
GAZİ ÜNİVERSİTESİ
Gazi
1926

BÜYÜK VERİ PROBLEMLERİNDE PERFORMANS ARTTIRMAYA YÖNELİK ÖZELLİK SEÇİMİ VE BOYUT İNDİRGEME OPTİMİZASYONU

Burhan Erdoğdu BEYAZIT

YÜKSEK LİSANS TEZİ
YÖNETİM BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

KASIM 2019

Burhan Erdoğdu BEYAZIT tarafından hazırlanan BÜYÜK VERİ PROBLEMLERİNDE PERFORMANS ARTIRMAYA YÖNELİK ÖZELLİK SEÇİMİ VE BOYUT İNDRİGEME OPTİMİZASYONU adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Yönetim Bilişim Sistemleri Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Cevriye Temel GENCER

Endüstri Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

.....

Üye: Doç. Dr. Hacer KARACAN

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

.....

Üye: Dr. Öğr. Üyesi Hakan ÖZKÖSE

Yönetim Bilişim Sistemleri Ana Bilim Dalı, Bartın Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

.....

Tez Savunma Tarihi: ...18./.11.../.2019

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

.....

Doç. Dr. Aslıhan TÜFEKÇİ

Bilişim Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Burhan Erdogdu BEYAZIT

18.11.2019

BÜYÜK VERİ PROBLEMLERİNDE PERFORMANS ARTIRMAYA YÖNELİK ÖZELLİK SEÇİMİ VE BOYUT İNDİRGEME OPTİMİZASYONU

(Yüksek Lisans Tezi)

Burhan Erdoğdu BEYAZIT

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Kasım 2019

ÖZET

Teknolojik gelişmeler ile veri boyutu, çeşitliliği ve akışında meydana gelen değişimler, BigData (Büyük veri) kavramın ve veriden bilgi elde etme sürecinde verinin toplanması, dönüştürülmesi, işlenmesi, saklanması ve sunulması gibi yeni bir paradigmanın ortaya çıkmasına neden olmuştur. Yeni paradigma veri toplama, işleme, saklama, bileşenleriyle, sıradan donanımlar üzerine kurulabilen, hata toleranslı, yatay genişleyebilen Hadoop ekosistemidir. Hadoop üzerinde paralel işlem çatısı olarak Apache Spark veri işleme süreçlerinde makine öğrenmesi kabiliyetlerini veri bilimcilerin kullanımına sunmaktadır. Günümüzde büyük veri kavramı ile bilgi keşfinin anlık olarak yapılabilmesi önemli bir ihtiyaç haline gelmiştir. Bu noktada büyük veri sistemleri üzerinde makine öğrenmesi ile veriden bilgi keşfi süreçlerinin otomatikleştirilmesi fikri ortaya çıkmıştır. Ancak literatürde tartışmalı bir husus olan otomatikleştirme fikirleri, için öncelikle çözüm bulunması gereken konuların başında özellik seçimi ve boyut azaltma işlemlerinin, en az alan bilgisi ve yüksek performans ile gerçekleştirilebilmesi gelmektedir. Bu çalışmada ülkemizde bir internet hizmet sağlayıcıdan elde edilen veriler ve açık kaynaklı telekomünikasyon veri seti ile Apache Spark makine öğrenmesi kütüphanesi kullanılarak özellik seçme ve boyut azaltma uygulaması gerçekleştirilmiştir. Özellik seçimi için Filter (Filtre), Embedded (Gömülü) ve Wrapper (Sarmalayıcı) metotlar, boyut azaltma için Principal Component Analysis (PCA) uygulanmıştır. F1- measure, Precision, Recall ve Accuracy başarım kriterlerine göre yapılan denemelerde Filter metotların bu kapsamda kullanışlı bir seçenek oldukları görülmüştür.

Bilim Kodu : 92414
Anahtar Kelimeler : Büyük Data, Yapay Öğrenme, Veri Madenciliği, Veri Yönetimi
Sayfa Adedi : 63
Danışman : Prof. Dr. Cevriye Temel GENCER

FEATURE SELECTION AND DIMENSIONALITY REDUCTION OPTIMIZATION TO IMPROVE PERFORMANCE IN BIG DATA PROBLEMS

(M. Sc. Thesis)

Burhan Erdoğdu BEYAZIT

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

November 2019

ABSTRACT

Technological developments and changes in volume, variety and velocity of data have led to define both new concept of Big data, and new paradigm in the process of acquiring information from data. The new paradigm is a fault-tolerant, scalable, built for commodity hardware, Hadoop ecosystem with data collection, data processing, data warehousing components. As a parallel processing framework on Hadoop, Apache Spark offers to the data scientists the ability of using machine learning libraries in easy way. In present, with the concept of big data, it has become an important necessity to make discovery of information instantaneously. The idea of automatizing information discovery based on machine learning on big data systems has been introduced. However, for automation ideas, which is a controversial issue in the literature, the first of the issues that need to be resolved is that feature selection and dimensionality reduction operations can be performed with minimum field knowledge and high performance. In this study feature selection and dimensionality reduction application were performed using Apache Spark machine learning library on the data obtained from an internet service provider and the open source telecommunication data set. The Filter, Embedded and Wrapper methods for Feature Selection were applied and Principal Component Analysis is used for dimensionality reduction. According to the tests measured by F1- measure, Precision, Recall Accuracy, filter methods have been seen to be a useful option in this context.

Science Code : 92414
Key Words : Big Data, Machine Learning, Data Mining, Data Management
Page Number : 63
Supervisor : Prof. Dr. Cevriye Temel GENCER

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren, kıymetli tecrübelerinden faydalandığım saygı değer hocam Prof. Dr. Cevriye Temel GENCER' e şükranlarımı sunarım. Akademik çalışmalara ve teknik gelişime verdiği değer ile beni cesaretlendiren ve bu çalışmanın hayata geçmesinde çok büyük katkıları olan Türksat Bilişim Sözleşme Yönetimi ve Mali Takip Direktörü Ahmet ASLANPINAR'a, aileme ve sevgili eşim Pınar BEYAZIT'a teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılan bazı kısaltmalar, açıklamaları ile aşağıda sunulmuştur.

Kısaltmalar	Açıklama
DCT	Ayrık kosinüs dönüşümü (Discrete cosine transform)
FN	Hatalı negatif (False negative)
FP	Hatalı pozitif (False positive)
GPS	Küresel konumlama sistemi (Global positioning system)
HDFS	Hadoop dağıtık dosya sistemi (Hadoop distribted file system)
IQR	Çeyreklikler arası aralık (Interquartile range)
MAE	Ortalama mutlak hata (Mean absolute error)
ML	Makine öğrenmesi (Machine learning)
MSE	Ortalama kare sapması (Mean square error)
PCA	Temel bileşen analizi (Principal component analysis)
PPV	Pozitif tahmin değeri (Positive predictive value)
RMSE	Ortalama kare sapmasının karekökü (Root mean square error)
SBE	Geriye doğru sıralı eleme (Sequential backward elimination)
SDK	Yazılım geliştirme kiti (Software development kit)
SFS	İleriye doğru sıralı seçme (Sequental forward selection)
SQL	Yapısal sorgu dili (Structured query language)
TN	Başarılı negatif (True negative)
TP	Başarılı pozitif (True positive)
TPR	Başarılı pozitif oranı (True positive rate)
XML	Genişletilebilir işaretleme dili (Extensible markup language)

1. GİRİŞ

Bilişim sektöründe sıklıkla rastladığımız veri kavramı esasında bilişim sektörünü de var eden olgudur. Verinin toplanması, saklanması, işlenmesi gibi bilgi işlem faaliyetlerinde ortaya çıkan ihtiyaçlar bilişim sektöründe veri tabanı uzmanlığı, dağıtık sistemler, iş zekâsı gibi birçok uzmanlık alanının da doğmasına yol açmıştır. Günümüzde verinin çeşitliliğinin ve boyutunun artışı karşısında geleneksel yolların kullanılabilirliğinin oldukça azaldığı görülmektedir.

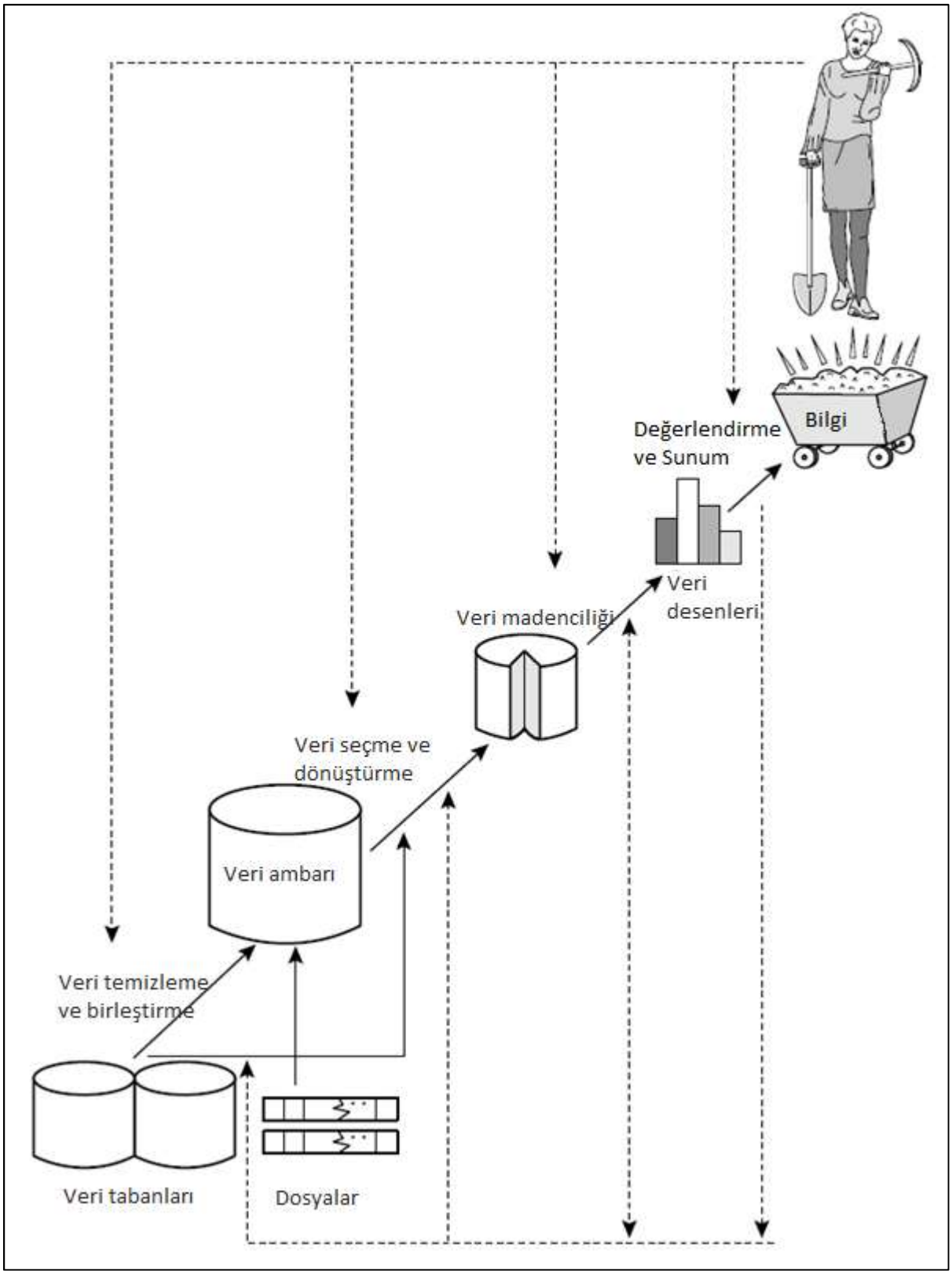
Verinin sınırsız sayıda tanımı olabilir. Şeker tarafından verinin tanımı "herhangi bir işleme tabi tutulmadan, gözlem veya ölçüm yöntemleri ile ortamdan elde edilen her türlü değerdir" şeklinde yapılmıştır (Şeker, 2013). Veri, elde edilen fayda bakımından veri (ham veri, data), enformasyon (information) ve bilgi (knowledge) olmak üzere üçe ayrılabilir (Aydıntan, 2009 : 7,8).

Bilişim sektöründe veri, henüz anlamlandırma işlemine tabi tutulmamış sensör verileri, işlem (Log) kayıtları, kamera kayıtları, ses dosyaları, ağ trafik kayıtları verisi gibi bir kaynaktan üretilen kayıtlar olarak tanımlanabilir. Bu kayıtların tutulması çeşitli regülasyonlarla zorunlu kılınmış olabilir, ancak bu verilerin tutulmasının organizasyon açısından faydaya dönüştürülmesi için işlenmesi gerekmektedir.

Log kayıtlarının işlenmesi ile hangi işlemin kim tarafından ne zaman gerçekleştirildiği bulunabilir. Video ve ses verisinden hangi anda kimin kayıtta göründüğü veya konuştuğu tespit edilebilir. Bu şekilde verinin anlamlandırılmasıyla birçok kaynakta enformasyon olarak tanımlanan bilgi üretilmektedir.

Veriden üretilen enformasyon, geçmişe dönüktür. Kayıt altına alınan veri üzerinde gerçekleştirilen işlem veriye ilişkin bir bilgi ortaya çıkarır. Böylelikle organizasyon, veri yığınını, istatistik gibi yöntemlerle analiz etme ve yorumlama imkânı bulur. Bu yorumlama ile geleceği de içine alan bir zaman dilimine ilişkin, verinin içerisinde olmayan birtakım değerler üretilebilir. Bu değerler bilgi olarak adlandırılmaktadır.

Bilgi, işletmeye değer yaratan bir tarzda organize edilebilen, gruplandırılabilen, modelleştirilebilen ve eyleme geçirilebilen veridir (Doğan ve Yörük, 2009 : 9). Veriden bilgi keşfi süreci Şekil 1.1'de gösterilmektedir.



Şekil 1.1. Veriden bilgi keşfi (Han ve Kamber, 2006:6)

Veriden bilgi keşfinin kuruluşlar açısından faydalarının görülmesine paralel olarak veri kaynakları ve türlerinde önemli değişimler yaşanmıştır (Xing, Jordan, ve Karp, 2001). Günümüzde akıllı cep telefonları ile cebimize giren bilgisayarlar, iş için kullandığımız

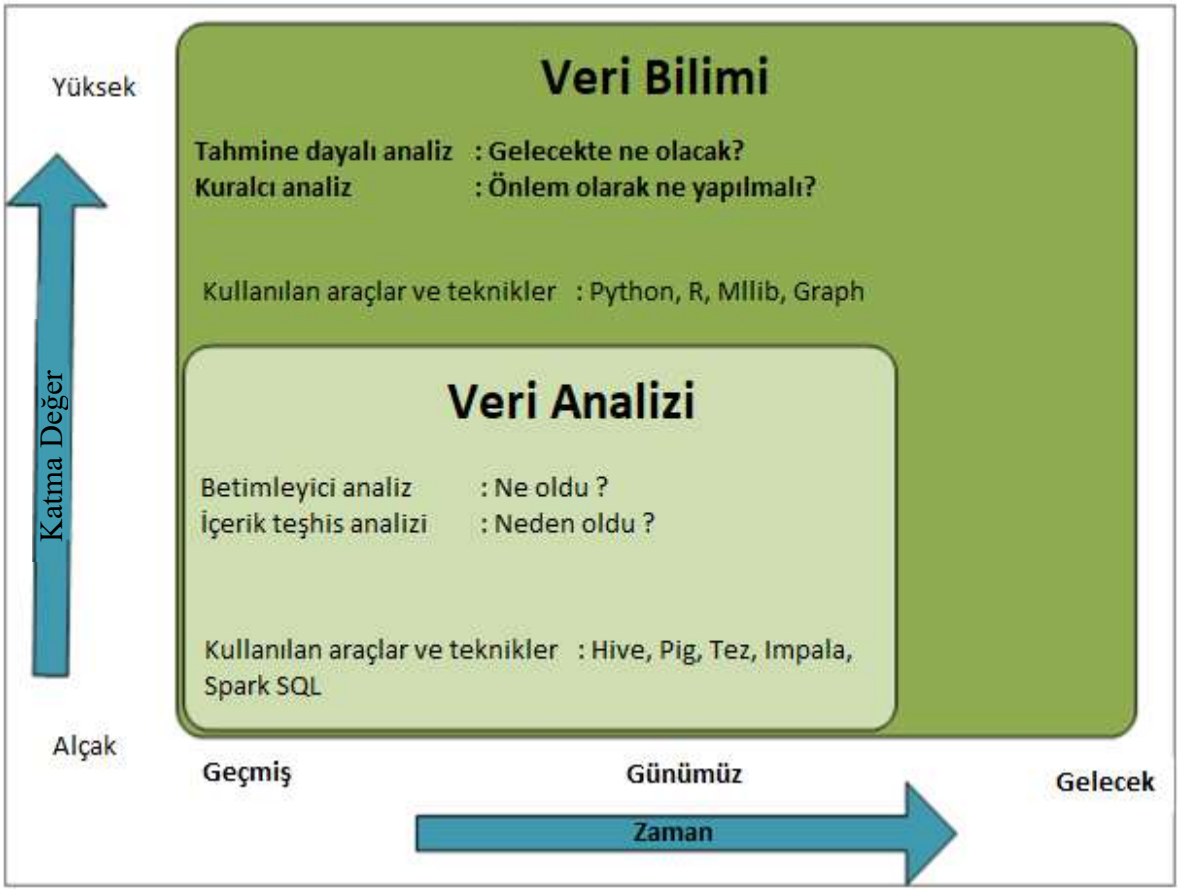
bilgisayarlardan çok daha fazla miktarda ve çeşitte veri üretmektedir. Örneğin gitmek istediğimiz adresi cep telefonumuzdaki navigasyon uygulamasına girerek yola çıktığımızda, cep telefonumuz anlık olarak konum verimizi üretmekte, üretilen veriyi uygulama sunucusuna göndererek işlemekte ve gitmek istediğimiz rotaya ilişkin yönlendirme bilgilerini bize sumaktadır. Bu süreci veri, enformasyon ve bilgi perspektifinde incelersek, koordinat verileri, yol verileri, ilgi çekici nokta verileri birbirleri ile eşleştirilerek adres enformasyonuna dönüştürülmüştür. Bu enformasyon zaman ve trafik yoğunluğu enformasyonu ile yorumlanarak bir noktadan bir noktaya gitmek istediğimiz zaman dilimi içerisinde trafik yoğunluğundan kaçınarak verimli şekilde hareket etmemizi sağlayacak bilgiye dönüştürülmüştür. Bu gibi örnekleri ses, fotoğraf video verileri, kişiler arasında sosyal medya üzerinde kurulan ilişki verileri gibi yapısal ve yapısal olmayan verileri de hesaba kattığımızda verinin boyutunun büyümesiyle birlikte büyük veri kavramının da tarihsel süreçte yeniden tanımlanmakta olduğu görülmektedir.

1960'lı yıllarda basit dosya sistemleri ile veri toplama yaygın iken 1970'lerle birlikte ve 1980'lerin başına gelindiğinde Hiyerarşik veri tabanlarının, ilişkisel veri tabanı sistemlerinin, veri modelleme araçlarının ilişki diyagramlarının kullanıldığı, dizin(indeks) mekanizmalarının ortaya çıktığı, yapısal sorgulama dilinin (SQL) kullanıldığı, kullanıcı ara yüzleri, form ve raporların oluşturulduğu, veri tutarlılığı mekanizmalarının geliştirildiği görülmektedir.

1980'lerin ortalarıyla birlikte gelişmiş veri modelleri, obje veri tabanları, mekânsal veri tipleri, çoklu ortam ve akan veri tipleri, sensör verileri gibi kavramların veri operasyonlarında ele alınabildiği gözlemlenmektedir. 1980'lerin sonlarından itibaren veri ambarı kavramının oluştuğu, veri ambarı kavramıyla veri madenciliği, veri sınıflandırma, kümeleme, ilişki desenlerinin ortaya çıkarılması, sapan verilerin analizi, eğilim analizi gibi operasyonların gerçekleştirilmekte olduğu; web madenciliği, ihlal tespiti gibi bilgi keşiflerinin gerçekleştirildiği gözlemlenmektedir. Bu gelişmelere paralel olarak 1990'lardan itibaren genişletilebilir işaretleme dili (XML) tabanlı veri tiplerinin ortaya çıktığı, veri tabanlarının bütünleştirilmesi ve transferi gibi operasyonlara ilişkin gelişmelerin yaşandığı gözlemlenmektedir.

Günümüzde ise yapısal ve yapısal olmayan yığın halinde veya akan verilerin analiz edilebilmesi için büyük veri platformları kullanılmaktadır (Han ve Kamber, 2006 : 2).

Bu noktada veri bilimi ile veri analizi arasındaki farkın da açıklanmasında fayda olacaktır. Veri analizi verinin toplanması ve yorumlanmasına yoğunlaşır, odak noktası genellikle geçmiş ve günümüzdür. Veri bilimi ise geçmiş ve günümüz verilerinden oluşturulan modele dayanarak geleceğe dair öneriler, tahminler ve keşfe yönelik analizler geliştirmeye yoğunlaşmıştır (Ankam, 2016 : 2). Şekil 1.2'de veri bilimi ve veri analizi kavramları şematize edilmiştir.



Şekil 1.2. Veri bilimi ve veri analizi (Ankam, 2016 : 2)

Veri madenciliği; veri ön işleme, uygun veri madenciliği algoritmasının seçilmesi, modelin kurulması, başarım testi aşamalarından oluşan bir veriden bilgi keşfi sürecidir. Bu aşamaların her biri model başarımı açısından vazgeçilmez öneme sahiptir. Verideki gürültü, sapan veriler, eksik veriler veri ön işleme sürecinde ele alınabilmektedir. Uygun veri madenciliği algoritmasının seçimi ağırlıklı olarak istatistik bilimi çerçevesinde çözüm bulmaktadır. Model başarımının ölçülmesi de istatistik ve alan bilgisinin de gerektiği disiplinler arası bir çözüm gerektirmektedir. Ancak veri ön işleme ile veri temizlenip model kurulması aşaması arasında kalan, hatta bazı durumlarda model kurulmasını tekrarlı bir girdi

olarak çalıştıran, verinin büyüklüğü ile önemi daha çok hissedilen, model başarımına doğrudan etki eden bir özellik seçimi süreci bulunmaktadır.

Özellik seçimi; mevcut özelliklerden başarım kriterini optimize eden bir alt kümenin seçilmesi; özelliklerin başarım kriterine göre en iyi belirli bir sayıda özelliğin seçilmesi ve seçilen özellik ve başarı kriteri arasında bir dengenin sağlanması olmak üzere üç şekilde tanımlanabilir (Molina, Belanche, ve Nebot, 2002).

Başka bir ifadeyle özellik seçimi, veri madenciliği yapmaktaki amacımıza ulaşmak için verinin hangi özelliklerinin sürece dâhil edileceğine dair bir ön hazırlıktır. Örneğin, kişisel verilerin girdi olarak alınarak bir kişinin uyuşturucu madde kullanımına yatkınlığı hakkında tahminleme probleminde hangi özelliklerin model için hangi ağırlıkta kullanılacağı model başarımı ve performansı açısından büyük öneme sahiptir.

Özellik seçimi ile birlikte bir diğer aşama da boyut azaltma çalışmasıdır. Özellik seçiminde ilgili özellikler seçilip ilgisiz özellikler elenirken; boyut azaltmada özelliklerin oluşturduğu boyutların temsil kabiliyetlerini çoğunlukla koruyarak indirgenmesi söz konusudur.

Makine öğrenmesi, veri madenciliği çalışmalarının veri ve teknolojide meydana gelen gelişmeler ile paralel olarak büyük miktarda ve sürekli erişim ihtiyacı ihtiva eden veri üzerinde gerçekleştirilen bilgi keşfi olarak tanımlanabilir. Bilişim dünyasında meydana gelen değişimler veri madenciliği uygulamalarını günümüzde makine öğrenmesi kavramıyla ortak platforma oturtmaya başlamaktadır.

Bu tez çalışmasında büyük veri ile makine öğrenmesi çalışmalarının otomatikleştirilebilmesinde, özellik seçimi ve boyut azaltma konusu araştırılmaktadır. Konu araştırılırken çözümler büyük veri sistemleri içerisinde ele alınacaktır.

Makine öğrenmesi günümüz itibariyle çok popüler bir konudur. Makine öğrenmesi çalışmaları, istatistik bilgilerinin yazılımsal olarak örneklenmesi şeklinde karşımıza çıkmaktadır. Bir istatistikçinin alet çantası olarak yüksek sayıda tekrar eden işlemlerin bilgisayara yaptırılması niteliği taşıyan bu yazılımsal örneklemelerin büyük ölçekte veri ile çalışan organizasyonlarda gerçek anlamda uygulanması istatistik, yazılım ve veri konusunda yüksek uzmanlık gerektirmektedir. Organizasyonlar için veriden bilgi keşfinin zamanında ve sürekli yapılabilmesi için veriden bilgi keşfi sürecinin otomatikleştirilmesi ihtiyacını doğurmuştur.

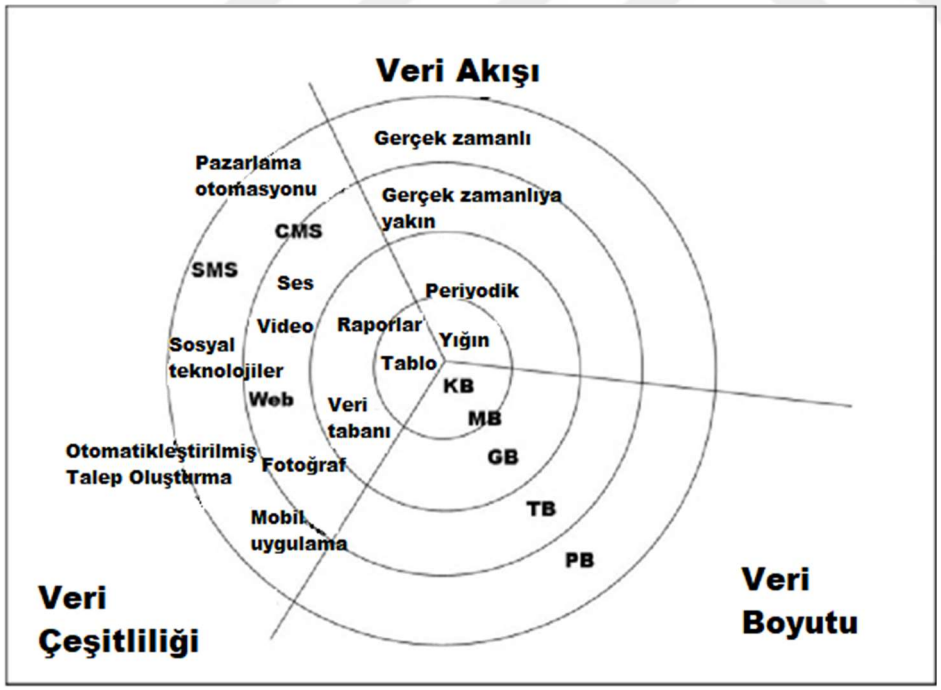
Bu tip çalışmalar için veri her şeydir. Ancak makine öğrenmesi konusunda çalışmaların çoğunlukla yabancı kaynaklarda var olması, ülkemizde bilimsel çalışmalarda anonim verinin kullanımına dair bir ön yargının olması, mevzuatların veriyi elinde bulunduran organizasyonlara veri gizliliği konusunda büyük sorumluluklar yüklemesi organizasyonların veriden bilgi elde edebilme oranını olumsuz etkilemektedir. Ayrıca istatistik bilimini, yazılım mühendisliği ve veri mühendisliği ile bir araya getirecek merkezi veri işleme alt yapılarının yetersizliği test çalışmalarını kısıtlamaktadır.

Çalışmanın ikinci bölümünde büyük veri kavramı ve büyük veri sistemleri kapsamında genel kabul gören Hadoop ekosistemi ile bu ekosistemde makine öğrenmesi çalışmalarında kullanılacak Spark çatısı; üçüncü bölümde özellik seçimi ve boyut azaltma konusunda literatür araştırması; dördüncü bölümde uygulama veri setinin tanıtılması ve çalışmaya yönelik verinin hazırlanması ile uygulama sonuçları; son bölümde değerlendirme ve öneriler yer almaktadır.

2. BÜYÜK VERİ KAVRAMI VE HADOOP EKOSİSTEMİ

İnternete bağlı cihazlar, işletmeler ve insanlar tarafından üretilen veri miktarı üstel olarak artmaktadır. Finans kuruluşları, işletmeler, sağlık kuruluşları gibi hizmet sağlayıcılarından, müşterileri, hastaları veya çalışanları ile olan iletişimleri neticesinde büyük miktarda veri ortaya çıkmaktadır. Bunların da dışında internet aramaları, sosyal medya, GPS sistemleri, borsa işlemleri neticesinde de yapısal veya yapısal olmayan büyük miktarda veri açığa çıkmaktadır. Veri üretimindeki bu çeşitlenme "veri devrimi" veya "Büyük Veri Çağı" şeklinde ifade edilmektedir (Monino ve Sedkaoui, 2016 : 1).

Verinin büyüdüğü aşikârdır. Ancak büyük veri (Big Data) kavramı veri büyüklüğünden ibaret değildir. Verinin hacimsel olarak büyük olması (Volume), verinin çok çeşitli olması (Variety), verinin yüksek hızda akması (Velocity) şeklinde Şekil 2.1'de büyük verinin üç boyutu şematize edilmiştir. Büyük veriye, organizasyon açısından ortaya çıkarılan katma değerinin (Value) de eklenmesiyle dört boyutlu bir büyük veri tanımı da yapılmaktadır (Monino ve Sedkaoui, 2016).



Şekil 2.1. Büyük veri diyagramı (Prajapati, 2013)

4V ile tanılanan büyük verinin işlenmesi yüksek donanım ihtiyaçlarını beraberinde getirmektedir. Donanımların kullanım ömürleri ve teknolojisinin hızla güncellenmesi, dikey genişleme maliyeti veri işleme süreçlerini maliyet sebebiyle sınırlandırmaktadır. Gelinen noktada veriden bilgi keşfi vazgeçilemez ihtiyaç olduğundan bu sınırlılıkları ortadan

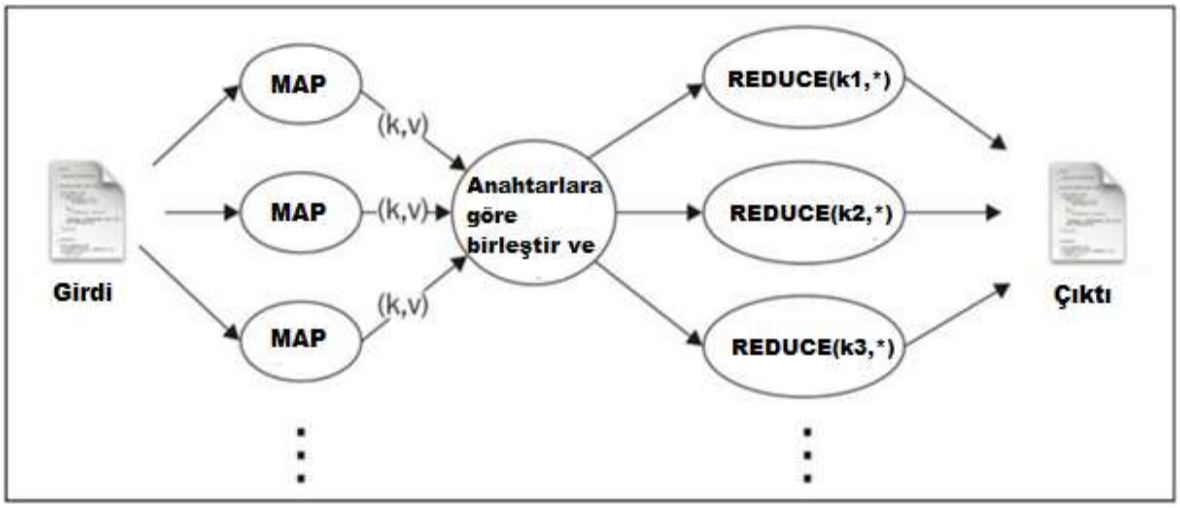
kaldıracak teknolojik gelişme ihtiyacı kendisini göstermiştir. Yatay genişleyebilen, sıradan donanımlar üzerine kurulabilen, hata toleranslı bir küme çalışma yapısı geliştirme çalışmaları Hadoop yapısının oluşmasını sağlamıştır.

Hadoop, Google dosya sistemi ve MapReduce çatısından ilham alan, büyük veri ve dağıtık sistemlerini yönetmek için tasarlanmış, Apache Software Foundation tarafından en üst dereceli proje olarak ele alınmış, açık kaynak kodlu bir Java çatısıdır (Prajapati, 2013).

Hadoop'un bir ekosistem olarak ele alınmasının sebebi bir yazılım geliştirme kitinden (SDK) ibaret olmamasıdır. Linux sistemler üzerine kurulur ve kaynak yönetimi kendi bileşenleri tarafından gerçekleştirilir. Hadoop yalın olarak bir küme bilgisayar çatısı olsa da üzerinde yazılım koşturulabilir olması, veri ambarı, veri tabanı gibi yapıları entegre edebilmesi Hadoop'u büyük veri çalışmalarında bir teknoloji standardı haline getirmiştir.

2.1. MapReduce Çatısı

Google bir arama motoru olarak kurulduğunda internet içeriğini kaydederek, bu verileri arama yapabilmek için indeksleme ihtiyacı duymuştur. Bunun için böylesine bir veriyi işleyebilecek fonksiyonel programlama paradigmasındaki "map" ve "reduce" fonksiyonlarından esinlenerek MapReduce isminde bir uygulama geliştirme çatısı oluşturmuştur (Perera ve Gunarathne, 2013 : 29). Şekil 2.2'de MapReduce çalışma mantığı şematize edilmiştir.



Şekil 2.2. MapReduce çalışma prensibi (Perera ve Gunarathne, 2013:33)

2.2. Spark Çatısı

Spark, ölçeklendirilebilirlik problemlerini verimli bir şekilde çözebilmek için paralelleştirilebilir programların yazılmasını basitleştirecek; MapReduce çatısının yerini alacak bir çatı geliştirme fikrinin ürünüdür (Ganelin, Orhian, Sasaki, ve York, 2016 : 5).

MapReduce çatısında Map ve Reduce görevleri esnasında dosya sistemi (HDFS) kullanıldığından bir darboğaz söz konusudur. Spark, MapReduce paradigmasına hafıza verimliliğinin entegre edilmesi üzerine kurulmuş, Scala, Java ve Python ile uygulama geliştirmeyi destekleyen dağıtık işleme çatısıdır.

Spark ile akan veri, graf verisi, metin analizi, makine öğrenmesi gibi birçok büyük veri operasyonu yapmak mümkündür. Bu çalışmanın konusu gereği Spark'ın yalnızca özellik seçimi ve boyut azaltma ile ilgili özelliklerinin açıklaması yapılacaktır.

Correlation (Korelasyon)

İki veri dizisi arasındaki korelasyonu hesaplamak istatistik çalışmalarında yaygındır. SparkML'de Pearson ve Spearman korelasyonları desteklenmektedir.

Hypothesis testing (Hipotez testi)

Hipotez testi, sonucun istatistiksel olarak anlamlı olup olmadığını, bu sonucun tesadüfen oluşup oluşmadığını belirlemek için istatistikte güçlü bir araçtır. SparkML, bağımsızlık için Pearson'in Ki-kare (ChiSquareTest , $\chi 2$) testlerini desteklemektedir.

Ki-kare, etikete karşı her özellik için Pearson'un bağımsızlık testini yapar. Her özellik için, (özellik, etiket) çiftleri, Ki-kare istatistiğinin hesaplandığı olasılık matrisine dönüştürülür. Tüm etiket ve özellik değerleri kategorik olmalıdır.

Principal component analysis (Boyut indirgeme)

Temel Bileşen Analizi ile Boyut indirgeme (PCA:Principal component analysis), Byrnes'e göre bir özellik uzayını daha az sayıda boyutla temsil etmeye yarayan matematiksel olarak $s = wx$ şeklinde tanımlanabilen doğrusal bir dönüştürmedir. Burada x orijinal veri seti, w ise dönüştürme matrisi ve s ise temsil uzayındaki veridir. Boyut indirgeme çok boyutlu analizde kullanılan en basit ve en yaygın yöntemdir (Byrnes, 2006 : 154).

Boyut indirgeme, olasılık korelasyonlu değişkenlerden oluşan bir gözlem setini, temel bileşenler olarak adlandırılan doğrusal olarak ilişkili olmayan değişkenler setine dönüştürmek için ortogonal (birbiriyle korelasyonu olmayan) bir dönüşüm kullanan istatistiksel bir prosedürdür (Apache Spark 2.4.1 MLlib, 2019).

Boyut indirgeme ve özellik seçimi birbirinden farklı işlemlerdir. Özellik seçiminde, özellik sayısı azaltılırken; boyut indirgemede özellikler aynen kalarak daha küçük vektörler ile ifade edilmektedir. Boyut indirgemede dikkat çeken husus boyut azaltılırken özelliklerde bir veri kaybının yaşanmıyor oluşudur.

PCA'in daha kolay anlaşılması için bir örnek üzerinde anlatım faydalı olacaktır. Çizelge 2.1'de özellikleri F1 ve F2 olan bir veri seti tanımlayalım. Her bir özelliğin aritmetik ortalaması özellik sütunlarının altında verilmiştir. Veri setindeki veriler rastgele seçilmiştir.

Çizelge 2.1. PCA Örnek veri seti

	F1	F2
	2	5
	4	10
	6	15
	8	18
	10	25
Ortalama	6	14,6

Aşağıdaki gibi örnek veri seti matrisimiz olsun. Ölçeklendirme için verilerin ortalamadan farkları alıyoruz:

$$\begin{bmatrix} -4 & -9{,}6 \\ -2 & -4{,}6 \\ 0 & 0{,}4 \\ 2 & 3{,}4 \\ 4 & 10{,}4 \end{bmatrix}$$

İlk aşama olarak kovaryans matrisi çıkartılır.

Kovaryans matrisimiz aşağıdaki gibi hesaplanacaktır:

$$\begin{bmatrix} cov(F1,F1) & cov(F1,F2) \\ cov(F2,F1) & cov(F2,F2) \end{bmatrix}$$

Kovaryans matrisi için değerler aşağıdaki gibi olacaktır:

$$\begin{bmatrix} 8 & 19{,}2 \\ 19{,}2 & 46{,}64 \end{bmatrix}$$

İkinci aşama öz değer ve öz vektörlerin hesaplanmasıdır. Öz değerler kovaryans matrisinden λI birim matrisinin farkının determinantını 0 yapan λ kökleridir.

Öz değerler Eş. 2.1'deki şekilde hesaplanacaktır:

$$\left|\begin{bmatrix} cov(F1,F1) & cov(F1,F2) \\ cov(F2,F1) & cov(F2,F2) \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right| = 0 \qquad (2.1)$$

$$\left|\begin{bmatrix} 8-\lambda & 19{,}2 \\ 19{,}2 & 46{,}64-\lambda \end{bmatrix}\right| = 0$$

Determinanttan kökler çözüldüğünde öz değerler $\lambda_1 \approx 0{,}082$ $\lambda_2 \approx 54{,}56$ olarak hesaplanır. Sıradaki işlem öz vektörlerin bulunmasıdır. Kovaryans matrisinden λ katsayılı birim matrisinin farkı ile matris çarpımında sonucu 0 yapan vektörler öz vektörlerdir. Eş. 2.2'de gösterilmektedir.

$$\begin{bmatrix} 8-\lambda & 19{,}2 \\ 19{,}2 & 46{,}64-\lambda \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \qquad (2.2)$$

Hesaplanan λ değerleri için işlem yapıldığında:

$\lambda = 0{,}082$ için :

$$\begin{bmatrix} 8-\lambda & 19{,}2 \\ 19{,}2 & 46{,}64-\lambda \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} 8-0{,}082 & 19{,}2 \\ 19{,}2 & 46{,}64-0{,}082 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} 7{,}918 & 19{,}2 \\ 19{,}2 & 46{,}558 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

$$7{,}918.x_1 + 19{,}2.x_2 = 0$$
$$19{,}2.x_1 + 46{,}558.x_2 = 0$$

$$\begin{bmatrix} 8-\lambda & 19{,}2 \\ 19{,}2 & 46{,}64-\lambda \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

$x_2 = a$ dersek $a\begin{bmatrix}0{,}412\\1\end{bmatrix}$ öz vektörünü elde ederiz.

$\lambda = 54{,}56$ için :

$$\begin{bmatrix}8-54{,}56 & 19{,}2\\ 19{,}2 & 46{,}64-54{,}56\end{bmatrix}.\begin{bmatrix}x_1\\x_2\end{bmatrix}=0$$

$$\begin{bmatrix}-48{,}56 & 19{,}2\\ 19{,}2 & -7{,}92\end{bmatrix}.\begin{bmatrix}x_1\\x_2\end{bmatrix}=0$$

$$-48{,}56.x_1 + 19{,}2.x_2 = 0$$
$$19{,}2.x_1 - 7{,}92.x_2 = 0$$

$x_2 = a$ dersek $a\begin{bmatrix}-2{,}42\\1\end{bmatrix}$ öz vektörünü elde ederiz.

Sıra temel bileşen seçimine geldiğinde, yani özellik uzayı kaç boyuta indirgenmek isteniyorsa λ değeri ve λ değerine bağlı olarak bulunan öz vektörler λ değeri büyük olandan başlanarak seçilir. Veri seti ile seçilen öz vektör veya öz vektörler çarpıldığında temel bileşenler elde edilmiş olur.

İki boyutlu olan örnek veri setimiz bir boyuta indirgenmek istenirse Eş. 2.3'teki temel bileşenler elde edilmiş olur:

$$Temel\ Bileşen\ Matrisi = \begin{bmatrix}-4 & -9{,}6\\ -2 & -4{,}6\\ 0 & 0{,}4\\ 2 & 3{,}4\\ 4 & 10{,}4\end{bmatrix}.\begin{bmatrix}x_1\\x_2\end{bmatrix} \qquad (2.3)$$

$$\begin{bmatrix}-4 & -9{,}6\\ -2 & -4{,}6\\ 0 & 0{,}4\\ 2 & 3{,}4\\ 4 & 10{,}4\end{bmatrix}.\begin{bmatrix}-2{,}42\\1\end{bmatrix}=\begin{bmatrix}0{,}08\\0{,}24\\0{,}4\\-1{,}44\\0{,}72\end{bmatrix}$$

Böylelikle veri setimizin iki boyutundaki bilgi tek boyuta indirgenmiş olmaktadır. Bu işlem daha fazla boyutun daha az boyuta indirgenmesi için de kullanılabilecektir.

Polinomial Expansion(Boyut yükseltme)

Boyut yükseltme (PolinomialExpansion) boyut indirgemenin tersine özelliklerin belirlenen bir dereceye genişletilmesini sağlar. Farklı veri setlerinin aynı boyutta birleştirilmesinde kullanılabilmektedir.

StringIndexer

StringIndexer, veri etiketlerini içeren metinsel bir kolonu indislere kodlar. Indisler etiket frekanslarına göre sıralıdır, en yüksek frekans 0 indisini alır. Frekansı 0 olan elemanların tutulması isteniyorsa "numLabels " indisine yerleştirilir. Girdi kolonu sayısal ise sayısal değer metne dönüştürülür ve metinsel değerler indekslenir. Estimator veya Trasformator kullanılacaksa girdi olarak metinsel indexlenmiş kolon kullanılmalıdır (Apache Spark 2.4.1 MLlib, 2019).

Aşağıdaki gibi bir veri setimiz olduğunu kabul edelim:

Çizelge 2.2. StringIndexer çalışma tablosu

SN	Kategori	Index
0	a	0,0
1	b	2,0
2	c	1,0
3	a	0,0
4	a	0,0
5	c	1,0

Çizelge 2.2 incelendiğinde "Kategori", "a", "b", "c" değerlerinden oluşan metinsel bir kolondur. StringIndexer "Kategori" kolonunu girdi olarak alır ve "Index" kolonunu oluşturulduğu görülmektedir. "a" en sık etiket olduğundan index numarası olarak 0'ı almıştır. Sırasıyla "c" 1, "b" ise 2 index numarasını almıştır (Apache Spark 2.4.1 MLlib, 2019).

IndexToString

StringIndexer ile frekanslarına göre indekslenen etiketleri, indeks numaralarından geri türetmek için kullanılır.

OneHotEncoderEstimator

Kategorik değerleri sayısal olarak indexlemek için kullanılır. StringIndexer'dan farkı ise indexleme sonucunda çıkan indislerin birbirine üstünlüğü olmayacak şekilde bir dönüşüm yapılmasıdır. StringIndexer'da indisler frekans değerlerine göre sıralı olarak verilmekteydi. OneHotEncoderEstimator'da ise üretilen indis değerlerinde bir sıralama söz konusu değildir. Ülke isimleri, kişi isimleri örnek olarak verilebilir.

VectorIndexer

VectorIndexer, vektör veri setlerindeki kategorik verileri indekslemeye yardımcı olur. VectorIndexer hem hangi özelliklerin kategorik olduğunu tespit edebilir, hem de orijinal veriyi kategori indisine dönüştürebilir. Aşağıdaki şekilde çalışır:

1. maxCategories parametresi ve vektör tipinde bir girdi kolonu al
2. En fazla maxCategories ile tanımlanan değere göre hangi özelliklerin kategorik olacağını belirle
3. Her bir kategorik özellik için 0 tabanlı indisleri hesapla
4. Kategorik özellikleri indexle ve orijinal değerleri indislere dönüştür (Apache Spark 2.4.1 MLlib, 2019).

Normalizer

Normalizer, vektörlerden oluşan bir veri setinin vektörlerinin hepsini p-norm parametresine dayanarak ortak ölçeğe dönüştüren bir özellik dönüştürücüdür. Makine öğrenmesi algoritmalarının performansının arttırılmasına yardımcı olabilmektedir (Apache Spark 2.4.1 MLlib, 2019).

StandardScaler

StandardScaler, vektörlerden oluşan bir veri setindeki satırları standart sapmaya veya ortalamaya göre ölçeklendiren bir dönüştürücüdür (Apache Spark 2.4.1 MLlib, 2019).

MinMaxScaler

MinMaxScaler, vektörlerden oluşan bir veri setindeki satırları belirli iki değer (genellikle 0 ile 1) arasında ölçeklendiren bir özellik dönüştürücüdür. Ölçeklendirilmiş değer Eş. 2.4'teki şekilde hesaplanır:

$$Ölçeklendirilmiş(ei) = \frac{ei - Emin}{Emax - Emin} * (max - min) + min \quad (2.4)$$

MaxAbsScaler

MaxAbsScaler vektörlerden oluşan bir veri setindeki her bir özelliğin değerini veri setinin maksimum değerinin mutlak değerine bölerek -1 ile 1 arasında ölçeklendirme yapan bir özellik dönüştürücüdür (Apache Spark 2.4.1 MLlib, 2019).

Bucketizer

Bucketizer, sürekli değerleri gruplara dönüştüren bir özellik dönüştürücüdür. Splits parametresi sürekli değerlerin bölüneceği sepetleri temsil eder. (n+1). değer için n tane sepet vardır. Son sepet hariç her bir sepet [x,y) aralığındadır, son sepet y yi içerir. Sepetler artan olmalıdır. Alt ve üst sınırlar bilinmiyorsa eksi sonsuz için "Double.NegativeInfinity", artı sonsuz için "Double.PositiveInfinity" kullanılır (Apache Spark 2.4.1 MLlib, 2019).

ElementwiseProduct

ElementwiseProduct, girdi vektöründeki her değeri ağrılık vektöründeki karşılığı ile skaler çarpan bir dönüştürücüdür (Apache Spark 2.4.1 MLlib, 2019).

VectorAssembler

VectorAssembler, özellik listesinin değerlerini tek bir vektörde birleştiren dönüştürücüdür. Makine öğrenmesinde lojistik regresyon ve karar ağaçlarında kullanışlıdır (Apache Spark 2.4.1 MLlib, 2019) .

Çizelge 2.3. VectorAssembler çalışma tablosu

ID	Hour	Mobile	UserFeatures	Clicked	Features
0	18	1,0	[0,0; 10,0; 0,5]	1,0	[18,0; 1,0; 0,0; 10,0; 0,5]

Çizelge 2.3'te VectorAssembler bileşeninin sayısal değer olan "Hour", "Mobile", "vektörel değer olan "UserFeatures" özelliklerini alarak "Features" isminde tek bir vektör oluşturduğu görülmektedir (Apache Spark 2.4.1 MLlib, 2019).

VectorSizeHint

VectorAssembler gibi özellikleri tek vektöre dönüştürür. Ancak vektörize ederken sabit bir genişlikte vektör üretmeyi sağlayan özellik dönüştürücüsüdür.

QuantileDiscretizer

QuantileDiscretizer, Bucketizer gibi sürekli değerleri kategorize etmek için kullanılır. Ancak QuantileDiscretizer sepetlere koyma işlemi için istatistikteki çeyreklik (approxQuantile) yöntemini kullanır (Apache Spark 2.4.1 MLlib, 2019).

Imputer

Veri ön işlemenin en önemli süreçlerinden birisi de eksik veriler ile başa çıkmaktır. Eksik veriler, sütun ortalaması, sınıf ortalaması, en çok tekrar eden değer, varsayılan boş değer veya belirli bir algoritmaya göre değer belirleme yöntemlerine göre ele alınmaktadırlar. Eksik değerler ile ilgili işlemler Spark çatısında Imputer vasıtasıyla yapılmaktadır.

Çizelge 2.4. Imputer çalışma tablosu

A	B	A1	B1
1,0		1,0	4,0
2,0		2,0	4,0
	3,0	3,0	3,0
4,0	4,0	4,0	4,0
5,0	5,0	5,0	5,0

Çizelge 2.4'te A özelliğinde eksik olan değerin A1 özelliğinde, B özelliğinde eksik olan değerlerin B1 özelliğinde Imputer vasıtasıyla hesaplandığı görülmektedir.

Özellik seçimi

Bu çalışmanın odak noktalarından biri de özellik seçimidir. Özellik seçimi daha önce de açıklandığı üzere veri madenciliği için çok önemli bir süreçtir. Bu kısımda Spark çatısı ile büyük verilerde özellik seçiminin nasıl yapılacağı incelenecektir.

VectorSlicer

VectorSlicer, özellik vektöründe verilen bir kritere göre bir alt vektör oluşmasını sağlayan özellik seçicidir.

RFormula

Spark veri madenciliğinde yaygın olarak kullanılan R dilinin '~', '.', ':', '+', '-' gibi operatörlerinin kullanılmasına imkân sağlamaktadır.

ChiSqSelector

İki verinin birbirleri ile korelasyonunun olup olmadığı, var ise korelasyonun tipi ile ilgili istatistikte sıkça kullanılan yöntemlerden birisi *Pearson* anlamlılık testidir. Eş. 2.5'te Pearson's product moment coefficient eşitliği verilmiştir.

$$r_{A,B} = \frac{\sum_{i=1}^{N}(a_i - \bar{A})(b_i - \bar{B})}{N_{\sigma_A \sigma_B}} = \frac{\sum_{i=1}^{N}(a_{ib_i}) - N\bar{A}\bar{B}}{N_{\sigma_A \sigma_B}} \quad (2.5)$$

Eş. 2.5'te N eleman sayısı, a_i ve b_i A ve B ye bağlı i sıralı elemanlar; $\bar{A}$, A' nın ortalaması ve $\bar{B}$ ise B'nin ortalama değeridir. σ_A A'nın, σ_B B'nin standart sapmasıdır.

Eğer $r_{A,B} > 0$ ise A ve B arasında pozitif korelasyon vardır. Yani A'nın değeri artarsa B'nin de değerinin artması beklenir. $r_{A,B}$ değeri ne kadar büyükse korelasyon o kadar güçlüdür. Eğer $r_{A,B} = 0$ ise A ve B arasında korelasyon yoktur. Eğer $r_{A,B} < 0$ ise A ve B arasında negatif korelasyon vardır. (Han ve Kamber, 2006, p. 68)

$r_{A,B}$ değeri :

- -0,09 ile 0 arasında ise veya 0 ile 0,09 arasında ise korelasyonun olmadığı,
- -0,3 ile -0,1 arasında ise negatif düşük korelasyon, 0,1 ile 0,3 arasında pozitif düşük korelasyon olduğu,
- -0,5 ile -0,3 arasında ise negatif orta dereceli korelasyon, 0,3 ile 0,5 arasında ise pozitif orta dereceli korelasyon olduğu,
- -1 ile -0,5 arasında ise güçlü negatif korelasyon, 0,5 ile 1 arasında ise güçlü pozitif korelasyon olduğu

kabul edilir.

Kategorik veride iki özellik arasında korelasyon olup olmadığı X^2 (chi-square, Ki-kare) ile bulunabilir. ChiSquare Eş. 2.6'ya göre hesaplanmaktadır.

$$X^2 = \frac{\sum(Gözlemlenen - Beklenen)^2}{Beklenen} \tag{2.6}$$

X^2 değeri için chi-square dağılımının kritik değer tablosuna bakılır.

Bu konu istatistikte genel bilinen bir yöntem olduğundan dolayı daha derine inilmeden Spark çatısı ile bu işlemlerin nasıl yapıldığına geçilecektir.

ChiSqSelector, hangi özellikleri seçeceğine karar vermek için Ki-kare bağımsızlık testini kullanır. "numTopFeatures", "percentile", "fpr", "fdr", "fwe" olmak üzere beş parametre desteklenmektedir.

- *numTopFeatures* : Ki-kare testine göre hedef değer ile korelasyonu en yüksek verilen sayı kadar özellik seçilir.
- *percentile* : numTopFeatures parametresi ile aynı mantıkla çalışır, ancak belirli bir sayıda özellik yerine özellikler içerisinden belirli bir yüzde seçilir.
- *fpr* : Belirli bir hata oranı (false positive p-values) değerinin altında kalan tüm özellikler seçilir.
- *fdr* : Benjamini-Hochberg procedure kullanılarak belirli bir hata oranı (false discovery rate) değerinin altında kalan tüm özellikler seçilir.
- *fwe* : Belirli bir hata oranı (family-wise) değerinin altında kalan tüm özellikler seçilir (Apache Spark 2.4.1 MLlib, 2019).

3. ÖZELLİK SEÇİMİ VE BOYUT AZALTMA

Özellik seçimi ve boyut azaltma konusu incelenirken öncelikle literatürde Türkçe "özellik" ve "boyut" olarak adlandırılan kavramları açıklamakta fayda olacaktır. Aşağıdaki gibi bir veri setimiz olduğunu düşünelim:

Çizelge 3.1. Özellik ve boyut kavramı için örnek veri

ID	Ad Soyad	Haftanın Günü	Hava Durumu	Pikniğe Gitme Durumu
1	Kişi 1	Pazartesi	Güneşli	Gitmiyor
2	Kişi 1	Cumartesi	Güneşli	Gidiyor
3	Kişi 1	Cumartesi	Yağmurlu	Gitmiyor
4	Kişi 2	Salı	Bulutlu	Gitmiyor
5	Kişi 2	Pazar	Güneşli	Gidiyor

Çizelge 3.1'deki veri setine göre bir kişinin pikniğe gidip gitmeyeceğinin tahmin edilmesi gibi bir problemimiz olduğunda "Ad Soyad", "Haftanın Günü", "Hava Durumu" bilgilerinin her birisi özellik olarak tanımlanır ("ID" hariç tutulur.). Veri madenciliği problemi, tahminleme veya betimleme karakterinde olabileceği gibi, aynı karakteristik problemde özellik ve boyutlara göre farklı veri madenciliği algoritmaları da kullanılabilir. "Pikniğe Gitme Durumu" ise hedef değerdir, bağımlı değişken olarak da isimlendirilir. Özetle bir veri madenciliği probleminde ulaşılmak istenen bilgiye hedef değer, hedef değeri bulmaya yarayan bilgilere özellik diyebiliriz. Boyut ise veri miktarını, özellik sayısını, özellik veri tiplerini de içine alan karmaşıklık düzeyine dair bir ifadedir. Özellik vektörü n boyutta olup, m sayıda özelliği içerebilir.

Teknolojide meydana gelen değişim, veri madenciliği süreçlerinde yüksek işlem kapasiteli bilişim sistemlerinin bütünleşik kullanımını sağlayarak veri madenciliği süreçlerini makine öğrenmesi alanında ele alma imkânı sağlamıştır.

Deng'e göre makine öğrenmesinde temel problemlerden birisi $X = \{x1, x2, \ldots, xM\}$ şeklinde ifade edilen özellikler ile Y şeklinde ifade edilen hedef değer arasında verilere dayalı bir fonksiyon bulabilmektir. Bazen hedef değer Y özelliklerin tamamıyla bulunamaz, özellikler içerisinden belirli alt küme seçilmelidir. Yeterli veri ve zaman olduğunda özelliklerin tamamını, ilişkisiz olanlar da dahil, kullanmak kolaydır. Fakat bu durumda uygulamada iki probleme karşılaşırız.

1) İlişkisiz özellikler işlem maliyetini arttırmaktadır. Özellik sayısının artması durumunda birçok algoritmanın çalışma zamanı artacaktır.

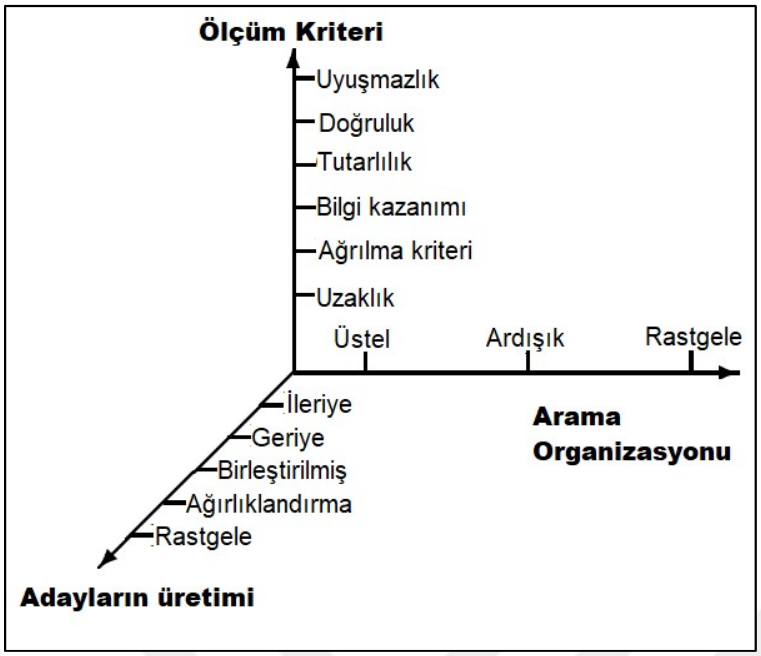
2) İlişkisiz özellikler aşırı öğrenmeye sebep olabilirler. Örneğin "ID" numarasının kullanılması tahminleme için aşırı öğrenmeye sebep olabilecektir (Deng, 1998 : 6,7).

Özellik seçiminde hangi özelliklerin seçileceği konusu detaylandırıldığında olması gereken özelliklerin tespiti kadar olmaması gereken özelliklerin ve olması ile olmamasının çok da etkisi olmayan özelliklerin tespit edilmesi hedeflenir.

Molina ve arkadaşlarına göre özellik seçimi aşağıdaki şekilde karakterize edilebilir:

1) Arama mekanizması: Özelliklerin ağırlıklandırılması ile ilgili süreçtir. Hipotez kurulurken elde edilen sezgisel bilgilerden faydalanılır. Arama üstel, sıralı ve rastgele karakteristikte olabilir.
2) Aday özelliklerin üretimi: Mevcut hipotez için olası değerlerin önerildiği mekanizmadır. Aşağıdaki beş şekilde aday özelliklerin seçimi söz konusudur:
   a) İleriye dönük seçim: Mevcut çözüme yeni bir özellik eklenir. Eğer başarım artıyorsa bu özellik seçime dâhil edilerek devam edilir. Tüm özellikler gezilerek özellik seçimi gerçekleştirilir.
   b) Geriye doğru eleme: Tüm özellikler dahil edilerek çözüme başlanır. Her turda bir özellik çıkarılır. Başarım artıyorsa bu özellik seçimden kaldırılır.
   c) Birleştirilmiş yöntem: Belirli bir sayıda ileriye dönük, belirli bir sayıda geriye dönük özellik seçim operasyonu çalıştırılır. Hangi operasyonda daha çok özellik seçilmişse o yönde devam edilir.
   d) Ağırlıklandırma: Tüm özellikler başarım kriterine göre test edilerek ağırlıkları elde edilir.
   e) Rastgele: Özellik grupları rastgele bir araya getirilerek optimizasyon aranır.
3) Değerlendirme ölçütü: Seçilen özellikler ile başarım fonksiyonu işletilerek bir başarım değeri elde edilir. İlişki değeri bir fonksiyonun döndürdüğü değerdir, bir özelliğin karakteristiği değildir (Molina, Belanche, ve Nebot, 2002) .

Özellik seçme işleminin karakteristiği Şekil 3.1'de şematize edilmiştir.



Şekil 3.1. Özellik seçme işleminin karakteristiği (Molina, Belanche ve Nebot, 2002)

Özellik, sürekli, kesikli, metinsel v.b olabilir. Özellikleri üç şekilse karakterize etmek mümkündür:

1) İlişkili: Bu özellikler hedef değer üzerinde etkilidirler ve bu özelliklerin hedef değer üzerindeki etkisi diğer özellikler tarafından temsil edilemezler.
2) İlişkisiz: Bu özellikler genellikle rastgele değerdedirler ve hedef değer üzerinde bir etkileri yoktur.
3) Fazlalık: Bu özellikler hedef üzerinde etkilidirler ancak hedef üzerindeki etkileri diğer değişkenler tarafından temsil edilebilir (Deepa ve Ladha, 2011).

Özellik seçme algoritmasında hedeflenen ilişkili özelliklerin tespiti ve aynı zamanda ilişkisiz ve fazlalık özelliklerin hariç tutulmasıdır. Bu şekilde veri setinin boyutu özellik seçilerek azaltılmış olur. Ayrıca vektör boyutunun değiştirilmesi ile boyut azaltma yöntemleri de mevcuttur. Özellik seçme bazı özelliklerin elenmesini sağlarken boyut azaltmada özellikler elenmeden bir kodlama yöntemiyle, örneğin PCA, boyut indirgenmiş olmaktadır.

3.1. Filtre (Filter) Metot

Bu yaklaşımda özellik seçme problemi makine öğrenmesi modeli kurmadan çözülmeye çalışılmaktadır. Bu algoritmalarda temel olarak hedef değer ile özellikler arasında korelasyon tespit edilmeye çalışılmaktadır. Bir korelasyonun varlığı tek başına yeterli olmayacaktır. Korelasyonu olan bir veya bir grup özellik tespit edilirken korelasyonu

olmayan özelikler elenmelidir. Bir ileri aşama olarak özelliklerin de birbirleri ile olan korelasyonu incelenmeli ve bir özelliğin korelasyonu başka özellikler ile de temsil edilebiliyorsa fazlalık olan özellikler de seçimden elenmelidir. Chi-Square Test, Fisher Score Test, Information Gain gibi istatistiksel algoritmalardan yoğun olarak faydalanılmaktadır.

Literatürde Filter metot olarak adlandırılan bu yaklaşım, bir makine öğrenmesi modeli kurulmadığından dolayı oldukça performanslı çalışmaktadır. Performans öncelikli, örneğin akan veri ile işlem yapılan, sistemlerde bu metot kullanılması öncelikli tercih olacaktır. Diğer yandan bu metotların her senaryo için çözüm olduğu da söylenememektedir. Korelasyon tabanlı olduklarından özelliklerin belirli permütasyonları ile ortaya çıkan korelasyonların tespiti oldukça zor bir konudur. Diğer yandan verinin karakteristiği ve veri madenciliği algoritması bir araya geldiğinde ortaya çıkabilecek ağırlıklandırmanın başarımı daha yüksek olabilir.

3.1.1. Ki-kare (Chi-Square)

Bölüm 3 içerisinde açıklanan Ki-kare test, korelasyon tabanlı özellik seçimi için oldukça kullanışlı bir algoritmadır. Daha önce açılandığından burada tekrar edilmeyecektir.

3.1.2. Öklid uzaklığı (Euclidian distance)

Öklid uzaklığı istatistikte de oldukça sık kullanılan bir uzaklık ölçümüdür. Birçok insan uzaklıktan bahsederken aslında Öklid uzaklığından bahseder. Öklid uzaklığı, yani kısaca uzaklık, iki noktanın koordinatlarının karesel uzaklıklarının kareköküdür.

Özellik seçiminde her özelliğin hedef değere olan uzaklığı analitik düzlemde hesaplanır. Burada bir özelliğin hedef değere veya bir referans özelliği olan uzaklığı, başka bir adlandırma ile komşuluğu ölçülmektedir. Öklid uzaklığı Eş. 3.1 ‘e göre hesaplanmaktadır.

$$Uzaklık(x, y) = \sqrt{\sum(x_i - y_i)^2} \tag{3.1}$$

Öklid uzaklığı genellikle standartlaştırılmış veriden değil ham veriden hesaplanır. Bu durum bazı avantajlar sağlar. Örneğin, özellik seçimi sapan veri olma ihtimali olan sonradan eklenecek verilerden daha az etkilenirler. Fakat ham veriden hesaplanma bazı dezavantajlar da getirir. Örneğin bir verinin milimetre cinsinde, diğer verinin kilometre cinsinde olduğu

durumda bu verilerin ham hallerinin kareleri alınarak işlem yapıldığında ağırlıklandırmada problemlerin çıkması kaçınılmazdır (Deepa ve Ladha, 2011).

3.1.3. Bilgi kazanımı

Bilgi kazanımı (Information Gain), bir özellik mevcut olduğunda veya olmadığında entropideki (belirsizlikteki) azalışı ölçer. Entropideki azalış bir özelliğin hedef değer üzerinde ne kadar etkili olduğu ile ilgili bir ölçüttür. Bir özelliğin kodlanması için ne kadar çok alan gerekli ise o özelliğin entropisi daha fazladır. Kodlama ise örneğin metin madenciliğindeki sıkıştırma veya özet (hash) değeri üretme olarak nitelendirilebilir. Kategorik verilerde sınıf sayısı ne kadar fazla ise entropinin yüksek olduğu kabul edilir. Hedef değerler bir özellik için tek sınıfta ise entropi bu özellik için düşüktür denilebilir (Deepa ve Ladha, 2011).

Bir karar ağacı oluşturmak için özellik seçimi yaptığımızı düşünelim. Seçtiğimiz özelliğe göre bilgi kazanımı yani entropi düşüşünü hesaplayarak hangi özellikte en çok bilgi kazanımı/en yüksek entropi düşüşü sağladığımızı bulalım.

$p_1, p_2, p_3 \dots p_i$ toplamları bir olan olasılıklar olsun. Entropi Eş. 3.2'ye göre hesaplanmaktadır.

$$Entropi = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{3.2}$$

Örneklerin hepsi aynı sınıfta ise entropi "0" olacaktır. Örnekler sınıflar arasında eşit dağılmışsa entropi "1" olacaktır. Örnekler sınıflar arasında rastgele dağılmışsa entropi "0" ile "1" arasında bir değer alacaktır.

p_i, D öğrenme kümesindeki bir varlığın C_i sınıfına ait olma olasılığı, $\frac{|C_{iD}|}{|D|}$ olarak ifade edilir. D içindeki bir varlığı sınıflandırmak için gerekli bilgi (D nin entropisi) Eş. 3.3'e göre hesaplanabilir.

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{3.3}$$

D kümesi A niteliğine göre v parçaya bölündükten sonra D'yi sınıflandırmak için gerekli olan bilgi Eş. 3.4'e göre hesaplanabilir:

$$Info_A(D) = \sum_{j=1}^{v} \left(\frac{|D_j|}{|D|} \right) Info(D_j) \tag{3.4}$$

A niteliğine göre bölünmeden dolayı bilgi kazancı Eş. 3.5'te ifade edilmiştir:

$$Gain(A) = Info(D) - Info_A(D) \tag{3.5}$$

Bilgi kazanımı yöntemini kullanan algoritmalara örnek olarak ID3 ve C4.5 algoritmaları verilebilir.

Eş. 3.6'da bir özelliğin entropiyi düşürme miktarı hesaplanmaktadır.

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \left(\frac{|D_j|}{|D|} \right) \log_2 \left(\frac{|D_j|}{|D|} \right) \tag{3.6}$$

Bilgi kazanım oranı Eş. 3.7'ye göre hesaplanmaktadır.

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)} \tag{3.7}$$

Ağaç yapılandırmasına GainRatio değeri en yüksek özellikten başlayarak seçilir.

3.1.4. Correlation-based feature selection (CFS)

Korelasyon tabanlı özellik seçiminde hedef değer ile yüksek korelasyonlu, diğer özelliklerle düşük korelasyonlu özellikler bulunması hedeflenir. Bilgi kazanımı ile bir özelliğin hedef değer ile olan korelasyonu tespit edilebilmekte iken özelliklerin birbirleri arasında korelasyon tespiti zayıf kalmaktadır. Korelasyon tabanlı özellik seçiminde özelliklerin en iyi kombinasyonu bulunmaya çalışılır. Bu yapılırken, forward selection, backward elemination, bi-directional search, best-first search ve genetik algoritmalardan faydalanılmaktadır.

$$r_{zc} = \frac{k\overline{r_{zi}}}{\sqrt{k + k(k-1)\overline{r_{ii}}}} \tag{3.8}$$

Eş. 3.8'de r_{zc} seçilen özellikler ile sınıf değişkeni arasındaki korelasyondur. k, seçilen özellik sayısı, $\overline{r_{zi}}$ seçilen özellikler ile sınıf değeri arasındaki korelasyonların ortalamasıdır.

r_{ii} ise özelliklerin birbirleri arasındaki korelasyonların ortalamasıdır (Deepa ve Ladha, 2011).

3.1.5. Markov blanket filtering

Tüm özellik kümesi F, seçilen özellikler F'nin alt kümesi G olsun. fG G'deki her bir f özelliğinin yansıması olsun. Bu yöntemde bağıl entropi ölçülerek $P(C|F = f)$ ile $P(C|G = fG)$ arasındaki tutarsızlık en aza indirilmeye çalışılır (Koller ve Sahami, 1996). Amaç bağıl entropisi düşük olan G özellik alt kümesini bulmaktır. Tutarsızlık Eş. 3.9'daki şekilde hesaplanır:

$$D(P||Q) = \sum_x P(x)\log(\frac{P(x)}{Q(x)})) \tag{3.9}$$

Bağıl entropi ise Eş. 3.10'a göre hesaplanır:

$$\Delta_G = \sum P(f)D(P(C|F = f)||P(C|G = f_G)) \tag{3.10}$$

M F_i'yi içermeyen bir özellik alt kümesi olsun. Eğer F_i verilen M setinden koşullu bağımsızsa ($G - M - \{F_i\}$) M, F_i için Markov blanket olarak adlandırılır.

G, bir özellik seti olsun ve F_i G içerisinde bir özellik olsun. M'nin G'nin bir alt kümesi ve F_i 'nin Markov Blanket'i olduğu varsayılsın. Yine $G' = G - F_i$ olduğunda $\Delta G' = \Delta G$ varsayılsın. G içinde bir F_i özelliği için Markov Blanket bulunduğunda F_i özelliği güvenle özellik seçiminden çıkarılabilir.

Markov Blanket değeri Eş. 3.11'e göre hesaplanır:

$$\Delta(F_i|M) = \sum P(M = f_M, F_i = f_i)D(P(C|M = f_M, F_i = f_i)||P(C|M = f_M)) \tag{3.11}$$

Algoritma (Deepa ve Ladha, 2011):

1. Başla $G = F$
2. Döngü başlat
3. Her $F_i \in G$ için $M_i = F_i$ ile F_j arasındaki korelasyonun en yüksek değeri için $F_j \in G - \{F_i\}$'nin k elemanlı alt kümesi
4. Her i için $\Delta(F_i|M)$ değerini hesapla.
5. EĞER $\Delta(F_i|M)$ değeri minimum İSE $G = G - \{F_i\}$
6. BİTİR

3.1.6. FCBFS (Fast correlation based feature selection)

Bu yöntemde "simmetric uncertanity" kavramı ile yola çıkılarak bir boyut "predominant" hale gelmişse onun ile ilgili diğer özellikleri yok sayarak ve bu işlemi her bir boyut için tek tek yaparak daha performanslı bir boyut azaltma algoritmasıdır.

Diğer algoritmalar ile karşılaştırıldığında hem hız hem de başarı yönünden iyi durumda olduğu görülmektedir. Relief algoritmasından boyut azaltma kabiliyeti bakımından üstündür (Yu ve Liu, 2003).

Algoritma:S= aday özellikler, $M = \emptyset$ seçilmiş özellikler

1. S içinde $Y \rightarrow \rho_{y,x^*}$ ile korelasyonunu maksimize eden X^* özelliğini ara
2. EĞER $\rho_{y,x^*} \geq \delta$ İSE X^* özelliğini M'ye ekle ve S'den çıkart
3. $\rho_{x,x^*} \geq \rho_{y,x^*}$ şartını sağlayan tüm özellikleri S'den çıkart
4. EĞER $S \neq \emptyset$ İSE GİT(2) DEĞİLSE BİTİR (Deepa ve Ladha, 2011)

3.2. Sarmalayıcı (Wrapper) Metot

Burada belirli özellik kabulü ile bir makine öğrenmesi algoritması kurularak başarım ölçülür. Başarım sonucuna göre özellik seçme işlemi yönetilir. Verinin karakteristiğine ve sektörüne ilişkin alan bilgisi büyük önem taşır. Bu özellik seçme yönteminin otomatikleştirilmesi boş küme ile başlanarak özellik ekleme veya tüm özellikler ile başlanıp özellik eleme şeklinde olabilmektedir.

Ayrıca veri madenciliği problemi için seçilen sınıflandırma, kümeleme, zaman serisi algoritmaları kurulan her bir özellik seçimi hipotezi için ileriye doğru veya geriye doğru sıralı şekilde işletilerek bir özellik seçimi yapılabilmektedir.

Literatürde Wrapper metot olarak adlandırılan bu yaklaşımda verinin karakteristiğine dair bilginin ve veri madenciliği algoritmasının bir araya gelmesiyle daha yüksek başarımlı modeller kurulduğu ve başarımı daha yüksek özellik seçimi yapıldığı söylenebilir. Ancak bu metotta her bir seçim hipotezi için model kurma işlemi gerçekleştirilir. Bu modellerden başarımı en yüksek model seçildiğinde bu modele ilişkin özellik ağırlıklandırması ile özellik seçimi gerçekleştirilmiş olmaktadır. Başarım ölçütünün çok hassas olduğu iş modellerinde performans kaygısına rağmen bu yaklaşım tercih edilebilir.

3.2.1. Sequental forward selection (SFS)

Aç gözlü arama (Greedy search) yöntemiyle çalışır. Seçilmiş özellikler için boş küme ile başlanır. Her bir özellik başarımı arttırıp arttırmadığına bakılarak başarımı arttırıyorsa seçime eklenir.

SFS optimal seçilmiş özellik kümesi küçük olduğunda çok iyi performans verir. Bu yöntemin en büyük dezavantajı yeni eklenen özellikler ile fazlalık haline gelen özelliği çıkaramamasıdır.

Algoritma:

1. Boş küme ile başla. $Y_0 = \emptyset$
2. $X^+ = argmax[J(Y_k + X)]; x \notin Y_k$
3. $Y_{k+1} = Y_k + X^+; k = k + 1$
4. GİT(2)
5. BİTİR (Deepa ve Ladha, 2011)

3.2.2. Sequential backward elimination (SBE)

SFS'nin tam tersi mantıkla çalışır. Tüm özelliklerden teker teker özellikler başarımı en az düşüren özellik çıkartılarak özellik seçimi yapılır. Ancak özelliğin çıkarılması başarımın artmasına da sebep olabilir.

SBS optimal seçilmiş özellik sayısı fazla ise çok iyi performans verir. En çok zamanı büyük özellik alt kümelerini dolaşırken harcar. En büyük dezavantajı çıkarılmış bir özelliğin bir sonraki durumda başarıma etkisinin ölçülememesidir.

Algoritma:

1. Özellik kümesinin tamamı ile başla. $Y_0 = X$
2. $X^- = argmax[J(Y_k - X)]; x \notin Y_k$
3. $Y_{k+1} = Y_k - X^-; k = k + 1$
4. GİT(2)
5. BİTİR (Deepa ve Ladha, 2011)

3.2.3. Genetik algoritma

Özünde bir optimizasyon algoritması olan genetik algoritma doğal seleksiyonun kodlamaya yansıtılmış halidir. Seçim, gen değişimi ve mutasyon fonksiyonları üzerine kurulmuştur. Optimize edilen değer genler ile kodlanarak kromozomlar halinde işlem görür. Genler kendi aralarında gen değişimi yaparlar. Bu tür çeşitliliği sağlar. Ayrıca mutasyon mekanizması ile gen ata bireylerden tamamen bağımsız genlere sahip kromozomları üretilmesini sağlanır. Başarım fonksiyonuna göre seçilim yapılır. Böylelikle hata minimize edilmiş olmaktadır. Özellik seçiminde ise yukarıda anlatılan süreçler ile en iyi özellik alt kümesine ulaşılmak hedeflenir. Ancak her bir nesilde bir model kurulmaktadır.

3.3. Gömülü (Embedded) Metot

Bu yaklaşımda özellik seçimi makine öğrenmesi modeli kurulurken yapılır. Özellik seçimi algoritması makine öğrenmesi algoritması içine gömüldüğünden gömülü metot olarak adlandırılmaktadır. Bu yaklaşımda makine öğrenmesi modellerinin özellik önem ağırlıklandırma yeteneğinden faydalanılır. Örneğin bir basit doğrusal regresyon modeli üretildiğinde:

$$y_i = \beta_0 + \beta_1 x_{i1} + \varepsilon_i \tag{3.12}$$

Eş. 3.12'de verilen basit regresyon denkleminde β_0, β_1, ε_i değerlerinin tespit edilmiş olması gerekmektedir. Bu durumda x_{i1} özelliğinin hedef değer açısından korelasyonu ve ağırlığı belirlenmiş olmaktadır. Destek vektör makineleri bu yaklaşımda genel olarak kullanılmaktadır.

Model kurulurken gerçekleştirilen özellik seçimi algoritmaları, model kurularak gerçekleştirilen özellik seçimi algoritmalarından daha hızlı çalışırlar. Model kurmadan özellik seçimine dayalı yöntem algoritmalarından ise başarımları daha yüksektir. Özellikler arasındaki etkileşimi tespit edebilirler ve makine öğrenmesi modeli eğitilerken özellik seçimi gerçekleştirilir. Bu yaklaşım için SVM-RFE algoritması örnek olarak verilebilir. Literatürde çok sayıda özellik seçme yöntemi bulunmakla birlikte bu kısımda araştırılan konu çerçevesindeki yöntemlere değinilmiştir.

3.4. Özellik Seçme Yöntemlerinin Karşılaştırılması

Çizelge 3.2'de özellik seçme algoritmalarının türleri, avantaj ve dezavantajları karşılaştırılarak bazı özellik seçme algoritmaları listelenmiştir.

Çizelge 3.2. Özellik seçme algoritmalarının karşılaştırılması (Deepa ve Ladha, 2011)

Yöntem		Avantajlar	Dezavantajlar	Örnek Algoritmalar
Filter	Tek Değişkenli	Hızlı Ölçeklendirilebilir Modelden bağımsız	Özellik bağımlılıklarını görmezden gelir Veri madenciliği modeli ile iletişimi yoktur	Ki-kare Öklid Uzaklığı Bilgi Kazanımı
	Çok Değişkenli	Özellik bağımlılıklarını modeller Modelden bağımsız Wrapper metotlardan daha düşük karmaşıklık	Tek değişkenli yöntemlerden daha yavaş çalışır Tek değişkenli modellere göre ölçeklenebilirliği düşüktür Veri madenciliği modeli ile iletişimi yoktur	CFS MBF FCBF
Wrapper	Belirleyici	Basittir Veri madenciliği modeli ile iletişimi vardır Özellik bağımlılıklarını modeller Rastgele yöntemlere göre daha az hesaplama karmaşıklığına sahiptir	Aşırı öğrenme riski Yerel minimum ve yerel maksimuma sıkışma ihtimali rastgele yöntemlerden fazladır Model bağımlı özellik seçimi	SFS SBE Plus L Minus R Beam Search
	Rastgele	Yerel minimum ve yerel maksimuma sıkışma ihtimali daha azdır Veri madenciliği modeli ile iletişimi vardır Özellik bağımlılıklarını modeller	Hesaplama karmaşıklığı fazladır Belirleyici yöntemlere göre aşırı öğrenme riski daha fazladır Model bağımlı özellik seçimi	Tavlama Benzetimi Randomized hill climbing Genetik algoritma Tahmin Dağıtım Algoritmaları
Embedded		Veri madenciliği modeli ile iletişimi vardır Wrapper yöntemlere göre hesaplama karmaşıklığı daha düşüktür Özellik bağımlılıklarını modeller	Model bağımlı özellik seçimi	Karar ağaçları Ağırlıklı Naive Bayes SVM Feature Selection

4. İNTERNET SAĞLAYICI VERİ SETİ İLE ÖZELLİK SEÇİMİ VE BOYUT İNDİRGEME UYGULAMASI

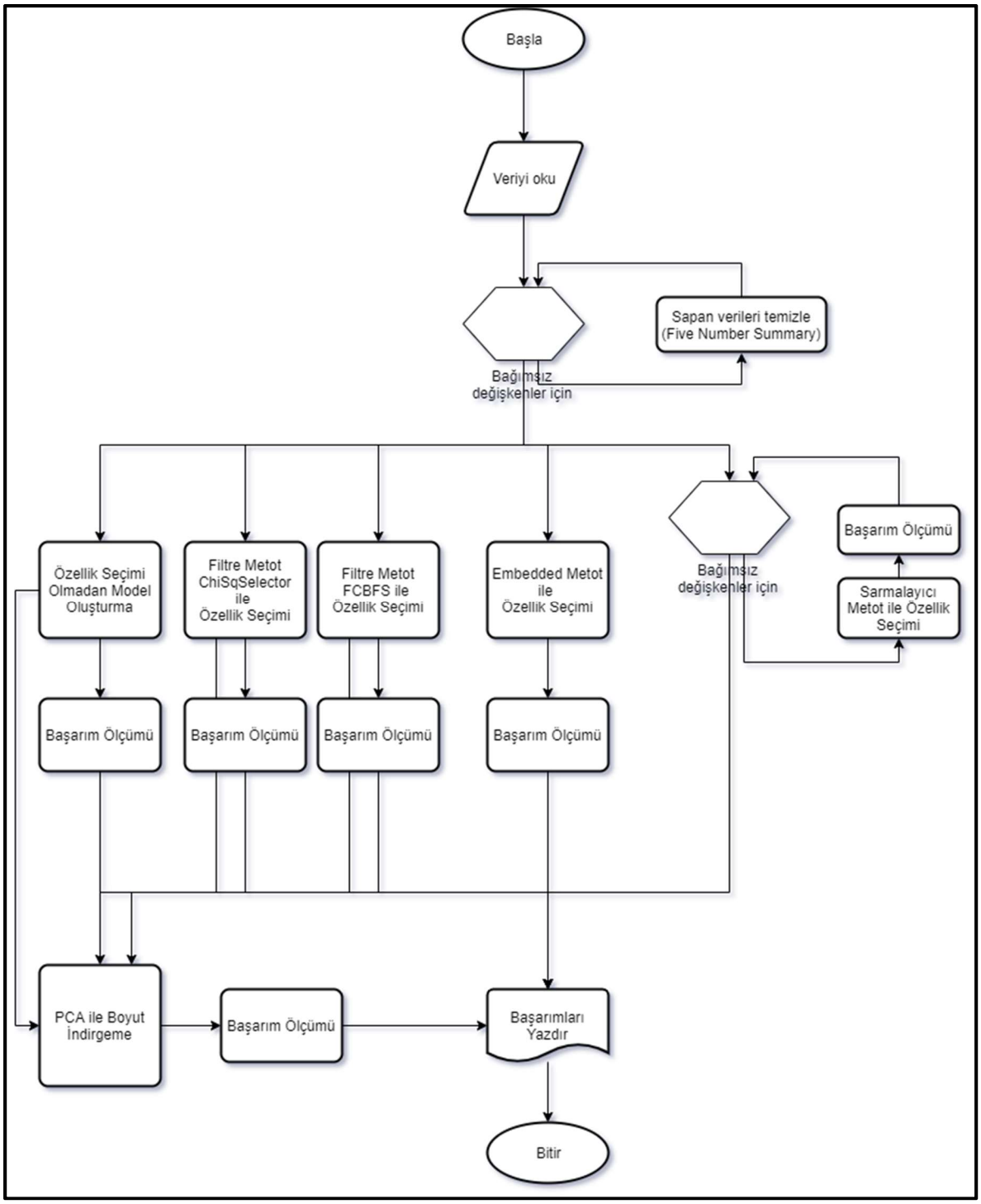
Bu kısımda internet sağlayıcıdan elde edilen verilerin tanıtılması yapılarak önceden tanımlanan başarım ölçütlerine göre oluşturulan makine öğrenmesi modeline göre başarım ölçümü yapılacaktır. Amaç en yüksek başarımlı modeli elde etmek değil, kabul edilebilir başarımı en az özellikle yakalamaktır.

Detayları ilerleyen kısımlarda açıklanacak uygulamaya geniş bir perspektiften bakarsak uygulama Hadoop üzerine kurulan Apache Spark veri işleme ve makine öğrenmesi çatısı kullanılarak kodlanmıştır. FCBFS algoritması Apache Spark çatısı içerisinde bulunmamakla birlikte Spark çatısı üzerinde çalışacak şekilde kodlaması yapılmıştır. Sunulan sonuçlar bu kodların çıktısıdır.

Sapan verilerin tespiti için kullanılacak FiveNumberSummary yöntemini kısaca bir örnekle özetleyebiliriz:

A={2,4,6,8,10} şeklinde bir veri kümemiz olsun. Bu veri setinin ortanca değeri 6, 1. Çeyreklik değeri (2+4)/2 = 3; 3. Çeyreklik değeri (8+10)/2 = 9 şeklinde bulunur. Inter quartile range (IQR) ise 3. Çeyreklik – 1. Çeyreklik yani 9-3=6 şeklinde hesaplanır. 1. Çeyreklikten 1,5*IQR çıkardığımızda elde ettiğimiz değerden yani 3-1,5*6 = -6 değerinden küçük veriler ile 3. Çeyrekliğe 1,5*IQR eklediğimizde yani 9+1,5*6 = 18 değerinden büyük veriler sapan veri olarak değerlendirilmelidir. A ile tanımlanan veri setinde minimum değer 2 -6'dan büyütür ve maksimum değer olan 10, 18'den küçüktür. Bu sebeple otomatik bir şekilde sapan veri olarak ele alınacak bir değer olmamıştır. Eğer olsaydı bu değerler ortanca ile değiştirilecekti. İlerleyen kısımda gerçek veride bu yönteme göre sapan veri şartlarına uyan veriler medyan ile değiştirilecektir.

Yapılacak işlemler Şekil 4.1'de akış şeması olarak gösterilmiştir.



Şekil 4.1 Uygulama ölçümleri için yapılacak işlemler

4.1. Veri Setinin Tanıtılması

Veri seti abonelik süresi, fatura ortalaması, aylık ortalama veri indirme miktarı, aylık ortalama veri yükleme miktarı, tarife fiyatı, hız, kota, üyelik durumu bilgilerinden oluşmaktadır. Veri seti 20000'i aktif, 50000 pasif olmak üzere 70000 kayıttan oluşmaktadır.

Uygulama boş kayıtlar göz ardı edilerek 69 790 kayıt üzerinden yani 488530 hücrelik veri ile gerçekleştirilmiştir.

Abonelik süresi

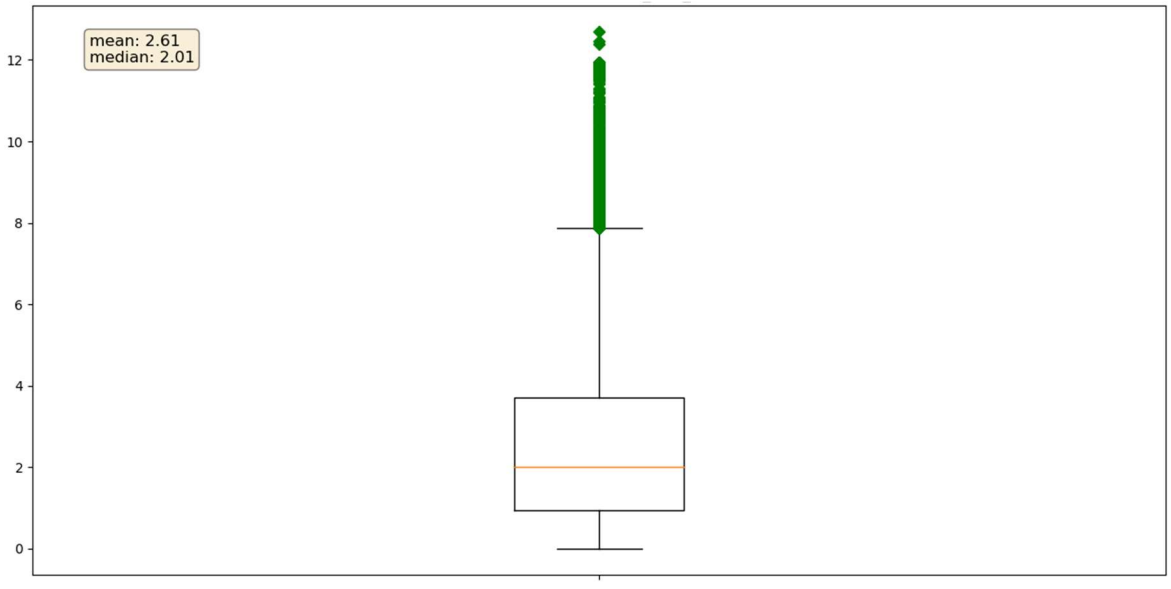
"customer_period" Bir üyenin üyelik yılını belirtir. Sayı tipinde sürekli bir değişkendir. Ortalama değeri 2,61'dir. Standart sapmasının değeri 2,19'dur. Minimum değeri 0, maksimum değeri 12,7'dir. 1.Çeyreklik değeri 0,94; 2.Çeyreklik değeri 2,01; 3.Çeyreklik değeri 3,71'dir.

$$IQR = Q_3 - Q_1$$
$$2{,}77 = 3{,}71 - 0{,}94$$

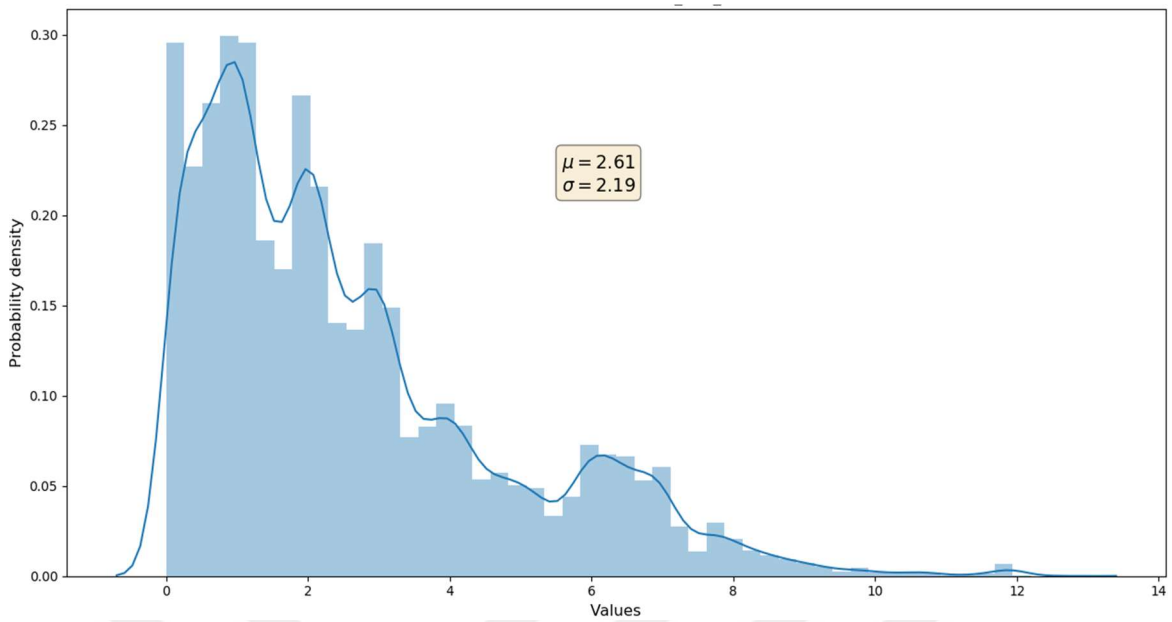
$$Q_3 + 1{,}5IQR < SapanVeri < Q_1 - 1{,}5IQR$$
$$3{,}71 + 1{,}5x2{,}77 < SapanVeri < 0{,}93 - 1{,}5x2{,}77$$
$$7{,}87 < SapanVeri < -3{,}23$$

-3,23'ten küçük, 7,87'den büyük veriler $SapanVeri$ olarak incelenmelidir. Bu şarta uyan 1641 adet veri bulunmaktadır.

Bu özelliğin BoxPlot analizi Şekil 4.2'de ve dağılımı Şekil 4.3'te görülmektedir.



Şekil 4.2. Abonelik süresi BoxPlot



Şekil 4.3. Abonelik süresi dağılım grafiği

Bu veri alanı ile hedef değerimizin arasındaki Pearson korelasyonu -0,109 değerindedir. Sapan veriler medyan ile değiştirildiğinde Pearson korelasyonu -0,087 olmaktadır.

Fatura ortalaması

"avg_invoice" Bir üyenin aylık fatura ortalamasını belirtir. Sayı tipinde sürekli bir değişkendir. Aslında faturanın ücretinin belirli olacağı ön görülerek kesikli olacağı değerlendirilen bu alan, faturanın kota aşımı, özel hizmet gibi etkenler ile değişken değerler

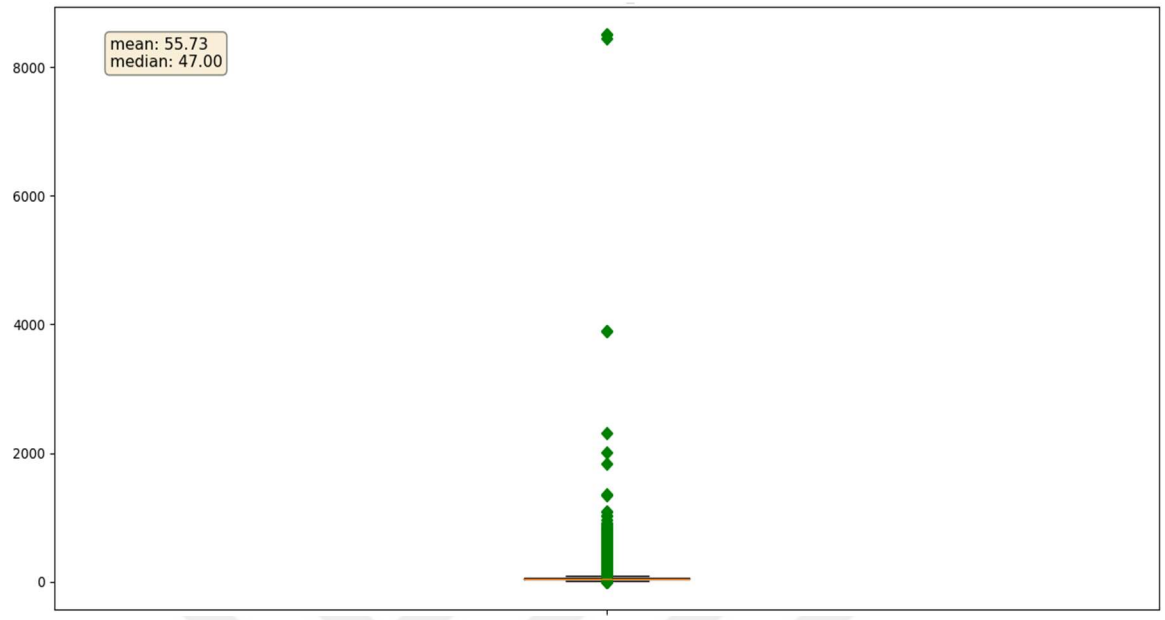
$$IQR = Q_3 - Q_1$$
$$19 = 58 - 39$$

$$Q_3 + 1{,}5IQR < SapanVeri < Q_1 - 1{,}5IQR$$
$$58 + 1{,}5x19 < SapanVeri < 39 - 1{,}5x19$$
$$86{,}5 < SapanVeri < 10{,}5$$

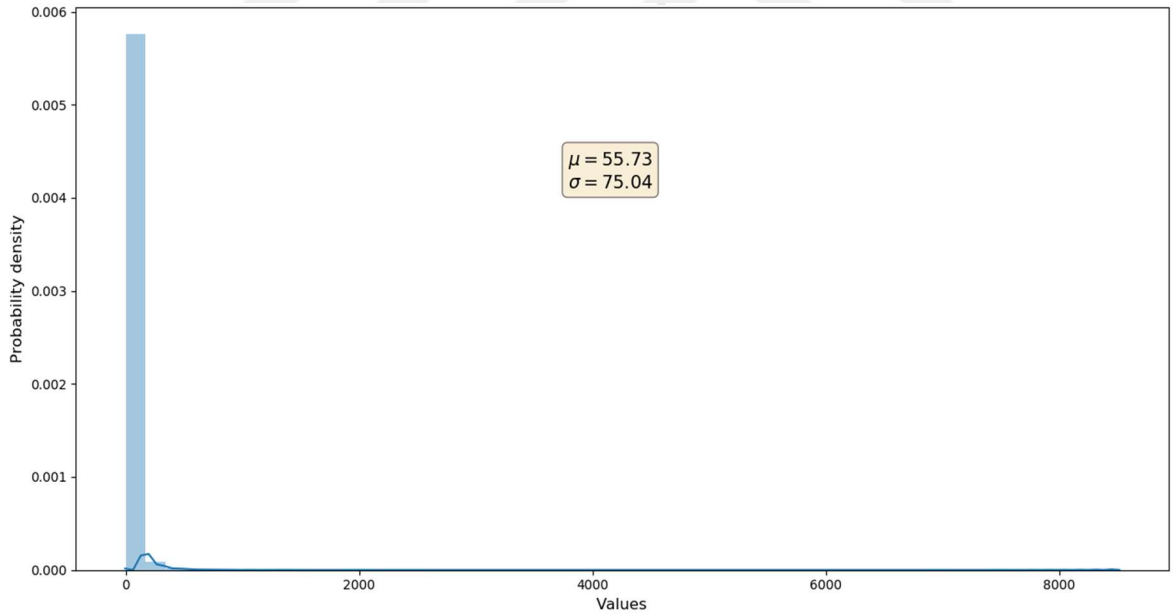
alabileceğinden sürekli değer olarak ele alınmıştır. Ortalama değeri 55,73'tür. Standart sapmasının değeri 75,04'tür. Minimum değeri 1, maksimum değeri 8509'dur 1.Çeyreklik değeri 39, 2.Çeyreklik değeri 47, 3.Çeyreklik değeri 58'dir.

10,5'ten küçük, 86,5'ten büyük veriler $SapanVeri$ olarak incelenmelidir. Bu alanda bu şarta uyan 4924 adet veri bulunmaktadır.

Bu özelliğin BoxPlot analizi Şekil 4.4'te ve dağılımı Şekil 4.5'te görülmektedir.



Şekil 4.4 Fatura ortalaması BoxPlot



Şekil 4.5. Fatura ortalaması dağılım grafiği

Bu veri alanı ile hedef değerimizin arasındaki Pearson korelasyonu 0,065 değerindedir. Sapan veriler medyan ile değiştirildiğinde Pearson korelasyonu -0,158 olmaktadır.

Veri indirme ortalaması

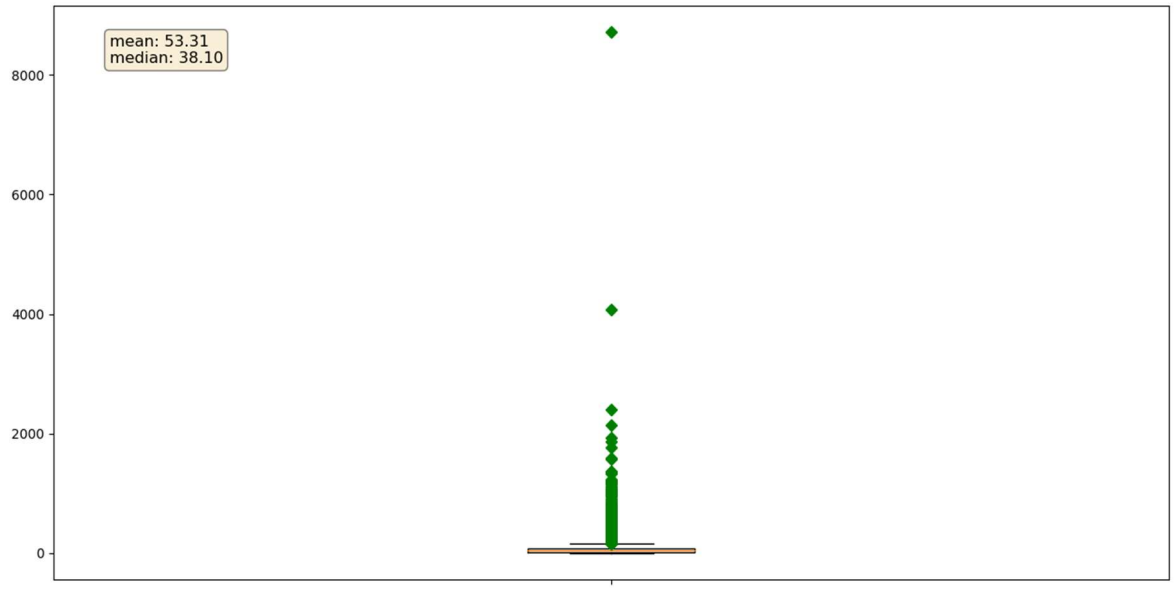
"avg_download_in_gb" Bir üyenin aylık Giga Byte ölçeğinde veri indirme ortalamasını belirtir. Sayı tipinde sürekli bir değişkendir. Ortalama değeri 53,31'dir. Standart sapmasının değeri 72,47'dir. Minimum değeri 0, maksimum değeri 8720,1'dur 1.Çeyreklik değeri 15,4; 2.Çeyreklik değeri 38,1; 3.Çeyreklik değeri 72,3'tür.

$$IQR = Q_3 - Q_1$$
$$56{,}9 = 72.3 - 15{,}4$$

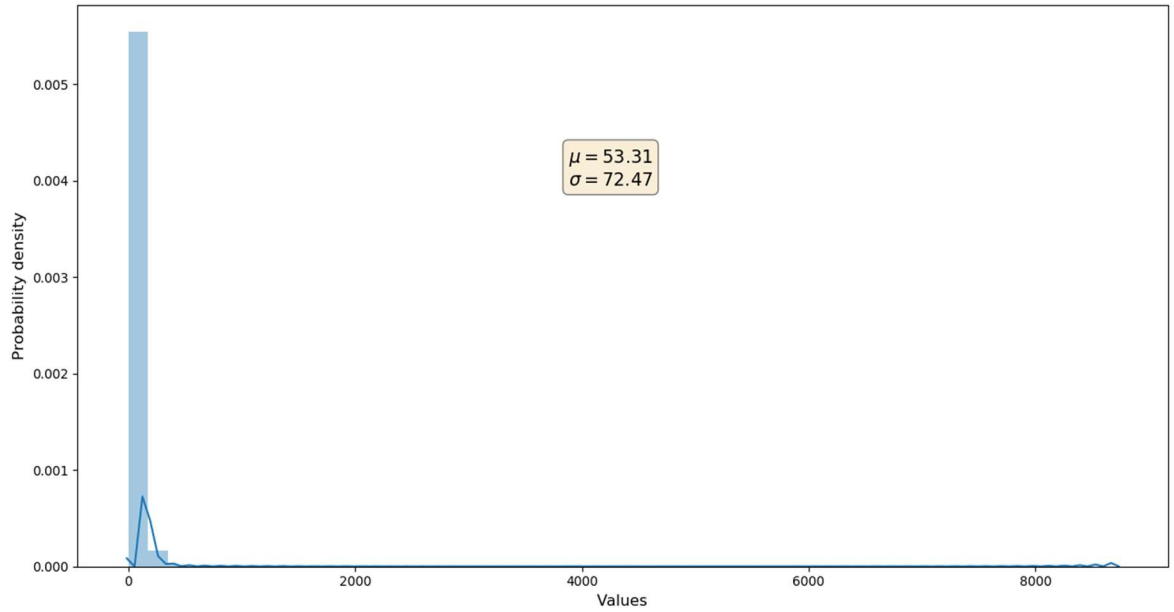
$$Q_3 + 1{,}5IQR < SapanVeri < Q_1 - 1{,}5IQR$$
$$72{,}3 + 1{,}5x56{,}9 < SapanVeri < 15{,}4 - 1{,}5x56{,}9$$
$$157{,}65 < SapanVeri < -69{,}95$$

$-69{,}95$'ten küçük, $157{,}65$'ten büyük veriler $SapanVeri$ olarak incelenmelidir. Bu alanda bu şarta uyan 3119 adet veri bulunmaktadır.

Bu özelliğin BoxPlot analizi Şekil 4.6'da ve dağılımı Şekil 4.7'de görülmektedir.



Şekil 4.6. Veri indirme ortalaması BoxPlot



Şekil 4.7. Veri indirme ortalaması dağılım grafiği

Bu veri alanı ile hedef değerimizin arasındaki Pearson korelasyonu -0,225 değerindedir. Sapan veriler medyan ile değiştirildiğinde Pearson korelasyonu -0,352 olmaktadır.

Veri yükleme ortalaması

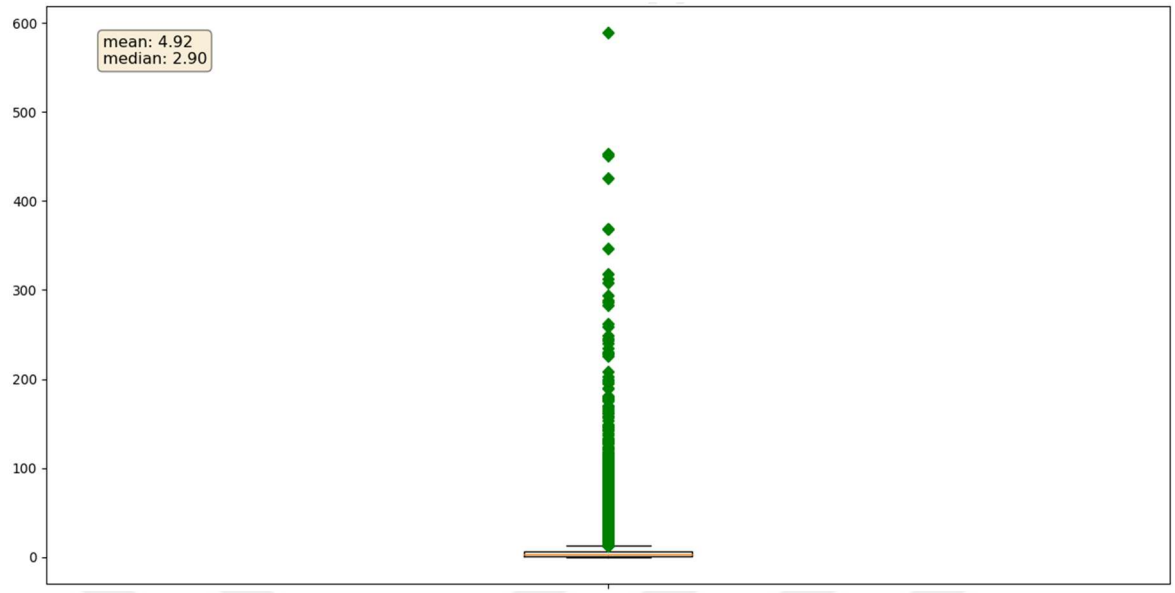
"avg_upload_in_gb" Bir üyenin aylık Giga Byte ölçeğinde veri yükleme ortalamasını belirtir. Sayı tipinde sürekli bir değişkendir Ortalama değeri 4,92'dir. Standart sapmasının değeri 10,34'tür. Minimum değeri 0, maksimum değeri 589,2'dir 1.Çeyreklik değeri 1,1; 2.Çeyreklik değeri 2,9; 3.Çeyreklik değeri 5,8'dir.

$$IQR = Q_3 - Q_1$$
$$4{,}7 = 5{,}8 - 1{,}1$$

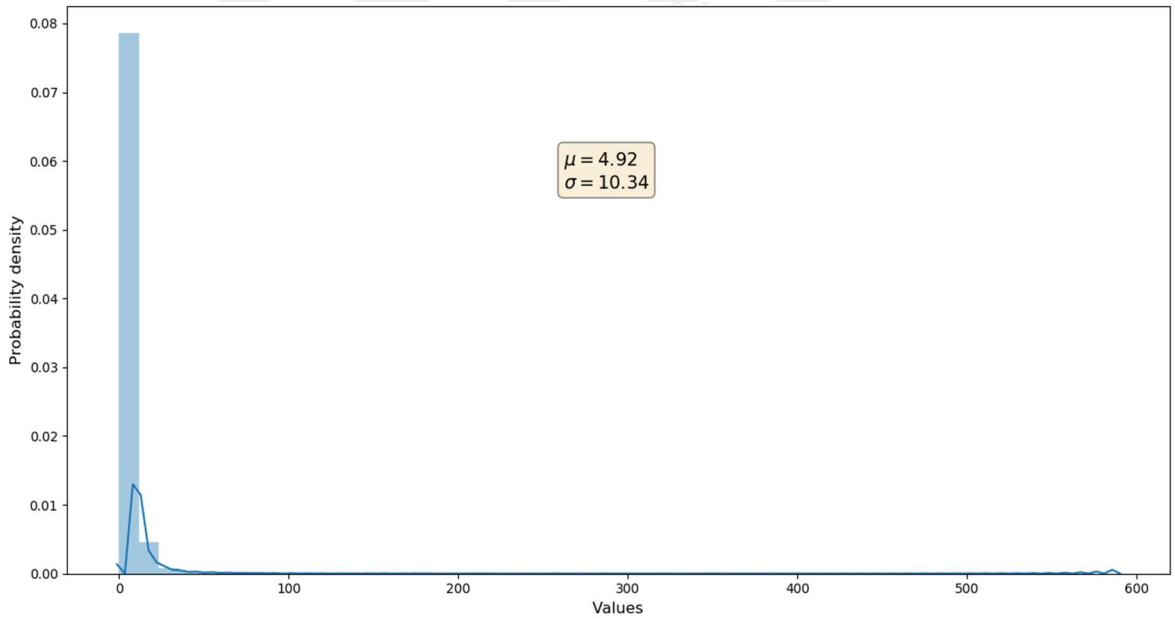
$$Q_3 + 1{,}5IQR < SapanVeri < Q_1 - 1{,}5IQR$$
$$5{,}8 + 1{,}5x4{,}7 < SapanVeri < 1{,}1 - 1{,}5x4{,}7$$
$$12{,}85 < SapanVeri < -5{,}95$$

$-5{,}95$'ten küçük, $12{,}85$'ten büyük veriler $SapanVeri$ olarak incelenmelidir. Bu alanda bu şarta uyan 4297 adet veri bulunmaktadır.

Bu özelliğin BoxPlot analizi Şekil 4.8'de ve dağılımı Şekil 4.9'da görülmektedir.



Şekil 4.8. Veri yükleme ortalaması BoxPlot



Şekil 4.9. Veri yükleme ortalaması dağılım grafiği

Bu veri alanı ile hedef değerimizin arasındaki Pearson korelasyonu -0,131 değerindedir. Sapan veriler medyan ile değiştirildiğinde Pearson korelasyonu -0,329 olmaktadır.

Tarife bilgisi

"tariff" Bir üyenin kullanmakta olduğu tarifenin bilgisini verir. Tarife ismi kategorik bir veridir ancak bir sıralama içermektedir. Bu sıralama fiyat bilgisi ile yapılabilmektedir. Fiyatın değerinden çok bir sıralama ölçütü olması önemlidir. Sayı tipindedir. 154 adet tarife

ismi 54 farklı fiyat bilgisi ile bir sıralamaya tabi tutulmaktadır. Bu özelliğin frekans dağılımı Çizelge 4.1'de görülmektedir.

Çizelge 4.1. Tarife kişi dağılımı

Tarife Fiyat Bilgisi	Üye Sayısı	Tarife Fiyat Bilgisi	Üye Sayısı	Tarife Fiyat Bilgisi	Üye Sayısı
0,25	19	28,88	1	71,70	420
0,50	640	31,08	28	79,67	63
7,50	9398	31,86	1417	95,61	51
7,89	10	35,06	2	100,00	8375
11,14	13	35,46	1	118,71	2
11,55	4	39,03	2	119,52	38
11,94	213	39,04	84	130,00	8935
12,74	22	39,08	71	144,00	9724
14,74	4	39,83	2157	159,36	72
15,93	34	43,49	3	175,00	8605
19,12	10	47,00	5	199,19	16
19,52	1	47,01	6	207,16	25
19,91	12	47,80	1612	239,03	2
19,92	6	54,97	35	278,87	4
20,71	9	55,77	1219	300,00	9200
22,11	1	62,95	5	318,72	2
23,11	802	63,74	350	398,40	3
23,90	400	70,91	2	420,00	5866

Hız bilgisi

"speed_in_mb" Bir üyenin modeminden okunan Mega Bit cinsinden hız bilgisidir. Sayı tipinde kesikli bir değişkendir. Sıralı kategorik bir özellik olarak ele alınmalıdır. Bu özelliğin frekans dağılımı Çizelge 4.2'de görülmektedir.

Çizelge 4.2. Hız kişi dağılımı

Hız(MiB)	Üye Sayısı	Hız(MiB)	Üye Sayısı	Hız(MiB)	Üye Sayısı
0	756	6	1	25	8035
1	7317	10	12274	40	6
2	32	15	9	50	3850
3	394	16	3060	60	4
4	9	20	649	100	1198
5	449	24	31957		

Kota bilgisi

"quota_in_gb" Bir üyenin sahip olduğu kota bilgisidir. Giga Byte cinsinden kesikli bir sayısal değerdir. Sıralı kategorik bir değişken olarak ele alınacaktır. Bu özelliğin frekans dağılımı Çizelge 4.3'te görülmektedir.

Çizelge 4.3. Kota bilgisi kişi dağılımı

Kota (GB)	Üye Sayısı	Kota (GB)	Üye Sayısı	Kota (GB)	Üye Sayısı
0	2709	20	223	100	744
1	9360	25	4035	151	9447
5	160	40	65	201	20
8	4	50	23683	252	2517
10	85	75	16419	504	358
15	120	80	51		

Uygulama, Apache Spark çatısının 2.4.1 versiyonu ile gerçekleştirilmiştir.

4.2. Uygulama

İnternet sağlayıcıdan elde edilen veriler Apache Spark çatısının 2.4.1 versiyonu ile işlenmiştir. Veriler ön işleme tabi tutulduktan sonra özellik seçimi olmadan ve özellik seçimli, boyut indirgeme öncesi ve boyut indirgeme sonrası başarımları tanımlanan başarım ölçütlerine göre ölçülmüştür.

4.2.1. Başarım ölçütleri

Bir makine öğrenmesi algoritmasının başarımını ölçmek için veri madenciliği yöntemine dayanan çok çeşitli yöntemler mevcuttur. Regresyon tabanlı algoritmalarda Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) gibi değerlendirme yöntemleri öne çıkarken, iki veya çok sınıflı sınıflandırma çalışmalarında Confusion Matrix, Accuracy, Precision (Positive Predictive Value), Recall (True Positive Rate), F-measure gibi değerlendirme yöntemlerinin kullanıldığı görülmektedir (Apache Spark 2.4.1 MLlib, 2019).

İki sınıflı sınıflandırmalarda Confusion Matrix başarım için önemli bir değerlendirme yöntemidir. Formüllerde kullanılmak üzere örnek harflendirmeli Confusion Matrix tablosu Çizelge 4.4'te incelenebilir.

Çizelge 4.4. İki sınıflı sınıflandırma için Confusion Matrix tablosu (Visa, Ramsey, Ralescu, ve Knaap, 2011)

	Tahmin edilen değer 0	Tahmin edilen değer 1
Gerçek değer 0	a	b
Gerçek değer 1	c	d

True Positive (TP) : Gerçek değeri 0 olan sınıfın 0 olarak tahmin edilme sayısıdır. Yukarıdaki tabloda d ile gösterilmektedir.

True Negative (TN) : Gerçek değeri 1 olan sınıfın 1 olarak tahmin edilme sayısıdır. Yukarıdaki tabloda a ile gösterilmektedir.

False Positive (FP) : Gerçek değeri 0 olan sınıfın 1 olarak tahmin edilme sayısıdır. Yukarıdaki tabloda b ile gösterilmektedir.

False Negative (FN) : Gerçek değeri 1 olan sınıfın 0 olarak tahmin edilmesidir. Yukarıdaki tabloda c ile gösterilmektedir (Apache Spark 2.4.1 Evaluation, 2019).

Accuracy : Sınıflandırmada doğruluk oranıdır. Eş. 4.1'e göre hesaplanmaktadır

$$Accuracy = \frac{a + d}{a + b + c + d} \tag{4.1}$$

Error : Sınıflandırmada hata oranıdır. Eş. 4.2'ye göre hesaplanmaktadır. (Visa, Ramsey, Ralescu, ve Knaap, 2011)

$$Error = \frac{b + c}{a + b + c + d} \tag{4.2}$$

Precision (Positive Predictive Value): Doğru tahmin edilen pozitif sınıfların pozitif tahminlere oranıdır. Kesinlik olarak adlandırılabilir. Eş. 4.3'e göre hesaplanmaktadır.

$$PPV = \frac{TP}{TP + FP} \tag{4.3}$$

Recall (True Positive Rate): Doğru tahmin edilen pozitif sınıfların pozitif doğru tahmin ve yanlış negatif toplamına oranıdır. Duyarlılık olarak adlandırılabilir.

F-measure : Precision ve Recall in birlikte değerlendirildiği bir ölçüttür. İki ölçüt arasındaki dengeyi belirlemek için β değişkeni kullanılır. Eş. 4.4'e göre hesaplanmaktadır.

$$F(\beta) = (1 + \beta^2).\left(PPV.\frac{TPR}{\beta^2.PPV + TPR}\right) \quad (4.4)$$

Veri setinin 70%'i eğitim için 30% i test için kullanılmaktadır. Veriler yerine konulmadan rastgele olarak seçilmektedir.

Testlerin başarımı Apache Spark çatısında mevcut olan MulticlassClassificationEvaluator başarım değerlendirme aracı ile ölçülmüştür.

4.2.2. Veri ön işleme

1. "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri için tespit edilen sapan veri sınırları dışında kalan veriler özelliklerin ortanca değerleri ile değiştirilmiştir.
2. "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri için frekans aralıkları $2^k \geq N$ ve $\mathrm{W} = \frac{\mathrm{X_{max}} - \mathrm{X}_{min}}{\mathrm{k}}$ formülleri kullanılarak gruplandırılmıştır. Formüller ile sınıf aralıkları tespit edilmiştir. Sınıf aralıkları belirlendikten sonra gruplar 1'den başlayarak etiketlenmiştir. Buna göre:

"customer_period" verisi için:
Sınırlar: [-∞, 0, 1, 2, 3, 4, 5, 6, +∞]
Gruplar Çizelge 4.5'te gösterildiği şekilde oluşmuştur.

Çizelge 4.5. customer_period frekans dağılımı

customer_period	Frekans
1	18779
2	15782
3	13976
4	7291
5	4522
6	3360
7	6080
Toplam	69790

"avg_invoice" verisi için:
Sınırlar: [-∞, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56, 61, 66, 71, 76, 81, +∞]

Gruplar Çizelge 4.6’da gösterildiği şekilde oluşmuştur.

Çizelge 4.6. avg_invoice frekans dağılımı

avg_invoice	Frekans	avg_invoice	Frekans
1	209	9	8690
2	475	10	5782
3	1440	11	3675
4	4247	12	2572
5	6926	13	1543
6	7488	14	1207
7	10442	15	998
8	14096		
Toplam	69790		

"avg_download_in_gb" verisi için:

Sınırlar: [-∞, 0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, +∞]

Gruplar Çizelge 4.7’de gösterildiği şekilde oluşmuştur.

Çizelge 4.7. avg_download_in_gb frekans dağılımı

avg_download_in_gb	Frekans
1	12984
2	8175
3	7821
4	10308
5	6024
6	5059
7	4197
8	3469
9	2758
10	2206
11	1862
12	1447
13	1202
14	944
15	822
16	512
Toplam	69790

"avg_upload_in_gb" verisi için:

Sınırlar: [-∞, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, +∞]

Gruplar Çizelge 4.8’de gösterildiği şekilde oluşmuştur.

Çizelge 4.8. avg_upload_in_gb frekans dağılımı

avg_upload_in_gb	Frekans
1	15298
2	10844
3	13631
4	7439
5	5707
6	4482
7	3329
8	2668
9	1981
10	1558
11	1227
12	1626
Toplam	69790

4.2.3. Makine öğrenmesi algoritması

Tahmin edilmek istenen veri bir kullanıcının üyeliliğinin durumudur. Mevcut veri setinde aktif ve pasif olmak üzere iki durum vardır, ancak yeni durumların da desteklenmesi adına sorun bir sınıflandırma problemi olarak ele alınmıştır. (Örneğin gelecekte “eski üyenin yeniden üye olması” gibi durumların desteklenmesi gibi).

Apache Spark çatısında Random Forest sınıflandırma algoritması, Ayrıca model kurulurken gerçekleştirilen özellik seçimi (Embedded) yöntemini de destekleyen aşırı öğrenmeye karşı dayanıklı bir sınıflandırma algoritmasıdır.

Ölçümlerin ortak bir algoritma üzerinden yürütülmesi adına özelliklerinin uygun olması sebebiyle uygulama Random Forest algoritması ile gerçekleştirilecektir.

4.2.4. Özellik seçimi olmadan başarım

"tariff", "speed_in_mb", "quota_in_gb", "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri ile “status” özelliği için sınıflandırma modeli kurulduğunda başarım ölçümleri Çizelge 4.9’da gösterildiği şekilde oluşmuştur.

Çizelge 4.9. Özellik seçimi olmadan başarım ölçümleri

Başarım Kriteri	Başarım
F1- measure	0,86
Precision	0,88
Recall	0,87
Accuracy	0,87

4.2.5. Özellik seçimi ile başarım

Bu bölümde Filter, Embedded ve Wrapper metotlar ile özellik seçimi yapılarak başarımlar ölçülecektir.

Filtre metot ile özellik seçimi

Filter metotlardan Ki-kare ve FCBFS ile başarım ölçümü gerçekleştirilecektir. Ki-kare ve FCBFS algoritmaları korelasyon tabanlı olarak model kurmadan özellik seçimi yapmaktadır.

Ki-kare ile başarım

"tariff", "speed_in_mb", "quota_in_gb", "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri Ki-kare özellik seçme algoritmasına selectorType="fpr", fpr=0,05 parametreleri ile verilmiştir. 7 özellikten elenen bir özellik olmamıştır. Başarım ölçümleri aşağıdaki Çizelge 4.10'daki şekilde olmuştur:

Çizelge 4.10. Ki-kare ile başarım ölçümleri

Başarım Kriteri	Başarım
F1- measure	0,86
Precision	0,88
Recall	0,87
Accuracy	0,87

FCBFS ile başarım

"tariff", "speed_in_mb", "quota_in_gb", "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri FCBFS algoritmasına verilmiş, "quota_in_gb", "speed_in_mb", "avg_download_in_gb" olmak üzere üç özellik seçilmiştir. Başarım ölçümleri Çizelge 4.11'de gösterildiği şekilde oluşmuştur.

Çizelge 4.11. FCBFS le başarım ölçümleri

Başarım Kriteri	Başarım
F1- measure	0,87
Precision	0,88
Recall	0,87
Accuracy	0,87

FCBFS algoritması ile seçilen “quota_in_gb”, “speed_in_mb”, “avg_download_in_gb” özelikleri haricindeki seçilmemiş olan "tariff", "customer_period", "avg_invoice", "avg_upload_in_gb” özelliklerinin etkisinin görülmesi için başarımı ölçüldüğünde Çizelge 4.12’deki sonuçlar elde edilmiştir:

Çizelge 4.12. FCBFS algoritması ile seçilen özellikler haricindeki özellikler ile başarım

Başarım Kriteri	Başarım
F1- measure	0,78
Precision	0,78
Recall	0,79
Accuracy	0,79

Seçilen özelliklerin başarımını gösteren Çizelge 4.11 ile seçilmemiş özelliklerin başarımını gösteren Çizelge 4.12 karşılaştırıldığında seçilmiş olan üç özelliğin hedef değeri tahmin etmek için başarım kaybı yaşanmadan kullanılabileceği görülmektedir.

Gömülü metot ile özellik seçimi

"tariff", "speed_in_mb", "quota_in_gb", "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri Random Forest Algoritmasına verilmiş, model kurulurken özelliklere dağıtılan ağırlıklar Çizelge 4.13’teki gibi ölçülmüştür.

Çizelge 4.13. Random forest algoritması ile ölçülen özellik önem dereceleri

Özellik	Önem Derecesi
speed_in_mb	0,4705
quota_in_gb	0,4203
avg_download_in_gb	0,0707
customer_period	0,0204
avg_invoice	0,0135
avg_upload_in_gb	0,0044
tariff	0,0002
TOPLAM	1

Çizelge 4.13'teki veriler incelendiğinde Embedded Metot ile seçilecek özelliklerin seçiminde bir eşik değerinin belirlenmesi gerektiği görülmektedir. Bu eşik değeri veri seti için alan bilgisi gerekecektir. Çünkü özellikler önem derecesine göre sıralanmıştır, ancak ilk kaç özelliğin seçilmesi gerektiği konusunda bir soyut bir çözüm önerilmemektedir. Alan bilgisi kullanılmadan başarım artış veya azalışına dayalı bir yöntem uygulanmak istenildiğinde yani Sequential Forward Selection veya Sequential Backward Elimination yöntemi uygulandığında özellik seçimi için Wrapper Metoda geçilmiş olacaktır. Çizelge 4.14 ve Çizelge 4.15 incelendiğinde Sequential Forward Selection veya Sequential Backward Elimination yöntemleri için belirleyici olacak özelliklerin başarımlarının alabileceği değerler görülebilecektir. Sonuç olarak gömülü metodun tek model kurarak tüm özellikler için önem ağırlık dağılımı yaptığı ancak özelliklerin seçim kriteri konusunda ekstra çalışma gerektirdiği görülmüştür. Önem derecesine göre ilk üç özellik yani 50% lik dilim seçildiğinde Filter method ile elde edilen başarıma ulaşılmış olacaktır.

Sarmalayıcı metot ile başarım

"tariff", "speed_in_mb", "quota_in_gb", "customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb" özellikleri RandomForest sınıflandırma algoritmasına teker teker verilerek model kurulmuştur. Ölçülen başarım aşağıdaki Çizelge 4.14'te sunulmuştur.

Çizelge 4.14. Model kurularak gerçekleştirilen özellik seçiminde başarımlar

	tariff	speed_in_mb	quota_in_gb	customer_period	avg_invoice	avg_download_in_gb	avg_upload_in_gb
F1-measure	0,64	0,84	0,84	0,60	0,61	0,78	0,77
Precision	0,78	0,84	0,85	0,51	0,77	0,79	0,77
Recall	0,73	0,84	0,85	0,72	0,72	0,79	0,78
Accuracy	0,73	0,84	0,85	0,72	0,72	0,79	0,78

Çizelge 4.15. Özellik gruplarının Wrapper metot ile başarımları

Özellikler	F1-measure	Precision	Recall	Accuracy
"customer_period"	0,60	0,51	0,72	0,72
"customer_period", "avg_invoice"	0,65	0,70	0,73	0,73
"customer_period", "avg_invoice", "avg_download_in_gb"	0,78	0,79	0,80	0,80

Çizelge 4.15. (devam) Özellik gruplarının Wrapper metot ile başarımları

Özellikler	F1-measure	Precision	Recall	Accuracy
"customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb"	0,78	0,80	0,80	0,80
"customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb", "speed_in_mb"	0,86	0,86	0,87	0,87
"customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb", "speed_in_mb", "quota_in_gb"	0,86	0,87	0,87	0,87
"customer_period", "avg_invoice", "avg_download_in_gb", "avg_upload_in_gb", "speed_in_mb", "quota_in_gb", “tariff”	0,86	0,87	0,87	0,87

Çizelge 4.15’te tek özellik ile başlanmış ve özellikler eklenerek başarımlar ölçülmüştür. Burada özellik gruplarını her bir özellik için döngüsel olarak oluşturursak yöntem Genetik algoritma yaklaşımına doğru bir geçiş olacaktır. Algoritmanın zaman karmaşıklığı ise n! düzeyinde olacaktır. n! kadar makine öğrenmesi modeli oluşturulması büyük veri sistemlerinde tercih edilebilecek bir maliyet değildir. Elde edilen başarımlar da incelendiğinde Filtre metot ile elde edilen sonuçların mevcut veri seti açısından daha verimli olduğu görülmektedir.

4.2.6. PCA ile boyut indirgeme

Bu kısımda özellik seçimi ile boyut azaltmadan farklı olarak herhangi bir özellikten vazgeçilmeden özelliklerin boyutu bir sayıya indirgenmektedir.

Tamamı 7 olan özellikler Ki-kare ile seçildiğinde elenen özellik olmazken, FCBFS ile çalıştırıldığında “quota_in_gb”, “speed_in_mb”, “avg_download_in_gb” özellikleri seçilmiş ve başarım düşüşü olmadığı gözlemlenmiştir. PCA ile boyut indirgeme uygulamasında elde edilen minimum boyut sayısı 3 olduğundan 2 boyut üzerinden bir test gerçekleştirilecektir. 2 boyutlu bir özellik uzayı gerçek senaryolar için oldukça küçüktür.

Ancak 3 boyutlu bir uzayın indirgendiğindeki başarımının test edilebilmesi için 2 sayısı seçilmiştir. Ayrıca özellik seçimi olmadan direkt PCA ile 7 ‘den 2 boyuta indirgendiği durumda başarım da incelenmiştir. Tüm durumları gösteren başarımlar Çizelge 4.16’da görülmektedir.

Çizelge 4.16. PCA ile 3 özelliğin 2 boyuta indirgendiği durumda başarım ölçümleri

	Özellik Seçimi Olmadan Başarım	Seçilen 3 özellik İle Başarım	PCA İle 3’ten 2 Boyuta İndirgendiğinde Başarım	PCA İle 7’den 2 Boyuta İndirgendiğinde Başarım
F1-measure	0,86	0,87	0,85	0,82
Precision	0,88	0,88	0,86	0,83
Recall	0,87	0,87	0,86	0,83
Accuracy	0,87	0,87	0,86	0,83

Çizelge 4.16 incelendiğinde PCA boyut indirgemenin özellik seçiminden sonra uygulanmasının daha başarılı bir sonuç verdiği görülmektedir.

4.3. Farklı Bir Veri Setinde Uygulama

İnternet sağlayıcı verileri ile yapılan çalışmada Filtre Metotlar ile özellik seçimi ve PCA boyut indirgemesi denenmiş ve başarım ölçülmüştür. Bu yöntemin başka bir veri setinde de başarılı sonuçlar verip vermeyeceğinin kontrolü için açık kaynaklı (erişim bilgileri kaynakçada adresi mevcut) (BigML, 2019) olarak temin edilmiş olan “Churn in Telecom's dataset” verisi ile uygulama yapılmıştır. Veri, 3333 satır ve 21 kolondan oluşmaktadır. Özellik seçimi senaryosu için "account_length", "area_code", "international_plan", "voice_mail_plan", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_calls", "total_intl_charge", "customer_service_calls" olmak üzere 18 özellik "churn" hedef değerini tahmin etmek için kullanılmıştır. "area_code", "international_plan", "voice_mail_plan" verileri kategorik özellik, diğer veriler ise sayısal (sürekli ve kesikli) özellik olarak ele alınmıştır. “churn” hedef değişkeni ise iki değerlidir.

4.3.1. Veri ön işleme

Sürekli değişkenler internet sağlayıcı verilerinde uygulanan yöntem ile gruplanmıştır. Burada aynı koşullar ile ölçüm hedeflenmektedir.

4.3.2. Makine öğrenmesi algoritması

Bu kısımda internet sağlayıcı verileri ile yapılan, ön işleme yöntemleri ve seçilen algoritma internet sağlayıcı veri seti uygulamasında kullanılan yöntem ve kodlar ile aynıdır.

4.3.3. Özellik seçimi olmadan başarım

Özellik seçimi olmadan elde edilen başarım sonuçları Çizelge 4.17'de sunulmuştur.

Çizelge 4.17. Özellik seçimi olmadan başarım

Başarım Kriteri	Başarım
F1- measure	0,83
Precision	0,84
Recall	0,87
Accuracy	0,87

4.3.4. Özellik seçimi ile başarım

Özellik seçiminde amaca en uygun özelliklerin seçilerek ilişkisiz özelliklerin modelden ayrı tutulması hedeflenmektedir. Bu durumda başarımın ne olacağı gözlemlenecektir.

Filtre metot ile özellik seçimi

Bu bölümde model kurulmadan özellik seçimi yapıldığında başarımın nasıl olacağı gözlemlenecektir.

Ki-kare ile başarım

İnternet sağlayıcı veri setinden farklı olarak bu uygulamada Ki-kare ile 18 özellik içerisinden 10 adet özellik ("international_plan", "voice_mail_plan", "number_vmail_messages", "total_day_minutes", "total_day_charge", "total_eve_minutes", "total_eve_charge", "total_intl_minutes", "customer_service_calls", "total_intl_calls") seçilmiştir. Başarım Çizelge 4.18'de görülmektedir.

Çizelge 4.18. Ki-kare ile başarım

Başarım Kriteri	Başarım
F1- measure	0,85
Precision	0,86
Recall	0,88
Accuracy	0,88

FCBFS ile başarım

FCBFS algoritması ile 18 özellikten 9 özelliğin ("international_plan", "total_intl_calls", "voice_mail_plan", "area_code", "customer_service_calls", "total_eve_minutes", "total_intl_charge", "total_night_calls", "number_vmail_messages") seçildiği görülmüştür.

Başarım Çizelge 4.19'da verilmiştir:

Çizelge 4.19. FCBFS ile başarım

Başarım Kriteri	Başarım
F1- measure	0,83
Precision	0,89
Recall	0,87
Accuracy	0,87

Ki-kare ve FCBFS ile başarım

Ki-kare ile seçilmiş olan 10 özellik FCBFS algoritmasına verilerek çalıştırıldığında 10 özellik içerisinden 8 özellik ("international_plan", "total_intl_calls", "total_day_charge", "voice_mail_plan", "total_intl_minutes", "customer_service_calls", "total_eve_minutes", "total_eve_charge") seçilmiştir. Başarım aşağıdaki Çizelge 4.20'de gösterildiği şekilde oluşmuştur.

Çizelge 4.20. Ki-kare ve FCBFS ile başarım

Başarım Kriteri	Başarım
F1- measure	0,84
Precision	0,86
Recall	0,87
Accuracy	0,87

Gömülü metot ile özellik seçimi

Gömülü metotlarda algoritma tarafından model kurulurken özellik seçimi gerçekleştirilmektedir. Gömülü (Embedded) metot ile yapılan ölçümler Çizelge 4.21 de sunulmuştur.

Çizelge 4.21. Embedded metot başarım ölçümleri

Özellik	Önem Derecesi
total_day_charge	0,2151
international_plan	0,1575
voice_mail_plan	0,0927
total_intl_minutes	0,0862
total_day_minutes	0,0823
total_intl_calls	0,0689
total_eve_charge	0,0477
number_vmail_messages	0,0414
total_eve_minutes	0,0402
total_night_charge	0,0389
total_eve_calls	0,0308
total_night_minutes	0,0238
total_night_calls	0,0229
account_length	0,0213
customer_service_calls	0,0173
total_day_calls	0,0102
area_code	0,0020
total_intl_charge	0,0006
Toplam	1

18 özellik ile Random Forest modeli kurulduğunda algoritma tarafından özelik önem aralıkları Çizelge 4.21'de görülmektedir. Daha önce olduğu gibi özelliklerin önem ağırlıklarına göre ilk 50% lik kısmıyla bir model kurma istersek başarımlar Çizelge 4.22'deki şekilde olmaktadır.

Çizelge 4.22. Embedded metot başarım ölçümleri

Başarım Kriteri	Başarım
F1- measure	0,87
Precision	0,88
Recall	0,89
Accuracy	0,89

Çizelge 4.22'deki başarımların Filter metotlar ile yakalanan başarımlardan çok da farklı olmadığı gözlemlenmiştir.

Sarmalayıcı metot ile başarım

Sarmalayıcı metot ile boş bir özellik uzayına özellik eklenerek model kurma yöntemi kullanılmıştır.

Çizelge 4.23. Wrapper metot başarım ölçümleri

Özellikler	F1-measure	Precision	Recall	Accuracy
"account_length", "number_vmail_messages"	0,80	0,74	0,86	0,86
"account_length", "number_vmail_messages", "total_day_minutes"	0,83	0,88	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls"	0,81	0,81	0,86	0,86
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge"	0,86	0,88	0,89	0,89
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes"	0,86	0,89	0,89	0,89
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls"	0,85	0,87	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge"	0,84	0,88	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes"	0,86	0,89	0,89	0,89
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls"	0,84	0,86	0,87	0,87
"account_length", "number_vmail_messages","total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge"	0,83	0,88	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge",…	0,84	0,87	0,88	0,88

Çizelge 4.23. (devam) Wrapper metot başarım ölçümleri

Özellikler	F1-measure	Precision	Recall	Accuracy
…"total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes"	0,84	0,87	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_charge"	0,84	0,87	0,88	0,88
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_charge", "customer_service_calls"	0,82	0,87	0,87	0,87
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge","total_intl_minutes", "total_intl_charge", "customer_service_calls", "total_intl_calls"	0,82	0,87	0,87	0,87
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_charge", "customer_service_calls", "total_intl_calls", "area_code"	0,81	0,85	0,87	0,87
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_charge", "customer_service_calls", "total_intl_calls", "area_code", "international_plan"	0,86	0,89	0,89	0,89
"account_length", "number_vmail_messages", "total_day_minutes", "total_day_calls", "total_day_charge", "total_eve_minutes", "total_eve_calls", "total_eve_charge", "total_night_minutes", "total_night_calls", "total_night_charge", "total_intl_minutes", "total_intl_charge", "customer_service_calls",…	0,86	0,89	0,89	0,89

Çizelge 4.23. (devam) Wrapper metot başarım ölçümleri

Özellikler	F1-measure	Precision	Recall	Accuracy
…"total_intl_calls", "area_code", "international_plan"	0,86	0,89	0,89	0,89

Çizelge 4.23 incelendiğinde Wrapper metot ile elde edilen başarımlar ile Filter metot ile elde edilen başarımların bu veri seti için yakın olduğu görülmektedir. Burada Wrapper metot ile tüm özellik kombinasyonları için model kurma maliyetinin kabul edilmesini sağlayacak düzeyde bir başarım artışı gözlemlenmemiştir.

4.3.5. PCA ile boyut indirgeme

Ki-kare ve FCBFS ile seçilen 8 özellik PCA ile 5 boyuta indirgendiğinde başarım Çizelge 4.24'te gösterildiği şekilde elde edilmiştir.

Çizelge 4.24. PCA ile 8 özelliğin 5 boyuta indirgendiği durumda başarım

	PCA Olmadan Başarım	PCA ile Başarım
F1- measure	0,84	0,85
Precision	0,86	0,86
Recall	0,87	0,88
Accuracy	0,87	0,88

5. DEĞERLENDİRME VE SONUÇ

Veri kavramı literatürde veri, enformasyon ve bilgi olmak üzere üç farklı şekilde kullanılmaktadır. Veri, kaynağından elde edilen ham veriyi; enformasyon, verinin işlenmesi ile elde edilen sonuçları; bilgi ise işlenmiş veriden elde edilen sonuçların birbirleri ile ilişkilendirilip, çeşitli yöntemlerle analiz edilerek, geçmiş ve günümüzü aydınlatmakla birlikte geleceğe dair planlamalar yapabilmemizi sağlayan değerler olarak tanımlanabilir. Teknolojide yaşanan değişimler verinin miktarını, çeşitliliğini, akış hızını ön görülemez derecede değiştirirken, aynı zamanda bu veriyi işlemek ve veriden bilgi elde etmek isteyen her organizasyon için bir fırsat haline gelmiştir.

Veri konusunda meydana gelen değişimler, veri işleme ile ilgili teknoloji paradigmalarının da yeniden tanımlanmasına neden olmuştur. Bugün "Big Data" olarak tanımlanan büyük veri teknolojileri sıradan donanımlar üzerine kurulabilen, ölçeklenebilir, hata toleranslı, paralel işleme özellikli veri işleme sitemleri olarak bilişim dünyasında yerlerini almışlardır. Böylece veriden bilgi elde etme sürecinin alt yapı sorununa verimli bir çözüm bulunmuş ve makine öğrenmesi çalışmaları konusunda büyük bir ivme sağlanmıştır.

Veriden bilgi keşfinin kritik süreçlerinden birisi de veriden özellik seçme konusudur. Özellik seçme kavramı veri madenciliğinin süreçlerinden biri olmakla birlikte "Büyük Veri" çalışmalarında kritik bir öneme sahiptir.

Bu çalışmada Hadoop büyük veri sistemi ve Spark Makine Öğrenmesi çatısı kullanılarak Filter, Embedded ve Wrapper Metotları ile özellik seçimi uygulaması gerçekleştirilmiştir. Makine öğrenmesi algoritması olarak Random Forest kullanılmış; başarım "F1-measure", "Precision", "Recall", "Accuracy" kriterlerine göre ölçülmüştür. Veri seti ülkemizde internet sağlayıcı bir firmadan elde edilen 70000 satır ve 7 sütun olmak üzere 490000 kayıttan oluşmaktadır.

Ölçümler ve kullanılan yöntemler Çizelge 5.1'de sunulmuştur. Çizelge 5.1 incelendiğinde Filter metotların başarımının Wrapper ve Embedded metotların başarımından düşük olmadığı görülmektedir. Teoride Wrapper metotların Filter ve Embedded Metotlardan başarım olarak daha iyi sonuçlar verdiği bilinmektedir. Özellik seçimine alan bilgisinin dahil edilmesi her üç yöntem açısından önemli başarım artışlarını sağlayacağı bir gerçektir. Bu faktör haricinde Filter metotlarda korelasyon, bilgi kazanımı gibi hesaplamalar ile model kurulmadan bir özellik seçimi yapılmaktayken Wrapper Metotlarda Sequential Forward

Selection ve Sequential Backward Elimination yöntemleri ve genetik algoritma yaklaşımı yöntemleri ile döngüsel olarak model kurulması söz konusudur. Wrapper Metotların başarımının yüksek olmasını sağlayan da her senaryo için bir modelin kuruluyor olmasıdır. Ancak bu durum tüm maliyete daha özellik seçme anında katlanılması anlamına gelmektedir. Yüksek başarım öncelikli senaryolar için Wrapper Metotlar iyi bir seçenek iken performans öncelikli senaryolar için Filter ve Embedded metotlar öncelikli olarak değerlendirilmelidir. Mevcut uygulamada Filter metotların başarımı Wrapper ve Embedde Metotların başarımından daha düşük olmamıştır. Embedded Metotlarda bir makine öğrenmesi modeli kurulmakta ve verilen parametrelere göre özellik önem dereceleri belirlenerek özellik seçimi model kurulurken yapılarak önemsiz özellikler modelden çıkarılabilmektedir. Embedded Metotlar bu özelliği ile başarım açısından Filter Metotlardan avantajlı iken performans yönünden de Wrapper Metotlara göre avantajlıdır. Ancak Embdded Metotlarda parametrelerin belirlenmesi kritik bir öneme sahiptir. Çizelge 5.1'den görüleceği üzere mevcut özelliklerin en başarılı 50% si seçildiğinde mevcut senaryoda Filter Metotlara göre başarım açısından bir avantaj sağlamadığı görülmüştür. Bu bilgiler ışığında mevcut uygulamada Filter metotların tercih edilmesinin başarım ve performans açısından uygun olacağı görülmüştür.

Çizelge 5.1. Uygulama başarım sonuçlarının karşılaştırılması

	Özellik Seçimi Olmadan	Özellik Seçimi İle			
		Filter		Wrapper (En yüksek)	Embedded (İlk 50% özellik kriteri ile)
		Ki-kare	FCBFS		
F1-measure	0,86	0,86	0,87	0,86	0,87
Precision	0,88	0,88	0,88	0,87	0,88
Recall	0,87	0,87	0,87	0,87	0,87
Accuracy	0,87	0,87	0,87	0,87	0,87

Yukarıda anlatılan uygulama aynı yöntemler ile 3333 satır ve 18 sütun olmak üzere 59.994 kayıtlık bir telekomünikasyon veri setinde gerçekleştirilmiştir. Özellik seçimi olmadan 18 özellik ile başarım 83% – 87% düzeyinde ölçülmüştür. Ki-kare ile 18 özellik içerisinden 10 özellik seçilmiş ve başarım 85% – 88% düzeyinde ölçülmüştür. FCBFS ile 18 özellik içerisinden 9 özellik seçilmiş ve başarım 83% – 89% düzeyinde olmuştur. Bu kısımda Ki-kare ve FCBFS algoritmalarının sırayla birlikte çalıştırıldığına nasıl bir başarım ortaya çıkabileceğini gözlemlemek adına Ki-kare ile seçilen özellikler FCBFS ile birbirleri ile korelasyonları kontrol edilerek yapılan özellik seçiminde özellik sayısı 8'e düşürülmüş,

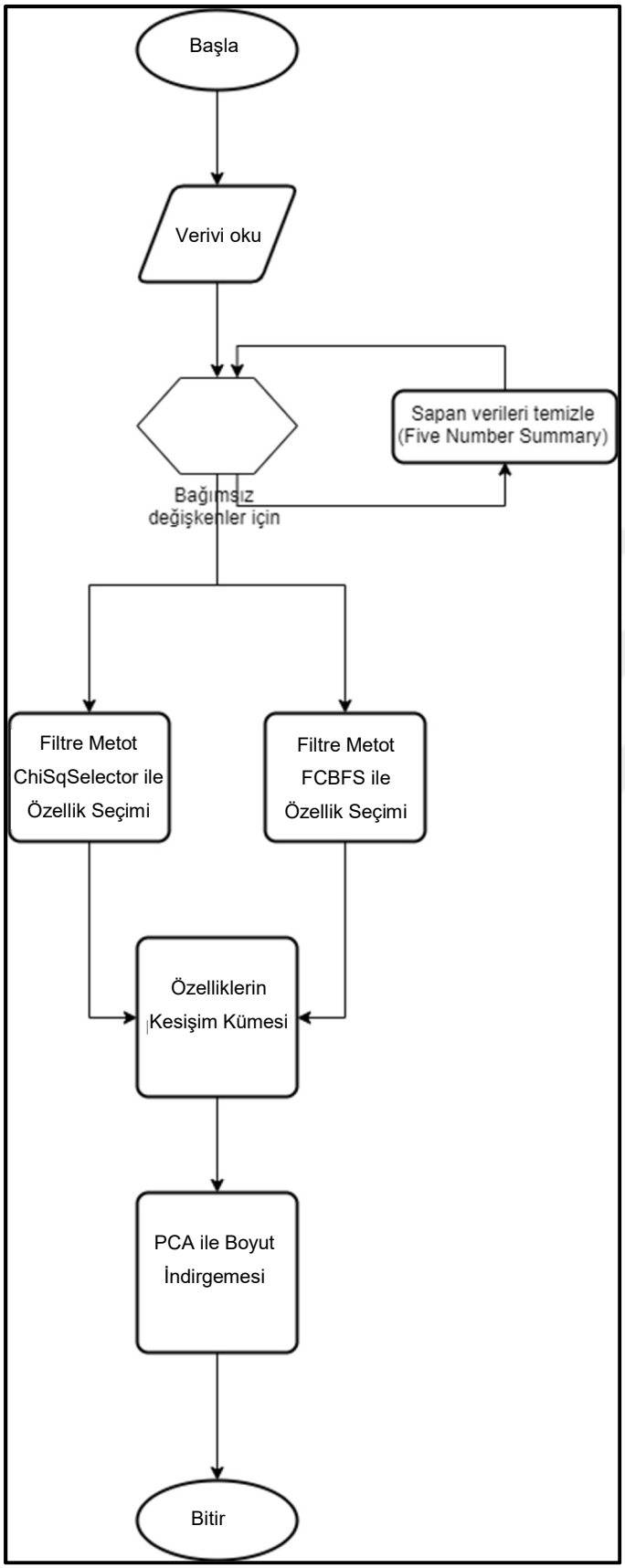
başarımın ise 84% – 87% düzeyinde olduğu görülmüştür. Seçilen bu 8 özellik PCA ile 5 boyuta indirgendiğinde ise 85% – 88% düzeyinde bir başarım ölçülmüştür.

Yapılan uygulamalarda performans öncelikli senaryolar için Apache Spark makine öğrenmesi kütüphanesi kullanılarak çalışmadaki veri yapısında Filter Metotlar ile başarımı yüksek özellik seçimlerinin yapılabileceği gözlemlenmiştir.

İnternet sağlayıcı verileri ile bir üyenin üyelik durumuna dair tahmin yapabilmek için "quota_in_gb", "speed_in_mb", "avg_download_in_gb" gerekli başarımı sağlamaktadır. Üretilecek makine öğrenmesine dayalı tahminleme yazılımlarında bu veri setinin yapısında Ki-kare ve FCBFS ile performanslı ve başarımı yüksek makine öğrenmesi modelleri kurulabilecektir. Şekil 5.1'de bu model şematize edilmiştir.

Diğer yandan bu çalışma makine öğrenmesi sürecini otomatize etme çalışmalarında Filter Metotların birçok senaryo için öncelikli olarak göz ününde bulundurulmasına dikkat çekmesi açısından bir gösterge olabileceği değerlendirilmektedir. Ayrıca her iki veri setinde de sürekli değerlerin gruplaması için soyut olarak tercih edilen yöntemin özellik seçimi için bir dezavantaj oluşturmadığı görüldüğünden bu sürecin otomatikleştirilmesi açısından de kullanışlı olabileceği değerlendirilmektedir.

Çalışmada kullanılan kodlar Apache Spark çatısı kullanılarak geliştirilmiştir. Veri madenciliği çalışmalarında bu çatı kullanılırken ek olarak çatı içerisinde var olmayan FCBFS algoritmasının gerçekleştirimi, sayısal verileri frekanslarına göre gruplara ayıran, sapan verileri ele alan yardımcı kütüphanelerin kodlanması yapılmıştır



Şekil 5.1 Özellik seçimi ve boyut indirgeme akış diyagramı

KAYNAKLAR

Ankam, V. (2016, Eylül). *Big Data Analytics.* Birmingham, United Kingdom: Packt Publishing.

İnternet: Apache Spark 2.4.1 Evaluation. URL: https://spark .apache.org/docs/2.4.1/mllib-evaluation-metrics.html, Son Erişim Tarihi: 06.03.2019.

İnternet: Apache Spark 2.4.1 MLLib. URL: https://spark.apache.org/docs/2.4.1/ml-statistics.html, Son Erişim Tarihi: 06.03.2019.

Aydıntan, B. (2009). *Örgüt Zekası ve Yönetimi.* Ankara: Gazi Kitabevi.

İnternet: BigML. URL: https://bigml.com/ user/francisco/ gallery /dataset/ 5163ad540c0b5e5b22000383, Son Erişim Tarihi: 11.05.2019

Byrnes, J. (2006). *Advances in Sensing with Security Applications.* Netherlands: Springer.

Deepa, T., and Ladha, L. (2011). Feature Selection Methods And Algorithms. *International Journal on Computer Science and Engineering*, 1787-1797.

İnternet: Deng, K. (1998, 11). Omega: On-Line Memory-Based General Purpose System Classifier. *Thesis*. U.S.A: The Robotics Institute School of Computer Science Carnegie Mellon University. URL: https://www.cs.cmu.edu/~kdeng/thesis/ thesis.pdf, Son Erişim Tarihi: 06.03.2019.

Doğan, E., ve Yörük, N. (2009). *Finansal Bilgi Manipülasyonu ve Finansal Bilgi Manipülasyonunun Belirlenmesine Yönelik İ.M.K.B'da Bir Uygulama.* Ankara: Detay Yayıncılık.

Doğan, K., ve Arslantekin, S. (2016). Büyük Veri: Önemi, Yapısı Ve Günümüzdeki Durum. *DTCF*, 15-36.

Ganelin, I., Orhian, E., Sasaki, K., and York, B. (2016). *Spark Big Data Cluster Computing in Production.* Indianapolis: John Wiley & Sons, Inc.

Han, J., and Kamber, M. (2006, Ocak). *Data Mining Concepts and Techniques.* San Francisco, United States of America. Elsevier Inc..

İnternet : İlter, H. Hadoop MapReduce Örnek Uygulama. URL: http://devveri.com /hadoop/hadoop-mapreduce-ornek-uygulama, Son Erişim Tarihi: 16.07.2019

İnternet: Koller, D., and Sahami, M. (1996). Toward Optimal Feature Selection. *International Conference on Machine Learning.* 2019 tarihinde URL: https://ai.stanford. edu/users/koller/Papers/Koller+Sahami:ICML96.pdf Son Erişim Tarihi: 16.07.2019

Lehner, F., and Maier, R. K. (2000, Kasım). How Can Organizational Memory Theories Contribute to Organizational Memory Systems? *Information Systems Frontiers (Vol 2)*,277-298.

Molina, L. C., Belanche, L., and Nebot, A. (2002, Aralık) Feature Selection Algorithms: A Survey and Experimental Evaluation. *IEEE International Conference on Data Mining* (s. 306-313). Maebashi City, Japan.

Monino, J. L., and Sedkaoui, S. (2016, Mart). *Big data, open data and data development.* United Kingdom : Wiley-Iste Publishing.

Perera, S., and Gunarathne, T. (2013, Ocak). *Hadoop MapReduce Cookbook.* Birmingham, United Kingdom : Pact Publishing.

Prajapati, V. (2013, Kasım). *Big Data Analytics with R and Hadoop.* Birmingham, United Kingdom: Packt Publishing.

Şeker, Ş. E. (2013,Temmuz). *İş Zekası ve Veri Madenciliği.* İstanbul: Cinius Yayınları.

Visa, S., Ramsey, B., Ralescu, A., and Knaap, E. v. (2011, Ocak). Confusion Matrix-based Feature Selection. *Midwest Artificial Intelligence and Cognitive Science Conference 2011* (s. 120 - 127). Cincinnati, United States of America

Xing, E. P., Jordan, M. I., and Karp, R. M. (2001, Haziran). Feature selection for high-dimensional genomic microarray data. *Proceedings of the 18th International Conference on Machine Learning* (s. 601-608). San Francisco, United States of America: Morgan Kaufmann Publishers Inc.

Yu, L., and Liu, H. (2003, Ocak). Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. *Proceedings of the Twentieth International Conference on Machine Learning.*

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: BEYAZIT, Burhan Erdoğdu
Uyruğu	: T.C
Doğum tarihi ve yeri	: 16/01/1988
Medeni hali	: Evli
Telefon	: +905325611688
Faks	:
e-posta	: erdogdubeyazit@gmail.com

Eğitim Derecesi	Okul/Program	Mezuniyet yılı
Yüksek lisans	Gazi Üniversitesi /Yönetim Bilişim Sistemleri	Devam Ediyor
Lisans	Hoca Ahmet Yesevi Üniversitesi/Bilgisayar Mühendisliği Bölümü	Devam Ediyor
Lisans	Polis Akademisi/Güvenlik Bilimleri Fakültesi	2010
Lise	Polis Koleji	2006

İş Deneyimi, Yıl	Çalıştığı Yer	Görev
2016 -	Türksat Uydu Haberleşme Kablo TV ve İşletme A.Ş.	Büyük Veri Uzmanı
2010 - 2016	Emniyet Genel Müdürlüğü	Yazılım Büro Amiri

Yabancı Dili

İngilizce

Yayınlar

1. Beyazıt, B. E., ve Gencer, C. T. (2019). Telekomünikasyon Verilerinde Hadoop ve Spark Teknolojileri İle Özellik Seçimi Uygulaması. VI. Uluslararası Fen, Mühendislik ve Mimarlık Bilimlerinde Akademik Çalışmalar Sempozyumu (s. 512 -522). Ankara: Asos Yayınevi.

Hobiler

Teknolojik gelişmeler, Doğa gezileri.

GAZİLİ OLMAK AYRICALIKTIR…