T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PERAKENDE SEKTÖRÜNDE MAKİNE ÖĞRENMESİNE DAYALI YAKLAŞIMLAR

TUĞÇE CEYLAN

YÜKSEK LİSANS TEZİ
İSTATİSTİK ANABİLİM DALI
İSTATİSTİK PROGRAMI

DANIŞMAN
PROF. DR. ALİ HAKAN BÜYÜKLÜ

İSTANBUL, 2018

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

PERAKENDE SEKTÖRÜNDE MAKİNE ÖĞRENMESİNE DAYALI YAKLAŞIMLAR

Tuğçe CEYLAN tarafından hazırlanan tez çalışması 06.12.2018 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Ali Hakan BÜYÜKLÜ

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Prof. Dr. Ali Hakan BÜYÜKLÜ

Yıldız Teknik Üniversitesi _____

Dr. Öğr. Üyesi Doğan YILDIZ

Yıldız Teknik Üniversitesi _____

Prof. Dr. Şahamet BÜLBÜL

Yıldız Teknik Üniversitesi _____

ÖNSÖZ

Çalışmam boyunca bana her konuda destek olan danışmanım ve aynı zamanda bölüm başkanım Prof. Dr. Ali Hakan Büyüklü'ye desteği ve yardımları için teşekkür ederim.

Tezin uygulama bölümünde kullandığım IPSOS Tüketici Hane Paneli verisine erişimim için bana yetki veren yöneticim Utku Doğruak'a ve direktörüm Ahu Şendilmen'e değerli katkıları için teşekkürlerimi sunarım.

Son olarak tüm eğitim hayatım boyunca bana katkı sağlayan değerli hocalarıma, maddi ve manevi desteğini her zaman hissettiğim aileme ve tez yazım sürecinde manevi olarak bana destek olan arkadaşlarıma sonsuz teşekkür ederim.

Aralık, 2018

Tuğçe CEYLAN

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ vi

KISALTMA LİSTESİ vii

ŞEKİL LİSTESİ viii

ÇİZELGE LİSTESİ x

ÖZET xi

ABSTRACT xii

BÖLÜM 1

GİRİŞ 1

1.1 Literatür Özeti 1
1.2 Tezin Amacı 2
1.3 Hipotez 2

BÖLÜM 2

MAKİNE ÖĞRENMESİ 3

2.1 Makine Öğrenmesi Nedir 3
2.2 Algoritma Öğrenme Yöntemleri 3
2.2.1 Denetimli Öğrenme 4
2.2.2 Denetimsiz Öğrenme 4
2.2.3 Yarı - Denetimli Öğrenme 4
2.3 Denetimli Öğrenme Yöntemleri 5
2.3.1 k - En Yakın Komşular 5
2.3.2 Naive Bayes Sınıflandırıcısı 6
2.3.3 Karar Ağaçları 7
2.3.3.1 ID3 8
2.3.3.2 C4.5 8
2.3.3.3 CART 8
2.3.4 Topluluk Öğrenme Yöntemleri (Ensemble Learning) 8

2.3.4.1 Torbalama Algoritması (Bagging) 9
2.3.4.2 Rasgele Ormanlar (Random Forests) 9
2.3.4.3 Güçlendirme Algoritması (Boosting) 10
2.3.5 Çoklu Doğrusal Regresyon 12
2.3.6 Lojistik Regresyon 13
2.3.7 Destek Vektör Makineleri (Support Vector Machines) 14
2.3.8 Yapay Sinir Ağları (Artificial Neural Networks) 15
2.3.8.1 Biyolojik ve Yapay Nöronlar 16
2.4 Denetimsiz Öğrenme Yöntemleri 18
2.4.1 Kümeleme 18
2.4.1.1 k - Means Kümeleme 18
2.4.1.2 Hiyerarşik Kümeleme 20
2.4.2 Boyut İndirgeme 20
2.5 Modelleri Test Etme ve Doğrulama 20
2.6 Modelleri Değerlendirme 21
2.6.1 Karışıklık Matrisi 21
2.6.2 ROC Eğrisi 23

BÖLÜM 3

UYGULAMA 25

3.1 SQL Database Server ile RStudio Bağlantısı 25
3.2 Veri Düzenleme ve Değişken Açıklama 26
3.3 Verilerin Görselleştirilmesi 29
3.4 XGBoost ile Sınıflandırma 32
3.4.1 XGBoost Modelinin Eğitilmesi 33
3.4.2 XGBoost Modelindeki Değişkenlerin Önem Matrisi 36
3.5 Naive Bayes ile Sınıflandırma 42
3.5.1 Naive Bayes Modelinin Eğitilmesi 43
3.6 Modellerin Tahmin Performanslarının Karşılaştırması 45

BÖLÜM 4

SONUÇ VE ÖNERİLER 46

KAYNAKLAR 48

ÖZGEÇMİŞ 50

SİMGE LİSTESİ

P	Olasılık
w	Yapay sinir ağı modeli için ağırlıklar
β	Regresyon katsayısı
ε	Hata terimi

KISALTMA LİSTESİ

AdaBoost	Adaptive Boosting
ANN	Artificial Neural Networks
AUC	Area Under Curve
CART	Classification and Regression Trees
CHAID	Chi-Squared Automatic Interaction Detector
FMCG	Fast Moving Consumer Goods
FN	False Negative
FP	False Positive
GBM	Gradient Boosting Machine Goods
ID3	Iterative Dichotomiser 3
kNN	k-Nearest Neighbours
ML	Machine Learning
MSE	Mean Square Error
O.O.B	Out Of Bag
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate
XGBoost	Extreme Gradient Boosting

ŞEKİL LİSTESİ

Sayfa

Şekil 2. 1 Karar ağacı 7
Şekil 2. 2 Ağırlık güncellemeleri ile AdaBoost sıralı eğitim 11
Şekil 2. 3 Regresyon tahmin yapısı 12
Şekil 2. 4 Doğrusal bir sınıflandırıcı örneği 14
Şekil 2. 5 Eğrisel bir sınıflandırıcı örneği 14
Şekil 2. 6 SVMs görsel örneği 15
Şekil 2. 7 Biyolojik sinir hücresi 16
Şekil 2. 8 Yönlendirilmiş ağ diyagramı 16
Şekil 2. 9 Gizli katmanlı tekrarlı ağ örneği 17
Şekil 2. 10 Üç kümeli bir kümeleme örneği 18
Şekil 2. 11 Dirsek metodu 19
Şekil 2. 12 Protein tüketimi bakımından kümelenen ülkelerin dendogram örneği 20
Şekil 2. 13 Karışıklık matrisi 21
Şekil 2. 14 Kesinlik versus anma örnek grafik 23
Şekil 2. 15 ROC eğrisi 24
Şekil 3. 1 SQL Database bağlantısının yapılması ve verinin aktarılması 26
Şekil 3. 2 Gerekli kütüphaneler, eksik verilerin tespiti ve veriden çıkarılması 27
Şekil 3. 3 Değişkenlerin yapısı 27
Şekil 3. 4 Markaların tarih detayında alımlarının grafiği 29
Şekil 3. 5 Markaların alışveriş kanalı detayında alımlarının grafiği 30
Şekil 3. 6 Markaların fiyat kategorileri 30
Şekil 3. 7 Sosya ekonomik statüye göre marka alımları grafiği 31
Şekil 3. 8 Hane kişi sayısına göre marka tercihlerinin grafiği 31
Şekil 3. 9 Marka tercihlerinin ürünün formatına göre grafiği 32
Şekil 3. 10 XGBoost ve karışıklık matrisi için gerekli kütüphaneler 32
Şekil 3.11 XGBoost için faktörden numeriğe dönüşüm sonrası veri yapısı 32
Şekil 3. 12 Eğitim ve test veri setlerini oluşturma 33
Şekil 3.13 Parametre ayarlarının yapılması ve açıklamaları 33
Şekil 3.14 Değişkenlerin tanımlanması ve etiketin sıfırdan başlatılması 34
Şekil 3. 15 Çapraz doğrulama ile minimum kayıp tespiti için iterasyon 34
Şekil 3. 16 Logaritmik minimum kayıbın gerçekleştiği adım ve kayıp değerleri 34
Şekil 3. 17 XGBoost modelinin eğitilmesi 35
Şekil 3. 18 Test veri setinde tahminleme 35
Şekil 3. 19 Etiketlerin dönüşümü 35

Şekil 3. 20 Karışıklık matrisinin hesaplanması 35
Şekil 3. 21 Karışıklık matrisi ve istatistikler 36
Şekil 3. 22 Önem matrisinin oluşturulması 36
Şekil 3. 23 Önem matrisi 37
Şekil 3. 24 Önem matrisinin grafiğini çizdirmek 37
Şekil 3. 25 Önem matrisi grafiği 37
Şekil 3. 26 Her bir sınıfın önemini incelemek 38
Şekil 3. 27 Her bir sınıfın önem matris grafiği 38
Şekil 3. 28 ABC sınıfının önem matris grafiği 39
Şekil 3. 29 Bingo sınıfının önem matris grafiği 39
Şekil 3. 30 Dalin sınıfının önem matris grafiği 40
Şekil 3. 31 Hes sınıfının önem matris grafiği 40
Şekil 3. 32 Other sınıfının önem matris grafiği 40
Şekil 3. 33 Private Label sınıfının önem matris grafiği 41
Şekil 3. 34 Vernel sınıfının önem matris grafiği 41
Şekil 3. 35 Viking sınıfının önem matris grafiği 41
Şekil 3. 36 Yumoş sınıfının önem matris grafiği 42
Şekil 3. 37 Naive Bayes için gerekli kütüphaneler 42
Şekil 3. 38 Veri setindeki verilerin yapısı 42
Şekil 3. 39 Eğitim ve test veri setlerini oluşturma 43
Şekil 3. 40 Eğitim setinde modelin eğitilmesi 43
Şekil 3. 41 Test veri setinde tahminleme, karışıklık matrisi ve tahminlerin doğruluğu 43
Şekil 3. 42 Karışıklık matrisi ve doğruluk 44
Şekil 3. 43 Naive Bayes model sonuçları oluşturmak, çalıştırmak ve kayıt etmek 44
Şekil 3. 44 Naive Bayes modelini iteratif çalıştırmak ve özet istatistikler 45
Şekil 3. 45 Naive Bayes model doğruluk değerleri 45

ÇİZELGE LİSTESİ

Sayfa

Çizelge 3. 1 Değişkenlerin açıklaması 26

ÖZET

PERAKENDE SEKTÖRÜNDE MAKİNE ÖĞRENMESİNE DAYALI YAKLAŞIMLAR

Tuğçe CEYLAN

İstatistik Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Prof. Dr. Ali Hakan Büyüklü

Son yıllarda teknolojinin hızla gelişmesi ve veriye ulaşmanın kolaylaşması ile veri biliminde makine öğrenimi algoritmaları önem kazanmıştır. Makine öğrenimi algoritmaları günümüzde bankacılık, finans, teknoloji, biyoistatistik, tıp, v.b. pek çok alanda uygulanmaktadır.

Bu çalışmada, çamaşır yumuşatıcı kategorisinde, tüketicilerin marka alım tercihlerinin sınıflandırmaya dayalı makine öğrenmesi teknikleri kullanılarak belirlenebilmesi amaçlanmıştır. Uygulamada IPSOS Hane Tüketici Paneli verisi kullanılmıştır. Extreme Gradient Boosting ve Naive Bayes Sınıflandırma algoritmaları kullanılarak modellenmiştir. Modeller, verinin %80'lik kısmı ile eğitilmiş ve kullanılmayan test kısmında tahminleme güçleri ölçülerek sınıflandıflandırma performansları karşılaştırılmıştır.

Veri analizi sırasında SQL ve R programı RStudio arayüzü kullanılmıştır. Bu çalışmanın bir bölümü EYİ 2018/Antalya sempozyumunda bildiri olarak sunulmuştur.

Anahtar Kelimeler: Sınıflandırma, makine öğrenmesi, extreme gradient boosting, naive bayes, veri madenciliği

ABSTRACT

APPROACHES TO THE APPLICATION OF MACHINE LEARNING IN THE RETAIL SECTOR

Tuğçe CEYLAN

Department of Statistics

MSc. Thesis

Adviser: Prof. Dr. Ali Hakan BÜYÜKLÜ

In recent years, with the rapid development of technology and easy access to data, machine learning algorithms have gained importance in data science. Machine learning algorithms are applied in many areas such as banking, finance, technology, biostatistics, medicine, etc.

In this study, it is aimed to determine the consumers' brand buying preferences in the category of the laundry softeners by using classification-based machine learning techniques. In practice, IPSOS Household Consumer Panel data is used. Modeled using Extreme Gradient Boosting and Naïve Bayes Classifier algorithms. Models are trained with %80 percent of data and the predictive power was measured in the rest test data which was not used and the classification performances were compared.

SQL and RStudio programs are used for data analysis. A part of this study is presented as a paper at the EYI 2018/Antalya symposium.

Keywords: Classification, machine learning, extreme gradient boosting, naïve bayes, data mining

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Literatür taramasında, makine öğrenmesini konu alan tez çalışmaları ve makaleler incelenmiştir.

Yıldız ve Karatana, şarkıların ait oldukları müzik türlerini belirlemek için Python programlama dili ile makine öğrenmesi algoritmalarını kullanarak sınıflandırma çalışması gerçekleştirmiştir [1]. Onan ve Korukoğlu, görüş madenciliği alanında gerçekleştirilen makine öğrenmesi yöntemlerinin kullanılması üzereine literatür araştırması yapmış ve kullanılan yöntemlerin güçlü ve zayıf yönlerini incelemiştir [2]. Ünsal, mesleki alan seçimleri üzerine makine öğrenmesi algoritması uygulayarak kişiye en uygun mesleğin sistem tarafından önerilebileceği bir yazılım geliştirmiştir [3]. Kartal, hastalarda kardiyolojik riski belirlemek için sınıflandırmaya dayalı makine öğrenmesi teknikleri kullanmıştır [4]. Cüvitoğlu, etkileşen ilaç kombinasyonlarının tahminlenebilmesi üzerine makine öğrenmesi yöntemleri uygulamıştır [5]. Nizam ve Akın, Weka kütüphanesinde yer alan sınıflandırma algoritmalarını kullanarak makine öğrenmesi ile sosyal medyada duygu analizinde dengeli ve dengesiz veri setlerinin performanslarını karşılaştırmıştır [6]. Akçetin ve Çelik, istenmeyen elektronik postaların (spam) tespiti için Weka makine öğrenmesi yazılımını kullanarak karar ağaçları algoritmalarının sınıflandırma performanslarını karşılaştırmıştır [7]. Myazina, bankalarda kredi notlama, kişinin aldığı krediyi ödeyip ödemeyeceği tahminlemesi, problemini makine öğrenimi yöntemleriyle incelemiştir [8]. Burrell, spam filtreleri, kredi kartı sahtekârlığı tespiti, arama motorları, haber eğilimleri, reklamcılık, sigorta

veya kredi yeterliliği ve kredi puanlama gibi sosyal olarak sonuçlanan sınıflandırma ve sıralama mekanizmaları için bir sorun olarak saydamlık konusunu makine öğrenimi algortimaları ile ele almıştır [9]. Bordoloi ve Biswas, duygu analizi için Naive Bayes, Destek Vektör Makinesi ve Maksimum Entropi algoritmalarını kullanarak algoritmaların başarısını karşılaştırmıştır [10]. Pashaei, kanser sınıflandırmasında yüksek düzeyde ayrımcı genlerin seçilmesi için makine öğrenmesine dayalı iki yaklaşım önermiştir [11]. Literatür taramasına bakıldığında, tüketici davranışlarını anlamaya yönelik, hızlı tüketim sektöründe (FMCG) sınıflandırmaya dayalı makine öğrenmesi yöntemlerinin R programı ile uygulanması bu çalışmayı diğer çalışmalardan ayırmaktadır.

1.2 Tezin Amacı

Tezin amacı, karmaşık verilerden anlamlı bilgiyi elde ederek iş dünyasında aksiyona döneşebilecek kararlar alabilmek için makine öğrenmesi algoritmaları ile yol gösterici modeller geliştirmektir. Bu amaçla, makine öğrenmesi kavramları açıklanarak, R programının RStudio arayüzü kullanılarak, perakende sektöründe tüketicilerin çamaşır yumuşatıcısı kategorisinde marka alımlarını sınıflandırmaya yönelik Extreme Gradient Boosting ve Naive Bayes sınıflandırma modelleri geliştirilmiş ve bu modellerin performansları karşılaştırılmıştır.

1.3 Hipotez

Çalışmada, perakende sektöründeki ürün kategorilerinden biri olan çamaşır yumuşatıcısı markalarının, tüketicilerin marka alım tercihleri bağlamında sınıflandırmasının yapılıp yapılamayacağı ve markaların tercih etkenlerinin ayrışıp ayrışmayacağı üzerine hipotezler kurulmuştur. Bu amaçla, Extreme Gradient Boosting ve Naive Bayes sınıflandırma algoritmalarının performansları da dikkate alınarak hipotezler test edilmiştir.

BÖLÜM 2

MAKİNE ÖĞRENMESİ

2.1 Makine Öğrenmesi Nedir

Makine öğrenimi (ML), karmaşık verileri eyleme geçirilebilir bilgiye dönüştüren bir dizi algoritmaların geliştirilmesini araştıran bilgisayar biliminin bir alt alanıdır. Bu alan, ulaşılabilir verinin varlığı, istatistiksel bilgi ve programlama gücünün bir arada kullanılması ile hızla gelişmiştir. Hızla büyüyen veri setlerinin analizinde istatistiksel yöntemleri kullanmak için programlama dillerine ihtiyaç artmasıyla bu alan bilimler arası bir uygulama alanı olmuştur [12].

ML uygulama alanlarının hepsini listelemek mümkün değildir ancak birkaç örnek verecek olursak, hava durumu ve iklim değişikliği tahminleme, evlerde ve ofis binalarında enerji kullanımının optimizasyonu, sahte kredi kartı işlemlerinin tespiti ve azaltılması gibi örnekler verilebilir [12].

Özetle, makine öğrenmesi algoritmaları geçmiş veriler ile eğitilerek, geçmişte bir durumu etkileyen faktörleri ve bu faktörlerin etki derecelerini anlamaya yardımcı olarak gelecekte benzeri durumları tahminlememizi ya da anlık durumu yorumlayabilmemizi, optimize edebilmemizi sağlar.

2.2 Algoritma Öğrenme Yöntemleri

ML algoritmalarının denetimli, denetimsiz ve yarı-denetimli olmak üzere 3 popüler öğrenme yöntemi vardır.

2.2.1 Denetimli Öğrenme

Denetimli öğrenme yöntemi, bir dizi girdi ve girdilerin çıktılarılarının (etiket olarakta adlandırılır) olduğu veri setleri ile kullanılır. Amaç, bilgisayarın girdilerden etiketlerin ne olduğunu öğrenmesi ve tahminleyebilmesidir [13]. Örneğin, araba ve bisiklet görsellerini kategorize etmek istesek algoritmaya neyin araba, neyin bisiklet olduğunu öğretmek için algoritmaya binlerce araba ve bisiklet görseli girilir ve her bir araba ve bisiklet görsel girdisi "araba" ve "bisiklet" olarak etiketlenir. Algoritmanın amacı; arabayı bisikletten ayıran özellikleri doğru bir şekilde ayırt ederek, girdileri başarılı bir şekilde kategorize etmektir. Bunu yapmak için algoritma her bir etiketli araba ve bisiklet görselini analiz eder.

Denetimli öğrenme algoritmaları, sınıflandırma ve regresyon problemleri için kullanılmaktadır. Lineer regresyon, karar ağaçları, destek vektör makineleri, naive bayes, neural networks algoritmaları denetimli öğrenme algoritmalarına örneklerdir [12].

2.2.2 Denetimsiz Öğrenme

Denetimsiz öğrenmede , algoritmaya bir dizi girdinin olduğu ancak etiketlerin olmadığı veri setleri ile başvurulur. Algoritma girdilerdeki yapıyı kendi kendine bulmak zorundadır [13]. Denetimli öğrenmede verilen örneği düşünecek olursak, araba ve bisiklet için girilen görsellere "araba" ve "bisiklet" etiketlemesi yapılmaz. Algoritma her bir görseli analiz ederek araba ve bisikleti birbirinden ayıran farklı özellikleri (teker sayısı, hacim, v.b.) keşfederek görselleri sınıflandırır. k-means kümeleme algoritması denetimsiz öğrenmeye örnek olarak verilebilir [12].

2.2.3 Yarı - Denetimli Öğrenme

Yarı – denetimli öğrenme, veri setinde etiketli ve etiketsiz girdilerin olduğu, başka bir deyişle, denetimli ve denetimsiz öğrenmenin bir kombinasyonudur[13]. Örneğin; analize etiketlenmemiş verilerle başlanırsa, sınıf etiketlerini oluşturmak için kümeleme algoritmaları (denetimsiz öğrenme) kullanılıp, sınıfların en önemli tahminleyicisini

bulmak için karar ağacı algoritması (denetimli öğrenme) kullanılarak yarı denetimli öğrenme yöntemi uygulanmış olur [12].

2.3 Denetimli Öğrenme Yöntemleri

2.3.1 k-En Yakın Komşu

k-NN algoritması, eğitim setinden k en yakın örnekleri kullanarak yeni bir örneği tahmin eder. k-NN bir model tarafından özetlenemez, yapısal olarak eğitim setinden bireysel örneklere dayanır. Yeni örnek için tahmin edilen cevap, k en yakın komşuların cevaplarının ortalamasıdır. Algoritma, kullanıcının örnekler arasındaki uzaklığı nasıl tanımladığına bağlıdır [14].

Örnekler arası uzaklığı tanımlamak için sık kullanılan metriklerden biri öklid uzaklığıdır. x_a , x_b iki ayrı örnek iken öklid uzaklığı

$$\left(\sum_{j=1}^{P}(x_{aj}-x_{bj})^2\right)^{\frac{1}{2}} \qquad (2.1)$$

Minkowski uzaklığı, Öklid uzaklığının bir genellemesidir ve q > 0 iken;

$$\left(\sum_{j=1}^{P}\left|x_{aj}-x_{bj}\right|^q\right)^{\frac{1}{q}} \qquad (2.2)$$

Minkowski uzaklığı, q=2 iken Öklid uzaklığına dönüşür. q=1 iken, Manhattan uzaklığı ile aynıdır. Manhattan uzaklığı, ikili kestiricilerde sıklıkla kullanılan bir metriktir. Daha bir çok uzaklık ölçüm matrisi vardır. Tanimoto uzaklığı, Hamming uzaklığı diğerlerine örnek olarak verilebilir [14] .

kNN, temelde uzaklığa dayalı bir algortima olduğundan dolayı, birbirinden oldukça farklı geniş ölçüdeki tahmin ediciler kestirimde yanlılığa neden olabilir. Potansiyel yanlılıktan kaçınmanın bir yolu olarak her bir tahmin edicinin, kestirimde eşit ağırlıkta katkı sağlaması için ölçeklendirilmesi tavsiye edilir [14].

Ölçekleme konusuna ek olarak, eksik verinin olması da kNN algoritması için bir problemdir. Çözümü için, eksik veri az sayıda ise çok önerilen bir yöntem olmasa da veri setinden çıkarılabilir. Alternatif bir yöntem olarak, eksik veri yerine tahmin edicinin ortalaması eksik gözlemlere atanabilir [14].

Uzaklık metriğini seçtikten sonraki adım en uygun komşu sayısını bulmaktır. k en yakın komşu sayısı tekrarlı örnekleme ile belirlenebilir [14].

2.3.2 Naive Bayes Sınıflandırıcı

Naive Bayes sınıflandırıcılar, "Bayes" teoremine dayanan ve tahmin edicilerin birbirinden bağımsız olduğunu varsayan bir sınıflandırma tekniğidir [15]. Basit bir ifadeyle, bir Naive Bayes Sınıflandırıcı, bir sınıftaki belirli bir özniteliğin varlığının başka herhangi bir özniteliğin varlığına bağlı olmadığını varsayar.

Bayes Teoremi:

$$P(B/A) = \frac{P(A/B)P(B)}{P(A)} \tag{2.3}$$

Burada;

P(B|A); sonsal olasılık.

P(A|B); koşullu olasılık, yani B olayının gerçekleştiği durumda A olayının meydana gelme olasılığıdır.

P(A); A olayının bağımsız olasılığı

P(B); B olayının bağımsız olasılığı , önsel olasılık. Bayes teoremine öznellik katar. P(B), B olayı hakkında öznel olarak sahip olunan bilgidir.

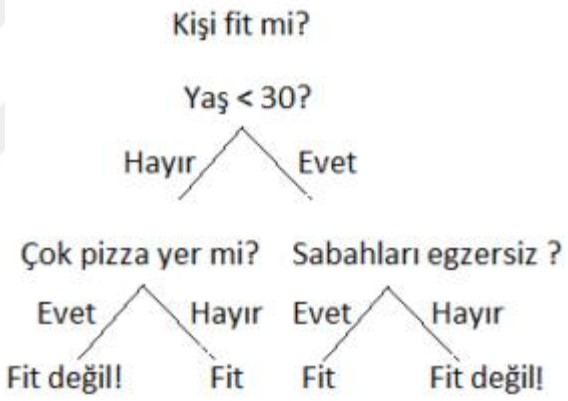
Naive Bayes, her bir sınıf için incelenen örneklerin var olan sınıflara ait olma olasılıklarını Bayes teoremini kullanarak tahmin eder. [8].

Ayrıca maksimum olabilirlik ilkesi kullanılarak parametre tahminlemesi yapıldığında, ortaya çıkan sınıflandırıcı Naive Bayes sınıflandırıcıdır [16].

Naive Bayes sınıflandırıcı, çok büyük veri setleri, ikili veya çoklu sınıflı kategorik veriler ve metin sınıflandırması için oldukça kullanışlı bir modeldir [17].

2.3.3 Karar Ağaçları

Karar ağaçları çok çeşitli problemleri çözmek için kullanılabilecek güçlü ve çok yönlü bir araçtır. Bir karar ağacı, olası sonuçları dikkate alırken karar vermemize yardımcı olan bir akış şemasıdır. Akış şemasında düğümler yer alır. Karar düğümü, iki seçenek arasında seçim yapmayı içeren herhangi bir düğümdür ve genellikle kare ile gösterilir. Olasılık düğümü, belirsiz bir sonucu temsil eden herhangi bir düğümdür ve genellikle daire şeklinde gösterilir. Bitiş düğümü cevaptır ve ağaç daha fazla bölünmez. Bitiş düğümü genellikle terminal düğümü veya yaprak düğümü olarak adlandırılır. Kök düğümü, ağacı başlatan ana soru / karardır. Bir düğümü iki ya da daha fazla alt gruba bölme işlemine bölünme denir. Bölünmenin tersi olarak, bir karar düğümü ile ilgili bir alt düğümü çıkarma işlemine ise budama adı verilir. Çeşitli nedenlerden dolayı düşük öneme sahip olan alt düğümler budama yöntemi ile ağaçtan çıkarılır. Düğümleri birbirine bağlayan tüm ağacın bir alt bölümü dal olarak adlandırılır [13].



Şekil 2.1 Karar ağacı [14]

Karar ağacı algoritmaları, sınıflandırma ve regresyon problemlerinde kullanılabilir. Makine öğrenmesinde karar ağacı oluşturmak için çeşitli algoritmalar vardır. En popüler olarak kullanılan algoritmalar ID3, C4.5, CHAID ve CART'tır [13].

Karar ağacının bölünecek düğümlerinin sırasına karar vermek için saflık metriği kullanılır. İlk düğüm, kök düğümdür. Saflık ölçülmezse, algoritma hangi düğümün ilk düğüm olacağına karar veremez ve ağaç oluşturulamaz. Saflığı ölçmek için Entropi, Gini, Ki-Kare, Information Gain, Variance Reduction ölçüleri kullanılabilir [13].

Karar ağacı algoritmalarının da diğer algoritmalar gibi güçlü ve zayıf yanları vardır. Zayıf yanları, algoritmalarda bilgi kazancı yanlılığı ve aşırı uyum sorunu olabilir [13].

2.3.3.1 ID3

Iterative Dichotomiser 3 (ID3), 1986 yılında Ross Quinlan tarafından geliştirilmiştir [12]. ID3 algoritması, sınıflandırma problemlerinde çok kullanışlıdır ancak regresyon problemlerinde kullanımı güçtür. Regresyon problemlerinde kullanılırken bilgi kazancı (information gain) ölçütü, standart sapma azaltma (standard deviation reduction) ile değiştirilerek kullanılabilir [13]. Algoritmanın uygulama kolaylığı bir avantajdır. Ancak, aşırı uyuma duyarlıdır ve eksik verilerle çalışmak mümkün değildir [13].

2.3.3.2 C4.5

C4.5 algoritması, ID3 algoritmasından sonra yine Ross Quinlan tarafından geliştirilmiştir [12]. Algoritma, kesikli ve sürekli değişkenlerle çalışabilir. Aynı zamanda regresyon ve sınıflandırma problemlerinde, kategorik ve sayısal verilerle çalışabilir. Eksik verilerle çalışmak mümkündür. Budama ve Gain Ratio kullanılarak aşırı uyum problemi çözülebilir [13].

2.3.3.3 CART

CART algoritması, 1984 yılında geliştirilmiştir ve "Sınıflandırma ve Regresyon Ağaçları" olarak bilinir [12]. ID3, C4.5, CHAID algoritmaları çoklu bölünmeler gerçekleştirebilirken, CART algoritması sadece ikili (binary) ağaçlara bölünebilir. Algoritmanın eksik verilerle çalışabilmesi, sınıflandırma ve regresyon problemlerinde kesikli ve sürekli değişkenlerin olduğu data setlerinde kullanılabilirliliği avantajlarıdır [13].

2.3.4 Topluluk Öğrenme Yöntemleri (Ensemble Learning)

Topluluk öğrenme yöntemlerinde topluluk, çeşitli modeller oluşturularak ve tahminler birleştirilerek oluşturulur. Birden çok modelin tahminleri kullanıldığından algoritmaların doğruluk oranı daha yüksek ve başarılıdır. En yaygın olarak kullanılan topluluk öğrenme yöntemlerine Bagging, Boosting, Random Forest ve Gradient Boosting örnek verilebilir [19].

2.3.4.1 Torbalama Algoritması (Bagging)

Karar ağaçları, budama yapılmadan genellikle aşırı uyum (overfit) eğilimindedir ve diğer algoritmalarla kıyaslandığında tahminleme gücü daha düşük olabilir. Ayrıca karar ağaçları yüksek eğitim varyansına sahiptir. Bu nedenlerden dolayı genellikle karar ağacı modellerini iyileştirmede bir topluluk öğrenme yöntemi olan bagging (boostrap aggregation) yöntemi kullanılır [18].

Bu yöntem, Breiman tarafından geliştirilen ilk topluluk öğrenme algoritmasıdır [19]. Bagging algoritmasında, boostrap rasgele örnekleme yöntemiyle (yerine koyarak) çok sayıda karar ağacı oluşturulur. Oluşturulan her bir karar ağacı birbirinden bağımsızdır ve ana veri setinden üretilir. Algortima, regresyon problemlerinde her bir karar ağacının tahminlerinin ortalamasını alarak, sınıflandırma problemlerinde en çok oyu alan sınıfı atayarak nihai modele ulaşır. Bagging karar ağaçları, varyansı düşürerek nihai modeli dengeler, bu durum doğruluğu (accuracy) arttırır [18].

2.3.4.2 Rasgele Ormanlar (Random Forests)

Bagging karar ağaçlarında kullanılan, boostrap örnekleme üretme yöntemi ağaç oluşturma sürecinde rasgeleliği sağlar. Ancak, bagging karar ağaçları birbirinden tamamen bağımsız değildir çünkü veri setindeki tüm tahminleyiciler her ağacın her bölünmesinde kullanılır. Bu durum, ağaç korelasyonu olarakta bilinir ve bagging yöntemiyle tahminlenen değerde varyansın azaltılmasına engel olur. Her ağaç benzersiz olsa da hepsi benzer bir yapıyla başlar ve sonuç olarak birbiriyle ilişkilidir. Bu nedenle, bagging yöntemiyle sağlanan varysans azaltımı daha da geliştirilebilir [15].

Random Forest yaklaşımı, her ağaç için değişken kümesini rasgele ayırarak ağaçların birbiriyle olan korelasyonunu azaltılır. Topluluktaki her bir ağaç için, öncelikle boostrap rasgele örnekler üretir. Her bir örnek için ve ağacın her bir düğümünde, veri setindeki tüm değişkenler arasından belirlenmiş sayıda rasgele değişkenler seçerek bir alt küme oluşturur ve değişkenlerin alt kümesinden en iyi bölünmeyi sağlayan en iyi değişkeni seçer. Nihai ağaç toplulukları, orman tahminlerini yapmak için torbalanır (bagged) [18].

Algoritma, en iyi bölünmeyi sağlayacak değişkeni seçmek için entropi ve her bir bölünmenin saflığını ölçmek için bilgi kazanımı ölçütünü kullanır. En yüksek saflığa sahip bölünme seçilir [13].

Rasgele ormanların değişken önem ölçütleri, modele en çok hangi değişkenlerin katkıda bulunduğunu belirlemede yardımcı olabilir. Random forests algoritmasında, model çıktıları iyileştirilebilir [18].

Rasgele ormanlar tahmini, oy çokluğuyla tahmin ve ortalama ile tahmin olmak üzere iki şekilde yaparlar. Genellikle, kategorik sınıflarda oy çokluğu tahminlemesi yaklaşımı kullanılır yani ormandaki tüm ağaçlar içinde en çok tahmin edilen sınıf nihai tahminleyici seçilir. Tahminleme için ikinci yöntemde ise ormandaki tüm ağaçların sonuçlarının ortalaması kullanılır [13].

Makine öğrenmesi algoritmalarında genellikle doğruluk, algoritma oluşturulduktan sonra özel olarak ayrılmış olan veri kümesinde test edilir. Rasgele ormanlar, algoritma çalıştırılırken doğruluğu test etmek için kullanılabilecek bir veri kümesine sahiptir. Out-of-Bag (O.O.B.) hata tahminleyicisi rasgele ormanların doğruluğunu test etmek için kullanılan bir yöntemdir. O.O.B hatası 0 ile 1 arasında bir değer alabilir ve sıfıra ne kadar yakınsa o kadar iyidir [13].

Yüzlerce veya binlerce değişken ile çalışılıyorsa ve hangi değişkenin önemli, hangilerinin önemsiz olduğu araştırılıyorsa rasgele ormanlar çok büyük ve yorumlanması zor algoritmalar olabilir. Ayrıca, değişkenlerin önemini anlamak için de O.O.B hatası yaygın olarak kullanılmaktadır [13].

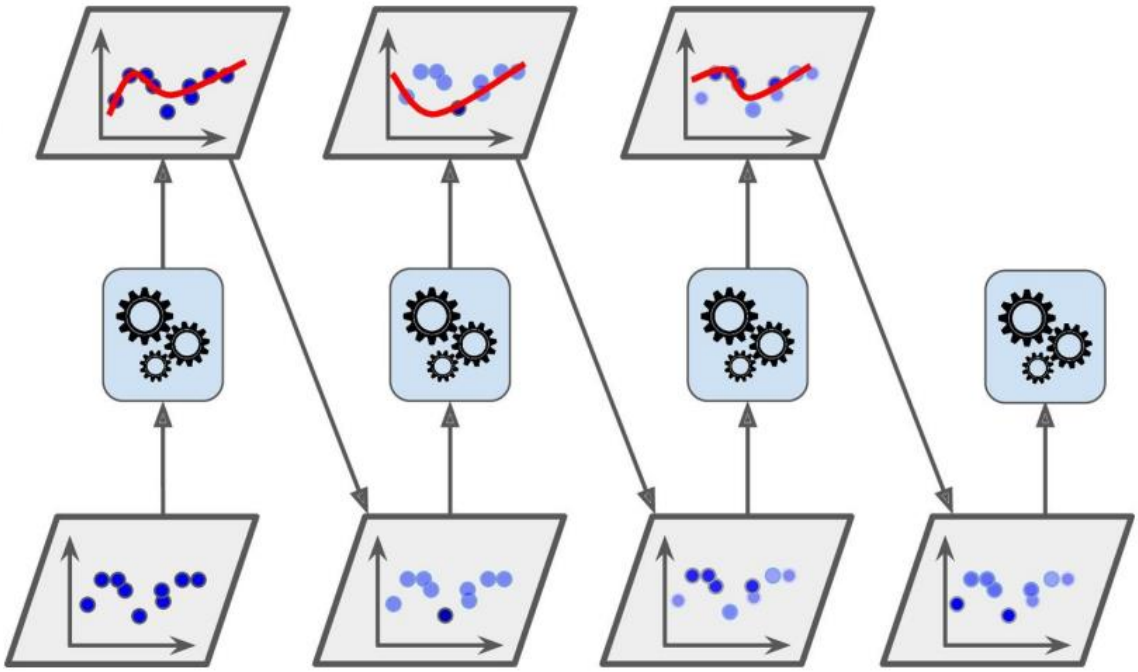
2.3.4.3 Güçlendirme Algoritması (Boosting)

Boosting yöntemi, Robert Schapire ve Yoav Freund tarafından geliştirilmiştir [8]. Topluluktaki zayıf sınıflandırıcıların bir araya getirilmesi ile güçlü bir sınıflandırıcı oluşturmayı amaçlayan zayıf öğrenme varsayımına dayanan iteratif bir yaklaşımdır [21].

Boosting algoritması, ilk olarak ikili sınıf problemleri için tasarlanmıştır. Daha sonra, AdaBoost (Adaptive Boosting) ve çeşitli varyasyonları çoklu sınıflandırma ve regresyon problemleri için geliştirilmiştir [20].

Pek çok boosting yöntemi vardır ve bu yöntemlerden en yaygın kullanılanlar Adaboost ve Gradient Boosting yöntemleridir [22].

Bir tahminleyicinin, kendinden önceki tahminleyicinin eğitim verilerini eksik öğrenmesinin düzeltilmesi Adaboost Algoritmasında kullanılan bir yöntemdir. Örneğin; bir Adaboost sınıflandırıcısı oluşturmak için, bir ilk temel sınıflandırıcı eğitilir ve eğitim seti üzerinde tahminler yapılır. Eksik sınıflandırılan eğitim örneklerinin nispi ağırlığı artar. İkinci bir sınıflandırıcı güncellenmiş ağırlıklar kullanılarak eğitilir ve eğitim setinde güncellenmiş ağırlıklar ile yeniden tahminler yapılır. İterasyon, belirli bir öğrenme sayısı kadar tekrar edilerek tüm tahmin ediciler eğitildikten sonra tahminleyicilerin ağırlıklı ortalaması alınarak ya da sınıf oylaması yapılarak, nihai tahmin elde edilir [22].



Şekil 2.2 Ağırlık güncellemeleri ile Adaboost sıralı eğitim [22]

AdaBoost yönteminde olduğu gibi Gradient Boosting yöntemi de topluluğa adımsal olarak tahminleyicileri ekler ve her bir tahminleyici, bir önceki tahminleyiciyi düzeltir. Ancak Adaboost yöntemindeki gibi her yinelemede tahminleyici ağırlıklarını yeniden düzenlemek yerine, bu yöntem önceki tahminleyici tarafından yapılan artık hatalara yeni tahminleyiciyi uydurmaya çalışır. Gradient Boosting algoritması, resgresyon problemlerinde de oldukça iyi çalışır [22].

Gradient boosting ağaç modellerinin hızını ve performansını yükseltmek için Tianqi Chen tarafından, gradient boosting karar ağacı algoritmasını uygulayan ve makine öğrenmesinde yaygın olarak kullanılan XGBoost (extreme gradient boosting)

kütüphanesi geliştirilmiştir. XGBoost, pek çok karmaşık sınıflandırma ve regresyon tahmin modelleme probleminde oldukça iyi performans gösterir ve aşırı uyum sorununu gradient boosting machine (gbm) kütüphanesine göre daha iyi kontrol ederek, model performansını arttırır [23].

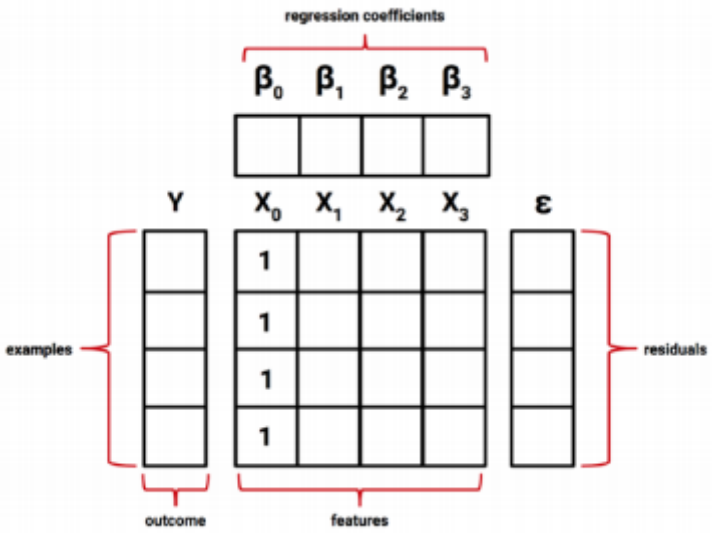
2.3.5 Çoklu Doğrusal Regresyon

Genel olarak, doğrusal bir model, girdi değişkenlerin ağırlıklı bir toplamını hesaplayarak bir tahmin yapar ve model bir kesişme terimi olarak adlandırılan bir sabit içerir [22].

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_i x_i + \varepsilon \tag{2.4}$$

Bu eşitlikte; y tahminlenen değeri, i değişken sayısını, x_i i. değişkenin değerini, β_1 ,..., β_i değişkenlerin bilinmeyen regresyon katsayılarını, α bağımsız değişkenlerin sıfır olması durumunda beklenen değeri (sabiti), ε hata terimini gösterir. Doğrusal regresyon modelinde tahminlenen değerler ile gerçek değerler arasındaki farklar, tahminlerin hatasıdır. Bir doğrusal regresyon modelini eğitmek için, ortalama hata kareyi (MSE) en aza indiren değer bulunmalıdır [22].

Regresyon parametrelerinin tahmini için, bağımlı değişken y de gözlenen her bir değer, x bağımsız değişkenlerinde gözlenen değerler ile ilişkili olmalıdır [12].



Şekil 2.3 Regresyon tahmin yapısı [12]

Şekil (2.3) de görülen satır ve sütun verileri, terimlerin her birinin birden çok değeri temsil ettiğini göstermek için kalın yazı tipi matris notasyonu kullanılarak yoğunlaştırılmış bir formülasyonda açıklanabilir:

$$Y = \beta X + \varepsilon \tag{2.5}$$

Her örnek için, bağımlı değişken Y artık bir satır vektörüdür. Bağımsız değişken X ler, her değişken için bir sütun içeren bir matriste birleştirilir, sabit terim için matrise 1 sütunu eklenir. β regresyon katsayıları ve ε hata terimleri de artık vektördürler. Amaç artık β vektörünü çözmektir ve tahminlemek için;

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{2.6}$$

formülüzasyonu kullanılabilir [12].

Gerçek dünyada genellikle analizlerde birden fazla bağımsız değişken ile karşılaşırız ve sayısal verileri modellemek için çoklu doğrusal regresyon, kullanılan en yaygın yaklaşımlardan biridir. Çoklu doğrusal regresyon modeli, değişkenler ve çıktı arasındaki ilişkilerin hem gücü hem de büyüklüğüne dair tahminler sağlar. Ancak, modeli kullanabilmek için sağlanması gereken varsayımlar, eksik verilerle kullanılmaması, sadece sayısal verilerle çalışması, kategorik verileri kullanabilmek için ekstra işlem gerekliliği modelin dezavantajlarıdır. Doğrusal regresyon modellerinde gerekli varsayımlar; hataların normal dağılması ve birbirinden bağımsız olması, değişkenler ile hata terimi arasında ilişki olmaması, bağımsız değişkenlerin birbirinden bağımsız olması ve her bir gözlem için hata terimi varyanslarının sabit olmasıdır [12].

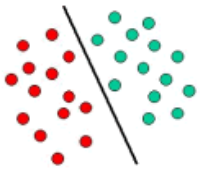
2.3.6 Lojistik Regresyon

Lojistik regresyon, genelleştirilmiş doğrusal modeller sınıfının en önemli ve sıklıkla kullanılan üyesidir. İkili sınıflandırma problemleri analizi için iyi bir ilk tercih olabilir. Lojistik regresyon her zaman 0 ve 1 arasında bir değer tahminler. Bu da gözlenen çıktı kategorik bir değer olduğunda, olasılıkları tahmin etmeyi veya gözlenen çıktı oran olduğunda, oranları tahmin etmeyi sağlar. Lojistik regresyon modeli, sadece tahminleme yapmakla kalmaz, doğrusal regresyon modelinde olduğu gibi girdi değişkenler ile sonuç arasındaki ilişki hakkında bilgi de verir [18].

Lojistik regresyon, bir örneğin belirli bir kategoriye ait olması olasılığını tahminler. Sigmoid işlevi, gerçek değerleri (0,1) aralığına yani olasılıklara eşler. Sigmoid'in tersi, log (p / (1-p)) olarak tanımlanan logittir (p bir olasılıktır). p/ (1-p) oranı, odds olarak bilinir. Odds oranı, bir olayın gerçekleşme olasılığının gerçekleşmeme olasılığına oranıdır. Lojistik regresyon, ilgilenilen olasılığın log-odds değerlerini bulan bir doğrusal regresyon olarak düşünülebilir [18].

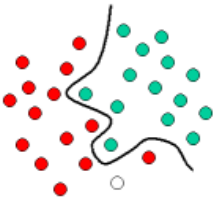
2.3.7 Destek Vektör Makineleri (Support Vector Machines)

Destek vektör makineleri, karar sınırlarını tanımlayan karar düzlemleri kavramına dayanır. Bir karar düzlemi, farklı sınıf üyeliklerine sahip nesneler kümesi arasında ayrım yapan bir düzlemdir. Şekil (2.4) te şematik bir örnek gösterilmiştir. Örnekte, nesneler yeşil veya kırmızı sınıfına aittir. Ayırma çizgisi, tüm nesnelerin yeşil olduğu ve solunda tüm nesnelerin kırmızı olduğu bir sınır tanımlar. Yani, sağa düşen herhangi bir nesne yeşil olarak sınıflandırılır [24].



Şekil 2.4 Doğrusal bir sınıflandırıcı örneği [24]

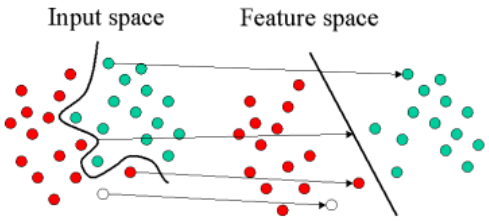
Çoğu sınıflandırma probleminde, mevcut olan örneklere (eğitim seti) dayanarak yeni nesneleri (test seti) doğru bir şekilde sınıflandırmak için daha karmaşık yapılara ihtiyaç duyulur.



Şekil 2.5 Eğrisel bir sınıflandırıcı örneği [24]

Şekil 2.5'i , Şekil 2.4 ile kıyaslarsak yeşil ve kırmızı nesneleri ayırmak için Şekil 2.5' te bir eğriye ihtiyaç duyulur. Destek vektör makineleri, faklı sınıf üyelikleri arasındaki

nesneleri ayırmak için ayırma çizgilerinin çizilmesine dayanan sınıflandırma problemleri kullanımında oldukça uygun bir yöntemdir [24].



Şekil 2.6 SVMs görsel örneği [24]

Şekil 2.6, Destek vektör makinelerinin arkasındaki temel fikri göstermektedir. Şemanın sol tarafındaki orjinal nesneler, kernels olarak bilinen bir dizi matematiksel fonksiyon kullanılarak yeniden düzenlenmiştir. Yeniden düzenleme işlemi, dönüşüm olarak bilinir [24].

SVM, farklı sınıf etiketlerini birbirinden ayıran çok boyutlu alanda hiper düzlemler oluşturarak sınıflandırma problemlerinde kullanılmak üzere geliştirilen bir metoddur. Hem regresyon hem de sınıflandırma problemlerinde kullanılabilir ve birden çok sürekli veya kategorik değişkenleri işleyebilir. Çok sınıflı kategorik değişkenler için, dummy değişkenler oluşturulabilir. Örneğin; üç sınıflı kategorik bir bağımlı değişken, üç kukla değişken ile temsil edilir [24].

SVM, optimal bir hiper düzlem oluşturmak için hata fonksiyonunu en aza indirmek amacı ile iteratif bir eğitim algoritması kullanır.

2.3.8 Yapay Sinir Ağları (Artificial Neural Networks)

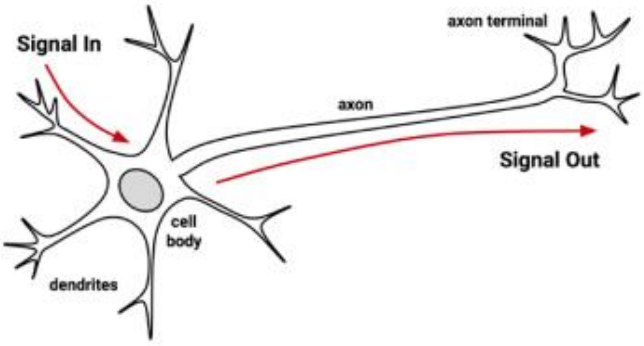
Sinir ağları, fonksiyonlar modellemek için hayvan beyninin yapısını taklit eder. Bir beyin, devasa bir paralel işlemci oluşturmak için nöron adı verilen birbirine bağlı hücreleri kullanırken, ANNs öğrenme problemlerini çözmek için yapay sinir ağı veya düğümleri kullanır.

Son yıllarda bilgisayarların güçlenmesi ile ANNs karmaşık yapısı da benzer şekilde artmış ve bazı pratik problemlere (ses ve el yazısı tanıma, akıllı cihazların otomasyonu, vb.) sıklıkla uygulanmaya başlamıştır. ANNs, sınıflandırma, sayısal tahmin ve hatta denetlenmeyen örüntü tanıma problemlerinde kullanılabilir. ANNs, iyi tanımlanmış

girdi ve çıktı verilerinin bulunduğu problemlerde uygulanabilir ancak girdi verileri çıktıyla ilişkilendiren süreç oldukça karmaşıktır [12].

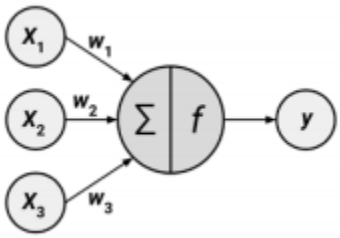
2.3.8.1 Biyolojik ve Yapay Nöronlar

Biyolojik nöronların nasıl işlediğini anlamak, insan beyni aktivitesini örnek alan yapay sinir ağlarını anlamak için yardımcı olabilir. Gelen sinyaller, biyokimyasal bir süreçle hücrelerin dendritleri tarafından alınır. İşlem, impulsun görece önemine veya sıklığına göre ağırlıklandırılmasını sağlar. Hücre gövdesi gelen sinyalleri biriktirmeye başladıkça, hücrenin patladığı ve çıkış sinyalinin akson boyunca bir elektrokimyasal işlem yoluyla iletildiği bir eşiğe ulaşılır. Aksosun terminallerinde, elektrik sinyali sinaps olarak bilinen komşu nöronlara geçecek bir kimyasal sinyal olarak işlenir [12].



Şekil 2.7 Biyolojik sinir hücresi [12]

Bir yapay nöronun modeli biyolojik nörona çok benzer olarak anlaşılabilir. Yönlendirilmiş bir ağ diyagramı, x değişkenleri (dendritler) tarafından alınan giriş sinyalleri ve y değişkeni (çıkış sinyali) arasındaki ilişkiyi tanımlar. Biyolojik nöronda olduğu gibi her bir dendritin sinyali, önemine göre ağırlıklandırılır (w değerleriyle). Giriş sinyalleri hücre gövdesi tarafından toplanarak f ile gösterilen aktivasyon işlevine göre iletilir [12].



Şekil 2.8 Yönlendirilmiş ağ diyagramı [12]

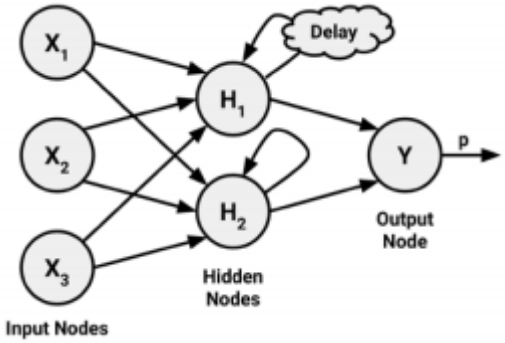
Hücre gövdesindeki sinyallerin toplamı, f(x) aktivasyon fonksiyonu tarafından kullanılır ve elde edilen sinyal y(x), çıktı aksonudur [12].

$$y(x) = f\left(\sum_{i=1}^{n} w_i x_i\right) \tag{2.7}$$

Özetle, yapay sinir ağları karmaşık veri modelleri oluşturmak için yapı taşı olan nöronları kullanarak girdi sinyallerini tek bir çıktı sinyaline dönüştüren bir aktivasyon fonksiyonu oluşturur. Ağ mimarisi, modeldeki nöronların bağlı olduğu katmanların ve nöronların sayısını açıklar. Eğitim algoritması, girdi sinyaline orantılı olarak nöronların uyarılması için bağlantı ağırlıklarını belirler.

Karmaşık ağlar oluşturmak için ek katmanlar eklenebilir. Çok katmanlı bir ağ, çıkış düğümüne ulaşmadan önce giriş düğümünden gelen sinyalleri işleyen bir veya daha fazla gizli katmana sahiptir. Çok katmanlı ağlar genellikle birbirine bağlantılıdır [12].

Bilgi akışı yönü tek yönlü olan ağlar ileri beslenme ağlarıdır ancak bilgi akışı, karmaşık yapılarda genellikle tek yönlü değildir. Tekrarlayan bir ağ (geri besleme ağı), sinyallerin döngülerini kullanarak her iki yönde hareket etmesine izin vererek karmaşık modellerin öğrenilmesini sağlar. Kısa süreli bir hafıza veya gecikme eklenmesi geri besleme ağlarının gücünü önemli ölçüde arttırır ve bir süre boyunca olayların sıralarını anlama yeteneği de sağlar [12].



Şekil 2.9 Gizli katmanlı tekrarlı ağ örneği [12]

Çok katmanlı ağlarda en sık kullanılan algoritma geri yayılım (back-propagating) ağlarıdır. Geri yayılma hataları stratejisi, ağ çıktısı ve gerçek çıktı arasında kabul

edilebilir minimum hata elde edilene kadar iteratif olarak girdilerin ağırlıklarını değiştirerek sinir ağını eğitmeye dayanır [12].

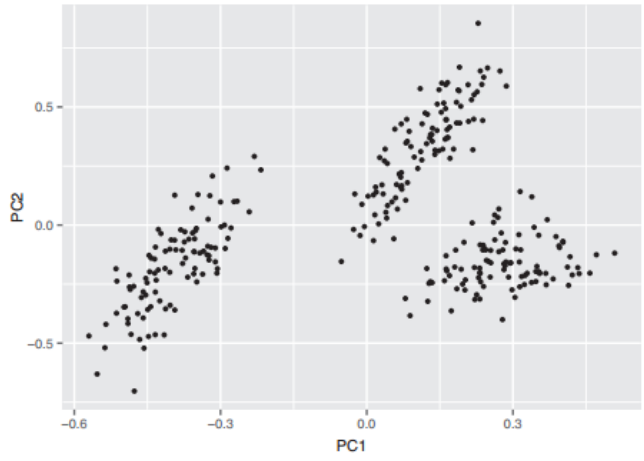
2.4 Denetimsiz Öğrenme Yöntemleri

2.4.1 Kümeleme

Kümeleme analizi, benzer türdeki nesneleri ilgili kategorilere ayırmak için bir dizi farklı algoritmayı ve yöntemi kapsayan bir denetimsiz öğrenme yöntemidir. Farklı ve çeşitli veriler az sayıda grup tarafından örneklendirilebiliyorsa, kümeleme yöntemi kullanmak uygundur. Karmaşıklığı azaltan ve ilişki kalıplarına içgörü sağlayan anlamlı veri yapıları oluşturur [12].

Denetimli öğrenme yöntemlerinin etiketlenmiş eğitim verilerini kullanarak hipotezlerin başarısını veya riskini tahminlemesinin aksine, kümeleme yönteminde tahmin edilmeye çalışılan hiçbir etiket yoktur. Bunun yerine amaç, verileri anlamlı bir şekilde düzenlemektir. Kümeleme için net bir başarı değerlendirme prosedürü yoktur [17].

Kümeleme için, benzerlik ve uzaklık kavramları kullanılır. En sık kullanılan benzerlik ve uzaklık ölçütleri; Öklid uzaklığı, Hamming uzaklığı, Manhattan uzaklığı ve Cosine benzerliğidir [18].



Şekil 2.10 Üç kümeli bir kümeleme örneği [18]

2.4.1.1 k-Means Kümeleme

k-Means algoritması, yaygın olarak kullanılan bir kümeleme yöntemidir. k-means, n sayıda örneğin her birini, önceden sayıca belirlenmiş k kümelerinden birine atar. Amaç,

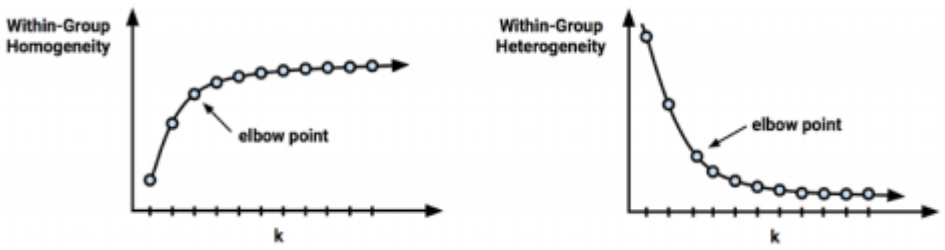
kümeler içi homojenliği arttırıp, kümeler arası heterojenliği sağlamaktır. İlk olarak, bir başlangıç kümeleri kümesine örnekler atar. Daha sonra, kümeye düşen örneklere göre kümelenme sınırlarını ayarlayarak günceller. Atama ve güncelleme işlemi, kümelenme uyumu daha fazla iyileştirilemeyene kadar devam eder ve nihai kümeler kesinleştirilerek süreç durdurulur. k-Means algoritmasında en sık kullanılan uzaklık ölçütü, Öklid uzaklığıdır [12].

Öklid uzaklığı;

$$dist(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.8}$$

formülü ile hesaplanır. Bu uzaklık fonksiyonu kullanılarak, her bir örneğin küme merkezine uzaklığı bulunur ve örnek, en yakın küme merkezine atanır. Uzaklık hesapları kullanılırken tüm değişkenlerin sayısal olması ve standart normalleştirme gereklidir.

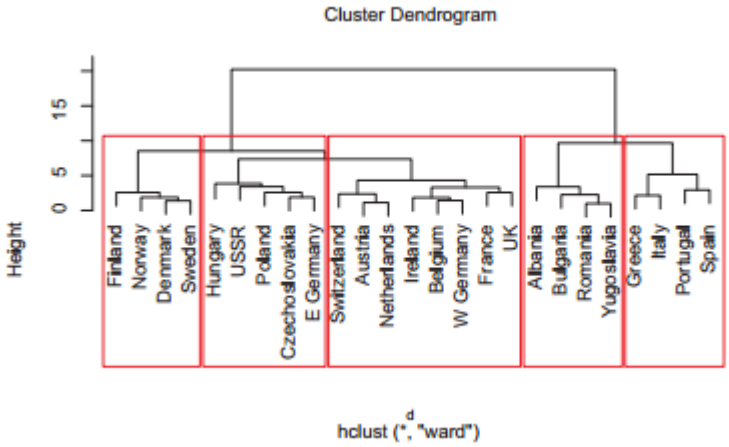
k-Means algoritmasında küme sayısı önceden kişi tarafından belirlenmelidir. Genellikle, analiz edecek kişinin önsel bilgisi ile küme sayısı belirlenebilir. Ya da örneğin bir şirketin üç farklı kampanya yapacağı kesindir ve direk üç küme oluşturmak gerekliliği vardır. Eğer araştırmacının önsel bilgisi yoksa veya net bir küme sayısına ihtiyacı yoksa, k küme sayısına karar vermek için dirsek metodu kullanılabilecek yöntemlerden biridir. Dirsek metodu, kümeler içi homojenliği veya heterojenliği değişik k değerlerinde ölçer. Dirsek noktası optimum k sayısını verir. Ancak bu yöntem çok büyük veri setlerini kümelemek gerektiğinde uzun zaman alabileceğinden, uygulanması her zaman mümkün olmayabilir. [12].



Şekil 2.11 Dirsek metodu [12]

2.4.1.2 Hiyerarşik Kümeleme

Hiyerarşik kümeleme, kümelenmiş kümelenme gruplarını bulur. Verilerin içindeki tüm nokta çiftleri arasındaki mesafeleri kaydeden bir uzaklık matrisini girdi olarak alarak, iç içe geçmiş kümeleri temsil eden bir ağaç olan dendograma döndürür [18].



Şekil 2.12 Protein tüketimi bakımından kümelenen ülkelerin dendogram örneği [18]

2.4.2 Boyut İndirgeme

Makine öğrenimi problemlerinde bazen eğitim örneği binlerce özellik içerebilir. Bu durum eğitimi aşırı yavaşlatabilir ve iyi bir çözüm bulmayı zorlaştırabilir. Boyut indirgeme yöntemleri bu sorunu çözmek için uygulanabilir. Bazı durumlarda, eğitim verilerinin boyutunun azaltılması gürültü ve gereksiz detayları filtreleyerek, algoritmanın performansını arttırabilir. Ancak, boyut indirgerken bir miktar bilgi kaybı da yaşanır. Bu yüzden, boyut indirgeme yöntemlerinin eğitimin aşırı yavaş olduğu durumlarda kullanılması tavsiye edilir. Temel Bileşenler Analizi (PCA), sıklıkla kullanılan bir boyut azaltma algoritmasıdır. PCA da amaç az sayıda değişkenle ana veriyi temsil edebilmektir [22].

2.5 Modelleri Test Etme Ve Doğrulama

Bir modelin yeni durumlara ne kadar iyi genelleneceğini bilmenin tek yolu, aslında yeni durumlarda denemektir.

Bunu yapmanın bir yolu, verileri iki gruba ayırmaktır: eğitim seti ve test seti. Eğitim seti kullanılarak model eğitilir ve test setini kullanarak test edilir. Yeni durumlarda hata

oranı, genelleme hatası (veya örneklem dışı hata) olarak adlandırılır ve modelinizi test kümesinde değerlendirerek, bu hatanın bir tahminini alırsınız. Bu değer, modelin daha önce hiç görmediği durumlarda ne kadar iyi performans göstereceğini anlatır [18].

Verilerin, genellikle 2/3'ü eğitim seti, kalan 1/3'ü test seti olarak bölünür. Bu yöntem, sınama seti yaklaşımı (holdout) olarak adlandırılır.

Sıklıkla kullanılan diğer bir yöntem ise, k katlı çapraz doğrulama (k-fold-cross-validation) yöntemidir. k katlı çapraz doğrulamanın temelindeki fikir, modelin yapımının mevcut eğitim verilerinin farklı alt kümeleri üzerinde tekrarlanması ve daha sonra modelin yalnızca son aşamada görülmeyen veriler üzerinde değerlendirilmesidir. Yani, gelecekteki veriler üzerine modelin performansını simüle etme girişimidir [18].

2.6 Modelleri Değerlendirme

Makine öğrenmesi algoritmalarında, kurulan modellerin başarısı değerlendirilmelidir. Modellerin değerlendirilmesi için doğruluk (accuracy), karışıklık matrisi (confusion matrix), kesinlik (precision), anma (recall), ROC eğrisi gibi yöntemler kullanılabilir [22].

2.6.1 Karışıklık Matrisi

Sınıflandırıcı performansının ölçülmesi için sıklıkla karışıklık matrisi (confusion matrix) kullanılır. Örneklerin sayısını, gerçek sınıf ve tahmin edilen sınıf oranları olarak özetleyen bir tablodur [25].

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Şekil 2.13 Karışıklık matrisi [25]

Bir karışıklık matrisinde her satır gerçek sınıfı, her sütun tahmini bir sınıfı temsil eder. Mükemmel bir sınıflandırıcı sadece gerçek pozitif (TP) ve gerçek negatif (TN) değerlere sahip olurken diğer gözeler sıfır olur [22].

Karışıklık matrisi yerine daha kısa bir metrik tercih edilmek istenirse, sınıflandırıcının kesinliğine bakılabilir.

Kesinlik;

$$precision = \frac{TP}{TP + FP} \tag{2.9}$$

formülü ile hesaplanır.

TP gerçek pozitiflerin sayısı, FP yalnış pozitiflerin sayısıdır. Kesinlik genellikle anma (recall) metriği ile birlikte kullanılır. Anma, gerçek pozitif oran (TPR) olarakta adlandırılır. Anma, sınıflandırıcı tarafından doğru bir şekilde saptanan pozitif örneklerin oranıdır [22].

Anma;

$$recall = \frac{TP}{TP + FN} \tag{2.10}$$

ile hesaplanır.

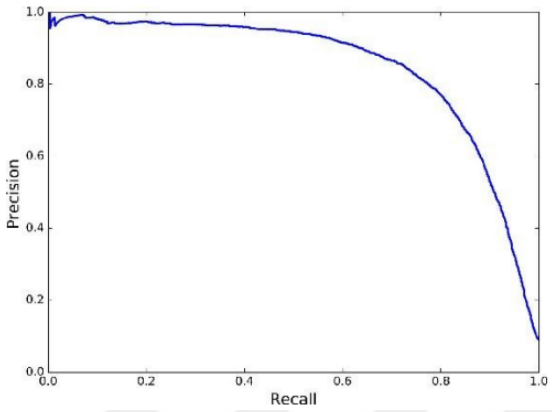
Özellikle iki sınıflandırıcıyı karşılaştırmak için basit bir yönteme ihtiyaç varsa, kesinlik ve anma metriklerini F1 skoru olarak adlandırılan tek bir metrikle birleştirebiliriz. F1 skoru, anma ve kesinliğin harmonik ortalamasıdır. Ortalama, tüm değerleri eşit olarak ele alırken, harmonik ortalama düşük değerlere daha fazla ağırlık verir. Sonuç olarak, hem kesinlik hem de anma değerlerinin yüksek olduğu durumda F1 skoru yüksek bir değer alacaktır [22].

F1 skoru;

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = 2 * \frac{precision * recall}{precision + recall} = \frac{TP}{TP + \frac{FN + FP}{2}} \tag{2.11}$$

formülü ile hesaplanır.

Artan kesinlik, anmayı azaltır ya da artan anma, kesinliği azaltır. Bu durum kesinlik/anma takası (precision/recall tradeoff) olarak adlandırılır. En iyi kesinlik/geri çağırma takası sağlayan eşik değeri seçilebilir.



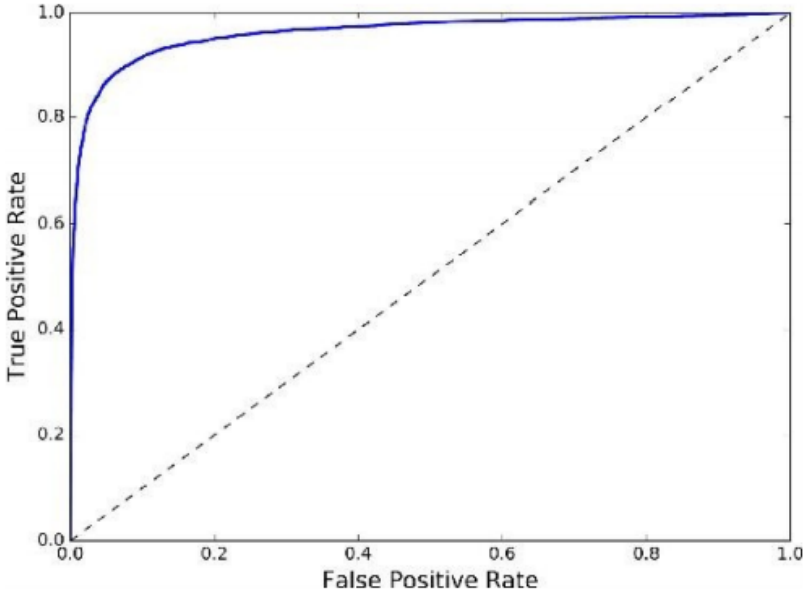
Şekil 2.14 Kesinlik versus anma örnek grafik [22]

Bu eşik değeri seçimi için yöntemlerden biri, kesinliğin anmaya karşı grafiğinin çizilmesidir. Eşik değeri seçimi, çözülmek istenilen probleme bağlı olarak, araştırmacı tarafından belirlenebilir [22].

2.6.2 ROC Eğrisi

Alıcı işlem karakteristiği eğrisi (ROC eğrisi), ikili sınıflandırıcılar için kullanılan yaygın bir model değerlendirme yöntemidir.

ROC eğrisi, yanlış pozitif orana (FPR) karşı gerçek pozitif oranı (TPR) çizer. FPR, yanlış olarak pozitif olarak sınıflandırılan negatif örneklerin oranıdır [22] .



Şekil 2.15 ROC Eğrisi [22]

Modelin tahmin başarısını değerlendirmenin bir yolu, eğri altındaki alanı (AUC) ölçmektir. Mükemmel bir sınıflandırıcı 1'e eşit bir AUC değerine sahip olacaktır. Yani AUC değeri 1'e ne kadar yakınsa, modelin sınıflandırma başarısı o kadar iyidir.

BÖLÜM 3

UYGULAMA

Çalışmada, IPSOS Araştıma Şirketi'nin Hane Tüketici Panel verisi kullanılmıştır. IPSOS Hane Tüketici Paneli, hızlı tüketim ürünlerinin (FMCG) 2004 yılından bu yana Türkiye'nin örneklemesi yapılarak her yıl yaklaşık 14000 haneden tüketim verilerinin toplanıp, depolandığı bir paneldir. İstatistiksel örnekleme yöntemiyle seçilen yaklaşık 14000 hanenin demografik özellikleri ve tüketim verisi ile tüketicilerin, Türkiye'deki alım davranışlarının incelenmesi hedeflenerek, şirketlere içgörü sağlanmaktadır. Çalışmanın amacı, çamaşır yumuşatıcısı kategorisinde, tüketicilerin marka alım tercihlerinin sınıflandırmaya dayalı makine öğrenmesi teknikleri kullanılarak belirlenebilmesidir. Uygulamada, eXtreme Gradient Boosting ve Naive Bayes Sınıflandırıcı algoritmaları kullanılarak, algoritmaların performans karşılaştırmaları yapılacaktır.

3.1 SQL Database Server ile RStudio Bağlantısı

Uygulama verisi, 90 aylık (Ocak 2011 – Haziran 2018 arası) çamaşır yumuşatıcısı kategorisi hane tüketim verisi olduğundan verinin boyutu çok büyüktür. Bu nedenle, RStudio programı ile SQL Database Server'ı arasında bağlantı kurularak, veri RStudio programına aktarılmıştır.

SQL Database Server'ı ile RStudio arasında bağlantı kurmanın birden fazla yöntemi vardır.

Uygulamada kullanılan yöntem:

```
library(rJava)
library(RJDBC)
library(DBI)


# Create connection driver and open connection
jdbcDriver <- JDBC("oracle.jdbc.OracleDriver",
                   classPath="C:/Program Files/Java/jre1.8.0_171/lib/ojdbc6.jar", " ")
jdbcConnection <- dbConnect(jdbcDriver, "jdbc:oracle:thin:@//database.hostname.com:port/service_name_or_sid",
                            "username", "password")

dataframe <- dbGetQuery(jdbcConnection, " SELECT ___ FROM ___
                        WHERE __
                        AND TRN00_PERIOD BETWEEN 201101 AND 201806
                        GROUP BY _____ ")
```

Şekil 3.1 SQL Database bağlantısının yapılması ve verinin aktarılması

3.2 Veri Düzenleme ve Değişken Açıklama

Uygulamaya ilk olarak 285.692 gözlem ve 19 değişken ile başlanmıştır. Değişkenlerin hepsi kategorik değişkendir. Tarih ve hanenin ekonomik sınıf (SES düzeyi) değişkenleri, birden fazla değişkende birbirinin farklı kodlanmış hali olduğundan 4 değişken çalışmadan çıkarılarak değişken sayısı 15 değişkene indirilmiştir.

Çizelge 3. 1 Değişkenlerin Açıklaması

Değişken Adı	Açıklaması
TRN00_PERIOD	Alışverişin yapıldığı tarih(yıl-ay)
SHP00_TYPE2	Alışverişin yapıldığı market (örn; Carrefour Total,Migros,...)
SHP00_CTYPE2	Alışverişin yapıldığı kanal(örn;discounter,local chain, national chain,...)
F46	Ürünün hacimsel aralık kategorisi
F47	Ürünün fiyat kategorisi
F55	Ürünün format (konsantre,...) kategorisi
F44	Ürünün renk kategorisi
F65	Ürünün varyant(koku detayı) kategorisi
FAM00_SES_GROUP	Hanenin SES düzeyi
FAM00_PAX_GROUP	Hane kişi sayısı kategorisi
FAM00_P_AGE_GRP	Hanedeki kişinin yaş aralığı
FAM00_P_EDU_GRP	Hanedeki eğitim düzeyi
FAM10_REGION	Hanenin kent-kır yaşam detayı
FAM10_EW	Hanenin bölge detayı
F34	Ürünün markası

Veride 18 kayıp gözlem tespit edilmiştir. 285.692 gözlem içinde çok az sayıda eksik veri olduğundan çalışmadan çıkarılmıştır. Ancak, eksik verilerin çok sayıda olduğu durumlarda çıkarılması tavsiye edilmez. Bunun yerine eksik veriler, farklı yöntemlerle

doldurulabilir. Analizler, eksik veriler çıkarıldıktan sonra 285.674 gözlem ve 15 değişken ile yapılmıştır.

Verilerin düzenlenmesinde kullanılan kütüphaneler, eksik verilerin tespiti ve çıkarılması:

```
library(dplyr)
library(tidyverse)
(lapply (dataframe, function (dataframe) which (is.na (dataframe))))
data <- na.omit(dataframe)
attributes(data)$na.action <- NULL
```

Şekil 3.2 Gerekli kütüphaneler, eksik verilerin tespiti ve veriden çıkarılması

Değişkenler kategorik olduğundan RStudio'ya faktör olarak tanımlanmalıdır. Verideki değişkenlerin yapısına bakılarak değişkenlerin karakter olarak tanımlandığı görülmüş ve faktör dönüşümü uygulanmıştır. Dönüşüm uygulandıktan sonra verinin yapısına bakacak olursak;

```
> print(str(data))
'data.frame':   285674 obs. of  15 variables:
 $ TRN00_PERIOD   : Factor w/ 90 levels "201101","201102",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ SHP00_TYPE2    : Factor w/ 9 levels "A-101","Bim",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ SHP00_CTYPE2   : Factor w/ 8 levels "B-C","BAZ","DSC",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ F46            : Factor w/ 5 levels "0 - 500","1001 - 2000",..: 2 2 2 2 2 2 2 3 3 3 ...
 $ F47            : Factor w/ 3 levels "CHEAPY","MEDIUM",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ F55            : Factor w/ 2 levels "CONCENTRATED",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ F44            : Factor w/ 19 levels "ASSORTMENT","BLUE",..: 17 17 17 17 17 17 13 11 11 11 ...
 $ F65            : Factor w/ 205 levels "ADA ERAHLIGI",..: 179 179 179 179 179 179 105 13 13 13 ...
 $ FAM00_SES_GROUP: Factor w/ 4 levels "AB","C1","C2",..: 1 1 2 3 3 3 2 1 1 1 ...
 $ FAM00_PAX_GROUP: Factor w/ 3 levels "1-2","3-4","5+": 2 2 2 3 1 2 3 3 3 3 ...
 $ FAM00_P_AGE_GRP: Factor w/ 4 levels "-25","26-35",..: 4 2 4 4 4 4 2 2 2 3 ...
 $ FAM00_P_EDU_GRP: Factor w/ 4 levels "COLLEGE","HIGH SCHOOL",..: 4 2 2 4 3 3 2 4 2 3 ...
 $ FAM10_REGION   : Factor w/ 3 levels "KENT","KIR","YARIKENT": 1 1 1 1 1 1 1 3 1 1 ...
 $ FAM10_EW       : Factor w/ 6 levels "Aegean","Blacksea",..: 5 1 5 5 1 1 4 5 5 5 ...
 $ F34            : Factor w/ 9 levels "ABC","BINGO",..: 6 6 6 6 6 6 6 6 6 6 ...
NULL
```

Şekil 3.3 Değişkenlerin yapısı

TRN00_PERIOD, 90 düzey içerir. 90 ay için alışverişin yapıldığı yıl ve ay bilgisidir.

SHP00_TYPE2, 9 düzey içerir. Alışverişin yapıldığı marketleri (Carrefour Total, Şok, Bim, Makromarket, Other Markets, Migros, A-101, Real, Kipa) gösterir.

SHP00_CTYPE2, 8 düzey içerir. Alışverişin yapıldığı kanal (BAZ, SPM, GRC, DSC, OTHER, B-C, NC, LC) bilgisini verir.

F46, 5 düzey içerir. Ürünün hacimsel olarak hangi mililitre aralığına (0-500, 501-1000, 1001-2000, 2001-4000, 4001+) düştüğünü gösterir.

F47, 3 düzey içerir. Ürünün hangi fiyat kategorisinde (Cheapy, Medium, Premium) olduğunu gösterir.

F55, 2 düzey içerir. Ürünün hangi formatta (Concentrated, Dilute) olduğunu gösterir.

F44, 19 düzey içerir. Ürünün hangi renk olduğu bilgisini verir (Purple, Dark Blue, Pink, Assortment, Red, Blue, Yellow, ... vb.).

F65, 205 düzey içerir. Ürünün koku varyantı bilgisini verir (Çiçek Buketi, Lavanta Rüzgarı, Bahar Coskusu, Gülün Büyüsü, ... vb.).

FAM00_SES_GROUP, 4 düzey içerir. Hanenin sosyo ekonomik statüsünü (AB, C1, C2, DE) gösterir.

FAM00_PAX_GROUP, 3 düzey içerir. Hanede yaşayan toplam kişi sayısı aralığını gösterir (1-2, 3-4, 5+).

FAM00_P_AGE_GRP, 4 düzey içerir. Hanedeki bireyin yaş aralığı (-25, 26-35, 36-45, 46+) bilgisini verir.

FAM00_P_EDU_GRP, 4 düzey içerir. Hanedeki bireyin eğitim durumunu (Primary, Middle, High School, College) gösterir.

FAM10_REGION, 3 düzey içerir. Hanenin hangi yerleşimde (Kent, Kır, Yarıkent) olduğu bilgisini verir.

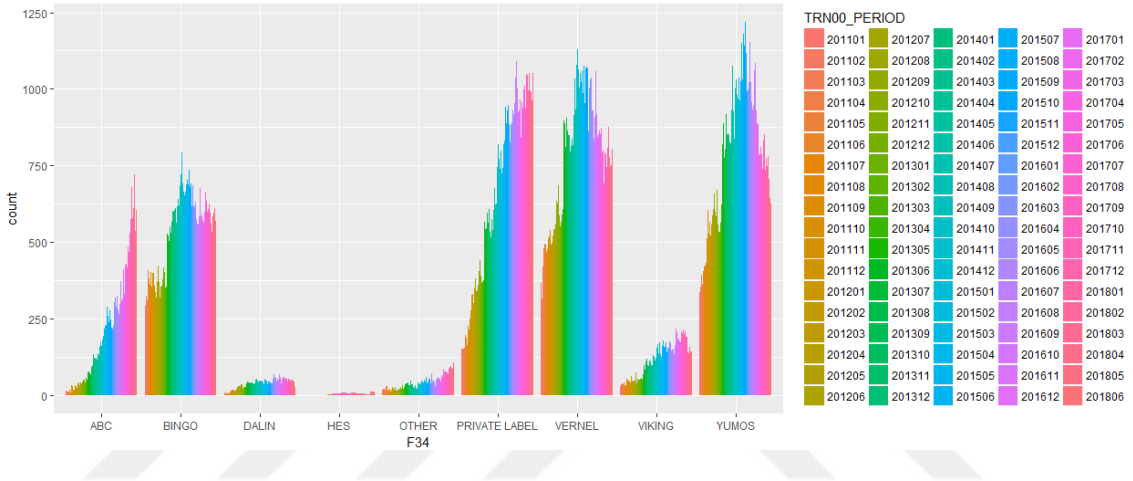
FAM10_EW, 6 düzey içerir. Hanenin hangi bölgede (Aegean, Blacksea, Mediterranean, Central, East and Southeast, Marmara) yaşadığını gösterir.

F34, 9 düzey içerir. Ürünün markasını (ABC, Bingo, Dalin, Hes, Other, Private Label, Vernel, Viking, Yumoş) gösterir.

Bağımlı değişken, F34 marka değişkenidir. Bağımsız değişkenlerin hepsi analizde kullanılmıştır.

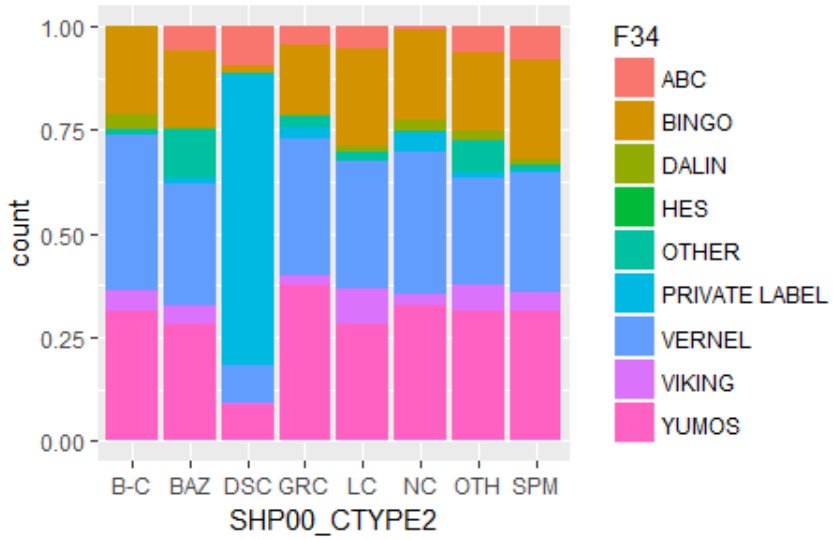
3.3 Verilerin Görselleştirilmesi

Şekil 3.4'de, 2011 ile 2018 Haziran yılları arası (90 ay) markaların tarih bazında alım grafiği gösterilmiştir. Grafikten de görüldüğü gibi çamaşır yumuşatıcısı pazar lideri 2015'lerde Yumoş markası iken, son dönemlerde Private Label alımları Yumoş markasını geride bırakmıştır. Yine son dönemlere bakıldığında, Private Label markalarından sonra tüketici tarafından en çok Vernel markası tercih edilmektedir. Genel olarak bakıldığında ise, çamaşır yumuşatıcı alımlarının son dönemlerde azalma eğiliminde olduğunu söyleyebiliriz.



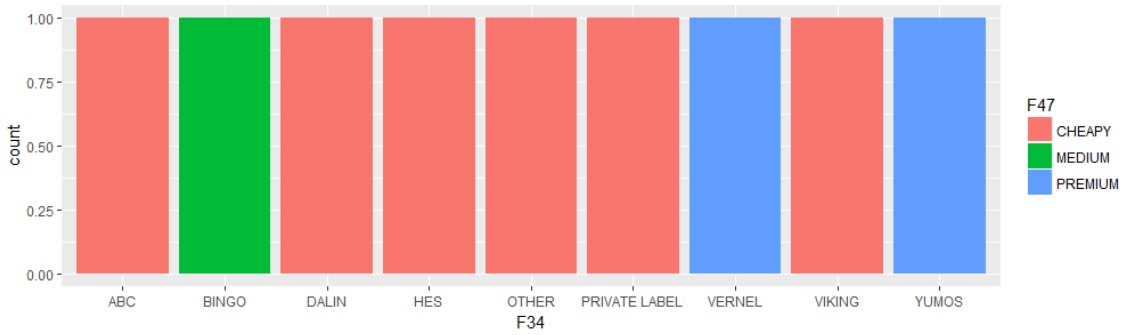
Şekil 3.4 Markaların tarih detayında alımlarının grafiği

Şekil 3.5'de çamaşır yumuşatıcı markalarının, alışveriş kanallarında genel alım dağılımına bakılmıştır. Private Label ve ABC markaları en çok DSC kanalından (A-101, Bim, Other Markets) satın alınırken, Bingo markası diğer kanallar ile karşılaştırıldığında en az DSC kanalından satın alınmaktadır. Ayrıca Yumoş ve Vernel markaları da diğer kanallara göre en az DSC kanalından satın alınmaktadır.



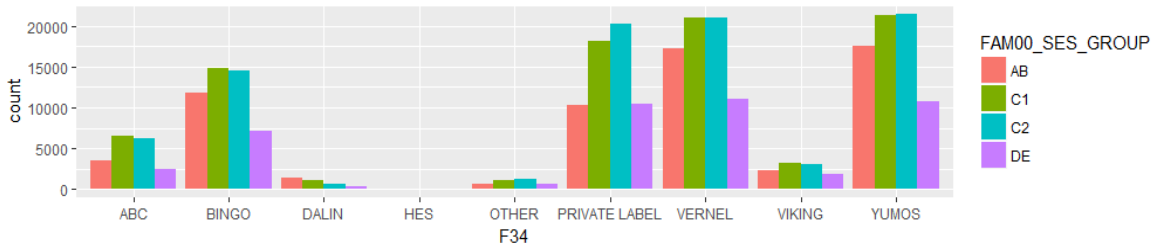
Şekil 3.5 Markaların alışveriş kanalı detayında alımlarının grafiği

Markalar, fiyat kategorileri bazında incelendiğinde Şekil 3.6'daki grafikte görüldüğü gibi ABC, Dalin, Hes, Other, Private Label ve Viking markaları Cheapy kategorisinde, Bingo markası Medium kategorisinde, Vernel ve Yumoş markaları ise Premium kategorisindedirler. Sadece Bingo markası, fiyat kategorisinde medium olarak konumlanmaktadır.



Şekil 3.6 Markaların fiyat kategorileri

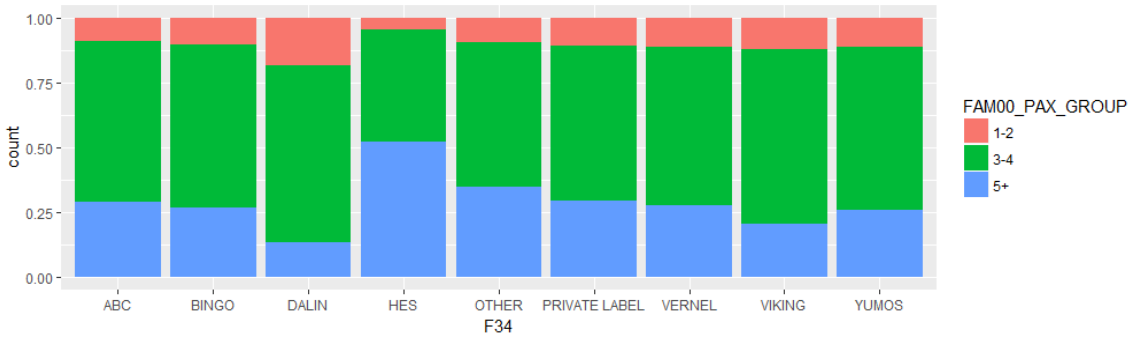
Şekil 3.7'de hanelerin sosya-ekonomik statüsü bakımından çamaşır yumuşatıcısı markalarını alım tercihleri grafiği görülmektedir.



Şekil 3.7 Sosya ekonomik statüye göre marka alımları grafiği

AB sosyo ekonomik statüdeki haneler, en çok Yumoş markasını ve neredeyse aynı oranda Vernel markasını tercih etmektedir. DE sosya ekonomik statüdeki haneler en çok Vernel markasını tercih etse de Yumoş ve Private Label tercihleri de yüksektir.

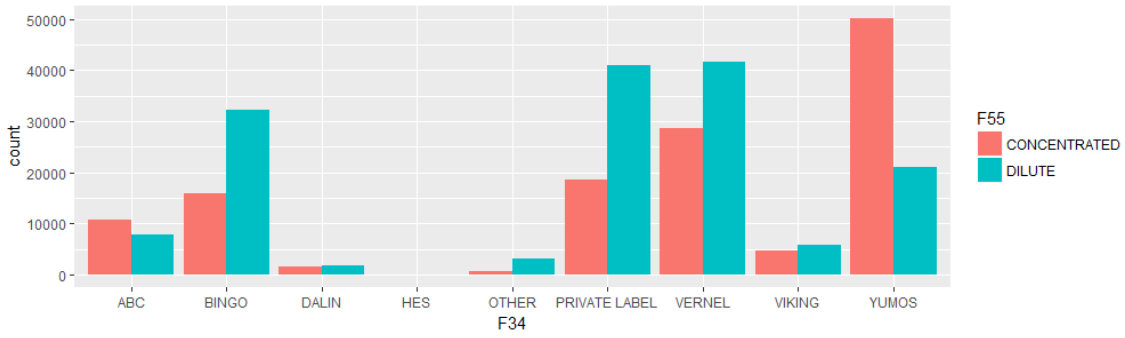
Yumoş ve Private Label markaları en çok C2 sosya ekonomik statüye sahip haneler tarafından tercih edilirken Vernel markası en çok C1 ve C2 sosyo ekonomik statüdeki haneler tarafından tercih edilmektedir.



Şekil 3.8 Hane kişi sayısına göre marka tercihlerinin grafiği

Şekil 3.8'de hane kişi sayısı kategorilerine göre çamaşır yumuşatıcısı markalarının tercih edilme grafiği gösterilmektedir. Dalin markası en çok, hanede kişi sayısı 1-2 olan hanelerde tercih edilmektedir. Hane kişi sayısı 3-4 olan hanelerde, her marka için çamaşır yumuşatıcı tüketimi diğer hane kişi sayısı kategorilerine göre daha yüksektir.

Şekil 3.9'daki grafikte çamaşır yumuşatıcı markalarının, ürünün formatına göre (concantrated, dilute) alım tercihlerine bakılmıştır.



Şekil 3.9 Marka tercihlerinin ürünün formatına göre grafiği

ABC ve Yumoş markaları konsantre formatta daha sık tercih edilirken Bingo, Private Label, Vernel markaları seyreltik formatta daha sık tercih edilmektedir.

3.4 XGBoost ile Sınıflandırma

RStudio programında, XGBoost modelini çalıştırmak için öncelikle yüklenmesi gereken, Şekil 3.10'da gösterilen kütüphaneler yüklenmiştir.

Kütüphaneler;

```
library(xgboost)
library(bst)
library(caret)      # Karısıklık matrisi icin
```

Şekil 3.10 XGBoost ve karışıklık matrisi için gerekli kütüphaneler

XGBoost, numerik verilerle çalışır. O yüzden faktör değişkenler numerik değişkenlere dönüştürülmüştür.

```
> str(data)
'data.frame':   285674 obs. of  15 variables:
 $ TRN00_PERIOD    : num  1 1 1 1 1 1 1 1 1 1 ...
 $ SHP00_TYPE2     : num  2 2 2 2 2 2 2 2 2 2 ...
 $ SHP00_CTYPE2    : num  3 3 3 3 3 3 3 3 3 3 ...
 $ F46             : num  2 2 2 2 2 2 2 3 3 3 ...
 $ F47             : num  1 1 1 1 1 1 1 1 1 1 ...
 $ F55             : num  2 2 2 2 2 2 2 2 2 2 ...
 $ F44             : num  17 17 17 17 17 17 13 11 11 11 ...
 $ F65             : num  179 179 179 179 179 179 105 13 13 13 ...
 $ FAM00_SES_GROUP: num  1 1 2 3 3 3 2 1 1 1 ...
 $ FAM00_PAX_GROUP: num  2 2 2 3 1 2 3 3 3 3 ...
 $ FAM00_P_AGE_GRP: num  4 2 4 4 4 4 2 2 2 3 ...
 $ FAM00_P_EDU_GRP: num  4 2 2 4 3 3 2 4 2 3 ...
 $ FAM10_REGION    : num  1 1 1 1 1 1 1 3 1 1 ...
 $ FAM10_EW        : num  5 1 5 5 1 1 4 5 5 5 ...
 $ F34             : num  6 6 6 6 6 6 6 6 6 6 ...
```

Şekil 3.11 XGBoost için faktörden numeriğe dönüşüm sonrası veri yapısı

3.4.1 XGBoost Modelinin Eğitilmesi

Veri kümesinin %80'i eğitim seti , %20'si test seti olmak üzere veri seti bölünmüştür. Bu bölünme ile eğitim veri setinde 228.539 gözlem, test veri setinde 57.135 gözlem bulunmaktadır.

Eğitim ve test veri setlerini oluşturma:

```
set.seed(100)
ind = sample(nrow(data),nrow(data)* 0.8)
training = data[ind,]
testing = data[-ind,]
```

Şekil 3.12 Eğitim ve test veri setlerini oluşturma

Çapraz doğrulama ve XGBoost için parametre ayarları yapılmıştır. Parametrelerin açıklamaları Şekil 3.13'de görülmektedir.

Parametre ayarlarının yapılması:

```
param = list("objective" = "multi:softmax", # coklu sınıf sınıflandırma
             "num_class"= 9 ,   # bagımlı degiskendeki sınıf sayısı
             "eval_metric" = "mlogloss",     # degerlendirme metrigi
             "nthread" = 8,          # kullanılacak thread sayısı
             "max_depth" = 16,         # agacın maksimum derinigi
             "eta" = 0.3,          # shrinkage adım boyu
             "gamma" = 0,          # minimum kayıp azaltma
             "subsample" = 0.8,        # agacın buyumesi icin veri örneklerinin bir kısmı
             "colsample_bytree" = 1,     # her agac olustururken sütunların alt örnek oranı
             "min_child_weight" = 12       # bir child için gereken  minimum örnek agırlıgın toplamı
)
```

Şekil 3.13 Parametre ayarlarının yapılması ve açıklamaları

Tahminleyiciler ve bağımlı değişken tanımlanır. XGBoost algoritması için bağımlı değişken veri setinin en son sütununda yer almalıdır. Veri setinde, bağımlı değişken F34 son sütundadır ancak olmasaydı, F34 sütunu son sütun olacak şekilde veri seti düzenlenmeliydi. XGBoost modelinde, bağımlı değişken aka etiketinin numerik olması gerekir ve numerik etiketler 0 değerinden başlamalıdır. Bu nedenle etiketin 0'dan başlatılması için Şekil 3.14'de gösterildiği gibi gerekli düzenlemeler yapılmıştır.

```
#tahminleyiciler ve bagımlı degisken, aka etiketi tanımlama.
predictors = colnames(training[-ncol(training)])

#xgboost numerik etiketlerde calısır. Bu nedenle etiketleri sayısal olarak dönüstür
label = as.numeric(training[,ncol(training)])
print(table (label))

#xgboost icin numerik etiket 0dan baslamalı. Bu yüzden etiketi sıfırdan baslat.
label = as.numeric(training[,ncol(training)])-1
print(table (label))
```

Şekil 3.14 Değişkenlerin tanımlanması ve etiketin sıfırdan başlatılması

Şekil 3.15'de gösterildiği gibi çapraz doğrulama ile en az kayıp veya hatayı veren basamak bulunur. Çalışmada, cv.nround iterasyonu 200 olarak seçilmiştir. İterasyon tur sayısı 150, 250, 300 gibi farklı değerler de seçilebilir.

```
# Step 1: En az kayıp veya hata ile turu tanımlamak icin capraz dogrulama calıstır
#         Note: xgboost verileri sayısal bir matris olarak bekler
set.seed(100)

cv.nround = 200;  # tur sayısı. 150,250,300 gibi farklı degerlere ayarlanabilir
bst.cv = xgb.cv(
  param=param,
  data = as.matrix(training[,predictors]),
  label = label,
  nfold = 3,
  nrounds=cv.nround,
  prediction=T)
```

Şekil 3.15 Çapraz doğrulama ile minimum kayıp tespiti için iterasyon

200 iterasyon içide, minimum loglossun gerçekleştiği adım bulunarak bu turdaki kayıp değerlerine bakılmıştır.

```
#minimum logaritmik kayıp hangi adımda
min.loss.idx = which.min(bst.cv$evaluation_log[, test_mlogloss_mean])
cat ("Minimum logloss occurred in round : ", min.loss.idx, "\n")

# minimum logaritmik kaybın olduğu adım
print(bst.cv$evaluation_log[min.loss.idx,])
```

```
Minimum logloss occurred in round :  120
> print(bst.cv$evaluation_log[min.loss.idx,])
   iter train_mlogloss_mean train_mlogloss_std test_mlogloss_mean test_mlogloss_std
1:  120         0.005691333       6.582975e-05           0.013359      0.0002809353
```

Şekil 3.16 Logaritmik minimum kayıbın gerçekleştiği adım ve kayıp değerleri

120. adımda minimum kayıp bulunmuştur. min.loss.idx = 120 olduğundan, XGBoost modeli iterasyon sayısı olarak 120 kullanılarak eğitilmiştir. Çünkü minimum hatanın alındığı adımda iterasyonu durdurmak gereklidir.

Modelin eğitilmesi:

```
# Step 2: bulunan min.loss.idx kullanarak xgboost modelini eğitin.
#         Not, minumum hatayı aldığımız yerde durmamız gerekiyor.
set.seed(100)

bst = xgboost(
  param=param,
  data =as.matrix(training[,predictors]),
  label = label,
  nrounds=min.loss.idx)
```

Şekil 3.17 XGBoost modelinin eğitilmesi

Model eğitildikten sonra tahminleme, test veri seti üzerinde yapılmıştır.

Test veri setinde tahminleme:

```
# Test verilerinde tahminleme
testing$prediction = predict(bst, as.matrix(testing[,predictors]))
```

Şekil 3.18 Test veri setinde tahminleme

XGBoost modeli için, bağımlı değişkeni sıfırdan başlatmak gerekliliğinden dönüşüm uygulanmıştı. Şekil 3.19'da gösterilen şekilde, etiketler yeniden ilk haline dönüştürülmüştür.

```
#Tahmini orijinal sınıfa çevir
testing$prediction = case_when(testing$prediction== "0" ~ "1", testing$prediction == "1" ~ "2",
                               testing$prediction == "2" ~ "3",testing$prediction == "3" ~ "4",
                               testing$prediction == "4" ~ "5",testing$prediction == "5" ~ "6",
                               testing$prediction == "6" ~ "7",testing$prediction == "7" ~ "8",
                               TRUE ~ "9")
```

Şekil 3.19 Etiketlerin dönüşümü

Modelin tahminleme doğruluğuna ve istatistiksel sonuçlarına, karışıklık matrisi ile bakılmıştır.

```
#tahminlerin dogrulugunu hesapla
confusionMatrix( testing$prediction,testing$F34)
```

Şekil 3.20 Karışıklık matrisinin hesaplanması

Modelin karışıklık matrisinin ve istatistiklerinin çıktısı Şekil 3.21'de gösterilmiştir.

```
Confusion Matrix and Statistics

          Reference
Prediction     1     2     3     4     5     6     7     8     9
         1  3710     0     0     3     3     4     0    33     0
         2     0  9770     0     0     0     0     0     0     0
         3     0     0   684     0    23     0     0     0     0
         4     1     0     0     6     1     0     0     0     0
         5     1     0    10     5   668     6     0     6     0
         6     3     0     0     0    11 11758     0     7     0
         7     0     0     0     0     0     0 14085     0    40
         8    42     0     0     5    15     8     0  2056     0
         9     0     0     0     0     0     0    42     0 14129

Overall Statistics

               Accuracy : 0.9953
                 95% CI : (0.9947, 0.9958)
    No Information Rate : 0.248
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9941
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3  Class: 4 Class: 5 Class: 6 Class: 7 Class: 8 Class: 9
Sensitivity           0.98749    1.000  0.98559 0.3157895  0.92649    0.9985    0.9970  0.97812    0.9972
Specificity           0.99919    1.000  0.99959 0.9999650  0.99950    0.9995    0.9991  0.99873    0.9990
Pos Pred Value        0.98854    1.000  0.96747 0.7500000  0.95977    0.9982    0.9972  0.96707    0.9970
Neg Pred Value        0.99912    1.000  0.99982 0.9997724  0.99906    0.9996    0.9990  0.99916    0.9991
Prevalence            0.06576    0.171  0.01215 0.0003325  0.01262    0.2061    0.2473  0.03679    0.2480
Detection Rate        0.06493    0.171  0.01197 0.0001050  0.01169    0.2058    0.2465  0.03598    0.2473
Detection Prevalence  0.06569    0.171  0.01237 0.0001400  0.01218    0.2062    0.2472  0.03721    0.2480
Balanced Accuracy     0.99334    1.000  0.99259 0.6578772  0.96300    0.9990    0.9980  0.98842    0.9981
```

Şekil 3.21 Karışıklık Matrisi ve İstatistikler

Karışıklık matrisi ve istatistiklere bakıldığında, modelin çoklu sınıf sınıflandırma doğruluğu (accuracy) 0.9953, kappa istatistiği ise 0.9941 bulunmuştur ve model rastlegelik ötesinde çok uyumludur.

3.4.2 XGBoost Modelindeki Değişkenlerin Önem Matrisi

XGBoost modelindeki bağımsız değişkenlerin, bağımlı değişken üzerindeki önem düzeylerini incelemek için önem matrisi (importance matrix) kullanılmıştır.

Bağımlı değişkendeki sınıfları etkileyen değişkenlerin önem matrisinin oluşturulması:

```
# bütün sınıfların bir arada önem matrisi
importance_matrix <- xgb.importance(model = bst)
```

Şekil 3.22 Önem Matrisinin Oluşturulması

Marka sınıflandırmasındaki değişkenlerin önemlerine ilişkin matris tablosu Şekil 3.23'te gösterilmiştir.

Filter

	Feature	Gain	Cover	Frequency	Importance
1	F47	0.4746411450	0.195299439	0.01122986	0.4746411450
2	F65	0.1940700781	0.260679547	0.17191283	0.1940700781
3	SHP00_TYPE2	0.1191162799	0.081768890	0.04237288	0.1191162799
4	F44	0.0772007298	0.139360683	0.06654421	0.0772007298
5	F55	0.0625252209	0.046383760	0.01782583	0.0625252209
6	SHP00_CTYPE2	0.0367814078	0.068351384	0.07739835	0.0367814078
7	F46	0.0236565152	0.042827915	0.04441847	0.0236565152
8	TRN00_PERIOD	0.0075582295	0.090639863	0.28412791	0.0075582295
9	FAM10_EW	0.0024420196	0.025645380	0.07915171	0.0024420196
10	FAM00_SES_GROUP	0.0005374423	0.011948419	0.05514737	0.0005374423
11	FAM00_P_EDU_GRP	0.0004983550	0.011688036	0.05669199	0.0004983550
12	FAM00_P_AGE_GRP	0.0004889260	0.011409865	0.05193287	0.0004889260
13	FAM00_PAX_GROUP	0.0002528932	0.007296345	0.02362862	0.0002528932
14	FAM10_REGION	0.0002307577	0.006700475	0.01761710	0.0002307577

Şekil 3.23 Önem Matrisi

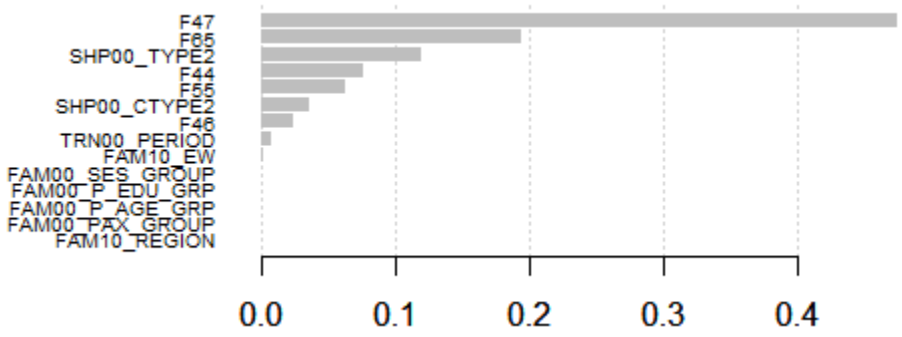
Matriste görüldüğü gibi, marka sınıfını tahminlemede genel olarak en önemli değişken markanın bulunduğu fiyat kategorisidir. Diğer önemli değişkenler, ürünün koku varyant kategorisi (F65) ve hangi marketten (SHP00_TYPE2) alındığıdır.

Önem matrisinin grafiğini çizdirmek:

```
xgb.plot.importance(importance_matrix)
```

Şekil 3.24 Önem matrisinin grafiğini çizdirmek

Önem matrisinin grafiksel gösterimi:



Şekil 3.25 Önem Matrisi Grafiği

Bağımlı değişkendeki her bir sınıfı etkileyen bağımsız değişkenlerin önem matrislerini oluşturmakta mümkündür.

Her bir sınıfın önem matrislerini ayrı ayrı incelemek:

```
# her bir sınıf için önemi ayrı ayrı incele
importance_matrixC1 <- xgb.importance(model = bst, trees = seq(from=0, by=9, length.out=min.loss.idx))
importance_matrixC2 <- xgb.importance(model = bst, trees = seq(from=1, by=9, length.out=min.loss.idx))
importance_matrixC3 <- xgb.importance(model = bst, trees = seq(from=2, by=9, length.out=min.loss.idx))
importance_matrixC4 <- xgb.importance(model = bst, trees = seq(from=3, by=9, length.out=min.loss.idx))
importance_matrixC5 <- xgb.importance(model = bst, trees = seq(from=4, by=9, length.out=min.loss.idx))
importance_matrixC6 <- xgb.importance(model = bst, trees = seq(from=5, by=9, length.out=min.loss.idx))
importance_matrixC7 <- xgb.importance(model = bst, trees = seq(from=6, by=9, length.out=min.loss.idx))
importance_matrixC8 <- xgb.importance(model = bst, trees = seq(from=7, by=9, length.out=min.loss.idx))
importance_matrixC9 <- xgb.importance(model = bst, trees = seq(from=8, by=9, length.out=min.loss.idx))
```

Şekil 3.26 Her bir sınıfın önemini incelemek

Şekil 3.26'da görüldüğü gibi marka bağımlı değişkeninde toplam 9 farklı marka sınıfı vardır. Her bir sınıf için ayrı inceleme yapılmak istendiğinde sınıf etiketleri sıfırdan başlamalıdır.

Her bir sınıf için önem matrislerinin grafiğini oluşturmak:

```
xgb.plot.importance(importance_matrixC1)
xgb.plot.importance(importance_matrixC2)
xgb.plot.importance(importance_matrixC3)
xgb.plot.importance(importance_matrixC4)
xgb.plot.importance(importance_matrixC5)
xgb.plot.importance(importance_matrixC6)
xgb.plot.importance(importance_matrixC7)
xgb.plot.importance(importance_matrixC8)
```

Şekil 3.27 Her bir sınıfın önem matris grafiği

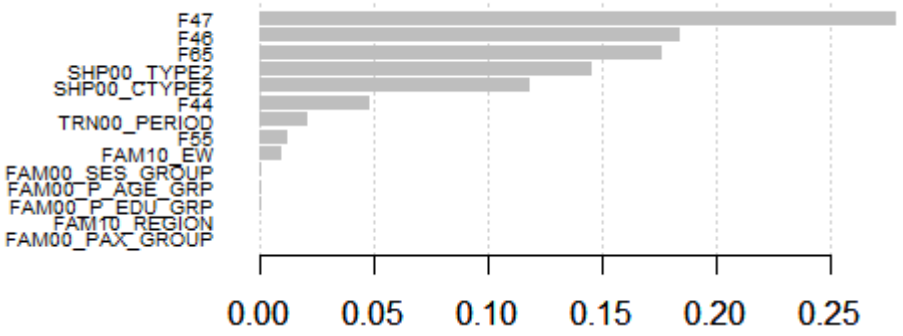
Şekil 3.27'de görülen;

- importance_matrixC1, ABC markasının,
- importance_matrixC2, Bingo markası markasının,
- importance_matrixC3, Dalin markasının,
- importance_matrixC4, Hes markasının
- importance_matrixC5, Other markalarının
- importance_matrixC6, Private Label (market) markasının
- importance_matrixC7, Vernel markasının

- importance_matrixC8, Viking markasının
- importance_matrixC9, Yumoş markasının önem matrislerinin grafikleridir.

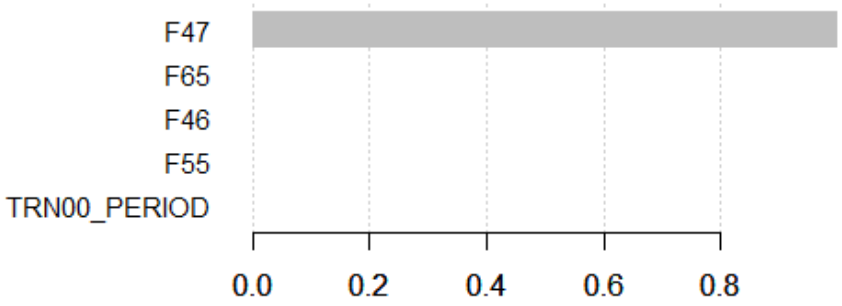
Grafikleri inceleyecek olursak:

ABC markasını etkileyen en önemli değişkenler, ürünün fiyat kategorisi, ürünün hacimsel kategorisi, ürünün koku varyantı, ürünün alındığı market ve satış kanalıdır.



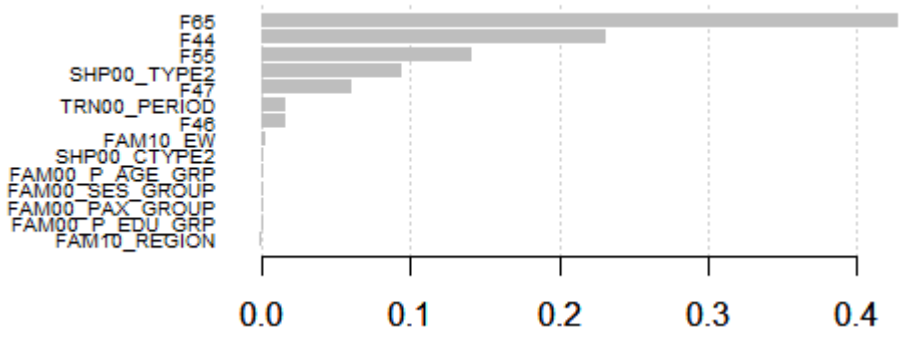
Şekil 3.28 ABC sınıfının önem matris grafiği

Bingo markasını etkileyen sadece 5 değişken vardır ve marka, ürün fiyat kategorisi değişkeninden çok yüksek oranda etkilenmektedir.



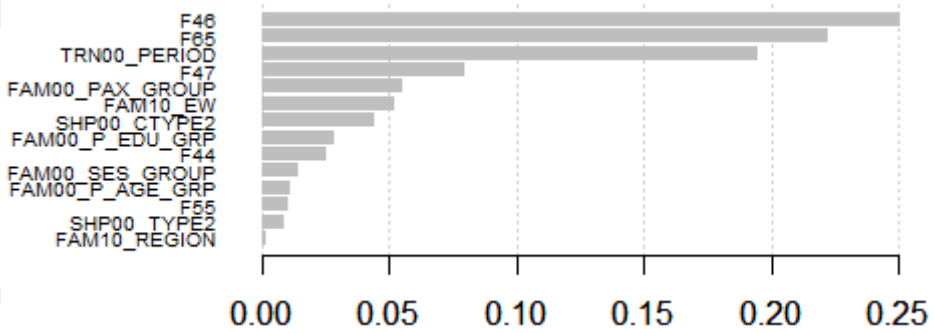
Şekil 3.29 Bingo sınıfının önem matris grafiği

Dalin markasını etkileyen en önemli faktör ürünün koku varyantıdır. Ürünün renk ve formatı (concantrated, dilute), koku varyantı değişkeninden sonraki önemli etkenlerdir.



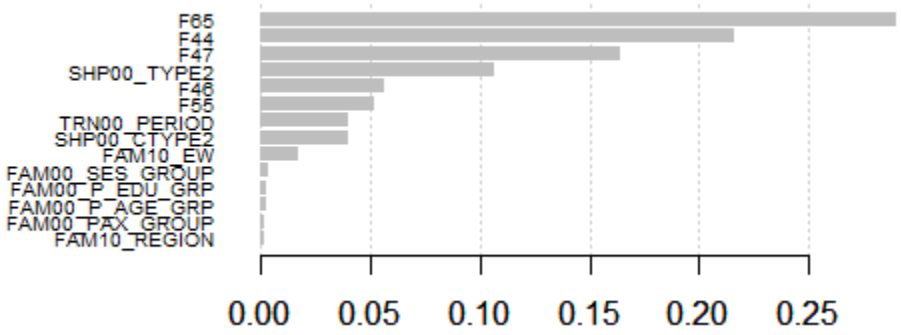
Şekil 3.30 Dalin sınıfının önem matris grafiği

Hes markasını etkileyen en önemli etkenler ürünün hacimsel kategorisi, koku varyantı ve ürünün alındığı tarihtir.



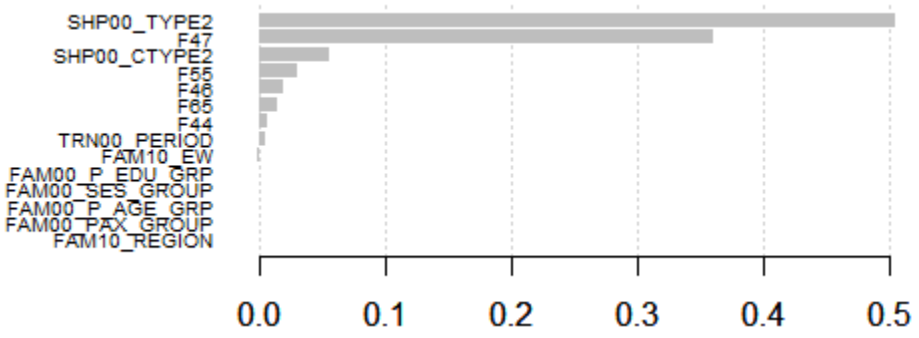
Şekil 3.31 Hes sınıfının önem matris grafiği

Other markasını yani diğer markaları etkileyen etkenlerden en önemlisi koku varyantıdır. Ürünün rengi ve fiyat kategorisi de etkilemektedir.



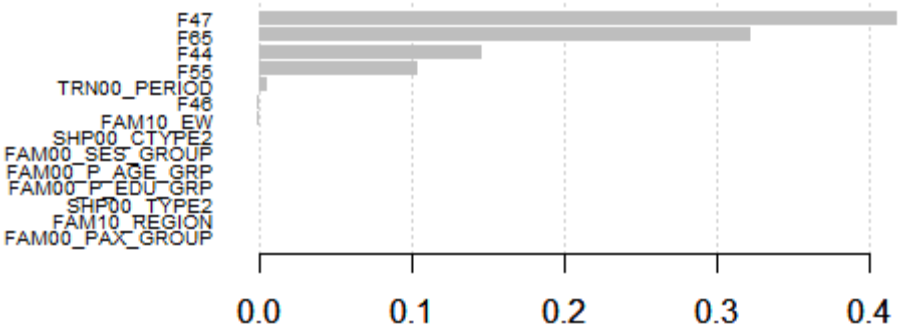
Şekil 3.32 Other sınıfının önem matris grafiği

Private Label (market markaları) sınıfını en çok ürünün satın alındığı market ve ürünün fiyat kategorisi etkilemektedir.



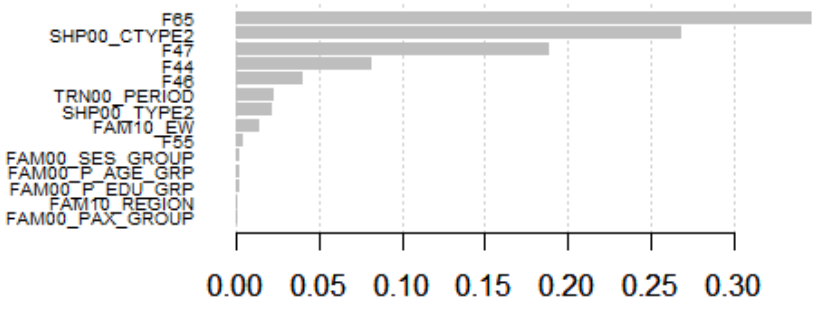
Şekil 3.33 Private Label sınıfının önem matris grafiği

Vernel markasını etkileyen en önemli değişkenler ürünün fiyat kategorisi, ürünün koku varyantı, ürünün rengi ve formatıdır.



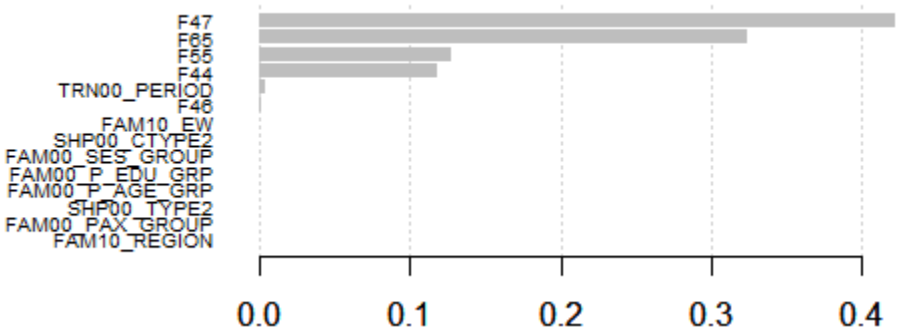
Şekil 3.34 Vernel sınıfının önem matris grafiği

Viking markasını etkileyen en önemli değişkenler koku varyantı , ürünün alındığı kanal ve ürünün fiyat kategorisidir.



Şekil 3.35 Viking sınıfının önem matris grafiği

Yumoş markası için en önemli değişkenler ürünün fiyat kategorisi, koku varyantı, formatı ve rengidir.



Şekil 3.36 Yumoş sınıfının önem matris grafiği

3.5 Naive Bayes ile Sınıflandırma

RStudio programında aynı veri seti ile Naive Bayes modeli çalıştırılmıştır. Naive Bayes modeli için gerekli kütüphaneler yüklenmiştir.

Kütüphanelerin yüklenmesi:

```
library(e1071)
library(caret)
library(mlbench)
```

Şekil 3.37 Naive Bayes için gerekli kütüphaneler

Naive Bayes modeli için XGBoost modelinde olduğu gibi, verilerin numerik olması gerekmemektedir. Veri setindeki veriler, faktör olarak düzenlenmiştir.

```
> print(str(data))
'data.frame':   285674 obs. of  15 variables:
 $ TRN00_PERIOD   : Factor w/ 90 levels "201101","201102",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ SHP00_TYPE2    : Factor w/ 9 levels "A-101","Bim",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ SHP00_CTYPE2   : Factor w/ 8 levels "B-C","BAZ","DSC",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ F46            : Factor w/ 5 levels "0 - 500","1001 - 2000",..: 2 2 2 2 2 2 2 2 3 3 3 ...
 $ F47            : Factor w/ 3 levels "CHEAPY","MEDIUM",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ F55            : Factor w/ 2 levels "CONCENTRATED",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ F44            : Factor w/ 19 levels "ASSORTMENT","BLUE",..: 17 17 17 17 17 17 13 11 11 11 ...
 $ F65            : Factor w/ 205 levels "ADA FERAHLIGI",..: 179 179 179 179 179 179 105 13 13 13 ...
 $ FAM00_SES_GROUP: Factor w/ 4 levels "AB","C1","C2",..: 1 1 2 3 3 3 2 1 1 1 ...
 $ FAM00_PAX_GROUP: Factor w/ 3 levels "1-2","3-4","5+": 2 2 2 3 1 2 3 3 3 3 ...
 $ FAM00_P_AGE_GRP: Factor w/ 4 levels "-25","26-35",..: 4 2 4 4 4 4 2 2 2 3 ...
 $ FAM00_P_EDU_GRP: Factor w/ 4 levels "COLLEGE","HIGH SCHOOL",..: 4 2 2 2 4 3 3 2 4 2 3 ...
 $ FAM10_REGION   : Factor w/ 3 levels "KENT","KIR","YARIKENT": 1 1 1 1 1 1 1 3 1 1 ...
 $ FAM10_EW       : Factor w/ 6 levels "Aegean","Blacksea",..: 5 1 5 5 1 1 4 5 5 5 ...
 $ F34            : Factor w/ 9 levels "ABC","BINGO",..: 6 6 6 6 6 6 6 6 6 6 ...
NULL
```

Şekil 3.38 Veri setindeki verilerin yapısı

3.5.1 Naive Bayes Modelinin Eğitilmesi

Veri kümesinin %80'i eğitim seti , %20'si test seti olmak üzere veri seti bölünmüştür. Bu bölünme ile eğitim veri setinde 228.539 gözlem, test veri setinde 57.135 gözlem bulunmaktadır.

Eğitim ve test veri setlerini oluşturma:

```
#egitim ve test setlerine böl
set.seed(100)
#Yeni bir train sütunu yaratın ve rasgele eşit dağılım ile 80/20 oranında 1 veya 0 atayın.
data[,"train"] <- ifelse(runif(nrow(data))<0.80,1,0)
#train / test göstergesinin sütun numarasını al
trainColNum <- grep("train",names(data))
#egitim ve test setini ayır training kolonunu modellemeden önce cıkar
traindata <- data[data$train==1,-trainColNum]
testdata <- data[data$train==0,-trainColNum]
```

Şekil 3.39 Eğitim ve test veri setlerini oluşturma

Eğitim ve test veri setlerini oluşturduktan sonra model eğitim setinde çalıştırılmıştır. Bağımlı değişken (F34) tanımlanmıştır ve geri kalan 14 bağımlı değişkenin hepsi modelde kullanılmıştır.

Eğitim setinde modelin eğitilmesi:

```
nb_model <- naiveBayes(F34~.,data = traindata)

nb_model
```

Şekil 3.40 Eğitim setinde modelin eğitilmesi

Naive Bayes modeli eğitim veri setinde eğitildikten sonra tahminleme, test veri seti üzerinde yapılmıştır. Modelin tahminleme doğruluğuna bakmak için karışıklık matrisi çizdirilmiş ve doğru tahminlerin oranına bakılmıştır.

Test veri setinde tahminleme, karışıklık matrisi ve tahminlerin doğruluk ortalaması:

```
#test verisinde tahminleme
nb_test_predict <- predict(nb_model,testdata[,-1])
#karısıklık matrisi
table(pred=nb_test_predict,true=testdata$F34)

#dogru tahminlerin oranı
mean(nb_test_predict==testdata$F34)
```

Şekil 3.41 Test veri setinde tahminleme, karışıklık matrisi ve tahminlerin doğruluğu

Naive Bayes modelinin karışıklık matrisi çıktısı ve modelin doğruluk tahminleme oranı Şekil 3.42'de gösterilmiştir.

```
> #confusion matrix
> table(pred=nb_test_predict,true=testdata$F34)
               true
pred              ABC BINGO DALIN   HES OTHER PRIVATE LABEL VERNEL VIKING YUMOS
  ABC            3521     0     0     0    31             40       0     12     0
  BINGO             0  9596     0     0     0              0       0      0     0
  DALIN             2     0   638     0    48              8       0      0     0
  HES               3     0     0    13    15              0       0     19     0
  OTHER            16     0     2     0   593            106       0     20     0
  PRIVATE LABEL    18     2     8     0    29          11573       0     10     0
  VERNEL            0     0     0     0     0              0   13947      0    60
  VIKING          120     0     4     2    35            128       0   2079     0
  YUMOS             0     0     0     0     0              5     238      0 14280
>
> #fraction of correct predictions
> mean(nb_test_predict==testdata$F34)
[1] 0.9828559
```

Şekil 3.42 Karışıklık Matrisi ve Doğruluk

Karışıklık matrisinde, gerçek değerler sütunlardaki değerler, tahmin edilen değerler satırlardaki değerlerdir. Naive Bayes modelin tahminlemede doğruluğu 0.98'dir. Model oldukça iyi sonuç verse de, bu durum veri seti seçiminden kaynaklanıyor olabilir. Algoritmanın etkinliğini kontrol etmek için farklı eğitim ve test veri kümelerinden sayısal bir ölçüm alınmalıdır.

Tahminlemenin başarısına bakmak için model belirli sayıda çalıştırılarak, her bir çalışma için doğru tahminlerin oranını içeren vektör oluşturan bir fonksiyon yazılmıştır.

```
#model sonuclarını olusturmak, calıstırmak ve kaydetmek icin fonksiyon
nb_multiple_runs <- function(train_fraction,n){
  fraction_correct <- rep(NA,n)
  for (i in 1:n){
    data[,"train"] <- ifelse(runif(nrow(data))<train_fraction,1,0)
    trainColNum <- grep("train",names(data))
    traindata <- data[data$train==1,-trainColNum]
    testdata <- data[data$train==0,-trainColNum]
    nb_model <- naiveBayes(F34~.,data = traindata)
    nb_test_predict <- predict(nb_model,testdata[,-1])
    fraction_correct[i] <- mean(nb_test_predict==testdata$F34)
  }
  return(fraction_correct)
}
```

Şekil 3.43 Naive Bayes model sonuçlarını oluşturmak, çalıştırmak ve kaydetmek

Naive Bayes modeli, bu çalışmada aynı eğitim setinde 20 kere çalıştırılmıştır. Sonuçların özet istatistiklerine bakılarak, sonuçlar arasındaki standart sapma hesaplatılmıştır.

```
#20 runs, Her bir çalışmada eğitim seti için rastgele seçilen verilerin% 80'i
fraction_correct_predictions <- nb_multiple_runs(0.8,20)
fraction_correct_predictions

#summary of results
summary(fraction_correct_predictions)

#standard deviation
sd(fraction_correct_predictions)
```

Şekil 3.44 Naive Bayes modelini iteratif çalıştırmak, özet istatistikler

Şekil 3.45'te görüldüğü gibi 20 adımın her bir doğruluk değeri hesaplanmıştır. Modelin tahminleme doğruluk oranı minimum 0,9820, maksimum 0.9839'dir. Tahminleme doğruluk oranlarının standart sapması yaklaşık olarak 0.0005 bulunmuştur.

```
> fraction_correct_predictions
 [1] 0.9828559 0.9831669 0.9832830 0.9830874 0.9827081 0.9829942 0.9822198 0.9826161 0.9829382 0.9838840 0.9819771
[12] 0.9831646 0.9835775 0.9828729 0.9825201 0.9830624 0.9836747 0.9831809 0.9821401 0.9826948
> summary(fraction_correct_predictions)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.9820  0.9827  0.9830  0.9829  0.9832  0.9839
> sd(fraction_correct_predictions)
[1] 0.0004944532
```

Şekil 3.45 Naive Bayes model doğruluk değerleri

3.6 Modellerin Tahmin Performanslarının Karşılaştırması

Çalışmada, XGBoost modeli 0.9953 doğruluk ile, Naive Bayes modeli 0.9820 doğruluk oranı ile sınıflandırma tahminlemesini gerçekleştirmiştir. İki algoritmada da oldukça başarılı sonuçlar alınmıştır. Ancak bu çalışmada, XGBoost modeli daha yüksek bir tahmin performansına sahiptir.

BÖLÜM 4

SONUÇ VE ÖNERİLER

Bu çalışmamızda, IPSOS Hane Tüketici Paneli'nden, 90 aylık (Ocak 2011 – Haziran 2018 arası) çamaşır yumuşatıcısı kategorisi hane tüketim verisi kullanılmıştır. 285.674 gözlem gibi büyük bir veri seti ile çalışıldığı için RStudio ile SQL Database Server'ının bağlantısının nasıl yapılacağına dair bilgi verilmiştir. Bağımlı değişken olan marka değişkeni 9 sınıflı olduğundan, çoklu sınıf sınıflandırma problemi ele alınarak makine öğrenmesi algoritmalarından Extreme Gradient Boosting ve Naive Bayes algoritmaları kullanılmıştır. Algortimaların RStudio programında nasıl çalıştığı anlatılmış ve iki algoritmanın tahminleme performansları karşılaştırılmıştır.

Yapılan çalışmada Extreme Gradient Boosting ve Naive Bayes algoritmaları markalara ilişkin tahminlerde yüksek doğruluk oranlarına ulaşmışlardır. Extreme Gradient Boosting modeli sınıflandırmada 0.99 doğruluğa sahipken Naive Bayes Sınıflandırıcı modeli sınıflandırmada 0,98 doğruluğa sahiptir. İki algoritma da, oldukça başarılı tahminlemeler yapsa da Extreme Gradient Boosting algoritmasının doğru sınıflandırma oranı, Naive Bayes Sınıflandırıcı'dan daha yüksektir.

Extreme Gradient Boosting sınıflandırma yöntemi için, her bir markada, marka alımını etkileyen değişkenlerin önemine bakılmıştır. Genel olarak, çamaşır yumuşatıcısı kategorisindeki markaların alımını etkileyen en önemli üç değişkenin fiyat, ürünün kokusu ve ürünün satın alındığı market olduğu saptanmıştır. Her bir markanın alımını etkileyen değişkenler incelendiğinde ise önemli olan değişkenlerin marka bazında farklılaştığı görülmüştür.

İş dünyasında, verinin ulaşılabilir olmasıyla birlikte şirketler, maksimum kazancı ya da müşteri davranışlarını anlamayı sağlayacak iş modellerine yoğunlaşmıştır. Uygulanan bu modelde, şirketler için çamaşır yumuşatıcısı sektöründe aksiyona dönüşebilecek anlamlı bilgiler elde etmek mümkündür. Bu çalışmada kullanılan algoritma, diğer hızlı tüketim ürün kategorilerine de uygulanarak, her hızlı tüketim ürün kategorisi için markalara göre tüketici alım davranışları incelenebilir.

KAYNAKLAR

[1] Yıldız, O. ve Karatana, A., (2017). "Music Genre Classification With Machine Learning Techniques", SUI Conference Proceedings, 15–18 May 2017, Antalya.

[2] Onan, A. ve Korukoğlu, S., (2016). "Makine Öğrenmesi Yöntemlerinin Görüş Madenciliğinde Kullanılması Üzerine Bir Literatür Araştırması", Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 22(2): 111-122.

[3] Ünsal, Ö., (2011). Mesleki Alan Seçimlerinin Makine Öğrenmesi Algoritması Kullanılarak Belirlenmesi, Yüksek Lisans Tezi, Gazi Üniversitesi Bilişim Enstitüsü, Ankara.

[4] Kartal, E., (2015). Sınıflandırmaya Dayalı Makine Öğrenmesi Teknikleri ve Kardiyolojik Risk Değerlendirmesine İlişkin Bir Uygulama, Doktora Tezi, İstanbul Üniversitesi Enformatik Anabilim Dalı, İstanbul.

[5] Cüvitoğlu, A., (2017). Prediction of Synergistic Drug Combinations by Using Machine Learning Methods, Master Thesis, Graduate School of Natural and Applied Sciences of Dokuz Eylül University, İzmir.

[6] Nizam, H. ve Akın, S.S., (2014). "Makine Öğrenmesi ile Duygu Analizinde Veri Seti Performansı", 14. Türkiye'de İnternet Konferansı, http://inet-tr.org.tr/inetconf19/ozet/10.html.

[7] Akçetin, E. Ve Çelik, U., (2014). "The Performance Benchmark of Decision Tree Algorithms for E-mail Detection", Journal of Internet Applications and Management, 5(2).

[8] Myazina, E., (2017). Machine Learning for Credit Scoring, Master Thesis, Department of Theoretical Computer Science and Mathematical Logic of Charles University, Prague.

[9] Burrell, J., (2016). How to Machine Thinks: Understanding Opacity in Machine Learning Algorithms, https://doi.org/10.1177/2053951715622512, 6 January 2016.

[10] Bordoloi, M. ve Biswas, S.K., (2018). "Sentiment Analysis of Product Using Machine Learning Technique: A Comparison Amoung NB, SVM and MaxEnt", International Journal of Pure and Applied Mathematics, 118(18): 71-83.

[11] Pashaei, E., (2017). Meta-Analysis of Microrna and Gene Selection Using Machine Learning, Doctoral Thesis, Graduate School of Natural and Applied Science of Yıldız Technical University, İstanbul.

[12] Brett, L., (2015). Machine Learning with R, Second Edition, Packt Publishing, Birmingham.

[13] Smith, C., (2017). Decision Trees and Random Forests: A Visual Introduction for Beginners, Blue Windmill Media.

[14] https://yazilimagiris.com/, Karar Ağacı Öğrenmesi, https://yazilimagiris.com/2017/11/karar-agaci/?print=print , 02 Ekim 2018.

[15] Khun, M. ve Johnson, K., (2013). Applied Predictive Modelling, Springer, New York.

[16] Pawlak, Z., (2002). "The Rough Set View on Bayes' Theorem", AFSS International Conference on Fuzzy Systems, 3-6 February 2002, India.

[17] Shalev-Shwartz, S. ve Shai-Ben, D., (2014). Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, Cambridge.

[18] Zumel, N. ve Mount, J., (2014). Practical Data Science with R, Manning Publications, New York.

[19] Elder, J.F. ve Giovanni, S., (2010). Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions (Synthesis Lectures on Data Mining and Knowledge Discovery), Morgan & Claypool Publishers, Chicago.

[20] Zhang, C. ve Ma, Y., (2012). Ensemble Machine Learning Methods and Applications, Springer, New York, Dordrecht, Heidelberg, London.

[21] Schapire, R.E. ve Freund, Y., (2012). Boosting Foundations and Algorithms, The MIT Press, Cambridge, Massachusetts, London, England.

[22] Geron, A., (2017). Hands on Machine Learning with Scikit Learn and TensorFlow, O Reilly Media, Sebastopol.

[23] https://machinelearningmastery.com, Decision Tree Figure, https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/, 02 Ekim 2018.

[24] http://www.statsoft.com, Support Vector Machines (SVM), http://www.statsoft.com/textbook/support-vector-machines, 04 Ekim 2018.

[25] https://rasbt.github.io/mlxtend/, Confusion Matrix, https://rasbt.github.io/mlxtend/user_guide/evaluate/confusion_matrix/, 07 Ekim 2018.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Tuğçe CEYLAN

Doğum Tarihi ve Yeri : 27.04.1991, Mersin

Yabancı Dili : İngilizce

E-posta : tugce.ceylan.es@gmail.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Lisans	İstatistik	Hacettepe Üniversitesi	2015
Lise	Fen Bilimleri	Özel Toros Fen Lisesi	2009

İŞ TECRÜBESİ

Yıl	**Firma/Kurum**	**Görevi**
2018 -	Ipsos	Data Executive
2017 - 2018	Nielsen	Modeling Analyst
2015 - 2016	CarrefourSa Genel Müdürlük	Sales Development Specialist

YAYINLARI

Bildiri

1. Ceylan T., Büyüklü, A. H., (2018). "Perakende Sektöründe Makine Öğrenmesine Dayalı Yaklaşımlar", 19. Uluslararası Ekonometri, Yöneylem Araştırması ve İstatistik Sempozyumu, 17-20 Ekim 2018, Antalya**.**