MAKİNE ÖĞRENMESİ SINIFLANDIRMA ALGORİTMALARININ İNCELENMESİ

ADEM KÜRŞAT KESKİN

YÜKSEK LİSANS TEZİ

MATEMATİK ANABİLİM DALI

T.C.

SİNOP ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ SINIFLANDIRMA ALGORİTMALARININ İNCELENMESİ

ADEM KÜRŞAT KESKİN

MATEMATİK ANABİLİM DALI

DANIŞMAN

Doç. Dr. B. BARIŞ ALKAN

SİNOP – 2018

İÇİNDEKİLER

Sayfa

KISALTMALAR LİSTESİ

BO	: Belirleyicilik Oranı
DO	: Doğruluk Oranı
DOR	: Duyarlılık Oranı
HO	: Hata Oranı
KNN	: K-En Yakın Komşuluk Sınıflandırıcısı
NBC	: Naive Bayes Sınıflandırıcısı

ŞEKİLLER

ÇİZELGELER

MAKİNE ÖĞRENMESİ SINIFLANDIRMA ALGORİTMALARININ İNCELENMESİ

ÖZET

Bu çalışmada, literatürde her birinin kendi içinde karmaşık bir teoriye sahip oldukları görülen, Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu sınıflandırma algoritmalarının R programında yazılan kodlar yardımıyla, farklı veri tiplerine uygulanması ile daha anlaşılır ve açık bir şekilde sunulması hedeflenmektedir. Ayrıca gerçek bir veri kümesi üzerinden, son dönemlerde makine öğrenmesi yöntemleri için son kullanıcı dostu yeniliklerle ön plana çıkan KNIME programı yardımıyla yukarıda bahsedilen dört makine öğrenme sınıflandırma algoritmasının bir performans karşılaştırması yapılmıştır. Çalışmanın birinci bölümünde, konuya giriş yapılmış olup, ikinci bölümde ise büyük veri, makine öğrenmesi, performans değerlendirme yöntemleri, sınıflandırma ve sınıflandırma başarı ölçüleri hakkında temel kavramlardan bahsedilmiştir. Daha sonraki bölümlerde sırasıyla, Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu sınıflandırma algoritmaları için gerekli matematiksel kavramlar verilmekte ve bu yöntemlerin kullanımının önemi vurgulanmaktadır. Çalışmanın dördüncü bölümünde ise farklı veri tipleri kullanılarak önceki bölümlerde ele alınan sınıflandırma algoritmalarının R programı uygulamalarına ve KNIME programı ile analiz edilen gerçek bir veri kümesi üzerinden bir performans karşılaştırması uygulamasına yer verilmektedir. Son bölümde ise çalışmadan elde edilen sonuçlar tartışılacaktır.

Anahtar Kelimeler: Sınıflandırma Algoritmaları, Makine Öğrenme, Büyük Veri.

INVESTIGATION of MACHİNE LEARNING CLASSIFICATION ALGORITHMS

ABSTRACT

In this study, it is aimed to present in a clearer and clear with the application of Decision Tree, Naive Bayes, Random Forest and K-nearest neighboring classification algorithms which are seen to have a complex theory within the literature, to different data types with the help of codes written in R program. In addition, a performance comparison of the above-mentioned four machine learning classification algorithms has been made with the help of the KNIME program, which has recently come to the forefront with end-user-friendly innovations for machine learning methods over a real data set. In the first section of the study, an introduction was made and in the second part, basic concepts about big data, machine learning, performance evaluation methods, classification and classification success measures were mentioned. In the following chapters, mathematical concepts for Decision Tree, Naive Bayes, Random Forest and K-nearest neighboring classification algorithms are given and the importance of using these methods is emphasized. In the fourth part of the study, using different data types, R applications of classification algorithms discussed in previous chapters and a performance comparison in KNIME is made over a real dataset. In the last section, the results of the study are discussed.

Key Words: Classification Algorithms, Machine Learning, Big Data.

TEŞEKKÜR

Bu tez çalışmasının konusunun belirlenme, hazırlanma ve tamamlanma aşamasında desteklerini benden bir an olsun esirgemeyen danışmanım Sayın Doç. Dr. B. Barış ALKAN'a sonsuz saygı ve teşekkürlerimi sunarım.

Fakültemiz Dekanı ve Matematik Anabilim Dalı Başkanı Prof. Dr. Kamil DEMİRCİ'ye yüksek lisans sürecim boyunca bana vermiş olduğu desteklerden dolayı saygı ve teşekkürlerimi sunarım.

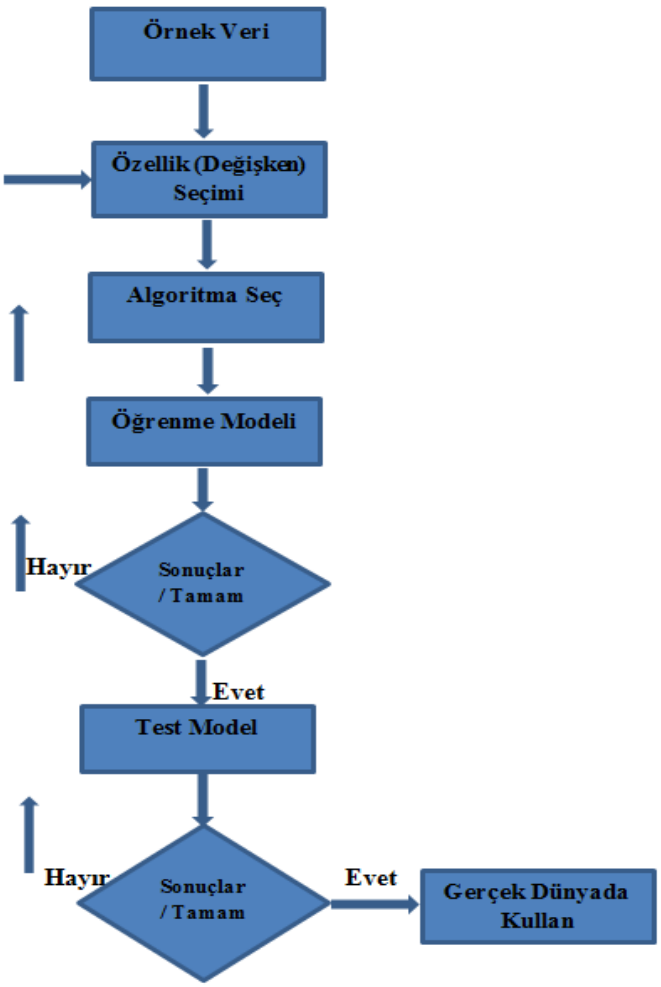
Yüksek Lisansa başlamama vesile olan Sayın Doç. Dr. Alper SİNAN'a bana vermiş olduğu fikir ve yönlendirmelerden dolayı teşekkürlerimi sunarım.

Ayrıca çalışmalarım boyunca beni cesaretlendiren Aylin ŞENOL'a, desteğini esirgemeyen ve vakit ayıran Dr. Öğr. Üyesi Yılmaz Mehmet DEMİRCİ, Dr. Öğr. Üyesi M. Duygu SAÇAR DEMİRCİ, Arş. Gör. Dr. Cumhur AVŞAR'a, ev arkadaşlarım Nuri SAĞOL, Kenan KAHİL, Koray SARIOĞLU'na ve sevgili annem, babam ve kardeşlerime sonsuz teşekkür ederim.

1. GİRİŞ

Makine Öğrenmesi, mevcut standart tahmin modellemesine alternatif bir yaklaşım sağlar ve büyük veriyi daha iyi açıklayan bir dönüştürme potansiyeline sahiptir (Obermeyer, 2016). Makine Öğrenmesi, hesaplamalı öğrenme ve yapı tanımlama çalışmalarında geliştirilmiştir. Bu çalışmalar, tahmin edilen ve gözlenen sonuçlar arasındaki hatayı en aza indirerek değişkenler arasındaki tüm karmaşık ve doğrusal olmayan etkileşimleri öğrenmek için yapılan bir hesaplamaya dayanır (Dreiseitl ve Ohno-Machado, 2002). Potansiyel olarak tahmini iyileştirmesinin yanı sıra, gözlenmesi mümkün olmayan fakat diğer değişkenlerden çıkarılabilen gizli değişkenleri de belirleyebilir (Berglund ve ark., 2013). Makine Öğrenmesi, herhangi bir veri kümesinin iç doğasının kavranmasını sağlayan algoritmaların bir topluluğudur. Veri kümesinin iç doğasının kavranması bir karar alma sürecinde insanlara ya da diğer makineler tarafından kullanılabilir. Örneğin, Nottingham üniversitesinde ilk bakım araştırmacıları ve bilgisayar bilim insanları kardiyovasküler hastalık riski değerlendirmesi için Makine Öğrenmesi algoritmaları kullanmışlardır. Bu Makine Öğrenmesi algoritmalarının kalp krizi riskini belirlemede deneyimli tıp doktorlarından daha iyi olduğunu tespit etmişlerdir (Weng ve ark., 2017). Makine Öğrenmesi, önleyici tedaviden yararlanabilecek kişilerin sayısını artırırken kalp krizi risk tahmininin doğruluk oranını önemli ölçüde iyileştirmektedir. Ayrıca, yanlış ön teşhis sonucu ortaya çıkabilecek gereksiz tedavilerden kaçınılmasını sağlayabilmektedir (Weng ve ark., 2017). Makine Öğrenmesi genellikle iki temel amaç için kullanılır. Bunlardan birincisi gelecekteki sonuçları öngörmektir. Örneğin, yoğun bir kavşakta trafik ışığı dizilerini optimize etmek istendiğinde, bir Makine Öğrenmesi algoritması beş dakika ilerideki trafik akışını kestirmek için geliştirilebilir. İkinci görevi ise nesnelerin özel gruplara sınıflandırılmasıdır. Örneğin, kahve çekirdekleri genellikle dört sınıftan biri olarak sınıflandırılır. Endüstriyel bir kahve çekirdeği alıcısı kahve çekirdeği kalitesini otomatik olarak derecelendiren bir Makine Öğrenmesi algoritması geliştirebilir.

Şekil 1.1 Makine Öğrenmesi modelleri geliştirmek için kullanılan tipik bir akış diyagramını göstermektedir.



Şekil 1.1. Makine Öğrenmesi modelleri geliştirmek için akış diyagramı

Eğer sınıflandırma ya da tahmin performansı kabul edilebilir bir seviyede ise, model ayrıca bağımsız bir test kümesi üzerinden doğrulanır. Bu, model tarafından daha önce görülmemiş orijinal örnek verisinin bir alt örneğidir. Tüm süreç iteratiftir. Genel olarak nihai bir model seçilmeden önce birçok yenileme gereklidir. Bu model ya da modeller yeni model tarafından daha önce görülmemiş modelden elde ettiğimiz verilere uygulanır.

Bu çalışmada, literatürde her birinin kendi içinde karmaşık bir teoriye sahip oldukları görülen, Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu sınıflandırma algoritmalarının R programında yazılan kodlar yardımıyla farklı veri tiplerine uygulanması ile daha anlaşılır ve açık bir şekilde sunulması hedeflenmektedir. Ayrıca gerçek bir veri kümesi üzerinden, son dönemlerde makine öğrenmesi yöntemleri için son kullanıcı dostu yeniliklerle ön plana çıkan KNIME programı yardımıyla yukarıda bahsedilen dört makine öğrenme sınıflandırma algoritmasının bir performans karşılaştırılması yapılmıştır.

Çalışmanın birinci bölümünde, konuya giriş yapılmış olup, ikinci bölümde ise büyük veri, makine öğrenmesi, performans değerlendirme yöntemleri, sınıflandırma ve sınıflandırma

başarı ölçüleri hakkında temel kavramlardan bahsedilmiştir. Daha sonraki bölümlerde sırasıyla, Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu sınıflandırma algoritmaları için gerekli matematiksel kavramlar verilmekte ve bu yöntemlerin kullanımının önemi vurgulanmaktadır. Çalışmanın dördüncü bölümünde ise farklı veri tipleri kullanılarak önceki bölümlerde ele alınan sınıflandırma algoritmalarının R programı uygulamalarına ve KNIME programı ile analiz edilen gerçek bir veri kümesi üzerinden bir performans karşılaştırması uygulamasına yer verilmektedir. Son bölümde ise çalışmadan elde edilen sonuçlar tartışılacaktır.

2. TEMEL KAVRAMLAR

2.1. Büyük Veri

Büyük veri, geleneksel veri işleme uygulamaları yetersiz kaldığında çok geniş veya karmaşık veri kümeleri için yaygın bir terimdir. Büyük veri durumunda karşılaşılan zorluklar arasında analiz, veri iyileştirme, arama, paylaşma, depolama, veri aktarımı ve bilgi gizliliği gibi konular başı çekmektedir. Büyük veri teriminin gerçek anlamı, veriden değer elde etmek için tahmin analizleri veya diğer ileri yöntemlerin kullanımıyla ilgilidir.

Büyük verideki doğruluk, daha güvenli karar vermeyi sağlayabilir. Daha iyi kararlar, daha yüksek operasyonel verimlilik, maliyet azaltma ve daha az risk anlamına gelmektedir. Büyük veri kümelerinin analizi ile iş trendlerini tespit etmek, hastalıkları önlemek, suçla mücadele etmek ve benzeri yeni ilişkiler bulunabilir. Bilim insanları, şirket yöneticileri, medya ve reklam uygulayıcıları ve hükümetler düzenli olarak internet arama, finans ve işletme bilişimi gibi alanlarda büyük veri kümeleri ile karşılaştıklarında bir takım zorluklarla yüzleşmektedirler. Veri kümeleri her geçen gün daha da büyümektedir. Çünkü ucuz ve sayısız bilgi verici mobil cihazlar, uzaktan algılama araçları, yazılım kayıtları, kornalar, mikrofonlar, radyo frekansı tanımlama okuyucuları ve kablosuz sensör ağları tarafından giderek daha fazla bilgi toplanmaktadır.

Dünyanın teknolojik bilgi depolama kapasitesi, 1980'lerden bu yana her 40 ayda birkaç katına çıkmıştır (Hilbert ve Lopez, 2011). 2012'den itibaren her gün 2.5 exabyte veri oluşturulmuştur. Büyük veri ile çalışmak ister istemez yaygın değildir. Birçok analiz standart veri kümelerinin çözümlenmesinde klasik notebook ve kişisel bilgisayarlar kullanılarak yapılır. İlişkisel veri tabanı yönetim sistemleri ve masaüstü istatistikleri ve görselleştirme paketleri genellikle büyük verileri ele almakta zorlanır. Büyük veri için, bu standart paketler yerine onlarca, yüzlerce ve hatta binlerce sunucu üzerinde çalışan çok paralel bir yazılım gerekmektedir (Jacobs, 2009).

Büyük veri genellikle, verileri tolere edilebilir bir zaman içerisinde yakalamak yönetmek ve işlemek için yaygın olarak kullanılan yazılım araçlarının gücünün üzerinde boyutlara sahip veri kümelerini içerir (Snijders ve ark., 2012). Büyük veri; Farklı, karmaşık ve büyük ölçekli veri kümelerinden gizli kalmış bilgileri ve değerleri ortaya çıkarmak için gerekli olan teknolojiler ve tekniklerin bir kümesidir (Hashem ve ark., 2015).

2013 yılında Gartner tarafından "Büyük Veri; yüksek karar verme, iç görü keşfi ve süreç optimizasyonu sağlamak için yeni veri işleme biçimleri gerektiren yüksek hacimli yüksek hızlı ve yüksek çeşitlilikte bilginin var olmasıdır." şeklinde tanımlanmıştır.

2.2. Makine Öğrenmesi

Birçok bilim uzmanı ve araştırmacılar tarafından şu an içinde yaşadığımız çağ "Büyük Veri" çağı olarak adlandırılmaktadır(Lohr,2012). Teknolojik gelişmenin başladığı ilk dönemlerde genellikle firmaların veya şirketlerin verilerinin depolandığı ve işlendiği merkezler vardı. Kişisel bilgisayarların kullanılmaya başlanmasıyla ve sonraki dönemlerde teknolojik ilerlemenin çok hızlı bir şekil atağa geçmesiyle, kablosuz iletişim gibi birçok yeni kavramın hayatın içine girdiği görülmektedir. Bu yeni kavramlar ve araçların sayesinde nerdeyse her bir birey veri üreticisi durumuna gelmiştir. Örneğin, bir kişi, bir ürünü satın aldığında, sinemada bir film tercihinde bulunduğunda, spor yaparken, beğenilerini veya fikirlerini sosyal iletişim programları üzerinden paylaştığında, yani günlük aktivitesi içerisinde sürekli veri üretmektedir. Her birey veri üreticisi olduğu kadar aynı zamanda da iyi bir veri tüketicisidir. Ayrıca, her birey ihtiyaçlarının en üst seviyede anlaşılması ve çıkarlarının önceden tespit edilmesini arzular. Örneğin, milyonlarca müşteriye hitap eden ve internet üzerinden ürün satan bir market zinciri ele alınsın. Bu marketten yapılan her işlemin kayıtları (tarih, müşteri kimliği, satın alınan mallar ve onların miktarı, harcanan toplam para, hangi kredi kartı ile alışveriş yapıldığı, müşterinin alışveriş sıklığı vb.) tutulur. Bu market zinciri, hangi müşterinin hangi ürünü satın alacağını öngörmek, satış ve karı en üst düzeye taşımak istemektedir. Benzer bir şekilde, her müşteri de kendi ihtiyaçlarını karşılayan en iyi ürün tercihini yapmak istemektedir. Zincir market karar vericileri veya ürün pazarlamacıları, örneğin, müşterinin bir yazarın bir sonraki kitabını satın alacağını tam olarak bilemez. Müşteri davranışı zaman içinde ve coğrafi konuma göre değişir. Fakat bu durumun, tamamen rastgele olmadığı araştırmacılar tarafından bilinmektedir. Örneği, kola aldıklarında cips alırlar. Verilerde belirli modeller var. Bilgisayar ile bir problemi çözmek için bir algoritmaya ihtiyaç vardır. Algoritma, girişi çıktıya dönüştürmek için yapılması gereken bir talimat dizisidir. Örneğin, bir sıralama için bir algoritma tasarlanabilir. Giriş bir sayı kümesidir ve çıktı onların sıralı listesidir. Aynı görev için çeşitli algoritmalar olabilir ve en az sayıda yönerge veya bellek veya her ikisini de gerektiren en verimli olan dizilim bulmak istenir. Fakat bazı görevler için, bir algoritma mevcut değildir. Örneğin, müşteri davranışını tahmin etmek ve spam e-postalarını meşru olanlardan ayırmak sıkıntılı bir süreçtir. Burada, giriş bir karakter dosyası olan bir e-posta belgesi, çıkış ise, mesajın spam olup olmadığını belirten bir evet / hayır çıktısıdır. Fakat girişi çıktıya nasıl dönüştürüleceği aşaması bilinmemektedir. Çünkü spam değişiklikleri, zaman içinde ve bireyden kişiye değişmektedir. Bu nedenle bilginin ulaşılabilir olmadığı veya var olmadığı durumlarda, bu bilgiye verilerden ulaşılmaya çalışılmaktadır (Alpaydin, 2016). Spam olduğu ve olmadığı bilinen binlerce örnek ileti

kolayca derlenebilir ve burada öğrenilmek istenen cevap, spam'i neyin oluşturduğudur. Bu, makine öğrenmesi işidir.

Makine öğrenmesi yöntemlerinin büyük veri tabanlarına uygulanması veri madenciliği olarak adlandırılır. Buradaki benzerlik, büyük miktarda toprak ve hammaddenin bir madenden çıkarılmasıdır; işlendiğinde, çok az miktarda değerli malzemeye yol açar. Veri madenciliğinde, yüksek tahmin doğruluğuna sahip, basit bir model oluşturmak için büyük miktarda veri işlenir (Alpaydin, 2016). Makine öğrenmesi uygulama alanları, finans bankalarında, kredi uygulamalarında, tıbbi teşhis alanında, dolandırıcılık tespitinde, eğitim-öğretim planlamasında ve borsada kullanılacak modelleri oluşturmak için geçmiş verilerini analiz eder. Makine öğrenmesi sadece bir veri tabanı problemi olmayıp, aynı zamanda yapay zekanın bir parçasıdır. Değişen bir ortamda bulunan bir sistem öğrenebilme yeteneğine sahip olmalıdır. Sistem bu değişiklikleri öğrenebilir ve uyarlayabilirse, sistem tasarımcısı tüm olası durumlar için öngörme ve çözüm sunma ihtiyacı duymaz.

Makine öğrenmesi, örnek verileri veya geçmiş deneyimleri kullanarak bir performans ölçütünü optimize etmek için bilgisayarları programlamaktır. Matematiksel modellerin oluşturulmasında istatistik teorisini kullanır, çünkü temel görev bir örneklemden çıkarım yapmaktır.

Makine öğrenmesinde bilgisayar biliminin iki önemli rolü vardır. Bunlardan birincisi, eğitim aşamasında, optimizasyon problemini çözmek için etkili algoritmalara ve genel olarak sahip olunan yüksek miktarda veriyi depolama ve işleme ihtiyacıdır. İkincisi, bir model öğrenildikten sonra, onun çıkarımı için temsili ve algoritmik çözümün de etkin olması gerekir (Alpaydin, 2016).

2.2.1. Makine Öğrenmesi Yöntemleri

Makine öğrenmesi yöntemleri genel literatürde, danışmanlı ve danışmansız öğrenme olarak ikiye ayrılmaktadır. Danışmanlı öğrenmede, danışman öğrenilmesi istenen olguya ait örneklerin her biri için yer alan girdileri ve o değerlere karşılık gelen çıktı değerleri sisteme tanıtır. Girdileri danışmanın belirlediği çıktılara göre şekillendirmek sistemin işidir. Bu yolla girdi-çıktı arasındaki ilişkiler öğrenilmektedir. Danışmansız öğrenme ise, herhangi bir danışmanın bulunmadığı, sadece girdi değişkenlerine sahip gözlemlerden oluşan öğrenme yöntemidir.

2.2.2. Makine Öğrenmesi Algoritması Performans Değerlendirme Yöntemleri

Makine öğrenmesi algoritmasının performansı eğitim ve test kümelerinin büyüklüğüne ya da hatalı sınıflandırma oranı ile değerlendirebilmektedir. Aşağıda performans değerlendirme yaklaşımları kısaca açıklanmıştır.

2.2.2.1. Holdout Yöntemi

Holdout yöntemi, mevcut veri örneğini iki alt kümede rastgele ayırmaktan oluşur: biri modeli eğitmek için kullanılır ve diğeri bunu test için etmek kullanılır. Sık kullanılan bir oran, eğitim için % 70 ve test için % 30'dur. Eğer küçük bir veri örneği varsa, ya çok küçük bir test kümesi ya da eğitim setinden çok fazla veri çıkartma tehlikesi vardır. Yani bu yöntem, tipik olarak sadece çok büyük veri kümeleri için tercih edilen bir yöntemdir (Torgo, 2016).

2.2.2.2. Çapraz Geçerleme Yöntemi

Çapraz geçerleme, literatürde iki şekilde uygulanır. Bunlar k-kat (k-fold) çapraz geçerleme ve birini dışarıda bırak (leave one out) çapraz geçerleme olarak adlandırılır. k-kat çapraz geçerleme, bir modelin kestirim performansının tahmin edilmesinde en yaygın kullanılan yöntemlerden biridir. Bu yöntemde, veri kümesi başlangıçta k eşit parçaya bölünür. Elde edilen bu k parçadan her seferinde bir tanesi test için kullanılırken, k-1 tanesi ise eğitim için kullanılmaktadır. Sonuç olarak k tane hata oranı bulunur ve genel tahmin hatasını hesaplanması için elde edilen hataların ortalaması alınmaktadır (Refaeilzadeh ve ark., 2009; Torgo, 2016).

2.2.3. Sınıflandırma

Sınıflandırma, girdi olarak tanımlanan her bir vektörü birbirinden bağımsız farklı gruplara atamayı amaçlamaktadır. Sınıflandırma probleminin çözümü için yazılan algoritmalar sınıflandırıcı olarak adlandırılmaktadır (Alpaydın, 2016; Bishop, 2007; Camastra ve Vinciarelli, 2008).

2.2.4. Sınıflandırma Başarı Ölçütleri

Sınıflandırıcıların birbirlerine göre üstünlüklerinin denetlenebilmesi için bazı ölçütler geliştirilmiştir. Bu ölçütler hata matrisi (confusion matrix) oluşturulması temeline dayanmaktadır. Gerçek çıktı değerleri ile modelden elde edilen tahmin değerlerinin oluşturduğu temsili tablo Çizelge 2.1'de verilmiştir (Balaban ve Kartal, 2015).

Çizelge 2.1. Hata Tablosu

			GERÇEK		
TAHMİN			**POZİTİF**	**NEGATİF**	**TOPLAM**
		POZİTİF	Doğru Pozitif (DP)	Yanlış Pozitif (YP)	TopP
		NEGATİF	Yanlış Negatif (YN)	Doğru Negatif (DN)	TopN
		TOPLAM	Pozitif (P)	Negatif (N)	Genel Toplam (G)

Çizelge 2.1'e göre sınıflandırma başarı ölçütleri aşağıdaki eşitlikler yardımıyla hesaplanır. Sınıflandırmanın doğruluğu (accuracy) ve hata oranı (error rate) aşağıdaki eşitliklerle bulunmaktadır.

$$Doğruluk\ Oranı\ (DO) = \frac{DP+DN}{G}$$

$$Hata\ Oranı\ (HO) = 1 - DO$$

Sınıflandırıcının pozitif sınıf etiketleri tahmin etmede sınıflandırıcının etkinliği duyarlılık oranı ile verilmektedir:

$$Duyarlılık\ Oranı\ (DOR) = \frac{DP}{P}$$

Sınıflandırıcının negatif sınıf etiketleri tahmin etmede sınıflandırıcının etkinliği belirleyicilik oranı ile verilmektedir:

$$Belirleyicilik\ Oranı\ (BO) = \frac{DN}{N}$$

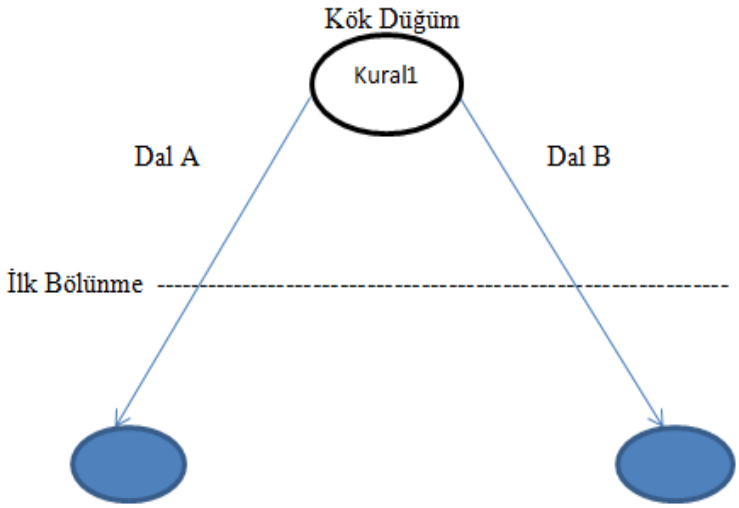
3. MATERYAL VE YÖNTEMLER

3.1. Karar Ağaçları

Karar ağaçları, karar modelinin en basit formlarından biri olup, bir ağaç yapısı şeklinde olan kurallarını oluşturmak için örnek veri özelliklerini kullanırlar. Karar ağaçlarının kökenleri insanların karar verme yoluna benzer. Karar ağaçları bilgiyi görsel olarak daha anlaşılır biçimde sundukları için oldukça popülerdir. Hem regresyon hem de sınıflandırma problemleri için kullanılabilir. Karar ağaçları bir sınıflandırma kararı vermek için kullanılabilen kontrol edilebilir bir kurallar kümesine bir veri örneğini indirgeyebilirler (Bohanec ve Bratka,1994; Lewis, 2017).

3.1.1. Böl ve Yönet Algoritması

Karar ağaçları düğümlere aşamalı olarak özellikler ve ilişkili bir kural eklenerek oluşturulur. Karar ağaçları genellikle böl ve yönet algoritması kullanılarak aşağıya doğru inşa edilirler. Şekil 3.1'de gösterildiği gibi temel yaklaşım ağacın üst düğümünde başlar ve orada veri özelliklerinden biri tarafından bir kural kullanılarak bölünür.



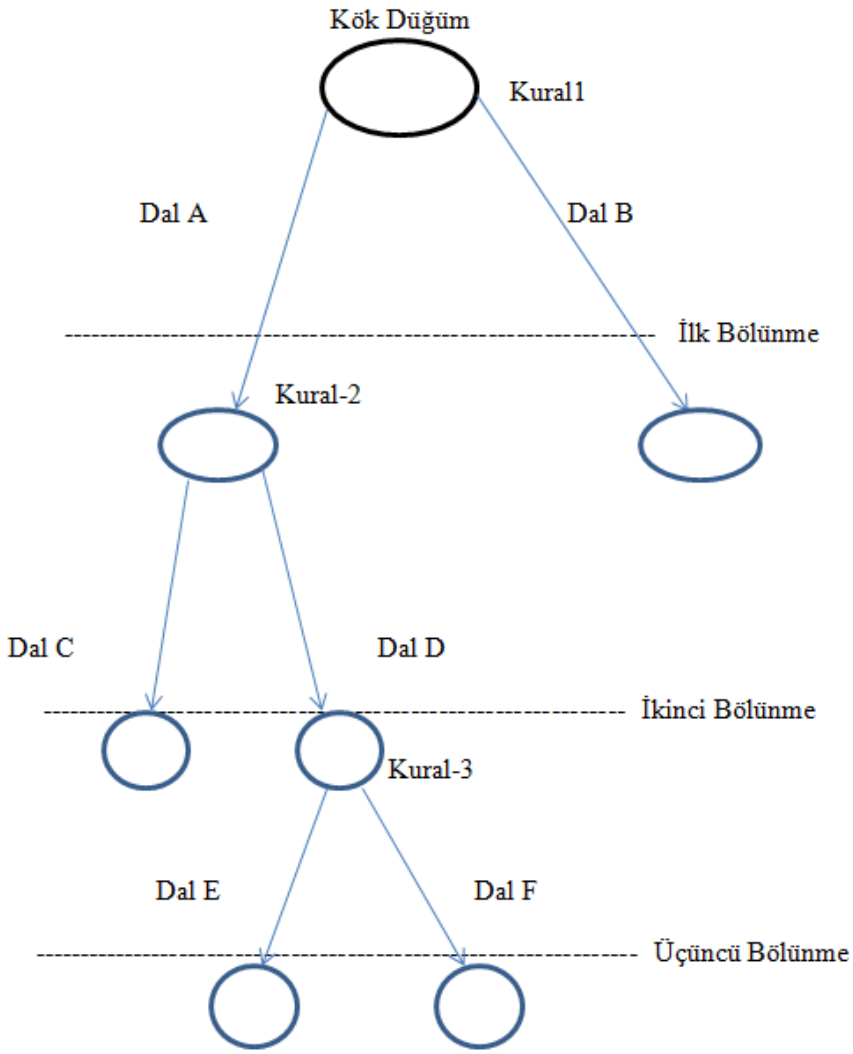
Şekil 3.1. Kök Düğüm ve bir Karar Ağacının ilk bölünmesi

Amaç; veriyi doğru sınıflandırılmış örneklere bölen bir özellik ve kural seçmektir. Bu süreç, sınıflandırma için en kullanışlı olan özelliğin tanımlanması ve bu özelliği kullanarak bir karar kuralı elde etmeyi içerir.

Tüm özellikler, hangi özelliğin verinin bölünmesi için en iyi olduğunun görülmesi için test edilir. İdeal olarak basit bir kural eğitim örneklerini kendi sınıflarına ayırır. Fakat çoğu durumda istenen şekilde gelişmez ve mümkün olduğu kadar net bir şekilde örnekleri ayıran

bir kural seçilir. Bu Şekil 3.1'deki Kural-1'dir. Bu aşamadan sonra ağaç, ilişkili düğümlere sahip iki yeni daldan oluşur.

Şekil 3.2'de görüldüğü gibi yeni dalın düğümleri daha sonra belirlenir. Eğer bu bölümde, bir düğümdeki tüm örnekler aynı sınıfa ait ise o zaman durulur ve bu sınıfın ismi ile her bir yaprak düğümü etiketlenir.



Şekil 3.2. Kök Düğüm ve Karar Ağacının Bölünmeleri

Verilerin sınıflandırma saflığını daha da iyileştirmek için örneklerin bölünmesini sağlayan bu ağaç oluşturma süreci her bir dal için tekrarlanır. Süreç bir durdurma kriterine kadar devam eder ya da düğümler bölümündeki tüm durumlar doğru bir şekilde sınıflandırılınca sona erer. Tipik bir durdurma kriteri, yeni özellik-kural kombinasyonları, alt grupların saflığını çok küçük miktarda arttırdığında verilerin bölünmesini durdurmaktır.

3.1.2. Karar Ağacı Yöntemlerinin Avantajları

Karar ağaçları, karar verme sürecindeki her adımı yakalar. Başka bir deyişle kurallar bir gözlemin sınıflandırılması için yol gösterir. Bir karar ağacı, bir dizi test sorusu ve koşulları olarak oluşturulduğu için kolay anlaşılır olacak bir şekilde sezgisel olarak net kurallarla tasarlanabilir. Karar ağaçları parametrik olmayan bir yöntemdir. Bu özelliklerin veya sınıfların olasılık dağılımı hakkında varsayımlar gerektirmediği anlamına gelir. Karar ağaçları kategorik ve sayısal veri kümeleri için kullanılabilirler. Ayrıca veri kümesinde yer alabilecek aykırı gözlemlere karşı dayanıklıdırlar ve kayıp değerlere sahip veri kümelerinin analizinde kullanılabilirler. Karar ağaçları hesaplama maliyeti açısından ucuzdur ve büyük veri kümeleri ile iyi çalışır. Son olarak, bir kez inşa edildikten sonra karar ağaçları yeni gözlemleri hızlı bir şekilde sınıflandırırlar.

3.2. Naive Bayes

Naive Bayes sınıflandırıcı (NBC) Bayes teoremini temel alan basit bir olasılık sınıflandırıcısıdır (Narayanan ve ark.,2003). Değişken sayısının çok fazla olduğu durumlarda oldukça kullanışlıdır. Thomas Bayes'in 18. Yüzyıldaki çalışmasına uzanan köklere sahip olan NBC, 1950'lerden beri uygulanmaktadır. NBC, sınıflandırma için basit, güçlü ve yüksek etkili bir araçtır. NBC, denetimli bir sınıflandırıcıdır. Danışmanlı sınıflandırma algoritmalarının iki genel tipi vardır. Bunlar;

1) Bayes Kuralının kullanımı yoluyla algoritmaları kullanan sınıflandırma,

2) Sınıflandırma sınırlarının doğrudan veriden öğrenildiği diskriminant analizi yoluyla sınıflandırma.

Naive Bayes, üretici bir sınıflandırıcıdır (Lewis, 2017).

3.2.1. Bayes Kuralının Temel Kavramı

Üretici algoritmalar bir fikir edinmenin en kolay yolu, küçük olasılık ve Bayes kuralını kullanmaktadır. Adını, İngiliz istatistikçi ve filozof olan Thomas Bayes'den alan Bayes kuralı, A ve B iki rasgele değişkeni verildiğinde, B verildiğinde A'nın koşullu olasılığını;

$$P(A/B)= P(B/A)P(A)/P(B)$$

eşitliği ile hesaplanabileceğini ifade etmektedir. P(A/B) sonsal olasılık olarak yaygın bir şekilde ifade edilirken, P(A) ve P(B) önsel olasılıklar ve P(B/A) ise olabilirlik olarak ifade edilir.

3.2.2. Üretici Algoritma

Bir Y sınıf değişkeni ve X açıklayıcı özellikleri, Y sınıf etiketlerinin bir kümesi ve X açıklayıcı değişkenlerin bir kümesi verildiğinde, X ve Y'nin ortak olasılık dağılımını açıkça belirtir. Bu ortak olasılık dağılımı P(y,n) ile gösterilsin. y verildiğinde n'in koşullu, olasılığı ise P(n/y) ile gösterilsin. Buradan Bayes kuralı kullanılarak, P(y/n);

$P(y/n)=P(n/y)P(y)/\int P\left(\frac{n}{y}\right)P(y)dy$

eşitliği ile elde edilir.

Burada, P(n/y)P(y), n ve y'nin ortak dağılımını ifade eder.

3.2.3. Ayırımcı Sınıflandırıcılar

Ayırımcı algoritmalarda, P(y/n) koşullu dağılımı doğrudan modellenmiştir. Bu sınıflandırma için gereklidir. Ayırımcı algoritmalar sadece koşullu dağılımı modellediğinden, üretkenliğe dayalı modellerden çok daha basit bir yapıya sahip olabilirler. Bazı ayırımcı sınıflandırıcılar, odaklarını belirli değerlerle sınırlandırarak işleri daha da kolaylaştırırlar.

Örneğin; Destek vektör makineleri, P(n/y)'nin 0,5'den daha küçük veya büyük olup olmadığına odaklanırlar.

3.2.4. Bayes Sınıflandırıcı

Bayes sınıflandırıcı sınıflandırma problemlerini çözmek için kullanılabilen bir üretici sınıflandırıcıdır. X=$[x_1, \ldots\ldots, x_n]$ özelliklerin bir giriş vektörü olarak K sınıflarıda C=$\{c_1, \ldots\ldots, c_k\}$ile gösterilsin. Bir Bayes sınıflandırıcı, X özelliklerinin bir kümesini göz önüne alarak verilen bir gözlem için, en büyük olasılıkla c_i sınıf değerini tahmin etmek için bir karar kuralıdır. Bayes sınıflandırıcı Bayes kuralını kullanılır.

$P(c_i/n)= P(c_i)P(n/c_i)$ (3.2.4.1)

Bayes kuralına göre Bayes sınıflandırıcı eğer;

$P(c_i)P(n/c_i)> P(c_j)P(n/c_j), \sum\{1,\ldots..k\}; j\neq k$ (3.2.4.2)

olduğunda x'i c_i'ye atayarak bir sınıf seçer.

P(x/c_i)'nin P(c_i) ile çarpılmasının arkasındaki anlam, daha sık oluşan sınıflara daha yüksek bir ağrılık vererek daha az olası sınıflara daha düşük bir ağırlık vermektir. İyi bir sınıflandırma prosedürü, hatalı sınıflandırma maliyetlerini en aza indiren bir prosedürdür. Bayes sınıflandırıcı, risk hatalı sınıflandırma olasılığı olarak tanımlandığında kaybı en aza indirmektedir. Bayes sınıflandırıcı, Bayes hata oranı olarak bilinen mümkün olan en küçük sınıflandırma hatasını verir. Bayes kuralı, verilerin dağılımı biliniyorsa en uygun

sınıflandırma yöntemidir. Naive Bayes sınıflandırıcı, Bayes sınıflandırıcısının bir yaklaşımıdır (Zang ve ark., 2009).

3.2.5. Temel Yaklaşımlar

Naive Bayes sınıflandırıcısı (NBC) iki önemli varsayımdan faydalanır; birincisi, NBC sınıflandırıcısının gizlenmiş ya da latent özellikleri yoktur. Yani X'deki özelliklerin kümesi tamamlanmıştır. İkincisi ise, verilen sınıftaki tüm özellikleri birbirinden bağımsızdır.
Öyleki;

$$P(n_1, \dots, n_n/c_j) \sim \prod_{i=1}^{n} p(n_i/c_j) \quad (3.2.5.1)$$

olarak ifade edilir.

Bu varsayım tahmin edilecek parametrelerin sayısını azaltır. Karşılıklı olarak koşullu bağımsızlık varsayımının tahmin doğruluğunu önemli ölçüde etkilemeyeceği düşünülmektedir. Gerçekten de koşullu bağımsızlık varsayımı önemli ölçüde ihlal edildiğinde bile Naive Bayes sınıflandırıcısının optimal olduğunu Washington Üniversitesi'nden Profesör Pedro Domingos ve Michael Pozzini göstermiştir (Domingos ve Pazzani, 1997).

3.2.6. Naive Bayes Sınıflandırıcının Avantajı

NBC'yi anlamak ve oluşturmak nispeten basittir. Sayısal olarak, basit hesaplamalar ve sıradan bölmeyi içeren işlemler topluluğundan oluşur. Bu NBC'yi büyük örneklerde çalışırken bile eğitim için son derece hızlı yapar. Yeni gözlemleri sınıflandırmak için mükemmel derecede hızlıdır.

NBC'de değişkenler bağımsız olduğundan, bir çözüm üzerinde lojistik regresyon gibi ayırıcı modellerden çok daha hızlı bir şekilde yakınsama sağlayacaktır. Bu, son derece doğru bir tahmin modeli oluşturmak için daha az eğitim verisine ihtiyaç olduğu anlamına gelmektedir. NBC, olasılıksal tahminlerin gerekli olduğu yerlerde kullanılabilir ve ilgisiz değişkenlere karşı duyarlı değildir. Örneklem genişliği ne kadar büyük olursa, ilgisiz değişkenlerin de o kadar az etkisi olur. Gerçek ve kesikli verilerle baş edebilir. Ayrıca NBC, gerçek zamanlı çevrimiçi sistemlerde kolayca kullanılabilir.

NBC, birçok alanda son zamanlarda başarılı bir şekilde kullanıldı. Birçok pratik veri bilimi tabanlı projede üstün performans sunmaya devam ediyor. Üç önemli alanda özellikle kendini kanıtlamıştır. Bu alanlar; İnsan hareketi tanımı, tıbbi araştırma ve trafik sıkışıklığı alanlarıdır (Lewis, 2017).

3.3. Rasgele Ormanlar

Rasgele ormanlar, eğitim verilerinin yeniden örneklenmesiyle karar ağaçlarının bir "orman" oluşturduğu parametrik olmayan bir yöntemdir. Genellikle tek bir karar ağacı üzerinde önemli bir performans iyileştirmeleri sağlarlar (Lewis, 2017).

Topluluk modeli birçok farklı modelin sınıflandırma ve tahmin sonuçlarını birleştirir. Bireysel modeller "zayıf öğrenici" olarak bilinir, çünkü bireysel olarak zayıf tahmin performansına sahiptirler. Zayıf bir öğrenici, bir sınıflandırma probleminde sadece rasgele bir şansın üzerinde bir tahmin doğruluğuna sahiptir. Zayıf öğreniciler benzer bir model olabilirler ya da çok farklı olabilirler. Ancak bir araya geldiklerinde yüksek tahmin performansına sahip bir "güçlü öğrenici" oluşturabilirler. Başka bir deyişle, bir topluluk modelinin performansını genellikle bireysel modellerin performansından daha iyidir.

Rasgele Ormanlar, çok sayıda karar ağacını bir veri setine uyduran ve ardından tahminleri tüm ağaçlardan birleştiren bir topluluk sınıflandırıcısıdır. Rasgele Ormanlar, sınıflandırma ve regresyon için kullanılabilirler. Sınıflandırma için, karar ağaçları ormandaki tüm ağaçlar üzerinde ağaç başına bir oy ile çoğunluk oyu kullanılarak birleştirilir. Regresyon için, ormanlar, ağaçların üzerinden ortalamalar alınarak oluşturulur (Liaw ve Wiener, 2002).

3.3.1. Rasgele Ormanlar Algoritması

Algoritma örnek verilerinden yerine koyarak örnekleme yoluyla rasgele örneklerin seçilmesi ile başlar. Bu, "Bootstrapped" örnekleme olarak adlandırılır. Bootstraptted örnekleme ile elde edilen bu alt örneklemden bir karar ağacı oluşturulur. Tipik bir rasgele örneklemde, orijinal gözlemin yaklaşık %63'ü en az bir kez yer almaktadır. Seçilmeyen orijinal veri kümesindeki gözlemler, torba dışı gözlemler olarak adlandırılır. Bu torba dışı gözlemler hata oranını tahmin etmek için kullanılırlar. Bu hata oranı genellikle torba dışı hata oranı olarak adlandırılırlar ve özelliğin (değişkenin) önemini tahmin etmek için kullanılırlar. Örneklerin rasgele seçim süreci, tek bir karar ağacı üreten her bir alt örnekte birçok kez tekrarlanır. Tekrarlama işlemi sayesinde her ağaç farklıdır. Bunun nedeni, her bir bölünmenin rasgele seçilen özelliklerden tespit edilmesidir. Bu karar ağaçlarının bir ormanı ile sonuçlanır. Her ağaç budanmadan, mümkün olan en geniş ölçüde büyümüştür. Bireysel ağaçlar bir sınıflandırma veya tahmin kararı yaparlar. Bir gözlemin en son tahmin edilen sınıfı, sınıflandırma için çoğunluk oyu ile veya regresyon için ağırlıklı ortalamalar yoluyla yapılır (Lewis, 2017).

Bir ormandaki her bir ağacın, klasik bir karar ağacından daha az doğruluk oranına sahip olma eğiliminde olmalarına rağmen birden fazla tahmini tek bir toplam tahminle birleştirmek

genellikle fazla doğruluğa sahip bir tahmin elde etmeyi sağlar. Bunun sebeplerinden biri, tek bir karar ağacının tahmin edilmesinin, eğitim setindeki küçük değişikliklere karşı oldukça hassas olma eğiliminde olmasıdır. Rasgele ormanlar genellikle tek bir karar ağacı ile ilgili yanı artırmadan genel varyansı azaltır. Rasgele ormandaki her ağaç zayıf bir öğrenicidir. Rasgele ormanın bütünü ise güçlü bir öğrencidir.

3.3.2. Rasgele Ormanların Avantajları

Rasgele Ormanlar, karar ağaçlarının avantajlarının çoğunu muhafaza ederler. Rasgele Ormanları oluşturmak ve çalıştırmak oldukça basittir ve genellikle çok iyi sonuç verirler. Son dönemlerde yapılan çalışmalar veri yapıları ve problemlerin birçoğunda ortaya çıkan regresyon ve sınıflandırma hatasını rasgele ormanlar yönteminin kullanılmasıyla azaltılabileceğini göstermektedir. Rasgele Ormanlar, yapay sinir ağları ve destek vektör makineleri yöntemleri ile rekabet etmektedir. Ancak rasgele ormanlar daha az parametreye sahip oldukları için eğitilmeleri daha hızlı olmaktadır. Rasgele ormanlar, minimum veri hazırlığına ihtiyaç duyarlar. Binlerce özelliğe sahip büyük veri tabanlarında bile etkin şekilde verimli çalışabilmektedir. Ayrıca, veri kümesinde yer alabilecek aykırı gözlemlere karşı dayanıklıdırlar. Rasgele ormanlar, orman inşa edilirken eğitim örneğinden genelleştirilmiş hatanın yansız bir tahminini ürettikleri için aşırı uyum problemi ile daha az karşı karşıya kalırlar. Bu hata oranı ormanı oluşturmak için uygun sayıda ağaç kullandığından çok doğru olma eğilimindedir (Lewis, 2017).

3.4. K-En Yakın Komşu Algoritması (KNN)

En yakın komşuluk algoritması, anlaşılması kolay ve pratikte oldukça iyi çalışan makine öğrenme algoritmalarından biridir. Bu algoritma parametrik olmayan bir algoritmadır. Burada parametrik olmayan terimi; örnek verileri üreten olasılık dağılımı hakkında hiçbir varsayımın olmadığı anlamına gelir. Gerçek dünya verileri teorik varsayımları ihlal etme eğilimindedir ve bu nedenle KNN gibi parametrik olmayan algoritmalar kullanışlıdır.

3.4.1. Uzaklık Ölçüleriyle Yakınlığın Ölçülmesi

KNN sınıflandırılması gereken gözlemlerle onların komşuları arasındaki uzaklığı hesaplar. Kestirilen sınıf bir çoğunluk oyu ile belirlenir. Bir komşunun yakınlığı nasıl belirlenir sorusuna cevap ararsak; Birçok yolla i ve j arasındaki uzaklık ölçülebilir. Öklid uzaklığı genellikle sürekli özellikler için kullanılır ve $D(x_i, x_j) = \sqrt{\sum_{m=1}^{n}(x_{im} - x_{jm})^2}$ şeklinde

gösterilir. Burada n özelliklerin sayısıdır. Diğer bir popüler ölçü ise; $D(x_i, x_j) = \sum_{m=1}^{n} | x_{im} - x_{jm}|$ şeklinde hesaplanan Manhattan Uzaklığıdır.

Minkowski Uzaklığı ise; $D(x_i, x_j) = \sum_{m=1}^{n}(| x_{im} - x_{jm}|^q)^{1/q}$ eşitliği ile elde edilir. Canberra Uzaklığı olarak adlandırılan diğer bir uzaklık ise;

$D(x_i, x_j) = \frac{\sum_{m=1}^{n} |x_{im} - x_{jm}|}{\sum_{m=1}^{n} |x_{im} + x_{jm}|}$ ile bulunur.

Hem Öklid hem de Manhattan uzaklığı, Minkowski uzaklığının özel durumları olarak değerlendirilmektedir. Minkowski uzaklığı, q=2 seçildiğinde Öklid, q=1 seçildiğinde Manhattan uzaklığının ağırlıklı versiyonudur.

Eğer bir uzaklık fonksiyonunu değiştirirseniz veri noktalarının nasıl sınıflandırılacağını değiştirirsiniz (Zhang ve ark., 2018).

3.4.2. K- En Yakın Komşu Yönteminin Avantajları

KNN yöntemini anlamak çok basittir. Ayrıca uygulamakta çok kolaydır. Süreç şeffaf olduğu için, makine öğrenme ve matematik konularında uzman olmayan profesyonellere açıklamak kolaydır. KNN ayrıca örnek tabanlı bir öğrenme algoritmasıdır. Örnek tabanlı öğrenme algoritmaları örnek verileri ezberlerler. Net bir eğitim veya modele sahip değildirler (Lewis, 2017).

4. BULGULAR VE TARTIŞMA

Çalışmanın bu aşamasında farklı alanlardan gerçek ve yapay veri kümeleri üzerine önceki bölümlerde incelenen yöntemlerin uygulamaları verilmiştir. R programında ve KNIME programında uygulamaları verilmiştir.

4.1. Karar Ağaçları Uygulaması

Karar ağacı sınıflandırma yönteminin uygulaması için R programında *mclust* paketindeki *banknote* veri kümesi kullanılmıştır. Bu veri kümesi, orijinal ve sahte İsviçre 1000 Franklık banknotlar üzerinde yapılan ölçümleri içerir. Çizelge 4.1, sınıf değişkeni ve özellikleri detaylı olarak açıklamaktadır. Veri kümesi, 100 sahte ve 100 orijinal banknottan oluşmaktadır.

Çizelge 4.1. İsviçre 1000 Franklık banknotlardaki değişkenler ve tipleri

Değişken	**Tanımlama**	**Tip**
DURUM	Gerçek veya sahte	Sınıf
UZUNLUK	Paranın uzunluğu	Özellik
SOL	Sol kenar genişliği	Özellik
SAĞ	Sağ kenar genişliği	Özellik
ALT	Alt kenar genişliği	Özellik
ÜST	Üst kenar genişliği	Özellik
DİYAGONAL	Çapraz (Köşegen) uzunluğu	Özellik

Çizelge 4.2, veri kümesinde yer alan değişkenler arasındaki korelasyonu veren tabloyu sunmaktadır. Sol uzunluk ve sağ uzunluk yaklaşık olarak %74'lük bir korelasyona sahiptir. Köşegen uzunluğu ve paranın uzunluğu tüm özellikler ile negatif korelasyona sahiptir.

Çizelge 4.2. Korelasyon matrisi

	UZUNLUK	**SOL**	**SAĞ**	**ALT**	**ÜST**	**DİYAGONAL**
UZUNLUK	1,000	0,231	0,152	-0,189	-0,061	0,194
SOL	0,231	1,000	0,743	0,414	0,362	-0,503
SAĞ	0,152	0,743	1,000	0,487	0,401	-0,516
ALT	-0,190	0,414	0,487	1,000	0,142	-0,622
ÜST	-0,061	0,362	0,401	0,142	1,000	-0,594
DİYAGONAL	0,194	-0,503	-0,516	-0,623	-0,594	1,000

sample fonksiyonu yardımıyla, eğitim seti için yerine koymaksızın rasgele olarak 150 gözlemi seçilir ve geriye kalan gözlemlerin test seti olarak kullanılması planlanır. Bu amaç için kullanılacak kod satırı,

N=nrow(banknote)

egitim<- sample(1:N, 150, FALSE)

şeklindedir.

Literatürde karar ağacı oluşturmada kullanılan algoritmaların performansları üzerine yapılan çalışmaların sonuçları göz önüne alınarak bu çalışmada karar ağacı oluşturmak için C5.0 algoritmasının kullanılmasına karar verilmiştir. Bu doğrultuda kullanılacak kod satırı,

library(C5.0)

sonuc<-C5.0 (durum~., data=banknote[egitim,])

şeklindedir.

sonuc karar ağacını görselleştirmek için *plot(sonuc)* komutu kullanılır.

Yaprak düğümleri, gözlemlerin sayısının ve oranının doğru sınıflandırıldığını gösterir. Sınıflandırma kurallarını görmek için kullanılacak kod satırı,

kurallar<-C50(durum~.data=banknote[egitim,],rules=True)

şeklinde verilir.

Daha sonra,

summary(kurallar)

R kodu kullanılarak *kurallar* görülebilir.

Kural 1: (70, Alt>8,6

Diyagonal<= 140,6

Durum → Sahte

Kural 2: (44, Uzunluk<=214,8
Diagonal<= 140,6
Durum → Sahte

Kural 3: (75, Diagonal>140,6
Durum → Gerçek

Kural 4: (30, Length>214,8
Bottom<=8,6
Durum → Gerçek

İlk kural, eğer alt>8,6 ve diyagonal<=140,6 ise sınıflandırmanın sahte olarak yapılacağını ifade eder. Dördüncü kural eğer alt<=8,6 ve uzunluk>214,8 ise sınıflandırmanın gerçek olarak yapılacağını ifade eder.

Karar ağacının eğitim örneğini ne kadar iyi sınıflandırdığını bulmak için, *predict* ve *table* fonksiyonlarının bir kombinasyonunu kullanılır.

Bunun için kullanılacak R fonksiyonu;

kestirim_egitim<-predict(sonuc,newdata=banknote[egitim,],type="sınıf")

ile hesaplanır. Elde edilen kestirim sonuçları *kestirim_egitim* de saklanır.

Karar ağacının, eğitim örneğini ne kadar iyi sınıflandırdığı *table* fonksiyonu kullanılarak aşağıda verilen R kodu yardımıyla bulunabilir:

table(banknote$durum[egitim],kestirim_egitim, dnn=c("gözlenen sınıf","gerçek sınıf")

Çizelge 4.3. İsviçre 1000 Franklık banknotlar için eğitim verisi hata tablosu

				GERÇEK		
	TAHMİN			**Gerçek Para**	**Sahte Para**	**TOPLAM**
			Gerçek Para	77	0	77
			Sahte Para	0	73	73
			TOPLAM	77	73	150

Çizelge 4.3 incelendiğinde, karar ağacının tüm örnekleri doğru sınıflandırdığı görülmektedir. Bu sonuç mükemmel bir sonuçtur. Fakat şuan eğitim setinde çalışıldığının unutulmaması gerekmektedir. Modeller genellikle eğitim kümesi verileri için çok iyi çıkabilir. Bu nedenle *test* kümesi için durum incelenmeden yorum yapılması yanılgıya düşürebilir.

Bu nedenle, sonraki adım olarak, modelin test örneği için ne kadar iyi performans gösterdiği araştırılır. Bu bağlamda, daha önce ana hatları verilen adımlar test kümesi için tekrarlanır. Bunun için kullanılacak kod satırı,

kestirim<-predict(sonuc,newdata=banknote[-egitim,], type="sınıf")

şeklinde verilmektedir. Bu R kodunda yer alan *–egitim* test örneğini ifade etmektedir.

Karar ağacının test örneğini ne kadar iyi sınıflandırdığını bulmak için, *predict* ve *table* fonksiyonlarının bir kombinasyonunu kullanılır.

table(banknote$durum[-egitim],kestirim, dnn=c("gözlenen sınıf","gerçek sınıf"))

Çizelge 4.4. İsviçre 1000 Franklık banknotlar için test verisi hata tablosu

			GERÇEK		
			Gerçek Para	**Sahte Para**	**TOPLAM**
TAHMİN		**Gerçek Para**	23	0	23
		Sahte Para	3	24	27
		TOPLAM	26	24	50

Çizelge 4.4 incelendiğinde, karar ağacının 50 gözlem içeren test örneğinden 3 tanesi hariç, geriye kalanların tümünü doğru olarak sınıflandırdığı görülmektedir. Sınıflandırma doğruluk oranı %94 olup, bu oran söz konusu veri kümesinin sınıflandırılması için oldukça iyi bir oran olarak yorumlanabilir.

4.2. Naive Bayes Uygulaması

Naive Bayes uygulaması için simülasyon kullanılarak elde edilen veri kümesi kullanılmıştır. Simülasyon verileri, Normal dağılımdan iki değişken (özellik), X_1 ve X_2 kullanılarak 100 gözlem için oluşturulur. Bu işlem için kullanılan R komutları:

degsay<-2

N<-100

*x<-matrix(rnorm(n*degsay), ncol=degsay)*

colnames(x)<-c("X_1","X_2")

y<-as.numeric(x[,1]^2+x[,2]^2)>2.3)

şeklinde verilir.

Bu adımlarda; 1. Satırda, değişkenlerin sayısı belirtilir. 2. Satırda ise örnek büyüklüğü 100 olarak ayarlanır. Değişkenleri isimlendirmek için *colnames* fonskiyonu kullanılır. X matrisi normal değerleri içerir ve bu matris *rnorm* fonksiyonu kullanılarak üretilir.

R nesnesi y, 0 veya 1 biçiminde sınıf etiketleri içerir. X nesnesi ise değişkenleri gösterir.

Sınıf nesnesi nümeriktir. Bu yüzden onu bir faktör içine

y<-as.factor(y)

komutu ile dönüştürürüz.

Daha sonra, değişkenleri ve sınıf etiketlerini veri nesnesi içinde birleştiririz. *as.data.frame* fonksiyonu bir matrisden bir veri çerçevesine dönüştürmek için kullanılır. Bu dönüşüm Naive Bayes fonskiyon kullanımını biraz daha kolaylaştırır. Bunun için gerekli R komutları:

data<-cbind(y,x)

data<-as.data.frame(data)

olarak verilir.

Sonraki adımda, *sample* fonksiyonu kullanarak yerine koymaksızın rasgele bir şekilde eğitim örneği için 70 gözlem seçilir. Bu işlem için gerekli R komut satırı:

egitim<-sample(1:N,70,FALSE)

şeklinde olmalıdır.

Naive Bayes modeli, *e1071* paketi kullanılarak veri üzerinden eğitilebilir. Bu ilk yüklenecek pakettir. Daha sonra *naiveBayes* fonksiyonu çağırılır. Bunun için aşağıdaki R komutu satırları yazılır.

library(e1071)

sonuc<-naiveBayes(x[egitim,],y[egitim])

Bu komutlar ile uydurulan model *sonuc* isimli R nesnesine yerleştirilmiş olur.

predict fonksiyonu sınıf tahminleri yapmak için kullanılır. Uydurulan modelin eğitim kümesindeki olasılıklarını görmek için *type="raw"* değeri aşağıdaki gibi eklenir.

kestirim_olasılıklar<-predict(sonuc,data[egitim,-1], type="raw")

Burada ilk birkaç sınıf olasılıkları, head fonksiyonu kullanılarak:

head (kestirim_olasılıklar)

komutu ile görülebilir.

	0	1
[1,]	0,586	0,413
[2,]	0,023	0,977
[3,]	0,845	0,155
[4,]	0,106	0,894
[5,]	0,023	0,977
[6,]	0,491	0,509

Burada, ilk gözlem sınıf 0 için 0,58 ve sınıf 1 için 0,41'lik bir sonsal olasılığa sahiptir. Kestirilen sınıf etiketi 0'dır. Altıncı gözlem, sınıf 0 için 0,49 ve sınıf 1 için 0,51'lik bir

olasılığa sahiptir. Burada sınıf 1, kestirilen sınıf etiketidir. Tahmin edilen sınıf etiketlerini görmek için:

kestirim<-predict(sonuc,data [egitim,-1], type=”sınıf”)

komut satırı kullanılır.

Eğitim örneğini ne kadar iyi sınıflandırdığı *table* fonksiyonu kullanılarak aşağıda verilen R kodu yardımıyla bulunabilir:

y_egitim<-y[egitim]

table(y_tegitim,kestirim)

Buna göre; 0 etiketleri yüksek doğruluk derecesi ile tahmin ediliyor. Sınıf 1 için eğitim kümesi daha az hassastır.

Çizelge 4.5. Simülasyon verisi eğitim hata tablosu

			GERÇEK		
TAHMİN			**0**	**1**	**TOPLAM**
		0	38	1	39
		1	2	29	31
		TOPLAM	40	30	70

Çizelge 4.5 incelendiğinde, bu modelde Sınıf 0 için 38 gözlem doğru sınıflandırılmıştır. Sınıf 1’de ise 30 gözlemin 29’u doğru sınıflandırılmıştır.

Eğitim kümesi için, genel doğruluk oranı %96 olarak bulunmuştur. Sonraki adımda *test* kümesinin performansının değerlendirilmesi kullanılacak komut satırı:

kestirim_test<-predict(sonuc, data [-egitim,-1], type=”sınıf”)

y_test<-y[-egitim]

table(y_test, kestirim_test)

olarak verilir.

Çizelge 4.6. Simülasyon verisi test hata tablosu

			GERÇEK		
TAHMİN			**0**	**1**	**TOPLAM**
		0	17	5	22
		1	0	8	8
		TOPLAM	17	13	30

Çizelge 4.6 incelendiğinde model, *sınıf 0* için tüm gözlemleri doğru sınıflandırmıştır. Fakat *sınıf 1* için 13 gözlemden 8 tanesini doğru sınıflandırmıştır. Test kümesi üzerinden modelin genel doğruluğu %83'tür. Eğitim örneği ile verilen performanstan biraz düşük bir oran elde edilmiştir. NBC, şartlı bağımsızlığı varsaydığı için sıklıkla eleştirilmektedir. Bu varsayım gerçek dünya verileri tarafından genellikle ihlal edilmektedir. Uygulamada, bu varsayım için açık ihlal durumlarında bile sonuç iyi olabilir. NBC, ampirik verilerden elde edilen olasılıkları tahmin ettiğinden, olasılıkları doğru olarak tahmin etmek için yeterli gözlem olmadığı durumlarda bazen zorluklarla karşılaşabilir. Bu zorluklar daha fazla gözlem toplayarak veya ampirik tahminlerin yerine ancak önceki bilgiler kullanılarak giderilebilir.

4.3. Rasgele Ormanlar Uygulaması

Rasgele Ormanlar sınıflandırma yönteminin uygulaması için R programında *mclust* paketindeki *thyroid* veri kümesi kullanılmıştır. Bu veri kümesi, 215 hastaya verilen 5 laboratuvar testinden oluşmaktadır. Testler bir hastanın tiroid fonksiyonunun doğru bir şekilde sınıflandırılıp sınıflandırılmayacağını tahmin etmek için kullanıldı. Çizelge 4.7 sınıf değişkeni ve özellikleri detaylı olarak açıklamaktadır.

Çizelge 4.7. Tiroid veri kümesi değişkenleri ve tipleri

Değişkenler	**Açıklama**	**Tip**
Teşhis	Hypo, Normal ve Hyper	Sınıf
TR3U	T3 Reçine alım testi	Özellik
T4	Toplam Serum Tiroksin	Özellik
T3	Toplam Serum Triiodothyronine	Özellik
TSH	Bazal Tiroid Uyarıcı Hormon	Özellik
DTSH	Tiroid Özelliğinin Maksimum Mutlak Farkı	Özellik

sample fonksiyonu yardımıyla, eğitim seti için yerine koymaksızın rasgele olarak 150 gözlemi seçilir ve geriye kalan gözlemlerin test seti olarak kullanılması planlanır. Bu amaç için kullanılacak kod satırı,

```
N=nrow(thyroid)
egitim<-sample(1:N, 150, FALSE)
```

şeklinde verilmiştir.

Model oluşturmak için *randomforest* paketi kullanılır. Rasgele Orman modelini oluşturmada iki temel seçenek vardır. Bunlardan birincisi ağaçların sayısı olup, ikincisi ise rasgele bir şekilde seçilecek özelliklerin sayısıdır. Bu bağlamda, 800 ağaca sahip bir orman oluşturulsun. Bu işlem için kullanılacak R komut satırları aşağıda verilmiştir.

library(randomforest)

num.trees=800

sonuc<-randomforest(teshis~.,data=thyroid[egitim,], ntree=num.trees, mytry=4)

Ormandaki her bir ağaç için, özelliklerin rasgele seçilen bir alt kümesi, her bir düğümü ayırmak için kullanılır. Bu *mytry* parametresi tarafından kontrol edilir. Yukarıdaki kodda beş özelliğin (değişken) dördü *mytry=4* şeklinde rasgele seçilmiştir.

Çizelge 4.8. Tiroid veri kümesi eğitim hata tablosu

			GERÇEK			
			Hypo	**Normal**	**Hyper**	**TOPLAM**
TAHMİN		**Hypo**	20	1	0	21
		Normal	1	109	0	110
		Hyper	0	1	18	19
		TOPLAM	21	111	18	150

Model ile ilgili ayrıntılar, *print* komutu kullanılarak görüntülenebilir. Çıktının ilk kısmı ağaç sayısını (800) ve her ayırmada kullanılan rasgele bir şekilde seçilen özelliklerin sayısını gösterir. Çizelge 4.8 incelendiğinde bu model % 2 civarında bir genel hataya sahiptir. Sınıflara göre doğruluk oranlarında önemli farklılıklar vardır. Hypo için hata oranı %1.3, normal için hata oranı %1.8 ve Hyper için hata oranı %21 bulunmuştur. Eğitim kümesi için, genel doğruluk oranı %98 olarak bulunmuştur.

Modelin test veri kümesinde nasıl çalıştığını değerlendirmek için *test* veri kümesi kullanılarak önceki adımlar tekrarlanır. Bunun için aşağıdaki R komutları kullanılır:

sonuc_test<-randomforest(teshis~.,data=thyroid[-egitim,],ntree=num.trees, mytry=4)

print(sonuc_test)

Çizelge 4.9. Tiroid veri kümesi için test hata tablosu

			GERÇEK			
			Hypo	**Normal**	**Hyper**	**TOPLAM**
TAHMİN		**Hypo**	7	2	0	9
		Normal	0	40	0	40
		Hyper	0	2	14	16
		TOPLAM	7	44	14	65

Çizelge 4.9 incelendiğinde, rasgele ormanlar yönteminin 65 gözlem içeren test örneğinden 4 tanesi hariç, geriye kalanların tümünü doğru olarak sınıflandırdığı görülmektedir. Sınıflandırma doğruluk oranı %93 olup, bu oran söz konusu veri kümesinin sınıflandırılması için oldukça iyi bir oran olarak yorumlanabilir.

4.4. K-En Yakın Komşu (KNN) Yöntemi Uygulaması

K-en yakın komşu sınıflandırma yönteminin uygulaması için R programında *rebmix* paketindeki *wine* veri kümesi kullanılmıştır. Bu veri kümesi İtalya'da aynı bölgede bulunan üç farklı tip üzümün kimyasal analizlerinden türetilen 13 özellik içermektedir. Çizelge 4.10 sınıf değişkeni ve özellikleri detaylı olarak açıklamaktadır.

Çizelge 4.10. Wine veri kümesi değişkenleri ve tipleri

Değişkenler	**Açıklama**	**Tip**
Alcohol	Sürekli	Özellik
Malic.Acid	Sürekli	Özellik
Ash	Sürekli	Özellik
Alcalinity.of.Ash	Sürekli	Özellik
Magnesium	Sürekli	Özellik
Total.Phenols.	Sürekli	Özellik
Flavanoids	Sürekli	Özellik
NonFlavanoid.Phenols	Sürekli	Özellik
Proanthocyanins	Sürekli	Özellik
Color.Intensity	Sürekli	Özellik
Hue	Sürekli	Özellik
OD280.OD315.of.Diluted.Wines	Sürekli	Özellik
Proline	Tam Sayı	Özellik
Cultivar	1, 2 veya 3	Sınıf

KNN'nin performansı uzaklık ölçüsüne ciddi bir şekilde bağlıdır. Bu nedenle genel ve yaklaşık olarak aynı aralıkta olsunlar diye özellikler ölçeklendirilir ya da normalleştirilir. Eğer bu yapılmazsa daha büyük aralıklara sahip özellikler daha önemli olarak değerlendirilecektir.

Normalleştirilmiş veri kümesi R'da;

veri <-wine[,1:13]

veri <-scale(veri)

komutları ile elde edilir.

Yerine koyma olmaksızın rasgele bir örnek, *sample* fonksiyonu ile belirlenir. Bu nedenle *replace* özelliği *FALSE* olarak ayarlanır. Eğitim örneği için yerine koymaksızın 89 gözlem aşağıdaki verilen R komut satırları kullanılarak seçilir

N=nrow(veri)

egitim<-sample(1:N, 89, replace=FALSE)

Burada veri kümesi, örnekteki toplam gözlem sayısı olan 178 veriden eğitim için rasgele seçilen 89 gözlemi içerir.

R'da bir KNN modeli tahmin etmek için çok sayıda seçenek vardır. *knnGarden* paketinden *knnVCN* fonksiyonu uzaklık ölçütlerini kullanmaya olanak sağlar. Burada çözülmesi gereken en yakın iki komşuyu kullanacak bir KNN modelinin nasıl bulunabileceğidir. Bunun için aşağıda verilen R komut satırları kullanır.

require (knnGarden)

sonuc <- knnVCN (veri [train ,],wine$Cultivar [egirim],veri [-egitim ,],K = 2,

method = " canberra ")

knnVCN'nin ilk argümanı eğitim kümesi özelliklerini alır. Bunu sınıf değişkeni takip eder. K parametresi analizde kullanılan komşuların sayısını ifade eder. Burada K=2 olarak ayarlanmıştır. Uzaklık ölçüsü için ise burada c*anberra* uzaklığı kullanılmıştır. Test veri kümesi için tahmin edilen değerler *sonuc$TstxIBelong* da saklanır.

Performansı değerlendirmek için bir hata matrisi,

table(sonuc$TstxIBelong,wine$cultivar[-egitim])

R komutu ile elde edilir.

Çizelge 4.11. Wine veri kümesi için test hata tablosu

			GERÇEK			
			1	**2**	**3**	**TOPLAM**
TAHMİN		**1**	30	2	0	32
		2	1	30	0	31
		3	0	1	25	26
		TOPLAM	31	33	25	89

Çizelge 4.11 incelendiğinde, rasgele ormanlar yönteminin 89 gözlem içeren test örneğinden 4 tanesi hariç, geriye kalanların tümünü doğru olarak sınıflandırdığı görülmektedir. Sınıflandırma doğruluk oranı %95,5 olup, bu oran söz konusu veri kümesinin

sınıflandırılması için oldukça iyi bir oran olarak yorumlanabilir. Bu hata matrisi bize KNN algoritmasının gözlemlerin çoğunu doğru bir şekilde sınıflandırdığını göstermektedir.

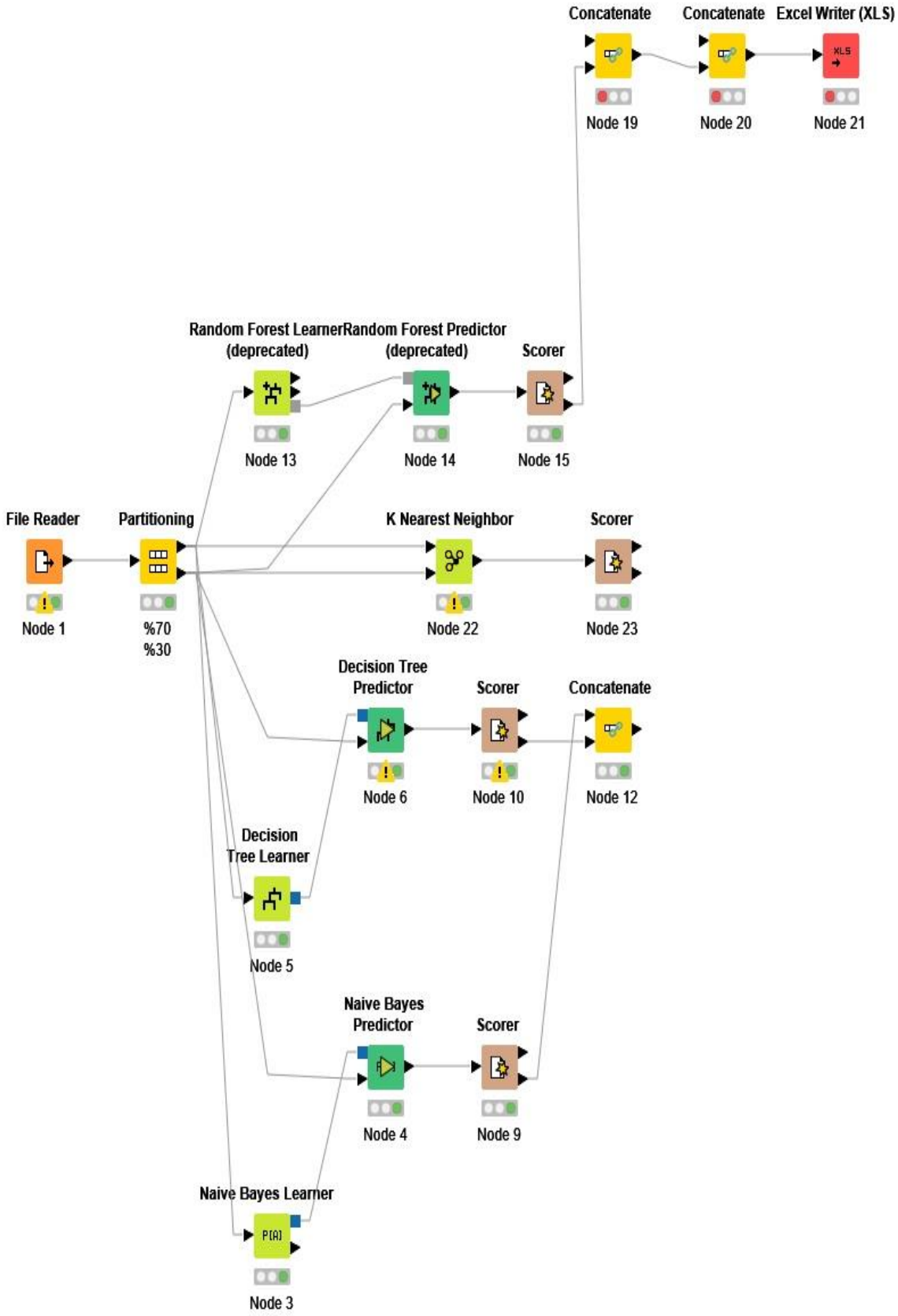
4.5. Makine Öğrenmesi Sınıflandırma Algoritmalarının KNIME Programı ile Performans Karşılaştırması

Bu bölümde kullanılacak veri kümesi UCI makine öğrenme havuzundan elde edilen Bankacılık Pazarlama verisidir. Veriler, bir Portekiz bankacılık kurumunun doğrudan pazarlama kampanyaları ile ilgilidir. Bu veri kümesi üzerinden son zamanlarda yıldızı parlayan KNIME programı yardımıyla önceki bölümlerde ayrıntılı olarak incelenen makine öğrenmesi sınıflandırma algoritmalarından Karar Ağacı, Naive Bayes, Rastgele Orman ve K-En yakın komşu yöntemlerinin performansları karşılaştırılmıştır. Sınıflandırma hedefi ise, müşterinin bir vadeli mevduat (Evet / Hayır) kabul edip etmeyeceğini tahmin etmektir.
Bankacılık veri kümesi KNIME programı kullanılarak analiz edilmiş dört makine öğrenmesi sınıflandırma algoritmasının performans ölçütleri Çizelge 4.12 de sunulmuştur. KNIME iş akış şeması Şekil 4.1 de verilmiştir.

Çizelge 4.12. Bankacılık Pazarlama Verisi için Performans Ölçütleri

	KARAR AĞACI	NAİVE BAYES	RASGELE ORMANLAR	K-EN YAKIN KOMŞULUK
DOĞRULUK ORANI	0,885	0,846	0,906	0,883
DUYARLILIK ORANI	0,501	0,403	0,681	0,319
BELİRLEYİCİLİK ORANI	0,930	0,946	0,922	0,923

Çizelge 4.12 incelendiğinde, Doğruluk oranına göre en iyi sınıflandırmayı Rastgele Ormanlar algoritması yapmış olup, diğer üç sınıflandırma algoritmasının performansları birbirlerine yakın olarak bulunmuştur. Duyarlılık oranına göre ise, en iyi sınıflandırmayı yine Rastgele Ormanlar algoritması yapmış olup, Karar Ağacı sonra gelen sınıflandırma algoritması olmuştur. Belirleyicilik oranı açısından Naive Bayes sınıflandırma algoritması en iyi performansı gösteren sınıflandırma algoritması olmaktadır. Tüm performans ölçütleri açısından incelendiğinde, en iyi sınıflandırma başarısını Rastgele Ormanlar algoritmasının gösterdiği tespit edilmiştir.



Şekil 4.1. KNIME iş akış şeması

5. SONUÇ

Teknolojik gelişmeler veri boyutunun çok daha fazla artmasına sebep olmuştur. Lohr (2012)'a göre veri büyümesinin her yıl bir önceki yılın %50 si kadar arttığı tahmin edilmektedir. Artan teknolojik gelişmelerden dolayı çok büyük miktarda veri elde edilmesine karşın böyle devasa çoklukta verinin işlenmesi, saklanması ve hızlı bir şekilde kaydedilmesi süreci oldukça zahmetlidir. Boyutsallıktaki büyük artıştan dolayı, verideki karmaşıklık ve heterojenlik araştırmacıların, verinin ölçülebilirliği, kalitesi ve görselleştirilebilmesi gibi yeni zorluklarla yüzyüze gelmesine neden olmaktadır. Bu nedenle, klasik istatistiksel yöntemlerin yetersiz kaldığı ve varsayımlarının sağlanamadığı durumlarda alternatif yaklaşımlara ihtiyaç duyulmaktadır. Bu bağlamda, istatistiğin, bilgisayar bilimi ve matematiğin bütünleşik bir şekilde süreçte yer alması gerekmektedir. Kategorik olarak düşündüğümüzde bu süreç makina öğrenme, istatistiksel çıkarım, optimizasyon, ağ analizi ve görselleştirme konuları üzerine yoğunlaşır.

Bu tez çalışmasında, makine öğrenmesi sınıflandırma algoritmalarından Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu yöntemleri R programında yazılan kodlar yardımıyla farklı veri tiplerine uygulanması ile daha anlaşılır ve açık bir şekilde sunulması amaçlanmıştır. Ayrıca gerçek bir veri kümesi üzerinden, son dönemlerde makine öğrenmesi yöntemleri için son kullanıcı dostu yeniliklerle ön plana çıkan KNIME programı yardımıyla yukarıda bahsedilen dört makine öğrenme sınıflandırma algoritmasının bir performans karşılaştırması yapılmıştır. Çalışmanın uygulama bölümünde, ilk dört uygulamada, farklı veri kümeleri için Karar Ağacı, Naive Bayes, Rastgele Orman ve K-en yakın komşu yöntemlerinin her birini R programının uygulama analiz adımları ayrıntılı bir şekilde ele alınmıştır. Beşinci uygulamada ise teorik alt yapısı önceki bölümlerde verilen sınıflandırma algoritmalarının gerçek bir veri kümesi üzerinden performans karşılaştırmaları KNIME programı kullanılarak yapılmıştır. Tüm performans ölçütleri açısından incelendiğinde, en iyi sınıflandırma başarısını Rastgele Ormanlar algoritmasının gösterdiği tespit edilmiştir.

Sonraki çalışmalarda, veri kümesinde aykırı gözlemlerin olması durumunda sınıflandırma yöntemlerinin performanslarının karşılaştırılması ve dayanıklı istatistik ölçüleri kullanılarak yöntemlerde yapılacak bir takım modifikasyonların genel başarı üzerindeki etkileri incelenebilir.

6. KAYNAKLAR

Alpaydin, E. (2016). Machine learning: the new AI. MIT Press.

Balaban, M.E., Kartal, E. (2015). Veri madenciliği ve Makine Öğrenmesi Temel Algoritmaları ve R Dili Uygulamaları, Çağlayan Kitabevi, İstanbul.

Berglund, E, Lytsy, P, Westerling, R. (2013). Adherence to and beliefs in lipid-lowering medical treatments: Astructural equation modeling approach including the necessity-concern framework. Patient Education and Counseling; 91(1): 105–12.

Bishop, C. M. (2007). Pattern Recognition and Machine Learning, Springer, ISBN: 0-387-31073-8.

Bohanec, M., Bratko, I. (1994). Trading accuracy for simplicity in decision trees, Machine Learning 15: 223–250.

Camastra, F. ve Vinciarelli, A. (2008). Machine Learning for Audio, Image and Video Analysis: Theory and Applications, Springer, ISBN: 1-84800-007-3.

De M´antaras, R. L. (1991). Distance Based Measures:, A distance-based attribute selection measure for decision tree induction. Mach. Learn., vol. 6, no. 1, pp. 81–92,.

Domingos, Pedro, and Michael Pazzani. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning 29.2-3, 103-130.

Dreiseitl, S., Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. Journal of biomedical informatics, 35(5-6), 352-359.

Flach, P. (2004). Tutorial on "The many faces of ROC analysis in machine learning." Tutorial Notes, 2004 Banff, Alberta, Canada, ICML 2004, 1-49.

Hashem Ibrahim Abaker, ve ark. (2015). The rise of “big data” on cloud computing: Review and open research issues. Information Systems 47 : 98-115.

Hilbert M., and Priscila L. (2011). The world's technological capacity to store, communicate, and compute information. Science,1200970.

Jacobs A. (2009). The pathologies of big data. Queue 7.6 (2009): 10.

Lewis, D. N. (2017). Machine Learning Made Easy with R: An Intuitive Step by Step Blueprint for Beginners, CreateSpace Independent Publishing Platform.

Liaw A., Matthew W. (2002). Classification and regression by randomForest. *R* news 2.3,18-22.

Lohr, S. (2012). The age of Big Data. N.Y. Times. February 11, http://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the world.html?_r=0j. Accessed on May 25, 2016.

Narayanan V., Ishan A., Arjun B. (2013). Fast and accurate sentiment classification using an enhanced Naive Bayes model. *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, Berlin, Heidelberg.

Obermeyer Z., Emanuel, EJ. (2016). Predicting the Future—Big Data, Machine Learning, and Clinical Medicine. The New England journal of medicine, 375(13): 1216–9

Quinlan, K., Kohavi, R. (1999) Decision Tree Discovery Data Mining,.

Refaeilzadeh, P., Tang, L. Liu, H. (2009). Cross-validation, Encyclopedia of database systems, Springer, 532–538.

Snijders C., Uwe M., Ulf-Dietrich R. (2012)."Big Data": big gaps of knowledge in the field of internet science. International Journal of Internet Science 7.1,1-5.

Torgo, L. (2016). Data mining with R: learning with case studies. Chapman and Hall/CRC.

Weng, S. F., Reps, J., Kai, J., Garibaldi, J. M., Qureshi, N. (2017). Can machine-learning improve cardiovascular risk prediction using routine clinical data?. PloS one, 12(4), e0174944.

Zhang, Min-Ling, José M. Peña, and Victor Robles. "Feature selection for multi-label naive Bayes classification." *Information Sciences* 179.19 (2009): 3218-3229.

Zhang, T., Fulk, G. D., Tang, W., Sazonov, E. S. (2013). Using decision trees to measure activities in people with stroke. In Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE (pp. 6337-6340). IEEE.

Zhang, S., Li, X., Zong, M., Zhu, X., & Wang, R. (2018). Efficient knn classification with different numbers of nearest neighbors. *IEEE transactions on neural networks and learning systems*, *29*(5), 1774-1785.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	: Adem Kürşat KESKİN
Doğum Tarihi ve Yeri	: 26.01.1986 / İSTANBUL
Yabancı Dili	: İngilizce
E-posta	: ademkursatkeskin@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Lise	Sayısal	Kayseri Hisarcıklıoğlu Fen Lisesi	2004
Lisans	İşletme	Sakarya Üniversitesi	2011

İŞ TECRÜBESİ

Yıl	Firma/Kurum	Görevi
2008-2009	Garanti Bankası Bodrum Şubesi	Gişe Memuru
2009-2011	Garanti Bankası Genel Müdürlük	Operasyon Memuru
2012-	Sinop Üniversitesi	Bilgisayar İşletmeni