TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ MADENCİLİĞİ TEKNİKLERİNİ KULLANARAK SOSYAL AĞ TABANLI SINIFLANDIRICI GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Yunuscan KOÇAK

Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Tansel ÖZYER

AĞUSTOS 2016

Fen Bilimleri Enstitüsü Onayı

.....
Prof.Dr. Osman EROĞUL
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

.....
Doç.Dr. Oğuz ERGİN
Anabilimdalı Başkan V.

TOBB ETÜ, Fen Bilimleri Enstitüsü'nün 141111017 numaralı Yüksek Lisans öğrencisi **Yunuscan KOÇAK** 'nın ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"VERİ MADENCİLİĞİ TEKNİKLERİNİ KULLANARAK SOSYAL AĞ TABANLI SINIFLANDIRICI GELİŞTİRİLMESİ"** başlıklı tezi 09.08.2016 tarihinde aşağıda imzaları olan jüri tarafından kabul edilmiştir.

Tez Danışmanı:	**Doç.Dr. Tansel ÖZYER** TOBB Ekonomi ve Teknoloji Üniversitesi	.....
Jüri Üyeleri:	**Doç.Dr. Hüsrev Taha SENCAR (Başkan)** TOBB Ekonomi ve Teknoloji Üniversitesi	.....
	Yrd.Doç.Dr. Reza Zare HASSANPOUR Çankaya Üniversitesi	.....

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Yunuscan KOÇAK

ÖZET

Yüksek Lisans Tezi

VERİ MADENCİLİĞİ TEKNİKLERİNİ KULLANARAK SOSYAL AĞ TABANLI SINIFLANDIRICI GELİŞTİRİLMESİ

Yunuscan KOÇAK

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç.Dr. Tansel ÖZYER

Tarih: AĞUSTOS 2016

AIDS HIV'in sebep olduğu ölümcül bir hastalıktır. Bağışıklık sistemine saldıran bu hastalık beyaz kan hücreleri üstünde çoğalarak bütün vücuda yayılmaktadır. Hastalığın yaşam döngüsünde HIV-1 protaz enzimi tarafından kırılan amino asit sekizlileri virüs tarafından kendi proteinlerini oluşturmakta kullanılmaktadır. Bu doğrultuda hangi sekizlilerin virüs tarafından kırılabileceğini tahmin etmek yenilikçi ve başarılı ilaçlar geliştirilmesi açısından önem arz etmektedir. Bu çalışmada farklı alanlarda da uygulanabilecek yenilikçi bir sınıflandırıcı önerilmektedir. Bu sınıflandırıcı veri madenciliği tekniklerini kullanarak oluşturulan bir sosyal ağ üzerinde analizler yaparak yeni örneklerin sınıflarını tahmin etmekte kullanılmaktadır. İki ana kısımdan oluşan çalışmamızda ilk olarak sık öğe kümelerinin öznitelik olarak değerlendirilme süreci anlatılmış, ikinci kısımda ise bu öznitelikleri kullanan sınıflandırıcıyı geliştirirken kullandığımız yaklaşım ve sınıflandırıcının çalışma mekaniği açıklanmıştır. Sonuçlarımız literatürde önerilen yöntem ve diğer makine öğrenme yöntemleri ile karşılaştırılmıştır ve bu sonuçlar ümit vericidir.

Anahtar Kelimeler: Veri madenciliği, Makine öğrenmesi, Sosyal ağ analizi, Sık öğe kümesi

ABSTRACT

Master of Science

DEVELOPMENT OF A CLASSIFIER BASED ON SOCIAL NETWORK ANALYSIS USING DATA MINING TECHNIQUES

Yunuscan KOÇAK

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Doç.Dr. Tansel ÖZYER

Date: August 2016

AIDS is a deadly disease that is caused by HIV. HIV attacks the immune system of the body and uses white blood cells to make replicates of itself and spreads them to the everywhere in the body. In the life cycle of disease HIV-1 protease enzyme is in charge of cleaving an amino acid octamer into peptides which are used to create proteins by virus. It is very critical to induce a model and predict cleavage of HIV-1 protease on octamers for developing successful medicine. In this work, a novel classifier is proposed which can also be used in different domains. This classifier analyzes a social network that is created by using data mining techniques to predict the class values of new instances. This work consists of two main parts, in the first part evaluation process of frequent itemsets as features is discussed. In the second part, our approach on developing the classifier and the working mechanism of classifier is explained. Our results are compared with the methodology that is proposed on the technical literature and with other machine learning methods and results are promising.

Keywords: Data mining, Machine learning, Social network analysis, Frequent itemsets

TEŞEKKÜR

Çalışmalarımın her aşamasında bana yol gösteren, bilgi ve deneyimlerinden yararlandığım hocam Doç. Dr. Tansel Özyer'e, kıymetli tecrübelerinden faydalandığım Bilgisayar Mühendisliği Bölümü öğretim üyelerine, destekleriyle her zaman yanımda olan aileme ve arkadaşlarıma, lisansüstü eğitimim boyunca vermiş olduğu burslardan dolayı TOBB Ekonomi ve Teknoloji Üniversitesi'ne teşekkür ederim.

İÇİNDEKİLER

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

KISALTMALAR

AIDS	: Edinilmiş Bağışıklık Eksikliği Sendromu
HIV	: İnsan Bağışıklık Yetmezlik Virüsü
SÖKA-SNF	: Sık Öğe Kümesi Ağı Tabanlı Sınıflandırıcı
OK	: Ortagonal Kodlama
PCA	: Temel Bileşenler Analizi
UCI	: Irvine Kaliforniya Üniversitesi
SVM	: Destek Vektör Makinesi
SÖK-TÜM	: Hem kırılmış hem de kırılmamış örneklerden elde edilen sık öğe kümeleri
SÖK-EVET	: Sadece kırılmış örneklerden elde edilen sık öğe kümeleri
SÖK-HAYIR	: Sadece kırılmamış örneklerden elde edilen sık öğe kümeleri
DESTEK-m	: Sık öğe kümeleri için asgari eşik değeri
SÖK-MERKEZ	: Sık öğe kümelerinden P_1 veya P_1' pozisyonundaki öğeleri bulunduranların öznitelik olarak seçilmesi.

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
P_x, P'_x	: Amino asitin sekizli içindeki pozisyonu
S_x, S'_x	: Protaz enzimi karşılıkları

1. GİRİŞ

AIDS HIV(İnsan Bağışıklık Yetmezliği) virüsünün sebep olduğu ölümcül bir hastalıktır. Bu hastalık hastanın bağışıklık sistemine saldırarak bağışıklık sisteminin diğer hastalıklarla mücadele etmesine engel olur. Bu durum kişinin hastalığın ilerleyen safhalarında eğer tedavi edilmez ise farklı hastalıklar tarafından zarar görmesine hatta ölümüne sebep olur.

Dünya üzerinde 2015 yılında yapılan bir araştırmaya göre 36.7 milyon kişi HIV virüsü ile yaşamaktadır [27]. Yine aynı yıl içinde AIDS hastalığı 1.1 milyon insanın ölümüne sebebiyet vermiştir [27]. Maalesef şuan hastalığa çare olabilecek bir ilaç bulunmamaktadır ancak araştırmacılar HIV virüsünün aktifliğini azaltacak ilaç ve tedaviler üstünde çalışmaktadırlar [6, 19]. Başarılı ilaçların geliştirilmesi ancak virüsün insan vücudunda nasıl çoğaldığının ve yayıldığının tam olarak anlaşılması ile mümkündür.

Virüsün yaşam döngüsünde proteinin peptit bağları HIV-1 protaz enzimi ile kırılarak virüs tarafından kendini çoğaltma işleminde kullanılmaktadır. Geliştirilen inhibitörler bu protaz enziminin normalde etkileyeceği peptit gibi davranarak enzimi tıkamakta ve proteinlerin aynı enzim tarafından etkilenmesi engellenmektedir. Bu yüzden eğer protaz enzimi ile amino asitler arasındaki ilişki anlaşılabilirse hastalığa çare olabilecek etkili ilaçlar geliştirilebilir.

Sınıflandırma problemi olarak bakıldığında, amaç belirli pozisyonlarda verilen sekiz adet amino asitin enzim tarafından kırılıp kırılmayacağının tahmin edilmesidir. Terminolojide amino asitlerin bulunduğu kökler $P_4, P_3, P_2, P_1, P'_1, P'_2, P'_3, P'_4$ ile gösterilmekte protaz enzimi tarafındaki karşılıkları ise $S_4, S_3, S_2, S_1, S'_1, S'_2, S'_3, S'_4$ olarak tanımlanmaktadır. Bir anahtar-kilit mekanizmasına benzetirsek kökler ile enzimin oturması sonucu protein kırılım işlemi gerçekleşmektedir. Bu kırılma köklerin orta noktası olan P_1 ve P'_1 pozisyonunda gerçekleşmektedir.

Her kök bir amino asit bulundurur ve toplam olarak 20 adet farklı amino asit bulunmaktadır. Bu yüzden olası bütün amino asit kombinasyonları için 20^8 adet sıralama bulunmaktadır. Bu sıralamaları kodlamak için kullanılan geçerli bir yöntem ortogonal kodlamadır. Bu kodlama içinde her bir öznitelik bir kökü ve amino asiti temsil eder ve toplam öznitelik sayısı 160 (20×8) olacak şekilde kodlanır. Problemin sınıf değeri de ikiliden oluşmaktadır, kırılmanın olması veya olmaması.

Problemin ilginç doğasından dolayı birçok araştırmacı bu konu üstünde çalışmıştır. Son 5 yıl içinde yapılan çalışmalar Rögnvaldsson [34] tarafından derlenmiş ve geliştirilen yöntemler kıyaslanmıştır. Bu çalışmalar içinde markov zincirlerini [26], birçok sınıflandırıcı birleştirme yöntemlerini [24], öznitelik seçimi için çok katmanlı perceptron yöntemini [16], destek vektör makinelerinde kernel kullanarak boyut azaltmayı [20] kullanan yöntemler bulunmaktadır. Bazı araştırmacılar ayrıca veriyi kodlamak için farklı kodlama teknikleri önermişler, OETMAP [12] ve genetik programlamayı

kullanan GP kodlama teknikleri önerilmiş [24] ama yapılan kıyaslamalar sonucunda Rögnvaldsson [34] ortogonal kodlamanın problemi doğrusal ayrılabilen bir problem halinde sunduğunu, bu kodlama ile birlikte destek vektör makinelerinin kullanılmasının yeterli olduğunu, farklı kodlama tekniklerinin daha başarılı sonuçlar vermediğini nitelemiştir.

Tezimin konusu veri madenciliğinin bir alt dalı olan ilişki kurallarını çıkarmak için kullanılan sık öğe kümeleri ile sosyal ağ analizinin birleştiği bir sınıflandırıcı geliştirmektir. Bu doğrultuda sık öğe kümelerinin birbirleri ile olan ilişkilerinden bir sosyal ağ üretilmiş daha sonra bu ağ sosyal ağ analizi teknikleri kullanılarak incelenmiştir. Analiz sonunda ağ üzerinden edinilen özellikler yeni sekizlilerin enzim tarafından kırılıp kırılmayacağının tahmin edilmesinde kullanılmıştır. Bu işlem içinde sık öğe kümeleri bir örneğin kırılıp kırılmayacağına karar vermek için oy vermektedirler. Birbirinden farklı oy etkisine sahip sık öğe kümelerinin etki gücü ağ üzerindeki konumuna ve veri seti genelindeki duruşuna bakılarak formüle dökülmüştür.

Tezin diğer bölümleri öncelikle ilgili konulardaki gerekli bilgilerin verilmesi, bu sınıflandırıcının geliştirilmesinde önemli bir etkisi bulunan yine sık öğe kümelerinin öznitelik olarak kullanılmasını kapsayan çalışmayı, tezin ana konusu olan sınıflandırıcının nasıl geliştirildiğini ve yapılan deney sonuçlarını kapsamaktadır.

2. GEREKLİ BİLGİLER

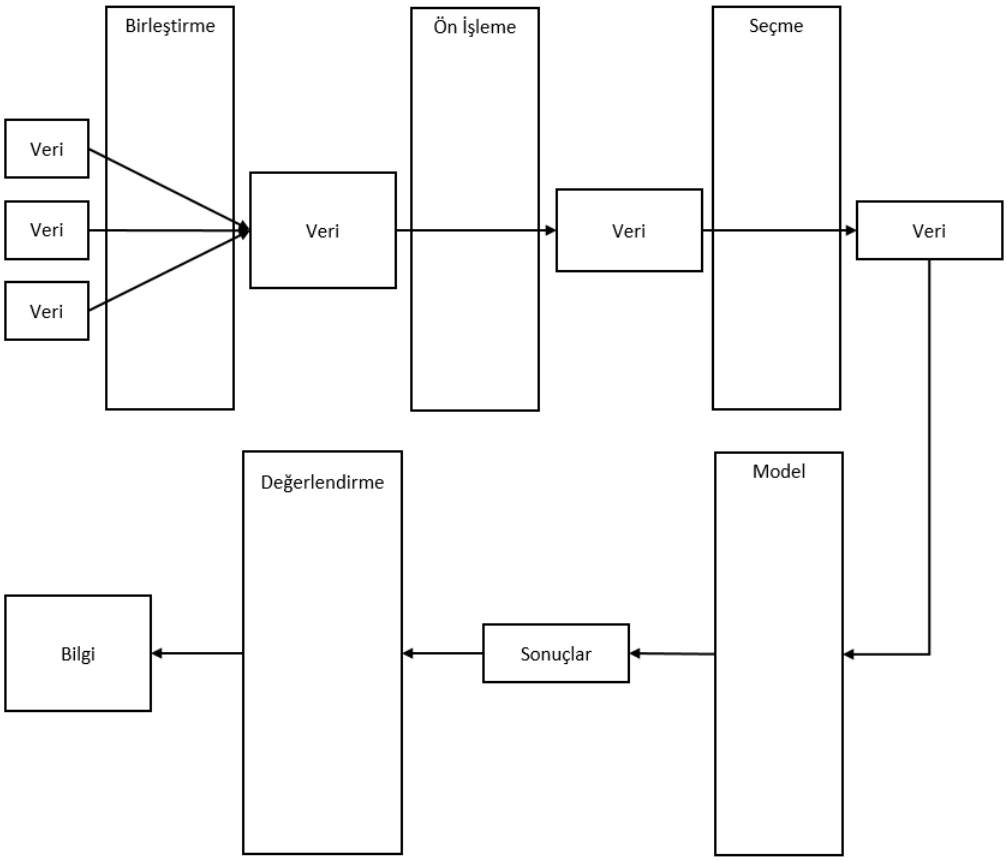
Bu bölümde SÖKA-SNF'nin geliştirilmesinde kullanılan ana yöntemler hakkında gerekli bilgiler verilmektedir.

2.1 Veri Madenciliği

Veri madenciliği belli bir boyuta sahip verinin bir amaç doğrultusunda incelenmesi ve içinden anlamlı bilgilerin çıkartılması işlemidir [7]. Bu amaçlar verilen girdilere bakılarak bir durumun tahmin edilmesi (sınıflandırma), belli bir özellikten dolayı birbirine benzeyen alt grupların bulunması (kümeleme) veya girdilerin birbirleri ile olan ilişkilerinin incelenmesi (ilişki kuralları çıkarma) olabilir [7]. Bu amaçları gerçekleştirme doğrultusunda veri madenciliği istatistik, bilgisayar bilimleri ve üzerine çalışan alan ile ilgili bilgilerin kullanıldığı çok disiplinli bir problem çözme işlemi olarak da tanımlanabilir.

Veri madenciliği veri tabanlarında bilgi keşfi adı verilen bir süreç içerisinde yapılır [8]. Bu süreç farklı kaynaklardan alınan bilgilerin tek bir yerde birleştirilmesi, veri ön işleme adı verilen veri üstünde ölçüm hatası veya kayıt eksikliklerinden dolayı oluşan hataların giderilmesi ve verinin analiz edilmek istenen şekle sokulması, veri içinden önemli kısımların belirlenmesi ve bunların analizde kullanılmak üzere seçilmesi, amaca uygun işlemleri yapacak modellerin tasarlanması ve veri ile birleştirilerek çalıştırılması ve bunların başarımlarının incelenmesi, gerekli raporların oluşturulması gibi alt adımlardan oluşmaktadır. Bu süreç görsel olarak Şekil 2.1'de sunulmuştur.

Sınıflandırma verilen girdileri sahip oldukları özniteliklere bakarak birbirlerinden farklı sınıflara ayırma işlemidir [2]. Bu problem iki boyutlu düzlem üzerinde şekildeki gibi gösterilebilir.

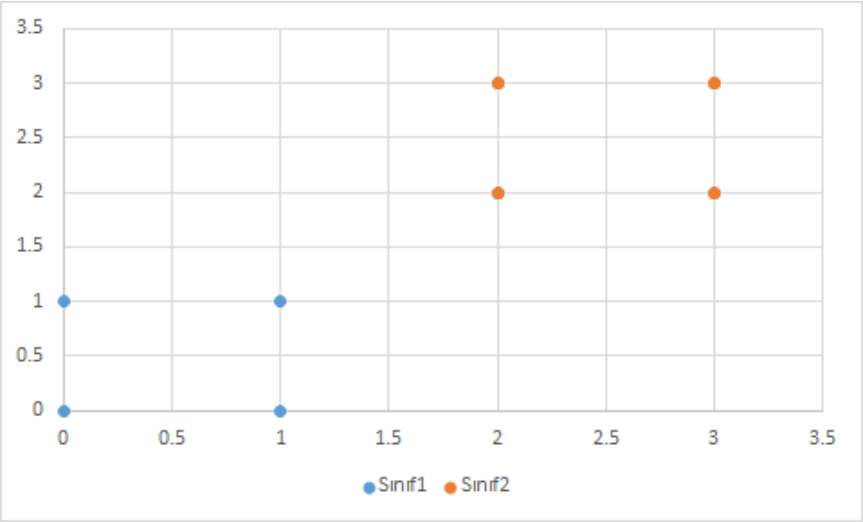


Şekil 2.1: Veri Tabanlarında Bilgi Keşfi: Farklı kaynaklardan alınan bilgiler ön işleme ve seçme aşamalarından geçirildikten sonra geliştirilen modellerde test edilir ve yapılan değerlendirmeler sonucunda bilgiye ulaşılır.

Şekil 2.2'de sunulan örnek problemde iki adet öznitelik bulunmaktadır. Eğitim veri setindeki örnekler X ve Y değerlerine göre noktalar koordinat düzlemine yerleştirilmiştir. Noktaların sahip olduğu sınıf değerleri turuncu ve mavi renkler ile gösterilmiştir. Bu problem için amacımız yeni gelen bir noktanın hangi renkte olacağını tahmin etmektir. Görselleştirme üzerinde de görüldüğü gibi mavi noktalar koordinat düzleminde sol alt kısımda yer almakta, turuncu noktalar ise sağ üst bölgede yogunlaşmaktadır. Bu bilgiyi kullanarak köşeden köşeye bir doğru çizip, yeni gelecek noktanın bu doğrunun hangi yüzünde kaldığına bakılarak bir tahmin işlemi gerçekleştirilebilir. Eğer yeni nokta doğrunun altında kalırsa sınıfı mavi, üstünde kalırsa turuncu olarak tahmin edilecektir. Bu işlem Şekil 2.3'te görselleştirilmiştir.

Sınıflandırma problemi için literatürde geliştirilmiş birçok yöntem bulunmaktadır. Bunlardan bazıları destek vektör makineleri [14], karar ağaçları [33] ve k-en yakın komşu [32] yöntemi olarak sıralanabilir.

Geometri tabanlı bir sınıflandırıcı olan destek vektör makineleri eğitim verisindeki noktaları inceleyerek bunları doğrusal bir biçimde ayıracak düzlemi bulur. Daha sonra bu düzlem yeni örnekleri sınıflandırmada kullanılır. Bu işlem Şekil 2.4'te görselleştirilmiştir. Ancak her problem doğrusal olarak ayrıştırılamamaktadır. Bu gibi problemler için destek vektör makineleri, noktaları bir üst düzleme taşıyarak bu düzlem içinde



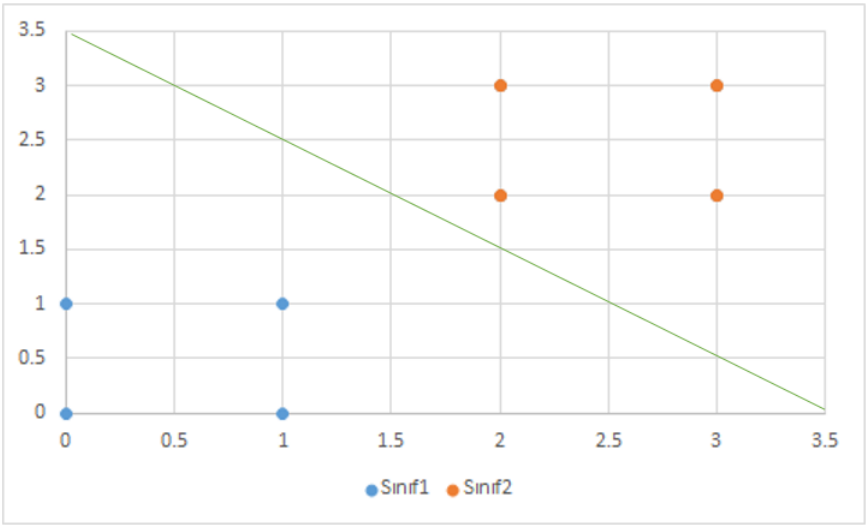
Şekil 2.2: Sınıflandırma Problemi. İki boyutlu düzlem üzerindeki noktaların sınıfları turuncu ve mavi renkler ile gösterilmiştir.

doğrusal olarak ayrıştırılabilmelerini sağlamaktadır. Noktaları taşıma işlemi tanımlanan kernel fonksiyonları aracılığıyla yapılır. Bu sayede doğrusal olmayan problemlerin de başarılı bir şekilde sınıflandırılması sağlanmaktadır.

Karar ağaçları verilen eğitim verisini inceleyerek bir örneği sınıflandırmada kullanılacak bir karar akış şeması oluşturur. Bu şema bir ağaç veri yapısı şeklinde olduğu için karar ağacı adı verilir. Karar ağacı üstünde yukarıdan aşağıya doğru ilerlerken, her seviyede bir özniteliğin durumuna bağlı olarak karar verilir. Karar ağacının yapraklarında ise sınıf değerleri bulunur. Verilen bir örnek için karar ağacında bir yaprağa ulaşmak, o örneğin sınıfının yaprakta bulunan sınıf olarak tahmin edildiği anlamına gelmektedir. Karar ağacının tasarımında özniteliklerin yukarıda aşağıya hangi sırada sıralanacağı önemlidir. Örneğin sınıfı hakkında karar vermemizi kolaylaştıracak belirtici öznitelikler karar ağacının üst dallarında yer almalıdır. Bunun için ağacın kökünden başlayarak özniteliklerin gini katsayısı [9] veya entropi değerleri [33] hesaplanır ve en yüksek bilgi kazanım değerine sahip öznitelik o dala atanır. Bu dalın altındaki dala atanacak öznitelik ise eğitim verisi içinde dalın sahip olduğu değeri içeren örnekler üstünde aynı işlem tekrarlanarak yapılır. Eğer bir dalın altındaki örneklerin hepsi aynı sınıf değerine sahip ise yaprağın değeri sınıf değeri olur. Öznitelik atama işlemi özyineli olarak tekrarlanır ve bu şekilde karar ağacı oluşturulmuş olur.

Örneğin bir üniversitede bulunan öğrencilerin öğlen yemeğini üniversite içindeki yemekhanede mi yoksa alışveriş merkezinde mi yiyeceğini belirten karar ağacı Şekil 2.5'teki gibidir. Karar ağacına bakıldığında yemeğin yemekhanede yenmesini sağlayan en belirtici öznitelik karar ağacının kök düğümünde bulunan bir sonraki derse kalan süredir. Eğer bu süre 45 dakikadan az ise diğer özniteliklere bakılmasına gerek kalmadan örneğin sınıfı tahmin edilir. Eğer bu süre 45 dakikadan çok ise bir sonraki özniteliğe, arkadaşlar ile birlikte yenilme durumuna bakılır. Verilen herhangi bir örnek için bu ağaç takip edilerek öğrencilerin yemeği nerede yiyecekleri tahmin edilebilir.

K-en yakın komşu yöntemi, tahmin edilecek örnekleri eğitim kümesindeki örnekler ile olan yakınlığına bakarak tahmin işlemini yapmayı amaçlayan bir yöntemdir. Bu



Şekil 2.3: Sınıflandırma probleminin çözümü. Sınıflandırılacak noktalar arasına çekilen bir doğrunun altında veya üstünde kalmalarına bakılarak yeni noktalar tahmin edilebilir. Eğer yeni nokta doğrunun altında kalırsa sınıfı mavi, üstünde kalırsa turuncu olarak tahmin edilecektir.

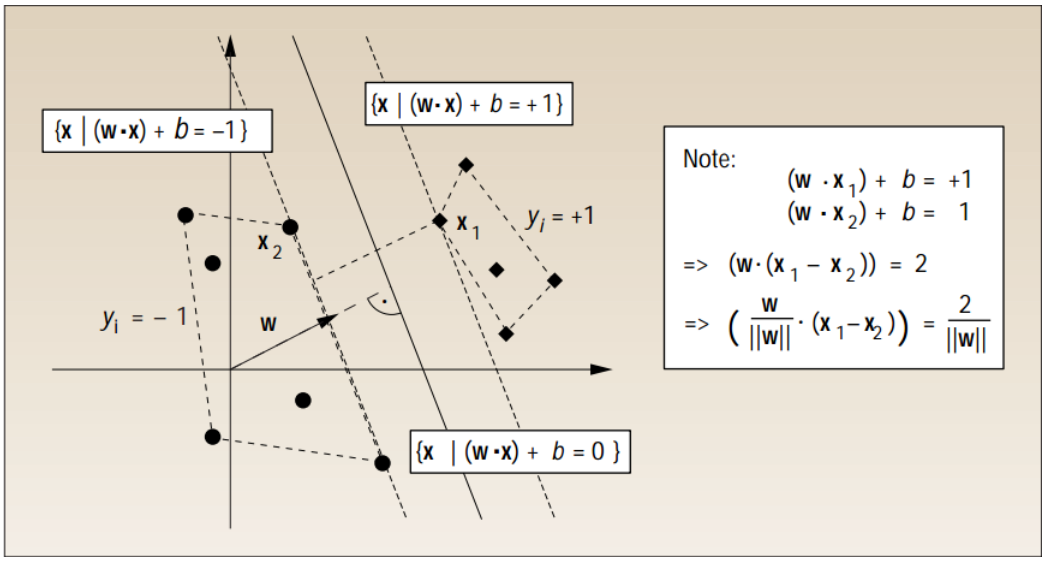
yöntemde örneğin sınıf değeri kendisine en yakın olan K örneğin sınıf değeri olarak tahmin edilir. Örnekler arasındaki yakınlık Öklit uzaklığı, Manhattan uzaklığı veya genel olarak Minkowski uzaklığı kullanılarak hesaplanabilir. Belirlenen K değeri küçük bir değer olması halinde eğitim setine çok bağlı kalacak ufak girdi değişikliklerinde tahmin çok oynayacaktır, çok büyük olması halinde ise her tahmin için veri setinin ortalama değerini dönecektir. Bu yüzden K değeri bu varyans ve sapma dengesinin iyi ayarlandığı bir değer seçilmelidir.

Şekil 2.6'daki örnekte siyah nokta K=3 değeri için sınıflandırılırken Manhattan uzaklığına bakıldığında kendisine en yakın 3 nokta (1,1), (2,2), (2,3) noktalarıdır. Bu noktalardan ikisi turuncu biri mavi olduğundan siyah noktanın sınıf değeri turuncu olarak tahmin edilir.

2.2 İlişki Kuralları Çıkarma

Veri madenciliğinde kullanılan önemli yöntemlerden biri de verinin sahip olduğu karakteristiğin anlaşılması için problemi oluşturan öznitelikler arasındaki örüntülerin bulunmasıdır. İlişki kuralları çıkarma olarak geçen bu işlem veri kümesindeki özniteliklerin örnekler içinde ne kadar beraber bulunduklarını hesaplayarak bu alt kümeler arasındaki ilişkileri gözler önüne serer [22].

Verilen bir öğe kümesine bakarak, birbirleri ile ilişkili olan öğeler farklı alt kümelere ayrılabilir. Bu alt kümeler içindeki ilişkilerin incelenmesi, öğeler arasındaki örtülü olan bilginin açığa çıkarılmasını sağlar. Bu şekilde örnekler içinde belirli bir sayıda görülen öğe kümeleri sık öğe kümesini oluşturur. Örnek olarak bir market zincirinde yapılan her bir alışverişte birbirinde farklı birçok ürün beraber satın alınmaktadır. Birçok kişinin yapmış olduğu alışverişlerin incelenmesi hangi ürünlerin daha sıklıkla beraber satın alındığı hakkında bir fikir verebilir. Bu amaçla yapılan tüm alışverişler içinde



Şekil 2.4: Destek vektör makineleri ile sınıflandırma. İki sınıf arasındaki ayırıcı düzlem sınıfların dış noktalarının oluşturduğu dışbükeyleri birleştiren doğruya dik olan doğru olarak belirlenir [14].

beraber geçen ürün kümelerinin bulunması sık öğe kümelerinin bulunması anlamına gelmektedir. Bulunan bu ürün kümeleri içinde birbirinden ayrık olacak şekilde ayrılan alt kümeler arasında bir alt küme sebebi temsil edecek diğeri sonuç olacak şekilde tasarlanan sebep-sonuç ilişkilerini bulmak mümkündür.

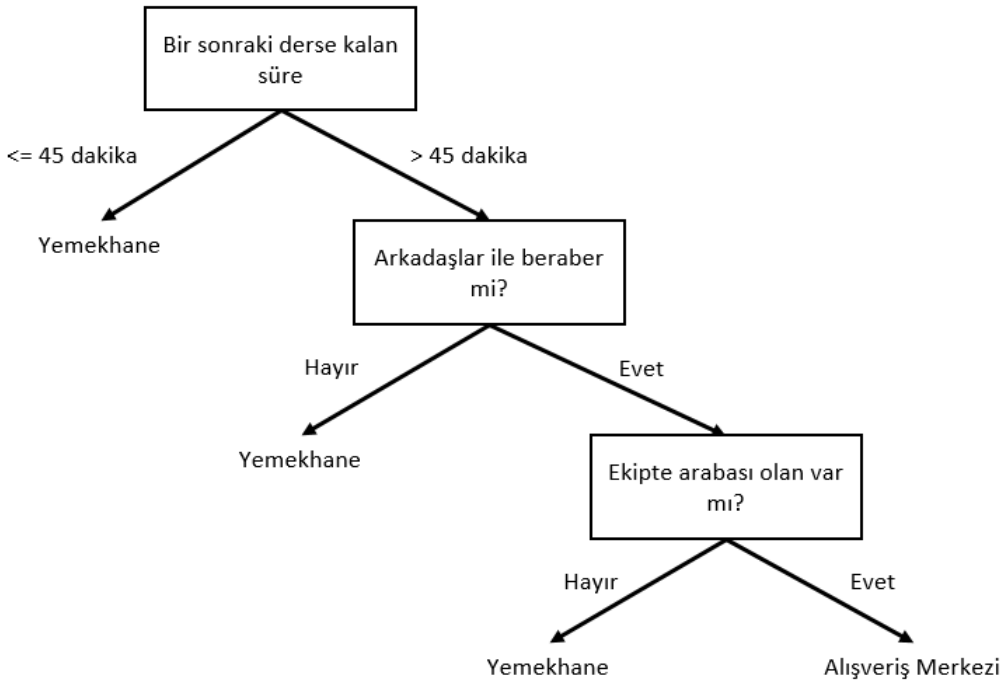
İlişki kuralları çıkarma literatürde üzerine çalışılmış bir konudur [1, 22]. Sık öğe kümeleri veri setinin özündeki yapıyı yakalamak için kullanılmaktadır. Formal bir dilde tanımlayacak olursak, $T = t_1, t_2, , .., t_n$ şeklinde n adet işlemden oluşan bir veri seti içinde, her işlem $t_i = \{I_{i1}, I_{i2}, ..., I_{ik}\}$ bütün öğelerden oluşabilecek $I_{ij} \in I$ şeklindeki kümelerden oluşmaktadır. I içindeki öğelerden oluşan bir öğe kümesi IS eğer daha önceden belirlenmiş bir destek eşik değerinden daha çok işlem içinde görülüyorsa sık olarak adlandırılır. Verilen bir öğe kümesinden oluşturulacak ilişki kuralları formal olarak $X \rightarrow Y$ öyle ki $X \bigcup Y = F, X \neq \phi, Y \neq \phi$ and $X \bigcap Y = \phi$ şeklinde tanımlanmaktadır. Öğe F işlemlerin ne kadarının alt kümesi olarak görüldüğünün oranı olarak belirtilen destek değeri ile nitelendirilmiştir. $X \rightarrow Y$ şeklinde tanımlanan bir ilişki kuralı değeri $X \bigcup Y$ destek değerinin X'in destek değerine bölünmesi şeklinde hesaplanan bir güven değerine sahiptir. İlişki kurallarının çıkarılmasında asgari destek eşik değeri ve asgari güven eşik değeri kullanılarak bu eşik değerlerinin üstünde kalan kurallar çıkartılır. Formal olarak öğe F'in destek değeri, |T| toplam işlem sayısını belirtmek üzere

$$destek(F) = \frac{|F \bigcap t_i \neq \phi|}{|T|}$$

şeklinde tanımlanır. Bir öğe setinin sıklığı ancak ve ancak

$$sık(F) = F \subseteq I \wedge \text{destek}\,(F) \geq asgari\ destek$$

durumunda mümkündür. Bir ilişki kuralı ancak asgari güven eşik değerini geçiyor ise önemlidir. s fonksiyonu sık olmayı belirtmek üzere güven değerinin formülü



Şekil 2.5: Karar ağacı ile sınıflandırma. Eğitim kümesindeki verilerden oluşturulan karar ağacı üstünde yeni örneğin özellikleri takip edilerek sınıflandırma yapılır.

$$güven(X \rightarrow Y) = \frac{s(F)}{s(X)}$$

şeklinde tanımlanır.

Sık öğe kümeleri kapalı sık öğe kümesi [29] ve azami sık öğe kümesi [23] olmak üzere 3 sınıfta incelenirler. Bir sık öğe kümesi eğer hiçbir üst kümesi kendi destek değerine sahip değilse buna kapalı sık öğe kümesi denir. Formal olarak,

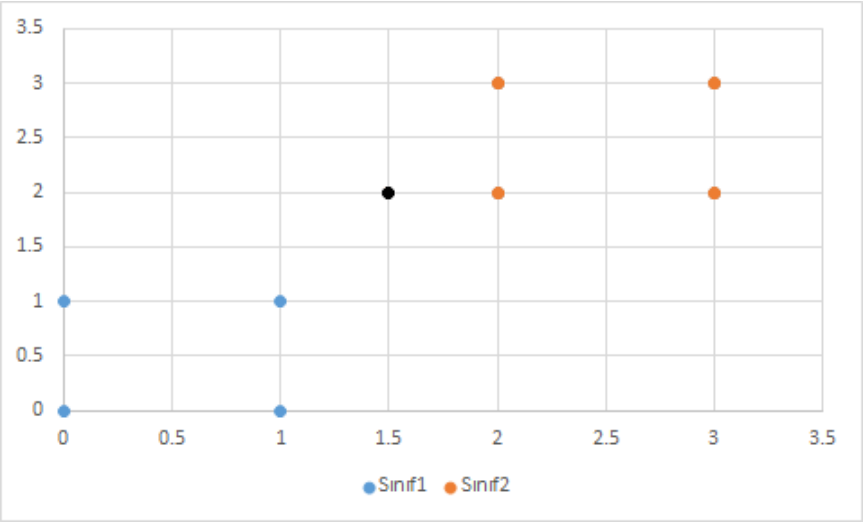
$$kapalıSık(F) = sık(F)\ \forall Z((Z \supset IS) \wedge (destek(F) \neq destek(Z)))$$

şeklinde gösterilir. Bir sık öğe kümesi eğer hiçbir üst kümesi sık değil ise azami sık öğe kümesi olarak adlandırılır. Formal gösterimde,

$$azamiSık(F) = sık(F)\ \forall Z((Z \supset F) \wedge (\neg\ sık(Z)))$$

Bu üç sık öğe kümesi içindeki sıralama $sık \supset kapalıSık \supset azamiSık$ şeklindedir (Şekil 2.7). Azami sık öğe kümeleri kapalı sık öğe kümelerinin bir alt kümesidir ve daha öz bilgi sunar. Kapalı sık öğe kümeleri ise sık öğelere kıyasla daha az fakat daha önemli öğeleri barındırır.

Sık öğe kümelerinin çıkarılması için literatürde birçok yöntem bulunmaktadır. Sık öğe kümelerini hızlı hesaplamak için bu yöntemler temel olarak bir kümenin elemanlarından biri sık değilse, o küme sık olamaz bilgisini kullanırlar. Apriori ve Fp-growth [13] algoritmaları bu yöntemlere örnektir.



Şekil 2.6: K-en yakın komşu yöntemi ile sınıflandırma. Örnek noktanın kendisine en yakın olan K nokta baz alınarak sınıf değeri tahmin edilir.

2.3 Sosyal Ağ Analizi

Sosyal ağ analizi, bir ağ olarak tanımlanabilen grupların içindeki aktörlerin birbirleri ile olan ilişkilerini belirli bir amaç doğrultusunda inceleme işlemidir. Bu analiz içinde ağ, çizge veri yapısı ile gösterilirken ağ üzerindeki aktörler düğümlerle, aktörler arasındaki ilişkiler kenarlar ile gösterilir. Bir ağ yapısı alt gruplarının incelenmesi amacıyla topluluklara bölünebilir, düğümler hangilerinin daha merkezde kaldığını bulmak amacıyla incelenebilir [36]. Bu şekilde incelenen alana bağlı olarak bulunan sonuçlar incelenir ve karar verme sürecine katkı sağlanmış olur.

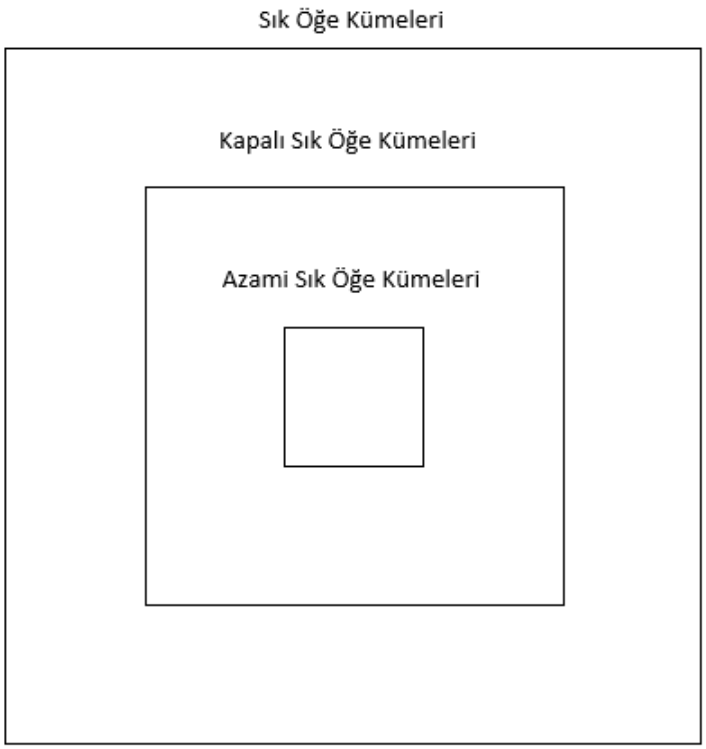
Sosyal ağ analizi içinde düğümler incelenirken kullanılan çeşitli merkeziyet metrikleri mevcuttur. Bu metriklerin ağın türüne (yönlü/yönsüz) veya kenarların ağırlıklı olup olmamasına göre farklı varyasyonları bulunmaktadır.

Derece merkeziyet değeri bir düğümün sahip olduğu komşu sayısını baz alan bir merkeziyet metriğidir. Bu metrik yönlü çizge yapısında iç-derece ve dış derece olmak üzere iki alt metriğe ayrılır. İç-derece merkeziyet değeri belirlenen düğüme gelen kenar sayısını, dış-derece metriği ise düğümden dışa giden kenar sayısının hesaplanmasıdır. Merkeziyet değerinin hesaplanmasına yönelik bir örnek Şekil 2.8'te verilmiştir.

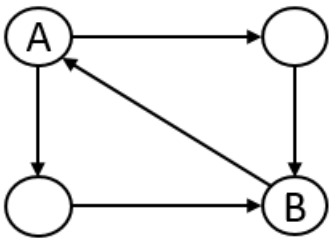
Arasındalık merkeziyet değeri bir düğümün diğer iki düğüm arasındaki en kısa yol içinde köprü olduğu durumların sayısıdır [10]. Bu durum çizge üzerindeki bütün kısa yollar için hesaplanarak bir düğümün arasındalık merkeziyet değeri bulunur. *yol* fonksiyonu iki düğüm arasındaki en kısa yolu belirtmek üzere arasındalık merkezinin formülü

$$A_m(v) = \sum_{s \neq t \neq v \in V} \frac{|yol_{st}(v)|}{|yol_{st}|}$$

şeklinde gösterilir.



Şekil 2.7: Sık, kapalı sık ve azami sık öğe kümeleri arasındaki ilişki. İçteki kümeler dıştakine göre sayıca daha az öğe barındırır ve daha öz bilgi sunar.



Şekil 2.8: Derece merkeziyet değeri. A ve B düğümlerinin derece merkeziyet değeri 3 olmakla birlikte A düğümünün iç-derece merkeziyet değeri 1, dış-derece merkeziyet değeri 2, B düğümünün ise iç-derece merkeziyet değeri 2 dış-derece merkeziyet değeri 1'dir.

Pagerank [28] algoritması Google'ın kendi arama motorlarında hangi sitelerin daha önemli olduğunu bulmak için kullandıkları bir algoritmadır. Pagerank algoritmasının hesapladığı skor internet üstünde herhangi bir siteden başlayan ve linkleri takip eden birisinin o siteyi gezme olasılığıdır [5]. Bu yöntemde sitelerin birbirlerine vermiş oldukları linkler yönlü bir çizge yapısına aktarılır ve hesaplama çizge yapısı üstünde yapılır.

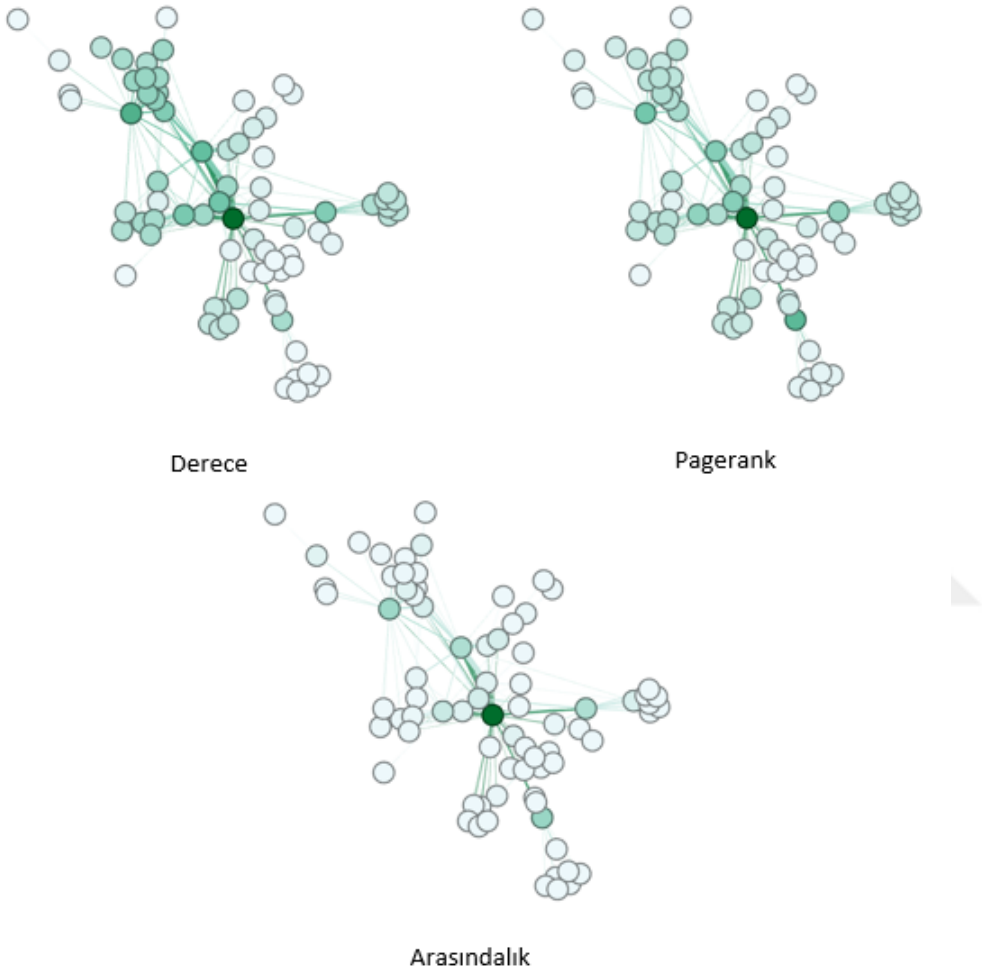
Pagerank değeri N fonksiyonu verilen bir düğümün iç komşularını, D fonksiyonu dış-derece değeri ve d sönüm değerini ifade etmek üzere

$$PR(v) = \frac{1-d}{N} + d \sum_{u \in N(v)} \frac{PR(u)}{D(u)}$$

iteratif bir şekilde hesaplanır ve ağ üzerinde bir değişim gözlenmeyene kadar devam ettirilir.

Çizge içinde bir düğümün pagerank skorunun yüksek olması için iki ihtimal vardır. Ya birçok farklı düğüm bu düğümü göstermelidir veya yüksek bir pagerank skoruna sahip bir düğüm bu düğümü işaret etmelidir.

Merkeziyet metrikleri 2.9 şeklinde Victor Hugo'nun Sefiller adlı romanındaki karakterlerin karşılaşma sıklığının ağı üstünde görselleştirilmiştir. Görselleştirme üstünde koyu renk daha yüksek bir merkeziyet değerinin hesaplandığı anlamına gelmektedir. Üç merkeziyet metriğinde de en yüksek skoru alan koyu yeşil düğüm hikayenin kahramanı Jean Valjean'ı göstermektedir. Derece ve pagerank merkeziyet skorlarında düğümler arasında biraz daha dengeli bir dağılım gözlenirken arasındalık merkeziyet skoru daha keskin bir dağılım göstermektedir.



Şekil 2.9: Derece, pagerank ve arasındalık merkeziyet metriklerinin Victor Hugo'nun Sefiller adlı romanındaki karakterlerin karşılaşma sıklığının ağı üstünde renklendirilmesi. Veri [17] çalışmasından alınmıştır.

3. ÖZNİTELİK OLARAK SIK ÖĞE KÜMELERİ

Yöntemimizin önemli parçalarından biri sık öğe kümelerinin öznitelik olarak kullanılmasından oluşmaktadır. Sık öğe kümelerinin bizim kırılma problemimiz için faydalı bir öznitelik olup olamayacakları bu bölümdeki çalışmalar sonucunda kararlaştırılmıştır.

Literatür araştırması sonucu, ikili sınıflandırma problemine giren protein kırılma probleminin özniteliklerinin ortogonal kodlama şeklinde kodlanması ve sınıflandırmanın destek vektör makineleri sınıflandırıcısı kullanılarak yapılmasının en başarılı yöntem olduğu anlaşılmıştır. Bu yüzden deneylerimizde sık öğe kümelerinin öznitelik olarak kullanılması ile ortogonal kodlamanın öznitelik olarak kullanılması karşılaştırılmıştır.

Bu deneyler aynı problem için oluşturulmuş 746 [38], 1625 [18], Impens [3, 11, 15, 25] ve Schilling [35] veri kümeleri üstünde yapılmıştır. Tüm veri setindeki sık öğe kümelerinin çıkarılmasının yanı sıra sadece aynı sınıf türüne sahip örneklerden sık öğe kümeleri çıkarılmış ve ayrı ayrı deneylere tabi tutulmuştur. Bu konu üstünde yaptığımız deneylerden bir diğeri ise sık öğe kümelerinin ortogonal kodlama ile çıkarılan özniteliklerle birleştirildiği durumun test edilmesidir. Bu kısımda yapılan bir diğer deney ise öznitelik çıkarma yöntemlerinin etkisinin denenmesi ve alan bilgisinin çıkarılan özniteliklere aktarıldığındaki performans değişiminin gözlemlenmesidir.

Bu deneylerin nasıl yapıldığı ve alınan sonuçlar aşağıdaki alt bölümlerde paylaşılmaktadır. Dört alt bölüm içinde ortogonal kodlama ile verilerin kodlanması, bu şekilde kodlanmış veri üstünden sık öğe kümelerinin çıkartılması, çıkarılan sık öğe kümelerinden gerekli olan sık öğe kümelerinin ayrıştırılması ve öznitelik olarak kullanılması anlatılmaktadır.

3.1 Veri Kümesinin Ortogonal Kodlanması

Yöntemimizin ilk adımı veri setinin gösteriminin sık öğe kümelerinin çıkarılabilmesi amacıyla değiştirilmesidir. Bu değişiklik amino asitlerin sekizli üstünde bulunduğu pozisyonların sık öğe kümesi çıkarma işlemine dahil edilebilmesi açısından oldukça önemlidir. Bu değişiklik içinde amino asitlerin başına sekizli içinde bulundukları pozisyon bilgisi eklenmiştir. Yapılan değişikliği örneklemek adına "746Dataset" veri kümesinin ilk örneği seçilmiştir. Bu örnek veri kümesi içinde;

$$AAAKFERQ$$

şeklindedir. Aynı örnek pozisyon bilgisi eklendikten sonra şu şekle çevrilmiştir.

$$P_4A, P_3A, P_2A, P_1K, P'_1F, P'_2E, P'_3R, P'_4Q$$

Bu gösterim aynı zamanda ortogonal kodlama olarak bilinmektedir. Bu gösterim içinde her örnek sekizli üstündeki pozisyonu gösteren 8 özniteliğe sahiptir ve her öznitelik 20 farklı amino asit değerinden birini alabilir.

Veri kümesi içindeki bütün örnekler bu gösterime çevrilerek kümemiz sık öğe kümelerinin çıkarılması aşamasına hazırlanmaktadır.

3.2 Sık Öğe Kümelerinin Çıkarılması

Veri seti yeni gösterime çevrildikten sonra sıralı amino asit dizilimine göre sık öğe kümeleri çıkartılmaktadır. Bu çıkartma işleminde belirli bir destek seviyesi üstündeki sık öğe kümeleri FP-Growth [1, 4] algoritması kullanılarak çıkartılmıştır. Bu çalışma içinde veri setindeki öğelerin bir özetini çıkarması amacıyla azami sık öğe kümeleri seçilmiştir. Bütün azami sık öğe kümeleri bir sık öğe kümesi olmak ile birlikte her alt kümesinin destek değeri bilinmemektedir.

Deneylerimizde sık öğe kümeleri 3 farklı şekilde çıkartılmıştır. (1) Sadece kırlmış olan örnekler üstünden, (2) Sadece kırılmamış örnekler üstünden (3) Sınıf değerine bağlı olmadan tüm veri seti üstünden. Bu 3 sık öğe kümesi deneylerimizde farklı kombinasyonlarda kullanılmıştır.

Veri setlerinin üstte belirtilen şekilde ayrılması belirli bir sınıf değeri içindeki örnekler arasındaki örüntüleri gözlemlemek içindir. Bu ayrım veri setini tümden incelediğimizde gözlemleyemeyeceğimiz bazı örüntüleri ortaya çıkarmak için yapılmıştır. Bazı örüntüler veri setinin tümü incelendiğinde az sayıda kaldıkları için destek değerinin altında kalabilirler fakat sınıf değerine bağlı olarak bir incelenme yapıldığında yüksek destek değerine sahip olup ortaya çıkabilirler.

3.3 Veri Setinin Güncellenmesi

Yeterli bir miktarda azami sık öğe kümesi çıkarımı ancak destek değerinin düşük tutulduğu durumlarda görülmüştür. Yapılan deneylerde bu değer 0.05 olarak belirlendiğinde yeterli miktarda azami sık öğe kümesi oluşturulduğu gözlemlenmiştir.

Öznitelikler üstünde yaptığımız çalışmalar bunlar ile sınırlı değildir. Çok sayıdaki öznitelik içinden en ayrıcı olanların seçilmesi sınıflandırıcının performansı açısından tercih edilmesi gereken bir yöntemdir. Veri setinin güncellenmesi işleminde sık öğe kümeleri öznitelik olarak kullanılmış, örnekler ile olan kesişim değerleri aynı pozisyon içinde kaç adet aynı amino asit bulundurduğu olarak belirlenmiştir. Bu fonksiyon *benzerlik* olarak adlandırılmıştır.

Örneğin A bir sık öğe kümesi olmak üzere $(P'_3D, P'_1Y, P'_4S, P_1Y, P_4S)$ öğelerinden oluşsun. B bir örnek olmak üzere $(P_4A, P_3A, P_2A, P_1A, P'_1Y, P'_2P, P'_3D, P'_4K)$ amino asit dizilimiden oluşsun. A ile B arasındaki benzerlik yalnızca 2 olmaktadır çünkü sadece P'_3D ve P'_1Y ikisinin kesişiminde bulunmaktadır. Benzerlik formülü şu şekilde ifade edilebilir:

$$benzerlik(A,B) = Aynı\ pozisyonda\ aynı\ amino\ asit\ bulundurma\ sayısı$$

Yeni veri seti *benzerlik* fonksiyonunun bütün örnek-öznitelik kombinasyonlarına uygulanması ile oluşturulur. M örnekten ve N sık öğe kümesinden oluşan bir veri seti, $M \times N$ büyüklüğündedir.

Öznitelik seçimi için ilk yaklaşımımız Temel bileşenler analizi (PCA, Principal component analysis) [37] yöntemini kullanmak olmuştur. Özetlemek gerekirse PCA, birbiri arasında korelasyon bulunan öznitelikleri lineer olarak korelasyonsuz özniteliklere çevirmek için kullanılır. Bu yüzden bu yöntem veri setindeki öznitelikleri filtrelemek için kullanılabilir.

Öznitelikleri filtrelemek için kullandığımız ikinci yaklaşım sık öğe kümelerinin bulundurduğu pozisyonları temel alarak bir filtreleme yapmaktır. Bu kısımda P_1 ve P'_1 pozisyonundaki öğelere sahip öznitelikler seçilmiştir. Bu poziyonların seçilmesindeki temel neden sekizli diziliminde kırılmanın gerçekleştiği orta pozisyonlar olmaları ve bununla ilgili çalışmalarda bu öğelerin daha bilgilendirici olduğunun belirtilmesidir.

3.4 Deney Sonuçları

4 veri seti deneylerimiz içinde kullanılmıştır. Bunlar 746Data, 1625Data, impensData and schillingData veri setleridir ve bu veri setlerine UCI makine öğrenme veri havuzu [21] üzerinden ulaşılabilir. Veri havuzu içinde veri setleri ile ilgili detaylı bilgi bulunmaktadır.

Deneylerimizde 10'lu çapraz doğrulama yöntemi, eğitim kümesinin gereğinden iyi sınıflandırıldığı ama bu durumun test kümesinde başarımı düşürmesine sebep olması durumunun üstesinden gelmek için kullanılmıştır. 10'lu çapraz doğrulama esnasında her bir test parçası için eğitim verisi üstünden sık öğe kümeleri bulunmuş bu öğelerin bir kısmı seçilmiş ve benzerlik fonksiyonu eğitim ve test verisi (satırlar) sık öğe kümeleri (sütunlar) üstüne uygulanarak yeni veri setleri oluşturulmuştur. Sınıflandırıcı model eğitim parçaları kullanılarak eğitilmiş ve modelin başarısı kalan parça üstüne test edilmiştir.

Sistemimiz python programlama dili ve scikit-learn [30] kütüphanesi kullanılarak geliştirilmiştir. SVC (Destek vektör makinesi) sınıflandırıcısı lineer kernel kullanılarak önceden atanmış parametreler ile çalıştırılmıştır. Pyfim [4] kütüphanesi sık öğe kümelerinin ayrıştırılmasında kullanılmıştır. Yaptığımız karşılaştırmalarda ortogonal kodlama ile kodlanmış orjinal veri setindeki sonuçları 10'lu çarpaz doğrulama yöntemi ile alarak taban değerleri hesaplanmıştır.

Sonuçların paylaşıldığı çizelgelerde kullanılan kısaltmaların açıklamaları Table 3.1 de verilmiştir.

Bu çizelgede verilen kısaltmalar yapılan deneyler içinde hangi yöntemlerin uygulandığını belirtmek için kullanılmaktadır. Örnek olarak OK + SÖK-TÜM + SÖK-MERKEZ + DESTEK-3 + PCA-100 kısaltması, "Ortogonal kodlama ile çıkarılmış özniteliklere tüm veri setinden %3'ten büyük destek değerine sahip sık öğe kümelerini ekle, bunlar arasından sadece merkez pozisyondaki öğeleri bulunduranları seç ve PCA yöntemi ile en iyi 100 tanesini öznitelik olarak kullan" cümlesinin yerini tutmaktadır.

Çizelge 3.1: Kısaltma & Açıklama

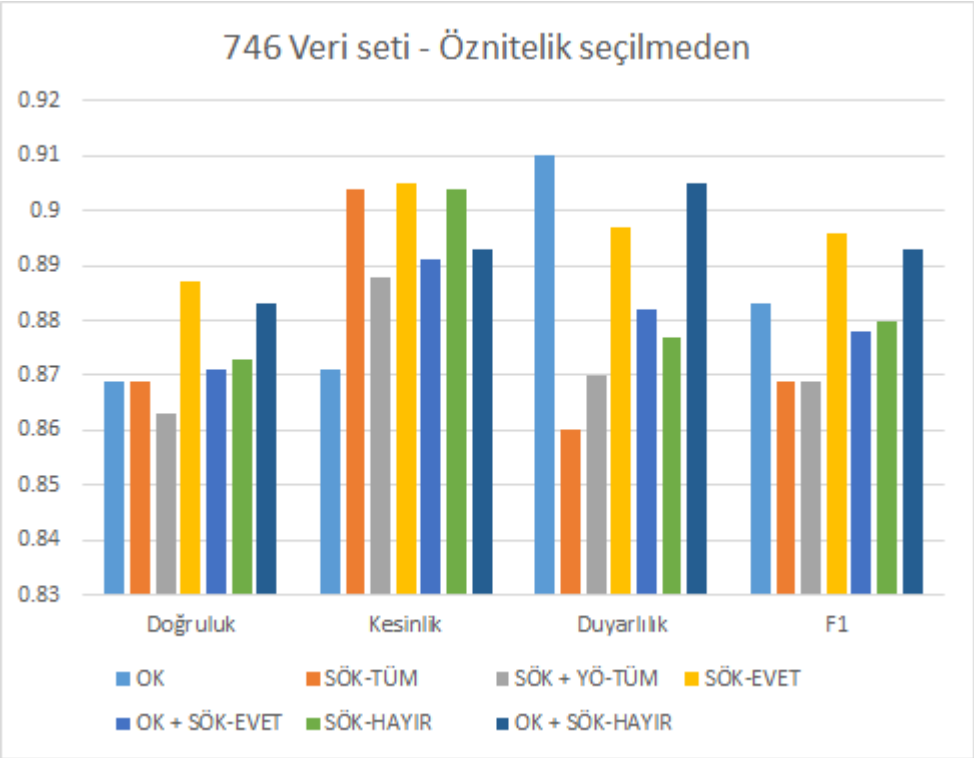
Kısaltma	**Açıklama**
OK	Ortogonal kodlama öznitelikleri.
SÖK-TÜM	Sık öğe kümeleri hem kırılmış hem de kırılmamış örneklerden elde edilmiştir.
SÖK-EVET	Sık öğe kümeleri sadece kırılmış örneklerden elde edilmiştir.
SÖK-HAYIR	Sık öğe kümeleri sadece kırılmamış örneklerden elde edilmiştir.
DESTEK-m	Sık öğe kümeleri için asgari destek eşiği m olarak belirlenmiştir. Eğer bir satırda geçmiyorsa ise asgari destek miktarı %3 olarak seçilmiştir.
PCA-100	Öznitelik seçme yöntemi olarak PCA kullanılmıştır ve 100 öznitelik seçilmiştir.
SÖK-MERKEZ	Sık öğe kümeleri P_1 veya P_1' pozisyonundaki öğeleri bulunduranlar öznitelik olarak seçilmiştir.

Deneylerimizin başarım metriği olarak doğruluk, kesinlik, duyarlılık ve F1 skoru seçilmiştir ve bu sonuçlar ilgili veri setleri için çizelgelerde toplarlanmış ve uygun diagramlar çizilmiştir.

746 veri setinin sonuçları Çizelge 3.2 ve 3.3'te toplanmıştır. Sonuçların görselleştirilmesi Şekil 3.1 ve 3.2'de verilmiştir. Bu veri seti içinde ortogonal kodlama ve ortogonal kodlamanın PCA yöntemi ile 100 öznitelik seçilme durumları taban durum olarak belirlenmiş ve deneylerimizde bu durumdan daha iyi sonuç alınması hedeflenmiştir. Çizelge 3.2 içinde üç farklı durum için sık öğe kümeleri çıkartılmış ve deneylerin performansları ölçülmüştür. Daha sonra ortogonal kodlama öznitelikleri sık öğe kümelerinden çıkartılan özniteliklere eklenerek bu durumun başarımı nasıl etkilediği gözlemlenmiştir. En iyi sonucun alındığı deney sonuçları paylaşıldığı çizelgede kalınlaştırılarak belirtilmiştir.

Çizelge 3.2: 746 Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK	0.869	0.871	0.910	0.883
SÖK-TÜM	0.869	0.904	0.860	0.869
OK + SÖK-TÜM	0.863	0.888	0.870	0.869
SÖK-EVET	**0.887**	**0.905**	**0.897**	**0.896**
OK + SÖK-EVET	0.871	0.891	0.882	0.878
SÖK-HAYIR	0.873	0.904	0.877	0.880
OK + SÖK-HAYIR	0.883	0.893	0.905	0.893



Şekil 3.1: 746 Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması görselleştirilmektedir.

Deneylerimiz sonucunda ilk farkettiğimiz durum sadece ortogonal kodlama kullanıldığı deneylerde PCA yöntemi ile öznitelikler seçildiğinde başarımın azalmasıdır. Ortogonal kodlama ile kıyaslandığında kırılmış örnekler üstünden oluşturulan sık öğe kümeleri (SÖK-EVET) daha iyi sonuç vermektedir. Tüm veri setinden oluşturulan öğelerin bulunduğu SÖK-TÜM durumu ortogonal kodlama ile benzer sonuçlar vermektedir. Bu üç sık öğe kümesi arasında SÖK-HAYIR en kötü sonuçlara sahiptir.

Ortogonal kodlama özniteliklerini sık öğe kümelerinden oluşturulan özniteliklerle birleştirdiğimizde ilginç sonuçlar gözlemlemekteyiz. SÖK-TÜM ve SÖK-EVET ile yapılan birleştirmelerde sadece SÖK-TÜM veya sadece SÖK-EVET kullanılan durumlara göre başarımın azaldığı ölçülmüştür. Bu durum aslında SÖK-TÜM ve SÖK-EVET sık öğe kümelerinin veri setini ortogonal kodlamaya benzer bir seviyede temsil edebilmesinden kaynaklanmaktadır. Sonuçların kötü alınmasının nedeni benzer nitelikteki özniteliklerin eklenerek öznitelik boyutunun herhangi yeni bir bilgi aktarılmadan arttırılmasıdır. SÖK-HAYIR sık öğe kümeleri her ne kadar öznitelik olarak tek başına kullanıldıklarında kötü performans göstermiş olsalar da ortogonal kodlama öznitelikleri ile birleştirildiklerinde yapılan deneyler arasındaki en başarılı sonucu göstermişlerdir.

Ayrıca özniteliklerin seçildiği durumda SÖK-MERKEZ ile seçilen özniteliklerin eklenmesi başarımı arttırmıştır.

Çizelge 3.3: 746 Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması gösterilmektedir.

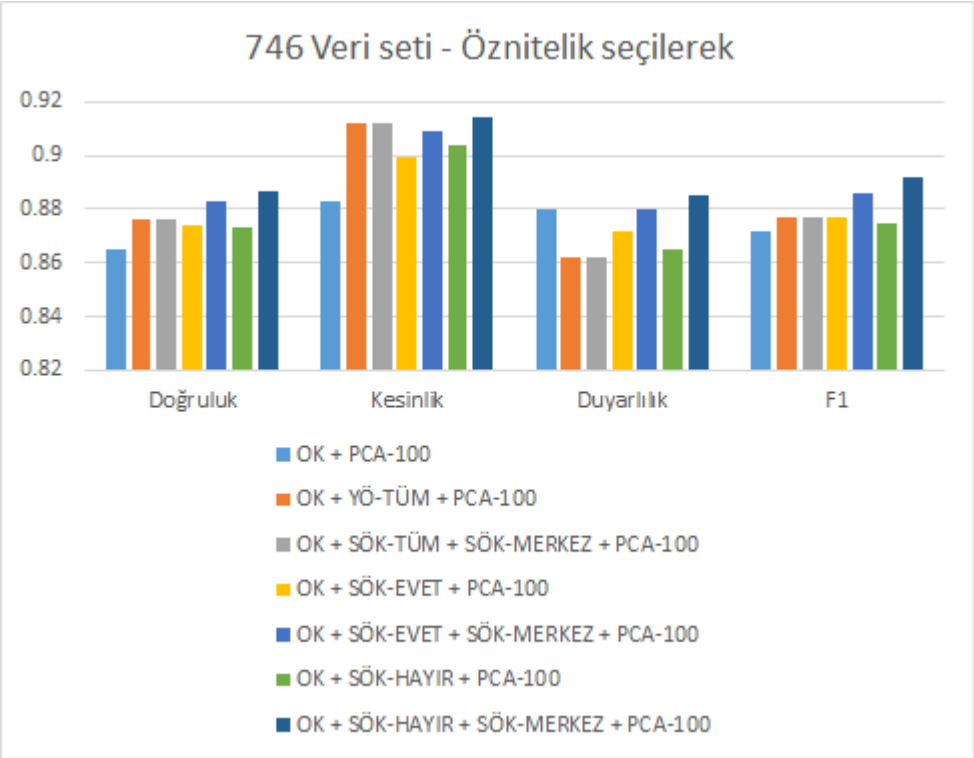
Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK + PCA-100	0.865	0.883	0.880	0.872
OK + SÖK-TÜM + PCA-100	0.876	0.912	0.862	0.877
OK + SÖK-TÜM + SÖK-MERKEZ + PCA-100	0.876	0.912	0.862	0.877
OK + SÖK-EVET + PCA-100	0.874	0.899	0.872	0.877
OK + SÖK-EVET + SÖK-MERKEZ + PCA-100	0.883	0.909	0.880	0.886
OK + SÖK-HAYIR + PCA-100	0.873	0.904	0.865	0.875
OK + SÖK-HAYIR + SÖK-MERKEZ + PCA-100	**0.887**	**0.914**	**0.885**	**0.892**

Impens veri setinin sonuçları Çizelge 3.4 ve 3.5'te toplanmıştır. Sonuçların görselleştirilmesi Şekil 3.3 ve 3.4'te verilmiştir. Taban durumumuz ile karşılaştırıldığında SÖK-TÜM ve SÖK-HAYIR daha iyi bir doğruluk değerine sahip olmasına rağmen daha düşük bir F1 skoruna sahiptir. SÖK-EVET bütün deneyler arasında en kötü başarıma sahiptir. Bu deneylerde görülen önemli sonuçlardan biri de ortogonal kodlama özniteliklerinin sık öğe kümeleri ile beraber kullanıldığı durumların F1 skorunu arttırmasıdır. Öznitelik seçimi olmadan yapılan bu deneylerde ortogonal kodlama SÖK-TÜM ile seçilen sık öğe kümelerinin kullanılması en iyi doğruluk sonucunu vermiştir.

Çizelge 3.4: Impens Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK	0.876	0.620	0.645	0.636
SÖK-TÜM	**0.889**	**0.675**	**0.603**	**0.628**
OK + SÖK-TÜM	0.885	0.654	0.696	0.659
SÖK-EVET	0.866	0.609	0.656	0.609
OK + SÖK-EVET	0.871	0.622	0.649	0.615
SÖK-HAYIR	0.882	0.634	0.569	0.590
OK + SÖK-HAYIR	0.879	0.625	0.67	0.634

Özniteliklerin seçildiği durumda ise SÖK-MERKEZ şeklinde seçilen öğeler doğruluk değerini her zaman arttırmaktadır. SÖK-HAYIR'ın kullanıldığı durumda SÖK-MERKEZ öğelerinin seçilmesi başarımı en yüksek oranda arttırarak impens veri seti üstündeki en iyi sonucun alınmasını sağlamıştır.



Şekil 3.2: 746 Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapıldığında performans açısından karşılaştırılması görselleştirilmektedir.

Çizelge 3.5: Impens Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK + PCA-100	0.871	0.620	0.623	0.662
OK + SÖK-TÜM + PCA-100	0.891	0.672	0.596	0.627
OK + SÖK-TÜM + SÖK-MERKEZ + PCA-100	0.893	0.706	0.576	0.621
OK + SÖK-EVET + PCA-100	0.881	0.650	0.596	0.611
OK + SÖK-EVET + SÖK-MERKEZ + PCA-100	0.888	0.669	0.602	0.622
OK + SÖK-HAYIR + PCA-100	0.879	0.640	0.570	0.593
OK + SÖK-HAYIR + SÖK-MERKEZ + PCA-100	**0.895**	**0.685**	**0.630**	**0.650**

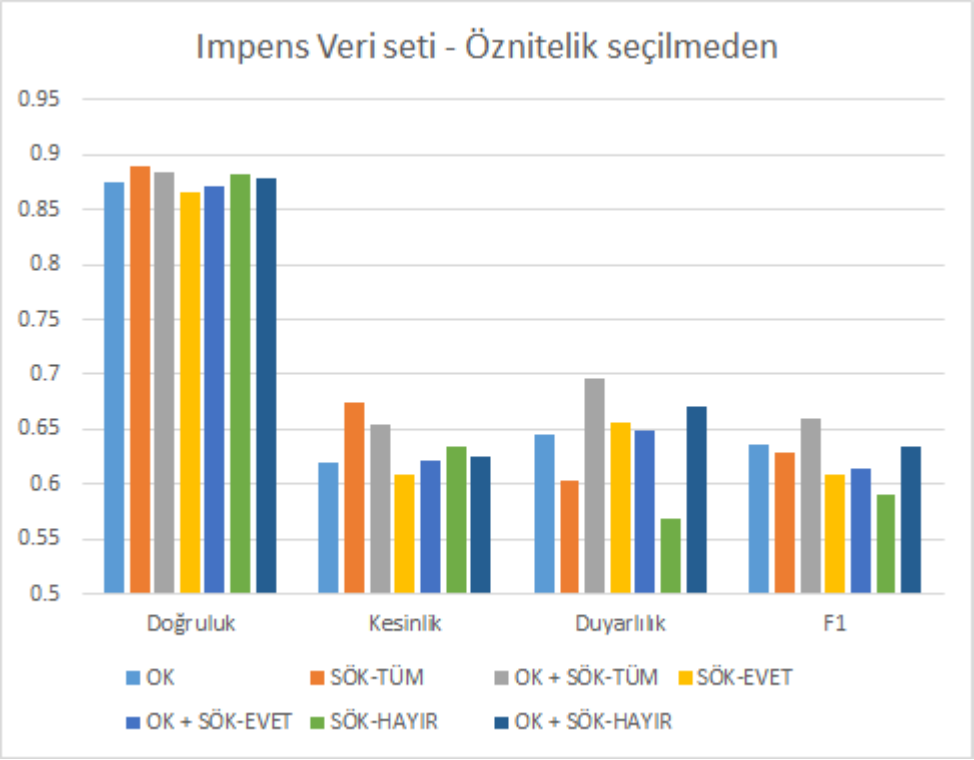
1625 veri setinin sonuçları Çizelge 3.6 ve 3.7'de toplanmıştır. Sonuçların görselleştirilmesi Şekil 3.5 ve 3.6'da verilmiştir. Önceki deneylerde denenen kombinasyonlar taban durumdan daha iyi bir sonuç almamızı sağlamamıştır. Bu nedenle taban durumu geçmek için sık öğe kümesi sayısını arttırmak denenmiştir. Bu nedenle %3 olarak kullanılan destek değeri %1 olarak değiştirilmiş ve daha çok sık öğe kümesinin seçilmesi

sağlanmıştır. Bu değişiklik SÖK-EVET ve SÖK-HAYIR için başarımı büyük ölçüde arttırmış ve SÖK-HAYIR durumu için en başarılı sonucun alınmasını sağlamıştır.

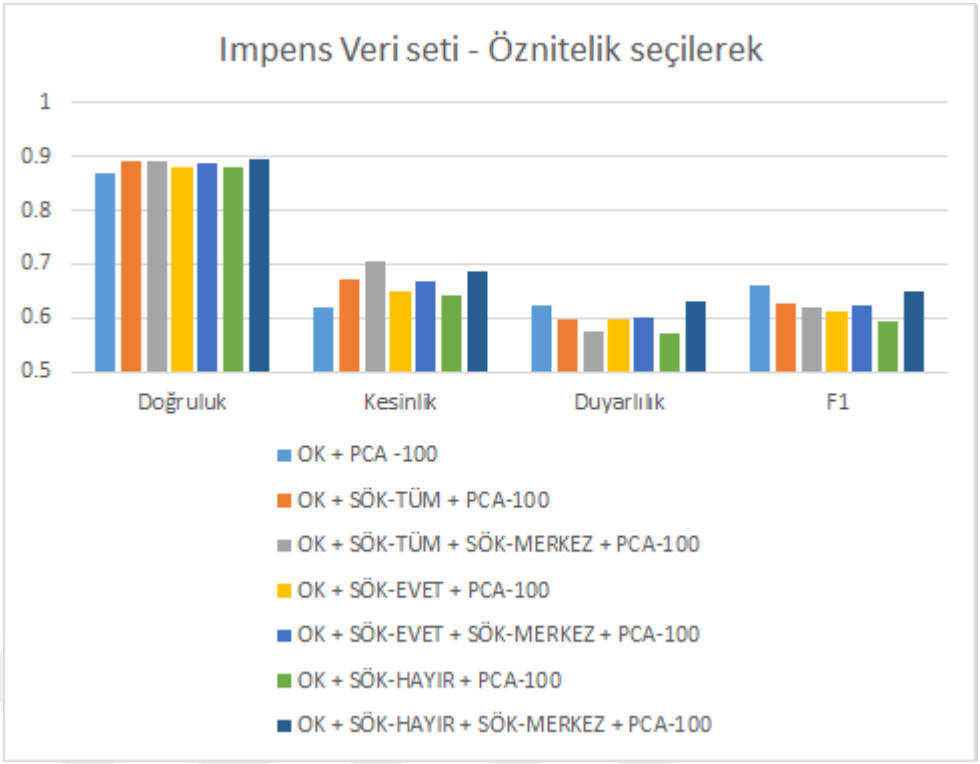
Çizelge 3.6: 1625 Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK	0.929	0.885	0.820	0.839
SÖK-TÜM	0.926	0.876	0.823	0.836
SÖK-TÜM + DESTEK-1	0.925	0.877	0.817	0.836
OK + SÖK-TÜM	0.926	0.872	0.820	0.836
SÖK-EVET	0.915	0.866	0.777	0.805
SÖK-EVET + DESTEK-1	0.924	0.874	0.820	0.834
OK + SÖK-EVET	0.923	0.874	0.807	0.828
SÖK-HAYIR	0.922	0.860	0.820	0.825
SÖK-HAYIR + DESTEK-1	**0.930**	**0.885**	**0.828**	**0.844**
OK + SÖK-HAYIR	0.928	0.875	0.828	0.841

Öznitelik seçiminin yapıldığı durumlarda ise ortogonal kodlama ve sık öğe kümeleri ile seçilen öznitelikler SÖK-MERKEZ ile filtrelendiği taktirde taban durumundan daha iyi bir sonuç vermemiştir. Üstteki deneyler sonucunda destek miktarının azaltılmasının başarımı olumlu etkileyebileceğinin anlaşılmasıyla aynı durum bu denenmiştir. Aynı

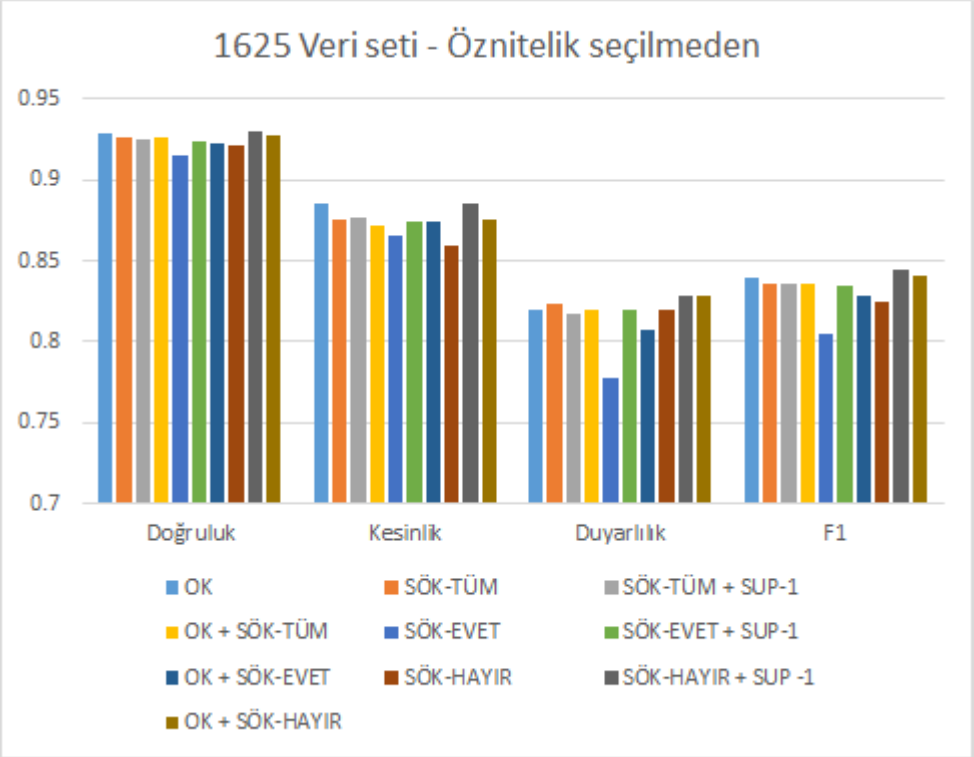


Şekil 3.3: Impens Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması görselleştirilmektedir.



Şekil 3.4: Impens Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması görselleştirilmektedir.

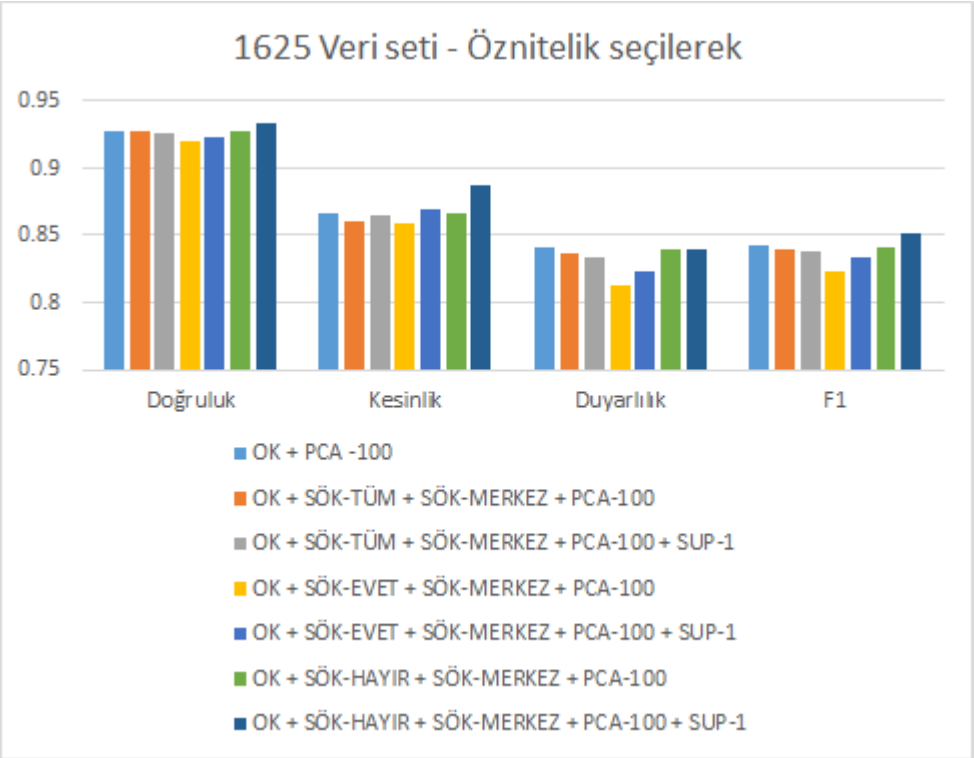
şekide en büyük başarım artışı SÖK-EVET ve SÖK-HAYIR için olmuştur. Bu değişikliğin yapılması sayesinde taban durumumuzdan daha iyi sonuçlar alınmıştır.



Şekil 3.5: 1625 Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması görselleştirilmektedir.

Çizelge 3.7: 1625 Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK + PCA-100	0.927	0.866	0.841	0.843
OK + SÖK-TÜM + SÖK-MERKEZ + PCA-100	0.927	0.861	0.836	0.839
OK + SÖK-TÜM + SÖK-MERKEZ + PCA-100 + DESTEK-1	0.926	0.865	0.833	0.838
OK + SÖK-EVET + SÖK-MERKEZ + PCA-100	0.920	0.859	0.812	0.823
OK + SÖK-EVET + SÖK-MERKEZ + PCA-100 + DESTEK-1	0.923	0.869	0.823	0.833
OK + SÖK-HAYIR + SÖK-MERKEZ + PCA-100	0.927	0.866	0.839	0.841
OK + SÖK-HAYIR + SÖK-MERKEZ + PCA-100 + DESTEK-1	**0.934**	**0.887**	**0.839**	**0.852**



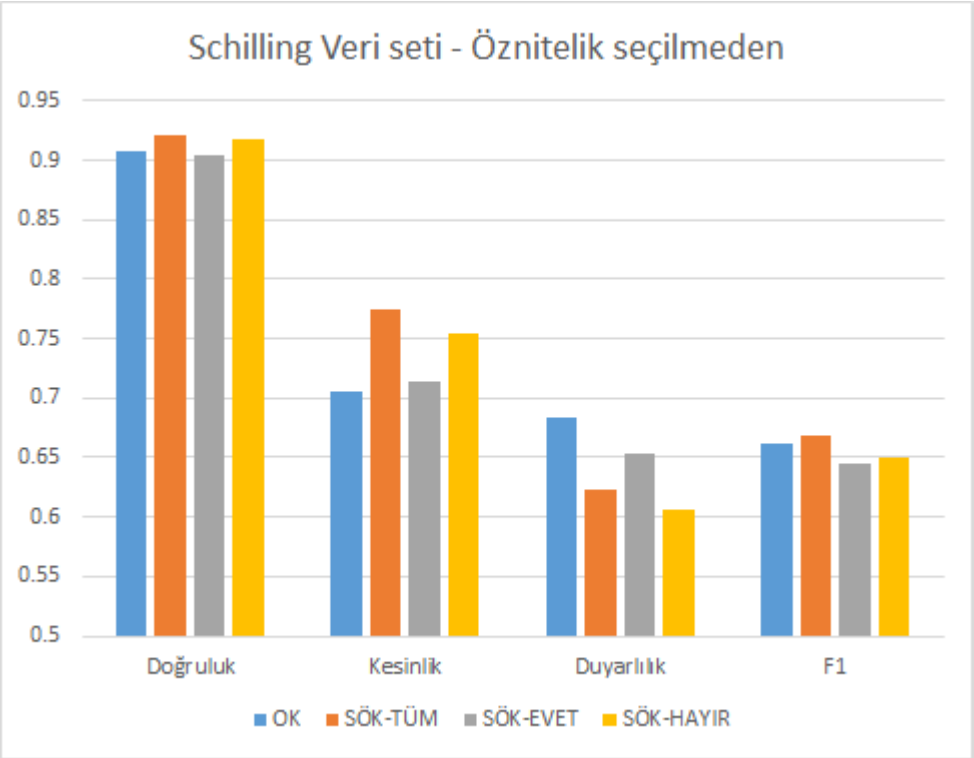
Şekil 3.6: 1625 Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması görselleştirilmektedir.

Schilling veri seti için sonuçlar Çizelge 3.8 ve 3.9'de toplanmıştır. Sonuçların görselleştirilmesi Şekil 3.7 ve 3.8'de verilmiştir. Sadece SÖK-HAYIR'ın kullanıldığı durum SÖK-EVET durumundan daha iyi sonuç göstermiş fakat en başarılı durum SÖK-TÜM kullanıldığında alınmıştır.

Öznitelik seçiminin yapıldığı durumda ise bütün durumlar öznitelik seçiminin yapılmadığı durumlara kıyasla daha iyi sonuçlar vermiştir. Bunlar arasından ortogonal kodlama ile SÖK-EVET yönteminin birleştirildiği, SÖK-MERKEZ ile filtrelendiği durum en başarılı durum olmuştur.

Çizelge 3.8: Schilling Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK	0.907	0.706	0.683	0.661
SÖK-TÜM	**0.922**	**0.774**	**0.623**	**0.668**
SÖK-EVET	0.904	0.714	0.653	0.645
SÖK-HAYIR	0.918	0.754	0.607	0.650



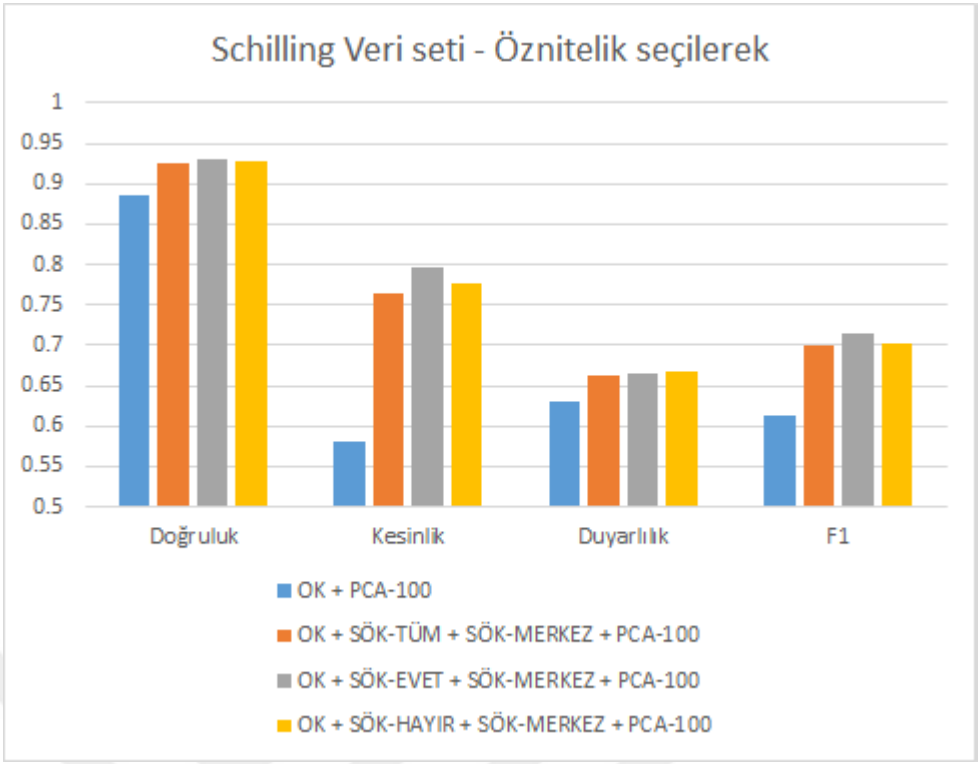
Şekil 3.7: Schilling Veri seti - Öznitelik seçme işlemi yapılmadan farklı sık öğe kümelerinin ayrıştırılmasının performans açısından karşılaştırılması görselleştirilmektedir.

Çizelge 3.9: Schilling Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması gösterilmektedir.

Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
OK + PCA-100	0.886	0.581	0.631	0.614
OK + SÖK-TÜM + SÖK-MERKEZ + PCA-100	0.926	0.765	0.663	0.699
OK + SÖK-EVET + SÖK-MERKEZ + PCA-100	**0.931**	**0.797**	**0.665**	**0.715**
OK + SÖK-HAYIR + SÖK-MERKEZ + PCA-100	0.927	0.777	0.667	0.701

3.5 Çıkarımlar

Aynı problem üstündeki farklı veri setlerinde yapılan deneyler sonucunda sık öğe kümelerinin protein kırılma probleminde ortogonal kodlamaya göre daha başarılı öznitelikler olabilecekleri, ayrıca ortogonal kodlama ile birleştirilerek ortogonal kodlamanın başarısını arttırabileceği gözlemlenmiştir. Sık öğe kümeleri için sınıf değeri baz alınarak veri setinin ayrılmasının genele bakıldığında az miktarda görülen öğelere ulaşmak için kullanılabileceği ve bu durumun başarımı arttırabileceği, bizim deneylerimizde yapmış olduğumuz merkezdeki amino asitlerin seçilmesi gibi alan bilgisinin daha kolaylıkla aktarılabileceği görülmüştür. Bu sonuçlarlar doğrultusunda sık öğe kümeleri-



Şekil 3.8: Schilling Veri seti - Farklı sık öğe kümelerinin ayrıştırıldığı ve ortogonal kodlama ile birlikte kullanıldığı durumlarda öznitelik seçilme işlemi yapılmasının performans açısından karşılaştırılması görselleştirilmektedir.

nin yeni örneklerin sınıflandırılması için kullanılabilecekleri sonucuna varılmıştır.

Bu çalışmada yapılan deneyler sonucunda sık öğe kümelerinden oluşturulan bir ağın önemli bulgular verebileceği ve kırılma problemini çözmek amacıyla bir sınıflandırıcıya dönüştürülebileceği fikri ortaya çıkmıştır. Bir sonraki bölümde sık öğe kümesi ağı tabanlı sınıflandırıcının (SÖKA-SNF) nasıl geliştirildiği, eğitim ve tahmin aşamalarını nasıl yaptığı detaylı bir şekilde anlatılmaktadır.

4. YÖNTEM

Bu bölüm içinde sık öğe kümesi ağı tabanlı sınıflandırıcının (SÖKA-SNF) oluşturulmasında kullanılan adımlar tanımlanmaktadır. Bu adımlar dört farklı alt bölümlerde açıklanacak şekilde ayrılmıştır. İlk alt bölüm, veri kümesinin temsilinin ortogonal kodlama (orthogonal encoding) şeklinde değiştirilmesini anlatmaktadır. İkinci alt bölümde, SÖKA-SNF'nin eğitim adımında kullanılan sık öğe kümelerinden ağ oluşturma işlemi açıklanmaktadır. Üçüncü alt bölümde oluşturulan ağın yeni verileri tahmin etmede nasıl kullanıldığı gösterilmektedir. Dördüncü bölümde ise tahmin aşamasında kullanılan eşik değerinin nasıl öğrenildiği açıklanmaktadır.

4.1 Veri Kümesinin Ortogonal Kodlanması

Yöntemimizin ilk adımı veri setinin gösteriminin sık öğe kümelerinin çıkarılabilmesi amacıyla değiştirilmesidir. Bu değişiklik 3.1 bölümünde anlatılan veri kümesinin ortogonal kodlanması ile aynı şekilde yapılmaktadır. Amacımız veri setindeki amino asitlerin üzerlerine bulunduklarını pozisyon bilgisini ekleyerek pozisyon bazındaki bilgileri veri setine aktarmaktır.

Veri kümesi içindeki bütün örnekler bu gösterime çevrilerek eğitim aşamasına hazırlanmaktadır.

4.2 SÖKA-SNF Eğitimi

Sık öğe kümelerinden ağ oluşturma işlemi veri kümesinden sık öğe kümelerinin bulunması ile başlamaktadır. Bizim problemimizde kırılma işleminin alabileceği iki farklı sınıf değeri bulunmaktadır, kırılmanın olması veya olmaması. Belirli bir sınıfa ait olan sık öğe kümelerinin bulunması amacıyla veri kümemiz bir alt kümesi kırılma olan örneklerden diğeri kırılma olmayan örneklerden oluşmak üzere iki alt kümeye ayrılmıştır. Bu alt kümeleri kullanarak iki ayrı kapalı sık öğe kümesi çıkartılmıştır. Kapalı sık öğe kümelerinin sık öğe kümelerine kıyasla seçilmesinin sebebi veri kümesi içindeki öğelere göre de daha sıkıştırılmış biçimde bulunmalarıdır. Bu durum aynı destek değerine sahip sık öğelerin küme içinden çıkartılmasıyla, sık öğe kümelerinin özetlenmesi açısından oldukça kullanışlıdır.

Bütün kapalı sık öğe kümeleri destek değerlerine göre büyükten küçüğe sıralanmış ve seçmeye üstten başlayarak 100'ü kırılmanın olduğu, 100'ü kırılmanın olmadığı öğe kümelerinden seçilen toplam 200 kapalı sık öğe kümesi, sık öğe kümesi ağında bulunması için seçilmiştir.

Seçilen sık öğe kümeleri SÖKA'nın içindeki düğümleri oluşturmaktadır ve bu düğümler arasındaki kenarların ağırlıkları öğe kümelerinin örnekler içinde birlikte bulunma

sayıları olarak belirlenmiştir. Eğer iki öğe kümesi hiçbir zaman aynı örnek içinde bulunmamışsa bu iki öğe kümesi arasında bir kenar oluşturulmamaktadır. SÖKA oluşturulduktan sonra her bir düğüm için ağırlıklı pagerank, arasındalık ve derece skorları hesaplanarak tahmin adımında kullanılmaktadır. Bu merkeziyet skorları bir düğümün ağ içindeki önemini çıkarmak için kullanılmaktadır. Geliştirdiğimiz sınıflandırıcıda bir düğüm ağ içinde ne kadar önemli ise tahmin aşamasında da o kadar çok söz hakkına sahiptir.

Ağ içindeki bütün düğümler için tanımladığımız bir diğer metrik ise sınıf skoru metriğidir. Sınıf skoru bir düğümün bu sınıf ile aynı sınıftaki kaç örnek içinde göründüğü olarak tanımlanmıştır. Örneğin kırılma skoru, kırılmış örneklerden kaç tanesinin bu düğümü içerdiğidir.

Tahmin aşamasında kullanmak amacıyla sınıf skoru metriğini kullanarak iki farklı metrik daha geliştirilmiştir. Bunlar normalize sınıf skoru ve sınıf güvenidir. Normalize sınıf skoru, sınıf skorunun bu sınıfa ait toplam örnek sayısına bölünmesi olarak tanımlanmıştır ve bu skor bir düğümün ağ içindeki belirli bir sınıfı temsil gücünü hesaplamak için kullanılmaktadır. Sınıf güveni, sınıf skorunun düğümün görüldüğü toplam örnek sayısına bölünmesi ile hesaplanmaktadır. Burada bulunması amaçlanan düğümün belirli bir sınıf değeri için ne kadar güven verdiğidir. Eğer bir düğüm büyük çoğunlukla belli bir sınıf değeri almış örneklerde bulunuyorsa, bu değer yüksek çıkacaktır. Bu da dolayısıyla düğümün bu değerinin örneğin bu sınıf değerini almasında etkili olabileceğini belirtmektedir. Tahmin aşamasında sınıf güveni ve sınıf skoru değerleri yüksek olan bir düğüm daha etkili bir rol oynamaktadır.

Bizim problemimizde kırılmış ve kırılmamış olmak üzere iki adet sınıf bulunmaktadır. Bu yüzden her düğüm için iki normalize sınıf skoru, iki sınıf güven skoru hesaplanmıştır. Bu hesaplamaların sonunda eğitim aşaması tamamlanmış olmaktadır ve bu skorlar tahmin aşamasında kullanılmaya hazırdır.

4.3 SÖKA-SNF Tahmini

Yeni örneklerin tahmini eğitim aşamasında öğrenilen bilgiler kullanılarak yapılmaktadır. Tahmin edilecek her örnek için bu örneğin alt kümesi olan bütün sık öğe kümeleri, örneğin sınıflandırılma işleminde oy kullanırlar. Düğümler her bir sınıf değeri için oy verirken, oylarının etkisi aşağıdaki formül ile hesaplanır.

$$\begin{aligned} oy\ etkisi(v) = arasındalık\ skoru(v) \times pagerank\ skoru(v) \times normalize\ sınıf\ skoru(v) \\ \times\ sınıf\ güveni(v) \times benzerlik\ skoru(v)^{\,2} \div derece\ skoru(v) \end{aligned}$$

Bu formül eğitim aşamasında tanımladığımız metrikleri birleştirmek amacıyla tasarlanmıştır. Fikirdeki ana motivasyonumuz ağ içinde merkezde bulunan bir düğümün, merkezde bulunmayan düğümlere kıyasla daha önemli olduğu ve bu yüzden tahmin aşamasında daha çok söz sahibi olması gerektiğidir. İki merkeziyet metriği, pagerank ve betweenness merkeziyet kavramının belirlenen amaca göre farklı tanımları olduğu için birleştirilmiştir. Formül içinde derece skorunun bölüm olarak gelmesinin sebebi, bir sık öğe kümesinin sadece veri setinde daha çok bulunan bir öğe içerdiği için yüksek merkeziyet değerine sahip olmasını engellemek amacıyladır.

Daha yüksek bir normalize sınıf skoruna sahip olmak, örnekler içinde bu sınıfa ait olanların bu düğümü daha sık bulundurduğu anlamına gelmektedir. Bu yüzden bu düğümün oy etkisi yüksek olmalıdır. Sık öğe kümeleri doğaları gereği iki sınıf içinde de bulunabilmektedirler. Eğer bir sık öğe kümesi iki sınıf içinde de eşit oranlarda bulunuyorsa bu sık öğe kümesi sınıflandırma işlemi için iyi bir belirteç değildir. Tam tersi durumda ise eğer bir sık öğe kümesi sadece bir sınıf içinde bulunuyorsa bu iyi bir belirteç olabilir ve bu yüzden oy etkisi daha yüksek olmalıdır. Sınıf güveni metriği bu özelliği yansıtmak için eklenmiştir.

Formül içinde bulunup eğitim bölümünde belirtilmemiş tek şey benzerlik skorudur. Benzerlik skoru örnek ile sık öğe kümesi arasındaki benzerlik olarak tanımlanmıştır. Bir örnek içinde 8 farklı öznitelik bulunmaktadır ve bu öznitelikler sık öğe kümelerini oluşturmaktadır. Sık öğe kümesi ile örnek arasındaki bu benzerlik ikisinin kesişimi arasındaki öznitelik sayısı olarak belirtilmiştir. Örneğin $P_4A, P_3A, P_2A, P_1K, P_1'F, P_2'E, P_3'R, P_4'Q$ örneği ile P_4A, P_3K, P_2Q, P_1K sık öğe kümesi arasındaki benzerlik skoru 2 olur çünkü sadece P_4A ve P_1K kesişim kümesindedir.

Örneğin daha büyük bir oranına sahip sık öğe kümelerinin oy etkisini arttırmak için benzerlik skorunun karesi kullanılmaktadır. İki sık öğe kümesi, P_4A ve P_4A, P_1K arasından ikincisi daha yüksek bir etkiye sahip olmalıdır çünkü örneğin daha büyük bir kısmını temsil etmektedir.

Oylama sonucunda bir örneğin sınıfı kırılmış sınıfı için verilen oyların kırılmamış sınıfı için verilen oylara bölünmesi ile bulunur. Eğer bu oran belirlenmiş bir eşik değerinden daha yüksek ise kırılmış, eğer düşük ise kırılmamış olarak tahmin edilir. Bu eşik değeri oldukça önemlidir çünkü sık öğe kümesi çoğu problemde tek taraflı olarak çıkmaktadır. Bizim problemimiz için enzimin kırabilmesi için belirli bir örüntü bulunmakta fakat kıramayacağı sekizliler daha büyük bir kümede oldukları için belirli bir örüntü bulunmamaktadır. Bu yüzden kırılmamış durumdaki örneklerden oluşturulan sık öğe kümeleri kırılmama durumundaki karakteristiği tam olarak gösterememektedir. Bu sebebten ötürü kırılmış örneklerden elde edilen sık öğe kümeleri genel olarak bütün örneklerin içinde daha sık bir şekilde görülecektir ve bu öğeler kırılma yönünde oy vereceklerdir. Bu iki oy arasındaki oransal farkın öğrenilmesiyle oluşturulan eşik değeri bu durumun tahmin aşamasını kötü etkilemesinin önüne geçebilir. En iyi eşik değerini bulmak bizim algoritmamızın önemli bir parçasıdır ve bu işlemin nasıl yapıldığı bir sonraki bölümde açıklanmaktadır.

4.4 SÖKA-SNF Eşik Öğrenimi

Eşik değeri kırılmış ve kırılmamış sınıflar için oyların oranlarından sınıf tahmini yaparken tahmin edilen sınıfı seçmede kullanılan değerdir. Eğer oyların oranı eşik değerinden büyük ise bakılan örneğin sınıfı kırılmış olarak, aksi durumda örneğin sınıfı kırılmamış olarak tahmin edilmektedir. Geliştirilen sınıflandırıcıda bu eşik değeri veri içinden yapılan ayrım içinde validasyon verisi üstünde en yüksek başarımı gösteren değer olarak seçilmektedir.

Eşik değerini öğrenme işleminde çapraz doğrulama yöntemi kullanılmıştır. Bu işlem veri kümesindeki verileri 10 parçaya ayırmak ile başlar. Bu ayrım yapılırken her bir

parça içindeki sınıf oranlarının genel veri içindeki sınıf oranlarına yakın olmasına özen gösterilir. Bu 10 parçanın 9'u eğitim verisinde kullanılır, oluşturulan model kalan bir parça üzerinde test edilir. Bu işlem esnasında 0 ile 5 arasında 0.2 aralıkları ile artan eşik değerleri model üstünde denenir ve en iyi başarımı veren eşik değeri saklanır. Bu işlem 10 kere tekrarlanır ve her seferinde test parçası değiştirilir. Saklanan başarımların medyanı bu işlem sonucunda seçilen eşik değeri olarak yeni örnekleri sınıflandırmada kullanılır.

5. DENEY SONUÇLARI VE ÖNERİLER

Önerilen yöntemin test edilmesinde 746 veri seti [38] kullanılmıştır. Bu veri setine UCI makine öğrenme veri havuzu [21] üzerinden ulaşılabilir.

Deneylerimizde başarım ölçümü 10'lu çapraz doğrulama yöntemi kullanılarak yapılmıştır. Bu ölçme yönteminin kullanılmasının en önemli sebebi sınıflandırıcının görmediği verileri nasıl sınıflandırdığının anlaşılmasıdır. 10 Parçaya ayrılan veri seti içindeki her parça veri seti içindeki sınıf dağılım oranlarını koruyacak şekilde dikkatlice ayrılmıştır. Bunun yapılmasının sebebi parçaların her birinin veri setindeki özellikleri göstermesinin istenmesidir. Sınıflandırıcının eğitilmesinde 10 parçadan 9'u kullanılmış, başarım kalan bir parça üzerinde test edilmiştir. SÖKA-SNF içinde eğitim verisi hem eğitim aşamasında hem de eşik değerinin öğrenilmesinde kullanılmıştır. Ayrılan test verisi ise SÖKA-SNF tahmin aşamasında tahmin edilmeye çalışılmıştır.

SÖKA-SNF'nin gerçekleştirilmesi python programlama dili kullanılarak yapılmıştır. Sık Öğe Kümesi Ağı'nın oluşturulması ve ağ üzerinden çıkarılan bilgiler graph-tool [31] kütüphanesi aracılığıyla yapılmıştır. Kapalı sık öğe kümelerinin çıkarılması Pyfim [4] kütüphanesi üzerinde fp-growth algoritmasının kullanılması ile gerçekleştirilmiştir.

SÖKA-SNF'nin başarımı Karar Ağacı ve K-en yakın komşu yöntemlerinin ortogonal kodlama ile kodlanmış veri seti üstünde yapılan deneyler ile karşılaştırılmış, önceki çalışmada kullanılan sık öğe kümelerinin öznitelik olarak değerlendirildiği ve destek vektör makineleri ile sınıflandırıldığı çalışmadan alınan en iyi sonuç eklenmiş, doğruluk kesinlik duyarlılık ve f1 skorları ölçülmüştür. K-en yakın komşu yöntemi için farklı K değerleri denenmiştir. Aldığımız sonuçlar 5.1 çizelgesinde paylaşılmıştır ve SÖKA-SNF ile alınan sonuçlar kalınlaştırılmıştır. Oluşturulan sık öğe kümesi ağının pagerank merkeziyet değerleri ile renklendirildiği bir görsel Şekil 5.2 ve 5.3 de paylaşılmıştır.

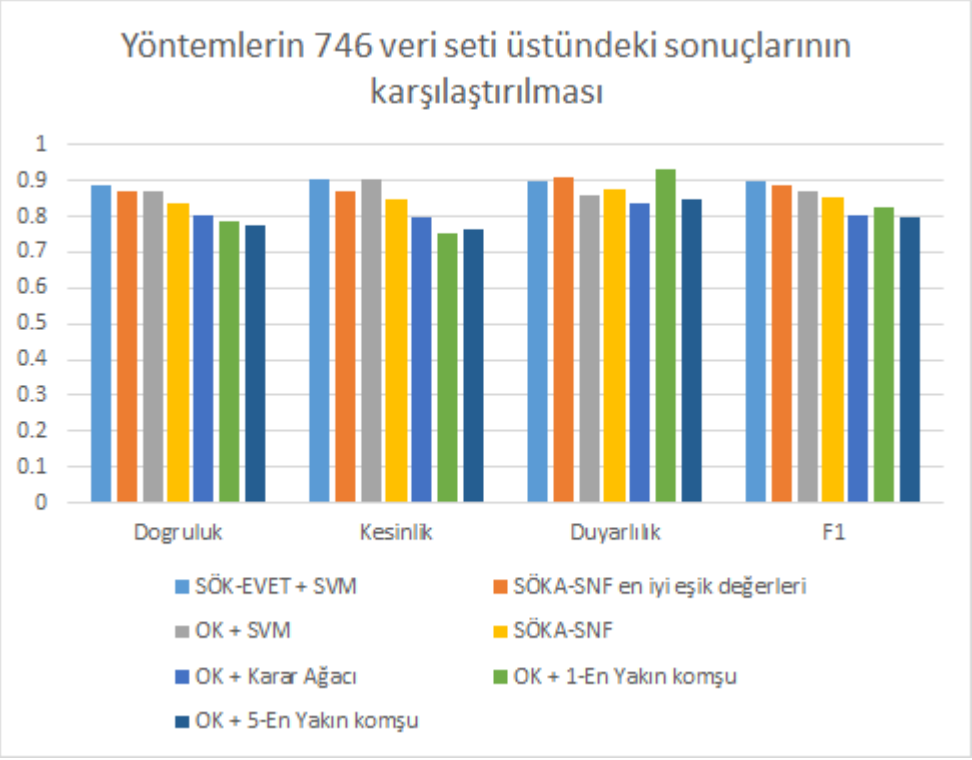
SÖKA-SNF için eşik değerinde yapılan değişikliğe bağlı olarak iki ayrı sonuç hesaplanmıştır. Bunlardan ilki eşik değerinin öğrenilmesinin yöntem kısmında belirtildiği şekilde yapılması, ikincisi ise sınıflandırıcının potansiyelini göstermesi açısından verilen veri kümesi üstünde alabileceği en iyi eşik değerleri için çalıştırılmasıdır. Eşik değerinin eğitim veri setinden öğrenilmesi geliştirilmeye açık bir problemdir ve en iyi eşik değerinin bulunduğu durumdaki başarımın gösterilmesi aslında sınıflandırıcının potansiyelinin anlaşılması için gereklidir.

Çizelgedeki sonuçlar doğruluk değerine bakılarak en başarılıdan en başarısıza göre sıralanmıştır. Yöntemler arasından alınan en iyi sonuç SÖK-EVET + SVM için alınmış olup 0.887 başarım değerine sahiptir. Bir alt sırada SÖKA-SNF içinde en iyi eşik değerleri kullanıldığında alınmış sonuç bulunmaktadır. Bu tablodaki önemli kıyaslamalardan biri yapılan çalışmalar sonucunda taban durum olarak tanımladığımız ortogonal kodlama ve destek vektör makineleri ile alınan sonuçlardan daha başarılı bir şekilde sınıflandırma yapabilen bir yöntem geliştirmiş olmamızdır.

Ayrıca eşik değerini kendimiz öğrendiğimiz yöntemimiz ise veri madenciliği alanında sıkça kullanılan karar ağacı ve K-en yakın komşu yöntemlerine göre daha başarılı sonuçlar göstermektedir.

Çizelge 5.1: Yöntemlerin 746 veri seti üstündeki sonuçlarının karşılaştırılması

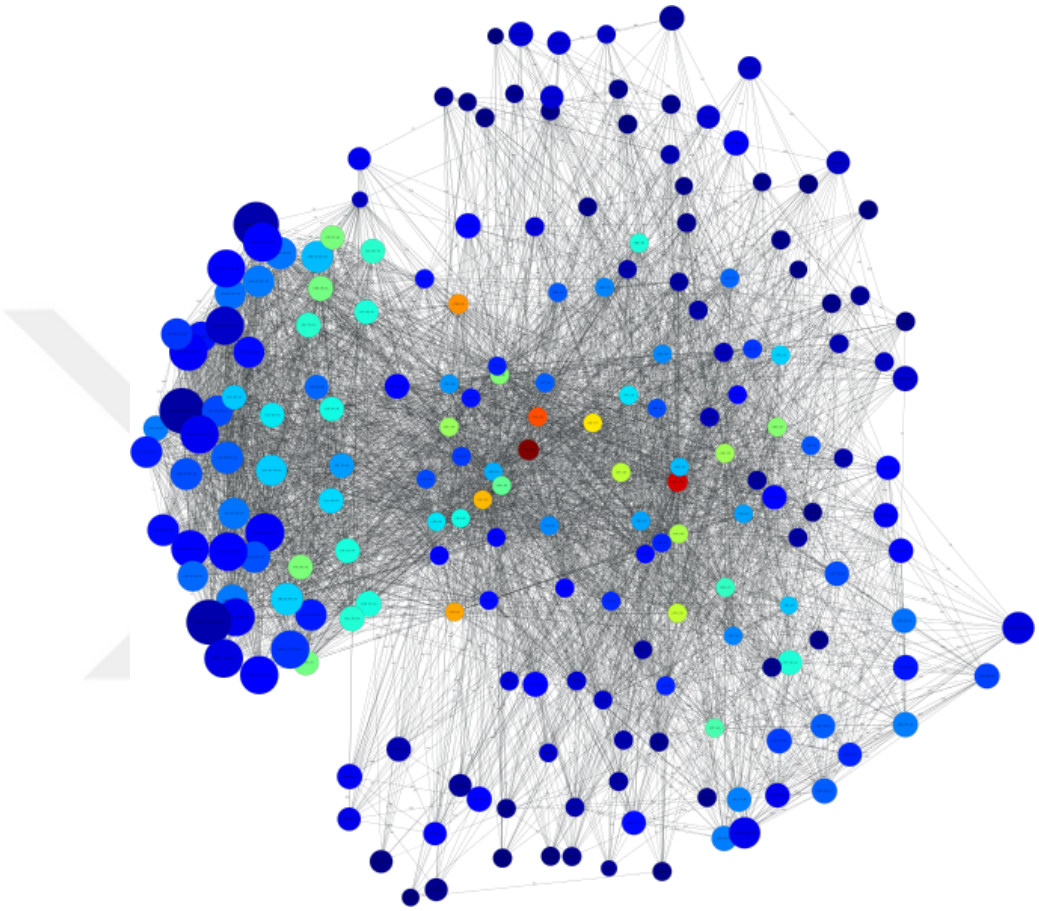
Yöntem	**Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1**
SÖK-EVET + SVM	0.887	0.905	0.897	0.896
SÖKA-SNF en iyi eşik değerleri	0.871	0.872	0.908	0.885
OK + SVM	0.869	0.904	0.860	0.869
SÖKA-SNF	0.835	0.845	0.877	0.852
Karar Ağacı	0.803	0.799	0.836	0.801
5-en yakın komşu	0.788	0.753	0.933	0.823
1-en yakın komşu	0.773	0.763	0.850	0.796



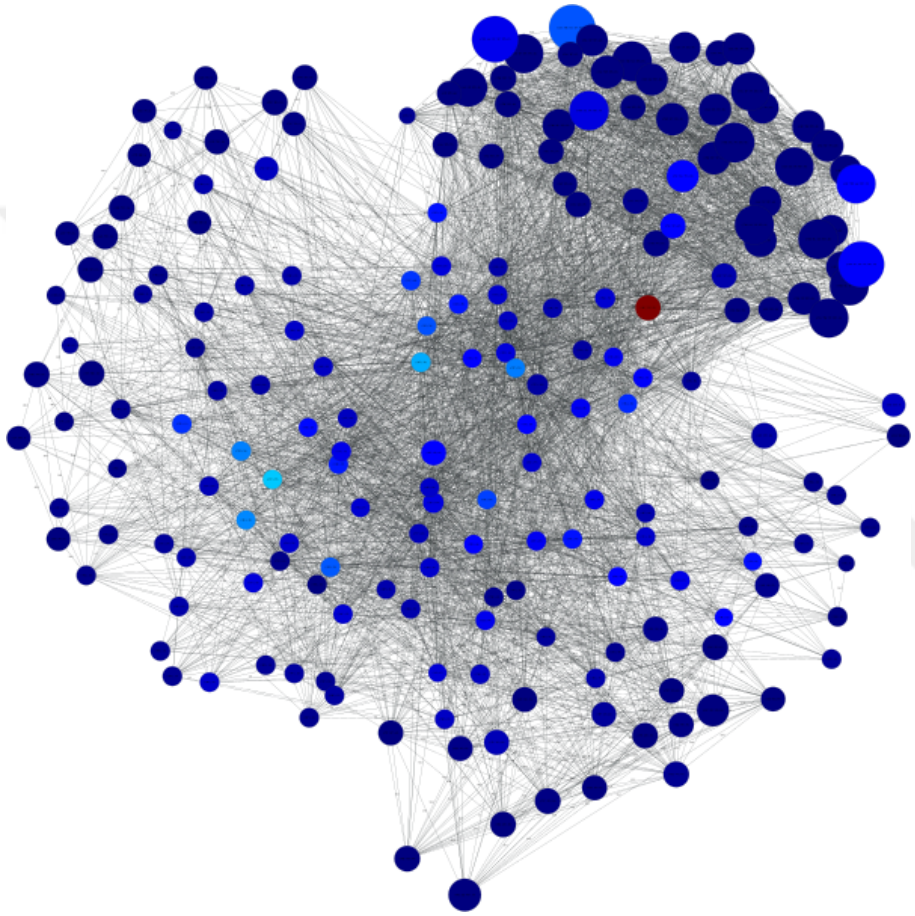
Şekil 5.1: Yöntemlerin 746 veri seti üstündeki sonuçlarının karşılaştırılması

Bu sonuçlar sosyal ağ analizini dikkate alan yenilikçi SÖA-SNF için ümit vericidir. Ağ içindeki düğümlerin merkeziyet skorları ile sık öğe kümelerinin sınıflandırmaya yardımcı olan ayırıcı gücünün birleştirilmesi başarılı tahminler yapılmasını sağlamıştır. Görüldüğü üzere, sık öğe kümelerinden oluşan bir ağın merkezinde olmak oylama sırasında daha etkili oy vermek anlamına gelmektedir çünkü bu beklenen bir durumdur. Normalize sınıf skoru sınıf skorunun tüm veri seti üstünde ölçülmesi ile hesaplanmıştır ve sınıf güveni sık öğe kümelerinin örnekler arasında bulundukları sınıf değerleri dikkate alarak hesaplanmıştır. Tahmin aşamasında örnek ile sık öğe kümesi arasındaki benzerliğin kullanılması, benzerlik değerinin karesi alındığı için başarımı önemli ölçüde etkilemiştir. Eşik değeri yöntemimize belirli bir sınıf için elde edilen sık öğe kümeleri arasındaki destek değeri dengesizliğini aşmak için eklenmiştir.

Her ne kadar problem alanımız biyoinformatik alanında olsa da, geliştirdiğimiz sınıflandırıcı başka alanlardaki problemleri sınıflandırmak için de kullanılabilir. Çünkü elde ettiğimiz doğruluk değeri sınıflandırma işleminde herhangi bir alan bilgisini kullanmamaktadır. Sürekli değerler içeren veri setlerinde ayrıklaştırma işlemi uygulanarak veri seti sık öğe kümelerinin çıkarılmasına uygun hale getirilmelidir. Çok sınıflı problemlerde eşik değeri ikili sınıflar arasında bulunabilir. Bu ikili sınıflar arasında bulunan eşik değerleri farklı eniyileme yöntemleri kullanılarak iyileştirilebilir.



Şekil 5.2: Sık öğe kümesi ağının pagerank merkeziyet değerlerine göre renklendirilmesi. Koyu renkler daha düşük merkeziyet değerini simgelerken açık renkler düğümün daha yüksek merkeziyet değerine sahip olduğunu belirtir.



Şekil 5.3: Sık öğe kümesi ağının arasındalık merkeziyet değerlerine göre renklendirilmesi. Koyu renkler daha düşük merkeziyet değerini simgelerken açık renkler düğümün daha yüksek merkeziyet değerine sahip olduğunu belirtir.

KAYNAKLAR

[1] **Agrawal, R., Srikant, R., et al.** Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB* (1994), vol. 1215, pp. 487–499.

[2] **Alpaydin, E.** *Introduction to machine learning*, 2 ed. MIT press, 2014. 5-9.

[3] **Álvarez, E., Castelló, A., Menéndez-Arias, L., and Carrasco, L.** Hiv protease cleaves poly (a)-binding protein. *Biochemical Journal 396*, 2 (2006), 219–226.

[4] **Borgelt, C.** An implementation of the fp-growth algorithm. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations* (2005), ACM, pp. 1–5.

[5] **Brin, S., and Page, L.** Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks 56*, 18 (2012), 3825–3833.

[6] **Deeks, S. G., Smith, M., Holodniy, M., and Kahn, J. O.** Hiv-1 protease inhibitors: a review for clinicians. *Jama 277*, 2 (1997), 145–153.

[7] **Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P.** From data mining to knowledge discovery in databases. *AI magazine 17*, 3 (1996), 37.

[8] **Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P.** The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM 39*, 11 (1996), 27–34.

[9] **Fayyad, U. M., and Irani, K. B.** The attribute selection problem in decision tree generation. In *AAAI* (1992), pp. 104–110.

[10] **Freeman, L. C.** A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

[11] **Gerenčer, M., and Burek, V.** Identification of hiv-1 protease cleavage site in human c1-inhibitor. *Virus research 105*, 1 (2004), 97–100.

[12] **Gök, M., and Özcerit, A. T.** A new feature encoding scheme for hiv-1 protease cleavage site prediction. *Neural Computing and Applications 22*, 7-8 (2013), 1757–1761.

[13] **Han, J., Pei, J., and Yin, Y.** Mining frequent patterns without candidate generation. In *ACM Sigmod Record* (2000), vol. 29, ACM, pp. 1–12.

[14] **Hearst, M. A., Dumais, S. T., Osman, E., Platt, J., and Scholkopf, B.** Support vector machines. *IEEE Intelligent Systems and their Applications 13*, 4 (1998), 18–28.

[15] **Impens, F., Timmerman, E., Staes, A., Moens, K., Ariën, K. K., Verhasselt, B., Vandekerckhove, J., and Gevaert, K.** A catalogue of putative hiv-1 protease host cell substrates. *Biological chemistry 393*, 9 (Sep 2012), 915–31.

[16] **Kim, G., Kim, Y., Lim, H., and Kim, H.** An mlp-based feature subset selection for hiv-1 protease cleavage site analysis. *Artificial intelligence in medicine 48*, 2 (2010), 83–89.

[17] **Knuth, D. E.** *The Stanford GraphBase: a platform for combinatorial computing*, vol. 37. Addison-Wesley Reading, 1993.

[18] **Kontijevskis, A., Wikberg, J. E., and Komorowski, J.** Computational proteomics analysis of hiv-1 protease interactome. *Proteins: Structure, Function, and Bioinformatics 68*, 1 (2007), 305–312.

[19] **Lam, P., Jadhav, P., Eyermann, C. J., Hodge, C. N., Ru, Y., Bacheler, L. T., Meek, J. L., Otto, M. J., Rayner, M. M., Wong, Y. N., et al.** Rational design of potent, bioavailable, nonpeptide cyclic ureas as hiv protease inhibitors. *Science 263*, 5145 (1994), 380–384.

[20] **Li, X., Hu, H., and Shu, L.** Predicting human immunodeficiency virus protease cleavage sites in nonlinear projection space. *Molecular and cellular biochemistry 339*, 1-2 (2010), 127–133.

[21] **Lichman, M.** UCI machine learning repository, 2013.

[22] **Ma, B. L. W. H. Y.** Integrating classification and association rule mining. In *Proceedings of the fourth international conference on knowledge discovery and data mining* (1998).

[23] **Mannila, H., and Toivonen, H.** Levelwise search and borders of theories in knowledge discovery. *Data mining and knowledge discovery 1*, 3 (1997), 241–258.

[24] **Nanni, L., and Lumini, A.** Using ensemble of classifiers for predicting hiv protease cleavage sites in proteins. *Amino Acids 36*, 3 (2009), 409–416.

[25] **Nie, Z., Bren, G. D., Vlahakis, S. R., Schimnich, A. A., Brenchley, J. M., Trushin, S. A., Warren, S., Schnepple, D. J., Kovacs, C. M., Loutfy, M. R., et al.** Human immunodeficiency virus type 1 protease cleaves procaspase 8 in vivo. *Journal of virology 81*, 13 (2007), 6947–6956.

[26] **Oğul, H.** Variable context markov chains for hiv protease cleavage site prediction. *BioSystems 96*, 3 (2009), 246–250.

[27] **Organization, W. H., et al.** Global health observatory (gho) data. *URL. Available form: http://www. who. int/gho/tb/en* (2015).

[28] **Page, L., Brin, S., Motwani, R., and Winograd, T.** The pagerank citation ranking: bringing order to the web. *Technical Report. from: http://ilpubs.Stanford.edu:8090/422/* (1999).

[29] **Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L.** Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory* (1999), Springer, pp. 398–416.

[30] **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.** Scikit-learn: Machine learning in python. *Journal of Machine Learning Research 12*, Oct (2011), 2825–2830.

[31] **Peixoto, T. P.** The graph-tool python library. *figshare* (2014).

[32] **Peterson, L. E.** K-nearest neighbor. *Scholarpedia 4*, 2 (2009), 1883.

[33] **Quinlan, J. R.** Induction of decision trees. *Machine learning 1*, 1 (1986), 81–106.

[34] **Rögnvaldsson, T., You, L., and Garwicz, D.** State of the art prediction of hiv-1 protease cleavage sites. *Bioinformatics* (2014), btu810.

[35] **Schilling, O., and Overall, C. M.** Proteome-derived, database-searchable peptide libraries for identifying protease cleavage sites. *Nature biotechnology 26*, 6 (2008), 685–694.

[36] **Serrat, O.** Social network analysis. *Knowledge Solutions* (2009), 28.

[37] **Wold, S., Esbensen, K., and Geladi, P.** Principal component analysis. *Chemometrics and intelligent laboratory systems 2*, 1-3 (1987), 37–52.

[38] **You, L., Garwicz, D., and Rögnvaldsson, T.** Comprehensive bioinformatic analysis of the specificity of human immunodeficiency virus type 1 protease. *Journal of virology 79*, 19 (2005), 12477–12486.

ÖZGEÇMİŞ

Ad-Soyad : Yunuscan KOÇAK

Uyruğu : T.C

Doğum Tarihi ve Yeri : 16.02.1992 ANKARA

E-posta : y.kocak@etu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** : 2014, TOBB ETÜ, Mühendislik Fakültesi, Bilgisayar Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

Yıl	Yer	Görev
2015-2016	TOBB ETÜ	Tam Burslu Yüksek Lisans Öğrencisi
2014-2015	TOBB ETÜ	Araştırma Burslu Yüksek Lisans Öğrencisi
2014-2015	Harezmi Bilişim Çözümleri	Bilgisayar Mühendisi

YABANCI DİL: İngilizce, İspanyolca

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- Kocak, Y., Özyer, T., Alhajj, R. "Classification of HIV data By Constructing A Social Network with Frequent Itemsets." Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2016. ACM, 2016