T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ YAKLAŞIMI İLE BANKA MÜŞTERİLERİNİN KREDİ RİSKİ TAHMİNİ VE KURAL ÇIKARIMI

Hilal MEYDAN

YÜKSEK LİSANS TEZİ

Matematik Mühendisliği Anabilim Dalı

Matematik Mühendisliği Programı

Danışman

Doç. Dr. Mert BAL

Haziran, 2025

T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNE ÖĞRENMESİ YAKLAŞIMI İLE BANKA MÜŞTERİLERİNİN KREDİ RİSKİ TAHMİNİ VE KURAL ÇIKARIMI

Hilal MEYDAN tarafından hazırlanan tez çalışması 19.06.2025 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Matematik Mühendisliği Anabilim Dalı, Matematik Mühendisliği Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

DOÇ. DR. Mert BAL
Yıldız Teknik Üniversitesi
Danışman

Jüri Üyeleri

DOÇ. DR. Mert BAL, Danışman
Yıldız Teknik Üniversitesi _____

DOÇ. DR. Ramazan TEKERCİOĞLU, Üye
Yıldız Teknik Üniversitesi _____

DR. ÖĞR. ÜYESİ Meryem UZUN PER, Üye
İstanbul Sağlık ve Teknoloji Üniversitesi _____

Danışmanım Doç. Dr. Mert BAL sorumluluğunda tarafımca hazırlanan “Makine Öğrenmesi Yaklaşımı ile Banka Müşterilerinin Kredi Riski Tahmini ve Kural Çıkarımı” başlıklı çalışmada veri toplama ve veri kullanımında yasal izin gerektirmeyen anonim bir veri seti kullandığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Hilal MEYDAN

İmza

Kıymetli
Aileme

TEŞEKKÜR

Tez çalışmam süresince bilgi ve deneyimiyle bana rehberlik eden, akademik gelişimime önemli katkılarda bulunan; sabrı, desteği ve yapıcı geri bildirimleriyle her aşamada yanımda olan saygıdeğer danışmanım Doç. Dr. Mert Bal'a en içten teşekkürlerimi sunarım.

Bu tezin konusunu bana önererek çalışmaya başlamamda büyük rol oynayan ve fikir aşamasından itibaren destek veren değerli arkadaşım Baş Veri Bilimci Sn. Murat Öztürkmen'e ayrıca teşekkür ederim.

Ayrıca, tezimin uygulama sürecinde sektör bilgisi ve deneyimleriyle genel anlamda katkı sunan GTech ailesine teşekkür ederim. Özellikle çalışma süremden zaman ayırmam konusunda gösterdikleri anlayış ve destek, süreci daha verimli ilerletmemi sağlamıştır. Bu kapsamda, haftalık iş programımda bana zaman tanıyarak hem motivasyonumu artıran hem de akademik çalışmama odaklanmamı kolaylaştıran GTech yönetimine ve ekip arkadaşlarıma da şükranlarımı sunarım.

Mesleki gelişimimde her zaman bana alan açan; deneyimi, öngörüsü ve yapıcı tavrıyla yalnızca çalışma hayatımda değil, kişisel gelişimimde de iz bırakan değerli yöneticim Kıdemli Büyük Veri Platformları Müdürü Sn. Tayfun Yalçınkaya'ya içten teşekkür ederim. Göstermiş olduğu destek ve ilham verici yaklaşımı, bu süreci benim için çok daha anlamlı kılmıştır.

Son olarak, her zaman yanımda olan, sabırları ve sevgileriyle bana güç veren aileme ve dostlarıma da teşekkür ederim.

Hilal MEYDAN

İÇİNDEKİLER

SİMGE LİSTESİ viii
KISALTMA LİSTESİ ix
ŞEKİL LİSTESİ xi
TABLO LİSTESİ xii
ÖZET xiii
ABSTRACT xv
1 GİRİŞ 1
1.1 Veri Seti İnceleme 5
1.2 Literatür Taraması 10
1.2.1 Kredi Riski Sınıflandırması ve Makine Öğrenmesi Yaklaşımları..... 11
1.2.2 Modelden Kural Çıkarımı Yöntemleri..... 15
1.3 Tez Çalışmasının Genel Çerçevesi ve Literatürdeki Diğer Yöntemlerle Karşılaştırması 20
2 YÖNTEMLER 23
2.1 Kredi Riski Tahmini için Makine Öğrenmesi Modelleri 23
2.1.1 Lojistik Regresyon..... 24
2.1.2 K En Yakın Komşular (K-Nearest Neighbors)..... 34
2.1.3 Naive Bayes..... 36
2.1.4 Karar Ağacı (Decision Tree)..... 40
2.1.5 Destek Vektör Makineleri..... 46
2.1.6 Rastgele Orman (Random Forest)..... 63
2.2 Kural Tabanlı Modellerin Değerlendirme Metrikleri 68
2.2.1 Children of the Tree Algoritması için Özelleştirilmiş Kural Değerlendirme Metrikleri..... 69
2.3 Veri dengesizliği ve Model Karmaşıklığı ile Başa Çıkma Yöntemleri..... 72
2.3.1 SMOTE..... 72

2.3.2 Minimum Açıklama Uzunluğu (MDL) İlkesi ..……73

3 GELİŞTİRİLEN ALGORİTMA: CHILDREN of the TREE (AĞACIN ÇOCUKLARI) – RASTGELE ORMAN TABANLI KURAL ÇIKARIMI YAKLAŞIMI 78

3.1 "Children of the Tree" Algoritmasının Temeli 78

3.2 Algoritma Adımları: Ağacın Çocukları (Children of the Tree) 79

4 UYGULAMA 82

4.1 Makine Öğrenmesi Modellerinin Uygulanması 82

4.1.1 Veri Ön İşleme ve Dengeleme……..82

4.1.2 Eğitim ve Test Verisinin Oluşturulması …….……..……...83

4.1.3 Özellik Ölçekleme ……...83

4.1.4 Uygulanan Modeller ……83

4.1.5 Performans Özeti ……84

4.1.6 ROC Eğrisi ve AUC Değerleri ……86

4.1.7 Yorum……...87

4.2 Children of the Tree Uygulaması..... 88

5 SONUÇ 93

5.1 Çalışmanın Sınırlılıkları……98

KAYNAKÇA 99

TEZDEN ÜRETİLMİŞ YAYINLAR 106

SİMGE LİSTESİ

δ	0 ile 1 arasındaki rasgele bir sayı
w	Ağırlık vektörü
w^T	Ağırlık vektörünün tersi (transpozu)
$w^\perp$	Ağırlık vektörüne dik vektör
U_{cf}	Alt güven sınırı
$\emptyset$	Boş küme
$L(\theta)$	Doğrulama fonksiyonu
F_1	Doğruluk (F1 Skoru)
$g(y)$	Doğrusal kombinasyon fonksiyonu
S_1 ve S_2	Farklı kümeler
$u(x)$	Fonksiyon
ξ_i	Gevşek değişken (slack variable)
q	Hata olasılığı
Θ	Her bir ağacın oluşturulmasında kullanılan rastgelelik kaynakları
ρ	İki farklı ağaç modelinin aynı veri noktası üzerindeki çıktıları arasındaki istatistiksel korelasyon
$\bar{\rho}$	İki farklı ağaç modelinin aynı veri noktası üzerindeki çıktıları arasındaki istatistiksel korelasyonun ortalama değeri
X	Karar değişkenleri
$f(d)$	Karar kuralları
$\mathcal{D}$	Kaynak (veri kümesi)
k	Koşul
(r)	Kök düğüm
r	Kural
$L(h)$	Kural karmaşıklığı
$\ln$	Logaritma (e tabanında)
μ	Ortalama
p	Olasılık
$\prod$	Olasılık çarpımı
$d(i,j)$	Öklid mesafesi
$mr(X, Y)$	Random Forest algoritması marjin fonksiyonu
$rmg(\Theta, X, Y)$	Random Forest algoritması ham marjin fonksiyonu
b	Sapma (bias) parametresi
$c(x)$	Sınıf etiketi
$S(x)$ veya $\sigma(\ )$	Sigmoid fonksiyonu
χ^2	Sonuç etiketi (Chi-Square)
∞	Sonsuzluk
$\sum$	Toplam (summation)
σ^2	Varyans
%	Yüzde

KISALTMA LİSTESİ

Anchors	Yerel açıklayıcı kurallar üreten yöntem
ANN	Yapay Sinir Ağı (Artificial Neural Network)
AUC	Eğri Altındaki Alan (Area Under the Curve)
BRL	Bayesyen Kural Listeleri
bkz	Bakınız
C4.5	Quinlan'ın C4.5 Karar Ağacı Algoritması
CART	Sınıflandırma ve Regresyon Ağaçları
CN2	Basit Kuralların Türetimi İçin Örtük (Kaplama) Algoritması
CORELS	Doğrulanabilir En İyi Kural Listeleri
CoT	Ağacın Çocukları Algoritması (Children of the Tree)
defragTrees	Ağaç Topluluklarını Bayesyen Model Seçimi ile Sadeleştirme
DT	Karar Ağaçları (Decision Trees)
EBM	Açıklanabilir Boosting Makineleri
Forest-ORE	Orman Tabanlı Optimal Kural Topluluğu
ForEx++	Genişletilmiş Orman Kural Topluluğu
GBDT	Gradyan Artırmalı Karar Ağaçları
GBM	Gradyan Artırmalı Makineler (Gradient Boosting Machine)
Grid Search	Izgara Arama (Grid Search - GS)
HC	Tepelik Arama (Hill Climbing)
IDS	Açıklanabilir Karar Kümeleri
inTrees	Açıklanabilir Ağaçlar
KKT	Karush-Kuhn-Tucker Koşulları
KNN	K-En Yakın Komşular (K-Nearest Neighbors)
LR	Lojistik Regresyon (Logistic Regression)
LSM	En Küçük Kareler Yöntemi (Least Squares Method)
MAP	Maksimum A Posteriori Sınıflandırması
MARS	MDL Tabanlı İlişkilendirme Kuralı Seçimi
MDL	Minimum Açıklama Uzunluğu (Minimum Description Length)
MIRCO	Kural Karmaşıklığı ve Çakışmayı Azaltan Yöntem
ML	Makine Öğrenmesi (Machine Learning)
MLM	Maksimum Olabilirlik Yöntemi (Maximum Likelihood Method)
MLP	Çok Katmanlı Algılayıcı Sinir Ağı
NB	Naive Bayes Sınıflandırıcısı
NN	Sinir Ağı (Neural Network)
OptExplain	Mantıksal Çıkarıma Dayalı Açıklanabilir Optimizasyon Yöntemi
PART	Kısmi Karar Ağaçları (Partial Decision Trees)
RF	Rastgele Orman (Random Forest)
RF+HC	Tepelik Arama ile Optimize Edilmiş Rastgele Orman
RFE	Özyineli Özellik Eleme (Recursive Feature Elimination)
RIPPER	Hata Azaltımı İçin Artımlı Budamalı Kural Öğrenici
RMSE	Kök Ortalama Kare Hatası (Root Mean Squared Error)
ROC	Alıcı Çalışma Özelliği Eğrisi (Receiver Operating Characteristic)

RuleFit	Doğrusal Modelli Kural Tabanlı Topluluk Yöntemi
SIRUS	Kararlı ve Açıklanabilir Kural Kümesi
SLIM	Çok Seyrek Doğrusal Tamsayılı Modeller
SMOTE	Sentetik Azınlık Aşırı Örnekleme Tekniği
SRL	Ölçeklenebilir Kural Öğrenici (Scalable Rule Learner)
SVM	Destek Vektör Makineleri (Support Vector Machines)
TE2Rules	Ağaç Topluluklarından Kural Türetme
UCI	California Üniversitesi Irvine Makine Öğrenmesi Havuzu
VC	Vapnik-Chervonenkis
WFS	Sarmal Özellik Seçimi (Wrapper Feature Selection)
XGBoost	Aşırı Gradyan Artırma

ŞEKİL LİSTESİ

Şekil 1.1 Alman kredi verisi – girdi veri deseni……**8**
Şekil 2.1 Lineer regresyon grafiği……..**25**
Şekil 2.2 Doğrusal olarak dağıtılmış verilere sahip bir veri kümesindeki doğrusal fonksiyon grafiği örneği…….……**26**
Şekil 2.3 Lojistik fonksiyon grafiği……...**28**
Şekil 2.4 Sigmoid fonksiyonun grafiği……..**29**
Şekil 2.5 Bir Lojistik Regresyon sınıflandırıcısı kullanılarak sınıflandırılan iki boyutlu örnek veri kümesi için karar sınırı…….……**30**
Şekil 2.6 Karar sınırının görselleştirilmesi……**30**
Şekil 2.7 Lineer regresyon.……**31**
Şekil 2.8 Lojistik regresyon……...**31**
Şekil 2.9 Gauss dağılımından tek boyutlu uzaydaki noktalar……**33**
Şekil 2.10 L(θ) grafiği……**33**
Şekil 2.11 Maksimum θ değerini gösteren ln L(θ) fonksiyonu……..**34**
Şekil 2.12 K-En Yakın Komşu algoritması için karar sınırı……**34**
Şekil 2.13 k=3 ile k-En Yakın Komşular sınıflandırma algoritması için görsel açıklama …….……**35**
Şekil 2.14 Sınıflandırılacak merkezli bir hiperküre noktası ve komşuları…….**36**
Şekil 2.15 T1, T2 ve A olayları……..**37**
Şekil 2.16 Naive Bayes algoritmasının karar sınırı……....**38**
Şekil 2.17 Sınıflandırma ağacı örneği……**40**
Şekil 2.18 İki boyutlu veri seti ve düğümler ve dallar ile karar ağacı algoritmasının karar sınırı……....**41**
Şekil 2.19 İki ve üç boyutlu veri kümeleri için sınıflandırma örneği (doğru-hiper düzlem
örnekleri)…….....**48**
Şekil 2.20 SVM ile sınıflandırma örneği……**49**
Şekil 2.21 En uygun hiper düzlem……..**50**
Şekil 2.22 Marjinin vektörel gösterimi……...**51**
Şekil 2.23 Keskin marjin örneği……**56**
Şekil 2.24 Esnek marjin örneği……...**56**
Şekil 2.25 Doğrusal olmayan destek vektör makinelerinin gösterimi…….....**59**
Şekil 2.26 Doğrusal olarak ayrılamayan bir veri kümesi……**60**
Şekil 2.27 Düzlem ile ayırma gösterimi…….…**61**
Şekil 4.1 Logistic Regresyon, KNN ve SVM modelleri için karmaşıklık matrisi sonuçları…….……**86**
Şekil 4.2 Decision Tree, Naive Bayes ve Random Forest modelleri için karmaşıklık matrisi sonuçları……..**86**
Şekil 4.3 Modellerin ROC eğrilerini gösteren grafik……..**87**

TABLO LİSTESİ

Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları……**6**
Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları (devamı)[6]…**7**
Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları (devamı)[6]…**8**
Tablo 1.2 Karar değişkenleri ve kategorilere dayalı etiket kodlama değerleri tablosu……..……**9**
Tablo 1.2 Karar değişkenleri ve kategorilere dayalı etiket kodlama değerleri tablosu (devamı)[6]…….....**10**
Tablo 1.3 Matematiksel programlama tabanlı algoritmalar……...**17**
Tablo 1.3 Matematiksel programlama tabanlı algoritmalar (devamı)[6]……**18**
Tablo 1.4 Makine öğrenmesi temelli algoritmalar……**18**
Tablo 1.4 Makine öğrenmesi temelli algoritmalar (devamı)[6]……**19**
Tablo 1.4 Makine öğrenmesi temelli algoritmalar (devamı)[6]……**20**
Tablo 3.1 "Children of the Tree" iterasyonlar[6]……**79**
Tablo 4.1 Model karşılaştırma tablosu……...**84**
Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar[6]…….....**89**
Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar (devamı)[6]……..**90**
Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar (devamı)[6]……..**91**
Tablo 5.1 RuleFit ve Children of the Tree algoritmalarının model performanslarının karşılaştırılması……...**95**
Tablo 5.2 Anchors algoritması sonuçları (en yüksek precision değerlerine göre)[6]……**96**
Tablo 5.3 Children of the Tree algoritması sonuçları (en yüksek confidence değerlerine göre)[6]……**97**

ÖZET

Makine Öğrenmesi Yaklaşımı ile Banka Müşterilerinin Kredi Riski Tahmini ve Kural Çıkarımı

Hilal MEYDAN

Matematik Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Danışman: Doç. Dr. Mert BAL

Bu çalışma, kredi riski sınıflandırma problemini ele almaktadır. Makine öğrenmesi modelleriyle bankanın geçmiş müşteri kitlesinin risk sınıflarından yola çıkarak, aday müşteriler için kredi riski sınıflandırmasına ve beraberinde hem açıklanabilir hem de denetlenebilir kurallar çıkarmaya odaklanmaktadır. Lojistik Regresyon, K-En Yakın Komşular, Naive Bayes, Karar Ağaçları, SVM ve Rastgele Orman gibi denetimli öğrenme algoritmaları, Alman kredi veri kümesi üzerinde değerlendirilmiştir. Veri kümesindeki dengesiz sınıflar problemine çözüm sunmak için SMOTE tekniği uygulanmış, algoritmaların her biri için aynı özellik mühendisliği çalışmaları gerçekleştirilmiştir. En iyi sonuç %83 doğruluk ve F1 puanı ile Random Forest algoritmasından elde edilmiştir. Bu çalışmayla, Random Forest modellerinin karar mekanizmalarını açıklamaya yönelik yeni bir kural çıkarım algoritması olan **"Children of the Tree (Ağacın Çocukları)"** geliştirilmiştir. Bu algoritma, model çıktılarını anlamlı ve yorumlanabilir IF-THEN kuralları biçiminde sunarak, aday müşterilerin kredi riskini şeffaf biçimde değerlendirmeyi amaçlamaktadır. Kuralların geçerliliği, doğruluğu, genellenebilirliği, yaygınlığı; "support", "confidence", "lift", "leverage" ve "conviction" gibi birliktelik kural madenciliği metrikleriyle sınanmıştır. Ayrıca algoritma içerisinde kullanılan **Minimum Açıklama Uzunluğu (MDL)** tekniği sayesinde, kural setlerinin basitliği sağlanmış ve modelin karmaşıklığı azaltılmıştır.

Gerçekleştirilen uygulamalar sonucunda, **Ağacın Çocukları** algoritması, geçmiş verilerden türetilmiş açıklanabilir kurallarla riskli ve risksiz müşteri ayrımını başarılı şekilde gerçekleştirmiştir. Çalışma, kredi riski analizi alanında açıklanabilir yapay zekâ ve karar denetlenebilirliği açısından yenilikçi bir katkı sunmaktadır.

Anahtar Kelimeler: Ağacın Çocukları, kural çıkarımı, Açıklanabilir Yapay Zeka, Minimum Açıklama Uzunluğu (MDL), kedi riski.

ABSTRACT

Bank Customers' Credit Risk Estimation and Rule Extraction with the Machine Learning Approach

Hilal MEYDAN
Department of Mathematical Engineering
Master of Science Thesis

Supervisor: Assoc. Prof. Dr. Mert BAL

This study addresses the problem of credit risk classification. It focuses on predicting credit risk for prospective bank customers by leveraging supervised machine learning models trained on historical risk profiles, while also extracting both explainable and auditable rules. Supervised learning algorithms such as Logistic Regression, K-Nearest Neighbors, Naive Bayes, Decision Trees, Support Vector Machines (SVM), and Random Forest were evaluated using the German Credit dataset. To overcome the class imbalance issue in the dataset, the SMOTE technique was applied, and the same feature engineering steps were carried out for each algorithm. The best result was obtained from the Random Forest model, achieving 83% accuracy and F1-score. As part of this work, a novel rule extraction algorithm called **"Children of the Tree"** was developed to explain the decision mechanisms of Random Forest models. The algorithm aims to transparently assess credit risk for new customers by expressing model outputs in the form of meaningful and interpretable IF-THEN rules. The validity, accuracy, generalizability, and prevalence of the rules were evaluated using association rule mining metrics such as **support**, **confidence**, **lift**, **leverage**, and **conviction**. Furthermore, the **Minimum Description Length (MDL)** principle was incorporated into the algorithm to ensure simplicity of rule sets and reduce model complexity. As a result of the experiments, the Children of the Tree algorithm successfully distinguished between risky and non-risky customers through

explainable rules derived from historical data. This study offers a novel contribution to the field of credit risk analysis by enhancing explainability and decision auditability in machine learning models.

Keywords: Children of the Tree, rule extraction, Explainable AI (XAI), Minimum Description Length (MDL), credit risk.

1
GİRİŞ

Kredi riski yönetimi, finansal kuruluşların sürdürülebilirliği için hayati öneme sahip bir faktördür. Bankalar, müşterilerine kredi verirken kredi riskini minimuma indirmek için çaba gösterirler. Kredi riskinin doğru bir şekilde değerlendirilmesi, bankaların finansal sağlığı açısından kritik bir konudur. Ayrıca aday müşteriler için de olası risk durumlarını belirlemek, bankanın gelecekteki riskini doğru yönetmesine olanak tanır. Makine öğrenmesi (ML) sınıflandırma yöntemleri son yıllarda, bankaların kredi riski tahmini yaparken verimli bir araç olarak kullanılmaya başlanmıştır[1]. Ancak, makine öğrenmesi modellerinin karar alma süreçlerinin genellikle "kara kutu" (black-box) yapıda olması ve insan yorumuna açık olmaması, finansal regülasyonlarla uyumlu ve açıklanabilir yapılar geliştirilmesini zorunlu hale getirmiştir[2]. *SHAP*[3] ve *LIME*[4] gibi yöntemler, makine öğrenmesi modellerinin kararlarını yorumlanabilir hale getirmek için modelin üzerine açıklayıcı bir katman ekleyerek çıktılara dair açıklamalar üretmek için son dönemde sıkça kullanılan yaklaşımlar olarak literatürdeki yerlerini almışlardır. Ancak bu tür yöntemler, karar sürecini dışarıdan yorumlama yaklaşımı benimserken; makine öğrenmesi modellerinin yapısından doğrudan kurallar çıkararak açıklanabilirlik sağlamaya yönelik çalışmalar literatürde görece daha sınırlı kalmıştır[5].

Bu çalışmada, kredi riski tahmini problemi bir uygulama alanı olarak ele alınmakta; esas bilimsel katkı ise makine öğrenmesi modellerinden anlamlı ve yorumlanabilir kurallar çıkarılmasına yönelik geliştirilen yeni bir algoritma olan *"Children of the Tree" (CoT)* ile sunulmaktadır[6]. Geliştirilen algoritma, model kararlarını insan yorumuna daha yakın, tutarlı ve veri temelli bir şekilde açıklayabilmeyi amaçlamaktadır. Bu çalışmada özetle; banka müşterilerinin kredi riski tahmininde kullanılan makine öğrenmesi yöntemleri detaylıca ve matematiksel alt yapısıyla

incelenecektir. Buna ek olarak *"Children of the Tree"* algoritmasına kıyas ölçütü (benchmark) olabilecek IF-THEN tabanlı kural çıkarma algoritmaları incelenecektir. Son olarak bankanın aday müşterileri için açıklanabilir bir yapı sunmak amacıyla modellerden kural çıkarımı yapılacak ve riskli- risksiz müşteriler için bankanın dikkatine sunulacak kurallar topluluğu oluşturulacaktır.

Bölüm 1'de kredi riski sınıflandırma ve makine öğrenmesi modellerinden kural çıkarımına dair uygulama bölümünde kullanılacak veri kümesinin tanıtımı verilmiştir. 2. Bölümde 2.1 maddesinde, çalışmada kullanılacak olan makine öğrenmesi sınıflandırma metotlarından Lojistik Regresyon (LR), K-En Yakın Komşular (KNN), Naive Bayes (NB), Karar Ağaçları (DT), Destek Vektör Makineleri (SVM) ve Rastgele Orman (Random Forest - RF) algoritmaları anlatılmıştır. Buna ek olarak, 2.2 başlığında kural tabanlı modellerin değerlendirme metriklerinden bahsedilmiştir. 2.3 başlığında veri dengesizliği ve model karmaşıklığı ile başa çıkma yöntemlerinden "*Synthetic Minority Oversampling Technique (SMOTE)"* ve *Minimum Açıklama Uzunluğu (MDL)* Tekniklerinden bahsedilmiştir, bunlar çalışmamızın değer önerisini güçlendiren tekniklerdir. Bölüm 3'de, uygulama bölümünde detaylıca sonuçları verilecek olan modelden kural çıkarımı adımında kullanılan "*Children of the Tree (Ağacın Çocukları)"* adlı yöntem tanıtılmıştır ve kıyas ölçütü olan bazı kural çıkarımı algoritmalarından bahsedilmiştir. 4. Bölümde ise Bölüm 2'de anlatılan makine öğrenmesi sınıflandırma metotları, Alman kredi veri seti üzerinde uygulanmış ve sonuçlar karşılaştırılmıştır. Daha sonra yeni gelecek müşteriler için kural çıkarımı uygulaması anlatılmıştır. Son bölümde ise karşılaştırılan sonuçların bir değerlendirmesi yer almaktadır. Ayrıca sonuçlar Alıcı İşletim Karakteristik Eğrisi *(Receiver Operating Characteristic - ROC)* eğrileri ile de grafikleştirilerek desteklenmiştir.

Bölüm 4 "Uygulama" aşamasında, banka müşterilerinin geçmiş kredi risk verilerine dayanarak, yeni kredi başvurularında bulunacak müşteriler için ikili risk sınıflandırması (riskli / değil) gerçekleştirilmiştir. Bu sınıflandırma, Bölüm 2'de anlatılan makine öğrenmesi modelleri aracılığıyla yapılmış; ardından bu modellerin karar süreçlerinden açıklanabilir kurallar türetilmiştir. Böylece sadece model çıktısı değil, bu çıktının altında yatan nedenler de görünür hale getirilmiştir. Yeni geliştirilen "Children of the Tree" algoritması sayesinde, aday müşteriler için

geçmiş veri desenlerinden elde edilen kurallarla şeffaf ve yorumlanabilir risk değerlendirme süreçleri oluşturulmuş; bu da kararların denetlenebilirliğini ve regülasyonlara uyumun izlenmesini mümkün kılmıştır[6]. Gerçekleştirilen deneysel analizler sonucunda, sınıflandırma algoritmalarının performansları; doğruluk (accuracy), sınıf bazlı F1 skorları ve Eğri Altında Kalan Alan *(Area Under Curve - AUC)* metrikleri kullanılarak değerlendirilmiştir. Bu kapsamda elde edilen değerler Tablo 4.1'de özetlenmiş, ROC eğrileri ise Şekil 4.2'de görselleştirilmiştir. Bölüm 5'te ise tezin genel değerlendirmesi yapılmış, algoritmanın sınırlarından da bahsedilmiştir.

Özetle çalışma sonuçları şöyledir; **RF** algoritması %83 doğruluk, 0.83 ortalama F1 skoru (her iki sınıfa ait F1 skorlarının eşit ağırlıkla alınarak hesaplandığı *macro average* kullanılmıştır) ve 0.92 AUC değeri ile en başarılı sonuçları elde etmiştir. Bu model, hem genel doğruluk hem de pozitif-negatif sınıflar arasında dengeli sınıflandırma başarımı sunarak çalışmanın en güçlü modeli olmuştur.

KNN algoritması %81 doğruluk, 0.815 ortalama F1 skoru ve 0.89 AUC değeriyle güçlü bir alternatif sunarken; **LR** %81 doğruluk, 0.81 ortalama F1 skoru ve 0.88 AUC ile benzer düzeyde performans göstermiştir. Bu iki model, özellikle istatistiksel olarak tutarlı sınıflandırmalar üretmiş ve açıklanabilirlik açısından avantajlı yapılarıyla öne çıkmıştır.

SVM modeli %80 doğruluk ve 0.80 ortalama F1 skoru ile genel başarıda geride kalmış olsa da, 0.90 AUC değeriyle ROC eğrisinde etkileyici bir eğri çizmiş; bu da modelin ayırt edicilik gücünü göstermiştir.

NB, %78 doğruluk ve 0.78 ortalama F1 skoru ile orta seviyede başarı sağlamış, sınıflar arasında dengeli ancak sınırlı performans sergilemiştir. **DT** modeli ise %75 doğruluk ve 0.76 ortalama F1 skoru ile en düşük başarıyı gösteren model olmuştur.

Tüm bu sonuçların ötesinde, geliştirilen **Children of the Tree** algoritması, Rastgele Orman modelinin yapısından doğrudan kural çıkarımı yaparak hem yorumlanabilirlik hem de tutarlılık açısından değerli katkılar sunmuştur[6]. Üretilen kurallar, yüksek güven, destek ve anlamlılık metrikleriyle hem açıklanabilir hem de veriyle tutarlı biçimde yapılandırılmıştır. Bu yönüyle algoritma, literatürdeki "IF–THEN" tabanlı yöntemlerle karşılaştırılabilir nitelikte güçlü bir performans ortaya koymuş; özellikle kararların şeffaflığı,

denetlenebilirliği ve regülasyonlara uyum açısından önemli avantajlar sağlamıştır[6].

1.1 Veri Seti İnceleme

Bu çalışmada, kredi riski analizi için literatürde yaygın olarak tanınan bir kaynak olan Statlog (Alman kredi veri seti)[1] veri seti kullanılmıştır[7]. Bu veri seti, banka müşterilerinin özelliklerini etkili bir şekilde yakalar ve müşteri risk seviyeleriyle ilgili kuralları çıkarmak için uygundur. Bu veri seti, banka pazarlama ve risk yönetimi departmanlarındaki potansiyel müşterilerin risk seviyesini belirlemek için kullanılır. Başvuranların demografik, finansal ve davranışsal özelliklerini bir araya getirdiği için değerlidir. Kredi riskini değerlendirirken, bir müşterinin yaşı, istihdam durumu, geçmiş kredi ödemeleri, kredi süresi ve talep edilen kredi tutarı gibi çok boyutlu veriler, olası riskler hakkında ayrıntılı içgörüler sağlar[7]. Bu tür dengesiz veri setlerinde, etkili karar almayı sağlamak için yüksek tahmin doğruluğuna sahip modeller geliştirmek ve istatistiksel olarak anlamlı kurallar üretmek kritik öneme sahiptir.

Veri setinin doğasını anlamak, çalışmanın hizmet ettiği alanı kavramak için önemlidir. Veri seti, kredi başvurularını değerlendirmek için kullanılan 20 özellik ve 1 hedef değişken içerir[7]. Bu özellikler, her başvuru sahibi hakkında demografik, finansal ve davranışsal bilgileri içerir. Yaş, başvuru sahibinin yaşını belirtirken, Kişisel Durum ve Cinsiyet, başvuru sahibinin medeni durumunu ve cinsiyetini belirtir. Konut, başvuru sahibinin ikametgahının mülkiyet durumunu temsil eder ve "kendi evi", "kira" veya "ücretsiz konaklama" olarak sınıflandırılır. Bakmakla Yükümlü Olunan Kişi Sayısı, başvuru sahibine maddi olarak kaç kişinin güvendiğini gösterir[7].

Veri setindeki finansal özellikler arasında, kredi tutarı talep edilen kredi tutarını; Süre (ay) kredinin geri ödeme süresini ay cinsinden belirtir. Mevduat tutarı (tasarruf) başvuranın tasarruf tutarını belirtir ve düşükten yükseğe doğru farklı aralıklara kategorik olarak ayrılır. Mevcut Mülk niteliği, başvuranın sahip olduğu mülklerin türünü ve değerini sınıflandırır. Diğer Taksit Planları, başvuranın ek kredi sözleşmelerinin olup olmadığını belirtir. Diğer Borçlular niteliği, başvuranın

[1] Veri seti UCI Makine Öğrenmesi Deposundan alınmıştır; belirli bir özellik kümesiyle tanımlanan kişileri iyi veya kötü kredi riskleri olarak sınıflandırır.

kredi başvurusunda kefil veya başka borçluların olup olmadığını belirtir. İstihdam durumu ve mesleği (iş), başvuranın istihdam durumunu ve mesleğini kategorik olarak ifade eden bir özelliktir. İstihdam Süresi, mevcut iş yerinde çalışılan zamandır ve bu sürenin uzunluğu finansal istikrarı ölçmek için bir ölçüt olabilir[7].

Davranışsal özellikler arasında, başvuranın sahip olduğu mevcut kredi sayısı, kredi başvurusunu değerlendirmede önemli bir faktördür. Öte yandan, kredinin amacı, kredinin hangi amaçla alındığını gösterir ve "araba", "mobilya", "eğitim" gibi kategorilere ayrılır. Kredi Geçmişi, başvuranın geçmiş kredi ödemelerinin bir özetini sunar ve ödemelerdeki düzenlilik, kredi riskini anlamada temel bir rol oynar[7].

Telefon sahipliği, başvuranın bir telefona sahip olup olmadığını gösterir, bu da özellikle iletişim için önemlidir. Yabancı Çalışan, başvuranın yabancı bir çalışan olup olmadığını gösterir. Bu iki özelliğin, bir kişinin kredi başvurusunu değerlendirmede modele dolaylı etkiler sunabileceği öngörülmektedir. Kredi Riski alanı ise, iki farklı sınıftan oluşan hedef değişkenimizi ifade eder. Sınıf 1, iyi (risksiz) durumdaki başvuruları ifade ederken, sınıf 2, kötü (riskli) durumdaki başvuruları ifade eder[7].

Tüm bu özellikler kredi riski analizinde farklı yönlerde bir araya gelerek başvuranın kredi geri ödeme kapasitesi hakkında bilgi sağlarken, çalışmamızda, kişinin sınıflandırma modelinde kredi verme açısından riskli olup olmadığı tahmin edilmiş ve bu risk sınıfına göre farklı kural çıkarımları ile bankadaki müşterilerin durumuna göre banka potansiyel müşterileri için anlamlı kurallar sağlanmaya çalışılmıştır.

Tablo 1.1'de veri setindeki özellikler aşağıdaki gibi tablo halinde özetlenebilir.

Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları[6]

Özellik Adı	**Veri Tipi**	**Açıklama**
Checking Account	Kategorik	Mevcut "vadesiz hesap" bilgisi
Duration	Sayısal	Kredi süresi (vade)

Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları (devamı)[6]

Credit History	Kategorik	Kredi geçmişi
Purpose	Kategorik	Kredi alınma amacı
Credit Amount	Sayısal	Kredi miktarı
Saving Account	Kategorik	Mevduat hesabı bilgisi
Employment Duration	Kategorik	Mevcut iş yerindeki çalışma süresi (zaman aralığı)
Installment rate	Sayısal	Aylık taksit oranı (harcanabilir gelirin yüzdesi olarak)
Personal Status and Sex	Kategorik	Medeni durum ve cinsiyet
Other Debtors	Kategorik	Diğer borçlular / kefiller

Tablo 1.1 Orijinal Alman kredi veri seti özellikleri ve açıklamaları (devamı)[6]

Present Residence	Sayısal	Şu anda ikamet edilen süre (yıl)
Property	Kategorik	Mal varlığı bilgisi
Age	Sayısal	Yaş
Other Installment Plans	Kategorik	Diğer ödeme planları
Housing	Kategorik	Konut durumu
Number of Credits	Sayısal	Aynı bankada bulunan diğer kredi sayısı
Job	Kategorik	Meslek
Dependents	Sayısal	Bakmakla yükümlü olunan kişi sayısı
Telephone	İkili (Binary)	Telefon sahipliği
Foreign Worker	İkili (Binary)	Yabancı çalışan
Credit Risk	İkili (Binary)	Kredi riski sınıfı (iyi / kötü) – Hedef değişken

Veri setine ilişkin ayrıntılı açıklamalar için bkz.[7].

Etiket kodlama (label encoding) yapılmadan önce veri setinin modelimize girişi aşağıdaki *Şekil 1.1'deki* gibidir.

checking	duration	credit_hi	purpose	credit_an	savings_	employm	installme	personal	other_de	present_	property	age	other_ins	housing	number_	job	depende	telephon	foreign_v
A11	6	A34	A43	1169	A65	A75	4	A93	A101	4	A121	67	A143	A152	2	A173	1	A192	A201
A12	48	A32	A43	5951	A61	A73	2	A92	A101	2	A121	22	A143	A152	1	A173	1	A191	A201
A14	12	A34	A46	2096	A61	A74	2	A93	A101	3	A121	49	A143	A152	1	A172	2	A191	A201
A11	42	A32	A42	7882	A61	A74	2	A93	A103	4	A122	45	A143	A153	1	A173	2	A191	A201
A11	24	A33	A40	4870	A61	A73	3	A93	A101	4	A124	53	A143	A153	2	A173	2	A191	A201

Şekil 1.1 Alman kredi verisi – girdi veri deseni

Her kategorik değişkenin içerdiği değerin sayısallaştırma sonucu aşağıdaki Tablo 1.2'de gösterildiği gibidir.

Tablo 1.2 Karar değişkenleri ve kategorilere dayalı etiket kodlama değerleri tablosu[6]

Özellikler	Kategori	Açıklamalar/Değerler	Etiketler
checking_account	A11	X < 0 DM (Alman Mark'ı)	0
checking_account	A12	0 <= X < 200 DM	1
checking_account	A13	X >= 200 DM / en az 1 yıllık maaş ödemeleri	2
checking_account	A14	checking account bulunmama durumu	3
credit_history	A30	hiç kredi alınmamış / tüm krediler zamanında ödenmiş	0
credit_history	A31	bu bankadaki tüm krediler zamanında ödenmiş	1
credit_history	A32	mevcut krediler şimdiye kadar zamanında ödenmiş	2
credit_history	A33	geçmişte ödemede gecikme olmuş	3
credit_history	A34	sorunlu hesap / başka bankalarda mevcut krediler	4
purpose	A40	araba (yeni)	0
purpose	A41	araba (ikinci el)	1
purpose	A410	diğer	2
purpose	A42	mobilya / ekipman	3
purpose	A43	radyo / televizyon	4
purpose	A44	ev aletleri	5
purpose	A45	tamirat	6
purpose	A46	eğitim	7
purpose	A48	yeniden eğitim	8
purpose	A49	iş / ticari amaçlı	9
savings_account	A61	X < 100 DM	0
savings_account	A62	100 <= X < 500 DM	1
savings_account	A63	500 <= X < 1000 DM	2
savings_account	A64	X >= 1000 DM	3
savings_account	A65	bilinmiyor / mevduat hesabı yok	4
employment_duration	A71	işsiz	0
employment_duration	A72	X < 1 yıl	1
employment_duration	A73	1 <= X < 4 yıl	2
employment_duration	A74	4 <= X < 7 yıl	3
employment_duration	A75	X >= 7 yıl	4
personal_status_sex	A91	erkek : boşanmış / ayrılmış	0
personal_status_sex	A92	kadın : boşanmış / ayrılmış / evli	1

Tablo 1.2 Karar değişkenleri ve kategorilere dayalı etiket kodlama değerleri tablosu (devamı)[6]

personal_status_sex	A93	erkek : bekar	2
personal_status_sex	A94	erkek : evli / dul	3
other_debtors	A101	yok	0
other_debtors	A102	birlikte başvuran	1
other_debtors	A103	kefil	2
property	A121	gayrimenkul	0
property	A122	yapı kooperatifi / hayat sigortası	1
property	A123	araba veya diğer (özellik 6'da olmayan)	2
property	A124	bilinmiyor / mal varlığı yok	3
other_installment_plans	A141	banka	0
other_installment_plans	A142	mağazalar	1
other_installment_plans	A143	yok	2
housing	A151	kiracı	0
housing	A152	kendi evi	1
housing	A153	ücretsiz	2
job	A171	işsiz / vasıfsız - yerleşik olmayan	0
job	A172	vasıfsız - yerleşik	1
job	A173	vasıflı çalışan / memur	2
job	A174	yönetici / serbest meslek / yüksek vasıflı / üst düzey çalışan	3
telephone	A191	yok	0
telephone	A192	var, müşteri adına kayıtlı	1
foreign_worker	A201	evet	0
foreign_worker	A202	hayır	1
credit_risk	1	İyi (Kredi Riski Yok / Kabul Edilebilir)	1
credit_risk	2	Kötü (Kredi Riski Yüksek / Reddedilebilir)	2

Bu veri setinde, her satır bankadan kredi alan bir başvuruyu temsil etmektedir, 1000 girişten oluşur. Her başvuru, bir dizi niteliklere göre iyi veya kötü kredi riski olarak sınıflandırılmış risk alanını da içermektedir. Bu veri seti kredi riski tahmini ve modelden kural çıkarımı adımında, kurallar çıkarmak amacıyla kullanılmıştır.

1.2 Literatür Taraması

Son yıllarda gelişen ML algoritmalarının, farklı sektörlerdeki uygulama başarıları sayesinde karar destek sistemlerinde daha sık tercih edilmesi, bankacılık ve finans gibi veri yoğun alanlarda da bu yöntemlerin kullanımını hızla artırmıştır. Bankacılık sektörü özelinde; kredi riski sınıflandırması gibi kritik problemlerde, kurumların

kredi tahsis süreçlerinde daha güvenilir, otomatik ve hızlı kararlar alması önem arz etmektedir. Müşterilerin geçmiş finansal davranışlarına dayalı olarak riskli/risksiz olarak sınıflandırılması da bu ihtiyaç için önemli bir çözüm imkânı sunmaktadır. Makine öğrenmesi modelleri, bu sınıflandırma problemlerinde sundukları yüksek doğruluk, esneklik ve veriyle öğrenme kabiliyeti sayesinde geleneksel istatistiksel yöntemlere kıyasla daha başarılı sonuçlar verebilmektedir[1].

Bu bağlamda, kredi riski tahmini alanında gerçekleştirilen akademik çalışmalarda; LR, DT, RF, SVM, NB ve KNN gibi çeşitli algoritmaların başarıyla uygulandığı ve birçok veri setiyle başarılı sınıflandırma performanslarının elde edildiği görülmektedir. Literatürdeki çalışmalar, hem modelleme tekniklerinin etkinliğini göstermekte hem de yeni yöntem önerileri için zemin oluşturmaktadır. Özellikle lojistik regresyon, karar ağaçları ve benzeri temel sınıflandırıcılar, kredi riski tahmini problemlerinde uzun süre yaygın olarak kullanılmış; bu yöntemler, daha sonra geliştirilen ensemble ve boosting tabanlı algoritmaların da temelini oluşturmuştur[1]. Bununla birlikte, bu modellerin açıklanabilirlik açısından sunduğu avantajlar —örneğin lojistik regresyondaki katsayıların doğrudan yorumu ya da karar ağaçlarındaki if-then kuralları— finansal karar destek sistemlerinde hâlen önemli bir referans noktası olma niteliklerini sürdürmektedir[9]. Öte yandan, bu modellerin karar alma süreçlerini açıklama gereksinimi, özellikle regülasyonlara tabi alanlarda daha da önem kazanmış; bu durum, makine öğrenmesi modellerinden açıklanabilir ve yorumlanabilir kurallar çıkarılması yönündeki araştırmaları tetiklemiştir[10].

Bu bölümde, kredi riski tahmini ve makine öğrenmesi temelli kural çıkarımı konularında literatürde çalışılan başlıca çalışmalar sunulacak; kullanılan yöntemler, elde edilen sonuçlar ve belirlenen sınırlılıklar ele alınarak, bu çalışmanın hangi boşluğu doldurmayı hedeflediği teorik bir çerçevede ortaya konacaktır.

1.2.1 Kredi Riski Sınıflandırması ve Makine Öğrenmesi Yaklaşımları

Addo ve arkadaşlarının (2018) çalışmasında, kredi riski sınıflandırması amacıyla çeşitli makine öğrenmesi ve derin öğrenme algoritmaları karşılaştırılmış; model performansları AUC ve *Root Mean Squared Error (RMSE)* gibi metriklerle değerlendirilmiştir. Ayrıca, dengesiz veri yapısıyla başa çıkmak amacıyla SMOTE tekniği uygulanmıştır. Bu yönüyle çalışma, hem veri ön işleme hem de model

değerlendirme kriterleri açısından kapsamlı bir yaklaşım sunmakta ve bu çalışmada kullanılan yöntemlerin seçimi için referans niteliği taşımaktadır[11]. Benzer şekilde bu tez kapsamında da veri dengesizliğini yönetmek amacıyla SMOTE yöntemi tercih edilmiştir; model başarımını ölçmek için ROC eğrisi ve AUC skoru temel alınmıştır.

Kredi riski sınıflandırmasında makine öğrenmesi algoritmalarının etkinliğini araştıran Mahbobi ve arkadaşları (2023), kredi kartı ödemelerine ilişkin "*University of California, Irvine Machine Learning Repository"* (UCI) veri seti üzerinde SVM ve KNN gibi algoritmaları farklı örnekleme teknikleriyle birlikte denemiştir. SVM algoritması, özellikle All-KNN[2] yeniden örnekleme yöntemi ile birlikte kullanıldığında %98.6 doğruluk ve 0.864 F1 skoru ile en başarılı sonuçları elde etmiştir. Bu çalışma, SVM ve KNN algoritmalarının kredi riski tahmini alanında yüksek potansiyele sahip olduğunu göstermiştir[12]. Buradan hareketle; bu tez çalışmasında da iki algoritmaya yer verilerek, benzer veri dengesizliği problemleri altında algoritmaların performanslarının değerlendirilmesi amaçlanmıştır.

Oğuz Koç ve arkadaşları (2023) tarafından gerçekleştirilen çalışmada[13], kredi skorlama problemlerinde çeşitli makine öğrenmesi algoritmalarının (SVM, Gaussian Naive Bayes, Karar Ağaçları, Rastgele Orman, Extreme Gradient Boosting (XGBoost), KNN, Çok Katmanlı Algılayıcı Sinir Ağları (MLP) ve Lojistik Regresyon) performanslarını, veri dönüşüm teknikleri ve özellik seçimi stratejileriyle birlikte karşılaştırmalı olarak değerlendirilmiştir. Alman (1000 gözlem) ve Avustralya (690 gözlem) kredi veri setleri kullanılarak yapılan analizlerde, özellikle Grid Search (GS) ve Wrapper Feature Selection (WFS) tekniklerinin modeller üzerindeki etkileri incelenmiştir. Sonuçlara göre, Alman veri setinde en yüksek doğruluk %82 ile MLP+GS kombinasyonunda elde edilirken, Avustralya veri setinde %89.05 doğrulukla XGBoost+GS yöntemi öne çıkmıştır. Lojistik regresyon, her iki veri setinde de %87.62'ye kadar doğruluk gösterebilmiş; KNN ise WFS ile %89.05'e ulaşarak yüksek başarı sağlamıştır. Çalışmada ayrıca, veri dönüşüm tekniklerinin (örneğin Min-Max ve Standardizasyon) özellikle KNN ve MLP gibi algoritmaların başarısını artırdığı belirtilmiş; karar ağaçları gibi

[2] **All-KNN (Tüm k-En Yakın Komşular Yöntemi)**, veri madenciliği ve makine öğrenmesi alanında, özellikle **anormal veri tespiti (anomaly detection)** için kullanılan bir yöntemdir. KNN tabanlı aykırı değer tespiti algoritmalarının optimize edilmiş bir versiyonudur.

gömülü özellik seçim mekanizmasına sahip modellerde bile WFS uygulamasının performansı iyileştirdiği vurgulanmıştır. Bu çalışma, sadece algoritma başarımını değil, aynı zamanda hiperparametre optimizasyonu, özellik seçimi ve ölçekleme işlemlerinin entegrasyonunun kredi riski sınıflandırmasındaki belirleyici rolünü göstermesi açısından literatüre kapsamlı bir katkı sunmaktadır[13]. Koç ve arkadaşlarının (2023) çalışması, kredi riski alanında çok sayıda makine öğrenmesi algoritmasının performansını detaylı biçimde incelemiş; bu yönüyle çalışmamıza kıyas ölçütü (benchmark) niteliğinde katkı sağlamıştır. Söz konusu makale, kredi skorlama problemlerine yönelik farklı algoritmaların uygulanabilirliği hakkında içgörü sunarak, tezimizde değerlendirdiğimiz model seçeneklerinin belirlenmesinde yol gösterici olmuştur.

Chen ve ekibinin 2009 tarihli çalışması, lojistik regresyon, karar ağacı, naive bayes gibi modellerden bazılarını birleştirerek kredi riskini tahmin etmek için hibrit bir yaklaşım sunmaktadır. Bu çalışmada lojistik regresyon, karar ağaçları, Naïve Bayes ve yapay sinir ağları gibi farklı sınıflandırma modelleri ile birlikte, bu yöntemlerin hibrit versiyonları da değerlendirilmiştir. Elde edilen sonuçlara göre, hibrit yaklaşımlar, tekil modellere kıyasla daha yüksek doğruluk oranlarına ulaşmış ve kredi riskinin tahmininde daha başarılı performanslar sergilemiştir. Özellikle lojistik regresyon temelli hibrit modellerin, veri seti üzerinde anlamlı iyileştirmeler sağladığı görülmektedir[14]. Bu çalışmada, lojistik regresyon modeli validasyon setinde %79.42, test setinde %72.17 doğruluk oranına sahipken; karar ağacı modeli validasyon setinde %72.97, test setin %65.60 doğruluk oranına sahiptir. NB algoritması ise validasyon setinde tek başına %77.16 doğruluk değerine sahipken, test setinde %71.20 doğruluk değeri almıştır. Hibrit model LR + DT %72.50 doğruluk oranına sahipken; hibrit model LR + NB %78.68 doğruluk oranına; hibrit model LR + LR ise %82.94 doğruluk oranına sahiptir. LR + NN (neural network) ise %83.44 doğruluk oranıyla en iyi hibrit model olmuştur[14]. Tez çalışmasında yapay sinir ağları kapsam dışı bırakılmıştır. Ancak incelenen kaynaklardaki sonuçlara bakılınca; makine öğrenmesi tekniklerinin kredi risk skorlamasında güçlü alternatifler sunduğunu ve özellikle açıklanabilir modellerle oluşturulan hibrit yapıların (yapay sinir ağları ve makine öğrenmesinin birlikte kullanımı, vb.) doğruluk açısından önemli avantajlar sağlayabileceğini göstermesi açısından değer taşımaktadır.

[15]'teki çalışma, kredi kartı kullanıcılarının temerrüde düşme olasılığını tahmin etmek amacıyla altı farklı veri madenciliği (makine öğrenmesi) yöntemini karşılaştırmaktadır. Kullanılan teknikler arasında KNN, Lojistik Regresyon, Diskriminant Analizi, Naive Bayes sınıflayıcı, yapay sinir ağı (ANN) ve Karar Ağaçları yer almaktadır. Tayvan'daki bir bankadan elde edilen 25.000 gerçek müşteri verisiyle yapılan analizde hem sınıflandırma doğruluğu (hata oranı ve area ratio[3]) hem de tahmin edilen temerrüt olasılığının güvenilirliği değerlendirilmiştir. Sınıflandırma sonuçlarına göre en düşük hata oranı KNN (0.16) ve karar ağaçlarında (0.17) gözlenmiştir; ancak en yüksek sınıflandırma başarısı (area ratio) yapay sinir ağına aittir (0.54), onu naive Bayes (0.53) ve karar ağaçları (0.536) izlemektedir. Analizde yalnızca sınıflandırma doğruluğu değil, aynı zamanda tahmin edilen temerrüt olasılıklarının gerçek olasılığa ne kadar yakın olduğu da değerlendirilmiştir. Bu amaçla geliştirilen "Sorting Smoothing Method" ile yapılan regresyon analizine göre ANN en iyi performansı göstermiştir ($R^2 = 0.965$). İkinci sırada naive Bayes sınıflayıcı ($R^2 = 0.899$), ardından KNN ($R^2 = 0.876$), lojistik regresyon ($R^2 = 0.794$), diskriminant analiz ($R^2 = 0.659$) ve son olarak karar ağaçları ($R^2 = 0.278$) gelmektedir. Çalışma, kredi riski değerlendirmesinde olasılık tahmininin önemini vurgulamaktadır[15].

Makine öğrenmesinde ensemble (topluluk) yöntemlerinin gelişimi, model çeşitliliğini artırarak daha kararlı ve yüksek performanslı sınıflandırıcılar elde etme amacını taşımaktadır. Bu kapsamda, 1990'lı yıllarda geliştirilen Bagging[16], Boosting[17,18] ve Stacking[19] gibi yöntemler, zayıf öğrenicilerin bir araya getirilerek daha güçlü modeller inşa edilmesini mümkün kılmıştır. Bu yöntemler arasında özellikle Bagging, bootstrap örneklemesi ile eğitilen çoklu modellerin ortalamasını alarak varyansı azaltmayı hedeflemektedir. RF algoritması ise, Breiman'ın Bagging yaklaşımına karar ağaçlarını temel öğrenici olarak eklemesi ve her ağaçta rastgele özellik altkümesi kullanılmasıyla geliştirilmiş, ensemble öğrenmenin en başarılı ve açıklanabilir uygulamalarından biri hâline gelmiştir[8]. Bu bağlamda RF, hem tarihsel gelişim süreci açısından hem de kredi riski gibi

[3] Kümülatif kazanımlar grafiğindeki (lift chart) model eğrisi ile baz çizgi arasındaki alanın, en iyi model ile baz çizgi arasındaki alana oranını ifade eden bir ölçüttür. Bu metrik 1'e yakınken modelin iyi sınıflandırdığı söylenebilirken, 0'a yakın durumlarda modelin başarısız olduğuna kanaat getirilebilir.

açıklanabilirliğin önemli olduğu alanlarda etkin bir yöntem olarak literatürdeki yerini almıştır.

Kurniawan'ın (2024) çalışmasında, kredi riski sınıflandırması için Rastgele Orman algoritması uygulanarak, kredi başvurularının "iyi" ya da "kötü" olarak ayrılmasında yüksek doğrulukla çalışan bir model geliştirilmiştir. Endonezya merkezli bir fintech şirketinden elde edilen 1.848.797 gözlem ve 503 değişkenden oluşan veri seti kullanılmıştır. Çalışma kapsamında modelin doğruluk oranı %97, F1 skoru ise 0.8571 olarak rapor edilmiştir. Özellikle "repeat_user", "monthly_salary" ve "year_of_service" gibi değişkenlerin kredi değerlendirme sürecinde belirleyici faktörler olduğu görülmüştür. Sınıflandırma temelli bu uygulama, kredi riski alanında RF algoritmasının etkinliğini vurgulayan güncel bir örnek teşkil etmektedir[20].

Bowen Han'ın (2025) çalışmasında, kredi riski tahmini amacıyla altı farklı makine öğrenmesi algoritması—RF, SVM, KNN, LR, Gradient Boosting Machine (GBM) ve Gradient Boosting Decision Tree (GBDT)—kapsamlı değerlendirme metrikleriyle karşılaştırılmıştır. Kaggle üzerinden elde edilen kredi başvuru verisi, ciddi şekilde dengesiz sınıf dağılımı içermektedir. Özellik seçimi Rastgele Orman özelliği önemi ve RFE (Recursive Feature Elimination) ile yapılmıştır. En iyi performans, RF + RFE kombinasyonunda elde edilmiş; bu model %71 doğruluk oranı ile öne çıkmıştır. KNN %69 doğrulukla ikinci sırada yer alırken, lojistik regresyon yalnızca %29 doğruluk göstererek en zayıf model olmuştur. Çalışma sınıflandırma temellidir ve yalnızca kredi riski alanında uygulanmıştır. Bulgular, dengesiz verilerde RF'nin yüksek doğruluk ve sınıf ayırt etme gücü sayesinde kredi riskinin değerlendirilmesinde etkili bir araç olduğunu göstermektedir[21].

1.2.2 Modelden Kural Çıkarımı Yöntemleri

Makine öğrenmesi modellerinden kural çıkarımı konusu, finans ve sağlık gibi pek çok alanda ele alınmış ve uygulanmıştır[22,23]. Bugüne kadar yapılan çalışmalarda; modellerin yorumlanmasından başlayarak, açıklanabilir/yorumlanabilir model tasarımlarına, modelin eğilimlerinin kurallar aracılığıyla ortaya konmasına ve hatta bu kuralların iş birimleri tarafından kullanılır hale getirilmesine kadar geniş bir yelpazede araştırmalar yürütülmüştür[2].

Özellikle ağaç tabanlı makine öğrenmesi modellerinden kural çıkarımı, modelin karar verme süreçlerini daha anlaşılır kılmak açısından önemli bir yaklaşımdır.

Rastgele Orman gibi topluluk modellerinde, birden fazla karar ağacından kurallar çıkararak analiz yapmak; hem modelin açıklanabilirliğini artırmakta hem de güvenilir kararlar elde edilmesini sağlamaktadır[8]. Bu bağlamda, literatürde "if-then" (eğer-o halde) biçiminde çalışan çeşitli algoritmalar öne çıkmaktadır:

- **inTrees Framework**[24]: Rastgele Orman'dan çıkarılan kuralları sıklık, hata oranı ve uzunluk gibi metriklere göre sıralar. Karmaşıklığı azaltarak kuralları daha sade ve anlaşılır hale getirir. Ancak, kural karmaşıklığını azaltma ve model hatasını dikkate alma gibi süreçlerde MDL (Minimum Description Length) tekniğini kullanmaz. Ayrıca kural çakışmaları ve açıklanabilirlik konusunda bazı sınırlamaları bulunmaktadır.
- **ExtractingRuleRF**[25]: RF modellerinden çıkarılan kuralları açgözlü (greedy) bir algoritma ile sıralar. Bu yöntem öngörüsel doğruluğu önceliklendirirken, açıklanabilirlik kapsamı sınırlı kalmaktadır.
- **SIRUS**[26]: RF'ten türetilmiş ve kural sıklığına dayalı bir yöntemdir. Daha kısa ve daha kararlı kurallar üretmeye odaklanır.
- **RF+HC (Hill Climbing)**[27]: Hill-Climbing algoritmasını kullanarak RF'ten çıkarılan kural sayısını azaltır. Küçük ve anlamlı kural kümeleri oluşturmak için optimizasyon uygular.
- **defragTrees**[28]: RF kurallarını Bayesyen model seçimi algoritması ile sadeleştirir ve öngörüsel performansı optimize eder. Karmaşıklığı azaltırken açıklanabilirliği korumayı amaçlar.
- **ForEx++**[29]: Karar ormanlarından yüksek kaliteli kural toplulukları üretir. Ortalama metriklere dayalı bir çerçeve sunar ve öngörüsel performansa odaklanır.
- **MIRCO**[30]: Matematiksel programlama kullanarak kural çeşitliliğini ve karmaşıklığını minimize eder. Bu yöntem özellikle azınlık sınıfı daha iyi temsil eden kurallar üretmeyi hedefler.

- **OptExplain**[31]: Mantıksal çıkarım, örnekleme ve optimizasyon teknikleri kullanarak kurallar üretir. Özellikle geniş veri kapsama alanı sunabilme kapasitesine sahiptir.
- **RuleFit**[32]: RF düğümlerinden çıkarılan kuralları seyrek doğrusal regresyon ile birleştirir. Ancak, yüksek korelasyonlu kurallarla çalışıldığında kararsızlık (instability) sorunları yaşayabilir.
- **Node Harvest**[33]: RF'ten çıkarılan kuralları ağırlıklı tahmin modeli olarak kullanır. Sadece pozitif ağırlıklarla çalışır ve daha açıklanabilir modeller sunmayı amaçlar.
- **Forest-ORE (Optimal Rule Ensemble)**[34]: Rastgele Orman modellerinden açıklanabilir bir kural topluluğu üretmeyi amaçlayan bir yöntemdir. Karmaşıklık, kapsama ve öngörü doğruluğu arasındaki dengeyi sağlamak için Karma-Tamsayılı Programlama (MIP) tekniğini kullanır.

Aşağıda Tablo 1.3'te literatürde kıyaslama oluşturan matematiksel programlama tabanlı algoritmaların temel özellikleri, avantajları, dezavantajları ve uygun oldukları uygulama alanının karşılaştırması yer almaktadır.

Tablo 1.3 Matematiksel programlama tabanlı algoritmalar[6]

Algoritma Adı	Temel Özellikleri	Avantajları	Dezavantajları	Uygulama Alanları
Forest-ORE[34]	Karma tamsayılı optimizasyon kullanarak Random Forest modellerinden açıklanabilir kural setleri çıkarır.	Öngörüsel başarı ile açıklanabilirlik arasında denge kurar.	Büyük veri setlerinde hesaplama maliyeti yüksektir ve zaman alıcıdır.	Global model açıklanabilirliği, azınlık sınıfları.
MIRCO[30]	Matematiksel programlama ile toplam kural karmaşıklığını ve çeşitliliğini minimize eder.	Azınlık sınıfı daha iyi temsil eden kurallar üretir.	Hesaplama maliyeti yüksektir.	Risk analizi, veri madenciliği.
OptExplain[31]	Mantıksal çıkarım, örnekleme ve optimizasyon teknikleri ile kurallar üretir.	Geniş veri kapsamını açıklama yeteneğine sahiptir.	Mantıksal işlemler ve optimizasyon süreçleri karmaşık olabilir.	Optimizasyon ve mantıksal çıkarım uygulamaları.

Tablo 1.3 Matematiksel programlama tabanlı algoritmalar (devamı)[6]

defragTrees [28]	Bayesyen model seçimiyle Random Forest kurallarını sadeleştirir ve optimize eder.	Doğruluğu koruyarak kuralları sadeleştirir.	Bayesyen süreçler büyük veri setlerinde yavaş çalışabilir.	Bayesyen modelleme, finans ve sağlık sektörleri.

Öte yandan Tablo 1.4'te literatürde kıyaslama noktası oluşturan makine öğrenmesi modelleri üzerinde çalışabilen kural çıkarma algoritmalarının temel özellikleri, avantajları, dezavantajları ve uygulama alanlarının karşılaştırması yer almaktadır.

Tablo 1.4 Makine öğrenmesi temelli algoritmalar[6]

Algoritma Adı	Temel Özellikleri	Avantajları	Dezavantajları	Uygulama Alanları
RuleFit[32]	Karmaşık modellerden türetilen if-then kurallarını doğrusal modellerle birleştirir.	L1 düzenlileştirme ile önemli kuralları seçer, açıklanabilirlik ve doğruluk dengesi kurar.	Dengesiz veri kümelerinde SMOTE veya benzeri optimizasyona ihtiyaç duyar.	Sınıflandırma ve regresyon, finansal analiz
RIPPER[35]	Art arda budama ile hataları azaltan if-then tabanlı bir algoritmadır.	Basit, hızlı ve açıklanabilir kurallar üretir.	Büyük veri setlerinde performansı düşebilir.	Tıp, finans, küçük veri setleri
CN2[36]	Örtük (covering) algoritmasıyla if-then kurallar üretir.	Dengesiz veri setlerinde güçlüdür, anlamlı kurallar üretir.	Karmaşık veri setlerinde doğruluğu düşebilir.	Biyoloji, tıp, sınıflandırma
PART[37]	Karar ağaçlarından kısmi kurallar çıkaran if-then tabanlı bir algoritmadır.	Açıklanabilir ve basit kurallar üretir; çok sınıflı problemlerde etkilidir.	Karmaşık ilişkilerde performansı sınırlıdır.	Eğitim, sınıflandırma
Bayesian Rule Lists (BRL)[38]	Bayesyen olasılık modeli ile kuralları sıralayan ve optimize eden bir if-then algoritmasıdır.	Açıklanabilirlik ve doğruluk dengesini sağlar; küçük veri setlerinde güçlüdür.	Büyük veri setlerinde hesaplama maliyeti yüksektir.	Sağlık, hukuk, finans

Tablo 1.4 Makine öğrenmesi temelli algoritmalar (devamı)[6]

TE2Rules[46]	Tree Ensemble modellerinden çıkarılan kuralları doğruluk ve açıklanabilirlik dengesi ile optimize eder.	Karar yollarının tümünü kapsayan yüksek açıklanabilirlik ve doğruluk dengesi sağlar.	Büyük veri setlerinde yüksek hesaplama maliyeti ve zaman gerektirir.	Makine öğrenmesi, açıklanabilir modeller
SIRUS[26]	Random Forest'tan türetilen kararlı ve açıklanabilir kural kümeleri oluşturur.	Kısa ve belirleyici kurallar üretir, kararlıdır.	Nadir ama önemli kuralları atlayabilir.	Regresyon ve sınıflandırma, stabil modeller
Anchors[39]	Her tahmin edilen durumu açıklamak için yerel kurallar üreten if-then tabanlı bir yöntemdir.	Karmaşık desenleri anlamada güçlüdür; sezgisel ve açıklayıcı kurallar üretir.	Büyük veri setlerinde ölçeklenebilirlik sınırlı olabilir.	Model açıklanabilirliği, mühendislik
C4.5 & CART[40,41]	Karar ağaçlarıyla if-then kurallar üretir.	Uygulaması kolay, hızlı ve açıklanabilir.	Overfitting'e yatkındır.	Sınıflandırma, regresyon, eğitim
Slipper[42]	Boosting kullanarak kuralların doğruluğunu artırır.	Performansı artırır, dengesiz veri setlerinde etkili olabilir.	Boosting nedeniyle hesaplama maliyeti artabilir.	İkili sınıflandırma
SRL (Scalable Rule Learner)[43]	Büyük veri setlerinde ölçeklenebilir kurallar üreten if-then tabanlı bir yöntemdir.	Büyük veri setlerinde hızlı, açıklanabilir ve optimize edilebilir.	Küçük veri setlerinde aşırı basitleştirilmiş sonuçlar üretebilir.	Büyük veri, gerçek zamanlı uygulamalar
IDS (Interpretable Decision Sets)[44]	Kurallar arası çakışmayı en aza indirerek düşük karmaşıklıkta kurallar üretir.	Açıklanabilirliğe öncelik verir.	Büyük veri setlerinde hesaplama maliyeti yüksektir.	Sağlık, hukuk, güvenilirlik gerektiren alanlar
EBM (Explainable Boosting Machines)[45]	Her özelliği bağımsız modelleyerek Gradient Boosting tabanlı açıklanabilir kurallar üretir.	GBM'ye yakın doğruluk, anlamlı açıklamalar sağlar.	Karmaşık ilişkilerde performansı sınırlı olabilir.	Sağlık, finans, kritik karar süreçleri
RF+HC (Hill-Climbing)[27]	Hill-Climbing algoritması ile RF içerisindeki kuralları optimize eder.	Küçük ve anlamlı kural setleri üretir.	Optimizasyon süreci zaman alabilir.	Optimize küçük veri setleri

Tablo 1.4 Makine öğrenmesi temelli algoritmalar (devamı)[6]

Node Harvest [33]	RF düğümlerinden elde edilen kuralları ağırlıklı tahmin modeliyle birleştirir.	Pozitif ağırlıklarla basit kural seti oluşturur.	Tahminsel performansı sınırlı olabilir.	Makine öğrenmesi, kural tabanlı analiz
ForEx++[29]	Tahmin performansını artırmak için yüksek kaliteli kural toplulukları üretir.	Tahminsel performansı optimize eder.	Optimize edilmiş kural boyutu, açıklanabilirliği sınırlayabilir.	Risk analizi, veri madenciliği
Children of the Tree[6]	RF modellerinden, dengesiz veri kümelerine uygun, SMOTE ile dengelenmiş kurallar çıkarır; MDL maliyetiyle sıralama yapar.	Her iki sınıfta yüksek doğruluk ve F1 skoru sağlar; en düşük maliyetli kuralları bulur.	Çok büyük veri setlerinde işlem maliyeti artabilir; bu durumda özellik temelli filtreleme önerilir.	Sağlık, finans, veri yoğun alanlar, dengesiz sınıflar, ML sınıflandırma

1.3 Tez Çalışmasının Genel Çerçevesi ve Literatürdeki Diğer Yöntemlerle Karşılaştırması

Bu çalışmada, kredi riski tahmini problemi kapsamında Lojistik Regresyon, Naive Bayes, KNN, SVM, Karar Ağaçları ve Rastgele Orman gibi literatürde sıklıkla kullanılan çeşitli sınıflandırma algoritmaları test edilmiştir. Ancak açıklanabilirlik ihtiyacı ve if-then biçiminde kural üretmeye elverişliliği dikkate alındığında, en yüksek doğruluk ve denge metriklerini de sağlayan RF modeli öne çıkmıştır. Bu nedenle çalışmada nihai model olarak Rastgele Orman seçilmiş ve önerilen "Ağacın Çocukları"[6] algoritması, bu modelin karar ağaçları üzerinden kural çıkarımı yapacak şekilde yapılandırılmıştır.

Önerilen algoritma "Children of the Tree" (Ağacın Çocukları), kuralları değerlendirme ve optimize etme sürecinde MDL[47] prensibini kullanmaktadır[6]. Bu yaklaşım, literatürdeki çoğu algoritmadan farklı bir değerlendirme mekanizması sunar. Örneğin, inTrees[24] algoritması da kural uzunluğu açısından değerlendirme yaparak kural setlerini sadeleştirmeye çalışsa da, MDL tabanlı bir optimizasyon süreci yürütmemektedir. RIPPER ve CN2 gibi algoritmalar ise genellikle destek

(support) ve güven (confidence) gibi klasik kural çıkarımı metriklerine odaklanırken[35,36]; tez çalışmasındaki yaklaşım, kuralların hem karmaşıklığını hem de doğruluğunu aynı anda değerlendirmektedir[6].

Bu farklı değerlendirme mantığı, algoritmamızın CN2[36] ve RIPPER[35] gibi yöntemlerle doğrudan karşılaştırılmasını zorlaştırmaktadır; çünkü söz konusu yöntemler, değerlendirme süreçlerine kural karmaşıklığını dâhil etmemektedir. Öte yandan, MDL tabanlı optimizasyon sayesinde elde edilen kurallar hem daha basit ve yoğun hem de daha açıklanabilir olmaktadır[47]. Bu özgün özellik, algoritmamızı literatürde farklı bir kategoriye yerleştirmekte; karmaşıklık ve doğruluk arasında denge kurarak açıklanabilirliğe öncelik veren bir yaklaşım ortaya koymaktadır.

Bu bağlamda, "Certifiably Optimal Rule Lists" (CORELS) algoritması da sade ve yorumlanabilir kural dizileri üretmeyi amaçlayan dikkat çekici bir çalışmadır[48]. CORELS[48], bir kural listesi oluştururken hem hata oranını hem de kural uzunluğunu optimize eder. Ancak burada da MDL doğrudan uygulanmaz; sadeleştirici bir ceza terimi (örneğin: kural başına sabit ceza) ile kısa ve doğru modeller üretilmeye çalışılır. Dahası, CORELS[48] sıfırdan bir kural listesi öğrenirken çalışır; halihazırda eğitilmiş bir modelin açıklanması gibi bir hedefi yoktur. Bu yönüyle CORELS, açıklanabilir modeller üretirken "**Children of the Tree[6]"** ise açıklanamayan modelleri anlamlı hale getirir — yani yöntemsel olarak biri doğal olarak yorumlanabilir (intrinsically interpretable), diğeri sonradan açıklanabilir (post-hoc explainable) yaklaşımı temsil eder.

Ayrıca "Children of the Tree" algoritmasının, dengesiz veri kümeleri üzerinde başarıyla çalışabilmesi önemli bir avantajdır[6]. CN2[36] ve RIPPER[35] gibi klasik kural çıkarım algoritmalarının, bu tür veri kümelerinde genellikle baskın sınıfa aşırı uyum sağlama (overfitting) eğiliminde olduğu bilinmektedir. Buna karşılık, önerdiğimiz algoritma azınlık sınıfı hedefleyerek bu sınıf için anlamlı kurallar çıkarabilmekte, böylece daha dengeli ve işlevsel bir sınıflandırma sağlamaktadır[6].

Algoritmanın kural çıkarım sürecinde MDL (Minimum Description Length) ilkesini doğrudan bir ölçüt olarak kullanması, onu geleneksel algoritmalardan temelde ayıran önemli bir özelliktir. Çünkü burada sadece doğruluğa değil;

kuralların ne kadar sade, açıklanabilir ve bilgi açısından anlamlı olduğuna da bakılmaktadır. Bu yaklaşımda, kuralları sadece klasik birliktelik kural madenciliği metrikleri ile değerlendirmek yeterli kabul edilmemiştir. Sadelik ve basitlik yönlerinden de değerlendirme sağlanmıştır. Bu yönüyle "Children of the Tree" algoritması klasik yöntemlerle birebir kıyaslanmayı da zorlaştırmaktadır[6]. Buna rağmen söz konusu çalışmalar bazı noktalarda CoT için birer kıyas ölçütü (benchmark) oluşturabilir. Her ne kadar bu çalışmalar doğrudan MDL ilkesini kural çıkarımı amacıyla kullanmamış olsa da, bazıları model uyumu ile karmaşıklık arasında denge kurmayı hedefleyen yaklaşımlarıyla MDL'in temel felsefesine dolaylı olarak benzemektedir[6].

Buna ek olarak, CoT yalnızca dengesiz veri kümelerinde azınlık sınıfına odaklanabilmesiyle değil, aynı zamanda bu tür veri kümesine uygulanan ön işleme tekniklerine karşı da yüksek uyum göstermesiyle öne çıkmaktadır[6]. Çalışmada kullanılan veri seti üzerinde, SMOTE gibi sentetik örnekleme teknikleriyle dengesizlik giderildikten sonra literatürdeki RuleFit[32] ve Anchors[39] kural çıkarımı algoritmaları da aynı önişleme adımlarından geçirilerek uygulanmıştır. Ancak, bu algoritmalar önişleme sayesinde elde edilen yapısal avantajlara rağmen, "Children of the Tree" algoritmasının ulaştığı başarı/genellik düzeyine aynı anda erişememiştir. Özellikle azınlık sınıfa ait doğru sınıflandırma oranları ve elde edilen kuralların doğruluk/karmaşıklık dengesi bakımından önerilen yöntem, istikrarlı biçimde daha başarılı sonuçlar üretmiştir[6]. Bu bulgular, tez çalışmasında önerilen algoritmanın (Children of the Tree - CoT) adilce ön işlenmiş veri ortamlarında dahi üstünlük sağladığını ve sadece teorik değil, pratik uygulamalar açısından da literatürdeki yöntemlere kıyasla anlamlı bir katkı sunduğunu ortaya koymaktadır.

2
YÖNTEMLER

Bölüm 2, çalışmada kullanılacak veri madenciliği yöntemlerine genel bir bakış içermektedir. Bu bölümde yöntemler hakkında literatür araştırması yapılmış ve bu yöntemlerin matematiksel altyapısı üzerinde durulmuştur. İncelenen yöntemler Lojistik Regresyon, Karar Ağacı, Naive Bayes ve K-En Yakın Komşular, Rastgele Orman'dır. Bu çalışma dengesiz veri kümesi üzerinde yapıldığından SMOTE[49] yöntemiyle veri dengeleme adımı da bu bölümde anlatılmış olacaktır. Ayrıca aday müşteriler için modelden kural çıkarımı aşamasında uygulanan algoritmik yöntem de bu bölümde sunulmaktadır. Ayrıca Minimum açıklama uzunluğu[47] tekniği ile de çıkarılan kuralların basitliğini ölçen bir ölçüt anlatılmaktadır. Bunun yanında birliktelik kural madenciliği başarı ölçüt metrikleri ve bunların "Children of the Tree" algoritmasındaki kullanılışı da bu bölümde özetlenmiştir[50].

2.1 Kredi Riski Tahmini için Makine Öğrenmesi Modelleri

Makine öğrenmesi (ML) modelleri, kredi riski tahmini gibi sınıflandırma problemlerinde sıkça kullanılmaktadır[1]. Çünkü bu problem doğası gereği, geçmişte borcunu ödemiş ya da ödememiş müşteriler gibi etiketlenmiş örneklerle çalışır. Bu durum denetimli öğrenme (supervised learning) **yöntemlerine** doğrudan uygundur. Bu kapsamda; **Lojistik Regresyon**, **KNN**, **Naive Bayes**, **SVM**, **DT** ve **Rastgele Orman** gibi yöntemler, farklı veri yapıları ve değişkenler arasındaki ilişkileri modelleyebilme yetenekleri sayesinde kredi riski analizinde yaygın olarak kullanılmaktadır. Bu modellerin her biri farklı varsayımlara ve sınıflandırma stratejilerine dayanmakta olup, bazıları doğrusal karar sınırlarına (örneğin lojistik regresyon)[51,52], bazıları ise daha karmaşık, doğrusal olmayan sınır yapılarına (örneğin Rastgele Orman)[8] daha uygundur.

Literatür araştırmalarında da bu modellerin benzer kredi riski problemlerinde kullanıldığı ve özellikle kullandığımız veri kümesine benzer yapıda olan veri kümeleri üzerinde test edildiği görülmüştür. Bu nedenle bu tez çalışmasında bu yaygın yöntemleri denenmiştir. Ayrıca, veri kümesinin sınıflar arasında **dengesiz dağılım göstermesi** nedeniyle, bu tür durumlarla başa çıkmak üzere önerilen güçlü yöntemlerden biri olan **SMOTE[49]** da uygulanmıştır ve tüm yöntemlerin sonuçları aynı ön işleme aşamalarından geçirilerek kaydedilmiştir. Tüm modeller sınıflandırma başarımı açısından değerlendirilmiş ve en başarılı performansı gösteren model üzerinden **kural çıkarımı** yapılmıştır. Bu bölümde denediğimiz ML algoritmalarının matematiksel temellerini ve işleyişini derinlemesine inceleyeceğiz.

2.1.1 Lojistik Regresyon

Lojistik Regresyon, tahmin edilmesi istenen sonucun olasılığı (y değeri) - sonuç değişkeni zorunlu olarak kategorik bir veridir (kişinin kredisini ödeyip ödeyemeyeceği) - ve nitelikler (x) arasındaki ilişkiye göre sınıflandırma yapan bir veri madenciliği algoritmasıdır[52,53]. Diğer bir deyişle, LR, verileri tanımlamak ve bağımlı bir ikili değişken ile bağımsız değişkenler arasındaki ilişkiyi açıklamak için kullanılır[52]. Bu yöntem, hem niceliksel hem de niteliksel verilerle çalışabilir[53]. Niceliksel veriler, sayısal değerlere sahip olan verilerdir. Örneğin, yaş, gelir, kredi tutarı gibi. Nitelliksel veriler ise, kategorik olarak sınıflandırılabilen verilerdir. Örneğin; cinsiyet, eğitim seviyesi, medeni durum gibi.

Lojistik regresyon, bağımlı değişkenin 0 ve 1 veya diğer karışık ikili değerlerle ifade edilebileceği bir model oluşturur. Bu model, verilerin incelenmesi ve sonuçların tahmin edilmesi için kullanılabilir[52,54].

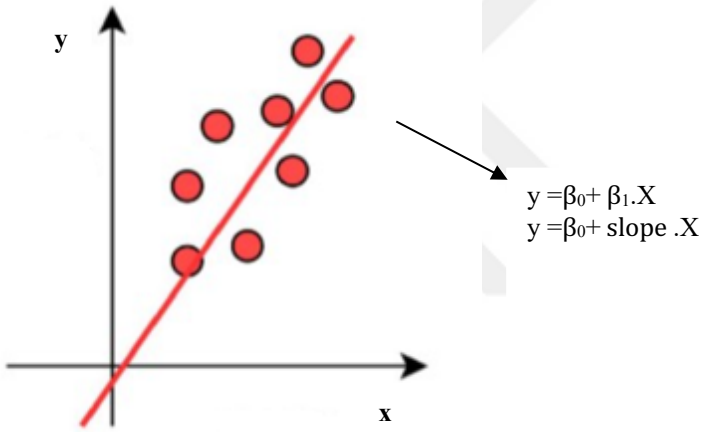
Aslında Lojistik Regresyon, "Genelleştirilmiş Doğrusal Model" olarak bilinen daha geniş bir tekniğin bir alt uygulamasıdır. 1972'de Nelder ve Wedderburn, lineer regresyon ile çözmeye uygun olmayan problemlerin çözümü için bazı teknikler önermiştir. Lojistik regresyon bunların özel bir durumu olarak ortaya çıkmıştır[55].

Lojistik Regresyon, [-∞, +∞] aralığındaki lineer regresyonun sigmoid fonksiyonu yardımıyla logit dönüşümü kullanılarak [0,1] aralığına dönüştürülmesiyle elde edilen istatistiksel bir regresyon modelidir[52,53].

Lojistik regresyonu anlamak için önce lineer regresyonu anlamak daha doğru olacaktır. Öncelikle lineer regresyondaki temel amacımız bağımlı ve bağımsız değişkenler arasındaki ilişkiyi doğrusal bir şekilde tanımlamaktır[56]. Bu amaçla doğrusal regresyon modeli, tahmin edilmeye çalışılan değerleri bir doğrusal fonksiyon yardımıyla veri setindeki gerçek değerlere en yakın şekilde bulmaya çalışır. Bunu, bağımlı değişkenin bağlı olduğu diğer tüm parametrelerin ağırlıklı toplamlarından oluşan genel tahmin denklemi (bağ fonksiyonu) ile yapar. Bu denklem aşağıdaki gibidir[52,54]:

$$g(y) = b_0 + b_1 x_1 + b_2 x_2 + \cdots = u(x) \tag{2.1}$$

Doğrusal regresyonun basit bir grafiği aşağıda Şekil 2.1'de verilmiştir. Burada y, g(y) korelasyon fonksiyonuna eşdeğer olarak gösterilmiştir.



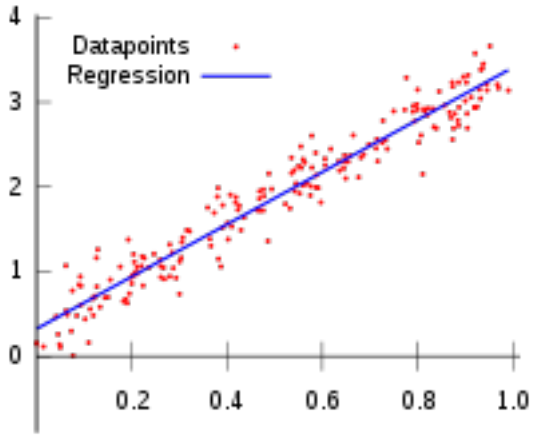
Şekil 2.1 Lineer regresyon grafiği

Doğrusal Regresyon modelinde, veri setindeki değerlerle hizalanmış en yakın doğrusal fonksiyonu çizmek için En Küçük Kareler Yöntemi (Least Squares Method - LSM) kullanılır[56].

En Küçük Kareler Yöntemi (Least Squares Method – LSM)

En küçük kareler yöntemi, iki bağımlı fiziksel büyüklük arasındaki matematiksel bağlantıyı olabildiğince gerçekçi bir denklem olarak yazmak için kullanılan standart bir regresyon yöntemidir. Başka bir deyişle, bu yöntem, ölçüm sonucundan elde edilen veri noktalarına "olabildiğince yakın" geçecek bir fonksiyon eğrisi bulmaya yarar. Gauss-Markov Teoremine göre en küçük kareler yöntemi, regresyon için en uygun yöntemdir[56]. Aşağıda Şekil 2.2'de Doğrusal olarak

dağıtılmış verilere sahip bir veri kümesindeki doğrusal fonksiyon grafiği örneği verilmiştir[57]. Bu örnek denklem (2.2)'deki gibi ifade edilebilir.



Şekil 2.2 Doğrusal olarak dağıtılmış verilere sahip bir veri kümesindeki doğrusal fonksiyon grafiği örneği[57]

Basit bir örnek vermek gerekirse, aralarında lineer bir bağlantı bulunan X ve Y adlı iki fiziksel niceliği göz önünde bulundurarak; Y, X'in bir fonksiyonu olarak yazılabilir. Bu iki miktar arasındaki bağlantı doğrusal olduğundan, bir denklem olarak ifade edilebilir[56]:

$$Y = AX + B \tag{2.2}$$

X ve Y arasındaki bağlantı tek bir doğrusal denklem olarak ifade edilecekse, bu noktalara mümkün olduğunca yakın geçecek bir çizgi bulunmalıdır. Başka bir deyişle, yukarıdaki denklemde a ve b, elde edilen çizgi veri noktalarına mümkün olduğunca yakın olacak şekilde değerler olarak seçilmelidir.

En küçük kareler yöntemi, denklemin verdiği (teorik) Y değerleri ile ölçümlerin verdiği (gerçek) Y değerleri arasındaki farkların karelerinin toplamının indirilmesi fikrine dayanmaktadır. Bu yöntem, denklemdeki a ve b sayılarını, söz konusu kareler toplamı en küçük (dolayısıyla adı buradan geliyor) olacak şekilde seçer[56].

Logit Dönüşümü (The Logit Transformation)

Lojistik regresyon, lineer regresyondan türetilen bir regresyon modeli olmasına rağmen, bağımlı ve bağımsız değişkenler arasında lineer bir ilişki varsaymadığı için lineer regresyondan farklıdır. Ancak bağımsız değişkenlerin logit değerleri ile sonuç değişkeni arasındaki doğrusallığı esas alır[52,53].

Lojistik regresyon sadece bağımlı değişken tarafından karakterize edilen olayın varlığı veya yokluğu ile ilgilendiğinden, bunu gösterecek g () bağ fonksiyonu g()>0 ve g()≤1 koşullarını da sağlamalıdır[52,53]. Diğer bir deyişle bağımlı değişken y başarı olasılığı (p) ve başarısızlık olasılığı (1-p) olduğundan; p≥0 ve p≤1 olduğu söylenebilir. Bu durumda g() fonksiyonunun her zaman pozitif olması durumu üstel bir fonksiyon kullanılarak (2.3) denklemindeki gibi ifade edilebilir[52,53].

$$g = e^{u(x)} \tag{2.3}$$

Aynı zamanda fonksiyonun g'den büyük bir sayıya bölünmesi gerekir ki g() her zaman 1'den küçük olsun. Burada oluşan p olasılığı (2.4)'deki gibi ifade edilebilir[52,53,54].

$$p = \frac{e^{u(x)}}{e^{u(x)} + 1} \tag{2.4}$$

Eğer denklem (2.1) , denklem (2.4)'te yerine konursa, olasılık fonksiyonu p aşağıdaki gibi olur[52,53].

$$p = \frac{e^{y}}{e^{y} + 1} \tag{2.5}$$

p başarı olasılığı ve q başarısızlık olasılığı ise [52]'deki gibi,

$$q = 1 - p \tag{2.6}$$

(2.5) denklemindeki p değeri (2.6)'da yerine yazılırsa, (2.7) denklemini oluşturur[52].

$$q = 1 - \frac{e^{y}}{e^{y} + 1} \tag{2.7}$$

p ve q arasındaki oran hesaplanırsa, denklem tek oran olarak adlandırılan denklem (2.8) olur[52,53].

$$\frac{p}{1-p} = e^{y} \tag{2.8}$$

(2.8) denkleminde eşitliğin her iki tarafının logaritması alınırsa y değeri aşağıdaki gibi elde edilir (Böylece logit dönüşümü yapılmıştır)[52,53,55].

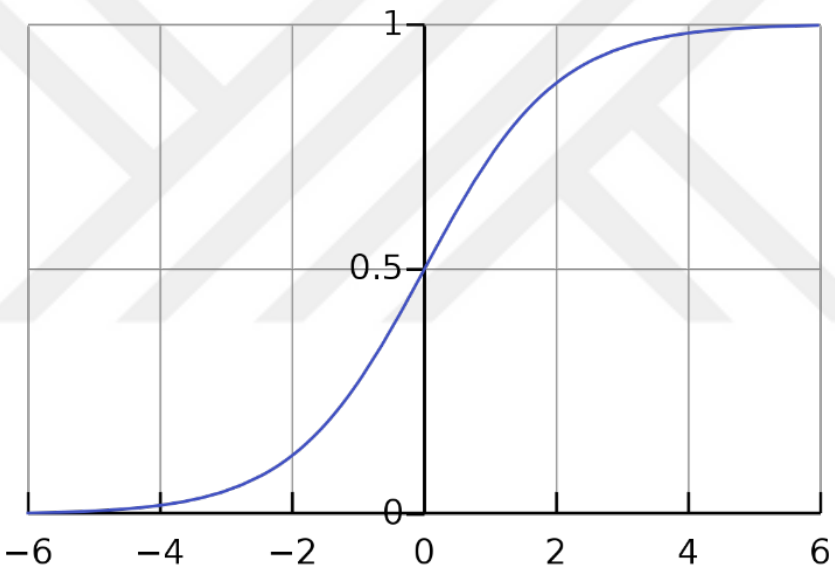
$$y = \log\left(\frac{p}{1-p}\right) \tag{2.9}$$

Buradan modele geri dönülür ve (2.1) denklemindeki u(x) denklem (2.9)'da yerine konursa (2.10) denklemi elde edilir[52,53,54].

$$u(x) = \log\left(\frac{p}{1-p}\right) \tag{2.10}$$

Yukarıda p'nin bağımsız değişkenlerin doğrusal bir fonksiyonu olduğu söylendi. Böylece doğrusal olmayan bir yapı doğrusal bir şekilde gösterilmiştir.

Denklemin (2.5) diğer adı lojistik fonksiyondur[52,53]. Sigmoid fonksiyonun yaygın olarak kullanılan bir şeklidir. Grafiği *Şekil 2.3*'te gösterildiği gibidir.

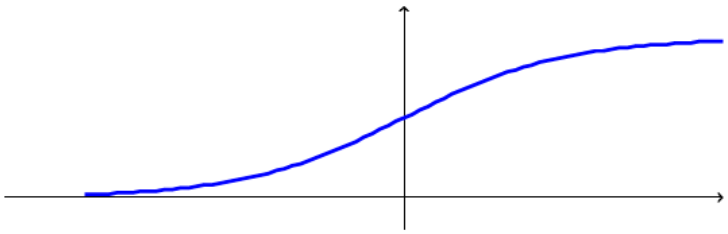


Şekil 2.3 Lojistik fonksiyon grafiği[53]

Sigmoid Fonksiyonu

Bir sigmoid fonksiyonu, karakteristik bir "S" şekilli eğriye veya sigmoid eğriye sahip olan matematiksel bir fonksiyondur. Sigmoid fonksiyonun yaygın bir örneği, Şekil 2.4'te gösterilen ve aşağıdaki formül ile tanımlanan lojistik fonksiyondur[52,53].

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} \tag{2.11}$$



Şekil 2.4 Sigmoid fonksiyonun grafiği[53]

Bir ikili sınıflandırma probleminde bazı eğitim verileri verilsin:

$\mathcal{D} = \{ (x^n, y^n), n = 1,2, \dots, N\}$, burada hedef değişken y∈{0,1}'dir. Yeni bir x girişinin sınıf 1'e ait olma olasılığı kullanılarak atanabilir[53,54].

$$p(y = 1|x) = f(x^T w) \tag{2.12}$$

Burada f fonksiyonu lojistik fonksiyondur. Lojistik regresyon modele karşılık gelir[53].

$$p(y = 1|x) = \sigma(b + x^T w) \tag{2.13}$$

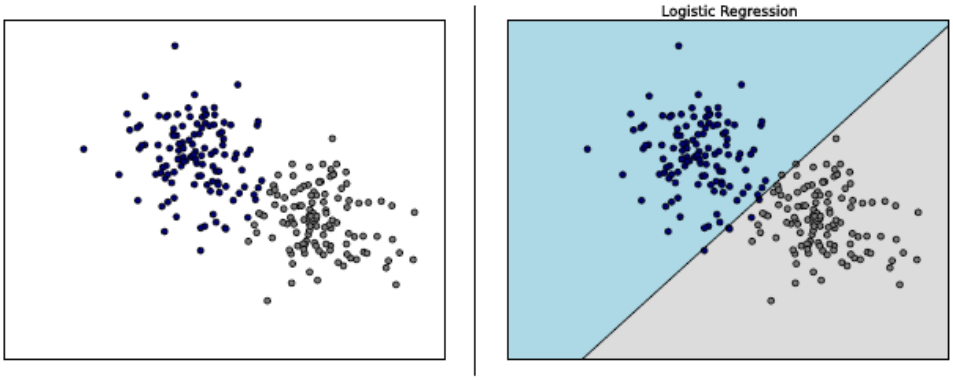
Burada b bir skalerdir ve w bir vektördür. Sigmoid fonksiyonunun $b + x^T w$ argümanı arttıkça, x'in 1. sınıfa ait olma olasılığı artar[53,54].

Karar Sınırı(Decision Boundary)

Karar sınırı, $p(c = |x) = p(c = 0|x) = 0.5$ olan x kümesi olarak tanımlanır. Bu, hiper düzlem tarafından verilir.

$$b + x^T w = 0 \tag{2.14}$$

Karar sınırı $p(c = |x) = 0.5$ (düz çizgi). İki boyutlu veriler için karar sınırı bir çizgidir. Sınıf 1 için tüm eğitim verileri (mavi bölgede koyu mavi daireler) çizginin bir tarafında ve sınıf 0 için (gri bölgede gri daireler) diğer tarafta yer alıyorsa, verilerin Şekil 2.5'te gösterildiği gibi doğrusal olarak ayrılabilir olduğu söylenir[58].



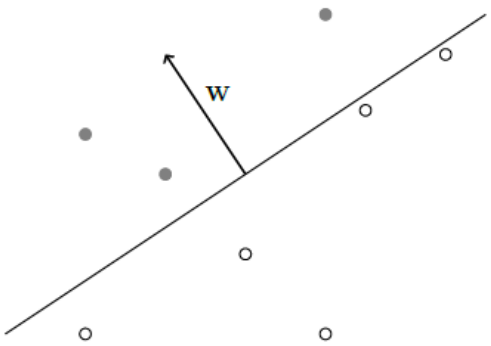
Şekil 2.5 Bir Lojistik Regresyon sınıflandırıcısı kullanılarak sınıflandırılan iki boyutlu örnek veri kümesi için karar sınırı[58]

Aşağıdaki Şekil 2.6'da hiperdüzlemin $b + x^T w > 0$ olan tarafında x girdilerinin 1'ler, diğer tarafında ise 0'lar olarak sınıflandırıldığı açıkça görülmektedir. "Sapma" parametresi (bias) b, karar sınırını sabit bir miktarda kaydırır. Karar sınırının oryantasyonu, hiper düzlemin normali olan w tarafından belirlenir[53,54], bkz. Şekil 2.6.

Geometrik yorumu açıklığa kavuşturmak için, x karar sınırında bir nokta olsun ve yeni bir $x^* = x + w^{\perp}$ noktası olsun. Burada w$^{\perp}$, w'ye dik bir vektördür, böylece $w^T w0^{\perp} = 0$. Daha sonra,

$$w^T x^* = b + w^T(x + w^{\perp}) = b + w^T x + w^T w^{\perp} = b + w^T x = 0 \qquad (2.15)$$

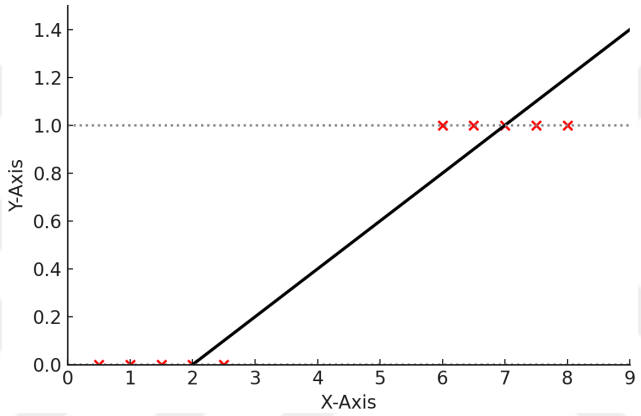
Böylece, eğer x karar sınırındaysa, x artı w'ye dik herhangi bir vektör de öyledir. D boyutlarında, w'ye dik olan vektörlerin uzayı, D - 1 boyutlu bir hiper düzlem oluşturur[53]. Örneğin, veri iki boyutluysa, karar sınırı Şekil 2.6'da gösterildiği gibi tek boyutlu bir hiper düzlem, bir çizgidir.



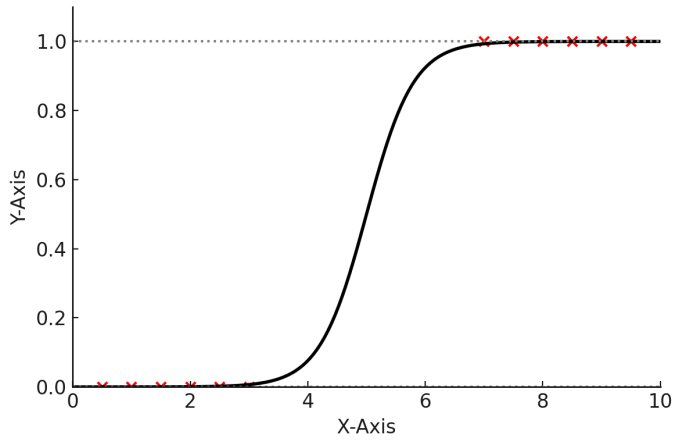
Şekil 2.6 Karar sınırının görselleştirilmesi[53]

Yukarıda belirtildiği gibi, gerçek değerler ile LSM tahmini kullanılarak tahmin edilen değerler arasındaki farkın doğrusal modelde en düşük olduğu regresyon modeli (2.2)'deki gibi bulunur[56]. Lojistik Regresyonda, LSM yerine Maximum Likelihood Method (MLM) kullanılır[52,53]. Çünkü Lojistik Regresyonda tahmin edilen durum sınıfların olasılık değerleridir.

Lojistik Regresyon ve Lineer Regresyon arasındaki fark aşağıdaki Şekil 2.7 ve Şekil 2.8'de olduğu gibi gösterilebilir - x ekseni argümanı, y ekseni sonuç değişkenini temsil eder.



Şekil 2.7 Lineer regresyon



Şekil 2.8 Lojistik regresyon

Şekil 2.7 ve Şekil 2.8'de görüldüğü gibi sonuç değişkeninin değerleri ({0,1}, olasılık değerleri) için doğrusal regresyon kullanılması uygun değildir.

Maximum Olabilirlik Yöntemi(Maximum Likelihood Method - MLM)

İstatistikte, maksimum olabilirlik tahmini, olabilirlik fonksiyonunu optimize ederek bir modelin parametrelerini tahmin eden, yani kullanılan istatistiksel modelde yapılan gözlemlerin en olası olduğu parametreleri seçen bir tahmin yöntemidir[59].

Maksimum olabilirlik yönteminin arkasındaki temel ilke beklentidir:

> "Rastgele bir olayın meydana gelmesi, olayın gerçekleşmesi en muhtemel olay olmasından kaynaklanır."

Bu yöntem 1920'lerde İngiliz istatistikçi Sir Ronald A. Fisher (1890-1962) tarafından keşfedildi[59].

Belirli bir yığından[4] X_1, X_2, X_3, …, X_n şeklinde çekilen örnekler olsun. Bu örneklerin ait olduğu yığının dağılım türünün bilindiği ve örneklerin birbirinden bağımsız olduğu varsayılır.

Amaç, yığının olasılık yoğunluk fonksiyonunu şekillendiren parametreleri bilinen örneklerden tahmin etmektir. Yığının olasılık yoğunluk fonksiyonu p (x|θ) olsun[52,53].

X_1, X_2, X_3, ..., X_n'nin birbirinden bağımsız olduğu bilindiğinden,

$$L(\theta) = \prod_{k=1}^{n} p(x_k \mid \theta) \tag{2.16}$$

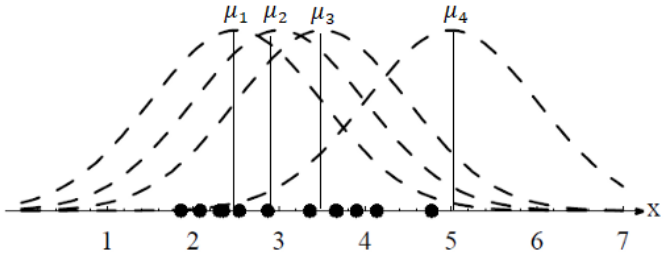
L(θ) "olabilirlik fonksiyonu" olarak adlandırılır. L()'yi maksimize etmek için parametresine göre türev alınırsa, türev ifadesini sıfır yapan değer, p (x|θ) 'nın θ parametresi (yığının olasılık fonksiyonu) için tahminimiz olacaktır[60].

Durum aşağıdaki Şekil 2.9 Gauss dağılımından tek boyutlu uzaydaki noktalar ile özetlenebilir.

Tek boyutlu uzaydaki noktalar aşağıdaki gibi Gauss dağılımından kaynaklanır. Noktaların kaynaklandığı dağılımın varyans (σ^2) parametresinin bilindiği ve ortalama (μ) parametresinin bilinmediği varsayılsın. Noktaların ait olduğu sonsuz

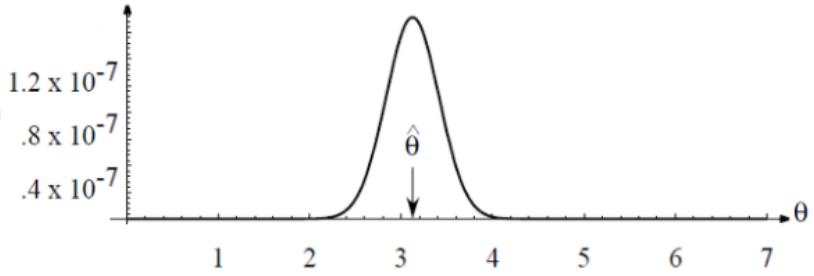
[4] Yığın (population), istatistiksel açıdan incelenmek istenen tüm birey, nesne, olay ya da ölçüm değerlerinin oluşturduğu, genellikle çok büyük veya sonsuz büyüklükte varsayılan temel veri evrenidir. Örneklem bu yığından rastgele seçilerek elde edilir ve yapılan analizlerin amacı, bu örnek üzerinden yığının özellikleri hakkında genelleme yapmaktır.

yığın adayı olabilir. Bu sonsuz yığın adaylarından dördü aşağıda gösterilmiştir. Ortalama (μ) parametresi nedeniyle her kesikli çizginin konumu farklıdır. Çünkü her birinin değeri farklıdır. Bu yüzden noktalarımızın ait olduğu sonsuz yığın adayı olabilir. Ancak burada μ parametresinin puanlara göre en uygun olanı tahmin edilecektir[59].



Şekil 2.9 Gauss dağılımından tek boyutlu uzaydaki noktalar[61]

Bu noktalar kullanılarak, yukarıdaki denklem (2.16)'daki formül yardımıyla Olabilirlik Fonksiyonu L(θ) çıkarılır. L(θ) grafiği aşağıda Şekil 2.10'daki gibidir.



Şekil 2.10 L(θ) grafiği[62]

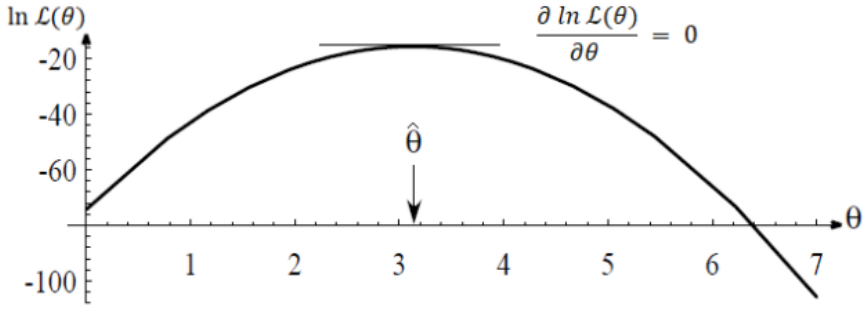
L(θ)'nin "*e*" tabanına göre logaritması alınırsa;

$$\ln L(\theta) = ln(p(x_1|\theta) \,.\, p(x_2|\theta) \;\ldots\; p(x_n|\theta))$$

$$= ln\, p(x_1|\theta) \;+\; ln\, p(x_2|\theta) \;+\; \ldots \;+\; ln\, p(x_n|\theta)$$

$$= \sum_{k=1}^{n} \ln(\mathrm{p}(x_k\,|\theta)) \qquad (2.17)$$

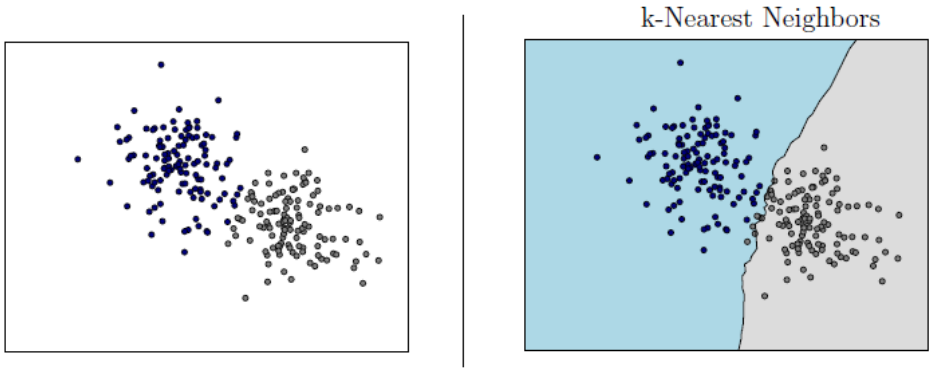
denklem (2.17) yukarıdaki gibi elde edilir. Daha sonra (2.17) denkleminin θ'ya göre türevi ve lnL(θ) 'yi maksimum yapan θ değeri bulunur. Bu değerin grafiği aşağıda Şekil 2.11'deki gibidir.



Şekil 2.11 Maksimum θ değerini gösteren ln L(θ) fonksiyonu[63]

2.1.2 K En Yakın Komşular (K-Nearest Neighbors)

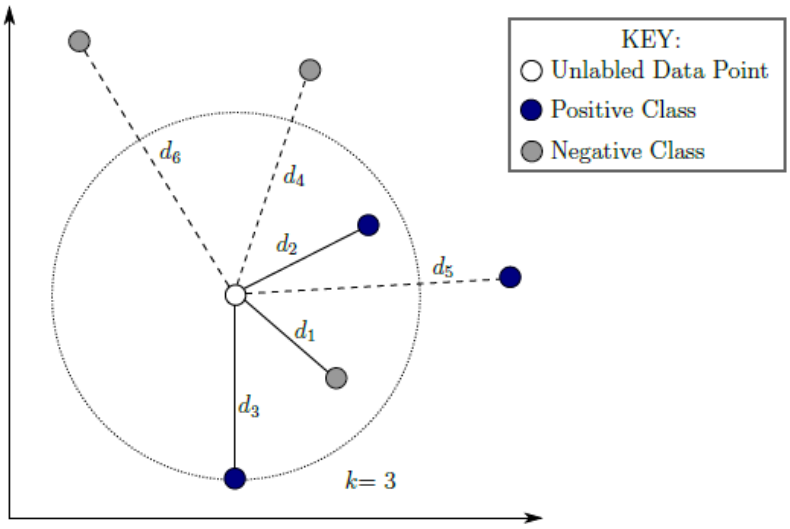
K-En Yakın Komşular, sınıflandırma işlemlerinde kullanılan hafıza tabanlı yöntemler arasında sayılan bir yöntemdir. Etiketlenmemiş yeni bir veri noktası, en yakın k komşusuyla ilişkili bir çoğunluk etiketi atanarak sınıflandırılır. Bu en yakın komşular, Öklid mesafesine dayalı olarak en yakın k veri noktasıdır. Aşağıda Şekil 12'de KNN için karar sınırı verilmiştir[58].



Şekil 2.12 K-En Yakın Komşu algoritması için karar sınırı[58]

Bu yöntemde gözlem değerleri arasındaki uzaklıklar kullanılarak sınıflandırma işlemi yapılır. Uzaklık ilişkisi olarak çoğunlukla Öklid uzaklık formülü kullanılır[64]. Şekil 2.13'te k=3 ile KNN için açıklamalı bir görsel verilmiştir.

$$d(i,j) = \sqrt{\sum_{k=1}^{p} \left(x_{ij} - x_{jk}\right)^2} \tag{2.18}$$

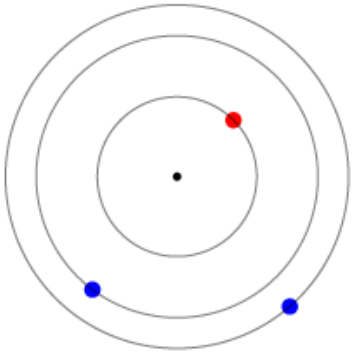


Şekil 2.13 k=3 ile k-En Yakın Komşular sınıflandırma algoritması için görsel açıklama[58]

Algoritma şu şekilde çalışır.

- Bir k parametresi belirlenir. Bu parametre, belirli bir noktaya en yakın komşu sayısıdır.
- Bu algoritma belirli bir noktaya en yakın komşuları belirlediğinden, o nokta ile diğer tüm noktalar arasındaki mesafeler ayrı ayrı hesaplanır.
- Yukarıda hesaplanan uzaklıklara göre sıralar büyükten küçüğe sıralanır ve aralarından en küçük k seçilir.
- Seçilen satırların ait olduğu kategori belirlenir ve en çok tekrarlanan kategori değeri seçilir.
- Seçilen kategori tahmin edilmesi beklenen gözlem değerinin kategorisi olarak kabul edilir[64].

Sınıflandırmayı yalnızca en yakın tek komşuya dayandırmak yanlışlıklara yol açabilir. Komşu basitçe yanılıyorsa (yanlış bir eğitim sınıfı etiketine sahipse) veya sınıfının özellikle temsili bir örneği değilse, bu durumlar tipik olarak yanlış bir sınıflandırmayla sonuçlanacaktır. Tek bir en yakın komşudan daha fazlasını dahil ederek, daha yumuşak bir karar sınırı ile daha sağlam bir sınıflandırıcı yapılabilir[65].



Şekil 2.14 Sınıflandırılacak merkezli bir hiperküre noktası ve komşuları[65]

K-en yakın komşularda, Şekil 2.14'te sınıflandırılacak noktanın (burada merkezi nokta) etrafında bir hiperküre ortalanır. İç daire en yakın komşuya karşılık gelir. Ancak, en yakın 3 komşuyu kullanarak iki "mavi" sınıf ve bir "kırmızı" sınıf olduğunu buluruz - ve bu nedenle nokta "mavi" sınıf olarak sınıflandırılır. Beraberlik durumunda, k beraberlik bozulana kadar uzatılabilir[65].

Farklılık ölçüsü olarak Öklid mesafesinin kabul edilmesinin nedeni budur, K-En Yakın Komşular algoritması, x test noktası merkezli bir hiperküreyi dikkate alır. Hiperküre eğitim verilerinde tam olarak K noktası içerene kadar r yarıçapı artırılır. Sınıf etiketi c(x) daha sonra hiperküre (hipersfer) içindeki en çok sayıda sınıf tarafından verilir[65].

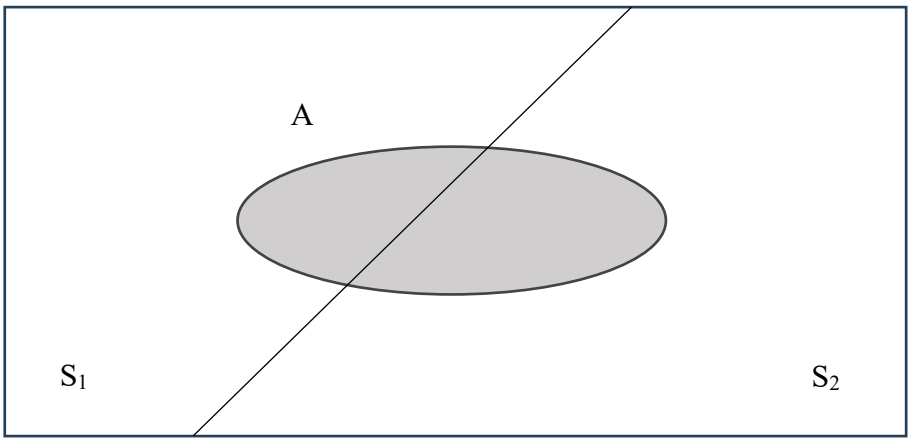
2.1.3 Naive Bayes

Naive Bayes sınıflandırıcısı, Bayes teoremine dayanan olasılıksal bir yaklaşımdır[66].

Bayes' Teoremi

Bayes teoremine dayalı sınıflandırmalar yapmak mümkündür. S_1 ve S_2 ayrık (uyumsuz) olaylardır [66,67].

$$S_1 \cap S_2 = \emptyset. \tag{2.19}$$



Şekil 2.15 S_1, S_2 ve A olayları[67]

A olayının T_1 ve T_2 cinsinden açıklanması. Bunun için koşullu olasılık kullanılır.

$$P(S_1|A) = \frac{P(A|S_1)}{P(A)} \tag{2.20}$$

Şekil 2.15'te görüldüğü gibi, A olayı S_1 ve S_2 olaylarından birinde gerçekleşir. A olayı için aşağıdaki bağıntı yazılabilir[67].

$$A = (S_1 \cap A) \cup (S_2 \cap A) \tag{2.21}$$

Böylece, A olayının olma olasılığı şu şekilde yazılabilir:

$$P(A) = P(S_1 \cap A) + P(S_2 \cap A) \tag{2.22}$$

Denklem (2.20) kullanılarak aşağıdaki bağıntı elde edilebilir.

$$P(S_1|A) = \frac{P(A|S_1)}{\mathrm{P}(S_1 \cap \mathrm{A}) + \mathrm{P}(S_2 \cap \mathrm{A})} \tag{2.23}$$

Bu sonuç genelleştirilebilir. Temel küme S_1, S_2, …, S_n olaylarından oluşuyorsa ve olasılıklar sıfır değilse, A olayının S_j olayında meydana gelme olasılığı aşağıdaki gibi hesaplanır:

$$P(S_j|A) = \frac{P(A|S_j)}{\sum_{i=1}^{n} \mathrm{P}(S_i \cap \mathrm{A})} \tag{2.24}$$

$P(A|S_j) = P(A|S_j)P(S_j)$ olduğu için aşağıdaki gibi yazılabilir[67]:

$$P(S_j|A) = \frac{P(A|S_j)\mathrm{P}(S_j)}{\sum_{i=1}^{n} \mathrm{P}(S_i \cap \mathrm{A})} \tag{2.25}$$

Bayes' Sınıflandırıcısı

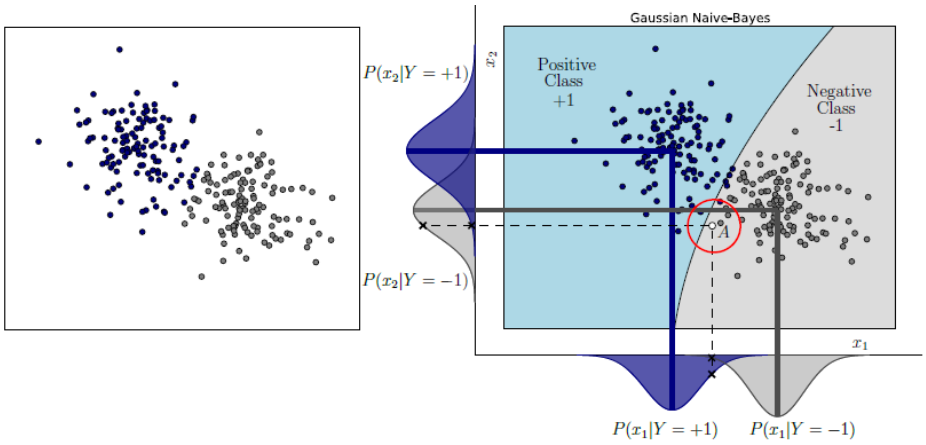
Bayes sınıflandırıcıları istatistiksel sınıflandırma teknikleri arasındadır. Sınıflandırma işleminin başında, X kümesinin sınıf üyeliği bilinmeyen bir veri kümesi olarak kabul edilirse ve bu X veri kümesinin C sınıfına ait olduğu hipotezi H ise, P(H|X) olasılığı hesaplanabilir. H'nin C sınıfına ait olduğu varsayımıyla. Burada P(H|X), H hipotezinin X üzerinde şartlandırılmasına ilişkin "olasılık sonrası" olarak bilinir. Bayes ilişkisi şu şekilde ifade edilebilir[67]:

$$\mathrm{P}(H|X) = \frac{\mathrm{P(X|H)P(H)}}{P(X)} \tag{2.26}$$

P(H), hipotezi oluşturan bir "öncel olasılıktır" (priori). P(X|H), H koşulunda bir "sonsal olasılık"tır.

Naive Bayes Sınıflandırıcısı

Naive Bayes sınıflandırıcısı, sınıflandırmadan önce veri setindeki gözlem değerlerinin her biri için olasılık değerlerini hesaplar. Algoritma, bir eleman için her durumun olasılığını hesaplayarak çalışır ve bunu olasılık değeri en yüksek olana göre sınıflandırır. Her eğitim örneği, bir hipotezin doğru olma olasılığını kademeli olarak artırabilir/azaltabilir -önceki bilgi gözlemlenen verilerle birleştirilebilir[66]. Naive Bayes algoritması, adını özellik bağımsızlığının saf varsayımından alır. Naive Bayes algoritmasının karar sınırı Şekil 2.16'da gösterilmektedir. Özellik eksenleri boyunca sınıf etiketleri (Y = [y_1 = +1; y_2 = -1]) için koşullu olasılıklar aşağıda Şekil 2.16'da gösterilmiştir. Koşullu olasılıkların bir Gauss dağılımından alındığı varsayılır[58].



Şekil 2.16 Naive Bayes algoritmasının karar sınırı[58]

Küçük bir eğitim verisi ile çok başarılı bir sınıflandırma yapabilmektedir. Öte yandan, veri girişi ve çıkışı değişken olan dinamik veri kümelerinde, her sınıflandırma işlemi için veri setini tekrar tekrar taramak zorunda olduğundan sınıflandırma yapmak zor (maliyetli) olacaktır. Algoritma şu şekilde açıklanabilir:

X, sınıf üyeliği bilinmeyen bir veri örneği olsun. Örnek X= {x_1, x_2, …, x_n} öznitelik değerlerinden oluşmalıdır. Bu örnek sınıfta m sınıf olduğunu varsayıldığında; C_1, C_2, …, C_m sınıf değerleri olsun. Sınıfı belirlenecek örnek için,

$$\mathrm{P}(C_i|\mathrm{X}) = \frac{\mathrm{P}(\mathrm{X}|C_i)\mathrm{P}(C_i)}{P(X)} \tag{2.27}$$

olasılıklar hesaplanır. $\mathrm{P}(\mathrm{X}|C_i)$ olasılığı için, örneğin x_i değerlerinin birbirinden bağımsız olduğu varsayılarak aşağıdaki denklem kullanılabilir[68].

$$\mathrm{P}(\mathrm{X}|C_i) = \prod_{k=1}^{n} \mathrm{P}(x_k|C_i) \tag{2.28}$$

Bilinmeyen X örneğini sınıflandırmak için, (2.27) denklemindeki $\mathrm{P}(C_i|\mathrm{X})$ içindeki paydalar birbirine eşit olduğundan, yalnızca pay değerlerini karşılaştırmak yeterlidir. Bu değerlerden en büyüğü seçilerek bilinmeyen örneğin bu sınıfa ait olduğu belirlenir[67].

$$A\mathrm{rg}\max_{C_i}\{\mathrm{P}(\mathrm{X}|C_i)\mathrm{P}(C_i)\} \tag{2.29}$$

Sonsal olasılıkları kullanan yukarıdaki ifade (2.29), "Maksimum A Posteriori Sınıflandırması (MAP)" olarak da bilinir. Son olarak, denklem (2.28) nedeniyle, aşağıdaki denklem Bayes sınıflandırıcısı olarak kullanılabilir[66]:

$$C_{MAP} = \arg\max_{C} \prod_{k=1}^{n} \mathrm{P}(x_k|C_i) \tag{2.30}$$

Koşullu olasılık değerlerini hesaplamak için sayısal veri dağılımının normal dağıldığı varsayılırsa Gauss dağılımı kullanılabilir. Olasılık yoğunluk fonksiyonu şu şekilde hesaplanabilir[68]:

$$\mathrm{P}(x_k|C_i) = f(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}.\sigma_{C_i}} e^{\frac{-\left(x_k-\mu_{C_i}\right)^2}{2\sigma_{C_i}^2}} \tag{2.31}$$

burada μ_{C_i} ortalama, σ_{C_i} ise standart sapmadır.

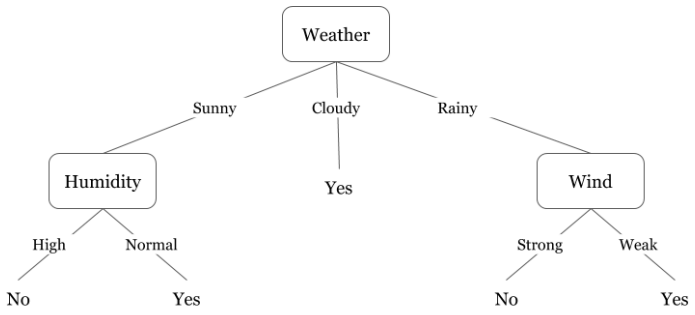
Sıfır Olasılık (Sıfır Frekans) Sorunundan Kaçınmak

Test setindeki bir değerin eğitim setinde gözlenemeyen bir değeri varsa olasılık değeri olarak sıfır verir yani tahmin edemez. Bu durum genellikle “Sıfır Frekans” olarak bilinir. Düzeltme teknikleri bu durumu çözmek için kullanılabilir. En basit düzeltme tekniklerinden biri “Laplacian Estimator (Laplacian Correction)” olarak bilinir. Laplace düzeltmesi, her duruma bir eklenerek uygulanabilir. Elde çok fazla veri olduğu düşünülürse her bir duruma ait özellik değerlerine 1 eklenmesi durumu genel olarak etkilemeyeceği gibi sıfır olasılıktan da kurtulmamıza yardımcı olacaktır[69].

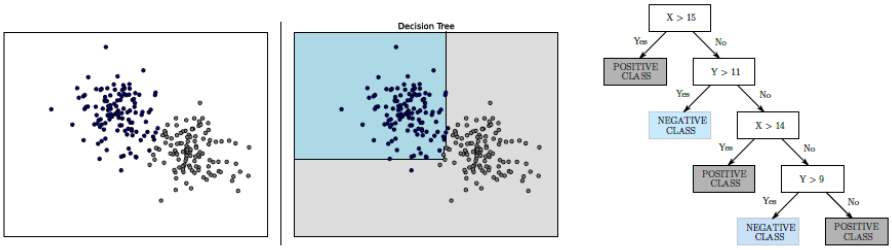
2.1.4 Karar Ağacı (Decision Tree)

Karar ağaçları, sıralı bir karar sürecini grafiksel olarak düzenlemenin bir yoludur[58]. Sınıflandırma ağaçları, akış şemalarına benzer yapılardır. Bir Karar Ağacı, her bir düğümün bir özelliğe uygulanan bir if-else testini temsil ettiği ve dalların olası sonuçları temsil ettiği düğümlerden ve dallardan oluşur[65]. Her öznitelik bir düğüm tarafından temsil edilir. Dallar ve yapraklar ağaç yapısının unsurlarıdır. Son yapıya "yaprak", üst yapıya "kök" ve aralarındaki yapılara "dal" adı verilir[69].

Sınıflandırma modelleri, çok sayıda öznitelik içeren bir veri kümesinden türetilebilir. Ancak, bazı özellikler bu modele dahil edilmeyebilir. Bir sınıflandırma ağacı örneği aşağıda Şekil 2.17'de verilmiştir.



Şekil 2.17 Sınıflandırma ağacı örneği[65]



Şekil 2.18 İki boyutlu veri seti ve düğümler ve dallar ile karar ağacı algoritmasının karar sınırı[58]

Her dikdörtgen bir düğümü temsil eder ve dallar Şekil 2.18'de oklarla temsil edilir.

Karar Ağaçlarında Dallanma Kriterleri

Sınıflandırma ağaçlarındaki en önemli sorunlardan biri de hangi kriterlere göre bölümleme yani herhangi bir kökten dallanma yapılacağıdır. Aslında, bir karar ağacı algoritması her bir farklı kritere karşılık gelir.

Entropi tabanlı segmentasyon kullanan algoritma örnekleri, ID3 ve C4.5 algoritmalarıdır.

Entropi

Bir sistemdeki belirsizliğin ölçüsüne "entropi" denir. S bir kaynak olsun. Bu kaynağın n mesaj, mesajlar $\{m_1, m_2, \ldots, m_n\}$ olmak üzere, üretebileceği ve tüm mesajların birbirinden bağımsız olduğu varsayılmaktadır. $P = \{p_1, p_2, \ldots, p_n\}$ olasılık dağılımı ile mi mesajları üretme olasılığı pi'dir mesajları üreten S kaynağının entropisi H(S) aşağıdaki gibidir[67]:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{2.32}$$

Karar Ağacında Entropi

Karar ağaçları oluşturulurken dallanmaya hangi öznitelikten başlanacağı önemlidir. Çünkü sınırlı sayıda kayıttan oluşan bir eğitim seti kullanarak olası tüm ağaç yapılarını ortaya çıkarmak ve aralarından en uygununu seçerek oradan başlamak zor bir iştir. Bu nedenle karar ağacı algoritmalarının çoğu baştan bazı değerleri

hesaplar ve buna göre bir ağaç oluşturur. Bu amaçla entropi kavramı kullanılabilir ve ağacın entropi değerine göre dallanması gerçekleşebilir.

Veri tabanından eğitim için elde edilen kayıt setinin eğitim seti sınıfının alacağı değerlere göre $\{C_1, C_2, \ldots, C_k\}$ şeklinde k sınıfa ayrıldığını düşünelim. Bu sınıflar hakkında ortalama bilgi miktarına ihtiyaç duyulabilir. Burada S, sınıf değerlerini içeren küme için P_S sınıflarının olasılık dağılımıdır ve aşağıdaki gibi hesaplanır[69].

$$P_S = \left(\frac{|C_1|}{|S|}, \frac{|C_2|}{|S|}, \ldots, \frac{|C_k|}{|S|}\right) \tag{2.33}$$

$|C_i|$ ifadesi, Ci kümesindeki eleman sayısını verir. Burada, örneğin, $p_1 = \frac{|C_1|}{|S|}$ olasılığı ifade eder. Dolayısıyla, S veya entropi için ortalama bilgi miktarı şu şekilde ifade edilir:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{2.34}$$

Dallanma ve Kazanç Kriterleri için Özniteliklerin Seçilmesi

Hedef özniteliği temsil eden S'nin, hedef özniteliği olmayan (yani bir sınıf özniteliği olmayan) bir X özniteliğinin değerlerine bağlı olarak $S_1, S_2, \ldots, S_n$ alt kümelerine bölünsün. S'nin bir elemanının sınıfını belirlemek için gereken bilgi, S_i'nin bir elemanının sınıfını belirlemek için gereken bilginin ağırlıklı ortalaması olarak kabul edilir. Dolayısıyla bu tanıma göre S'nin bir elemanının sınıfını belirlemek için gerekli olan bilgi aşağıdaki gibi hesaplanır[69].

$$H(X,S) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} H(S_i) \tag{2.35}$$

S setinin X testine[5] göre bölünmesi sonucunda elde edilen bilgiyi ölçmek için "kazanç kriteri" adı verilen bir ifade kullanılmaktadır. Bu kriter [67]'deki gibi aşağıda denklem 2.36'daki olarak tanımlanabilir:

$$Kazanç\ (X,S) = H(S) - H(X,S) \tag{2.36}$$

[5] Veri kümesi S'yi, öznitelik X'in değerlerine göre alt parçalara ayıran bir karar koşulu veya dallanma kuralıdır**.** Örneğin; Eğer X = "Yaş" değişkeni ise; test = "Yaş > 30?" olabilir.

Burada ayrıştırma yapılırken Kazanç (X, S) değerinin maksimize edilmesi amaçlanmaktadır. En yüksek bilgi kazancını sağlayan ve kazancı maksimize eden X özelliği seçilir.

Kazanç kriteri hesaplanırken eğer bir özellik diğerlerinden çok farklı değerler içeriyorsa (2.35)'teki *H (X,S)* değeri sıfır veya sıfıra yakın olacaktır; (2.36)'daki bilgi kazancı en yüksek değere ulaşacağı için yanıltıcı sonuçlar elde edilebilir. Söz konusu problemin çözümü için veri setinde böyle bir öznitelik varsa bilgi kazanımı yerine "kazanç oranı" adı verilen bir ilişki kullanılmaktadır[67].

Kazanç oranının hesaplanmasında bölünmüş bilgi adı verilen kavram Quinlan [69] tarafından ortaya atılmıştır. Bu yol, S kümesi için X özniteliğinin değerini belirlemek için gerekli bilgi miktarını ortaya çıkarmak amacı ile bulunmuştur. Söz konusu bilgi miktarı $H(P_{X,S})$ şeklinde ifade edilebilir. Burada $P_{X,S}$ ifadesi X öznitelik değerlerinin olasılık dağılımıdır ve aşağıdaki gibi hesaplanır[67]:

$$P_{X,S} = \left(\frac{|S_1|}{|S|}, \frac{|S_2|}{|S|}, \dots, \frac{|S_k|}{|S|}\right) \tag{2.37}$$

Burada $H(P_{X,T})$ miktarı, X özniteliği için bölünmüş bilgi olarak bilinir. Bu değer şu şekilde hesaplanır:

$$H(P_{X,S}) = H\left(\frac{|S|}{|S|}, \frac{|S|}{|S|}, \dots, \frac{|S_k|}{|S|}\right) \tag{2.38}$$

Bunun yerine aşağıdaki ifade kullanılabilir:

$$H(P_{X,S}) = \sum_{i=1}^{k} \frac{|S_i|}{|S|} H(S_i) \tag{2.39}$$

$H(P_{X,S})$ değeri ve yukarıda elde edilen kazanç kriteri yardımıyla kazanç oranı hesaplanır[67]:

$$Kazanç\ Oranı\ (X,S) = \frac{Gain(X,S)}{\mathrm{H}(P_{X,S})} \tag{2.40}$$

ID3 Algoritma Adımları

Quinlan tarafından karar ağaçları ile sınıflandırma yapmak için geliştirilen birçok algoritmadan biri de ID3'tür. Bu yöntemde ve C4.5'te karar ağacında dallanmanın

hangi özniteliğe göre yapılacağını belirlemek için entropi kavramı kullanılmaktadır[67].

ID3 algoritmasının adımları şu şekilde verilebilir:

1. Sınıf niteliği için entropi hesaplanır. Bunun için denklem (2.41) kullanılır.

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{2.41}$$

2. Sınıf kalitesi dikkate alınarak her bir özellik için ağırlıklı ortalamalar hesaplanır. Bu amaçla aşağıdaki (2.42) eşitliği kullanılır.

$$H(X,S) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} H(S_i) \tag{2.42}$$

3. Her özellik için bilgi kazancı aşağıdaki (2.43) kullanılarak hesaplanır:

$$Bilgi\ Kazancı\ (X,S) = H(S) - H(X,S) \tag{2.43}$$

4. En büyük bilgi kazancını sağlayan özellik seçilir. Bu öznitelik, sınıflandırma ağacının bir düğümüdür.
5. Bir önceki düğümden ayrılan her dal için veri seti yeniden düzenlenir ve 1. adıma geri dönülür. Her adımda veri seti satırları azaltılır. İşlem, işlenecek başka satır kalmayıncaya kadar devam eder.

C4.5 Algoritması

Quinlan tarafından ID3 algoritmasının genişletilmiş bir versiyonu olarak geliştirilmiştir. Bu algoritma, ID3 algoritmasının adımlarını birebir takip etmektedir. ID3'ten farklı olarak, sayısal değerlerle özniteliklerin (niteliklerin) karar ağaçlarını oluşturma olanağı sağlamıştır. Ek olarak, bilinmeyen öznitelik değerlerine sahip örnek kümeler için, karar ağacının ne şekilde oluşturulacağına dair bir yöntem sağlar[69].

Sayısal Değere Sahip Nitelikler

C4.5 algoritması araştırmacılara sayısal özellik değerlerini kategorik değerlere dönüştürmenin bir yolunu sunar. Bunun için öznitelik değerleri bir t eşiğine göre iki parçaya bölünür ve her bölüm için bilgi kazancı hesaplanır. İşlemler sonucunda en büyük kazancı sağlayacak olan bölünmeye ulaşılması hedeflenir[67].

Bilinmeyen Öznitelik Değerleri

Veritabanındaki bazı bilgiler bazen kaybolabilir. Veriler toplanırken bazı değerler kaydedilmemiş veya eksik kaydedilmiş olabilir. Kaybolan değerlerin neden olduğu sorunları önlemek için aşağıdaki yollardan biri izlenebilir[67]:

a) Eksik verilerle ilgili tüm değerler veri tabanından silinir.

b) Kayıp verilerle çalışmak için yeni bir algoritma kullanılır.

İlk çözümün basit olmasına rağmen; kayıp veriler büyükse, bu çözüm kabul edilemez. Böyle bir durumda ikinci seçeneği tercih etmek daha akıllıca olacaktır.

C4.5'te eksik verileri olan numuneler için kazanç kriterini hesaplamak için bir düzeltme faktörü kullanılır. H(X) ve H(X, S) değerlerinin daha önce belirtildiği gibi hesaplandığını varsayalım. Bu hesaplamalarda sadece özellikleri bilinen numuneler dikkate alınır. Yani eksik değerleri içeren satırlar eğitim setinden çıkarılmıştır. Bu durumda kazanç kriteri F faktörü kullanılarak düzeltilir[67]:

$$F = \frac{Veritabanında\ bilinen\ değere\ sahip\ örnek\ sayısı}{Veritabanında\ tüm\ değerlere\ sahip\ örnek\ sayısı} \quad (2.44)$$

Yeni kazanç kriteri şöyle olacaktır:

$$Kazanç\ (X) = F(H(S) - H(X,S)) \quad (2.45)$$

Benzer şekilde değeri bilinmeyen örnekler ek bir gruba bölünerek $H(P_{X,S})$ bilgi bölme değeri hesaplanabilir. X testinin bu test veri setini $n + 1$ alt kümeye böldüğü varsayılırsa, $H(P_{X,S})$ değeri hesaplanır. Bu, kazanç oranının değerini doğrudan etkiler.

Budama Karar Ağaçları

Karar ağaçları karmaşık bir görünüme sahip olabilir. Bir karar ağacı, bir alt ağaç atılarak bir yaprak ile değiştirilebilir. Bu işleme "karar ağacını budama" denir. Budama, "tahmin hata oranını" azaltmayı ve sınıflandırma modelinin kalitesini iyileştirmeyi amaçlar[70].

Ancak hata oranını hesaplamak kolay değildir. Tahmini hata oranını belirlemek için yeni bir dizi ek test örneği kullanılabilir. Bu teknik, önceden var olan örnekleri eşit büyüklükte bloklara böler ve her blok için o bloğu oluşturan tüm örneklerden bir ağaç oluşturulur, ardından bu ağaç verilen örnek bloğu ile test edilir. Budama

işleminin temeli, sınıflandırma doğruluğuna katkı sağlamayan alt ağaçların çıkarılmasıdır. Böylece daha az karmaşık bir ağaç üretilir.

Özyinelemeli bölme yönteminde düzenleme yapmanın iki yolu vardır:

a) Bazı durumlarda örneklem setinin daha fazla bölünmemesine karar verilir. Bölmenin sona ermesi olarak χ^2 gibi istatistiksel testlere dayanır. Bölmeden önce ve sonra gözle görülür bir fark yoksa söz konusu düğüm yaprak olarak gösterilir. Yani ağaç dallanmadan karar verilir. Bu tekniğe "ön budama" denmesinin nedeni budur.

b) Bazı ağaçlar seçilen bir "doğruluk ölçüsü" kullanılarak budanabilir. Bu yöntemde ağaç oluşturulduktan sonra budama yapılır. C4.5 sınıflandırma yöntemi bu tekniği kullanır.

C4.5'te budama

C4.5'te beklenen hata oranını tahmin etmek için özel bir yöntem kullanılır. Bu yönteme "kötümser budama" denir. Ağaçtaki her bir düğüm için U_{cf} üst güven sınırı, iki terimli dağılımların istatistiksel tabloları kullanılarak elde edilebilir[70].

Verilen düğüm U_{cf}, S_i ve E'nin bir fonksiyonudur. C4.5 algoritması %25 güven sınırını kullanır ve verilen her düğümdeki S_i'yi $U_{\%25}\left(\frac{|S_i|}{E}\right)$düğümünün güven aralığıyla karşılaştırır. Her yaprağın üzerindeki ağırlıklar toplam vaka sayısıdır. Alt ağaçtaki kök düğümün beklenen hatası, yapraklardaki toplam ağırlığın $U_{\%25}$'inden azsa (alt ağacın beklenen hatası), alt ağaç yok edilir. Yerine bir kök düğüm yerleştirilir. Böylece budanan ağaçta yeni bir yaprak olarak görünür.

2.1.5 Destek Vektör Makineleri

Destek Vektör Makineleri (Support Vector Machines - SVM), ilk olarak Cortes ve Vapnik tarafından 1995 senesinde tanıtılmıştır. Özellikle de doğrusal olmayan veriler üzerinde sağladığı etkin ayırma yeteneği ile makine öğrenmesinde öne çıkan yöntemlerden biri hâline gelerek, birçok sınıflandırma problemlerine yüksek doğruluk sunduğu görülmüştür. SVM, örnekleri yüksek boyutlu bir özellik uzayına taşıyarak, bu uzayda optimal bir ayrım hiperdüzlemi bulmayı hedefler[71].

SVM, verilerin türüne bağlı olmak üzere, algoritmanın çalışması sırasında çekirdek (kernel) fonksiyonları da kullanılabilmektedir. Bu sayede hem doğrusal hem de doğrusal olmayan sınıflandırma işlemlerini gerçekleştirilebilmektedir. Eğer sınıflandırma işleminde, tam ayrıştırılabilir veriler kullanılırsa genellikle tüm veriler bir hiper düzlem ile sınıflandırılabilmektedir. Fakat, eğer tam ayrıştırılamayan veriler kullanılırsa, çoğunlukla aynı boyutta tek bir düzlem ile sınıflandırılamamaktadır[72]. Bu nedenle de farklı çekirdek fonksiyonları kullanılmaktadır.

Destek Vektör Makineleri'nin en önemli avantajı, sınıflandırma problemini *"kareli optimizasyon"* problemine dönüştürüp çözmesidir. Böylece problemin çözümüne ilişkin öğrenme aşamasında işlem sayısı azalmakta ve diğer teknik/algoritmalara göre daha hızlı çözüme ulaşılmaktadır[73]. Bu niteliğinden dolayı, özellikle büyük veri setlerinde avantaj sağlamaktadır. Buna ek olarak, optimizasyon temelli olduğundan sınıflandırma performansı, kullanışlı oluşu ve hesaplama karmaşıklığı açısından diğer tekniklere göre daha başarılı olduğu neticesine Nitze ve arkadaşları tarafından varıldığı söylenebilir[74].

Algoritmanın temelde işleyişine bakacak olursak; bir düzlem üzerine yerleştirilmiş noktaları ayırmak için bir doğru çizer. Bu doğrunun, farklı iki sınıfın tüm noktaları için maksimum uzaklıkta olmasını amaçlar ve buna ek olarak tek bir sınıf içinde bulunan farklı sınıftan gözlem sayısının en az olmasına dikkat eder. Karmaşık olsa bile, küçük ve orta ölçekteki veri setleri için uygundur.

Destek Vektör Makineleri'nin uygulanması sürecinde çekirdek fonksiyonu seçimi ve parametre optimizasyonu önemli rol oynamaktadır[75].

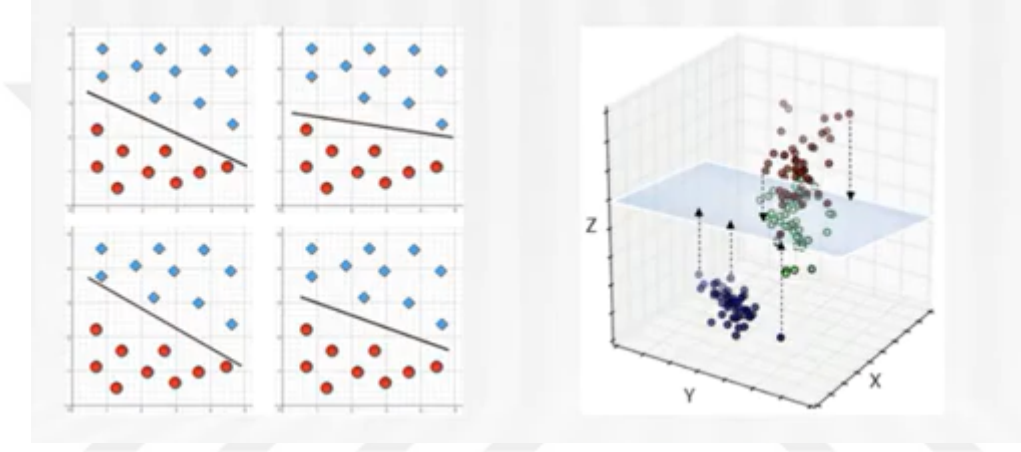
Destek Vektör Makinelerinin Matematiksel Temelleri

Destek Vektör Makineleri'nin (SVM) matematiksel temelleri, kalkülüs, vektör geometrisi ve kısıtlı optimizasyon teorilerine dayanmaktadır. Bu yöntem hem doğrusal hem de doğrusal olmayan sınıflandırma problemleri için uygulanabilir bir yapı sunar. Ayrıca veri kümesinin doğrusal olarak ayrılabilir olup olmaması, algoritmanın uygulama biçimini doğrudan etkilemektedir[52,71].

Veri noktaları, bir boyutta bir sayı doğrusu üzerinde; iki boyutta doğru ile; üç boyutta düzlem ile ve daha yüksek boyutlarda ise hiper düzlem (hyperplane) yardımıyla ayrılabilir. Bu nedenle SVM algoritmasının çalışma mantığını

kavrayabilmek için özellikle hiper düzlem kavramının temel aldığı vektör uzaylarının geometrisine aşinalık gereklidir[53,54].

Aşağıda gösterilen Şekil 2.19'da iki farklı sınıfı temsil eden veri kümeleri yer almaktadır. Sol tarafta iki boyutlu düzlem üzerinde kolaylıkla ayrılabilen sınıflar doğru ile ayrılmışken, sağ tarafta daha yüksek boyutlu bir özellik uzayında sınıflandırılması gereken veriler, ancak bir hiper düzlem aracılığıyla ayrılabilmektedir. Bu durum, özellikle çok boyutlu veri setleri üzerinde çalışan makine öğrenmesi algoritmaları için geometrik sezginin önemini ortaya koymaktadır[53].



Şekil 2.19 İki ve üç boyutlu veri kümeleri için sınıflandırma örneği (doğru-hiper düzlem örnekleri)

Doğrusal ayrılma durumu mükemmel doğrusal ayrılma ve belli bir hata oranıyla doğrusal ayrılma olarak iki ayrı kategoride incelenebilir.

Mükemmel Doğrusal Ayrılma (Doğrusal Ayrılabilme)

Regresyon modellerinde, bir doğrusal regresyon modeli aşağıda denklem (2.46)'deki gibi bir doğru denklemiyle ifade edilebilir[56].

$$y = h_{\Theta}(x) = \Theta_0 + \Theta_1(x) \tag{2.46}$$

Θ : Ağırlık değerleri,

x : Bağımsız değişkeni temsil etmek üzere.

Denklem (2.46)'da $x=x_1$ ve $y=x_2$ olarak yazılırsa ve sağ taraf sıfır olacak şekilde denklem düzenlenirse, denklem (2.47) elde edilmiş olur.

$$x_2 = \Theta_0 + \Theta_1 x_1 \rightarrow \Theta_1 x_1 - x_2 + \Theta_0 = 0 \quad (2.47)$$

x ve w vektörleri iki boyutlu vektörler olmak üzere, x=(x_1,x_2) ve w=(Θ_1,-1) vektörlerinin iç çarpımı:

$$\Theta_1 x_1 - x_2 = < w, x > \quad (2.48)$$

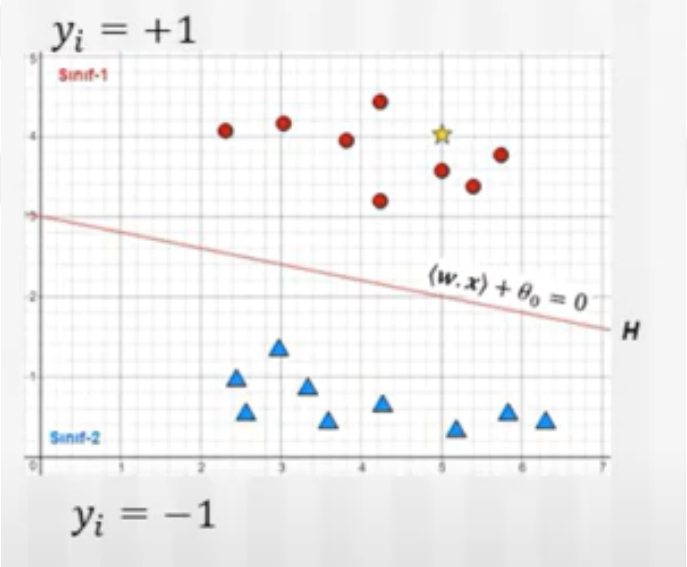
(2.48) şeklinde ifade edilebilir[53].

Burada Denklem (2.47)'de Denklem (2.48)'te bulunan ifade yerine konulursa,

$$< w, x > + \Theta_0 = 0 \quad (2.49)$$

(2.49) denklemi elde edilir[53,54].

Böylece doğru ile gösterilen doğrusal regresyon modeli kullanılarak bir hiper düzlemin matematiksel gösterimi elde edilmiştir.



Şekil 2.20 SVM ile sınıflandırma örneği

Yukarıda Şekil 2.20'de göründüğü gibi $y_i \in \{-1,1\}$ olmak üzere her bir x_i vektörünün bir y_i sınıf etiketine sahip olduğu ve bu iki sınıf etiketinin şekildeki gibi doğrusal olarak ayrılabildiği görülmüştür[53]. Burada;

- **H:** bir hiper düzlemdir.
- $\mathbf{y_i}$=+1 ve $\mathbf{y_i}$=-1 iki farklı sınıf etiketidir.

O halde H hiper düzleminin üzerinde kalan ve $\mathbf{y_i}$=**+1** sınıfına ait olan noktaların denklemi (2.50),

$$< w, x_i > +\Theta_0 \geq +1 \quad (2.50)$$

ve H hiper düzleminin altında kalan ve $\mathbf{y_i}$**=-1** sınıfına ait olan noktaların denklemi (2.51),

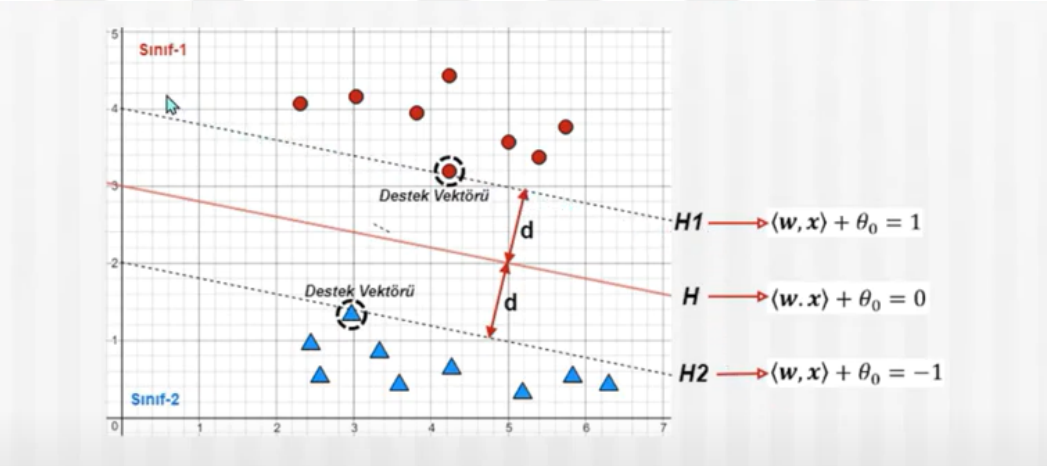
$$< w, x_i > +\Theta_0 \geq -1 \quad (2.51)$$

şeklinde ifade edilebilir[54].

Denklem (2.50) ve Denklem (2.51) tek bir denklem sistemi ile gösterilmek istenirse aşağıda (2.52) gibi gösterilebilir[54].

$$y_i(< w, x_i > +\Theta_0) \geq 1 \quad (2.52)$$

En Uygun Hiper Düzlemin Seçilmesi



Şekil 2.21 En uygun hiper düzlem

Yukarıda Şekil 2.21'de de görüldüğü gibi algoritma en uygun hiper düzlemi seçmek için iki koşulu önceler ve her iterasyonda bu iki koşulu sağlayan en iyi hiper düzlemi bulmak üzere hesaplamalarına devam eder[53].

Algoritmanın sağlamaya çalıştığı koşullar:

1) Her bir H_1 ve H_2 hiper düzlemleri arasındaki mesafe (2d, marjin) en büyük olmalıdır.
2) Sınıf 1 ve Sınıf 2'ye ait örnekler en iyi şekilde birbirinden ayrılabilmelidir.

Bu durumları sağlayan H hiper düzlemi en iyi hiper düzlem (optimal hyper plane) olarak ifade edilir[54].

Burada yukarıda tanımlanan Denklem (2.49), H optimal hiper düzlemini ifade ederken,

$$< w, x > +\Theta_0 = 0 \ , < w, x > +\Theta_0 - d = 0 \tag{2.53}$$

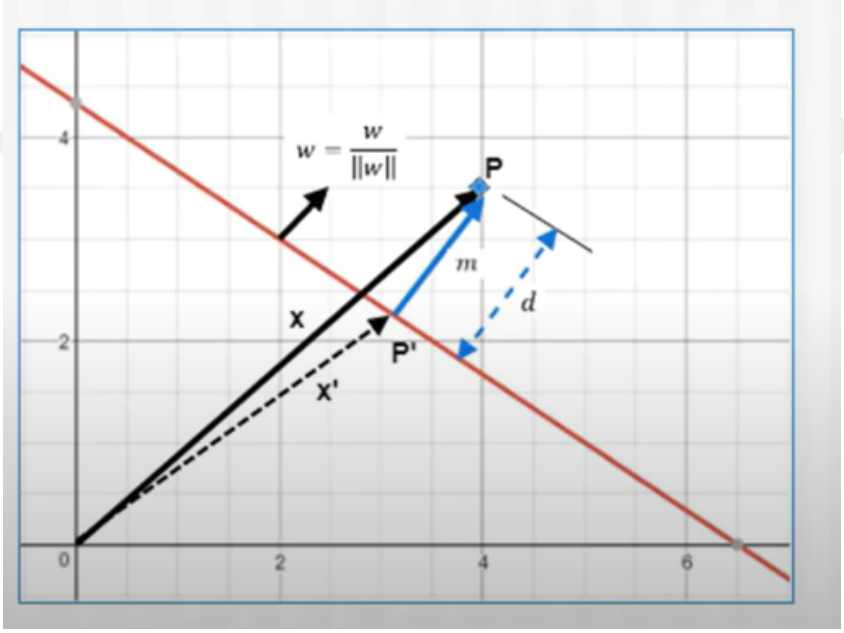
(2.53) denklemi H_1 hiper düzlemini ve (2.54) denklemi,

$$< w, x > +\Theta_0 + d = 0 \tag{2.54}$$

H_2 hiper düzlemini ifade etmektedir. Burada d (marjin) mesafesinin değeri hesaplama kolaylığı açısından 1 olarak alınabilir[53].

Bu şartlar altında H_1 ve H_2 hiper düzlemleri sırasıyla, $< w, x > +\Theta_0 = 1$ ve $< w, x > +\Theta_0 = -1$ olarak ifade edilebilir.

Marjinin geometrik karşılığı aşağıdaki Şekil 2.22'de verilmiştir. Marjinin uzunluğu şekilden de anlaşılacağı üzere, bir veri örneği ile hiper düzlem arasındaki Öklid uzaklığıdır[53,54].



Şekil 2.22 Marjinin vektörel gösterimi

x ve ***x'*** vektör ve w bir birim vektör olmak üzere; **P'** noktası, P noktasının hiper düzlem üzerine dik iz düşümüdür. ***m*** ise ***w*** birim vektörü ile aynı yöndedir ve normu ***d*** mesafesi kadardır. O halde,

$$m = d.\frac{w}{||w||} \tag{2.55}$$

Şeklinde (2.55) denklemi yazılabilir. Vektörlerin çıkarma özelliğinden de

$$x'=x-m \quad => \quad x' = x - d.\frac{w}{||w||} \tag{2.56}$$

(2.56) şeklinde ifade edilebilir.

P' noktası hiper düzlemin üzerinde olduğundan H hiper düzlem denklemini sağlayacaktır[53]. O halde,

$$< w, x' > +\Theta_0 = 0$$

$$< w, x - d.\frac{w}{||w||} > +\Theta_0 = 0 \quad =>$$

$$< w, x > - d.\frac{<w,w>}{||w||} + \Theta_0 = 0 \quad =>$$

$$< w, x > - d.\frac{||w||^2}{||w||} + \Theta_0 = 0 \quad =>$$

$$< w, x > - d.||w|| + \Theta_0 = 0 \quad =>$$

$$d = \frac{|< w, x > +\Theta_0|}{||w||} = \frac{1}{||w||} \tag{2.57}$$

$d = \frac{1}{||w||}$ olduğuna göre maksimize edilmek istenilen marjin değeri olan 2d değeri[54],

$$2d = \frac{2}{||w||} \tag{2.58}$$

(2.58) olacaktır. Böylece maksimizasyonun amaç fonksiyonu elde edilmiş olur.

Algoritmanın sağlamaya çalıştığı ikinci koşul ise maksimizasyonun hangi şart altında yapılacağını belirleyen kısıtı verecektir. Bu ise Denklem (2.52)'yi kullanarak ifade edilebilir Burada y_i değerlerinin değişimine göre Denklem (2.50)'nin veya Denklem (2.51)'in oluşacağı görülmektedir.

Marjinin En Büyüklenmesi

Yukarıda Denklem (2.57)'de elde edilen marjin fonksiyonuna bakıldığında marjinin en büyüklenmesi için paydadaki w birim vektörü normunun en küçüklenmesi yani minimize edilmesi gerekmektedir. İşlem kolaylığı ve

türevlenebilirliğin kolay olması amacıyla paydadaki ||w|| ifadesi yerine $\frac{1}{2}||w||^2$ yazılabilir[53,54].

Bu durum denklem (2.59),

$$2d = \frac{1}{2}||w||^2 \tag{2.59}$$

şeklinde ifade edilebilir. Bu da vektörlerin özelliklerinden bilindiği üzere (2.60) olarak ifade edilebilir.

$$2d = \frac{1}{2} < w, w > \tag{2.60}$$

Böylece artık algoritma tarafından çözülmesi beklenen problem karesel bir minimizasyon problemidir.

Amaç fonksiyonu: $\frac{1}{2} < w, w >$

Kısıt fonksiyonu: $y_i(< w, x_i > +\Theta_0) \geq 1$, $\forall_i$

Karesel programlamada kısıtların doğrusal olması gerekmektedir.

Yukarıdaki minimizasyon problem Lagrange Çarpanları yardımıyla çözülebilir. Lagrange çarpanları en küçükleme tipindeki problemi dual probleme dönüştürerek problemin daha kolay çözülmesine imkan verir. Problemin çözümünde kullanılan Lagrange fonksiyonu Eşitlik (2.61)'de verilmiştir[72].

$$L_p(w, \Theta_0, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i [\, y_i(< w, x_i > +\Theta_0)] + \sum_{i=1}^{n} a_i \tag{2.61}$$

Bu eşitliklerde $\alpha i \geq 0$ olmak üzere, her bir αi Lagrange çarpanı olarak ifade edilir. L_p, w ağırlık vektörü ve Θ_0 sabitini en küçükleyen ve negatif olmayan dual değişken α_i'yi en büyükleyen bir fonksiyondur[76].

Lagrange fonksiyonunun w ve Θ_0'ya göre kısmi türevler alınarak Eşitlik (2.62) ve (2.63)'teki Karush-Kuhn-Tucker (KKT) koşulları elde edilir[77].

$$\frac{\partial L_p(w,\Theta_0,\alpha)}{\partial w} = w - \sum_{i=1}^{n} y_i \alpha_i x_i = 0 \;\; => w = \sum_{i=1}^{n} y_i \alpha_i x_i \tag{2.62}$$

$$\frac{\partial L_p(w,\Theta_0,\alpha)}{\partial \Theta_0} = 0 => \sum_{i=1}^{n} y_i \alpha_i = 0 \quad (2.63)$$

Elde edilen eşitlikler Lagrange fonksiyonunda ilgili yerlere koyularak problem en büyükleme tipindeki dual Lagrange problemine (L_D (α)) dönüşür. Denklem (2.64) olarak aşağıda ifade edilmiştir.

$$max\, L_D(\alpha) = L_D(w, \Theta_0, \alpha) =$$

$$= \sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j < x_i, x_j > \quad (2.64\ a)$$

şu kısıtlar altında,

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \text{ ve } \forall \alpha_i \geq 0 \quad (2.64\ b)$$

Optimal hiper düzlemin belirlenmesi için (2.64 a) ve (2.64 b)'den oluşan model (2.64) çözülerek, dual Lagrange L_D (α)'yı maksimum yapan αi değerleri elde edilir. αi Lagrange çarpanlarından sıfırdan büyük değer alan eğitim örnekler "destek vektörleri" olarak ifade edilir. Optimal ayırma hiper düzlemi, sıfırdan büyük değer alan bu Lagrange çarpanları ile belirlenir [72].

αi 'nin çözümü ile optimal hiper düzlemin Eşitlik (2.65) ve (2.66)'da verilen ağırlık vektörü w ve Θ_0 sabit parametreleri belirlenir.

$$w^* = \sum_{i=1}^{n} y_i \alpha_i x_i \quad (2.65)$$

$$\Theta_0{}^* = -\frac{1}{2} < w, x_s > \quad (2.66)$$

Sonuç olarak elde edilen hiper düzeleme bağlı olarak sınıflandırıcı Eşitlik (2.67)'de verilmiştir.

$$f(x) = sgn\big((w^*, x_i) + \Theta_0{}^*\big) = sign(\sum_{i=1}^{n} y_i \alpha_i < x_i, x_j >) \quad (2.67)$$

Ayrıca Eşitlik (2.67)'deki sınıflandırıcıya alternatif olarak, algoritmanın gerçek değerler üretmesinden dolayı Eşitlik (2.68)'de verilen sınıflandırıcının kullanımı daha uygun olmaktadır.

$$f(x) = h\left((w^*, x_i) + \Theta_0{}^*\right) => h(z) = \begin{cases} -1\,, & z < -1 \\ z\,, & -1 \leq z \leq 1 \\ 1, & z > 1 \end{cases} \quad (2.68)$$

Hata Payı ile Doğrusal Olarak Ayrılma

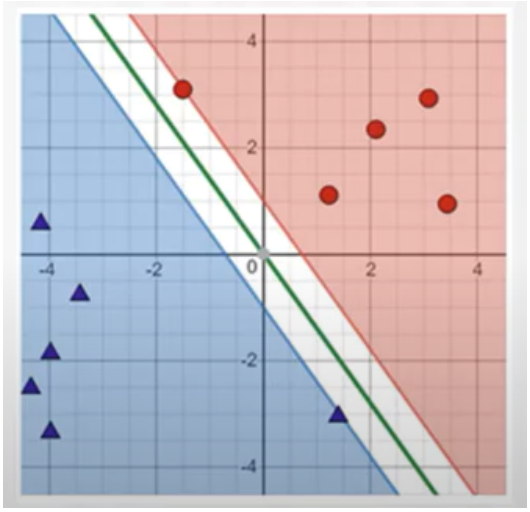
Veri setinin belirli bir hata ile doğrusal olarak ayrılma durumu, yani buna mükemmel ayrılamama da denir; veri setinin gürültülü veri içermesi, çok boyutlu olması veya karmaşık yapısından kaynaklanmaktadır[75,78]. Belirli bir hata ile doğrusal ayrılma durumunda iki sınıflı veri setini ayırmak için gevşek sınır (soft margin) yaklaşımı kullanılmaktadır[75].

Esnek Marjin

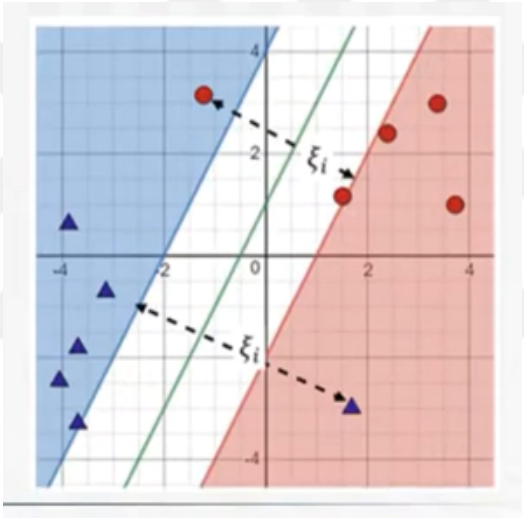
Gürültülü bir veri seti içerisinde Sınıf 1'e ait olması gereken veriler Sınıf 2'nin içerisinde bulunabilirken, Sınıf 2'ye ait olması gereken bazı veriler de Sınıf 1 içerisinde bulunabilir. Yani aslında veri setlerini düzgün doğrularla tam olarak ayıramıyor olduğumuzu düşünmek mümkündür. Bu tip durumlarda gevşek değişkenlerden (slack variable) faydalanılabilir[71]. Bu gevşek değişkenler marjini büyütmek amacıyla kullanılabilecek hata payı olarak ifade edilebilir. Marjin ne kadar büyükse SVM algoritması o kadar optimal bir sonuca ulaşabilecek anlamına gelir. Bu sebeple hata oranını da minimumda tutmaya çalışacak şekilde marjini büyütmek, daha iyi sonuç alabileceğimiz bir optimizasyon problem için önemlidir[71].

Gürültü içermeyen bir veri seti, yukarıda da anlatıldığı gibi, düzgün ayrışabilen doğrusal bir probleme örnektir.

Aşağıda Şekil 2.23'te "keskin marjin" örneğine, Şekil 2.24'te ise "esnek marjin" örneğine yer verilmiştir.



Şekil 2.23 Keskin marjin örneği



Şekil 2.24 Esnek marjin örneği

Yukarıda anlatılan amaç fonksiyonu ve kısıt fonksiyonlarına ufak eklemeler yaparak belli bir hata payı ile optimizasyon yapabilen bir SVM modeli elde edebilmek mümkün.

Hiper parametre düzlemine ξ_i gevşek değişkeni uygulanır[72].

$$y_i(< w, x_i > +\Theta_0) + \xi_i \geq 1 \text{ , } \xi_i \geq 0 \text{ ,} \forall_i \qquad (2.69)$$

Marjin Esnetme Kuralı

Esnek marjin kullanılarak yapılan bir sınıflandırmada, marjininn çok artması hatalı sınıflandırılan örneklerin çoğalmasına neden olacaktır. Bu sebeple hatalı sınıflandırılan örnek sayısı da minimum tutulacak şekilde marjin esnetilmelidir[71].

Burada yeni minimizasyon fonksiyonumuz:

$$\frac{1}{2} < w, w > + c \sum_{i=1}^{n} \xi_i \qquad (2.70)$$

Kısıt fonksiyonumuz :

$$y_i(< w, x_i > + \Theta_0) - 1 + \xi_i \geq 0 \,, \quad \xi_i \geq 0 \,, \quad \forall_i : i=1,..,n \qquad (2.71)$$

Burada dikkat edilmesi gereken minimizasyon fonksiyonunda eklenen esnek değişkenlerin toplamının minimum olacağıdır. Negatif esnek değişkenlerin problem oluşturmaması için ikinci bir kısıt eklenerek onların hesaba katılması engellenmiştir. Ancak bu artık problemin doğrusal olarak çözülebilmesini imkansız kılar. Artık elimizde kısıtlı doğrusal olmayan bir optimizasyon problem bulunmaktadır[71].

Burada eğer kısıtlarda "eşitlik" var ise çözüm yoluna giderken "lagrange çarpanları" kullanılır. Eğer kısıtlarda "eşitsizlik" durumu mevcut ise "lagrange fonksiyonu" oluşturularak "Karush-Kuhn-Tucker (KKT)" koşulları üzerinden çözüme ulaşılır[77].

Ayrıca buradaki c parametresinin nasıl hesaplanacağına dikkat etmek gerekir. "c" burada esnek marjin için bir hiper parametredir ve eğer küçük bir "c" parametresi seçilirse yanlış sınıflandırılmış örneklere tolerans gösterilerek çok daha geniş bir marjin elde etme durumu ortaya çıkar. "c" çok büyük seçilirse de dar bir marjin elde edileceğinden sonuç yine istenilen gibi olmayacaktır[76,77].

Burada en uygun "c" değeri araştırılırken çapraz doğrulama ya da ızgara arama(grid search) gibi yöntemler kullanılabilir.

Esnek Marjin için Dual Çözüm

Bir doğrusal programlama problemi kendisi ile ilişkili başka bir doğrusal programlama problemine dönüştürülebilir. Bu tarz ikili yapılarda dönüşümden önceki problemin kendisine Primal, problemin dönüştürülmüş haline ise dual adı verilir.

Eğer primal bir amaç fonksiyonu maximize edilecekse bunun dual hali olan amaç fonksiyonu minimize edilmelidir. Tersi durumda ise dual maximize edilmelidir.

Daha önceki problem dikkate alınacak olursa, elimizde bulunan problem marjini en büyük yapma problem idi. Yani maximizasyon yapmak gerekiyordu. O halde bu

problemin dual hali için Denklem (2.70) ve (2.71)'de sırasıyla, açıklanan amaç ve kısıt fonksiyonları, minimizasyon yapmak üzere yazılabilir[53,54].

Bunların kullanılmasıyla oluşturulan Lagrange Fonksiyonu ise:

$$L_p(w,\Theta_0,\xi;\alpha,\beta) = \frac{1}{2} < w,w > +c\sum_{i=1}^{n}\xi_i$$
$$-\sum_{i=1}^{n}\alpha_i[\,y_i(< w,x_i > +\Theta_0) - 1 + \xi_i] - \sum_{i=1}^{n}\beta_i\xi_i \tag{2.72}$$

α ve β çarpanları ayarlanarak, w, Θ_0 ve ξ parametrelerini göre Lagrange fonksiyonunun en küçüklenmesi ile problemin dual çözümü şöyle yazılabilir[76]:

$$\max_{\substack{\alpha,\beta:\\ \alpha_i\geq 0,\beta_i\geq 0}} \left(\min_{\substack{w,\Theta_0,\xi:\\ \xi_i\geq 0}} L_p(w,\Theta_0,\xi;\alpha,\beta)\right) \tag{2.73}$$

Lagrange fonksiyonunun minimize edilmesi için w, Θ_0, ξ parametrelerini göre kısmi türevlerinin alınması gerekmektedir[76].

$$\frac{\partial L_p(w,\Theta_0,\xi;\alpha,\beta)}{\partial w} = w - \sum_{i=1}^{n} y_i\alpha_i x_i = 0 \;\; => w = \sum_{i=1}^{n} y_i\alpha_i x_i \tag{2.74}$$

$$\frac{\partial L_p(w,\Theta_0,\xi;\alpha,\beta)}{\partial \Theta_0} = 0 => \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{2.75}$$

$$\frac{\partial L_p(w,\Theta_0,\xi;\alpha,\beta)}{\partial \xi} = 0 \leq \alpha_i \leq c \tag{2.76}$$

Bu değerler denklem 2.26'da yerlerine yazılırsa;

$$\begin{aligned} L_p(w,\Theta_0,\xi;\alpha,\beta) = \frac{1}{2} < \sum_{i=1}^{n} y_i\alpha_i x_i\,, \sum_{j=1}^{n} y_j\alpha_j x_j \\ > +c\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[\,y_i\left(< \sum_{i=1}^{n} y_i\alpha_i x_i\,, x_i > +\Theta_0\right) - 1 \\ + \xi_i] - \sum_{i=1}^{n}\beta_i\xi_i \end{aligned} \tag{2.77}$$

Denklem (2.77)'de gerekli düzenlemeler yapıldıktan sonra denklem (2.78) elde edilir[76].

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j < x_i, x_j > \quad (2.78)$$

Dual en iyileme fonksiyonu ise artık

Amaç fonksiyonu: $L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j < x_i, x_j >$

Kısıtlar ise: $0 \leq \alpha_i \leq c$, i=1

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

şeklindedir. Dual en iyileme problem için üretilen çözüm KKT koşullarını sağlarsa, bu çözüm için en uygun çözüm denilebilir. Bu problem için KKT koşulları nelerdir, Eşitlik (2.71), (2.74), (2.75), (2.76)'teki tüm koşullardır. Bunlara ek;

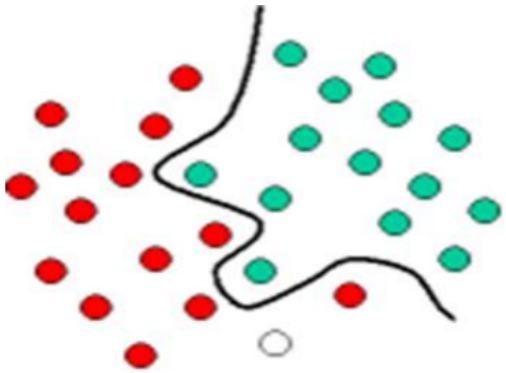
$\alpha_i \geq 0, \quad \beta_i \geq 0, \quad \beta_i \xi_i = 0\,, \quad \alpha_i[y_i(< w, x_i > +\Theta_0) - 1 + \xi_i] = 0$

koşulları sayılabilir[77].

Doğrusal Olmayan Sınıflandırmada Destek Vektör Makineleri

Gerçek yaşam problemlerinde çoğunlukla, bir veri setinin hiper düzlem ile doğrusal olarak ayrılması mümkün değildir. Doğrusal olmayan SVM de tam bu noktalarda kullanılmaktadır. Veri setinin doğrusal bir fonksiyonla tam veya belirli bir gevşek değişken kullanılarak ve hata payı kabul edilerek ayrılamaması durumunda kullanılan algoritmalardır. Burada sınıfları ayırma işlemi, ayırma düzleminin tahmin edilmesiyle mümkün olabilir denilebilir. Ancak bu işlem de oldukça zordur[75].

Aşağıda Resim 2.25'te doğrusal olmayan destek vektörlerinin modellemesi gösterilmiştir.



Şekil 2.25 Doğrusal olmayan destek vektör makinelerinin gösterimi

Buradan da anlaşılacağı üzere iki sınıflı bir problemde sınıfları doğrusal olarak ya da belli bir hata payı kabul edilerek ayırabilecek herhangi lineer bir düzlem çizilememektedir. Bu gibi durumlarda SVM kernel fonksiyonlarına başvurur. Bu amaç için literatürde kullanılan farklı tipte çekirdek fonksiyonları bulunmaktadır[71,72]. Bunlardan bazıları;

• Doğrusal (K(x,z)=<x,z>),

• Polinom (K(x,z) =$(< x, z > +1)^d$),

• Gaussian Kernel (RBF Kernel) (K(x,z) = $e^{-\frac{||x-z||^2}{2\sigma^2}}$),

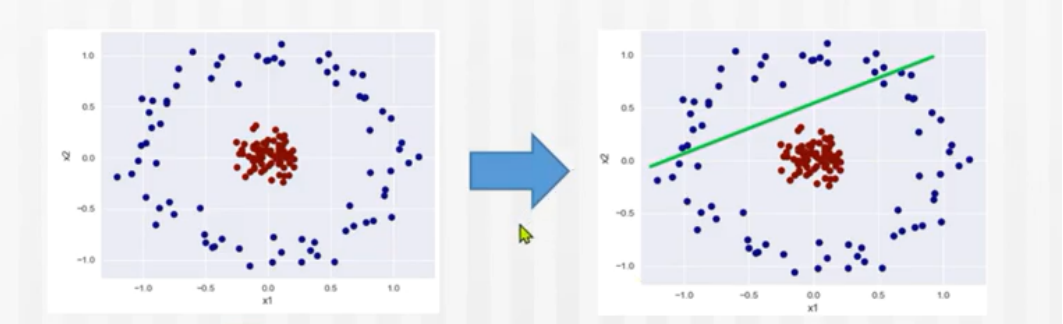
• Laplace,

• Anova,

• Sigmoid,

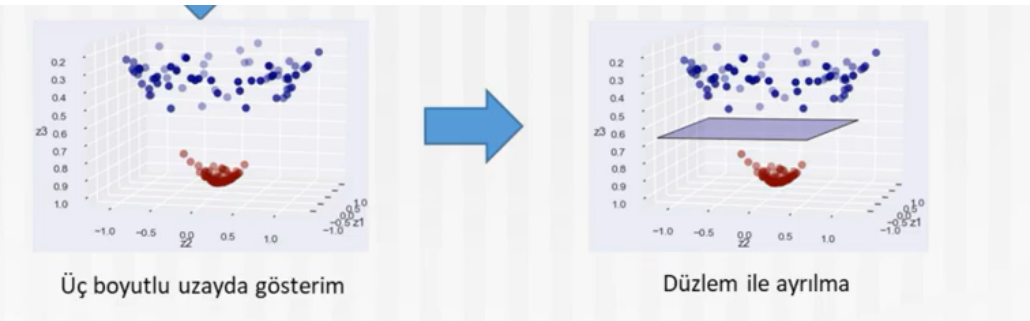
• Bessel çekirdek fonksiyonlarıdır [72].

Aşağıda Şekil 2.26'da doğrusal olarak ayrılamayan veri kümesi bulunmaktadır.



Şekil 2.26 Doğrusal olarak ayrılamayan bir veri kümesi

Ancak bu şekli 3 boyutlu uzayda gösterirsek, bir düzlem vasıtasıyla bu şekildeki sınıfları birbirlerinden ayırmak mümkün olabilir. Aşağıda Şekil 2.27'deki gibi, üç boyutlu bir uzayda bir düzlemle ayrışabilir.



Şekil 2.27 Düzlem ile ayırma gösterimi

İki boyutlu uzaylardan yüksek boyutlu uzaylara eşleme(mapping) işlemi sırasında öznitelik sayısı arttıkça yüksek işlem yükü gerektiren matematiksel ifadelerin kullanılması gerekmektedir. Bu da modelin hesaplamalarında zorluk yaratan bir faktör olacaktır. Bu zorluğun aşılması noktasında çekirdek(kernel) fonksiyonları devreye girer[72,76].

Çekirdek fonksiyonları genel itibariyle herhangi bir doğrusal sınıflandırma probleminde karşılaşılan, $< x_i, x_j >$ iç çarpım ifadesinin yerine yazılabilecek ve iki boyutlu bir problem daha yüksek boyutlara taşıyormuş gibi davranıp, aslında normun karesini hesaplamak yoluyla, algoritmayı fazla işlem yükünden kurtaracak bir fonksiyondur.

$K(x_i, x_j) = (< x_i, x_j >)^2$ şeklinde genel bir gösterimi vardır. Bu gösterim $x, z \in R^2$ olmak üzere, $x=(x_1, x_2)$ ve $z=(z_1, z_2)$ veri setindeki girdi uzayına ait vektörlerdir. $a,b \in R^3$ olmak üzere, x ve z vektörlerinin φ dönüşümü sonrasında yüksek boyutlu öznitelik uzayında a ve b vektörleri ile temsil edildiği varsayılırsa;

$$(x_1, x_2) \xrightarrow{\varphi} (a_1, a_2, a_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \quad , \quad (z_1, z_2) \xrightarrow{\varphi} (b_1, b_2, b_3) = (z_1^2, \sqrt{2}z_1z_2, z_2^2) \tag{2.79}$$

Φ dönüşümü sonrasında öznitelik uzayında oluşan yeni koordinatlı vektörlerin iç çarpımları aşağıdaki gibi yazılır.

$$(< a, b >) =< \varphi(x), \varphi(z) > \Rightarrow$$

$$=< x_1^2, \sqrt{2}x_1x_2, x_2^2, z_1^2, \sqrt{2}z_1z_2, z_2^2 >$$

$$= x_1^2 z_1^2 + 2x_1 x_2 z_1 z_2 + x_2^2 z_2^2$$

$$= (x_1 z_1 + x_2 z_2)^2$$

$$= (< \mathrm{x}, \mathrm{z} >)^2 \qquad (2.80)$$

Buradan anlaşılıyor ki dönüşüm gerçekleştirmeye gerek duyulmadan, sadece girdi uzayındaki vektörlerin iç çarpımlarının karesini hesaplayarak dönüşümden alınacak sonuçlar hesaplanabilmektedir. Öznitelik uzayına taşınan vektörlerin iç çarpımlarını ifade eden fonksiyona da **çekirdek fonksiyonu** denir. Bu dönüşüme gerek duyulmadan yapılan iç çarpımlar yoluyla sonuç hesaplama yöntemine de bazı kaynaklarda ***çekirdek hilesi*** adı verilmektedir[76].

Yukarıda bahsedildiği gibi çekirdek fonksiyonları, herhangi bir doğrusal sınıflandırma probleminde karşımıza çıkan, , $< x_{i,} x_j >$ iç çarpım ifadesinin yerine yazılabilir. Örneğin bir önceki esnek marjinin kullanıldığı en iyileme problemindeki L(α) amaç fonksiyonunda iç çarpım $< x_{i,} x_j >$

Ifadesi yerine $K(x_{i,} x_j)$ fonksiyonu yazılırsa yeni amaç fonksiyonu;

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j K(x_{i,} x_j) \qquad (2.81)$$

Kısıtlar ise: $0 \leq \alpha_i \leq c$, i=1

$\sum_{i=1}^{n} \alpha_i y_i = 0$ şeklinde yeniden tanımlanabilir. Böylece doğrusal olarak sınıflandırılamayan öznitelik vektörlerinin yüksek boyutlu bir uzaya taşınarak elde edilen çekirdek fonksiyonu yardımıyla sınıflandırılması da sağlanmış olmaktadır[71,76].

Mercer Koşulu

"Her fonksiyon çekirdek fonksiyonu olabilir mi?" sorusunun cevabını arar.

Bir fonksiyonun çekirdek fonksiyonu olabilmesi için Mercer koşulları adı verilen matematiksel koşulları sağlaması gerekmektedir.

Bu koşullar; fonksiyonun sürekli olması, simetrik olması ve pozitif yarı tanımlı olması şeklindedir[76].

Sonuç olarak, doğrusal olmayan SVM için özellik uzayında tanımlı ayırma hiper düzlemine bağlı olarak sınıflandırıcı karar fonksiyonu Denklem (2.82) ile ifade edilebileceği gibi,

$$f(x) = sign\left((w^*, x_i) + \Theta_0{}^*\right) = sign(\sum_{i=1}^{n} y_i \alpha_i < \varphi(x)_i, \varphi(x_j) >) \qquad (2.82)$$

ile genelleştiriledebilir[75].

2.1.6 Rastgele Orman (Random Forest)

"Random Forest" algoritması, Leo Breiman ve Adele Cutler tarafından geliştirilmiş, 2001 yılında Breiman'ın yayımladığı "Random Forests" makalesi[8] ile tanıtılmıştır. Algoritma temelde, birden fazla karar ağacı (decision tree) kullanan bir topluluk (ensemble) öğrenme yöntemidir. Temellerinin, 1990'lı yıllarda Breiman ve arkadaşlarının Bagging (Bootstrap Aggregating) yöntemi üzerinde çalışmaları sırasında atılmıştır.

Karar ağaçlarının tek başlarına aşırı öğrenmeye (overfitting) yatkın olabilmesi durumuna çalışmalar sırasında rastlanılmaktadır. Diğer ensemble algoritmaları gibi Random Forest'te de birçok ağaçtan oluşan topluluk yapısı bu riski azaltır[8]. Breiman ve arkadaşları bu durumu "Güçlü Sayılar Yasası"na dayandırarak açıklamaktadır. Güçlü Sayılar Yasası, bir dizi rastgele değişkenin ortalamasının, yeterince büyük bir örneklem büyüklüğüyle, bu değişkenlerin beklenen değerine (yani ortalamasına) yakınsamasını garanti eder. RF algoritmasında, birçok karar ağacı (her biri rastgele olarak farklı özelliklerle eğitilmiş) bir araya getirilerek bir sınıflandırıcı oluşturulur. Bu durumda, her ağacın tahmini rastgele bir değişken olarak düşünülebilir. Ayrıca RF'in genelleme yeteneği karar ağaçlarına göre daha yüksektir. Bunun sebebi çok sayıda karar ağacının tahminde söz sahibi olmasıdır. Bu durum, modelin daha dengeli ve genelleme yeteneğinin yüksek olmasını sağlar[8,54].

Modelin açıklanabilirliğini artıran önemli bir özellik olan, değişken önem dereceleri (feature importance) doğrudan RF modelinden elde edilebilir. Bu da, karmaşık ağaçlı modeller için tahminde neyin daha çok önemsendiğini gösterme konusunda anlamlı sonuçları gözler önüne serer[79].

"Random Forest"ın Temel İlkeleri

"Random Forest", temelinde Bagging yöntemine dayanır. Bu yöntemde, eğitim verisinden farklı alt kümeler (bootstrap örneklemeleri) alınarak her biri için ayrı bir model oluşturulur. Bu modellerin sonuçları birleştirilerek nihai tahmin yapılır[8].

Her bir karar ağacı, farklı bir bootstrap örneklemesi kullanılarak eğitilir. Ayrıca, her düğümde rastgele bir alt özellik kümesi seçilir ve bu özellikler arasından en iyi ayrımı yapan özellik seçilir. Bu, ağaçlar arasındaki korelasyonu azaltır ve modelin genelleme yeteneğini artırır[8].

Tahminler, ağaçların oylarına göre yapılır. Oylama kısmında literatürde birçok "Voting" yöntemi verilmektedir. Ancak "Random Forest"ta regresyon problemlerinde ortalama alınırken, sınıflandırma problemlerinde en fazla oy alan sınıf seçilir. En fazla oy alan sınıfın seçilmesi durumuna 'demokrasi usulü' oylama da denilmektedir[8,54].

"Random Forest"ın Matematiksel Temelleri

$h_1(\mathbf{x}),h_2(\mathbf{x}), \ldots ,h_K(\mathbf{x})$ sınıflandırıcılardan oluşan bir topluluk verildiğinde ve Y, X rastgele vektörünün dağılımından rastgele çekilen eğitim seti ile kenar fonksiyonu (marjin) aşağıdaki gibi tanımlanabilir[8].

$$mg(X,Y) = av_k\, I(h_k(X)=Y) - \max_{j\neq Y} av_k\, I(h_k(\mathbf{X}) = j) \qquad (2.83)$$

Burada I(•) gösterge fonksiyonudur. Marjin, X ve Y'de sağ sınıfa verilen ortalama oy sayısının, diğer sınıflara ait ortalama oy sayısını ne ölçüde aştığını ölçer. Marjin ne kadar büyük olursa, sınıflandırmaya olan güven de o kadar artar. Genelleme hatası şu şekilde verilir[8]:

$$PE^* = P_{\mathbf{X},Y}(mg(\mathbf{X},Y) < 0) \qquad (2.84)$$

$\mathbf{X}$, Y alt simgeleri olasılığın $\mathbf{X}$, Y uzayının üzerinde olduğunu gösterir.

Rastgele ormanlarda, $h_k(\mathbf{X}) = h(\mathbf{X},\Theta_k)$. Çok sayıda ağaç için "Büyük Sayıların Gücü Yasası"ndan (Strong Law of Large Numbers) ve ağaç yapısından şu sonuç çıkar:

Teorem 2.1 *Ağaç sayısı arttıkça, hemen hemen tüm diziler için* Θ_1 *, ...PE * şuna yakınsar[8]:*

$$P_{X,Y}\left(P_{\Theta}\left(h(X,\Theta)=Y\right)-\max_{j\neq Y}P_{\Theta}(h(\mathbf{X},\Theta)=j)\ <\ 0\right) \tag{2.85}$$

Bu sonuç, daha fazla ağaç eklendikçe rastgele ormanların neden aşırı uyum sağlamadığını, ancak genelleme hatasının sınırlayıcı bir değerini ürettiğini açıklamaktadır.

Güç ve Korelasyon

Rastgele ormanlar için, bireysel sınıflandırıcıların ne kadar doğru olduğunun ve aralarındaki bağımlılığın ölçüsü olan iki parametre açısından genelleme hatası için bir üst sınır türetilebilir. Bu ikisi arasındaki etkileşim, rastgele ormanların işleyişini anlamanın temelini oluşturur. Bu ilişkinin matematiksel temeli, Amit ve Geman'ın 1997 tarihli analizine dayanmaktadır[8,80].

Tanım 2.1 Rastgele bir orman için marjin fonksiyonu

$$mr(\mathbf{X},Y)=P_{\Theta}\left(h(X,\Theta)=Y\right)-\max_{j\neq Y}P_{\Theta}(h(\mathbf{X},\Theta)=j) \tag{2.86}$$

ve sınıflandırıcılar kümesinin gücü $\{h(x,\Theta)\}$ ve Θ her bir ağacın oluşturulmasında kullanılan rastgelelik kaynaklarını (bootstrap örneklemesi ve özellik seçimleri) temsil eder,

$$s=E_{\mathbf{X},Y}\,mr(\mathbf{X},Y). \tag{2.87}$$

$s \geq 0$ varsayarsak Chebychev'in eşitsizliği şunu verir:

$$PE^{*} \leq \mathrm{var}(mr)/s^2 \tag{2.88}$$

mr varyansı için daha açıklayıcı bir ifade aşağıda türetilmiştir:

$$\hat{j}(\mathbf{X},Y)=\ \arg\max_{j\neq Y}P_{\Theta}(h(\mathbf{X},\Theta)=j) \tag{2.89}$$

olsun.

Böylece,

$$mr(\mathbf{X},Y) = P_{\Theta}\,(h(\mathbf{X},\Theta)=Y) - P_{\Theta}\,(h(\mathbf{X},\Theta)=j\hat{}(\mathbf{X},Y)) \tag{2.90}$$

$$= E\Theta[I(h(\mathbf{X},\Theta)=Y) - I(h(\mathbf{X},\Theta)=j\hat{}(\mathbf{X},Y))] \tag{2.91}$$

olur.

Tanım 2.2 *Ham marjin fonksiyonu*

$$rmg(\Theta,\mathbf{X},Y) = I(h(\mathbf{X},\Theta)=Y) - I(h(\mathbf{X},\Theta)=j\hat{}(\mathbf{X},Y)) \tag{2.92}$$

Dolayısıyla, $mr(\mathbf{X},Y)$, $rmg(\Theta,\mathbf{X},Y)$*)'nin* Θ'ye göre beklenen değeridir. Tanım fonksiyonundaki (identity) herhangi bir işlev için;

$$[E_{\Theta} f(\Theta)]^2 = E_{\Theta,\Theta'} f(\Theta) f(\Theta') \tag{2.93}$$

Θ , Θ' 'nin aynı dağılımla bağımsız olduğu durumu tutar, bu da şunu gösterir:

$$mr(\mathbf{X},Y)^2 = E_{\Theta,\Theta'} rmg(\Theta,\mathbf{X},Y) rmg(\Theta',\mathbf{X},Y) \tag{2.94}$$

(2.94)'ü kullanmak şunu verir:

$$\text{var}(mr) = E_{\Theta,\Theta'}\,(\text{cov}_{\mathbf{X},Y}\, rmg(\Theta,\mathbf{X},Y) rmg(\Theta',\mathbf{X},Y))$$

$$= E_{\Theta,\Theta'}\,(\rho(\Theta,\Theta')\, sd(\Theta) sd(\Theta')) \tag{2.95}$$

burada ρ(Θ,Θ'), rmg(Θ,X,Y) ile rmg(Θ',X,Y) arasındaki korelasyon olup Θ,Θ' sabittir ve sd(Θ) rmg(Θ,X,Y) Θ'yi sabit tutuyor (**ρ**, iki farklı ağaç modelinin aynı veri noktası üzerindeki çıktıları, örneğin; doğru sınıfı tahmin edip etmemesi, arasındaki **istatistiksel bağımlılığı/korelasyonu** ölçer.).

Daha sonra,

$$\text{var}(mr) = \text{ρ}(E_{\Theta} sd(\Theta))^2$$

$$\leq \text{ρ} E_{\Theta}\,\text{var}(\Theta) \tag{2.96}$$

burada ρ-korelasyonun ortalama değeridir; şöyle ki;

$$\text{ρ} = E\Theta,\Theta'\,(\rho(\Theta,\Theta')\, sd(\Theta) sd(\Theta'))/\, E\Theta,\Theta'\,(sd(\Theta) sd(\Theta'))\,. \tag{2.97}$$

Yerine yazılırsa;

$$E_{\Theta}\,\mathrm{var}(\Theta) \leq E_{\Theta}(E_{\mathbf{X},Y} rmg(\Theta,\mathbf{X},Y))^2 - s^2 \qquad (2.98)$$
$$\leq 1 - s^2$$

(2.88), (2.96) ve (2.98)'in bir araya getirilmesi şunu verir:

Teorem 2.2 Genelleme hatası için bir üst sınır şu şekilde verilir[8]:

$$PE^* \leq \bar{\rho}(1-s^2)/s^2 \qquad (2.99)$$

Sınırın gevşek olması muhtemel olmasına rağmen, Vapnik-Chervonenkis (VC) tipi sınırların diğer sınıflandırıcı türleri için yaptığı gibi, rastgele ormanlar için de aynı anlamlı işlevi yerine getirir. Rastgele ormanlar için genelleme hatasında yer alan iki bileşenin, ormandaki bireysel sınıflandırıcıların gücü ve ham marjin fonksiyonları açısından aralarındaki korelasyon olduğunu göstermektedir. C/s2 oranı korelasyonun gücün karesine bölünmesiyle elde edilir. Rastgele ormanların işleyişini anlamada bu oran yararlı bir rehber olacaktır; ne kadar küçükse o kadar iyidir.

Tanım 2.4 Rastgele bir orman için c/s2 oranı şu şekilde tanımlanır:

$$c\,/\,s2 = \bar{\rho}\,/\,s^2 \qquad (2.100)$$

İki sınıf durumunda basitleştirmeler vardır. Marjin fonksiyonu,

$$mr(\mathbf{X},Y) = 2P_{\Theta}(h(\mathbf{X},\Theta)=Y) - 1 \qquad (2.101)$$

Gücün pozitif olması gerekliliği (bkz. (2.88)) tanınan zayıf öğrenme koşuluna benzer hale gelir EX,YPΘ (h(X, Θ) = Y) >. 5. Ham marjin fonksiyonu 2I(h(X,Θ)=Y) −1'dir ve $\bar{\rho}$ korelasyonu I(h(X,Θ)=Y) ve I(h(X,Θ')=Y) arasındadır. Özellikle Y için değerler +1 ve -1 olarak alınırsa, o zaman;

$$\bar{\rho} = E_{\Theta,\Theta'}\,[\rho(h(\cdot,\Theta),h(\cdot,\Theta'))] \qquad (2.102)$$

böylece $\bar{\rho}$ ormanın iki farklı üyesi arasındaki Θ, Θ' dağılımı üzerinden ortalaması alınan korelasyondur.

$\hat{j}(\mathbf{X}, Y)$'yi belirleyen orman olduğundan, kuvvet ölçüsü, tek tek ağaçlara olduğu kadar ormana da bağlıdır . Başka bir yaklaşım da mümkündür. Böylece genel hata oranı PE* (2.103)'deki gibi olur:

$$\text{PE*} = \text{P}_{\text{X,Y}}\,(\text{P}_{\Theta}(\text{h}(\text{X},\Theta)=\text{Y}\,) - \max_{j \neq Y} P_{\Theta}(h(\mathbf{X},\Theta) = j) < 0) \leq$$
$$\sum_j P_{X,Y}\,(P_{\Theta}(h(\mathbf{X},\Theta) = Y) - P_{\Theta}(h(\mathbf{X},\Theta) = j) < 0). \qquad (2.103)$$

Bu doğrultuda, her sınıf için marjin değeri (2.104) denklemindeki gibi tanımlanır:

$$s_j = P_{X,Y}\,(P_{\Theta}(h(\mathbf{X},\Theta) = Y) - P_{\Theta}(h(\mathbf{X},\Theta) = j) \qquad (2.104)$$

j sınıfına göre sınıflandırıcılar kümesinin $\{h(\mathbf{x},\Theta)\}$ kuvveti olsun. Bu güç tanımı ormana bağlı değildir. Bu tanım, sınıflandırıcılar kümesinin j sınıfı üzerindeki ayrım gücünü ifade eder. Chebyshev eşitsizliği kullanıldığında, tüm $s_j>0$ için aşağıdaki denklem (2.105)'te hata oranı üst sınırı şu şekilde verilir:

$$PE^* \leq \sum_j var(P_{\Theta}(h(\mathbf{X},\Theta) = Y) - P_{\Theta}(h(\mathbf{X},\Theta) = j)/s_j^2 \qquad (2.105)$$

ve (2.96)'nın türetilmesinde kullanılanlara benzer tanımlar kullanılarak, (2.105)'deki varyanslar ortalama korelasyonlar cinsinden ifade edilebilir.

2.2 Kural Tabanlı Modellerin Değerlendirme Metrikleri

Birliktelik Kuralı Madenciliği (Association Rule Mining), büyük veri kümeleri içerisinde ögeler (özellikle değişken, ürün, durum vb.) arasındaki bağımlılıkları ve örüntüleri keşfetmek için veri madenciliğinde sıkça kullanılan bir tekniktir. Literatürde bu yöntemle ilişkili olarak **"ilginçlik ölçütleri" (interestingness metrics)** terimi de kullanılmaktadır. Özellikle "**support**" (destek), "**confidence**" (güven) ve "**lift**" (kaldırma oranı) gibi ölçütler, birliktelik kuralı madenciliğinde en yaygın kullanılan metrikler arasındadır. Bu metriklerin geliştirilmesindeki temel amaç, verideki ögeler arasında anlamlı ilişkileri ortaya koyarak kullanıcıya değerli bilgiler sunmaktır. Böylece geçmiş ilişkilerin analizi yapılabilir ve bu analiz, gelecekteki çalışmalar veya karar alma süreçleri için yön gösterici olabilir[81, 82].

Support ve **confidence** metrikleri, ilk olarak Agrawal ve Srikant tarafından önerilmiştir ve birliktelik kuralı madenciliğinin temelini oluşturur. Bu metrikler, belirli ögelerin bir veri kümesinde birlikte görülme olasılığını ifade eder. Daha

sonra geliştirilen **lift** gibi ek metrikler ise, kuralların birbirine ne kadar bağlı olduğunu ya da bağımsız olup olmadığını tespit etmeye yardımcı olur. Pozitif ya da negatif bağımlılıkların derecesini belirler. Ayrıca, **certainty factor** ve **netconf** gibi ilginçlik ölçütleri, özellikle yanıltıcı veya bağımsız kuralların etkisini azaltarak daha anlamlı sonuçlar elde edilmesini sağlar[82].

2.2.1 Children of the Tree Algoritması için Özelleştirilmiş Kural Değerlendirme Metrikleri

Bu bölümde, Children of the Tree algoritması tarafından çıkarılan kuralların değerlendirilmesinde kullanılan ve klasik birliktelik kuralı metriklerinden uyarlanarak sınıflandırma bağlamına adapte edilen ölçütler sunulmaktadır. Children of the Tree algoritması, çıkarılan kuralların veri kümesinde ne kadar anlamlı ve ilgi çekici olduğunu değerlendirmek amacıyla çeşitli istatistiksel ölçütler kullanmaktadır[6]. Bunlar arasında en temel olanı **support (destek)** ölçütüdür ve bir kuralın veri kümesinde ne sıklıkla geçtiğini, yani kaç örneğe uygulandığını gösterir. **Confidence (güven)** değeri ise bir kuralın doğruluk oranını temsil eder; yani öncül koşul gerçekleştiğinde sonucun ne kadar sıklıkla gerçekleştiğini ölçer. **Lift (kaldırma)** metriği, kuralın tahmin başarısını tesadüfi sınıflandırmaya göre ne kadar iyileştirdiğini gösterir; değer 1'den büyükse kural, sonucu beklenenden daha iyi açıklıyor demektir. **Leverage (kaldıraç)** ise kuralın gerçekleşme oranı ile, öncül ve sonucun birbirinden bağımsız olduğu varsayımına dayalı beklenen oran arasındaki farkı ölçerek birlikteliğin gücünü değerlendirir. Son olarak, **conviction (ikna gücü)** metriği, kuralın doğruluğunun ne kadar ikna edici olduğunu, özellikle de kuralın yanlış çıktığı durumlara kıyasla nasıl performans gösterdiğini hesaplar[6,83]. Bu metrikler birlikte kullanılarak, Children of the Tree algoritmasının ürettiği kurallar yalnızca doğruluklarına göre değil, aynı zamanda ne kadar açıklayıcı ve anlamlı olduklarına göre de sıralanmakta ve seçilmektedir.

Support (Destek): Bir kuralın, veri kümesinde ne sıklıkla geçtiğini gösterir. Yani, kuralın kaç örneğe uygulandığını belirtir[6].

$$\text{Support} = \frac{N\text{rule}}{N\text{total}} \tag{2.106}$$

- *Nrule: Kurala uyan örnek sayısı*

- *Ntotal: Toplam örnek sayısı*

Yorum: Yüksek destek değeri, kuralın veri kümesinde sık tekrarlandığını gösterir.

Confidence (güven): Bir kuralın doğru olma olasılığını ölçer. Özellikle belirli bir özellik ya da koşul verildiğinde, kuralın ne sıklıkla doğru çıktığını ifade eder[6].

$$\text{Confidence} = \frac{N\text{correct}}{N\text{rule}} \tag{2.107}$$

Ncorrect: Kuralın hem öncül hem sonuç kısmının doğru olduğu örnek sayısı

Nrule: Kurala uyan toplam örnek sayısı

Yorum: Güven değeri, kuralın doğruluk oranını verir. 1'e yakınsa, kural oldukça güvenilirdir.

Lift(kaldırma): Bir kuralın başarımının beklenen doğruluğa göre ne ölçüde iyi olduğunu değerlendirir. Lift, bir kuralın sadece tesadüfen mi çalıştığını, yoksa gerçekten güçlü bir ilişkiye mi dayandığını anlamamıza yardımcı olur. Eğer A meydana geldiğinde B'nin olma olasılığı, B'nin genel olma olasılığına kıyasla artıyorsa, bu kural "tesadüfi olmayan", anlamlı bir ilişkidir. Lift bunu **"kaç kat daha iyi?"** sorusuyla cevaplar. Lift değeri 1'den büyükse, A'nın olması B'nin olma olasılığını artırıyor demektir. 1'den küçükse, A varken B'nin olma ihtimali azalıyor demektir[83].

$$\text{Lift} = Confidence / \frac{N\text{class}}{N\text{total}} \tag{2.108}$$

Confidence: Kuralın güven değeri

$\frac{Nclass}{Ntotal}$ *: Veri kümesinde ilgili sınıfın genel oranı*

Yorum:

- Lift = 1 → Öncül ve sonuç bağımsız
- Lift > 1 → Kural, sonuç sınıfı üzerinde pozitif etkili
- Lift < 1 → Kural, sonucu beklenenden daha az açıklıyor

Leverage(kaldıraç): Bir kuralın gerçekleşme sıklığı ile bu kuralın öncül ve sonucunun istatistiksel olarak birbirinden bağımsız olması durumunda beklenen gerçekleşme sıklığı arasındaki farkı ölçer. Başka bir deyişle, kuralın gözlemlenen destek (support) değeri ile öncül ve sonucun bağımsız kabul edildiği durumda

hesaplanan teorik destek değeri arasındaki farktır. Bu fark pozitifse, kural beklenenden daha sık gerçekleşiyor demektir ve aralarında anlamlı bir ilişki olabileceğine işaret eder; negatifse, beklenenden daha seyrek görülüyor demektir[6,83].

$$\text{Leverage} = Support - \frac{N\text{antecedent}}{N\text{total}} x \frac{Nclass}{Ntotal} \quad (2.109)$$

Support: Kuralın destek değeri

$\frac{N\text{antecedent}}{N\text{total}}$*: Öncül koşulun gerçekleşme oranı*

$\frac{Nclass}{Ntotal}$ *: Sınıfın genel oranı*

Yorum:

- Leverage değeri 0 ise → kuralın etkisi tesadüfle aynı
- 0'dan büyükse → anlamlı bir birliktelik vardır

Conviction(ikna gücü): Bir kuralın doğruluğunun ne kadar güvenilir ve istisnasız olduğunu ölçer. Özellikle, A gerçekleştiğinde B'nin gerçekleşmeme olasılığı ne kadar düşükse, conviction değeri o kadar yüksek olur. Bu da kuralın istatistiksel olarak ne kadar tutarlı ve ikna edici olduğunu gösterir. Kuralın yanlışlandığı durumlar nadirse, conviction metriği yükselir; dolayısıyla yüksek conviction, kuralın güvenilirliğinin güçlü bir göstergesidir[83].

$$\text{Conviction} = \frac{1 - \frac{Nclass}{Ntotal}}{1 - \text{Confidence}} \quad (2.110)$$

Confidence: Kuralın güven değeri

$\frac{Nclass}{Ntotal}$ *: Sınıfın genel oranı*

Yorum:

- Conviction değeri 1'e eşitse → kural ve sonuç birbirinden bağımsız
- 1'den büyükse → kural ikna edici, sonuçla ilişkili
- Çok yüksek conviction → kuralın yanlış tahmin yapma olasılığı düşüktür

2.3 Veri Dengesizliği ve Model Karmaşıklığı ile Başa Çıkma Yöntemleri

Makine öğrenmesi uygulamalarında sınıf dağılımındaki dengesizlik ve model karmaşıklığı, doğru ve açıklanabilir tahminler elde etmenin önündeki başlıca engellerdendir. Dengesiz veri kümeleri, özellikle azınlık sınıfın ihmal edilmesine neden olarak sınıflandırma performansını düşürürken; yüksek model karmaşıklığı, aşırı uyuma (overfitting) yol açarak hem genellenebilirliği zayıflatmakta hem de yorumlanabilirliği azaltmaktadır. Bu çalışmada söz konusu iki problemle başa çıkmak amacıyla iki farklı yaklaşım benimsenmiştir: **SMOTE** tekniği, sınıflar arası dağılımı dengelerken; **Minimum Description Length (MDL)** ilkesi ise geliştirilen kural çıkarım algoritmasında, modelden elde edilen kuralların **basit, açıklanabilir ve düşük karmaşıklığa sahip** olmasını sağlamaktadır. MDL, yalnızca doğruluk değil, aynı zamanda **bilgi yoğunluğu yüksek ve yorumlanabilir** kuralları önceliklendirerek modelin açıklanabilirliğini artırmaya katkı sunmaktadır.

2.3.1 SMOTE

SMOTE, dengesiz sınıf yapısına sahip veri kümelerinde, **azınlık sınıfın örnek sayısını artırmak** amacıyla kullanılan ***sentetik veri üretme*** yöntemidir[49]. Günümüzde makine öğrenmesi uygulamalarında çokça kullanılmaktadır.

Dengesiz veri kümesi problemi, sınıflandırma performansını olumsuz yönde etkileyen önemli bir faktördür[49]. Dengesiz sınıf dağılımına sahip veri kümelerinde, öğrenme algoritmaları çoğunluk sınıfı lehine öğrenme eğilimindedir; bu durum azınlık sınıfına ait örneklerin doğru tahmin edilememesine yol açar. Bu çalışmada, azınlık sınıfına ait örnek sayısını artırmak için **SMOTE (Synthetic Minority Over-sampling Technique)** yöntemi kullanılmıştır.

SMOTE, Chawla ve arkadaşları (2002) tarafından önerilen, veri kümesindeki azınlık sınıf örneklerinden yeni, sentetik örnekler üretmeye dayalı bir aşırı örnekleme (over-sampling) yöntemidir[49]. SMOTE, klasik çoğaltma yöntemlerinin aksine, azınlık sınıfındaki örneklerin doğrudan kopyalanması yerine, bu örnekler ile komşuları arasında yeni örnekler oluşturarak veri setini genişletir. Böylece, öğrenme algoritmasının karar sınırları daha genelleyici hâle gelir ve modelin aşırı öğrenme (overfitting) riski azaltılır[49]. Diğer bir deyişle modele veri

kümesindeki örneklemler arasında ara alternatif noktalar sunarak desenin daha iyi anlaşılmasına olanak tanır diyebiliriz.

Matematiksel olarak SMOTE işlemi şu şekilde tanımlanabilir: Azınlık sınıfına ait her bir örnek x_i için, aynı sınıftan k adet en yakın komşusu belirlenir (genellikle $k = 5$). Bu komşular arasından rastgele bir tanesi seçilir (x_{zi}). Daha sonra, bu iki örnek arasındaki fark vektörü ile rastgele bir sayı ($\delta \in [0,1]$) çarpılarak, x_i'ye eklenir ve yeni bir örnek aşağıdaki (2.111) gibi oluşturulur:

$$x_{yeni} = x_i + \delta \cdot (x_{zi} - x_i) \tag{2.111}$$

Bu işlem sonucunda elde edilen x_{yeni} örneği, veri uzayında x_i ile x_{zi} arasındaki doğru parçası üzerinde yer alır. Bu sayede üretilen örnek, mevcut veri yapısına uygun, gerçekçi bir azınlık sınıf örneği olur[49].

SMOTE yöntemi, özellikle veri kümesinin azınlık sınıfa ait örneklerinin sayıca yetersiz olduğu durumlarda sınıflandırma algoritmalarının performansını artırmakta etkili bir yöntem olarak kabul edilmektedir. Bununla birlikte, yöntemin bir sınırlılığı, gürültülü veya hatalı azınlık örneklerinden de yeni örnekler üretebilecek olmasıdır.

Bu çalışmada SMOTE yöntemi, *Alman Kredi Veri Seti* üzerindeki "iyi" ve "kötü" sınıflar arasındaki dengesizliği azaltmak ve makine öğrenmesi modellerinin her iki sınıfta da dengeli performans göstermesini sağlamak amacıyla uygulanmıştır.

2.3.2 Minimum Açıklama Uzunluğu (MDL) İlkesi

Minimum Açıklama Uzunluğu (MDL) ilkesi, başta **Jorma Rissanen** tarafından yayımlanan bir dizi çalışma ile geliştirilmiş, bilgi kuramına dayanan bir model seçme yöntemidir[84, 86, 87]. Bu ilkenin kökleri, 1960'larda **Solomonoff**, **Kolmogorov** ve **Chaitin** tarafından geliştirilen **Kolmogorov ya da algoritmik karmaşıklık kuramına** dayanmaktadır[88].

MDL ilkesi, makine öğrenmesi ve istatistikte **Occam'ın Usturası** ilkesinin biçimsel (formel) bir ifadesi olarak kabul edilir. Model seçimi sürecinde MDL, modelin karmaşıklığı ile veriye uyum düzeyi arasında bir denge kurmayı amaçlar[85].

MDL Kavramı

MDL'ye göre, bir veri kümesi için en uygun model, **veriyi en iyi sıkıştıran modeldir**. Bu yaklaşım iki temel bileşenden oluşur[89,90]:

- **Model Karmaşıklığı (L(h))**: Hipotezin veya modelin tanımı için gereken bit sayısını ifade eder. Basit modeller genellikle daha kısa tanım uzunluğuna sahiptir.
- **Veriye Uyum Maliyeti (L(D | h))**: Model kullanılarak verinin açıklanması için gereken bit sayısını ifade eder. Veriye iyi uyum sağlayan bir model, hataları ya da sapmaları kodlamak için daha az bit gerektirir.

Zamanla MDL, örüntü madenciliğinden model sadeleştirmeye kadar geniş bir alanda kullanılmaya başlanmıştır. Örneğin, **KRIMP** algoritması sık itemset'lerden oluşan örüntü tablosunu MDL'ye göre optimize ederek veriyi en iyi sıkıştıran desenleri seçer[91]. Benzer şekilde, **SLIM (Supersparse Linear Integer Models)** gibi açıklanabilir modellerde L0 ve L1 ceza terimleri kullanılarak modelin karmaşıklığı kontrol altına alınır; bu yapı MDL'ye kuramsal olarak benzer olsa da doğrudan MDL ilkesiyle tanımlanmış değildir[92].

Ancak bu yaklaşımlar **doğrudan IF–THEN biçimli kural çıkarımı amacıyla MDL kullanan yapılar değildir.** Bu açıdan istisna sayılabilecek örneklerden biri, **MARS** (MDL-based Association Rule Selection) algoritmasıdır. MARS, birliktelik kuralı madenciliği sonucunda elde edilen kurallar arasından, veriyi en kısa açıklayanları seçmek için MDL hesaplamaları yapar. Burada hem L(R) hem de L(D|R) hesaplanır ve en sade kural kümesi belirlenir[93]. MARS önceden verilen bir kural listesi üzerinden çalıştırılır, Children of the Tree algoritmamız gibi herhangi bir makine öğrenmesi modelini açıklama yeteneği yoktur.

Bu bağlamda, **Children of the Tree** algoritması dikkat çeken bir yaklaşımdır. Yöntem, Rastgele Orman gibi ağaç tabanlı modellerin içindeki karar yollarının üzerinden geçerek **IF–THEN** kurallar üretir**.** Üretilen bu kurallar daha sonra sadece support ve confidence gibi klasik ölçütlerle değil, aynı zamanda her bir kuralın **açıklama uzunluğuna (MDL temelli)** göre de sıralanır. Böylece yalnızca yüksek doğruluklu değil, aynı zamanda **en sade ve açıklanabilir** olan kurallar öne çıkarılır[6].

Sonuç olarak, MDL ilkesi birçok örüntü tanıma, sıklıkla birlikte geçen veri yığını tespiti ve hali hazırda eldeki kural listesinden seçim gibi örneklerde daha yaygın şekilde kullanılmakla birlikte, doğrudan model içinden kural çıkarımı ve açıklama üretimi amacıyla kullanımına araştırmalarımız sırasında rastlanmamıştır, etkin ve birebir kullanım bulunamamıştır. **Children of the Tree**, bu yönde MDL'yi tam anlamıyla kullanan ender örneklerden biridir ve hem bilgi kuramsal hem de açıklanabilirlik odaklı güçlü bir zemin sunar[6].

MDL'nin Matematiksel Formülasyonu

Matematiksel olarak, toplam tanım uzunluğu L(D, h) şu şekilde ifade edilir[90]:

$$L(D, h) = L(h) + L(D \mid h) \tag{2.112}$$

Burada;

$\boldsymbol{L(h)}$: Modelin kendisinin tanım uzunluğudur (karmaşıklığı),

$\boldsymbol{L(D \mid h)}$: Verinin modele göre tanım uzunluğudur (hata ya da artık bilginin kodlanması).

MDL ilkesinde amaç, bu toplam tanım uzunluğunu minimuma indirmektir[90]. Bu yaklaşım, modelin **sadeliği** ile verinin doğruluğunu (temsil kabiliyetini) dengede tutmayı hedeflemektedir.

Kural Çıkarımında MDL'nin Uygulanması

Minimum Açıklama Uzunluğu (MDL) ilkesi, kural çıkarımı bağlamında bir kural kümesinin "maliyetini" değerlendirmek amacıyla kullanılabilir. Bu bağlamda, MDL'in matematiksel formülasyonu yukarıda ilgili bölümde verilmiştir, bkz. *MDL'in Matematiksel Formülasyonu*[6]. Buna göre:

- Çok karmaşık bir kural (yani çok fazla koşul içeren bir kural), yüksek bir **L(h)** (modelin tanımlama uzunluğu) değerine sahip olacaktır.
- Veriye zayıf uyum sağlayan bir kural ise yüksek bir **L(D|h)** (veriyle modelin uyuşmazlığı) değerine sahip olur.

En iyi kural kümesi, bu iki bileşenin toplamı olan açıklama uzunluğunu (description length) **minimize eden** kümedir. Geliştirdiğimiz algoritma, literatürde tanımlanan bu prensiplere uygun şekilde **MDL hesaplamalarını** uygulamaktadır[6].

MDL Maliyetinin Hesaplanması: Kural Karmaşıklığı (L(h))

Eğer bir kural r, toplamda k adet koşul içeriyorsa, her bir koşul modelin karmaşıklığına katkı yapar. Her koşulun belirli sayıda bit gerektirdiği varsayımıyla kuralın toplam karmaşıklığı yaklaşık olarak şu şekilde hesaplanır[89]:

$$L(h) = k \cdot log_2(N) \tag{2.113}$$

Burada:

- *k*: kuraldaki koşul sayısıdır.
- *N*: veri kümesindeki örnek sayısını temsil eder. Bu değer, hesaplamada **çözünürlük düzeyi** olarak düşünülebilir ve modelin karmaşıklığını belirler.

Hata Maliyeti (L(D | h))

Bu bileşen, kural tarafından **yanlış sınıflandırılan örneklerin sayısını** yansıtır. Eğer E yanlış sınıflandırma varsa[89]:

$$L(D \mid h) = E \cdot log_2(N) \tag{2.114}$$

Yani, kuralın tahmin edemediği örnekler, modelin veriyle olan uyuşmazlık maliyetini artırır.

Toplam MDL Maliyeti

Her iki bileşen birleştirildiğinde, bir kuralın toplam MDL maliyeti şu şekilde hesaplanır[89]:

$$MDL(r) = k \cdot log_2(N) + E \cdot log_2(N) \tag{2.115}$$

Bu fonksiyon, hem **basit kuralları (küçük k)** hem de **doğru kuralları (küçük E)** teşvik eder.

MDL Değerinin Yorumu

- **Düşük bir MDL değeri**, model karmaşıklığı ve doğruluğu arasında iyi bir denge kurulduğunu gösterir.
- **Yüksek bir MDL değeri** ise kuralın ya çok karmaşık olduğunu (yüksek k), ya da çok fazla hata içerdiğini (yüksek E) ve bu nedenle tercih edilmemesi gerektiğini gösterir.

MDL değeri **mutlak bir kriter değildir**; her bir kural için hesaplanan MDL değeri, diğer kurallarla karşılaştırmalı olarak değerlendirilmelidir[89].

3
GELİŞTİRİLEN ALGORİTMA: CHILDREN of the TREE (AĞACIN ÇOCUKLARI) – RASTGELE ORMAN TABANLI KURAL ÇIKARIMI YAKLAŞIMI

Bu çalışma, **Rastgele Orman** gibi ağaç tabanlı sınıflandırma modellerinden kural çıkarımı gerçekleştiren yeni bir algoritmayı tanıtmaktadır. Özellikle **bankacılık gibi kurala dayalı tahminlerin kritik önem taşıdığı alanlarda**, potansiyel müşterilere yönelik anlamlı kurallar, geliştirilen algoritmanın temelinde önemli bir rol oynamaktadır[6]. Literatürdeki mevcut yöntemlerle karşılaştırıldığında, bu algoritmanın banka **risk yönetimi** ve **pazarlama departmanları** için **basit fakat etkili** bir çözüm sunması beklenmektedir.

Algoritma, MDL yöntemi sayesinde gereksiz karmaşıklık yaratan kuralları azaltmakta ve böylece daha anlaşılır ve optimize edilmiş bir kural kümesinin oluşturulmasına olanak sağlamaktadır[6].

3.1 "Children of the Tree" Algoritmasının Temeli

Rastgele Orman modeli, çok sayıda karar ağacından oluşan bir toplu öğrenme (ensemble learning) yöntemidir. Her bir karar ağacı, veri kümesindeki bir örnek T_i (i = 1,2,...,N) **belirli kurallar aracılığıyla sınıflandırmakta veya tahmin etmektedir**.

Karar ağaçları genellikle bir **kök düğüm (root node)** ile başlar. Bu kök düğüm *r*, veri kümesinin başlangıç noktasıdır. Eğer bir düğüm *d* **yaprak düğüm (leaf node)** ise, bu düğümde artık veri ayrıştırması yapılmaz ve düğüm doğrudan bir **sınıf etiketini** veya **tahmin edilen değeri** temsil eder.

Yaprak düğüm tamamlandığında, bu düğüm üzerinden çıkarılan kuralın anlamlılığını ve açıklanabilirliğini değerlendiren **ölçütler (M(d))** hesaplanır. Bu

ölçütler arasında **support (destek)**, **confidence (güven)** ve **Minimum Description Length (MDL)** gibi kriterler bulunmaktadır[6].

$$M(d) = f(d) \tag{3.1}$$

Burada; *f(d)*, yaprak düğüm ddd'den çıkarılan kuralın anlamlılığını temsil eden metrikleri göstermektedir[6]. Bu çalışmada kullanılan metrikler: **support, confidence, lift, leverage, conviction** ve **mdl** değerleridir.

Eğer düğüm d bir **iç düğüm (inner node)** ise, bu düğüm iki çocuk düğüme (sol: d_L ve d_R) sahip olabilir. Bu iç düğüm, veri üzerinde belirli bir özellik X_j ve eşik değeri θ_j kullanılarak ayrım yapar. Yani, karar kuralı bu düğümde belirlenir[6].

Matematiksel olarak, iç düğüm d için karar kuralı φ(d) şu şekilde tanımlanır:

$$\varphi(d) = \begin{cases} Xj \leq \theta j \text{ eğer sol çocuk düğüme geçiliyorsa} \\ Xj > \theta j \text{ eğer sağ çocuk düğüme geçiliyorsa} \end{cases} \tag{3.2}$$

Bu kural, ilgili düğümden ayrılan veriyi ikiye ayırır. **Sol ve sağ çocuk düğümler**, bu kural aracılığıyla oluşturulan iki alt veri kümesini temsil eder[6].

3.2 Algoritma Adımları: Ağacın Çocukları (Children of the Tree)[6]

Bu bölümde, ağaç tabanlı makine öğrenmesi modellerinden kural çıkarımı yapmayı amaçlayan **Children of the Tree** algoritmasının temel işleyiş adımları anlatılmaktadır. Algoritma detay adımları aşağıda Tablo 3.1'de sunulmaktadır[6].

Tablo 3.1 "Children of the Tree" iterasyonlar[6]

```
current_node          root
current_node_type     internal
current_node          is not null
while current_node is not null, do
   if current_node is leaf, do
      add rule for current_node to rule_list
   else
      identify left_child and right_child of current_node
      split_data using feature Xj and threshold θj
      generate rules:
         if (Xj ≤ θj) move to left_child
         if (Xj > θj) move to right_child
   end
   move to the next node in the tree (left_child or right_child)
end
```

[6] Kaynak [6] Children of the Tree'ye bakınız.

Aşağıda algoritmanın genel döngüsü yer almaktadır:

- **current_node** değişkeni, başlangıçta karar ağacının **kök düğümüne (root)** atanır.
- Düğüm tipi "iç düğüm (internal node)" olarak başlatılır.
- **current_node** boş olmadığı sürece, algoritma aşağıdaki adımları tekrarlar:

Başlangıç Noktası

Karar ağacı, **kök düğüm (r)** üzerinden başlatılır.

Karar Fonksiyonu

Herhangi bir düğüm d, veri kümesini sınıflandırmak için bir **karar fonksiyonu** f(d) kullanır. Bu fonksiyon, verinin düğümde nasıl ayrıldığını ve hangi yöne gideceğini belirler.

$$f(d)=\begin{cases}\text{Sınıf } C1 & \text{Eğer } d \text{ yaprak düğümse} \\ \text{Karar } (Xj \le \theta j \text{ ve } d_L) \text{veya } (Xj > \theta j \text{ ve } d_R) & \text{Eğer } d \text{ iç düğümse}\end{cases} \quad (3.3)$$

- Burada C1, yaprak düğümdeki sınıf etiketini veya tahmin edilen değeri temsil eder.
- İç düğümlerde karar fonksiyonu, veri kümesini ikiye ayıran kuralı ifade eder.

Yaprak Düğüm Kontrolü

Eğer d düğümü bir **yaprak düğüm** ise, o düğüme ait kural, (3.4)'teki gibi **kural listesine eklenir**:

$$\text{Rule List} \leftarrow \text{Rule List} \cup \{(D)\} \quad (3.4)$$

İç Düğüm İşlemleri

Eğer d bir **iç düğüm (internal node)** ise (3.5) gibidir:

$$\text{Rules(d)} \leftarrow \{(\text{Xj} \le \theta\text{j and } d_L), (\text{Xj} > \theta\text{j and } d_R)\} \quad (3.5)$$

- Sol çocuk düğüm d_L ve sağ çocuk düğüm d_R belirlenir.
- Özellik X_j ve eşik değeri θ_j kullanılarak karar kuralları oluşturulur:

Her bir çocuk düğüm için aynı işlem tekrarlanır. Bu süreç, ağaç boyunca tüm yolların (yani kuralların) çıkarılması tamamlanana kadar devam eder[6].

Kural Değerlendirme ve Seçimi

Tüm kural çıkarımı işlemi tamamlandıktan sonra, her bir kural için support, confidence, lift, leverage, conviction **değerlendirme metrikleri** hesaplanır.

Ayrıca, her kuralın **karmaşıklık ve hata maliyetleri** dikkate alınarak **MDL (Minimum Description Length)** değeri hesaplanır. Tüm metrikler bir arada değerlendirilerek, **en anlamlı ve sade (optimum) kurallar** seçilir[6].

4
UYGULAMA

Bu bölümde, 2. Bölümde detaylı biçimde açıklanan makine öğrenmesi modelleri, kredi riski tahmini problemi kapsamında Alman kredi veri seti üzerinde uygulanmış ve elde edilen sonuçlar karşılaştırılmıştır. En iyi performansı gösteren model, elde edilen çıktılar yorumlanarak belirlenmiştir. Bu model temel alınarak, 4.2 numaralı bölümde geliştirdiğimiz **"Children of the Tree"** adlı kural çıkarımı algoritmasının (3. Bölümde ayrıntılı biçimde açıklanmıştır) uygulaması gerçekleştirilmiş ve elde edilen sonuçlara yer verilmiştir.

4.1 Makine Öğrenmesi Modellerinin Uygulanması

Bu çalışmada, kredi riski tahminine yönelik olarak farklı makine öğrenmesi algoritmaları değerlendirilmiştir. Kullanılan veri kümesi, UCI Machine Learning Repository'de yer alan ve literatürde sıklıkla tercih edilen Alman kredi veri kümesidir. Veri kümesi, bankaya kredi başvurusunda bulunan bireylere ait demografik ve finansal bilgileri içermekte ve her bireyin kredi riski durumu ("iyi" ya da "kötü" müşteri) etiketlenmiş durumdadır. Bu yönüyle problem, **denetimli öğrenme (supervised learning)** sınıflandırma problemi olarak ele alınmıştır.

4.1.1 Veri Ön İşleme ve Dengeleme

İlk olarak, veri kümesi içerisindeki hedef değişken olan "**credit_risk**" ayrılmış ve model girişlerini oluşturan tüm özellikler (X) ile hedef sınıf (y) ayrıştırılmıştır. Veri kümesinde bazı değişkenler kategorik olduğu için bu değişkenler **etiketleme yöntemiyle (Label Encoding)** sayısal forma dönüştürülmüştür. Bu dönüşüm, ağaç tabanlı algoritmalar dışındaki modellerin, özellikle doğrusal ve mesafe tabanlı algoritmaların (örneğin Lojistik Regresyon, KNN, SVM) doğru şekilde çalışabilmesi açısından kritik bir ön işleme adımıdır.

Veri kümesi incelendiğinde, hedef değişken sınıflarının *dengesiz* olduğu gözlemlenmiştir. Bu tür durumlar, öğrenme sürecinde modellerin azınlık sınıfı doğru öğrenememesiyle sonuçlanabilir. Bu sorunu gidermek amacıyla, ***SMOTE (Synthetic Minority Over-sampling Technique)*** yöntemi kullanılmıştır. SMOTE, azınlık sınıfa ait örneklerden benzer fakat yeni örnekler üretir ve böylece sınıflar arasında dengeli bir dağılım sağlar.

Tüm modellerin eşit koşullar altında karşılaştırılabilmesi amacıyla, özellikler ayrıca **standartlaştırılmıştır (StandardScaler)**. Bu işlem, özellikle **SVM ve KNN gibi mesafeye duyarlı** algoritmalarda model başarımı üzerinde doğrudan etkili olsa da, RF **ve** DT **gibi ağaç tabanlı modellerde teknik olarak zorunlu değildir.** Ancak modellerin aynı ölçekte eğitilmesini sağlamak ve değerlendirmede adaletli bir karşılaştırma yapabilmek adına, tüm modellere **ölçeklenmiş veri** kullanılarak eğitim ve test uygulanmıştır.

4.1.2 Eğitim ve Test Verisinin Oluşturulması

SMOTE uygulamasının ardından veri kümesi, %80 eğitim ve %20 test oranı ile rastgele bölünmüştür. Böylece modellerin hem öğrenme yetenekleri hem de genellenebilirlikleri tarafsız bir şekilde test edilmiştir.

4.1.3 Özellik Ölçekleme

SVM (Support Vector Machine) ve **KNN (K-Nearest Neighbors)** gibi modeller, uzaklık ve sınır temelli çalıştıkları için değişkenler arası farklı ölçeklerden olumsuz etkilenebilmektedir. Bu nedenle yalnızca bu iki model için **Z-normalizasyonu (StandardScaler)** uygulanmış, tüm değişkenler ortalaması 0 ve standart sapması 1 olacak şekilde dönüştürülmüştür.

4.1.4 Uygulanan Modeller

Çalışmada toplam **altı farklı makine öğrenmesi algoritması** değerlendirilmiştir:

- Lojistik Regresyon (Logistic Regression)
- K-en Yakın Komşu (KNN)
- Destek Vektör Makineleri (SVM)
- Karar Ağaçları (Decision Tree)
- Naive Bayes

- Rastgele Orman (Random Forest)

Bu modeller, hem doğrusal (örneğin lojistik regresyon) hem de doğrusal olmayan (örneğin karar ağaçları, rastgele orman) karar sınırlarına sahip olup, farklı varsayımlarla çalışmaları sayesinde model karşılaştırmasında kapsamlı bir bakış açısı sunmuştur.

Her bir model, eğitim verisi üzerinde eğitilmiş ve test verisi üzerinde değerlendirilmiştir. Bu süreçte karmaşıklık matrisi (confusion matrix), sınıflandırma raporu (classification report) **ve** ROC eğrisi gibi yaygın ölçütler kullanılmıştır.

4.1.5 Performans Özeti

Her bir modelin doğruluk (accuracy) ve sınıflar bazında F1 skorları tablo halinde sunulmuştur. F1 skoru, özellikle dengesiz veri kümelerinde değerlendirme için uygun bir ölçüttür çünkü hem precision (kesinlik) hem recall (duyarlılık) bileşenlerini dikkate alır.

Aşağıda Tablo 4.1'de, kredi riski sınıflandırma sürecinde uygulama bölümünde denediğimiz 6 farklı algoritmanın sınıflandırma sonuçlarına ait karşılaştırmalar verilmiştir. Gözlemlenen tüm makine öğrenmesi algoritmalarına aynı ön işleme adımları (sayısallaştırma, SMOTE, standardizasyon) uygulanmış ve karşılaştırma adil koşullar altında gerçekleştirilmiştir. Rapor tablosuna göre her iki sınıfın F1 skorları göz önüne alındığında en iyi performans gösteren algoritma Sınıf 1 ve Sınıf 2 için sırasıyla 0.84 ve 0.82 F1 skorları ile "Random Forest" olmuştur.

Tablo 4.1 Model karşılaştırma tablosu

Model Adı	**Accuracy**	**F1 Score (Class 1)**	**F1 Score (Class 2)**
Logistic Regression	0.81	0.81	0.80
KNN	0.81	0.82	0.81
SVM	0.80	0.81	0.79
Decision Tree	0.75	0.76	0.75
Naive Bayes	0.78	0.77	0.79
Random Forest	**0.83**	**0.84**	**0.82**

Gözlemlenen her 3'er algoritmanın karmaşıklık matrislerinin detayı aşağıda Şekil 4.1 ve Şekil 4.2'de verilmiştir. Buna göre dikkat çeken bazı yorumlar şöyle verilebilir:

Karmaşıklık matrisinin hedef değişkenlere (target) uyarlaması şöyledir: doğru negatifler 1-1 olarak; doğru pozitifler 2-2 olarak gözlemlenmektedir.

Modellerin **doğru negatif (True Negative)** ve **doğru pozitif (True Positive)** sınıflandırma başarıları, karmaşıklık matrisleri üzerinden incelendiğinde, **Rastgele Orman** modeli, **sınıf 1'i (negatif sınıf) doğru tahmin etme konusunda** en başarılı model olarak öne çıkmaktadır. Gerçekten de, RF modeli toplam **124 örneği doğru şekilde sınıf 1 olarak** tahminlemiş ve bu açıdan en yüksek doğru negatif sayısına ulaşmıştır. Bu durum, modelin iyi sınıfı (borcunu ödeyen müşteri) ayırt etme kabiliyetinin oldukça güçlü olduğunu göstermektedir.

Ancak **doğru pozitif eşleştirmeler (2–2 tahminleri)** dikkate alındığında, **LR**, **KNN** ve **NB** modelleri, yüzeysel olarak **RF**'den daha iyi görünmektedir. Bunun nedeni, bu modellerin genellikle **sınıf 2'ye (riskli müşteri) yönelme eğiliminde olmalarıdır**. Bu eğilim, doğru pozitif sayısının yüksek görünmesine neden olsa da, yanıltıcıdır. Çünkü aynı modellerde **yanlış pozitif (1 sınıfına aitken 2 olarak tahmin edilen) örnek sayısı da oldukça fazladır.**

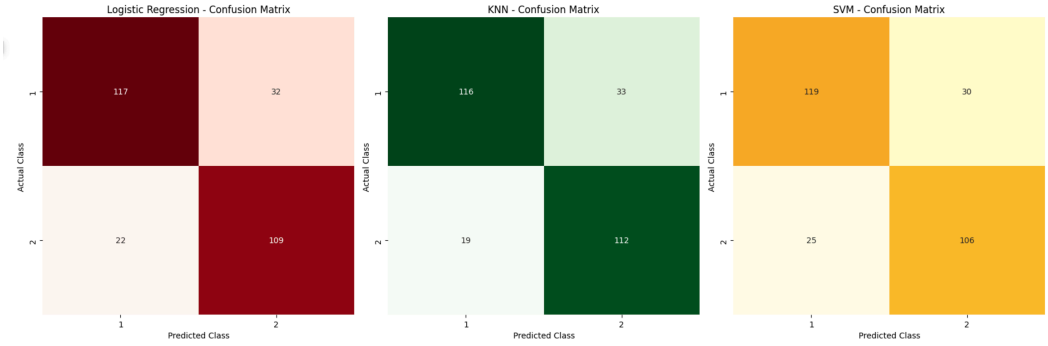
Bu durum, modellerin sınıf 2'yi fazla tahminleme eğiliminden kaynaklanmaktadır ve **modelin gerçek başarı oranını düşüren bir unsur olarak değerlendirilmelidir**. Diğer bir ifadeyle, Lojistik Regresyon, KNN ve Naive Bayes modelleri sınıf 2 için daha fazla tahminde bulundukları için daha fazla doğru pozitif elde etmişlerdir; ancak bu tahminlerin birçoğu aynı zamanda **yanlış pozitiflere de yol açmıştır**.

Buna karşılık, **RF** modeli dengeli bir şekilde hem **doğru negatif (1–1)** hem de **doğru pozitif (2–2)** sınıflandırmalarda güçlü performans göstermiştir. Ayrıca bu modelin hem **yanlış pozitif** hem de **yanlış negatif** tahmin sayıları diğer modellere göre belirgin biçimde düşüktür. Bu da "Random Forest"ın sadece sınıf 1'e değil, aynı zamanda **azınlık sınıfı olan sınıf 2'ye dair de daha isabetli ve temkinli kararlar verebildiğini göstermektedir.**

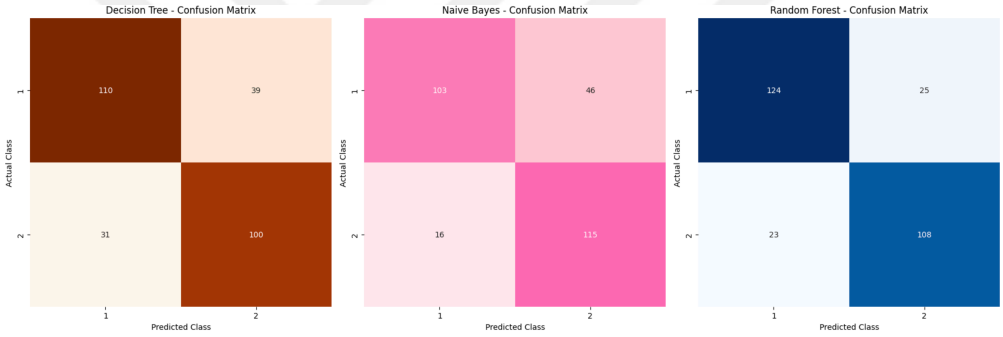
Sonuç olarak, yalnızca doğru pozitif sayısına bakarak model başarısını yorumlamak yanıltıcı olabilir. Daha bütüncül bir değerlendirme yapıldığında, RF modeli, her iki

sınıf için de yüksek doğruluk ve düşük hata oranlarıyla öne çıkan, genel model başarımı açısından en güçlü adaydır.

Algoritmaların karmaşıklık matrislerinin detayı aşağıda Şekil 4.1 ve Şekil 4.2'de verilmiştir.



Şekil 4.1 Lojistik Regresyon, KNN ve SVM modelleri için karmaşıklık matrisi sonuçları

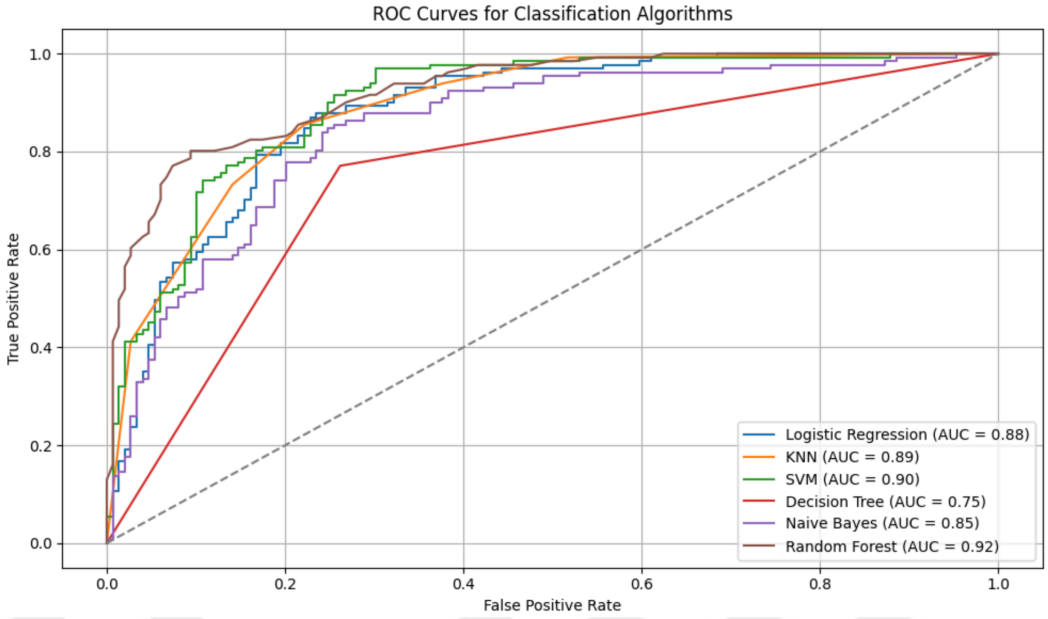


Şekil 4.2 Decision Tree, Naive Bayes ve Random Forest modelleri için karmaşıklık matrisi sonuçları

4.1.6 ROC Eğrisi ve AUC Değerleri

Model başarımının sınıflar arası ayrım gücünü değerlendirmek amacıyla, her bir model için **ROC eğrileri çizilmiş** ve **AUC** değerleri hesaplanmıştır. ROC eğrisi altında kalan alan ne kadar büyükse, modelin sınıfları doğru ayırt etme gücü de o derece yüksek olmaktadır. Elde edilen sonuçlara göre, "Random Forest" ve SVM modelleri en yüksek AUC değerlerini elde etmiş, bu da onların kredi riski tahmini görevinde daha başarılı olduğunu göstermiştir.

Aşağıda Şekil 4.3'te, modellerin karşılaştırmaları için ROC (Receiver Operating Characteristic) eğrisi verilmiştir.



Şekil 4.3 Modellerin ROC eğrilerini gösteren grafik

ROC eğrisi, sınıflandırma algoritmalarının pozitif sınıfı (sınıf 2 – riskli müşteri) ne kadar başarılı ayırt ettiğini görsel olarak göstermektedir. Eğriler ne kadar sol üst köşeye yakınsa, modelin sınıflandırma gücü o kadar yüksektir.

Grafiğe göre:

- **RF** modeli, **AUC = 0.92** ile en yüksek doğruluk alanına sahiptir ve pozitif sınıfı ayırt etmede en başarılı modeldir.
- **SVM (0.90)**, **KNN (0.89)** ve **Logistic Regression (0.88)** da yüksek performans göstermiştir ve birbirlerine oldukça yakındır.
- **Naive Bayes (0.85)** orta düzeyde başarılı bir ayrım yaparken,
- **Decision Tree (0.75)** modeli ROC eğrisine göre diğer modellere kıyasla daha düşük ayrım gücüne sahiptir.

Genel olarak, ROC eğrisi RF'nin hem düşük yanlış pozitif (false positive) oranıyla hem de yüksek doğru pozitif (true positive) oranıyla **daha dengeli ve üstün bir sınıflama performansı** sergilediğini ortaya koymaktadır.

4.1.7 Yorum

Elde edilen sonuçlara göre, **Rastgele Orman** modeli hem genel başarı hem de sınıflar arası dengeli F1 skorları açısından en başarılı model olarak öne çıkmıştır. Bu nedenle sonraki bölümde uygulanacak olan **"Children of the Tree"**

algoritması ile kural çıkarımı işlemi, Rastgele Orman modeli üzerinden gerçekleştirilmiştir. Böylece hem sınıflandırma başarısı yüksek hem de açıklanabilirliği arttırılmış bir model yapısı elde edilmiştir[6].

4.2 Children of the Tree Uygulaması

Bu bölümde, geliştirdiğimiz algoritmanın uygulanması sunulmaktadır. "Children of the Tree" algoritmasının işleyişi, Bölüm 3'te ayrıntılı şekilde açıklanmıştır. Bu bölümde ise, 1.5 "Veri Kümesinin Tanıtımı" başlığında sunulan veri kümesi üzerinde "Children of the Tree" algoritması uygulanmış ve elde edilen sonuçlar aşağıdaki Tablo 5'te verilmiştir.

"Children of the Tree" algoritması, literatürde benzer bir çözüme rastlanmaması nedeniyle özgün fikirle geliştirilmiştir. Özellikle, makine öğrenmesi modellerinden kural çıkarımı yapılırken **MDL (Minimum Description Length)** ilkesinin uygulanması, algoritmanın literatüre **özgün bir katkı** sunduğunu göstermesi açısından önem taşımaktadır.

"Children of the Tree" algoritması, **Rastgele Orman** temelli bir makine öğrenmesi modelinden **kural çıkarımı** gerçekleştirmek üzere tasarlanmış bir **optimizasyon algoritmasıdır**. Çalışmada, algoritmanın **dengesiz veri kümeleri** üzerinde de çalışabilmesini sağlamak amacıyla **SMOTE (Synthetic Minority Oversampling Technique)** yöntemi kullanılmıştır.

Çalışma iki ana bileşene odaklanmaktadır. İlk olarak, dengesiz veri kümesi SMOTE tekniği ile dengelenmiş; ardından, RF modeli üzerinde çeşitli hiperparametre optimizasyonları uygulanarak genel F1 skoru yaklaşık **0.83** olacak şekilde bir performans elde edilmiştir. Bu kapsamda, Sınıf 1 (iyi sınıf) için F1 skoru **0.84**, Sınıf 2 (kötü sınıf) için ise **0.82** olarak hesaplanmıştır. Bu sonuçlara dayanarak, Bölüm 2'de ayrıntıları verilen "Children of the Tree" algoritmasının kural çıkarım mekanizması uygulanmıştır.

Kural çıkarımı sonrasında her bir kural için **support (destek)**, **confidence (güven)**, **lift (kaldırma)**, **leverage (kaldıraç)** ve **conviction (ikna gücü)** gibi metrikler hesaplanmıştır. Ayrıca, her bir kuralın **karmaşıklık** ve **hata maliyeti** dikkate alınarak **MDL değeri** belirlenmiştir. Bu metriklerin tamamı değerlendirildikten sonra, **optimal kurallar** seçilmiştir. Aşağıdaki **Tablo 4.2'de**, seçilen optimal

kuralları, bu kuralların temsil ettiği sınıfları ve ilgili metrik değerlerini (geçerlilik ve yaygınlık dahil olmak üzere) sunmaktadır. Ayrıca tabloda, kuralların **sadelik derecelerini yansıtan MDL maliyet değerleri** de yer almaktadır[6].

Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar[6]

Rule	Class	Support	Confidence	Lift	Leverage	Conviction	MDL Cost
duration > 6.50 and savings_account <= 2.50 and checking_account <= 2.50 and other_debtors <= 1.50 and credit_history <= 3.50.	2	0,48	0,78	1,56	0,24	2,28	9176,16
savings_account <= 2.50 and other_debtors <= 1.50 and checking_account <= 2.50 and credit_history <= 3.50 and duration > 10.50.	2	0,45	0,79	1,59	0,22	2,42	9479,25
savings_account <= 1.50 and credit_history <= 3.50 and checking_account <= 2.50 and other_debtors <= 1.50 and personal_status_sex <= 2.50.	2	0,49	0,79	1,57	0,22	2,34	9521,05
present_residence <= 3.50 and checking_account <= 2.50 and foreign_worker <= 0.50 and other_debtors <= 1.50 and purpose <= 8.50.	2	0,43	0,76	1,52	0,22	2,07	9865,94
checking_account <= 2.50 and other_debtors <= 1.50 and credit_history <= 3.50 and duration > 12.50 and savings_account <= 3.50.	2	0,39	0,80	1,62	0,19	2,62	10116,77

Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar (devamı)[6]

checking_account <= 2.50 and other_debtors <= 1.50 and credit_amount > 1044.50 and number_credits <= 1.50 and credit_amount > 1173.00.	2	0,43	0,70	1,41	0,22	1,70	10189,93
credit_history <= 3.50 and checking_account <= 2.50 and duration > 7.50 and credit_history > 1.50 and other_debtors <= 1.50.	2	0,43	0,70	1,39	0,21	1,65	10325,80
credit_amount <= 10918.00 and telephone <= 0.50 and checking_account <= 2.50 and employment_duration <= 3.50 and dependents <= 1.50	2	0,40	0,74	1,48	0,20	1,92	10398,96
checking_account <= 2.50 and duration > 10.50 and other_debtors <= 1.50 and employment_duration <= 3.50 and telephone <= 0.50.	2	0,37	0,78	1,56	0,19	2,29	10409,41
employment_duration > 1.50 and purpose <= 8.50 and duration <= 25.50 and credit_history > 1.50 and credit_amount <= 7452.00.	1	0,40	0,69	1,38	0,20	1,61	10628,88
duration <= 27.50 and housing > 0.50 and employment_duration > 1.50 and credit_history > 1.50 and other_installment_plans > 1.50.	1	0,29	0,78	1,55	0,15	2,24	11339,56
duration <= 24.50 and savings_account <= 3.50 and employment_duration > 1.50 and credit_amount <= 7881.00 and duration > 7.00.	1	0,39	0,59	1,17	0,19	1,21	11381,37
other_installment_plans > 1.50 and employment_duration > 1.50 and credit_history > 1.50 and credit_amount <= 3897.50 and housing > 0.50.	1	0,28	0,79	1,58	0,14	2,37	11444,08

Tablo 4.2 Children of the Tree algoritması sonuçları: kurallar, metrikler ve sınıflar (devamı)[6]

personal_status_sex > 1.50 and duration > 8.50 and duration <= 24.50 and credit_amount > 1081.00 and credit_amount <= 7521.00.	1	0,3	0,66	1,33	0,15	1,49	11768,06
credit_history > 1.50 and foreign_worker <= 0.50 and other_installment_plans > 1.50 and credit_amount <= 3916.00 and number_credits <= 1.50.	1	0,33	0,59	1,18	0,17	1,23	11788,97
savings_account <= 2.50 and duration <= 24.50 and employment_duration > 1.50 and duration > 7.50 and credit_history > 1.50	1	0,31	0,63	1,27	0,15	1,37	11830,77
checking_account > 2.50 and age > 24.50 and duration <= 45.00 and credit_amount <= 9569.00 and other_installment_plans > 1.50.	1	0,20	0,93	1,86	0,10	7,13	11914,38
other_installment_plans > 1.50 and personal_status_sex > 1.50 and credit_history > 1.50 and property <= 2.50 and savings_account <= 3.50.	1	0,27	0,70	1,40	0,13	1,66	11966,64

Yukarıdaki **Tablo 4.2'de**, "Children of the Tree" algoritmamız Alman kredi veri kümesine uygulanmış ve bankaların potansiyel müşterileri üzerinden kurallar çıkarılmıştır. Tablonun ilk sütununda çıkarılan kurallar yer almakta, ikinci sütunda ise bu kuralların ait olduğu sınıflar belirtilmektedir. Diğer bir deyişle, bir kural bankanın potansiyel bir müşterisine uygulanıyorsa, o müşterinin **riskli olup olmadığı** ya da **düşük riskli bir müşteri** olup olmadığı belirtilmektedir.

Tablo 4.2'de yer alan **support (destek)** değerleri, her bir kuralın veri kümesinde ne sıklıkla gerçekleştiğini, yani **yaygınlığını** göstermektedir. **Confidence (güven)** değerleri ise bu kuralların **doğruluk oranlarını**, yani **geçerliliklerini** yansıtır. Bilindiği üzere, **MDL Cost** (Minimum Açıklama Uzunluğu Maliyeti) alanı, tabloda sunulan her bir kuralın **sadelik** derecesini matematiksel olarak yansıtan bir değerdir[6]**.**

MDL maliyetleri, probleme ve veri kümesine özgüdür; bu nedenle farklı çalışmalar ve veri setlerinde gözlemlenen değerlerden **daha yüksek veya daha düşük** olabilir. Ancak bu çalışmada elde edilen en düşük MDL maliyeti, **Sınıf 2 (kötü sınıf)** için hesaplanmıştır ve şu kurala aittir: **"duration > 6.50 ve savings_account <= 2.50 ve checking_account <= 2.50 ve other_debtors <= 1.50 ve credit_history <= 3.50"**, bu kural için hesaplanan MDL maliyeti **9176.16** olarak bulunmuştur[6].

Sınıf 1 (iyi sınıf) için elde edilen en düşük MDL maliyeti ise **"employment_duration > 1.50 ve purpose <= 8.50 ve duration <= 25.50 ve credit_history > 1.50 ve credit_amount <= 7452.00"** kuralına aittir ve bu kuralın **MDL Cost** değeri **10628.88** olarak hesaplanmıştır[6].

5
SONUÇ

Bu tez çalışmasında, kredi riski sınıflandırmasına yönelik olarak çeşitli makine öğrenmesi algoritmaları test edilmiş ve açıklanabilirlik odaklı yeni bir algoritma olan "Children of the Tree" önerilmiştir. Lojistik Regresyon (LR), KNN, SVM, Karar Ağaçları (DT), Naive Bayes (NB) ve Rastgele Orman (RF) gibi yaygın sınıflandırma algoritmaları, aynı ön işleme adımlarına tabi tutulmuş ve SMOTE yöntemiyle dengelenmiş veri kümesi üzerinde değerlendirilmiştir. Bu sayede modeller arası karşılaştırma adil, tutarlı ve karşılaştırılabilir bir yaklaşımla gerçekleştirilmiştir.

Elde edilen sonuçlara göre, KNN ve LR algoritmaları da sırasıyla %81 doğruluk, 0.815 ve 0.81 F1 skoru, 0.89 ve 0.88 AUC değerleriyle yüksek performans sergilemiştir. SVM %80 doğruluk ve 0.90 AUC değeriyle özellikle ayrıştırıcı gücü açısından dikkat çekmiştir. NB ve DT ise sırasıyla %78 ve %75 doğruluk ile daha sınırlı sonuçlar üretmiştir. En yüksek doğruluk (Accuracy: 0.83, F1-Sınıf 1: 0.84, F1-Sınıf 2: 0.82) ve AUC (0.92) değerleri RF algoritmasından elde edilmiştir. Bu model, pozitif ve negatif sınıflar arasında dengeli bir sınıflandırma başarımı göstermiş ve çalışmanın en güçlü temel algoritması olmuştur. Bu güçlü temel üzerine kurulan ve açıklanabilir kural çıkarımını hedefleyen "Children of the Tree" algoritması ise hem daha basit kurallar oluşturabilme kapasitesi hem de yorumlanabilirlik açısından öne çıkmıştır.

Tez çalışmasında, kuralların açıklanabilirliğini ve sadeliğini artırmak amacıyla **Minimum Description Length (MDL)** ilkesi kullanılmıştır. MDL, bir kuralın **karmaşıklığını** (koşul sayısı) ve **hata maliyetini** (yanlış sınıflandırma sayısı) dikkate alarak tanım uzunluğunu en aza indirmeyi amaçlar. Bu yaklaşım sayesinde, yalnızca doğru olan değil, aynı zamanda daha **az karmaşık** ve daha **açıklanabilir**

kuralların teşvik edilmesi sağlanmaktadır. Bu çalışmada, **daha az koşula sahip kuralları önceliklendiren**, sadelik ölçütünü MDL maliyeti ile değerlendiren ve **dengesiz veri kümelerinde azınlık sınıfını dikkate alarak daha genelleyici kurallar üreten** yeni ve etkili bir algoritma tanıtılmaktadır. Bu yönüyle yöntemimiz, kural optimizasyonuna yönelik alternatif yaklaşımlardan ayrılmaktadır[6].

İlerleyen çalışmalarda, algoritmanın rakiplerinde olduğu gibi **daha kararlı kurallar üretebilme kapasitesinin artırılması**, buna ek olarak daha **yüksek doğruluk** ve **daha sade** genelleyici kurallar oluşturulması hedeflenmektedir. Ayrıca, belirli özellikler üzerinden kural çıkarımı yapılmasını sağlayan hesaplamaların da sisteme entegre edilmesi planlanmaktadır. Böylelikle, kullanıcıların veri seti içindeki en önemli değişkenleri belirleyerek, bu özelliklere odaklanan kurallar elde etmesi mümkün olacaktır[6].

Aşağıda sunulan **Tablo 5.1**, 1.2.2 numaralı "Modelden Kural Çıkarımı Yöntemleri" başlıklı bölümde incelenen **RuleFit** algoritması ile önerdiğimiz algoritmanın model başarımı açısından karşılaştırmasını içermektedir. RuleFit algoritması, kendi framework'ü içerisinde **tree_generator parametresi random_forest_classifier** olarak ayarlanarak çalıştırılmıştır. Her iki algoritma için de aynı **ön işleme adımları** uygulanmış, **aynı özellikler** modele girdi olarak verilmiş ve her iki veri kümesi **SMOTE** yöntemi ile dengelenmiştir. Ayrıca, her iki modelin RF yapılandırmaları da **aynı hiperparametreler** kullanılarak gerçekleştirilmiştir[6].

Elde edilen sonuçlara göre, "Children of the Tree" algoritması hem **Sınıf 2 (kötü)** hem de **Sınıf 1 (iyi)** için RuleFit'e kıyasla daha başarılı performans göstermiştir. **Tablo 5.2'deki** veriler, modelin daha **güvenilir tahminler** yapmasını ve **sınıflar arasında daha iyi ayrım** sağlamasını mümkün kıldığını göstermektedir. RuleFit algoritması, tıpkı önerdiğimiz algoritma gibi bir RF modeli üzerinde eğitim yapabilme kapasitesine sahip olduğundan, her iki yöntem aynı veri ön işleme adımları ve özellik seçimiyle çalıştırılarak **adil bir karşılaştırma** yapılması sağlanmıştır[6].

Tablo 5.1 RuleFit ve Children of the Tree algoritmalarının model performanslarının karşılaştırılması[6]

Algoritma	Accuracy	F1 Score (Sınıf 2)	F1 Score (Sınıf 1)
Rule Fit	0.74	0.75	0.73
Children of the Tree	0.83	**0.82**	**0.84**

Ardından, karşılaştırma **Anchors** algoritması ile devam ettirilmiştir. Bu sefer, Children of the Tree algoritmasında kullanılan aynı RF modeli üzerinde **Anchors algoritması ile çıkarılan kurallar** analiz edilmiştir. **Tablo 5.2**, Anchors algoritması tarafından elde edilen bazı kuralları ve bunların değerlendirme metriklerini göstermektedir. Anchors, kurallarını **precision (kesinlik)** ve **coverage (kapsama oranı)** metrikleri üzerinden değerlendirir. Precision, kuralın doğruluğunu gösterirken; coverage, kuralın veri kümesindeki ne kadar örneğe uygulandığını belirtir[6].

Tablo 5.2 incelendiğinde, **yüksek precision değerine sahip kuralların**, genellikle veri kümesinin oldukça küçük bir alt kümesine uygulandığı görülmektedir. Buna karşılık, Children of the Tree algoritması kuralları **daha geniş metriklerle değerlendirmektedir**. Her ne kadar doğrudan karşılaştırılabilir olmasalar da, Anchors'ta kullanılan **coverage** metriği Children of the Tree'deki **support (destek)** ile, **precision** metriği ise **confidence (güven)** ile anlam olarak karşılaştırılabilir[6].

Tablo 5.2 Anchors algoritması sonuçları (en yüksek precision değerlerine göre)[6]

Sınıf	Anchors	Precision	Coverage
2	checking_account = 0 AND personal_status_sex = 1 AND duration > 28.00	1.0	0.04
2	checking_account = 1 AND credit_amount > 4176.25 AND telephone = 0 AND property = 2 AND other_installment_plans = 2	1.0	0.01
1	employment_duration = 4 AND duration <= 19.00 AND age > 39.25 AND other_installment_plans = 2 AND 1361.75 < credit_amount <= 2319.50	1.0	0.01
1	duration <= 12.00 AND present_residence > 3.00 AND housing = 1 AND property = 0 AND credit_amount <= 2319.50	0.99	0.03

Children of the Tree algoritmasında elde edilen en yüksek **support** ve **confidence** değerleri sırasıyla **Sınıf 2 için 0.48 ve 0.78**, **Sınıf 1 için 0.40 ve 0.69** olarak gözlemlenmiştir. Anchors algoritmasında Sınıf 1'e ait Children of the Tree'ye benzer uzunlukta olan "employment_duration = 4 AND duration <= 19.00 AND age > 39.25 AND other_installment_plans = 2 AND 1361.75 < credit_amount <= 2319.50" kuralı için **precision** 1.0 iken, **coverage** değeri yalnızca 0.0104'tür. Bu da kuralın veri kümesinde **uygulanabilirliğinin oldukça düşük** olduğunu göstermektedir. Benzer şekilde, Sınıf 2'ye ait ve benzer uzunlukta olan bir başka Anchors kuralı, 1.0 precision ile birlikte yalnızca 0.0147 coverage sağlamıştır[6].

Aşağıda sunulan **Tablo 5.3'te**, Children of the Tree algoritmasının **en yüksek confidence** değerlerine göre sıralanmış bazı kural örneklerini göstermektedir. Eğer **confidence = precision** ve **support = coverage** olarak düşünülürse, Tablo 7 ile yapılan kıyaslama sonucunda Children of the Tree algoritmasının **daha uygulanabilir ve açıklanabilir kurallar ürettiği** görülmektedir[6].

Tablo 5.3 Children of the Tree algoritması sonuçları (en yüksek confidence değerlerine göre)[6]

Kural	Sınıf	Confidence	Support
age > 33.50 and purpose <= 4.50 and credit_amount <= 4814.00 and checking_account > 2.50 and duration <= 16.50	1	1.0	0.05
checking_account > 2.50 and duration <= 16.50 and other_installment_plans > 1.50 and credit_history > 3.50	1	1.0	0.04
housing <= 0.50 and checking_account <= 2.50 and present_residence <= 3.50 and dependents <= 1.50 and other_installment_plans <= 1.50	2	1.0	0.04
credit_history <= 3.50 and personal_status_sex <= 1.50 and credit_history <= 1.50 and savings_account <= 1.50 and installment_rate > 2.50	2	1.0	0.04

Children of the Tree algoritması, hem Anchors hem de RuleFit ile yapılan karşılaştırmalarda öne çıkmaktadır. Anchors algoritması her ne kadar **çok yüksek doğruluklu (precision) kurallar** üretse de, bu kuralların **veri kümesinde uygulanabilirlik oranı çok düşüktür**. RuleFit ise, aynı hiperparametreler ve ön işleme adımları altında daha düşük model performansı sergilemiştir. Buna karşılık, Children of the Tree algoritması, hem **confidence (güven)** hem de **support (destek)** arasında denge kurarak, her iki sınıf için de **daha yüksek F1 skorlarına** ulaşmıştır. Bu da özellikle **dengesiz veri kümelerinde** anlamlı ve ölçeklenebilir kurallar üretme gereksinimi olan uygulamalarda (örneğin kredi riski analizi) Children of the Tree algoritmasını **daha üstün bir alternatif haline getirmektedir**. MDL maliyeti ve destek gibi ölçütleri kullanan algoritma, **hem açıklanabilir hem de pratik** kurallar üretme yetkinliği ile mevcut yöntemlerden ayrışmaktadır[6].

5.1 Çalışmanın Sınırlılıkları

“Children of the Tree” algoritması, kural çıkarımı konusunda önemli katkılar sunmakla birlikte, bazı sınırlılıklar da barındırmakta ve bu sınırlıklar gelecekteki geliştirmelere yön verebilir. Algoritma, Alman kredi veri kümesi üzerinde güçlü performans sergilemiş olsa da, farklı özelliklere sahip veri kümeleri üzerindeki **genellenebilirliği henüz test edilmemiştir**. Ayrıca, algoritmanın **kullanıcı tanımlı özelliklere veya alan bilgisini içeren önceliklendirmelere göre kuralları sıralama yetisi** bulunmamaktadır. Bu durum, algoritmanın gerçek dünya uygulamalarındaki kullanım esnekliğini sınırlayabilir. Son olarak, bu çalışmada sadece **Anchors** ve **RuleFit** ile karşılaştırma yapılmıştır; gelecekteki çalışmalarda, algoritmanın **diğer güncel kural çıkarım ve açıklanabilir yapay zekâ yöntemleriyle** de karşılaştırılarak avantajlarının ve geliştirilmesi gereken yönlerinin daha derinlemesine analiz edilmesi amaçlanmaktadır[6].

KAYNAKÇA

[1] S. Lessmann, B. Baesens, H. V. Seow, and L. C. Thomas, "Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research," *European Journal of Operational Research*, vol. 247, no. 1, pp. 124–136, 2015. Doi : 10.1016/j.ejor.2015.05.030.

[2] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint*, arXiv:1702.08608, 2017. Doi : 10.48550/arXiv.1702.08608.

[3] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017. Doi : 10.48550/arXiv.1705.07874.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144. Doi : 10.1145/2939672.2939778.

[5] N. Barakat and A. P. Bradley, "Rule extraction from support vector machines: A review," *Neurocomputing*, vol. 74, no. 1–3, pp. 178–190, 2010. Doi : 10.1016/j.neucom.2010.08.013.

[6] H. Meydan and M. Bal, "Children of the Tree: Optimised rule extraction from machine learning models," *Journal of Digital Data & Artificial Intelligence*, vol. 1, no. 1, pp. 14–35, 2025. Doi : 10.26650/d3ai.1606958.

[7] Statlog (German Credit Data), UCI Machine Learning Repository. [Online]. Available: https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data. [Accessed: Nov. 15, 2022]

[8] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. Doi : 10.1023/A:1010933404324.

[9] D. Martens, J. Vanthienen, W. Verbeke, and B. Baesens, "Performance of classification models from a user perspective,"*Decision Support Systems*, vol. 51, no. 4, pp. 782–793, 2011. Doi : 10.1016/j.dss.2011.01.007.

[10] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," arXiv preprint arXiv:1702.08608, 2017. Doi : 10.48550/arXiv.1702.08608.

[11] P. M. Addo, D. Guegan, and B. Hassani, "Credit risk analysis using machine and deep learning models," *Risks*, vol. 6, no. 2, pp. 1–20, 2018. Doi : 10.3390/risks6020038.

[12] M. Mahbobi, S. Kimiagari and M. Vasudevan, "Credit Risk Classification: An Integrated Predictive Accuracy Algorithm Using Artificial and Deep Neural Networks," *Annals of Operations Research*, vol. 330, pp. 609–637, 2023. Doi : 10.1007/s10479-021-04114-z.

[13] O. Koç, Ö. Uğur, and A. S. Kestel, "The impact of feature selection and transformation on machine learning methods in determining the credit scoring," arXiv preprint arXiv:2303.05427, Mar. 2023. Doi : 10.48550/arXiv.2303.05427.

[14] C.-F. Tsai and M.-L. Chen, "Credit rating by hybrid machine learning techniques," *Applied Soft Computing*, vol. 10, no. 2, pp. 374–380, Mar. 2010. Doi : 10.1016/j.asoc.2009.08.003.

[15] C. Yeh and C.-H. Lien, "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2473–2480, Mar. 2009. Doi : 10.1016/j.eswa.2007.12.020.

[16] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996. Doi : 10.1007/BF00058655.

[17] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990. Doi : 10.1007/BF00116037.

[18] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001. Doi : 10.1214/aos/1013203451.

[19] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992. Doi : 10.1016/S0893-6080(05)80023-1.

[20] R. Kurniawan, "Application of random forest algorithm on credit risk analysis," *Procedia Computer Science*, vol. 245, pp. 740–749, 2024. doi: 10.1016/j.procs.2024.10.300.

[21] B. Han, "Evaluating machine learning techniques for credit risk management: An algorithmic comparison," in *Proceedings of the 5th International Conference on Signal Processing and Machine Learning*, 2024. Doi : 10.54254/2755-2721/112/20251785.

[22] X. Gao, S. Alam, P. Shi, F. Dexter, and N. Kong, "Interpretable machine learning models for hospital readmission prediction: a two-step extracted regression tree approach," *BMC Medical Informatics and Decision Making*, vol. 23, no. 104, 2023. Doi : 10.1186/s12911-023-02193-5.

[23] B. H. Misheva, A. Hirsa, J. Osterrieder, O. Kulkarni, and S. F. Lin, "Explainable AI in credit risk management," arXiv preprint arXiv:2103.00949, Mar. 2021. Doi : 10.48550/arXiv.2103.00949.

[24] H. Deng, "Interpreting tree ensembles with inTrees," *International Journal of Data Science and Analytics*, vol. 7, no. 4, pp. 277–287, Dec. 2019. Doi : 10.1007/s41060-018-0144-8.

[25] L. T. K. Phung, V. T. N. Chau, and H. P. Nguyen, "Extracting rule RF in educational data classification: From a random forest to interpretable refined rules," in *Proceedings of the 2015 International Conference on Advanced*

Computing and Applications (ACOMP). 2015, pp. 20–27. Doi : 10.1109/ACOMP.2015.13.

[26] C. Bénard, G. Biau, S. da Veiga, and E. Scornet, "SIRUS: Stable and interpretable rule set for classification," arXiv preprint arXiv:1908.06852, 2019. Doi : 10.48550/arXiv.1908.06852.

[27] M. Mashayekhi and R. Gras, "Rule extraction from random forest: The RF+HC methods," in *Advances in Artificial Intelligence*, T. Kanade, J. Kittler, and J. M. Kleinberg, Eds. Berlin, Germany: Springer, 2015, vol. 3060, pp. 223–237. Doi : 10.1007/978-3-319-18356-5_20.

[28] S. Hara and K. Hayashi, "Making tree ensembles interpretable: A Bayesian model selection approach," arXiv preprint arXiv:1606.09066, 2017. Doi : 10.48550/arXiv.1606.09066.

[29] M. N. Adnan and M. Z. Islam, "ForEx++: A new framework for knowledge discovery from decision forests," *Australasian Journal of Information Systems*, vol. 21, Nov. 2017. Doi : 10.3127/ajis.v21i0.1539.

[30] S. I. Birbil, M. Edali, and B. Yuceoglu, "Rule covering for interpretation and boosting," arXiv preprint arXiv:2007.06379, 2020. Doi : 10.48550/arXiv.2007.06379.

[31] G. Zhang, Z. Hou, Y. Huang, J. Shi, H. Bride, J. S. Dong, and Y. Gao, "Extracting optimal explanations for ensemble trees via logical reasoning," arXiv preprint arXiv:2103.02191, 2021. Doi : 10.48550/arXiv.2103.02191.

[32] J. H. Friedman and B. E. Popescu, "Predictive learning via rule ensembles," *Annals of Applied Statistics*, vol. 2, no. 3, pp. 916–954, Sep. 2008. Doi : 10.1214/07-AOAS148.

[33] N. Meinshausen, "Node harvest," *Annals of Applied Statistics*, vol. 4, no. 4, pp. 2049–2072, Dec. 2010. Doi : 10.1214/10-AOAS367.

[34] M. Haddouchi and A. Berrado, "Forest-ORE: Mining optimal rule ensemble to interpret random forest models," arXiv preprint arXiv:2403.17588, 2024. Doi : 10.48550/arXiv.2403.17588.

[35] M. Duma, B. Twala, and T. Marwala, "Improving the performance of the RIPPER in insurance risk classification: A comparative study using feature selection," arXiv preprint arXiv:1108.4551, 2011. Doi : 10.48550/arXiv.1108.4551.

[36] P. Clark and T. Niblett, "The CN2 induction algorithm," *Machine Learning*, vol. 3, pp. 261–283, 1989. Doi : 10.1007/BF00116835.

[37] E. Frank and I. H. Witten, "Generating accurate rule sets without global optimization," in *Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98)*, 1998, pp. 144–151.

[38] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model," *Annals of Applied Statistics*, vol. 9, no. 3, pp. 1350–1371, 2015. Doi : 10.1214/15-AOAS848.

[39] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," in *Proc. 32nd AAAI Conf. Artificial Intelligence (AAAI)*, 2018, pp. 1527–1535.

[40] S. L. Salzberg, "C4.5: Programs for machine learning by J. Ross Quinlan," *Machine Learning*, vol. 16, pp. 235–240, 1994. Doi : 10.1007/BF00993309.

[41] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*, Belmont, CA: Wadsworth Int. Group, 1984. Doi : 10.1201/9781315139470.

[42] W. W. Cohen and Y. Singer, "A simple, fast, and effective rule learner," in *Proc. 16th Nat. Conf. Artificial Intelligence (AAAI/IAAI),* 1999, pp. 335–342.

[43] Z. Wang, W. Zhang, N. Liu, and J. Wang, "Scalable rule-based representation learning for interpretable classification," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[44] H. Lakkaraju, S. H. Bach, and J. Leskovec, "Interpretable decision sets: A joint framework for description and prediction," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD*), 2016, pp. 1675–1684. Doi : 10.1145/2939672.2939874.

[45] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "Interpretml: A unified framework for machine learning interpretability," arXiv preprint arXiv:1909.09223, 2019. Doi : 10.48550/arXiv.1909.09223.

[46] G. R. Lal, X. Chen, and V. Mithal, "TE2Rules: Explaining tree ensembles using rules," arXiv preprint arXiv:2206.14359, 2022. Doi : 10.48550/arXiv.2206.14359.

[47] P. D. Grünwald, *The minimum description length principle*, Cambridge, MA: MIT Press, 2007. Doi : 10.7551/mitpress/4643.001.0001.

[48] E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, and C. Rudin, "Learning certifiably optimal rule lists for categorical data," *Journal of Machine Learning Research*, vol. 18, no. 234, pp. 1–78, 2017. Doi : 10.48550/arXiv.1704.01701.

[49] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002. Doi : 10.1613/jair.953.

[50] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1993, pp. 207–216.

[51] D. R. Cox, "The analysis of binary data," *Journal of the Royal Statistical Society: Series B*, vol. 20, pp. 215–242, 1958.

[52] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, 3rd ed., Hoboken, NJ: Wiley, 2013.

[53] C. M. Bishop, *Pattern recognition and machine learning*, New York: Springer, 2006.

[54] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2nd ed., New York: Springer, 2009.

[55] J. A. Nelder and R. W. M. Wedderburn, "Generalized linear model," *J. R. Stat. Soc. Series A (General*), vol. 135, no. 3, pp. 370–384, 1972.

[56] G. A. F. Seber and A. J. Lee, *Linear regression analysis*, 2nd ed., Hoboken, NJ: Wiley-Interscience, 2003.

[57] "Linear regression illustration," *Wikimedia Commons*, https://upload.wikimedia.org/wikipedia/commons/thumb/4/41/LinearRegression.svg/500px-LinearRegression.svg.png, Date of access: May 1, 2025.

[58] A. D. Weigman, "Fault-tolerant sensor fusion for aircraft height estimation," MSc thesis, Stellenbosch Univ., Cape Town, 2019.

[59] A. Gelman et al., *Bayesian data analysis*, 3rd ed., Boca Raton, FL: CRC Press, 2013.

[60] D. Wackerly, W. Mendenhall, and R. L. Scheaffer, *Mathematical statistics with applications*, 7th ed., Duxbury, 2007.

[61] "Linear regression illustration 1," *Medium*, https://miro.medium.com/max/537/1*3RbN-F3W7hURvTzsNR3OuQ.png, Date of access: Dec. 1, 2020.

[62] "Linear regression illustration 2," *Medium*, https://miro.medium.com/max/700/1*mWD7FHIl15y-iSKw3MWb6A.png, Date of access: Dec. 1, 2020.

[63] "Linear regression illustration 3," *Medium*, https://miro.medium.com/max/691/1*lPCKC5-8ktYmspjBPfo44g.png, Date of access: Dec. 1, 2020.

[64] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967. Doi : 10.1109/TIT.1967.1053964.

[65] D. Barber, *Bayesian reasoning and machine learning*, Cambridge Univ. Press, 2012.

[66] D. T. Larose, *Data mining: Methods and models*, Hoboken, NJ: John Wiley & Sons, 2006.

[67] Y. Özkaya, *Veri madenciliği yöntemleri*, İstanbul: Papatya Bilim Yayınevi, 2016.

[68] J. Han and M. Kamber, *Data mining: Concepts and techniques*, Burlington, MA: Morgan Kaufmann, 2006.

[69] J. R. Quinlan, *C4.5: Programs for machine learning*, Burlington, MA: Morgan Kaufmann, 1993.

[70] M. Kantardzic, *Data mining: Concepts, methods and algorithms*, Hoboken, NJ: John Wiley & Sons, 2003.

[71] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. Doi : 10.1007/BF00994018.

[72] S. Metlek and K. Kıyas, "Makine öğrenmesinde, teoriden örnek MATLAB uygulamalarına kadar destek vektör makineleri," Iksad Publications, p. 22, 2020.

[73] S. Osowski et al., "MLP and SVM networks – a comparative study," in *Proc. 6th Nordic Signal Process. Symp. (NORSIG),* 2004.

[74] I. Nitze, U. Schulthess, and H. Asche, "Comparison of machine learning algorithms: Random forest, artificial neural network and support vector machine to maximum likelihood for supervised crop type classification," in *Proc. 4th GEOBIA*, Jan., Brazil, 2012, pp. 35–40.

[75] S. Ayhan and E. Şenol, "Destek vektör makineleriyle sınıflandırma problemlerinin çözümü için çekirdek fonksiyonu seçimi," *Eskişehir Osmangazi Üniversitesi İİBF Dergisi*, vol. 9, no. 1, pp. 175–198, Apr. 2014.

[76] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[77] I. Carmichael and J. S. Marron, "Geometric insights into support vector machine behavior using the KKT conditions," arXiv preprint arXiv:1704.00767, 2017. Doi : 10.48550/arXiv.1704.00767.

[78] S. Li, H. Li, M. Li, Y. Shyr, L. Xie, and Y. Li, "Improved prediction of lysine acetylation by support vector machines," *Protein and Peptide Letters*, vol. 16, pp. 977–983, 2009.

[79] A. Cutler and G. Zhao, "PERT – perfect random tree ensembles," unpublished manuscript, 2001.

[80] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Computation*, vol. 9, no. 7, pp. 1545–1588, 1997.

[81] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Record*, vol. 22, no. 2, pp. 207–216, Jun. 1993. Doi : 10.1145/170036.170072.

[82] E. V. Altay and B. Alataş, "Nicel birliktelik kural madenciliği için baskın olmayan sıralama genetik algoritma-II'nin duyarlılık analizi," *Bilişim Teknolojileri Dergisi*, vol. 13, Jan. 2020.

[83] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to data mining*, Boston, MA: Pearson Addison Wesley, 2005.

[84] J. Rissanen, "Modeling by the shortest data description," *Automatica*, vol. 14, pp. 465–471, 1978.

[85] T. Roos, "Minimum description length principle," in *Encyclopedia of Machine Learning and Data Mining*, C. Sammut and G. I. Webb, Eds., Springer, 2017, pp. 823–827. Doi : 10.1007/978-1-4899-7687-1_894.

[86] J. Rissanen, *Stochastic complexity and statistical inquiry*, Singapore: World Scientific, 1989.

[87] A. R. Barron, J. Rissanen, and B. Yu, "The minimum description length principle in coding and modeling," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2743–2760, 1998. Doi : 10.1109/18.720554.

[88] M. Li and P. Vitányi, *An introduction to Kolmogorov complexity and its applications,* 3rd ed., Springer, 2019. ISBN: 978-3-030-11297-4.

[89] P. Grünwald and T. Roos, “Minimum description length revisited,” *Int. J. Mathematics for Industry*, vol. 11, no. 1, 1930001, 2019. Doi : 10.1142/S2661335219300018.

[90] P. D. Grünwald, *The minimum description length principle*, Cambridge, MA: MIT Press, 2007. Doi : 10.7551/mitpress/4643.001.0001.

[91] J. Vreeken, M. van Leeuwen, and A. Siebes, “Krimp: Mining itemsets that compress,” *Data Mining and Knowledge Discovery*, vol. 23, no. 1, pp. 169–214, 2011.

[92] B. Ustun and C. Rudin, “Supersparse linear integer models for optimized medical scoring systems,” *Machine Learning*, vol. 102, pp. 349–391, 2016.

[93] M. Adnan, M. Z. Islam, and H. A. Abbass, “A novel method for selecting interesting association rules using MDL principle,” in *Proc. AI 2012: Advances in Artificial Intelligence*, Springer, 2012, pp. 65–76.

TEZDEN ÜRETİLMİŞ YAYINLAR

Makaleler

1. H. Meydan and M. Bal, “Children of the Tree: Optimised rule extraction from machine learning models,” *Journal of Digital Data & Artificial Intelligence*, vol. 1, no. 1, pp. 14–35, 2025. Doi : 10.26650/d3ai.1606958.