T.C.
EGE ÜNİVERSİTESİ
Sağlık Bilimleri Enstitüsü

HASTALIK TANISI VERİLERİNDE VERİ ÖN İŞLEMENİN TOPLULUK ÖĞRENME SINIFLANDIRMA ALGORİTMALARI ÜZERİNDEKİ ETKİSİNİN İNCELENMESİ

Yüksek Lisans Tezi

Yüksel ÖZKAN

Biyoistatistik ve Tıbbi Bilişim Anabilim Dalı

İzmir
2019

T.C.
EGE ÜNİVERSİTESİ
Sağlık Bilimleri Enstitüsü

HASTALIK TANISI VERİLERİNDE VERİ ÖN İŞLEMENİN TOPLULUK ÖĞRENME SINIFLANDIRMA ALGORİTMALARI ÜZERİNDEKİ ETKİSİNİN İNCELENMESİ

Yüksel ÖZKAN

Danışman
Aslı SUNER KARAKÜLAH

Biyoistatistik ve Tıbbi Bilişim Anabilim Dalı
Biyoistatistik

İzmir
2019

TEZ ONAY SAYFASI

Kurum Adı	**: Ege Üniversitesi Sağlık Bilimleri Enstitüsü Müd.**
Anabilim Dalı	**: Biyoistatistik ve Tıbbi Bilişim**
Program	**: Biyoistatistik Yüksek Lisans**
Tez Konusu	**: Hastalık Tanısı Verilerinde Veri Ön İşlemenin Topluluk Öğrenme Sınıflandırma Algoritmaları Üzerindeki Etkisinin İncelenmesi**
Danışman	**: Dr. Öğr. Üyesi Aslı SUNER KARAKÜLAH**
Tezi Hazırlayan	**: Yüksel ÖZKAN**
Değerlendirme Kurulu Üyeleri	**:**

Adı Soyadı	**: Yüksel ÖZKAN**
Başkan(Danışman)	**: Dr. Öğr. Üyesi Aslı SUNER KARAKÜLAH**
Üye / İmza	**: Doç. Dr. Timur KÖSE**
Üye / İmza	**: Doç. Dr. Eralp DOĞU**
Üye / İmza	**:**
Üye / İmza	**:**

Tezin Kabul Edildiği Tarih **: 13.09.2019**

Önsöz

Yapay zeka teknolojisinin dahil olmadığı, geliştirmediği ve hatta değiştirmediği hiçbir alan neredeyse kalmamıştır. İnsan hayatının en önemli konusu olan sağlık alanında da yapay zeka teknolojisi devrim niteliğinde yenilikler sağlamaktadır. Her geçen gün giderek artan kardiyovasküler, diyabet, kanser ve nadir hastalıklarda erken tanı tartışılamaz öneme sahiptir. Bu durumda özellikle, hastalık tanısının hızlı ve etkili bir şekilde tespit edilmesi için yapay zeka yöntemlerinin gücünden faydalanmak gerekmektedir. Bu yöntemler sayesinde, daha hasta odaklı kişiselleştirilmiş sağlık hizmetlerinin sunulmasına imkan sağlanacaktır. Böylelikle, kişiye özgü tıbbın gelişimi hızlandırılmış olacaktır. Kişiye özgü tıpta, bireylerin geçmişteki ve gelecekteki tüm hasta bilgilerinin bulunduğu büyük verilerin dijital ortama aktarılması ve yapay zeka teknolojisi ile ilerleyen zamanda hastalık tanısının doğru konması hedeflenmektedir. Bu çalışmada, hastalık tanısı verilerinde veri ön işlemenin topluluk öğrenme sınıflandırma algoritmaları üzerindeki etkileri araştırılmış ve kullanılan algoritmaların performansları ile çalışma süreleri kıyaslanmıştır. Bu tez çalışmasının konusunun belirlenmesinden hazırlanmasına kadar sürecin her aşamasında kıymetli bilgilerini, tecrübesini ve zamanını benden esirgemeyerek her fırsatta çalışmamla en yakından ilgilenen, eleştirileriyle yol gösteren danışman hocam Aslı SUNER KARAKÜLAH'a teşekkürü ve özellikle minnetimi belirtmeyi bir borç bilirim.

İzmir, 16.09.2019 Yüksel ÖZKAN

Özet

Hastalık Tanısı Verilerinde Veri Ön İşlemenin Topluluk Öğrenme Sınıflandırma Algoritmaları Üzerindeki Etkisinin İncelenmesi

Sağlık alanında hastalığın tanımlanması ve incelenmesi için sınıflandırma yaparken, özellikle karmaşık verilerden anlamlı bilginin ortaya çıkarılmasında, yapay zekâ teknolojisini kullanarak hesaplama yapabilen denetimli makine öğrenme yöntemleri kullanılmaktadır. Topluluk öğrenme yöntemleri ise aynı problemi çözmek için birden fazla öğreniciyi aynı anda eğiterek daha başarılı modellerin kurulmasını sağlamaktadır. Bu çalışmada, sağlık verilerinde doğru hastalık tanısı koymak için kullanılan veri setlerinde olası karşılaşılabilecek kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi problemlere veri ön işleme yapıldıktan sonra, sınıflandırma algoritmalarının performanslarının karşılaştırılması amaçlanmıştır. Çalışmada, KEEL veri tabanından kalp hastalığı, tiroid, hepatit, lenfödem, meme kanseri ve diyabet gibi hastalıkların tanısı için toplanmış veriler kullanılmıştır. Sınıflandırma yapmak amacıyla, torbalama algoritmalarından rastgele orman ve ağırlıklı alt uzay rastgele orman algoritmaları kullanılırken; artırma algoritmalarından eklemeli lojistik regresyon ve gradyan artırma makinaları algoritmaları kullanılmıştır. Algoritmaların performanslarının karşılaştırılmasında doğruluk, duyarlılık/hassaslık, seçicilik, kesinlik, Kappa istatistiği, Youden indeksi, F - ölçütü ve ROC ölçüm metrikleri kullanılmıştır. Aynı zamanda, algoritmaların çalışma süreleri hesaplanmıştır. Tüm istatistiksel analizler, RStudio 1.2.1335 - Windows 7+ (64-bit) programı ile yapılmıştır. Orijinal veriler ve işlenmiş veriler için algoritmaların performansları karşılaştırıldığında, veri ön işlemeden sonra algoritmaların performans başarılarının arttığı görülmüştür. Genel olarak, artırma algoritmalarının performansları torbalama algoritmalarına göre daha yüksek sonuçlar vermiştir. Algoritmalar çalışma süreleri açısından kıyaslandığında ise, artırma algoritmaları en uzun süre çalışan algoritmalardır. Sonuç olarak, araştırmalar tarafından yüksek performans başarısı hedefleniyorsa, veri ön işleme göz ardı edilmemelidir. Veri ön işlemede, parametrelerin ayarlanma ve değişken seçimi gibi farklı konularda eklenerek benzetim çalışmaları yapılabilir.

Anahtar Kelimeler: Tanısı; Veri Ön İşleme; Kayıp Gözlem; Sınıf Gürültüsü; Sınıf Dengesizliği; Topluluk Öğrenme

Abstract

Investigation of the Effect of Data Preprocessing on Ensemble Learning Classification Algorithms in Disease Diagnosis Data

In the field of health, while classifying for identification and examination of disease, supervised machine learning methods are used, which are able to compute using artificial intelligence technology, in order to extract meaningful information from complex data. Ensemble learning methods enable establishment of more successful models by training multiple learners at the same time to solve same problem. In this study, it is aimed to compare performance of classification algorithms after data preprocessing to problems such as missing values, class noise and class imbalance that may be encountered in data sets used to diagnose accurate disease in health data. In the study, data collected from KEEL database were used to diagnose diseases such as heart disease, thyroid, hepatitis, lymphedema, breast cancer and diabetes. In order to make classification, while random forest and weighted subspace random forest were used as bagging algorithms; additive logistic regression and gradient boosted machines algorithms were used as boosting algorithms. Accuracy, sensitivity, specificity, precision, Kappa statistic, Youden index, F - measure and ROC measurement metrics were used to compare performance of algorithms. At the same time, run times of algorithms were calculated. All statistical analyzes were performed with RStudio 1.2.1335 - Windows 7+ (64-bit) program. When performances of algorithms were compared for original data and processed data, it was seen that performance success of algorithms increased after data preprocessing. In general, performance of boosting algorithms yielded higher results than bagging algorithms. When algorithms were compared in terms of run time, boosting algorithms were the longest running algorithms. As a result, data preprocessing should not be overlooked if research is aimed at high performance success. In data preprocessing, simulation studies can be performed by adding different topics such as tuning parameters and selecting variables.

Keywords: Diagnosis; Data Preprocessing; Missing Values; Class Noise; Class Imbalance; Ensemble Learning

İçindekiler

Tablolar Dizini

Şekiller Dizini

Grafikler Dizini

Kısaltma Listesi

IBM SSEC	:	Seçici Sıra Elektronik Hesap Makinesi
EDSAC	:	Elektronik Gecikmeli Depolama Otomatik Hesap Makinesi
CAPTCHA	:	Bilgisayarlar ve İnsanlar için Ayrı Ayrı İşlem Yapabilen Tamamen Otomatikleştirilmiş Turing Testi
CNNs	:	Konvolüsyonel Sinir Ağları
API	:	Uygulama Programlama Arayüzü
CPT	:	Mevcut İşlem Terminolojisi
FAHES	:	Gizlenmiş Kayıp Değerler Dedektörü
CNC-NOS	:	Sınıf Gürültü Filtresi
SMOTE	:	Sentetik Azınlıklı Aşırı Örnekleme
IPF	:	Yinelemeli Bölme Filtresi
AE	:	Kümelendirilmiş Topluluk
WELM	:	Ağırlıklı Aşırı Öğrenme Makinesi
PubMed	:	Ulusal Tıp Kütüphanesi
KEEL	:	Evrimsel Öğrenmeye Dayalı Bilgi Çıkarımı
GPLv3	:	Genel Kamu Lisansı Sürüm 3
MAR	:	Rastgele Kayıp
MCAR	:	Tamamıyla Rastgele Kayıp
MNAR	:	Rastgele Olmayan Kayıp
HRF	:	Melez Yeniden Etiketleme Filterisi
CART	:	Regresyon ve Sınıflandırma Ağaçları

Easy-SMT	:	Topluluk Öğrenmenin SMOTE Yöntemine Entegre Edilmesi
RUS	:	Rastgele Az Örnekleme Yöntemi
NCL	:	Komşuluk Temizleme Kuralı
ROC	:	Alıcı İşletim Karakteristiği
KDD	:	Veri Tabanlarından Bilgi Keşfi
CRAN	:	Kapsamlı R Arşiv Ağı
MICE	:	Zincirleme Denklemlerle Çok Değişkenli Atama
caret	:	Sınıflandırma ve Regresyon Eğitimi
DMwR	:	R ile Veri Madenciliği
JM	:	Birleşik Modelleme
FCS	:	Tamamen Koşullu Tanımlama
MCMC	:	Markov Zinciri Monte Carlo
mipo	:	Çoklu Atanan Havuzlanmış Sonuçları
mids	:	Çarpımla Atanmış Veri Seti Sınıfı
mira	:	Çarpımlı Atanmış Tekrarlanan Analiz
VIM	:	Kayıp Değerlerin Göreselleştirilmesi ve Atanması
PM	:	Kayıp Gözlem Yüzdesi
NR	:	Sınıf Gürültüsü Yüzdesi
IR	:	Sınıf Dengesizliği Oranı
AdaBoost	:	Uyarlanabilir Artırma
rf	:	Rastgele Orman

wsrf	:	Ağırlıklı Alt Uzay Rastgele Orman
logitboost	:	Eklemeli Lojistik Regresyon
gbm	:	Gradyan Artırma Makinaları
OOB	:	Test Veri Seti
IGR	:	Bilgi Kazanma Oranı
ELLOSS	:	Üssel Kayıp Fonksiyonunun Beklentisi
ACC	:	Doğruluk
SEN	:	Duyarlılık
SPE	:	Seçicilik
PRE	:	Kesinlik
KAP	:	Kappa İstatistiği
YI	:	Youden İndeksi
F	:	F - ölçütü
DP	:	Doğru Pozitif
YP	:	Yanlış Pozitif
YN	:	Yanlış Negatif
DN	:	Doğru Negatif
pmm	:	Tahmini Ortalama Eşleştirme
logreg	:	Lojistik Regresyon
polr	:	Oransal Regresyon
polyreg	:	Çoklu Kategorili Logistik Regresyon

1. Giriş

Giriş kısmında, araştırmanın problemi, araştırma sorusu, araştırmanın hipotezleri, varsayımları, sınırlılıkları ve amacına yönelik ilgili literatür çerçevesinde genel bilgiler yer almaktadır. *Genel bilgiler* kısmında, makine öğrenme yöntemlerinin tarihsel gelişimi, sağlık alanında makine öğrenme yöntemlerinin kullanılması ve hastalık tanısı verilerinde olası karşılacak problemler konuları üzerinde durulmuştur. *Gereç ve yöntemler* kısmında, çalışmada kullanılan veri setleri, veri ön işleme (data pre-proccesing) yaklaşımları, kayıp gözlem yüzdesi, sınıf gürültüsü yüzdesi ve sınıf dengesizliği oranının hesaplanması, kullanılan R paketleri, makine öğrenme yöntemleri, topluluk öğrenme yöntemleri, torbalama (bootstrap aggregating - bagging) ve artırma (boosting) algoritmaları, peformans ölçüm metrikleri ve algoritmaların hesaplama süreleri konuları hakkında ayrıntılı bilgi verilmiştir. *Bulgular* kısmında, orijinal veri setlerinin performans karşılaştırması, veri ön işleme yapılması, işlenmiş veri setlerinin performans kaşılaştırması ve algoritmaların hesaplama sürelerinin karşılaştırması yer almaktadır. *Tartışma* kısmında, literatürde yer alan benzer çalışmalara dayandırılarak elde edilen bulguların değerlendirilmesine yer verilmiştir. *Sonuç ve öneriler* kısmında, çalışmadan elde edilen bulgulara bağlı olarak genel bir değerlendirme yapılmış ve çalışma hipotezinin geçerliliği konusunda yargıda bulunulmuştur. Aynı zamanda, bu tür çalışmaların önemi vurgulanarak ileride benzer çalışmalar yapılırken karşılaşılacak olası durumlar hakkında önerilere de yer verilmiştir. *Kaynaklar* kısmında, tez çalışması sırasında yararlanılan literatüre yer verilmiştir.

1.1. Araştırmanın Problemi

Makine öğrenme, bilgisayarlar aracılığıyla verilerden bilginin ortaya çıkmasını sağlayan bilimsel bir disiplindir (Lip, Nieuwlaat, Pisters, Lane, & Crijns, 2010; O'Mahony et al., 2014). Makine öğrenme, istatistiksel yaklaşımların temeline dayanan bilgi işlem algoritmaları ile bilgisayar bilimleri teknolojisi kullanılarak verilerden öğrenmeyi amaçlamaktadır. İstatistik, matematik ve bilgisayar bilimleri arasındaki bu ilişki, milyarlarca veya trilyonlarca veri noktasını içerebilen, büyük veri kümelerinden istatistiksel modeller oluşturularak hesaplamaların yapılması prensibine dayanmaktadır.

Günümüze kadar birçok olması mümkün olmayan teknolojik gelişmelerin önünü açan yapay zeka teknolojisi (artificial intelligence technology) sayesinde, farklı alanlardaki pek çok endüstri karmaşık veri analizini kolaylıkla uygulama imkanı bulmuştur. Özellikle, sağlık alanındaki çalışmalarla ilgilenen endüstriler, makine öğrenme yöntemleriyle bu alana özgü

sorunların üstesinden gelebilmek için çalışmalarını sürdürmektedirler. Oldukça büyük tıbbi veri setlerinin var olmasından bu yana öğrenme algoritmalarının sağlık alanında uygulanmasına ilişkin çalışmaların sayısı da artmıştır. Tıbbi verilere makine öğrenme algoritmaları uygulayan binlerce çalışmanın olmasına rağmen, çok az çalışma klinik bakıma (clinical care) önemli bir şekilde katkıda bulunmuştur (Kononenko, 2001). Deo (2015) yayınladığı bir derlemede, bu eksikliği vurgulayarak, istatistiksel öğrenme yaklaşımlarının tıbbi pratiklikte uygulanması sırasında hangi engellerle karşılaştığını ele alarak bu eksikliğin nasıl üstesinden gelinebileceğini tartışmıştır (Deo, 2015). Baem ve Kohane (2016) ise derlemelerinde, yapay zekanın sağlık alanındaki uygulamalarda hala hakettiği yeri alamadığını ifade etmişlerdir. Klinik bakım, sağlık hizmetleri bilgi işlemlerini elektronik sağlık kayıtları biçiminde otomatikleştirmesiyle makine öğrenmenin bu alanda daha fazla uygulanarak sağlık giderlerinin düşürülebileceğini savunmuşturlar (Beam & Kohane, 2016).

Sağlık alanında sınıflandırmalar, hastalığın tanınması ve incelenmesi için temel sağlayan sağlık çalışmalarının bir parçasıdır (Jutel, 2011). Böylelikle, hekimler karmaşık verilerin anlamlı parçalara nasıl ayrılacağına karar vermesine yardımcı olmaktadır. Günümüze kadar, elektrokardiyografi sonuçlarının yorumlanabilmesi, hastalık tanılarının konulması ve uygun tedavilerin önerilmesi, klinik akıl yürütmenin yorumlanması ve hekimlere karmaşık hasta vakalarında tanı hipotezlerinin kurulması gibi birçok konuda makine öğrenmeden yararlanılmıştır. Farklı alanlarda çalışan sağlık uzmanları, tıbbi bilginin ortaya çıkarılmasında ve dayanıklı karar kurallarının oluşturulmasında, yapay zeka teknolojisiyle karmaşık veri yapılarında hesaplama yapabilen makine öğrenme yöntemlerini kullanmayı tercih etmektedir (Deo, 2015). Bu duruma bir örnek olarak, Hindistan'da bulunan Aravind Eye Care System'deki göz doktorları ve bilgisayar bilimciler, diyabetik hastaların milyonlarca retina (diabetic retinopathy) fotoğrafını taramak için otomatik bir görüntü sınıflandırma sistemini test etmek ve uygulamak için birlikte çalışmalarını sürdürmektedirler (Simonite, 2017). Bir başka çalışmada ise, DXplain adı verilen bilgisayar uzmanlığı olmayan hekimler tarafından kullanılmak üzere bilgisayar tabanlı bir tanısal karar destek sistemi tasarlanmıştır. Bu sistem makine öğrenme teknolojisini kullanarak, hastalığın tanısında ve sağlık verilerinin sınıflandırılmasında oldukça başarılı bir şekilde uygulanmaktadır. Bunun sebebi ise, sınıflandırmadaki başarısı sayesinde hekimlerin klinikte karar verme süreçlerini hızlandırmasıdır (Barnett, Cimino, Hupp, & Hoffer, 1987). Sınıflandırma algoritmalarının başarılı bir şekilde uygulandığı diğer bir çalışma ise, Derin Sinir Ağları (Deep Neural Networks) ile cilt kanserinin dermatolog düzeyinde sınıflandırılması çalışması olmuştur.

Çalışmada, cilt kanseri hastalarına ait muayene görüntüleri kullanılarak cilt lezyonlarının otomatik sınıflandırılması amaçlanmıştır. Konvolüsyonel sinir ağları (Convolutional Neural Networks - CNNs) algoritması kullanılarak eğitilen bu uygulama, cilt kanserinin tanısında dermatologlarla karşılaştırılabilir düzeyde sınıflandırabilen bir yapay zeka teknolojisi uygulaması olma özelliği taşımaktadır. Bu uygulamanın 2021 yılına kadar geliştirilerek dermatologların mobil cihazlarıyla klinik dışında da erişimlerinin sağlanması hedeflenmektedir. Böylelikle, tanı bakımından hayati önemi olan bir durum karşısında düşük maliyetli evrensel bir erişim sağlanabileceği öngörülmektedir (Esteva et al., 2017). Yapay zeka giderek tıbbi uygulamaların seyrini değiştirmektedir. Daha önce sadece sağlık uzmanlarının ilgilendiği konular olarak düşünülen birçok konu, artık makine öğrenme ile yayılmaktadır (Yu, Beam, & Kohane, 2018). Yapay zeka ile birlikte geleneksel bilgisayar destekli tanıda sınırlamalar ortadan kaldırılmıştır. Böylelikle, tanı koymada aynı anda birden fazla hipotezin test edilmesi sağlanarak daha dayanıklı sonuçlar elde edilmesi sağlanmıştır (Szolovits & Pauker, 1978).

Bu çalışmada, sağlık verilerinde doğru hastalık tanısı koymak için olası karşılaşılabilecek kayıp gözlem problemi, gürültülü veri problemi ve sınıf dengesizliği problemine ilişkin sorunların göz ardı edilmeden veri ön işlemeye tabi tutulduktan sonra, makine öğrenme sınıflandırma algoritmalarının uygulanması, sonrasında algoritmalarının performanslarının ve çalışma sürelerinin karşılaştırılması amaçlanmaktadır.

1.2. Araştırmanın Sorusu

Hastalık tanısı verilerinde veri ön işlemenin topluluk öğrenme algoritmaları üzerinde etkisi var mı?

1.3. Araştırmanın Hipotezleri

Bu tez çalışması kapsamında ilgilenilen hipotezler,

- *Cleveland, Heart, Hepatitis, Lymphography, Mammography, Newthyroid, Pima ve Thyroid* verilerin bağımsız sürekli değişkenleri normal dağılıma sahip değildir.
- *Cleveland, Heart, Hepatitis, Lymphography, Mammography, Newthyroid, Pima ve Thyroid* verileri bağımsız sürekli değişkenleri birbirleriyle ilişkili değildir.

- Orijinal veri setleri kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi olası karşılaşılacak problemlere sahipse, sınıflandırma algoritmaları performans başarıları azalır.
- Kayıp gözleme sahip olmayan verilerin sınıflandırma algoritmaları performans başarıları yüksektir.
- Sınıf gürültüsüne sahip olmayan verilerin sınıflandırma algoritmaları performans başarıları yüksektir.
- Sınıf dengesizliğine sahip olmayan verilerin sınıflandırma algoritmaları performans başarıları yüksektir.
- İşlenmiş veri setlerinin sınıflandırma algoritmaları performans başarıları artar.
- Artırma algoritmaları ile torbalama algoritmalarının sınıflandırma performans başarıları birbirinden farklıdır.
- Artırma algoritmaları ile torbalama algoritmalarının hesaplama süreleri birbirinden farklıdır.
- Veri ön işlemeden sonra hastalık tanısı verilerinin tanı koyma doğruluğu artar.

1.4. Araştırmanın Varsayımları

Bu tez çalışması kapsamında, araştırmanın bazı aşamaları için geçerli olan varsayımlardan bahsedilmiştir. Tüm analiz aşamalarında, R fonksiyounu olan $seed.set()$ fonksiyonu kullanılarak rastgele sayıların üretimi sabit tutulmuştur. Buna bağlı olarak, yapılan hesaplamaların tamamının her defasında aynı sonucu vermesi sağlanmıştır. Bu tez çalışmasında, birden çok veri seti kullanıldığı için parametre optimizasyonu yapılmayarak sadece bazı parametre değerleri sabit tutulmuştur. Böylelikle, algoritmalara ait parametre değerleri hem orijinal hem de işlenmiş veriler için aynıdır. Kayıp gözlem ataması için kullanılan çoklu atama (multiple imputation) modellerine ait olası atama değerleri her yinelemede aynı değerlere sahiptir. Çoklu atama modelleri, rastgele kayıp varsayımı altında atama yapmaktadır. Buna göre, değikenlerdeki eksik veri değerlerinin diğer değişkenlere bağlı olduğu varsayılmaktadır. Sınıf gürültüsünün tespit edilmesinde kullanılan filtreleme yöntemi her yinelemede aynı sınıfın gürültüye sahip olduğu varsayımına dayanmaktadır. Sınıfların dengelenmesi aşamasında üretilen verilerden her yinelemede aynı sınıf değerlerinin elde edildiği varsayılmaktadır.

1.5. Araştırmanın Sınırlılıkları

Bu tez çalışmasının sahip olduğu bazı sınırlılıklar vardır:

- Çalışmada kullanılan veri setleri, sadece bir veri tabanından elde edilmiştir. Bu yüzden araştırma konusuyla ilgili diğer veri tabanlarından yararlanılmamıştır.
- Çalışmada sadece hastalık tanısı verileri kullanılmıştır. Bu veriler, altı farklı hastalık tanısına (kalp, hepatit, lenfoma, meme kanseri, tiroid ve diyabet hastalıkları) ait verilerdir. Diğer hastalıklara yer verilmemiştir.
- Araştırma kapsamında elde edilen veri setlerine ait değişkenler çok fazla olduğu için tamamının değişken bilgisine sahip olmak mümkün değildir. Bu yüzden sadece bazılarının açıklamalarına yer verilmiştir.
- Sınıf gürültüsü probleminin çözümü için kullanılan filtre yöntemi kayıp gözleme karşı duyarlı olduğu için öncelikle kayıp gözlem probleminin çözülmesi gerekmektedir.
- Bağımsız kategorik değişken düzeylerinde sadece tek bir sınıf düzeyi için değerler varsa algoritmalar, ilgili değişkene ait veri seti için hesaplama yapmamaktadır. Bu değişkenin modelden çıkarılması gerekmektedir.
- Algoritmaların hesaplama süreleri oldukça zaman aldığı için bilgisayar işlem hızının beşinci nesil bir bilgisayar olması önerilir.

1.6. Araştırmanın Amacı

Bu çalışmada, hastalık tanısı konulması amacıyla elde edilen verilerde olası karşılaşılabilecek kayıp gözlem, gürültülü veri ve sınıf dengesizliği problemlerinin göz ardı edilmeden veri ön işlemeye tabi tutulduktan sonra topluluk öğrenme sınıflandırma algoritmalarının performanslarının ve çalışma sürelerinin karşılaştırılması amaçlanmaktadır.

2. Genel Bilgiler

Tezin bu bölümde, araştırma problemine yönelik literatür taraması, kullanılacak yöntemlerin tarihsel gelişimine ilişkin değerlendirme ve sağlık alanındaki uygulama örnekleriyle problem türlerine ilişkin genel bilgiler sunulmaktadır.

2.1. Makine Öğrenme Yöntemlerinin Tarihsel Gelişimi

Tezin bu bölümde, makine öğrenmenin yıllar içerisinde nasıl geliştiği ve bugün geldiği noktalar hakkında bilgi verilecektir. Makine öğrenme yöntemlerinin kronolojik gelişimi detaylı olarak aşağıda incelenmiştir.

1940'dan önce - İstatistiksel Yöntemlerin Keşfi

- 18. yüzyılda, İngiliz istatistikçi Thomas Bayes, bir binom dağılımına ait parametrenin olasılık dağılımını hesaplayabilen bir teorem keşfetmiştir. Bu teoreme, Bayes Teoremi (Bayes Theorem) adı verilmiştir (Dale, 1988; Earman, 1990; Gillies, 1987; Poitras, 2013).
- 1795 yılında Carl Friedrich Gauss tarafından en küçük kareler yöntemi (least squares method) geliştirilmiştir (Hogan, 1977; Stigler, 1981; Wight & Gable, 2005).
- 1854 yılında George Boole mantıksal akıl yürütmenin, sistematik olarak denklem çözme ile birbirine benzediği düşüncesini savunmuştur (Ledesma, Pérez, Borrajo, & Laita, 1997).
- 1913 yılında Andrey Markov tarafından veri analiz yöntemlerinden olan Markov Zinciri (Markov Chain) yöntemi geliştirilmiştir (Lloyd, 1974; Takano, 1973).
- 1914 yılında Leonardo Torres y Quevedo hiçbir insan yönlendirmesi olmadan kendi kendine oyunu devam ettirebilen ilk santraç makinesini tasarlamıştır (Weverbergh, 2017).
- 1921 yılında Çek yazar Karel Capek, "robot" kelimesini literatüre kazandırmıştır (Hor´kov´ & Kelemen, 2011; Horáková & Kelemen, 2009).

1940 – 1950 yılları arası - Depolanabilir Bilgi İşlem Makineleri ve Zekâ

- 1940'ların sonlarında, verilerin elektronik bir bellekte saklanabilmesi için Depoyabilen Program Bilgisayar (Stored-Program Computer) geliştirilmiştir (Fjeldaas & Lygre Furevik, 2016; Neumann et al., 1973).

- 1948 yılının Ocak ayında, IBM tarafından elektromanyetik bigisayar olarak adlandırılan seçici sıra elektronik hesap makinesi (IBM Selective Sequence Electronic Calculator - IBM SSEC) tasarlanmıştır (Norman, 2004).
- 1948 yılının Mayıs ayında Londra Üniversitesi'nde Andrew Booth ve Kathleen Booth ilk dönen tambur depolama cihazına (drum storage device) sahiplerdir (Johnson, 2010). Haziran ayında ise, Manchester Bebeği (Manchester Baby) olarak adlandırılan depolanmış bir programı uygulayan, tamamen elektronik Manchester küçük ölçekli deney makinesi (Manchester small-scale experimental machine) adında bir bilgisayar yapılmıştır (Ward, 2010).
- 1949 yılının başlarında, ilk programını çalıştırabilen Manchester Mark 1, Mayıs ayında ise, Cambridge Üniversitesi'nde, elektronik gecikmeli depolama otomatik hesap makinesi (Electronic Delay Storage Automatic Calculator - EDSAC) adı verilen ilk programını çalıştırabilen, tam ölçekli bir işletim bigisayarı (full-scale operational computer) geliştirilmiştir (Shelburne & Burton, 1998; Wilkes, 1997).
- 1950 yılında Alan Turing tarafından yayınlanan makalade, "Makinalar düşünebilir mi?" sorusu tartışılmış ve Turing testi oluşturulmuştur (Aamoth, 2014; French, 2000; Korukonda, 2003; McCoy & Ullman, 2018; Turing, 1950). 1950'li yıllarda tekillik (singularity) terimi, teknoloji alanında ilk kez Jon von Neumann tarafından kullanılmıştır (Walsh, 2017). Neumann'a göre, giderek artan bilginin ve gelişen teknolojinin insan ırkı benzeri bir tekillik noktasına doğru götürdüğünü savunmaktadır (Magee & Devezas, 2011).

1950 – 1960 yılları arasında - İlk Yapay Sinir Ağı ve Yapay Zeka Dönemi

- 1951 yılında Marvin Minsky ve Dean Edmonds, beyinlerin biyolojik çalışma disiplininden yola çıkarak bilgisayar temeline dayanan bir simülasyon yaparak ilk yapay sinir ağını geliştirmişlerdir (Minsky, 1961; Mochari, 2016; Verwijnen, 2016).
- 1958 yılında, John McCarthy tarafından yapay zeka çalışmalarında kullanılan programlama dili olan Lisp geliştirilmiştir (McCarthy, 1960, 1979).
- 1959 yılında Arthur Samuel, makine öğrenme (machine learning) terimini ortaya atmıştır (Samuel, 1959).

- 1960'lı yıllarda Geoffrey Hinton, çoklu öğrenme tanımını ortaya atarak derin öğrenme (deep learning) kavramını ilk kez dile getirmiştir. Günümüzde Hinton derin öğrenmenin babası olarak da anılmaktadır (Rumelhart, Hinton, & Williams, 1986).

1960 – 1970 yılları arasında - Robot Dönemi

- 1966 yılında, Stanford Araştırma Enstitüsü Yapay Zeka Merkezi tarafından kendi hareketlerinin sorumluluğunu alabilen Shakey the Robot adı verilen ilk robot geliştirilmiştir (Darrach, 1970; Kuipers, Feigenbaum, Hart, & Nilsson, 2017; Press, 2016; Speck, Dornhege, & Burgard, 2017).
- 1969 yılında Arthur Bryson ve Yu-Chi Ho tarafından, çok katmanlı bir dinamik sistem optimizasyon yöntemi olan geri yayılım (backpropagation) tanımlanmıştır (Bryson & Ho, 1969; Schmidhuber, 2014; Vamsidhar, Varma, Rao, & Satapati, 2010).

1970 – 1980 yılları arasında

- 1973 yılında, James Lighthill, Artificial Intelligence: A General Survey adı verdiği Lighthill Raporu'nu parlementoya sunmuştur (Boden, 1984; McCarthy, 1974, 2000).
- 1974 yılında ilk yapay zeka kışı (artificial intelligence winter – AI Winter) yaşanmıştır (Dickson, 2018).

1980 - 1990 yılları arasında - Uzman Sistemler Dönemi

- 1980 yılında, dünyadaki şirketler tarafından uzman bilgisayarlar (expert systems) adı verilen bir yapay zeka programı kabul edilmiştir (Fogg, 2017).
- 1985 yılında, sinirbilimci Terry Sejnowski tarafından NETtalk adı verilen bir yazılım geliştirilerek, İngilizce kelimelerin nasıl telaffuz edilmesi gerektiğine yardımcı olmak amaçlanmıştır (Fogg, 2017).
- 1989 yılında, Axcelis, kişisel bilgisayarlarda genetik algoritmaların kullanımını ticarileştiren ilk yazılım paketi olan Evolver paketini yayınlamıştır (Markoff, 1990).

1990 – 2000 yılları arasında - Teknolojik Tekillik

- 1993 yılında Vernor Vinge tarafından "Yaklaşmakta Olan Teknolojik Tekillik" adlı bir makale yayınlanmıştır (Vinge, 1993). Böylelikle, Vinge de tekillik terimini kullanan araştırmacılardan biri olmuştur. Vinge göre, gelecek 30 yıl içinde insanüstü zeka yaratmak için gerekli olan teknolojiye sahip olunacağını belirtilmektedir. Hatta, bu

teknolojilerden dolayı bir zaman sonra insanlık çağının sona ereceğini ifade etmektedir (Earman & Eisenstaedt, 1999; Vinge, 2008).

- 1996 yılında, yapay zekaya olan ilginin yeniden artmasıyla birlikte IBM tarafından Deep Blue adı verilen satranç oynayabilen bir bilgisayar geliştirilmiştir (Campbell, Hoane, & Hsu, 2002). 1997 yılında ise saniyede 200 milyon pozisyon deneyebilen Deeper Blue adı verilen ikinci bir bilgisayar daha geliştirmiştir (Higgins, 2017; Hsu, 1999; IBM, 2011).

2000 – 2010 yılları arasında - Makine Öğrenme Uygulamaları Dönemi

- 2000 yılında, Carnegie Mellon Üniversitesi'nden Luis von Ahn, Manuel Blum, Nicholas Hopper ve John Langford tarafından Turing testine dayanan bilgisayarlar ve insanlar için ayrı ayrı işlem yapabilen tamamen otomatikleştirilmiş Turing testi (Completely Automated Public Turing test to tell Computers and Humans Apart - CAPTCHA) adı verilen bir uygulama geliştirilmiştir. Bu uygulama, internet üzerinden erişim sağlayan kullanıcıların insan mı yoksa makina mi olduğunu anlayabilmek için kullanılmaktadır (Rao, Sri, & Sai, 2016).
- 2005 yılında Ray Kurzweil'in "Tekillik Yakın: İnsanlar Biyolojinin Ötesine Geçtiğinde" adlı kitabı yayınlanmıştır (Kurzweil, 2005).
- 2006 yılında geri yayılım, çok katmanlı algılayıcı (multilayer perceptron) olarak adlandırılan yapay sinir ağlarının eğitimi için kullanılan bir öğrenme algoritması geliştirilmiştir (Hecht-Nielson, 1992; H.-M. Lee, Chen, & Huang, 2001).
- 2012 yılında Geoffrey Hinton, sinir ağları çalışmalarında, Boltzmann makineleri (Boltzmann machines), zaman gecikmeli sinir ağları ve varyasyonel öğrenme gibi birçok farklı çalışmaya yer vermiştir (Amari, Kurata, & Nagaoka, 1992; Golovko, Kroshchanka, Turchenko, Jankowski, & Treadwell, 2015; Hinton et al., 2012; Le Roux & Bengio, 2008).
- 2006 yılında, Oren Etzioni, Michele Banko ve Michael Cafarella yaptıkları bir çalışmada, yazıyı otomatik olarak anlayabilme kabiliyetine sahip TextRunner adını verdikleri bir sistem geliştirmişlerdir (Etzioni, Banko, & Cafarella, 2006; Manning & Schütze, 1999; Yates et al., 2007).

2010 - Günümüz

- 2011 yılında konvolüsyonel sinir ağı kullanımı farklı alanlarda giderek artmıştır (D. C. Ciresan, Meier, Gambardella, & Schmidhuber, 2011; D. Ciresan, Meier, Masci, & Schmidhuber, 2012; Lecun, Bottou, Bengio, & Haffner, 1998; Sharma, Jain, & Mishra, 2018).
- 2014 yılında Google tarafından DeepMind teknolojisi kullanılarak ekrandaki piksellerin davranışını analiz ederek ve insanların video oyunlarını nasıl oynadığından esinlenerek, kendi kendine öğrenen bir yapay sinir ağı geliştirilmiştir. Sonrasında, Sinir Turing Makinası (Neural Turing Machine) adı verilen harici belleğe erişebilen bir sinir ağı da şirket tarafından geliştirilmiştir (Graves & Wayne, 2014; Khayut, Fabri, & Avikhana, 2016).
- 2014 yılının Haziran ayında geliştirilen Eugene Goostman adlı yazılım, jüriye kendisinin bir insan olduğuna ikna ederek Turing testini geçmeyi başaran ilk yapay zeka yazılım olmuştur (Aamoth, 2014).
- 2014 yılında Ergun Ekici'nin yapımcılığını üstlendiği Amelia geliştirilmiştir. Amelia, yapay zeka kullanılarak geliştirilen birçok akıllı makineden farklı olarak insanlara özgü davranışları taklit etmek yerine insanların düşünme yapısını anlayabilme üzerine kurgulanmıştır (Martinez, 2018).
- 2016 yılında, DeepMind, bir tıbbi sistemi eğitmek için Kraliyet Devlet Hastanesi'nden NHS hasta verilerinin kullanımıyla ilgili bir çalışmaya dahil olmuştur. Fakat bu durum, İngiltere Bilgi Komiserleri Ofisi tarafından veri korunma yasasını ihlal ettiği gerekçesiyle iptal edilmiştir (Hill, 2018).
- 2016 yılında IBM tarafından yapay zeka teknoloji kullanılarak AlphaGo geliştirilmiştir (Browne et al., 2012; Chao, Kou, Li, & Peng, 2018; J. X. Chen, 2016; H. Li, Xiong, Ohno-Machado, & Jiang, 2014; Y. Pan, 2016). 2017 yılında ise AlphaGo modeline göre daha gelişmiş bir sürüm olan DeepMind teknolojisinin yeni yapay zeka modeli Alpha Zero tasarlanmıştır (Silver et al., 2018, 2017).
- 2017 yılında Google, video içeriğini tanıyıp aranabilir hale getiren makine öğrenme uygulama programlama arayüzünü (Application Programming Interface - API) yayınlamıştır (Pathak, Pandey, & Rautaray, 2018). Bunun yanı sıra, derin öğrenme

teknolojisiyle geliştirdiği GoogleNet sayesinde kanserli hücreleri tespit etmeyi amaçlamıştır (Callaghan, 2014; Szegedy et al., 2015).

- 2017 yılından bu yana Geoffrey Hilton ve Toronto'daki araştırma grubu, konuşma tanıma ve nesne sınıflandırma konularında başarılar yakalayarak derin öğrenmede büyük gelişmeler göstermektedir (Guatto, 2017).
- 2017 yılında Google'da çalışmalarına devam eden Hilton, geri yayılmanın makine öğrenme sistemlerinin gelişmesine katkı sağladığını belirtmiştir (Google, 2019).
- 2018 yılında Microsoft, kanser tedavi dahil birçok sağlık sorununun tedavisi için yapay zeka ve makine öğrenme teknolojisini kullanarak Healthcare NExT sistemini tasarlamıştır (P. Lee, 2018).

Gelecek Yıllarda…

- 2021 yılı itibariyle Microsoft'un insan vücudu içinde kanserli hücrelerin tespit edilebileceği ve bu hücreleri yeniden programlayarak yararlı hale getirebileceği bir yapay zeka geliştirmesi beklenmektedir.
- Google mühendislik direktörü Ray Kurzweil'in tahminlerine göre, teknolojik tekilliğin 30 yıl içerisinde gerçekleşeceği yönündedir.
- Kurzweil tahminlerine göre 2029 yılında, yapay zekanın Turing testini geçeceğini ifade etmiştir.
- Kurzweil, 2030'larda yapay zekayla gelişen teknolojilerin insan beyninin içerisine yerleştirilip hafızanın güçlendireceğini de söylemektedir.
- Bilim insanlarının bu tahminlerinin gerçekleşmesi halinde tasarlanan insan-makine birleşimi makinaların geliştirildiği Transhümanist Çağ ya da İnsan Ötesi Çağ (Transhumanism) olarak adlandırılan dönemin başlayacak olmasıdır.
- Yapay zeka teknolojisi sayesinde, bu makinaların hızı ve zekâsı insan zihnine entegre edilerek insanlığı birkaç seviye atlatacağı ve daha güçlü bir beyin yapısının ortaya çıkacağı düşünülmektedir (Kostick, 2017; Popoveniuc, 2013).
- Tekilliğin gerçekleşebilme olasılığı, Cambridge Varoluşsal Risk Araştırmaları Merkezi gibi kuruluşlar tarafından incelenmektedir. Öte yandan, tekilliğin gerçekleşme olasılığı, bilim dünyasında çok büyük bir etki yaratmaktadır. Bu etkiye

de, birisini veya bir şeyi adlandırmaktan korkmak anlamına gelen Voldemort Etkisi (Voldemort Effect) adı verilmektedir (Alegre, 2017).

Bu tez kapsamında ise, yapay zekanın öğrenme yöntemlerinden makine öğrenme sınıflandırma algoritmaları üzerinde durulacaktır. Öncelikle sağlık alanında makine öğrenme sınıflandırma algoritmalarının kullanımı, literatürdeki çalışmalarla değerlendirilecek, sonrasında sağlık verilerinde ortaya çıkabilecek olası problemlerden bahsedilecek ve bu probleme ilişkin çözüm yaklaşımları anlatılacaktır.

2.2. Sağlık Alanında Makine Öğrenme Yöntemlerinin Kullanımı

Sağlık alanında üretilen bilginin hızı bir şekilde artmasına paralel olarak bu alandaki karar verme süreçlerine ilişkin karar desteği uygulamalarında da sayıca bir artış söz konusu olmaktadır (Osamor & Grady, 2016; Sirovich, Gallagher, Wennberg, & Fisher, 2008; Wickremasinghe, Hashmi, Schellenberg, & Avan, 2016). Bu bağlamda, makine öğrenmenin tıbbi tanıdaki uygulamaları, gelişmiş ve karmaşık veri analizini mümkün kılan algoritmaların, sistemlerin ve metodolojilerin ortaya çıktığına işaret etmektedir (Kononenko, 2001). Yakın bir gelecekte, akıllı veri analizinin bir parçası olan makine öğrenmesi yöntemlerinin, modern teknoloji tarafından üretilen ve depolanan büyük miktarda bilginin işlenmesinde kullanılacağı yaygın olarak öngörülmektedir. Halen mevcut makine öğrenmesi yöntemleri, biyomedikal alandaki verilere ait tanımlanmamış ilişkilerin ve gizli örüntülerin (patterns) ortaya çıkarılmasında oldukça önemli bir potansiyele sahiptir. Aynı zamanda, makine öğrenmesi, sağlık uzmanlarının hassas tıp (precision medicine) olarak bilinen kişiselleştirilmiş bakım hizmetine geçmelerini sağlayan büyük veri setlerini otomatik olarak anlamlı ve yorumlanabilir hale getirmelerine yardımcı olabilmektedir (Azencott, 2018).

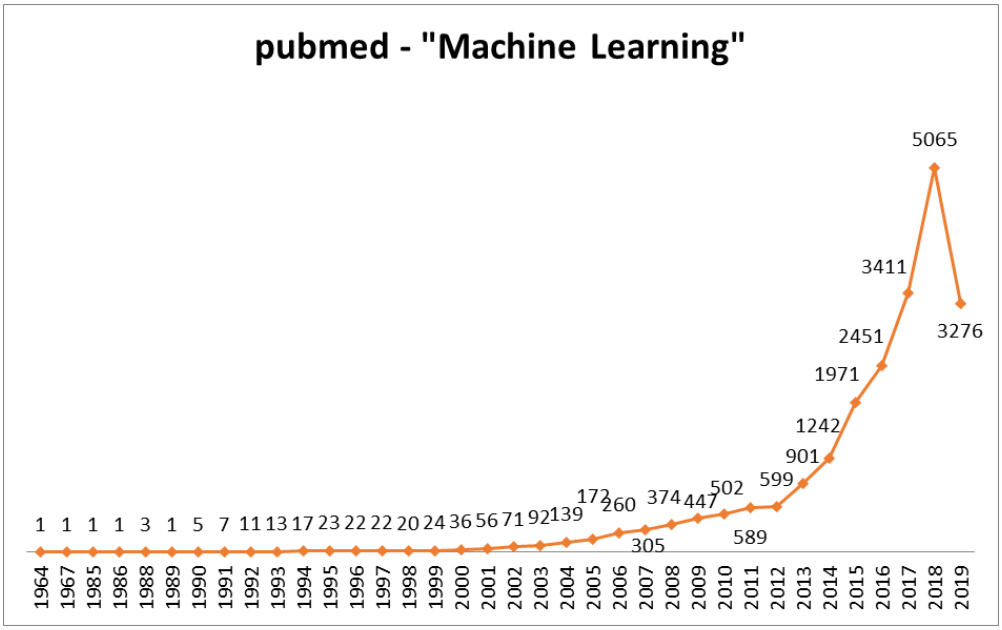
Jutel (2011), sınıflandırmanın tıpta pratiğe rehberlik ettiğini ve sınıflandırmayı anlamanın, tanı sonuçlarını daha iyi anlama arayışının bir parçası olması gerektiğini savunmuştur (Jutel, 2011). Subbulakshmi ve Deepa (2015), tıbbi veri sınıflandırmasını, dünya çapında birçok araştırmacının ilgisini çeken temel veri madenciliği (data mining) problemi olarak tanımlamışlardır. Çalışmalarında, makine öğrenmesi temeline dayanan medikal veri setlerini sınıflandıran aşırı öğrenme makine sınıflandırıcı (extreme learning machine classifier) ile parçacık sürü optimizasyonu (particle swarm optimization) yöntemlerini entegre eden melez bir makine öğrenme yaklaşımı önermişlerdir (Subbulakshmi & Deepa, 2015). Thazin ve diğerleri (2014), tıbbi tanı koymada sınıflandırma problemlerine karşı makine öğrenme algoritmaları ile bir sınıflandırma modeli oluşturarak çözüm sunulması gerektiğini ifade

etmişlerdir. Tıbbi veri setlerinin sınıflandırılması için ileri beslemeli yapay sinir ağları algoritmasına (feedforward neural networks algorithm) dayanan ve göç temelli diferansiyel evrim algoritması (migration based differential evolution algorithm) ile ağ öğrenimi, yakınsama oranı ve sınıflandırma doğruluğunu sağlayan bir yaklaşım geliştirmişlerdir (Thazin, Thein, Mo, & Tun, 2014). Raval ve diğerleri (2015), hastalık tahmini için geliştirilen tıbbi tanı yazılımlarının giderek arttığına dikkat çekmişlerdir. Özellikle, yapay zekâ ve yapay sinir ağları, bu tür tıbbi tanı problemlerini çözmek için kullanılan yöntemlerin başında gelmektedir. Makine öğrenme yöntemleri, tıbbi tanı koymada, hastaların klinik ve laboratuvar semptomlarına dayalı olarak, hastalıklarının uygun verilerle analiz edilebilmesi ve belli hastalıklar için daha verimli sonuç elde edilmesini sağlayan yöntemler olarak kullanılmaktadır. Bu yüzden, makine öğrenme yöntemleri tanı koyma aşamasında oldukça sorunsuz çalışan ve zaman kaybını önleyen yöntemlerdir (Raval, Bhatt, Kumhar, Parikh, & Vyas, 2015).

Medikal tanı (medical diagnosis), sağlık uzmanları açısından kritik bir karar verme sürecidir (Foster, Koprowski, & Skufca, 2014). Tanı kelimesinin kökeni, Fransızca diagnostic kelimesinden gelmektedir. Hastalığın ne olduğunu araştırıp ortaya koyma, tanılama ve teşhis anlamında kullanılmaktadır (Türk Dil Kurumu, 2006). Tanı, bir hastalık ya da bozukluğu hastanın tıbbi geçmişini, hastalık belirtilerini ve bulgularını değerlendirerek ve hastayı çeşitli yöntemlerle muayene ederek tanımlama işlemidir. Tanı terimi, hekim ya da muayene eden kişinin vardığı kararı belirtmek için de kullanılmaktadır. Buna karşılık olarak tıbbi tanı ise, bir kişiye ait semptomların ve belirtilerin hangi hastalık veya durumun açıkladığını belirleme işlemi olarak tanımlanmaktadır. Tıbbi tanı, tıp sözlüğünde D_x veya D_S olarak kısaltılmıştır. Bu kısaltmalar, ondalık sayı basamağına denk gelen mevcut işlem terminolojisi (Current Procedural Terminology - CPT) kodlarıdır. CPT kodları, doktorların uyguladıkları tedaviye yönelik süreçlerini ve tedavi maliyetlerini ilişkilendiren uluslararası puanlama sistemidir (Dotson, 2013). Sağlık alanında hastalık tanısını konulabilmesi için klinik bulgular ile biyokimyasal analiz sonuçlarının birlikte değerlendirilmesi gerekmektedir. Bu sonuçlar bütünleşik olarak hastalık tanısının konulmasında destekleyici rol oynamaktadır. Medikal tanının konulmasında hekimlerin kararlarını daha etkin hale getirmek için sınıflandırma algoritmalarından yararlanılmaktadır (Gammerman, 2010; Magoulas & Prentza, 2001). Fakat bu algoritmalarla hastalık tanısı konurken, karşılaşılan yetersiz bilgi, hatalı veri toplama ve/veya hatalı doğrulamadan kaynaklanan bilişsel hatalar (inadequate knowledge), kayıp gözlem problemi (missing value problem), dengesiz sınıf problemi (imbalanced class

problem), gürültülü veri problemi (noisy data problem), yanıt değişkeni ve açıklayıcı değişkenlerin tanımlanmasında karşılaşılan problemler gibi birçok problem tanısal hataya yol açmaktadır (Kübler, Liu, & Sayyed, 2018; Wan, Duan, & Zou, 2017). Bu problemlerin varlığında tanı koymada kullanılan algoritmalar, bu durumdan etkilenmekte ve yanlış tanı konulmasına sebep olabilmektedir. Aynı zamanda, veri setlerinde küçük n ve büyük p durumu olarak literatürde adlandırılan $p > n$ boyut laneti (curse of dimensionality) durumu da söz konusu olabilmektedir (Altman & Krzywinski, 2018; S. Chen, Montgomery, & Bolufé-Röhler, 2015). Sınıflandırma algoritmalarının sağlıkta sıkça rastlanan bu tür problemlere sahip verilerle maksimum verimle çalışabilmesi için veri ön işleme yapılmalıdır. Böylelikle hasta güvenliği, kalite geliştirme, karar verme ve problem çözme gibi çeşitli alanlarda çalışan araştırmacılar, tıbbi tanının konulması konusunda daha güvenilir sonuçlar elde edebilirler.

Makine öğrenme yöntemlerine ilişkin medikal alanda yapılan literatür çalışmalarını incelemek amacıyla, fen ve sağlık bilimleri alanındaki dergilerden, online kitaplardan ve MEDLINE'dan elde edilen biyomedikal literatürü içeren Ulusal Tıp Kütüphanesi'nin bir servisi olan Public Medline (PubMed) arama sonuçlarından yararlanılmıştır. PubMed veri tabanında Grafik 1'de gösterildiği gibi "Machine Learning"[Mesh] arama terimi kullanılarak Haziran 2019'da literatür taraması yapıldığında, 1964 yılından itibaren sağlık alanında yayınlanmış 22131 makaleye ulaşılmıştır.



Grafik 1: PubMed Veri Tabanı Arama Sonuçları

2.3. Sağlık Verilerinde Karşılaşılabilecek Olası Problemlere İlişkin Literatür Taraması

Bu tez kapsamında sağlık verilerinin sınıflandırılması için elde edilen veri setlerinde karşılaşılabilecek kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi olası problemler üzerinde durulacak ve bu problemlere ilişkin çözüm önerilerinde bulunulacaktır.

Literatürde, sağlık verilerinde kayıp gözlem problemine yönelik yapılan çalışmalara bakıldığında, Wood ve diğerleri (2004) kayıp değerlere epidemiyolojik ve klinik araştırmalarda oldukça fazla rastlanılması ile kayıp değerlerin araştırma sonuçlarının geçerliliğini zayıflatma potansiyeline sahip olmalarına rağmen çoğu kez tıbbi literatürde göz ardı edilmiş olduğunu vurgulamışlardır (Wood, White, & Thompson, 2004). Qahtan ve diğerleri (2018) genel olarak iki tür kayıp değer olduğunu, bunlardan birinin açık değerler diğerinin ise örtülü kayıp değerler olduğunu ve örtülü kayıp değerlerin algılanmadığını belirtmişlerdir. Bu algılanamayan örtülü kayıp değerleri ortaya çıkarmak için Dayanıklı Gizlenmiş Kayıp Değerler Dedektörünü (Robust Disguised Missing Values Detector - FAHES) önermişlerdir. Fakat bu çalışma da yine spesifik bir problem üzerine yoğunlaşıldığı için, diğer problemler göz ardı edilmiştir ve diğer problemler açısından çalışma eksik kalmıştır (Qahtan, Elmagarmid, Castro, Ouzzani, & Tang, 2018). Lee ve diğerleri (2017), kütle spektrometresi verilerinin kimyasal türlere dayalı biyolojik olayları analiz etmek için kullanıldığını; ancak, bu verilerin teknik veya biyolojik faktörlerden dolayı beklenmedik yinelenen kayıtları ve eksik değerleri içerdiğini belirtmişlerdir. Kayıp değer problemlerinin istatistiksel ve makine öğrenme yöntemlerinin gücünü etkileyebildiğini; bu nedenle, bu sorunları hafifletmek için yöntemler geliştirilmesi gerektiğini savunmuşlardır. Aynı zamanda bu yöntemlerin kütle spektrometresi verilerinin yapısı için uygun olmadığını belirterek, bunun için açık kaynak kodlu bir yazılım olan R programlama dilinde Eksik Değerler Ön işlemcisi (Missing Values Preprocessor) uygulamasını geliştirmişlerdir. Geliştirilen bu uygulama da, sadece kayıp gözlem problemiyle baş etmek için tasarlandığından, sınıf dengesizliği problemi ve sınıf gürültüsü problemi gibi problemler görmezden gelinmektedir (G. Lee, Lee, Jung, & Nam, 2017).

Sınıf gürültüsü problemi için Frenay ve Verleysen (2014) sınıf gürültüsünün tahminlerin doğruluğunu azaltırken, modellerin karmaşıklığı ve gerekli eğitim örneklerinin sayısını arttırabileceğini belirtmişlerdir (Frenay & Verleysen, 2014a). Luengo ve diğerleri (2018) sınıflandırma problemlerinde sınıf gürültüsüyle başa çıkmak için ortaya çıkan stratejiler arasında en çok filtrelemenin kullanıldığını belirterek, sadece yanlış etiketlenmiş bireyleri

doğru bir şekilde filtrelemeyi değil, aynı zamanda bunları mümkün olduğunda düzeltmeyi amaçlayan bir sınıf gürültü filtresi (Class Noise Cleaner with Noise Scoring - CNC-NOS) önermişlerdir (Luengo, Shim, Alshomrani, Altalhi, & Herrera, 2018). Saez ve diğerleri (2015) sentetik azınlıklı aşırı örnekleme (Synthetic Minority Oversampling Technique - SMOTE) yönteminin gürültülü örneklemlerin ürettiği problemi arttırdığını savunarak, SMOTE yönteminin geliştirilmiş hali olan yinelemeli bölme filtresi (Iterative Partitioning Filter - IPF) adı verilen yinelemeli topluluk tabanlı bir gürültü filtresi geliştirmişlerdir (Sáez, Luengo, Stefanowski, & Herrera, 2015). Fakat bu filtre sadece sınıf gürültüsü problemi ile başa etmek üzere geliştirildiği için problemlere karşı dayanaklı değildir. Başka bir çalışmada ise, Zhang ve diğerleri (2009) gürültülü verideki öğrenme problemlerini inceleyerek gelecekteki bireyleri doğru bir şekilde tahmin etmek için gürültülü akış verisinden dayanıklı bir tahmin modeli oluşturmayı amaçlayan kümelendirilmiş topluluk (Aggregate Ensemble - AE) yöntemini önermişlerdir (P. Zhang, Zhu, Shi, & Wu, 2009).

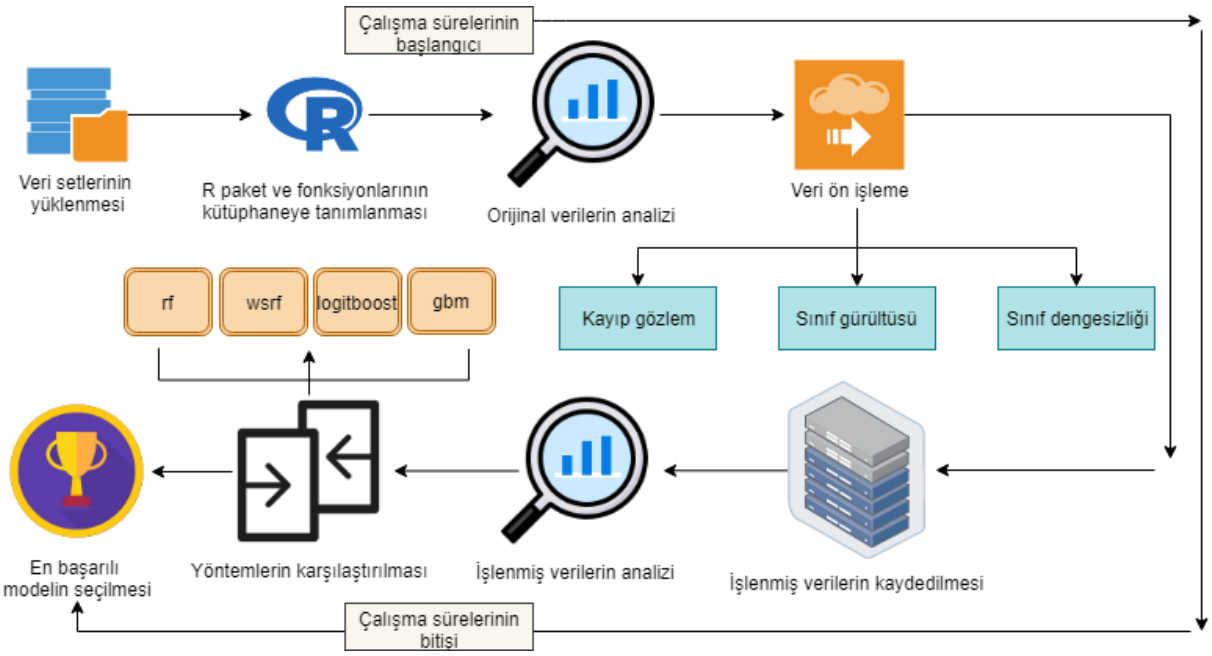
Sağlık alanında dengesiz sınıflandırma problemi ile başa çıkmak için çeşitli çalışmalar yapılmıştır. Krawczyk (2016), meme kanseri malignite derecelendirme için görüntü işleme ve makine öğrenme tekniklerini kullanarak sınıflandırma probleminde az sayıda hasta olması ve yüksek malignite derecesinin erken tespit edilmesi zor ve önemli olduğu için EUSBoost adında bir klinik karar destek sistemi önermişlerdir (Krawczyk, 2016). Liu ve diğerleri (2017), kanser mikrodizin gen ifade verilerinde çoklu sınıf dengesizliğinin ve yüksek boyutun zorluklarıyla başa çıkmak için ağırlıklı aşırı öğrenme makineleri (Weight Extreme Learning Machine - WELM) topluluğuna dayanan bir melez yöntemi önermişlerdir (Z. Liu, Tang, Cai, Wang, & Chen, 2017). Wan ve diğerleri (2017), tarafından proteinlerin hücre içi lokasyonunu tahmin etmek için bir dizi makine öğrenme yaklaşımını kullanılmasıyla karşılaşılan çok etiketli sınıflandırma problemlerinde dengesiz verilerin ele alınması yönelik İnsan Protein Subselüler Lokasyon Tahmini (Human Protein Subcellular Location Prediction - HPSLPred) adı verilen bir sınıflandırıcı araç geliştirilmiştir (Wan et al., 2017). Zhang ve diğerleri (2017), dengesiz zihinsel iş yükü verisinin sınıflandırma problemini ele alarak maliyet duyarlı bir çoğunluk ağırlıklı azınlıklı aşırı örnekleme stratejisini önermişlerdir (J. Zhang, Cui, Li, & Wang, 2017).

Aynı zamanda, kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği problemlerinin çözümü için kullanılan yöntemler ile makine öğrenme yöntemlerine ilişkin medikal alanda yapılan literatür çalışmaları PubMed arama sonuçlarına bakılmıştır. Bu çalışmalar arasında "Machine Learning"[Mesh] AND "class imbalanced" arama terimleriyle 2013 yılından itibaren

yayınlanmış 12 makaleye, "Machine Learning"[Mesh] AND "missing data" arama terimleriyle 2012 yılından itibaren yayınlanmış 42 makaleye, "Machine Learning"[Mesh] AND "noisy data" arama terimleriyle ise 2011 yılından itibaren yayınlanmış 18 makaleye ulaşılmıştır. Bu sonuçlar doğrultusunda gün geçtikçe makine öğrenmesi yaklaşımlarının popüleritesinin arttığını ve son zamanlarda veri ön işlemeye ilişkin problemlerin uluslararası literatürde önem kazanmaya başlandığını söyleyebiliriz.

3. Gereç ve Yöntem

Bu tezin gereç ve yöntem kısmında, kullanılan veri setleri, modelleme algoritmaları, performans ölçüm metrikleri, kayıp gözlem, sınıf gürültüsü, sınıf dengesizliği problemleri çözümünde kullanılan yöntemlerden ve R paketleri hakkında ayrıntılı bilgi verilmiştir. Çalışmanın genel bir akış şeması Şekil 1'de verilmektedir.



Şekil 1: Çalışmanın Genel Akış Şeması

Çalışmanın uygulama aşamasında, Intel(R) Core(TM) i5-6200U CPU @ 2.3GHz 2.40GHz işlemci özelliği, 8,00 GB yüklü bellek(RAM) ve 64 bit İşletim Sistemi, x64 tabanlı işlemci sistem özelliğine sahip bir bilgisayar kullanılmıştır. Ayrıca, çalışmadaki tüm analizler RStudio 1.2.1335 - Windows 7+ (64-bit) programında yapılmıştır. Çalışmada kullanılacak veriler indirildikten ve R paketleriyle fonksiyonları bilgisayara yüklendikten sonra, orijinal verilerin topluluk öğrenme yöntemleri kullanılarak analizi yapılmıştır. Daha sonra orijinal verilerde yer alan olası problemler için veri ön işleme aşaması tamamlanmıştır. Böylelikle, işlenmiş veriler kaydedildikten sonra analizler tekrar yapılmıştır. Son olarak, hem orijinal verilerin hem de işlenmiş verilerin performansları karşılaştırılıp, algoritmaların çalışma süreleri de hesaplanarak en başarılı modele karar verilmiştir.

3.1. Çalışmada Kullanılan Veri Setleri

Çalışmada kullanılan *Cleveland*, *Heart*, *Hepatitis*, *Lymphography*, *Mammography*, *Newthyroid*, *Pima* ve *Thyroid* veri setleri Evrimsel Öğrenmeye Dayalı Bilgi Çıkarımı (Knowledge Extraction based on Evolutionary Learning - KEEL) adı verilen açık kaynaklı bir java yazılım aracı olan Genel Kamu Lisansı sürüm 3 (General Public License version 3 - GPLv3) ile tasarlanmış veri tabanından elde edilmiştir. KEEL, regresyon, sınıflandırma, denetimsiz öğrenme gibi birçok veri madenciliğini barındıran bir yazılım aracı olarak geliştirilmiştir. Farklı yaklaşımlara dayanan çeşitli evrimsel öğrenme algoritmaları içeren KEEL, aynı zamanda kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi birçok probleme sahip veri setleri için farklı veri ön işleme yöntemlerini de içinde barındırmaktadır (Alcala-Fdez et al., 2011). KEEL yazılımının veri seti havuzunda, literatürde tüm araştırmacıların çalışmalarında kullanılması için halihazır bulunan çok çeşitli alanlardan elde edilmiş veri setleri mevcuttur. Bu çalışma kapsamında, sağlık alanından elde edilmiş hastalık tanısı sınıflandırma veri setleri ile çalışılmıştır. Bu veri setlerinin iki tanesi kalp hastalığı tanısı (*Cleveland* ve *Heart*), iki tanesi tiroid hastalığı tanısı (*Newthyroid* ve *Thyroid*) hepatit hastalığı tanısı (*Hepatitis*), lenfödem hastalığı tanısı (*Lymphography*), meme kanseri hastalığı tanısı (*Mammographic*) ve diyabet hastalığı tanısı (*Pima*) verileridir. Bu veri setleriyle ilgili bilgiler detaylı olarak bahsedilmiştir.

- ***Cleveland:*** Bu veri, 1988 yılında Cleveland Clinic adıyla kurulan, Cleveland, Ohio merkezli bir Amerikan Akademik Tıp Merkezi'ndeki kalp hastalarından elde edilmiş bir kalp hastalığı veri setidir. 1989 yılı itibariyle birçok çalışmada kullanılan bu kalp hastalığı veri seti, toplamda 76 değişken içermektedir. Araştırmacılar tarafından veri tabanında yayınlanan veri seti ise, 14 değişkenden ve 303 gözlemden oluşmaktadır. *Cleveland* kalp hastalığı verisinin bağımlı değişkeni beş sınıftan oluşmaktadır. Bu sınıflar kalp hastalığı tanısının var olup olmadığını ifade etmektedir. 0 değeri, kalp hastalığının yokluğunu ifade ederken, 1, 2, 3 ve 4 değerleri ise farklı şiddetlerdeki kalp hastalığının varlığını ifade etmektedir. Bağımsız sürekli değişkenler, *Age* (hasta yaşı - yıl), *Trestbps* (dinlenme kan basıncı - mmHg), *Chol* (serum kolesterol - mg/dl), *Thalach* (elde edilen maksimum kalp atış hızı), *Oldpeak* (dinlenmeye kıyasla egzersizle ortaya çıkan ST depresyonu), *Ca* (floroskopi ile renklendirilmiş ana damarların sayısı - 0 ile 3 arasında tam sayı değeri) değişkenleridir. Bağımsız kategorik değişkenler ise, *Sex* (hasta cinsiyeti - 1: erkek; 2: kadın), *Cp* (göğüs ağrısı

tipi - 1: tipik anjin; 2: atipik anjin; 3: anjinal olmayan ağrı; 4: asemptomatik), *Fbs* (açlık kan şekeri $>$ 120 mg / dl - 1: doğru; 0: yanlış), *Restecg* (elektrokardiyografik sonuçların dinlenmesi - 0: normal; 1: ST-T dalga anormalliğe sahip (T dalgası inversiyonları ve/veya ST)), *Exang* (egzersize bağlı anjin - 1: evet; 0: hayır), *Slope* (egzersiz ST segment tepesinin eğimi - 1: yukarı; 2: düz; 3: aşağı), *Thal* (3: normal; 6: kalıcı kusur; 7: düzelebilir kusur) değişkenleridir.

- ***Heart:*** Bu veri, 90'lı yıllarında başında Avrupa Birliği destekli bir proje olan Avrupa Statlog projesi (European Statlog project) kapsamında elde edilmiş bir veri setidir. Bu projenin amacı, farklı çalışma alanlarından elde edilmiş büyük ölçekli gerçek dünya verilerinin çeşitli sınıflandırma algoritmaları ile performanslarının karşılaştırılmasıdır. Bu proje kapsamında, *Heart* verisi dahil olmak üzere 20 farklı veri seti ve 23 farklı sınıflandırma algoritması kullanılmıştır. *Heart* verisi, 1998 yılı itibariyle birçok çalışmada kullanılan bir kalp hastalığı tanısı veri setidir. Bu veri seti, toplamda 75 değişken içermesine rağmen, proje araştırmacıları tarafından yayınlanan 14 değişkenden ve 270 gözlemden oluşmaktadır. *Heart* kalp hastalığı verisinin bağımlı değişkeni iki sınıftan oluşmaktadır. Bu sınıflar, kalp hastalığı tanısının var olup olmadığını ifade etmektedir. 1 değeri, kalp hastalığının yokluğunu; 2 değeri ise kalp hastalığının varlığını ifade etmektedir. Bu veriye ait bağımsız değişkenler, *Cleveland* kalp hastalığı verisi ile aynıdır.
- ***Hepatitis:*** Bu veri, 1988 yılında Carnegie-Mellon Üniversitesi'nden Gail Gong tarafından bağışlanmıştır. *Hepatitis* verisi, 19 değişkenden ve 155 gözlemden oluşmaktadır. Bu veri setinin bağımlı değişkeni, hepatit hastalarının test sonuçlarıyla ilgilendiği için iki sınıftan oluşmaktadır. Bu sınıflardan 1 değeri, hastanın pozitif test sonucunu ifade ederken; 2 değeri, hastanın negatif test sonucunu ifade etmektedir. Bağımsız sürekli değişkenler, *Age* (hasta yaşı - yıl), *Bilirubin* (mg/dL), *AlkPhosphate* (U/L), *Sgot* (U/L), *AlbuMin* (g/dL), *ProTime* (s) değişkenleridir. Bağımsız kategorik değişkenler ise, *Sex* (hasta cinsiyet - 0: erkek; 1: kadın), *Steroid* (0: yok; 1: var), *Antivirals* (0: yok; 1: var), *Fatigue* (0: yok; 1: var), *Malaise* (0: yok; 1: var), *Anorexia* (0: yok; 1: var), *LiverBig* (0: yok; 1: var), *LiverFirm* (0: yok; 1: var), *SpleenPalpable* (0: yok; 1: var), *Spiders* (0: yok; 1: var), *Ascites* (0: yok; 1: var), *Varices* (0: yok; 1: var) ve *Histology* (0: yok; 1: var) değişkenleridir.

- ***Lymphography:*** Bu veri, 1988 yılında Yugoslavya'da bulunan Üniversite Tıp Merkezi, Onkoloji Enstitüsü'nde lenfografi alanında çalışan Zwitter ve Soklic tarafından bağışlanmıştır (Garcia ve Ramani, 2012). Lymphography verisi, 19 değişkenden ve 148 gözlemden oluşmaktadır. Bu veri setinin bağımlı değişkeni, normal (0), metastaz (1), malign lenf (2), fibroz (3) olmak üzere dört sınıftan oluşmaktadır. Lenfografi verisinin bağımsız sürekli değişkenleri, *lym nodes diminish* (lenf düğümlerinin azalması - 0 ile 3 arasında bir tam sayı), *lym nodes enlarge* (lenf bezleri genişlemesi - 1 ile 4 arasında bir tam sayı) ve *no of nodes in* (düğüm sayısı - 0 ile 70 arasında bir tam sayı) değişkenleridir. Bağımsız kategorik değişkenler ise, *Lymphatics* (lenfatikler - 1: normal; 2: kemerli; 3: deforme olmuş; 4: yerinden olmuş), *Block of afferent* (afferent bloğu - 0: hayır; 1: evet), *Block of lymph.c* (c lenf bloğu, alt ve üst kapaklar - 0: hayır; 1: evet), *Block of lymph.s* (s lenf bloğu, tembel insizyon - 0: hayır; 1: evet), *Bypass* (baypas ameliyatı geçirme - 0: hayır; 1: evet), *Extravasates* (lenf kuvveti - 0: hayır; 1: evet), *Regeneration* (yenilenme - 0: hayır; 1: evet), *Early uptake in* (erken alım - 0: hayır; 1: evet), *Changes in lymph* (lenf değişiklikleri - 1: fasulye şekli; 2: oval; 3: yuvarlak), *Defect in node* (düğümdeki bozukluk - 0: yok, 1: lacunar; 2: marjinal lacunar; 3: merkezi lacunar), *Changes in node* (düğümdeki değişiklik - 0: yok, 1: lacunar; 2: marjinal lacunar; 3: merkezi lacunar), *Changes in structure* (yapıdaki değişiklikler - 0: yok; 1: grenli; 2: damla benzeri; 3: kaba; 4: seyreltilmiş; 5: retiküler; 6: soyulmuş; 7: soluk), *Special forms* (özel şekiller - 0: yok; 1: chalices; 2: vesicles), *Dislocation of* (yerinden çıkma - 0: hayır; 1: evet) ve *Exclusion of no* (dışta bırakılanlar - 0: hayır; 1: evet) değişkenleridir.
- ***Mammographic:*** Bu veri, 2003-2006 yılları arasında Erlangen-Nürnberg Üniversitesi, Radyoloji Enstitüsü'nde toplanmıştır. Meme kanseri taramasında elde edilen mamografik test verisi, 6 değişkenden ve 829 gözlemden oluşmaktadır. *Mammographic* verisi bağımlı değişkeninin 0 değeri, negatif test sonucunu, 1 değeri, pozitif test değerini temsil eden ikili sınıf değişkenidir. Bu veri setinin bağımsız sürekli değişkeni *Age* (hasta yaşı) değişkeni iken, bağımsız kategorik değişkenleri ise, Meme Görüntüleme Raporlama ve Veri Sistemi'nin kısaltması olan *BIRADS* (Breast Imaging Reporting and Data System) değişkenidir. *BIRADS* kodları, bir radyolog tarafından verilen 0 ile 6 arasında değişen değerler alabilen mamografi değerlendirme kodlarıdır. 0 değeri, iyi huylu anlamına gelirken, 1 ile 5 arasındaki değerler malign olduğu anlamını taşımaktadır. *Shape* (kitle şekli - 1: yuvarlak; 2: oval; 3: lobüler; 4:

düzensiz), *Margin* (kitle marjini - 1: sınırlı; 2: çevrelenmiş; 3: lekelenmiş; 4: gizlenmiş; 5: kötü tanımlanmış; 6: belirtilmiş) ve *Density* (kitle yoğunluğu - 1: yüksek; 2: iso; 3: düşük; 4: yağ içeren) değişkenleridir.

- ***Newthyroid:*** Bu veri, 1987 yılında düzenlenen Makine Öğrenme Atölyesi (Machine Learning Workshop)'ne katılan Ross Quinlan'ın Irvine Kaliforniya Üniversitesi (University of California Irvine - UCI)'ne ziyareti sırasında bırakılmıştır. *Newthyroid* verisi, tiroid hastalığı tanısına ait altı veri setinin depolandığı bir veri tabanında bulunmaktadır. Bu veriler, 6 değişkenden ve 215 gözlemden oluşmaktadır. *Newthyroid* verisinin bağımlı değişkeni olan sınıf değişkeni, normal (1), hipertiroid (2) ve hipotiroid (3) olmak üzere üç düzeyden oluşmaktadır. Bağımsız sürekli değişkenler ise, *T3resin* ($\mu g/dl$), *Thyroxin* ($\mu g/dl$), *Triiodothyronine* ($\mu g/dl$), *Thyroidstimulating* ($\mu g/dl$) ve *TSH* (tiroid uyarıcı hormon - mIU / L) tiroid hormonu değişkenleridir.
- ***Pima:*** Bu veri, 2000 - 2004 yılları arasında Phoenix, Arizona'da yaşayan 21 yaşından büyük Pima Hintli kadınlardan toplanmıştır. Diyabet hastalığı verisinin aslı, Ulusal Diyabet Enstitüsü ve Sindirim, Böbrek Hastalıkları Enstitüsü (National Institute of Diabetes and Digestive and Kidney Diseases)'nde bulunmaktadır. Pima verisi, 9 değişkenden ve 768 gözlemden oluşmaktadır. Bağımlı değişken, diyabet hastalığının yokluğu (0: test negatif) ve diyabet hastalığının varlığı (1: test pozitif) olmak üzere iki sınıftan oluşmaktadır. Diyabet hastalığı tanısı koyulurken, *Preg* (hamilelik sayısı), *Plas* (plazma glikoz konsantrasyonu), *Pres* (diyastolik kan basıncı - $mm\,Hg$), *Skin* (triceps cilt kıvrım kalınlığını - mm), *Insu* (2 saatlik serum insülini - mu U/mL), *Mass* (vücut kitle indeksi - $kg\ /\ m\text{^}2$), *Pedi* (diyabet soyağacı fonksiyonu) ve *Age* (hasta yaşı - yıl) bağımsız sürekli değişkenlerden yararlanılmıştır.
- ***Thyroid:*** Bu veri, *Newthyroid* verisi gibi tiroid hastalığı veri tabanından alınmıştır. Ross Quinlan'ın UCI ziyareti sırasında bırakılan bu veri, 1984 yılından 1987 yılının başlarına kadar Sidney, Avustralya'da bulunan Garvan Enstitüsü'nde toplanmış bir tiroid hastalığı tanısı verisidir. *Thyroid* verisi, orijinalinde 9172 hasta kaydından oluşmasına rağmen araştırmacılar tarafından *thyroid0387* koduna sahip bir sürümü yayınlanmıştır. *Thyroid* verisi, 22 değişkenden ve 7200 gözlemden oluşmaktadır. *Thyroid* verisinin bağımlı değişkeni olan sınıf değişkeni, normal (1), hipertiroid (2) ve hipotiroid (3) olmak üzere üç düzeyden oluşmaktadır. Bağımsız sürekli değişkenler ise, *Age* (hasta yaşı - yıl) ve *TSH, T3 (Triiodothyronine, T3-RIA), TT4 (total serum*

thyroxine), T4U (thyroxine) ve *FTI (free thyroxine index)* tiroid hormonları ölçüm değerlerine ait değişkenlerdir. Bağımsız kategorik değişkenler ise, *Sex* (hastanın cinsiyeti - 0: erkek; 1: kadın), *On thyroxin* (0: yok; 1: var), *Query on thyroxine* (0: yok; 1: var), *On antithyroid medication* (0: yok; 1:var), *Sick* (0: yok; 1:var), *Pregnant* (0: yok; 1: var), *Thyroid surgery* (0: yok; 1: var), *I131 treatment* (0: yok; 1: var), *Query hypothyroid* (0: yok; 1: var), *Query hyperthyroid* (0: yok; 1: var), *Lithium* (0: yok; 1: var), *Goitre* (0: yok; 1: var), *Tumor* (0: yok; 1: var), *Hypopituitary* (0: yok; 1: var) ve *Psych* (0: yok; 1: var) değişkenleridir.

3.2. Veri Ön İşleme

Tezin bu bölümünde, makine öğrenme sınıflandırma algoritmaları kullanılmadan önce, sağlık verilerinde karşılaşılabilecek kayıp gözlem, gürültülü veri ve sınıf dengesizliği gibi olası problemler üzerinde durulacaktır. Sağlık verileriyle yapılan çalışmalarda sıklıkla ortaya çıkan bu problemler sonraki alt başlıklarda detaylıca anlatılmıştır ve bu problemlere ilişkin çözüm önerileri sunulmuştur.

3.2.1. Problem1: Kayıp Gözlem

Sağlık araştırmalarında kayıp veri sorununun bulunması olağan bir durumdur. İyi tasarlanmış ve kontrollü bir çalışmada bile, eksik veriler ortaya çıkabilmektedir. Kayıp gözlem (missing value) olarak adlandırılan bu durumda, değişken ya da değişkenlerde gözlem değerlerinin bulunmaması sonucunda bu gözlemlere ilişkin hücreler boş kalmaktadır. Bu durumda standart istatistiksel yöntemlerin kullanıldığı yazılım ve programlama dilleri kullanılarak yapılan analizlerde yer alan tüm değişkenlerde kayıp bilgi olmadığı varsayılması problem yaratmaktadır. Kayıp gözlem probleminden dolayı bazı değişkenler için nispeten az sayıda gözlem bulunurken, örneklem genişliği de aynı oranda azalabilmektedir. Bu durumda, güven aralıkları daha az güvenilir, istatistiksel güç daha düşük ve parametre tahminleri yanlı (bias) olabilmektedir (R. Little & Rubin, 1987).

Kayıp gözlem tipini belirlemek ve kayıp gözlem problemiyle baş etmek için önerilen farklı yöntemlerinin nasıl uygulanacağı literatürde tartışma konusu olmuştur (Dong & Peng, 2013; Rubin, 1987). Kang (2013) kayıp gözlem varlığının istatistiksel gücü azaltabileceğini ifade etmiştir. Bunun yanı sıra, kitle parametrelerinin yanlı tahminlenebileceğini ve örneklemin temsil gücünün azalabileceğini vurgulamıştır. Kayıp gözlem, birim düzeyinde yanıtlanmama (unit nonresponse) veya madde düzeyinde yanıtlanmama (item nonresponse) olmak üzere iki

durumda gerçekleşmektedir (Kang, 2013). Örneğin, yapılan bir sağlık araştırmasında, bir katılımcı araştırmaya katılmayı reddedebilir veya araştırmanın yapıldığı zamanda orada bulunmayabilir. Bu durumda, katılımcının hiçbir bilgisi alınmadığından birim düzeyinde yanıtlanmama durumu söz konusudur. Diğer bir durumda ise, katılımcının bazı değişkenlere ait bilgilerinin elde edilememesi veya bir değişken ile ölçülmek istenen özelliğin yeterli bulunmaması gibi nedenlerle kayıp gözlemler oluşmakta ve bu problem madde düzeyinde yanıtlanmama olarak adlandırılmaktadır (R. Little & Rubin, 1987). Sağlık araştırmalarında, karşılaşılacak kayıp gözlem tipine göre hangi çözüm önerilerinin sunulacağı ve kullanılacak atama yönteminin belirlenmesi kayıp gözlem mekanizmalarına (missing data mechanisms) bağlıdır. Kayıp veri tipleri, kayıp veri mekanizmaları ile kayıp ve gözlemlenen değerler arasındaki ilişkiye dayanarak üç mekanizma için incelenmektedir. Kayıp gözlem mekanizmalarının matematiksel gösterimi için kullanılan ifadeler aşağıda kısaca özetlenmiştir.

$\boldsymbol{Y}$**:** Ham veri setinin (bağımlı ve bağımsız değişkenler) hem gözlemlenmiş değerlerini $\boldsymbol{Y_{obs}}$ ve hem de kayıp değerlerini $\boldsymbol{Y_{mis}}$ içeren bir matristir.

$\boldsymbol{R}$**:** $\boldsymbol{Y}$ matrisindeki ilgili değişken değerlerinin gözlenen (0) veya kayıp (1) olup olmadığını ifade eden bir eksiklik matrisidir (missingness matrix).

$\boldsymbol{q}$**:** Eksiklik matrisi $\boldsymbol{R}$ ile ham veri seti $\boldsymbol{Y}$ arasındaki ilişkiyi tanımlayan bir parametreler vektörürdür. Burada $\boldsymbol{q}$ parametresi, kayıp veri mekanizması olarak ifade edilir ve rastgele kayıp (missing at random - MAR), tamamıyla rastgele kayıp (missing completely at random - MCAR) ve rastgele olmayan kayıp (missing not at random - MNAR) arasında ayırım yapılmasını sağlar.

Eksiklik ve ihmal edilebilirlik kavramları her üç mekanizma için Şekil 2'de ve Tablo 1'da özetlenmiştir (Nakagawa & Freckleton, 2011). Tablo 1'de, üç değişkene sahip bir veri seti üzerinden $\boldsymbol{Y}$ ve $\boldsymbol{R}$ matrisi gösterilmiştir. Buna göre, gözlenen değerler, obs, kayıp değerler, mis olarak ifade edilmiştir. $\boldsymbol{Y_{obs}}$ için her üç değişken $\boldsymbol{v_1}$, $\boldsymbol{v_2}$ ve $\boldsymbol{v_3}$ olarak gösterilirken, $\boldsymbol{Y_{mis}}$ için her üç değşkeni ikili olarak $\boldsymbol{m_1}$, $\boldsymbol{m_2}$ ve $\boldsymbol{m_3}$ olarak gösterilmektedir. Örnek veri setinde, $\boldsymbol{v_3}$ değişken değerleri ölçülememiştir. Sadece mekanizmaların anlaşılması için eklenmiştir, fakat genellikle bu tür değişkenler, $\boldsymbol{Y}$ ve $\boldsymbol{R}$ matrislerine dahil edilemezler.

- **MAR:** Kayıp gözlem olasılığının gözlenen değişkenlere bağlı olduğu durum, rastgele kayıp olarak adlandırılmaktadır. Bu durumda, MAR mekanizmasında eksik olma durumu, diğer gözlenen değişkenlerle ilişkilidir (R. J. A. Little & Rubin, 1991). Diğer bir deyişle, MAR aslında bir veya daha fazla ölçülen değişken ile eksik verilerin

olasılığı arasında sistematik bir ilişki olduğu anlamına gelmektedir. Kayıp gözlem mekanizması, MAR olarak tanımlandığında beklenti maksimizasyonu (expectation maximization), regresyon atama (regression imputation) ve çoklu atama yöntemleri kayıp gözlem probleminin çözümünde uygun atama yöntemleri olarak kullanılmaktadır (Sandip Sinharay, Stern, & Russell, 2001). MAR mekanizmasının olasılık dağılımı aşağıdaki gibidir:

$$p(\boldsymbol{R}|\boldsymbol{Y_{obs}}, \boldsymbol{q}) \quad (1)$$

Bu formüle göre, eksikliğin yalnızca gözlenen değişkenlere $\boldsymbol{Y_{obs}}$ bağlı olduğu ve bu ilişkilerin $\boldsymbol{q}$ tarafından kontrol edildiği anlamına gelmektedir. Tablo 1'e göre, $\boldsymbol{v_2}$ değişkeni kayıp değerleri, gözlenen $\boldsymbol{v_1}$ değişkenine bağlıysa, $\boldsymbol{v_2}$ MAR'dir.

Tablo 1: Eksiklik ve İhmal Edilebilirlik Kavramlarının Matris Gösterimi

Birey	$Veri[\boldsymbol{Y} = (\boldsymbol{Y_{obs}}, \boldsymbol{Y_{mis}})]$			$Eksiklik[\boldsymbol{R}]$		
	$\boldsymbol{v_1}$	$\boldsymbol{v_2}$	$\boldsymbol{v_3}$	$\boldsymbol{m_1}$	$\boldsymbol{m_2}$	$\boldsymbol{m_3}$
1	obs	mis	mis	0	1	1
2	obs	obs	mis	0	0	1
3	obs	obs	mis	0	0	1
4	obs	mis	mis	0	1	1
5	obs	obs	mis	0	0	1
6	obs	obs	mis	0	0	1
7	obs	obs	mis	0	0	1
8	obs	mis	mis	0	1	1
9	obs	mis	mis	0	1	1
10	obs	obs	mis	0	0	1

obs: gözlemlenmiş değerler, mis: kayıp değerler

- **MCAR:** Kayıp gözlem olasılığının gözlenen veya gözlemlenmeyen değişkenlerle ilişkili olmadığı durum, tamamıyla rastgele kayıp mekanizması olarak tanımlanmaktadır (R. Little & Rubin, 1987). Burada en dikkat edilmesi gereken varsayım ise, rastgelelik varsayımıdır. Bu varsayımın sağlanması için, eksik değişken

değerlerinin herhangi bir değişkenin değerleriyle ilişkili olmaması gerekmektedir. MCAR varsayımı sağlandığında, liste bazında silme (listwise deletion) veya çiftler bazında silme (pairwise deletion) yaklaşımları ile tam gözlemlerin kullanılması sağlanmaktadır (Acock, 2005). Bu yöntemlerin basit kullanıma sahip olmaları ve hesaplama sürelerinin kısalığı, yöntemlerin tercih nedenleri arasında yer almaktadır. Kayıp gözlem mekanizmasının MCAR özelliği taşımadığı durumda bu yöntemlerin kullanılmasıyla yanlı çözümler elde edilebilmektedir. Bu nedenle, MCAR özelliğinin bulunmadığı kayıp gözlem mekanizmalarında, dayanıklı yöntemlerin tercih edilmesi çözüm için daha uygun olmaktadır (R. J. A. Little & Rubin, 1991). Aynı zamanda, MCAR mekanizması, MAR mekanizmasına göre daha tutucu bir yaklaşım göstererek eksikliğin verilerle tamamen ilişkisiz olduğunu varsayımına dayanmaktadır. MCAR mekanizmasının olasılık dağılımı aşağıdaki gibidir:

$$p(\boldsymbol{R}|\boldsymbol{q}) \tag{2}$$

Bu formüle göre, eksiklik olasılığının verilere (ne gözlenen $\boldsymbol{Y_{obs}}$ ne de kayıp $\boldsymbol{Y_{mis}}$) bağlı olmadığını, fakat $\boldsymbol{R}$ matrisindeki bir değişken değerinin 0 veya 1 olmasının hala $\boldsymbol{q}$ tarafından kontrol edildiği anlamına gelmektedir. Tablo 1'e göre, kayıp değerler gözlenen bir değişkene ($\boldsymbol{v_1}$) veya $\boldsymbol{v_2}$ değişken değerlerine bağlı değilse, $\boldsymbol{v_2}$ MCAR'dır.

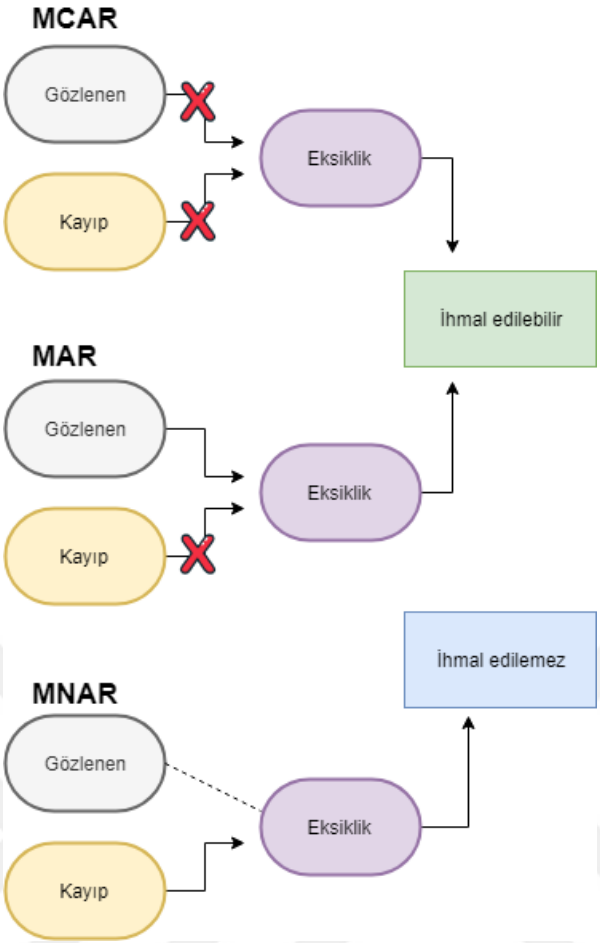
- **MNAR:** Kayıp gözlem olasılığının gözlenen veya gözlemlenmeyen değişkenlerle ilişkili olduğu durum, rastgele olmayan kayıp mekanizma olarak tanımlanmaktadır. Diğer bir ifade ile verideki eksiklik rastgele değildir ve kayıp veri diğer değişkenleri kullanarak tahmin edilemediğinden bu durumda, MNAR mekanizmasında eksik değerler gözlemlenmeyen değerlere bağlıdır. Atama yöntemlerinin tümü NMAR mekanizması durumunda yanlı olmaktadırlar (R. Little & Rubin, 2002). MNAR mekanizmasının olasılık dağılımı aşağıdaki gibidir:

$$p(\boldsymbol{R}|\boldsymbol{Y_{obs}}, \boldsymbol{Y_{mis}}, \boldsymbol{q}) \tag{3}$$

Bu formüle göre, $\boldsymbol{R}$ matrisindeki bir değişken değerinin 0 veya 1 olması olasılığının hem gözlenen $\boldsymbol{Y_{obs}}$ hem de kayıp $\boldsymbol{Y_{mis}}$ değişkenlere bağlı olduğunu ve bu ilişkinin $\boldsymbol{q}$ tarafından kontrol edildiği anlamına gelmektedir. Tablo 1'e göre, $\boldsymbol{v_2}$ değişkeni kayıp değerleri yine $\boldsymbol{v_2}$ değişkenine bağlıysa, $\boldsymbol{v_2}$ MNAR'dır. Bu gibi kayıp değerler, tamamen gözlenen bir değişken olan $\boldsymbol{v_1}$ değişkeni ile de ilişkili olabilir. Bu durum özel bir durum olduğundan MNAR eksikliğinin olasılığı (probability of MNAR

missingness) tamamen, hem $\boldsymbol{v_1}$ hem de $\boldsymbol{v_2}$ ile ilgili olan kayıp değerlere sahip bireylere $\boldsymbol{Y_{mis}}$, diğer bir ifadeyle, $p(\boldsymbol{R}|\, \boldsymbol{Y_{mis}}, \boldsymbol{q})$ olasılığına bağlıdır. Daha karmaşık bir başka MNAR tipi, $\boldsymbol{v_2}$ değişkeninin tamamen gözlemlenmemiş bir değişkene bağlı olduğu, örneğin Tablo 1'deki $\boldsymbol{v_3}$ değişkenidir. Tablo 1'deki örnek için, eğer $\boldsymbol{v_3}$ değişkeni için ölçüm yapılmazsa, $\boldsymbol{v_2}$ değişkeninin MAR varsayımı kayıp değerleri MNAR olacaktır. Böyle bir araştırmada sadece MNAR varsayılır. Bunun sebebi ise, $\boldsymbol{Y_{mis}}$'teki gözlemlenmemiş değerlere bağlı olmasıdır.

Şekil 2'de, üç kayıp veri mekanizması (MCAR, MAR ve MNAR) ve ihmal edilebilirliği (kayıp veri mekanizmasını modellememiz gerekip gerekmediği) gözlenen verilerle ($\boldsymbol{Y_{obs}}$), kayıp verilerle ($\boldsymbol{Y_{mis}}$), eksiklik matrisi ($\boldsymbol{R}$) ve ilişkileri ($\boldsymbol{q}$; eksikliği açıklayan parametreler - mekanizma) gösterilmiştir. Kesintisiz oklar, noktalı oklar ve çarpı işaretli oklar sırasıyla "bağlantı", "olası bağlantı" ve "bağlantı yok" anlamına gelmektedir. İhmal edilebilirliği ve eksikliği birleştiren çizgiler, üç mekanizmayı iki ihmal edilme kategorisinde gruplandırmaktadır. Ayrıca, ham MCAR, MAR ve MNAR tipleri mevcut olmadığından tüm eksiklikler MAR eksikliğinin bir tipi olarak düşünülebilmektedir (Nakagawa & Freckleton, 2011).



Şekil 2: MAR, MCAR ve MNAR Mekanizmalarının Eksikliği ve İhmal Edilebilirliği

Kayıp gözlem mekanizmalarında ihmal edilebilirlik (ignorability), bir başka önemli kavramdır. MNAR eksikliği (MNAR missingness), ihmal edilemez (non-ignorable) iken; MAR ve MCAR eksikliği, ihmal edilebilir (ignorable) olarak ifade edilmektedir (R. J. Little & Rubin, 2002). İhmal edilebilirlik, kayıp verilere atama ve arttırma yaparken verinin eksikliğinin görmezden gelinip gelinemeyeceğini ifade ettiğinden, kayıp verilerin silineceği anlamına gelmemektedir. MAR ve MCAR mekanizmalarında atama ve arttırma yapılırken, verilerin nasıl eksik olduğuna dair özel bir varsayımda bulunulması gerekmemektedir. Diğer yandan, ihmal edilemez MNAR eksikliği, bu tür varsayımların kayıp değerlerini tamamlamak için bir varsayım gerektirmektedir (Papageorgiou, Grant, Takkenberg, & Mokhles, 2018).

Bazı çalışmalarda, tam veri seti elde edilebilmesi araştırmanın kapsamı nedeniyle mümkün olmadığında, kayıp verilerin ihmal edilebilir olduğu düşünülerek analiz sırasında kayıp gözlemler göz ardı edilebilmektedir. Kayıp verilerin, ihmal edilebilirliğine karşı geliştirilen kayıp gözlem mekanizmaları, uygun analiz yöntemlerinin belirlenmesinde ve sonuçların

yorumlanmasında oldukça önemlidir. Rastgele kayıp ve tamamıyla rastgele kayıp varsayımlarına bağlı olarak kayıp veri mekanizmasının "ihmal edilebilir" olup olmadığı kararı verilebilmektedir. Kayıp veri mekanizması, MAR ya da MCAR varsayımlarının sağlanması ve kayıp verilerin varlığında kestirilen parametreler ile normal şartlar altındaki kestirim parametreleri arasında bir ilişki varsa ihmal edilebilmektedir. MAR varsayımı sağlanmadığında ise, kayıp veri mekanizması ihmal edilebilir değildir. İhmal edilebilir kayıp veri mekanizması olması durumu kayıp verilerin analiz dışı bırakılabilmesini sağlayacağından araştırmacının işini kolaylaştıracak, aksi halde kayıp veri mekanizması modellenecektir. Veri seti ihmal edilemez veriler içerdiğinde genellikle hangi modellerin daha uygun olacağı konusunda yeterli bilgi bulunmamaktadır. Allison (2009), araştırma sonuçları seçilen modele oldukça duyarlı olacağından, kayıp veri sürecinin doğasına yönelik oldukça sağlam ön bilgiler olduğunda ihmal edilemez kayıp verilerle etkili bir kestirim yapılabileceğini belirtmiştir (Allison, 2009).

Literatürde, kayıp gözlemlerin kabul edilebilir bir oranına ilişkin tam olarak belirlenmiş bir kesim noktası (cutoff) bulunmamaktadır. Bu konuyla ilgili olarak Schafer (1999), %5 veya daha az bir kayıp gözlem oranının önemsiz olduğunu savunmuştur (Joseph L Schafer, 1999). Bennett (2001), verilerin %10'undan daha fazlasının kayıp gözlem olduğu durumlarda istatistiksel analizlerin yanlı bir kestirime sahip olacağını vurgulamıştır (Bennett, 2001). Yine benzer şekilde Acuña ve Rodriguez (2004), kayıp değerin istatistiksel analizde yaygın bir sorun olduğunu, kayıp değer oranının %1'den az olduğu durumda önemsiz kabul edildiğini; %1-5 arasında idare edilebilir olduğunu; %5-15 arasında gelişmiş yöntemler gerektiğini ve %15'ten fazla olduğunda büyük oranda verinin yapısını etkileyeceğini belirtmiştir (Acuña & Rodriguez, 2004).

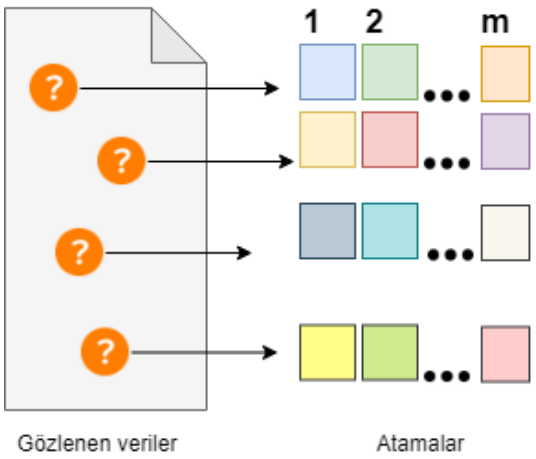
Kayıp gözlem ile baş etmenin en iyi yöntemi, çalışmayı doğru planlamak ve veriyi özenle toplamak olsa da; özellikle klinik çalışmalarda bu konu daha hassas hale geldiğinden; kayıp veri miktarını en aza indirmek için bazı atama yöntemleri önerilmiştir. Önerilen yöntemler, iki ana başlıkta yer almaktadır. Klasik yöntemler (conventional methods) olarak; liste bazında silme, çiftler bazında silme yöntemi, hot-deck atama, son gözlemi ileri taşıma yöntemi (last observation carried forward), regresyon atama, interpolasyon atama yöntemi (interpolation imputation method) gibi yöntemler bulunmaktadır (Acock, 2005). Kayıp gözlem mekanizmaları için ileri atama yöntemleri olarak ise, en çok olabilirlik yöntemi (maximum likelihood method), çoklu atama yöntemi, tekli rastgele atama (single random imputation),

çoklu rastgele atama (multiple random imputation) yöntemleri geliştirilmiştir (Sandip Sinharay et al., 2001).

3.2.1.1. Çoklu Atama Yöntemi

Çoklu atama yöntemi, kayıp gözleme sahip veriler için aynı anda birden fazla değişken için kayıp veri ataması yapmaktadır. Kayıp verinin ait olduğu değişken için belli bir modele bağlı dağılım belirlenerek, bu dağılımdan rastgele üretilen olası atama değerleri ile atama tamamlanmaktadır. Her bir değişken için yapılan atama sonucunda, veri setlerinin sonuçları birleştirilerek tamamlanmış veri setleri elde edilmektedir. Bu tamamlanmış veri setlerinden değişken için en olası değerler, tahmin değerleri olarak atanmaktadır (Jakobsen, Gluud, Wetterslev, & Winkel, 2017; P. Li, Stuart, & Allison, 2015).

Çoklu atama yönteminin atama değeri (imputation), *m* ve yineleme sayısı (iteration), *t* olmak üzere iki önemli parametresi vardır. Atama sayısı, bir değişken için ne kadar atama yapılacağını ifade ederken, yineleme sayısı ise, atama modeli kurulurken ne kadar tekrar edeceğini ifade etmektedir. Şekil 3'de çoklu atama çalışma disiplini gösterilmiştir. Literatürde kullanılan birçok çoklu atama yöntemi vardır. Bunlar değişken tipine göre farklılık göstermektedir. Eğer bağımsız sürekli değişken normal dağılıma sahip değilse, tahmini ortalama eşleştirme (predictive mean matching - pmm) yöntemi veya farklı regresyon yöntemleri tercih edilebilmektedir. Eğer bağımsız kategorik değişken ikili sınıfa sahipse, lojistik regresyon (logistic regression - logreg) modeli; çoklu sınıfa sahipse, çoklu kategorili logistik regresyon (multinomial logit regression - polyreg) modeli; sıralı çoklu sınıfa sahipse, oransal regresyon (proportional odds model - polr) tercih edilebilmektedir. Atama modellerine karar verilirken, modeller ile değişkenler arasındaki ilişkiyi en iyi şekilde tahminleyebilecek bir yapı oluşturulmalıdır. Çoklu atamalar yapılırken değişkenler için atanan olası değerler, değişkenin sahip olduğu gerçek değerlere yaklaşık değerler olmalıdır. Atanan değerler, gerçek hayatta bir karşılığı olan makul ve mantıklı değerler olmalıdır. Bu yüzden atama yapmadan önce çoklu atama parametrelerine ve atama modellerine dikkate ederek benzetim çalışmalarının yapılması gerekmektedir (S Sinharay, Stern, & Russell, 2001).



Şekil 3: Çoklu Atama Modeli Çalışma Disiplini

Çoklu atama yöntemiyle kayıp gözlem ataması yapmak üç adımdan oluşmaktadır: (1) *m* tane tamamlanmış veri seti elde etmek için *m* kez olası kayıp değer ataması (missing value imputation) yapmak; (2) istatistiksel yaklaşımlar kullanarak atama modellerini belirlemek; ve (3) en iyi atama verisini seçmek. Literatürde atama değerinin kaç olması gerektiğiyle ilgili birçok çalışma bulunmaktadır. Schafer (1998) çalışmasında, *m* değerinin 3 ile 5 arasında almasının yeterli olduğunu savunmuştur (Joseph L. Schafer & Olsen, 1998). Rubin (1987) çalışmasında ise *m* değerinin 2 ile 5 arasında olmasını önermiştir (Rubin, 1987). White ve diğerleri (2010) çalışmalarında, Monte Carlo hatası olarak tanımladıkları bir standart sapma yaklaşımı kullanarak yeterli sayıda olası kayıp değer üretebilmek için atama değerinin kayıp gözlemlerin yüzdesine eşit veya ondan daha büyük olması gerektiğini belirtmişlerdir. Diğer yandan bazı çalışmalarda ise, atama değerinin belirlenmesinde standart sapmanın yanı sıra, model anlamlılığına (p-değeri) ve istatistiksel güce bakılmasının da önemli olduğu vurgulanmıştır (White, Daniel, & Royston, 2010). Genel olarak, atama sayısı ne kadar küçük olursa modellerin sahip olduğu standart hataların da o kadar küçük olacağı savunulmaktadır. Atama sayının belirlenmesinde benzetimler yapılarak en az model standart hatasını veren atama değerine karar verilebilmektedir. Aynı zamanda, atama hesaplama süreleri (imputation computation times) açısından değerlendirildiğinde, en kısa sürede hesaplama yapabilecek atama değerinin seçilmesi önerilmektedir (Q. Pan, Wei, Shimizu, & Jamoom, 2014; Rezvan, Lee, & Simpson, 2015)

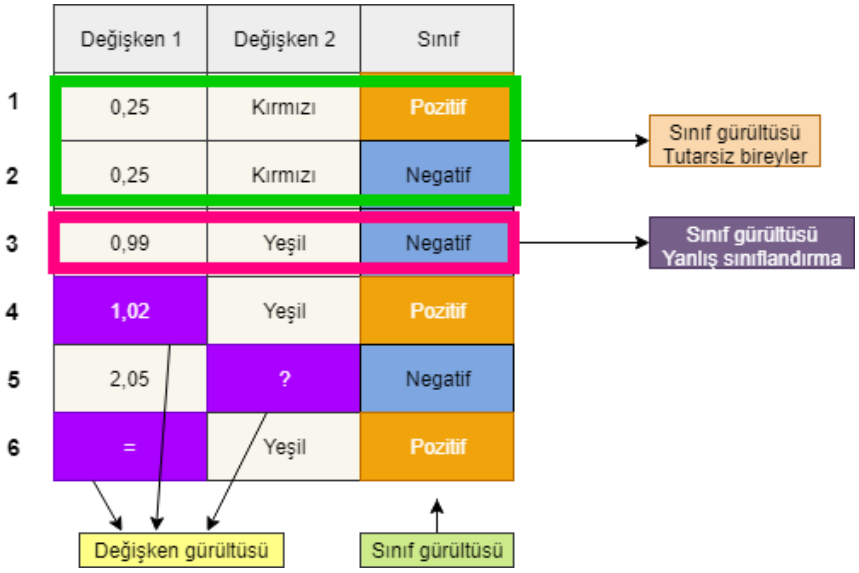
3.2.2. Problem2: Sınıf Gürültüsü

Verilerin toplanması ve hazırlanması sırasında genellikle hatalar ve gürültüler ortaya çıkabilmektedir. Gürültülü veri, gürültü denilen büyük miktarda anlamsız bilgi içeren veriler olarak tanımlanmaktadır. Gürültü terimi, genellikle bozuk veriler (corrupt data) olarak da ifade edilmektedir. Bu veriler, makine öğrenme yöntemleri tarafından doğru anlaşılamayan ve yorumlanamayan verilerdir. Sağlık verilerinde sıklıkla karşılaşılan gürültü sorunu, kurulan sınıflandırma modeline ölçüm hatası olarak katılmaktadır. Bir sınıflandırma probleminde, olası karşılaşılabilecek gürültünün özelliklerini analiz ederek oluşturulacak gürültü uzayında çeşitli etkiler gözlemlenebilmektedir. Gürültü, başka bir sınıfa karşılık gelen örneklerin kendi içinde yeniden küçük bir sınıf kümesi oluşturabilir, somut bir sınıf içindeki kilit alanlarda bulunan örneklerin yerini alabilir ya da kaldırabilir, hatta sınırların üst üste binmesine neden olan sınıflara ait sınırlarını bozabilir. Tüm bu hatalardan dolayı sınıf gürültüsü problemi varlığında verilerin kalitesi, modellerin yorumlanabilirliği, sınıflandırma doğruluğunun belirlenmesi, sonuçların yorumlanabilirliği, makine öğrenme yöntemlerinin performansları ve modellere ait kararların alınması etkilenmektedir (Wang, Storey, & Firth, 1995; Xingquan Zhu & Wu, 2004).

Gürültü genellikle, veri toplama ve veri hazırlama sırasında meydana gelmekte ve temelde iki ana kaynağa bağlı olmaktadır. Bunlar, ölçüm araçlarından kaynaklanan örtük hatalar (implicit errors) ve uzmanlar tarafından veri girişi sırasında kaynaklanan rastgele hatalar (random errors) olarak tanımlanmaktadır (Xingquan Zhu & Wu, 2004). Bu gibi durumlarda kullanılan makine öğrenme yöntemlerinin yüksek sınıflandırma performansına sahip olması beklenirken, eğitim veri setlerinin kalitesinin düşmesi ve aynı zamanda sınıflandırma algoritmasının gürültüye karşı dayanıklı olmaması, yöntemlerin performanslarını düşürmektedir (X. Wu & Zhu, 2008). Gürültünün etkilerini azaltmak için öncelikle etkilenebilecek verilerin bileşenlerinin tanımlanması ve hesaplanması gerekmektedir. Bir veri setini içeren çok sayıda bileşen mevcuttur. Bunlar, sınıf etiketleri (class labels) ve değişken değerleri (attribute values) olmak üzere sınıflandırma verilerindeki iki temel unsurdur (Xingquan Zhu, Wu, & Chen, 2006). Bu temel unsurlara bağlı olarak, literatürde yaygın olarak bahsedilen iki tür gürültü tanımlanmıştır (Xingquan Zhu & Wu, 2004; Xingquan Zhu et al., 2006). Bunlar birisi sınıf gürültüsü (class noise) olarak da bilinen etiket gürültüsü (label noise), diğeri ise öznitelik gürültüsüdür (attribute noise). Bir sınıflandırma veri setinin kalitesini sınıf etiketleri ve değişken değerleri bileşenleri doğrudan etkilemektedir. Sınıf etiketlerinin kalitesi, her bir

bireyin sınıfının doğru bir şekilde atanıp atanmadığını ifade etmektedir. Değişken değerlerinin kalitesi ise, hastalık tanısının sınıflandırma için bireylerin doğru bir şekilde tanımlanabilme becerisini ifade etmektedir. Gürültülü verilerin varlığı, hastalık tanısının sınıflandırması problemlerinde tahmin hataları yapılmasına neden olmaktadır. Tanı sınıflarının sınırları gürültülü veriler nedeniyle güvenilirliğini kaybetmekte ve karar kurallarının geliştirilmesi zorlaşmaktadır (Xingquan Zhu & Wu, 2004). Bu tez kapsamında, gürültülü veri tiplerinden sadece sınıf gürültüsü ele alınacaktır.

- **Sınıf gürültüsü/etiket gürültüsü (class noise):** Bu gürültü, gözlemin yanlış etiketlenmesi durumda ortaya çıkmaktadır. Sınıf gürültüsü, etiketleme işlemi sırasında öznel davranma, veri girişi hataları veya her bir gözlemi etiketlemek için kullanılan bilgilerin yetersizliği gibi çeşitli nedenlere bağlı olarak meydana gelmektedir. Sınıf gürültüsü, çelişkili gözlemleri yani farklı sınıf etiketlerine sahip aynı değişken değerlerine sahip gözlemleri ve yanlış sınıflandırmaları (yanlış olarak etiketlenmiş gözlemler) içermektedir (Hernández & Stolfo, 1998; Xingquan Zhu & Wu, 2004). Çelişkili gözlemleri ortaya çıkarmak, yanlış sınıflandırmaları tanımlamaktan daha zor olduğu için literatürde birçok çalışma yanlış sınıflandırma gürültüsüne odaklanmıştır (Lin et al., 2018; Sáez, Galar, Luengo, & Herrera, 2013). Sınıf gürültüsü temelde ikiye ayrılmaktadır:
  - **Tutarsız bireyler (contradictory examples):** Benzer bireylerin farklı sınıf etiketlerine sahip olması durumudur (Hernández & Stolfo, 1998). Örneğin Şekil 4'te bir veri setinde oluşan gürültü tiplerine ilişkin bir örnek yer almaktadır. Burada, 1. ve 2. bireye (birinci birey için değişken 1 değeri "0,25", değişken 2 değeri "kırmızı", sınıfı "pozitif"; ikinci birey için değişken 1 değeri "0,25", değişken 2 değeri "kırmızı", sınıfı "negatif") ait değişken değerleri aynı olmasına rağmen bu bireyler farklı sınıflarda yer almaktadır.



Şekil 4: Gürültü Tiplerine İlişkin Bir Örnek

– **Yanlış sınıflandırmalar (misclassifications):** Gerçekte ait olduğu sınıftan, farklı bir sınıfa etiketlenmiş bireyler ortaya çıkmasına sebep olan durumdur (Xingquan Zhu, Wu, & Chen, 2003): Örneğin, Şekil 4'teki veri setinde yer alan 3. bireyin (değişken 1 değeri "0,99", değişken 2 değeri "yeşil", sınıfı "negatif") aldığı değişken değerlerine göre "pozitif" sınıfa ait olması gerekirken, yanlış sınıflandırılarak "negatif" sınıfta yer aldığı görülmüştür.

- **Değişken gürültüsü (variable noise):** Bu gürültü, bağımsız değişken değerlerindeki bozulmalara karşılık gelmektedir. Hatalı değişken değerleri, eksik değerler ve tamamlanmamış değişkenler (incomplete variables) veya "umursamama" olarak ifade edilen değerleri içermektedir (Frenay & Verleysen, 2014a). Eksik değerler literatürde genellikle gürültülü veri konusundan bağımsız olarak değerlendirilmekte, bu nedenle değişken gürültüsü terimi temel olarak hatalı değerler için kullanılmaktadır (Xingquan Zhu & Wu, 2004). Değişken gürültüsü tipine örnek olarak Şekil 4'teki veri setindeki durumlar aşağıda göz önünde bulundurulmuştur;

  – **Hatalı değişken değerleri (*Erroneous variable values*):** Örneğin, 4. birey (değişken 1 değeri "1,02", değişken 2 değeri "yeşil", sınıfı "pozitif"), yanlış bir değişken değeri aldığı için (hatalı veri girişi) ilk değişkende gürültülü veriye sahiptir.

- **Eksik veya bilinmeyen özellik değerleri (*Missing or unknown variable values*):** Örneğin, 5. birey (değişken 1 değeri "2,05", değişken 2 değeri "?", sınıfı "negatif"), ikinci değişken değeri bilmediğinden bilinmeyen özellikte değişken gürültüsüne sahiptir.
- **Tamamlanmamış özellik değerleri veya "umursamama" değerleri (*Incomplete variables or "do not care" values*):** Örneğin, 6. birey (değişken 1 değeri "=", değişken 2 değeri "yeşil", sınıfı "pozitif") için ilk değişken değeri diğer değişkenler üzerinde herhangi bir etkisi olmadığından bireyin sınıfını etkilememektedir.

Literatürde sınıf gürültüsü ile ilgili iki ana yaklaşım bulunmaktadır (Frenay & Verleysen, 2014a). Bunlardan ilki olan, algoritma seviyesi yaklaşımları ile gürültülü verilerin varlığından çok daha az etkilenebilecek dayanıklı sınıflandırma algoritmalarının elde edilmesi amaçlanmaktadır. Bu durum, mevcut algoritmaların, etiketleme gürültüsüyle baş edebilmek için sınıflandırma yapısında modellenmektedir (J. Ross Quinlan, 1993). Bu modelleme ile karşılaşılacak olası bir aşırı uyumdan (overfitting) kaçınmak için budama stratejileri uygulanmakta veya temiz olan veriye göre gürültülü örneklerin önemi azaltılmaktadır. Bahsedilen bu iki yaklaşımı birleştiren yeni öneriler de bulunmaktadır. Bu önerilerin temel amacı, sınıflandırma sürecinde gürültünün modellenmesini ve potansiyel gürültülü verilerin göz ardı edilmesini sağlamaktır. Diğer bir yaklaşım olan, filtreler olarak da adlandırılan veri seviyesi yaklaşımları (data level approaches) ise, veri setinin sınıflandırıcı topluluklarını oluşturarak, gürültülü örnekleri yinelemeli olarak filtrelemektedir. Sonrasında, temizleme stratejileri geliştirerek sınıflandırıcı kümeleri oluşturmaktadır. Veri seviyesi yaklaşımları, tüm bu süreçlerin birkaçını birleştiren hibrid/melez yaklaşımlardır (Bouveyron & Girard, 2009).

3.2.2.1. Sınıf Gürültüsünün Yeniden Etiketlenmesi Yöntemleri

Sınıf gürültüsünün çözümünde birçok farklı yaklaşım sunulmaktadır. Literatürde, gürültünün etkilerinin azaltılması için iki temel yaklaşım önerilmiştir (Xingquan Zhu & Wu, 2004). Bu yaklaşımlardan ilki gürültüyü düzgün bir şekilde işlemek için algoritmaların uyarılmasıdır. Bu amaçla kullanılan yöntemler, dayanıklı sınıflandırıcılar (robust classifiers) olarak bilinmekte ve gürültülü verilerden daha az etkilenecek şekilde tanımlanmaktadırlar (Salzberg, 1994). İkinci yaklaşım olarak ise gürültülü veriye sahip gözlemleri silmeyi veya yeniden etiketlemeyi sağlayan veri setlerinin ön işlenmesi önerilmektedir (Brodley & Friedl, 1999). Gürültünün etkisinin azaltılması için önerilen bu iki yaklaşım da iyi performans göstermelerine rağmen

bazı dezavantajlara sahiptir. İlk yaklaşım olan algoritmaların uyarılması yaklaşımı, sınıflandırma algoritmasına dayandığı için aynı sonuçların elde edilmesi modellerde aşırı uyum oluşturması nedeniyle diğer makine öğrenme algoritmalarına genellenememektedir. İkinci yaklaşım olan veri ön işleme yaklaşımında ise, yine ilk yaklaşımda bahsedilen durumlarla karşılaşılırken, aynı zamanda belirli bir gürültü türünü algılaması için tasarlanmış yapılar olduğundan elde edilen veriler yine de mükemmel bir veri setine ulaşmak için yeterli olmamaktadır (X. Wu & Zhu, 2008). Bu nedenlerden dolayı, gürültünün neden olduğu etkilerin azaltılmasına yol açabilecek diğer mekanizmaların araştırılması önemlidir, bu da her bir farklı makine öğrenme algoritmasına uyum sağlamaya gerek duymadan veya verilerde mevcut gürültü seviyesi ve tipi hakkında varsayımlar yapmak zorunda kalmadan gerçekleştirilebilir.

Bu tez çalışmasında sınıf gürültüsünün çözümde, gözlemlere ait sınıfların silinmeden sınıflarının yeniden etiketlenmesini sağlayacak bir yöntem tercih edimiştir. Melez yeniden etiketleme filterisi (Hybrid Repair Filter - HRF) adı verilen bu yaklaşım, gözlem sınıflarının yeniden sınıflandırılmasına dayanan topluluk tabanlı bir filtre olarak tanımlanmaktadır (Morales, Luengo, Garcia, Lorena, Carvalho, et al., 2017). *HRF* fonksiyonu, sınıflandırma ön işleme sırasında veri setindeki sınıf gürültüsünü ortadan kaldırmak (removing) ya da sınıfı yeniden etiketlemek yani sınıflandırmak (repairing) için topluluk temelli bir filtredir. HRF algoritması destek vektör makinesi, yapay sinir ağı, regresyon ve sınıflandırma ağaçları (Classification and Regression Trees - CART), k-en yakın komşu ($k = 1,\ 3,\ 5$) algoritmalarının (k-nearest neighbors algorithm) bir topluluğudur. *HRF* algoritması, bu algoritmaların, tahminlerine ve oylama (voting) şekillerine göre, bir örneklemin alt örneklemini oluşturarak bunları gürültü olarak etiketlemektedir. *HRF* algoritması fonksiyonu için oylama ve yöntem olmak üzere iki önemli parametre vardır. Oylama parametresi, sınıf gürültülerin hangi yöntemle seçileceğini; yöntem parametresi ise, belirlenen sınıf gürültüsü ile filtrenin ne yapacağına karar vermesini ifade etmektedir. Oylama yöntemi, çoğunluk (majority) ve oy birliği (consensus) olmak üzere iki farklı şekilde olmaktadır. Algoritma genellikle çoğunluk yaklaşımını kullanılmaktadır. HRF algoritması yöntem parametresine, silme (remove), yeniden etiketleme (repair) ve melezleme (hybrid) olmak üzere üç farklı şekilde karar verebilmektedir. Eğer gürültülü verilerin silinmesine karar verilirse, sınıf gürülütüsüne sahip gözlemler veri setinden ortadan kaldırılmaktadır. Sınıf gürültüsünün yeniden etiketlenmesine karar verildiğinde, sınıf gürültüsüne ait gözlem sınıfları, topluluk içinde en çok oy alan sınıflar ile değiştirilerek onarılmaktadır. Melezlemeye karar verildiğinde

ise, k-en yakın komşular algoritmasının gürültülü sınıfa verdiği oy, ortadan kaldırmaya mı yoksa yeniden etiketlemeye mi yönelik olacak konusunda karar verilmektedir. Tüm bu süreç, sınıflar gürültüden arındırılıncaya kadar devam etmektedir. Son olarak, arındırılan veri setiyle eğitilen topluluğun doğruluğu, orijinal veri setinin doğruluğunu geçene kadar tekrarlanmakta ve sınıf gürültüsünden arındırılmış veri setleri elde edilmektedir.

3.2.3. Problem3: Sınıf Dengesizliği

Sınıf dengesizliği probleminde, eğer veri setindeki örneklemin büyük bir bölümü bir sınıfta yer alıyorsa bu sınıfa çoğunluk sınıfı (majority class), daha küçük bir bölümü diğer sınıfta yer alıyorsa buna da azınlık sınıfı (minority class) denmektedir. Dengesizlik seviyesi (level of imbalance), çoğunluk sınıfı örneklem sayısının azınlık sınıfı örneklem sayısına bölünmesiyle elde edilmektedir (Leevy, Khoshgoftaar, Bauder, & Seliya, 2018). Sınıf dengesizliği (class imbalanced / unbalanced) durumunda, algoritmaların doğruluklarının yüksek olması çoğu zaman anlamsızdır. Bunun nedeni ise, çoğunluk sınıfının azınlık sınıfı üzerinde baskısının olmasıdır. Bu baskı her zaman sınıf dengesizliğine yol açmayabilmektedir. Eğer çoğunluk sınıfı azınlık sınıfından daha önemliyse, çoğunluk sınıfının öğrenme süreci üzerindeki baskısı sorun olmayabilmektedir. Fakat, azınlık sınıfı görmezden gelinmeyecek kadar önemliyse çoğunluk sınıfının üstünlüğü doğruluk açısından yanlılığa neden olabilmektedir. Bu durumda, sınıf dengesizliği öğrenmesi gereklidir. Başka bir deyişle, sınıf dengesizliği öğrenmesinde, azınlık sınıfının çoğunluk sınıfından daha yüksek maliyete sahip olduğu varsayılmaktadır. Ayrıca, son teknoloji (state-of-the-art) olarak adlandırılan sınıflandırma algoritmaları, verilerin çarpık dağılıma sahip olması durumunda azınlık sınıfını görmezden gelerek çoğunluk sınıfına yönelik bir öğrenme yanlılığı geliştirdiğinden, dengesiz veri setlerinden öğrenmek zor bir durum haline gelmektedir. Birçok sınıflandırma algoritması, sınıflara ait örneklerin sayısı arasındaki büyük farklarla baş edebilecek kadar dayanıklı bir yapıya sahip değildir (Pozzolo, Caelen, Maintainer, & Dal Pozzolo, 2013). Özellikle tıbbi tanı gibi birçok uygulama alanında veri setinin dengesiz bir yapıya sahip olması büyük risk taşımaktadır. Bu durum, dengesiz verilerle başa çıkmak için çeşitli yöntemlerin gelişmesinin yolu açılmıştır.

Sınıf dengesizliği problemi, yetersiz temsil edilen veri ve sınıf dağılımı çarpıklıklarının varlığında öğrenme algoritmalarının performanslarını azaltabilmektedir. Bunun yanı sıra, belirli bir sınıfın diğerlerine göre daha az temsil edilmesine ve çoğunluk sınıflarına yönelik bir öğrenme yanlılığına yol açmaktadır. Bu durum genellikle azınlık sınıfını görmezden gelme eğiliminin ortaya çıkması sonucu oluşan yanlış sınıflandırma sorunu olarak da

tanımlanmaktadır. Böylece, çoğunluğa yönelik bir öğrenme yanlılığı meydana gelmektedir. Yanlılık genellikle, azınlık sınıfını görmezden gelme eğiliminden kaynaklanan yanlış sınıflandırma problemi olarak tanımlandığından, bu problem tanısal hataya yol açmaktadır (Kübler et al., 2018; Y. Zhang, Liu, Cai, & Zhang, 2016). Dengesiz sınıf probleminin varlığında tanı koymada kullanılan algoritmalar bu durumdan etkilenmekte ve yanlış tanı konulmasına sebep olabilmektedir. Sınıflandırma algoritmalarının sağlıkta sıkça rastlanan bu tür bir probleme sahip verilerle maksimum verimle çalışabilmesi için, büyük miktarda ham veriyi ve bilgiyi temsil etmede çeşitli algoritmaların ve/veya yaklaşımların kullanılması gerekmektedir. Böylelikle, hasta güvenliği/kalite geliştirme, karar verme ve problem çözme gibi çeşitli alanlarda çalışan araştırmacılar, tıbbi tanıyı daha güvenilir hale getirmek için bu algoritmaları ve/veya yaklaşımları güvenle kullanabilmektedir.

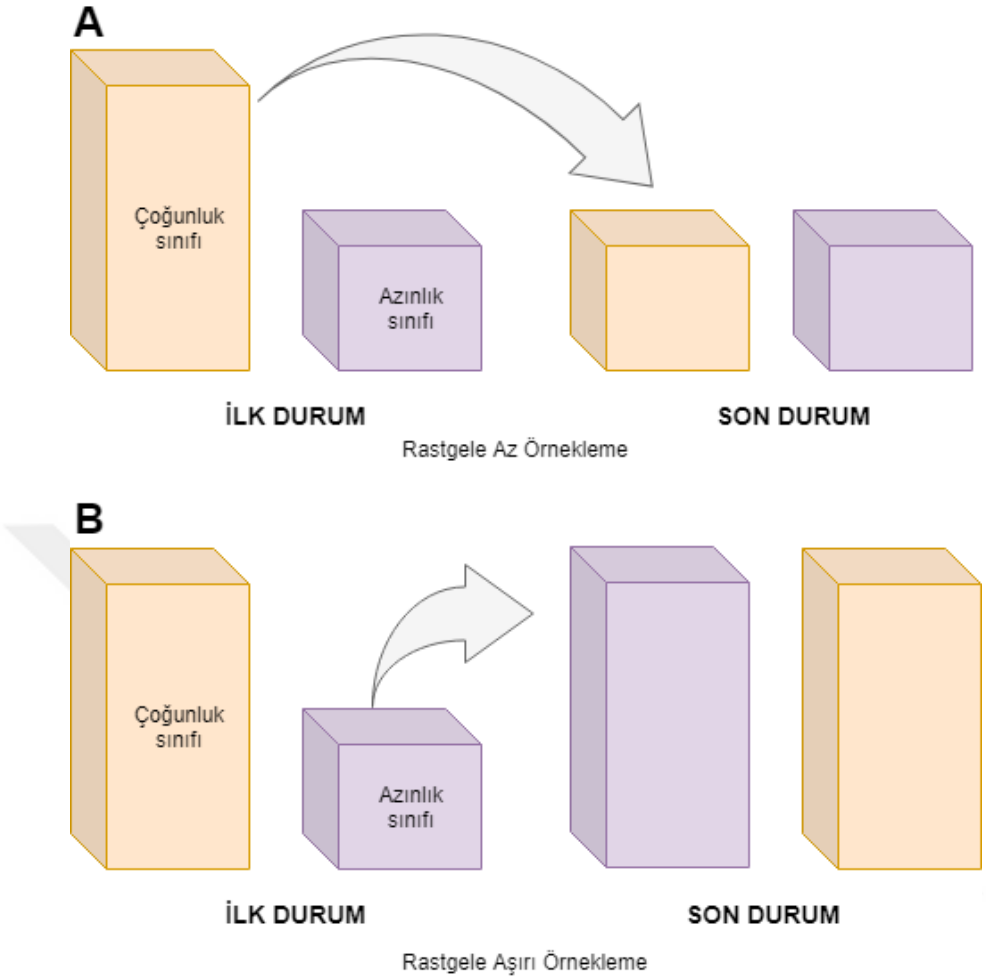
Literatürde tıbbi tanı ile ilgili yapılan çalışmalarda karşılaşılan problemlerin, sınıflandırma performansını düşürdüğünü savunulmuştur. Buna yönelik olarak doğru tanının konulması için dengesiz sınıf problemiyle baş edilmesine yönelik geliştirilen algoritmalar ile mevcut makine öğrenme algoritmalarının entegre bir şekilde kullanılması gerekmektedir. Wu ve diğerleri (2018), endüstriyel sistemlerde büyük çaplı endüstriyel verilerin toplanması nedeniyle bu sistemlerde veriye dayalı hatalı tanılar ve prognostikler bir çalışma alanı oluşturduğunu vurgulamışlardır (Z. Wu, Lin, & Ji, 2018). Bu sistemlerden elde edilen dengesiz veri problemine çözüm sağlaması için topluluk öğrenmenin *SMOTE* yöntemine entegre edilmesiyle oluşturulan (integrated ensemble-based method with SMOTE - Easy-SMT) dengesiz öğrenme algoritmasını önermişlerdir. Önerilen Easy-SMT algoritması, dengesiz azınlık sınıfları genişletmek için *SMOTE* yöntemine dayanan aşırı örnekleme yöntemi ve dengesiz bir sınıf öğrenme problemini topluluk temelli dengeli bir öğrenme alt problemine aktarmak için kullanılan EasyEnsemble yöntemini içeren entegre bir topluluk temelli yöntem olarak tanımlanmıştır. Literatürdeki bu tür çalışmalarla daha doğru bir şekilde hastalık tanısı konulması sağlanmaya çalışılmaktadır. Özellikle kronik hastalıkların sınıflandırılması medikal tanı koymada büyük öneme sahiptir. Kronik hastalıkların en başında gelen hastalıklardan biri olan diyabet hastalığının görülme sıklığı, tahmin edilenden daha hızlı bir şekilde artmaktadır (Fatima & Pasha, 2017). Folorunso ve Adeyemo (2013), sınıf dengesizliği probleminin etkisini incelemek ve diyabet tanısına ait dengesiz veri setindeki sınıflandırma performansını iyileştirmek için literatürde kullanılan Rastgele Az Örnekleme Yöntemi (Random Undersampling - RUS) ve Komşuluk Temizleme Kuralı (Neighborhood Cleaning Rule - NCL) yöntemlerini kullanmışlardır. Orijinal veri seti, karar ağacı algoritması kullanılarak

sınıflandırılmıştır. Kappa istatistiği, Matthew's korelasyon katsayısı, doğruluk ve hata kareler ortalaması karekökü kriterleri ile RUS ve NCL yöntemine göre dengelenen veri setlerinin performansları karşılaştırılmıştır. En iyi performansa sahip yöntem, NCL alt örnekleme yöntemi olmuştur (Folorunso & Adeyemo, 2013). Alghamdi ve diğerleri (2017), dengesiz sınıf problemi ile baş etmede *SMOTE* yaklaşımını kullanarak dengeli sınıflar elde ettikten sonra topluluk öğrenme yöntemlerinden karar ağacı, naive Bayes, lojistik regresyon, lojistik model ağacı ve rastgele orman yöntemlerini kullanarak diyabet tanısını sınıflandırmışlardır. Kappa istatistiği, duyarlılık, seçicilik, hassaslık, doğruluk ve F-ölçütü ile yöntemlerin performansları karşılaştırılmıştır. Rastgele orman ve naive Bayes algoritmaları en iyi performansa sahip algoritmalar olarak bulunmuştur (Alghamdi et al., 2017). Khalilia vd. (2011), aşırı derecede dengesiz verilerde hastalık risklerini tahmin etmeyi amaçlamışlardır. Arasında diyabet tanısı veri setinin bulunduğu toplam sekiz kronik hastalığın tanısının sınıflandırılmasında destek vektör makinesi, torbalama, artırma ve rastgele orman algoritmalarının performansları karşılaştırılmıştır. Bu veri setleri aşırı derece dengesiz sınıflara sahip oldukları için tekrarlı rastgele alt örnekleme (repeated random sub-sampling) yöntemi entegre edilerek probleme çözüm getirilmiştir. Alıcı işletim karakteristiği eğrisi - ROC eğrisi (receiver operating characteristic curve - ROC curve) kriteri ile yapılan performans kıyaslanmasında en iyi performansa sahip algoritma, rastgele orman algoritması olmuştur (Khalilia, Chakraborty, & Popescu, 2011). Tüm bu çalışmalara bakıldığında, doğrudan orijinal veri setine mevcut makine öğrenme algoritmalarını uygulamak yerine öncelikle dengesiz sınıf problemiyle baş edilmiş, sonrasında mevcut makine yöntemlerinin sınıflandırma performanslarının karşılaştırılmasına yönelik bir çalışma yapılması amaçlanmıştır. Literatürde yaygın olarak kullanılan yeniden örnekleme yöntemleri (resampling techniques) ile dengesiz sınıf problemiyle baş edilmeye çalışılmıştır. Yeniden örnekleme yöntemleri, örneklemdeki verileri yeniden kullanan istatistiksel prosedürlerdir. Sınıf dengesizliği problemini gidermek için literatürde birçok yöntem bulunmakla birlikte, yeniden örnekleme yöntemleri bu problemi gidermek için en yaygın olarak kullanılan yöntemler olarak belirtilmiştir (Hu, Liang, Ma, & He, 2009). Sınıf dengesizliği probleminin çözümünde kullanılan yöntemler temel olarak, yapay veri üretme, azınlık sınıfının sayıca çoğunluk sınıfına yakınlaştırma veya çoğunluk sınıfın sayıca azınlık sınıfa yakınlaştırma olmak üzere üç farklı yeniden örnekleme yaklaşımına dayanmaktadır.

3.2.3.1. SMOTE

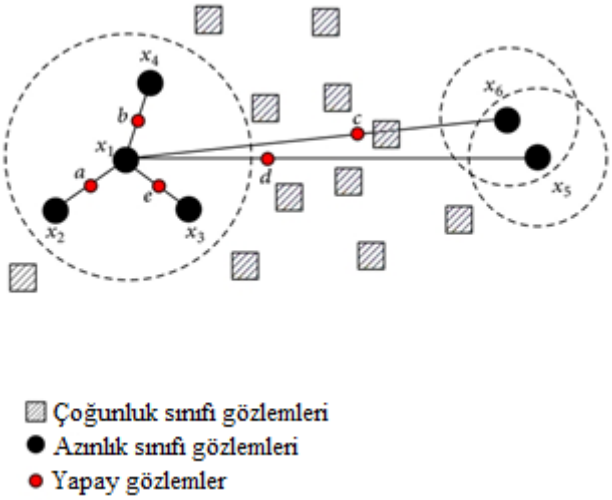
Sentetik azınlık aşırı örnekleme yöntemi bir yeniden örnekleme yöntemidir (Noorhalim, Ali, & Shamsuddin, 2019). *SMOTE* algoritması sayesinde, sınıflandırmada gözlemlenen sınıfların frekanslarındaki dengesizlik ile baş edilerek sınıf dengesizliği probleminin modeller üzerindeki etkisi azaltılmaya çalışılmaktadır (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). *SMOTE* yaklaşımına değinmeden önce, kısaca az örnekleme (undersampling) ve aşırı örnekleme (oversampling) yaklaşımlarından bahsetmek gerekir.

- **Az Örnekleme**: Bu yeniden örnekleme yaklaşımı, veri setindeki gözlem sayısının çok olduğu sınıfın (çoğunluk sınıf), gözlem sayısının az olduğu sınıfa (azınlık sınıf) sayıca rastgele olarak yaklaştırılması temel mantığına dayanmaktadır (Şekil 5a). Veri setinde yapılan bu işlem, eğitim süresini önemli ölçüde düşürmekte ve bellek açısından önemli bir tasarruf sağlayarak sınıflandırma algoritmalarının hesaplama sürelerini azaltmaktadır (A. Y. Liu, 2004).
- **Aşırı Örnekleme:** Bu yeniden örnekleme yaklaşımı, veri setindeki azınlık sınıfın, çoğunluk sınıfa sayıca rastgele olarak yaklaştırılması temel mantığına dayanmaktadır (Şekil 5b). Aşırı örnekleme yöntemi, veri setinin genişliğini büyük oranda arttırdığı için eğitim süresinde artış gözlenmekte ve buna bağlı olarak, kullanılan bellekte büyük yer kaplamaktır. Diğer yandan, sıınıflandırma algoritmalarının hesaplama süreleri artmaktadır (A. Y. Liu, 2004).



Şekil 5: Rastgele Az Örnekleme ve Rastgele Aşırı Örnekleme

Az örnekleme ve aşırı örneklemenin yukarıda bahsedilenler dışında farklı avantaj ve dezavantajları da bulunmaktadır. Bunlardan en önemlisi, az örnekleme veri setindeki örneklem sayısını azaltarak bilgi kaybına yol açarken; aşırı örnekleme, örneklem sayısını arttırarak aşırı uyum problemine sebep olabilmektedir (Blagus & Lusa, 2013). Fakat, azınlık sınıfları örneklem sayısını genişletmek için aşırı örnekleme yöntemini temel alarak geliştirilmiş olan *SMOTE* yeniden örnekleme yaklaşımı kullanılarak da sınıf dengesizliği problemi çözülebilmektedir (Vaughan, 2017). *SMOTE* yaklaşımı Şekil 6'da gösterildiği gibi, azınlık sınıfı örneklemi ile aynı sınıfta bulunan en yakın komşular arasında rastgele interpolasyonla yapay gözlemler üretmektedir.



Şekil 6: SMOTE Algoritması Çalışma Prensibi (Minh, 2018)

SMOTE yaklaşımının çalışma şeklinden kısaca bahsetmek gerekirse (Şekil 6),

- Azınlık sınıfında yer alan her bir gözlemin *k* yakın komşusu bulunur.
- Azınlık sınıfında yer alan gözlem (x_i) ile *k*-en yakın komşusu olan gözlem (x_j) arasındaki uzaklık farkı alınır.
- Bu fark, 0 ile 1 arasında bir rastgele sayı (α) çarpılır ve yeni yapay gözlem elde edilir.

$$x_{new} = x_i + (x_j - x_i) * \alpha \quad (4)$$

- SMOTE algoritmasıyla ne kadar gözlem üretilecekse, tüm adımlar t kez yinelenir $t = 1, \dots, T$ ve üretilen yapay gözlemler eğitim veri setine eklenir.

SMOTE yaklaşımında, k sayısı farklı değerler alması durumunda genel uzaklık hesaplama formülü aşağıdaki gibi tanımlanmıştır.

Eğer $k = 1$ ise (1- en yakın komşudan yapay veri üretme),

$$x_{new} = x_i + (x_1 - x_i)\alpha \quad (5)$$

$x_i \in S_{min}$: azınlık sınıfı gözlemi

$\hat{x}_1$: x_i'in k-en yakın komşulardan biri, $\hat{x}_1 \in S_{min}$

$\alpha \in [0,1]$: rastgele sayı

Buna göre, $k = 1$ durumunda,

$$x_{new} = x_i + (x_1 - x_i)\alpha = x_i + (x_1\alpha - x_i\alpha) = x_i - x_i\alpha + x_1\alpha = (1-\alpha)x_i + \alpha x_1 \quad (6)$$

şeklinde hesaplanır.

Eğer $k = 2$ ise (2 - en yakın komşulardan yapay veri üretme),

$$x_{new;2} = \alpha x_i + \alpha_1 x_1 + \alpha_2 x_2 \quad (7)$$

Bu durumda, rastgele sayı, $\alpha = 1 - \alpha_1 - \alpha_2$ eşit olur,

$$x_{new;2} = (1 - \alpha_1 - \alpha_2)x_i + \alpha_1 x_1 + \alpha_2 x_2 \quad (8)$$

$$x_{new;2} = x_i - \alpha_1 x_i - \alpha_2 x_i + \alpha_1 x_1 + \alpha_2 x_2 \quad (9)$$

$$x_{new;2} = x_i + (x_1 - x_i)\alpha_1 + (x_2 - x_i)\alpha_2 \quad (10)$$

şeklinde hesaplanır.

Eğer *k*-en yakın komşular varsa,

$$x_{new;k} = x_i + (x_1 - x_i)\alpha_1 + (x_2 - x_i)\alpha_2 + \cdots + (x_k - x_i)\alpha_k \quad (11)$$

$$x_{new;k} = x_i + \sum_{j=1}^{k}(x_k - x_i)\alpha_k \quad (12)$$

şeklinde hesaplanır ve bu durumda, *k*-en yakın komşular için rastgele sayılar,

$$\alpha \in [0,1]$$

$$\alpha_1 \in [0, 1 - \alpha]$$

$$\alpha_2 = [0, 1 - \alpha - \alpha_1]$$

...

$$\alpha_k = [0, 1 - \alpha - \alpha_1 - \alpha_{k-1}] \quad (13)$$

olarak hesaplanır ve $\alpha, \alpha_1, \alpha_2, \ldots, \alpha_k$ rastgele sayılar, $\sum \alpha_i = 1$ eşittir.

SMOTE yaklaşımın kullandığı k-en yakın komşular algoritması, regresyon analizinde öklid uzaklığı (euclidean distance) ile hesaplanırken; sınıflandırmada komşu gözlem ile azınlık gözlemi arasındaki uzaklığın farkı alınarak hesaplanmaktadır. Sonuç olarak *SMOTE*, var olan azınlık sınıfı gözlemlerinin komşuluk ilişkilerine bağlı rastgele herhangi bir noktada yeni gözlemler üreterek çalışmaktadır. Böylelikle, *SMOTE* rastgele aşırı örneklemeye benzer şeklinde, azınlık sınıfındaki gözlem sayısını arttırmaktadır. Ancak, aynı gözlemi yeniden üretmemektedir. Diğer bir ifadeyle, mevcut ya da gerçek azınlık sınıfı gözlemlerini uygun

şekilde birleştirerek yeni bir gözlem oluşturmaktadır. Böylelikle, aşırı örnekleme yöntemindeki aşırı uyum problemini önemli derecede önleyebilmektedir (Chawla et al., 2002).

3.3. Veri Ön İşleme R Paketleri

Gelişen dünya ile birlikte, birçok farklı kaynaktan elde edilen veri büyüklüğü artmaktadır. Bu artışa bağlı olarak, veri madenciliği alanında karşılaşılacak olası zorlukların da çeşitliliği ve sayısı aynı derecede artmaktadır. Bundan dolayı araştırmacıların, sadece verinin hacmini değil, aynı zamanda veriye ait sorunları da ele almaları gerekmektedir. Bu anlamda, veri ön işleme (data preprocessing), Veri Tabanlarından Bilgi Keşfi (Knowledge Discovery from Databases - KDD) olarak adlandırılan sürecin önemli bir parçası haline gelmektedir (Famili, Shen, Weber, & Simoudis, 1997).

Veri ön işleme, tüm KDD işleminde en çok zaman alan adımlardan biri olarak bilinmektedir (Tomar & Agarwal, 2014). Bu adımlar içerisinde değişken seçimi (variable selection), kayıp gözlem değerlerinin alınması, sınıf gürültüsünün tespit edilmesi ve sınıf dengesizliği ile baş edilmesi gibi birçok işlem yer almaktadır. Değişken seçimi, öğrenme süreci için en uygun nitelikleri ortaya çıkarmayı amaçlamakta, böylece modellerin karmaşıklığı için harcanan hesaplama süresinin azaltılmasını sağlamaktadır. Kayıp gözlem değerlerinin giderilmesi, veri setinde mümkün olduğu kadar fazla bilgi tutarak bilgi kaybını önlemektedir. Sınıf gürültüsü varlığında ise, genel olarak temel veri dağılımdan kaynaklı yanlış sınıflandırılmış gözlemlerin tespit edilmesi amaçlanmaktadır. Sınıf dengesizliği sürecinde ise, çoğunluk sınıfına karşı olan öğrenme yanlılığı ortadan kaldırılmaktadır. Veri ön işleme sonrasında uygulanacak öğrenme algoritmaları sadece verilerden anlamlı ve ilişkili bilgiler elde etmekle kalmayıp; aynı zamanda yüksek tahmin (predictive) veya tanımlayıcı (descriptive) performanslarına sahip modellerin kurulmasını da sağlamaktadır (García, Ramírez-Gallego, Luengo, Benítez, & Herrera, 2016).

Tüm bu veri ön işleme süreçleri için geliştirilmiş pek çok yazılım bulunmaktadır. Bu yazılımlar içerisinden en çok bilinenleri; Java yazılımı aracı olan KEEL değişken seçimi, eksik değerler ve gürültülü veriler için kapsamlı bir veri işleme algoritmasına sahip açık kaynaklı (GPLv3) bir Java yazılım aracıdır (Alcalá-Fdez et al., 2011, 2009; Triguero et al., 2017). Diğer yandan, WEKA (Witten & Frank, 2005), KNIME (Berthold et al., 2009), RapidMiner (Hofmann & Klinkenberg, 2013) veya istatistiksel programlama dili R gibi veri ön işleme fonksiyonlarına sahip başka birçok veri madenciliği yazılımı bulunmaktadır. R, veri

ön işleme süreçlerini ele almak için Kapsamlı R Arşiv Ağı (Comprehensive R Archive Network - CRAN) adı verilen R paketleri deposunda birçok paket bulunmaktadır. Eksik değerlerin ele alınması için Zincirleme Denklemlerle Çok Değişkenli Atama (Multivariate Imputation by Chained Equations - MICE) paketi başta olmak üzere (Stef van Buuren & Oudshoorn, 2011) Amelia, Hmisc ve missForest (Honaker, King, & Blackwell, 2011) paketleri en çok tercih edilen paketlerdir. Sınıf gürültüsü probleminin çözümünde kullanılan kapsamlı bir paket olan *NoiseFiltersR* paketi, filtreleme yöntemleri (filtering methods) olan birçok yöntemi içinde barındırarak eğitim setindeki gürültülü gözlemlerin tespit etmesinde ve ortadan kaldırmasında kullanılan bir veri ön işleme mekanizmasıdır (Frenay & Verleysen, 2014b; Morales, Luengo, Garcia, Lorena, De Carvalho, et al., 2017). Bununla birlikte, *NoiseFiltersR* paketi bazen yardımcı fonksiyonlar olarak ihtiyaç duyduğundan sınıf dengesizliği probleminde kullanılan aşırı örnekleme yöntemlerinin yanı sıra Tomek Links (Tomek, 1976) veya ENN (Wilson, 1972) gibi temel klasik filtre sürümlerini içermektedir. Ayrıca, sınıf dengesizliği problemi ile baş edebilmek için sınıflandırmada kullanılacak kapsamlı bir paket olan *unbalanced* paketi geliştirilmiştir (Pozzolo et al., 2013).

Bu tez çalışması kapsamında, kayıp veri problemi için *MICE* paketi içinde yer alan mice algoritması, sınıf gürültüsü problemi için *NoiseFilters* paketinde yer alan *HybridRepairFilter* algoritması ve sınıf dengesizliği problemi için geliştirilmiş olan sınıflandırma ve regresyon eğitimi (Classification And REgression Training - caret) ve R ile veri madenciliği (Data Mining with R - DMwR) paketleri içinde yer alan *SMOTE* algoritması kullanılarak bu problemler çözülecektir.

3.3.1. Kayıp Gözlem Veri Ön İşleme için: MICE Paketi

Çoklu atama, karmaşık eksik veri (complex incomplete data) problemleri için en çok tercih edilen yöntemdir. Birden fazla değişkende meydana gelen eksik verilerle baş etmek özel bir çaba istemektedir. Bunun için *MICE* olarak adlandırılan yaklaşım, çok değişkenli verilerin kayıp gözlemlere sahip olduğu durumda etkili bir şekilde kullanılmaktadır. Tek bir atamaya kıyasla (örneğin; ortalama) çoklu atama, eksik değerlerdeki belirsizliği göz ardı etmeyerek önemsemektedir. *MICE* yaklaşımı, birleşik modelleme (Joint Modeling - JM) ve tamamen koşullu tanımlama (Fully Conditional Specification - FCS) olmak üzere iki farklı yaklaşımı temel almaktadır. Schafer (1997), çok değişkenli normal (multivariate normal), log-linear ve genel konum modeli (general location model) gibi atamalar için çeşitli JM yöntemleri geliştirmiştir (J.L. Schafer, 1997). JM yaklaşımı, eksik veriler için çok değişkenli bir dağılım

belirlemek ve Markov Zinciri Monte Carlo (Markov Chain Monte Carlo - MCMC) yöntemleriyle koşullu dağılımların elde edilmesini sağlamaktadır. Böylelikle, çok değişkenli dağılıma sahip verinin kabul edilebilir bir sonuç elde edilebilmesine imkan vermektedir.

FCS yaklaşımı, değişkenden değişkene çok değişkenli atama modelini, her bir eksik değişken için bir tane olmak üzere koşullu yoğunluk kümesi tarafından belirlemektedir. Bu yaklaşım ile, ilk atamadan başlayarak, FCS koşullu yoğunlukları yinelenerek atamalar elde edilmektedir. Genellikle 10 ile 20 arası yineleme yeterlidir (J.L. Schafer, 1997). FCS yaklaşımı, çok değişkenli en uygun dağılımın bulunamadığı durumlarda JM yaklaşımına alternatif olarak tercih edilebilmektedir.

MICE paketi, JM yaklaşımının çok eski kalmasından dolayı çoklu atama için tamamen koşullu olarak belirlenmiş modeller adı verilen FCS yaklaşımını temel alan modeller olarak uygulanmaya başlanmıştır. MICE 1.0 sürümü ilk olarak 2000 yılında bir S-PLUS kütüphanesi olarak piyasaya sürülmüştür (S van Buuren & Oudshoorn, 2000). Yıllar içerisinde kullanım alanının gelişmesiyle kullanıcı tarafından R paketine dönüştürülmüştür (R Development Core Team, 2011).

Bu tez kapsamında, kayıp gözlem için çoklu atama paketi olan *MICE* paketinin 2.9 sürümü kullanılmıştır (Stef van Buuren & Oudshoorn, 2011). Çoklu atama yaklaşımı, geleneksel istatistiksel yaklaşımlardan temel olarak farklı bir yönteme sahiptir. Kullanıcılar, kayıp gözlemle baş etmede modellere, eksik veri mekanizmalarına ve değişken sayısına bağlı olarak farklı birçok durumunu bir arada değerlendirmek zorunda kalmaktadır. *MICE* yaklaşımı, bu süreçlerin tamamının bir arada değerlendirilesine imkan vererek, kayıp gözlem problemiyle baş etmede pratiklik sağlamaktadır. *MICE* yaklaşımı arka planında çoklu atamanın, her biri tanısal kontrol gerektirebilecek adımların sırasıyla yapabilmesi yatmaktadır. Tanısal kontrol, birden fazla çoklu atama işlemi sırasında atamaların anlamlı olup olmadığını değerlendirmektir. Atamalar, verilere yakın değerler olmalıdır. Bunun yanı sıra, olası atanmış değerler, mantıklı ve anlamlı değerler olmalıdır aksi takdirde uyuşmayan veriler (örneğin; negatif sayımlar, hamile babalar) atanmış veride bulunmamalıdır. Atamalarda değişkenler arasındaki ilişki göz önünde bulundurulmalı ve "doğru" atama değerleri hakkındaki olası belirsizlik miktarını yansıtmalıdır. Böylelikle, atanmış yani tahmin verilerindeki tanı kontrolleri, atamaların uygunluğunun veya anlamlılığının kontrol edilmesini sağlamaktadır (Stef van Buuren & Oudshoorn, 2011).

MICE yaklaşımının notasyonunda,

Y_j, p eksik değişkenlerden birini temsil etmektedir $j = 1, \dots, p$.

Bu durumda, $Y = (Y_1, \dots, Y_p)$ olur.

Y_j'nin gözlenen (observed) ve eksik (missing) kısımları sırasıyla Y_j^{obs} ve Y_j^{mis} olarak tanımlanır.

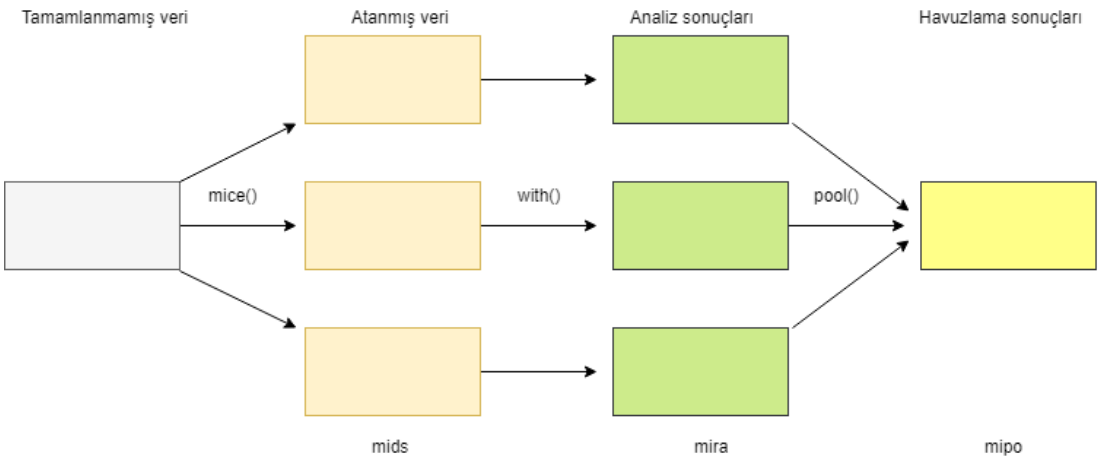
Bu durumda, $Y^{obs} = (Y_1^{obs}, \dots, Y_p^{obs})$ ve $Y^{mis} = (Y_1^{mis}, \dots, Y_p^{mis})$, Y'de gözlenen ve eksik verileri ifade etmektdedir.

Atama sayısı, $m \geq 1$'e eşittir.

Atama veri setleri, $Y^{(h)}$ olarak ifade edilmektedir $h = 1, \dots, m$.

$Y_{-j} = (Y_1, \dots, Y_{j-1}, Y_{j+1}, \dots, Y_p)$, Y_j dışındaki $p-1$ tane değişkenin tamamı Y olarak gösterilmektedir.

Q, bir katsayıyı (örneğin; bir regresyon katsayısı) ifade etmektedir. Uygulamada, $\boldsymbol{Q}$ genellikle çok değişkenli bir vektörü (multivariate vector) ifade etmektedir. Daha genel ifade etmek gerekirse, Q, katsayının herhangi bir modelini kapsamaktadır (Rubin, 1987).



Şekil 7: Çoklu Atama Adımları (Stef van Buuren & Oudshoorn, 2011)

Çoklu atama yöntemine Şekil 7'deki gibi modüler bir yaklaşımla bakıldığında, çoklu atamadaki üç ana adımı göstermektedir: atama (imputation - mids), analiz (analysis - mira) ve havuzlama (pooling - mipo). Bunun için geliştirilen *MICE* paketi her adımın sonuçlarını mids,

mira ve mipo olmak üzere üç farklı sınıfta saklamaktadır. Şekil 7'nin en sol tarafı analizin gözlemlenen, yani eksik bir veri seti temsil eden Y^{obs} ile başlamaktadır. Buradaki sorun, genel olarak, gözlemlenmemiş veriler için gerçekçi olmayan varsayımlar yapmadan, Y^{obs}'dan Q'yu tahmin edilememesidir. Çoklu atama, eksik değerleri olası veri değerleri (plausible values) ile değiştirerek verinin birkaç atanmış modelinin (imputed versions) yani tahmin edilen modelinin genel bir çerçevesini oluşturmaktadır. Bu olası değerler, her eksik veri için özel olarak modellenen bir dağılımından elde edilmektedir. *MICE* paketi bu işlemi, $mice()$ adı verilen fonksiyonla yapmaktadır (Rubin, 1987).

Şekil 7'de, atama sayısı $m = 3$ olarak belirlenmiş ve atanmış veri setleri, Y^1, Y^2, Y^3 ile gösterilmektedir. Atanmış üç veri seti, eksik veri girişleri için aynıdır, ancak tahminlenen değerlere yani atanmış değerlere (imputed values) göre farklılık göstermektedir. Bu farkın büyüklüğü, hangi değerin atanacağı konusunda bir belirsizliği yansıtmaktadır. *MICE* paketi, bu değerlere ait verileri saklamak için çarpımla atanmış veri seti sınıfı (multiply imputed dataset of class - mids) adı verilen özel bir sınıfa sahiptir.

İkinci adımda, eğer veriler tamamlanmışsa, her bir veri setindeki Q değeri, kullanılan yöntemle tahmin edilmektedir. Bu işlem tüm veriler tamamlandığı için kolay uygulanmaktadır. Uygulanan model $Y^1, \dots, Y^m$ için genel olarak aynıdır. *MICE* 2.9 paketinde, bu analiz $with.mids()$ fonksiyonu ile yapılırken; bu fonksiyon, $lm.mids()$ ve $glm.mids()$ fonksiyonlarının yerine kullanılmaktadır. Girdi verilerinin farklı olmasından dolayı, tahminler olan $\hat{Q}^{(1)}, \dots, \hat{Q}^{(m)}$ birbirlerinden farklı olacaktır. Bu farklılıkların, hangi değeri etkileyeceği konusundaki belirsizlikten kaynaklandığını anlamak önemlidir. *MICE* paketinde, analiz sonuçları toplu olarak, çarpımlı atanmış tekrarlanan analiz (multiply imputed repeated analysis - mira) olarak depolanan R nesnesi sınıfı içinde saklanmaktadır (Rubin, 1987).

Son adımda ise, m tahminlerini, $\hat{Q}^{(1)}, \dots, \hat{Q}^{(m)}$, bir $\bar{Q}$ tahminin içine havuzlamakta ve varyanslarını tahmin etmektedir. Yaklaşık olarak normal dağılmış Q miktarları yani katsayıları için ortalama $\hat{Q}^{(1)}, \dots, \hat{Q}^{(m)}$ üzerinden hesaplanırken, Rubin tarafından belirtilen yönteme göre atama içi ve arasındaki varyans (within- and between-imputation variance) toplanmaktadır (Rubin, 1987). $pool()$ fonksiyonu, Rubin tarafından belirlenen kurallara göre miktarları havuzlamak için yöntemler içermektedir. Fonksiyondan elde edilen sonuçlar, çoklu atanan havuzlanmış sonuçları (multiple imputed pooled outcomes - mipo) adı verilen sınıf nesne içinde depolanarak saklanmaktadır.

Atama modeli için, kayıp veriyi meydana getiren sürecin göz önüne alınması yani bu sürecin öneminin belirlenmesi, verideki ilişkilerin korunması ve bu ilişkilerle ilgili belirsizliğin korunması gerekmektedir. Bu duruma bağlı kalınarak Rubin (1987)'in hesaplamasında olduğu gibi Q'nun geniş aralığı için istatistiksel olarak doğru atama değerlerinin elde edilmesi sağlanmaktadır. Çok değişkenli eksik verilere atama yaparken ortaya çıkabilecek birkaç sorun olabilir (Rubin, 1987). Bu sorunlar genel olarak,

- Belirli bir Y_j için, atama modelinde kullanılan Y_{-j} tahmincilerinin kendileri tamamlanmamış olabilir.
- Y_1'in Y_2'ye bağlı olduğu ve Y_2'nin Y_1'e bağlı olduğu durumlarda, genel olarak Y_1ve Y_2, diğer değişkenler de dahil olmak üzere, korelasyon gösterdiği için dairesel bağımlılık (circular dependence) oluşabilir.
- Özellikle, büyük p (değişken sayısı) ve küçük n (örneklem sayısı) yani boyutluluk sorunu olarak da bilinen durumda, doğrusal bağlantı (collinearity) ve boş hücreler oluşabilir.
- Satırlar veya sütunlar, örneğin boylamsal verilerde (longitudinal data) olduğu gibi sıralı olabilir.
- Değişkenler farklı tiplerde olabileceğinden (örneğin; ikili, sırasız, sıralı, sürekli), çok değişkenli normal dağılım varsayımıyla modellerin kurulmasının teorik olarak uygun olmaması durumu söz konusu olabilir.
- Y_j ve Y_{-j} arasındaki ilişki karmaşık olabilir, örneğin doğrusal olmayabilir veya sansürleme işlemlerine tabi tutulabilir.
- Atama, olası olmayan kombinasyonlar yaratabilir (örneğin; hamile babalar) veya verilerdeki deterministik ilişkileri yok edebilir (örneğin; toplam puanlar).
- Atamalar, anlamsız olabilir (örneğin; ölülerin vücut ısısı).
- Atanamış verilere uygulanacak Q modelleri henüz bilinmiyor olabilir.

Bahsedilen bu sorunların tamamı her zaman için geçerli değildir ve belirli veriler için başka karmaşıklıklar görünebilir. Verilerin gerçek hayattaki karmaşıklıklarının ortaya çıkardığı sorunları ele almak için, verilerdeki her sütun için ayrı ayrı atama modeli belirlemek gerekmektedir. Bundan dolayı, zincirleme denklemler (chained equations) yönteminin geliştirilmesinin önü açılmıştır. Bu yöntemde, tüm atamalar değişken seviyesinde gerçekleşir.

Buna bağlı olarak, zincirleme denklemler yönteminde tek değişkenli atama yöntemleri oldukça iyi geliştirilmiştir (Rubin, 1987).

Varsayımsal olarak tamamlanmış verilerin Y, p değişkene sahip çok değişkenli dağılımı $P(Y|\theta)$ kısmen gözlenen rastgele bir örneklemden gelmektedir. Y'nin çok değişkenli dağılımının, bilinmeyen parametreleri, θ vektörü tarafından belirlendiği varsayılmaktadır. Buradaki sorun, θ'nın çok değişkenli dağılımını doğrudan veya dolaylı olarak elde etmektir. *MICE* algoritması, koşullu dağılımlarından yinelemeli olarak örnekleme yaparak θ'nin sonsal dağılımını (posterior distribution) elde etmektedir.

$$P(Y_1|Y_{-1}, \theta_1)$$

.

.

.

$$P\left(Y_p|Y_{-p}, \theta_p\right)$$

Parametreler $\theta_1, \dots, \theta_2$, ilgili koşullu yoğunluklara özgüdür ve mutlaka "doğru" bileşik dağılımının (joint distribution) $P(Y|\theta)$ çarpanlara ayrılmasıyla elde edilen bir sonuç değildir.

$$\theta_1^{*(t)} \sim P\left(\theta_1|Y_1^{obs}, Y_2^{(t-1)}, \dots, Y_p^{(t-1)}\right)$$

$$Y_1^{(t)} = \left(Y_1|Y_1^{obs}, Y_2^{(t-1)}, \dots, Y_p^{(t-1)}, \theta_1^{*(t)}\right)$$

...

$$\theta_p^{*(t)} \sim P\left(\theta_p|Y_p^{obs}, Y_1^{(t)}, \dots, Y_{p-1}^{(t)}\right)$$

$$Y_p^{(t)} = \left(Y_p|Y_p^{obs}, Y_1^{(t)}, \dots, Y_p^{(t)}, \theta_p^{*(t)}\right) \quad (14)$$

Gözlemlenen marjinal dağılımlardan başlayarak, zincirleme denklemlerin t.yinelemesi, $Y_j^{(t)} = \left(Y_j^{obs}, Y_j^{*(t)}\right)$ t. yinelemedeki j. atanmış değişkeni için alternatif koşullu örnekleme olarak adlandırılan bir Gibbs örneklemleyicidir. Önceki atamaların $Y_j^{*(t-1)}$, diğer değişkenlerle olan ilişkisinin $Y_j^{*(t)}$ doğrudan girip girmediği gözlemlenebilir. Bu yüzden yakınsama, diğer birçok MCMC yönteminin aksine oldukça hızlı olabilmektedir. Yakınsaklığı gözlemlemek önemlidir, fakat kullanıcı tecrübelerine bağlı olarak yinelemelerin sayısı m, en az 10 ile 20 arasında olabilir (Rubin, 1987). Zincirleme denklemler, *MICE* algoritmasının,

eksik verilerle baş edebilmek için tek değişkenli yaklaşımlarının bir birleşimi olarak kolayca uygulanabileceği anlamına gelmektedir. $Mice$ () fonksiyonu, her biri bir atanmış veri seti üreten m akışlarını paralel olarak uygulamaktadır. *MICE* paketi ile birlikte uygulanan Kayıp Değerlerin Göreselleştirilmesi ve Atanması (Visualization and Imputation of Missing Values - VIM) paketi fonksiyonları, eksik verileri görselleştirmek için kullanılan bir grafik paketidir (Kowarik & Templ, 2016).

3.3.1.1. Atama Modellerinin Seçimi

Atama modellerine karar vermek için bazı durumlar göz önünde bulundurulmalıdır. Öncelikle hangi kayıp veri mekanizması varsayımının sağlandığına karar verilmelidir. *MICE* algoritması, eksik verilerin rastgele kayıp olduğunu varsaymaktadır. MAR, bir değerin eksik olma ihtimalinin yalnızca gözlenen değere bağlı olduğunu ve bunları kullanarak tahmin edilebileceği anlamına gelmektedir (Bodner, 2008; Bull, Lewinger, & Lee, 2007). MAR varsayımı, birçok uygulamada uygun bir yöntemdir. *MICE* paketi, hem MAR hem de MNAR ile baş edebilmektedir, fakat MNAR için çoklu atama modeline ek bazı modellemeler gerekmektedir. Atama yöntemlerinin genellikle MAR olduğu varsayılmaktadır, fakat çoklu atama yaklaşımı tamamen geneldir ve MNAR için de geçerli olabilmektedir. MNAR varsayımı altında tamamlanmış veriyle kurulan modeller, kayıp gözlem için kurulan modeller için doğru değildir. Bu yüzden atama için bu yaklaşım kullanılmaz (Bernaards, Belin, & Schafer, 2007).

İkinci olarak atama modelinin tipine karar vermek gerekmektedir. Seçilen model hem yapısal hem de varsayılan hata dağılımını kapsamalıdır. *MICE* paketi içinde veri setinde bulunan her değişken için farklı atama modelleri bulunmakta ve bu modeller bağımlı değişkenin tipine bağlı olarak değişmektedir. Her bir değişken için tek değişken atama yöntemleri uygulanmaktadır (Clogg, Rubin, Schenker, Schultz, & Weidman, 1991). Bu yöntem, tahminlenen veri setini ve eksik değer bulunan ilgili değişkene tek bir atama yapmaktadır. *MICE* paketi içinde, Tablo 2'de varsayılan bazı atama modelleri belirtilmiştir.

Tablo 2: MICE Algoritması Atama Modelleri

Yöntem	Tanım	Değişken tipi
pmm	Tahmin ortalama eşleştirme	sürekli
norm	Bayes doğrusal regresyon	sürekli
norm.nob	Bayes olmayan doğrusal regresyon	sürekli
mean	Koşulsuz ortalama atama	sürekli
2L.norm	İki-düzeyli doğrusal model	sürekli
logreg	Lojistik regresyon	kategorik, 2-düzeyli
polyreg	Çok kategorili logit model	kategorik, > 2-düzeyli
polr	Sıralı logit model	sıralı, > 2-düzeyli
lda	Doğrusal diskriminant analizi	kategorik
sample	Gözlenen veriden rastgele örnekleme	hepsi*

*Bağımsız değişken tipinin hem sürekli hem de kategorik olduğu durumda kullanılabilir.

3.3.2. Sınıf Gürültüsü Veri Ön İşleme: NoiseFiltersR Paketi

NoiseFiltersR paketi, veri seviyesi yaklaşımlarını temel almaktadır. Bunun sebebi ise, veri ön işlemesinin sadece bir kez yapılmasına izin verilmesi ve daha sonra herhangi bir sınıflandırma algoritmasının uygulanabilmesidir. Algoritma seviyesi yaklaşımları her sınıflandırma algoritması için özel yaklaşımlardır. Morales ve diğerleri (2017) çalışmasında, gürültülü veri konusu çeşitli yönleriyle göz önünde bulundurularak en popüler klasik ve son teknoloji filtrelere (state-of-the-art filters) genel bir bakış açısıyla ele almışlardır. (Morales, Luengo, Garcia, Lorena, De Carvalho, et al., 2017). Sınıf gürültüsünde gözlemler, topluluk temelli, benzerlik temelli ve veri karmaşıklığı temelli algoritmalar olmak üzere üç tipte tanımlanmaktadır. Topluluk temelli algoritmalar, farklı bölümler veya eğitim verisinin yeniden örnekleri üzerine kurulu sınıflandırma topluluklarının tahminlerinden oluşmaktadır. Benzerlik temelli algoritmalar, her örneğin en yakın komşularının etiket dağılımına dayanmaktadır. Veri karmaşıklığı temelli algoritmalar ise, gürültü verilerin varlığına bağlı olarak karmaşıklığa ait ölçümlerin azaltılması temeline dayanmaktadır (García, Luengo, & Herrera, 2016). Tablo 3'de yer alan *NoiseFiltersR* paketi fonksiyonları her üç durum için farklı çözümler sunmaktadır.

Tanımlanan her üç gürültü temeliyle nasıl başa çıkılacağı konusunda, iki farklı strateji geliştirilmiştir. Bunlar, gürültü sınıfının ortadan kaldırılması (noise removal) ve gürültü sınıfının yeniden etiketlenmesi (noise reparation) stratejileridir. İlk stratejide, gürültülü örnekleri ortadan kaldırırken; ikincisi stratejide ise, bu örnekleri mevcut bilgilere dayanarak en olası etiketle yeniden etiketlemektedir. Eğer yeni etiket yeterince güven veriyorsa gözlem için tekrar etiketleme yapan karma yaklaşımlar (hybrid approaches) da vardır. Aksi takdirde, gürültülü örnek ortadan kaldırılmaktadır. Gürültü sınıfının ortadan kaldırılması ve gürültü sınıfının yeniden etiketlenmesi ve bunun gibi stratejiler birçok çalışmanın konusudur. Çoğu çalışma, yeniden etiketlemenin olası sorunları üzerinde dururken, diğer çalışmalar, hibrid yaklaşımlarının tehlikeleri üzerinde durmuştur. Tablo 3'de gösterildiği gibi bu iki strateji baz alınarak *NoiseFiltersR* paketi içerisinde birçok farklı çözüm sunulmuştur.

NoiseFiltersR paketi, sınıflandırma problemlerinde sınıf gürültüsü veri ön işlemesi için kullanılan bir R paketidir (Morales, Luengo, Garcia, Lorena, De Carvalho, et al., 2017). Bu paket, 24 araştırma makalesinde yayınlanan toplam 30 filtre içermektedir. Gürültülü verinin tanımlanmasında, 13'ü topluluk tabanlı filtreden (ensemble), 14'ü benzerlik (similarity) temeline dayanan filtreden ve diğer üç tanesi veri karmaşıklığını (data complexity) önleyebilme temeline dayanan gürültü filteleme fonksiyonlarından oluşmaktadır. Sınıf gürültüsünün giderebilmesi yaklaşımı dikkate alındığında, dördü yeniden etiketleme veya sınıflama olasılığını (possibility of relabelling) birleştirirken, diğer 26 tane fonksiyon sadece sınıfı ortadan kaldırma (removing) olmak üzere bu iki yaklaşım kullanılmaktadır. Tablo 3'de, literatürde en çok tercih edilen *NoiseFiltersR* paketinin filtre fonsiyonlarının tam bir listesi yer almaktadır. Paket içerisinde yer alan fonksiyonların tamamının referansları ilgili dökümanda mevcuttur (Morales, Luengo, Garcia, Lorena, De Carvalho, et al., 2017).

Tablo 3: NoiseFilters Paketinin Sınıf Gürültüsü Filtreleme Algoritması

		Gürültü Verinin Tanımlanması		
		Topluluk Tabanlı Filtreler	**Benzerlik Tabanlı Filtreler**	**Veri Karmaşıklığı Tabanlı Filtreler**
Sınıf Gürültüsünün Giderebilmesi	**Sınıfı Ortadan Kaldırma**	C45robustFilter C45votingFilter C45iteratedVotingFilter CVCF dynamicCF edgeBoostFilter EF HARF INFFC IPF ORBoostFilter PF	AENN BBNR CNN DROP1 DROP2 DROP3 ENG ENN PRISM RNN TomekLinks	saturationFilter consensusSF classifSF
	Yeniden Sınıflama	hybridRepairFilter	EWF GE ModeFilter	

Bu tez kapsamında, sınıflandırmada sınıf gürültüsü problemi ele alınacaktır ve literatürde en çok kullanılan *NoiseFiltersR* paketi fonksiyonlarından *hybridRepairFilter* algoritması kullanılacaktır.

3.3.3. Sınıf Dengesizliği Veri Ön İşleme: SMOTE Fonksiyonu

SMOTE fonksiyonu *k, perc.over* ve *perc.under* olmak üzere üç farklı parametreye sahiptir. Fonksiyonda yer alan *k* parametresi, azınlık sınıfından yeni gözlemler üretmek için kullanılan en yakın komşu sayısını; *pec.over* parametresi, aşırı örnekleme olarak da bilinen azınlık sınıfından ne kadar gözlem üretileceğine belirleyen bir sayıyı; *perc.under* parametresi ise, düşük örnekleme olarak bilinen azınlık sınıfında bulunan her bir gözlem için çoğunluk sınıflarından ne kadar gözlem seçilebileceğini belirleyen bir sayıyı ifade etmektedir. *SMOTE* algoritması, sınıf dengesizliği probleminin çözümünde en çok tercih edilen algoritmaların başında gelmektedir (Chawla et al., 2002). Bu yöntem, azınlık sınıflarında bulunan gözlemlerin belli bir sayıda en yakın komşular algoritması kullanılarak azınlık sınıfı için yeni

yapay gözlemlerin üretilmesi temel mantığı dayanmaktadır. Aynı zamanda, çoğunluk sınıfı gözlemlerini de düşük sayıda örneklemdirilerek daha dengeli bir veri seti oluşturmaktadır. *Perc.over* ve *perc.under* parametreleri, azınlık sınıfının aşırı örneklemesinin ve çoğunluk sınıfının düşük örneklemesinin kontrol edilmesini sağlamaktadır. *Perc.over* parametresinin 100 üzerinde bir sayı verilmesi önerilmektedir. Bu tür değerler ile azınlık sınıfına ait orijinal veri setindeki her bir gözlem için, o sınıfın $perc.over/100$ kadar yeni gözlem üretmesini sağlamaktadır. Eğer *perc.over* parametresi 100 değerinin altında bir değer alırsa, orijinal veri setinde azınlık sınıfına ait gözlemler için rastgele seçilen bir oranında tek bir gözlem üretmektedir. *Perc.under* parametresi, son dengeli veri seti için rastgele seçilecek olan çoğunluk sınıfındaki gözlemlerin oranını kontrol ettiğinden, bu oran yeni oluşturulan azınlık sınıfı gözlem sayısına bağlı olarak hesaplanmaktadır. Örneğin, azınlık sınıfı için 200 yeni gözlem oluşturulduğu varsayıldığında, 100'lük bir *perc.under* parametre değeri rastgele olarak, orijinal veri setinden final veri seti için çoğunluk sınıflarına ait 200 gözlem seçecektir. Fonksiyondaki *k* parametresi yeni gözlemlerin üretilme şeklini kontrol ettiğinden, mevcut olan herhangi bir azınlık sınıfı gözlemine bağlı olarak yeni gözlemler oluşturulacaktır. Aynı zamanda *k* parametresinin ne kadar gözlem üreteceği ise yine belirtildiği gibi parametrik algılayıcı olarak adlandırılan *perc.over* parametresi tarafından kontrol edilmektedir. Bu *k* parametresine bağlı olarak üretilen gözlemler, azınlık sınıfının her bir gözlemindeki en yakın komşulardan gelen bilgiler kullanılarak üretilmektedir. Böylelikle, *k* parametresi de bu komşuların kaçının kullanıldığını kontrol etmektedir. Bu tez kapsamında, yeniden örnekleme yöntemlerinden SMOTE yöntemi kullanılarak sınıf dengesizliği problemi çözülmeye çalışılacaktır (Kuhn, 2008).

3.4. Kayıp Gözlem Yüzdesi, Sınıf Gürültüsü Yüzdesi ve Sınıf Dengesizliği Oranının Hesaplanması

Çalışmada kullanılan veri setlerinin kayıp gözlem yüzdesi (proportion of missing value - PM), sınıf gürültüsü yüzdesi (noise rate - NR) ve sınıf dengesizliği oranı (imbalanced ratio - IR) aşağıdaki formüllere göre hesaplanmıştır.

- **Kayıp gözlem yüzdesi:** Bu yüzde, bağımsız değişkenlerin sahip olduğu toplam kayıp gözlem değeri sayısının örneklem genişliğine bölünmesiyle hesaplanmakta, 0 ile 100 arasında değer almaktadır (Joseph L Schafer & Graham, 2002).

$$PM = \frac{\sum_{i=1}^{m} x_i}{n} * 100 \quad (15)$$

x_i: bağımsız değişken kayıp gözlem sayısı, $i = 1, \dots, m$

n: toplam gözlem sayısı

- **Sınıf gürültüsü yüzdesi:** Bu yüzde, yanlış sınıflandırılmış gözlemlerin oranı olarak tanımlanmaktadır. Aynı zamanda, bağımlı kategorik değişken sınıflarının yeniden etiketlenerek onarılmasını ifade etmekte, 0 ile 100 arasında değer almaktadır (T. Liu & Tao, 2014; Natarajan, Dhillon, Ravikumar, & Tewari, 2018).

$$NR = \frac{\sum_{i=1}^{n} y_i}{\sum_{j}^{m} y_j} * 100 \tag{16}$$

y_i: toplam yanlış sınıflandırılmış gözlem sayısı, $i = 1, \dots, n$

y_j: bağımlı değişken toplam sınıf sayısı, $j = 1, \dots, m$

- **Sınıf dengesizliği oranı:** Sınıflandırmada, çoğunluk sınıfının azınlık sınıflarına bölünmesiyle elde edilen bir orandır. Bu oran, azınlık ile çoğunluk sınıfları arasındaki denge hakkında bilgi vermektedir (Noorhalim et al., 2019). İkili sınıf için dengesizlik oranın hesaplanması aşağıdaki gibidir.

$$IR(c) = \frac{max_c}{min_c} \tag{17}$$

max_c: Çoğunluk sınıfı örneklem genişliği

min_c: Azınlık sınıfı örneklem genişliği

Sınıf dengesizliği oranı, eğer $IR = 1$ ise tam dengeli veri seti olarak adlandırılır. Dengeli bir sınıf dağılımı için, bu oranın $IR \leq 1$ olması gerekmektedir. Eğer $IR \geq 1$ ise, sınıf dağılımının dengesiz olduğu anlaşılmaktadır.

3.5. Çalışmada Kullanılan R Paketleri ve Fonksiyonları

Bu tez çalışması kapsamında, bulguların elde edilmesi için kullanılan R paketleri ve fonksiyonları Tablo 4'de verilmiştir. Bu paketler, R ortamına indirildikten sonra, R kütüphanesine tanımlanarak kullanılır.

Tablo 4: Çalışmada Kullanılan R Paketleri ve Fonksiyonları

İşlem	Paket	Fonksiyon
Normallik test	*goftest*	*ad.test*
		shapiro.test
Korelasyon	*corrplot*	*corrplot*
		cor
Modelleme	*caret*	
	Proc	*multiclass.roc*
	purrr	*roc*
	randomForest	
	wsrf	
	caTools	
	gbm	
Kayıp gözlem	*mice*	*mice*
	VIM	*stripplot*
		plot
Sınıf gürültüsü	*NoiseFiltersR*	*hybridRepairFilter*
Sınıf dengesizliği	*caret*	*SMOTE*
	DMwR	
Grafikler	*ggplot2*	*ggplot*

Çalışmada kullanılan veri setlerine ait bağımsız sürekli değişkenlerin normallik sınaması için *shapiro.test* (Shapiro-Wilk testi) ve *ad.test* (Anderson-Darling testi) fonksiyonları kullanılmaktadır. Anderson-Darling testinin kullanılması için, *goftest* paketinin tanımlanması gerekmektedir. Klasik uyum iyiliği testleri olarak ifade edilen *goftest* paketi, Cramer-von Mises ve Anderson-Darling sürekli tek değişkenli dağılımları hesaplamak için kullanılan uyum testi algoritmalarını içermektedir. Sürekli değişkenler arasında ilişkinin test edilmesi

için, *cor* fonksiyonu kullanılmaktadır. Bu ilişkinin görselleştirilmesi için, *corrplot* paketinin içerisindeki *corrplot* fonksiyonu kullanılmaktadır. Orijinal ve işlenmiş veri setlerinin modellenmesi için kullanılan en temel paket, *caret* paketidir (Kuhn, 2008). Bu paket, regresyon ve sınıflandırma modellerinin eğitimi için kullanılan birçok fonksiyonu içermektedir. Bu pakete ek olarak, model kurulumunda gerekli diğer paketler ise, *randomForest, wsrf, caTools* ve *gbm* paketleridir (Blagus & Lusa, 2015; Breiman, 2001; Natekin & Knoll, 2013; Zhao, Williams, & Huang, 2017). Bu ek paketler, tez çalışmasında kullanılacak rastgele orman, ağırlıklı alt uzay rastgele orman, eklemeli lojistik regresyon ve gradyan artırma makinaları algoritmaları için gerekli olan paketlerdir. *Logitboost* algoritması için kullanılan *caTools* paketi, aynı zamanda istatistiksel bazı temel hesaplamaların yapılması ve eğri altında kalan alanın hızlı hesaplanması için kullanılmaktadır. Algoritmaların performans ölçüm metriklerinin karşılaştırması yine bu paket ile yapılmaktadır. Ayrıca, ROC eğrisi değerleri ve grafiği için Proc ve purr paketleri gerekmektedir. Bu paketler içinde yer alan *roc* fonksiyonu ikili sınıf bağımlı değişkeni için kullanılırken, *multiclass.roc* fonksiyonu çoklu sınıf bağımlı değişkenleri için kullanılmaktadır. Kayıp gözlem probelminin analizi için, mice paketi içerisindeki mice fonksiyonu kullanılarak çoklu atamalar yapılmaktadır. Kayıp değerlerin görselleştirilmesi ve atamaları için ek olarak *VIM* paketi içerisindeki *stripplot* ve *plot* fonksiyonları kullanılmaktadır (Kowarik & Templ, 2016). Sınıf gürültüsü problemi analizi için, *NoiseFiltersR* paketinde yer alan *hybridRepairFilter* fonksiyonu kullanılarak sınıflar gürültüden arındırılmaktadır (Morales, Luengo, Garcia, Lorena, De Carvalho, et al., 2017). Sınıf dengesizliği problemi analizinde, *caret* paketine ek olarak *DMwR* paketi kullanılarak *SMOTE* fonksiyonu ile sınıflar dengelenmektedir. Tüm ölçüm metriklerinin hem orijinal hem de işlenmiş veri setleri için analiz sonuçlarının görselleştirilmesinde *ggplot2* paketinde yer alan *ggplot* fonksiyonu kullanılmaktadır.

3.6. Makine Öğrenme Algoritmaları

Makine öğrenme algoritmaları, birden çok disiplini içinde barındıran ve verilere dayalı öğrenmeyi temel alarak geliştirilen algoritmaların süreçlerini konu almaktadır (Al-Jarrah, Muhaidat, Karagiannidis, & Taha, 2015). Bu disipinler arasında istatistik, olasılık, karar kuramı, yapay zeka, veri madenciliği, örüntü tanıma, bilgisayar bilimi gibi alanlar yer almaktadır (Murphy, 2012). Makine öğrenme yöntemlerinin gelişimi sayesinde; tıbbi tanı, DNA dizilerinin sınıflandırılması, biyoinformatik, doğal dil işleme, konuşma ve elyazısı tanıma arama motorları, örüntü tanıma, beyin-makine arayüzleri, kredi kartı dolandırıcılığı

denetimi, nesne tanıma, oyun oynama, dinamik web siteleri ve yazılım mühendisliği gibi birçok uygulama alanında hızlı gelişmeler olmuştur (Ongsulee, 2017). Günümüzde ise makine öğrenme sayesinde kendi kendini süren arabalar, sesle çalışan asistanlar gibi birçok teknolojik gelişmenin önü açılmaktadır (Angra & Ahuja, 2017).

Yapay zeka alanında yapılan gelişmelerle makine öğrenme yöntemleri de birçok alanda kullanılmaya başlanmıştır. Özellikler sağlık alanında yapılan uygulamalardan biri de tıbbi tanının konmasıdır. Dünyada erken tanı son derecede önemli bir durumdur. Bu yüzden, hastalık tanısının doğru sınıflandırmasını sağlayacak istatistiksel sınıflandırma model yaklaşımlarının kullanımı da büyük önem kazanmaktadır. Bu amaçla, makine öğrenme yöntemleri, tıbbi tanı ve tedavinin önemli görevlerine yardımcı olmak amacıyla faydalı bilgileri sunmak için uygulanmaktadır. Bu yöntemler, kullanılan tıbbi verilerin farklı perspektiflerden analiz edilmesini ve yorumlanmasını sağlamaktadır (Kavakiotis et al., 2017). Makine öğrenme yöntemleri, temelde üç tipte incelenmektedir.

- Denetimli (*Supervised*) Makine Öğrenme Algoritmaları
- Denetimsiz (*Unsupervised*) Makine Öğrenme Algoritmaları
- Yarı denetimli (*Semi - supervised*) Makine Öğrenme Algoritmaları

Makine öğrenme yöntemleri, çalışma verisinin ilgilendiği sonuca bağlı olarak Tablo 5'de gösterildiği gibi farklılık göstermektedir.

Tablo 5: Makine Öğrenme Yöntemleri Tipleri

	Denetimli öğrenme	**Denetimsiz öğrenme**
Kategorik veri	Sınıflandırma	Kümeleme
Sürekli veri	Regresyon	Boyut indirgeme

Eğer bağımlı değişken kategorik veri (categorical data) tipinde ise denetimli öğrenme sınıflandırma amacıyla, denetimsiz öğrenme ise kümeleme yapmak için kullanılmaktadır. Bağımlı değişkenin sürekli veri (continuous data) tipinde olduğu durumda ise, denetimli öğrenme regresyon modeli oluşturmak, denetimsiz öğrenme ise boyut indirgeme amacıyla tercih edilmektedir.

3.6.1. Denetimli Makine Öğrenme Algoritmaları

Denetimli makine öğrenme algoritmaları, etiketlenmiş bir eğitim veri setinden girdi ile çıktılar arasında ilişki kurulması mantığına dayanmaktadır (Witten, Frank, & Hall, 2011). Eğitim veri setiyle kurulan modelle, yeni girişler için doğru çıktıyı tahminleme öğrenilmektedir. Denetimli öğrenme, eğitime dahil edilemeyen veriler için genellenebilir tahminler ile gözlenenler arasındaki sapmaları en aza indirgemek için algoritmalarda en uygun parametreleri belirlemek üzere tasarlanmıştır. Modelin genelleştirilebilirliği test seti ile tahmin edilebilmektedir (Szolovits & Pauker, 1978). Temel amaç, sonuçları bilinen veri setinden yapılan sınıflandırmadan hareketle, sonuçları bilinmeyen yeni veri setine dair etkili tahminler yapılmasını sağlamaktır. Sağlık alanında denetimli öğrenmeye örnek olarak; radyolojide, bir akciğer nodülünün göğüs röntgeninden otomatik olarak algılanması gösterilebilir. Bu tür denetimli öğrenme modellerinin kullanılmasıyla hekim tarafından bazı öngörülemeyen yeni ilişkiler ortaya çıkarılabilmektedir. Böylelikle sağlıkta risk tahminleri denetimli öğrenmeyle yapılabilmektedir (Roberts et al., 2016). Denetimli öğrenme sınıflandırma algoritmaları genel tanımı,

$$D_{train} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_{train}} \tag{18}$$

$\boldsymbol{x}_i \in X$: girdi vektörü

$y_i \in Y$: sınıf etiketi, $Y \in \{c_1, \dots, c_K\}$ Ksınıf etiket sayısı

N_{train}: Eğitim veri setindeki gözlem sayısı

olarak ifade edilmektedir. Sınıflandırmada, girdi vektörü ile sınıf etiketleri arasındaki ilişki modellenmektedir. Genellikle, gözlemlerin sabit fakat bilinmeyen bir S dağılımdan gelen bağımsız ve özdeş dağılıma (independently and identically distributed - iid) sahip rastgele değişkenler olduğu varsayılmaktadır.

Bir h hipotezi, giriş vektörleri setini (X) karşılık gelen sınıf etiketleri setine (Y) bir fonksiyondur. Haritalama fonksiyonu (mapping function) olarak da adlandırılan bu fonksiyon, X giriş bölgesini K farklı bölgelere böler. Bu bölgelerin her biri bir sınıfa karşılık gelmektedir. Bu bölgeler arasındaki ayrımlara karar sınırları (decision boundaries) denir. Genelleme hatası, S dağılımından gelen yanlış sınıflandırma gözleminin $(\boldsymbol{x}, y)$ olasılığı ifade etmektedir.

$$Error(h) = Pr_{(\boldsymbol{x},y)\sim S}[h(x) \neq y] \tag{19}$$

Kesin genelleme hatasının hesaplanması nadirde olsa mümkündür çünkü olasılık dağılımının altta yatan dağılımı genellikle bilinmemektedir (S. B. Kotsiantis, 2007). Bu nedenle, verilen bir hipotezi değerlendirmek için test veri seti üzerinde deneysel hata (empirical error) hesaplanır. Aynı şekilde, test veri seti içinde aynı varsayımlar geçerlidir.

$$D_{test} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_{test}} \tag{20}$$

$\boldsymbol{x}_i \in X$: girdi vektörü

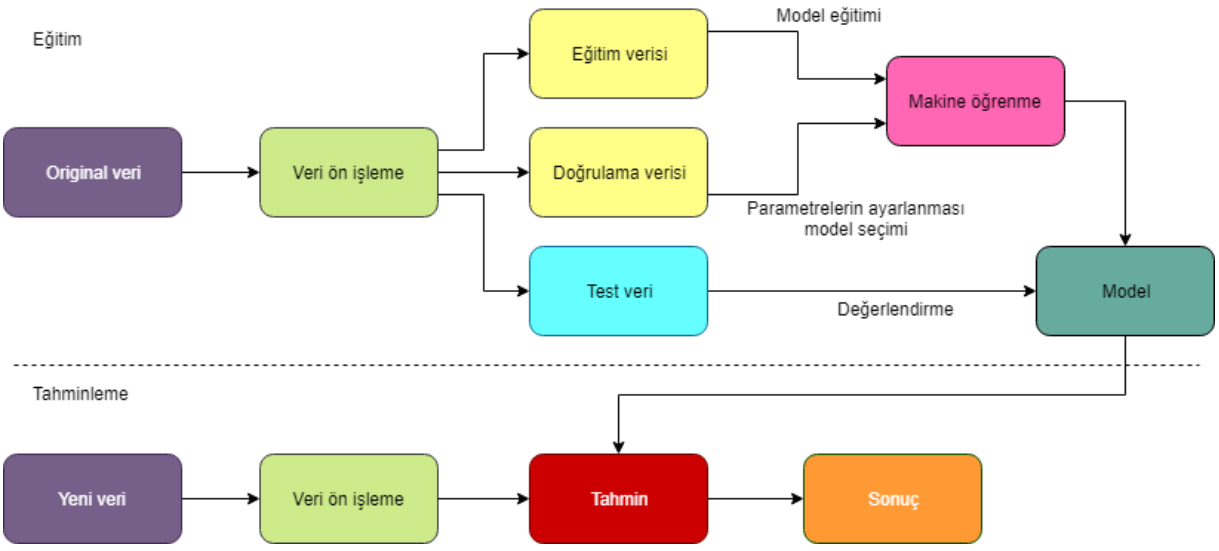
$y_i \in Y$: sınıf etiketi, $Y \in \{c_1, \dots, c_K\}$ Ksınıf etiket sayısı

N_{train}: Test veri setindeki gözlem sayısı

Deneysel hata, D_{test}'teki gözlemlerin ortalama yanlış sınıflandırma hatasını ifade etmektedir.

$$Error_{test}(h) = \frac{1}{N_{test}} \sum_{j=1}^{N_{test}} 1h(\boldsymbol{x_j}) \neq y_j \tag{21}$$

1 değeri, işaret fonksiyonunu (indicator function) temsil etmektedir. Sınıflandırmada, hipotez, eğitim veri setlerinden öğrenerek modellenmektedir. Modellenmiş hipotezin iyi bir genelleme yeteneğine sahip olması gerekmektedir. Başka bir ifadeyle, yeni gözlemler için de doğru tahminler yapması beklenmektedir.



Şekil 8: Denetimli Öğrenme Yöntemlerinin Genel Çalışma Prensibi

Denetimli makine öğrenmesi yaklaşımlarının genel iş akışı Şekil 8'de gösterildiği gibi,

- Öncelikle veriden eğitim ve test veri setleri ayrılır. Bu ayırım için 80/20, 70/30 ve 60/40 gibi farklı oranlar literatürde önerilmektedir (Dobbin & Simon, 2011; Y. Xu & Goodacre, 2018).
- Daha sonra, eğitim setinin bir kısmı tahmin modelini oluşturmak için, diğer kısmı ise modeli ayarlamak (tune) ve doğrulamak (validate) için kullanılır.
- Makine öğrenme modeli tamamlandıktan sonra oluşturulan model, test veri kümesi için tahminler yapmak için kullanılır ve modelin performansı, tahmin edilen sonuçlar ile test veri kümesi için gözlemlenen sonuçlar karşılaştırılarak tahmin edilir.

3.6.2. Denetimsiz Makine Öğrenme Algoritmaları

Denetimsiz makine öğrenme algoritmaları, yalnızca giriş verilerini içeren verilerin altında yatan yapıları veya dağılımları modellemektedir (Witten et al., 2011). Denetimsiz öğrenme, ham verilerin alt kümelerini bulmak, verilerdeki aykırı değerleri belirlemek veya büyük verilerde boyut indirgemek için etiketlenmemiş verilerin altında yatan modelleri oluşturmaktadır (Berner et al., 1994). Buradaki temel amaç, etiketsiz veriyi değişkenler arasındaki ilişkilere dayalı olarak kümeleyerek çeşitli modeller ya da yapılar oluşturabilmektir. Böylelikle denetimsiz öğrenmede, veriler arasındaki benzerliğe dayanarak veri yapısı korunmakta ve boyutsallığın azaltılmasıyla kümelendirme yapılmaktadır. Denetimsiz öğrenmede, tahmin edilecek çıktılar bulunmamaktadır. Bunun yerine, denetimsiz öğrenmede verilerin yapısından dolayı doğal olarak oluşan desenler ve gruplamalar bulunmaktadır. Sağlık alanında, hassas tıp olarak adlandırılan alanda daha çok tercih edilmektedir. Genellikle nadir görülen hastalıklar içinde değil de daha çok bilindik hastalıkların tanımlanması için kullanılması önerilmektedir. Özellikle karmaşık çoklu kategorili hastalıklar için bu tür mekanizmaların kullanılarak tanımlanması çok zor bir durumdur. Denetimli öğrenmeyle karşılaştırıldığında hastalık tanısı için öngörülen bir sonuç yoktur. Denetimli öğrenmede sadece, verilerinin yapısıyla ilgilenilmektedir. Burada hastalığı bir kategoride tanımlamak yerine alt gruplar oluşturularak olası yeni hastalık mekanizmaları tanımlanabilmektedir (Alashwal, El Halaby, Crouse, Abdalla, & Moustafa, 2019).

Denetimsiz makine öğrenmesi, kümeleme (clustering), anormalliği tespit etme (anomaly detection) ve boyut indirgeme (dimensionality reduction) amacıyla kullanılır. Kümeleme algoritmaları, veri setlerini benzer ölçümlerle kümelere ayırmayı amaçlamaktadır. Anomali

tespiti, veri kümesindeki aykırıkları tanımlamaktadır. Boyut indirgeme ise, verileri tanımlamak için kullanılan rastgele değişken sayısını azaltmaktadır. Buna örnek olarak, binlerce parametreli bir görüntüyü daha küçük bir özet parametreli vektöre indirgemek verilebilir. Elde edilen özet vektörü ham verilerdeki önemli bilgileri korumaktadır. Buna bağlı olarak, benzer görüntülerden gelen özet vektörler, alakasız görüntülerden elde edilen özet vektörlere göre daha fazla benzerlik gösterecektir (Chimienti et al., 2016; Sidey-Gibbons & Sidey-Gibbons, 2019).

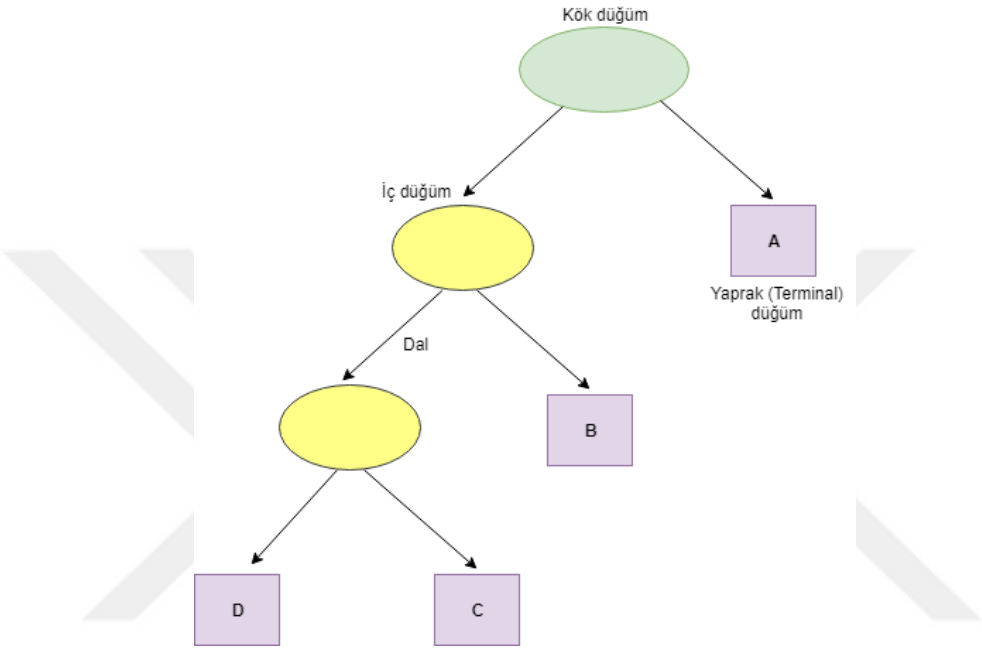
3.6.3. Yarı denetimli Makine Öğrenme Algoritmaları

Yarı denetimli makine öğrenme algoritmaları ise, eğitim veri seti için etiketlenmemiş verilerin kullanımını da sağlayan denetimli öğrenme yöntemlerinin mantığıyla benzerdir. Yarı denetimli öğrenme, denetimsiz öğrenme (herhangi bir etiketli eğitim verisi olmadan) ve denetimli öğrenme (tamamen etiketli eğitim verileriyle) arasındadır. Sağlık araştırmalarında sıklıkla raslanan bir diğer durum ise eğitim setinde yanıt değişkeni etiketlerine (örneğin, hasta ve sağlıklı, kanser ve kanser değil gibi) tam olarak ulaşılamıyor olması durumudur. Bu durumda kullanılan sınıflandırma yöntemleri etkinliğini kaybetmekte ve mevcut veriler ile etkili bir sınıflandırma yaklaşımı geliştirilebilmesi için, yarı denetimli öğrenme yöntemleri kullanılabilmektedir. Yarı denetimli öğrenme yöntemleri, büyük oranda etiketlenmemiş verilerle etiketlenmiş küçük miktardaki veriyi kullanarak, etiketlenmemiş veriyi eğitmek üzere kullanan yöntemlerdir (Xiaojin Zhu, 2008). Yarı denetimli öğrenme, denetimsiz öğrenme (herhangi bir etiketli eğitim verisi olmadan) ve denetimli öğrenme (tamamen etiketli eğitim verileriyle) arasında yer almakta ve gerçek hayat verilerinde eğitim verilerinin etiketlenmesinde kullanıldığında zor, pahalı ya da zaman alıcı bir yöntem olmaktadır (Hady & Schwenker, 2013). Son yıllarda oldukça fazla ilgi duyulan yarı denetimli öğrenme yöntemleri; üretken modeller, düşük yoğunluklu ayırma, grafik tabanlı yöntemler ve sezgisel yaklaşımlar olmak üzere dört ana başlık altında incelenmektedir (Chapelle, Scholkopf, & Zien, Eds., 2009).

3.7. Karar Ağaçları

Topluluk öğrenme algoritmalarına geçmeden önce kısaca karar ağaçlarından bahsetmek gerekir. Karar ağaçlarının diğer bir adı ağaç tabanlı öğrenme algoritmalarıdır. Literatürde sınıflandırma problemi için en çok tercih edilen karar ağaçları algoritmaları, CART ve C4.5 algoritmalarıdır (Gupta, Rawat, Jain, Arora, & Dhami, 2017). Sınıflandırma probleminde ilk adım, eğitim veri seti ile model kurulmasıdır. İkinci adım ise, test veri kümesiyle kurulan

modelin test edilmesidir. Sınıflandırma karar ağaçlarında hedef değişken yani bağımlı değişken kategorileri önceden bilinmektedir. Karar ağaçları kayıp gözlemlere karşı duyarlı değildir. Aynı zamanda karar ağacı modelinde aşırı uyum yaşanabileceğinden model parametrelerinin ayarlanması ve budama yapılması önerilmektedir (Kingsford & Salzberg, 2008).



Şekil 9: Bir Karar Ağacının Oluşturulması

Karar ağaçları yapısal olarak üç aşamadan oluşmaktadır (Şekil 9).

1. Orijinal veri setinden eğitim ve test verileri belli oranlara göre ayrılmaktadır. Eğitim veri seti örneklem genişliği oransal olarak test veri setinden büyük olmaktadır. Karar ağacı bölünmeye tüm eğitim veri setini içinde barındıran kök düğümden (root node) başlamaktadır. Eğer bağımsız değişken sürekli ise, bölünme noktası için tek bir değer hesaplanarak kök dallara ayrılmaktadır (Mathuria, 2013). Bağımlı değişkene ait değerler, bölünme noktası değerinden büyük ise, karar sağ düğüme geçmekte; küçük ve eşitse karar sol düğüme geçmektedir. Eğer bağımsız değişken kategorik ise, bölünme kuralına göre kategorilerin bir alt kümesini sol düğüme, geri kalanını ise sağ düğüme geçirmektedir. Kök düğümün bölünmesi için kullanılan bölünme kuralı, karar ağacının oluşmasında kullanılacak tüm bağımsız değişkenlere uygulanmaktadır. Tüm

bağımsız değişkenler için hesaplanan bölünme noktaları içerisinden, en iyi ayrımı sağlayan noktalar seçilerek düğümler elde edilir. Diğer bir deyişle, kök düğüm alt dallara ayrıldıktan sonra, iç düğümler elde edilir. İç düğümler (interior nodes) ise, aynı bölünme kuralına göre tekrarlı olarak bölünmeye devam etmektedir. Bölünmenin durdurulması yine bir durdurma kuralına bağlıdır. Durdurma kuralları genel olarak, düğümdeki minimum değişken sayısına ve maksimum ağaç derinliğine bağlıdır. Ağaç derinliği, kök düğümden başlayarak en uzaktaki yaprak düğüme kadar olan iç düğüm sayısı olarak tanımlanmaktadır (Zhou, 2012).

2. Karar ağaçları belli bir durdurma kuralı uygulanmadıkça olabildiğince büyük bir ağaç oluşturmaktadır. Bunun önüne geçebilmek için budama (pruning) işlemi uygulanmaktadır. Budama yapmak, karar modeli için aşırı uyum problemini ortadan kaldırmayı sağlamaktadır (Zhou, 2012). Aynı zamanda, yanlış sınıflandırma hatasını (misclasification error) en aza indirgeyerek karar ağacının en uygun genişliğine karar verilmesini sağlamaktadır. Böylelikle bazı iç düğümlerin kaldırılmasıyla elde edilen alt ağaç gruplarından en uygun olanı seçilmektedir.
3. Eğitim veri setiyle kurulan karar ağacı modelinin test veri setiyle test edilmesinde (örneğin; çapraz geçerlilik yöntemi), budama sonucu elde edilen alt ağaç gruplarından en iyi olan karar ağacı seçilir (Witten & Frank, 2005).

Karar ağaçları algoritmalarındaki en önemli sorunlardan biri, hangi kökten itibaren bölünmenin başlayacağı ve hangi kritere göre (bölünme kuralına göre) bölünmenin yapılacağıdır. Aynı zamanda, karar ağaçları algoritmalarında kök düğümünün nasıl bölüneceği konusu, ağacın doğruluğunu etkileyen faktörlerden birisidir. Sınıflandırma problemi için bu bölünme kriterleri, entropi, bilgi kazancı (information gain), Gini indeksi, Towing indeksi, karışıklık (impurity) ölçütü, benzerlik oran ki-kare istatistikleri (chi-square measure for association) ve Kolmogorov-Smirnov ölçütü ile sağlanmaktadır. ID3 (Iterative Dichotomiser 3) ve C4.5 algoritması, entropi ve bilgi kazancına dayalı algoritmalarıyken; CART algoritması, Towing ve Gini indeksi temeline dayalı bir algoritmadır. Genel olarak CART algoritması, Gini indeksine bağlı ikili (binary) bölünme gerçekleştirmektedir (Song & Lu, 2015). Bu yüzden son düğümde iki dal bulunmaktadır. Bu algoritmada budama yapılması, ağacın karmaşıklık ölçüsüne bağlıdır. C4.5 algoritması, ID3 karar ağacı modelinin daha gelişmiş bir versiyonudur. Bu algoritma, kategorik bağımlı değişkenin ikiden fazla kategori seviyesine yani çok sınıflı (multi class) sahip olduğu durumda kullanılmaktadır. Bu durumda düğümlerde bulunan dalların sayısı bağımlı değişken kategori sayısına eşittir. C4.5

algoritması, tek bir sınıflandırıcı için birden çok ağacını bir araya getirerek birleştirmektedir. C4.5, düğümün bölünmesi için bilgi kazancı algoritmasını kullanırken, her yapraktaki hata oranına göre budama yapmaktadır (Zhou, 2012).

Özellikle sağlık alanında hastalıkların sınıflandırılması için tercih edilen karar ağaçları algoritmalarının sağladığı bazı avantajlar bulunmaktadır (Song & Lu, 2015). Değişkenler arasında ilişkinin belirlenmesi, kayıp gözlem problemine karşı duyarlı olma, sapan gözlemlerden (outliers) etkilenmeme, hem sürekli hem de kategorik değişkenlerin bir arada değerlendirilmesi bunlardan bazılarıdır. Bunların yanı sıra karar ağaçlarının en önemli dezavantajı genellenebilir yani kesin sonuçlar vermemesidir. Bu kesin sonuçlardan kasıt ise, sınıflandırma modelleme performanslarıyla ilgilidir. Burada modellere ait doğruluk değerlerinin olabildiğince yüksek olması beklenmektedir (Bae, 2014).

Son zamanda gelişen teknoloji ile birlikte karar ağaçları algoritmaları da bu gelişime ayak uydurmuştur. Özellikle, topluluk öğrenme olarak bilinen ağaç tabanlı topluluk yöntemler (tree-based ensembles methods) kullanılarak, farklı sınıflandırma algoritmalarıyla elde edilen algoritmalar birleştirilmekte, böylece daha kesin sonuçların elde edilmesi sağlanmaktadır (J. R. Quinlan, 1986; Zhou, 2012).

3.7.1. Entropi ve Bilgi Kazancı

Bir sistemdeki belirsizliğin veya düzensizliğin ölçüsü, entropi olarak tanımlanmaktadır. C4.5 algoritması, sınıflandırmada ağacın bölünme kriteri olarak, bölünme bilgisi kavramını kullanan bilgi kazancı yönteminden yararlanarak hesaplanan kazanç oranlarını kullanmaktadır (J.R Quinlan, 1999). Entropi değer aralığı, sonuç sayısına (m) bağlı olarak değişmektedir. Buna göre, $0 \leq Entropi \leq \log_2(m)$ arasında değer almaktadır. Eğer entropi değeri 0 (minimum entropi) olarak hesaplanırsa, olasılıklardan biri 1 olur ve diğerleri 0 değerini alır. Eğer $\log_2(m)$ (maksimum entropi) entropi değeri en yüksek değeri alırsa, tüm olasılıklar $1/m$ değerine eşit olur. 1 değerine ne kadar yaklaşırsa belirsizlik artmaktadır. $P(p_1, \ldots, p_m)$ sınıf dağılım olasılıklarını sahip bir veri setinin S entropi değeri,

$$E(S) = -\sum_{k=1}^{m} p_i \log_2(p_i) \tag{22}$$

olarak tanımlanır.

p_i: S veri setindeki i. sınıfın olasılığı

i. sınıfına ait gözlem sayısının tüm veri setindeki toplam gözlem sayısına bölünmesiyle elde edilmektedir. S veri seti için X değişkeni üzerinden n tane alt bölüme ayrılacağını varsayalım. Buna göre, X değişkenine bağlı bilgi kazancı aşağıdaki gibi hesaplanmaktadır,

$$Bilgi\ Kazancı = (S, X) = E(S) - \sum_{k=1}^{m} p(S_i)E(S_i) \tag{23}$$

$E(S)$: S veri setinin X değişkeni üzerinde bölünmeden önceki entropisi

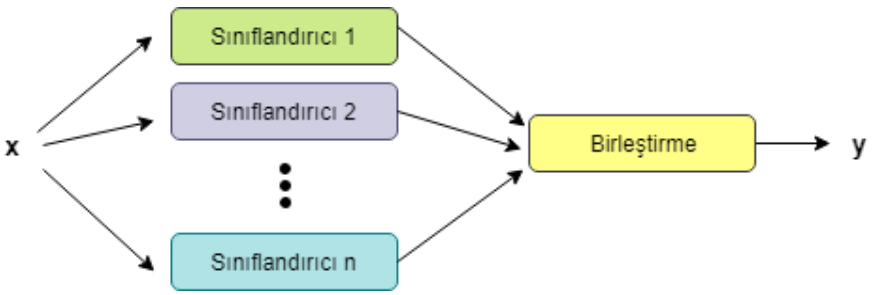
$p(S_i)$: i. sınıfına ait alt bölümünün X değişkeni üzerinde bölünme gerçekleştikten sonraki olasılığı

$E(S_i)$: i. sınıfına ait alt bölümünün X değişkeni üzerinde bölünme gerçekleştikten sonraki entropisi

Sonuç olarak, bilgi kazancının en yüksek olduğu değişken en iyi dallara ayırma kriteri değişken olarak seçilir. Böylelikle, bölünmeye bu değişkenden üzerinden başlanılmaktadır (Witten & Frank, 2005).

3.8. Topluluk Öğrenme

Komite temelli öğrenme (committee-based learning) veya çoklu sınıflandırıcı öğrenme sistemleri (learning multiple classifier systems) olarak adlandırılan topluluk öğrenme yöntemleri (ensemble learning methods), aynı problemi çözmek için birden fazla sınıflandırıcıyı aynı anda eğitmektedir (Yang, Yang, Zhou, & Zomaya, 2010). Eğitim verilerinden tek bir sınıflandırıcı ile model kurmaya çalışan öğrenme yaklaşımlarının aksine, topluluk öğrenme yöntemleri birden çok sınıflandırıcı ile model kurmaya ve birleştirmeye çalışmaktadır (Şekil 10).



Şekil 10: Genel Bir Topluluk Öğrenme Çalışma Modeli (Zhou, 2012)

Topluluk öğrenme yöntemlerinde bir topluluk, temel sınıflandırıcılar (base classifiers) olarak adlandırılan çok sayıda sınıflandırıcıyı içermektedir. Temel sınıflandırıcılar, genellikle eğitim verilerinden k-en yakın komşular, naive Bayes, karar ağacı, yapay sinir ağı veya diğer temel öğrenme algoritmaları (base learning algorithms) tarafından oluşturulmaktadır. Çoğu topluluk öğrenme yöntemi, aynı türden sınıflandırıcılar kullanarak homojen topluluklar (homogeneous ensembles) oluşturmaktadır. Bu durumda, homojen temel sınıflandırıcılar (homogeneous base learners) oluşturulurken, tek bir temel öğrenme algoritması kullanılmaktadır. Diğer bir durumda ise, farklı türden sınıflandırıcılar kullanarak heterojen topluluklar (heterogeneous ensembles) oluşturulmaktadır. Böylelikle, heterojen topluluklar oluşturulurken çoklu öğrenme algoritmaları (multiple learning algorithms) kullanılarak, birden çok yöntem bir arada bulunmaktadır. Heterojen topluluk durumunda, tek bir temel öğrenme algoritması olmadığından; genellikle sınıflandırıcıları, temel sınıflandırıcılar yerine bireysel sınıflandırıcıları (individual classifiers) veya bileşen sınıflandırıcıları (component classifiers) tercih etmektedir. Heterojen topluluk öğrenme yöntemlerin başarısı, temel sınıflandırıcıların öğrenme başarısı ve bu sınıflandırıcıların birbirlerinden farklılıkları olmak üzere bu iki ölçüte göre kıyaslanmaktadır (Zhou, 2012).

Topluluk öğrenme yöntemleri, tek bir sınıflandırıcı modeline kıyasla daha güçlü ve genellenebilir sonuçlar elde etmek amacıyla birden fazla modelin tahmin sonuçlarını birleştirmektedir. Bir topluluğun genellenebilirliği, genellikle temel sınıflandırıcıların genellenebilirliği özelliğinden çok daha güçlüdür (Verma & Hassan, 2011). Topluluk öğrenme yöntemlerinin daha fazla tercih edilmesinin sebebi, rastgele tahminler yapabilen zayıf sınıflandırıcıların (weak classifiers), kesin tahminler yapabilen güçlü sınıflandırıcılara (strong classifiers) göre daha da güçlendirebilmeleridir. Bu sayede, zayıf sınıflandırıcıların tahminleme gücünün artmasını sağlamaktadır. Bundan dolayı, temel sınıflandırıcılara aynı zamanda zayıf sınıflandırıcılar da denmektedir. Diğer yandan, topluluk öğrenme yöntemleri, kullanılan tek bir sınıflandırma algoritmasının eğitim veri setinde ezbere başvurmasının önüne geçmekte ve sınıflandırma başarısının daha yüksek hesaplanmasının beklendiği durumlarda kullanılmaktadır (Dietterich, 2000b).

Topluluk öğrenme yöntemlerinde, toplulukların birleştirmesi konusunda birçok çalışma yapılmıştır. Genel olarak birleştirme kurallarının gelişimi konusu üç farklı şekilde ortaya çıkmıştır (Zhou, 2012). Bunlar, sınıflandırıcıların birleştirilmesi (combining classifiers), zayıf sınıflandırıcıların topluluklarının birleştirilmesi (ensembles of weak learners) ve uzmanlarının karışımının birleştirilmesidir (mixture of experts).

Sınıflandırıcıların birleştirilmesi, çoğunlukla örüntü tanıma topluluğu (pattern recognition community) konusunda incelenmektedir. Bu konuda, araştırmacılar genellikle güçlü sınıflandırıcılar kullanarak daha güçlü birleşik sınıflandırıcılar (stronger combined classifiers) elde etmek için güçlü birleştirme kuralları geliştirmektedirler. Bununla birlikte, farklı birleştirme kurallarının geliştirilmesi ve kullanılması artmaktadır (Hastie, Tibshirani, & Friedman, 2009).

Zayıf sınıflandırıcıların topluluklarının birleştirilmesi konusu çoğunlukla makine öğrenme topluluğunda incelenmektedir. Bu konuda ise, araştırmacılar genellikle zayıf sınıflandırıcılar üzerinde çalışmaktadırlar. Zayıf sınıflandırıcıların performanslarını zayıftan güçlüğü doğru artırmak için güçlü algoritmalar geliştirilmiştir. Bu durum, artırma, Uyarlanabilir Artırma (Adaptive Boost - AdaBoost), ve torbalama gibi literatürde çok fazla kullanılan topluluk öğrenme yöntemlerinin ortaya çıkmasına neden olmuştur (Freund & Schapire, 1997). Böylelikle, zayıf sınıflandırıcıların ne kadar zayıf olduğunun anlaşılmasını sağlayan teorik bilgi araştırmacılara sunulmuştur.

Uzmanların karışımının birleştirilmesi, çoğunlukla sinir ağları topluluğu konusunda incelenmiştir. Bu konuda, araştırmacılar genellikle böl ve yönet stratejisini (divide-and-conquer strategy) dikkate almaktadırlar. Böylelikle, parametrik modellerin ortak bir karışımını öğrenmeye çalışarak genel bir çözüm elde etmek için birleştirme kuralları kullanılmaktadır (Zhou, 2012).

Topluluk öğrenme modellerinin geliştirilmesinin kesin bir tarihi olmadığından, çoklu modellerin kullanılmasıyla birlikte topluluk öğrenme yöntemleri, 1990'lı yıllardan bu yana yapılan iki farklı çalışmanın sonucunda daha da gelişim göstermiştir. Bunlardan ilki, Hansen ve Salamon (1990) tarafından yapılan deneysel çalışmadır. Bu çalışmada, birden çok sınıflandırıcı birleştirilmesiyle yapılan tahminlerin başarısının, en iyi tek sınıflandırıcı tarafından yapılan tahminlerin başarısından daha yüksek olduğu saptanmıştır (Hansen & Salamon, 1990). Diğer çalışmada ise, Schapire (1990) bir teorik çalışma yapılmıştır. Bu çalışma sonucunda, zayıf sınıflandırıcıların performanslarının güçlü sınıflandırıcıların performanslarına yükseltilebileceği kanıtlanmıştır. Zayıf sınıflandırıcıların sonuçlarının elde edilmesi güçlü sınıflandırıcılardan daha kolay olmasına rağmen topluluk öğrenme yöntemleri güçlü sınıflandırıcıların sonuçlarının elde edilmesi konusunda kolaylık sağlamaktadır (Schapire, 1990).

Genel anlamda, bir topluluğun kurulması temelde iki aşamadan oluşmaktadır. Bunlar, temel sınıflandırıcıların oluşturulması ve bu sınıflandırıcıların birleştirilmesidir. İyi bir topluluğun oluşturulması için, genellikle mümkün olabildiğince doğru ve çeşitli temel sınıflandırıcıların kullanılması gerektiği düşünülmektedir (Maimon & Rokach, 2005). Bir topluluğun oluşturulma maliyeti, tek bir sınıflandırıcı oluşturulması maliyetinden daha büyük değildir. Bunun nedeni ise, tek bir sınıflandırıcının kullanılmasında genellikle model seçimi (model selection) veya parametre ayarlanması (parameter tuning) için sınıflandırıcının birden fazla türünün oluşturulmasının gerekmesidir. Bu durum, topluluklarda temel sınıflandırıcıların oluşturulması sırasında tüm durumların aynı anda karşılaştırılmasıyla gerçekleştirilmektedir. Bu yüzden, topluluklardaki sınıflandırıcıların birleştirilmesindeki hesaplama maliyeti, çoğu birleştirme stratejisinin kolay hesaplanmasından dolayı tek bir sınıflandırıcının maliyetinden genellikle daha düşüktür (J R Quinlan, 1996).

Topluluk yöntemlerini birbirinden farklılaştıran ana faktörler vardır (Sewell, 2007). Bunlar,

1. İç sınıflandırıcıların ilişkisi (inter-classifiers relationship): Bireysel sınıflandırıcılar birbirini nasıl etkilediği konusuyla ilgilenilmektedir. Buna bağlı olarak, topluluk yöntemleri sıralı (sequential) ve eş zamanlı (concurrent / parallel) olmak üzere iki ana yaklaşımda incelenmektedir.
2. Birleştirme yöntemleri (combination methods): Bir topluluğu oluşturan sınıflandırıcı modellerin birleştirilmesi konusudur.
3. Topluluk çeşitliliği (ensemble diversity): Topluluğun başarısını arttırmak için, sınıflandırıcılar arasında bir çeşitlilik olmalıdır. Çeşitlilik, sınıflandırıcıların seçilmesinde sınıflandırıcılar arasında farklılığı ve çeşitliliği sağlamak amacıyla çıktılara bir ceza yani hata ekleyerek elde edilebilmektedir.
4. Topluluk genişliği (ensemble size): Topluluktaki sınıflandırıcıların sayısıdır.

3.8.1. İç Sınıflandırıcıların İlişkisi

Bireysel sınıflandırıcıların topluluk içerisinde birbirleriyle ilgili olan ilişkilerini sıralı ve eş zamanlı olmak üzere iki farklı yaklaşımla incelenmektedir.

3.8.1.1. Sıralı Yaklaşımlar

Sıralı yaklaşımlar, öğrenme topluluklarında öğrenme sıraları arasında bir etkileşim olduğunu varsaymaktadır. Bu yaklaşımla, sonraki tekrarlarda yani yinelemelerde öğrenmeyi ilerletmek

için önceki tekrarlarda elde edilen bilgiden yararlanılması sağlanmaktadır. Sıralı yaklaşım kendi içinde model güdümlü örnek seçimi (model-guided instance selection) ve artımlı toplu öğrenme (incremental batch learning) olmak üzere iki temel yaklaşıma ayrılmaktadır (F. J. Provost & Kolluri, 1997).

Model güdümlü örnek seçimi yaklaşımında, önceki yinelemelerde oluşturulan sınıflandırıcılar, sonraki yinelemeye yönelik eğitim setini değiştirmek için kullanılmaktadır. Bu süreç, temel öğrenme algoritmasına dahil edebilmektedir. Dolaylı (constructive) veya tutucu (conservative) yöntemler olarak da bilinen bu yöntemler, genellikle ilk sınıflandırıcılarının doğru olduğu tüm veri örneklerini göz ardı etmekte ve yalnızca yanlış sınıflandırılmış örneklerden öğrenmektedir. Model güdümlü örnek seçimi yaklaşımı da kendi içinde belirsizlik örneklemesi (uncertainty sampling), artırma ve pencereleme (windowing) olmak üzere farklı yaklaşımlar barındırmaktadır (Maimon & Rokach, 2005).

3.8.1.2. Artırma

Uyarlanabilir yeniden örnekleme ve birleştirme (adaptive resampling and combining) olarak da bilinen artırma algoritması, bir öğrenme algoritmasının performansını iyileştirmek için genel bir yöntemdir (Opitz & Maclin, 1999). Bu yöntem, çeşitli dağılımlara sahip eğitim verilerinde zayıf sınıflandırıcıların sonuçlarını tekrarlı olarak toplayarak güçlü bir sınıflandırıcı oluşturur. Her yeni zayıf sınıflandırıcı bir sonuç elde ettiğinde veri yeniden ağırlıklandırılır. Böylelikle, toplama işlemi devam ederken, daha önceden oluşturulmuş zayıf sınıflandırıcılardan eğitim veri setindeki gözlemler için yanlış sınıflandırma yapanlara daha fazla odaklanılarak güçlü bir sınıflandırıcı elde edilir. Daha sonra, iyileştirilen zayıf sınıflandırıcılar bir araya getirilerek daha güçlü sınıflandırıcı haline gelmektedirler.

Artırma algoritması, birden çok sayıda kurulan karar ağacının sonuçlarını kullanarak ağırlıklı oylama (weighted voting) yöntemiyle final sınıf tahminini yapmaktadır (S. Kotsiantis, 2011). Bu algoritmada, bir sonraki kurulan karar ağacı ile bir önceki karar ağacına bağımlıdır. Daha önce kurulan karar ağaçlarından yanlış sınıf tahmininde bulunan tahmin edicilere daha fazla ağırlık verilerek ard arda yineleme ile yeni karar ağaçları kurulmaktadır. Son olarak, final tahmini için ağırlıklı oylama yapılarak gözleme ait sınıf tahminlenmektedir.

Schapire 1990 yılında ilk artırma algoritmasını geliştirmiştir. Daha sonra 1995 yılında ise, Freund ve Schapire birlikte AdaBoost algoritmasını geliştirmişlerdir (Freund & Schapire, 1997; Schapire, 1990). Bu algoritmanın temel mantığında, eğitim veri setindeki her bir gözlem için bir ağırlık belirlenmektedir. Başlangıçta, tüm ağırlıklar eşitken her yinelemede,

yanlış sınıflandırılmış tüm gözlemlerin ağırlıkları güncellenerek arttırılmakta, doğru sınıflandırılmış gözlemlerin ağırlıkları ise azaltmaktadır. Bunun nedeni ise, zayıf sınıflandırıcıların doğru sınıflandırma ile çok az ilişkili sınıflandırıcılar iken; güçlü sınıflandırıcıların doğru sınıflandırma ile çok fazla ilişkili olmalarıdır. Bunun sonucunda, zayıf sınıflandırıcılar ile güçlü sınıflandırıcıların ağırlıkları dengelenmektedir. Böylelikle, zayıf sınıflandırıcılar eğitim veri seti içerisinde güçlendirilerek güçlü sınıflandırıcılara dönüştürülmektedir. Bu süreç, birbirini tamamlayan bir dizi ardışık güçlü sınıflandırıcı oluşturmaktadır.

Tüm artırma yaklaşımlarının temeli AdaBoost algoritmasına dayandığı için kısaca adaboost algoritmasından da bahsedilecektir. Bu algoritma, ikili sınıflandırmada (binary classification) için geçerli bir algoritmadır. AdaBoost algoritması, eğitim veri setinin ikili sınıflandırılmış (-1, +1) olarak etiketlenmiş m gözlemden oluştuğunu varsaymaktadır. Yeni bir gözlemin sınıflandırması için her biri α_t ağırlığına sahip olan sınıflandırıcılara $\{C_t\}$ oy verilerek yapılır:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t * C_t(x)\right) \tag{24}$$

Freund ve Schapire (1996), ikili sınıflandırmaya eşdeğer olan ve çoklu sınıflandırma problemlerinin ele alınması için AdaBoost algoritmasının AdaBoost.Ml ve AdaBoost.M2 olmak üzere iki farklı versiyonunu geliştirmişlerdir. Çoklu sınıflandırma durumunda, yeni bir gözlemin sınıflandırması aşağıdaki gibidir:

$$H(x) = \underset{y \in dom(y)}{argmax}\left(\sum_{t:C_t(x)=y}^{T} \log\frac{1}{\beta_t}\right) \tag{25}$$

Tüm bu bahsedilen artırma algoritmaları, zayıf uyarıcıların ağırlıklı gözlemlerle birlikte sorunun çözülebileceğini varsaymaktadır. Aksi halde, ağırlıklı verilerden yeniden ağırlıklandırma yöntemi ile ağırlıklandırılmamış bir veri seti üretilmektedir. Böylelikle, gözlemler ağırlıklarına göre olasılıkla seçilmekte ve bu veri seti, orijinal eğitim veri seti genişliğine ulaşıncaya kadar bu seçim devam etmektedir. Artırma, iki ana nedenden dolayı sınıflandırma algoritmalarının performanslarının başarılarının arttırmaktadır (Freund & Schapire, 1996):

1. Eğitim veri seti hatasını azaltacak büyük hatalara sahip zayıf sınıflandırıcıların hipotezi birleştirerek daha küçük hatalı bir sınıflandırıcı oluşturmak.

2. Varyansı büyük olan zayıf sınıflandırıcılardan daha düşük varyanslı bir sınıflandırıcı elde etmek.

Diğer yandan, artırma algoritması bazen genelleme performansında azalmalara neden olabilmektedir. Quinlan'a (1996) göre, artırma algoritmasının başarısızlığının temel nedeni aşırı uyum problemidir. Artırma algoritmasının amacı, veriler üzerinde iyi performans gösteren bir bileşik sınıflandırıcı oluşturmaktır, ancak yineleme sayısının artmasına bağlı olarak, bireysel sınıflandırıcılardan elde edilen doğruluk önemli ölçüde azalacağından, bileşik sınıflandırıcının başarısının azalmasına yol açacaktır. Bu nedenle, aşırı uyum pobleminden kaçınmanın olası bir yolu, yineleme sayısını olabildiğince küçük tutmaktır (J R Quinlan, 1996).

Sonuç olarak, hem pencereleme hem de belirsizlik örneklemesi, yalnızca son bir örneklem elde etmek için bir sınıflandırıcı dizisi oluşturmaktadır. Yöntemler arasındaki fark, pencerelemede, gözlemlerin önceden etiketlenmiş (labeled) olması; belirsizlik durumunda ise, gözlemlerin önceden etiketlenmemiş (unlabeled) olmasıdır. Bu nedenle, yeni eğitim veri setindeki gözlemler birbirinden farklı seçilmiştir. Artırma yaklaşımı ise, bir sınıflandırıcı dizisi oluşturmakta, ancak hepsinden bilgi edinmek için bunları birleştirmektedir. Pencereleme ve belirsizlik örneklemesi sınıflandırıcıları birleştiren en iyi sınıflandırıcıyı kullanmaktadır (Maimon & Rokach, 2005).

Artırma topluluk öğrenme yaklaşımında ise, bir yinelemeden elde edilen sınıflandırıcı, takip eden yinelemede (bu yinelemenin alt örneklemiyle birlikte) öğrenme algoritmasına "ön bilgi (prior knowledge)" olarak verilmektedir. Öğrenme algoritması, önceki sınıflandırıcıyı değerlendirmek için mevcut alt örneklemeyi kullanmakta ve bir sonraki sınıflandırıcıyı oluşturmak için önceki sınıflandırıcıdan yararlanmaktadır. Son yinelemede oluşturulan sınıflandırıcı, son sınıflandırıcı (final classifier) olarak seçilmektedir (Maimon & Rokach, 2005).

3.8.1.3. Eş zamanlı Yaklaşımlar

Paralel yaklaşımlar olarak da bilinen eş zamanlı topluluk yaklaşımında, orijinal veri seti, birden fazla sınıflandırıcıyla (multiple classifiers) eş zamanlı olarak uyarılarak birkaç alt gruba ayrılmaktadır. Orijinal eğitim veri setinden oluşturulan alt kümeler birbirinden ayrık (mutually exclusive) veya örtüşen (overlapping) olabilmektedirler. Belirli bir örnek için tek bir sınıflandırma sonucu elde etmek için bir birleştirme yöntemi uygulanmaktadır. Uyarlanmış sınıflandırıcıların sonuçlarını birleştirme yöntemleri genellikle uyarlama algoritmalarından

(induction algorithms) bağımsız olduğundan, her alt kümede farklı uyarıcılarla birlikte kullanılabilmektedir. Bu eş zamanlı yöntemler, sınıflandırıcıların tahminleme gücünü (predictive power of classifiers) artırmayı veya toplam uygulama süresini (total execution time) azaltmayı amaçlamaktadır. Eş zamanlı yaklaşım, kendi içinde farklı yaklaşımlar barındırmaktadır (Breiman, 1996). Bu yaklaşımlar, torbalama ve çapraz doğrulanmış topluluklar (cross-validated committees) yaklaşımlarıdır.

3.8.1.4. Torbalama

Torbalama algoritması sınıflandırmada, doğru sınıflandırma oranını artıran ve model varyansını azaltan bir algoritmadır (Witten et al., 2011). Bunun yanında, hem aşırı uyuma hem de kayıp gözleme karşı duyarlıdır. Torbalama algoritması bir çok sınıflandırma algoritması modeli kurabilmesine rağmen, daha çok karar ağaçları algoritmasını kullanmaktadır. Bu algoritma, bootstrap örnekleme yöntemini kullanarak eğitim veri setinden rastgele alt örneklemler elde ederek karar ağaçları modelleri kurmaktadır. Bu karar ağaçları, eş zamanlı olarak gözlemler için sınıf tahminlerini oylamaktadır. Burada kullanılan oylama yöntemi, çoğunluk oylama (majority voting) yöntemidir (Zhou, 2012). Bu oylama sonucu en çok oyu alan sınıfın final sınıf tahmini belirlenerek öğrenme tamamlanmaktadır. Torbalama yaklaşımında, eş zamanlı diğer bir ifadeyle aynı anda kurulan karar ağaçları birbirine bağlı değildirler. Bunun nedeni ise, bu karar ağaçlarının eğitim veri setinden bootstrap örnekleme yöntemiyle birbirinden bağımsız olarak alt örneklemlerinin oluşturulmasıdır. Bu yaklaşımda, alt örneklem seçimi için yapılan her yinelemede tüm gözlemler eğitim veri setinden eşit olasılıkla seçilir. Yeniden örneklemeden sonra kurulan sınıflandırıcıların performansları birbirinden farklıdır. Bootstrap yöntemiyle elde edilen alt örneklemlere ait karar ağaçları birbirinden farklı olmadıkları durumda, karar ağaçları benzer hatalar yapma eğilimi göstereceklerdir. Alt örneklemler için eğitim veri setinin yaklaşık %63,2'i kadarı kullanarak bootstrap yeniden örnekleme ile elde edilen gözlemlerden oluşmaktadır (Breiman, 1996).

$$\left(1-\frac{1}{n}\right)^{n} \approx e^{-1} = 0{,}368 \qquad (26)$$

Burada n, gözlem sayısını ifade etmektedir. n sonsuza giderken doğal logaritma tabanında $1-e^{-1} = 1-0{,}368 = 0{,}632$ olur.

Bu yaklaşım sayesinde, sınıflandırıcılarına ait farklı sonuçlar tek bir tahmin değeri olarak birleştirerek, genellendirilmiş bir bileşik sınıflandırıcı oluşturarak sınıflandırma doğruluğunu

arttırmayı amaçlamaktadır (Breiman, 1996). Genellikle, yeniden örnekleme yöntemiyle elde edilen her alt örneklem genişliği (subsample size), eğitim veri setinin genişliğine eşittir.

Alt örneklem eğitim veri setlerindeki gözlemlerin bazılarının alt örneklemlerde bir defadan fazla yer alabileceği gibi, bazılarının hiç dahil edilmeyebileceği unutulmamalıdır. Bu yüzden, alt örneklem eğitim verileri birbirinden farklı, ancak kesinlikle bağımsız değildirler. Yeni bir gözlemi sınıflandırmak için her sınıflandırıcı, bilinmeyen gözlem için sınıf tahmini döndürmektedir. Bileşik torbalanmış sınıflandırıcı en sık tahmin edilen sınıfı tahminlemektedir (örneğin, çoğunluk oylama yöntemi kullanır). Böylelikle torbalama yaklaşımı, çoğu zaman orijinal tek verilerden oluşturulan zayıf modellerden daha iyi performans gösteren topluluk bir model oluşturmaktadır. Breiman (1996), bunun özellikle kararsız uyarıcılar (unstable inducers) adı verilen sınıflandırma modelleri için geçerli olduğunu belirtmiştir. Bunun nedeni ise, torbalama yaklaşımının kararsızlığı ortadan kaldırabileceğini savunmasıdır. Bu anlamda, öğrenilmiş veri setini bozmak eğer kurulan sınıflandırıcıda önemli değişikliklere neden olabilirse, uyarıcı kararsız olarak kabul edilmektedir (Breiman, 1996). Bununla birlikte, torbalama yaklaşımının analiz edilmesi oldukça zor olduğundan, geliştirilmiş kararlar için faktörlerin ve nedenlerin anlaşılması kolay değildir.

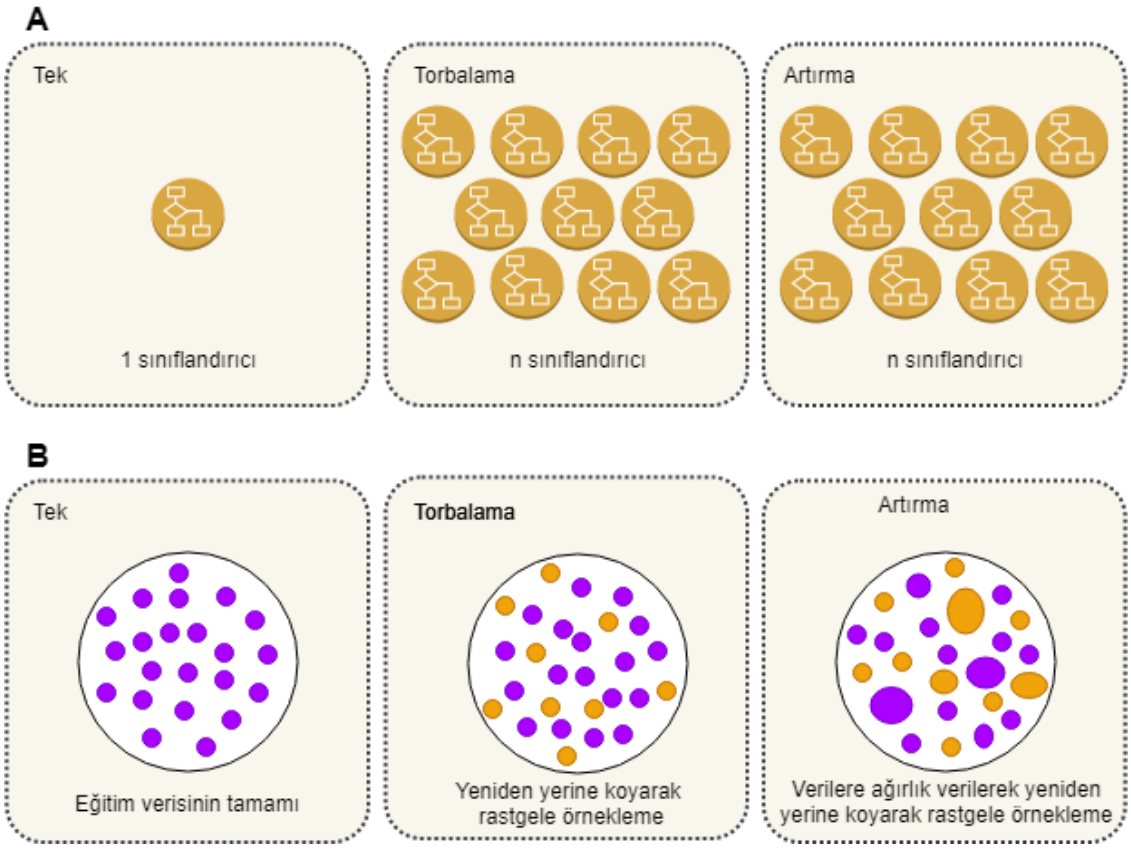
Torbalama yaklaşımı, artırma yaklaşımı gibi, farklı sınıflandırıcılar üreterek ve birden fazla modeli birleştirerek bir sınıflandırıcının doğruluğunu arttırmaya yönelik bir yöntemdir. Her ikisi de aynı tipteki farklı sınıflandırıcıların çıktılarını birleştirmek için sınıflandırmada bir tür oy kullanmaktadır. Artırma yaklaşımında, torbalamanın aksine, her bir sınıflandırıcı daha önceki performanslardan etkilenmektedir. Bu nedenle yeni sınıflandırıcı, öncekilerde yapılan hatalara ve performanslara daha fazla dikkat etmeye çalışmaktadır. Torbalamada, her bir örnek eşit olasılıkla seçilirken, artırma yaklaşımında örnekler ağırlıklarına göre orantılı olarak seçilmektedir. Ayrıca, yukarıda da belirtildiği gibi Quinlan'a (1996) göre torbalama yaklaşımı, hata oranlarının 0,5'in altında tutulması şartıyla, artırma yaklaşımının kararsız öğrenme sistemlerinin kullanımını engellemediği durumlarda öğrenme sisteminin kararlı olmamasına gerek duymaktadır (J R Quinlan, 1996).

3.8.1.5. Torbalama ve Artırma Yöntemlerinin Karşılaştırılması

Topluluk öğrenme algoritmaları için yapılan birçok çalışma genel olarak artırma algoritmalarının torbalama algoritmalarından daha başarılı sonuçlar verdiğini göstermektedir (Brown, Wyatt, & Tino, 2005; Kumar, Kongara, & Ramachandra, 2013; Z. Zhang, Zhao,

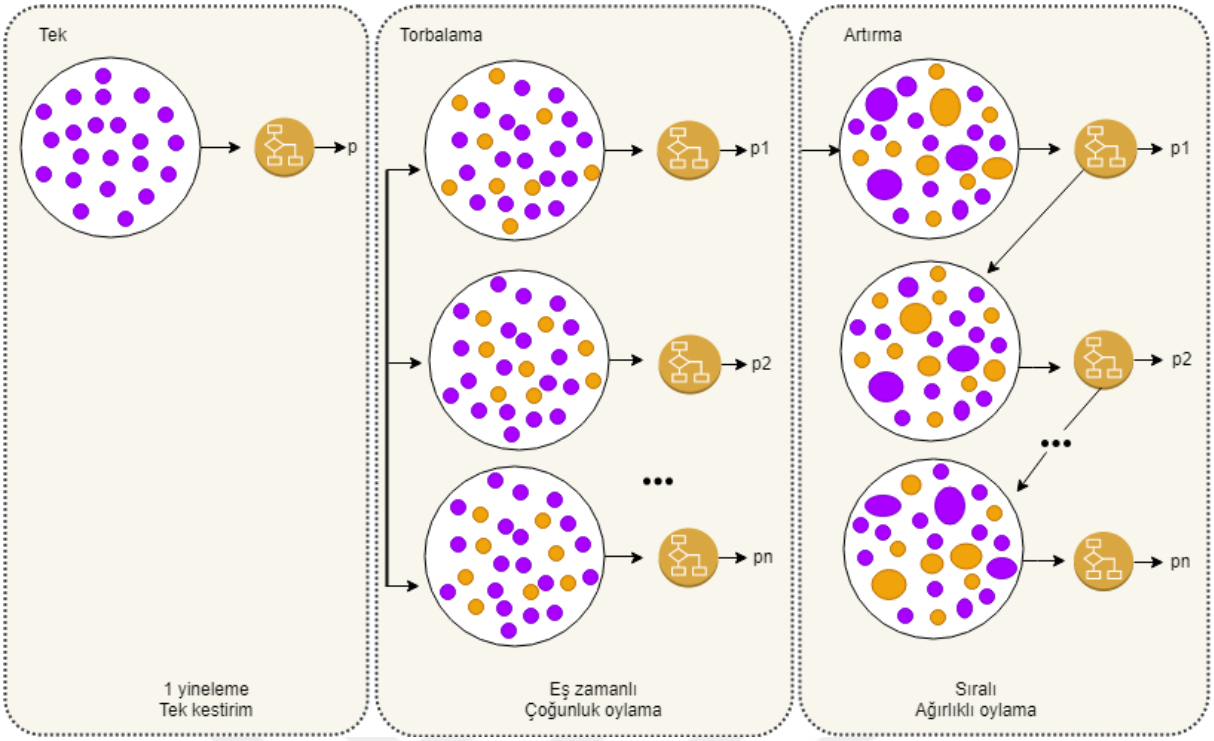
Canes, Steinberg, & Lyashevska, 2019). Bu sonuçlar, öğrenmedeki bazı hatalardan etkilenmektedir. Bu hataların ana nedenleri arasında gürültü, yanlılık ve sapmalar yer almaktadır. Topluluk öğrenme, bu faktörleri en aza indirmek ve makine öğrenme algoritmalarının kararlılığını ve doğruluğunu artırmak için tasarlanmıştır. Birden fazla sınıflandırıcının birleştirilmesi, özellikle kararsız sınıflandırıcılar varlığında, tek bir sınıflandırıcıdan daha güvenilir bir sınıflandırma modeli kurabilmektedir (Dietterich, 2000b).

Tek sınıflandırıcılar tüm eğitim veri setini kullanırken, torbalama ve artırma, eğitim veri setinden N tane sınıflandırıcı için N yeni eğitim veri setini, orijinal setten yeniden yerine koyarak rastgele örneklemeyle üretir (Zhou, 2012). Yeniden yerine koyarak örnekleme ile elde edilen her yeni eğitim veri setinde bazı gözlemler tekrarlanabilmektedir. Torbalama durumunda, herhangi bir gözlem, yeni bir eğitim veri setinde aynı görülme olasılığına sahiptir. Bununla birlikte, artırma durumunda gözlemler ağırlıklandırıldığından, bazıları yeni eğitim veri setlerinde daha sık yer almaktadır. Bu çoklu veri setleri, aynı öğrenme algoritmasını eğitmek için kullanılarak farklı sınıflandırıcılar elde edilmektedir (Şekil 11a).



Şekil 11: Topluluk Öğrenme Algoritmalarının Karşılaştırılması

Eğitim veri seti içinde bulunan gözlemlerin ağırlıklandırılması durumu iki yöntem arasındaki temel fark Şekil 11b'de gösterilmektedir. Eğitim aşamasında yeni sınıflandırıcı torbalama yönteminde paralel olarak eğitilmesine rağmen, artırma yönteminde sıralı bir şekilde eğitilerek oluşturulmaktadır. Buna göre, torbalama yöntemiyle eğitilen sınıflandırıcıların her bir modeli, birbirinden bağımsız olarak kurulmaktadır. Artırma yönteminin sıralı öğrenmesinden dolayı, her bir sınıflandırıcı, önceki sınıflandırıcıların başarısı dikkate alınarak veriler üzerinde eğitilmektedir. Her eğitim aşamasından sonra, ağırlıklar yeniden dağıtılarak güncellenmektedir. Yanlış sınıflandırılmış veriler, en zor durumlarla baş edebilmesi için ağırlıklarını artırmaktadır. Bu şekilde, sonraki sınıflandırıcılar eğitimleri sırasında yanlış sınıflandırılmış verilere odaklanmaktadır. Sınıflandırma aşamasında, yeni veri sınıfını tahmin etmek için sadece N öğrencisini yeni gözlemlere uygulanması gerekmektedir. Torbalama yönteminde sonuç, N tane öğrenenin verdiği yanıtların çoğunluk oyları alınarak elde edilmektedir. Bununla birlikte, artırma yönteminde ise, N tane sınıflandırıcılarının tahminlerinin ağırlıklı oylamasını almak için ikinci bir ağırlık seti atanmaktadır. Artırma yönteminin eğitimi aşamasında, her modele ağırlık atanmaktadır (S. B. Kotsiantis & Pintelas, 2004; Maimon & Rokach, 2005). Eğitim veri setlerinde yapılan sınıflandırma sonucu iyi olan güçlü bir sınıflandırıcıya, zayıf olan sınıflandırıcıdan daha yüksek bir ağırlık verilmektedir. Bu nedenle, yeni bir sınıflandırıcı modeli kurulurken, artırma yöntemi torbalama yöntemi gibi sınıflandırıcıların hatalarını göz önünde bulundurmak zorundadır. Diğer yandan, artırma yöntemler topluluk içerisindeki zayıf sınıflandırıcıları güçlü sınıflandırıcılara dönüştürerek zayıf olanların olasılıksal tahminlerinin güçlendirilmesini sağlamaktadır (Şekil 12).



Şekil 12: Topluluk Öğrenme Algoritmalarının Sonuçlarının Birleştirilmesi

Her iki yöntemin farklılıkları ise, bazı artırma yöntemleri tek bir öğrenciyi saklamak veya silmek için ekstra bir koşul içermektedir. Örneğin, AdaBoost algoritması ile model kurmak için %50'den az bir hata gerekmektedir. Aksi durumda, yineleme işlemi rastgele bir tahminden daha iyi bir sınıflandırıcı başarısı elde edinceye kadar tekrarlanmaktadır. Buna göre, artırma yönteminde bir sonraki eğitim ve sınıflandırma aşamasında kullanılacak ağırlıkları belirlemek için farklı artırma yöntemler içeren çeşitli alternatifler vardır (Freund & Schapire, 1997).

Torbalama ve artırma yöntemi, tekli sınıflandırıcı modelden elde edilen tahminin varyansını, farklı modellerden gelen tahminleri birleştirdikleri için azaltmaktadır. Dolayısıyla final modeli daha yüksek kararlılığa sahip bir model olabilmektedir. Tek bir modelin çok düşük bir performans göstermesi durumu söz konusu ise, torbalama nadirde olsa yanlılığa sahip olmaktadır. Bununla birlikte, artırma, tek bir modelin performansını optimize ettiği ve hatalarını azalttığı için daha düşük hatalarla birleştirilmiş bir final modeli kurabilmektedir. Bunların yanı sıra, tekli sınıflandırıcı model aşırı uyuma sahipse, bu durumda torbalama en iyi yöntem olarak öne çıkabilmektedir. Artırma yönteminin aşırı uyum durumunda kullanılması önerilmemektedir. Aslında, artırma yöntemi aşırı uyum probleminin kendisiyle karşı

karşıyadır diğer bir ifadeyle artırma algoritması en büyük sorunu aşırı uyum problemidir. Bu nedenle, bazı durumlarda torbalama yöntemi, artırma yönteminden daha etkili olabilmektedir (Dietterich, 2000a).

Sonuç olarak, torbalama ve artırma yöntemlerinin karşılaştırılmasında hangi yöntemin daha iyi olduğu ile ilgili çalışmalar ele alındığında kesin bir kazananı yoktur. Bu durum veriye, benzetimlere ve koşullara bağlıdır (Dietterich, 2000b).

3.8.2. Birleştirme Yöntemleri

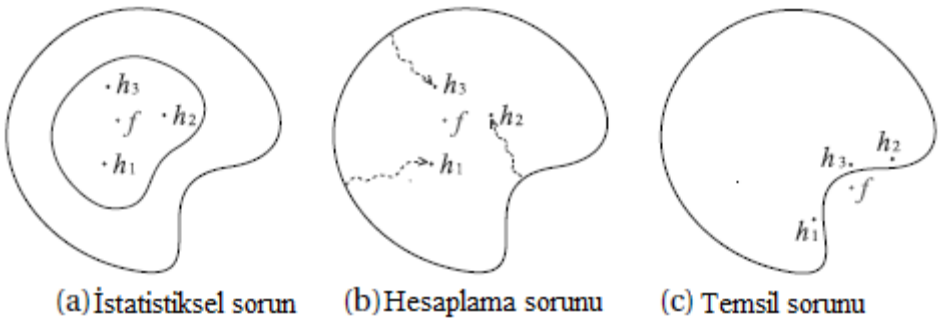
Sınıflandırma algoritmaları ile bir bireyin veya nesnenin hangi sınıfa dâhil olacağı tahminlenebilmektedir. Birçok çalışmada, sınıflandırma algoritması yöntemleri arasından probleme uygun olanı seçilerek gerekli optimizasyonlar yapılmaktadır ve en yüksek doğruluk oranları elde edilmeye çalışılmaktadır. Bir sınıflandırma problemi için, birden çok sınıflandırıcının eğitilmesi veya aynı sınıflandırıcıyı aynı eğitim setinin farklı alt kümeleri ile eğitmek gerekmektedir. Sonrasında hangi sınıflandırma algoritmasının en yüksek sınıflandırma başarısına sahip olduğu çalışılabilmektedir. Buna imkân sağlayan topluluk öğrenme yöntemleri (sınıflandırıcı topluluklar) sınıflandırma problemi için birden fazla sınıflandırıcıyı kullanmaktadırlar. Bunun sonucunda temel mantık olarak farklı doğruluk skorlarına sahip sınıflandırıcıların bu sonuçlarını farklı yöntemlerle (oylama, ortalama vb.) birleştirilmesi (combination methods) mantığına dayanmaktadır. Böylelikle tek bir sınıflandırıcıdan daha iyi sonuçlar elde etme imkanı sağlamaktadır. Sınıflandırıcı topluluklar sadece sınıflandırma doğruluğu başarısını yükseltmesinin yanı sıra, eğitim veri setinin ezberini bozmaktadır. Sınıflandırma modelinin yanlılığını ve varyans hatasını da azaltmaktadır. Buradaki önemli noktalardan biri de, topluluk öğrenme yöntemi kullanıldığında elde edilen sonucun, tek bir sınıflandırıcıdan elde edilen sonuçtan daha iyi olmalıdır, aksi halde topluluk öğrenme yöntemi kullanmanın hiçbir anlamı kalmamaktadır. Aynı zamanda, topluluk öğrenme yöntemini tek bir sınıflandırıcıdan farklı kılan diğer bir durum, hesaplama maliyetinin daha düşük olmasıdır.

Bir problemin çözümünde, birden fazla temel sınıflandırıcı ile modeller kurulduktan sonra en iyi model başarısına sahip algoritmayı bulmak yerine topluluk öğrenme yöntemlerinin birleştirme yöntemleri özelliği sayesinde daha dayanıklı bir genelleme yapılmaktadır. Birleştirme yöntemleri, istatistiksel sorunun (statistical issue), hesaplama sorununun (computational issue) ve temsil sorununun (representational issue) ortadan kaldırılması için kullanılmaktadır.

İstatistiksel sorun, hipotez alanının, sınırlı eğitim verilerini eğitmek için çok büyük olduğu ve eğitim verilerinde aynı doğruluk oranını veren birkaç farklı hipotez olabileceği durumudur. Öğrenme algoritması bu hipotezlerden birini seçerse, yanlışlıkla seçilen hipotezin gelecekteki verileri doğru bir şekilde tahminleyememe riski ortaya çıkmaktadır. Şekil 13a'da gösterildiği gibi, hipotezleri birleştirerek, yanlış bir hipotez seçme riski azaltılabilir.

Hesaplama sorununda, birçok öğrenme algoritması, yerel en idealde saplanıp kalabilecek bir tür yerel arama gerçekleştirir. Yeterli eğitim verisi olsa bile, en iyi hipotezi bulmak çok zor olabilir. Bu sorunun üstesinden gelmek için birleştirme yöntemleri, yerel aramayı birçok farklı başlangıç noktasından yaparak bilinmeyen gerçek hipotezi seçebilmek için daha iyi bir yaklaşım sağlayabilmektedir. Şekil 13b'de gösterildiği gibi, hipotezleri birleştirerek, yanlış bir yerel minimum seçilme riski azaltılabilir.

Temsil sorunu, birçok makine öğrenme algoritmasında gerçek bilinmeyen hipotez, hipotez alanındaki hiçbir hipotez ile temsil edilememiş olabilmektedir. Bu durumda, Şekil 13c'de gösterildiği gibi, hipotezleri birleştirerek, temsil edilebilir fonksiyonların alanını genişletmek mümkün olabilir ve böylece öğrenme algoritması, gerçek bilinmeyen hipoteze daha doğru bir yaklaşım oluşturabilmektedir.



Şekil 13: Birleştirmenin Üç Temel Nedeni (Dietterich, 2000b)

Şekil 13'te birleştirmenin üç temel nedeni: (a) istatistiksel sorun, (b) hesaplamalı sorun ve (c) temsil sorunu göstermektedir. Şekil 13'te dış eğri (outer curve), hipotez uzayını temsil eder ve Şekil 13a'daki iç eğri (inner curve), eğitim verilerinde aynı doğrulukta hipotezleri temsil eder. f noktası etiketi, gerçek hipotezdir (true hypothesis) ve h_i'ler bireysel hipotezleri (individual hypotheses) ifade etmektedir. Yukarıda bahsedilen her üç sorun, geleneksel öğrenme

yaklaşımlarının başarısız olmasının en önemli nedenlerindendir. İstatistiksel sorundan kaynaklı bir öğrenme algoritmasının genellikle yüksek bir varyansa sahip olduğu belirtilmektedir. Aynı zamanda, hesaplama sorunundan kaynaklı bir öğrenme algoritmasının da yüksek bir hesaplama varyansına sahip olduğu ifade edilemektedir. Temsil sorundan kaynaklı bir öğrenme algoritmasının ise, genellikle yüksek bir yanlılığa sahip olduğu söylenebilmektedir (Dietterich, 2000b). Bu nedenle, birleştirme yoluyla, öğrenme algoritmalarının yanlılığının yanı sıra varyans da azaltılabilmektedir. Bu durum, birçok deneysel çalışma ile doğrulanmıştır (Opitz & Maclin, 1999; F. Provost & Kohavi, 1998).

Sınıflandırıcıları birleştirmesi temel olarak iki şekilde yapılmaktadır. Bunlar, basit çoklu sınıflandırıcı birleştirilmesi (simple multiple classifier combinations) ve meta birleştiricilerdir (meta-combiners). Basit birleştirme yöntemleri, bireysel sınıflandırıcıların aynı problemin (regresyon veya sınıflandırma) çözümünü sağlayan en uygun yöntemler olarak tanımlanmaktadır. Eğer regresyon problemi ile ilgileniliyorsa yani bağımlı değişken sürekli ise ortalama (averaging) birleştirme yöntemleri kullanılmaktadır. Literatürdeki en popüler ortalama yöntemleri, basit ortalama (basic averaging) ve ağırlıklı ortalama (weighted averaging) yöntemleridir. Eğer sınıflandırma problemi ile ilgileniliyorsa yani bağımlı değişken kategorik ise oylama birleştirme yöntemleri kullanılır. Literatürdeki en popüler oylama yöntemleri, çoğunluk oylama, çoğulluk oylama (plurality voting), ağırlıklı oylama ve şeffaf oylama (soft voting) yöntemleridir. Basit birleştirme yöntemleri, aykırı değerlere ve aynı şekilde performans gösteren sınıflandırıcılara karşı dayanıklı değillerdir. Diğer yandan, meta birleştiriciler teorik olarak daha güçlüdürler ve ek öğrenmeyle ilgili tüm problemlere karşı duyarlılardır (örneğin; aşırı uyum, uzun eğitim süresi). Meta birleştirme yaklaşımları, istifleme (stacking), belirleyici ağaçlar (arbiter trees), birleştirici ağaçlar (combiner trees) ve derecelendirme (grading) yaklaşımlarıdır. Meta birleştiricilerin en önemli yöntemlerinden istifleme yöntemi, genellikle en yüksek doğruluk genellemesini sağlamaktadır (Maimon & Rokach, 2005).

Oylama sürecinde, T tane bireysel sınıflandırıcı (individual classifiers) olduğunu varsayalım.

- Bireyler sınıflandırıcılar, $\{h_1, \dots, h_T\}$ ile ifade edilmektedir.
- h_i'lerin birleştirilmesi için, I tane olası sınıf etiketinden $\{c_1, \dots, c_I\}$ sınıf etiketi tahminlenmektedir.

- Genel olarak, bir x örneği için, h_i sınıflandırıcısının sonuçlarının, $h_i^j(x)$'nin c_j sınıfı etiketi için h_i sonucuna sahip olduğu bir I-boyutlu etiket vektörü $\left(h_i^1(x), \ldots, h_i^I(x)\right)^T$ olarak verildiği varsayılmaktadır.
- $h_i^j(x)$, bireysel sınıflandırıcılar tarafından elde edilen bilgilere göre farklı tiplerde değerler alabilmektedir:
    - Kesin etiket (crisp label): h_i, sınıf etiketi olarak c_j tahminlerse $h_i^j(x) \in \{0,1\}$ 1 değerini alır aksi takdirde 0 değerini alır.
    - Sınıf olasılığı (class probability): $h_i^j(x) \in [0,1]$, sonsal olasılık $P(c_j|\, x)$ tahmini olarak kabul edilebilmektedir.

Bu sonuçların ait olasılıkların dönüşümü için ayarlama yöntemleri kullanılmaktadır. Çoğu sınıflandırıcı tarafından tahmin edilen olasılıklar zayıftır. Bununla birlikte, sınıf olasılıklarına dayalı birleştirme yöntemleri için ayarlamadan sonra, kesin etiket sonucu veren olasılıklar oldukça dayanıklıdır. Basit çoklu sınıflandırıcı birleştirilmesi yönteminden çoğunluk oylama yöntemi, sınıflandırma problemlerinde en çok tercih edilen yöntemdir (Zhou, 2012).

3.8.2.1. Çoğunluk Oylama

Çoğunluk oylama, en popüler oylama yöntemidir. Bu yöntemde, her sınıflandırıcı bir sınıf etiketine oy vermekte ve son çıkış sınıfı etiketi oyların yarısından fazlasını alan etikettir. Sınıf etiketlerinden hiçbiri oyların yarısından fazlasını alamazsa, bir reddetme seçeneği (rejection option) verilmekte ve birleştirilmiş sınıflandırıcı (combined classifier) tahmini yapılmamaktadır. Diğer bir ifadeyle, topluluğun çıkış sınıfı etiketi aşağıdaki gibi hesaplanabilmektedir,

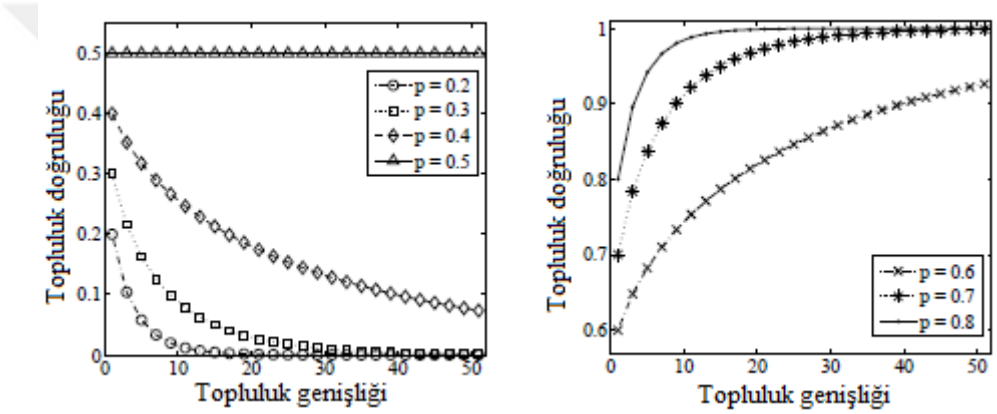
$$H(x) = \begin{matrix} c_j \\ red \end{matrix} \quad \begin{matrix} \sum_{i=1}^{T} h_i^j(x) > \frac{1}{2} \sum_{k=1}^{I} \sum_{i=1}^{T} h_i^k(x) \\ diğer\ durum \end{matrix} \tag{27}$$

İkili sınıflandırma problemi için toplam T sınıflandırıcısı varsa, en az $\lfloor T/2 + 1 \rfloor$ kadar sınıflandırıcı doğru sınıf etiketini seçerse, topluluk kararı doğru olmaktadır. Sınıflandırıcılara ait çıktılarının birbirinden bağımsız olduğunu ve her sınıflandırıcının bir doğruluk değeri olan p olasılığına sahip olduğu varsayılmaktadır. Bu durumda, her sınıflandırıcının p olasılıkla doğru bir sınıflandırma yaptığı anlamına gelmektedir. Topluluğun doğru bir karar verme

olasılığı, binom dağılımına bağlı olarak hesaplanabilmektedir. T sınıflandırıcı için, en az $\lfloor {}^{T}/_{2} + 1 \rfloor$ kadar doğru sınıflandırıcı olasılık değeri alması aşağıdaki gibi hesaplanmaktadır (Hansen & Salamon, 1990):

$$P_{mv} = \sum_{k = \lfloor {}^{T}/_{2}+1 \rfloor}^{T} \binom{T}{k} p^{k}(1-p)^{T-k} \tag{28}$$

İkili sınıflandırmada için topluluğun farklı p doğruluk değeri ile T bağımsız sınıflandırıcılarının çoğunluk oylamasının topluluk doğruluğunun alacağı değerler Şekil 14'te gösterilmektedir.



Şekil 14: İkili Sınıflandırmada p Doğruluk Değeri ile T Bağımsız Topluluk Sınıflandırıcının Çoğunluk Oylaması (Lam & Suen, 1997)

- Eğer $p > 0{,}5$ ise, P_{mv}, T'de mononotonik olarak artar ve

$$\lim_{T \to +\infty} P_{mv} = 1\,;$$

- Eğer $p < 0{,}5$ ise, P_{mv}, T'de mononotonik olarak azalır ve

$$\lim_{T \to +\infty} P_{mv} = 0\,;$$

- Eğer $p = 0{,}5$ ise, herhangi bir T için $P_{mv} = 0{,}5$ olur.

Bahsedilen bu sonuçlar, bireysel sınıflandırıcıların istatistiksel olarak bağımsız olduğu varsayımına dayanarak elde edilmektedir. Ancak uygulamada sınıflandırıcılar aynı sorun üzerinde eğitildiklerinden, genellikle yüksek oranda ilişkilidirler. Bu nedenle, çoğunluk

oylama doğruluğunun, bireysel sınıflandırıcı sayısının artmasıyla 1'e yaklaşması beklenmez (Lam & Suen, 1997).

3.8.2.2. Ağırlıklı Oylama

Çoğunluk oyu, topluluk öğrenme için en basit ve en etkili birleştirme yöntemleri arasında olmasına rağmen, oylamaya dayalı çoğunluk temelli topluluk yöntemleri üzerine yapılan çalışmalarda kullanımı giderek artmıştır. Bu çalışmalar, sınıflandırıcı topluluğu dayanıklılığının ve doğruluğunun, ağırlıklı oylama yöntemleri kullanılarak arttırılabileceğini göstermektedir (Zhou, 2012). Bazı araştırmalar arasında, ağırlıklı oylamaya dayalı topluluk sınıflandırma (weighted voting-based ensemble classification) yöntemlerinin kullanılarak sınıflandırma açısından daha genel sonuçlar elde edileceği savunulmuştur. Özellikle hastalık tanısı ve biyoinformatik alanında kullanılan topluluk öğrenme algoritmalarında, ağırlıklı oylama yöntemi çok fazla tercih edilmektedir (Breiman, Friedman, Olshen, & Stone, 1984).

En temel ağırlıklı oylamada, her bir sınıflandırıcı için ağırlık değerleri (w_k), eğitim seti ile belirli bir sınıflandırıcının tahminleme performansına dayanarak belirlenmektedir:

$$w_k = \frac{a_k}{\sum_l a_l} \tag{29}$$

a_k: eğitim setindeki k. sınıflandırıcısının tahmin doğruluğunu,

l: sınıflandırıcı topluluğundaki sınıflandırıcı sayısını,

a_l: l. sınıflandırıcının tahmini doğruluğunu, ifade etmektedir.

Ağırlıklı oylamada, N/M oranı veya daha az doğruluk performansı gösteren sınıflandırıcılara eğitim veri setleriyle sıfır ağırlığı verilir. Böylelikle, tahminleme performansı eşik değerden daha düşük olan sınıflandırıcılar, sınıflandırma topluluğundan çıkarılır. Ağırlık değerleri yeniden kullanılarak orantılı olarak ölçeklendirilir (Maimon & Rokach, 2005).

$$a_k = max\left\{0{,}1 - \frac{M.e_k}{N(M-1)}\right\} \tag{30}$$

e_k: belirli bir D_k sınıflandırıcısı tarafından elde edilen hata sayısını,

N: örneklem sayısını,

M: sınıf sayısını, belirtmektedir.

Sınıflandırıcı ağırlıkları, 0 ile 1 arasında değer almaktadır.

3.8.3. Topluluk Çeşitliği

Topluluk öğrenme temel konularından biri de topluluk çeşitliliği (ensemble diversity), yani bireysel sınıflandırıcılar arasındaki farklılık konusudur. Bireysel sınıflandırıcıların birleştirilmesi sırasında en yüksek peformans kazancını sağlamak için**,** bireysel öğrencilerin farklı olması gerekmektedir. Aksi durumda, aynı bireysel sınıflandırıcıların bir araya getirilmesiyle performans artışı olmamaktadır. Tumer ve Ghosh (1995), karar sınır analizini kullanarak, bireysel sınıflandırıcılar arasındaki genel korelasyonu tanımlamak için θ parametresine dayalı basit yumuşak oylama (simple soft voting) topluluğunun performansını analiz etmişlerdir (Tumer & Ghosh, 1995). Topluluğun beklenen ek hatasını (expected added error),

$$err_{add}^{ssv}(H) = \frac{1+\theta(T-1)}{T}\overline{err}_{add}(h) \tag{31}$$

$\overline{err}_{add}(h)$ bireysel sınıflandırıcıların beklenen hatası (Tüm bireysel sınıflandırıcıların eşit hataya sahip olduğu varsayılmaktadır.) ve T topluluk genişliğini ifade etmektedir. Bu formülde bulunan sınıflandırıcıların bağımsız olması durumunda, $\theta = 0$ olmakta ve topluluk, bireysel sınıflandırıcılardan T'nin hata azaltma faktörünü elde etmektedir. Sınıflandırıcılar tamamen korelasyon gösterirse, yani $\theta = 1$ ise, birleşmeden performans kazancı elde edilememektedir. Tumer ve Ghosh (1995), çeşitliliğin topluluk performansı için çok önemli olduğunu göstermektedir. Benzer sonuçlar diğer tüm birleştirme yöntemleri için de elde edilebilmektedir (Tumer & Ghosh, 1995).

Öte yandan, farklı bireysel sınıflandırıcılar bir topluluğun oluşturulması kolay değildir. Bu durumun başlıca nedeni, bireysel sınıflandırıcıların aynı sınıflandırma ya da regresyon problemi için aynı eğitim veri setlerinden eğitilmiş olmaları ve bu nedenle genellikle yüksek derecede korelasyonlu olmalarıdır (S. B. Kotsiantis & Pintelas, 2004). Teorik olarak olası pek çok yaklaşım, örneğin, en uygun ağırlıklı ortalama birleştirme yöntemini uygulamak pek işe yarar sonuçlar elde etmeyi sağlamamaktadır. Bunun nedeni ise, bireysel sınıflandırıcıların birbirinden bağımsız ya da daha az ilişkili olma varsayımlarına dayanmasıdır (Tumer & Ghosh, 1995). Topluluğun beklenen ek hatasının hesaplama formülündeki bireysel öğrenciler arasındaki yüksek korelasyonu göz önüne alsa bile, bireysel öğrencilerin sonsal olasılıklarla ilgili bağımsız tahminler ürettiği varsayımına dayansa da uygulamada bu durum böyle değildir. Aslında, çeşitli bireysel sınıflandırıcıların üretilmesi sorununda, bireysel öğrencilerin çok zayıf olmaması gerektiğine, aksi takdirde birleştirmelerinin iyileşmeyeceğine ve

performansın daha da düşeceğine inanılmaktadır (Brown, Wyatt, & Tino, 2005). Bireysel sınıflandırıcıların performansının oldukça zayıf olduğu durumlarda, basit yumuşak oylanmış topluluğun ek hatasının büyük olabilecektir. Bu nedenle, bireysel sınıflandırıcıların doğru (accurate) ve çeşitli (diverse) olması istenmektedir. Tamamlayıcılık saf doğruluktan daha önemli olduğundan, yalnızca doğru sınıflandırıcıları birleştirmek çoğu zaman bazı zayıf sınıflandırıcıları bir araya getirmekten daha kötü olabilmektedir (Maimon & Rokach, 2005). Sonuç olarak, topluluk içi öğrenmenin başarısı, bireysel performans ile çeşitlilik arasındaki iyi bir denge ile sağlanmaktadır. Ne yazık ki, çeşitlilik çok önemli olmasına rağmen, hala net bir çeşitlilik anlayışına sahip olunmamakta, şimdiye kadar kabul görmüş kesin bir çeşitlilik tanımı bulunmamaktadır (Zhou, 2012).

Çeşitliliğin ölçülmesiyle ilgili olarak, regresyon problemleri için varyans genellikle çeşitliliği ölçmek için kullanılmaktadır (Krogh & Vedelsby, 1994). Regresyon problemi için, tek bir modelin ortalama hatası korunurken, topluluk hatasının çeşitliliğin artmasına bağlı olarak modelin ortalama hatası azaltılabilmektedir. Sınıflandırma problemlerinde ise, çeşitliliği ölçmek daha karmaşık bir işlemdir. Kuncheva ve diğerleri (2003) yaptıkları çalışmalarda birçok çeşitlilik ölçümünü karşılaştırmışlar ve çoğunun birbirileriyle ilişkili olduğu sonucuna varmışlardır (Kuncheva, Whitaker, Shipp, & Duin, 2003). Aynı zamanda, artan çeşitliliğe bağlı olarak topluluk hatasının (ensemble error) azaltabileceği varsayılmaktadır (Cunningham & Carney, 2000).

Topluluk hatası, hataların ayrıştırılması (error decomposition) konusunda incelenmektedir (Zhou, 2012). Bir topluluğun genelleme hatası (generalization error of ensemble), çeşitliliğe bağlıdır. Topluluk öğrenme yöntemlerinin genelleme hatası iki farklı şekilde değerlendirilmektedir. Bunlardan ilki, hata - belirsizlik ayrıştırması (error-ambiguity decomposition) diğeri ise, yanlılık - varyans - kovaryans ayrıştırması (bias-variance-covariance decomposition) olarak tanımlanmaktadır. Her iki hata ayrıştırılması konusunda, regresyon problemleri için daha genel sonuçlar elde edilmesine rağmen, sınıflandırma problemi için benzer sonnuçları elde etmek oldukça zordur (Geman, Bienenstock, & Doursat, 1992; Krogh & Vedelsby, 1994).

Bir regresyon problemi için hata - belirsizlik ayrıştırması ile hesaplanan topluluğun genelleme hatası, T bireysel sınıflandırıcıdan oluşan bir topluluk $h_1, \dots, h_T$, yaklaşık fonksiyonuna $f : R^d \rightarrow R$ tanımlanır (Krogh & Vedelsby, 1994). Topluluğun final tahmininin yani bireysel sınıflandırıcıların birleştirilmesi yönteminin ağırlıklı ortalama olduğu varsayıldığında,

$$H(x) = \sum_{i=1}^{T} w_i * h_i(x) \tag{32}$$

w_i: h_i sınıflandırıcısı için ağırlık ve bu ağırlıklar, $w_i \geq 0$ ve $\sum_{i=1}^{T} w_i = 1$ ile sınırlıdır.

Bir x örneği için h_i bireysel sınıflandırıcının belirsizliği,

$$ambi\ (h_i|x) = \left(h_i(x) - H(x)\right)^2 \tag{33}$$

Topluluğun belirsizliği ise,

$$\overline{ambi}(h|x) = \sum_{i=1}^{T} w_i *\ ambi\ (h_i|x) = \sum_{i=1}^{T} w_i \left(h_i(x) - H(x)\right)^2 \tag{34}$$

Bir x örneği için h_i bireysel sınıflandırıcının hatası,

$$err\ (h_i|x) = \left(f(x) - h_i(x)\right)^2 \tag{35}$$

Topluluğun hatası H ise,

$$err\ (H|x) = \left(f(x) - H(x)\right)^2 \tag{36}$$

Hata - belirsizlik ayrıştırmasına dayalı topluluk genelleme hatası,

$$err(H) = \overline{err}(h) - \overline{ambi}(h) \tag{37}$$

$\overline{err}(h) = \sum_{i=1}^{T} w_i * err(h)$: bireysel genelleme hatalarının ağırlıklı ortalamasıdır.

$\overline{err}(h) = \sum_{i=1}^{T} w_i * ambi(h)$: topluluk belirsizliği (ensemble ambiguity) olarak da adlandırılan belirsizliklerin ağırlıklı ortalamasıdır.

Topluluk genelleme hatasına ait formülde, ilk terim $\overline{err}(h)$, bireysel sınıflandırıcıların genelleme yapabilmesine bağlı bireysel sınıflandırıcıların ortalama hatasını ifade etmektedir. İkinci terim ise $\overline{ambi}(h)$, topluluk çeşitliliğine bağlı olarak bireysel sınıflandırıcıların tahminleri arasındaki değişkenliği yani varyansı ölçebilen belirsizliği ifade etmektedir. Topluluk genelleme hatası formülünde, bireysel sınıflandırıcılar ne kadar doğru ve ne kadar çeşitli olursa, topluluğun daha iyi olduğu gösterilmektedir. Tüm bu hesaplamaların regresyon problemi için olduğunu unutmamak gerekmektedir. Sınıflandırma problemi için yapılacak olan benzer hesaplamalar daha zor olduğundan, aynı şekilde deneysel olarak belirsizliği $\overline{ambı}$ tahmin etmek de zordur. Genel olarak, $\overline{ambı}$ tahmin değeri, err tahmin değerinden $\overline{err}$ tahmin değeri çıkarılarak elde edilmektedir. Bu nedenle bu tahmin değeri, sadece çeşitliliği ifade etmekle kalmaz, aynı zamanda topluluk hatası ve bireysel hata arasındaki farkı da ifade

etmektedir. $\overline{ambı}$ tahmin değeri, pozitif değeri sahip olması gerekirken bazen bu hesaplamadan dolayı farklı bir değer de alabilmektedir. Bu hesaplama formülü, araştırmacıya bazı bilgiler sunmasına rağmen, topluluk çeşitliliğinin birleşik bir genel formülünü sağlamamaktadır (Krogh & Vedelsby, 1994).

Bir regresyon problemi için yanlılık - varyans - kovaryans ayrıştırması ya da diğer adıyla yanlılık - varyans ayrıştırması, öğrenme algoritmalarının performansını analiz etmek için önemli bir yöntemdir (Geman et al., 1992). Bir öğrenme hedefi ve eğitim veri setinin genişliği göz önüne alındığında, bir sınıflandırıcının genelleme hatası diğer ifadeyle içsel gürültü (intrinsic noise), yanlılık ve varyans olmak üzere üç bileşen altında incelenmektedir. İçsel gürültü, hedef üzerindeki herhangi bir öğrenme algoritmasının beklenen hatası üzerindeki alt sınırı ifade etmektedir. Yanlılık, öğrenme algoritmasının ortalama tahmininin hedefi nasıl yaklaştırabileceğini ölçmektedir. Varyans ise, öğrenme yaklaşımı tahmininin aynı büyüklükteki farklı eğitim setleri için ne kadar sapacağını belirtmektedir. İçsel gürültünün tahmin edilmesi zor olduğundan, genellikle yanlılık ile birlikte incelenmektedir. Bu nedenle genelleme hatası, sınıflandırıcının beklenen hatasını ifade eden yanlılık ve sınıflandırıcının eğitim veri setlerindeki varyasyonlara duyarlılığını ifade eden varyans ile ölçülmektedir.

Hesaplama formülünde, f, tahmini ve h, sınıflandırıcıyı ifade etmektedir.

$$err(h) = \mathbb{E}\left[(h - f)^2\right] \tag{38}$$

$$err(h) = (\mathbb{E}[h] - f)^2 + \mathbb{E}[(h - \mathbb{E}[h])^2] \tag{39}$$

$$err(h) = bias(h)^2 + variance(h) \tag{40}$$

h sınıflandırıcının yanlılık ve varyans ayrıştıması,

$$bias(h) = \mathbb{E}[h] - f \tag{41}$$

$$variance(h) = \mathbb{E}(h - \mathbb{E}[h])^2 \tag{42}$$

T bireysel sınıflandırıcıdan oluşan bir topluluk $h_1, \ldots, h_T$, yanlılık - varyans - kovaryans ayrışması genişletilmiş hali aşağıdaki formül gibidir (Ueda & Nakano, 1996). Genellenebilirlik kaybı (loss of generality) olmadan bireysel sınıflandırıcıların eşit ağırlıklarla birleştirildiği varsayılmaktadır. Bireysel sınıflandırıcıların ortalama yanlılıkları, ortalama varyansları ve ortalama kovaryansları sırasıyla,

$$\overline{bias}(H) = \frac{1}{T}\sum_{i=1}^{T}(\mathbb{E}[h_i] - f\ ) \tag{43}$$

$$\overline{variance}(H) = \frac{1}{T}\sum_{i=1}^{T}\mathbb{E}(h_i - \mathbb{E}[h_i])^2 \tag{44}$$

$$\overline{covariance}\,(H) = \frac{1}{T(T-1)}\sum_{i=1}^{T}\sum_{\substack{j=1 \\ j\neq i}}^{T}\mathbb{E}(h_i - \mathbb{E}[h_i])\,\mathbb{E}\big(h_j - \mathbb{E}\big[h_j\big]\big) \tag{45}$$

Topluluk hata karesinin yanlılık - varyans - kovaryans ayrıştırması,

$$err(H) = \overline{bias}(H)^2 + \frac{1}{T}\overline{variance}(H) + \left(1 - \frac{1}{T}\right)\overline{covariance}\,(H) \tag{46}$$

$err(H)$, topluluğun hata karesinin büyük çoğunlukla bireysel sınıflandırıcılar arasındaki korelasyonu modelleyen kovaryans terimine bağlı olduğunu göstermektedir. Kovaryans ne kadar küçük olursa, topluluk daha iyi performansa sahip olmaktadır. Tüm sınıflandırıcılar benzer hatalar yaparsa, kovaryansın büyük olacağı açıkça gösterilmektedir. Bu nedenle bireysel sınıflandırıcıların farklı hatalara sahip olması tercih edilmektedir. Aynı zamanda, $err(H)$ için hesaplanan kovaryans terimiyle de bir kez daha çeşitliliğin topluluk performansı için önemli olduğu gösterilmektedir. Yanlılık ve varyans terimlerinin pozitif değer olması gerekmektedir, kovaryans terimi negatif olabilmektedir. Ayrıca, yanlılık - varyans - kovaryans ayrıştırması yöntemine dayalı hesaplanan topluluğun genelleme hatası burada sadece regresyon problemleri için hesaplandığı için sınıflandırma problemi için araştırmacıya bazı bilgiler sunmasına rağmen topluluk çeşitliliğinin bir genel formülünü sağlamamaktadır. Brown ve diğerleri (2005) çalışmalarında hata - belirsizlik ayrışması ve yanlılık - varyans - kovaryans ayrışması arasındaki ilişkiyi açıklamıştır (Brown, Wyatt, & Tino, 2005). Bireysel sınıflandırıcıların eşit ağırlıklarla birleştirildiği varsayıldığında,

$$\overline{err}(H) - \overline{ambi}(H) = \mathbb{E}\left[\frac{1}{T}\sum_{i=1}^{T}(h_i - f\,)^2 - \frac{1}{T}\sum_{i=1}^{T}(h_i - H)^2\right] \tag{47}$$

$$\overline{err}(H) - \overline{ambi}(H) = \overline{bias}(H)^2 + \frac{1}{T}\overline{variance}(H) + \left(1 - \frac{1}{T}\right)\overline{covariance}\,(H) \tag{48}$$

Buna göre,

$$\overline{err}(H) = \mathbb{E}\left[\frac{1}{T}\sum_{i=1}^{T}(h_i - f\,)^2\right] = \overline{bias}^2(H) + \overline{variance}(H) \tag{49}$$

$$\overline{ambı}(H) = \mathbb{E}\left[\frac{1}{T}\sum_{i=1}^{T}(h_i - H)^2\right] = \overline{variance}(H) - variance(H)$$

$$= \overline{variance}(H) - \frac{1}{T}\overline{variance}(H) - \left(1 - \frac{1}{T}\right)\overline{covariance}\,(H) \qquad (50)$$

Böylece, $\overline{variance}$ terimi, hem ortalama hata kare terimi hem de ortalama belirsizlik teriminde görülmektedir. $\overline{variance}$ terimi, belirsizliği hata teriminden çıkarılmasıyla yok edilir yani $\overline{variance}$ teriminin hem $\overline{err}$ hem de $\overline{ambı}$ terimlerinde görünmesi durumu, yanlılık terimini etkilemeden belirsizlik terimini maksimize etmenin zor olduğunu göstermektedir. Bu durum, farklı sınıflandırıcıların bir araya getirilmesinin zor olduğu anlamına gelmektedir (Brown, Wyatt, Harris, & Yao, 2005).

3.8.4. Topluluk Genişliği

Topluluk öğrenme yöntemlerinin önemli diğer bir konusu da, toplulukta kaç bireysel sınıflandırıcının olması gerektiğini ifade eden topluluk genişliğini belirtmektir. Bu sayı, temel olarak üç farklı soruna göre tanımlanabilmektedir:

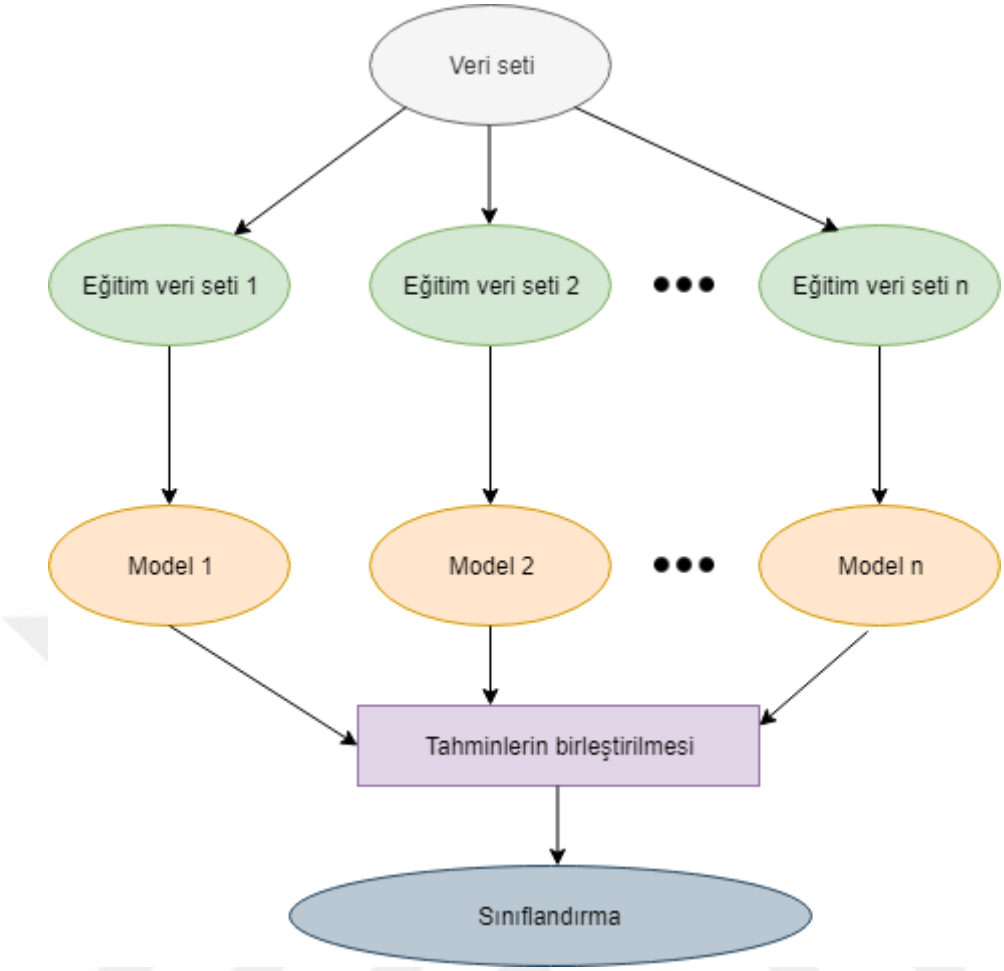
1. İstenen doğruluk (desired accuracy) - Hansen ve Salamon (1990), 10 farklı bireysel sınıflandırıcı içeren toplulukların, hata oranını azaltmak için yeterli olduğunu savunmuştur (Hansen & Salamon, 1990). Bununla birlikte, AdaBoost algoritmasının karar ağaçları kullanması durumunda, 25 sınıflandırıcı içeren göreceli olarak büyük topluluklarda bile hatanın azalması gözlemlendiğine dair deneysel çalışmalar vardır (Opitz & Maclin, 1999). Özellikle, ayrık bölümleme yaklaşımlarında (disjoint partitioning approaches), alt kümelerin sayısı ile son hesaplanan doğruluk arasında bir sapma olabilmektedir. Her bir alt kümenin genişliği çok küçük olmamalıdır çünkü etkili bir sınıflandırıcı üretmek için her öğrenme süreci için yeterli veri bulunması gerekmektedir. Chan ve Stolfo (1998) yaptıkları çalışmada, toplulukta bulunan ağaçlardaki alt kümelerin sayısını 2 ile 64 arasında değiştirmiş ve bu durumun topluluk doğruluğu düzeyi üzerindeki etkisini incelemiştir (Chan & Stolfo, 1998).
2. Kullanıcı tercihleri (user preferences) - Sınıflandırıcıların sayısının arttırılması genellikle hesaplama karmaşıklığını arttırmakta ve anlaşılabilirliği azaltmaktadır. Bu nedenle kullanıcılar, topluluk genişiliği sınırını önceden tanımlayarak tercihlerini belirleyebilmektedirler (Zhou, 2012).

3. Kullanılabilir işlemci sayısı - Eş zamanlı yaklaşımlarda, paralel öğrenme için kullanılabilir işlemci sayısı, paralel işlemde ele alınan sınıflandırıcıların sayısına bağlı olabilmektedir (Zhou, 2012).

Caruana ve diğerleri (2004), binlerce modelin bulunduğu bir kütüphaneden topluluklar kurmak için bir yöntem geliştirmişlerdir. Topluluğun performansını en üst seviyeye çıkaracak modelleri seçmek için ileriye doğru seçim (forward stepwise selection) yapılmasını önermişlerdir. Topluluk seçimi (ensemble selection), toplulukların doğruluk, çapraz entropi (cross entropy), ortalama hassaslık (mean precision) veya ROC eğrisi gibi performans ölçütlerine göre optimize edilmesini sağlamaktadır (Caruana, Crew, & Ksikes, 2004). Topluluk genişliğinin önemine dikkat çeken bazı çalışmalar, farklı veri yapılarında topluluk çeşitliliğinin ve performans ölçütlerinin farklı topluluk genişliklerinde nasıl değiştiğini vurgulamışlardır (Shuo Wang & Xin Yao, 2013). Sınıflandırma algoritmaları, topluluk genişliğinden ve eğitim veri setlerindeki sınıf dengesizliğinden ciddi derecede etkilenmektedir.

3.9. Ağaç Tabanlı Topluluk Öğrenme Algoritmaları

Ağaç tabanlı topluluk öğrenme, diğer topluluk öğrenme yöntemlerindeki gibi tek bir sınıflandırıcı kullanmak yerine çoklu sınıflandırıcıların kullanıldığı hassas bir yöntemdir. Buradaki temel mantık, Şekil 15'te gösterildiği gibi çok farklı sayıda ağacın tahminlerini bir araya getirmektedir. Torbalama ve artırma yöntemleri ise iki önemli temel topluluk öğrenme yöntemi olarak belirtilmektedir. Sağlık alanında en çok tercih edilen torbalama algoritmaları, rastgele orman (*random forest - rf*) ve ağırlıklı alt uzay rastgele orman (*weighted subspace random forests - wsrf*) iken; yaygın olarak kullanılan artırma algoritmaları, eklemeli lojistik regresyon (*additive logistic regression - logitboost*) ve gradyan artırma makinaları (*gradient boosting machines - gbm*) algoritmalarıdır (Witten et al., 2011).



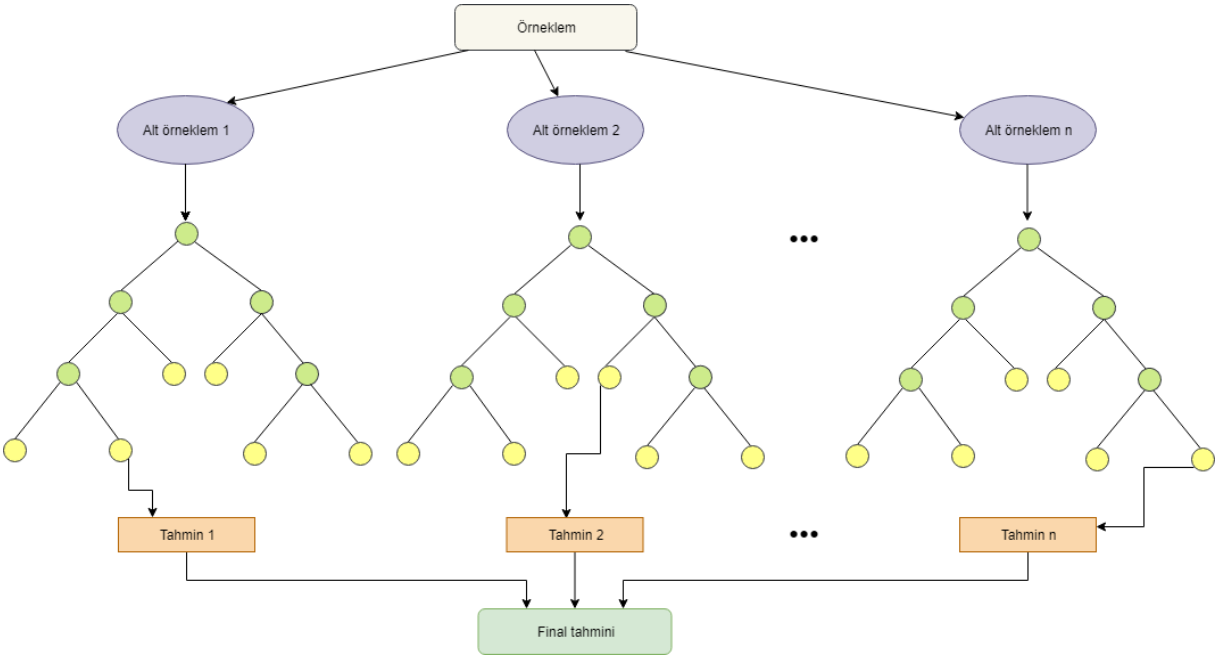
Şekil 15: Ağaç Tabanlı Topluluk Öğrenme

Şekil 15'te gösterildiği gibi, ağaç tabanlı bir topluluk öğrenme yönteminde orijinal veri setinden farklı eğitim veri setleri alt kümeleri elde edilir. Daha sonra bu eğitim veri setleri için modeller tahminlenir ve sonuçlar birleştirilir. Son olarak, final modeli kurulur ve test verileri ile modeller test edilir (Dietterich, 2000b).

3.9.1. Rastgele Orman

Rastgele orman algortiması, 2000 yılında Leo Breiman tarafından önerilen torbalama yöntemi ve Ho'nun 1998 yılında önermiş olduğu rastgele alt uzay yönteminin bir arada kullanılmasına imkan sağlayan bir torbalama yöntemdir (Breiman, 2001; Ho, 1998) . Rastgele orman, 2001 yılında Breiman tarafından artırma algoritmalarına göre daha fazla randomizasyon sağladığı gerekçesiyle önerilmiştir (Breiman, 2001). Rastgele orman algoritması, torbalama algoritması temeline dayandığından aynı zamanda eş zamanlı bir yaklaşımdır. Bu algoritmada, orman içerisinde yer alan bireysel sınıflandırıcılar olarak ifade edilen karar ağaçları, CART

algoritmasıyla elde edilmektedir (Sutton, 2005). Bu bireysel sınıflandırıcılar bir araya gelerek, karar ormanı topluluğunu oluşturmaktadır. Karar ormanını oluşturan her bir karar ağacı, orijinal veri setinden rastgele yerine koyarak seçilen bootstrap örnekleme yöntemiyle farklı örneklemler oluşturmaktadır. Karar ormanı topluluğu oluşturulduktan sonra tahmin sonuçları bir araya getirilerek birleştirilerek son tahmin elde edilmektedir (Şekil 16).



Şekil 16: Rastgele Orman Çalışma Prensibi

Bootstrap örnekleme yöntemiyle elde edilen örneklemler birbirinden bağımsızdır. Rastgele alt-uzay yöntemi sayesinde, tüm değişkenler arasından rastgele olarak m tane değişken oluşturulmaktadır. m değişken sayısı, toplam değişken sayısından daha küçük olmalıdır. Bu m tane değişken arasından karar ağacının her bir düğümü için en iyi dallara bölünmesine sağlayacak değişken belirlenmektedir. Orman içerisinde oluşturulan her bir karar ağacı en geniş haliyle bırakılmaktadır ve bu karar ağaçları budanmamaktadır. Eğer sınıflandırma problemi ise, karar ağaçlarında her bir yaprak düğüm sadece bir sınıfın kategorilerini içererek oluşturulmaktadır. Eğer regrasyon problemi ise, yaprak düğümde en az sayıda örnek kalacak şekilde karar ağaçları bölünmeyi sürdürmektedir (Breiman et al., 1984).

Rastgele orman algoritması iki parametreye sahiptir (Breiman, 2001).

- Oluşturulacak ağaç sayısı (B)
- Her bir düğümün bölünmesinde rastgele seçilecek değişken sayısı (M)

Bir rastgele orman algoritması oluşturulurken sırasıyla (Breiman, 2001),

1. Her bir karar ağacının oluşturulması için orijinal veri setinden n genişliğinde bootstrap örnekleme yöntemiyle örneklem seçilir. Bu örneklemin $^2/_3$ oranı ile karar ağacını elde etmek için kullanılan eğitim veri seti (inBag) ve $^1/_3$ oranı ile de kurulan modelin iç hata oranını test etmek için test veri seti (out of bag - OOB) olmak üzere ikiye ayrılır.
2. Eğitim veri setinde, CART karar ağacı algoritmasıyla kurulan en geniş karar ağacı kurulur ve bu karar ağacı budanmaz.
3. CART algoritmasıyla kurulan karar ağacı oluşturulurken, her bir düğümün bölünmesinde kullanılacak toplam M tane bağımsız değişken arasından rastgele m tane bağımsız değişken seçilir. Burada dikkat edilmesi gereken durum, $m < M$ koşulu sağlanmalıdır. Rastgele seçilen m bağımsız değişkenden bilgi kazancı en yüksek olan değişkenle düğümler dallara ayrılır. Bilgi kazancı en yüksek olan değişkenin hangi değerine göre bölünmenin olacağına, gini indeksi yöntemiyle karar verilmektedir. Bu süreç, her bir düğüm için yeni oluşturulacak dal kalmayana kadar tekrarlanır (Breiman et al., 1984).
4. Karar ağacı oluşturulduktan sonra, her bir yaprak düğüm bir sınıfa atanır. Böylelikle karar ağacı model kurulumu tamamlanır.
5. Model kurulduktan sonra, test veri seti kök düğümden yani ağacın en tepesinden bırakılır ve test veri setinde bulunan her bir örnek yani gözlem için atanan sınıf kaydedilir.
6. 1. adımdan 5. adıma kadar tüm adımlar B defa tekrarlanır.
7. Test veri setine ait gözlem ile bir değerlendirme yapılarak, belli bir örneğin hangi kategorilerde kaç defa sınıflandırıldığı sayılır.
8. Her bir gözleme, karar ağacı kümeleri üzerinden belirlenen çoğunluk oylama ile sınıf ataması yapılır (örneğin, ikili sınıflandırma modelinde, belli bir gözlem tüm karar ağaçlarından en az %51 oranında çoğunluk oyu aldığı sınıfın etiketini alır. Böylelikle, bu gözlem için atanan sınıf, gözlemin tahminlenen sınıfı olur).

Rastgele orman algoritmasının belli başlı bazı önemli özellikleri vardır. Bunlar, genelleme hatası (generalization error), parametreleri ayarlama (tunning parameters), değişken önemliliği (variable importance), farklı sınıf büyüklükleri (unequal class sizes), örnekler arası uzaklık (proximity) ve kayıp değer atama özellikleridir.

Genelleme hatası, orijinal veri setinden bootstrap örnekleme yöntemiyle çekilen bir örneklemin içerisinde bazı gözlemler karar ağacı oluşturulması sırasında yer almadığı durumda ortaya çıkmaktadır (Maimon & Rokach, 2005). Örneklem içerisinde yer almayan bu gözlemler OOB yani test veri seti olarak adlandırılır ve bu gözlemlerle genelleme hatasına bağlı bir modelin iç hata oranı tahmin edilir. OOB hata oranının tahmin edilmesinde, her bir karar ağacı OOB veri seti için bir sınıf değeri tahmin edilir ve bu tahminler kaydedilir. Genelleme hatası, karar ağacı modellerinin herhangi bir noktasında, örneklem içerisindeki her bir gözlemin OOB olduğu karar ağaçlarındaki hata oranı tahminlerinin ortalamasıyla hesaplanmaktadır. Bu durumda genel anlamda bir hata oranı tüm gözlemlerin ortalaması alınarak hesaplanabilmektedir (Hastie et al., 2009).

Rastgele orman sınıflandırıcısının genelleme hatasının OOB tahmini hesaplanmasında, N orijinal veri setinden rastgele olarak S test veri seti ve yeniden yerine koyarak T eğitim veri seti seçilir. T'den herhamgi bir N gözlemin seçilme olasılığı, $1/N$ ve seçilmeme olasılığı $1 - 1/N$ olarak hesaplanır. Aynı gözlemin her N kez denemede seçilmeme olasılığı,

$$\left(1 - \frac{1}{N}\right)^N$$

T'deki tüm gözlemler aynı şekilde her N kez denemede seçilmeme olasılığı,

$$\lim_{n\to\infty}\left(1 + \frac{1}{N}\right)^N = \frac{1}{e} \approx 0{,}37 \qquad (51)$$

N yeterince büyük olduğu durumda, T gözlemlerinin yaklaşık % 37'si S veri setinin gözlemleri değildir yani gözlemlerin yaklaşık % 37'si S 'nin dışında kalır. $T_1, \dots, T_M$ rastgele bir ormandaki M tane karar ağacı sınıflandırıcıları $f_1, \dots, f_M$ için kullanılan M giriş kümeleri olduğunu varsayalım. Yalnızca eğitim verileri üzerinde OOB sınıflandırıcı f_{oob}, örneklem, T_m'de olmaması koşuluyla, sınıflandırıcı f_m'in bir eğitim örneklemi için oy vermesine izin verilerek oluşturulabilir.

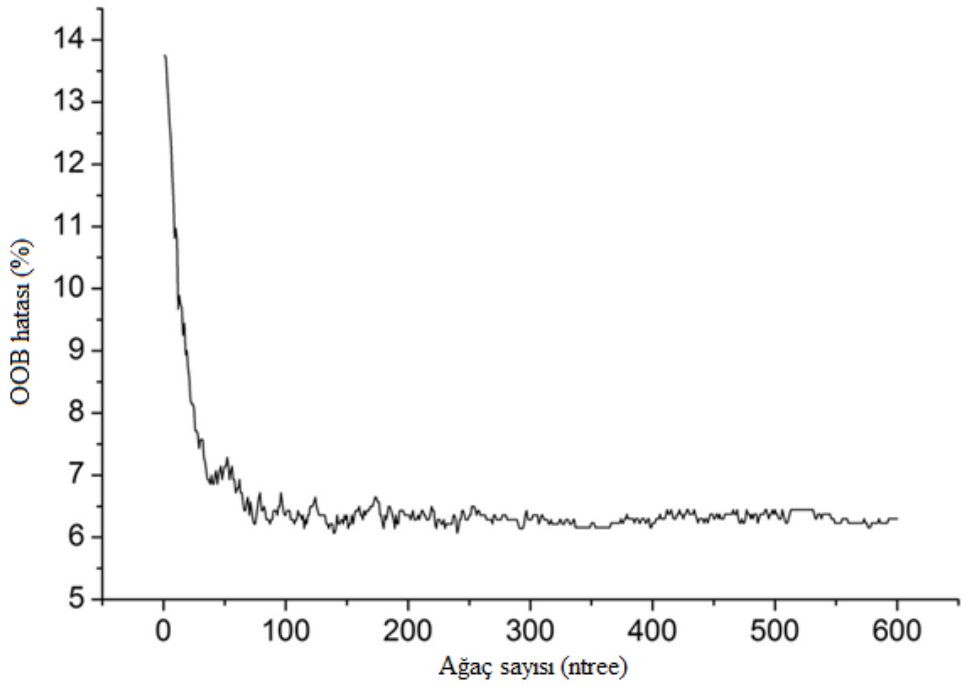
$$v(y) = |\{m|(x, y') \notin T_m \; ve \; f_m(x) = y\}| \qquad (52)$$

ve böylelikle, çoğunluk oyu $\arg max_y v(y)$ hesaplanır. OOB hatası, f_{oob}'nin eğitim hatasıdır ve

$$\frac{1}{N}\sum_{n=1}^{N} I\left(y_n \neq f_{oob}(x_n)\right) \tag{53}$$

rastgele ormanın genelleme hatasının yansız kestiricisi (unbiased estimate) olarak hesaplanır (Breiman, 1996).

Parametreleri ayarlamasında, oluşturulacak ağaç sayısı (B) ve her bir düğümün bölünmesinde rastgele seçilecek değişken sayısı (m) parametrelarinin rastgele orman algoritması için ayarlanması gerekmektedir. Breiman (2001) her iki parametrenin ayarlanması için farklı öneriler bulunmuştur. Breiman (2001), B parametresi için 500 adet ağaçtan oluşan bir karar ormanı yeterli sayılabilmektedir. Sınıflandırma problemlerinde m parametresi için $m = \sqrt{M}$ olarak hesaplanmaktadır. Regresyon problemlerinde ise, $m = {}^{M}/_{3}$ olarak hesaplanmaktadır (M, orijinal veri setindeki toplam bağımsız değişken sayısı). Rastgele orman algoritması karar ağacı ormanına daha fazla karar ağacı eklemek yani B sayısını arttırmak aşırı uyuma neden olmadığından bu sayısının yeterli büyüklükte olması önemlidir. B sayısının kontrol edilebilmesi için OOB hata oranına dikkate edilmelidir (Breiman, 2001).



Grafik 2: Rastgele Orman OOB Hatası (Q. Feng, Liu, & Gong, 2015)

Grafik 2'de, OOB hata oranı belirli bir ağaç sayısından sonra sabit bir değere yakınsadığı görülmektedir. Bir regresyon probleminde ise bu parametrelerin ayarlanması, ağaçların derinliği ya da yaprak düğümlerde kalan minimum gözlem sayısının kontrol edilmesiyle sağlanmaktadır.

Değişken önemliliği, bağımsız değişkenlerin tahmin ediciliğinin ölçülmesi anlamına gelmektedir. Değişken önemliliğinin amaçlarından biri, karar ormanı modelinin yorumlanabilirliğini sağlamak; diğeri ise, değişken seçimini kolaylaştırmaktır. En önemli amacı ise, model performansını geliştirerek aşırı uyumu ortadan kaldırmaktır. Rastgele orman algoritması sınıflandırma problemlerinde kurulan modeller için doğrudan değişken seçimi yapmaktadır (Breiman, 2001; Maimon & Rokach, 2005). Örneğin, bazı boyut indirgeme algoritmaları doğrudan değişken seçimi yapamadığından, tahmin modeli için önemli ölçüde bilgi kaybı ortaya çıkmaktadır. Değişken önemliliğinin hesaplanmasında birbirlerine yaklaşık aynı sonuçları veren gini önemliliği ve permütasyona dayalı değişken önemliliği ile hesaplanabilmektedir.

Farklı sınıf genişliği durumu sınıfllandırma probleminde, sınıflar arasındaki örneklem sayısı farklarından meydana gelmektedir. Bu durum, aynı zamanda sınıf dengesizliği problemi olarak adlandırılmaktadır. Sınıf dengesizliğinde, sınıflandırıcılar azınlık sınıfını görmezden gelerek çoğunluk sınıfına odaklandığı için, çoğunluk sınıflarına yönelik bir hata oranına sebep olmaktadır. Rastgele orman algoritması, sınıf dengesizliğina sahip veri setlerinde dengeli sonuçlar elde edebilmek için sınıfları ağırlıklandırır. Bu algoritma sayesinde, azınlık sınıflarına odaklanılarak önemli bağımsız değişkenlerde farklılıklar görülebilinmektedir. Unutulmaması gerekilen durum ise, sınıf dengesizliğine sahip olmayan yani dengeli veri setlerinde bile büyük ölçüde yanlış sınıflandırma maliyetine sahip kararlara daha düşük hata oranları verilerek ağırlıklandırmalar düzenlenebilmektedir (Hastie et al., 2009).

Örnekler arası uzaklık, çok büyük veri setlerinde ortaya çıkan bir durumdur. Bu durum, veri setinin tutarlı olup olmadığını açık bir şekilde gözlemleyememekten kaynaklanmaktadır. Belli sınıflarda alt kümelerin oluşumu veya bu sınıflarda benzer örüntülerin var olup olmaması, sapan değerler, bağımlı değişkenin çoklu sınıfa sahip olması durumunda bazı gözlemlerin birbirileriyle örtüşmesi veya bazılarının birbirinden ayrık olması gibi durumlar veri setinin tutarlılığını etkilemektedir. Rastgele orman algoritması, bu durumları anlayabilmek için rehberlik etmektedir. Rastgele orman algoritması, gözlem çiftleri arasındaki uzaklıkları ölçerek (proximity measure) hesaplanmaktadır. İki gözlem arasındaki uzaklık, iki gözlemin aynı yaprak düğümde (terminal node) sonlanma oranlarına eşittir. Bu oran, ormandaki ağaçlar

üzerinden hesaplanarak gözlemlerin birbirleriyle ne derecede ilişkili olduğunu, başka bir ifade ile korelasyon gösterdiğinin anlaşılmasını sağlamaktadır. Karar ağaçları arasındaki korelasyonun artmasına bağlı olarak ormandaki ağaçların performansı düşmektedir. Ağaçların performansını arttırmak yani ağaçlar arasındaki korelasyonu azaltmak için her bir düğümde rastgele değişken seçerek seçilen değişkenlerle karar ağacı modelinin kurulması gerekmektedir. Bu durum, oluşan farklı karar ağaçlarının performansını arttırmakta, böylelikle çeşitli karar ağaçlarından oluşan ormanın daha doğru kararlar vermesi sağlanmaktadır (Breiman et al., 1984). Rastgele orman algoritması, uzaklık ölçüsünü kullanarak $n \times n$ boyutlu simetrik bir uzaklık matrisi (proximity matrix) oluşturmaktadır. n, karar ağacı oluşturulurken veri setinde bulunan toplam gözlem sayısını ifade etmektedir. Her bir bireysel sınıflandırıcı karar ağacı oluşturulduktan sonra eğitim ve test veri setleri (OOB ve inBag) karar ağacına yukarıdan aşağı doğru birlikte yerleştirilmektedir. Aynı yaprak düğümde yer alan gözlemlerin sayısı kaydedilmekte ve bir uzaklık matrisi oluşturulmaktadır. Uzaklık matrisinin her bir hücresi ormandaki toplam ağaç sayısına bölünmekte ve böylece uzaklık oranları hesaplanmış olmaktadır. Eğer iki gözlemde her zaman aynı yaprak düğümde sonlanırsa, uzaklık değerleri 1 değerine eşit olmaktadır. İki gözlemin hiçbir zaman aynı yaprak düğümünde sonlanmadığı durumda ise uzaklık değerleri, 0 değerine eşit olmaktadır. Aynı zamanda, diğer gözlemlerle arasındaki uzalık oranları oldukça düşük olan gözlemler sapan değerler olarak yorumlanabilmektedir. Yine benzer olarak, diğer gözlemlerle arasındaki uzaklık oranları oldukça yüksek olan gözlemlerin, birbirlerine daha benzer yapılar gösterdiği anlamına gelmektedir (Hastie et al., 2009; Maimon & Rokach, 2005).

Kayıp değer atama konusunda, rastgele orman algoritması kendi içinde bir algoritma geliştirerek kayıp gözleme sahip değişkenler için atamalar yapmaktadır. Rastgele orman algoritmasının geliştirildiği bu algoritmanın temelini uzaklık ölçüsü oluşturmaktadır (Breiman, 2001). İlk aşamada, veri setindeki kayıp gözlemler belirlenmekte; eğer kayıp gözlem ya da gözlemler bir sürekli değişkene aitse, medyan değeri bulunarak atamalar yapılmaktadır. Kayıp gözlem ya da gözlemlerin kategorik değişken olduğu durumda ise, en yüksek frekans değerine sahip kategoriyle atama yapılmaktadır. İkinci aşamada, tamamlanmış veri setiyle bir rastgele orman modeli kurularak, bu modelden bir uzaklık matrisi oluşturulmaktadır. Bu matristeki uzaklıklar, ağırlıklandırma ölçüsü olarak kullanılmaktadır. Eğer sürekli değişkense, kayıp değerler için tamamlanmış verilerin uzaklık oranları kullanılarak ağırlıklı ortalaması hesaplanarak atama yapılır. Eğer kategorik değişkense, kayıp değerler için tamamlanmış verilerin uzaklık oranı en yüksek kategori değeri kullanılarak

atama yapılır. Üçüncü aşamada, yeniden atama yapılarak tamamlanmış veri seti ile yeni bir rastgele orman modeli kurularak yeni bir uzaklık matrisi oluşturulur. Bu uzaklık matrisi kullanılarak kayıp gözlemler için tutarlı değerler elde edilene kadar beş defa bu süreç tekrarlanır. Bu süreç aynı zamanda, k-en yakın komşu yaklaşımı olarak adlandırılmıştır (Breiman, 2001).

Rastgele orman algoritması birçok öğrenme algoritmasına göre daha genellebilir ve daha geçerli tahminlere sahip modeller oluşturmaktadır. Bu sayede, modelin yorumlanabilirliği de sağlanmaktadır. Bu algoritma, hem rastgele örnekleme yöntemini hem de topluluk öğrenme algoritmasını bir arada sunmasından dolayı tahminlerin geçerliliği diğer algoritmalara göre daha iyidir. Rastgele orman algoritmasının tahminlerinin yanlılığının düşük olması ve karar ağaçları arasındaki korelasyonun düşük olmasından dolayı tahminlerinin doğruluğu daha yüksektir. Yanlılık miktarının düşük olmasının sebebi, olabildiğince büyük karar ağaçlarının oluşturulması iken; karar ağaçları arasındaki korelasyonun düşük olmasının sebebi, olabildiğince birbirinden farklı karar ağaçlarının oluşturulmasıdır. Aynı zamanda, bu algoritma aşırı uyuma karşı dayanıklıdır (Maimon & Rokach, 2005; Zhou, 2012).

Rastgele orman algoritmasını diğer algoritmalardan ayıran en önemli özelliklerden biri, her ağaçtaki değişkenlerin farklı olmasından dolayı aşırı uyum probleminin ortaya çıkmamasıdır. Bu yüzden, rastgele orman algoritması aşırı uyuma karşı dayanıklıdır. Bununla birlikte rastgele orman algoritması yapısı gereği hızlı ve istenildiği kadar ağaçla çalışabilmesi nedeniyle şu ana kadarki algoritmalar arasında en yüksek doğruluğa sahip algoritma olarak tanımlanmaktadır (Breiman, 2001).

3.9.2. Ağırlıklı Alt Uzay Rastgele Orman

Ağırlıklı alt uzay rastgele orman algoritması, Xu (2012) tarafından önerilmiş torbalama yöntemdir (B. Xu, Huang, Williams, Wang, & Ye, 2012). *Wsrf* algorirması, *rf* algoritmasında olduğu gibi karar ağaçları kullanmakta ve düğümler için değişkenlerden yararlanmaktadır. Özellikle değişkenlerin önemlilik dereceleri *wsrf* algoritmasının performansı için büyük öneme sahiptir. Bunun nedeni ise, karar ağaçlarının bölünmeye hangi değişkeni kullanarak başlayacağını belirlemesidir. Eğer önem derecesi yüksek değişkenlerle karar ağaçaları kurulmazsa, ormanda zayıf karar ağaçları modelleri kurulacaktır. Böylelikle, *wsrf* modelinin genel doğruluğunun azalmasına neden olacaktır (S. Kotsiantis, 2011). Amaratunga, Cabrera ve Lee (2008) yaptıkları çalışmada bu sorunun çözümü için alt uzay örneklemesi için bir değişken ağırlıklandırma yöntemini (weighting method for subspace sampling method)

önermişlerdir. Bu yöntemle, bir değişkenin ağırlığı, varyans analizi için kullanılan t-testi ile değişken ve sınıf arasındaki korelasyondan hesaplanmaktadır. Bu ağırlık, bir alt uzay için seçilen değişkenin olasılığı olarak kullanılmaktadır. Değişken ile sınıf arasındaki korelasyon ne kadar yüksek olursa, değişkeninin sahip olacağı ağırlıkta bir o kadar yüksek olacaktır. Böylelikle değişken ağırlığı yüksek olan değişkenle kurulan karar ağaçları performansında da artış olacaktır (Amaratunga, Cabrera, & Lee, 2008). *Wsrf* modelinde, bilgilendirici değişkenlerin rastgele bir ormanda karar ağacı yetiştirilirken seçilmeleri daha olası olduğundan, ormanı oluşturan ağaçların ortalama başarısında bir artışa neden olmaktadır. Xu (2012), Amaratunga'nın önerdiği yöntemi, çoklu sınıf durumunda uygulanabilmesi için değişken ağırlıklarının hesaplanabilmesi için bilgi kazanma oranını (Information Gain Ratio - IGR) kullanan ve çok yüksek boyutlu verileri sınıflandırmak için ağırlıklı alt uzay rastgele orman algoritmasına dayanan rastgele bir orman algoritması olan *wsrf* algoritmasını önermektedir (Amaratunga et al., 2008; B. Xu et al., 2012). Bu algoritma, çoklu sınıf durumunda uygulanabilir olmasının yanı sıra, p toplam değişken sayısı olan $\log_2(M) + 1$ sabit alt uzay büyüklüğünü (fixed subspace size) kullanırken Breiman'ın (2001) önerdiği rastgele ormanı algoritmasıyla karşılaştırılabilir modeller oluşturacaktır (Breiman, 2001) (Breiman 2001). Bununla birlikte, oluşturulan sıralı algoritmalar kullanarak oluşturulan rastgele orman için hesaplama süresi uzun olabilmektedir.

3.9.2.1. Alt Uzay Seçimi için Değişken Ağırlıklandırma

Rf algoritması, bootstrap örnekleme ile tüm eğitim verilerinden hem gözlemleri hem de değişkenleri ($m < M$ koşuluna bağlı düğümün bölünlemesi için değişken seçimi) örneklemdirmektedir. Bu nedenle, ormandaki tüm ağaçlar birbirinden farklıdır ve her ağaç, eğitim veri setini oldukça farklı açılardan modelleyerek ağaçlar arasında yüksek çeşitlilik göstermesini sağlamaktadır (Zhou, 2012). Bu çeşitlilik, ortaya çıkan topluluk modelinin genel doğruluğunun artmasını sağlamaktatır. *Wsrf* algoritması ise, *rf* algoritmasının bu özelliklerini de kullanarak yüksek boyutlu verilerde aynı başarısı yakalamaya çalışmıştır. *Wsrf* algoritmasında düğümün bölünmeye başlayacağı değişkeni belirleyen parametre olan *mtry*, *rf* algoritmasının kullandığı geleneksel rastgele değişken örneklemenin (random variable sampling) yerine değişken alt uzay seçimi için yeni bir değişken ağırlıklandırma yöntemi kullanılmaktadır. $m < M$ koşuluna bağlı olarak,

$$m = \log_2(M) + 1 \tag{54}$$

şeklinde hesaplanmaktadır. Xu (2012)'nun bu yöntemi önermesine yol açan nedenlerden bazıları, yüksek boyutlu veriler için hesaplanan *mtry* parametresinin ($m = \sqrt{M}$ sonucuna göre) yetersiz kalması ve topluluk çeşitliliğinin azalmasıdır (B. Xu et al., 2012). Bu yöntemde, bir değişkenin sınıfa göre bilgilendiriciliği bir bilgi kazanma oranı ile ölçülmektedir. Bu ölçüm, karar ağacının oluşturulması sırasında belirli bir düğümü bölerken değişken alt uzayına dahil edilmek üzere seçilen değişkenin olasılığını kullanılmaktadır. Bu nedenle, ölçüme göre en yüksek olasılık değerine sahip değişkenlerin, değişken seçimi sırasında aday olarak seçilmesi daha olasıdır. Böylelikle daha güçlü bir karar ağacı elde edilebilinmektedir (Ho, 1998).

Alt uzay seçimi için değişken ağırlıklandırması için yapılan hesaplamalar aşağıdaki gibidir. Val gözlemlerine sahip bir veri seti D verildiğini varsayalım ve sınıflandırmada bağımlı değişken Y olsun. Y değişkenin sınıf düzeyleri y_j olarak ifade edilsin. $j = 1, \dots, q$. D veri setindeki bir A kategorik değişkenini p farklı değere sahip olan a_i olarak tanımlanır ve $i = 1, \dots, p$. D veri setindeki gözlem sayısı, $A = a_i$ ve $Y = y_j$ durumunda val_{ij} olduğu koşulunu sağlamaktadır. Daha sonra, A'ya karşı Y'nin kontejan tablosu elde edilir, burada A için $val_{i.} = \sum_{j=1}^{q} val_{ij}$ marjinal toplamları $i = 1, \dots, p$ ve Y için $val_{.j} = \sum_{i=1}^{p} val_{ij}$ marjinal toplamları $j = 1, \dots, q$ ifade etmektedir. D'deki sınıf saflığını (class purity) ölçen D veri setinin bilgi ölçüsü (information measure) olarak tanımlanır:

$$Info(D) = -\sum_{j=1}^{q} \frac{val_{.j}}{val} \log_2 \frac{val_{.j}}{val} \tag{55}$$

bir $D_{A= a_i}$ alt kümenin bilgi ölçüsü ise aşağıdaki gibi tanımlanır:

$$Info\left(D_{A= a_i}\right) = -\sum_{j=1}^{q} \frac{val_{ij}}{val_{i.}} \log_2 \frac{val_{ij}}{val_{i.}} \tag{56}$$

ve A'ya bağlı tüm alt kümeler için bilgi entropisinin ağırlıklı toplamı (weighted sum of the information entropy):

$$Info_A(D) = -\sum_{i=1}^{p} \frac{val_{i.}}{val} Info\left(D_{A= a_i}\right) \tag{57}$$

Son olarak, bilgi kazanma oranı aşağıdaki gibi tanımlanır:

$$Gain(A) = Info(D) - Info_A(D) \tag{58}$$

$$SplitInfo(A) = -\sum_{i=1}^{p} \frac{val_{i.}}{val} \log_2 \frac{val_{i.}}{val} \tag{59}$$

$$IGR(A,Y) = \frac{Gain(A)}{SplitInfo(A)} \tag{60}$$

D veri setinin toplam M tane değişkeni olduğunu varsayalım, $A_1, A_2, \dots, A_M$. Ağırlıklı alt uzay rastgele orman yöntemi, bu değişkenlerin bilgilendiriciliğini ölçmek için $IGR(A,Y)$ bilgi kazanma oranını kullanmaktadır (Breiman, 1996; Maimon & Rokach, 2005). Bu değerleri değişken ağırlıkları olarak kabul ederek, bu oranları, seçilen bir her değişkenin olasılığı olması için normalleştirmektedir:

$$w_i = \frac{\sqrt{IGR(A_i,Y)}}{\sum_{i=1}^{M} \sqrt{IGR(A_i,Y)}} \tag{61}$$

Bu yöntem ile Breiman'ın (2001) yöntemiyle karşılaştırıldığında aralarında fark, her düğümde değişken alt uzayının seçilmesidir. Breiman (2011) yönteminde, $m < M$ koşulunu sağlayan rastgele değişkenlerini seçtikten sonra, düğümü bölmek için bir değişken seçilmektedir (Breiman, 2001). Ağırlıklı alt uzay rastgele orman yöntemi, aynı boyuttaki alt uzayları seçmektedir, ancak değişkenlerin her birinin dahil edilme olasılıkları için w_i ağırlığı kullanmaktadır. Ayrıca, *wsrf* algoritması için karar ağaçlarını oluştururken CART yerine C4.5 algoritmasını kullanmaktadır. Böylelikle, kategorik değişkenler, k sınıf düzeyi kadar bölünürken (k-way split); sürekli değişkenler, ikili bölünmektedir (binary split).

3.9.3. Eklemeli Lojistik Regresyon

Eklemeli lojistik regresyon algoritması, Friedman tarafından 1998 yılında önerilmiş bir artırma algoritmasıdır (J. Friedman, Hastie, & Tibshirani, 2000). Eklemeli regresyon, doğrusal regresyon modelinde kullanıldığı gibi sınıflandırma modelinde de uygulanabilmektedir. *Logitboost* sınıflandırıcısı, lojistik kayıp fonksiyonunu en aza indirgeyebilen eklemeli bir model kuran bir meta-tahmin edicidir. Tüm artırma yöntemlerde olduğu gibi *logitboost* yönteminde temeli AdaBoost algoritmasına dayanmaktadır. Diğer bir deyişle, *logitboost* algoritması, AdaBoost algoritmasının genelleştirilmiş halidir. Sınıflandırmada, AdaBoost algoritması sadece ikili sınıflar için model kurubilirken, *logitboost* algoritması hem ikili hem de çoklu sınıflar için model kurmaktadır. AdaBoost algoritması, monoton ve düz bir üssel kayıp fonksiyonunun beklentisini (expectation of exponential loss function) $ELLOSS(F) = J(F) = E\left(e^{-yF(x)}\right)$, en aza indirmek için geliştirilmiş ekleme lojistik regresyon modelinin

$F(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t f_t(\boldsymbol{x})$ en temel halidir (J. Friedman et al., 2000). Bununla birlikte, üssel kayıp fonksiyonu, sınıflandırma hatasıyla (classification error) birlikte üssel olarak değişmesinden dolayı, bu durum AdaBoost algoritmasının gürültülü verilerin karşı eğilimli göstermektedir. *Logitboost* algoritması, AdaBoost algoritmasının çözüm bulamadığı gürültülü verilerden kaynaklı aşırı uyum problemini ortadan kaldırmak için geliştirilmiştir. *Logitboost* algoritması, aşırı uyum problemi için sınıflandırma hatasını doğrusal bir şekilde azaltarak daha doğru bir genelleme sağlamaktadır. Bu sorunu çözmek için Friedman ve diğerleri (2000), sınıflandırma hatası ile doğrusal olarak değişebilen ve gürültü veri, aykırı değerlere karşı daha az duyarlı olduğu ortaya çıkan bir binom log-olabilirlik kayıp fonksiyonu (binomial log-likelihood loss function) önermiştir (J. Friedman et al., 2000).

$$LLOSS(F) = E\left[-\log\left(1 + e^{-yF(x)}\right)\right] \tag{62}$$

Böylelikle, *logitboost* algoritması lojistik kayıp fonksiyonunu kullanarak aşırı uyum problemine neden olan verilerin ağırlığını arttırmaktadır. AdaBoost algoritmasının karşı koyamadığı aşırı uyum probleminin üstesinden gelmektedir. *Logitboost* algoritmasının ise artırma yineleme (boosting iteration) sayısıdır ve T ile ifade edilir. Bu parametre aynı zamanda, *logitboost* algoritması için kurulacak modeldeki ağaç sayısını ifade etmektedir. Artırma yineleme parametresinin optimizasyonu için OOB hata oranı ve aşırı uyum problemi dikkate alınarak parametre ayarı yapılmaktadır. Bu parametrenin ayarlanmasında önemli bir noktası ise, değişken önemliliğidir. *Logitboost* algoritması modelinde bulunan değişkenlerin model için sıfırdan farklı önemli derecesine sahip değişkenler yer almalıdır. Bu durum dikkate alınarak artırma yineleme sayısı belirlenmektedir. İdeal artırma parametresinin belirlenmesi için erken durdurma (early stopping) adı verilen bir takım benzetimler yapılmalıdır. Bu benzetimde, test veri seti için OOB hata oranının en aza indirgendiği durum ve aşırı uyum probleminin var olmaya başladığı andaki durum birlikte değerlendirilerek durdurma işlemi yapılmakta ve en iyi artırma yineleme parametresine karar verilmektedir (Blagus & Lusa, 2015).

Zayıf sınıflandırıcıların oluşturulması, artırma algoritmaları ve diğer tüm topluluk öğrenme algoritmalarının performansını etkileyen en önemli faktörlerden biridir. *Logitboost* algoritmasında, zayıf sınıflandırıcılar $f_t(x)$, ağırlıklandırılarak aşırı uyuma karşı daha dayanıklı hale getirilmektedir. En küçük kareler yönteminin hata oranını en aza indirgemek için zayıf sınıflandırıcılar seçilmektedir. *Logitboost* algoritmasının çalışma disiplini aşağıdaki gibidir:

- N tane gözlemden oluşan bir S eğitim veri seti giriş verisi olarak alınır. X bağımsız değişken sınıf uzayından $\boldsymbol{x}_i$'ler ve $\boldsymbol{x}_i$'lerle ilişkili Y bağımsız değişken uzayının bir elemanı olan y_i ikili sınıflar çekilir. $\boldsymbol{x}_i \in X$ ve $y_i \in Y = \{-1,1\}$ olarak ifade edilir.

$$S = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_N, y_N)\} \tag{63}$$

- *Logitboost* algoritması, zayıf sınıflandırıcı $f_t(x)$, olarak bir sınıflandırıcı seçer.
- Bu sınıflandırıcı, t kez çalıştırılır $t = 1, \dots, T$.
- Zayıf sınıflandırıcı $f_t(x)$ ilk defa çalışırken S eğitim veri setindeki tüm gözlemlerinin ağırlığını eşit olarak alır ($i = 1, \dots, N$).

$$w_i = \frac{1}{N} \tag{64}$$

Burada topluluk fonksiyonu $F(\boldsymbol{x}) = 0$ ve olasılıklar, $p(\boldsymbol{x}) = P(y = 1|x) = 1/2$ eşittir.

- Tüm gözlemlerin ağırlıkları, zayıf sınıflandırıcının her çalışmasında tekrar güncellenir. Bu güncelleme işlemi, zayıf sınıflandırıcının test ettiği hipotezlerin yapmış olduğu yanlış sınıflandırmalara göre yapılır.

$$y_i^* = \frac{(y_i + 1)}{2} \tag{65}$$

$$w_i = p_i(\boldsymbol{x})[1 - p_i(\boldsymbol{x})] \tag{66}$$

$$z_i = \frac{y_i^* - p_i(\boldsymbol{x})}{w_i} \tag{67}$$

- $f_t(\boldsymbol{x})$ fonksiyonunu, w_i ağırlıklarını kullanarak, z_i'den $\boldsymbol{x}_i$'ye ağırlıklı en küçük kareler regresyon (weighted least-squares regression) modelini kurar.
- Çalışmamızda, $\{(\boldsymbol{x}_1, z_1), (\boldsymbol{x}_2, z_2), \dots, (\boldsymbol{x}_N, z_N)\}$ verisi için w_i ağırlıklarını kullanarak regresyon karar ağacı algoritması modeli kurulur.

Bir önceki hipotezde yanlış olarak sınıflandırılan gözlemlerin ağırlıkları arttırılarak, daha sonra, bir sonraki hipotezin oluşturulması için gerekli olan eğitim veri setinin yanlış sınıflandırılmış gözlemlerinin yer alma olasılıkları arttırılmaktadır. Böylelikle, bu süreç için t tane hipotez oluşturulmaktadır. Bu hipotezler daha sonrasında birleştirilerek tek bir hipotez haline getirilmektedir.

- Son sınıflandırıcı $LF(\boldsymbol{x}) = sing[F(X)]$ olarak kaydedilir.

Buraya kadar yapılan tüm hesaplamalar ikili *logitboost* algoritması için verilse de bu algoritma çoklu sınıf durumunda da kullanılmaktadır.

Logitboost algoritmasında, çoklu sınıflar için iki farklı çözüm yolu vardır. Bunlardan biri, biri diğerlerine karşı *logitboost* (one-vs.-others logitboost) yöntemi; ikinci yol ise çoklu sınıf *logitboost* (multi-class logitboost) yöntemidir. Çoklu sınıf *logitboost* yöntemi, büyük boyutlu verilerde tanımlanması çok zor olduğundan, biri diğerlerine karşı *logitboost* yöntemi tercih edilmektedir. Bu yöntemde, tüm eğitim veri seti, her bir sınıf için sırayla iki gruba ayrılarak; tek tek sınıflara ait birer veri seti elde edilmektedir. Bir bağımlı kategorik değişkene ait k tane sınıf olduğunu varsayılarak k ikili *logitboost* sınıflandırıcılar (binary logitboost classifiers) oluşturulmaktadır. Her bir *logitboost* sınıflandırıcı için sınıflara ait test verisinin olasılığını hesapladıktan sonra, k ikili *logitboost* sınıflandırıcıları, sınıflandırma olasılığının bir vektörünü oluşturmaktadır:

$$P(\boldsymbol{x}) = [p_1(\boldsymbol{x}), p_2(\boldsymbol{x}), \dots, p_k(\boldsymbol{x})] \tag{68}$$

Test verilerinin en yüksek olasılığa sahip olduğu sınıf tahminlenir:

$$C(x) = argMax_i[p_i(x)] \tag{69}$$

Tüm sınıfların birbirinden ayrık (mutually exclusive) olduğu varsayılmaktadır. Böylelikle, *logitboost* algoritması çoklu sınıflar için ikili lojistik regresyon modelini hesaplayabilmektedir.

Sonuç olarak, tüm artırma algoritmalarında olduğu gibi *logitboost* algoritmasının da temel amacı, zayıf sınıflandırıcıları birleştirerek güçlü bir sınıflandırıcı elde etmektir. Bir zayıf sınıflandırıcı için hata oranı yeniden hesaplanarak ağırlıklar güncellenmektedir. Sonrasında, değişen ağırlıklarla yeni bir sınıflandırıcı tanımlanmaktadır. Bu işlem, kullanıcı tanımlı maksimum yineleme sayısına ulaşına kadar devam etmektedir. Daha sonra, final modeli için genel doğruluk oranı son yineleme sayısı için hesaplanmaktadır. Aynı zamanda, aşırı uyuma neden olan verilerin ağırlıkları da güncellendiği için, bu probleme de çözüm bulunmaktadır (J. Friedman et al., 2000).

3.9.4. Gradyan Artırma Makinaları

Gradyan artırma makinaları algoritması, 2001 yılında Friedman tarafından önerilmiş bir artırma algoritmasıdır (J. H. Friedman, 2001). *Gbm* algoritmasının temel amacı, AdaBoost algoritmasında olduğu gibi temel sınıflandırıcıları yeniden ağırlıklandırarak güncellemek yerine, bir önceki yinelemede elde edilen kayıp fonksiyonunu (loss function) negatif gradyan

vektörüne uydurmaktır. Gradyan, çok değişkenli bir fonksiyonun kısmi türev bilgisine sahiptir. Skaler değere sahip bir çok değişkenli fonksiyonun $f(x, y, \dots)$ gradyanı,

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \tag{70}$$

olarak ifade edilmektedir. Kısmi türev bilgisinin tamamı bu vektörde verilmektedir. Eğer fonksiyonun f bir girdi uzayında bir $(x_0, y_0, \dots)$ noktasında olduğu varsayılırsa, $\nabla f(x_0, y_0, \dots)$ vektörü f değerini en hızlı bir şekilde yükseltmesiyle hangi yöne gidileceği belirlenmektedir. Tüm gradyan vektörleri $\nabla f(x_0, y_0, \dots)$, f fonksiyonunun eş yükselti eğrilerine diktir. f fonksiyonunun girdileri iki boyutlu durumdan daha fazla olduğunda da f'in gradyanı girdi uzayında f fonksiyonunu en hızlı arttıran yönde bir vektör vermektedir. *Gbm* algoritması, fonksiyon uzayındaki en dik türevlenebilir fonksiyonların hatalarını en aza indirgemeyi başaracak şekilde tanımlanmıştır. Genel olarak, bağımlı değişken ile bağımsız değişkenler arasındaki ilişkinin istatistiksel model tahmin regresyon fonksiyonu olarak tanımlanmaktadır:

$$\hat{f}(.) = \underset{f(.)}{argmin}\{E_{Y,X}[\rho(Y, f(X))]\} \tag{71}$$

$\rho(.)$: türevlenebilir kayıp fonksiyonu

En yaygın kullanılan kayıp fonksiyonunun L_2 kaybı $\rho(y, f(.)) = (y - f(.)^2)$ ortalamasının en küçük kareler yöntemiyle elde edilen regresyon modeli, $f(x) = E(Y|X = x)$ olarak bulunmaktadır. Gözlemlerin eğitim veri setindeki $(y_1, x_1), \dots, (y_n, x_n)$ hatası,

$$\hat{f}(.) = argmin_{f(.)} = \left\{\frac{1}{n}\sum_{i=1}^{n} \rho(y_i, f(x_i))\right\} \tag{72}$$

olarak yeniden tanımlanmaktadır.

Gbm algoritmasında, her yinelemede permütasyon örnekleme yöntemi kullanılarak eğitim veri setinden rastgele alt örneklem seçilmektedir. Böylelikle, bu algoritmaya rastgelelik dahil edilerek doğru tahminleme olasılığının artması sağlanmaktadır. Seçilen bu alt örneklemler, tüm sınıflandırmalarda model güncellemesinin hesaplanması ve ağaçlar arasındaki korelasyonun azaltılması için kullanılmektedır. Bunların yanı sıra alt örneklemler sayesinde

aşırı uyum problemine karşı dayanıklılık artmaktadır. Her yinelemede (m) değerlendirilen kayıp fonksiyonu şu şekilde hesaplanmaktadır:

$$u^{[m]} = \left(u_i^{[m]}\right) = \left(-\frac{\partial}{\partial f}\rho(y_i, f)|f = \hat{f}^{[m-1]}(.)\right) \quad i = 1, \dots, n \tag{73}$$

Burada, $\rho\big(y, f(.)\big) = \frac{1}{2}\big(y - f(.)\big)^2$ eşitliği $y - f(.)$ artıklarından yeniden güncellenmesini sağlayarak, bir önceki yineleme olan $y - f^{[m-1]}$ ifadesine kadar devam etmektedir.

Burada dikkat edilmesi gereken durumlardan biri, tahminlenmesi zor olan gözlemlere odaklanarak yineleme ile değiştirme işlemi yoluyla temel sınıflandırıcıların başarı performanslarının arttırılmasıdır. AdaBoost algoritmasında bu işlem, daha öncesinde yanlış sınıflandırılmış gözlemlerin ağırlıkları arttırılarak yapılırken, *gbm* algoritmasında daha önceki yinemelerden hesaplanmış büyük artık değerlerine sahip gözlemler tanımlanarak yapılmaktadır (Natekin & Knoll, 2013).

Ağaç tabanlı topluluk öğrenme algoritması olan *gbm*, ağaç sayısı, yaprak sayısı, öğrenme oranı ve ağaçlardaki yaprak düğümlerin sayısı olmak üzere dört parametreye sahiptir. Öğrenme oranı parametresi, ne kadar küçük olursa, genelleme hatasının o kadar düşük olacağı varsayımından dolayı, genellikle bu değer 0,1 olarak ayarlanmaktadır (J. Friedman et al., 2000). Literatürde, öğrenme oranı 0,1'e eşitse ve 100 ile 500 arasında ağaç varsa, ağaçlardaki yaprak düğümlerinin sayısının 2 ile 8 arasında olabileceği belirtilmektedir (J. H. Friedman, 1999). Bunların yanı sıra, Friedman (2001) çalışmalarında, *gbm* algoritmasıyla kurulan karar ağaçlarının yaprak sayılarının parametre değerinin 1 ile 9 arasında olmasını önermiştir (J. H. Friedman, 2001).

Bu çalışmada, varsayılan olarak bağımlı değişkeni ikili olan veri setleri için lojistik regresyon kayıp fonksiyonu kullanılırken, çoklu kategorili olan veri setleri için çoklu kategorili lojistik regresyon kayıp fonksiyonunun kullanımı tercih edilmiştir.

3.10. k-katlı Çapraz Doğrulama Yöntemi

k-katlı çapraz geçerlilik yada k-katlı çapraz doğrulama (k-folds cross validation) olarak adlandırılan yönteminin temel mantığı, eğitim veri setini birbirinden farklı eğitim ve test veri setleri olmak üzere alt kümelerine ayırmaktır. Sonrasında, eğitim veri setlerini kullanarak ardışık ağaçlar elde edilmekte ve test veri setleri kullanılarak sınıflandırma model hataları tahminlenmektedir (Maimon & Rokach, 2005; Schaffer, 1993). Başka bir deyişle, eğitim veri

seti L öncelikle k tane eşit genişlikte parçalara $L_1, \dots, L_k$ ayrılır ve bu parçalar, her bir adım içinde farklı bir doğrulama için tutulmaktadır. Daha sonra, geriye kalan $k-1$ parça öğrenme için kullanılmaktadır. Bu işlem ardışık k kez tekrarlanmaktadır.

$$L^{(k)} = L - L_k \qquad k = 1, \dots, K \tag{74}$$

$L^{(k)}$: $k-1$ test veri seti alt küme parçası

L_k: L eğitim veri setinin k tane alt küme parçası

Örneğin, ilk adımda $L_2, \dots, L_k$ alt kümeleri ilk modelin kurulmasında eğitim veri setleri olarak kullanılmaktadır. Daha sonra, kurulan bu model L_1 alt kümesi ile test edilmektedir. İkinci adımda ise, $L_1, L_3 \dots, L_k$ alt kümeleri ikinci modelin kurulmasında eğitim veri setleri olarak kullanılmakta ve kurulan bu model benzer şekilde L_2 alt kümesi ile test edilmektedir. Bu süreç, 10 kez tekrarlanarak devam etmektedir. K-katlı çapraz geçerlilik yöntemi, diğer örnekleme tiplerinden farklı olarak her bir örneklem eğitim veri seti için aynı sayıda kullanılırken, test veri seti için sadece bir kez kullanılmaktadır. Breiman (1996), parça sayısı değerini Tablo 6'da gösterildiği gibi $k = 10$ olacak şekilde önermiştir (Breiman, 1996).

Tablo 6: k-katlı Çapraz Doğrulama

Parça	Adım1	Adım2	Adım3				Adım10
1	Test						
2		Test					
3			Test				
4				.	.	.	
5							
6							
7							
8							
9							
10							Test

Her bir parça için kurulan modelin performansı doğruluk gibi bir performans metriği ile ölçülmektedir. Sınıflandırma için doğruluk kestirimi, k adımdaki doğru sınıfladırmaların orijinal veri setinin örneklem genişliği sayısına bölünmesiyle elde edilmektedir. Diğer bir ifade ile, hata tahmini k adımda elde edilen toplam hatanın orijinal veri setinin örneklem sayısına bölünmesidir. Kestirim hatasının çapraz doğrulama tahmini aşağıdaki gibi hesaplanmaktadır:

$$\hat{R}_{min}^{CV} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i^{-L_k}\right) \tag{75}$$

$\hat{y}_i^{-L_k}$: L_k parçasının çıkarılmasıyla tahmin edilen modelin i. gözlemi için tahmin değeri

Doğru sınıflandırmanın 0, yanlış sınıflandırmanın 1 olarak kaydedildiği bu değerler, aynı zamanda sınıflandırma maliyetleridir. Daha sonrasında hesaplanan kestirim hatalarından en küçük olan k parçası, sınıflandırma ağacı olarak seçilmektedir (Witten et al., 2011). Bu çalışmada da 10-katlı çapraz doğrulama yöntemi ile 70/30'lük eğitim ve test veri ayırma oranı kullanılmıştır.

3.11. Sınıflandırmada Kullanılan Performans Ölçüm Metrikleri

Bu tez çalışmada kullanılacak topluluk öğrenme algoritmalarının performansları karşılaştırılırken; doğruluk (accuracy - ACC), duyarlılık / hassaslık (sensitivity / recall - SEN), seçicilik (specificity - SPE), kesinlik (precision - PRE), Kappa istatistiği (Kappa statistic - KAP), Youden indeksi (Youden index - YI), F - ölçütü (F-measure - F) ve ROC ölçüm metrikleri kullanılacaktır. Bu ölçüm metrikleri, hata matrisi (confusion matrix) adı verilen Tablo 7'ye göre hesaplanacaktır (Chawla et al., 2002). Performans ölçüm metrikleri 0 ile 1 arasında değer almaktadır. Ölçüm metrik değerlerinin 1'e çok yakın değerler alması, sınıflandırma başarısının yüksek olduğu anlamına gelmektedir. Tablo 7'ye göre, DP: doğru pozitifi, YP: yanlış pozitifi, YN: yanlış negatifi ve DN: doğru negatifi ifade etmektedir.

Tablo 7: Performans Ölçüm Metriklerinin Hesaplanması için Hata Matrisi

		Gerçek Durumu	
		Pozitif	**Negatif**
Tahmin Durumu	**Pozitif**	DP	YP
	Negatif	YN	DN

- **Doğruluk:** Bir testin doğruluğu, hasta ve sağlıklı bireyleri doğru olarak ayırt edebilme gücüdür (Swets, 1988). Tanı testinin doğruluğu hesaplanırken, tüm hasta ve sağlıklı bireyler için doğru pozitif ve doğru negatif oranı hesaplanmaktadır. Sınıflandırma doğruluğu ne kadar yüksek olursa, sistem o kadar iyi performans göstermektedir.

$$ACC = \frac{DP + DN}{DP + DN + YP + YN} \tag{76}$$

- **Duyarlılık:** Bir testin duyarlılığı, bir bireyin gerçekten hasta olduğu bildiğinde, test sonucunda da hasta sonucuna varılmasıdır (Maimon & Rokach, 2005). Hasta bireylerin doğru olarak ayırt edilmesi için doğru pozitif oranı (True Positive Rate - TPR) hesaplanmaktadır. Duyarlılık değeri aynı zamanda testin gücüne de eşittir ($power = 1 - \beta$). Duyarlılık, hassaslık değeri ile benzer formülle hesaplanmaktadır.

$$SEN = REC = \frac{DP}{DP + YN} \tag{77}$$

- **Seçicilik:** Bir testin seçiciliği, bir bireyin gerçekten hasta olmadığı bilindiği durumda, tanı testi sonucunun negatif çıkması olasılığı olarak tanımlanmaktadır (Maimon & Rokach, 2005). Sağlıklı bireylerin doğru ayırt edilmesi için doğru negatif oranı (True Negative Rate - TNR) hesaplanmaktadır.

$$SPE = \frac{DN}{DN + YP} \tag{78}$$

- **Kesinlik:** Bir testin kesinliği, tanı testinin pozitif kestirim değeri (Positive Predictive Value - PPV) veya pozitif kestirimlerin gerçekten pozitif olması durumda hesaplanmaktadır (Buckland & Gey, 1994). Başka bir ifade ile, tanı testi sonucu pozitif olan bir bireyin, hasta olma olasılığı olarak tanımlanmaktadır.

$$PRE = \frac{DP}{DP + YP} \tag{79}$$

- **Kappa istatistiği:** Gözlemciler arası güvenilirliği test etmek için sıklıkla kullanılan Kappa istatistiği, ilk olarak Cohen tarafından önerilen bir uyum ölçümü metriğidir (Cohen, 1960; McHugh, 2012). -1 ile 1 arasında değişen değerler alabilen Kappa istatistiğinin önceden belirlenmiş bir kabul edilebilirlik seviyesi olmamakla birlikte, 0,75 değerinden büyük olması mükemmel uyumu; 0,40 değerinden küçük olması zayıf uyumu ve 0,40 ile 0,75 arasındaki değerler alması ise kabul edilebilir uyumu

göstermektedir. p_o, gözlenen uyum oranı ve p_e, beklenen uyum oranı olarak alındığında κ ile gösterilmektedir.

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{80}$$

- **Youden indeksi:** Bu metrik, ROC eğrisi gibi tanı testinin etkinliğini ölçmede ve en uygun eşik değerinin belirlenmesinde kullanılan bir kriterdir (Fluss, Faraggi, & Reiser, 2005; Youden, 1950). Tanısal doğruluğun en sık kullanılan ölçütü olan ROC eğrisine rağmen, Youden indeksi de tercih edilmektedir. Youden indeksi, 0 ile 1 arasında değer almaktadır. Hasta ve sağlıklı bireylerin tam olarak ayrılması durumunda $J = 1$; tam örtüşme durumunda $J = 0$ değerini vermektedir.

$$J = max_c\{duyarlılık(c) + seçicilik(c) - 1\} \tag{81}$$

- **F-ölçütü:** Bu metrik, hassaslık ile kesinlik değerlerinin harmonik ortalaması alınarak hesaplanmaktadır (Buckland & Gey, 1994). Bu durumda, hem yanlış pozitif hem de yanlış negatif değerleri aynı anda dikkate alarak hesaplama yapmasından dolayı doğruluk olarak anlaşılması kolay değildir, ancak dengesiz bir sınıf dağılımı varsa F-ölçütü değerinin doğrulukla birlikte değerlendirilmesi önerilmektedir. Eğer, pozitif ve yanlış negatifler benzer maliyetlere sahipse doğruluk değerine bakmak; yanlış pozitiflerin ve yanlış negatiflerin maliyeti farklıysa, F-ölçütü değerine bakmak daha faydalıdır (Watson & Petrie, 2010).

$$F = 2 * \frac{\left(\frac{DP}{DP + YN}\right) * \left(\frac{DP}{DP + YP}\right)}{\left(\frac{DP}{DP + FN}\right) + \left(\frac{DP}{DP + YP}\right)} = 2 * \frac{REC * PRE}{REC + PRE} \tag{82}$$

- **ROC eğrisi:** Eğri altında kalan alan (Area Under the Curve - AUC) olarak da ifade edilen ROC eğrisi tanı testlerinin ve tahmin modellerinin değerlendirilmesi, uygun eşik değerinin ve testin ayırt ediciliğinin belirlenmesi, iki ya da daha fazla tanı testinin performanslarının karşılaştırılması gibi durumlarda kullanılan bir yöntemdir (Hajian-Tilaki, 2013). Eğri altında kalan alan, hasta ve sağlıklı bireylerde tanı koymada ne kadar bir ayırt etme gücüne sahip olunduğunu göstermektedir. ROC eğrisi, uygun eşik değerinin belirlenmesinde, tüm olası eşik değerleri (c) için duyarlılığa $duyarlılık(c))$ karşı $1 - seçicilik(c)$ değerlerine ait bir grafik vermektedir. Eğri altında kalan alan, 0 ile 1 arasında değer almaktadır ve kitlenin prevalansından etkilenmemektedir. Rastgele olarak bir tanı testinin alabileceği eğri atında kalan alan değeri 0,5 iken; mükemmel bir

tanı testinin doğruluğu 1 olarak ifade edilmektedir (Bradley, 1997; Maimon & Rokach, 2005).

4. Bulgular

Tezin bulgular kısmında, sekiz farklı veri setine ait kayıp gözlem yüzdesi, sınıf gürültüsü yüzdesi ve sınıf dengesizliği oranı verilmiştir. Veri ön işleme aşamalarına geçilmeden önce, orijinal veri setleriyle sınıflandırma modellerinin performansları hesaplanmıştır. Ardından, veri ön işleme sonrasındaki işlenmiş veri setlerinin performansları tekrar hesaplanarak sonuçlar karşılaştırılmıştır. Veri ön işleme aşamalarında kullanılan sınıf gürültüsü paketinin kayıp gözleme karşı duyarlı olması nedeniyle, ilk aşama olarak öncelikle kayıp gözleme sahip veri setlerine çoklu atama yöntemiyle gerekli atamalar yapılmıştır. İkinci aşama olarak gürültülü sınıflarla dengelenme işleme yapılmaması için sınıf gürültüsüne sahip veriler arındırılmıştır. Son aşamada ise, tüm veri setleri için sınıf dengesizliği problemi ortadan kaldırılarak, dengeli sınıflar elde edilmiş ve yeni veri setleri kaydedilmiştir. Daha sonra sınıflandırma aşamasına geçilerek, işlenmiş veri setleri ile en başarı sınıflandırma performans ölçüm metriklerine sahip olan algoritma/algoritmalar belirlenmiştir. Tüm bu işlemler sırasında algoritmaların çalışma süreleri kaydedilmiştir. Algoritmaların çalışma sürelerinin ilk aşaması, orijinal verilerin analizi, ikinci aşaması, veri ön işleme ve son aşama ise, işlenmiş verilerin analizidir.

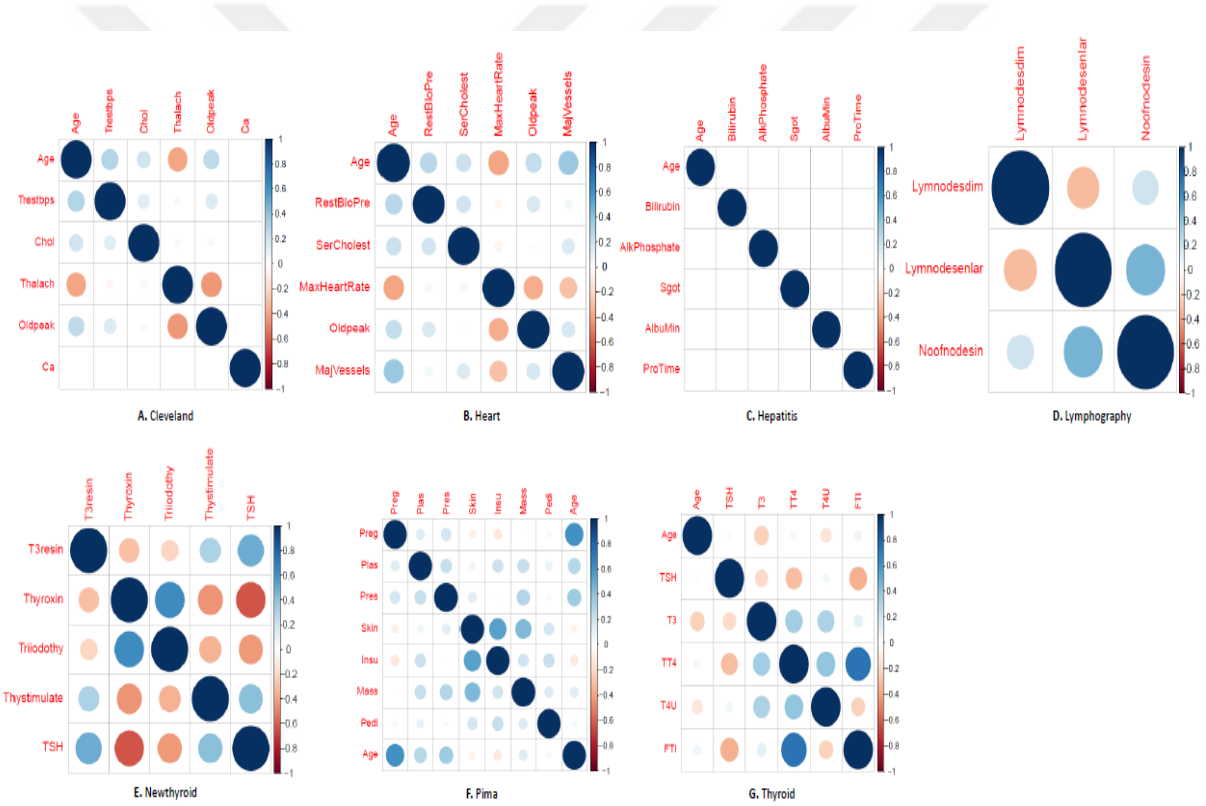
Tablo 8'de orijinal veri setleriyle ilgili genel bilgiler yer almaktadır. Örneklem genişlikleri, değişken sayıları, kayıp gözlem yüzdeleri, sınıf gürültüsü yüzdeleri ve sınıf dengesizliği oranları birbirinden farklı olan sekiz veri setinin; hem sürekli hem de kategorik değişken sayıları da benzer değildir. En az örneklem genişliğine sahip olan veri seti *Lymphography* verisi iken, en fazla örneklem genişliğine sahip olan veri seti *Thyroid* verisidir. Bunun yanı sıra, en az değişkene sahip olan veri setleri *Mammographic* ve *Newthyroid* verisi iken, en fazla değişkene sahip olan veri seti *Thyroid* verisidir. Özellikle *Cleveland, Heart, Hepatitis* ve *Lymphography* veri setlerinin, örneklem genişliklerinin az ve değişken sayılarının çok fazla olması dikkat çekmektedir. Bu durumda, örneklem genişliği ile değişken sayıları arasında bir boyut problemi söz konusudur. *Newthyroid* ve *Pima* veri setleri sadece sürekli değişkenlere sahipken, diğer veri setlerinde her iki değişken tipi de yer almaktadır. Burada yine önemli bir nokta, veri setlerinin sürekli ve kategorik değişken sayılarının birbirilerine göre farklı olmasıdır. *Hepatitis, Lymphography* ve *Thyroid* veri setlerinin kategorik bağımsız değişken sayıları, sürekli bağımsız değişken sayılarına göre oldukça fazladır. Orijinal veri setleri, bağımlı değişkenin sınıf düzeyleri açısından ele alındığında ise, *Heart, Hepatitis, Mammographic* ve *Pima* verileri iki kategoriye sahipken, çoklu sınıflara sahip veri setleri olan

Newthyroid ve *Thyroid* üç kategorili, *Lymphography* dört kategorili ve *Cleveland* beş kategorili sınıf düzeyine sahip verilerdir. Veri ön işleme aşamasında karşılaşılabilecek olası problemler açısından incelendiğinde ise *Cleveland, Hepatitis* ve *Mammographic* veri setleri, kayıp gözleme sahip verilerdir. En az kayıp gözleme sahip olan veri seti *Cleveland* verisi iken (%1,980), en fazla kayıp gözleme sahip olan veri seti ise *Hepatitis* verisidir (%48,387). En az sınıf gürültüsüne sahip veri seti *Thyroid* verisi iken (%0,056), en fazla sınıf gürültüsüne sahip veri seti ise *Cleveland* verisidir. Tüm veri setlerinde sınıf dengesizliği problemi vardır. En az sınıf dengesizliğine sahip olan veri seti *Mammographic* verisi iken (1,159), en fazla sınıf dengesizliğine sahip olan veri seti ise *Thyroid* verisidir (12,483). Burada dikkat edilmesi gereken ise, *Cleveland, Hepatitis* ve *Mammographic* verileri her üç probleme de sahip olma özelliğini taşımaktadır. *Heart, Newthyroid, Pima* ve *Thyroid* verileri hem sınıf gürültüsüne hem de sınıf dengesizliğine sahip veri setleridir. Sadece tek bir veri ön işleme problemini içeren veri seti olan *Lymphography* verisi sınıf dengesizliği problemine sahiptir. Tüm bu durumlar açısından orijinal veri setleri ele alındığında, veri setlerinin birbirilerine göre üstünlükleri ve farklılıkları ortaya çıkmaktadır.

Tablo 8: Veri Setlerine ait Genel Bilgiler

Veri seti	Örneklem genişliği	Değişken sayısı	Sürekli değişken sayısı	Kategorik değişken sayısı	Sınıf düzeyi sayısı	Kayıp gözlem yüzdesi	Sınıf gürültüsü yüzdesi	Sınıf dengesizliği oranı
Cleveland	303	13	5	8	5	1,980	19,802	1,179
Heart	270	13	6	7	2	-	4,074	1,250
Hepatitis	155	19	6	13	2	48,387	4,516	3,844
Lymphography	148	18	3	15	4	-	-	1,209
Mammographic	961	5	1	4	2	13,632	9,886	1,159
Newthyroid	215	5	5	-	3	-	2,791	2,308
Pima	768	8	8	-	2	-	7,943	1,866
Thyroid	7200	21	6	15	3	-	0,056	12,483

Cleveland, Heart, Hepatitis, Lymphograpy, Mammography, Newthyroid ve *Pima* verilerine ait bağımsız sürekli değişkenlerinin normal dağılım varsayımına ilişkin durumları Shapiro-Wilk testi ile sınanmıştır. *Thyroid* verisinin örneklem genişliği $n > 5000$ olduğundan, bu veri için Anderson-Darling normallik testi yapılmıştır. Bu sonuçlara göre, veri setlerine ait tüm bağımsız sürekli değişkenlerin normal dağılıma sahip olmadığı görülmüştür $(p < 0{,}05)$. Daha sonra bağımsız sürekli değişkenler arası korelasyon yapısının incelenmesinde Spearman'ın korelasyon katsayısı kullanılmıştır. Elde edilen sonuçlara göre, Şekil 17c'de, *Hepatitis* verisinin sürekli değişkenleri arasında ilişki yokken $(\rho = 0)$, diğer tüm verilere ait sürekli değişkenler arasındaki korelasyon değerlerinin büyük çoğunluğu $-0{,}6 \leq \rho \leq 0{,}5$ arasında değer almaktadır. Böylelikle, sürekli değişkenler arasında düşük düzeyde bir ilişki olduğu söylenebilmektedir.



Şekil 17: Veri Setlerine ait Bağımsız Sürekli Değişkenler Korelasyon Değerleri

Verilerin dağılımları ve korelasyon durumları incelendikten sonra, herhangi bir veri ön işleme yaklaşımı uygulanmadan ve uygulandıktan sonra, orijinal ve işlenmiş veri setleri analiz edilmiş, sonrasında kullanılan algoritmaların performans ölçüm metrikleri hesaplanmıştır.

4.1. Orijinal Veri Setlerinin Analizi

Öncelikle orijinal veri setleriyle sınıflandırma yapmak amacıyla farklı modeller kurulmuş ve kullanılan algoritmaların performans ölçüm metrikleri hesaplanmıştır. Tablo 9'da gösterildiği gibi, kurulan model geçerliliklerini test etmek için çalışmada yer alan tüm veri setleri %30 oranında test ve %70 oranında eğitim veri setleri olmak üzere ikiye ayrılmıştır.

Tablo 9: Orijinal Veri Setlerine ait Test ve Eğitim Veri Setleri

Veri setleri	Örneklem genişliği	Eğitim veri seti (% 70)	Test veri set (% 30)
Cleveland	303	215	88
Heart	270	189	81
Hepatitis	155	110	45
Lymphography	148	105	43
Mammographic	961	674	287
Newthyroid	215	151	64
Pima	768	538	230
Thyroid	7200	5042	2158

Orijinal veri setleri için her bir model açısından Tablo 10'daki değerlere göre model parametreleri ayarlanmıştır.

Tablo 10: Kullanılan Algoritmalara ait Parametre Değerleri

	Cleveland	**Heart**	**Hepatitis**	**Lymphography**	**Mammographic**	**Newthyroid**	**Pima**	**Thyroid**
rf								
ntrees	500	500	500	500	500	500	500	500
mtry	4	4	4	4	2	2	3	5
wsrf								
ntrees	500	500	500	500	500	500	500	500
mtry	4	4	5	5	3	3	4	5
logitboost								
nIter	1300	1300	1900	1800	500	500	800	2100
gbm								
n.trees	650	650	950	900	250	250	400	1050
interaction.depth	1, 5, 9	1, 5, 9	1, 5, 9	1, 5, 9	1, 5, 9	1, 5, 9	1, 5, 9	1, 5, 9
shrinkage	0,1	0,1	0,1	0,1	0,1	0,1	0,1	0,1
n.minobsinnode	10	10	10	10	10	10	10	10

Tüm orijinal veri setlerinde *rf* ve *wsrf* algoritmalarına ait ağaç sayısı $ntrees = 500$ olarak belirlenmiştir. Burada yine her bir bölünme için kullanılacak değişken sayısı $mtry$ parametresi, *rf* ve *wsrf* algoritmaları için farklı hesaplandığından, veri setlerine göre değişiklik göstermektedir. *Logitboost* algoritmasına ait artırma yineleme sayısı $nIter$ parametresi, veri setlerindeki bağımsız değişken sayısının 100 katı olacak şekilde ayarlanmıştır. gbm algoritması ait ağaç sayıları parametresi $n.trees$, bağımsız değişken sayının 50 katı olarak belirlenmiştir. Diğer parametreler olan yaprak sayıları $interaction.depth$, öğrenme oranı $shrinkage$ ve ağaçlardaki yaprak düğümlerin sayısı $n.minobsinnode$ olan üç parametre için literatürde varsayılan değerler kullanılmıştır. Bu çalışmada, en düşük büyüklükteki yaprak sayısı 1, orta büyüklükteki yaprak sayısı 5 ve en yüksek büyüklükteki yaprak sayısı 9 olarak belirlenmiştir. Diğer yandan, ağaçlardaki yaprak düğümlerin sayısına ait parametreyi sabitlemek açısından her veri seti için bu değer 10 olarak ayarlanmıştır. Bu parametre değerlerine göre orijinal veri setleriyle kurulan her bir modelin performans değerleri Tablo 11'de verilmiştir. Tablo 11'de koyu olarak belirtilen sonuçlar, orijinal veri setlerine ait model performans ölçüm metriklerinin en iyi değerlerini ifade etmektedir.

Cleveland verisinin tüm algoritma modelleri, performans ölçüm metrikleri açısından değerlendirildiğinde, modelleme performans başarıları çok düşük sonuçlar vermiştir. Bunun nedenlerinden biri, örneklem genişliğinin az olmasıdır. Diğer nedenler arasında, %1,98 kayıp gözleme sahip olması, %19,802 sınıf gürültüsüne sahip olması ve 1,179 oranında sınıf dengesizliğine sahip olması yer almaktadır. Her dört algoritmanın da ölçüm metrikleri olabildiğince birbirine yakın değerler vermesine rağmen, bu değerler model başarıları açısından değerlendirildiğinde yeterli sayılmamıştır. Bu arada artırma algoritmalarında ROC eğrisi değeri hariç (%0,802) diğer tüm ölçüm metriklerinde benzer sonuçlar vermiştir. Bundan dolayı en iyi sınıflandırma başarısına sahip algoritmalar, *logitboost* ve *gbm* algoritmalarıdır.

Heart verisi, genel anlamda tüm ölçüm metriklerinde yüksek değerler vermesine rağmen *logitboost* algoritmasına ait Kappa istatistiği (%0,631) ve Youden indekisi (%0,627) biraz daha düşük sonuçlar vermiştir. Diğer yandan, *rf* ve *gbm* algoritmaları aynı duyarlılık değerine (%0,933) sahiptir. Aynı zamanda, *Heart* verisi, %4,074 sınıf gürültüsüne ve 1,25 oranında sınıf dengesizliği oranına sahip bir veridir. *Heart* verisinin en iyi sınıflandırma algoritması, *rf* algoritmasıdır.

Hepatitis verisinin ölçüm metriklerinin genel olarak düşük sonuçlar vermesinin nedenleri arasında %48,387 kayıp gözleme sahip olması, %4,516 sınıf gürültüsüne sahip olması ve 3,844 oranında sınıf dengesizliğine sahip olması yer almaktadır. Özellikle, doğruluk ve seçicilik değerleri dışındaki tüm ölçüm değerleri oldukça düşük sonuçlar vermiş olmasından anlaşılacağı gibi, tek bir ölçüm değerine bakarak genelleme yapmak doğru olmamaktadır. Bu sonuçlara göre, *wsrf* ve *gbm* algoritmaları birbirilerine çok yakın değerler vediğinden, her iki algoritma da en başarılı sınıflandırma algoritmaları olarak seçilmiştir. Diğer yandan, ROC eğrisi değeri (%0,813) en yüksek algoritma, *rf* algoritmasıdır.

Lymphography verisi de *Hepatitis* verisi gibi oldukça düşük sınıflandırma başarılarına sahip olan bir veri setidir fakat *Lymphography* verisi sadece 1,209 oranında sınıf dengesizliğine sahiptir. Tüm ölçüm değerleri açısından en iyi sınıflandırma başarına sahip algoritma, *logitboost* algoritması modeli olmasına rağmen, ROC eğrisi değeri (%0,857) en yüksek algoritması, *rf* algoritması olmuştur.

Tablo 11: Orijinal Veri Setlerine ait Model Performans Ölçüm Metrikleri Değerleri

Veri Seti	Algoritma	ACC	SEN	SPE	PRE	Kappa	F	YI	ROC
Cleveland	rf	0,568	0,244	0,854	0,238	0,216	0,241	0,099	0,766
	wsrf	0,579	0,264	0,857	0,289	0,235	0,277	0,121	**0,802**
	logitboost	0,575	**0,351**	0,866	0,355	0,276	**0,353**	**0,217**	0,782
	gbm	**0,591**	0,293	**0,878**	**0,426**	**0,305**	0,347	0,169	0,777
Heart	rf	**0,926**	**0,933**	**0,917**	**0,933**	**0,850**	**0,933**	**0,850**	**0,961**
	wsrf	0,877	0,889	0,861	0,889	0,750	0,889	0,750	0,949
	logitboost	0,817	0,892	0,735	0,786	0,631	0,835	0,627	0,909
	gbm	0,914	**0,933**	0,889	0,913	0,825	0,923	0,822	0,949
Hepatitis	rf	0,800	0,444	0,889	0,500	0,348	0,471	0,333	**0,813**
	wsrf	**0,867**	0,556	**0,944**	**0,714**	**0,546**	0,625	0,500	0,796
	logitboost	0,800	0,556	0,861	0,500	0,400	0,526	0,417	0,741
	gbm	0,844	**0,667**	0,889	0,600	0,533	**0,632**	**0,556**	0,756
Lymphography	rf	0,767	0,385	0,881	0,382	0,529	0,384	0,266	**0,857**
	wsrf	0,721	0,365	0,861	0,357	0,442	0,361	0,226	0,843
	logitboost	**0,786**	**0,401**	**0,895**	**0,391**	**0,577**	**0,396**	**0,296**	0,528
	gbm	0,767	0,389	0,892	0,389	0,547	0,389	0,281	0,836
Mammographic	rf	0,843	0,864	0,819	0,847	0,684	0,855	0,683	0,889
	wsrf	0,822	0,844	0,797	0,828	0,642	0,836	0,641	0,868
	logitboost	**0,878**	**0,909**	**0,844**	**0,866**	**0,755**	**0,887**	**0,753**	0,865
	gbm	0,843	0,870	0,812	0,843	0,684	0,856	0,682	**0,895**
Newthyroid	rf	**0,922**	**0,826**	**0,912**	**0,967**	**0,813**	**0,891**	**0,738**	0,881
	wsrf	0,859	**0,826**	0,899	0,811	0,701	0,819	0,725	0,857
	logitboost	0,891	0,811	0,900	0,882	0,749	0,845	0,711	**0,894**
	gbm	0,859	0,767	0,876	0,858	0,677	0,810	0,643	0,844
Pima	rf	0,761	0,562	0,867	0,692	0,449	0,621	0,429	0,838
	wsrf	0,752	0,575	0,847	0,667	0,436	0,617	0,422	0,832
	logitboost	0,748	**0,588**	0,828	0,635	0,424	0,611	0,416	0,742
	gbm	**0,778**	0,538	**0,907**	**0,754**	**0,476**	**0,628**	**0,445**	**0,851**
Thyroid	rf	0,995	0,986	0,992	0,959	0,964	0,972	0,978	0,921
	wsrf	0,997	**0,996**	**0,997**	0,963	0,977	0,979	**0,993**	0,833
	logitboost	**0,998**	0,972	0,988	0,958	0,959	0,965	0,961	0,815
	gbm	**0,998**	0,993	**0,997**	**0,981**	**0,984**	**0,987**	0,989	**0,969**

ACC: Doğruluk, SEN: Duyarlılık, SPE: Seçicilik, PRE: Hassaslık, Kappa: Kappa istatistiği, F: F-ölçütü, YI: Youden indeksi, ROC: ROC eğrisi

Mammographic verisi, tüm modeller açısından genel anlamda yüksek bir sınıflandırma başarısına sahip algoritmadır. Buna rağmen *Mammographic* verisi, %13,632 kayıp gözleme sahip olması, %9,886 sınıf gürültüsüne sahip olması ve 1,159 oranında sınıf dengesizliğine sahip olması nedeniyle, modellerin sınıflandırma başarıları yeterli değildir. *Mammographic* verisi için en başarı sınıflandırma algoritmasına sahip model, *logitboost* algoritması olmasına rağmen, ROC eğrisi değeri (%0,895) en yüksek algoritma, *gbm* algoritmasıdır.

Newthyroid verisi, genel olarak tüm algoritmalarda yüksek sınıflandırma başarısına sahip olsa da %2,791 sınıf gürültüsüne ve 2,308 oranında sınıf dengesizliğine sahiptir. Diğer tüm algoritmalarda yaklaşık olarak benzer sonuçlar vermesine rağmen, en iyi sınıflandırma başarısı *rf* algoritmasına aittir. Fakat ROC eğrisi değeri açısından (% 0,894) en yüksek değere sahip algoritma, *logitboost* algoritmasıdır.

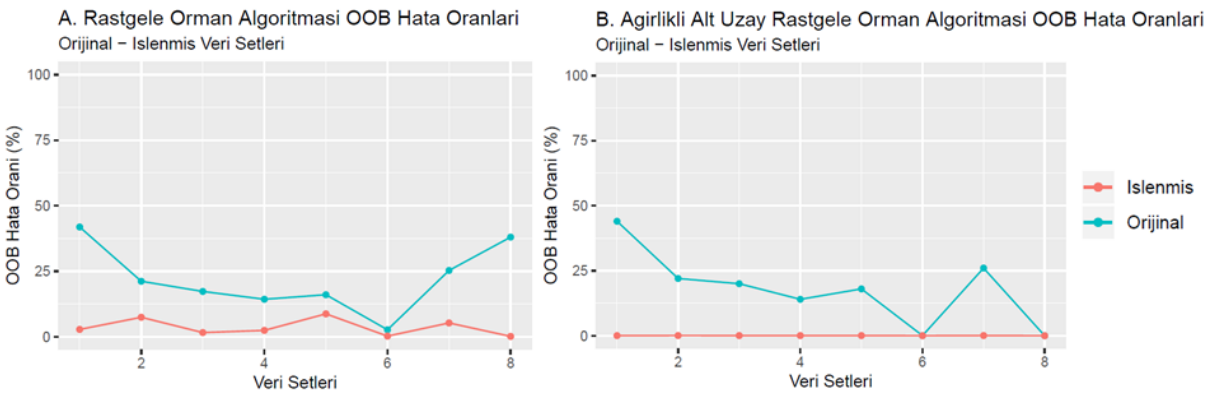
Pima verisi, seçicilik ve ROC eğrisi değerleri hariç diğer tüm ölçüm metriklerinde oldukça düşük değerler vermiştir. Bunun olası sebepleri olarak *Pima* verisinin, %7,943 sınıf gürültüsüne ve 1,866 oranında sınıf dengesizliğine sahip olması gösterilebilmektedir. Tüm bu durumlar göz önünde bulundurulduğunda ve diğer algoritmalar da yaklaşık olarak benzer sonuçlar vermesine rağmen, en iyi sınıflandırma başarısı *gbm* algoritmasına aittir.

Thyroid verisi, tüm algoritmalar açısından oldukça yüksek sonuçlar vermesine rağmen, %0,056 sınıf gürültüsüne ve 12,483 oranında sınıf dengesizliğine sahiptir. *Thyroid* verisi için en iyi başarılı sınıflandırma performansına sahip algoritma, *gbm* algoritmasıdır. Şekil 22'de ifade edildiği gibi, orijinal verilere ait tüm performans ölçüm metrikleri %0,20 ile %0,99 arasında değerler almaktadır. Veri ön işlemeden sonra bu değer aralığının daha da daralması beklenmiştir. Böylelikle, model performanslarındaki iyileştirmelerin daha açık gözlemlenmesi sağlanmıştır. Orijinal veri setleri için kurulan modellerin en iyi parametre değerleri ve bu modellere ait bazı sonuçlar Tablo 12'de verilmiştir.

Tablo 12: Orijinal Veri Setlerine ait Modellerin En İyi Parametre Değerleri ve Sonuçları

	Cleveland	**Heart**	**Hepatitis**	**Lymphography**	**Mammographic**	**Newthyroid**	**Pima**	**Thyroid**
rf								
OOB hata oranı	% 41,86	% 21,16	% 17,27	% 14,29	% 16,02	% 2,65	% 25,28	% 38
wsrf								
OOB hata oranı	% 44	% 22	% 20	% 14	% 18	% 0,05	% 26	% 0,00
strength	0,20	0,37	0,49	0,58	0,55	0,89	0,39	0,99
korelasyon	0,31	0,20	0,22	0,18	0,28	0,07	0,24	0,01
logitboost								
nIter	1300	1300	1900	1800	500	500	800	2100
gbm								
n.trees	50	50	100	200	100	250	50	180
interaction.depth	1	1	1	5	1	9	1	5
shrinkage	0,1	0,1	0,1	0,1	0,1	0,1	0,1	0,1
n.minobsinnode	10	10	10	10	10	10	10	10

Tüm bu sonuçlara göre, *Cleveland* verisi için kurulan *rf* algoritması test veri seti tahmin hata oranı (%41,86), en yüksek değere sahiptir. *Newthyroid* verisi için kurulan *rf* algoritması test veri seti tahmin hata oranı (%2,65) ise en düşük değere sahiptir. Buna göre, *Cleveland* verisinin model tahmini oldukça düşükken, *Newthyroid* verisinin model tahmininin yüksek olduğu söylenebilmektedir. Buna bağlı olarak, modellerin performans ölçüm metriklerine ait değerler de bu sonuçları desteklemektedir. *Wsrf* algoritmasına ait OOB hata oranı *Cleveland* verisinde %44 oranı ile en yüksek değere sahipken, en düşük OOB hata oranına sahip veri *Thyroid* verisidir. Şekil 18a'da ve Şekil 18b'de gösterildiği gibi, orijinal verilere ait OOB hata oranları hem *rf* algoritmasında hem de *wsrf* algoritmasında %0 ile %50 arasında değer almaktadır. Şekil 18a ve Şekil 18b'de, *Cleveland* veri 1, *Heart* veri 2, *Hepatitis* verisi 3, *Lymphography* verisi 4, *Mammography* verisi 5, *Newthyroid* verisi 6, *Pima* verisi 7 ve *Thyroid* verisi 8 ile ifade edilmiştir.



Şekil 18: Rastgele Orman ve Ağırlıklı Alt Uzay Rastgele Orman OOB Hata Oranları

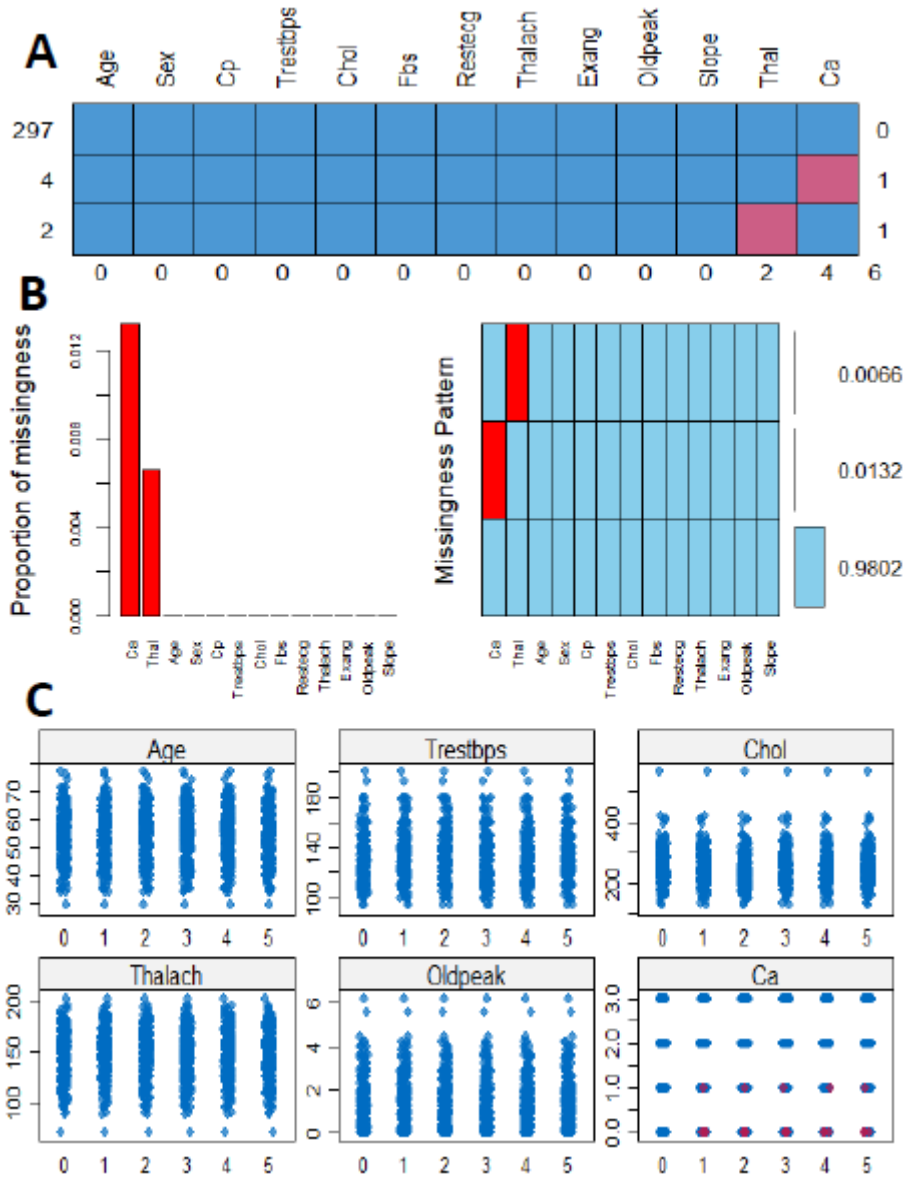
Wsrf algoritmasına özgü bir sonuç olan *strength* değeri, en yüksek değeri 0,99 oranla *Thyroid* verisine aitken, en düşük değeri 0,20 oranla *Cleveland* verisine aittir. Buna göre, *Thyroid* verisine ait *wsrf* algoritması modeli için kurulan bireysel rastgele orman ağaçlarının performanslarının toplamda çok başarılı olduğu anlamına gelmektedir. *Wsrf* algoritmasına ait diğer bir sonuç ise, *wsrf* modelindeki ağaçlarının çeşitliliği ölçümünü sağlayan korelasyon değeridir. *Wsrf* modeline ait en yüksek değer 0,31 oranla *Cleveland* verisine aitken, en düşük değer 0,01 oranla *Thyroid* verisine aittir. Genel olarak tüm bu sonuçlar göz önüne alındığında, *Tyriod* verisi için kurulan *wsrf* algoritması modeli sınıflandırma başarısı yüksek bir algoritma olmuştur. *Logitboost* algoritması modeli en iyi parametre değerini, en yüksek artırma yineleme sayısında almıştır. Bağımsız değişken sayısı en fazla olan *Tyhroid* verisi en yüksek artırma yineleme sayısına sahip olduğundan, en iyi modeli 2100. yinelemede kurmuştur. *Mammographic* ve *Newthyroid* veri setleri en az ve aynı bağımsız değişken sayılarına sahip olduğundan en iyi modeli 500. yinelemede kurmuştur. *Gbm* algoritması modeline ait parametre değerlerine bakıldığında, öğrenme oranı *shrinkage* ve ağaçlardaki yaprak düğümlerin sayısı $n.minobsinnode$ parametreleri tüm veri setleri için sabit tutulmuştur. Bu nedenle diğer parametrelerle birlikte en iyi modeller kurulmuştur. Tüm orijinal veri setleri için en iyi *gbm* algoritma modeli en düşük 50. ve en yüksek 250. ağaçta kurulmuştur. *Gbm* algoritmasının diğer parametresi olan yaprak sayıları parametresi için *Newthyroid* verisinin en iyi *gbm* algoritması modeli 9 final yaprak düğümüne sahipken, *Lymphography* ve *Thyroid* veri setleri için en iyi *gbm* algoritması modeli 5 final yaprak düğümüne sahiptir. Diğer tüm orijinal veri setleri tek bir final yaprak düğümüne sahiptir.

4.2. Veri Ön İşleme: Kayıp Gözlem, Sınıf Gürültüsü ve Sınıf Dengesizliği

Tüm bu sonuçlar orijinal veri setleri için elde edildikten sonra veri ön işleme aşamasına geçilmiştir.

1. **Aşama:** Kayıp gözlem ataması

Veri ön işleme aşamasında, öncelikle kayıp gözleme sahip *Cleveland, Hepatitis* ve *Mammographic* veri setleri için MAR varsayımı altında çoklu atama yaklaşımı kullanılarak, kayıp gözlem ataması yapılmıştır. Tüm veri setlerinin atama modelleri için, çoklu atama sayısı parametresi $m = 5$ ve yineleme sayısı $maxit = 10$ olarak ayarlanmıştır. Çoklu atama modelleriyle simüle edilerek elde edilen olası atama değerleri 5 farklı tamamlamış veri setinde kaydedilmiştir. *Cleveland* verisine ait kayıp gözlem atama sonuçları Şekil 19c'de gösterildiği gibidir. *Cleveland* veri setinde, *Ca* ve *Thal* bağımsız değişkenleri kayıp gözleme sahip değişkenlerdir. *Ca* değişkeninde 4 (%0,0132) ve *Thal* değişkeninde ise 2 (%0,0066) kayıp gözlem bulunmaktadır. Genel olarak, Cleveland veri seti %1,980 (6) kayıp gözlem yüzdesine sahiptir (Tablo 8). Tablo 19b'de, eksikliğin hangi değişkenlerde meydana geldiği ve bu değişkenlerin birlikte ne kadar bir kayıp gözlem yüzdesine sahip olduğu konusunda bilgi verilmektedir. *Cleveland* veri setinde bulunan kayıp verilerin oranları Şekil 19b'de verilmiştir. Buna göre, veri seti %0,0198 oranında kayıp gözleme sahipken, bu oranın % 0,0132 oranı *Ca* değişkeninden ve % 0,0066 oranı *Thal* değişkeninden kaynaklanmaktadır. *Ca* değişkeni sürekli bir değişken olduğundan, atama modeli varsayılan *pmm* modeli olarak ayarlanmıştır. *Thal* değişkeni çoklu kategorik bir değişken olduğundan, atama modeli varsayılan *polyreg* modeli ayarlanmıştır. Şekil 19a'da 0, kayıp gözlem yokluğunu; 1, kayıp gözlem varlığını ifade etmektedir.

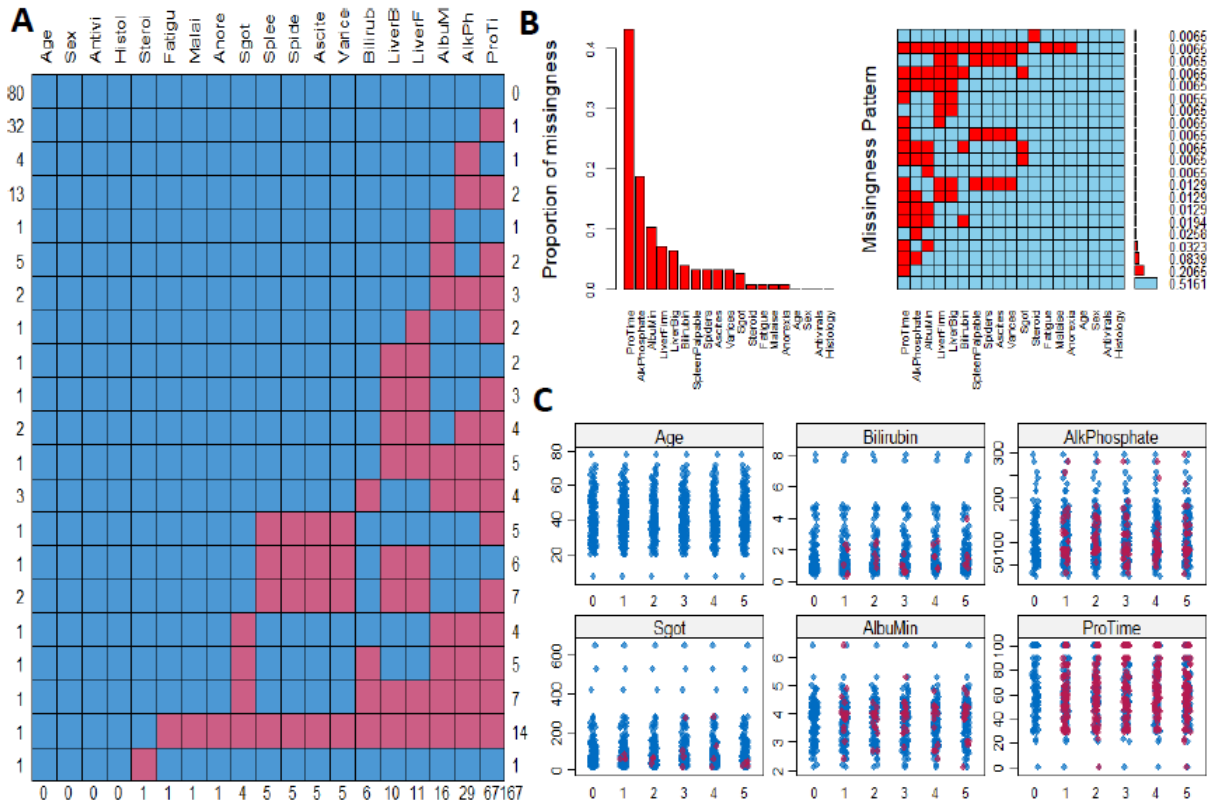


Şekil 19: Cleveland Verisi Kayıp Gözlem Oranları ve Atama Verileri

Sürekli değişkenler için yapılan olası atama değerlerine bakıldığında, her bir 5 farklı atama sonucunda veri setleri için benzer sonuçlar elde edildiği Şekil 19c'de gösterilmiştir. Bu sonuçlara göre *Cleveland* tamamlanmış veri seti, 5. atama sonucu elde edilen olası atama değerleri olmasına karar verilmiştir. Daha sonra, eksik değerleri tamamlanmış veri seti, yeni bir veri seti olarak kaydedilmiştir.

Hepatitis verisine ait kayıp gözlem atama sonuçları Şekil 20'de gösterildiği gibidir. *Hepatitis* veri setinde, kategorik değişkenlerden *Steroid* 1 (%0,645), *Fatigue* 1 (%0,645), *Malaise* 1 (%0,645), *Anorexia* 1 (%0,645), *LiverBig* 10 (%6,452), *LiverFirm* 11 (%7,097),

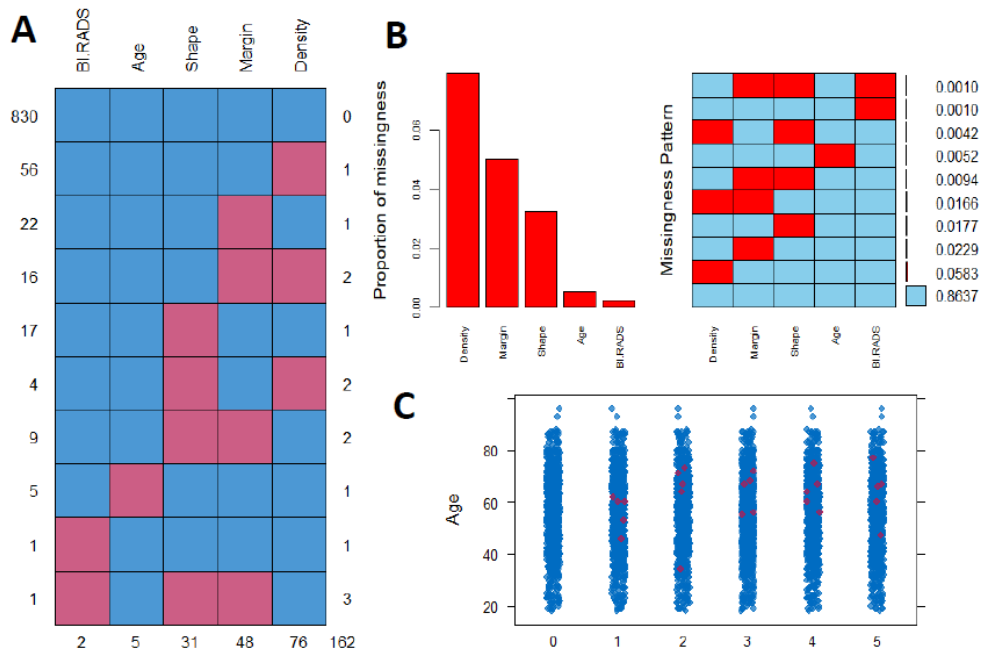
SpleenPalpable 5 (%3,226), *Spiders* 5 (%3,226), *Ascites* 5 (%3,226) ve *Varices* 5 (%3,226) kayıp gözleme sahiptir. Sürekli değişkenlerden *Bilirubin* 6 (%3,871), *AlkPhosphate* 29 (%18,710), *Sgot* 4 (%2,581), *AlbuMin* 16 (%10,323) ve *ProTime* 67 (%43,226) kayıp gözleme sahiptir. Genel olarak, *Hepatitis* veri setinde %48,387 (167) kayıp gözlem yüzdesi hesaplanmıştır (Şekil 20a). *Hepatitis* veri setinde bulunan kayıp verilerin oranları Şekil 20b'de verilmiştir. Buna göre, *ProTime* değişkeni tek başına kayıp gözlemin %0,2065 oranına sahipken, *ProTime* ve *AlkPhosphate* değişkenlerinin ikisi birlikte %0,2065 oranında kayıp gözleme sahiptir. Diğer durumlar için de aynı şekilde yorumlanabilmektedir. *Hepatitis* veri setinin tamamlaması için kullanılan çoklu atama modelleri değişken tiplerine göre belirlenmiştir. Kategorik değişkenler ikili sınıflara sahip oldukları için, atama modelleri *logreg* olarak ayarlanmıştır. Sürekli değişkenler için atama modelleri, varsayılan *pmm* olarak ayarlanmıştır. *Hepatitis* veri setinde bulunan tüm bağımsız değişkenler için atama veri setlerinin görselleştirmesinde sadece bazı bağımsız değişkenlerin olası atama değerlerinin yer verildiği Şekil 20c'den yararlanılmıştır. Şekil 20a'da 0, kayıp gözlem yokluğunu ifade etmektedir.



Şekil 20: Hepatitis Verisi Kayıp Gözlem Oranları ve Atama Verileri

Bu sonuçlara göre, *Hepatitis* verisi için elde edilen olası atamalarda veri setlerinin benzer sonuçlar verdiği görülmektedir. Buna göre, Hepatitis tamamlanmış veri setlerinden 5. atama uygun bulunarak yeni veri kaydedilmiştir.

Mammographic verisine ait kayıp gözlem atama sonuçları Şekil 21'de gösterildiği gibidir. *Mammographic* veri setindeki tüm bağımsız değişkenler kayıp gözleme sahiptir. Kategorik değişkenlerden *BI.RADS* 2 (%0,208), *Shape* 31 (%3,226), *Margin* 48 (%4,995), *Density* 76 (%7,908) ve sürekli değişken olan *Age* 5 (%0,503) kayıp gözleme sahiptir. Genel olarak, *Mammographic* veri setinde %13,632 (162) kayıp gözlem yüzdesi bulunmaktadır (Şekil 21a). *Mammographic* veri setinde bulunan kayıp verilerin oranları Şekil 21b'de verilmiştir. Buna göre, *Density* değişkeninde tek başına % 0,0583 oranında kayıp gözlem varken, *Density* ve *Margin* değişkenlerinin ikisinde birden %0,0166 oranında kayıp gözlem yer almaktadır. Benzer şekilde, *Margin, Shape* ve *BI.RADS* değişkenleri birlikte %0,001 oranında kayıp gözleme sahiptirler. Diğer tüm durumlar için benzer yorumlar yapılabilmektedir. *Mammographic* veri seti için atama modelleri belirlenirken sıralı çoklu kategorik değişken olan *BI.RADS* ve *Density* değişkenleri için *polr* atama modeli ayarlanmıştır. *Shape* ve *Margin* değişkenleri çoklu kategorik değişkenler olduğundan, atama modeli varsayılan *polyreg* olarak ayarlanmıştır. *Age* değişkeni sürekli bir değişken olduğu için atama modeli varsayılan *pmm* olarak ayarlanmıştır. Şekil 21a'da 0, kayıp gözlem yokluğunu ifade etmektedir.



Şekil 21: Cleveland Verisi Kayıp Gözlem Oranları ve Atama Verileri

Mammographic veri için olası atama değerleri değerlendirilirken, sadece tek bir sürekli değişken olan *Age* değişkenine ait veri setleri ele alınmıştır (Şekil 21c). Diğer veri setlerinde olduğu gibi tüm atanmış veri setlerinin de benzer sonuçlar verdiği gözlemlenmiştir. Buna bağlı olarak *Mammographic* tamamlanmış veri setlerinden 5. atama uygun bulunmuş ve yeni veri kaydedilmiştir.

Cleveland, *Hepatitis* ve *Mammographic* veri setlerine ait kayıp gözlem problemi ortadan kaldırılmış ve böylelikle ilk aşama tamamlanmıştır. Atama parametresinin kaç olması gerektiğiyle ilgili olarak farklı atama ve yineleme değerlerinde benzetimler yapılabilmektedir. Diğer yandan hangi modellerin atamada kullanıldığı da bu parametrenin belirlenmesinde etkili olabilmekte veya tam tersi durumlarda geçerli olabilmektedir. Bu nedenle *MICE* paketiyle kayıp gözlem ataması yapılacaksa, öncelikle her değişken tipine göre bir atama modeli belirlenmelidir. Sonra bu modellerden elde edilen olası atama değerlerinin kaydedildiği tamamlanmış veri setlerinden birine karar verilmelidir. Daha sonra, tamamlanmış atama veri seti, yeni veri seti olarak kaydedilmelidir.

İlk veri ön işleme aşaması olan kayıp gözlem ataması tamamlanarak sınıf gürültüsü aşamasına geçilmiştir.

2. **Aşama:** Sınıf gürültüsünün filtrelenmesi

Bu çalışma kapsamında sadece sınıf gürültüsüne sahip veri setleri ele alınmıştır. *Lymphography* veri seti hariç diğer tüm veri setlerinde sınıf gürültüsü problemi mevcuttur. Sınıf gürültüsü probleminin çözümü için kullanılan *NoiseFiltersR* paketi içerisinden *hybridRepairFilter* fonksiyonu kullanılmıştır. Bu fonksiyon sayesinde sınıf gürültüsüne sahip gözlemler veri setlerinden çıkarılmadan, gözlemlerin yeniden etiketlenmesi sağlanmıştır. Sınıf gürültülerinin yeniden etiketlenmesinde çoğunluk oyu yöntemi kullanılmıştır. Tablo 8'de gösterildiği gibi, *Cleveland* verisinde 60 (%19, 802), *Heart* verisinde 11 (%4,074), *Hepatitis* verisinde 7 (%4,516), *Mammographic* verisinde 95 (%9,886), *Newthyroid* verisinde 6 (%2,791), *Pima* verisinde 61 (%7,943) ve *Thyroid* verisinde 4 (%0,056) gözlemde sınıf gürültüsü tespit edilmiştir. Bu veri setleri sınıf gürültüsünden arındırıldıktan sonra yeni veri setleri olarak kaydedilmiştir.

Sınıf gürültüsü filtrelenmesi veri ön işlemesi tamamlandıktan sonra sınıfların dengelenmesi aşamasına geçilmiştir.

3. **Aşama:** Sınıfların dengelenmesi

Tüm orijinal veri setleri, sınıf dengesizliği problemine sahiptirler. Bu problemin çözümünde *caret* ve *DMwR* paketleri birlikte kullanılarak *SMOTE* fonksiyonu ile sınıf dengesizliği problemi çözülmüştür. Bu çalışmada *SMOTE* fonksiyonu için *k* parametresi, bağımsız değişken sayısının karekökü olarak ayarlanmıştır. *Perc.over* ve *perc.under* parametreleri ise, en uygun sınıf dengesini verecek şekilde ayarlanarak benzetim yapılmıştır. Bu sonuçlara göre, azınlık ve çoğunluk sınıfları dengelendikten sonra elde edilen yeni sınıf dağılımları Tablo 13'de ifade edildiği gibidir. Sınıf dağılımları yüzdeleri sınıflar arasında yakın olacak şekilde ayarlandığından, bazı verilerde orijinal veri setlerine göre örneklem genişliklerinde artma olurken bazılarında azalma görülmüştür. Veri setlerindeki sınıf dağılımlarının 10 katından fazla bir örneklem genişliği elde edilmeye çalışıldığında, bazı veri setlerinde kayıp gözlemler meydana gelmektedir. Buna örnek olarak, *Lymphography* ve *Thyroid* verileri verilebilmektedir. Sınıf dağılımlarının ayarlanmasında, sınıf dengesizliği oranı *(imbalanced ratio - IR)* kontrol edilerek, $IR \leq 1$ durumunda dengeli veriler elde edilmiştir. Tüm orijinal veri setleri için sınıflar dengelendikten sonra elde edilen yeni örneklem genişlikleri Tablo 13'de verilmiştir.

Tablo 13: Dengeli Sınıfların Örneklem Genişlikleri ve Sınıf Dengesizliği Oranları

Veri seti	Örneklem genişliği	Sınıf dengesizliği oranı
Cleveland	871	0,906
Heart	654	1,000
Hepatitis	270	1,000
Lymphography	235	0,880
Mammographic	2652	1,000
Newthyroid	650	0,806
Pima	1386	1,000
Thyroid	3425	0,523

Bu sonuçlara göre, *Thyroid* verisi hariç diğer tüm veri setlerinin örneklem genişliklerinde artış meydana gelmiştir. *Thyroid* veri seti, en uygun dengeleme parametreleri ayarlanmasına rağmen 7200 örneklem genişliğine sahip orijinal verinin neredeyse yarısı kadar bir örneklem büyüklüğüne sahip olmuştur. Bu bir anlamda bilgi kaybına yol açmış gibi görünse bile aslında

SMOTE algoritması bu soruna karşı dayanıklı bir algoritma olduğu için, bu durumun üstesinden gelmiştir. Aynı şekilde, diğer veri setlerinde örneklem genişliklerinin artmasından kaynaklı aşırı uyum sorununa neden olmamıştır. Tüm verilerin veri ön işlemeden önce sahip oldukları sınıf dengesizliği oranlarında düşüşler meydana geldiğinden, sınıf dağılımları arasında daha dengeli bir yapı elde edilmiştir. Tablo 8'e göre *Thyroid* verisi 12,483 oranla en yüksek sınıf dengesizliği oranına sahip olan veri seti iken, veri ön işlemeden sonra 0,523 oranla en düşük sınıf dengesizliği oranı sahip olan veri seti olmuştur. Aynı şekilde, diğer tüm verilerin sınıf dengesizlik oranları, *Cleveland* verisi 1,179'dan 0,906'a, *Heart* verisi 1,25'ten 1'e, *Hepatitis* verisi 3,844'den 1'e, *Lymphography* verisi 1,209'dan 0,88'e, *Mammographic* verisi 1,159'dan 1'e, *Newthyroid* verisi 2,308'den 0,806'ya ve *Pima* verisi 1,866'dan 1'e düşmüştür.

Tablo 14'de sınıflandırmada kullanılan her bir veri setinin bağımlı değişken düzeyleri bazında üç farklı durum için sınıf dağılımları ele alınmıştır. İlk durum, tüm orijinal veri setlerinin en ham haldeyken sınıf dağılımlarını ifade ederken; ikinci durum ise, sınıf gürültüsü filtrelenmesi aşamasından sonraki sınıf dağılımlarını ifade etmektedir. Son durum ise, sınıfların dengelenmesinden sonra elde edilen yeni örneklem genişliklerinin sınıf dağılımlarını göstermektedir.

Tüm orijinal veri setlerinin sınıf dağılımları yüzdelerinde artış sağlanmasına rağmen, yukarıda bahsedilen bazı durumlar için *Lymphography* ve *Thyroid* verilerinin artışlar gözlenememiştir. *Lymphography* verisinde her dört sınıf düzeyinin sadece iki tanesinde bu artışın gözlenememesinin nedeni, veri setinde kayıp gözlemler üretmeye başlamasıdır. Aynı şekilde *Thyroid* verisinde 1. ve 2. düzeyinin örneklem temsil gücünün 3. düzeye oranla oldukça düşük olmasından dolayı, dengeyi sağlayabilmek için 3. düzeyde artış gözlemlenememiştir. Tüm veri setlerinde sınıf dengesizliği problemi ortadan kaldırıldıktan sonra yeni veri setleri kaydedilmiştir. Böylelikle tüm veri ön işleme aşamaları gerçekleşmiş olup, işlenmiş veri setlerinin performanslarının karşılaştırılması aşamasına geçilmiştir.

Tablo 14: Orijinal Veri, Sınıf Gürültüsü Filtrelenmesi ve Sınıfların Dengelenmesi Durumlarındaki Sınıf Dağılımları

Veri seti	Bağımlı değişken sınıf düzeyleri	Orijinal	Sınıf gürültüsü	Sınıf dengesizliği
Cleveland	**0: Kalp hastalığı yok**	164 (%0,543)	204 (%0, 673)	414 (%0,475)
	1: 1. derece şiddetli	55 (%0,182)	33 (%0,109)	65 (%0,075)
	2: 2. derece şiddetli	36 (%0,119)	35 (%0,116)	69 (%0,079)
	3: 3. derece şiddetli	35 (%0,116)	26 (%0,086)	43 (%0,049)
	4: 4. derece şiddetli	13 (%0,043)	5 (%0,017)	280 (%0,321)
Heart	**1: Kalp hastalığı yok**	150 (%0,556)	161 (%0,596)	327 (%0,500)
	2: Kalp hastalığı var	120 (%0,444)	109 (%0,404)	327 (%0,500)
Hepatitis	**1: Test pozitif**	32 (%0,206)	27 (%0,174)	135 (%0,500)
	2: Test negatif	123 (%0,794)	128 (%0,826)	135 (%0,500)
Lymphography	**0: normal**	2 (%0,014)	*	28 (%0,119)
	1: metastaz	81 (%0,547)	*	54 (%0,229)
	2: malign lenf	61 (%0,412)	*	43 (%0,183)
	3: fibrozis	4 (%0,027)	*	110 (%0,468)
Mammographic	**0: Test negatif**	516 (%0,537)	519 (% ,540)	1326 (%0,500)
	1: Test pozitif	445 (%0,463)	442 (%0,460)	1326 (%0,500)
Newthyroid	**1: normal**	150 (%0,698)	156 (%0,726)	290 (%0,446)
	2: hipertiroid	35 (%0,163)	34 (%0,158)	85 (%0,131)
	3: hipotiroid	30 (%0,139)	25 (%0,116)	275 (%0,423)
Pima	**0: Test negatif**	500 (%0,651)	537 (%0,699)	693 (%0,500)
	1: Test pozitif	268 (%0,349)	231 (%0,301)	693 (%0,500)
Thyroid	**1: normal**	166 (%0,023)	169 (%0,023)	1113 (%0,325)
	2: hipertiroid	368 (%0,051)	369 (%0,052)	1176 (%0,343)
	3: hipotiroid	6666 (%0,926)	6662 (%0,925)	1136 (%0,332)

* Lymphography veri seti, sınıf gürültüsüne sahip değildir.

4.3. İşlenmiş Veri Setlerinin Analizi

Veri ön işlemeden sonra algoritma performanslarının karşılaştırılması için işlenmiş veri setlerin eğitim ve test veri setlerinin dağılımı Tablo 15'de yer verilmiştir.

Tablo 15: İşlenmiş Veri Setlerine ait Test ve Eğitim Veri Setleri

Veri seti	Örneklem sayısı	Eğitim veri seti (% 70)	Test veri set (% 30)
Cleveland	871	612	259
Heart	654	458	196
Hepatitis	391	275	116
Lymphography	501	353	148
Mammographic	2652	1858	794
Newthyroid	650	456	194
Pima	1386	972	414
Thyroid	3425	2400	1025

Bu sonuçlara göre, orijinal veri setlerine göre işlenmiş veri setlerindeki eğitim ve test veri setlerindeki dağılımlar artmıştır. Bu dağılımlar elde edildikten sonra Tablo 10'da yer alan model parametrelerine göre tüm veri setleri için algoritmaların parametreleri ayarlanmıştır. Sadece *Thyroid* verisine ait bazı değişiklikler yapılmıştır. SMOTE algoritması *Thyroid* verisinin sınıf dağılımını dengelerken orijinal veri setinin *Query on thyroxine, On antithyroid medication* ve *Hypopituitary* kategorik değişkenlerinde sadece tek düzeyde (0) yeni gözlemler üretilmiştir. Bu nedenden dolayı kullanılan algoritmalar bu değişkenler için duyarlılık göstererek katsayılarını hesaplayamamışlardır. Böylelikle, bu problemin ortadan kaldırılması için her üç değişken de *Tyhroid* veri setinden kaldırılmıştır. Bundan dolayı 21 bağımsız değişkene sahip olan *Thyroid* verisi 18 bağımsız değişkene sahip olmuştur. İşlenmiş veri setlerinin algoritmalara göre performanslarının karşılaştırılması Tablo 16'te verilmiştir. Tablo 16'da koyu olarak belirtilen sonuçlar, işlenmiş veri setlerine ait model performans ölçüm metriklerinin en iyi değerlerini ifade etmektedir.

Tablo 16: İşlenmiş Veri Setlerine ait Model Performans Ölçüm Metrikleri Değerleri

Veri Seti	Algoritma	ACC	SEN	SPE	PRE	Kappa	F	YI	ROC
Cleveland	rf	0,946	0,865	0,982	**0,927**	0,915	0,895	0,847	0,895
	wsrf	0,927	0,838	0,977	0,882	0,886	0,859	0,815	0,895
	logitboost	0,931	0,825	0,981	0,855	0,892	0,839	0,806	0,728
	gbm	**0,954**	**0,884**	**0,985**	0,922	**0,928**	**0,903**	**0,869**	**0,914**
Heart	rf	0,939	0,949	0,929	0,930	0,878	0,939	0,878	0,992
	wsrf	0,939	**0,979**	0,898	0,906	0,878	0,941	0,878	0,988
	logitboost	0,934	0,957	0,912	0,917	0,869	0,936	0,869	0,970
	gbm	**0,959**	**0,979**	**0,939**	**0,941**	**0,918**	**0,960**	**0,918**	**0,997**
Hepatitis	rf	0,963	0,950	**0,975**	0,974	0,925	0,962	0,925	**0,997**
	wsrf	0,963	0,950	**0,975**	0,974	0,925	0,962	0,925	0,994
	logitboost	**0,975**	**0,975**	**0,975**	**0,975**	**0,950**	**0,975**	**0,950**	0,995
	gbm	0,963	0,950	**0,975**	0,974	0,925	0,962	0,925	0,994
Lymphography	rf	0,928	0,911	0,977	0,909	0,893	0,910	0,889	0,894
	wsrf	0,913	0,896	0,973	0,893	0,871	0,894	0,869	0,764
	logitboost	0,884	0,865	0,964	0,865	0,829	0,865	0,829	0,776
	gbm	**0,942**	**0,927**	**0,982**	**0,927**	**0,914**	**0,927**	**0,909**	**0,930**
Mammographic	rf	0,917	0,897	0,937	0,934	0,834	0,915	0,834	0,975
	wsrf	0,933	0,942	0,924	0,926	0,867	0,934	0,867	0,987
	logitboost	**0,958**	**0,972**	0,942	**0,948**	**0,915**	**0,960**	**0,914**	0,972
	gbm	0,947	0,950	**0,945**	0,945	0,894	0,947	0,894	**0,992**
Newthyroid	rf	0,995	0,996	0,998	0,987	0,992	0,992	0,994	0,891
	wsrf	0,985	0,969	0,992	0,979	0,974	0,974	0,961	0,901
	logitboost	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**0,907**
	gbm	0,995	0,996	0,998	0,987	0,992	0,992	0,994	0,961
Pima	rf	**0,944**	**0,971**	0,918	0,922	**0,889**	**0,946**	**0,889**	0,986
	wsrf	**0,944**	0,952	**0,937**	**0,938**	**0,889**	0,945	**0,889**	0,979
	logitboost	0,894	0,889	0,899	0,898	0,788	0,894	0,788	0,927
	gbm	0,942	0,956	0,927	0,929	0,884	0,943	0,884	**0,980**
Thyroid	rf	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,809
	wsrf	0,998	0,998	0,999	0,998	0,997	0,998	0,997	0,835
	logitboost	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	**1,000**	0,833
	gbm	0,998	0,998	0,999	0,998	0,997	0,998	0,997	**0,987**

ACC: Doğruluk, SEN: Duyarlılık, SPE: Seçicilik, PRE: Hassaslık, Kappa: Kappa istatistiği, F: F-ölçütü, YI: Youden indeksi, ROC: ROC eğrisi

Cleveland verisinin tüm algoritma modellerine ait ölçüm metrikleri değerlendirildiğinde, modelleme performans başarılarının orijinal veriye göre oldukça yüksek olduğu gözlemlenmiştir. *Cleveland* verisi için en iyi sınıflandırma başarısına sahip algoritma, *gbm* algoritmasıdır. Fakat sadece hassaslık ölçütü bakımından %0,927 oranla en iyi değer, *rf* algoritmasında elde edilmiştir.

Heart verisi, genel anlamda tüm ölçüm metriklerinde yüksek değerler vermiştir. Sadece duyarlılık ölçütü bakımından *wsrf* ve *gbm* algoritmaları %0,979 oranla aynı değere sahiptirler. Buna göre, *Heart* verisinin en başarılı sınıflandırma algoritması, *gbm* algoritmasıdır.

Hepatitis verinin genel olarak ölçüm metrikleri yüksek sonuçlar vermiştir. Özellikle, seçicilik ölçütü bakımından her dört algoritmada da aynı sonuçlar (%0,975) elde edilmiştir. Bu sonuçlara göre, *Hepatitis* verisinin en başarılı algoritması, *logitboost* algoritması olmuştur.

Lymphography verisi için tüm ölçüm değerlerinde en iyi sınıflandırma başarına sahip algoritma, *gbm* algoritmasıdır.

Mammographic verisi, tüm modeller açısından genel anlamda oldukça yüksek bir sınıflandırma başarısına sahiptir. Sadece seçicilik ölçütünde %0,945 oranla ve ROC eğrisi ölçütünde %0,992 oranla *gbm* algoritmasında en iyi değeri almasına rağmen en iyi sınıflandırma başarısını *logitboost* algoritmasında elde etmiştir.

Newthyroid verisi tüm sınıflandırma algoritmalarında yüksek sınıflandırma başarısı göstermiştir. *Newthyroid* verisinin en iyi sınıflandırma başarısı gösteren algoritması *logitboost* algoritması olmuştur. Bu algoritma tüm performans ölçüm metriklerinde %100 başarı göstermesine rağmen, sadece %0,907 ROC eğrisi değerine sahiptir.

Pima verisi sadece *gbm* algoritmasında ROC eğrisi değeri %0,98 oranla en iyi sınıflandırma değerine sahiptir. Bunun yanında, *rf* ve *wsrf* algoritmalarında %0,944 oranında doğruluk, %0,889 oranında Kappa istatistiği ve %0,889 oranında Youden indeksi değerleri elde edilmiştir. Tüm bu sonuçlara göre, *Pima* veri için *rf* ve *wsrf* algoritmaları en başarılı sınıflandırmaya sahip algoritmalardır.

Thyroid verisi orijinal veri setinde olduğu gibi yine tüm algoritmalar açısından oldukça yüksek sonuçlar vermiştir. *Thyroid* verisi, *rf* ve *logitboost* algoritmalarında %100 başarıya sahip olmasına rağmen sadece ROC eğrisi değerinde %0,987 oranla *gbm* algoritmasında en iyi sınıflandırma başarısı göstermiştir. Bu sonuçlara göre, *Thyroid* veri için en başarılı

sınıflandırma algoritmaları, *rf* ve *logitboost* algoritmalarıdır. Şekil 22'de ifade edildiği gibi, işlenmiş verilere ait tüm performans ölçüm metrikleri %0,70 ile %0,100 arasında değerler almaktadır. Böylelikle, veri ön işlemeden sonra bu değer aralığının daraldığı görülmüştür. Böylelikle, model performanslarındaki iyileşmeler açık bir şekilde gözlemlenmiştir.

İşlenmiş veri setleri için kurulan modellerin en iyi parametre değerleri ve bu modellere ait bazı sonuçlar Tablo 17'de verilmiştir.

Tablo 17: İşlenmiş Veri Setlerine ait Modellerin En İyi Parametre Değerleri ve Sonuçları

	Cleveland	**Heart**	**Hepatitis**	**Lymphography**	**Mammographic**	**Newthyroid**	**Pima**	**Thyroid**
rf								
OOB hata oranı	% 2,78	% 7,42	% 1,58	% 2,41	% 8,72	% 0,22	% 5,25	% 0,12
wsrf								
OOB hata oranı	% 0,04	% 0,07	% 0,06	% 0,05	% 0,06	% 0,01	% 0,07	% 0,00
strength	0,82	0,68	0,81	0,86	0,85	0,96	0,71	0,99
korelasyon	0,07	0,09	0,08	0,06	0,08	0,02	0,10	0,00
logitboost								
nIter	1300	1300	1900	1800	500	500	800	1800
gbm								
n.trees	400	550	150	50	250	200	400	800
interaction.depth	5	9	5	9	9	5	9	9
shrinkage	0,1	0,1	0,1	0,1	0,1	0,1	0,1	0,1
n.minobsinnode	10	10	10	10	10	10	10	10

Tüm bu sonuçlara göre, orijinal veri setlerinin sahip olduğu *rf* algoritması OOB hata oranları ciddi anlamda azalmalar meydana gelmiştir. Bunun sebeplerin birisi veri setlerine ait örneklem genişliklerinin artması iken, diğeri veri setlerinin veri ön işlemeye tabi tutularak olası tüm problemlerden arındırılmasıdır. *Mammographic* verisinin *rf* algoritmasına ait test veri seti tahmin hata oranı %8,72 ile en yüksek değere sahipken; *Thyroid* verisi %0,12 ile en düşük değere sahiptir. Bu sonuçlara göre, orijinal veri setine göre işlenmiş veri setlerinde *rf* algoritması OOB hata oralarında %10 ile %40 arasında bir iyileşme gerçekleşmiştir.

Böylelikle, kurulan *rf* modellerinin model tahminleme başarılarının veri ön işlemeden sonra oldukça arttığı söylenebilmektedir. Aynı zamanda, veri ön işlemeden sonra kurulan *rf* algoritması performans ölçüm metrikleri değerleri de bu sonuçları desteklemektedir.

Wsrf algoritmasına ait OOB hata oranı da *rf* algoritması hata oranları gibi %10 ile %40 arasında bir iyileşme göstermiştir. *Heart* ve *Pima* verileri %0,07 oranı ile en yüksek değere sahipken, en düşük OOB hata oranına sahip olan veri seti *Thyroid* verisidir. *Wsrf* algoritmasına özgü bir sonuç olan *strength* değeri, orijinal veri setinde olduğu gibi en yüksek değeri 0,99 oranla *Thyroid* verisine aitken, en düşük değeri 0,68 oranla *Heart* verisine aittir. Bu durum, *Thyroid* verisine ait *wsrf* algoritması modeli için kurulan bireysel rastgele orman ağaçları performanslarının genel başarısının oldukça yüksek olduğu anlamına gelmektedir. *Wsrf* modelindeki ağaçlarının çeşitliliğinin ölçümünü sağlayan korelasyon değeri için en yüksek değer 0,10 oranla *Pima* verisine aitken, en düşük değer *Thyroid* verisine aittir. Buna göre, *Thyriod* verisine ait korelasyon değeri 0 olduğundan *wsrf* modelindeki rastgele orman ağaçlarının birbirileriyle ilişkili olmadığı söylenebilmektedir. Böylelikle topluluk öğrenme algoritmalarının başarıları bu çeşitlilikten dolayı oldukça artmıştır.

Logitboost algoritması modeli, en yüksek artırma yineleme sayısında en iyi parametre değerini aldığından, her veri setinin bağımsız değişken sayının 100 katında en yüksek değer elde edilmiştir. Bundan dolayı bağımsız değişken sayısı (19) en fazla olan *Hepatitis* verisi en yüksek artırma yineleme sayına sahip olduğundan en iyi modeli 1900. yinelemede kurmuştur. *Mammographic* ve *Newthyroid* veri setleri en az ve aynı bağımsız değişken sayılarına (5) sahip olduğundan en iyi modeli 500. yinelemede elde edilmiştir.

Gbm algoritması modeline ait parametre değerlerine bakıldığında, öğrenme oranı $shrinkage$ ve ağaçlardaki yaprak düğümlerin sayısı $n.minobsinnode$ parametreleri orijinal veri setlerinde olduğu gibi sabit tutulmuştur. İşlenmiş veri setlerinde en iyi *gbm* algoritma modeli, *Lymphography* veri için en düşük 50. ve *Thyroid* verisi için en yüksek 800. ağaçta kurulmuştur. *Gbm* algoritmasının diğer parametresi olan final yaprak sayıları için *Cleveland, Hepatitis* ve *Newthyroid* verilerinin en iyi *gbm* algoritması modeli 5 final yaprak düğümüne sahipken, diğer tüm işlenmiş veri setleri için en iyi *gbm* algoritması modeli 9 final yaprak düğümüne sahiptirler.

Şekil 18a'da ve Şekil 18b'de gösterildiği gibi, işlenmiş verilere ait OOB hata oranları *rf* algoritmasında 0 ile %0,10 arasında değer alırken, *wsrf* algoritmasında yaklaşık sıfıra çok

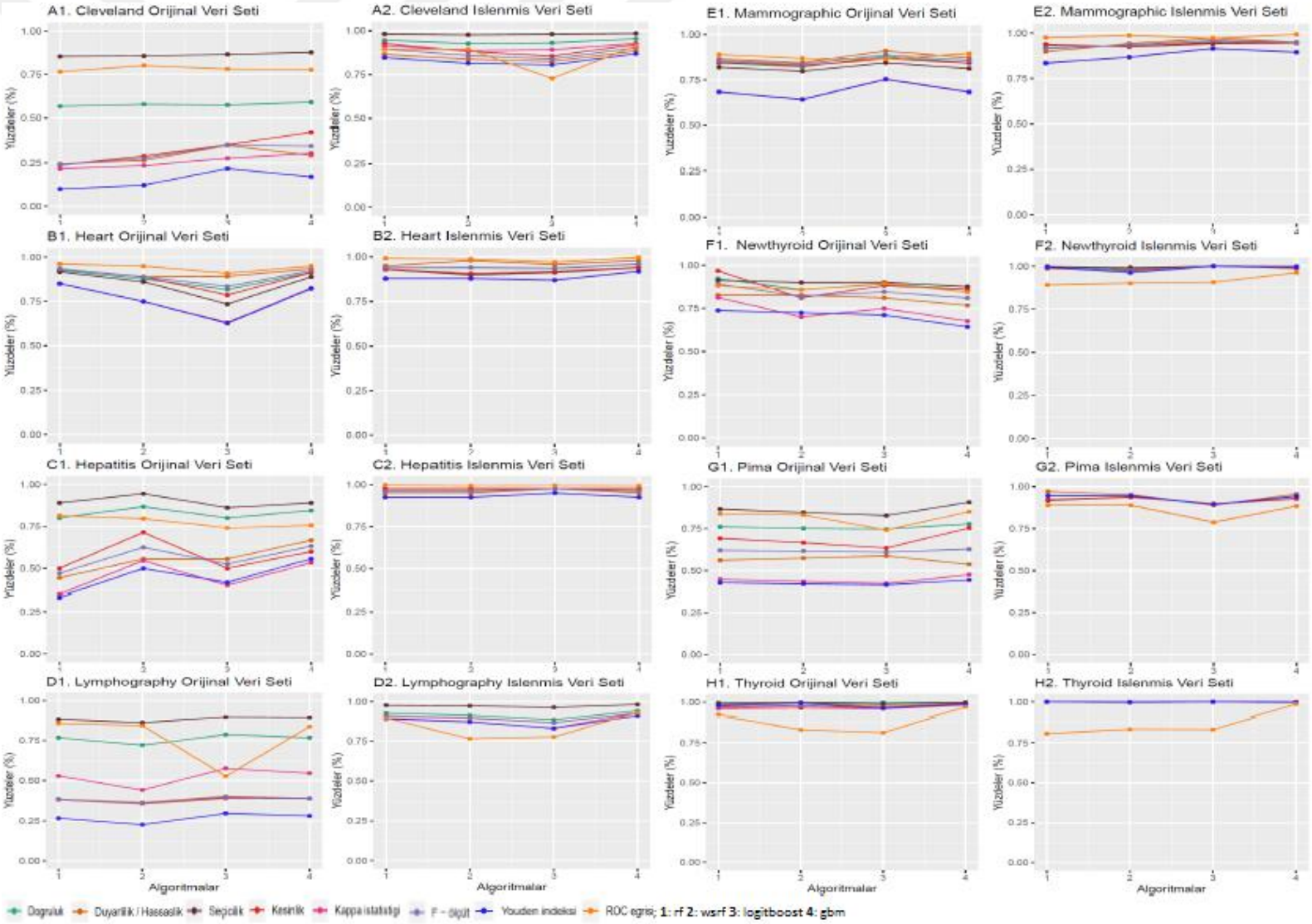
yakın değerler almıştırlar. Buradan da anlaşılacağı gibi, veri ön işlemeden sonra OOB hata oranlarında önemli derecede azalmalar meydana gelerek algoritmaların sınıflandırma başarıları artmıştır.

Genel olarak hem orijinal hem de işlenmiş veri setlerinin en başarılı algoritmaları Tablo 18'de verilmiştir.

Tablo 18: Orijinal ve İşlenmiş Veri Setlerinin En Başarılı Algoritmaları

Veri seti	Orijinal	İşlenmiş
Cleveland	logitboost gbm	gbm
Heart	rf	gbm
Hepatitis	wsrf gbm	logitboost
Lymphography	logitboost	gbm
Mammography	logitboost	logitboost
Newthyroid	rf	logitboost
Pima	gbm	rf wsrf
Thyroid	gbm	rf logitboost

Bu sonuçlara göre, artırma algoritmaları torbalama algoritmalarına göre daha başarılı sonuçlar vermiştir. Bunun nedenlerinden biri, bu veri setlerinde torbalama algoritmalarının daha zor eğitildiği ve modelleme başarılarının artırma algoritmalarına göre daha düşük olmasıdır. Diğer bir neden ise, artırma algoritmalarının gözlem odaklı olarak modelleri her defasında güncellemesidir. Bu güncellemeler sayesinde yanlış sınıflandırılmış gözlemlerin doğru sınıflandırılabilme şansını arttırmasıdır.



Şekil 22: Orijinal ve İşlenmiş Verileri için Kurulan Algoritmaların Performans Ölçüm Metriklerinin Karşılaştırılması

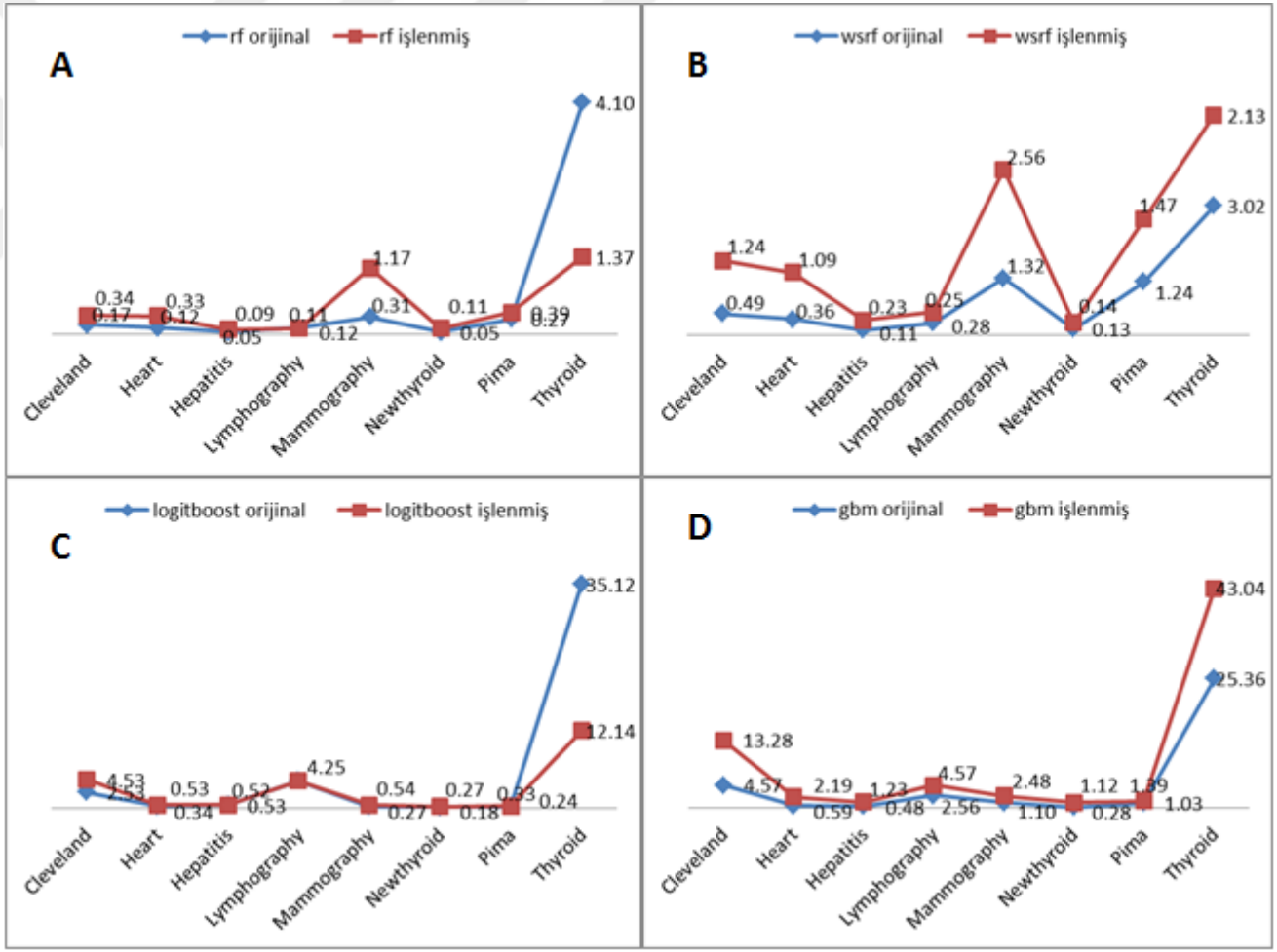
4.4. Algoritmaların Çalışma Süresi

Çalışma kapsamında yapılan tüm işlemlerin hesaplama süreleri kaydedilmiştir. Tablo 19'da gösterildiği gibi, ilk aşama, orijinal verilerin analizinde kullanılan algoritmalar; ikinci aşama, veri ön işleme analizi ve son aşama ise, işlenmiş verilerin analizinde kullanılan algoritmaların sonucu kaydedilen çalışma sürelerini ifade etmektedir. Grafik 3'te, *rf, wsrf, logitboost* ve *gbm* algoritmalarının hem orijinal verilerde hem de işlenmiş verilerde kaydedilmiş çalışma sürelerini göstermektedir.

Tablo 19'dan ve Grafik 3'ten anlaşılacağı gibi, artırma algoritmalarının çalışma süreleri torbalama algoritmalarına göre daha uzundur. Veri setinin genişliği arttıkça algoritmaların çalışma süreleri de aynı şekilde artmaktadır. İşlenmiş verilerin, orijinal verilere göre daha uzun süre çalışmaktadır. *Gbm* algoritması, en uzun çalışma süresine sahip algoritma iken; *rf* algoritması, en kısa çalışma süresine sahip algoritmadır. *Thyroid* verisi en geniş örneklem büyüklüğüne sahip olan algoritma olduğundan, hem orijinal veride hem de işlenmiş veride en uzun çalışma süresine (168,77 dk) sahip olan veri setidir. Diğer yandan, *Newthyroid* verisi hem orijinal veride hem de işlenmiş veride en kısa çalışma süresine (2,36 dk) sahiptir. Veri ön işleme aşamasında, sınıf gürültüsü problemi için filtreleme işlemi en uzun süren aşamadır.

Tablo 19: Algoritmaların Çalışma Süreleri

	Aşama1: Orijinal veriler				Aşama2: Veri ön işleme			Aşama3: İşlenmiş veriler				Üç aşama için
Veri seti	**rf**	**wsrf**	**logitboost**	**gbm**	**Kayıp gözlem**	**Sınıf gürültüsü**	**Sınıf dengesizliği**	**rf**	**wsrf**	**logitboost**	**gbm**	**toplam süre (dk)**
Cleveland	0,17	0,49	2,53	4,57	0,06	0,15	0,01	0,34	1,24	4,53	13,28	27,37
Heart	0,12	0,36	0,34	0,59	-	0,12	0,01	0,33	1,09	0,53	2,19	5,68
Hepatitis	0,05	0,11	0,53	0,48	0,10	0,12	0,01	0,09	0,23	0,52	1,23	3,47
Lymphography	0,12	0,28	4,25	2,56	-	-	0,01	0,11	0,25	4,25	4,57	16,40
Mammography	0,31	1,32	0,27	1,10	0,26	0,32	0,01	1,17	2,56	0,54	2,48	10,34
Newthyroid	0,05	0,13	0,18	0,28	-	0,07	0,01	0,11	0,14	0,27	1,12	2,36
Pima	0,27	1,24	0,24	1,03	-	0,18	0,01	0,39	1,47	0,33	1,39	6,55
Thyroid	4,10	3,02	35,12	25,36	-	42,48	0,01	1,37	2,13	12,14	43,04	168,77



Grafik 3: Topluluk Öğrenme Yöntemlerinin Çalışma Süreleri

Tartışma

Tezin tartışma kısmında, ele edilen sonuçların hipotezlere bağlı kalınarak literatürdeki benzer ve farklı çalışmalarla kıyaslaması yapılmıştır. Literatürde bu konu üzerine yapılan çalışmalar sadece modellerin genel doğruluk değerleri açısından değerlendirilmiştir. Literatürdeki çalışmaların tamamında ve bizim çalışmamızda dahil olmak üzere model geçerliliği için 10-katlı çapraz doğrulama yöntemi kullanılmıştır. Bunun yanı sıra, çalışmamızda kullandığımız veri setleri örneklem genişlikleri, sınıf düzeyleri, kayıp gözlem yüzdeleri, sınıf gürültüsü yüzdeleri ve sınıf dengesizliği oranları bakımından birbirinden farklı seçilmiştir.

Moon ve diğerleri (2007) yaptıkları çalışmada, topluluk öğrenme tabanlı bir rastgele bölümlerden toplulukla sınıflandırma (classification by ensembles from random partitions) yaklaşımı geliştirmişlerdir. Bu yaklaşımın performansını, rastgele orman, Adaboost, logitboost, karar ormanı, destek vektör makinası, diagonal doğrusal diskriminant analizi, çökmüş sentroidler (shrunken centroids), CART, yansız etkileşim seçimi ve tahmini olan sınıflandırma kuralı ve hızlı, yansız ve verimli istatistik ağacı algoritmaları ile karşılaştırmışlardır. Çalışmada *Lymphoma, Lung Cancer* ve *Breast Cancer* hastalıklarına ait genomik veriler kullanılmıştır. Bu verilerin boyutları bizim çalışmamızda kullandığımız veri setlerinden oldukça büyüktür. *Lymphoma* verisi, 4026 gene sahip 47 örneklemden oluşan bir veri setidir. Kayıp gözlem problemine sahip olduğu için 10-en yakın komşu yöntemi ortalamaları kullanılarak atama yapılmıştır. *Lung cancer* verisi 12533 gene sahip 181 örneklemden oluşan veri setidir. Bu veriden gürültülü veriler filtrelendiğinden 5000 gen ile analiz yapılmıştır. *Breast cancer* verisi 25000 gene sahip 78 örneklemden oluşan bir veri setidir. Bu veri setinde 5000 anlamlı gen olduğundan geriye kalan genler veri setinden çıkarılmıştır. Tüm bu veri ön işlemeler sonucunda her üç veri seti için model performansları kıyaslandığında, rastgele bölümlerden toplulukla sınıflandırılma algoritması en yüksek başarıya sahip algoritma olmuştur. Bu çalışma, bizim çalışmamızdan farklı olarak genomik veriler kullanarak hastalık tanısının sınıflandırılmasını çalışmıştır (Moon et al., 2007).

Verma ve Hassan (2011) tarafından yapılan çalışmada, denetimsiz topluluk öğrenme yaklaşımları kullanılarak *Mammography*, *Wisconsin Breast Cancer* ve *Pima Diabetes* verilerinde hastalık tanısı tanımlanmıştır. Çalışmada kullanılan hibrid topluluk öğrenme yöntemleri, bizim tez çalışmamızda olduğu gibi *Mammography* verisi için %100 doğruluk sonucuna vermiştir. Bu sonuç, eğer veriler arasında mükemmel bir modelleme başarısı

yakalanıyorsa; eğitim ve test verilerinde bu doğruluğa ulaşmanın mümkün olabileceğini göstermektedir (UCI, 2019; Verma & Hassan, 2011).

Lavanya ve Rani (2012) çalışmalarında, topluluk karar ağaçları sınıflandırıcısı olan CART torbalama algoritmasını, CART algoritmasına entegre çalışan değişken seçim yaklaşımını ve veri ön işleme yapabilen hibrid yaklaşımını kullanarak, UCI veri tabanından alınan *Breast Cancer, Wisconsin Breast Cancer (Original)* ve *Wisconsin Breast Cancer (Diagnostic)* olmak üzere üç farklı meme kanseri veri seti için model performanslarını karşılaştırmışlardır. Bu veri setlerinin örneklem genişlikleri sırasıyla, 286, 699 ve 569 gözlemden oluşmaktadır. Veri ön işleme aşamasında, en anlamlı bağımsız değişkenler seçilmiş ve kayıp gözlemler veri setlerinden silinmiştir. Çalışmada sadece doğruluk kriterini göz önünde bulundurarak model çalışma süreleri kaydedilmiştir. Bu sonuçlara göre, hibrid yaklaşımı her üç veri seti için en iyi doğruluk değerlerine (ACC: %74,47; %97,85; %95,96) sahip yaklaşımken, en uzun çalışma süresi *Breast Cancer* verisi için 28,25 dk olarak kaydedilmiştir. Bu çalışmada, *Breast Cancer* verisinin en az örneklem genişliğine sahip olmasına rağmen en uzun çalışma süresine sahip olmasının nedeni olarak veriden öğrenmenin ne derece zor olduğu söylenebilmektedir. Bu durumda, en küçük örneklemde en hızlı çalışma sürenin elde edilmesi gibi bir yargıya varmak her zaman doğru değildir. Bunu en açık görebileceğimiz durum ise, bizim çalışmamızda kullandığımız orijinal verilerin çalışma süreleri ile işlenmiş verilerin çalışma süreleri arasında önemli derecede bir fark olmamasıdır. Sadece *Thyroid* verisinin çalışma süreleri arasında bariz bir fark vardır. Bunun sebebi ise, örneklem genişliğinin yarı yarıya azalmasıdır (Lavanya D & Rani, 2012; UCI, 2019).

Budnik ve Krawczyk (2013) çalışmalarında, veri ön işlemede parametrelerin ayarlanması konusu üzerinde durmuşlardır. Bu konu bizim çalışmamızda ele almadığımız bir konu olduğundan diğer çalışmalardan da farklılık göstermektedir. Çalışmada, parametrelerin ayarlanması için topluluk öğrenme yöntemleri kullanarak bir karar destek sistemi sunmuşlardır. Bu çalışmada, bizim kullandığımız *Cleveland, Heart, Hepatitis, Lymphography* ve *Pima* verilerinin yanı sıra UCI veri tabanından elde edilen 10 farklı hastalık tanısı veri seti kullanılmıştır. Farklı örneklem genişliklerinin ve farklı değişken sayılarının ele alındığı çalışmada, C4.5, CART ve azaltılmış hata budaması (reduced-error pruning) temel sınıflandırıcıları ile topluluk öğrenme yöntemlerinden torbalama, rotasyon ormanı ve rastgele orman algoritmalarının performansı karşılaştırılmıştır. Genel olarak, tüm algoritmalar açısından sonuçlar kıyaslandığında topluluk öğrenme algoritmalarının diğer temel sınıflandırıcılara göre başarıları %80'in üzerinde bulunmuştur (Budnik & Krawczyk, 2013;

UCI, 2019). Buna göre, bizim çalışmamızda da olduğu gibi veri setlerinin farklı genişliklerde olması ve farklı değişken sayılarına sahip olması gibi durumlarda topluluk öğrenme algoritmalarının öğrenme becerisi diğer algoritmalara göre daha yüksektir. Bizim çalışmamız, bu çalışmadan farklı olarak artırma algoritmalarına da yer vermiştir.

Kumar ve diğerleri (2013) çalışmalarında, UCI veri tabanından *Wisconsin Breast Cancer* ve *Pima Diabetes* verilerini kullanarak torbalama, artırma, AdaBoost, Multiboosting ve rastgele orman algoritmalarının doğruluk, duyarlılık, seçicilik, hata oranı metrikleri ile performanslarını ve algoritmaların çalışma sürelerini kıyaslamışlardır. Veri ön işleme kısmında ilişkisiz ve gürültülü bağımsız değişkenler için herhangi bir düzenleme yapılmadan bu değişkenler veri setinden silinmiştir. Rastgele orman, meme kanseri veri seti için en başarılı algoritma iken (ACC: %94,49) ve diyabet veri seti için *MultiBoost* algoritması en başarılı algoritma olarak bulunmuştur (ACC: %76,31). Bunun yanı sıra, torbalama algoritmaları en uzun çalışma süresine sahip algoritmalardır. Bu süreler, *Wisconsin Breast Cancer* veri için yaklaşık 0,17 dk iken, *Pima Diabetes* veri için yaklaşık 0,50 dk civarındadır (Kumar et al., 2013; UCI, 2019).

Srimani ve Koti (2013) yaptıkları çalışmada, UCI veri tabanından *Thyroid, Bupa Liver, Haberman, Hepatitis* ve *Wisconsin Breast* verilerini kullanmışlardır. Bu çalışmada, alternatif karar ağacı (alternating decision tree), en iyi ilk karar ağacı (best-first decision tree), karar kütüğü (decision stump), fonksiyonel ağaçlar (functional trees), J48 karar ağaçları, logitboost karar ağaçları, lojistik model ağaçları, rastgele orman ve naïve Bayes ağaçları gibi çeşitli topluluk öğrenme yaklaşımlarının doğruluk, Kappa istatistiği ve ROC eğrisi sonuçlarını kıyaslamışlardır. Bu çalışma bizim çalışmamızdan farklı olarak, veri setleri için veri ön işleme yapmamıştır. Genel olarak, tüm algoritmalar için *Thyroid* verisi logistik model ağaçlarında %97,7 doğruluğa, *Hepatitis* verisi %83,6 doğruluğa, *Haberman* verisi karar ağaçlarında %74,8 doğruluğa, *Wisconsin Bearst* verisi karar ağaçlarında %86,8 doğruluğa ve *Bupa Liver* verisi karar ağaçlarında %75,1 doğruluğa sahiptir. Tüm veri setlerinde, lojistik model ağaçları yaklaşımı en uzun çalışma süresine sahip algoritma olarak bulunmuştur (*Thyroid*: 6,4 dk; *Bupa Liver*: 7,24 dk; *Haberman*: 2,82 dk; *Hepatitis*: 2,76 dk; *Wisconsin Breast*: 18,03 dk). Bu sonuçlar veri ön işleme yapılmadan doğrudan sınıflandırma yöntemlerinin kullanılmasının model başarılarını düşürebildiğini göstermektedir (Srimani & Koti, 2013; UCI, 2019).

Gerçek hayat verilerinde karşılaşılan problemlerden biri olan sınıf gürültüsü, Saez ve diğerleri (2016) tarafından yapılan çalışmada detaylı olarak ele alınmıştır. Bu çalışmada, tıbbi verilerin sınıflandırılmasında sınıf gürültüsünün insan hataları, makine hataları, dijitalleştirme ve

arşivleme hataları gibi birçok kaynaktan meydana geldiğini belirtilmiştir. Bu problemin çözümünde, algoritma seviyesi (algorithm-level) ve veri seviyesi yaklaşımları önerilmiştir (Sáez, Krawczyk, & Woźniak, 2016).

Alanis-Tamez ve diğerleri (2017) ise bizim tez çalışmamızla benzer şekilde KEEL veri tabanından hastalık tanısı veri setlerini kullanmışlardır. Bu çalışmada, kayıp gözlem ve sınıf dengesizliği problemlerine dikkat çekilmesine rağmen hiçbir veri ön işleme yapılmamıştır. Bu orijinal verilerle; lojistik, çok kategorili lojistik, k-en yakın komşular, destek vektör makinaları ve C4.5 karar ağacı zayıf sınıflandırıcıları kullanılarak sınıflandırma modelleri elde edilmiştir. Ayrıca diğer çalışmalardan farklı olarak, 5-katlı çapraz geçerlilik yöntemi ile model doğrulama yapılmıştır (Alanis-Tamez, Villuendas-Rey, & Yáñez-Márquez, 2017; Alcalá-Fdez et al., 2009). Tüm bu sonuçlar Friedman test istatistiği ile test edildiğinde, lojistik model en iyi sınıflandırma algoritması olarak bulunmuştur ($p < 0{,}05$).

Jan ve diğerleri (2018) çalışmalarında, UCI veri tabanlarından elde edilen *Cleveland* ve *Hungarian* kardiyovasküler hastalık verisi için destek vektör makinesi, yapay sinir ağı, naive Bayes, regresyon analizi ve rastgele orman algoritmalarını bir arada kullanabilen bir topluluk öğrenme yaklaşımı ortaya koymuşlardır. Bu yaklaşım, akıllı bir kalp hastalığı tanısı tahminleme sistemi olarak geliştirilmiştir. Sınıflandırma algoritmalarını kullanmadan önce; kayıp gözlem, sınıf dengesizliği ve değişkenlerin normalleştirilmesi veri ön işleme adımları uygulanmıştır. Fakat bu çalışmada, bizim çalışmamızdan farklı olarak verilerin orijinal durumdaki performansları değerlendirilmemiştir. Eğer orijinal verilerin performansları değerlendirilseydi, algoritmaların performanslarına ne kadarlık bir iyileşme olduğu gözlenebilirdi. Bu sonuçlara göre, sistemin genel doğruluğu %98,14 olarak bulunmuştur (Jan, Awan, Khalid, & Nisar, 2018; UCI, 2019).

Guncar ve diğerleri (2018) ise, hemotoloji hastalığı tanısı sınıflandırılması için iki farklı topluluk tabanlı akıllı kan analizi algoritması (smart blood analytics algorithm) geliştirmişlerdir. Bu iki yaklaşıma SBA-HEM61 ve SBA-HEM181 adı verilmiştir. Bu yaklaşımlar, sadece hemotolojik verilerin kan testleri sonuç parametrelerini kullanarak tahminleme yapabilmektedir. Bu veriler, 43 farklı hemotolojik kategoriden oluşan 8233 hastanın bulunduğu bir veri tabanından alınmıştır. Bu çalışmada, veri ön işleme sürecinde, aşırı sapan gözlemler veri setlerinden çıkartılmış ve bizim çalışmamızda olduğu gibi kayıp gözlem problemi için *MICE* algoritmasıyla atama yapılmıştır. Tüm bu sonuçlara göre, her iki algoritmanın modelleme genel doğrulukları sırasıyla %59 ve %57 değerlerinden %88 ve %86 değerlerine yükselmiştir (Gunčar et al., 2018).

Feng ve diğerleri (2018), sınıf gürültüsü veri ön işlemesi için *SMOTE* algoritmasına entegre olarak çalışabilen topluluk tabanlı UnderBagging ve SMOTEBagging yaklaşımını önererek denetimli ve denetimsiz torbalama modellerinin genel doğruluğunu %96 değerine kadar yükseltmişlerdir (W. Feng et al., 2018).

Sınıf dengesizliği problemi için literatürde temel olarak önerilen aşırı örnekleme ve az örnekleme yöntemleri, sınıf dağılımlarını dengelemek için kullanılsa da, yöntemlerin kullanımında bazı ciddi handikapların olduğu Fernández ve diğerleri (2013) tarafından vurgulanmıştır. Aşırı örneklemeden dolayı veri setinde aşırı uyum problemi meydana geldiğinden, eğitim veri setinin doğruluğu yüksek olsa bile; test veri setindeki sınıflandırma performansı genellikle çok daha kötü sonuçlar verebilmektedir (Fernández, López, Galar, del Jesus, & Herrera, 2013). Bu tez çalışmasında ise, bu durum göz önünde bulundurularak aşırı uyum problemine daha dayanıklı olan SMOTE yaklaşımı tercih edilmiştir. Böylelikle, aşırı örneklemenin çoğunluk sınıfı hakkında bilgi kaybetmeden dengeli bir dağılım sağlayabilme özelliği kullanılmıştır.

Zhang ve diğerleri (2019) yaptıkları çalışmada, ikili sınıfa sahip bir hastalık tanısı veri setini simüle ederek *Gbm* algoritmasının performansını değerlendirmeyi amaçlamışlardır. Buna bağlı olarak, benzetim yapılarak elde edilmiş verilerin karmaşık ilişkileri modellenirken kullanılan g*bm* algoritması, genelleştirilmiş doğrusal model tabanlı olan lojistik regresyon modeli yaklaşımdan daha iyi sonuçlar verdiği gösterilmiştir. *Gbm* ile lojistik regresyon algoritmasının performansı, ROC değerleri ilişkisini ölçmeyi sağlayan DeLong's test ile test edilmiştir (Z. Zhang et al., 2019). Bu sonuca göre, *gbm* algoritmasının lojistik regresyon algoritmasından istatistiksel olarak daha anlamlı olduğu sonucuna varılmıştır (0,98; %95 CI: 0,972–0,997).

Literatürde *rf, wsrf, logitboost* ve *gbm* algoritmalarını bir arada değerlendiren bir çalışmaya rastlanmamıştır. Aynı zamanda, tüm bu çalışmalara bakıldığında, bizim çalışmamızda olduğu gibi kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi problemlerin var olduğu durumlarda hem orijinal hem de işlenmiş veri setleri için topluluk öğrenme algoritmalarının performanslarının karşılaştırıldığı benzer bir çalışmaya da rastlanmamıştır. Bu çalışmalar da gösteriyor ki, verilerin sahip olduğu birçok problem olabilmektedir. Bu problemler çözülmeden veri üzerinde herhangi bir öğrenme algoritması uygulandığında modelleme performansları düşmektedir. Sorunlar çözüldükten sonra, bu modellerden öğrenmeye çalışmak daha doğru olmaktadır.

Sonuç ve Öneriler

Tezin sonuç ve öneriler kısmında, çalışma sonucu elde edilen tüm bulgular değerlendirilmiştir. Tüm orijinal veri setlerine ait bağımsız sürekli değişkenlerin birbirileriyle ilişkili olmadığı görülmüştür. Orijinal veri setlerinin performans karşılaştırmasında genel olarak, verilere ait sınıflandırma performanslarının önemli derecede başarılı olmadığı gözlemlenmiştir. Bunun nedenleri arasında, bazı veri setlerinin örneklem genişliklerinin değişken sayılarına göre oldukça düşük olması ve örneklem temsil gücünün yetersiz olması söylenebilmektedir. Diğer önemli sebepler ise, bu veri setlerinin sahip olduğu olası kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği gibi problemlerdir. Bu problemlerden arındırılan işlenmiş veri setleri aynı parametrelerle tekrar modellendiğinde, tüm veri setlerinde oldukça yüksek sınıflandırma başarıları elde edilmiştir. Artırma ve torbalama algoritmaları çalışma süreleri açısından değerlendirildiğinde ise, en uzun çalışma süresine sahip algoritmaların artırma algoritmaları olduğu saptanmıştır. Bunun en başlıca nedeni ise, artırma algoritmalarının veriden öğrenirken, sıralı olarak geliştirilen karar ağaçları modellerinin zayıf olanlarının birleştirilerek güçlü modellere çevrilmesidir. Böylelikle, zayıf sınıflandırıcıda eğitilen gözlemlerin ağırlıkları güncellenmektedir. Her yinelemede bu işlem devam ettiği için artırma algoritmalarının çalışma süreleri uzundur. Torbalama algoritmalarında bu durum söz konusu değildir. Buna bağlı olarak, torbalama algoritmalarının çalışma süreleri veri setlerinin genişliklerinin küçük veya büyük olmasına bağlı değildir ve her durumda hızlı çalışmaktadırlar. Özellikle büyük veri setlerinde torbalama algoritmalarının kullanılması önerilebilir fakat modelleme doğruluğunun yüksek olması verinin temsiliyet gücüne ve algoritmanın öğrenme becerisine bağlıdır. Veri setinin oldukça geniş olması durumunda da bu süreler uzayabilmektedir. Çalışma sürelerinin uzun olmasından dolayı artırma algoritmalarının kullanılması önerilmektedir.

Makine öğrenme algoritmalarının karşılaştırılmasında en iyi yaklaşımın belirlenmesi kolay olmamaktadır. Bu durumu, tek bir nedene veya birden fazla nedene bağlamak mümkündür. Örneklem genişliğinin arttırılması her zaman avantaj sağlamayabilmektedir. Bunun sebebi ise, algoritmanın doğru öğrenmesini sağlayacak verinin ortaya çıkarılmasıdır. Genel olarak, makine öğrenme algoritmalarının kullanıldığı bu tip çalışmalarda, bir veya birden fazla algoritmanın başarılı olması yapılacak olan benzer çalışmalarda aynı başarılı sonuçları vermeyebilir. Özellikle, büyük veride makine öğrenme algoritmaları kullanılacaksa, veri ön işlemenin göz ardı edilmemesi gerekmektedir. Bu çalışmada bahsedilen kayıp gözlem, sınıf gürültüsü ve sınıf dengesizliği problemleri dışında veri setine has problemler çözüldükten

sonra analizler yapılmalıdır. Böylelikle, gözle görülür bir şekilde verilerin sınıflandırma başarılarında artış olması beklenmektedir.

Bu tez çalışmasında da, makine öğrenme yöntemleri içerisinde yer alan topluluk öğrenme sınıflandırma algoritmalarında veri ön işlemenin önemi vurgulanarak, araştırmacılara veri ön işleme sürecini göz ardı etmemeleri önerilmektedir. Böylelikle, araştırmacılar veri ön işleme yöntemleriyle kullandıkları algoritmaların başarılarının artmasında önemli derecede katkı sağlayacaklardır. Son yıllarda yapılan birçok araştırmada, özellikle makine öğrenme yöntemlerinde veri ön işleme sürecine dikkat çekilmiştir. Gelecek dönemlerde yapılacak olan benzer çalışmalarda, bu tür yöntemlerin kullanılmadan önce verilerin yapısının dikkatli bir şekilde özümsenmesi ve anlaşılması gerekmektedir. Daha sonra, kullanılan algoritmaların çalışma disiplinleri öğrenilmelidir. Eğer bu çalışmada olduğu gibi birçok veri seti ve modelleme algoritmasıyla çalışılmayacaksa, modellerin parametreleri farklı durumlar için ayarlanarak benzetimler yapılabilmektedir. Böylelikle, modellerin en iyi parametre değerleri seçilerek en başarılı algoritmalar elde edilebilmektedir. Benzer şekilde, modelleme performansını arttıracak başka bir yol ise, model için en önemli değişkenlerin seçilmesidir. Bunun için modelleme yapmadan önce değişken seçimi yöntemlerine başvurulabilmektedir. Bu çalışma kapsamında ele alınan veri ön işleme problemleri dışında parametre ayarlanması ve değişken seçimi gibi konuların ileriki çalışmalarda ele alınması planlanmaktadır.

Kaynaklar

Aamoth, D. (2014). Interview with Eugene Goostman, the Fake Kid Who Passed the Turing Test. Retrieved January 25, 2019, from http://time.com/2847900/eugene-goostman-turing-test/

Acock, A. C. (2005). *Working With Missing Values*. Oregon. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.521.9011&rep=rep1&type=pdf

Acuña, E., & Rodriguez, C. (2004). The Treatment of Missing Values and its Effect on Classifier Accuracy. In *Classification, Clustering, and Data Mining Applications* (pp. 639–647). Berlin, Heidelberg: Springer Berlin Heidelberg.

Al-Jarrah, O. Y., Muhaidat, S., Karagiannidis, G. K., & Taha, K. (2015). Efficient Machine Learning for Big Data: A Review. *Big Data Research*, *2*(3), 87–93.

Alanis-Tamez, M. D., Villuendas-Rey, Y., & Yáñez-Márquez, C. (2017). Computational Intelligence Algorithms Applied to the Pre-diagnosis of Chronic Diseases. *Research in Computing Science*, *138*, 41–50.

Alashwal, H., El Halaby, M., Crouse, J. J., Abdalla, A., & Moustafa, A. A. (2019). The Application of Unsupervised Clustering Methods to Alzheimer's Disease. *Frontiers in Computational Neuroscience*, *13*(31), 1–9.

Alcala-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Journal of Multiple-Valued Logic & Soft Computing*, *17*(2–3), 255–287. Retrieved from http://www.keel.es/

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Journal of Multiple-Valued Logic and Soft Computing*, *17*, 255–287.

Alcalá-Fdez, J., Sánchez, · L, García, · S, Del Jesus, · M J, Ventura, · S, Garrell, · J M, … Del Jesus, M. J. (2009). KEEL: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, *13*(3), 307–318.

Alegre, S. M. (2017). *Voldemort and the Failure of Magic in the Harry Potter Series: The*

Post-human Monster in Fantasy and Science Fiction. Barcelona. Retrieved from www.editorialaluvion.com,

Alghamdi, M., Al-Mallah, M., Keteyian, S., Brawner, C., Ehrman, J., & Sakr, S. (2017). Predicting diabetes mellitus using SMOTE and ensemble machine learning approach: The Henry Ford ExercIse Testing (FIT) project. *PloS One*, *12*(7), 1–15.

Allison, P. D. (2009). *Missing Data*. Retrieved from https://pdfs.semanticscholar.org/58de/eac621923189c28cae06066652c218e126e4.pdf

Altman, N., & Krzywinski, M. (2018). The curse(s) of dimensionality. *Nature Methods*, *15*(6), 399–400.

Amaratunga, D., Cabrera, J., & Lee, Y. S. (2008). Enriched random forests. *Bioinformatics*, *24*(18), 2010–2014.

Amari, S., Kurata, K., & Nagaoka, H. (1992). Information geometry of Boltzmann machines. *IEEE Transactions on Neural Networks*, *3*(2), 260–271.

Angra, S., & Ahuja, S. (2017). Machine learning and its applications: A review. In *International Conference on Big Data Analytics and Computational Intelligence* (pp. 57–60). Chirala: IEEE.

Azencott, C. (2018). Machine learning and genomics : precision medicine vs . patient privacy, (1), 1–14.

Bae, J.-M. (2014). The clinical decision analysis using decision tree. *Epidemiology and Health*, *36*, e2014025.

Barnett, G. O., Cimino, J. J., Hupp, J. A., & Hoffer, E. P. (1987). DXplain. An evolving diagnostic decision-support system. *JAMA*, *258*(1), 67–74.

Beam, A. L., & Kohane, I. S. (2016). Translating Artificial Intelligence Into Clinical Care. *JAMA*, *316*(22), 2368–2369. https://doi.org/10.1001/jama.2016.17217

Bennett, D. A. (2001). How can I deal with missing data in my study? *Australian and New Zealand Journal of Public Health*, *25*(5), 464–469.

Bernaards, C. A., Belin, T. R., & Schafer, J. L. (2007). Robustness of a multivariate normal approximation for imputation of incomplete binary data. *Statistics in Medicine*, *26*(6), 1368–1382.

Berner, E. S., Webster, G. D., Shugerman, A. A., Jackson, J. R., Algina, J., Baker, A. L.,

Taunton, O. D. (1994). Performance of Four Computer-Based Diagnostic Systems. *New England Journal of Medicine*, *330*(25), 1792–1796.

Berthold, M. R., Cebron, N., Dill, F., Gabriel, T. R., Kötter, T., Meinl, T., … Wiswedel, B. (2009). KNIME - the Konstanz information miner: version 2.0 and beyond. *ACM SIGKDD Explorations Newsletter*, *11*(1), 26.

Blagus, R., & Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced data. *BMC Bioinformatics*, *14*, 106–140.

Blagus, R., & Lusa, L. (2015). Boosting for high-dimensional two-class prediction. *BMC Bioinformatics*, *16*(1), 300–317. https://doi.org/10.1186/s12859-015-0723-9

Boden, M. A. (1984). Impacts of artificial intelligence. *Futures*, *16*(1), 60–70.

Bodner, T. E. (2008). What Improves with Increased Missing Data Imputations? *Structural Equation Modeling: A Multidisciplinary Journal*, *15*(4), 651–675.

Bouveyron, C., & Girard, S. (2009). Robust supervised classification with mixture models: Learning from data with uncertain labels. *Pattern Recognition*, *42*(11), 2649–2658.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification And Regression Trees* (1st ed.). New York: Routledge.

Brodley, C. E., & Friedl, M. A. (1999). Identifying Mislabeled Training Data. *Journal of Artificial Intelligence Research*, *11*, 131–167. https://doi.org/10.1613/jair.606

Brown, G., Wyatt, J., Harris, R., & Yao, X. (2005). Diversity creation methods: a survey and categorisation. *Information Fusion*, *6*(1), 5–20.

Brown, G., Wyatt, J. L., & Tino, P. (2005). Managing Diversity in Regression Ensembles. *Journal of Machine Learning Research*, *6*, 1621–1650.

Browne, C., Powley, E., Whitehouse, D., Lucas, S., Member, S., Cowling, P. I., … Colton, S. (2012). A Survey of Monte Carlo Tree Search Methods. *The IEEE Transactions on Computational Intelligence and AI in Games*, *4*(1), 1–49.

Bryson, A., & Ho, Y.-C. (1969). *Applied optimal control : optimization, estimation, and control: Arthur E. Ho, Yu-Chi, Bryson: Amazon.com: Books* (1st ed.). Waltham: Blaisdell Publishing Company.

Buckland, M., & Gey, F. (1994). The relationship between Recall and Precision. *Journal of the American Society for Information Science*, *45*(1), 12–19.

Budnik, M., & Krawczyk, B. (2013). On optimal settings of classification tree ensembles for medical decision support. *Health Informatics Journal*, *19*(1), 3–15.

Bull, S. B., Lewinger, J. P., & Lee, S. S. F. (2007). Confidence intervals for multinomial logistic regression in sparse data. *Statistics in Medicine*, *26*(4), 903–918.

Callaghan, V. (2014). Micro-Futures. In J. C. Augusto & T. Zhang (Eds.), *10th International Conference on Intelligent Environments* (pp. 256–267). Shanghai: IOS Press.

Campbell, M., Hoane, A. J., & Hsu, F.-H. (2002). Deep Blue. *Artificial Intelligence*, *134*, 57–83.

Caruana, R., Crew, G., & Ksikes, A. (2004). Ensemble Selection from Libraries of Models. In *21st International Conference onMachine Learning* (pp. 18–23). Banff, Canada,.

Chan, P. K., & Stolfo, S. J. (1998). Toward Scalable Learning with Non-uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection. In *4th International Conference on Knowledge Discovery and Data Mining* (pp. 164–168). New York.

Chao, X., Kou, G., Li, T., & Peng, Y. (2018). Jie Ke versus AlphaGo: A ranking approach using decision making method for large-scale data with incomplete information. *European Journal of Operational Research*, *265*(1), 239–247.

Chapelle, O., Scholkopf, B., & Zien, Eds., A. (2009). Semi-Supervised Learning. *IEEE Transactions on Neural Networks*, *20*(3), 542–542.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chen, J. X. (2016). The Evolution of Computing: AlphaGo. *Computing in Science & Engineering*, *18*(4), 4–7.

Chen, S., Montgomery, J., & Bolufé-Röhler, A. (2015). Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution.

Applied Intelligence, *42*(3), 514–526.

Chimienti, M., Cornulier, T., Owen, E., Bolton, M., Davies, I. M., Travis, J. M. J., & Scott, B. E. (2016). The use of an unsupervised learning approach for characterizing latent behaviors in accelerometer data. *Ecology and Evolution*, *6*(3), 727–741.

Ciresan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2011). Convolutional Neural Network Committees for Handwritten Character Classification. In *2011 International Conference on Document Analysis and Recognition* (pp. 1135–1139). Beijing: IEEE.

Ciresan, D., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks*, *32*, 333–338.

Clogg, C. C., Rubin, D. B., Schenker, N., Schultz, B., & Weidman, L. (1991). Multiple Imputation of Industry and Occupation Codes in Census Public-use Samples Using Bayesian Logistic Regression. *Journal of the American Statistical Association*, *86*(413), 68–78.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, *20*(1), 37–46.

Cunningham, P., & Carney, J. (2000). Diversity versus Quality in Classification Ensembles Based on Feature Selection. In López de Mántaras R. Plaza E. (Ed.), *European Conference on Machine Learning* (pp. 109–116). Springer, Berlin, Heidelberg.

Dale, A. I. (1988). On Bayes' Theorem and the Inverse Bernoulli Theorem. *Historia Mathematica*, *15*, 348–360.

Darrach, B. (1970). Meet Shaky, the first electronic person. *Life Magazine*, *69*(21), 56–59.

Deo, R. C. (2015). Machine Learning in Medicine. *Circulation*, *132*(20), 1920–1930.

Dickson, B. (2018). What is the AI winter? Retrieved January 25, 2019, from https://bdtechtalks.com/2018/11/12/artificial-intelligence-winter-history/

Dietterich, T. G. (2000a). An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning*, *40*(2), 139–157.

Dietterich, T. G. (2000b). Ensemble Methods in Machine Learning. In *International Workshop on Multiple Classifier Systems* (pp. 1–15). Springer, Berlin, Heidelberg.

Dobbin, K. K., & Simon, R. M. (2011). Optimally splitting cases for training and testing high dimensional classifiers. *BMC Medical Genomics*, *4*(31), 1–8.

Dong, Y., & Peng, C.-Y. J. (2013). Principled missing data methods for researchers. *SpringerPlus*, *2*(1), 222.

Dotson, P. (2013). CPT® Codes: What Are They, Why Are They Necessary, and How Are They Developed? *Advances in Wound Care*, *2*(10), 583–587. https://doi.org/10.1089/wound.2013.0483

Earman, J. (1990). Bayes' Bayesianism. *Studies in History and Philosophy of Science Part A*, *21*(3), 351–370.

Earman, J., & Eisenstaedt, J. (1999). Einstein and Singularities. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, *30*(2), 185–235.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, *542*(7639), 115–118.

Etzioni, O., Banko, M., & Cafarella, M. J. (2006). *Machine Reading*. Washington. Retrieved from www.aaai.org

Famili, A., Shen, W.-M., Weber, R., & Simoudis, E. (1997). Data preprocessing and intelligent data analysis. *Intelligent Data Analysis*, *1*(4), 3–23.

Fatima, M., & Pasha, M. (2017). Survey of Machine Learning Algorithms for Disease Diagnostic. *Journal of Intelligent Learning Systems and Applications*, *09*(01), 1–16.

Feng, Q., Liu, J., & Gong, J. (2015). Urban Flood Mapping Based on Unmanned Aerial Vehicle Remote Sensing and Random Forest Classifier - A Case of Yuyao, China. *Water*, *7*(12), 1437–1455.

Feng, W., Huang, W., Ren, J., Feng, W., Huang, W., & Ren, J. (2018). Class Imbalance Ensemble Learning Based on the Margin Theory. *Applied Sciences*, *8*(5), 815–843.

Fernández, A., López, V., Galar, M., del Jesus, M. J., & Herrera, F. (2013). Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches. *Knowledge-Based Systems*, *42*, 97–110.

Fjeldaas, S., & Lygre Furevik, M. (2016). The Principle of the Stored Program Applied to

Servo Motors. *Procedia CIRP*, *54*, 71–76.

Fluss, R., Faraggi, D., & Reiser, B. (2005). Estimation of the Youden Index and its Associated Cutoff Point. *Biometrical Journal*, *47*(4), 458–472.

Fogg, A. (2017). A History of Deep Learning.

Folorunso, S. O., & Adeyemo, A. B. (2013). Alleviating Classification Problem of Imbalanced Dataset. *African Journal of Computing & ICT*, *6*(2), 137–144.

Foster, K. R., Koprowski, R., & Skufca, J. D. (2014). Machine learning, medical diagnosis, and biomedical engineering research - commentary. *Biomedical Engineering Online*, *13*(94), 1–17.

Frenay, B., & Verleysen, M. (2014a). Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(5), 845–869.

Frenay, B., & Verleysen, M. (2014b). Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(5), 845–869.

French, R. M. (2000). The Turing Test: the first 50 years. *Trends in Cognitive Sciences*, *4*(3), 115–122.

Freund, Y., & Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *13th International Conference on Machine Learning* (pp. 148–156). Morgan Kaufmann.

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139.

Friedman, J. H. (1999). Stochastic Gradient Boosting. *Computational Statistics and Data Analysis*, *38*(1), 367–378.

Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1189–1232.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive Logistic Regression: A Statistical View of Boosting. *The Annals of Statistics*, *28*(2), 337–407.

Gammerman, A. (2010). Modern Machine Learning Techniques and Their Applications to Medical Diagnostics (pp. 2–2). Springer, Berlin, Heidelberg.

García, S., Luengo, J., & Herrera, F. (2016). Tutorial on practical tips of the most influential

data preprocessing algorithms in data mining. *Knowledge-Based Systems*, *98*, 1–29.

García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, *1*(9), 1–22.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, *4*(1), 1–58.

Gillies, D. A. (1987). Was Bayes a Bayesian? *Historia Mathematica*, *14*(4), 325–346.

Golovko, V., Kroshchanka, A., Turchenko, V., Jankowski, S., & Treadwell, D. (2015). A new technique for restricted Boltzmann machine learning. In *8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications* (pp. 182–186). Warsaw: IEEE.

Google. (2019). Geoffrey E. Hinton. Retrieved January 25, 2019, from https://ai.google/research/people/GeoffreyHinton

Graves, A., & Wayne, G. (2014). *Neural Turing Machines*. London.

Guatto, J. (2017). Artificial Intelligence at U of T. Retrieved January 25, 2019, from https://www.utoronto.ca/news/artificial-intelligence-u-t

Gunčar, G., Kukar, M., Notar, M., Brvar, M., Černelč, P., Notar, M., & Notar, M. (2018). An application of machine learning to haematological diagnosis. *Scientific Reports*, *8*(1), Sci. Rep.

Gupta, B., Rawat, A., Jain, A., Arora, A., & Dhami, N. (2017). Analysis of Various Decision Tree Algorithms for Classification in Data Mining. *International Journal of Computer Applications*, *163*(8), 975–8887.

Hady, M. F. A., & Schwenker, F. (2013). Semi-supervised Learning. In M. Bianchini, M. Maggini, & L. C. Jain (Eds.), *Handbook on Neural Information Processing* (1st ed., pp. 215–239). Springer, Berlin, Heidelberg.

Hajian-Tilaki, K. (2013). Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation. *Caspian Journal of Internal Medicine*, *4*(2), 627–635.

Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(10), 993–1001.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data*

Mining, Inference and Prediction (2nd ed.). New York: Springer.

Hecht-Nielson, R. (1992). Theory of the Backpropagation Neural Network. *Neural Networks for Perception*, 65–93.

Hernández, M. A., & Stolfo, S. J. (1998). Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Data Mining and Knowledge Discovery*, *2*(1), 9–37.

Higgins, C. (2017). A Brief History of Deep Blue, IBM's Chess Computer. Retrieved January 25, 2019, from http://mentalfloss.com/article/503178/brief-history-deep-blue-ibms-chess-computer

Hill, R. (2018). Audit of DeepMind deal with NHS trust: It checks out, nothing to see here. Retrieved January 25, 2019, from https://www.theregister.co.uk/2018/06/13/royal_free_deepmind_deal_audit/

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., … Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(8), 832–844.

Hofmann, M., & Klinkenberg, R. (2013). *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Chapman & Hall/CRC. Retrieved from www.RapidMiner.com.

Hogan, E. R. (1977). Robert Adrian: American mathematician. *Historia Mathematica*, *4*(2), 157–172.

Honaker, J., King, G., & Blackwell, M. (2011). Amelia II: A Program for Missing Data. *Journal of Statistical Software*, *45*(7), 1–47.

Hor´kov´, J., & Kelemen, J. (2011). Some Impacts of Karel Capek's Concept of Robots as Artificial Organisms. In *2011 Fourth International Conference on Emerging Trends in Engineering & Technology* (pp. 49–54). Port Louis, Mauritius: IEEE.

Horáková, J., & Kelemen, J. (2009). Artificial living beings and robots: one root, variety of influences. *Artificial Life and Robotics*, *13*(2), 555–560.

Hsu, F.-H. (1999). IBM's Deep Blue Chess grandmaster chips. *IEEE Micro*, *19*(2), 70–81.

Hu, S., Liang, Y., Ma, L., & He, Y. (2009). MSMOTE: Improving classification performance when training data is imbalanced. *2nd International Workshop on Computer Science and*

Engineering, WCSE 2009, *2*, 13–17.

IBM. (2011). Deep Blue. Retrieved January 25, 2019, from https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/team/

Jakobsen, J. C., Gluud, C., Wetterslev, J., & Winkel, P. (2017). When and how should multiple imputation be used for handling missing data in randomised clinical trials – a practical guide with flowcharts. *BMC Medical Research Methodology*, *17*(1), 162.

Jan, M., Awan, A. A., Khalid, M. S., & Nisar, S. (2018). Ensemble approach for developing a smart heart disease prediction system using classification algorithms. *Research Reports in Clinical Cardiology*, *9*, 33–45.

Johnson, R. G. (2010). Andrew D. Booth – Britain's Other "Fourth Man." In A. Tatnall (Ed.), *IFIP International Federation for Information Processing 2010* (Vol. 325, pp. 26–37). Brisbane, Australia: Springer.

Jutel, A. (2011). Classification, Disease, and Diagnosis. *Perspectives in Biology and Medicine*, *54*(2), 189–205.

Kang, H. (2013). The prevention and handling of the missing data. *Korean Journal of Anesthesiology*, *64*(5), 402–406.

Kavakiotis, I., Tsave, O., Salifoglou, A., Maglaveras, N., Vlahavas, I., & Chouvarda, I. (2017). Machine Learning and Data Mining Methods in Diabetes Research. *Computational and Structural Biotechnology Journal*, *15*, 104–116.

Khalilia, M., Chakraborty, S., & Popescu, M. (2011). Predicting disease risks from highly imbalanced data using random forest. *BMC Medical Informatics and Decision Making*, *11*(1), 51.

Khayut, B., Fabri, L., & Avikhana, M. (2016). Modeling of Computational Systemic Deep Mind Under Uncertainty. *Procedia Computer Science*, *95*, 135–144.

Kingsford, C., & Salzberg, S. L. (2008). What are decision trees? *Nature Biotechnology*, *26*(9), 1011–1013.

Kononenko, I. (2001). Machine learning for medical diagnosis: History, state of the art and perspective. *Artificial Intelligence in Medicine*, *23*(1), 89–109.

Korukonda, A. R. (2003). Taking stock of Turing test: a review, analysis, and appraisal of issues surrounding thinking machines. *International Journal of Human-Computer*

Studies, *58*(2), 240–257.

Kostick, K. (2017). Of man and machine: The evolution of transhumanism. Retrieved January 25, 2019, from https://blogs.bcm.edu/2017/03/22/man-machine-evolution-transhumanism/

Kotsiantis, S. (2011). Combining bagging, boosting, rotation forest and random subspace methods. *Artificial Intelligence Review*, *35*(3), 223–240.

Kotsiantis, S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, *31*, 249–268.

Kotsiantis, S. B., & Pintelas, P. E. (2004). Combining Bagging and Boosting. *Computational Intelligence*.

Kowarik, A., & Templ, M. (2016). Imputation with the R Package VIM. *Journal of Statistical Software*, *74*(7), 1–16.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221–232.

Krogh, A., & Vedelsby, J. (1994). Neural Network Ensembles, Cross Validation, and Active Learning. In G. Tesauro, D. S. Touretzky, & T. K. Leen (Eds.), *7th International Conference on Neural Information Processing Systems* (pp. 231–238). Cambridge, MA: MIT Press.

Kübler, S., Liu, C., & Sayyed, Z. A. (2018). To use or not to use: Feature selection for sentiment analysis of highly imbalanced data. *Natural Language Engineering*, *24*(1), 3–37.

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, *28*(5), 1–24.

Kuipers, B., Feigenbaum, E. A., Hart, P. E., & Nilsson, N. J. (2017). Shakey: From Conception to History. *AI Magazine*, *38*(1), 88–103.

Kumar, G. R., Kongara, V. S., & Ramachandra, G. A. (2013). An Efficient Ensemble Based Classification Techniques for Medical Diagnosis. *International Journal of Latest Technology in Engineering, Management & Applied Science*, *2*(8), 5–9.

Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., & Duin, R. P. W. (2003). Limits on the Majority Vote Accuracy in Classifier Fusion. *Pattern Analysis and Applications*, *6*, 22–

31.

Kurzweil, R. (2005). *The Singularity Is Near: When Humans Transcend Biology*. (R. Kot & C. Ferraro, Eds.) (1st ed.). London: Penguin.

Lam, L., & Suen, S. Y. (1997). Application of majority voting to pattern recognition: an analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *27*(5), 553–568.

Lavanya D, & Rani, U. (2012). Ensemble Decision Tree Classifier For Breast Cancer Data. *International Journal of Information Technology Convergence and Services*, *2*(1), 17–24.

Le Roux, N., & Bengio, Y. (2008). Representational Power of Restricted Boltzmann Machines and Deep Belief Networks. *Neural Computation*, *20*(6), 1631–1649.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Ledesma, L. de, Pérez, A., Borrajo, D., & Laita, L. M. (1997). A computational approach to George Boole's discovery of mathematical logic. *Artificial Intelligence*, *91*(2), 281–307.

Lee, G., Lee, H. B., Jung, B. H., & Nam, H. (2017). mvp - an open-source preprocessor for cleaning duplicate records and missing values in mass spectrometry data. *FEBS Open Bio*, *7*(7), 1051–1059.

Lee, H.-M., Chen, C.-M., & Huang, T.-C. (2001). Learning efficiency improvement of back-propagation algorithm by error saturation prevention method. *Neurocomputing*, *41*(1–4), 125–143.

Lee, P. (2018). Microsoft's focus on transforming healthcare: Intelligent health through AI and the cloud - The Official Microsoft Blog. Retrieved January 25, 2019, from https://blogs.microsoft.com/blog/2018/02/28/microsofts-focus-transforming-healthcare-intelligent-health-ai-cloud/

Leevy, J. L., Khoshgoftaar, T. M., Bauder, R. A., & Seliya, N. (2018). A survey on addressing high-class imbalance in big data. *Journal of Big Data*, *5*(42), 1–30.

Li, H., Xiong, L., Ohno-Machado, L., & Jiang, X. (2014). Privacy preserving RBF kernel support vector machine. *BioMed Research International*, *2014*, 827371.

Li, P., Stuart, E. A., & Allison, D. B. (2015). Multiple Imputation: A Flexible Tool for

Handling Missing Data. *JAMA*, *314*(18), 1966–1967.

Lin, C. T., Hsieh, T. Y., Liu, Y. T., Lin, Y. Y., Fang, C. N., Wang, Y. K., … Chuang, C. H. (2018). Minority Oversampling in Kernel Adaptive Subspaces for Class Imbalanced Datasets. *IEEE Transactions on Knowledge and Data Engineering*, *30*(5), 950–962.

Lip, G. Y. H., Nieuwlaat, R., Pisters, R., Lane, D. A., & Crijns, H. J. G. M. (2010). Refining Clinical Risk Stratification for Predicting Stroke and Thromboembolism in Atrial Fibrillation Using a Novel Risk Factor-Based Approach. *Chest*, *137*(2), 263–272.

Little, R. J. A., & Rubin, D. B. (1991). Statistical Analysis with Missing Data. *Journal of Educational Statistics*, *16*(2), 150–155.

Little, R. J., & Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. *Statistical analysis with missing data Second edition*.

Little, R., & Rubin, D. (1987). *Statistical analysis with missing data* (1st ed.). Canada: Wiley.

Little, R., & Rubin, D. (2002). *Statistical Analysis with Missing Data*. (D. J. Balding & P. Bloomfiel, Eds.), *John Wiley & Sons, Inc.* (2nd ed.). Hoboken, New Jersey.

Liu, A. Y. (2004). *The Effect of Oversampling and Undersampling on Classifying Imbalanced Text Datasets*. The University of Texas at Austin.

Liu, T., & Tao, D. (2014). Classification with Noisy Labels by Importance Reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(3), 1–19.

Liu, Z., Tang, D., Cai, Y., Wang, R., & Chen, F. (2017). A hybrid method based on ensemble WELM for handling multi class imbalance in cancer microarray data. *Neurocomputing*, *266*, 641–650.

Lloyd, E. H. (1974). What is, and what is not, a Markov chain? *Journal of Hydrology*, *22*(1–2), 1–28.

Luengo, J., Shim, S.-O., Alshomrani, S., Altalhi, A., & Herrera, F. (2018). CNC-NOS: Class noise cleaning by ensemble filtering and noise scoring. *Knowledge-Based Systems*, *140*, 27–49.

Magee, C. L., & Devezas, T. C. (2011). How many singularities are near and how will they disrupt human history? *Technological Forecasting and Social Change*, *78*(8), 1365–1378.

Magoulas, G. D., & Prentza, A. (2001). Machine Learning in Medical Applications (pp. 300–

307). Springer, Berlin, Heidelberg.

Maimon, O., & Rokach, L. (2005). *Data Mining and Knowledge Discovery Handbook*. Heidelberg: Springer-Verlag Berlin.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing* (2nd ed.). Cambridge: MIT Press.

Markoff, J. (1990). Business Technology; What's the Best Answer? It's Survival of the Fittest. Retrieved March 5, 2019, from https://www.nytimes.com/1990/08/29/business/business-technology-what-s-the-best-answer-it-s-survival-of-the-fittest.html

Martinez, J. (2018). Amelia: The First Line of Defense for Data Security. Retrieved January 25, 2019, from https://www.ipsoft.com/2018/04/12/amelia-the-first-line-of-defense-for-data-security/

Mathuria, M. (2013). Decision Tree Analysis on J48 Algorithm for Data Mining. *Intrenational Journal of Advanced Research in Computer Science and Software Engineering*, *3*(6), 1114–1119.

McCarthy, J. (1960). *Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I*. Cambridge. Retrieved from http://www-formal.stanford.edu/jmc/

McCarthy, J. (1974). Artificial intelligence: a paper symposium: Professor Sir James Lighthill, FRS. Artificial Intelligence: A General Survey. In: Science Research Council, 1973. *Artificial Intelligence*, *5*(3), 317–322.

McCarthy, J. (1979). *History of Lisp*. California. Retrieved from http://jmc.stanford.edu/articles/lisp/lisp.pdf

McCarthy, J. (2000). *Review of Artificial Intelligence: A General Survey*. Stanford. Retrieved from http://www-formal.stanford.edu/jmc/reviews/lighthill.pdf

McCoy, J. P., & Ullman, T. D. (2018). A Minimal Turing Test. *Journal of Experimental Social Psychology*, *79*, 1–8.

McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia Medica*, *22*(3), 276–282.

Minh, H. (2018). How to Handle Imbalanced Data in Classification Problems. Retrieved July 26, 2019, from https://medium.com/james-blogs/handling-imbalanced-data-in-

classification-problems-7de598c1059f

Minsky, M. (1961). Steps Toward Artificial Intelligence. *Proceedings of the IRE*, *49*(1), 8–30.

Mochari, I. (2016). Remembering Marvin Minsky, a Founding Father of Artificial Intelligence. Retrieved January 25, 2019, from https://www.inc.com/ilan-mochari/marvin-minsky-artificial-intelligence-pioneer.html

Moon, H., Ahn, H., Kodell, R. L., Baek, S., Lin, C.-J., & Chen, J. J. (2007). Ensemble methods for classification of patients for personalized medicine with high-dimensional data. *Artificial Intelligence in Medicine*, *41*(3), 197–207.

Morales, P., Luengo, J., Garcia, L. P. F., Lorena, A. C., Carvalho, A. C. P. L. ., & Herrera, F. (2017). The NoiseFiltersR Package: Label Noise Preprocessing in R. *The R Journal*, *9*(1), 219–228.

Morales, P., Luengo, J., Garcia, L. P. F., Lorena, A. C., De Carvalho, A., & Herrera, F. (2017). The NoiseFiltersR Package: Label Noise Preprocessing in R. *The R Journal*, *9*(1), 219–228.

Murphy, K. P. (2012). *Machine learning: A Probabilistic Perspective* (1st ed.). Cambridge: MIT Press. Retrieved from https://mitpress.mit.edu/books/machine-learning-1

Nakagawa, S., & Freckleton, R. P. (2011). Model averaging, missing data and multiple imputation: a case study for behavioural ecology. *Behavioral Ecology and Sociobiology*, *65*(1), 103–116.

Natarajan, N., Dhillon, I. S., Ravikumar, P., & Tewari, A. (2018). Cost-Sensitive Learning with Noisy Labels. *Journal of Machine Learning Research*, *18*, 1–33. Retrieved from http://jmlr.org/papers/v18/15-226.html.

Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, *7*(21), 1–20.

Neumann, J., Mauchly, J. W., Burks, A. W., Goldstine, H. H., Williams, F. C., Kilburn, T., … Renwick, W. (1973). Stored Program Electronic Computers. In *The Origins of Digital Computers* (pp. 349–401). Berlin, Heidelberg: Springer Berlin Heidelberg.

Noorhalim, N., Ali, A., & Shamsuddin, S. M. (2019). Handling Imbalanced Ratio for Class Imbalance Problem Using SMOTE. In L. Kor, A. Ahmad, Idrus Z., & Mansor K. (Eds.), *Third International Conference on Computing, Mathematics and Statistics* (pp. 19–30).

Singapore: Springer Singapore.

Norman, J. (2004). IBM's SSEC, the First Computer that Can Modify a Stored Program. Retrieved January 24, 2019, from http://www.historyofinformation.com/detail.php?entryid=851.

O'Mahony, C., Jichi, F., Pavlou, M., Monserrat, L., Anastasakis, A., Rapezzi, C., … Hypertrophic Cardiomyopathy Outcomes Investigators. (2014). A novel clinical risk prediction model for sudden cardiac death in hypertrophic cardiomyopathy (HCM Risk-SCD). *European Heart Journal*, *35*(30), 2010–2020.

Ongsulee, P. (2017). Artificial intelligence, machine learning and deep learning. In *15th International Conference on ICT and Knowledge Engineering* (pp. 1–6). Bangkok: IEEE.

Opitz, D., & Maclin, R. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*, *11*, 169–198. Retrieved from https://www.d.umn.edu/~rmaclin/publications/opitz-jair99.pdf

Osamor, P. E., & Grady, C. (2016). Women's autonomy in health care decision-making in developing countries: a synthesis of the literature. *International Journal of Women's Health*, *8*, 191–202.

Pan, Q., Wei, R., Shimizu, I., & Jamoom, E. (2014). Determining Sufficient Number of Imputations Using Variance of Imputation Variances: Data from 2012 NAMCS Physician Workflow Mail Survey. *Applied Mathematics*, *5*, 3421–3430.

Pan, Y. (2016). Heading toward Artificial Intelligence 2.0. *Engineering*, *2*(4), 409–413.

Papageorgiou, G., Grant, S. W., Takkenberg, J. J. M., & Mokhles, M. M. (2018). Statistical primer: how to deal with missing data in scientific research? *Interactive CardioVascular and Thoracic Surgery*, *27*(2), 153–158.

Pathak, A. R., Pandey, M., & Rautaray, S. (2018). Application of Deep Learning for Object Detection. *Procedia Computer Science*, *132*, 1706–1717.

Poitras, G. (2013). Richard Price, miracles and the origins of Bayesian decision theory. *The European Journal of the History of Economic Thought*, *20*(1), 29–57.

Popoveniuc, B. (2013). Pro and cons singularity: Kurzweil's theory and its critics. In *Proceedings of the Virtual Reality International Conference* (pp. 1–6). New York: ACM

Press.

Pozzolo, D., Caelen, O., Maintainer, G. B., & Dal Pozzolo, A. (2013). *Unbalanced: Racing for Unbalanced Methods Selection*. *Intelligent Data Engineering and Automated Learning - IDEAL 2013*. Springer Berlin Heidelberg. Retrieved from https://cran.r-project.org/web/packages/unbalanced/unbalanced

Press, G. (2016). A Very Short History Of Artificial Intelligence (AI).

Provost, F. J., & Kolluri, V. (1997). A Survey of Methods for Scaling Up Inductive Learning Algorithms Primary contact. In *3rd International Conference on Knowledge Discovery and Data Mining*.

Provost, F., & Kohavi, R. (1998). On Applied Research in Machine Learning. *Machine Learning*, *30*(2/3), 271–274.

Qahtan, A. A., Elmagarmid, A., Castro, F. R., Ouzzani, M., & Tang, N. (2018). FAHES: A Robust Disguised Missing Values Detector. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18* (pp. 2100–2109). New York, New York, USA: ACM Press.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Quinlan, J. Ross. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Quinlan, J.R. (1999). Simplifying decision trees. *International Journal of Human-Computer Studies*, *51*(2), 497–510.

Quinlan, J R. (1996). Bagging, Boosting, and C4.5. In *Thirteenth National Conference on Artificial Intelligence* (pp. 725–730). Retrieved from www.aaai.org

R Development Core Team. (2011). R: The R Project for Statistical Computing. Retrieved July 27, 2019, from https://www.r-project.org/

Rao, K., Sri, K., & Sai, G. (2016). A Novel Video CAPTCHA Technique To Prevent BOT Attacks. *Procedia Computer Science*, *85*, 236–240.

Raval, D., Bhatt, D., Kumhar, M. K., Parikh, V., & Vyas, D. (2015). Medical Diagnosis System Using Machine Learning. *International Journal of Computer Science & Communication*, *7*, 177–182.

Rezvan, P. H., Lee, K. J., & Simpson, J. A. (2015). The rise of multiple imputation: a review

of the reporting and implementation of the method in medical research. *BMC Medical Research Methodology*, *15*, 30–73.

Roberts, K., Boland, M. R., Pruinelli, L., Dcruz, J., Berry, A., Georgsson, M., … Brennan, P. F. (2016). Biomedical informatics advancing the national health agenda: the AMIA 2015 year-in-review in clinical and consumer informatics. *Journal of the American Medical Informatics Association*, *24*(1), 185–190.

Rubin, D. (1987). *Multiple imputation for nonresponse in surveys* (2nd ed.). Canada: Wiley.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

Sáez, J. A., Galar, M., Luengo, J., & Herrera, F. (2013). Tackling the problem of classification with noisy data using Multiple Classifier Systems: Analysis of the performance and robustness. *Information Sciences*, *247*(1), 1–20.

Sáez, J. A., Krawczyk, B., & Woźniak, M. (2016). On the Influence of Class Noise in Medical Data Classification: Treatment Using Noise Filtering Methods. *Applied Artificial Intelligence*, *30*(6), 590–609.

Sáez, J. A., Luengo, J., Stefanowski, J., & Herrera, F. (2015). SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, *291*, 184–203.

Salzberg, S. L. (1994). C4.5: Programs for Machine Learning. *Machine Learning*, *16*(3), 235–240.

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*(3), 210–229.

Schafer, J.L. (1997). *Analysis of Incomplete Multivariate Data* (1st ed.). London: Chapman & Hall.

Schafer, Joseph L., & Olsen, M. K. (1998). Multiple Imputation for Multivariate Missing-Data Problems: A Data Analyst's Perspective. *Multivariate Behavioral Research*, *33*(4), 545–571.

Schafer, Joseph L. (1999). Multiple imputation: a primer. *Statistical Methods in Medical Research*, *8*(1), 3–15.

Schafer, Joseph L, & Graham, J. W. (2002). Missing data: our view of the state of the art.

Psychological Methods, *7*(2), 147–177.

Schaffer, C. (1993). Selecting a Classification Method by Cross-Validation. *Machine Learning*, *13*(1), 135–143.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227. https://doi.org/10.1007/BF00116037

Schmidhuber, J. (2014). *Deep Learning in Neural Networks: An Overview*. Manno Lugano. Retrieved from http://www.idsia.ch/˜juergen/DeepLearning8Oct2014.texCompleteBIBTEXfile

Sewell, M. (2007). *Ensemble Methods*. London. Retrieved from https://pdfs.semanticscholar.org/2cb6/dbf69c43e0c7b863bef8f38007582d7d1203.pdf

Sharma, N., Jain, V., & Mishra, A. (2018). An Analysis Of Convolutional Neural Networks For Image Classification. *Procedia Computer Science*, *132*, 377–384.

Shelburne, B. J., & Burton, C. P. (1998). Early programs on the Manchester Mark I Prototype. *IEEE Annals of the History of Computing*, *20*(3), 4–15.

Shuo Wang, & Xin Yao. (2013). Relationships between Diversity of Classification Ensembles and Single-Class Performance Measures. *IEEE Transactions on Knowledge and Data Engineering*, *25*(1), 206–219.

Sidey-Gibbons, J. A. M., & Sidey-Gibbons, C. J. (2019). Machine learning in medicine: a practical introduction. *BMC Medical Research Methodology*, *19*(64), 1–18.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., … Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* , *362*(6419), 1140–1144.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., … Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, *550*(7676), 354–359.

Simonite, T. (2017). Google's AI eye doctor gets ready to go to work in India. Retrieved January 30, 2019, from https://www.wired.com/2017/06/googles-ai-eye-doctor-gets-ready-go-work-india/

Sinharay, S, Stern, H. S., & Russell, D. (2001). The use of multiple imputation for the analysis of missing data. *Psychological Methods*, *6*(4), 317–329.

Sinharay, Sandip, Stern, H. S., & Russell, D. (2001). The Use of Multiple Imputation for the Analysis of Missing Data. *Psychological Methods*, *6*(4), 317–329.

Sirovich, B., Gallagher, P. M., Wennberg, D. E., & Fisher, E. S. (2008). Discretionary decision making by primary care physicians and the cost of U.S. Health care. *Health Affairs (Project Hope)*, *27*(3), 813–823.

Song, Y.-Y., & Lu, Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai Archives of Psychiatry*, *27*(2), 130–135.

Speck, D., Dornhege, C., & Burgard, W. (2017). Shakey 2016 - How Much Does it Take to Redo Shakey the Robot? *IEEE Robotics and Automation Letters*, *2*(2), 1203–1209.

Srimani, P. K., & Koti, M. S. (2013). Medical Diagnosis Using Ensemble Classifiers - A Novel Machine-Learning Approach. *Journal of Advanced Computing*, *1*, 9–27.

Stigler, S. M. (1981). Gauss and the Invention of Least Squares. *The Annals of Statistics*, *9*(3), 465–474.

Subbulakshmi, C. V., & Deepa, S. N. (2015). Medical Dataset Classification: A Machine Learning Paradigm Integrating Particle Swarm Optimization with Extreme Learning Machine Classifier. *The Scientific World Journal*, *2015*, 1–12.

Sutton, C. D. (2005). Classification and Regression Trees, Bagging, and Boosting. In R. C. Rao & A. S. R. S. Rao (Eds.), *Handbook of Statistics* (24th ed., pp. 303–330). Amsterdam: Elsevier.

Swets, J. (1988). Measuring the accuracy of diagnostic systems. *Science*, *240*(4857), 1285–1293.

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9). Boston: IEEE.

Szolovits, P., & Pauker, S. G. (1978). Categorical and probabilistic reasoning in medical diagnosis. *Artificial Intelligence*, *11*(1–2), 115–144.

Takano, S. (1973). Interaction information and Markov chain. *Journal of Multivariate Analysis*, *3*(1), 93–101.

Thazin, H., Thein, T., Mo, K., & Tun, M. (2014). Medical Diagnosis Classification Using Migration Based Differential Evolution Algorithm, 153–163.

Tomar, D., & Agarwal, S. (2014). A Survey on Pre-processing and Post-processing Techniques in Data Mining. *International Journal of Database Theory and Application*, *7*(4), 99–128.

Tomek, I. (1976). Two Modifications of CNN. *IEEE Transactions on Systems, Man, and Cybernetics*, *6*(11), 769–772.

Triguero, I., González, S., Moyano, J. M., García, S., Jes´, J., Alcalá-Fdez, J., Herrera, F. (2017). KEEL 3.0: An Open Source Software for Multi-Stage Analysis in Data Mining. *International Journal of Computational Intelligence Systems*, *10*, 1238–1249.

Tumer, K., & Ghosh, J. (1995). *Theoretical Foundations Of Linear And Order Statistics Combiners For Neural Pattern Classifiers*. Austin.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, *59*(236), 433–460.

Türk Dil Kurumu. (2006). Tanı. Retrieved February 1, 2019, from http://www.tdk.gov.tr/index.php?option=com_gts&arama=gts&guid=TDK.GTS.5c540eb88958a5.60838109

UCI. (2019). UC Irvine Machine Learning Repository. Retrieved July 27, 2019, from https://archive.ics.uci.edu/ml/index.php

Ueda, N., & Nakano, R. (1996). Generalization error of ensemble estimators. In *International Conference on Neural Networks* (Vol. 1, pp. 90–95). Washington: IEEE.

Vamsidhar, E., Varma, K., Rao, S., & Satapati, R. (2010). Prediction of Rainfall Using Backpropagation Neural Network Model. *International Journal on Computer Science and Engineering*, *2*(4), 1119–1121.

van Buuren, S, & Oudshoorn, K. (2000). *Multivariate Imputation by Chained Equations MICE V1.0 User's manual*. Amsterdam. Retrieved from https://stefvanbuuren.name/publications/MICE V1.0 Manual TNO00038 2000.pdf

van Buuren, Stef, & Oudshoorn, K. (2011). mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, *45*(3), 1–67.

Vaughan, R. (2017). Oversampling in Health Surveys: Why, When, and How? *American Journal of Public Health*, *107*(8), 1214–1215.

Verma, B., & Hassan, Z. S. (2011). Hybrid ensemble approach for classification. *Appl Intell*, *34*, 258–278.

Verwijnen, J. (2016). *Marvin Minsky - the father of AI*. Helsinki. Retrieved from https://pdfs.semanticscholar.org/ef31/300364bb686178d7013f8d53ae91da26bac9.pdf

Vinge, V. (1993). The Coming Technological Singularity: How to Survive in the Post-Human Era. In G. A. Landis (Ed.), *Vision-21: Interdisciplinary Science and Engineering in the Era of Cyberspace* (pp. 11–22). Westlake: NASA Publication.

Vinge, V. (2008). Signs of the Singularity. Retrieved January 25, 2019, from https://spectrum.ieee.org/biomedical/ethics/signs-of-the-singularity

Walsh, T. (2017). The Singularity May Never Be Near. *AI Magazine*, *38*(3), 58–62.

Wan, S., Duan, Y., & Zou, Q. (2017). HPSLPred: An Ensemble Multi-Label Classifier for Human Protein Subcellular Location Prediction with Imbalanced Source. *Proteomics*, *17*(17–18), 1–12.

Wang, R. Y., Storey, V. C., & Firth, C. P. (1995). A Framework for Analysis of Data Quality Research. *IEEE Transactions on Knowledge and Data Engineering* , *7*(4), 623–640.

Ward, M. (2010). Giving birth to a binary "Baby." Retrieved January 24, 2019, from http://news.bbc.co.uk/2/hi/technology/8493062.stm

Watson, P. F., & Petrie, A. (2010). Method agreement analysis: A review of correct methodology. *Theriogenology*, *73*(9), 1167–1179.

Weverbergh, R. (2017). The world's first game developer: Leonardo Torres y Quevedo. Retrieved January 24, 2019.

White, I. R., Daniel, R., & Royston, P. (2010). Avoiding bias due to perfect prediction in multiple imputation of incomplete categorical variables. *Computational Statistics & Data Analysis*, *54*(10), 2267–2275.

Wickremasinghe, D., Hashmi, I. E., Schellenberg, J., & Avan, B. I. (2016). District decision-making for health in low-income settings: a systematic literature review. *Health Policy and Planning*, *31*(2), 12–24.

Wight, R. D., & Gable, P. A. (2005). Gauss, Johann Carl Friedrich. *Encyclopedia of Statistics in Behavioral Science*, *2*, 694–696.

Wilkes, M. V. (1997). Arithmetic on the EDSAC. *IEEE Annals of the History of Computing*, *19*(1), 13–15.

Wilson, D. L. (1972). Asymptotic Properties of Nearest Neighbor Rules Using Edited Data.

IEEE Transactions on Systems, Man, and Cybernetics, *2*(3), 408–421.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques* (2nd ed.). San Francisco: Morgan Kaufmann.

Witten, I. H., Frank, E., & Hall, M. a. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. *Annals of Physics* (Vol. 54).

Wood, A. M., White, I. R., & Thompson, S. G. (2004). Are missing outcome data adequately handled? A review of published randomized controlled trials in major medical journals. *Clinical Trials: Journal of the Society for Clinical Trials*, *1*(4), 368–376.

Wu, X., & Zhu, X. (2008). Mining With Noise Knowledge: Error-Aware Data Mining. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *38*(4), 917–932.

Wu, Z., Lin, W., & Ji, Y. (2018). An Integrated Ensemble Learning Model for Imbalanced Fault Diagnostics and Prognostics, *6*.

Xu, B., Huang, J. Z., Williams, G., Wang, Q., & Ye, Y. (2012). Classifying Very High-Dimensional Data with Random Forests Built from Small Subspaces. *International Journal of Data Warehousing and Mining*, *8*(2), 44–63.

Xu, Y., & Goodacre, R. (2018). On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. *Journal of Analysis and Testing*, *2*(3), 249–262.

Yang, P., Yang, J. Y. H., Zhou, B., & Zomaya, A. (2010). A Review of Ensemble Methods in Bioinformatics. *Current Bioinformatics*, *5*(4), 296–308.

Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., & Soderland, S. (2007). *TextRunner: Open Information Extraction on the Web*. New York.

Youden, W. J. (1950). Index for rating diagnostic tests. *Cancer*, *3*(1), 32–35.

Yu, K.-H., Beam, A. L., & Kohane, I. S. (2018). Artificial intelligence in healthcare. *Nature Biomedical Engineering*, *2*(10), 719–731. https://doi.org/10.1038/s41551-018-0305-z

Zhang, J., Cui, X., Li, J., & Wang, R. (2017). Imbalanced classification of mental workload using a cost-sensitive majority weighted minority oversampling strategy. *Cognition, Technology and Work*, *19*(4), 633–653.

Zhang, P., Zhu, X., Shi, Y., & Wu, X. (2009). An Aggregate Ensemble for Mining Concept Drifting Data Streams with Noise. In *Advances in Knowledge Discovery and Data Mining* (pp. 1021–1029). Springer, Berlin, Heidelberg.

Zhang, Y., Liu, B., Cai, J., & Zhang, S. (2016). Ensemble weighted extreme learning machine for imbalanced data classification based on differential evolution. *Neural Computing and Applications*, *28*(s1), 1–9.

Zhang, Z., Zhao, Y., Canes, A., Steinberg, D., & Lyashevska, O. (2019). Predictive analytics with gradient boosting in clinical medicine. *Annals of Translational Medicine*, *7*(7), 152–159.

Zhao, H., Williams, G. J., & Huang, J. Z. (2017). wsrf: An R Package for Classification with Scalable Weighted Subspace Random Forests. *Journal of Statistical Software*, *77*(3), 1–30.

Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Cambridge: CRC Press.

Zhu, Xiaojin. (2008). Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, *2*(3), 1–60.

Zhu, Xingquan, & Wu, X. (2004). Class Noise vs. Attribute Noise: A Quantitative Study. *Artificial Intelligence Review*, *22*(3), 177–210.

Zhu, Xingquan, Wu, X., & Chen, Q. (2003). Eliminating class noise in large datasets. In *Twentieth International Conference on International Conference on Machine Learning* (pp. 920–927). Washington, DC, USA: AAAI Press.

Zhu, Xingquan, Wu, X., & Chen, Q. (2006). Bridging Local and Global Data Cleansing: Identifying Class Noise in Large, Distributed Data Datasets. *Data Mining and Knowledge Discovery*, *12*, 275–308.

Ekler

Ek 1: Orijinal Verilerin Analizi R Kodları

```
library(caret)
#cleveland
clevelanddata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/cleveland_stddata.csv"), sep = ";",
header = T)
#data partition
set.seed(1234)
trainIndex <- createDataPartition(clevelanddata$Class, p = .7, list = FALSE, times = 1)
head(trainIndex)
clevelandTrain <- clevelanddata[ trainIndex,]
clevelandTest  <- clevelanddata[-trainIndex,]
#Basic Parameter Tuning (10-fold CV / repeated ten times)
set.seed(1234)
fitControl <- trainControl(method = "repeatedcv", number = 10, repeats = 10)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(clevelandTrain[,1:13]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = clevelandTrain, method = "rf", trControl = fitControl,verbose = FALSE, tuneGrid
= tunegrid)
fitRF$finalModel
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, clevelandTrain)
confusionMatrix(trainRF_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, clevelandTest)
confusionMatrix(testRF_predict, clevelandTest$Class, positive = '0')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(clevelandTrain[,1:13]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = clevelandTrain, method = "wsrf", trControl = fitControl,
verbose = FALSE, tuneGrid = tunegrid)
fitwsrf$finalModel
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, clevelandTrain)
confusionMatrix(trainwsrf_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, clevelandTest)
confusionMatrix(testwsrf_predict, clevelandTest$Class, positive = '0')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(clevelandTrain[,1:13])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = clevelandTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE,tuneGrid = tunegrid)
fitlogitboost
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, clevelandTrain)
confusionMatrix(trainlogitboost_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, clevelandTest)
confusionMatrix(testlogitboost_predict, clevelandTest$Class, positive = '0')
```

```
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:13)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = clevelandTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
fitgbm
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, clevelandTrain)
confusionMatrix(traingbm_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, clevelandTest)
confusionMatrix(testgbm_predict, clevelandTest$Class, positive = '0')
resamps <- resamples(list(RF = fitRF, WSRF = fitwsrf, logitboost = fitlogitboost, gbm = fitgbm))
resamps
summary(resamps)
theme1 <- trellis.par.get()
theme1$plot.symbol$col = rgb(.2, .2, .2, .4)
theme1$plot.symbol$pch = 16
theme1$plot.line$col = rgb(1, 0, 0, .7)
theme1$plot.line$lwd <- 2
trellis.par.set(theme1)
bwplot(resamps, layout = c(3, 1))
splom(resamps)
#heart
heartdata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/heart_stddata.csv"),
                     sep = ";",  header = T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(heartTrain[,1:13]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = heartTrain, method = "rf", trControl = fitControl, verbose = FALSE, tuneGrid =
tunegrid)
fitRF
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, heartTrain)
confusionMatrix(trainRF_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, heartTest)
confusionMatrix(testRF_predict, heartTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(heartTrain[,1:13]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = heartTrain, method = "wsrf", trControl = fitControl, verbose = FALSE, tuneGrid
= tunegrid)
fitwsrf$finalModel
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, heartTrain)
confusionMatrix(trainwsrf_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, heartTest)
confusionMatrix(testwsrf_predict, heartTest$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(heartTrain[,1:13])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
```

```
fitlogitboost <- train(Class ~ ., data = heartTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
fitlogitboost
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, heartTrain)
confusionMatrix(trainlogitboost_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, heartTest)
confusionMatrix(testlogitboost_predict, heartTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:13)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = heartTrain, method = "gbm", trControl = fitControl, verbose = FALSE, tuneGrid
= tunegrid)
fitgbm
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, heartTrain)
confusionMatrix(traingbm_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, heartTest)
confusionMatrix(testgbm_predict, heartTest$Class, positive = '1')
#hepatitis
hepatitisdata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/hepatitis_stddata.csv"),  sep = ";",
header = T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(hepatitisTrain[,1:19]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = hepatitisTrain, method = "rf", trControl = fitControl, verbose = FALSE, tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, hepatitisTrain)
confusionMatrix(trainRF_predict, hepatitisTrain$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(hepatitisTrain[,1:19]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = hepatitisTrain, method = "wsrf", trControl = fitControl,  verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, hepatitisTrain)
confusionMatrix(trainwsrf_predict, hepatitisTrain$Class, positive = '1')
library(caTools)

#logitboost expand.grid / parameters tuning
niter <- ncol(hepatitisTrain[,1:19])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = hepatitisTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, hepatitisTrain)
confusionMatrix(trainlogitboost_predict, hepatitisTrain$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:19)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
```

```
fitgbm <- train(Class ~ ., data = hepatitisTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, hepatitisTrain)
confusionMatrix(traingbm_predict, hepatitisTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, hepatitisTest)
confusionMatrix(testgbm_predict, hepatitisTest$Class, positive = '1')
#lymphography
lymphographydata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/lymphography_stddata.csv"),
sep = ";",  header = T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(lymphographyTrain[,1:18]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = lymphographyTrain, method = "rf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, lymphographyTrain)
confusionMatrix(trainRF_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, lymphographyTest)
confusionMatrix(testRF_predict, lymphographyTest$Class, positive = 'normal')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(lymphographyTrain[,1:18]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = lymphographyTrain, method = "wsrf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, lymphographyTrain)
confusionMatrix(trainwsrf_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, lymphographyTest)
confusionMatrix(testwsrf_predict, lymphographyTest$Class, positive = 'normal')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(lymphographyTrain[,1:18])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = lymphographyTrain, method = "LogitBoost", trControl = fitControl,
verbose = FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, lymphographyTrain)
confusionMatrix(trainlogitboost_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, lymphographyTest)
confusionMatrix(testlogitboost_predict, lymphographyTest$Class, positive = 'normal')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:18)*50, shrinkage = 0.1, n.minobsinnode =10)

set.seed(825)
fitgbm <- train(Class ~ ., data = lymphographyTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, lymphographyTrain)
confusionMatrix(traingbm_predict, lymphographyTrain$Class, positive = 'normal')
```

```
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, lymphographyTest)
confusionMatrix(testgbm_predict, lymphographyTest$Class, positive = 'normal')
#mammographic
mammographicdata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/mammographic_stddata.csv"),
sep = ";",  header = T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(mammographicTrain[,1:5]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = mammographicTrain, method = "rf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, mammographicTrain)
confusionMatrix(trainRF_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, mammographicTest)
confusionMatrix(testRF_predict, mammographicTest$Class, positive = '0')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(mammographicTrain[,1:5]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = mammographicTrain, method = "wsrf", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, mammographicTrain)
confusionMatrix(trainwsrf_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, mammographicTest)
confusionMatrix(testwsrf_predict, mammographicTest$Class, positive = '0')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(mammographicTrain[,1:5])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = mammographicTrain, method = "LogitBoost", trControl = fitControl,
verbose = FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, mammographicTrain)
confusionMatrix(trainlogitboost_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, mammographicTest)
confusionMatrix(testlogitboost_predict, mammographicTest$Class, positive = '0')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:5)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = mammographicTrain, method = "gbm", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, mammographicTrain)
confusionMatrix(traingbm_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, mammographicTest)
confusionMatrix(testgbm_predict, mammographicTest$Class, positive = '0')
#newthyroid
newthyroiddata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/newthyroid_stddata.csv"), sep = ";",
header = T)
```

```
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(newthyroidTrain[,1:5]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = newthyroidTrain, method = "rf", trControl = fitControl,verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, newthyroidTrain)
confusionMatrix(trainRF_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, newthyroidTest)
confusionMatrix(testRF_predict, newthyroidTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(newthyroidTrain[,1:5]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = newthyroidTrain, method = "wsrf", trControl = fitControl,verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, newthyroidTrain)
confusionMatrix(trainwsrf_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, newthyroidTest)
confusionMatrix(testwsrf_predict, newthyroidTest$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(newthyroidTrain[,1:5])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = newthyroidTrain, method = "LogitBoost", trControl = fitControl,verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, newthyroidTrain)
confusionMatrix(trainlogitboost_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, newthyroidTest)
confusionMatrix(testlogitboost_predict, newthyroidTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:5)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = newthyroidTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, newthyroidTrain)
confusionMatrix(traingbm_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, newthyroidTest)
confusionMatrix(testgbm_predict, newthyroidTest$Class, positive = '1')
#pima
pimadata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/pima_stddata.csv"), sep = ";",  header =T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(pimaTrain[,1:8]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = pimaTrain, method = "rf", trControl = fitControl,verbose = FALSE, tuneGrid =
tunegrid)
```

```
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, pimaTrain)
confusionMatrix(trainRF_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, pimaTest)
confusionMatrix(testRF_predict, pimaTest$Class, positive = 'tested_positive')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(pimaTrain[,1:8]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = pimaTrain, method = "wsrf", trControl = fitControl,verbose = FALSE, tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, pimaTrain)
confusionMatrix(trainwsrf_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, pimaTest)
confusionMatrix(testwsrf_predict, pimaTest$Class, positive = 'tested_positive')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(pimaTrain[,1:8])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = pimaTrain, method = "LogitBoost", trControl = fitControl,verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, pimaTrain)
confusionMatrix(trainlogitboost_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, pimaTest)
confusionMatrix(testlogitboost_predict, pimaTest$Class, positive = 'tested_positive')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:8)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = pimaTrain, method = "gbm", trControl = fitControl,verbose = FALSE, tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, pimaTrain)
confusionMatrix(traingbm_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, pimaTest)
confusionMatrix(testgbm_predict, pimaTest$Class, positive = 'tested_positive')
#thyroid
thyroiddata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/thyroid_stddata.csv"),  sep = ";",
header = T)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(thyroidTrain[,1:21]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = thyroidTrain, method = "rf", trControl = fitControl, verbose = FALSE, tuneGrid =
tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, thyroidTrain)
confusionMatrix(trainRF_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, thyroidTest)
confusionMatrix(testRF_predict, thyroidTest$Class, positive = '1')
```

```
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(thyroidTrain[,1:21]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = thyroidTrain, method = "wsrf", trControl = fitControl,verbose =
FALSE,tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, thyroidTrain)
confusionMatrix(trainwsrf_predict, thyroidTrain$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(thyroidTrain[,1:21])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = thyroidTrain, method = "LogitBoost", trControl = fitControl,verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, thyroidTrain)
confusionMatrix(trainlogitboost_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, thyroidTest)
confusionMatrix(testlogitboost_predict, thyroidTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:21)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = thyroidTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, thyroidTrain)
confusionMatrix(traingbm_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, thyroidTest)
confusionMatrix(testgbm_predict, thyroidTest$Class, positive = '1')
```

Ek 2: Veri Ön işleme: Kayıp Gözlem Atama

```
library(mice)
library(VIM)
#cleveland
#missing data
p <- function(x) {sum(is.na(x))/ length(x)*100}
apply(clevelanddata, 2, p)
#the pattern of the missing data in mice package by
md.pattern(clevelanddata[,-14], rotate.names = TRUE)
md.pairs(clevelanddata)
#Missingness pattern can also be visualised in VIM package by
cleveland_aggr = aggr(clevelanddata[,-14], numbers=TRUE, sortVars=TRUE,
                  labels=names(clevelanddata),cex.axis=.7, gap=3,
                  ylab=c("Proportion of missingness","Missingness Pattern"))
#impute
impute <- mice(clevelanddata[,1:14], m = 5, maxit = 10, printFlag = FALSE, seed = 123)
#complete data
nomissing_cleveland <- complete(impute, 5)
summary(nomissing_cleveland)
str(nomissing_cleveland)
#export data nomissing
write.table(nomissing_cleveland, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nomissing_cleveland.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
nomissing_clevelanddata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez
veriseti/nomissing_cleveland.csv"), sep = ",",  header = T)
#grafik
stripplot(impute, pch = 20, cex = 1.2)
#hepatitis
#missing data
p <- function(x) {sum(is.na(x))/ length(x)*100}
apply(hepatitisdata, 2, p)
hepatitis_aggr = aggr(hepatitisdata[,-20], numbers=TRUE, sortVars=TRUE,
                labels=names(hepatitisdata),cex.axis=.7, gap=3,
                cex.numbers= 0.8, prop = TRUE,
                ylab=c("Proportion of missingness","Missingness Pattern"))
md.pattern(hepatitisdata[,-20], rotate.names = TRUE)
md.pairs(hepatitisdata)
#impute
impute <- mice(hepatitisdata[,1:20], m = 5, maxit = 10, printFlag=FALSE, seed = 123)
#complete data
nomissing_hepatitis <- complete(impute, 5)
summary(nomissing_hepatitis)
str(nomissing_hepatitis)
#export data nomissing
write.table(nomissing_hepatitis, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nomissing_hepatitis.csv",
          col.names = TRUE, row.names=FALSE, sep = ",")
nomissing_hepatitisdata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/nomissing_hepatitis.csv"),
                           sep = ",",  header = T)

summary(nomissing_clevelanddata)
str(nomissing_clevelanddata)
stripplot(impute, pch = 20, cex = 1.2)
plot(impute)
#mammographic
#missing data
p <- function(x) {sum(is.na(x))/ length(x)*100}
apply(mammographicdata, 2, p)
mammographic_aggr = aggr(mammographicdata[,-6], numbers=TRUE, sortVars=TRUE,
```

```
            labels=names(mammographicdata),cex.axis=.7, gap=3,
            ylab=c("Proportion of missingness","Missingness Pattern"))
md.pattern(mammographicdata[,-6], rotate.names = TRUE)
md.pairs(mammographicdata)
#impute
impute <- mice(mammographicdata[,1:6], method = c("polr", "pmm", "polyreg", "polyreg", "polr", " "),
         m = 5, maxit = 10, printFlag=FALSE, seed = 123)
print(impute)
#complete data
nomissing_mammographic <- complete(impute, 5)
summary(nomissing_mammographic)
str(nomissing_mammographic)
#export data nomissing
write.table(nomissing_mammographic, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/nomissing_mammographic.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
nomissing_mammographicdata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez
veriseti/nomissing_mammographic.csv"),
                        sep = ",",  header = T)
summary(nomissing_mammographicdata)
str(nomissing_mammographicdata)
stripplot(impute, pch = 20, cex = 1.2)
plot(impute)
```

Ek 3: Veri Ön işleme: Sınıf Gürültüsü Filtreleme

```
library(NoiseFiltersR)
#cleveland
set.seed(1234)
outcleveland_repair <- hybridRepairFilter(Class~., clevelanddata,
                          noiseAction = "repair", consensus = TRUE)
str(outcleveland_repair)
summary(outcleveland_repair)
#gürültüsüz veri seti
nonoise_cleveland <- outcleveland_repair$cleanData
str(nonoise_cleveland)
summary(nonoise_cleveland)
#export data nomissing
write.table(nonoise_cleveland, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_cleveland.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
#heart
set.seed(1234)
outheart_repair <- hybridRepairFilter(heartdata, noiseAction = "repair",
                           classColumn = ncol(heartdata), consensus = TRUE)
str(outheart_repair)
summary(outheart_repair)
#gürültüsüz veri seti
nonoise_heart <- outheart_repair$cleanData
str(nonoise_heart)
summary(nonoise_heart)
#export data nomissing
write.table(nonoise_heart, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_heart.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
#hepatitis
set.seed(1234)
outhepatitis_repair <- hybridRepairFilter(hepatitisdata, noiseAction = "repair",
                           classColumn = ncol(hepatitisdata), consensus = TRUE)

str(outhepatitis_repair)
summary (outhepatitis_repair)
#gürültüsüz veri seti
nonoise_hepatitis <- outhepatitis_repair$cleanData
str(nonoise_hepatitis)
summary(nonoise_hepatitis)
#export data nomissing
write.table(nonoise_hepatitis, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_hepatitis.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
#mammographic
set.seed(1234)
outmam_repair <- hybridRepairFilter(mammographicdata, noiseAction = "repair",
                           classColumn = ncol(mammographicdata), consensus = TRUE)
str(outmam_repair)
summary(outmam_repair)
#gürültüsüz veri seti
nonoise_mammographic <- outmam_repair$cleanData
str(nonoise_mammographic)
summary(nonoise_mammographic)
#export data nomissing
write.table(nonoise_mammographic, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/nonoise_mammographic.csv",
        col.names = TRUE, row.names=FALSE, sep = ",")
#newthyroid
```

```
set.seed(1234)
outnewthyroid_repair <- hybridRepairFilter(newthyroiddata, noiseAction = "repair",
                                  classColumn = ncol(newthyroiddata), consensus = TRUE)
str(outnewthyroid_repair)
summary(outnewthyroid_repair)
#gürültüsüz veri seti
nonoise_newthyroid <- outnewthyroid_repair$cleanData
str(nonoise_newthyroid)
summary(nonoise_newthyroid)
#export data nomissing
write.table(nonoise_newthyroid, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_newthyroid.csv",
          col.names = TRUE, row.names = FALSE, sep = ",")
#pima
set.seed(1234)
outpima_repair <- hybridRepairFilter(pimadata, noiseAction = "repair",
                               classColumn = ncol(pimadata), consensus = TRUE)
str(outpima_repair)
summary(outpima_repair)
#gürültüsüz veri seti
nonoise_pima <- outpima_repair$cleanData
str(nonoise_pima)
summary(nonoise_pima)
#export data nomissing
write.table(nonoise_pima, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_pima.csv",
          col.names = TRUE, row.names = FALSE, sep = ",")
#thyroid
set.segment(1234)
outthyroid_repair <- hybridRepairFilter(thyroiddata, noiseAction = "repair",
                                 classColumn = ncol(thyroiddata), consensus = TRUE)
str(outthyroid_repair)
summary(outthyroid_repair)
#gürültüsüz veri seti
nonoise_thyroid <- outthyroid_repair$cleanData
str(nonoise_thyroid)
summary(nonoise_thyroid)
#export data nomissing
write.table(nonoise_thyroid, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/nonoise_thyroid.csv",
          col.names = TRUE, row.names = FALSE, sep = ",")
```

Ek 4: Veri Ön işleme: Sınıfların Dengelenmesi

```
library(caret)
library(DMwR)
#cleveland
#dengeli veri seti
set.seed(1234)
noimbalanced_cleveland <- SMOTE(Class ~ ., data = clevelanddata,  k = sqrt(14), perc.over = 5555, perc.under
= 215)
#export data noimbalanced
write.table(noimbalanced_cleveland, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/noimbalanced_cleveland.csv", col.names = TRUE, row.names=FALSE, sep = ",")
#heart
set.seed(1234)
noimbalanced_heart <- SMOTE(Class ~ ., data = heartdata, k = sqrt(14), perc.over = 200, perc.under = 150)
#export data noimbalanced
write.table(noimbalanced_heart, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/noimbalanced_heart.csv",
col.names = TRUE, row.names=FALSE, sep = ",")
#hepatitis
set.seed(1234)
noimbalanced_hepatitis <- SMOTE(Class ~ ., data = hepatitisdata, k = sqrt(20), perc.over = 450, perc.under =
125)
#export data noimbalanced
write.table(noimbalanced_hepatitis, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/noimbalanced_hepatitis.csv", col.names = TRUE, row.names=FALSE, sep = ",")
#lymphography
set.seed(1234)
noimbalanced_lymphography <- SMOTE(Class ~ ., data = lymphographydata, k = sqrt(19), perc.over = 1000,
perc.under = 125)
#export data noimbalanced
write.table(noimbalanced_lymphography, file = "C:/Users/ Casper/Dropbox/tez/tez
veriseti/noimbalanced_lymphography.csv",col.names = TRUE, row.names=FALSE, sep = ",")
#mammographic
set.seed(1234)
noimbalanced_mammographic <- SMOTE(Class ~ ., data = mammographicdata, k = sqrt(6), perc.over = 200,
perc.under = 150)
#export data noimbalanced
write.table(noimbalanced_mammographic, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/noimbalanced_mammographic.csv", col.names = TRUE, row.names=FALSE, sep = ",")
#newthyroid
set.seed(1234)
noimbalanced_newthyroid <- SMOTE(Class ~ ., data = newthyroiddata, k = sqrt(6), perc.over = 1000,
perc.under = 150)
#export data noimbalanced
write.table(noimbalanced_newthyroid, file = "C:/Users/Casper/Dropbox/tez/tez
veriseti/noimbalanced_newthyroid.csv", col.names = TRUE, row.names=FALSE, sep = ",")
#pima
set.seed(1234)
noimbalanced_pima <- SMOTE(Class ~ ., data = pimadata, k = sqrt(9), perc.over = 200, perc.under = 150)
#export data noimbalanced
write.table(noimbalanced_pima, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/noimbalanced_pima.csv",
col.names = TRUE, row.names=FALSE, sep = ",")
#thyroid
set.seed(1234)
noimbalanced_thyroid <- SMOTE(Class ~ ., data = thyroiddata, k = sqrt(21), perc.over = 300, perc.under = 255)
#export data noimbalanced
library(xlsx)
write.xlsx(noimbalanced_thyroid, file = "C:/Users/Casper/Dropbox/tez/tez veriseti/noimbalanced_thyroid.xlsx",
col.names = TRUE, row.names = FALSE)
```

Ek 5: İşlenmiş Verilerin Analizi R Kodları

```
#cleveland
clevelanddata<- read.csv(file = ("C:/Users/Casper/Dropbox/tez/tez veriseti/noimbalanced_cleveland.csv"), sep =
",",  header = T)
#data partition
set.seed(1234)
trainIndex <- createDataPartition(clevelanddata$Class, p = .7, list = FALSE, times = 1)
head(trainIndex)
clevelandTrain <- clevelanddata[ trainIndex,]
clevelandTest  <- clevelanddata[-trainIndex,]
#Basic Parameter Tuning (10-fold CV / repeated ten times)
set.seed(1234)
fitControl <- trainControl(method = "repeatedcv", number = 10, repeats = 10)
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(clevelandTrain[,1:13]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = clevelandTrain, method = "rf",trControl = fitControl,verbose = FALSE,tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, clevelandTrain)
confusionMatrix(trainRF_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, clevelandTest)
confusionMatrix(testRF_predict, clevelandTest$Class, positive = '0')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(clevelandTrain[,1:13]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = clevelandTrain, method = "wsrf", trControl = fitControl,verbose =
FALSE,tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, clevelandTrain)
confusionMatrix(trainwsrf_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, clevelandTest)
confusionMatrix(testwsrf_predict, clevelandTest$Class, positive = '0')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(clevelandTrain[,1:13])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = clevelandTrain, method = "LogitBoost", trControl = fitControl,verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, clevelandTrain)
confusionMatrix(trainlogitboost_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, clevelandTest)
confusionMatrix(testlogitboost_predict, clevelandTest$Class, positive = '0')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:13)*50,shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = clevelandTrain, method = "gbm", trControl = fitControl,verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
```

```
traingbm_predict <- predict(fitgbm, clevelandTrain)
confusionMatrix(traingbm_predict, clevelandTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, clevelandTest)
confusionMatrix(testgbm_predict, clevelandTest$Class, positive = '0')
#heart
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(heartTrain[,1:13]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = heartTrain, method = "rf", trControl = fitControl,verbose = FALSE, tuneGrid =
tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, heartTrain)
confusionMatrix(trainRF_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, heartTest)
confusionMatrix(testRF_predict, heartTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(heartTrain[,1:13]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = heartTrain, method = "wsrf", trControl = fitControl,verbose = FALSE,tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, heartTrain)
confusionMatrix(trainwsrf_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, heartTest)
confusionMatrix(testwsrf_predict, heartTest$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(heartTrain[,1:13])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = heartTrain, method = "LogitBoost", trControl = fitControl,verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, heartTrain)
confusionMatrix(trainlogitboost_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, heartTest)
confusionMatrix(testlogitboost_predict, heartTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:13)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = heartTrain, method = "gbm", trControl = fitControl,verbose = FALSE,tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, heartTrain)
confusionMatrix(traingbm_predict, heartTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, heartTest)
confusionMatrix(testgbm_predict, heartTest$Class, positive = '1')
#hepatitis
library(randomForest)
#random forest expand.grid / parameters tuning
```

```
mtry <- sqrt(ncol(hepatitisTrain[,1:19]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = hepatitisTrain, method = "rf", trControl = fitControl, verbose = FALSE,tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, hepatitisTrain)
confusionMatrix(trainRF_predict, hepatitisTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, hepatitisTest)
confusionMatrix(testRF_predict, hepatitisTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(hepatitisTrain[,1:19]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = hepatitisTrain, method = "wsrf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, hepatitisTrain)
confusionMatrix(trainwsrf_predict, hepatitisTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, hepatitisTest)
confusionMatrix(testwsrf_predict, hepatitisTest$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(hepatitisTrain[,1:19])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = hepatitisTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, hepatitisTrain)
confusionMatrix(trainlogitboost_predict, hepatitisTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, hepatitisTest)
confusionMatrix(testlogitboost_predict, hepatitisTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:19)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = hepatitisTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, hepatitisTrain)
confusionMatrix(traingbm_predict, hepatitisTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, hepatitisTest)
confusionMatrix(testgbm_predict, hepatitisTest$Class, positive = '1')
#lymphography
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(lymphographyTrain[,1:18]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = lymphographyTrain, method = "rf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, lymphographyTrain)
confusionMatrix(trainRF_predict, lymphographyTrain$Class, positive = 'normal')
```

```
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, lymphographyTest)
confusionMatrix(testRF_predict, lymphographyTest$Class, positive = 'normal')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(lymphographyTrain[,1:18]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = lymphographyTrain, method = "wsrf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, lymphographyTrain)
confusionMatrix(trainwsrf_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, lymphographyTest)
confusionMatrix(testwsrf_predict, lymphographyTest$Class, positive = 'normal')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(lymphographyTrain[,1:18])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = lymphographyTrain, method = "LogitBoost", trControl =
fitControl,verbose = FALSE,  tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, lymphographyTrain)
confusionMatrix(trainlogitboost_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, lymphographyTest)
confusionMatrix(testlogitboost_predict, lymphographyTest$Class, positive = 'normal')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:18)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = lymphographyTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, lymphographyTrain)
confusionMatrix(traingbm_predict, lymphographyTrain$Class, positive = 'normal')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, lymphographyTest)
confusionMatrix(testgbm_predict, lymphographyTest$Class, positive = 'normal')
#mammographic
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(mammographicTrain[,1:5]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = mammographicTrain, method = "rf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, mammographicTrain)
confusionMatrix(trainRF_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, mammographicTest)
confusionMatrix(testRF_predict, mammographicTest$Class, positive = '0')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(mammographicTrain[,1:5]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
```

```
fitwsrf <- train(Class ~ ., data = mammographicTrain, method = "wsrf", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, mammographicTrain)
confusionMatrix(trainwsrf_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, mammographicTest)
confusionMatrix(testwsrf_predict, mammographicTest$Class, positive = '0')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(mammographicTrain[,1:5])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = mammographicTrain, method = "LogitBoost", trControl = fitControl,
verbose = FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, mammographicTrain)
confusionMatrix(trainlogitboost_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, mammographicTest)
confusionMatrix(testlogitboost_predict, mammographicTest$Class, positive = '0')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:5)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = mammographicTrain, method = "gbm", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, mammographicTrain)
confusionMatrix(traingbm_predict, mammographicTrain$Class, positive = '0')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, mammographicTest)
confusionMatrix(testgbm_predict, mammographicTest$Class, positive = '0')
#newthyroid
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(newthyroidTrain[,1:5]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = newthyroidTrain, method = "rf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, newthyroidTrain)
confusionMatrix(trainRF_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, newthyroidTest)
confusionMatrix(testRF_predict, newthyroidTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(newthyroidTrain[,1:5]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = newthyroidTrain, method = "wsrf", trControl = fitControl,verbose =
FALSE,tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, newthyroidTrain)
confusionMatrix(trainwsrf_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, newthyroidTest)
confusionMatrix(testwsrf_predict, newthyroidTest$Class, positive = '1')
```

```
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(newthyroidTrain[,1:5])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = newthyroidTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, newthyroidTrain)
confusionMatrix(trainlogitboost_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, newthyroidTest)
confusionMatrix(testlogitboost_predict, newthyroidTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:5)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = newthyroidTrain, method = "gbm", trControl = fitControl,verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, newthyroidTrain)
confusionMatrix(traingbm_predict, newthyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, newthyroidTest)
confusionMatrix(testgbm_predict, newthyroidTest$Class, positive = '1')
#pima
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(pimaTrain[,1:8]))
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitRF <- train(Class ~ ., data = pimaTrain, method = "rf", trControl = fitControl, verbose = FALSE, tuneGrid =
tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, pimaTrain)
confusionMatrix(trainRF_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, pimaTest)
confusionMatrix(testRF_predict, pimaTest$Class, positive = 'tested_positive')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(pimaTrain[,1:8]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = pimaTrain, method = "wsrf", trControl = fitControl, verbose = FALSE, tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, pimaTrain)
confusionMatrix(trainwsrf_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, pimaTest)
confusionMatrix(testwsrf_predict, pimaTest$Class, positive = 'tested_positive')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(pimaTrain[,1:8])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = pimaTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
```

```
trainlogitboost_predict <- predict(fitlogitboost, pimaTrain)
confusionMatrix(trainlogitboost_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, pimaTest)
confusionMatrix(testlogitboost_predict, pimaTest$Class, positive = 'tested_positive')
library(gbm)
#gbm expand.grid / parameters tuning
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:8)*50, shrinkage = 0.1, n.minobsinnode = 10)
set.seed(825)
fitgbm <- train(Class ~ ., data = pimaTrain, method = "gbm", trControl = fitControl, verbose = FALSE, tuneGrid
= tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, pimaTrain)
confusionMatrix(traingbm_predict, pimaTrain$Class, positive = 'tested_positive')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, pimaTest)
confusionMatrix(testgbm_predict, pimaTest$Class, positive = 'tested_positive')
#thyroid
library(randomForest)
#random forest expand.grid / parameters tuning
mtry <- sqrt(ncol(thyroidTrain[,1:18]))
tunegrid <- expand.grid(.mtry = mtry)

set.seed(825)
fitRF <- train(Class ~ ., data = thyroidTrain, method = "rf", trControl = fitControl, verbose = FALSE, tuneGrid =
tunegrid)
# Prediction & Confusion Matrix - train data
trainRF_predict <- predict(fitRF, thyroidTrain)
confusionMatrix(trainRF_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testRF_predict<- predict(fitRF, thyroidTest)
confusionMatrix(testRF_predict, thyroidTest$Class, positive = '1')
library(wsrf)
#wsrf expand.grid / parameters tuning
mtry <- log2(ncol(thyroidTrain[,1:18]))+1
tunegrid <- expand.grid(.mtry = mtry)
set.seed(825)
fitwsrf <- train(Class ~ ., data = thyroidTrain, method = "wsrf", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainwsrf_predict <- predict(fitwsrf, thyroidTrain)
confusionMatrix(trainwsrf_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testwsrf_predict<- predict(fitwsrf, thyroidTest)
confusionMatrix(testwsrf_predict, thyroidTest$Class, positive = '1')
library(caTools)
#logitboost expand.grid / parameters tuning
niter <- ncol(thyroidTrain[,1:18])*100
tunegrid <- expand.grid(.nIter = niter)
set.seed(825)
fitlogitboost <- train(Class ~ ., data = thyroidTrain, method = "LogitBoost", trControl = fitControl, verbose =
FALSE, tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
trainlogitboost_predict <- predict(fitlogitboost, thyroidTrain)
confusionMatrix(trainlogitboost_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testlogitboost_predict<- predict(fitlogitboost, thyroidTest)
confusionMatrix(testlogitboost_predict, thyroidTest$Class, positive = '1')
library(gbm)
#gbm expand.grid / parameters tuning
```

```
tunegrid <- expand.grid(interaction.depth = c(1, 5, 9), n.trees = (1:18)*50, shrinkage = 0.1, n.minobsinnode =10)
set.seed(825)
fitgbm <- train(Class ~ ., data = thyroidTrain, method = "gbm", trControl = fitControl, verbose = FALSE,
tuneGrid = tunegrid)
# Prediction & Confusion Matrix - train data
traingbm_predict <- predict(fitgbm, thyroidTrain)
confusionMatrix(traingbm_predict, thyroidTrain$Class, positive = '1')
#Prediction & Confusion Matrix - test data
testgbm_predict<- predict(fitgbm, thyroidTest)
confusionMatrix(testgbm_predict, thyroidTest$Class, positive = '1')
```

Teşekkür

1999 yılında başladığım eğitim ve öğretim hayatımın 20 yılı geri kalmaktadır. Her yılını en etkin ve verimli geçirmeyi kendime bir görev bildim. Bana hayat boyu koşulsuz olarak maddi ve manevi desteklerini esirgemeyen, sevgiyi, saygıyı ve güzel ahlaklı insan olmayı öğreten, hayatın her şartlarında ne olursa olsun her zaman sabırlı olmayı, Atatürk'ün açtığı yolda, Atatürk ilke ve inkılaplarına bağlı bir vatandaş olmayı öğreten en değerli varlıklarım annem Fatma ÖZKAN ve babam Ahmet ÖZKAN'a teşekkür ederim. Bir elin beş parmağı olduğumuz canımın içi kardeşlerim Mustafa ÖZKAN, Tülin ÖZKAN, Soner ÖZKAN ve en küçük yavrumuz Arya Lisa ÖZKAN'a teşekkür ederim. Tüm arkadaşlarım ve en yakın arkadaşım, dostum, yol arkadaşım Halil İbrahim ADALARLI'ya teşekkür ederim.

Lisans eğitim ve öğretim hayatım boyunca tecrübelerini, güler yüzlerini, samimiyetlerini benden esirgemeyen, istatistik bilimini öğreten ve gelecekteki mesleki hayatımda da bana verdikleri değerli bilgilerden faydalandığım Dokuz Eylül Üniversitesi İstatistik Bölümü'ndeki tüm hocalarıma teşekkür ederim.

Yüksek lisans eğitim ve öğretimim boyunca bana karşı gösterdikleri samimiyet, hoşgörü, nazik davranışları ve sundukları tüm imkânlar için danışmanım Dr. Öğr. Üyesi Aslı SUNER KARAKÜLAH'a, desteklerini esirgemeyen Prof. Dr. Mehmet N. ORMAN'a ve Doç. Dr. Timur KÖSE'ye; Ege Üniversitesi, Tıp Fakültesi, Biyoistatistik ve Tıbbi Bilişim Anabilim Dalı ailesinden Uzm. Dr. Gül KİTAPÇIOĞLU'na, Arş. Gör. Semiha ÖZGÜL'e, Arş. Gör. Ömer Faruk DADAŞ'a, Arş. Gör. Gülden HAKVERDİ'ye ve Fatoş DOĞAN'a sonsuz teşekkürler ederim. Ayrıca, desteklerinden dolayı Muğla Sıtkı Koçman Üniversitesi, Fen Fakültesi İstatistik Bölümü'nden Doç. Dr. Eralp Doğu ve yüksek lisans tez öğrencisi Sultan Turhan'a teşekkür ederim.

Son olarak, bilgisini ve deneyimleri bizlerle paylaşan bölümümüzün kurucusu değerli hocamız emekli Prof. Dr. Fikret İKİZ'e teşekkürü bir borç bilirim.

İzmir, 16.09.2019 Yüksel ÖZKAN

Özgeçmiş

KİŞİSEL BİLGİLER

Adı Soyadı : Yüksel ÖZKAN

Adresi : Ege Üniversitesi, Tıp Fakültesi, Biyoistatistik ve Tıbbi Bilişim Anabilim Dalı, Bornova, İzmir

Tel : +90 531 271 4213

Mail : yukselozkan03@gmail.com

ÖĞRENİM BİLGİLERİ

- **Yüksek Lisans:** Ege Üniversitesi – Sağlık Bilimleri Enstitüsü - Biyoistatistik Tezli Yüksek Lisans Programı [Eylül 2017 – Devam ediyor]

  Tez Danışmanı: Doktor Öğretim Üyesi Aslı SUNER KARAKÜLAH

- **Lisans:** Dokuz Eylül Üniversitesi - Fen Fakültesi - İstatistik Bölümü [Eylül 2013 - Temmuz 2017]

  Danışman: Prof. Dr. Aylin Alın

- **Lisans**: Dokuz Eylül Üniversitesi - Yabancı Diller Yüksekokulu - İngilizce Hazırlık [Eylül 2012 -Temmuz 2013]

GÖREVLER

- **Kısmi Zamanlı Çalışma:** Ege Üniversitesi, Madde Bağımlılığı, Toksikoloji ve İlaç Bilimleri Enstitüsü [Ekim 2018 – Mayıs 2019]
- **Lisans Stajı:** Marmara Üniversitesi, Tıp Fakültesi, Biyoistatistik Anabilim Dalı - Yaz stajı [Temmuz 2016 - Ağustos 2016]

PROJELER

- **Kariyer Başlangıç Destek Projesi:** Yönetici: Dr. Öğr. Üyesi Aslı Suner Karakülah, *Sağlık Verilerinde Veri Ön İşleme Sonrası Sınıflandırma Algoritmalarının Performans Değerlendirmesi*, Ege Üniversitesi, Bilimsel Araştırmalar Birimi (BAP) tarafından desteklenen TKB-2019-20199 nolu proje, Araştırmacı [Mayıs 2019 - Devam ediyor]

- **Lisans Bitirme Projesi:** Danışman: Prof. Dr. Aylin Alın, *Yüksek Çıktılı Gen Verilerinde Diferansiyel Ekspresyon Analizi için İstatistiksel Testler,* İstatistik Bölümü Lisans Bitirme Projeleri, Mayıs 2017, Buca, İzmir, Sözlü Bildiri ve Poster Bildiri.

EĞİTİMLER

- *Julia ile Derin Öğrenmeye Giriş*, Prof. Dr. Deniz Yüret, Haziran 2019, Sarıyer, İstanbul.
- *Julia ile Derin Öğrenme Uygulamaları*, Prof. Dr. Deniz Yüret, Haziran 2019, Sarıyer, İstanbul.
- *Count Data Modelling and Analysis with Applications in Medicine*, Professor Dankmar Böhning, Assoc. Prof. Dr. Stefanie Biederman, Prof. Dr. Mehmet Orman, Mr. James Gallagher, May 2018, Çeşme, İzmir.
- *Rasch Analizine Giriş*, Prof. Dr. Reha Alpar, Ekim 2016, Belek, Antalya.

Ulusal ve Uluslar Arası Bilimsel Toplantılarda Sunulan Bildiriler

- **Yüksel Özkan**, Timur Köse, Mehmet Nurullah Orman, Aslı Suner, *Makine Öğrenmesi Yöntemleriyle Çalışanların Sağlık Durumlarına Göre İşe Devamsızlıklarının Tahminlenmesinde Karar Desteği*, 11. Tıp Bilişimi Kongresi, Kasım 2018, Altındağ, Ankara, Poster Bildiri.
- Sultan Turhan, **Yüksel Özkan**, Banu Şarer Yürekli, Aslı Suner, Eralp Doğu, *Comparison of Supervised Disease Diagnosis Methods for Unbalanced Classes: The Case of Diabetes Diagnosis*, 11. International Statistics Days Conference, Ekim 2018, Turgutreis, Muğla, Sözlü Bildiri.
- **Yüksel Özkan**, Banu Sarer Yürekli, Aslı Suner, *Performance Evaluation of Supervised Machine Learning Algorithms for Predicting Diabetes Mellitus*, 4th International Researchers, Statisticians and Young Statisticians Congress, Nisan 2018, Çeşme, İzmir, Sözlü Bildiri.
- **Yüksel Özkan**, Cevriye Cezayir, Hüseyin Sağıroğlu, Sezer Baysal, *Yüksek Çıktılı Gen Verilerinde Diferansiyel Ekspresyon Analizi için İstatistiksel Testler*, 14. Uluslararası İstatistik Öğrenci Kolokyumu, Nisan 2017, Tepebaşı, Eskişehir, Sözlü Bildiri.
- Gülnaz Nural Bekiroğlu, **Yüksel Özkan**, Esra Akdeniz, *CART modellemede Eğitim ve Test Veri Seti için En Uygun Ayırma Oranın Belirlenmesi*, 18. Ulusal Biyoistatistik

Kongresi ve 1. Uluslararası Biyoistatistik Kongresi, Ekim 2016, Belek, Antalya, Sözlü Bildiri.

- Esra Akdeniz, Gülnaz Nural Bekiroğlu, **Yüksel Özkan,** *Cox Regresyon Modelinde Değişken Seçimi Yöntemleri*, 18. Ulusal Biyoistatistik Kongresi ve 1. Uluslararası Biyoistatistik Kongresi, Ekim 2016, Belek, Antalya, Sözlü Bildiri.
- **Yüksel Özkan**, Esra Akdeniz, Gülnaz Nural Bekiroğlu, *Veri Biliminde Parametrik, Parametrik Olmayan ve Kara Kutu Modellerin Karşılaştırılması ve Bir Uygulama*, 18. Ulusal Biyoistatistik Kongresi ve 1. Uluslararası Biyoistatistik Kongresi, Ekim 2016, Belek, Antalya, Sözlü Bildiri.

BİLİMSEL KURULUŞLARA ÜYELİKLER

- İzmir İstatistiksel Düşünce Topluluğu, Eğitim Koordinatörlüğü [2017 – 2018]
- Dokuz Eylül Üniversitesi, İstatistik Topluluğu, Başkan Yardımcılığı [2016 - 2017]
- Dokuz Eylül Üniversitesi, İstatistik Topluluğu, Kariyer Koordinatörlüğü Üyesi [2015 - 2016]

SERTİFİKA

- Koç Üniversitesi ve Bilim Akademisi, *Kuzeybatıda Yapay Öğrenme Yaz Okulu - Sağlık Bilimlerinde, Robotikte ve Endüstride Yapay Öğrenme*, Haziran 2019, Sarıyer, İstanbul, Katılımcı Öğrenci.
- İstanbul İşletme Enstitüsü, *Dijital Pazarlama Uzmanlığı*, Şubat 2019, Online Eğitim.
- İstanbul İşletme Enstitüsü, *İş Hukuku Uygulamaları*, Şubat 2019, Online Eğitim.
- Gençlik ve Spor Bakanlığı Akademisi, *Kariyer Semineri*, Mart 2013, Buca, İzmir.
- Boğaziçi Üniversitesi ve Bilim Akademisi, *Boğaz'da Yapay Öğrenme İsmail Arı Yaz Okulu - Konuşma, Dil İşleme ve Biyoenformatik*, Temmuz 2018, Bebek, İstanbul, Katılımcı Öğrenci.
- Ege Üniversitesi Tıp Fakültesi Biyoistatistik ve Tıbbi Bilişim Anabilim Dalı Yaz Okulu, *Count Data Modelling and Analysis with Applications*, Mayıs 2018, Çeşme, İzmir, Katılımcı Öğrenci.
- Türk İstatistik Derneği, Anadolu Üniversitesi, *14. Uluslararası İstatistik Öğrenci Kolokyumu*, Nisan 2017, Eskişehir, Katılımcı Öğrenci.

- Yaşamartı Personal Development Academy, *Art of Effective Elocution - Presentation Techniques, Elocution, Phonetics, Speaking, Breathing Techniques*, May 2014, Konak, İzmir.
- Yaşamartı Personal Development Academy, *Personal Brand Image in Business Life, Relationship Management and Leadership Competencies*, May 2014, Konak, İzmir.
- Yaşamartı Personal Development Academy, *You Have the Control on the Way to Success - Motivation and Stress Management, Decision - Making Under Difficult Conditions and Problem Solving*, May 2014, Konak, İzmir.
- Kalkınma Bakanlığı, Birleşmiş Milletler Kalkınma Programı, Habitat Kalkınma ve Yönetişim Derneği ve Visa Europe, *Paramı Yönetebiliyorum Projesi - Finansal Bilinç Eğitimi*, Nisan 2014, Konak, İzmir.