T.C.
İSTANBUL SABAHATTİN ZAİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ BİLİM DALI

KARA PARA AKLAMANIN ÖNLENMESİ İÇİN DERİN ÖĞRENME

DOKTORA TEZİ

Mert Yılmaz ÇAKIR

İstanbul
Ocak-2024

T.C.
İSTANBUL SABAHATTİN ZAİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ BİLİM DALI

KARA PARA AKLAMANIN ÖNLENMESİ İÇİN DERİN ÖĞRENME

DOKTORA TEZİ

Mert Yılmaz ÇAKIR

Tez Danışmanı
Dr. Yahya ŞİRİN

İstanbul
Ocak-2024

TEZ ONAYI

Lisansüstü Eğitim Enstitüsü Müdürlüğüne,

Bu çalışma, jürimiz tarafından Bilgisayar Mühendisliği Anabilim Dalı, Bilgisayar Bilimleri ve Mühendisliği Bilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman Dr. Öğr. Üyesi Yahya ŞİRİN

Üye Dr. Öğr. Üyesi Mehtap YALÇINKAYA

Üye Dr. Öğr. Üyesi Abdullah SÖNMEZ

Üye Dr. Öğr. Üyesi Şengül BAYRAK HAYTA

Üye Dr. Öğr. Üyesi Sümeyra BEDİR

Onay

Yukarıdaki imzaların, adı geçen öğretim üyelerine ait olduğunu onaylarım.

Prof. Dr. Erhan İÇENER

Enstitü Müdürü

BİLİMSEL ETİK BİLDİRİMİ

Doktora tezi olarak hazırladığım **"Kara Para Aklamanın Önlenmesi İçin Derin Öğrenme"** adlı çalışmanın öneri aşamasından sonuçlandığı aşamaya kadar geçen süreçte bilimsel etiğe ve akademik kurallara özenle uyduğumu, tez içindeki tüm bilgileri bilimsel ahlak ve gelenek çerçevesinde elde ettiğimi, tez yazım kurallarına uygun olarak hazırladığımı, bu çalışmamda doğrudan veya dolaylı olarak yaptığım her alıntıya kaynak gösterdiğimi ve yararlandığım eserlerin kaynakçada gösterilenlerden oluştuğunu beyan ederim.

Mert Yılmaz ÇAKIR

ÖN SÖZ

Bu tez çalışmasında öncelikle benden desteklerini esirgemeyen her zaman yanımda olan anneme, babama, kardeşlerime, eşime, değerli önerileri ve her türlü yardımlarıyla beni yönlendiren kıymetli danışmanım Dr. Yahya ŞİRİN'e, hocalarım Dr. Mehtap YALÇINKAYA'ya ve Dr. Abdullah SÖNMEZ'e, bölüm arkadaşım Dr. Mehmet Ali KUTLUGÜN'e, moral ve desteklerinden dolayı yakınlarıma ve sağladığı burs için İstanbul Sabahattin Zaim Üniversite'sine gönülden teşekkür ederim.

Mert Yılmaz ÇAKIR
İstanbul - 2024

ÖZET

KARA PARA AKLAMANIN ÖNLENMESİ İÇİN DERİN ÖĞRENME

Mert Yılmaz ÇAKIR

Doktora, Bilgisayar Bilimleri ve Mühendisliği

Tez Danışmanı: Dr. Yahya ŞİRİN

Ocak-2024, 108 +XV Sayfa

Bu tez, finansal suçlar aracılığıyla gerçekleşen kara para aklama sorununu ele almak ve özellikle kredi kartı dolandırıcılığı gibi suçların tespitindeki sınıf dengesizliği zorluğunu çözmek amacıyla derin öğrenme yaklaşımlarını araştırmaktadır. Dolandırıcılığı tespit etmedeki ana zorluk, normal ve sahte örnekler arasındaki doğal sınıf dengesizliğinden kaynaklanmaktadır. Bu sorunu çözmek adına otokodlayıcı tabanlı Gürültü Faktörü Kodlama yöntemi ile sentetik azınlık üst örnekleme tekniği birleştirilerek yeni bir sınıf dengeleme önerimi sunulmaktadır. Farklı veri kümelerinde çeşitli otokodlayıcı varyantları üzerinde gerçekleştirilen deneyler, önerilen yaklaşımın geleneksel üst örnekleme yöntemlerine kıyasla daha etkili olduğunu göstermektedir. Ayrıca çalışma, dolandırıcılık tespiti için çeşitli sınıflandırma yöntemlerini karşılaştırarak, çok aşamalı derin öğrenme modellerinin genel performans üzerindeki olumlu etkilerini ortaya koymaktadır. Otokodlayıcı tabanlı Gürültü Faktörü Kodlama yaklaşımının, sınıf dengesizliği sorununu başarılı bir şekilde ele alarak yüksek doğruluk, geri çağırma, özgüllük, hassasiyet, F_1 puanı, AUC-ROC ve MCC değerlerinde etkileyici sonuçlar elde ettiği görülmektedir. Ancak, bu yaklaşımın performansındaki veri kümesine bağlı değişkenlik, gelecekteki çalışmalarda model parametrelerinin daha etkin bir şekilde optimize edilmesi ve genelleme yeteneklerinin artırılması açısından dikkate alınmalıdır.

Anahtar Kelimeler: Kara para aklama, derin öğrenme, sınıf dengesizliği, finansal suçlar, dolandırıcılık tespiti

ABSTRACT

DEEP LEARNING FOR ANTI MONEY LAUNDERING

Mert Yılmaz ÇAKIR

Ph. D. Computer Sciences and Engineering

Supervisor: Dr. Yahya ŞİRİN

January-2024, 108 +XV Pages

This thesis investigates deep learning approaches to address the issue of money laundering, particularly in the detection of crimes such as credit card fraud, within the context of financial crimes. The primary challenge in fraud detection stems from the natural class imbalance between genuine and fraudulent examples. To tackle this issue, a novel proposal is presented by combining the autoencoder-based Noise Factor Encoding method with synthetic minority oversampling technique. Experiments conducted on various datasets with different autoencoder variants demonstrate the effectiveness of the proposed approach compared to traditional oversampling methods. Furthermore, the study compares various classification methods for fraud detection, highlighting the positive impact of multi-stage deep learning models on overall performance. The autoencoder-based Noise Factor Encoding approach successfully addresses the class imbalance problem, yielding impressive results in terms of high accuracy, recall, specificity, precision, F_1 score, AUC-ROC, and MCC values. However, the variability in performance based on dataset characteristics emphasizes the need for more effective optimization of model parameters and enhanced generalization capabilities in future research endeavors.

Keywords: Money laundering, deep learning, class imbalance, financial crimes, fraud detection

İÇİNDEKİLER

ÜÇÜNCÜ BÖLÜM

TABLO LİSTESİ

ŞEKİL LİSTESİ

SEMBOLLER LİSTESİ

α	: Önerilen yaklaşım için gürültü faktörü
δ	: SMOTE için yapay katsayı olmak üzere (0,1) arasında rastgele bir sayı
D_{KL}	: VAE için Kullback-Leibler sapması
J_E	: CAE için Jacobian matrisi
λ	: CAE için Jacobian matrisinin Frobenius normu
μ	: Gaussian dağılımı için ortalama değeri
$\mathcal{L}$	: AE, CAE ve VAE için kayıp fonksiyonu
$\mathcal{N}$	: Önerilen yaklaşım için Gaussian dağılımı
σ	: Gaussian dağılımı için standart sapma değeri
w	: ADASYN için ağırlık hesabı

KISALTMALAR LİSTESİ

ACC : Doğruluk (Accuracy)

ADASYN : Uyarlanabilir Sentetik Örnekleme (Adaptive Synthetic Sampling)

AE : Otokodlayıcı (Autoencoder)

AI : Yapay Zekâ (Artificial Intelligence)

AML : Kara Para Aklamayı Önleme (Anti Money Laundering)

ANN : Yapay Sinir Ağları (Artificial Neural Networks)

AUC : Area Under the Receiver Operating Characteristic Curve (AUC-ROC)

CAE : Büzülmeli Otokodlayıcı (Contractive Autoencoder)

CCF : Kredi Kartı Sahtekârlığı (Credit Card Fraud)

CCU : Küme Merkezli Alt Örnekleme (Cluster Centroid Undersampling)

CIP : Sınıf Dengesizliği Problemi (Class Imbalanced Problem)

CNN : Evrişimsel Sinir Ağları (Convolutional Neural Networks)

CSL : Maliyet Duyarlı Öğrenme (Cost-Sensitive Learning)

DL : Derin Öğrenme (Deep Learning)

DNN : Derin Sinir Ağları (Deep Neural Networks)

DT : Karar Ağaçları (Decision Tree)

EDA : Keşifsel Veri Analizi (Exploratory Data Analysis)

F_1 : F_1 score

FATF : KPA Önlenmesinde Mali Eylem (Financial Action Task Force on ML)

GAN : Çekişmeli Üretici Ağ (Generative Adversarial Network)

KPA : Kara Para Aklama (Money Laundering)

MCC : Matthew's Korelasyon Katsayısı (Matthew's Correlation Coefficient)

ML : Makine Öğrenmesi (Machine Learning)

NFE : Gürültü Faktörü Kodlaması (Noise Factor Encoding)

NMU : Ramak Kala Alt Örnekleme (Near Miss Undersampling)

P : Hassasiyet (Precision)

PCA : Temel Bileşen Analizi (Principal Component Analysis)

R : Duyarlılık (Recall/Sensitivity)

ReLU : Düzeltilmiş Lineer Ünite (Rectified Linear Unit)

RF : Rastgele Orman (Random Forest)

ROS : Rassal Üst Örnekleme (Random Oversampling)

RUS : Rassal Alt Örnekleme (Random Undersampling)

SMOTE : Sentetik Azınlık Üst Örn. Tekniği (Synthetic Minority OS Technique)

SMOTEN : Sembolik SMOTE (SMOTE for Nominal)

SVM : Desktek Vektör Makineleri (Support Vector Machines)

SP : Özgüllük (Specificity)

VAE : Değişken Otokodlayıcı (Variational Autoencoder)

BİRİNCİ BÖLÜM

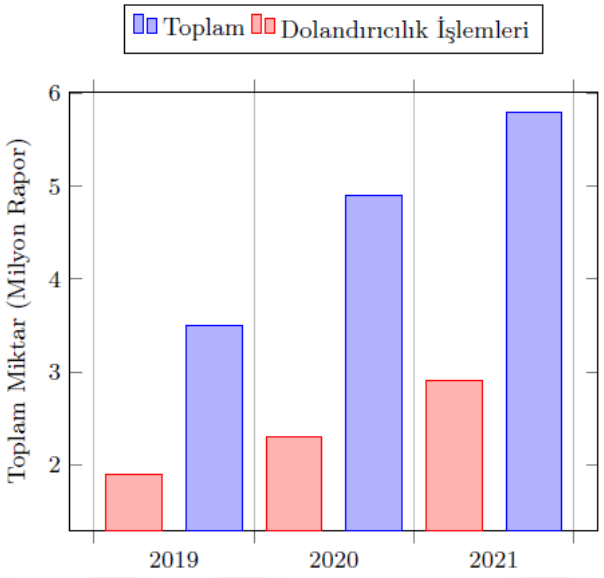
GİRİŞ

Kara para aklama (KPA), suç faaliyetlerinden elde edilen parayı, görünüşte meşru ve temiz bir kaynaktan gelmiş gibi gösterme sürecini ifade eder (Korejo et al., 2021). Bu illegal uygulama genellikle insan ticareti, vergi kaçakçılığı, yasadışı kumar, terörizm ve hırsızlık gibi suçlardan elde edilen fonları içerir. KPA, küresel ekonominin yaklaşık %2- 5'ini oluşturan veya mevcut Amerikan doları cinsinden 800 milyar ila 2 trilyon dolar arasında bir büyüklüğe sahip olan önemli bir sorun olarak kabul edilmektedir (Nations, 2022). KPA ile mücadele, uluslararası iş birliği, etkili finansal takip sistemleri ve güçlü yasal düzenlemeler gerektirir. Bu hem finansal kurumlar hem de devletlerarası çeşitli önlemleri içermelidir. Ayrıca, KPA tespiti (AML) ve önleme konusundaki teknolojik gelişmelerin izlenmesi ve bu alandaki uzmanlığın sürekli olarak güncellenmesi de kritik öneme sahiptir. KPA için birçok yöntem bulunmaktadır. KPA ve mali dolandırıcılıkla birlikte mücadele ederken bunları çözmenin çok az yöntemi vardır (Goecks et al., 2022). Bu nedenle, bu çalışma finansal dolandırıcılık tespitine yönelik yöntemleri belirlemeyi amaçlamaktadır.

Finansal dolandırıcılık yöntemlerinden birisi olan kredi kartı dolandırıcılığı (CCF), sahte veya çalıntı kredi kartı bilgilerini kullanarak, genellikle çevrimiçi alışverişlerde veya diğer finansal işlemlerde, meşru gibi görünen ancak aslında illegal olan fon transferlerini gerçekleştirmeyi içerir (Bin Sulaiman et al., 2022). Bu tür dolandırıcılık yöntemleri, teknolojik gelişmelerle birlikte daha gelişmiştir ve finansal sistemlere zarar verme potansiyeli taşımaktadır. Son on yıl içerisinde yaygınlaşan e-ticaret ile kredi kartı kullanımı önemli ölçüde artarak hileli işlemlerde yükselişi beraberinde getirmiştir (Randhawa et al., 2018). Ayrıca, bu yükselişle birlikte çevrimiçi ticaret kurumları için ekonomik kayıp ve müşteri güvenini kaybetme riski bulunmaktadır. Bu sebepler ile dolandırıcılık yönetimi; sigorta talepleri, KPA, elektronik ödemeler, banka işlemleri vb. gibi birçok alan için temel bir araştırma konusu olmuştur (Al-Hashedi & Magalingam, 2021).

2017 Yılında CCF yüzünden gerçekleşen yaklaşık 23,97 milyar dolarlık meblağ, 2018'de yüzde 16,2 oranında artış göstererek 27,85 milyar dolara ulaşmıştır. Bununla birlikte 2023 yılına kadar bu kaybın yıllık 35 milyar dolara ulaşacağı tahmin

edilmektedir (Tingfei et al., 2020). Şekil 1.1, 2021'de bildirilen 5,8 milyon tüketici raporunun (dolandırıcılık, kimlik hırsızlığı ve diğer tüketici koruma konuları) 2,9 milyonunun dolandırıcılıkla ilgili olduğunu ve bunların %25'inin kayıp para olduğunu göstermektedir[1]. Bu kayıpların, etkili dolandırıcılık önleme yoluyla azaltılması ve kullanıcı hareketlerinde mümkün olan en kısa sürede saptanması gerekir. Öte yandan finans sektörünün buradaki zorluğu, müşteri hareketlerini etkilemeden hileli işlemleri hızlı bir şekilde tespit edebilmektir.



Şekil 1.1: Tüketici Gözcü Ağı Raporunun Yıllık Dağılımı

Zamanında müdahalenin kritik olduğu dolandırıcılık tespitinde, sezgisel yöntemler veya kural tabanlı modeller geleneksel olarak sınırlı bir başarıyla kullanılmaktadır (Interceptd, 2022). Sezgisel yöntemler genellikle manuel çözümleri oluştururken, kural tabanlı yaklaşımlar karmaşık koşullarla ilgilenir. Ancak her iki yöntem de bir işlemin hileli olarak nitelendirilebilmesi için yeterli pratikliğe sahip değildir. Günümüzde ise dolandırıcılık tespit yaklaşımı gibi anomali tespit problemleri, hızlı aksiyon alabilmek için makine öğreniminden (ML) yararlanır (Salazar et al., 2018).

AML için ML (Gao et al., 2019; Yee et al., 2018) ve özellikle büyük hacimli veriler üzerinde daha karmaşık örüntüleri tanıyabilen derin öğrenme (DL) (Zioviris et al., 2022; Alarfaj et al., 2022; Esenogho et al., 2022) yöntemleri kullanılır. Bu alanda ML uygulanması, bankacılık kurumlarının işlemleri dolandırıcılık olaylarından gerçek

[1] https://www.ftc.gov/reports/consumer-sentinel-network-data-book-2021

zamanlı olarak ayırt etmesine olanak sağlar. Bunun yanı sıra ML, diğer yöntemlerden daha yüksek doğruluk sergileyen modellerin tanımlamasına yardımcı olur. ML, CCF problemleri için gözetimli, (Salazar et al., 2018; Carcillo et al., 2018), yarı gözetimli (Roy et al., 2018; Xuan et al., 2018) ve gözetimsiz öğrenme (Carcillo et al., 2021; Eshghi & Kargari, 2019) çözümleri sunar [46 Sayfadaki].

Dolandırıcılık tespiti, finans, siber güvenlik ve e-ticaret gibi kritik sektörlerde hayati bir rol oynamakta olup, gün geçtikçe artan işlem verileriyle birlikte giderek daha karmaşık bir görev haline gelmektedir (Abdallah et al., 2016). Bu önemli görev, dolandırıcılık faaliyetlerini tanımlamak ve engellemek amacıyla geliştirilen sistemler aracılığıyla gerçekleştirilir. Ancak, bu süreç, işlem verilerindeki hızlı büyüme ve çeşitlenme nedeniyle daha da karmaşık hale gelmektedir. Ayrıca, normal işlemlerle sahte işlemler arasındaki doğal sınıf dengesizliği problemi (CIP), bu sorunu daha da derinleştirmekte ve sahtecilikle mücadeleyi güçleştirmektedir.

CCF tespiti için kullanılan veri setleri genellikle, meşru veya normal işlemlerin sayısının sahte işlemlerden önemli ölçüde daha fazla olması nedeniyle dengesiz bir dağılıma sahiptir (Provost & Fawcett, 2001; Dal Pozzolo et al., 2015). Bu durum, modelin eğitildiği veri setinde negatif sınıfın (meşru işlemler) daha baskın olmasına ve pozitif sınıfın (sahte işlemler) sayısının göreceli olarak az olmasına yol açar. Bu dengesizlik, eğitilen modelin pozitif durumları gürültü olarak değerlendirmesine ve sınıflandırma sonuçlarının negatif sınıfa doğru yanlı bir sapma göstermesine neden olabilir (Patel et al., 2020). Bu durum, sahte işlemlerin doğru bir şekilde tespit edilmesini zorlaştırabilir ve modelin performansını olumsuz etkileyebilir. Bu nedenle, CCF tespiti için kullanılan modellerde dengesiz veri setleriyle başa çıkma stratejileri, örneğin üst örnekleme (oversampling) veya sınıf ağırlıkları kullanma gibi yöntemlerin uygulanması önemlidir [19 Sayfadaki]. Geleneksel ML teknikleri, bu özel sorunu etkili bir şekilde ele almakta zorlanmıştır (Fernando & Tsokos, 2021). Bu nedenle, CIP ile başa çıkabilen ve dolandırıcılık tespitinde daha etkili sonuçlar sağlayabilen özel tekniklerin ve algoritmaların kullanılması önemlidir.

Son yıllarda, araştırmacılar dolandırıcılık tespiti konusunda CIP etkilerini azaltmak amacıyla çeşitli yaklaşımları detaylı bir şekilde incelemişlerdir. Bu alanda yaygın olarak kullanılan stratejilerden biri, veri kümesindeki dengesizliği gidermek için azınlık sınıfa ait sentetik örnekler oluşturmaya odaklanmaktadır. Bu amaca hizmet eden yöntemler arasında öne çıkanlardan biri Sentetik Azınlık Üst Örnekleme Tekniği

(SMOTE) olarak bilinir (Feng et al., 2021). SMOTE, azınlık sınıfa ait örnekler arasında interpolasyon yaparak sentetik örnekler oluşturur ve bu şekilde veri setini daha dengeli hale getirmeye çalışır (Yi et al., 2020). Ancak, SMOTE'un vaat ettiği avantajlara rağmen, bazı önemli kısıtlamalara sahiptir. SMOTE, dolandırıcılık işlemlerinin içsel özelliklerini ve değişen karmaşıklıklarını yeterince içselleştirmeme eğilimindedir. Bu kısıtlama, dolandırıcılık altındaki desenleri doğru bir şekilde tanıma ve modelleme etkinliğini sınırlayabilir, bu da dolandırıcılık tespiti performansını olumsuz yönde etkileyebilir. Bu bağlamda, CIP ile mücadelede daha etkili ve kapsamlı yöntemlere odaklanan yeni araştırmaların önemli olduğu söylenebilir.

Ayrıca, dolandırıcılık tespiti araştırmaları, model geliştirme ve değerlendirmesi için sıklıkla kullanılan açık kaynak veri kaynaklarının kendi özel zorluklarını ve sınırlamalarını beraberinde getirir. Bu açık kaynak veri kümeleri, içerdikleri sınırlı çeşitlilik, tutarsız veri kalitesi, sahte örneklerin doğru etiketlenmesi konusundaki zorluklar ve dolandırıcılık yöntemlerinin güncel eğilimleri yansıtmada potansiyel gecikmeler gibi kısıtlamalardan etkilenebilir. Bu zorlukları göz önünde bulundurarak, dolandırıcılık tespit doğruluğunu artırmak amacıyla yeni bir yaklaşım öneriyoruz: Autoencoder (AE) tabanlı Gürültü Faktörü Kodlama (NFE) ile SMOTE yöntemlerinin birleştirilmesi. Otokodlayıcılar, boyut azaltma ve gözetimsiz öğrenme için kullanılan sinir ağı modelleridir ve sahte örneklerin karmaşık özelliklerini yakalama yeteneği sunarlar (Wang et al., 2014). Gerçek dolandırıcılık verileri üzerinde otokodlayıcıları eğitip, kodlama süreci sırasında gürültü faktörü ekleyerek, dolandırıcılığı daha etkili bir şekilde ayırt etme yeteneğini artırmayı hedefliyoruz. Bu, mevcut açık kaynak veri setlerinin zorluklarını aşmak ve daha güvenilir dolandırıcılık tespiti modelleri geliştirmek için özgün bir yaklaşım olarak öne çıkmaktadır.

Bu tez kapsamında, AE, Variational Autoencoder (VAE), ve Contractive Autoencoder (CAE) gibi üç farklı otokodlayıcı varyantının, NFE (Autoencoder tabanlı Gürültü Faktörü Kodlama) tekniği tarafından geliştirilen performansını değerlendirdik. Bu bağlamda NFE ile dolandırıcılık verilerinin öğrenilmiş temsillerini gerçek dolandırıcılık verileriyle birleştirerek, eğitim için daha zengin ve çeşitli bir veri kümesi oluşturduk. Daha sonra, bu birleştirilmiş veri kümesine CIP ile başa çıkmak amacıyla SMOTE tekniğini uyguladık. Önerilen yaklaşımın etkinliğini değerlendirmek için çeşitli değerlendirme metrikleri kullanıldık. Bu metrikler arasında Accuracy (ACC), F_1 score (F_1), Recall/Sensitivity (R), Specificity (SP), Precision (P), Area Under the

Receiver Operating Characteristic Curve (AUC-ROC) ve Matthew Correlation Coefficient (MCC) yer almaktadır (Chicco & Jurman, 2020). Bu kapsamlı metrik seti, modelin dolandırıcılık tespitindeki doğruluğunu geniş bir perspektiften değerlendirme imkânı sağlamaktadır. Bu çalışmanın sonuçları, dolandırıcılık tespiti alanında daha etkili ve güvenilir modellerin geliştirilmesine katkıda bulunabilir.

Bu makalenin geri kalanı aşağıdaki gibi yapılandırılmıştır: Bölüm 2, KPA konularındaki ilgili çalışmaların genel bir bakışını sunar. Bölüm 3 CIP üzerinde durulmaktadır. Bölüm 4 ML ve öğrenme türleri üzerinde bilgi vermektedir. Bölüm 5'te DL ilgili teknik bilgiler paylaşılmaktadır. Bölüm 6'da önerilen çalışma, Bölüm 7'de deneysel çalışma sunulmuştur. Bölüm 8'de çalışmanın sonuç ve değerlendirilmesi yapılmıştır ve dolandırıcılık tespiti doğruluğunu artırmada yaklaşımımızın önemi vurgulanmaktadır.

1.1. Tezin Amacı

Bu araştırmanın temel amacı, kara para aklamanın derin öğrenme teknikleri ile tespitinde geçmişten günümüze yapılan çalışmaları incelemek ve bu teknikleri daha etkili hale getirmek için verimli yöntemler ve öneriler sunmaktır. Bu hedef doğrultusunda, mevcut literatür detaylı bir şekilde taranmış, belirlenen tekniklerin avantajları ve eksikleri üzerinde ayrıntılı bir değerlendirme yapılmıştır. Bu değerlendirmeler sonucunda, önerilen bir sistem tasarlanmıştır.

1.2. Tezin Kapsamı

Bu çalışma, literatürde mevcut kara para aklamanın derin öğrenme teknikleri ile tespiti ve önerilen sistemlerin yapısal analizini içermekte olup, bu bağlamda kullanılan veri setlerini incelemektedir. Yapılan bu inceleme sonucunda, önerilen sistemin etkinliği deneysel bir çalışma ile değerlendirilmiştir.

1.3. Sınırlar

DL ile KPA tespiti çalışmaları birçok teknik, hukuki ve etik sınırlamayla karşı karşıyadır. Bu sınırlamalar şu şekilde sıralanabilir:

- **Veri Erişimi ve Gizlilik:** KPA tespiti için kullanılacak verilere erişim sınırlıdır. Bu tür verilere ulaşmak, yasal ve etik sorumluluklar gerektirmektedir. Veri gizliliği, özellikle finansal verilerle ilgili çalışmalarda

önemlidir. Bu kapsamda çalışmada erişilen üç farklı kredi kartı sahtekârlığı veri seti bulunmaktadır.

- **Hukuki Sınırlamalar:** KPA suçları ciddi bir hukuki boyuta sahiptir. Bu tür çalışmalar, yerel ve ulusal yasalarla uyumlu olmalıdır. Hukuki düzenlemelere riayet etmeyen çalışmalar, yasal sorunlara yol açabilir. Bu çalışmada düzenlemelere riayet edilmiştir.
- **Etik İkilemler:** KPA tespiti, özel bir hassasiyet gerektirir. Kişisel gizlilik hakları ile bu tür suçların tespiti arasında bir denge sağlamak önemlidir. Bu, etik bir ikilemi gündeme getirebilir. Bu çalışmada kullanılan veri setlerinde kişisel bilgiler anonimleştirilmiştir.
- **Veri Kalitesi:** DL modellerinin başarısı, veri kalitesine bağlıdır. Eğer veriler eksik, hatalı veya yanıltıcı ise, sonuçlar güvenilir olmayabilir. Bu, özellikle finansal verilerin temiz ve doğru olmasını gerektirir. Bu çalışmada birden çok veri seti kullanılarak bu durumun minimize edilmesi sağlanmıştır.
- **Bilgisayar Gücü ve Kaynaklar:** DL modelleri genellikle büyük hesaplama kaynakları gerektirir. Bu, maliyetleri artırabilir ve sınırlamalar getirebilir. Bu kapsamda önerimi yapılacak çalışmanın kullanılacağı gerçek dünya verilerinde bilgisayar gücü ve kaynaklarının iyileştirilmesi önerilmektedir.
- **Ölçekleme Zorlukları:** Gerçek dünya finansal verileri büyük ölçüde karmaşıktır ve çok büyük veri setleri içerebilir. Bu verileri işlemek ve analiz etmek zor olabilir. Çalışmada kullanılan veriler, bu verilere kıyasla daha azdır.
- **Hata Oranları:** DL modelleri, hata yapabilir ve yanlış pozitif veya yanlış negatif sonuçlar verebilir. Bu, finansal kuruluşlar ve yetkililer için yanıltıcı olabilir. Çalışmamızda bu veriler kabul gören performans metrikleri ile sunulmuştur.
- **Yetersiz Eğitim Verisi:** DL modelleri, yeterli ve temsilci eğitim verileri gerektirir. KPA vakalarının nadir olması, bu verilerin yetersiz olmasına yol açabilir. Çalışmamızda bu durumun iyileştirilmesi için sentetik veri üretilmesi önerilmektedir.
- **Öğrenme Süreçleri:** DL modellerinin eğitimi uzun sürebilir ve bazen öğrenme süreçleri belirli dönemlerde optimize edilmelidir. Gerçek dünya problemlerine

göre daha az bir veri seti ile çalışıldığından farklı veri setleri için öğrenme süreçlerinin optimize edilmesi gerekebilir.

- **İnsan Faktörü:** İnsan değerlendirmeleri ve uzman bilgisi, DL modellerinin sonuçlarını yorumlamada ve filtrelemede önemlidir. İnsan faktörü, otomatik tespit sistemlerinin sınırlamalarını etkileyebilir. Bu kapsamda çalışmamızda sadece teknik performans değerlendirilmesi yapılmıştır.

Özetlemek gerekirse, DL ile KPA tespiti çalışmaları birçok teknik, hukuki ve etik sınırlamayla karşılaşabilir. Bu çalışmaların güvenilirlik, gizlilik ve yasal uyum konularına dikkat etmeleri önemlidir. Ayrıca, insan faktörünün dikkate alınması ve DL modellerinin sınırlamalarının anlaşılması da kritik öneme sahiptir.

1.4. Varsayımlar

Bu çalışma, aşağıda belirtilen varsayımlar temel alınarak gerçekleştirilmiştir:

1. Çalışma kapsamında genel olarak sınırlı CCF verileri kullanılarak sistem eğitilmiştir. Sistemin eğitim seviyesine dayanarak, deneysel çalışma sürecinde yapılan testlerin yeterli olduğu kabul edilmiştir.
2. Literatür taraması sonucunda ortaya çıkan tekniklerin, tasarlanan sistemin KPA tespiti konusundaki yeterliliği kabul edilmiştir.

1.5. Katkılar

Çalışma kapsamında geliştirilen metodolojinin ve elde edilen sonuçların akademik literatüre sağladığı katkılar aşağıda sıralanmıştır.

1. Literatür Tarama: İncelenen hile tespiti çalışmaları için kullanılan teknikler detaylı olarak sunulmuş, eksikleri vurgulanarak iyileştiren çalışma önerimi sunulmuştur.
2. Sınıflandırma: DL modelleri, çok aşamalı DL modelleri, ML modelleri ve hibrit modeller karşılaştırılarak sınıflandırma aşamasındaki performans değerlendirmeleri sunulmuştur.
3. Autoencoder Tabanlı Gürültü Faktörü Kodlama (NFE) Yöntemi: Bu tez kapsamında, dolandırıcılık tespitinde kullanılmak üzere özgün bir AE tabanlı NFE yöntemi geliştirilmiştir. Bu yöntem, normal ve sahte işlem verileri

arasındaki doğal sınıf dengesizliğini ele alarak etkili bir dolandırıcılık tespit modeli oluşturulmasını sağlar.

4. SMOTE ile Entegrasyon: Bu tez, sınıf dengesizliğini gidermek amacıyla SMOTE ile AE tabanlı NFE yöntemini başarıyla birleştirmektedir. Bu entegrasyon, daha dengeli bir veri kümesi elde edilmesini sağlayarak dolandırıcılık tespit modelinin performansını artırmaktadır.
5. Üç Farklı Otokodlayıcı Varyasyonunun Karşılaştırılması: Çalışma, AE, VAE ve CAE olmak üzere üç farklı otokodlayıcı varyasyonunu içerir. Bu varyasyonlar, NFE tekniği ile geliştirilmiş versiyonlarıyla karşılaştırılarak, her birinin dolandırıcılık tespitindeki etkinliği detaylı bir şekilde incelenmiştir.
6. Farklı Dolandırıcılık Verileri ile Model Eğitimi: Tez, farklı dolandırıcılık verileri kullanılarak otokodlayıcı modellerinin eğitildiği bir süreci içermektedir. Bu, modelin gerçek dünya dolandırıcılık faaliyetlerini daha doğru bir şekilde tanımlamasını sağlayarak, modelin güvenilirliğini artırmaktadır.
7. Performans Metriklerinin Kapsamlı Değerlendirmesi: Çalışma, çeşitli performans metriklerini kullanarak geliştirilen yöntemlerin etkinliğini değerlendirmektedir. Bu metrikler arasında hassasiyet, özgüllük, doğruluk, F_1 skoru, AUC ve MCC gibi önemli ölçütler bulunmaktadır.
8. Otokodlayıcı Varyantları ile NFE Yöntemlerinin Karşılaştırılması: Tez, AE-NFE, VAE- NFE ve CAE- NFE yöntemlerini karşılaştırarak her birinin avantajlarını ve sınırlamalarını belirlemektedir. Bu kıyaslamalar, her yöntemin farklı senaryolarda ne kadar etkili olduğunu ortaya koymaktadır.

İKİNCİ BÖLÜM

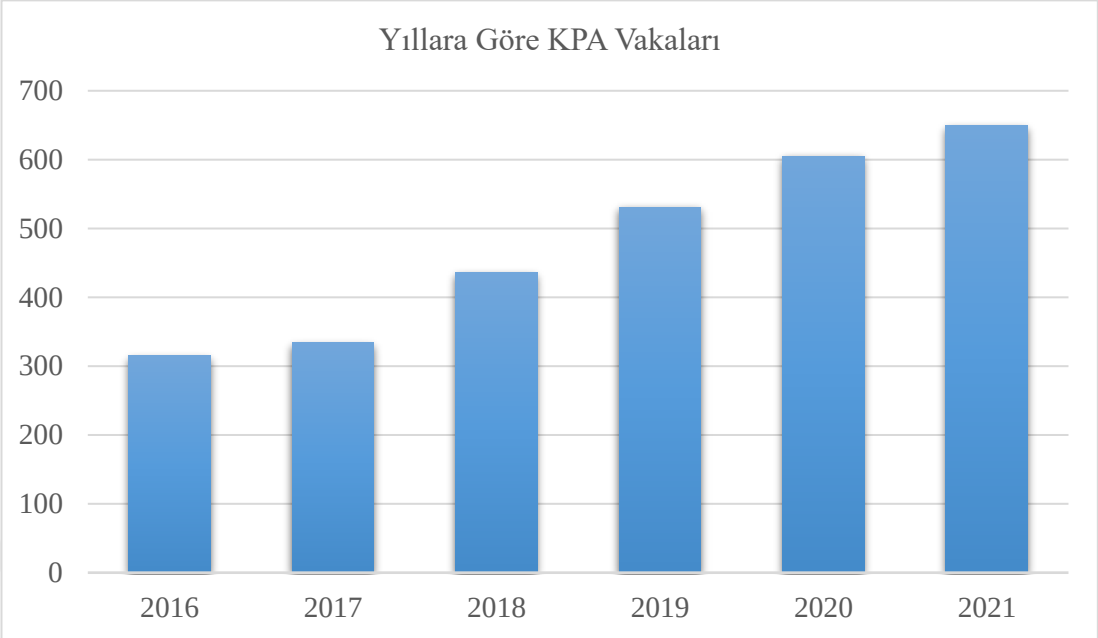
KARA PARA AKLAMA

KPA, yasa dışı yollardan elde edilen geliri veya varlıkları, yasal bir kaynak gibi göstermek veya kullanmak üzere gizleme sürecidir (Ogbeide et al., 2023). Temelde suç gelirlerini yasa dışı faaliyetlerden elde eden kişilerin, bu gelirleri açık bir şekilde gizleyerek ve yasal kaynaklardan elde edilmiş gibi kullanarak izlenebilirliklerini ortadan kaldırma çabasıdır. KPA genellikle suç örgütleri, terör örgütleri, organize suçlar ve diğer yasa dışı faaliyetlerle ilişkilidir (Chitimira & Animashaun, 2023). Bu, suç gelirlerinin kökenini gizleyerek ve yasal olmayan faaliyetlerle bağlantısını örtbas etmeyi amaçlayan karmaşık bir faaliyettir. KPA tarihi, farklı zamanlarda ve kültürlerde değişiklik göstermiştir, ancak modern anlamda bu yöntem, 20. yüzyılın ortalarında özellikle uyuşturucu ticareti ve diğer büyük suç faaliyetlerinin artışıyla birlikte daha fazla dikkat çekmeye başlamıştır (Sullivan, 2015). Bu süreç, daha etkili ve karmaşık aklama yöntemlerinin kullanılmasına ve yetkililer tarafından tespit edilmesini zorlaştırmaya yönelik sürekli bir evrimi beraberinde getirmiştir.

Tablo 2.1'de belirtildiği üzere Avrupa Birliği ülkeleri arasında yargı iş birliği kurumu olan Eurojust'ın 2022 haberinde kayıtlı KPA vaka sayısı 2016 yılından bu yana istikrarlı bir şekilde artmaktadır[2]. 2021 yılında Ajansa 600'den fazla vaka getirildiği belirtilmiştir. Bu rakam 2016 yılında kaydedilen vakaların iki katından fazlasını temsil etmektedir.

[2] https://www.eurojust.europa.eu/news/money-laundering-cases-registered-agency-doubled- last-6-years-according-eurojusts-new-report

Tablo 2.1: Yıllara göre Eurojust'a bildirilen KPA vaka sayıları

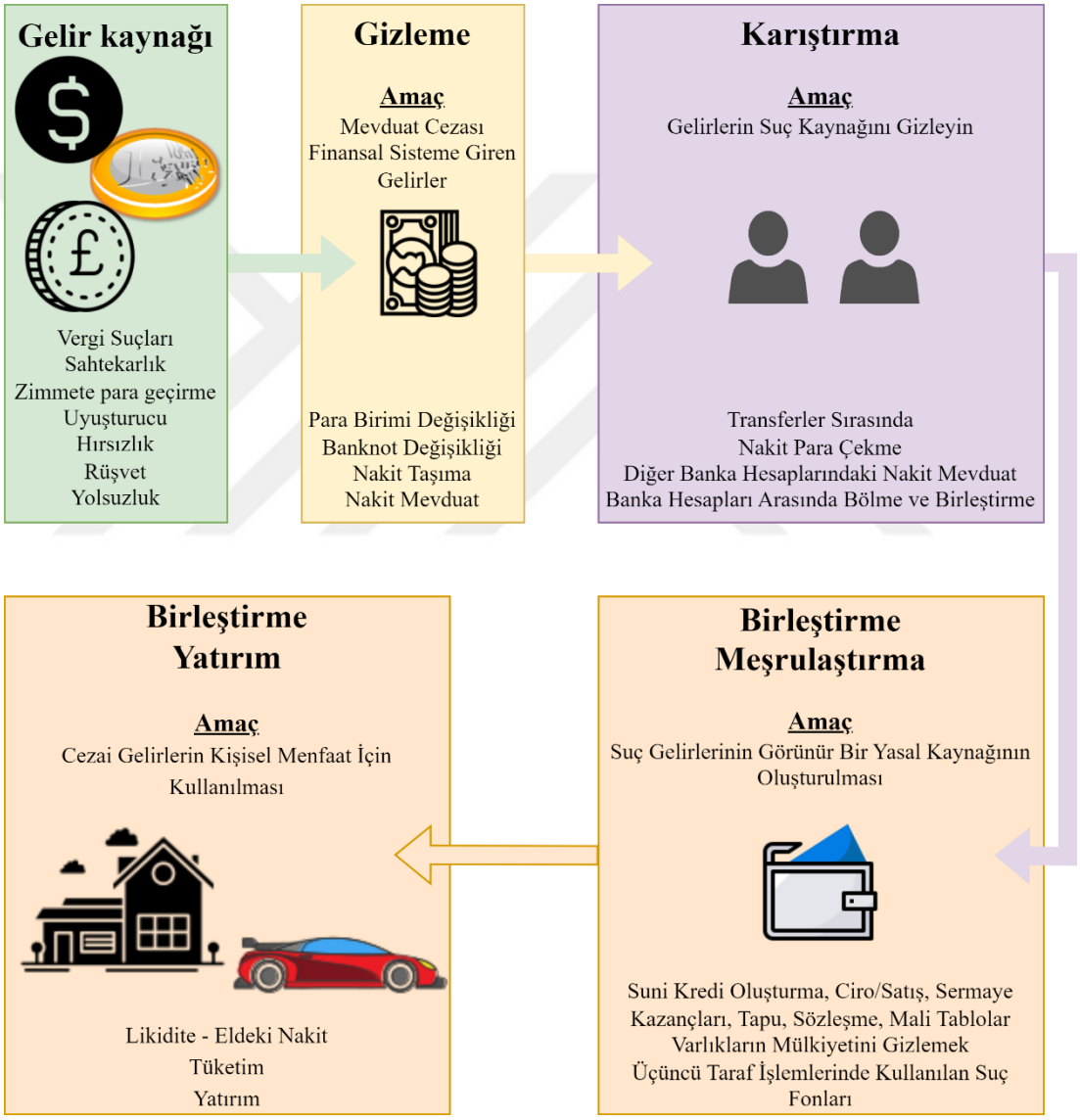


KPA sorununa uluslararası düzeyde ciddi bir yaklaşım, özellikle 1989'da kabul edilen FATF tarafından belirlenen kılavuzlarla birlikte başladı (Jakobi, 2018). FATF, üye ülkeler arasında KPA ile mücadelede iş birliği sağlamak ve en iyi uygulamaları teşvik etmek amacıyla kuruldu. Bu dönem, KPA tespiti ve önlenmesi konusunda önemli bir dönemeç olarak öne çıkmaktadır. KPA türleri, çeşitli yöntemler ve stratejiler kullanarak gerçekleştirilebilir, ancak genellikle finansal sistemleri istismar etme eğilimindedir. FATF rehberliğinde, uluslararası topluluğun AML için daha etkili bir mücadele yürütebilmesi adına adımlar atılmıştır. Bu çerçeve, suç gelirlerinin yasal finans sistemine etkileşimini önleme, şüpheli finansal faaliyetleri tanımlama ve takip etme ve uluslararası iş birliği ile suçla mücadelede daha koordineli bir yaklaşım benimseme amacını taşımaktadır. KPA çeşitleri çok çeşitli olabilir, ancak genellikle şu ana başlıklar altında toplanır (Teichmann, 2019):

- **Yasal İşletmeler Yoluyla Aklama:** Suç gelirlerinin yasal işletmelerde aklanması anlamına gelir, bu da genellikle restoranlar, kumarhaneler, gayrimenkul yatırımları ve benzeri yasal işletmelerin kullanımını içerir.
- **Nakit Aklama:** Büyük miktardaki nakit parayı bankalar aracılığıyla aklama veya yurtdışına transfer etme sürecini içerir, bu da kaynağını gizlemek isteyenler için bir yöntem olabilir.

- **Mülk Aklama:** Gayrimenkul yatırımları ve mülk edinimini içerir ve bu şekilde suç gelirleri yasal bir şekilde gösterilebilir.
- **Para Transferi ve Kripto Paralar:** KPA için yeni ve karmaşık yollar sunar, ancak bu tür varlıkların izlenmesi ve tespiti genellikle zorlu bir süreci içerir. Bu yöntemlerin hepsi, suç gelirlerini gizleme ve yasa dışı kaynakları meşru gibi gösterme amacını taşıyan çeşitli stratejileri temsil eder.

Şekil 2.1'de gösterildiği üzere KPA aşamaları sunulmuştur.



Şekil 2.1: KPA aşamaları

Kaynak: (Jayantilal et al., 2017)

AML, yasa dışı yollarla elde edilen paraya yasal bir görünüm kazandırmak amacıyla kullanılan süreci ifade eder. KPA aşamaları genellikle üç aşamada gerçekleşir (Cassella, 2018):

1. **Gizleme (Placement):** Bu aşamada, suç gelirleri yasal finansal sistem içine sokulmaya çalışılır. Bu genellikle küçük miktarlarda yapılan işlemleri içerir. Örneğin, büyük miktarlardaki kara paralar, küçük miktarlarda yapılan çok sayıda nakit işlemle karıştırılır. Banka hesaplarına veya diğer finansal araçlara aktarılabilir.
2. **Karıştırma (Layering):** Bu aşamada, kara paraların kökenini gizlemek ve izini kaybettirmek için karmaşık finansal işlemler yapılır. Bu, çoklu hesaplar, farklı ülkelerdeki banka hesapları, yatırımlar, hisse senetleri, gayrimenkuller ve diğer varlık türlerini içerebilir. Amaç, parayı takip edenlerin izini kaybettirmek ve suç gelirini karmaşık hale getirmektir.
3. **Birleştirme (Integration):** Bu aşamada, kara para yasal ekonomiye dâhil edilir ve suç geliri, yasal olarak elde edilmiş gelir gibi gösterilmeye çalışılır. Bu süreç genellikle gayrimenkul alımı, iş kurma, yasal ticaret ve yatırımlar gibi yöntemleri içerir. Bu aşama, kara parayı yasal bir şekilde kullanma ve suç gelirini gizleme amacına hizmet eder.

AML, uluslararası düzeyde önlenmeye çalışılan bir suçtur ve birçok ülkede sıkı yasalar ve düzenlemelerle kontrol edilmeye çalışılmaktadır. Finansal kurumlar, müşterilerinin işlemlerini izleyerek ve şüpheli durumları raporlayarak bu tür faaliyetlere karşı mücadele ederler.

2.1. Kredi Kartı Sahtekârlığı

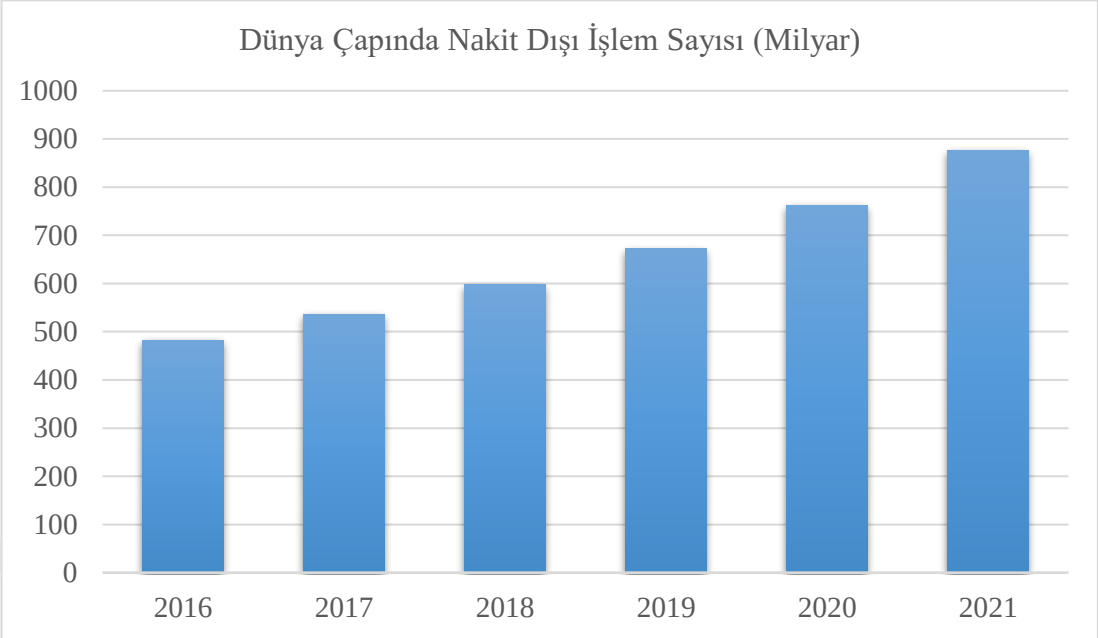
CCF, kredi kartları veya banka kartlarının izinsiz ve haksız yolla kullanılmasıyla ortaya çıkan bir suç biçimidir (Srivastava et al., 2008). Bu suç, genellikle kart sahibinin bilgisi veya onayı olmadan gerçekleştirilen, haksız maddi kazanç elde etmeye yönelik işlemleri içerir (Dal Pozzolo et al., 2014). CCF, dolandırıcılık amacı güden ve finansal kaynakları kötüye kullanarak yasadışı kazanç elde etmeyi hedefleyen bir tür finansal suçlar kategorisine dâhil edilebilir. Bu tür suçlar genellikle teknolojik gelişmelerin ve dijital ödeme sistemlerinin yaygınlaşmasının etkisiyle karmaşıklaşmış ve çeşitlenmiştir, bu da yetkililerin bu tür faaliyetlere karşı önleyici ve yaptırım uygulama çabalarını artırmalarını gerektirmektedir (Benson Edwin Raj & Annie Portia, 2011).

CCF, çeşitli yöntemler aracılığıyla gerçekleştirilebilen karmaşık bir suç türüdür. Suçlular, bu eylemi gerçekleştirmek için farklı stratejiler kullanabilirler (Bhatla et al., 2003).

- **Kart Kopyalama:** Fiziksel bir kopya oluşturarak veya kartın manyetik şeridini kopyalayarak kredi kartı verilerini sahtekârca kullanma yöntemini içerir.
- **Kredi Kartı Bilgilerinin Çalınması:** Suçluların çevrimiçi alışveriş siteleri veya veri ihlalleri yoluyla kredi kartı bilgilerini ele geçirmelerini içerir, bu bilgiler daha sonra çeşitli işlemlerde kullanılabilir.
- **Sahte Kart Üretimi:** Suçluların sahte kredi kartları üreterek bu kartları kullanmalarını içerir; bu sahte kartlar, gerçek kart bilgilerini içerebilir, bu da sahtekârlık girişimlerini daha zor algılanır hale getirir
- **Kimlik Hırsızlığı:** Bireylerin kimlik bilgilerini çalarak bu bilgilerle yeni kredi kartları açma veya mevcut kartları kullanma yolunu içerir.

CCF, bu çeşitli yöntemler aracılığıyla sadece bireylerin maddi kayıplarına neden olmakla kalmaz, aynı zamanda finansal kurumları ve güvenlik uzmanlarını sürekli olarak yeni koruma önlemleri geliştirmeye zorlayarak geniş çapta bir güvenlik sorununu da temsil eder (Richhariya & Singh, 2014). Suçlular, bu tür sahtekârlık eylemlerini genellikle maddi kazanç elde etmek, KPA veya yasa dışı mal ve hizmetleri satmak amacıyla kullanabilirler. Bu durum, finansal kuruluşları ve bireyleri sürekli olarak gelişen tehditlere karşı koruma önlemleri almaya zorlamaktadır (Dal Pozzolo et al., 2014). Tablo 2.2'de sunulduğu üzere kredi kartı işlemleri sürekli artış göstermektedir. Bu sebeple CCF önlenmesi kritik bir görevdir. Bunun için uygulanan tedbirler arasında, güvenlik protokollerinin sürekli olarak güçlendirilmesi, analitik araçların kullanılması ve kart sahiplerine düzenli olarak güvenlik bilinci eğitimleri verilmesi bulunmaktadır (Zaabi & Tubaishat, 2015). Bu önleyici yaklaşımlar, finansal kayıpları en aza indirmek ve suçluların faaliyetlerini tespit ederek engellemek amacıyla geniş bir çaba içerisinde uygulanmaktadır.

Tablo 2.2: Yıllara göre nakit olmayan banka işlemlerinin sayısı[3]

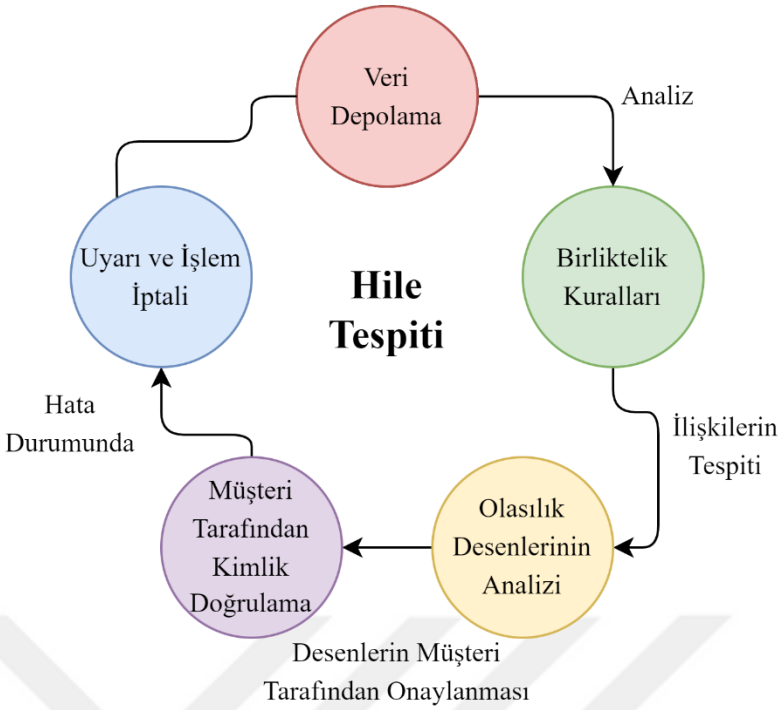


AML, geleneksel yöntemlerle giderek zorlaşan bir görev haline gelmiştir (Yang et al., 2023). Suçlular, izlerini gizlemek için her geçen gün daha karmaşık yöntemlere başvurmaktadır. Bu nedenle DL, AML için umut vaat eden potansiyel bir araç olarak öne çıkmaktadır. DL, verileri otomatik olarak işleyebilir ve sürekli olarak yeni desenleri öğrenme yeteneğine sahiptir (Fekri et al., 2021; Jan et al., 2019). AML için kullanıldığında, büyük finansal veri setlerini analiz edebilir, anormallikleri tespit edebilir ve şüpheli işlemleri belirleyebilir. Ancak, bu güçlü yöntem, veri gizliliği, etik sorunlar ve eğitim için yeterli temsilci veri gibi bir dizi sınırlamayla karşılaşabilir. DL, AML için potansiyel bir çözüm sunsa da bu tür teknolojilerin kullanımında yasal, etik ve veri gizliliği konularına özel dikkat gösterilmesi önemlidir. Geliştirilmeye devam eden DL modelleri, finansal kuruluşlar ve yetkililere KPA ile mücadelede daha etkili bir destek sağlama potansiyeline sahiptir.

Şekil 2.2'de gösterildiği gibi hile tespiti, çeşitli yöntemler ve teknolojiler kullanılarak potansiyel hileli faaliyetleri tanımlama ve engelleme sürecidir.

[3] https://www.paymentscardsandmobile.com/

world-payments-report-2018-digital-payments-booming/



Şekil 2.2: ML ile hile tespit sisteminin aşamaları[4]

1. **Veri Toplama:** İlk aşama, potansiyel hileli faaliyetleri tespit etmek için geniş bir veri kümesini toplamayı içerir. Bu veriler genellikle finansal işlemler, kullanıcı etkileşimleri, sistem günlükleri, coğrafi konum bilgileri, cihaz bilgileri ve diğer ilgili verileri içerir.
    a. **Veri Analizi:** Toplanan veriler, analitik ve veri madenciliği teknikleri kullanılarak incelenir. Özellikle, tipik kullanım desenlerinden sapmaları ve anomaliyi tespit etmek amacıyla istatistiksel analizler uygulanır.
    b. **Proaktif İzleme:** Gerçek zamanlı olarak veya belirli aralıklarla, sistemdeki işlemler ve etkileşimler sürekli olarak izlenir. Kritik noktalarda, anlık uyarılar veya bildirimlerle durum tespiti yapılır.
2. **Birliktelik Kuralları:** ML algoritmaları, mevcut ve geçmiş verileri kullanarak hileli desenleri öğrenir ve tanımlar. Yapay zekâ (AI) tabanlı sistemler, karmaşık ve dinamik hileli faaliyetleri tespit etme yeteneğine sahiptir.

[4] https://spd.tech/machine-learning/credit-card-fraud-detection/

3. **Olasılık Desenlerinin Analizi:** Belirli kurallar ve politikalar, potansiyel hileli faaliyetleri tanımlamak için kullanılır. Önceden belirlenmiş kurallar, şüpheli durumları belirleme ve raporlama sürecine dâhil edilir.
4. **Müşteri Tarafından Kimlik Doğrulama:** Kullanıcıların kimliklerini doğrulamak ve güvenliği artırmak amacıyla biyometrik veriler kullanılabilir. Parmak izi, retina taraması, yüz tanıma gibi biyometrik veri türleri, kimlik hırsızlığını ve diğer hileli faaliyetleri azaltabilir.
5. **Uyarı / İşlem İptali:** Hile tespit sistemleri tarafından belirlenen şüpheli durumlar, uzman analistler tarafından daha detaylı bir şekilde incelenir ve doğrulanır. Yanlış pozitifleri azaltmak ve gerçek hileli durumları belirlemek için bu aşama önemlidir.
    a. **Geri Bildirim ve Güncelleme:** Hile tespit sistemleri sürekli olarak güncellenir ve iyileştirilir. Bu, yeni hileli desenlere karşı daha etkili bir koruma sağlar. Analistlerden gelen geri bildirimler ve sistemin performansını değerlendiren veriler, sürekli iyileştirmeleri destekler.

E-ticaretin hızla yayılması, kredi kartı kullanımındaki artış ve işlem düzeyindeki bilgi toplamanın kolaylaşması, büyük veri setlerinin oluşmasını sağlamıştır. Bu durum, CCF tespit etme üzerine yoğun bir araştırma çabasını beraberinde getirmiştir. Araştırmacılar, bu alanda çeşitli teknikleri kullanarak dolandırıcılığı tespit etmeye yönelik çözümler geliştirmişlerdir. Bunlar arasında, kümelenme teknolojisi (clustering technology) (Bolton & Hand, 2002), akran grubu analizi (peer group analysis) (Weston et al., 2008), genetik algoritmalar (genetic algorithms) (Duman & Ozcelik, 2011), birliktelik kuralları analizi (association rules analysis) (Sánchez et al., 2009), Bayes çıkarımı (Bayesian inference) (Bahnsen et al., 2013), makine öğrenimi (Sailusha et al., 2020) ve sinir ağları (neural networks) (Maes et al., 2002) gibi çeşitli yöntemler bulunmaktadır. Bu çeşitlilik, büyük veri setlerindeki anormallikleri tespit etme ve dolandırıcılığı önleme konusundaki araştırmalara geniş bir perspektif sunmaktadır. Bu teknikler, e-ticaret platformlarında güvenliği artırmak için geliştirilmekte olup, finansal işlemlerdeki güvenlik açıklarını azaltma amacını taşımaktadır.

CCF tespitinde toplanan büyük veri kümelerinin incelenmesinde ML ve sinir ağları önemli yöntemler olarak öne çıkmaktadır (Bhattacharyya et al., 2011; Thennakoon et al., 2019). Bu alandaki çeşitli çalışmalarda, gözetimli makine öğrenimi, CCF

sorunlarını çözmek için sıkça tercih edilen bir yöntemdir (Roy et al., 2018; Xuan et al., 2018). Özellikle hile tespiti sürecinde veri ön işleme tekniklerinin iyileştirilmesi ve model performansını artırmak amacıyla geliştirilmiş sınıflandırıcıların kullanımı önde gelen araştırma konularındandır. Bu yöntemler, büyük veri setlerindeki karmaşıklıkları ele alarak CCF ile mücadelede etkili bir çözüm sunma potansiyeline sahiptir.

Dolandırıcılık tespiti, araştırma topluluğunun öncelikli ilgi alanlarından biri olup, bu kompleks görevle ilgili çeşitli zorluklarla başa çıkmak adına bir dizi teknik geliştirilmiştir (Habibpour et al., 2023; Wei et al., 2023). Özellikle CIP, dolandırıcılık faaliyetlerinin doğru bir şekilde tespitini zorlaştırabilir (Strelcenia & Prakoonwit, 2023a; Sinayobye et al., 2023). Geleneksel yöntemler arasında çoğunluk sınıfından örneklerin eksiltildiği (undersampling) ve azınlık sınıfından örneklerin çoğaltıldığı (oversampling) teknikler yer alır (Wongvorachan et al., 2023). Ancak, bu yaklaşımlar, değerli bilgi kaybına neden olabilir veya sentetik örneklerin eklenmesiyle dolandırıcılık desenlerini doğru temsil etme konusunda sorunlar yaşatabilir (Rathore et al., 2022). Son yıllarda, CIP ile başa çıkmak için odaklanılan yöntemler arasında SMOTE gibi üst örnekleme teknikleri ön plana çıkmıştır (Dablain et al., 2022). SMOTE, azınlık sınıf örnekleri arasında interpolasyon yaparak sentetik örnekler oluşturur ve bu şekilde azınlık sınıfın etkili bir şekilde temsilini artırır (Zakariah et al., 2023). Ancak, SMOTE'un sahtekârlık işlemlerine özgü karmaşık özellikleri yakalama yeteneği sınırlıdır, bu da sahtekârlık tespitinde etkinliği azaltabilir (Strelcenia & Prakoonwit, 2023b). Geleneksel üst örnekleme tekniklerinin sınırlamalarını aşmak için otokodlayıcı tabanlı yaklaşımlar ortaya çıkmıştır. Otokodlayıcılar, giriş verisinin sıkıştırılmış temsillerini öğrenmek için tasarlanmış sinir ağı mimarileridir ve anormallik tespiti görevlerinde üstün performans sergileyebilir, bu da dolandırıcılık tespiti alanında yeni ve umut verici bir alternatif sunmaktadır (Laakom et al., 2022; Takiddin et al., 2022; Fanai & Abbasimehr, 2023).

Bu bölümde, ULB Machine Learning Grubu tarafından sağlanan zorlu bir kredi kartı işlem veri kümesi[5] üzerinde uygulanan otokodlayıcı tabanlı yöntemleri inceleyeceğiz. Bu veri kümesi, aynı zamanda çalışmamızda Veri Kümesi-I olarak isimlendirilmiştir. İlk olarak, (Zou et al., 2019) çalışmasında, yazarlar veri kümesini dengelemek için üst

[5] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

örnekleme, gürültü azaltma ve özellik öğrenme için denoising autoencoder sinir ağları ile sınıflandırma için derin tam bağlı sinir ağlarını birleştirirler. Deneysel sonuçlar, bu yaklaşımın azınlık sınıf örneklerinin sınıflandırma doğruluğunu önemli ölçüde artırdığını ve en iyi doğruluk performansını %97.93'e ulaştığını göstermektedir.

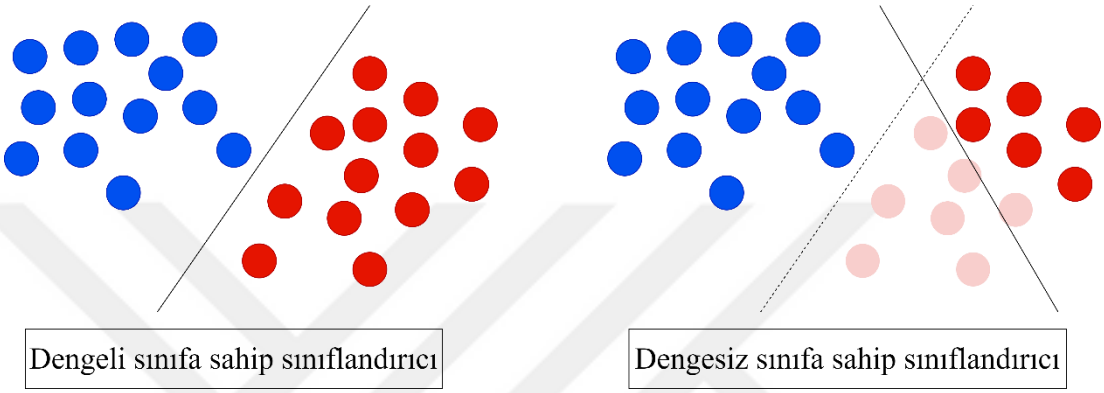
(Tingfei et al., 2020) çalışmasında, yazarlar veri kümesini dengelemek için azınlık sınıf örneklerini üst örnekleme amacıyla VAE kullanarak yenilikçi bir yaklaşım önerirler. VAE tarafından oluşturulan örnekleri geleneksel derin öğrenme teknikleri ile birleştirerek, SMOTE, Generative Adversarial Networks (GANs) ve temel modellere kıyasla üstün performans elde ederler. (Misra et al., 2020) çalışmasında, yazarlar, autoencoder tabanlı özellik çıkarmanın ardından MLP, KNN ve LR gibi algoritmalarla sınıflandırmayı içeren iki aşamalı bir metodoloji sunarlar. (T.-H. Lin & Jiang, 2021) çalışmasında, yazarlar, özellik çıkarma için AE, sınıflandırma için ise olasılıksal Random Forest (RF) kullanarak yenilikçi bir CCF tespit yöntemi olan AE-PRF'i tanıtırlar. (Zioviris et al., 2022) çalışmasında, yazarlar, işlem veri akışlarındaki etkili CCF tespiti için çok aşamalı bir DL modeli sunarlar. Model, özellik seçimi ve gizli veri temsilini elde etmek için iki autoencoder kullanır ve ardından dolandırıcılık tanımlaması için derin evrişimli sinir ağı (CNN) kullanır.

(Du et al., 2023) çalışmasında, yazarlar, aşırı dengesiz veri kümelerinde CCF tespit etmek için kullanılan yenilikçi bir yaklaşım olan Autoencoder with Probabilistic LightGBM (AED-LGB) olarak adlandırılan bir yaklaşımı tanıtırlar. Simetrik bir autoencoder, yüksek boyutlu kredi kartı verilerinden düşük boyutlu özellikler çıkarmak için kullanılır ve ardından LightGBM kullanılarak sınıflandırma yapılır. (Y. Ding et al., 2023) çalışmasında, yazarlar CCF tespitinde dengesiz veri sorununu ele almak için Variational Autoencoder Generative Adversarial Network (VAEGAN) tabanlı geliştirilmiş bir üst örnekleme tekniği tanıtırlar. Çalışma, bu yaklaşımı SMOTE, GAN ve VAE gibi mevcut üst örnekleme yöntemleriyle karşılaştırırken beş sınıflandırma modelini değerlendirir. Çalışma, geliştirilmiş VAEGAN ile XGBoost sınıflandırma algoritmasının diğer yöntemleri geride bıraktığı sonucuna varır.

ÜÇÜNCÜ BÖLÜM

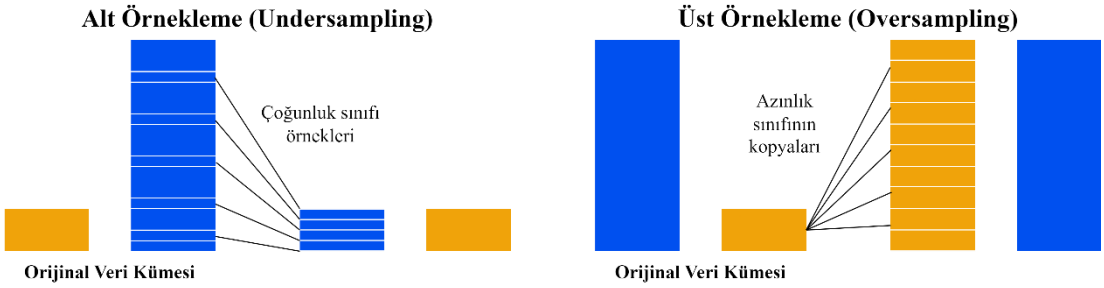
SINIF DENGESİZLİĞİ PROBLEMİ

CIP, Şekil 3.1'de gösterildiği üzere bir ML modelinin eğitildiği veri kümesindeki farklı sınıflar arasında belirgin bir dengesizlik durumunu ifade eder. Bu durumda, bir veya birkaç sınıf, diğerlerine kıyasla çok daha fazla örneğe sahiptir (Guo et al., 2008).



Şekil 3.1: Sınıf dengesizliği problemi

AI uygulamalarında CIP, modelin eğitimi sırasında potansiyel olumsuz etkilere neden olabilir, çünkü modeller genellikle çoğunluk sınıfındaki örnekleri daha iyi öğrenirken, azınlık sınıfındaki örnekleri ihmal edebilir. Bu durum, modelin çoğunluk sınıfındaki desenlere aşırı uyum sağlayarak azınlık sınıfındaki önemli desenleri doğru bir şekilde öğrenememesine ve bu sınıfı doğru bir şekilde sınıflandırmamasına neden olabilir. CIP ile başa çıkmak, modelin tüm sınıfları dengeli bir şekilde öğrenmesini sağlamak ve genel performansını artırmak için önemlidir (Dai et al., 2023). Şekil 3.2'de sunulduğu üzere CIP ile başa çıkabilmek için üst örnekleme ve alt örnekleme teknikleri kullanılmaktadır.



Şekil 3.2: CIP önlemek için en çok kullanılan Undersampling ve Oversampling teknikleri

3.1. Undersampling

CIP, bir sınıfın diğerinden belirgin şekilde daha fazla örneğe sahip olduğu durumu ifade eder (S. M. Liu et al., 2023). Bu durum, ML modelleri için önemli bir zorluktur çünkü model, çoğunluk sınıfındaki örnekleri daha iyi öğrenirken, azınlık sınıfını ihmal edebilir. Bu dengesizlik, modelin azınlık sınıfındaki nadir örnekleri doğru bir şekilde öğrenmesini zorlaştırabilir. CIP çözümü için kullanılan bir veri düzeltme tekniği olan alt örnekleme (undersampling), çoğunluk sınıfına ait örneklerin sayısını azaltarak sınıflar arasındaki dengesizliği düzeltmeye çalışır (Soltanzadeh et al., 2023). Ancak alt örnekleme, özellikle azınlık sınıfındaki nadir örneklerin tamamen kaybedilmesine yol açabilir, bu da modelin azınlık sınıfını etkili bir şekilde öğrenmesini engelleyebilir (Kalaycıoğlu et al., 2023). Bu durumu aşmak için dikkatli bir örnekleme stratejisi seçmek ve azınlık sınıfındaki önemli desenlere odaklanmak gerekmektedir.

Alt örnekleme, özellikle büyük veri setlerinde hesaplamalı açıdan daha hızlı olması nedeniyle tercih edilen bir CIP çözüm yöntemidir. Bu yöntem, çoğunluk sınıfındaki fazla örnekleri çıkartarak dengesizliği azaltmaya çalışır. Ancak, azınlık sınıfındaki nadir örneklerin tamamen çıkarılması, bilgi kaybına yol açabilir. Bu durum, özellikle medikal, finansal ve endüstriyel alanlarda, azınlık sınıfındaki kritik örneklerin dikkatlice korunması gereken durumları kapsar. Alt örneklemenin etkili bir şekilde uygulanabilmesi için, CIP özel gereksinimleri dikkate alınmalı ve uygun örnekleme stratejileri geliştirilmelidir. Ayrıca, modelin genel performansını değerlendirmek için uygun metriklerin seçilmesi önemlidir, çünkü sınıf dengesizliği çözümünün başarısı, sadece azınlık sınıfındaki örnekleri çıkartmakla değil, aynı zamanda modelin genel doğruluğu ve performansıyla da ölçülmelidir.

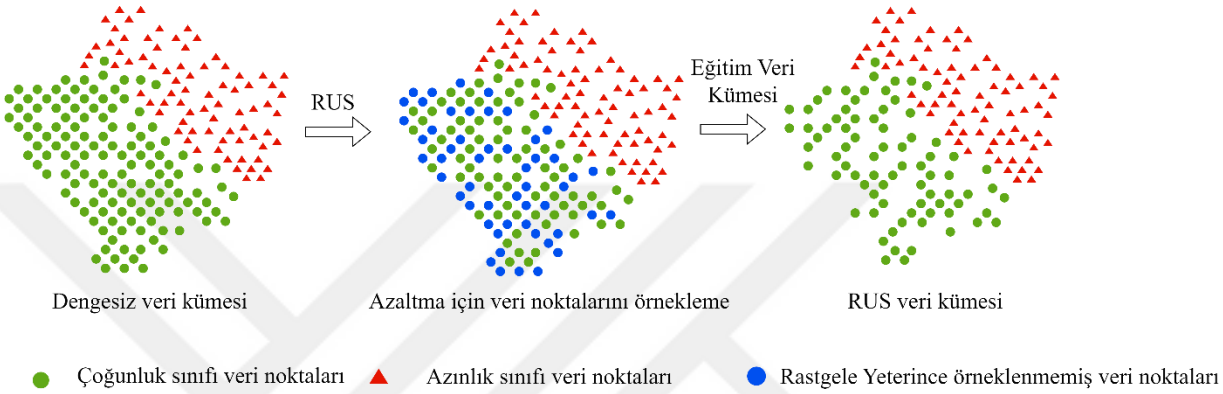
3.1.1. Random Undersampling

RUS, Şekil 3.3'de sunulduğu üzere sınıf dengesizliği sorununu çözmek veya hafifletmek amacıyla kullanılan bir veri örnekleme tekniğidir (Leevy et al., 2023). Bu teknikte, çoğunluk sınıfındaki fazla örnekler rastgele seçilerek azaltılır, böylece sınıflar arasındaki örnek sayısı dengelenir (Rafrastara et al., 2023). RUS, genellikle büyük bir çoğunluk sınıfı ve daha küçük bir azınlık sınıfı içeren veri setlerinde kullanılır. Aşağıda RUS adımları sırasıyla sunulmuştur.

1. **Azınlık Sınıfındaki Örneklerin Sayısını Belirleme:** İlk adım olarak, azınlık sınıfındaki örnek sayısının, çoğunluk sınıfındaki örnek sayısına eşit olacak

şekilde belirlenmesi gerekmektedir. Bu sayı, sınıflar arasındaki dengeyi sağlamak için seçilen bir hedef oran.

2. **Çoğunluk Sınıfından Rastgele Örnek Seçme:** Çoğunluk sınıfındaki örnekler içinden belirlenen azınlık sınıfı örnek sayısı kadar rastgele örnek seçilir. Bu seçim işlemi, çoğunluk sınıfındaki fazla örneklerin azaltılmasını sağlar.
3. **Yeni Veri Setini Oluşturma:** Seçilen rastgele örneklerle birlikte azınlık sınıfındaki orijinal örneklerden oluşan yeni bir veri seti oluşturulur.



Şekil 3.3: Random Undersampling

Kaynak: (Rafrastara et al., 2023)

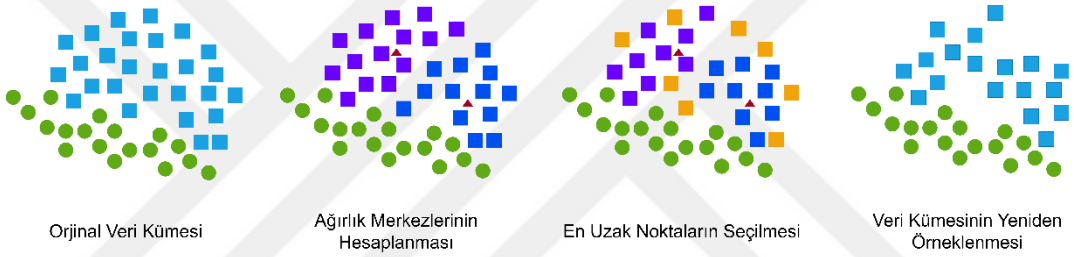
RUS, basit bir uygulama ve yüksek hesaplama verimliliği nedeniyle popüler bir tekniktir. Ancak, bu yöntemin bazı dezavantajları vardır. Özellikle, çoğunluk sınıfındaki rastgele seçilen örneklerin önemli bilgileri kaybetme olasılığı yüksektir. Ayrıca, azınlık sınıfındaki tüm önemli varyasyonları temsil edemeyebilir. Bu nedenle, RUS kullanılmadan önce dikkatlice düşünülmesi ve veri setinin özelliklerine uygunluğunun değerlendirilmesi önemlidir. Ayrıca, diğer örnekleme teknikleriyle birleştirilerek veya çeşitli dengeleme stratejileriyle birlikte kullanılarak daha etkili sonuçlar elde edilebilir.

3.1.2. Cluster Centroid Undersampling

CCU, Şekil 3.4' sunulduğu üzere CIP ile başa çıkmak amacıyla kullanılan bir veri örnekleme tekniğidir. Bu teknik, özellikle ML modellerini eğitirken, sınıflar arasındaki gözlemler arasındaki dengesizlikleri azaltmak için kullanılır. CCU, önce veri kümesindeki azınlık sınıfının örneklerini kümeleme (clustering) işlemine tabi tutar. Daha sonra, her bir kümenin merkezini alır ve bu merkezleri, azınlık sınıfının yeni temsilcileri olarak seçer. Bu işlem, azınlık sınıfındaki örnek sayısını azaltırken,

orijinal örneklerin genel özelliklerini korumaya çalışır (W.-C. Lin et al., 2017). Aşağıda CCU adımları sırasıyla sunulmuştur.

1. **Azınlık Sınıfının Kümeleme (Clustering):** Azınlık sınıfındaki örnekler, belirli bir kümeleme algoritması (örneğin, K-Means) kullanılarak kümelere ayrılır. Her küme, azınlık sınıfının örneklerinden oluşur.
2. **Küme Merkezlerini Hesaplama:** Her bir kümenin merkezi hesaplanır. Bu merkezler, azınlık sınıfının yeni temsilcileri olarak kullanılacaktır.
3. **Yeni Temsilcileri Oluşturma:** Azınlık sınıfının orijinal örnekleri yerine, her bir kümenin merkezi, azınlık sınıfının yeni temsilcisi olarak seçilir.
4. **Yeni Veri Setini Oluşturma:** Yeni temsilcilerle birlikte çoğunluk sınıfındaki orijinal örneklerden oluşan yeni bir veri seti oluşturulur.



Şekil 3.4: Cluster Centroid Undersampling

Bu teknik, azınlık sınıfındaki örnek sayısını azaltarak ve çoğunluk sınıfındaki örneklerle daha dengeli bir veri seti elde ederek model performansını artırmayı amaçlar. Ancak, her veri seti ve problem bağlamı farklı olduğu için, CCU her zaman uygun bir çözüm olmayabilir. Bu nedenle, farklı tekniklerin ve yöntemlerin dikkatlice değerlendirilmesi ve uygulanması gerekmektedir.

3.1.3. Near Miss Undersampling

NMU, Şekil 3.5'de sunulduğu üzere CIP ile mücadele etmek için kullanılan bir veri örnekleme tekniğidir. Bu teknik, özellikle CIP olduğu veri setlerinde, azınlık sınıfındaki örnekleri azaltarak modelin dengeli bir şekilde eğitilmesini sağlamayı amaçlar (Mqadi et al., 2021). NMU, azınlık sınıfındaki örnekleri çoğunluk sınıfındaki örneklerle "yakın" olanları koruyarak seçer. Aşağıda NMU adımları sırasıyla sunulmuştur.

1. **Örneklerin Gruplandırılması (Clustering):** Önce, azınlık sınıfındaki örnekler benzerlik temelinde belirli bir kümeleme algoritması kullanılarak

kümelere ayrılır. Bu, azınlık sınıfındaki örneklerin birbirlerine ne kadar benzediğini belirlemek için yapılır.

2. **Her Kümeden Temsilci Seçme:** Her küme için, bu kümedeki azınlık sınıfı örneklerinden çoğunluk sınıfındaki örneklerle benzerliği en düşük olan birkaç örnek seçilir. Bu, çoğunluk sınıfındaki örneklerle daha yakın olan azınlık sınıfındaki örnekleri koruma stratejisini benimser.
3. **Yeni Veri Setini Oluşturma:** Seçilen örneklerle birlikte çoğunluk sınıfındaki orijinal örneklerden oluşan yeni bir veri seti oluşturulur.

Şekil 3.5: Near Miss Undersampling

NMU, azınlık sınıfındaki örneklerin çoğunluk sınıfındaki örneklerle benzerliklerine dayanarak örnek seçimi yapması nedeniyle dikkate değer bir tekniktir. Bu yaklaşım, sınıf dengesizliğini azaltmaya çalışırken, azınlık sınıfındaki önemli örnekleri korumayı hedefler. Ancak, bu yöntemin dezavantajlarından biri, çoğunluk sınıfındaki örneklerle benzerliği düşük olan azınlık sınıfındaki örnekleri göz ardı edebilmesidir. Bu nedenle, veri seti ve problem bağlamına bağlı olarak, farklı örnekleme tekniklerinin ve değerlendirme stratejilerinin kullanılması gerekebilir.

3.2. Oversampling

Üst örnekleme (Oversampling), CIP çözme amacı güden bir stratejidir. Bu yöntemde, azınlık sınıfa ait örneklerin sayısı artırılarak, modelin eğitim sürecinde azınlık sınıfının daha etkili bir şekilde öğrenmesi hedeflenir (Mohammed et al., 2020). Bu, genellikle azınlık sınıfındaki örneklerin kopyalanması veya benzer örneklerin sentetik olarak oluşturulmasıyla gerçekleştirilir. Bu yaklaşım, azınlık sınıfının bilgi kaybına uğramadan daha etkili bir şekilde temsil edilmesini sağlamak için kullanılır. Azınlık sınıfa ait örneklerin çoğaltılması, modelin eğitimini daha dengeli ve adil hale getirerek, sınıflar arasındaki dengesizliği azaltır. Ancak, doğru bir şekilde uygulanması ve

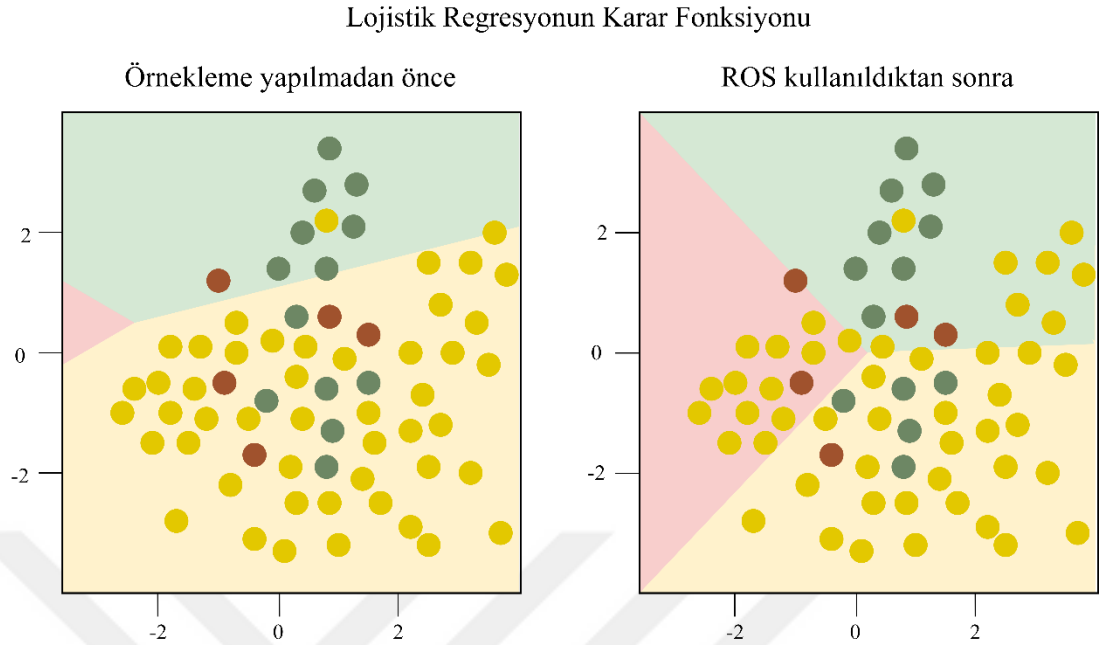
sentetik örneklerin gerçek veriye uygunluğunun sağlanması önemlidir, aksi takdirde modelin performansı olumsuz etkilenebilir.

3.2.1. Random Oversampling

ROS, Şekil 3.6'da sunulduğu üzere CIP çözümü amacıyla kullanılan bir veri örnekleme tekniğidir. Bu teknik, azınlık sınıfındaki örnekleri çoğaltarak sınıflar arasındaki örnek sayısını eşitlemeyi amaçlar. ROS, basit bir uygulama ve anlama kolaylığı nedeniyle popüler bir yöntemdir (A. Liu et al., 2007). Aşağıda ROS oranı Denklem **(3-1)**'de ve ROS adımları sırasıyla sunulmuştur.

1. **Azınlık Sınıfının Belirlenmesi:** İlk adım olarak, azınlık sınıfının belirlenmesi gerekmektedir. Azınlık sınıfı, daha az örneğe sahip olan sınıfı temsil eder.
2. **Azınlık Sınıfındaki Örneklerin Sayısının Belirlenmesi:** Azınlık sınıfındaki örneklerin sayısı belirlenir. Bu sayı, çoğunluk sınıfındaki örnek sayısıyla aynı hale getirilerek sınıflar arasında bir denge sağlanmaya çalışılır.
3. **Rastgele Örnekleme (Random Sampling):** Azınlık sınıfındaki örnekler içinden belirlenen sayıda örnek rastgele seçilir. Bu seçim işlemi, azınlık sınıfındaki örnek sayısını çoğaltarak denge sağlamayı amaçlar.
4. **Seçilen Örneklerin Yeniden Eklenmesi:** Rastgele seçilen örnekler, aynı örnekleri tekrar ekleyerek veya çoğaltarak azınlık sınıfındaki örnek sayısını belirlenen hedefe ulaştırır.
5. **Yeni Veri Setinin Oluşturulması:** Rastgele seçilen ve çoğaltılan azınlık sınıfı örnekleri ile orijinal veri seti birleştirilerek yeni bir veri seti oluşturulur.

$$\text{Random Oversampling Oranı} = \frac{\text{Azınlık Sınıfındaki Örnek Sayısı} \times \text{Hedef Oran}}{\text{Azınlık Sınıfındaki Örnek Sayısı}} \qquad (3\text{-}1)$$



Şekil 3.6: Random Oversampling

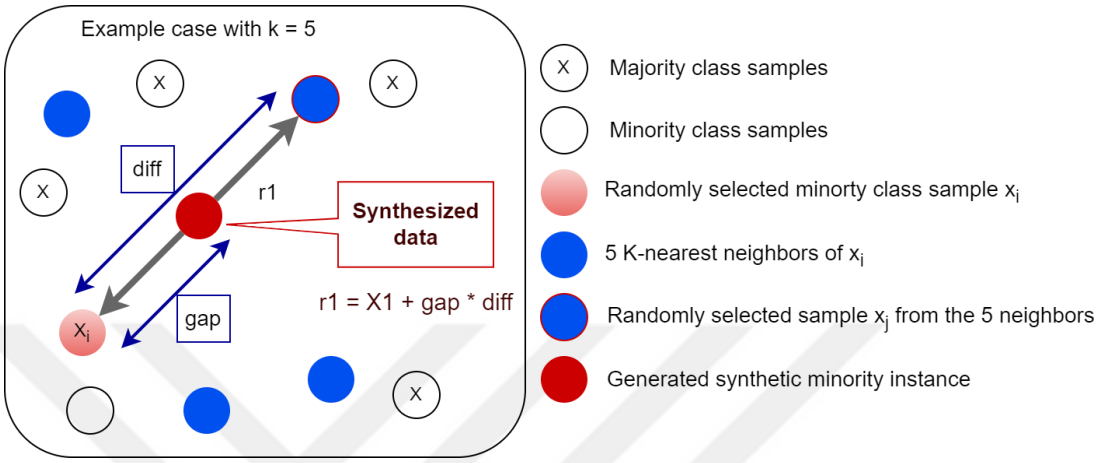
ROS, basit ve hızlı bir çözüm olabilir, ancak veri setinin özelliklerine bağlı olarak bazı dezavantajları vardır. Bu yöntem, çoğunluk sınıfındaki örnekleri modelin ezmesine ve aşırı uyarlamaya yol açabilir. Ayrıca, azınlık sınıfındaki örneklerin tamamını tekrar ekleyerek veri setini çoğaltmak, orijinal veri setinden gelen bilgiyi kaybetme riskini artırabilir. Bu nedenle, ROS tek başına kullanılmadan önce dikkatlice değerlendirilmeli ve diğer yöntemlerle birleştirilerek veya çeşitli dengeleme stratejileriyle birlikte kullanılarak daha etkili sonuçlar elde edilebilir.

3.2.2. SMOTE

Sentetik Azınlık Üst Örnekleme Tekniği (SMOTE), özellikle CIP çözümü için geliştirilen bir üst örnekleme tekniğidir (T. Zhu et al., 2017). Bu yöntem, azınlık sınıfındaki örneklerin sayısını artırmak ve sınıf dağılımını dengelemek için sentetik örnekler oluşturur. SMOTE, özellikle azınlık sınıfındaki örnekler arasında interpolasyon yaparak sentetik örnekler üretir (Chawla et al., 2002). Yetersiz örnekleme yöntemleri, çoğunluk sınıfındaki örnek sayısını azaltarak dengesizliği gidermeye çalışsa da bu yaklaşım bazı önemli bilgilerin kaybına neden olabilir (Dal Pozzolo et al., 2015).

Dolandırıcılık tespiti bağlamında kullanıldığında SMOTE, dolandırıcılık altındaki özellikleri içeren sentetik örnekler oluşturarak azınlık sınıfın, yani hileli işlemlerin,

temsilini artırır. Bu yaklaşım, dengesizlik sorununu hafifletmeye yardımcı olur ve dolandırıcılık tespiti modellerinin daha dengeli bir şekilde eğitilmesine olanak tanır. Şekil 3.7'de açıklanan SMOTE, özünde azınlık sınıfının özelliklerini çeşitlendirmek ve güçlendirmek için sentetik örnekler üreterek modelin genelleme yeteneğini artırmayı amaçlayan etkili bir yöntem olarak öne çıkar.



Şekil 3.7: SMOTE

Üst örnekleme stratejileri, özellikle CIP ile mücadele etmek isteyen birçok veri madenciliği ve makine öğrenimi uygulamasında önemli bir rol oynar. Bunlar arasında SMOTE popüler ve etkili bir yaklaşımdır.

$$x_{new} = x_i + \delta(x_j - x_i) \tag{3-2}$$

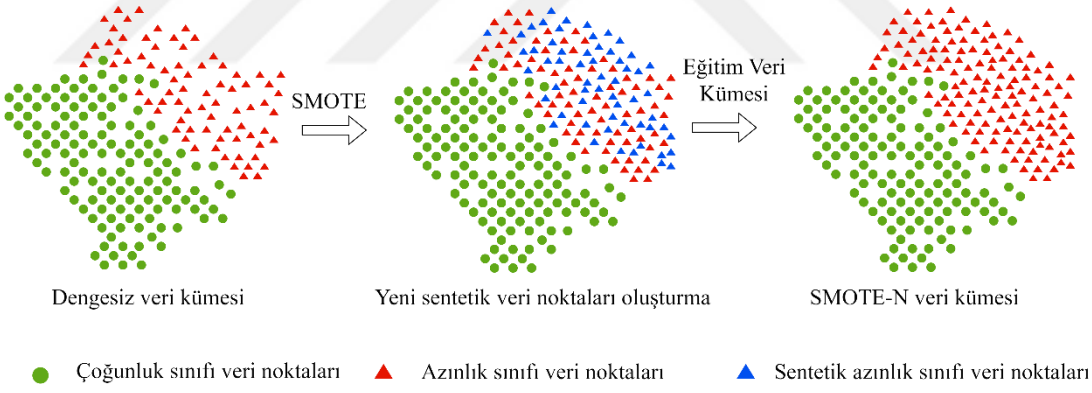
SMOTE örnekleri x_{new} (Denklem (3-2)'e bakınız), azınlık sınıfından iki benzer örneğin (x_i ve x_j) lineer kombinasyonlarıdır. $0 \leq \delta \leq 1$ için; x_j, x_i'nin k en yakın azınlık sınıfından rastgele seçilir.

3.2.3. SMOTE-N

SMOTE for Nominal (SMOTE-N) ya da başka bir adıyla SMOTE for Categorical Variables, Şekil 3.8'de sunulduğu üzere CIP için kullanılan bir veri sentezleme yöntemidir. Geleneksel SMOTE, özellikle sayısal veriler üzerinde çalışırken, SMOTE-N, kategorik (nominal) özelliklere sahip veri setleri için uyarlanmış bir versiyonudur (Fithriasari et al., 2020). SMOTE-N, sınıflandırma problemlerindeki CIP çözümü amacıyla kullanılır ve kategorik özelliklere sahip veri setlerinde sentetik örnekler oluşturur. Geleneksel SMOTE, sayısal özellikler arasında lineer bir

interpolasyon kullanırken, SMOTE-N kategorik verilerdeki özellik değerlerini koruyarak sentetik örnekler üretir. Aşağıda SMOTE-N adımları sırasıyla sunulmuştur.

1. **Komşu Seçimi:** Geleneksel SMOTE gibi, her kategorik örnek için k-en yakın komşular belirlenir.
2. **Azınlık Sınıfı İçin Sentetik Örnek Oluşturma:** Her kategorik örnek için k-en yakın komşular arasından rastgele bir komşu seçilir. Seçilen komşu ile kategorik örnek arasında, kategorik özelliklerin her biri için farklar hesaplanır.
3. **Rastgele Örnek Oluşturma:** Her bir kategorik özellik için farklar arasında rastgele bir ağırlık belirlenir. Ağırlıklar kullanılarak yeni sentetik örnek oluşturulur. Özelliklerin değerleri, orijinal örnek ve seçilen komşu arasındaki farka göre belirlenir.
4. **Sentetik Örneğin Eklenmesi:** Oluşturulan sentetik örnek, azınlık sınıfına eklenir.
5. **Bu İşlemin Tüm Azınlık Sınıfı Örnekleri İçin Yapılması:** Bu işlem, azınlık sınıfındaki her örnek için tekrarlanır.



Şekil 3.8: SMOTE-N

SMOTE-N, özellikle kategorik veri setleri üzerinde çalışırken sınıf dengesizliğini hafifletmek için kullanılır. Ancak dikkatlice uygulanmalıdır, çünkü kategorik özellikler arasındaki farkların rastgele ağırlıklarla çarpılması, sentetik örneklerin gerçek örnekleri doğru bir şekilde temsil etmemesine neden olabilir. Bu nedenle, veri setinin özelliklerine bağlı olarak farklı dengeleme stratejileri ve oversampling teknikleri kullanılabilir.

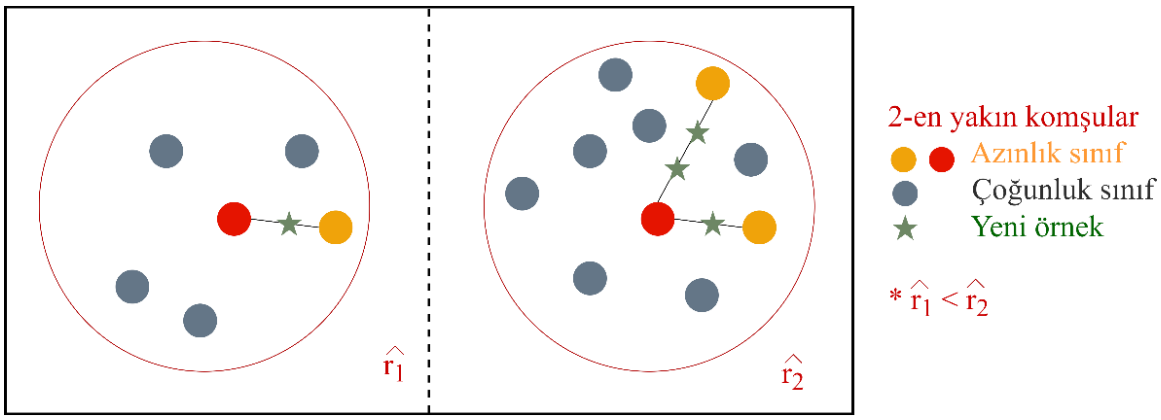
3.2.4. ADASYN

ADASYN üst örnekleme yöntemi, Şekil 3.9'da sunulduğu üzere CIP çözümünde kullanılan bir veri sentezleme tekniğidir. ADASYN, özellikle azınlık sınıfındaki örnek sayısını artırmak ve böylece ML modellerinin daha iyi genelleme yapmasını sağlamak için tasarlanmıştır (Haibo He et al., 2008). Aşağıda ADASYN adımları sırasıyla sunulmuştur.

1. **Her Azınlık Örneği İçin K-En Yakın Komşu Belirleme:** Her azınlık sınıfı örneği için k-en yakın komşular belirlenir.
2. **Her Azınlık Örneği İçin Ağırlık Hesaplama:** Ağırlık, her örnek için aşağıdaki Denklem **(3-3)**'deki formül kullanılarak hesaplanır:

$$w_i = \frac{\text{Azınlık Sınıfı Komşularının Sayısı}}{\text{Toplam Komşuların Sayısı}} \quad (3\text{-}3)$$

   Ağırlık, bir örneğin sentetik örnek oluşturulma ihtiyacını belirlemede kullanılır.
3. **Sentetik Örneklerin Oluşturulması:** Her azınlık sınıfı örneği için ağırlığa göre sentetik örnek oluşturulur. Ağırlığı daha yüksek olan örnekler, daha fazla sentetik örnek oluşturulması ihtimaliyle karşılaşır. Yüksek ağırlıklı örnekler, k-en yakın komşular arasında fark vektörlerini kullanarak sentetik örnekler üretir. Bu, azınlık sınıfındaki örneklerin, çoğunluk sınıfına daha yakın olan bölgelerde daha fazla örneklerle desteklenmesini sağlar.
4. **Sentetik Örneklerin Eklenmesi:** Oluşturulan sentetik örnekler, azınlık sınıfına eklenir.
5. **Bu İşlemin Tüm Azınlık Sınıfı Örnekleri İçin Yapılması:** Bu işlem, azınlık sınıfındaki her örnek için tekrarlanır.



Şekil 3.9: ADASYN

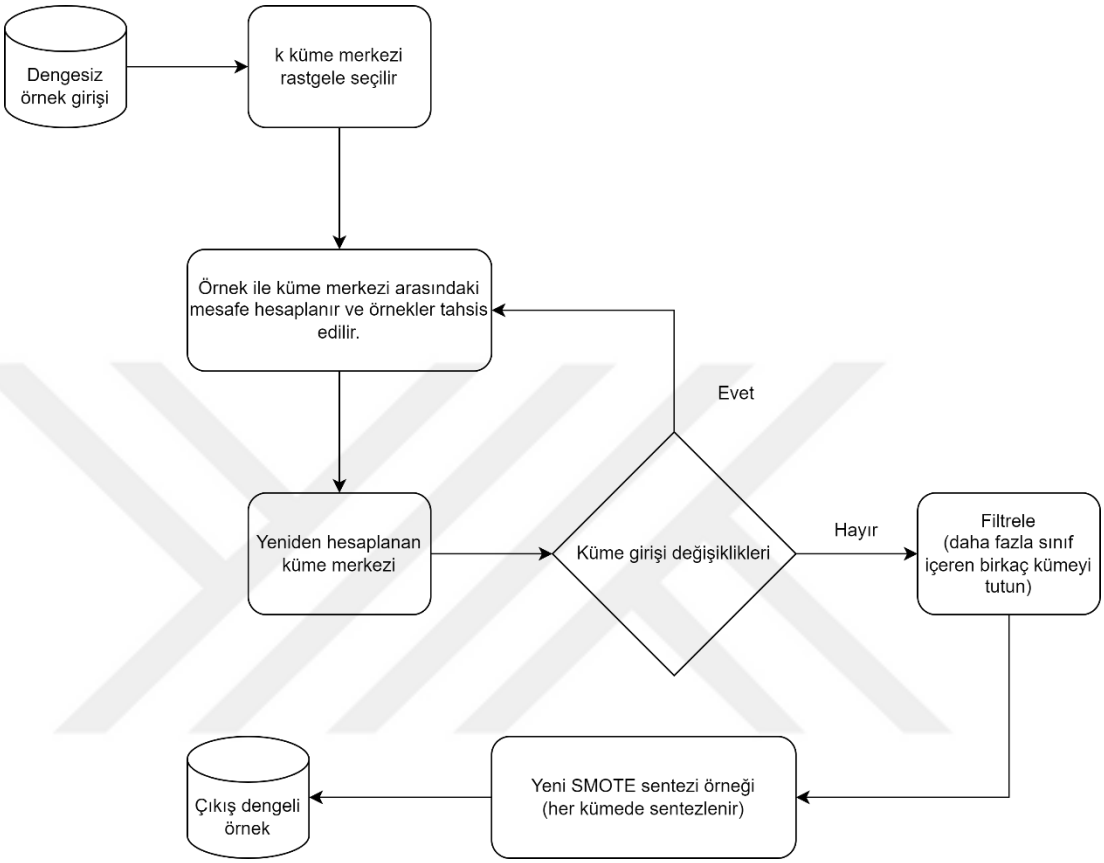
ADASYN, CIP sorunu için SMOTE'a dayanarak, azınlık sınıfındaki örneklerin daha dengeli bir şekilde sentezlenmesini sağlayabilir. Ağırlıkları adaptif olarak hesaplamak, veri setinin özelliklerine bağlı olarak sentetik örnek oluşturma işlemini daha akıllıca yapabilir. Bu sayede, modelin azınlık sınıfındaki nadir durumları daha iyi öğrenmesi ve genelleme yapması amaçlanır.

3.2.5. K-Means SMOTE

K-Means SMOTE, Şekil 3.10'da sunulduğu üzere CIP çözümü için kullanılan bir veri sentezleme yöntemidir. Bu yöntem, özellikle azınlık sınıfındaki örneklerin sayısını artırarak ML modellerinin daha iyi performans göstermesini hedefler. K-Means SMOTE, SMOTE algoritması ile k-ortalama (k-means) kümeleme algoritmasını birleştirerek sentetik örnekler üretir (Fonseca et al., 2021). Bu sayede, sentetik örneklerin daha iyi dağıtılmasını ve azınlık sınıfının daha iyi temsil edilmesini sağlamayı amaçlar. Aşağıda K-Means SMOTE adımları sırasıyla sunulmuştur.

1. **Azınlık Sınıfındaki Örneklerin K-Ortalama ile Kümeleme:** Azınlık sınıfındaki örnekler, k-ortalama algoritması kullanılarak belirli sayıda küme (cluster) içine ayrılır. Bu kümeleme işlemi, örneklerin benzerliklerine dayanarak gruplara ayrılmasını sağlar.
2. **Her Küme İçin Merkez Belirleme:** Her küme için merkez, o kümedeki örneklerin ortalaması olarak belirlenir.
3. **Her Küme İçin SMOTE Uygulaması:** Her küme için SMOTE algoritması uygulanır. Ancak, bu adımda normal SMOTE'tan farklı olarak, sentetik örnekler sadece o küme içindeki örneklerle ilişkilendirilir.

4. **Sentetik Örneklerin Eklenmesi:** Oluşturulan sentetik örnekler, orijinal azınlık sınıfına eklenir.
5. **Bu İşlemin Tüm Kümeler İçin Yapılması:** Bu işlem, her küme için tekrarlanır.



Şekil 3.10: K-Means SMOTE

Kaynak: (Chen & Zhang, 2021)

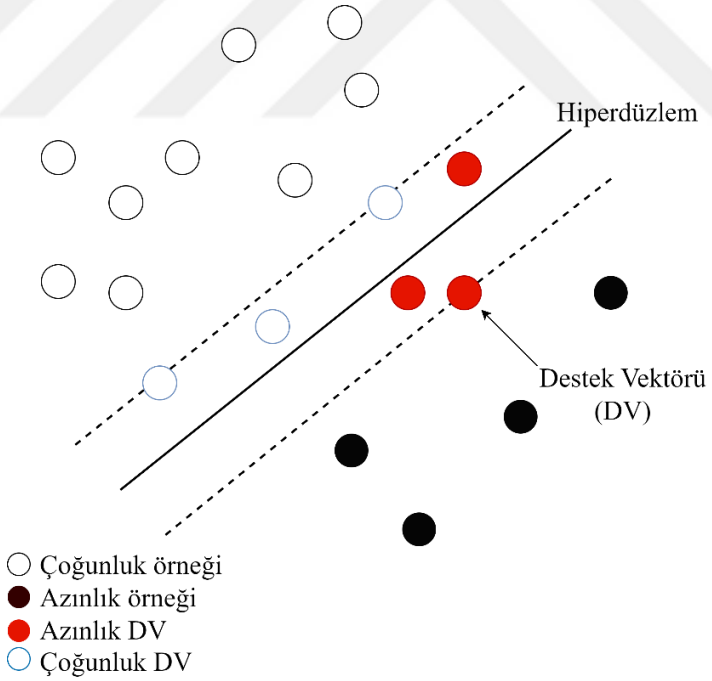
K-Means SMOTE, azınlık sınıfındaki örneklerin kümeleme ile daha homojen bir şekilde gruplandırılmasını sağlar ve SMOTE algoritmasının sentetik örnek üretme mekanizmasını kullanarak çeşitli sentetik örnekler oluşturur. Bu yöntem, CIP çözümü için etkili olabilir, ancak veri setinin özelliklerine bağlı olarak dikkatlice kullanılmalıdır.

3.2.6. SVM SMOTE

SVM-SMOTE, Şekil 3.11'de sunulduğu üzere CIP çözümü amacıyla kullanılan bir veri sentezleme tekniğidir. SVM-SMOTE, SMOTE ve SVM algoritmalarını birleştirerek, azınlık sınıfındaki örneklerin sentetik örneklerle güçlendirilmesini

amaçlar (Q. Wang et al., 2017). Aşağıda SVM-SMOTE adımları sırasıyla sunulmuştur.

1. **Azınlık Sınıfındaki Örneklerin Belirlenmesi:** İlk adım olarak, azınlık sınıfındaki örnekler tespit edilir.
2. **Azınlık Sınıfındaki Örneklerin SVM ile Sınıflandırılması:** Azınlık sınıfındaki örnekler, bir SVM modeli kullanılarak sınıflandırılır. SVM, örneklerin sınıflarını belirleyen bir sınıflandırma modeli oluşturur.
3. **Azınlık Sınıfındaki Örneklerin İncelenmesi:** SVM tarafından yanlış sınıflandırılan azınlık sınıfındaki örnekler belirlenir.
4. **SMOTE Uygulanması:** SMOTE algoritması, yanlış sınıflandırılan azınlık sınıfındaki örnekler üzerinde uygulanır. SMOTE, her bir örneği, o örneğin k-en yakın komşuları arasında sentetik örnekleri oluşturarak artırır.
5. **Sentetik Örneklerin Eklenmesi:** Oluşturulan sentetik örnekler, orijinal azınlık sınıfına eklenir.



Şekil 3.11: SVM-SMOTE

Kaynak: (J.-B. Wang et al., 2021)

SVM-SMOTE, SMOTE algoritmasının yanı sıra SVM kullanılması ile azınlık sınıfındaki örnekleri daha iyi bir şekilde modellemeyi ve öğrenmeyi amaçlar (Wu & Chang, 2005). Bu, özellikle SVM'nin karar sınırlarını daha doğru bir şekilde

ayarlamasına ve azınlık sınıfındaki nadir durumları daha etkili bir şekilde öğrenmesine katkıda bulunabilir. Ancak, bu yöntemin kullanılması veri setinin özelliklerine ve problem bağlamına bağlı olarak dikkatlice değerlendirilmelidir.

3.3. Algoritma Ayarları

Sınıf dengesizliğinden kurtulabilmek için kullanılan algoritma ayarlarından ikisi sınıf ağırlıkları (class weights) ve maliyet duyarlı öğrenme (cost-sensitive learning) teknikleridir. Her iki teknikte, CIP için kullanılan yöntemlerdir, ancak veri setinin özelliklerine ve problem bağlamına bağlı olarak hangi yöntemin daha iyi performans göstereceğini belirlemek gerekmektedir.

3.3.1. Sınıf Ağırlıkları

CIP çözümü için sınıflara ağırlıklar atanabilir. Bu, modelin daha az örneğe sahip sınıfı daha fazla vurgulamasına yardımcı olur (Cano et al., 2013). Sınıf ağırlıkları, modelin eğitimi sırasında farklı sınıflara ait örneklerin önemini belirlemek için kullanılır. Bu ağırlıklar, daha az örnek içeren sınıflara daha yüksek ağırlıklar atanarak modelin bu sınıfları daha fazla önemsemesini sağlar. Örneğin, eğer iki sınıfınız varsa ve birinci sınıf daha az örneğe sahipse, birinci sınıfa daha yüksek ağırlıklar atanabilir.

3.3.2. Cost-Sensitive Learning

CSL, farklı sınıfların hatalarının model tarafından farklı şekilde cezalandırılmasını sağlar. Hata cezaları, sınıfların dengesizlik durumuna göre ayarlanabilir. Yani, daha az örneğe sahip sınıfların hataları daha fazla cezalandırılabilir. Bu yaklaşım, maliyet (cost) kavramını kullanarak, hataların genel maliyetini dengelemeye çalışır (Fernández et al., 2018). CSL, öğrenme algoritmalarına ek maliyet matrisleri ekleyerek veya sınıf ağırlıkları kullanarak uygulanabilir. Bu matris, her bir hata türü (örneğin, false positive ve false negative) için farklı maliyetleri içerir.

3.4. Ensemble Modeller

Ensemble modeller (topluluk yöntemleri), bir dizi sınıflandırıcı oluşturan ve daha sonra tahminlerinin (ağırlıklı) oylarını alarak yeni veri noktalarını sınıflandıran öğrenme algoritmalarıdır (Dietterich, 2000). Ensemble modeller CIP ile başa çıkmak için etkili olabilir. Bu modeller, birden fazla temel modelin kombinasyonunu içeren bir yaklaşımı ifade eder. Random Forest ve Gradient Boosting gibi ensemble modeller, CIP için kullanılabilir. Ensemble modeller, genellikle tek bir modelden daha iyi

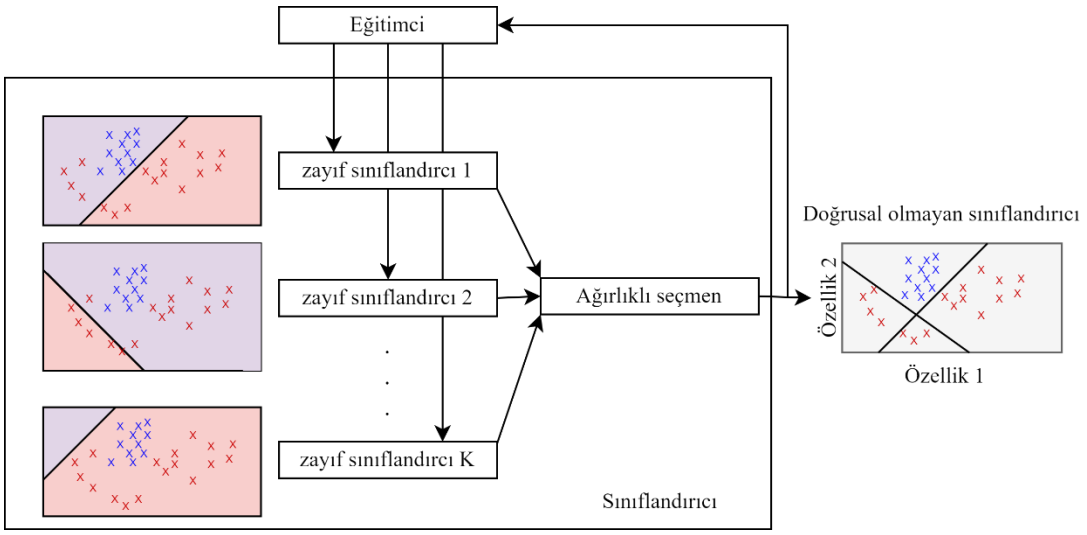
performans sağlayabilir çünkü farklı modellerin güçlü yönlerini birleştirirler. Ancak, her modelin avantajları ve dezavantajları olduğundan en iyi performansı elde etmek için farklı ensemble modelleri arasında deneme yapmak ve parametre ayarlamak önemlidir.

3.4.1. Random Forest

RF, birçok karar ağacının bir araya getirilmesiyle oluşturulan bir ensemble modelidir (Yadav & Pal, 2020). RF, her bir karar ağacını farklı bir alt küme (bootstrap örnekleme) ve farklı özellik alt kümesi (random feature selection) kullanarak eğitir (Özçift, 2011). Bu, her bir ağacın farklı bir perspektiften öğrenmesini sağlar. CIP ile başa çıkmak için, RF sınıf ağırlıklarını (class weights) kullanma yeteneğine sahiptir ve bu sayede daha az örneğe sahip sınıflara daha fazla önem verebilir (Detaylı bilgi 50).

3.4.2. AdaBoost

AdaBoost, zayıf öğrenicileri (genellikle karar ağaçları) bir araya getirerek güçlü bir öğrenici oluşturan bir ensemble algoritmasıdır (Schapire, 2013). Şekil 3.12'de sunulan AdaBoost, her bir öğreniciyi, önceki öğrenicinin hatalarına odaklanarak eğitir. Bu, özellikle azınlık sınıfındaki örnekleri daha fazla vurgulayarak CIP ile başa çıkmaya yardımcı olabilir.



Şekil 3.12: Adaboost sınıflandırıcı

Kaynak: (Z. Wang et al., 2015)

Adaboost sınıflandırıcının temel aşamaları aşağıda sunulmuştur.

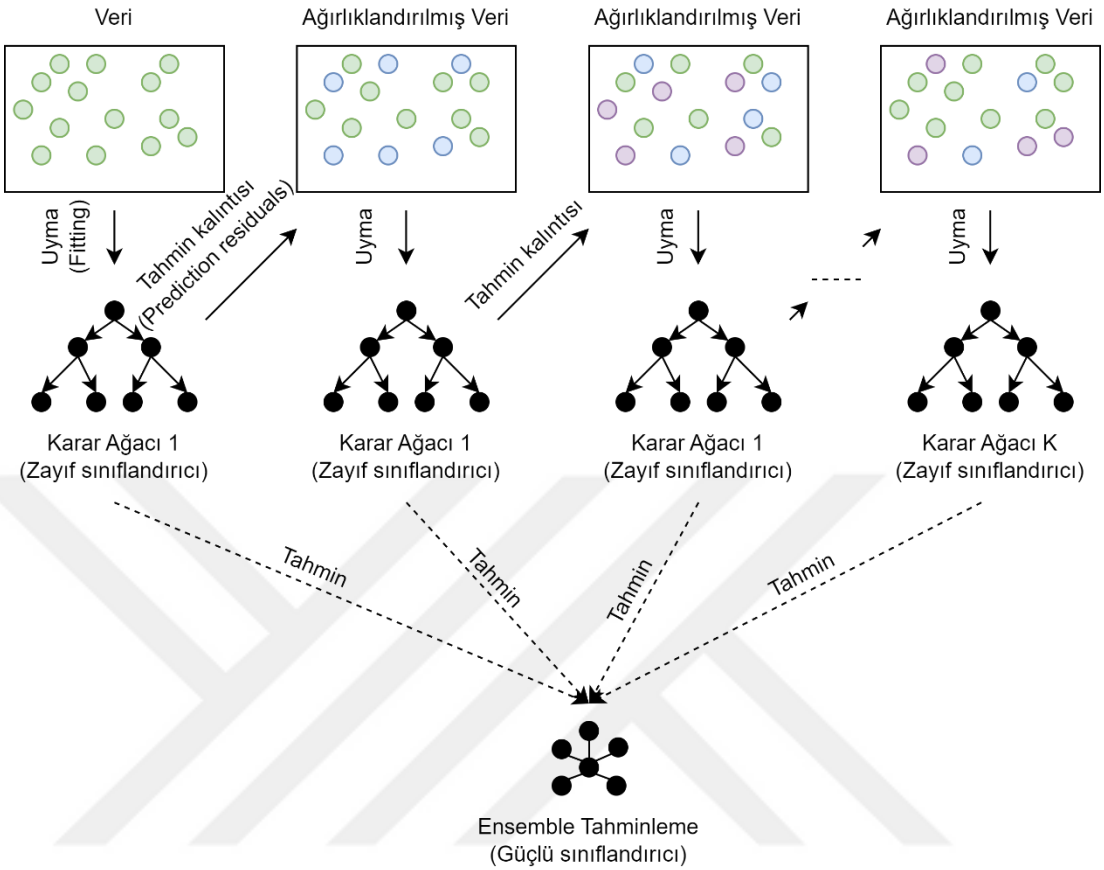
1. **İlk Modelin Eğitimi:** İlk olarak, veri seti üzerinde bir zayıf öğrenen model (genellikle bir karar ağacı) eğitilir. Model, veri setindeki örnekleri iyi tahmin etse bile genellikle düşük bir doğruluk oranına sahiptir.
2. **Hataların Hesaplanması:** İlk modelin eğitimi sonrasında, her bir örnek için modelin yaptığı tahminlerle gerçek etiketler arasındaki hatalar hesaplanır. Hataların büyüklüğü, bir örnek doğru tahmin edildiyse düşük, yanlış tahmin edildiyse yüksektir.
3. **Ağırlıkların Atanması:** Hatalı tahmin edilen örneklerin ağırlığı artırılır. Yani, hatalı tahmin edilen örneklerin üzerine daha fazla odaklanılır. Doğru tahmin edilen örneklerin ağırlığı düşer.
4. **Yeni Modelin Eğitimi:** Ağırlıklarla ayarlanmış veri seti üzerinde bir sonraki zayıf öğrenen model eğitilir. Bu model, önceki modelin yanlış tahmin ettiği örnekler üzerinde daha fazla odaklanarak genellikle daha iyi bir performans gösterir.
5. **Ağırlıklandırılmış Modellerin Birleştirilmesi:** Her modelin performansına göre belirli bir ağırlık atanır. Bu modeller birleştirilirken, daha iyi performans gösteren modellerin daha büyük bir etkiye sahip olması sağlanır.
6. **Adımın Tekrarlanması:** Adım 3'ten sonra, hala hatalar varsa, bu hatalar üzerine odaklanan bir sonraki zayıf öğrenen model eğitilir. Bu adımlar, belirlenen bir iterasyon sayısına veya belirli bir performans eşiğine ulaşılıncaya kadar tekrarlanır.
7. **Final Modelin Oluşturulması:** Belirlenen sayıda zayıf öğrenen model eğitildikten sonra, bu modeller birleştirilerek güçlü bir ensemble modeli elde edilir. Her modelin ağırlığı, performansına ve hatalarına göre belirlenir.

AdaBoost, hatalı tahmin edilen örnekler üzerine odaklanarak bu hataları düzeltmeye çalışır ve modelin genel performansını artırmayı hedefler.

3.4.3. Gradient Boosting

Gradient Boosting, zayıf öğrenicileri (genellikle karar ağaçları) bir araya getirerek güçlü bir öğrenici oluşturan bir ensemble algoritmasıdır (Natekin & Knoll, 2013). Gradient Boosting, önceki ağaçların yaptığı hataları düzeltmeye odaklanarak çalışır.

Şekil 3.13'de gösterildiği üzere Gradient Boosting, CIP ile başa çıkmak için içsel olarak ağırlıklandırma mekanizmalarını kullanabilir.



Şekil 3.13: Gradient Boosting mimarisi

Kaynak: (Zhang et al., 2021)

Gradient Boosting'in temel aşamaları aşağıda sunulmuştur.

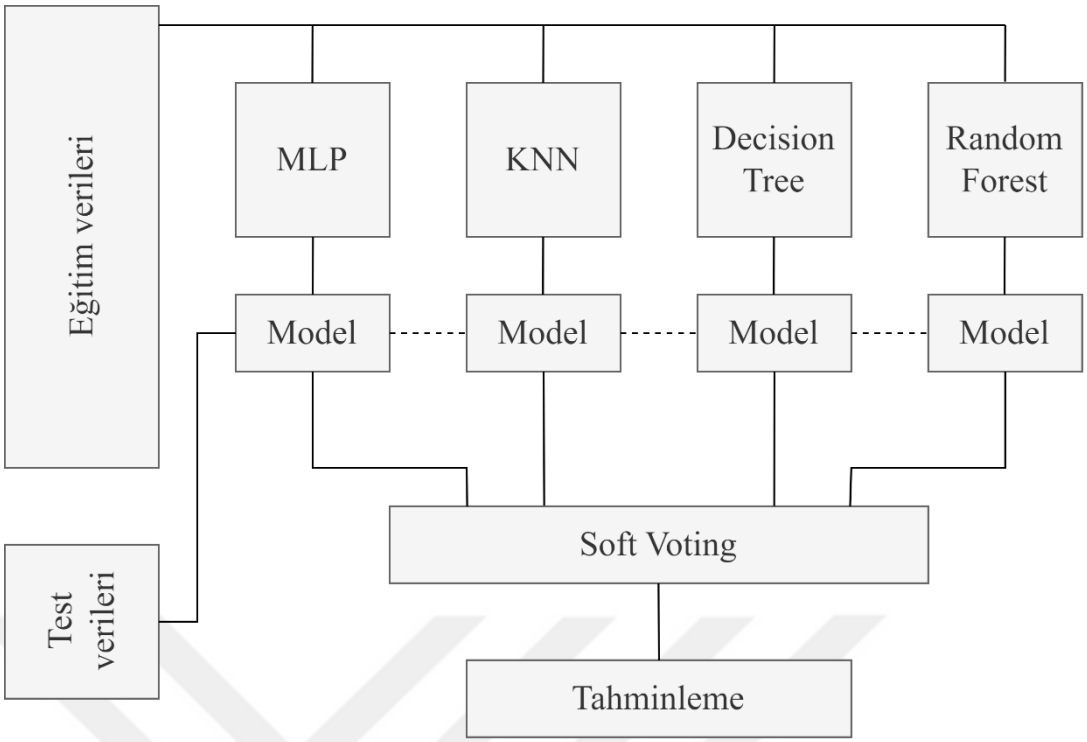
1. **İlk Modelin Eğitimi:** İlk olarak, veri seti üzerinde bir zayıf öğrenen model (genellikle bir karar ağacı) eğitilir. İlk model, veri setindeki örnekleri tahmin eder.
2. **Hataların Hesaplanması:** İlk modelin tahminleri ile gerçek etiketler arasındaki hatalar hesaplanır. Hatalar, her bir örneğin gerçek etiketi ile tahmin edilen etiketi arasındaki farktır.
3. **Hata (Gradient) ile Ağırlıklı Öğrenme:** İlk modelin hatalarına odaklanan bir sonraki model eğitilir. Ancak, bu sefer hatalar üzerine odaklanılır ve bu hataları azaltmaya çalışan bir model oluşturulur. Gradyan iniş (gradient descent) yöntemi kullanılarak, önceki modelin hatalarına göre ağırlıklı olarak yeni bir model oluşturulur.

4. **Ağırlıklı Modellerin Birleştirilmesi:** Yeni model, önceki modelin hatalarını düzeltmeye yönelik olarak eğitildiği için, bu modeller birleştirildiğinde genel hata azalır. Modeller genellikle ağırlıklandırılarak birleştirilir, yani daha düşük hata yapan modeller daha fazla etkiye sahip olur.
5. **İteratif Süreç:** 2-4 adımları, belirli bir iterasyon sayısına veya belirli bir performans eşiğine ulaşılıncaya kadar tekrarlanır. Her iterasyonda, önceki modellerin hatalarını düzelten yeni bir model eklenir.
6. **Öğrenme Oranının Kontrolü:** Öğrenme oranı, her yeni modelin katkısının ne kadar olduğunu kontrol eden bir parametredir. Düşük bir öğrenme oranı, her modelin daha küçük bir etkisi olduğu ve genellikle daha fazla iterasyon gerektiği anlamına gelir.
7. **Final Modelin Oluşturulması:** Belirlenen sayıda iterasyon sonunda, oluşturulan modeller birleştirilerek güçlü bir ensemble modeli elde edilir. Genellikle, her modelin ağırlığı, performansına ve hatalarına göre belirlenir.

Gradient Boosting, özellikle büyük ve karmaşık veri setlerinde güçlü bir performans sergileyen etkili bir öğrenme yöntemidir. XGBoost, LightGBM ve CatBoost gibi kütüphaneler, Gradient Boosting algoritmalarının optimize edilmiş ve hızlandırılmış versiyonlarını içermektedir.

3.4.4. Voting Classifier

Şekil 3.14'de gösterildiği üzere Voting Classifier, farklı temel modellerin çoğunluk oylamasıyla sınıflandırma yaparak ensemble bir model oluşturur (Ruta & Gabrys, 2005). CIP ile başa çıkmak için, temel modellerin sınıf ağırlıklarını uygun bir şekilde belirlemek gerekir.



Şekil 3.14: Soft Voting Classifier örneği

Kaynak: (Awan et al., 2020)

Voting Classifier, bir dizi farklı öğrenme algoritmasının tahminlerini birleştiren bir topluluk öğrenme yöntemidir. Bu yöntem, farklı modellerin güçlü yönlerini birleştirerek daha genel ve dengeli bir tahmin sağlamayı amaçlar. Voting Classifier, sınıflandırma (classification) problemleri için kullanılır ve iki ana türü vardır: Hard Voting ve Soft Voting (Oliveira et al., 2022).

1. **Hard Voting:** Hard Voting, birden fazla öğrenici modelin tahminlerini birleştirirken, her bir modelin sınıf tahminlerini sayarak en fazla oyu alan sınıfı seçer. Örneğin, üç farklı modelin sınıf tahminleri [A, B, B] ise, hard voting yöntemiyle seçilen sınıf "B" olacaktır.
2. **Soft Voting:** Soft Voting, öğrenici modellerin olasılık tahminlerini kullanarak, bu olasılıkları toplayarak ve ortalamalarını alarak bir tahminde bulunur. Her modelin sınıf olasılıkları ağırlıklı olarak toplanır ve en yüksek toplam olasılığa sahip sınıf seçilir.

Voting Classifier'ın aşamaları şu şekildedir:

1. **Farklı Modellerin Seçimi:** Farklı öğrenici modelleri seçilir. Bu modeller genellikle farklı algoritmalar, farklı parametre ayarları veya farklı veri alt kümeleri üzerine eğitilmiş modeller olabilir.
2. **Modellerin Eğitimi:** Seçilen modeller, eğitim verileri üzerinde ayrı ayrı eğitilir.
3. **Hard Voting veya Soft Voting Seçimi:** Hard voting veya soft voting yönteminden biri seçilir. Hard voting, sınıf etiketlerinin doğrudan sayılmasını içerirken, soft voting, sınıf olasılıklarını kullanır.
4. **Tahmin:** Hard voting kullanılıyorsa, her modelin tahminleri sayılarak en fazla oyu alan sınıf seçilir. Soft voting kullanılıyorsa, her modelin sınıf olasılıkları toplanarak en yüksek toplam olasılığa sahip sınıf seçilir.
5. **Performans Değerlendirmesi:** Oluşturulan Voting Classifier'ın performansı, doğruluk, hassasiyet, duyarlılık gibi metrikler kullanılarak değerlendirilebilir.

Voting Classifier, genellikle farklı modellerin güçlü ve zayıf yönlerini birleştirerek daha genel bir model elde etmek için kullanılır. Bu yöntem, ensemble learning (topluluk öğrenmesi) kavramının bir örneğidir ve çeşitli modellerin bir araya getirilerek geçerli bir çözüm elde etmeye çalışır.

3.5. Eğitim Süreci ve Değerlendirme

CIP çözümü için eğitim ve değerlendirme süreçleri, dengesizliği ele alacak ve modelin performansını daha iyi değerlendirecek şekilde ayarlanmalıdır.

3.5.1. Stratified Split

Katmanlı bölünme ile yanıltıcı sonuçların önlenmesi için veri kümesi bölünürken, sınıflar arasındaki oranların eğitim ve test kümelerinde de korunduğundan emin olunmalıdır (Farias et al., 2020).

3.5.2. Doğru Metrik Seçimi

CIP ile mücadele ederken, model performansını değerlendirmek için doğru metrikleri seçmek kritiktir. Bu, genellikle Accuracy gibi basit metriklerin yanıltıcı olabileceği durumlarda önemli hale gelir. Çünkü CIP, modelin çoğunluk sınıfını doğru bir şekilde tahmin etmesini sağlayabilir, ancak azınlık sınıfındaki örnekleri ihmal etmesine neden

olabilir. Bu nedenle, Precision, Recall, F_1-score gibi metrikler, özellikle CIP durumlarında model performansını daha doğru bir şekilde değerlendirmek için tercih edilmelidir (Gu et al., 2009).

Ayrıca, CIP genellikle özel durumlarla ilişkilidir ve bu sorunu çözmek için kullanılacak yöntemler, problem bağlamına ve veri kümesinin özelliklerine göre uyarlanmalıdır. Bu çözümler, örneğin üst örnekleme, sınıf ağırlıkları veya özel algoritmalar gibi çeşitli teknikleri içerebilir. Doğru metrik seçimi ve uygun çözüm stratejileri, modelin gerçek performansını daha iyi anlamak ve CIP ile başa çıkmak için önemlidir.

1. **Performans Metrikleri:** Bir ML modelinin veya sınıflandırma algoritmasının performansını değerlendirmek için kullanılan ölçütlerdir (Erickson & Kitamura, 2021). Bu metrikler, modelin ne kadar iyi çalıştığını ve hangi alanlarda iyileştirme yapılması gerektiğini anlamak için kullanılır.
    a. **Accuracy (ACC):** Bir modelin doğru sınıflandırma oranını ölçen temel bir metriktir. Genel doğruluk, doğru tahmin edilen örneklerin toplam örnek sayısına oranıdır (Y. Liu et al., 2014).

        TP (True Positive): Doğru pozitif tahminlerin sayısı.

        TN (True Negative): Doğru negatif tahminlerin sayısı.

        FP (False Positive): Yanlış pozitif tahminlerin sayısı.

        FN (False Negative): Yanlış negatif tahminlerin sayısı.

    b. **Precision (P):** Pozitif olarak tahmin edilen örneklerin kaçının gerçekten pozitif olduğunu ölçer. Pozitif olarak tahmin edilen örnekler arasındaki yanlış pozitif tahminlerin oranını gösterir (Gray et al., 2011).

    c. **Recall (R):** Gerçekten pozitif olan örneklerin kaçının doğru bir şekilde pozitif olarak tahmin edildiğini ölçer. Gerçek pozitifler arasındaki yanlış negatif tahminlerin oranını ifade eder (Sofaer et al., 2019).

    d. **Specificity (SP):** Gerçekten negatif olan örneklerin kaçının doğru bir şekilde negatif olarak tahmin edildiğini ölçer. Gerçek negatifler arasındaki yanlış pozitif tahminlerin oranını gösterir (Arabzadeh et al., 2020).

e. **F_1 score:** Precision ve recall'in harmonik ortalamasını temsil eden bir performans metriğidir. Bu metrik, precision ve recall arasındaki dengeyi sağlamak için kullanılır (Chicco & Jurman, 2020).

Bu performans metrikleri, sınıflandırma modellerinin performansını değerlendirmek ve modelin gücü ve zayıflıkları hakkında bilgi sağlamak için yaygın olarak kullanılır. Hangi metriğin kullanılacağı, spesifik bir problemin gereksinimlerine ve veri setine bağlı olarak değişebilir.

2. **Confusion Matrix Analizi:** Her bir sınıfın tahminleri ve gerçek değerleri arasındaki ilişkiyi gösteren bir matristir. Bu matris, modelin hangi sınıfları ne kadar doğru veya yanlış tahmin ettiğini gösterir. Özellikle false positive (FP) ve false negative (FN) değerlerine odaklanmak, modelin CIP ile başa çıkma yeteneğini değerlendirmek için önemlidir (Luque et al., 2019).
3. **Precision-Recall Eğrisi:** Precision-recall eğrisi, hassasiyet ve duyarlılığı farklı kesme noktalarında gösteren bir eğridir. Bu eğri, özellikle azınlık sınıfındaki performansı değerlendirmek için kullanışlıdır. Precision-recall eğrisinin altındaki alan (AUC-PR) modelin performansını ölçmek için kullanılabilir (Davis & Goadrich, 2006).
4. **Hassasiyet (Precision) ve Duyarlılık (Recall) Optimizasyonu:** Modelin hassasiyeti artırılırken, duyarlılık düşebilir veya tersi bir durum söz konusu olabilir. İdeal bir model, her iki metriği de yüksek bir şekilde optimize eder. Ancak, bu iki metrik arasında bir denge kurmak gerekir, çünkü birini artırmak diğerini genellikle düşürür. Ayrıca modelin optimize edilecek metriklerini seçerken, problem bağlamına ve hedeflere bağlı olarak hassasiyet, duyarlılık veya F_1 puanını seçmek önemlidir.
5. **Eşik Değer Ayarlanması:** CIP için modelin tahminlerini yaparken kullanılan eşik değeri önemlidir. Eşik değeri, modelin sınıfları tahmin etme hassasiyetini ve duyarlılığını etkileyebilir. Eşik değeri, hassasiyet ve duyarlılık arasında bir denge kurmaya yardımcı olabilir.
6. **MCC:** Sınıflandırma modellerinin performansını ölçen bir metriktir. MCC, özellikle dengesiz sınıflar veya eğitim verilerindeki dengesiz sınıf dağılımlarıyla başa çıkmak için kullanışlı bir metrik olarak kabul edilir. MCC,

dengesiz sınıflara sahip veri setlerinde daha dengeli sonuçlar verir. Bu, azınlık sınıfındaki örnekleri yanıltıcı olmayan bir şekilde değerlendirmesini sağlar. MCC hem duyarlılık hem de hassasiyet değerlerini içerir. Bu sayede modelin her iki metrik arasında bir denge kurmasını sağlar. MCC, CIP durumlarında doğru bir performans gösterir ve hesaplama stabilitesi açısından güvenilirdir (Boughorbel et al., 2017).

DÖRDÜNCÜ BÖLÜM

MAKİNE ÖĞRENMESİ

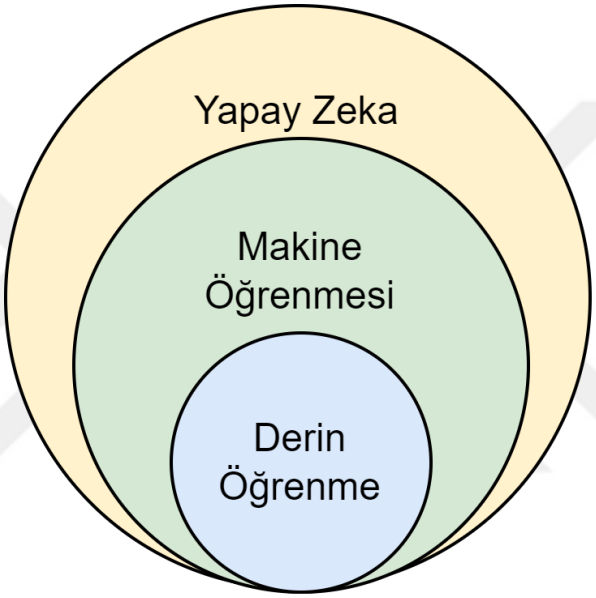
ML, bilgisayar sistemlerine insan müdahalesi olmadan öğrenme ve geliştirme yeteneği kazandıran bir AI dalıdır (Moustakis & Herrmann, 1997; Gillies et al., 2016). Bu disiplin, büyük miktarda veri kullanarak algoritmaların öğrenmesine dayanır ve modelin belirli görevleri gerçekleştirebilmesi için veri setlerindeki örüntüleri analiz etmesine olanak tanır (Zanoni et al., 2015). ML, temel olarak dört ana bileşeni içerir. İlk olarak, geniş veri setleri bu algoritmaların öğrenme sürecinde kullanılır. Ardından, bu veriler üzerinde çalışan çeşitli algoritmalar, belirli görevleri gerçekleştirmek üzere tasarlanan modelleri oluşturur. Bu modeller, veri setlerindeki örüntüleri tanımak ve genelleme yeteneği kazanmak için eğitilir. Eğitim süreci, modelin etiketli veri setleri üzerinde öğrenmesini ve belirli görevleri gerçekleştirmesini sağlar. Eğitim sürecinin ardından, model, genelleme yeteneğini test etmek ve performansını değerlendirmek üzere ayrı bir doğrulama veri seti üzerinde test edilir. Bu değerlendirme, modelin gerçek dünya verileri üzerindeki başarısını doğrulamak için önemlidir (Brink et al., 2016).

ML çeşitli tiplerde uygulanabilir. Gözetimli öğrenme, etiketli veri setleri üzerinde çalışır ve bir girdiye karşılık gelen doğru çıkışı öğrenir. Gözetimsiz öğrenme ise etiketlenmemiş veri setleri üzerinde çalışır ve veri içindeki örüntüleri öğrenir (Alloghani et al., 2020). ML uygulama alanları oldukça geniştir. Görüntü işlemeden doğal dil işlemeye, sağlık sektöründen finansa, üretim ve lojistikten oyun geliştirmeye kadar birçok endüstri ve alanda kullanılmaktadır. Sürekli olarak gelişen algoritmalar ve teknikler ile ML daha da güçlü ve geniş kapsamlı hale gelmektedir.

4.1. Ortaya Çıkış

Şekil 4.1'de AI ve DL ile ilişkisi gösterilen ML terimi, kökenini 1959 yılına dayandıran ve IBM çalışanı Arthur Samuel tarafından icat edilen bir kavramdır. Samuel, bilgisayar oyunları ve AI alanındaki öncü çalışmalarıyla tanınırken, ML terimi de kendi kendine eğitim veren bilgisayarları tanımlamak için kullanılmaya başlandı (Samuel, 1959). Bu dönemde, Raytheon Company tarafından geliştirilen bir deneyimsel "öğrenme makinesi" olan Cybertron, temel takviye öğrenimini kullanarak sonar sinyallerini, elektrokardiyogramları ve konuşma kalıplarını analiz etmeyi

amaçlıyordu (Yaqoob et al., 2023). 1960'ların başlarında, bu makine, kalıpları tanımak için bir insan operatör tarafından eğitildi ve yanlış kararları düzeltmek için bir "aptal" düğmesiyle donatıldı. ML araştırmalarına ilişkin temsili bir kitap olan Nilsson'un "Öğrenme Makineleri" kitabı, 1960'ların başlarında model sınıflandırması üzerine odaklandı (Nilsson, 1965). Örüntü tanıma konusundaki ilgi, 1970'lerde Duda ve Hart'ın çalışmalarıyla devam etti (Duda et al., 1973). 1981'de ise bir bilgisayar terminalinden 40 karakteri tanımayı öğrenen bir sinir ağının öğretme stratejileri kullanarak başarı elde ettiği bir rapor sunuldu. Bu tarihler, ML disiplininin gelişiminde önemli kilometre taşlarını temsil etmektedir.



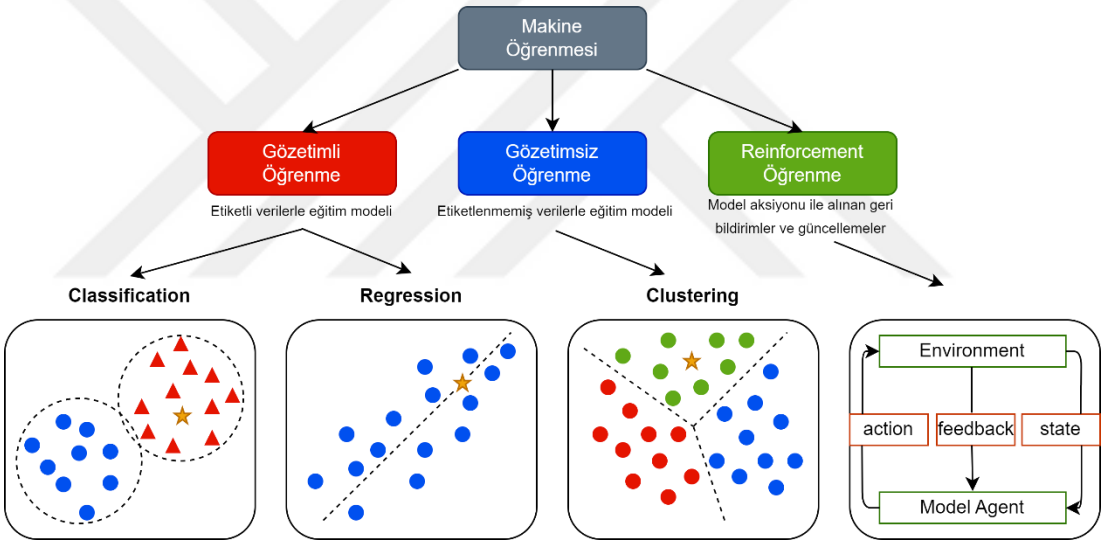
Şekil 4.1: Yapay zekâdan derin öğrenmeye
Kaynak: (Sarker, 2022)

ML, Tom M. Mitchell tarafından öne sürülen bir tanıma göre, bir bilgisayar programının deneyimden öğrendiği ve bu öğrenme süreciyle ilgili görevlerde performans gösterdiği bir alandır (Carbonell et al., 1983). Bu tanım, bir programın deneyim (E) ile belirli görev sınıflarını (T) öğrenip, performansını ölçtüğü bir metrik (P) kullanarak gelişimini ifade eder. Bu perspektif, ML çalışmalarını bilişsel süreçlerden ziyade işlevsel bir bakış açısıyla ele almaktadır. Bu, Alan Turing'in klasik sorusu olan "Makineler düşünebilir mi?" sorusuna karşılık olarak, "Bizim yaptıklarımızı makineler yapabilir mi?" sorusunu öne çıkarır. Günümüzde makine öğreniminin iki temel amacı, geliştirilen modellerle veri sınıflandırmak ve bu modelleri kullanarak gelecekteki sonuçlar hakkında tahminlerde bulunmaktır. Örneğin, bir dermatoloji uygulamasında, gözetimli öğrenme algoritmaları kullanılarak

bilgisayarın kanserli benleri sınıflandırması öğretilebilir. Aynı şekilde, finans alanında bir makine öğrenimi algoritması, geçmiş verilere dayanarak yatırımcıya gelecekteki potansiyel tahminler hakkında bilgi sağlayabilir.

4.2. Öğrenme Türleri

Şekil 4.2'de gösterildiği üzere ML alanında öğrenme türleri, insan öğrenimini taklit etmek için verileri ve algoritmaları kullanarak makinelerin zaman içinde gelişmesine, tahminler veya sınıflandırmalar yaparken veya veriye dayalı içgörüleri ortaya çıkarırken giderek daha doğru hale gelmesine olanak tanır (Bhalekar & Shaikh, 2019). ML, genel olarak şu şekilde öğrenme türlerine ayrılır: gözetimli öğrenme (supervised learning), yarı gözetimli öğrenme (semi-supervised learning), gözetimsiz öğrenme (unsupervised learning) ve pekiştirmeli öğrenme (reinforcement learning) (Sarker, 2021).



Şekil 4.2: ML ile öğrenme türleri

Kaynak: (Peng et al., 2021)

4.2.1. Gözetimli Öğrenme

Gözetimli öğrenme (supervised learning), AI ve ML alanında kullanılan temel bir öğrenme türüdür. Bu öğrenme yöntemi, bilgisayar modellerinin girdi verileriyle birlikte bu verilerin istenen çıktıları (etiketleri veya hedefleri) öğrenmeye çalıştığı bir süreçtir. Gözetimli öğrenme, verilerin temsilini öğrenme, sınıflandırma ve regresyon gibi çeşitli görevler için kullanılır. Gözetimli öğrenme, veri madenciliği, görüntü işleme, doğal dil işleme, tıp, otonom sürüş, öneri sistemleri ve daha birçok uygulama

alanında önemli bir rol oynar (Cunningham et al., n.d.; Hastie et al., 2009; B. Liu, 2011).

Gözetimli öğrenme, etiketli verilerle çalışır. Veriler, giriş özelliklerinden oluşur ve her bir veri örneği ile ilişkilendirilen bir hedef çıktı (etiket) bulunur (Nasteski, 2017). Bu etiketler, modelin istenen sonucu öğrenmesine yardımcı olur. Örneğin, bir görüntü işleme görevinde, görüntüler giriş verilerini temsil ederken etiketler, görüntülerin içerdiği nesnelerin türlerini veya sınıflarını ifade edebilir. Gözetimli öğrenme süreci, bir modelin eğitimi ile başlar. Model, verileri analiz edip giriş verileriyle hedef çıktıları eşleştirme yeteneğini öğrenir. Eğitim sırasında model, bir kayıp fonksiyonu kullanarak tahminlerini gerçek etiketlerle karşılaştırır. Amacı, bu kaybı minimize ederek doğru tahminler yapma yeteneğini artırmaktır. Gözetimli öğrenme için birçok farklı model türü kullanılabilir. Örnekler arasında karar ağaçları (DT), RF, SVM ve daha birçok algoritma bulunur. Modelin türü, verinin yapısı ve görevin karmaşıklığına bağlı olarak seçilir (Jiang et al., 2020).

Gözetimli öğrenme, temel olarak iki ana görevi çözmek için kullanılır: sınıflandırma ve regresyon (Sen et al., 2020).

- **Sınıflandırma:** Sınıflandırma görevi, giriş verilerini belirli sınıflara veya kategorilere ayırmayı amaçlar (Michie et al., 1994). Örneğin, e-postaları "spam" ve "spam değil" olarak sınıflandırmak, bir sınıflandırma örneğidir.
- **Regresyon:** Regresyon görevi, giriş verileri ile sürekli bir sayısal çıktı arasındaki ilişkiyi modellemeyi hedefler (Rong & Bao-wen, 2018). Örneğin, ev fiyatlarını evin özelliklerine dayalı olarak tahmin etmek bir regresyon görevine örnektir.

Modelin eğitim süreci, veri setinin bir eğitim kümesi ve bir doğrulama kümesi olarak bölünmesini içerir (Nasteski, 2017). Model, eğitim verilerini kullanarak öğrenilir ve ardından doğrulama verileri üzerinde değerlendirilir. Değerlendirme, modelin performansını ölçmek ve aşırı uydurmayı (overfitting) önlemek için önemlidir. Gözetimli öğrenme, geniş bir uygulama yelpazesi ile verilerden anlamlı bilgi çıkarmak ve öngörü yapmak için güçlü bir araçtır. Verilerin etiketlenmiş olması, modelin doğru sonuçlar üretebilmesi için kritik bir öneme sahiptir (J. Wei et al., 2019).

4.2.2. Gözetimsiz Öğrenme

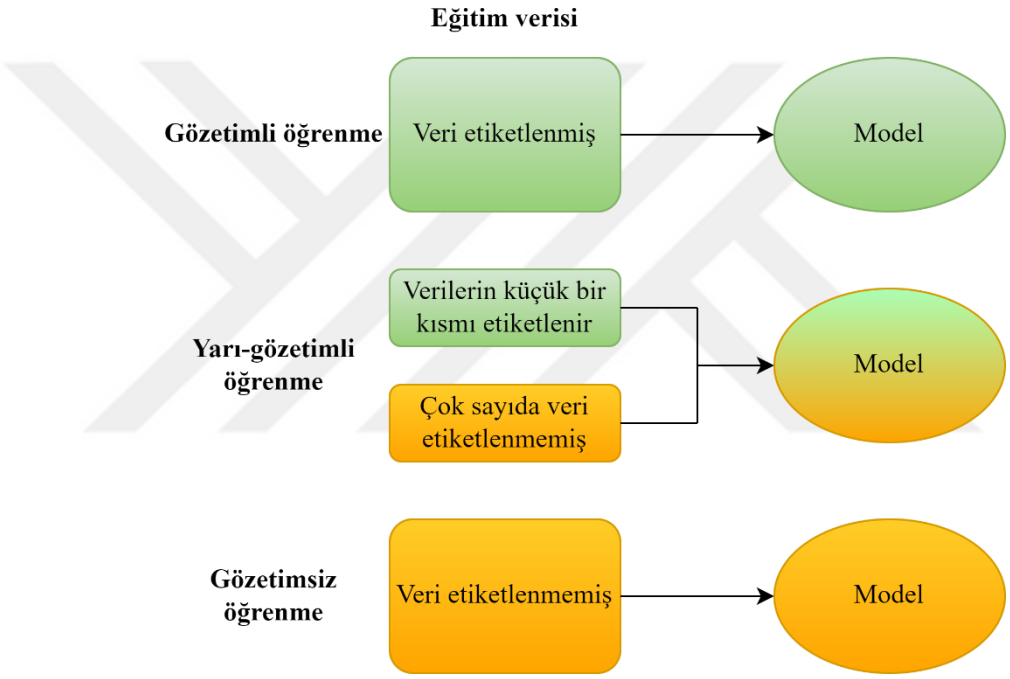
Gözetimsiz öğrenme, bir algoritmanın etiketlenmemiş verileri kullanarak kalıpları ve ilişkileri keşfettiği bir tür ML tekniğidir (Ghahramani, 2004). Gözetimli öğrenmenin aksine, gözetimsiz öğrenme algoritmaya etiketli hedef çıktılar sağlamayı gerektirmez (Hastie et al., 2009b; Mahesh, 2020). Gözetimsiz öğrenmenin temel amacı genellikle veriler içindeki gizli kalıpları, benzerlikleri veya kümelerini keşfetmektir (Corral et al., 2007) bunlar daha sonra veri araştırması, görselleştirme, boyut azaltma ve daha fazlası gibi çeşitli amaçlar için kullanılabilir. İki ana gözetimsiz öğrenme kategorisi vardır: Kümeleme ve boyut azaltma.

Gözetimsiz öğrenim, segmentasyon, anormallik tespiti ve veri araştırması gibi görevler için kullanılır (Omar et al., 2013). Etiketlenmiş verilere ihtiyaç duymaz ve veri etiketleme zahmetini azaltır. Bununla birlikte gözetimsiz öğrenimde etiketleri kullanmadan model çıktısının kalitesini tahmin etmek zor olabilir. Küme yorumlanabilirliği açık olmayabilir ve anlamlı yorumlara sahip olmayabilir. Ham verilerden anlamlı özellikler çıkarmak için kullanılabilecek otokodlayıcı ve boyut azaltma gibi tekniklere sahiptir. Gözetimsiz öğrenmenin bazı yaygın uygulamaları şunlardır (Usama et al., 2019):

- **Kümeleme:** Benzer veri noktalarını kümeler halinde gruplandırmayı sağlar.
- **Anormallik tespiti:** Verilerdeki aykırı değerleri veya anormallikleri belirlemeyi sağlar.
- **Boyut azaltma:** Temel bilgileri korurken verinin boyutunu azaltmayı sağlar.
- **Öneri sistemleri:** Geçmiş davranışlar veya tercihlere göre önerim sunulmasını sağlar.
- **Yoğunluk tahmini:** Verilerin olasılık yoğunluk fonksiyonunun tahmin edilebilmesi sağlar.
- **Veri ön işleme:** Veri temizleme, eksik değerlerin atanması ve veri ölçeklendirme gibi veri ön işleme görevlerini yerine getirebilir.
- **Keşifsel veri analizi (EDA):** Belirli görevleri tanımlamadan önce verileri keşfetmeyi ve öngörüler elde etmeyi sağlar.

4.2.3. Yarı Gözetimli Öğrenme

Yarı gözetimli öğrenme, Şekil 4.3'de sunulduğu üzere ML alanında etiketli ve etiketsiz verilerin bir arada kullanıldığı bir öğrenme yaklaşımıdır (Learning, 2006; Pise & Kulkarni, 2008; Ouali et al., 2020; Cao et al., 2021). Bu yöntem, etiketli veri elde etmenin maliyetli, zaman alıcı veya kaynak yoğun olduğu durumlarda özellikle avantajlıdır. Veri kümesinin etiketlenmesi zor veya pahalı olduğunda, yarı gözetimli öğrenme, sınırlı etiketli verilerle daha geniş bir etiketsiz veri kümesinden öğrenme yeteneği sunarak modelin performansını artırabilir. Bu yaklaşım, özellikle büyük veri setlerinde etiketleme sürecinin zorluğu veya yüksek maliyeti nedeniyle tercih edilir.



Şekil 4.3: Yarı gözetimli öğrenme

Bu yöntem, çeşitli veri türlerine ve senaryolara uygulanabilir. Ancak, yarı gözetimli yöntemlerin uygulanması genellikle diğer öğrenme yaklaşımlarına göre daha karmaşık olabilir. Bu yöntemin başarısı, hâlâ elde edilmesi zor veya maliyetli olan bazı etiketli verilere ihtiyaç duyduğu durumlarda sınırlanabilir. Etiketlenmemiş veriler, modelin performansını etkileyebilir ve bu durum, özellikle her zaman mevcut olmayan etiketli verilere erişimdeki zorluklar nedeniyle ortaya çıkabilir. Yarı gözetimli öğrenme, etiketleme sürecindeki zorlukları aşarak modelin güçlü genelleme yeteneklerinden yararlanma potansiyeli sunar.

4.2.4. Reinforcement Öğrenme

Reinforcement öğrenimi, makinelerin çevreleriyle etkileşime geçerek eylemler ürettiği ve bu eylemlerin sonuçlarını deneme, yanılma ve gecikme yoluyla keşfettiği bir öğrenme yöntemidir (Barto, 1997; Kaelbling et al., 1996; Y. Li, 2017). Bu yöntem, belirli bir bağlam içinde ideal davranışı otomatik olarak belirleyebilen makinelerin performansını en üst düzeye çıkarmak amacıyla kullanılır. Reinforcement öğrenme, özellikle öngörülemeyen ve değişken ortamlarda karar alma yeteneği gerektiren uygulamalarda kritik bir rol oynar. Deneme-yanılma süreçleri ve gecikme faktörleri, bu öğrenme yaklaşımının temel özellikleridir. Bu yöntem, makinelerin etraflarındaki dünyayı keşfetmelerini ve optimize etmelerini sağlayarak adaptasyon yeteneklerini artırır.

Reinforcement öğrenimi, özellikle karmaşık problemleri çözmek ve uzun vadeli hedeflere ulaşmak için ideal olan bir öğrenme yaklaşımıdır (Wiering & van Otterlo, 2012). Bu metot, otonom karar verme mekanizmaları için uygun bir çerçeve sunar, çünkü makineler bir dizi karar almayı öğrenir ve geleneksel tekniklerle çözülemeyen görevleri başarıyla gerçekleştirebilir. Ancak, bu avantajların yanında Reinforcement öğrenimi, hesaplama açısından maliyetli ve zaman alıcı olabilir. Uygulama alanına bağlı olarak çok miktarda veri ve hesaplama gücü gerektirdiğinden, basit problemler için tercih edilmez. Bu durum, yöntemin kullanımını sınırlayabilir ve maliyeti artırabilir (Gosavi, 2004). Buna rağmen, zorlu görevlerin üstesinden gelme yeteneği, özellikle uzun vadeli ve karmaşık sonuçlara ulaşma noktasında Reinforcement öğrenimini değerli kılar.

4.3. Modeller

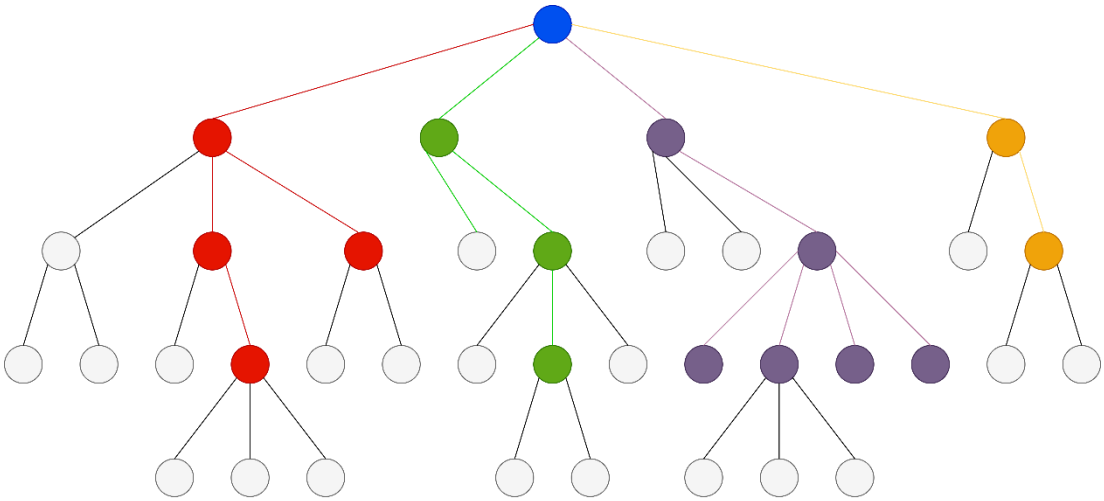
ML, eğitim verileri üzerinde öğrenen ve bu öğrenmeyi kullanarak yeni verilerle tahminlerde bulunan çeşitli model türlerini içeren bir süreçtir. Bu modeller, genellikle karmaşık algoritmalar kullanarak eğitildikleri veri setlerinden örüntüleri çıkarmayı amaçlarlar. Bu öğrenilen örüntüler, daha sonra öngörü yapmak, sınıflandırma yapmak veya belirli görevleri gerçekleştirmek için kullanılabilir. ML modelleri arasında popüler olanlar arasında SVM, DT, RF, k-NN (k en yakın komşu), doğrusal regresyon ve daha pek çok algoritma bulunmaktadır (Mishra & Rath, 2021). Bu modeller, geniş bir uygulama yelpazesi üzerinde kullanılarak, çeşitli endüstrilerde ve disiplinlerde başarıyla uygulanabilir.

4.3.1. Decision Tree

DT, veri madenciliği ve ML alanlarında sıklıkla kullanılan bir sınıflandırma ve regresyon algoritmasıdır (Kingsford & Salzberg, 2008; de Ville, 2013). Bu algoritma, sınıflandırma ve regresyon görevlerini gerçekleştirebilen bir model oluşturur. Şekil 4.4'de gösterildiği gibi temelde ağaç yapısına dayanan bir model olan karar ağacı, kök düğümden başlayarak her düğümde belirli bir özellik veya öznitelik üzerinde bir test gerçekleştirir. Bu testler, veri noktalarını belirli kriterlere göre sınıflandırmak veya sayısal tahminler yapmak için kullanılır. Karar ağacında iki temel düğüm türü bulunur (Magee, 1964; Kotsiantis, 2013):

- **İç Düğümler (Inner Nodes):** Bu düğümler, özellik testlerini temsil eder ve bu testlerin sonuçlarına göre ağaç dallara ayrılır. Her dal, belirli bir alt kümenin temsilcisidir.
- **Yaprak Düğümler (Leaf Nodes):** Bu düğümler, sonuç sınıflandırmasını veya tahminini içerir. Örneğin, bir karar ağacı veri noktalarının yaşını belirli bir sınıra göre kontrol edebilir ve sonuç olarak bu veri noktalarını belirli sınıflara atayabilir.

Bu esnek ve anlaşılır yapısıyla DT veri analizi ve model yorumlaması için ideal bir araç haline getirir. DT, eğitim sürecinde veri kümesindeki her düğümün en uygun özellik seçimini ve sınıflandırma kriterini belirleyerek oluşturulur. Bu süreç, ağacın veri noktalarını daha etkili bir şekilde sınıflandırabilmesi için en iyi özellikleri ve kararları seçmesini içerir. Bu seçim, entropi, Gini impurity veya diğer ölçütler gibi belirli kriterlere dayanabilir. DT genişliği ve derinliği, modelin karmaşıklığını belirler. Derin bir ağaç, veriye daha fazla uyum sağlayarak daha karmaşık bir model oluştururken, geniş bir ağaç daha basit bir model ortaya koyar. Modelin en uygun yapısını belirlemek, genellikle ağacın aşırı uydurmasını önlemek ve genelleme performansını artırmak için dikkatlice ayarlanan bir süreçtir. Bu şekilde, karar ağaçları, veri setlerindeki karmaşıklığı anlamak ve yorumlamak için güçlü bir araç oluşturur.



Şekil 4.4: Decision Tree

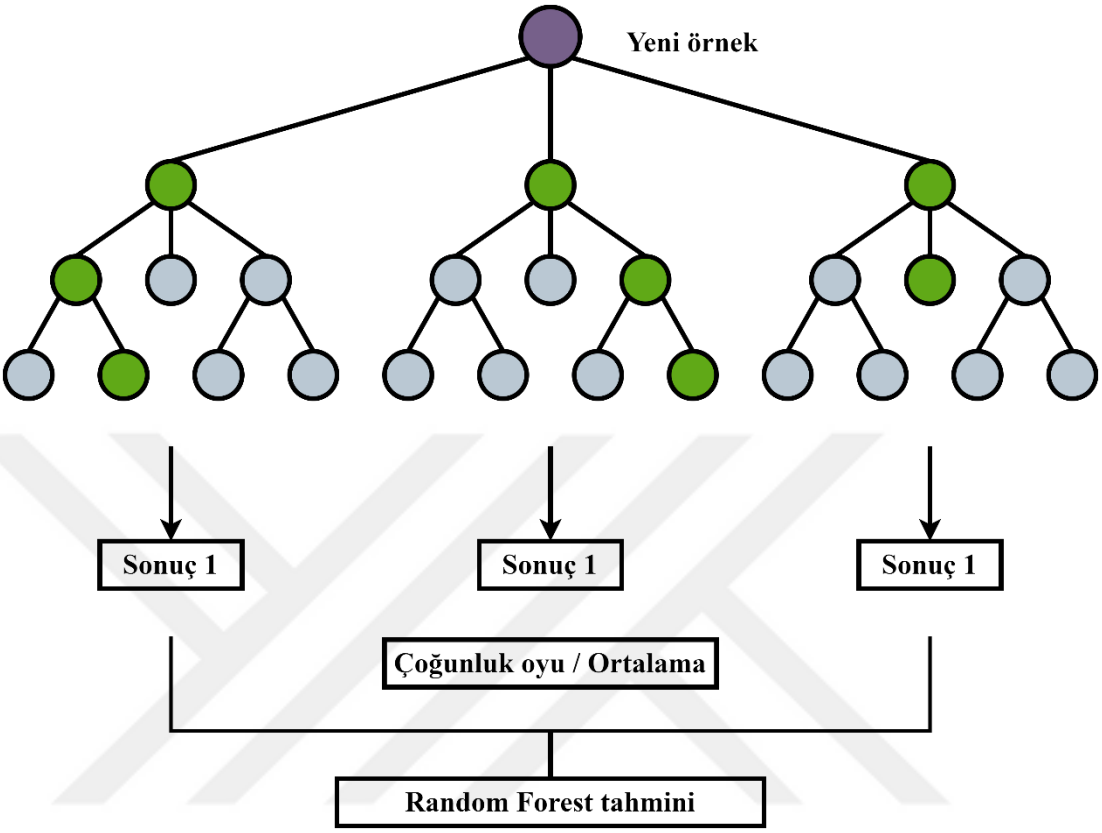
Kaynak: (C. Wang et al., 2019)

Eğitim sürecinden sonra, bir DT yeni veri örneklerini sınıflandırmak veya sayısal tahminler yapmak için kullanılabilir. Ağaç, kök düğümünden başlayarak her düğümdeki testlere göre veri noktasını ilgili dala yönlendirir ve nihayetinde bir yaprak düğümünde sonuca ulaşır. Karar ağaçları, basit ve anlaşılır modeller oluşturabilme avantajına sahiptir. Ancak, aşırı öğrenmeye yatkın olabilirler, bu nedenle ağacın derinliği dikkatlice kontrol edilmelidir. Ayrıca, CIP gibi durumlarla başa çıkmak için geliştirilmiş tekniklerle birleştirilebilirler. Genellikle topluluk tabanlı sınıflandırma ve tahmin görevlerinde kullanılan DT, çeşitli uygulama alanlarında etkili ve esnek bir çözüm sunar.

4.3.2. Random Forest

RF, sınıflandırma ve regresyon görevlerinde kullanılan etkili bir ML algoritmasıdır. Şekil 4.5'de sunulan bu yöntem, birçok karar ağacını bir araya getirerek daha güçlü ve kararlı bir model oluşturur. Her DT, farklı veri alt kümesi ve rastgele özellik seçimleri kullanılarak eğitilir (Rigatti, 2017; Biau & Scornet, 2016; Qi, 2012). Bu çeşitlilik, modelin aşırı uydurmaya karşı direnç göstermesini sağlar. DT, veri noktalarını sınıflandırmak veya sayısal tahminler yapmak için kullanılır. Her ağaç, veriye dayanarak bağımsız tahminlerde bulunur. RF, her karar ağacının farklı bir alt küme ve rastgele özelliklerle eğitildiği bir eğitim sürecini içerir. Bu, her ağacın benzersiz bir bakış açısına sahip olmasını sağlar. Ağaçlar, veri kümesi üzerinde bootstrap örnekleme kullanılarak oluşturulur, bu da her ağacın eğitim verilerini rastgele seçilen veri noktalarıyla çeşitlendirmesini sağlar. Ayrıca, her düğümde en iyi bölünme yerine

rastgele bir alt küme özellikleri kullanılması, her ağacın farklı özelliklere odaklanmasını sağlar. Bu özellikler ile RF yüksek performans ve genelleme sağlayabilir (Y. Liu et al., 2012; Schonlau & Zou, 2020; Oshiro et al., 2012).



Şekil 4.5: Random Forest

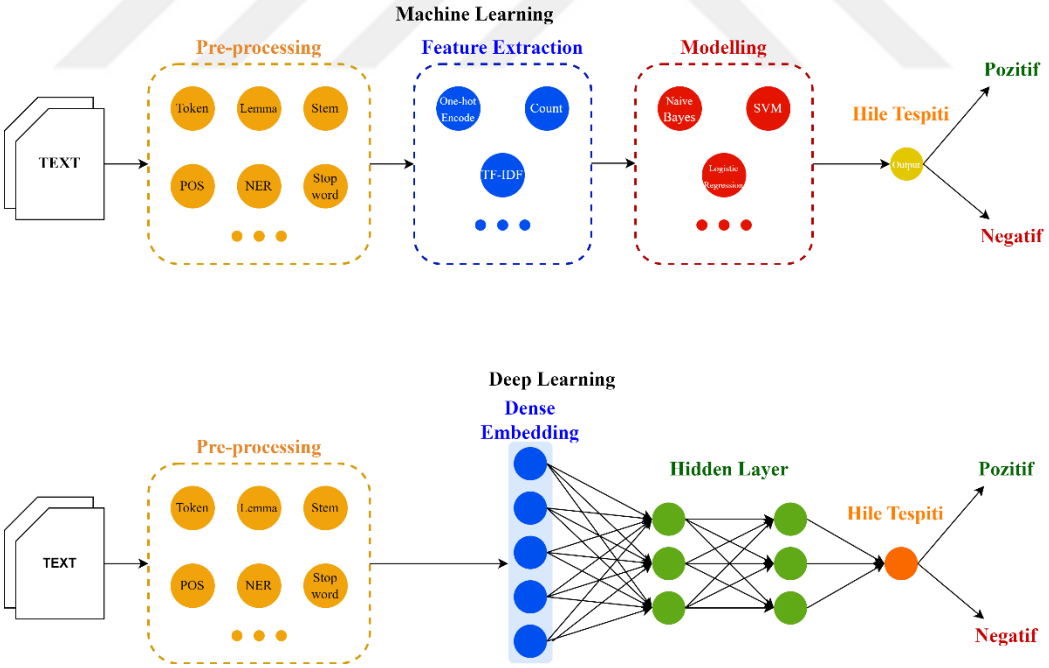
Kaynak: (Khan et al., 2021)

Bir yeni veri noktasının sınıflandırılması veya tahmin edilmesi gerektiğinde, tüm karar ağaçlarının tahminleri toplanır veya bir ortalaması alınır (sınıflandırma için). Sınıflandırma durumunda, sınıf tahminleri çoğunluğa dayalı olarak belirlenir. Regresyon durumunda ise tahminlerin ortalaması alınır. RF, birden fazla ağacın bir araya getirilmesiyle oluşturulduğu için, genellikle daha kararlı ve aşırı uydurmaya karşı daha dirençlidir. Ayrıca, ağaçların çeşitliliği sayesinde daha iyi genelleme yapabilir. Modelin performansı, ağaç sayısı ve derinliği gibi hiperparametrelerin ayarlanmasıyla iyileştirilebilir. RF, sınıflandırma, regresyon, özellik, sıralama ve veri madenciliği gibi birçok uygulama alanında kullanılır.

BEŞİNCİ BÖLÜM

DERİN ÖĞRENME

Bu tez, finansal suçların önceden engellenmesi ve KPA önleme amacıyla DL tekniklerinin nasıl kullanılabileceğini araştırmaktadır. Şekil 5.1'de ML ile arasındaki farkın gösterildiği DL, AI alanındaki hızlı gelişmelerle birlikte büyük veri analizi ve desen tanıma konularında önemli başarılar elde etmiş bir alt dal olarak öne çıkmaktadır (LeCun et al., 2015). KPA genellikle finansal kurumları hedef alan bir suç türüdür ve bu suçun karmaşıklığı, teknolojinin ilerlemesi ve finansal piyasaların genişlemesiyle birlikte daha da artmıştır. Dünya genelinde milyarlarca dolarlık kaynakların yasa dışı yollarla temizlenmesi ve terörizmin finansmanının engellenmesi gibi nedenlerle AML önemi giderek artmaktadır (Bakınız İKİNCİ BÖLÜM). Geleneksel yöntemlerin, bu suçun karmaşıklığı ve sürekli evrimi karşısında yetersiz kaldığı gerçeğiyle karşı karşıyayız. Bu bağlamda, yeni ve etkili çözüm yöntemlerine duyulan ihtiyaç önemli bir hal almaktadır.



Şekil 5.1: ML ile DL arasındaki fark

Kaynak: (Dang et al., 2020)

Geleneksel finansal suç tespit yöntemleri genellikle otomatize edilmemiş işlemler ve kural tabanlı yaklaşımlarla sınırlıdır, bu da hızla değişen suç modellerine karşı etkisiz olabilir. AML için daha gelişmiş ve otomatize bir yaklaşım gerekmektedir. İşte bu noktada, DL teknolojisi devreye girebilir. DL modelleri, büyük veri kümelemelerini analiz ederek potansiyel suç desenlerini tespit edebilir ve geleneksel yöntemlere kıyasla daha hızlı ve hassas sonuçlar sağlayabilir.

5.1. Yapay Sinir Ağları

Yapay sinir ağları (ANN), günümüz AI alanının temel taşlarından biri olarak kabul edilir. ANN, biyolojik sinir sistemlerinden ilham alarak oluşturulan matematiksel modellerdir. Bu modeller, yapay nöronlar adı verilen işlem birimlerinden oluşur ve bu nöronlar birbiriyle etkileşime girerek bilgi işleme görevlerini gerçekleştirirler. Temel olarak, ANN, girdi verilerini alır, bu verileri işler ve belirli bir çıktıyı üretirler (Mehrotra et al., 1997). İnsan beyninin bilgi işleme mekanizmasından esinlenerek geliştirilen bu teknoloji, karmaşık problemleri çözme, öğrenme ve desen tanıma gibi birçok alanda kullanılmaktadır. ANN fikri, insan beyninin işleyişini anlamaya ve onu taklit etmeye yönelik uzun bir tarihe sahiptir (Zou et al., 2008). İlk düşünceler, 1940'ların sonlarına dayanmaktadır (Bielecki, 2019).

- **1943: McCulloch-Pitts Modeli:** Warren McCulloch ve Walter Pitts, sinir hücrelerinin basit mantıksal işlevlerle tanımlanabileceğini gösterdiler. Bu model, ANN temelini atmıştır (McCulloch & Pitts, 1943).
- **1950'ler: Perceptron Modeli:** Frank Rosenblatt'ın geliştirdiği Perceptron modeli, basit bir ANN modelidir. İkili sınıflandırma problemlerini çözmek için kullanıldı (Block, 1962).
- **1960'lar-1980'ler: Durgun Dönem:** ANN bu dönemde sınırlı işlem gücü ve veri eksikliği gibi nedenlerle ilgi görmemekteydi (Haglin et al., 2019).
- **1986: Geri Yayılım Algoritması:** David Rumelhart, Geoffrey Hinton ve Ronald Williams tarafından tanıtılan geri yayılım algoritması, sinir ağlarının yeniden canlanmasına neden oldu. Bu algoritma, sinir ağlarının öğrenme yeteneklerini önemli ölçüde artırdı (Rumelhart et al., 1986).
- **1990'lar: Çok Katmanlı Ağlar ve Uygulamaları:** Bu dönemde çok katmanlı yapay (derin) sinir ağları (DNN) geliştirildi ve konuşma tanıma, görüntü işleme

ve doğal dil işleme gibi uygulamalarda kullanılmaya başlandı (S. Ding et al., 2013).

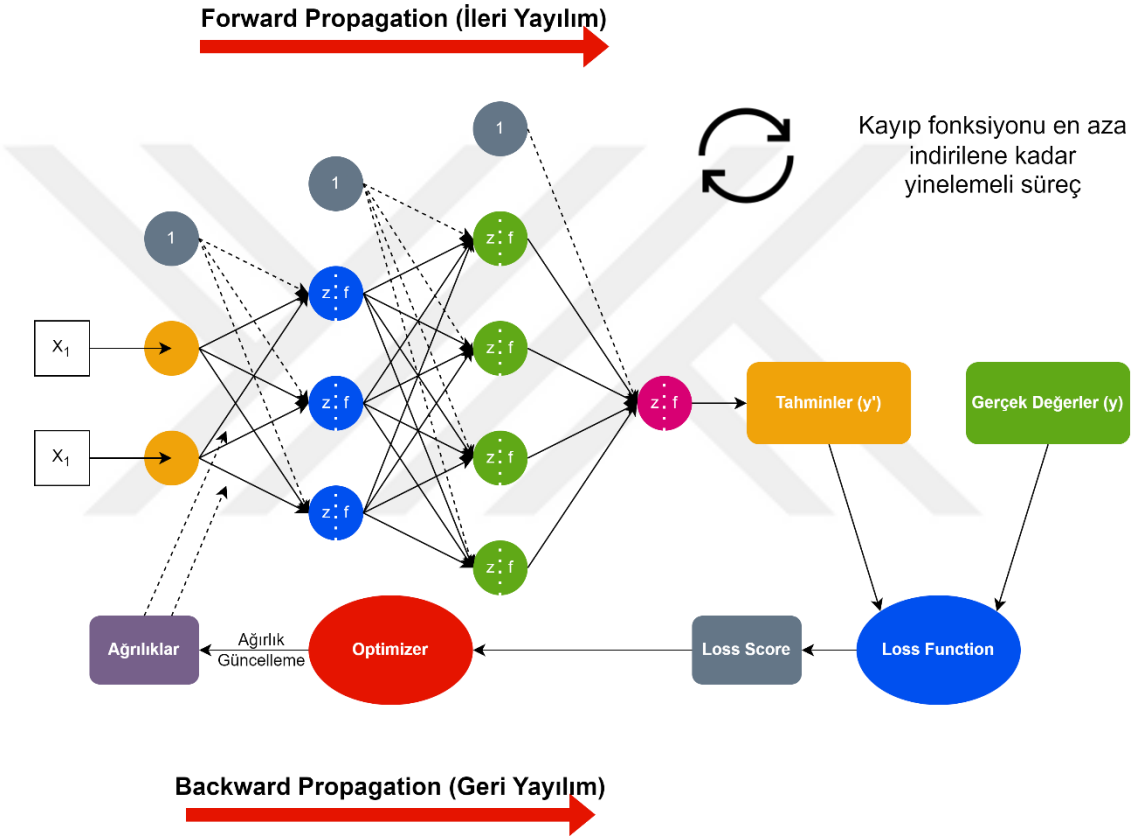
- **2000'ler ve Sonrası: Büyük Veri ve Derin Öğrenme:** ANN, büyük veri setlerinin kullanılabilir hale gelmesi ve gelişmiş hesaplama kaynakları sayesinde büyük bir ivme kazandı. DL yöntemleri, karmaşık problemleri çözmek için başarıyla kullanılmaya başlandı (H. Zhu, 2020).

ANN ana bileşenleri şunlardır:

- **Yapay Nöronlar:** ANN temel işlem birimleri olan yapay nöronlar, girdi verilerini alır ve bir aktivasyon fonksiyonu kullanarak bu girdileri işlerler. Sonuç olarak bir çıktı üretirler.
- **Katmanlar:** Yapay nöronlar, katmanlar halinde düzenlenir. Giriş katmanı, girdileri alır ve ardışık katmanlar, girdileri giderek daha soyut ve karmaşık temsilcilere dönüştürmek için kullanılır.
- **Ağırlıklar ve Bağlantılar:** Her nöronun girdi bağlantıları üzerinde bir dizi ağırlık bulunur. Bu ağırlıklar, girdi verileriyle çarpılarak nöronun çıktısını hesaplamak için kullanılır.
- **Öğrenme Algoritmaları:** ANN, eğitim verilerini kullanarak ağırlıklarını ayarlarlar. Geri yayılım (backpropagation) gibi algoritmalar, hata düzeltmeyi ve ağırlıkların güncellenmesini yönetir.
- **Aktivasyon Fonksiyonları:** Nöronların çıktılarını belirler. Sigmoid, ReLU ve tanh gibi fonksiyonlar bu bağlamda sıkça tercih edilenler arasındadır.

ANN, gelişim süreci ve temel tanımı itibariyle, bilgi işleme ve öğrenme kapasitesiyle büyük bir öneme sahip bir AI teknolojisidir. ANN, büyük veri analizi, görüntü işleme, doğal dil işleme, konuşma tanıma, otonom araçlar ve birçok farklı uygulama alanında başarıyla kullanılmaktadır (Kumar & Thakur, 2012). Bu teknoloji, özellikle son yıllarda DL alanında elde edilen büyük başarılarla ön plana çıkmıştır. ANN gelişimi, bilgisayar teknolojisinin ilerlemesiyle paralel olarak büyük bir ivme kazanmıştır. Özellikle büyük veri setlerinin kullanılabilir hale gelmesi, ANN ile daha karmaşık ve büyük ölçekli problemleri çözme yeteneğini artırmıştır. DL, çok katmanlı ANN gelişimini temsil eder ve karmaşık desenleri tanıma, veri madenciliği, otomatik çeviri ve benzeri uygulamalarda büyük bir rol oynamıştır.

Ancak bazı ANN sınırlamaları da vardır. Özellikle büyük modeller, eğitim süreçleri ve yeterli veri gereksinimleri gibi zorluklarla karşılaşabilirler. Ayrıca, ağ yapılandırması, aktivasyon fonksiyonu seçimi ve hiper parametrelerin ayarlanması gibi konular da önemlidir. Sonuç olarak, mimarisi Şekil 5.2'de gösterilen ANN bilgisayar biliminde önemli bir rol oynar ve gelecekte daha da geliştirilerek birçok alanda daha etkili bir şekilde kullanılmaya devam edecektir. Bu teknoloji, insan beyninin işleme mekanizmasından ilham alarak karmaşık problemleri çözme kapasitesine sahip bir AI modelini temsil eder.



Şekil 5.2: ANN mimarisi

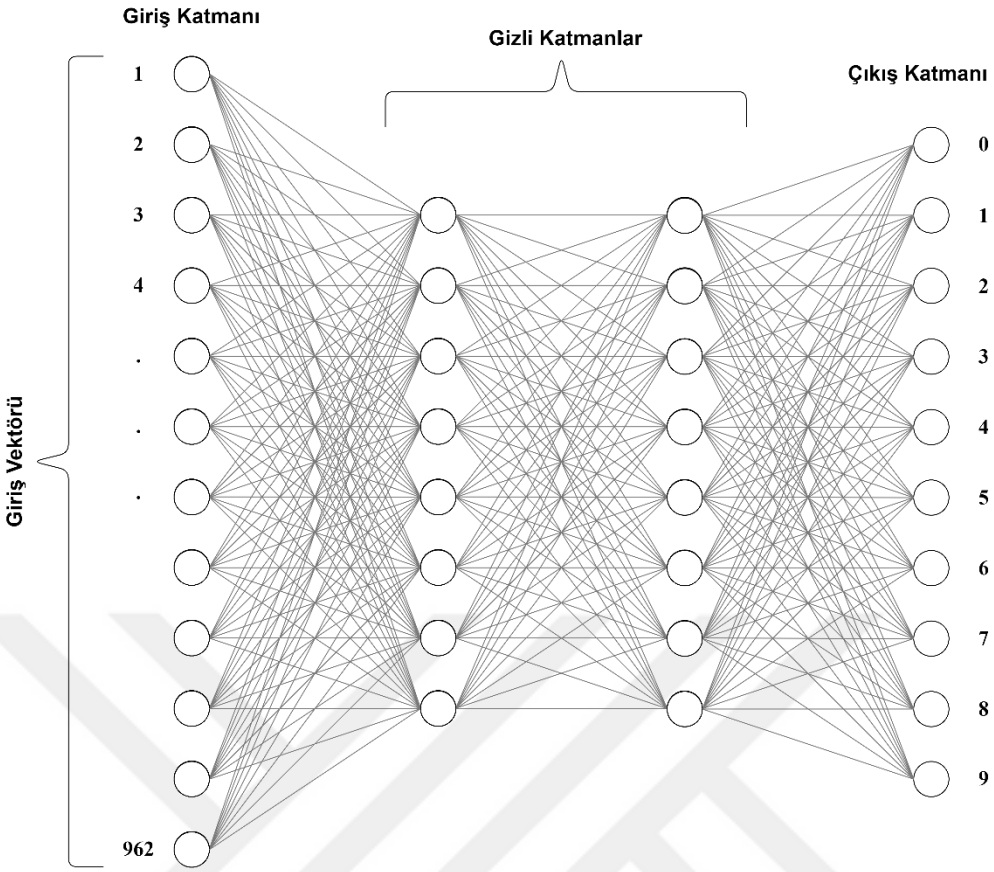
5.2. Derin Sinir Ağları

DNN, sinir ağlarının evrimleşmiş ve karmaşıklaşmış bir türüdür ve özellikle karmaşık veri analizi, öğrenme ve desen tanıma görevlerinde kullanılır. DNN, birçok katmandan oluşan çok katmanlı bir ANN yapısına sahiptir ve büyük veri setlerini işlemek için tasarlanmıştır. DNN, AI ve ML alanında büyük bir çığır açmış ve birçok uygulama alanında devrim yaratmıştır. DNN, biyolojik sinir sistemlerinden ilham alınarak geliştirilmiş matematiksel modellemelerdir. İnsan beyni, karmaşık bilgi işleme görevlerini gerçekleştirirken birçok nöronun birbirleriyle etkileşimde bulunmasına

dayanır. DNN de benzer bir mantığı takip eder. Temel DNN yapı taşları, yapay nöronlardır. Bu yapay nöronlar, girdi verilerini alır, bu girdileri işler ve çıktı üretir. Her yapay nöron, bir dizi ağırlık ve eşik değeri ile karakterize edilir. Girdi verileri bu ağırlıklarla çarpılarak toplanır ve ardından bir aktivasyon fonksiyonu kullanılarak işlenir. Bu işlemin sonucu, nöronun çıktısını verir. DNN, çok katmanlı bir yapıya sahiptir. Her katmanda bir veya daha fazla yapay nöron bulunur. Geleneksel ANN aksine, DNN birçok gizli katmana sahiptir. Bu katmanlar, girdi verilerini giderek daha karmaşık ve soyut temsilcilerine dönüştürme görevini üstlenirler. Bu çok katmanlı yapısı, DL yeteneklerini sağlar.

DNN eğitim süreci, ağın ağırlık değerlerinin belirlenmesi için gerçekleşir. Bu ağırlıklar, girdi verilerini istenilen çıktılara dönüştürmek için ayarlanır. Eğitim verileri, ağın performansını iyileştirmek için kullanılır. Geri yayılım (backpropagation) adı verilen bir teknik, ağın hatalarını geriye doğru ileterek ağırlıkların güncellenmesini sağlar. Her yapay nöronun aktivasyon fonksiyonu, nöronun çıktısını belirler. Bu fonksiyonlar, nöronların non-lineer davranışlarını modellemek için kullanılır. Sigmoid, ReLU ve tanh gibi yaygın aktivasyon fonksiyonları bulunur.

Şekil 5.3'de mimarisi de sunulan DNN, veriler arasındaki karmaşık ilişkileri öğrenme kapasitesine sahiptir. Bu nedenle, derin öğrenme ve temsil öğrenme görevlerinde büyük bir başarı elde ederler. DNN, otomatik olarak özelliklerin çıkarılmasında etkili bir şekilde kullanılır ve bu, büyük ölçüde veri analizi ve desen tanıma alanlarında fayda sağlar. Başarılarına rağmen DNN için bazı sınırlamalar ve zorluklar vardır. Büyük veri setleri ve karmaşık modeller gerektirirler. Ayrıca, eğitim süreçleri zaman alabilir ve donanım gereksinimleri yüksek olabilir. DNN, AI alanındaki en önemli gelişmelerden biri olarak kabul edilir. Karmaşık veri analizi, öğrenme ve temsil öğrenme görevlerinde büyük bir potansiyele sahiptirler. Bu nedenle, birçok uygulama alanında büyük bir etki yaratmış ve gelecekte daha da önemli bir rol oynayacakları düşünülmektedir.



Şekil 5.3: Derin Sinir Ağları mimarisi

5.3. Derin Öğrenme

DL, AI ve ML alanlarında büyük bir çığır açmış ve birçok uygulama alanında büyük başarılar elde etmiştir. DL, ANN üzerine kurulu karmaşık ve çok katmanlı bir ML alt dalıdır. Temel olarak, bu yöntem, büyük veri setleri üzerinde öğrenme yapma yeteneği ve veriler arasındaki karmaşık ilişkileri çıkarma kapasitesi ile tanınır. DL, bilgisayarlar aracılığıyla desen tanıma, sınıflandırma, çeviri, doğal dil işleme ve benzeri karmaşık görevleri gerçekleştirebilme yeteneği sunar. DL, ismini sinir ağlarının çok katmanlı yapısından alır. Bu yöntem, verileri temsil eden katmanlar (genellikle gizli katmanlar olarak adlandırılır) ve bu katmanları birbirine bağlayan ağırlıklardan oluşur. DL, bu ağırlıkları eğitim verileri üzerinden otomatik olarak ayarlayarak öğrenme sürecini gerçekleştirir (Dablain et al., 2022). Bu nedenle, bu yöntem özellikle büyük veri analizi için uygundur. DL, birçok uygulama alanında büyük bir etki yaratmıştır. Bu alanlardan bazıları şunlardır:

- **Görüntü İşleme:** DL, görüntü tanıma, nesne tespiti ve yüz tanıma gibi görsel işleme görevlerinde kullanılır.

- **Doğal Dil İşleme:** Dil modelleri ve otomatik metin çevirisi gibi doğal dil işleme görevlerinde büyük başarılar elde edilir.
- **Konuşma Tanıma:** Ses tanıma ve konuşma tanıma uygulamalarında kullanılır.
- **Otonom Araçlar:** Sürücüsüz araçlar, DL kullanarak çevrelerini algılayabilirler.
- **Sağlık Alanı:** Tıbbi görüntülerin analizi ve hastalık teşhisi gibi sağlık alanında birçok uygulama bulunur.

DL, büyük başarılar elde etmiş olsa da bazı zorluklarla karşılaşabilir. Büyük veri setleri, eğitim süreçleri ve donanım gereksinimleri gibi faktörler, bu teknolojinin sınırlamalarını oluşturabilir. Ayrıca, ağ yapısı, aktivasyon fonksiyonu seçimi ve hiper parametrelerin ayarlanması gibi konular da önemlidir. DL, AI ve ML alanlarında önemli bir role sahip bir teknolojidir. Karmaşık problemleri çözme, büyük veri analizi ve otomatik desen tanıma gibi görevlerde büyük bir potansiyele sahiptir. Bu nedenle, DL gelecekte daha da geliştirilerek birçok alanda daha etkili bir şekilde kullanılacaktır.

5.4. Derin Öğrenme Teknikleri

Bu bölümde, DL tekniklerinin temel prensiplerini ve bu tekniklerin çalışma mantığını ayrıntılı bir şekilde açıklayacak, nöron ağlarının ve DL modellerinin temel bileşenleri incelenecektir. Ayrıca, bu bölümde DL alanında önemli rol oynayan özel modeller, özellikle otokodlayıcı türleri ve CNN gibi öncü konular üzerinde durulacaktır.

5.4.1. Autoencoder

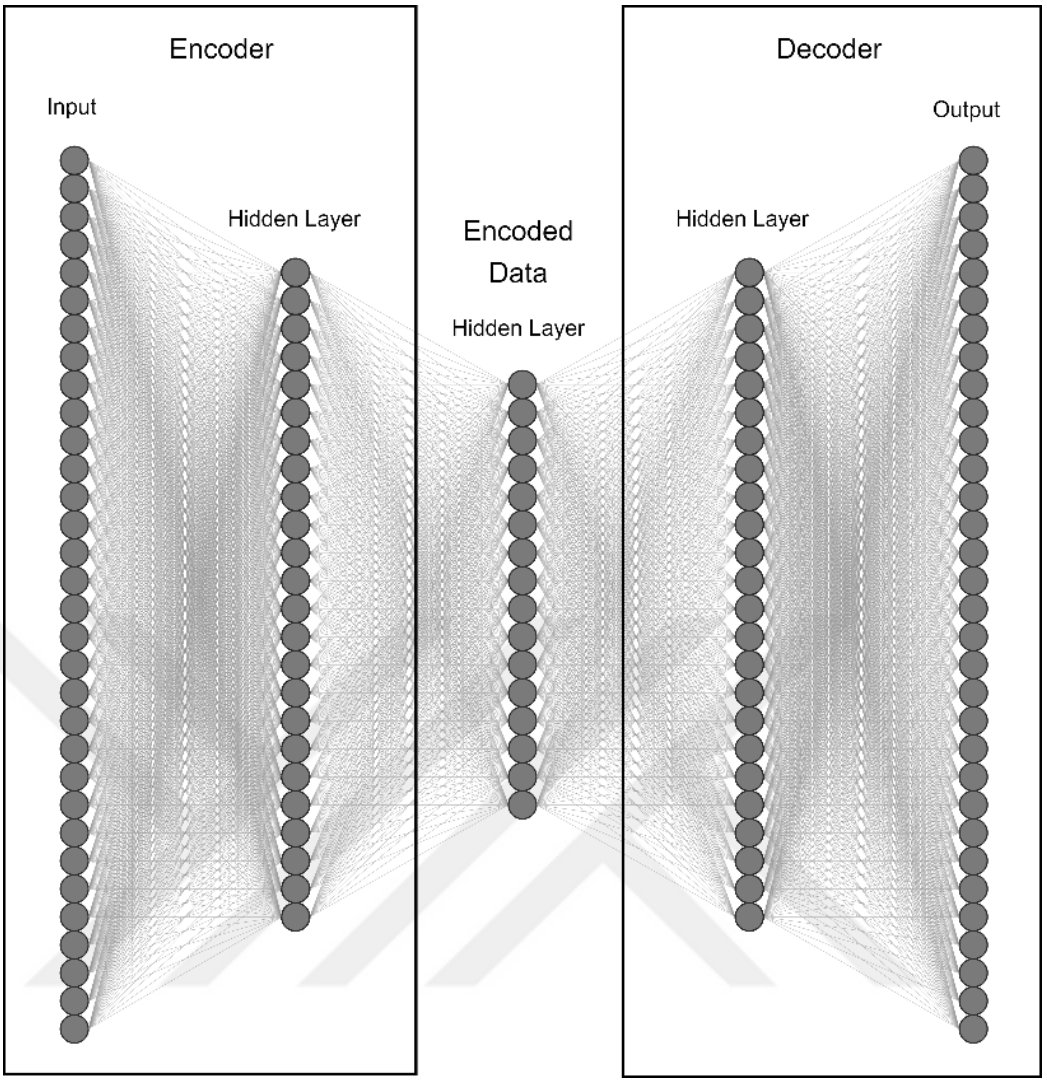
AE, gözetimsiz öğrenme ve boyut azaltma görevlerinde etkili olan sinir ağı modelleridir (Çakır & Şirin, 2023). Bu yapılar, bir encoder ağı aracılığıyla giriş verilerini daha düşük boyutlu bir gizli uzaya eşleyen ve ardından bir decoder ağı aracılığıyla bu gizli temsil ile orijinal girişi yeniden oluşturan bir yapıya sahiptir. Giriş verilerini etkili bir şekilde kodlamayı ve çözmeyi öğrenen AE, verilerdeki belirgin özellikleri ve desenleri yakalamayı hedeflerler (Hinton & Salakhutdinov, 2006). Örnek bir AE modeli Şekil 5.4'de gösterildiği gibidir.

Autoencoder'lar, özellikle anormallik tespiti görevlerinde umut vaat edici bir rol oynamaktadır. Bu modeller, normal örnekler üzerinde eğitildiklerinde, bu örnekleri doğru bir şekilde yeniden oluşturmayı öğrenirler. Ancak, dolandırıcılık gibi

anormallikler, öğrenilmiş desenlerden sapar ve bu durum daha yüksek yeniden oluşturma hatalarına yol açar. Bu AE özelliği, dolandırıcılığı tespit etmek için uygun hale getirir. Yani, sistem normalde beklenen desenlerden sapmaları belirleyerek ve yüksek yeniden oluşturma hataları üzerinden anormallikleri vurgulayarak dolandırıcılığı tespit edebilir. Bu özellikleri ile AE dolandırıcılık tespiti alanında etkili ve umut vadeden bir araç haline gelmiştir.

$$\mathcal{L}(x,\, x') \;=\; ||\, \tilde{x} \;-\; x' \,||^2 \qquad (5\text{-}1)$$

Denklem (5-1)'de, $\tilde{x}$, giriş verisinin AE tarafından üretilen tahmini veya yeniden oluşturulan çıktısını temsil ederken x', gerçek ve orijinal giriş verisinin yeniden oluşturulmuş çıktısını ifade eder. AE eğitme sürecinde temel amaç $\tilde{x}$ ile x' arasındaki benzerliği maksimize etmek ve bu sayede mümkün olduğunca gerçekçi bir yeniden oluşturma yapabilmektir. Bu eğitim, Denklem (5-1)'de gösterildiği gibi genellikle bir kayıp fonksiyonu olan $\mathcal{L}(x,\, x')$ aracılığıyla gerçekleştirilir. Bu kayıp fonksiyonu, tahmini $\tilde{x}$ ile gerçek x' arasındaki farkı ölçerek, AE modelinin eğitim sırasında bu farkı en aza indirgeyerek giriş verisinin anlamlı ve sıkıştırılmış bir temsilini öğrenmesini hedefler.



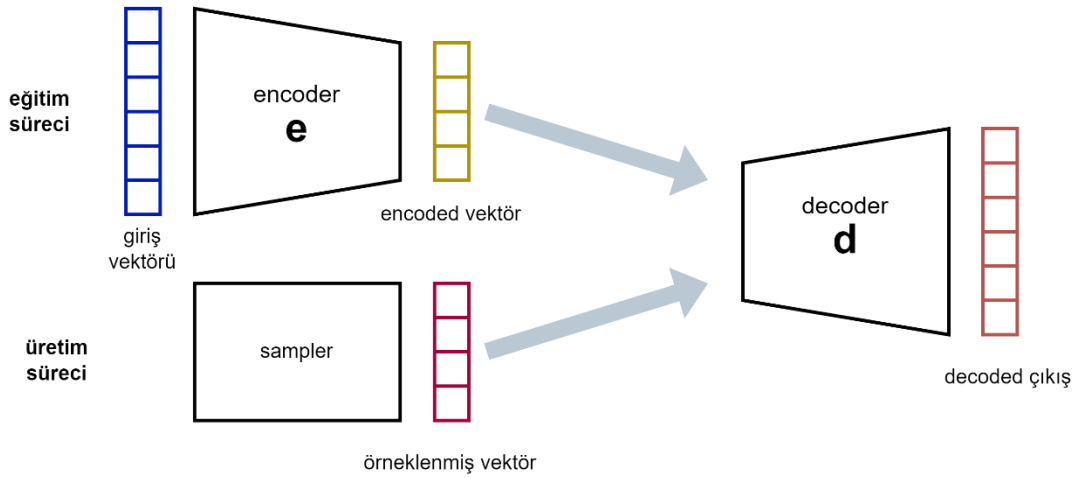
Şekil 5.4: Otokodlayıcılar, genellikle boyut azaltma için kullanılan veriyi temsil eden sembolik bir vektör öğrenmeye tasarlanmıştır. Genel mimarileri, kodlayıcı (encoder) ve çözücü (decoder) modüllerini içerir.

Kaynak: (Çakır & Şirin, 2023)

5.4.2. Variational Autoencoder

VAE, otokodlayıcı ailesine ait bir model olup, veri dağılımını olasılıksal bir yaklaşımla modeller. Bu model, geleneksel otokodlayıcıların aksine, giriş verisini gizli uzaya doğrudan eşleme yerine, gizli uzayın üzerinde bir olasılık dağılımını öğrenir. Bu dağılım, veri noktalarının gizli uzay içindeki konumlarına dair bir belirsizlik sağlar. Şekil 5.5'de sunulan VAE'nin özelliği, bu olasılık dağılımı aracılığıyla yeni örnekler oluşturabilme yeteneğidir, bu da modelin daha önce görmediği ancak öğrendiği veri dağılımına uygun yeni verileri üretebilme yeteneğini ifade eder. (Kingma & Welling,

2013) tarafından tanıtılan VAE, özellikle veriye dayalı olasılıksal modelleme ve örüntü oluşturma konularında başarılı bir şekilde kullanılmaktadır.



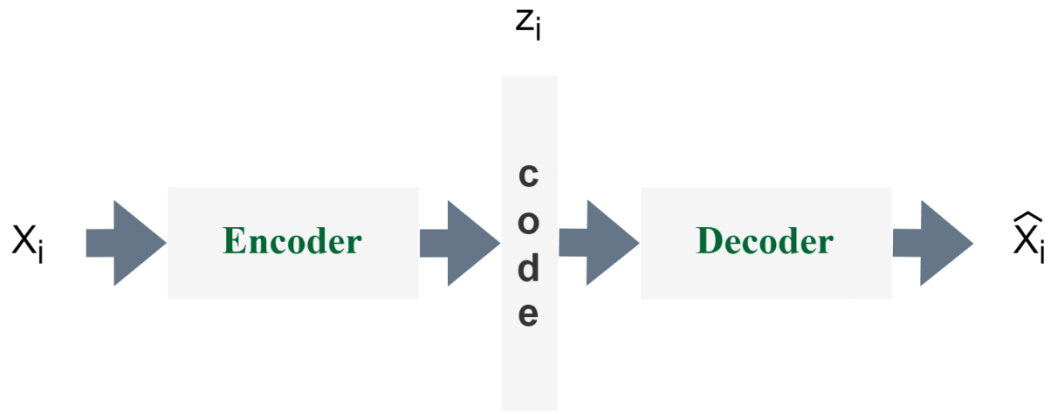
Şekil 5.5: Variational Autoencoder

$$\mathcal{L}(\theta, \phi) = E_{q_\phi}[log\, p_\theta(x \mid z)\ ] - D_{KL}[q_\phi(z \mid x)\, ||p(z)] \qquad (5\text{-}2)$$

Denklem (5-2)'de gösterildiği üzere $q_\phi(z \mid x)$ ve $p(z)$ Gauss dağılımlarıdır ve $\mathcal{L}$, kayıp fonksiyonunun sonucudur. İlk terim, $E_{q_\phi}[\log p_\theta\,(x \mid z)]$ olan değişken alt sınırdır ve ikinci terim, $D_{KL}[q_\phi(z \mid x)\, ||p(z)]$ olan Kullback-Leibler sapmasını temsil eder. Bu kayıp fonksiyonunu kullanarak VAE, giriş verisinin belirli özelliklerine odaklanmayı amaçlar ve bu özellikleri öncelikle korumayı hedefler.

5.4.3. Contractive Autoencoder

Şekil 5.6'da sunulan CAE, (Rifai et al., 2011) tarafından tanıtılan bir otokodlayıcı türüdür ve öğrenilen temsillerin giriş uzayındaki küçük değişikliklere karşı dayanıklılığını artırmak amacıyla geliştirilmiş bir düzenleme terimi içermektedir. Bu düzenleme terimi, öğrenilen temsillerin giriş verilerindeki küçük değişikliklere karşı hassas olmamasını sağlayarak, otokodlayıcının daha genel ve anlamlı temsiller öğrenmesini teşvik eder. CAE modelinin dayanıklılığı, özellikle sentetik örnekler oluşturulması gibi uygulamalarda, daha anlamlı ve tutarlı sonuçların elde edilmesine katkı sağlayabilir. Bu özellikleriyle CAE, özellik çıkarma ve örüntü oluşturma görevlerinde etkili bir şekilde kullanılabilir.



Şekil 5.6: Contractive Autoencoder

$$\mathcal{L}(x, \hat{x}) = || x - \hat{x} ||_2^2 + \lambda || J_E(x) ||_F^2 \quad (5\text{-}3)$$

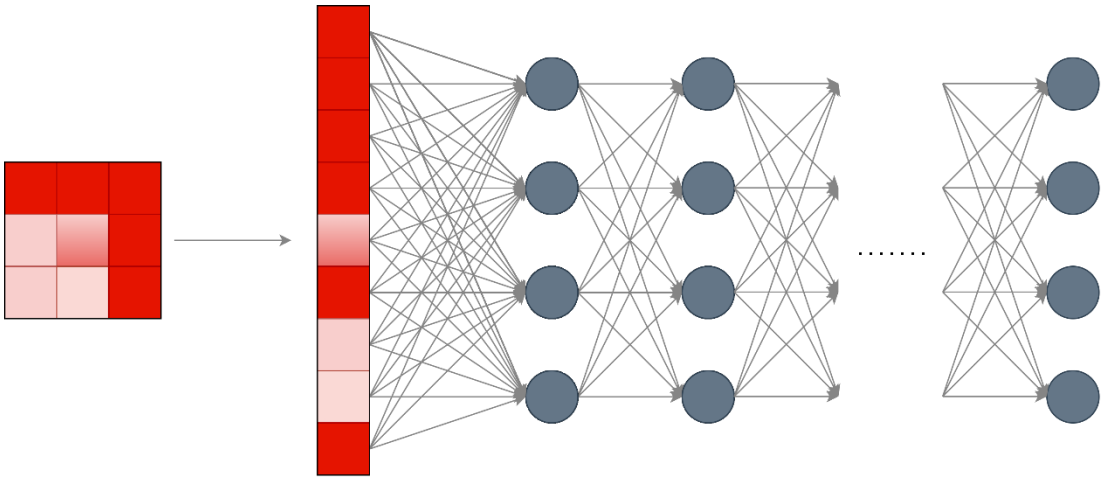
Denklem (5-3)'de gösterildiği üzere kayıp fonksiyonu $\mathcal{L}(x, \hat{x})$, farklı bileşenlere ayrılabilir. İlk olarak, giriş verisi x ile yeniden oluşturulmuş çıktı $\hat{x}$ arasındaki karekök Euclidean uzaklığını hesaplar, $|| x - \hat{x} ||_2^2$ ile gösterilir. Bu, otokodlayıcının orijinal girişin kopyasının doğruluğunu nicelendirir. İkinci olarak, düzenlemeyi kontrol eden hiperparametre λ tarafından yönetilen bir düzenleme terimi ekler. Bu terim, Jacobian matrisinin Frobenius normunu içerir, $|| J_E(x) ||_F^2$, burada $J_E(x)$, encoder'ın çıkışının giriş x 'e göre Jacobian matrisini temsil eder. Düzenleme, öğrenilen temsillerin küçük giriş değişikliklerine karşı dayanıklılığını teşvik etmeyi amaçlar ve bu formül, CAE modeli eğitirken merkezi bir rol oynar.

CAE, büyük bir metin veri kümesi üzerinde eğitildiğinde, encoder bölümü giriş metin verisinin en önemli özelliklerini yakalayan sıkıştırılmış bir temsil öğrenir. Aynı anda, decoder bölümü bu sıkıştırılmış temsilden orijinal metin verisini başarılı bir şekilde yeniden oluşturmayı öğrenir. Elde edilen bu sıkıştırılmış temsil, metin verilerini etkili bir şekilde sıkıştırmak, metin benzerlik aramak veya metin verilerinin genel bir temsilini oluşturmak gibi çeşitli doğal dil işleme görevlerinde kullanılabilir. CAE, özellikle büyük metin veri kümeleri üzerinde DL yöntemlerini kullanarak veri temsilini geliştirmek isteyen araştırmacılar ve uygulamacılar için güçlü bir araç olabilir.

5.4.4. Convolutional Neural Network

Şekil 5.7'de minimal bir gösterimi sunulan CNN, özellikle görüntü işleme ve tanıma gibi görsel veri analizi görevlerinde etkili olan bir DL türüdür. CNN temel ilkeleri, bir dizi katmandan oluşur (Z. Li et al., 2022).

- **Girdi Katmanı (Input Layer):** Bir CNN başlangıç noktasıdır.
- **Evrişim Katmanları (Convolutional Layers):** Temel CNN özelliği, evrişim katmanlarının kullanılmasıdır. Evrişim işlemi, girdi katmanının farklı özelliklerini öğrenmek için filtrelerin (kernels) kullanılması anlamına gelir. Bu filtreler, girdi verileri üzerinde gezinirken birer ölçü (çerçeve) uygular ve bu çerçevedeki özellikleri çıkarır. Her evrişim katmanı, birden fazla filtre içerebilir.
- **Aktivasyon Fonksiyonları:** Evrişim katmanlarının ardından, elde edilen sonuçlar üzerine bir aktivasyon fonksiyonu (genellikle ReLU) uygulanır. Bu, ağırlık matrisleriyle lineer işlemlerin ardından daha karmaşık ve öğrenilebilir özelliklerin elde edilmesine yardımcı olur.
- **Ara Katmanlar (Intermediate Layers):** Evrişim katmanları arasında, daha derin özelliklerin öğrenildiği ve daha fazla öğrenme için daha fazla aktivasyon işlemi uygulandığı ara katmanlar bulunabilir.
- **Ara Katmanların Havuzlama Katmanları (Pooling Layers):** Ara katmanların ardından genellikle havuzlama katmanları gelir. Havuzlama işlemi, özellik haritalarını küçültmek ve hesaplama yükünü azaltmak için kullanılır. Bu işlem, her bir özellik haritasındaki en büyük özellikleri seçerek veya ortalama değerleri hesaplayarak gerçekleştirilir.
- **Tam Bağlantılı Katmanlar (Fully Connected Layers):** CNN sonunda, bir veya daha fazla tam bağlantılı katman bulunur. Bu katmanlar, öğrenilen CNN özelliklerini kullanarak sınıflandırma veya regresyon görevlerini gerçekleştirir. Genellikle softmax aktivasyonu kullanılarak çok sınıflı sınıflandırma işlemleri gerçekleştirilir.



Şekil 5.7: Convolutional Neural Network

Eğitim sürecinde CNN, öğrenmeye rastgele başlar, ancak geriye doğru yayılan (backpropagation) hata azaltma süreciyle ağırlıklarını günceller. Bu, ağın veriye daha iyi uyan ağırlıkları öğrenmesini sağlar. Büyük veri seti üzerinde eğitilen derin bir CNN, genellikle özellik çıkarımında oldukça başarılıdır ve özellikle görüntü sınıflandırma, nesne tanıma ve yüz tanıma gibi görevlerde kullanılır.

- **Otomatik özellik öğrenme:** CNN, belirli özellikleri el ile tanımlamak yerine, verilerden özellikleri otomatik olarak öğrenir.
- **Maksimum invariyans:** Evrişim işlemi, bir nesnenin farklı konumlarında veya açılarda nasıl göründüğünü öğrenir, bu da nesne tanıma ve sınıflandırma için avantaj sağlar.
- **İşlem hızı:** Havuzlama katmanları sayesinde, özellik haritalarının boyutu küçültülerek işlem hızı artırılır.

ALTINCI BÖLÜM

ÖNERİLEN ÇALIŞMA

Bu bölümde, DL tekniklerinin KPA ile önlenmesinde nasıl kullanılabileceğini detaylı bir şekilde inceleyecek ve bu teknolojinin finansal güvenlik açısından taşıdığı potansiyel vurgulanacaktır. Tezin bu bölümü, DL yaklaşımının finansal suç tespiti ve KPA önleme süreçlerine nasıl bütünleştirilebileceğini, mevcut yöntemlere göre avantajlarını ve potansiyel katkılarını öne çıkaracaktır. Sonuç olarak, bu tez, finansal suçlarla mücadelede DL teknolojisinin önemini vurgulayarak KPA önlenmesinde yeni bir perspektif sunacaktır.

6.1. Motivasyon

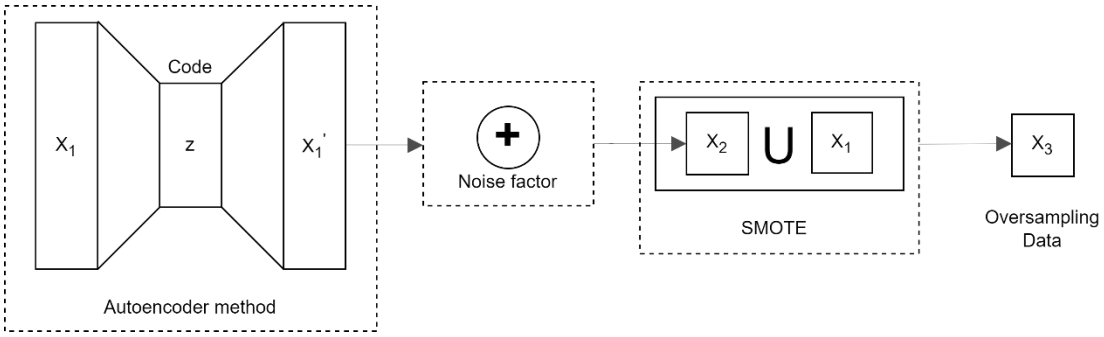
SMOTE, CIP çözümünde etkili bir araç olmasına rağmen, hileli işlemlere özgü karmaşık özellikleri ve detayları tam olarak yakalayamayabilir. Bu nedenle, dolandırıcılık tespiti performansını artırmak amacıyla otokodlayıcı tabanlı yeni bir yöntem öneriyoruz, bu yöntemde NFE (Noise Factor Encoding) ve SMOTE'un birleşimi bulunmaktadır. Önerdiğimiz yaklaşımın temelinde yatan motivasyon, otokodlayıcının normal verinin temel desenlerini etkili bir şekilde öğrenebileceği ve anormallikleri tespit edebileceği gözlemine dayanmaktadır. Otokodlayıcıları gerçek dolandırıcılık verileri üzerinde eğitirken kodlama süreci sırasında bir gürültü faktörü ekleyerek, hileli işlemlerin temsilini geliştirmeyi amaçlıyoruz. Bu gürültü faktörü, otokodlayıcıyı daha fazla ayırt edici özellik öğrenmeye ve dolandırıcılığın benzersiz özelliklerini yakalamaya zorlayarak, hileli işlemleri daha etkili bir şekilde tanımlamayı hedefliyoruz.

Otokodlayıcı tabanlı NFE tekniğinin SMOTE ile birleşimi, sınıf dengesizliğini ele almanın ötesinde, dolandırıcılık örneklerinin artırılmış temsilleri ile veri kümesini zenginleştirmemize olanak tanır. Bu yaklaşım, otokodlayıcının benzersiz yeteneklerini anormallik tespiti için kullanmaktadır ve aynı zamanda CIP çözümü adına SMOTE'un avantajlarından faydalanmaktadır. Otokodlayıcı tabanlı NFE, gerçek dolandırıcılık verilerini öğrenirken eklenen gürültü faktörleri ile özellikleri daha etkili bir şekilde vurgular, bu da dolandırıcılık örneklerini daha hassas bir şekilde tanımlamamıza yardımcı olur. Bu bütünleşik yaklaşım, modelin güvenilirliğini ve doğruluğunu artırmayı amaçlamaktadır. Yapılan deneyler ve değerlendirme ölçütleri aracılığıyla,

önerdiğimiz yöntemin dolandırıcılık tespiti performansını artırma konusundaki etkinliğini detaylı bir şekilde inceleyeceğiz. Bu bağlamda, otokodlayıcı tabanlı NFE ve SMOTE'un birleşimi ile elde edilen sonuçlar, dolandırıcılık tespiti modellerinin daha doğru ve güvenilir olmasını sağlayarak, bu alandaki literatüre katkıda bulunmayı hedeflemekteyiz. İlerleyen bölümler, yöntemimizi, deneysel kurulumumuzu ve önerdiğimiz yaklaşımımızın etkinliğini değerlendirmek için kullanılan değerlendirme ölçütlerini açıklayacaktır.

6.2. Yöntemsel Temeller

Bu bölümde, CIP varlığında dolandırıcılık tespit performansını artırmak için AE tabanlı NFE ile SMOTE'u birleştiren önerdiğimiz yaklaşımımızı sunuyoruz. Yaklaşımımızın temel adımlarını açıklıyor ve her adım için bir açıklama sunuyoruz. Hileli işlemlerin temsilini geliştirmek için otokodlayıcı tabanlı NFE tekniğini kullanıyoruz. Gerçek dolandırıcılık verileri üzerinde bir otokodlayıcı modeli eğitirken kodlama işlemi sırasında bir gürültü faktörü ekliyoruz. Bu gürültü faktörü, otokodlayıcının daha ayırt edici özellikleri öğrenmesini ve dolandırıcılığın benzersiz özelliklerini yakalamasını teşvik eden pertürbasyonlar (yaklaşık bir çözüm elde etmek için metotlar) olarak hizmet eder. Gürültü faktörünü içererek amacımız, hileli işlemlerin saklı temsilini zenginleştirmek ve bunları normal işlemlerden ayırt etme yeteneklerini artırmaktır. Otokodlayıcı modelini NFE ile eğittikten sonra, gerçek dolandırıcılık verilerini ve NFE ile geliştirilmiş dolandırıcılık verilerini birleştiriyoruz. Bu adım, dolandırıcılık işlemlerinin zenginleştirilmiş temsillerini orijinal dolandırıcılık verileri ile birleştirmeyi amaçlar, sonuç olarak dolandırıcılık tespiti için daha kapsamlı ve temsilci bir veri kümesi oluşturur. Şekil 6.1'de gösterildiği gibi gerçek dolandırıcılık verilerinin NFE ile geliştirilmiş verilerin birleştirilmesindeki amacımız, daha geniş bir dolandırıcılık deseni yelpazesini yakalamak ve dolandırıcılık tespit modellerimizin genelleme yeteneğini artırmaktır.



Şekil 6.1: Kodlama yöntemli sentetik dolandırıcılık verilerini SMOTE ile birleştirme. X_1 dolandırıcılık verilerini temsil eder, X_2 kodlu gürültü ile sentetik dolandırıcılık verilerini temsil eder ve X_3, X_1 ve X_2'yi birleştirerek SMOTE ile üst örnekleme yaparak elde edilen verileri temsil eder.

CIP çözümü için yaygın bir şekilde tercih edilen SMOTE yöntemini kullanıyoruz. SMOTE, komşu azınlık sınıfı örnekleri arasında interpolasyon yaparak sentetik örnekler oluşturur. Gerçek dolandırıcılık verileri ve NFE ile geliştirilmiş verilerin birleştirilmiş veri kümemize SMOTE uygulayarak, azınlık sınıfın temsilini (dolandırıcılık örnekleri) arttırır ve dolandırıcılık tespit modellerimizi eğitmek için daha dengeli bir veri kümesi oluştururuz. SMOTE, modellerimizin daha çeşitli dolandırıcılık örnekleri üzerinden öğrenmelerini sağlar, bu da normal ve hileli işlemler arasındaki ayrım yeteneklerini artırır. Ayrıca, sentetik dolandırıcılık verilerini oluşturmak için AE-NFE yerine CAE-NFE ve VAE-NFE karşılaştırmaları da sunulmuştur. Sonuçlar, AE-NFE'nin en iyi performansı gösterdiğini ve AE-NFE kullanımının üretilen sentetik dolandırıcılık verilerinin kalitesini artırabileceğini göstermiştir.

6.3. Veri Kümeleri

Çalışmamızda üç farklı veri kümesi kullandık. İlk veri kümesi olan Şekil 6.2'de sunulan Veri Kümesi-I, daha önce ilgili çalışmalarda kullanılmış olan ve kamuya açık bir veri kümesidir. Bu veri kümesi, 2013 Eylül ayında Avrupalı kart sahipleri tarafından gerçekleştirilen kredi kartı işlemlerini içermektedir. Bu veri kümesi 284807 işlem içerir ve bunlardan 492'si (%0.17) dolandırıcılık içeren veridir. Her işlem, gizlilik nedeniyle bir PCA dönüşümü sonucu elde edilen 30 sayısal giriş özelliği içerir. Orijinal özellikler ve açıklamaları, gizlilik nedeniyle ifşa edilmemiştir.

Out[3]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V21
223361	1.955041	-0.380783	-0.315013	0.330155	-0.509374	-0.086197	-0.627978	0.035994	1.054560	-0.030441	...	0.238197
165061	-0.400975	-0.626943	1.555339	-2.017772	-0.107769	0.168310	0.017959	-0.401619	0.040378	0.611115	...	-0.153485
238186	0.072509	0.820566	-0.561351	-0.709897	1.080399	-0.359429	0.787858	0.117276	-0.131275	-0.638222	...	-0.314638
150562	-0.535045	1.014587	1.750679	2.769390	0.500089	1.002270	0.847902	-0.081323	0.371579	0.560595	...	0.063525
138452	-4.026938	1.897371	-0.429786	-0.029571	-0.855751	-0.480406	-0.435632	1.313760	0.536044	1.221746	...	-0.480691
...	...	...	...	...	...	...	...	...	...	...	...	...
119879	1.173488	0.100792	0.490512	0.461596	-0.296377	-0.213165	-0.165254	0.119221	-0.114199	0.079128	...	-0.186027
259178	-0.775981	0.144023	-1.142399	-1.241113	1.940358	3.912076	-0.466107	1.360620	0.400697	-0.654029	...	0.037078
131932	-0.146609	0.992946	1.524591	0.485774	0.349308	-0.815198	1.076640	-0.395316	-0.491303	-0.212753	...	0.052649
146867	-2.948638	2.354849	-2.521201	-3.798905	1.866302	2.727695	-0.471769	2.217537	0.580199	-0.027572	...	-0.332759
121958	1.233174	-0.784851	0.386784	-0.698559	-1.034018	-0.637028	-0.502369	-0.188057	-0.749637	0.543016	...	0.027634

227845 rows × 30 columns

Şekil 6.2: Veri Kümesi-I

Şekil 6.3'de sunulan Veri Kümesi-II[6] bir mobil para işlem ağı simülasyonundan elde edilmiştir ve genellikle gelişmekte olan ülkelerde sıkça kullanılan finansal hizmetleri modellemek amacıyla oluşturulmuştur. Bu veri kümesi 6362620 işlem içerir ve bunlardan 8213'ü (%0.13) dolandırıcı verisidir. Sütun adları step, type, amount, nameOrig, oldbalanceOrg, newbalanceOrig, nameDest, oldbalanceDest, newbalanceDest ve isFraud'ı içerir. Step sütunu simülasyondaki zaman adımını temsil ederken, type sütunu işlem türünü gösterir. nameOrig ve nameDest sütunları sırasıyla müşteri ve tüccar hesap kimliklerine karşılık gelir. oldbalanceOrg ve newbalanceOrig sütunları gönderen hesabın başlangıç ve bitiş bakiyelerini belirtirken, oldbalanceDest ve newbalanceDest sütunları alıcı hesabın başlangıç ve bitiş bakiyelerini yansıtır. Son olarak, isFraud sütunu işlemin dolandırıcılık içerip içermediğini gösterir.

Out[3]:

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	is
0	1	PAYMENT	9839.64	C1231006815	170136.00	160296.36	M1979787155	0.00	0.00	
1	1	PAYMENT	1864.28	C1666544295	21249.00	19384.72	M2044282225	0.00	0.00	
2	1	TRANSFER	181.00	C1305486145	181.00	0.00	C553264065	0.00	0.00	
3	1	CASH_OUT	181.00	C840083671	181.00	0.00	C38997010	21182.00	0.00	
4	1	PAYMENT	11668.14	C2048537720	41554.00	29885.86	M1230701703	0.00	0.00	
...	...	...	...	...	...	...	...	...	...	...
6362615	743	CASH_OUT	339682.13	C786484425	339682.13	0.00	C776919290	0.00	339682.13	
6362616	743	TRANSFER	6311409.28	C1529008245	6311409.28	0.00	C1881841831	0.00	0.00	
6362617	743	CASH_OUT	6311409.28	C1162922333	6311409.28	0.00	C1365125890	68488.84	6379898.11	
6362618	743	TRANSFER	850002.52	C1685995037	850002.52	0.00	C2080388513	0.00	0.00	
6362619	743	CASH_OUT	850002.52	C1280323807	850002.52	0.00	C873221189	6510099.11	7360101.63	

Şekil 6.3: Veri Kümesi-II

[6] https://www.kaggle.com/datasets/ealaxi/paysim1

Şekil 6.4'de sunulan Veri Kümesi-III[7] IEEE tarafından sağlanmış ve dolandırıcılık tespit yarışması temelinde hizmet vermiş bir veri kümesidir. Bu veri kümesi, anonim bir banka tarafından 2019 Eylül ayında iki günlük bir süre içinde gerçekleştirilen işlemleri kapsar. İşlem dosyası 590540 işlem içerir ve bunlardan 5471'i (%0.93) dolandırıcı verisidir. Ayrıca 135229 benzersiz kimliği temsil eden 144233 girişi içeren bir kimlik dosyası bulunmaktadır. Üç veri kümesinde de dolandırıcılık içeren işlemler, toplam işlem sayısına göre küçük bir yüzdeyi içerir ve bu durum Şekil 6.5 ve Şekil 6.6 görsellerinde sunulmuştur.

Out[2]:

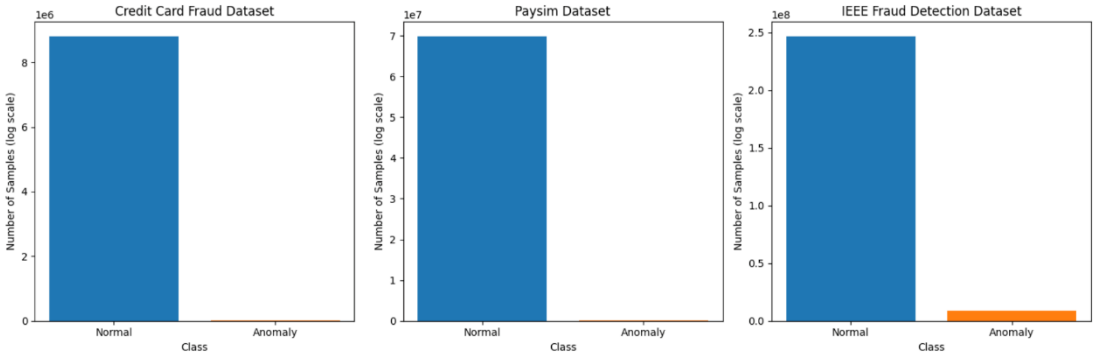
	TransactionID	isFraud	TransactionDT	TransactionAmt	ProductCD	card1	card2	card3	card4	card5	...	id_31	id_32
0	2987000	0	86400	68.50	W	13926	NaN	150.0	discover	142.0	...	NaN	NaN
1	2987001	0	86401	29.00	W	2755	404.0	150.0	mastercard	102.0	...	NaN	NaN
2	2987002	0	86469	59.00	W	4663	490.0	150.0	visa	166.0	...	NaN	NaN
3	2987003	0	86499	50.00	W	18132	567.0	150.0	mastercard	117.0	...	NaN	NaN
4	2987004	0	86506	50.00	H	4497	514.0	150.0	mastercard	102.0	...	samsung browser 6.2	32.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
590535	3577535	0	15811047	49.00	W	6550	NaN	150.0	visa	226.0	...	NaN	NaN
590536	3577536	0	15811049	39.50	W	10444	225.0	150.0	mastercard	224.0	...	NaN	NaN
590537	3577537	0	15811079	30.95	W	12037	595.0	150.0	mastercard	224.0	...	NaN	NaN
590538	3577538	0	15811088	117.00	W	7826	481.0	150.0	mastercard	224.0	...	NaN	NaN
590539	3577539	0	15811131	279.95	W	15066	170.0	150.0	mastercard	102.0	...	NaN	NaN

590540 rows × 434 columns

Şekil 6.4: Veri Kümesi-III

Veri ön işleme aşaması, her veri kümesi için özgü adımlar içermektedir. Veri Kümesi-I için, 'Amount' özelliği StandardScaler kullanılarak standartlaştırıldı ve 'scaledAmount' adı verilen bir özellik oluşturuldu. Benzer şekilde, 'Time' özelliği de standartlaştırıldı ve 'scaledTime' olarak dönüştürüldü. Daha sonra, orijinal 'Amount' ve 'Time' sütunları veri kümesinden kaldırıldı.

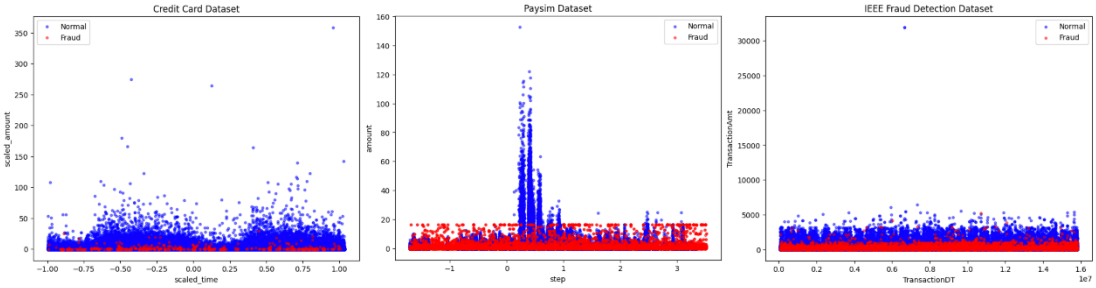
[7] https://www.kaggle.com/competitions/ieee-fraud-detection/data



Şekil 6.5: Üç veri kümesi için normal ve anormal verilerinin görünümü

Veri Kümesi-II için, 'nameOrig' ve 'nameDest' gibi gereksiz sütunlar daha fazla analiz için dışlandı. Kategorik özellik 'type' sayısal olarak kodlandı ve farklı işlem türlerine sayısal değerler atandı. Ayrıca, 'step,' 'amount,' 'oldbalanceOrg,' 'newbalanceOrig,' 'oldbalanceDest' ve 'newbalanceDest' özellikleri StandardScaler kullanılarak standartlaştırıldı.

Veri Kümesi-III için, ön işleme adımları 'transactions' ve 'identity' veri kümelerini paylaşılan bir endekse göre birleştirme işlemi içermekteydi. Kategorik sütunlar tanımlandı ve bu sütunlardaki eksik değerler 'unknown' etiketi ile dolduruldu. Ayrıca, kategorik özellikleri sayısal temsillere dönüştürmek için etiketleme kodlaması uygulandı. Son olarak, sayısal sütunlar veri kümesi genelinde tutarlı bir ölçekleme sağlamak için StandardScaler kullanılarak standartlaştırıldı.



Şekil 6.6: Üç farklı veri kümesi için normal (mavi) ve anormallik (kırmızı) durumlarının dağılımları

Bu ön işleme prosedürleri, veriyi sonraki analiz ve modelleme işlemleri için optimize etmek amacıyla gerçekleştirildi. Bu şekilde dolandırıcılık tespit algoritmalarının doğruluğu ve etkinliğinin arttırılması hedeflenmiştir.

YEDİNCİ BÖLÜM

DENEYSEL ÇALIŞMA

Bu çalışmada kullanılan veri kümelerini rastgele bir şekilde iki alt kümeye ayırdık: eğitim için %80 ve test için %20. AE, CAE ve VAE ile verinin gizli temsilini öğrenmek için çeşitli otokodlayıcı modellerini kullandık. Bu modellerin dayanıklılığını artırmak için giriş verilerine gürültü ekledik. Yaklaşımımız, sentetik hileli işlemler oluşturmak için bir gürültü faktörü olarak adlandırılan α ile kodlama yöntemlerini (AE, VAE, CAE) kullanmayı içerir. Bu kodlama yöntemleri, orijinal hileli olmayan işlemleri giriş olarak alır ve bunları daha düşük boyutlu gizli bir uzayda kodlar. Daha sonra her kodlama yönteminin kodlayıcısı, gizli uzaydan örneklem yaparak ve gürültü ekleyerek sentetik dolandıcılık içeren işlemler oluşturur. Elde edilen sentetik veri, orijinal dolandırıcı olmayan işlemlerin bir kombinasyonunu oluşturur. Bu tanımı aşağıdaki Denklem (7-1) şeklinde tanımlarız:

$$\mathrm{x} = \left| \mathrm{X}_{synthetic} \right| \tag{7-1}$$

Burada x, sentetik verinin boyutunu temsil eder. Ayrıca aşağıdaki Denklem **(7-2)**'de normal (Gaussian) dağılımı kullanırız:

$$\mathcal{N}(\mu, \sigma, \mathrm{x}) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \mathrm{e}^{-\frac{(\mathrm{x}-\mu)^2}{2\sigma^2}} \tag{7-2}$$

Gürültü eklemek için burada μ ve σ, sırasıyla normal dağılımın ortalama ve standart sapmasını temsil eder. Son sentetik veri, $\mathbf{X}_{synthetic\_noisy}$ aşağıdaki Denklem **(7-3)**'deki gibi elde edilir:

$$\mathbf{X}_{synthetic\_noisy} = \mathrm{X}_{synthetic} + \alpha \cdot \mathcal{N}(\mu, \sigma, x) \tag{7-3}$$

Burada, $\mathbf{X}_{synthetic\_noisy}$, gürültü eklenmiş sentetik veriyi temsil eder ve $\mathcal{N}(\mu, \sigma, x)$ sentetik veri ile eşleşen boyuta sahip bir normal dağılımı, μ ortalamayı, σ standart sapmayı temsil eder.

Yeni sentetik dolandırıcılık içeren veri oluşturmak için eğitilmiş otokodlayıcı modellerini kullanarak dolandırıcılık içeren veriyi yeniden oluşturuyoruz ve ek gürültü

ekliyoruz. Bu işlem, orijinal dolandırıcılık içeren veriye oldukça benzeyen ancak tamamen aynı olmayan yeni veri noktaları üretir, böylece değişkenlik eklenir. Ardından bu sentetik dolandırıcılık veri noktalarını orijinal doalndırıcılık içeren veri ile birleştiriyor ve SMOTE kullanarak fazladan örnekleme uyguluyoruz. Daha sonra bu fazladan örnekleme verisi üzerinde RF sınıflandırıcı ile model eğitiyoruz ve test verisi üzerinde performansını değerlendiriyoruz. Bu işlem, AE, CAE ve VAE olmak üzere üç otokodlayıcı modelinin her biri için tekrarlanır ve performanslarını karşılaştırmamıza olanak tanır.

Deneyimizin uygulanması ve sonuçların oluşturulması için çeşitli amaçlar için geniş kabul görmüş kütüphanelerden yararlandık (Açıklamalar için pypi.org[8] sitesinden faydanlandık):

- Yapay sinir ağları işlemlerini yürütmek için TensorFlow (tensorflow 2.14.0)
- Veri yapılarını manipüle etmek ve veri analizi yapmak için Pandas (pandas 2.1.2)
- Sayısal işlemlerle başa çıkmak için NumPy (numpy 1.26.1)
- Veriyi görselleştirmek için Matplotlib (matplotlib 3.8.1)
- Makine öğrenme görevleri ve değerlendirme metriklerinin hesaplanması için Scikit-Learn (scikit-learn 1.3.2)
- Yapay sinir ağlarını tanımlamak ve eğitimlerini denetlemek için Keras (keras 2.15.0)
- Sınıf dengesizliği sorunlarını etkili bir şekilde ele almak için Imbalanced-Learn (imbalanced-learn 0.11.0)

Şeffaflığı sağlamak ve çoğaltılabilirliği teşvik etmek amacıyla, tüm kod tabanını, tüm betikler ve ilgili belgeler dâhil, GitHub deposu üzerinden erişilebilir hale getirdik. Kaynaklara şu adresten ulaşabilirsiniz: [https://github.com/mert-cakir/ae-based-nfe]. Önerilen yöntemin sözde kodu Algoritma 1'de sunulmuştur.

[8] https://pypi.org/

Algorithm 1 Proposed method for generating synthetic fraud data

Load credit card data($\mathbf{X}$)
$\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{test}} \leftarrow \text{SplitData}(\mathbf{X}, 0.8)$
autoencoder $\leftarrow \text{TrainAutoencoder}(\mathbf{X}_{\text{train\_normal}})$
$\mathbf{X}_{\text{synthetic\_fraud}} \leftarrow \text{GenerateSyntheticTransactions}(\text{autoencoder}, \mathbf{X}_{\text{train\_fraud}})$
$\mathbf{X}_{\text{synthetic\_fraud\_noisy}} \leftarrow \text{AddNoiseToTransactions}(\mathbf{X}_{\text{synthetic\_fraud}})$
$\mathbf{X}_{\text{new\_train\_fraud}} \leftarrow \mathbf{X}_{\text{synthetic\_fraud\_noisy}} \cup \mathbf{X}_{\text{train\_fraud}}$
$\mathbf{X}_{\text{new\_train}} \leftarrow \mathbf{X}_{\text{train\_normal}} \cup \mathbf{X}_{\text{new\_train\_fraud}}$
$\mathbf{X}_{\text{resampled}}, \mathbf{y}_{\text{resampled}} \leftarrow \text{SMOTE}(\mathbf{X}_{\text{new\_train}}, \mathbf{y}_{\text{new\_train}})$
rf $\leftarrow \text{TrainRandomForest}(\mathbf{X}_{\text{resampled}}, \mathbf{y}_{\text{resampled}})$
$\mathbf{y}_{\text{pred}} \leftarrow \text{rf.predict}(\mathbf{X}_{\text{test}})$
function TrainAutoencoder($\mathbf{X}_{\text{train}}$)
    Initialize autoencoder model: autoencoder
    Set learning rate: 0.001, network initiation seed: 42
    Set batch size: 32, epoch: 100
    Set activation function: tanh, ReLU
    Set input dimension: 28, encoding dimension: 14
    Set number of layers: 4
    if autoencoder is AE or CAE **then**
        Set loss function: MSE
    else
        Set loss function: KL
    end if
    Set optimization algorithm: Adam
    Compile autoencoder model with specified parameters
    Train autoencoder on $\mathbf{X}_{\text{train}}$ using specified metrics
    return trained autoencoder model
end function
function GenerateSyntheticTransactions(autoencoder, $\mathbf{X}_{\text{normal}}$)
    Initialize empty synthetic fraud data: $\mathbf{X}_{\text{synthetic}}$
    for each normal transaction $\mathbf{x}$ in $\mathbf{X}_{\text{normal}}$ **do**
        Encode $\mathbf{x}$ using the trained autoencoder:
        $\mathbf{z} \leftarrow \text{Encode}(\text{autoencoder}, \mathbf{x})$
        Generate synthetic fraud transaction:
        $\mathbf{x}_{\text{synthetic}} \leftarrow \text{Decode}(\text{autoencoder}, \mathbf{z})$
        Add $\mathbf{x}_{\text{synthetic}}$ to $\mathbf{X}_{\text{synthetic}}$
    end for
    return $\mathbf{X}_{\text{synthetic}}$
end function
function AddNoiseToTransactions($\mathbf{X}_{\text{synthetic}}$)
    Set noise factor $\alpha = 0.2$, Normal Distribution $= \mathcal{N}$
    Set normal distribution parameters: mean $(\mu) = 0.0$, sd $(\sigma) = 1.0$
    $\mathbf{x}_{\text{synthetic}} \leftarrow \mathbf{x}_{\text{synthetic}} + \alpha \times \mathcal{N}(\mu, \sigma, \text{size}(\mathbf{x}_{\text{synthetic}}))$
    return $\mathbf{X}_{\text{synthetic}}$
end function

$\mathcal{N}$, ortalama 0 ve standart sapma 1 olan normal dağılımı temsil ederken, α kodlanmış veriye eklenen gürültü miktarını belirler. SMOTE algoritması, her devirde (epoch) sentetik dolandırıcılık içeren verilerini fazladan örneklendirmek için kullanılır. Kodlama ve çözme modelleri, orijinal dolandırıcılık içeren veri kümesini kullanarak önceden eğitilmiştir. Model performansını AUC-ROC ve F_1 skoru kullanarak değerlendirdik. Ayrıca CCF algılamadaki etkinliğini değerlendirmek için geleneksel SMOTE yöntemiyle yaklaşımımızı karşılaştırdık.

Accuracy (ACC):

$$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \tag{7-4}$$

Precision (P):

$$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{7-5}$$

Recall (R):

$$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{7-6}$$

Specificity (SP):

$$\frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}} \tag{7-7}$$

F_1 score (F_1):

$$\frac{2 * \mathrm{Precision} * \mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} = \frac{2 * \mathrm{TP}}{2 * \mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \tag{7-8}$$

Matthews Correlation Coefficient (MCC):

$$\frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}} \tag{7-9}$$

Çalışmamızda, önerilen üst örnekleme yöntemlerimizin bir sınıflandırma görevindeki performansını değerlendirmek için çeşitli değerlendirme metrikleri kullandık. Genel sınıflandırma performansını ölçmek için doğruluk (accuracy, Denklem **(7-4)**)

kullandık. Ancak, dengesiz veri kümeleri içeren durumlarda, bir sınıfın diğerini belirgin şekilde aştığı durumlarda, yalnızca doğruluk kullanmak yanıltıcı olabilir. Daha kapsamlı bir değerlendirme sağlamak amacıyla ayrıca kesinlik (precision Denklem **(7-5)**) ve duyarlılık (recall, Denklem **(7-6)**) kullanıldı. Kesinlik, tüm pozitif tahminler arasındaki doğru pozitif tahminlerin doğruluğunu ölçerken; duyarlılık, aynı zamanda hassasiyet veya doğru pozitif oran olarak bilinir ve sınıflandırıcının pozitif örnekleri doğru bir şekilde tanımlama yeteneğini nicelendirir. Ayrıca, negatif örnekleri doğru bir şekilde tanımlama yeteneğini ölçmek için özgüllük (specificity, Denklem **(7-7)**) olarak bilinen, aynı zamanda doğru negatif oran olarak bilinen bir metrik kullanıldı. F_1 skoru (Denklem (7-8)), dengesiz sınıflandırma problemleri için yaygın olarak kullanılan bir metrik olup, kesinlik ve duyarlılık aritmetik ortalamasını temsil eder, ikisi arasında bir denge kurar. Ayrıca, pozitif ve negatif örnekleri ayırt etme yeteneğini değerlendirmek için AUC-ROC metriğini kullandık. AUC-ROC, çeşitli eşik değerleri üzerinde doğru pozitif oranı (duyarlılık) ve yanlış pozitif oranı dikkate alır. Bununla birlikte doğru ve yanlış pozitifler ile negatifleri hesaba katan bir korelasyon katsayısı olan MCC (Denklem (7-9)) kullanıldı. MCC değeri, -1 (tam anlaşmazlık) ile 1 (mükemmel sınıflandırma) arasında değişmektedir. Bu metrikler, önerilen üst örnekleme yöntemlerimizin sınıflandırma zorluklarını ele almadaki etkinliğini kapsamlı bir şekilde değerlendirmemizi sağlamıştır.

SEKİZİNCİ BÖLÜM

SONUÇLAR VE ÖNERİLER

Bu çalışma, KPA tespiti konusundaki zorlukları ele almak ve finansal sistemlerin güvenliğini artırmak amacıyla geliştirilen yenilikçi yaklaşımın etkililiğini değerlendirmektedir. Bu kapsamda dolandırıcılık tespiti, normal ve sahte işlemler arasındaki doğal sınıf dengesizliği nedeniyle önemli bir zorluk içermektedir. Bu tez ayrıca, otokodlayıcı tabanlı NFE yöntemi ile SMOTE'u birleştiren bir modelin kullanımını önermekte ve bu yöntemin dolandırıcılık tespiti performansını nasıl etkilediğini değerlendirmektedir. Otokodlayıcının farklı varyasyonlarını içeren bu yaklaşım, dolandırıcılık tespiti modellerinde sınıf dengesizliğini ele alarak daha kesin sonuçlar elde etmeyi amaçlamaktadır.

8.1. Sınıflandırma Aşamasında Derin Öğrenme ve Kıyaslamaları

CCF tespitinde kullanılan farklı sınıflandırma yöntemlerinin performansını değerlendiren bu çalışmayla geleneksel ML teknikleri, DL yaklaşımları ve hibrit modeller ile çok aşamalı derin öğrenme modelleri oluşturulmuştur. Bu modeller karşılaştırılarak sonuçları analiz edilmiştir.

Autoencoder modelinin kullanılan parametre değerlerinde modelin eğitimi için belirlenen epoch sayısı 20'dir. Giriş verisinin özellik sayısı 30 olarak belirlenmiştir. Autoencoder'ın ilk kodlama katmanında 27 nöron, ikinci kodlama katmanında 24 nöron ve üçüncü kodlama katmanında 21 nöron bulunmaktadır. Gizli katmanın nöron sayısı 15 olarak seçilmiştir. Aktivasyon fonksiyonlarında L1 düzenleme kullanıldığı durumda öğrenme oranı 1e-7 olarak belirlenmiştir. Bu parametreler, Autoencoder modelinin mimarisini oluşturur. Modelin derlenmesi için 'adam' optimizer ve 'mean_squared_error' kayıp fonksiyonu kullanılmıştır. Modelin eğitimi sırasında, her epoch'ta eğitim ve test kayıp değerleri saklanır. Eğitim süreci sonunda, model giriş verisini daha düşük boyutlu bir temsile (encoding) dönüştürmeyi ve ardından orijinal veriyi yeniden oluşturmayı öğrenir.

VAE modelinin kullanılan parametre değerlerinde ilk kodlama katmanında 27 nöron, ikinci kodlama katmanında 24 nöron ve üçüncü kodlama katmanında 21 nöron bulunmaktadır. Gizli katman 15 nöron içermektedir. Model, öğrenilen latent değişken

sayısı olarak 2'yi kullanmaktadır. Optimizasyon işlemi için RMSprop optimizer'ı ve öğrenme oranı 1e-3 olarak belirlenmiştir. Modelin eğitileceği epoch sayısı 20'dir ve her eğitim adımında kullanılacak mini-batch boyutu 128 olarak seçilmiştir. VAE modeli, giriş verisini daha düşük boyutlu bir temsile dönüştürmeye çalışan üç kodlama katmanı içerir. Aynı zamanda, bu temsilin istatistiksel özelliklerini tutan bir gizli katman vardır. Model, orijinal veriyi bu düşük boyutlu temsil kullanarak yeniden oluşturmaya çalışarak eğitilir. Kayıp fonksiyonu, yeniden yapılandırma kaybı Kullback-Leibler (KL) kaybı kullanılarak hesaplanır. Model, RMSprop optimizer ile derlenir ve eğitilirken her epoch'ta eğitim ve doğrulama kayıp değerleri kaydedilir. Sonuç olarak, VAE modeli veriyi düşük boyutlu bir temsil ile öğrenir ve bu temsilin öğrenilmiş bir dağılım ile birleştirerek, veri setindeki varyansı daha etkin bir şekilde ele alabilir. Bu sayede, model yeni veriler üretebilir ve öğrendiği latent uzayı kullanarak üretken bir model oluşturabilir.

Tablo 8.1, Tablo 8.2, Tablo 8.3, Tablo 8.4 ve Tablo 8.5'de sunulduğu üzere RF ve DT gibi klasik ML yöntemleri, DNN ve CNN gibi derin öğrenme modelleri ile karşılaştırılmış ve tespit hassasiyeti üzerindeki etkileri incelenmiştir. Ayrıca, veri dengesizliği sorununu aşmak için oversampling teknikleri olarak SMOTE, SMOTEN, ADASYN, KMeans SMOTE ve SVM SMOTE kullanılmıştır. Bu çalışmada, AE ve VAE gibi özellik çıkarma yöntemlerinin de etkisi göz önüne alınmıştır. Elde edilen sonuçlar, farklı tekniklerin performansını karşılaştırarak, CCF tespiti alanında daha etkili yöntemlerin belirlenmesine katkı sağlamaktadır.

Tablo 8.1: Decision Tree ve Çoklu Model Karşılaştırmaları

Techniques	ACC %	P %	R %	SP %	F_1 %
DT	0.9992	0.8134	0.7414	0.6031	0.7758
AE + DT	0.9981	0.4635	0.4761	0.4873	0.4697
VAE + DT	0.9986	0.6133	0.6258	0.4867	0.6195
ROS + DT	0.9991	0.7448	0.7346	0.5131	0.7397
SMOTE + DT	0.9978	0.4264	0.7891	0.1657	0.5536
SMOTEN + DT	0.9991	0.7819	0.7074	0.5972	0.7428
ADASYN + DT	0.9975	0.3838	0.7414	0.1784	0.5058
Kmeans SMOTE + DT	0.9976	0.4057	0.7619	0.1758	0.5295
SVM SMOTE + DT	0.9990	0.6982	0.8027	0.3625	0.7468
ROS + AE + DT	0.9990	0.7464	0.7210	0.5324	0.7335
SMOTE + AE + DT	0.9993	0.8188	0.7687	0.5762	0.7929
SMOTEN + AE + DT	0.9989	0.6824	0.6870	0.4946	0.6847
ADASYN + AE + DT	0.9992	0.7762	0.7551	0.5294	0.7655
Kmeans SMOTE + AE + DT	0.9991	0.7659	0.7346	0.5416	0.7499
SVM SMOTE + AE + DT	0.9989	0.7000	0.6666	0.5384	0.6829
ROS + VAE + DT	0.9006	0.0150	0.8775	0.0021	0.0295
SMOTE + VAE + DT	0.8993	0.0148	0.8775	0.0020	0.0291
SMOTEN + VAE + DT	0.9986	0.5919	0.7006	0.3826	0.6417
ADASYN + VAE + DT	0.8077	0.0080	0.9047	0.0159	0.0008
Kmeans SMOTE + VAE+DT	0.9066	0.0155	0.8571	0.0026	0.0306
SVM SMOTE + VAE + DT	0.9768	0.0616	0.8775	0.0090	0.1152

Sınıflandırma aşamasında DT ile yapılan deneyler sonucunda en başarılı sonuçları SMOTE + AE + DT ile sağlamıştır.

Tablo 8.2: Random Forest ve Çoklu Model Karşılaştırmaları

Techniques	ACC %	P %	R %	SP %	F_1 %
RF	0.9995	0.9491	0.7619	0.8536	0.8452
AE + RF	0.9990	0.8235	0.5714	0.7777	0.6746
VAE + RF	0.9993	0.8617	0.7210	0.7068	0.7851
ROS + RF	0.9995	0.9576	0.7687	0.8717	0.8528
SMOTE + RF	0.9995	0.9090	0.8163	0.6923	0.8602
SMOTEN + RF	0.9994	0.9406	0.7551	0.8372	0.8377
ADASYN + RF	0.9994	0.8872	0.8027	0.6590	0.8428
Kmeans SMOTE + RF	0.9995	0.9083	0.8095	0.700	0.8561
SVM SMOTE + RF	0.9995	0.9160	0.8163	0.7105	0.8633
ROS + AE + RF	0.9994	0.9724	0.7210	0.9318	0.8281
SMOTE + AE + RF	0.9995	0.9491	0.7619	0.8536	0.8452
SMOTEN + AE + RF	0.9993	0.9494	0.6394	0.9137	0.7642
ADASYN + AE + RF	0.9995	0.9411	0.7619	0.8333	0.8421
Kmeans SMOTE + AE + RF	0.9993	0.9439	0.6870	0.8846	0.7952
SVM SMOTE + AE + RF	0.9993	0.9504	0.6530	0.9107	0.7741
ROS + VAE + RF	0.9584	0.0346	0.8639	0.0056	0.0667
SMOTE + VAE + RF	0.9577	0.0346	0.8775	0.0049	0.0666
SMOTEN + VAE + RF	0.9992	0.8195	0.7414	0.6129	0.7785
ADASYN + VAE + RF	0.8772	0.0127	0.9183	0.0011	0.0251
Kmeans SMOTE +VAE+RF	0.9565	0.0332	0.8639	0.0053	0.0639
SVM SMOTE + VAE + RF	0.9855	0.0945	0.8639	0.0161	0.1704

Sınıflandırmada RF modeli için kullanılan parametrelerde n_estimators parametresi 100 olarak belirlenmiştir. Bu parametre, Random Forest modelinin kaç tane ağaç içereceğini belirler. Bu durumda, 100 ağaç içeren bir Random Forest modeli oluşturulmuştur. RF ile yapılan deneyler sonucunda en başarılı sonuçları SVM SMOTE + RF sağlamıştır.

Tablo 8.3: DNN ve Çoklu Model Karşılaştırmaları

Techniques	ACC %	P %	R %	SP %	F_1 %
DNN	0.9993	0.8507	0.7755	0.6226	0.8113
AE + DNN	0.9991	0.8256	0.6122	0.7500	0.7031
VAE + DNN	0.9991	0.8446	0.5918	0.7894	0.6960
ROS + DNN	0.9984	0.5344	0.8435	0.1755	0.6543
SMOTE + DNN	0.9982	0.4901	0.8503	0.1447	0.6218
SMOTEN + DNN	0.9992	0.9207	0.6326	0.8709	0.7500
ADASYN + DNN	0.9980	0.4679	0.8435	0.1402	0.6019
Kmeans SMOTE + DNN	0.9984	0.5234	0.8367	0.1764	0.6439
SVM SMOTE + DNN	0.9988	0.6119	0.8367	0.2352	0.7068
ROS + AE + DNN	0.9993	0.8444	0.7755	0.6111	0.8085
SMOTE + AE + DNN	0.9920	0.1593	0.8435	0.0339	0.2681
SMOTEN + AE + DNN	0.9994	0.8750	0.7619	0.6862	0.8145
ADASYN + AE + DNN	0.9993	0.8455	0.7823	0.6037	0.8127
Kmeans SMOTE + AE + DNN	0.9993	0.8720	0.7414	0.7037	0.8014
SVM SMOTE + AE + DNN	0.9994	0.8692	0.7687	0.6666	0.8158
ROS + VAE + DNN	0.9898	0.1293	0.8571	0.0241	0.2247
SMOTE + VAE + DNN	0.9402	0.0248	0.8843	0.0033	0.0484
SMOTEN + VAE + DNN	0.9993	0.8201	0.7755	0.5689	0.7972
ADASYN + VAE + DNN	0.8929	0.0140	0.8843	0.0018	0.0276
Kmeans SMOTE+ VAE+DNN	0.9807	0.0715	0.8503	0.0133	0.1319
SVM SMOTE + VAE + DNN	0.9929	0.1789	0.8571	0.0350	0.2961

Sınıflandırmada DNN modeli için kullanılan parametre değerlerinde ilk katman, 15 giriş özelliği ile 16 ReLU aktivasyon fonksiyonuna sahip nöron içerir. İkinci katman, 10 ReLU aktivasyon fonksiyonuna sahip nöron içerir. Bu katmandan sonra bir Dropout katmanı gelir, bu katmanın dropout oranı 0.5'tir. Üçüncü katmanda tekrar 10 ReLU aktivasyonlu nöron bulunur. Dördüncü katmanda ise 12 ReLU aktivasyonlu nöron bulunmaktadır. Son katman, binary sınıflandırma için bir Sigmoid aktivasyon fonksiyonuna sahip tek bir nöron içerir. DNN modeli, 'adam' optimizer kullanılarak derlenir ve kayıp (loss) fonksiyonu olarak 'binary_crossentropy' seçilir. Eğitim sırasında, toplam 5 epoch boyunca, her biri 15 örneği içeren batch'ler kullanılarak

model eğitilir. DNN ile yapılan deneyler sonucunda en başarılı sonuçları SVM SMOTE + AE + DNN sağlamıştır.

Tablo 8.4: CNN ve Çoklu Model Karşılaştırmaları

Techniques	ACC %	P %	R %	SP %	F_1 %
CNN	99.94	92.30	73.46	81.25	81.81
AE + CNN	99.93	85.18	78.23	61.53	81.56
VAE + CNN	99.93	84.49	74.14	65.51	78.98
ROS + CNN	99.94	84.28	80.27	56.86	82.22
SMOTE + CNN	99.93	86.25	76.87	65.38	81.29
SMOTEN + CNN	99.93	85.71	77.55	63.46	81.42
ADASYN + CNN	99.94	89.14	78.23	69.56	83.33
Kmeans SMOTE + CNN	99.94	88.80	75.51	72	81.61
SVM SMOTE + CNN	99.94	89.43	74.82	74	81.48
ROS + AE + CNN	99.94	88.18	76.19	99.98	81.75
SMOTE + AE + CNN	99.94	88.28	76.87	99.98	82.18
SMOTEN + AE + CNN	99.93	88	74.82	99.98	80.88
ADASYN + AE + CNN	99.94	87.9	74.15	99.98	80.44
Kmeans SMOTE + AE + CNN	99.93	83.45	75.51	99.97	79.28
SVM SMOTE + AE + CNN	99.93	88.88	70.74	99.98	78.78
ROS + VAE + CNN	98.31	8.2	86.39	98.34	14.99
SMOTE + VAE + CNN	97.12	5.06	88.43	97.14	9.57
SMOTEN + VAE + CNN	99.93	82.6	77.55	99.97	80
ADASYN + VAE + CNN	88.5	1.38	93.87	88.5	2.73
Kmeans SMOTE+VAE+ CNN	97.43	5.48	85.71	97.45	10.30
SVM SMOTE + VAE + CNN	98.93	12.46	85.71	98.96	21.76

Sınıflandırmada CNN bölümünde kullanılan parametre değerlerinde ilk Conv1D katmanında 32 filtre ve 2 çekirdek boyutu kullanılmıştır. Bu katmanın aktivasyon fonksiyonu ReLU'dur. İlk katmandan sonra Batch Normalization ve Dropout (0.2 oranında) katmanları gelmektedir. Ardından, ikinci Conv1D katmanında 64 filtre ve 2 çekirdek boyutu kullanılmıştır. Bu katman da ReLU aktivasyon fonksiyonuna sahiptir. İkinci katmandan sonra da Batch Normalization ve Dropout (0.5 oranında) katmanları bulunmaktadır. CNN modelinin devamında Flatten katmanı gelir, bu katman veriyi

düzleştirir. Daha sonra iki yoğun (Dense) katman gelir. İlk yoğun katmanda 64 nöron bulunmaktadır ve ReLU aktivasyon fonksiyonu kullanılmıştır. Bu katmandan sonra bir Dropout (0.5 oranında) katmanı yer almaktadır. Son olarak, ikinci yoğun katman sadece 1 nöron içermekte ve aktivasyon fonksiyonu olarak Sigmoid kullanılmıştır. Bu yapı, binary sınıflandırma için uygundur. CNN ile yapılan deneyler sonucunda en başarılı sonuçları ADASYN + CNN sağlamıştır. Sınıflandırma aşamasında kullanılan tekniklerin en başarılı sonuçlarını kendi içerisinde kıyasladığımızda Tablo 8.5 elde edilmiştir.

Tablo 8.5: Çoklu Model Karşılaştırmaları

Techniques	ACC %	P %	R %	SP %	F_1 %
SMOTE + AE + DT	99.93	81.88	76.87	57.62	79.29
SVM SMOTE + RF	99.95	91.60	81.63	71.05	86.33
SVM SMOTE + AE + DNN	99.94	86.92	76.87	66.66	81.58
ADASYN + CNN	99.94	89.14	78.23	69.56	83.33

Çok aşamalı model karşılaştırmaları sonucunda hibrit modeller ile performansın iyileştirildiği gözlenmiştir. Bununla birlikte SVM SMOTE + RF en başarılı sonuçları vermiştir. F_1 sonuçlarında RF içeren çoklu modelden sonra ADASYN + CNN ve SVM SMOTE + AE + DNN sonuçları başarılı olmuştur. Bu sonuçlara istinaden veri sentezleme öneriminde sınıflandırma aşamasında RF tekniği kullanılmıştır.

8.2. Performans İyileştirmesi İçin Veri Sentezleme Önerimi

Bu bölümde, sınıf dengesizliğinin giderilmesi adına deneylerimizin sonuçlarını sunuyor ve analiz ediyoruz. Farklı yaklaşımların performansını, SMOTE, AE-NFE, VAE-NFE ve CAE-NFE dâhil olmak üzere, dolandırıcılık tespiti bağlamında karşılaştırıyoruz. Sonuçları Tablo 8.6 içerisinde sunuyoruz. Ayrıca, bu yaklaşımların üç farklı veri kümesi üzerindeki etkilerini tartışıyoruz.

Tablo 8.6: Deneyler için performans metrikleri

Data	Model	ACC %	R %	SP %	P %	F_1 %	AUC %	MCC %
1	SMOTE	99.95	84.69	99.98	89.24	86.91	98.99	86.91
	AE-NFE	99.95	82.65	99.98	93.10	87.56	99.41	87.70
	VAE-NFE	99.95	83.67	99.98	90.10	86.77	98.95	86.81
	CAE-NFE	99.95	85.71	99.98	89.36	87.50	98.98	87.49
2	SMOTE	99.91	95.98	99.92	61.73	75.13	99.88	76.94
	AE-NFE	99.92	96.04	99.93	63.79	76.69	99.86	78.24
	VAE-NFE	99.92	96.48	99.92	62.99	76.22	99.84	77.93
	CAE-NFE	99.92	96.04	99.92	63.51	76.46	99.85	78.07
3	SMOTE	97.85	45.07	99.82	90.61	60.20	91.85	63.07
	AE-NFE	97.91	46.53	99.82	90.92	61.56	93.14	64.22
	VAE-NFE	97.92	46.72	99.83	91.29	61.81	93.14	64.49
	CAE-NFE	97.90	46.34	99.82	90.85	61.37	92.93	64.06

Tablo 8.6, sonuçları ayrıntılı bir şekilde sunar ve AE-NFE yaklaşımımızın sürekli olarak Veri Kümesi-I ve Veri Kümesi-II'de SMOTE, VAE-NFE ve CAE-NFE yöntemlerini geride bıraktığını göstermektedir. AE-NFE yaklaşımı, kodlama süreci sırasında tanıtılan gürültü faktörünü kullanarak dolandırıcı örneklerin temsilini artırmak için kullanır ve bu da dolandırıcılık tespiti performansını artırır. Bu gelişme, AE-NFE'nin diğer yaklaşımlarla karşılaştırıldığında elde ettiği daha yüksek doğruluk, F_1 puanı, hassasiyet, duyarlılık, özgüllük, AUC-ROC ve MCC değerlerinde açıkça görülmektedir.

Ancak ilginç bir şekilde, Veri Kümesi-III'te VAE-NFE yöntemi en başarılı yaklaşım olarak ortaya çıkmaktadır. Bu, değişkenliklerin dolandırıcılık içeren işlemlerin belirsizliklerini daha hassas bir şekilde yakalama yeteneğine sahip varyasyonlu otokodlayıcının bu belirli veri kümesinde üstün performansına katkı sağladığını

göstermektedir. Bu, dolandırıcılık tespiti görevleri için farklı otokodlayıcı varyasyonlarını düşünmenin önemini vurguluyor, çünkü veri kümesi özelliklerine bağlı olarak farklı etkililik dereceleri sergileyebilmektedir. Tablo 8.7, çeşitli modellerin performansını, önerdiğimiz AE-NFE yöntemi de dâhil olmak üzere Veri Kümesi-I üzerinde karşılaştırmalı bir şekilde sunmaktadır.

Tablo 8.7: Veri Seti-I kullanılarak yapılan farklı modellerin performansının karşılaştırılması

Model	ACC %	R %	SP %	P %	F_1 %	AUC %	MCC %
AE - NFE	99.95	82.65	99.98	93.1	87.56	99.41	87.7
(Zou et al., 2019)	97.93	84	-	-	-	-	-
(Tingfei et al., 2020)	99.96	83.66	99.98	90.64	87	-	-
(Misra et al., 2020)	99.94	80.15	-	85.34	82.65	-	-
(T.H. Lin&Jiang,2021)	99.94	81.42	99.98	-	-	96.2	84.41
(Zioviris et al., 2022)	87.59	93.50	-	1.29	2.54	96.6	10.16
(Du et al., 2023)	99.93	80.39	99.97	-	-	-	85.06
(Y. Ding et al., 2023)	-	83.85	99.91	92.15	87.93	98.47	-

Tablo 8.7'de sunulan dolandırıcılık tespitinde sınıf dengesizliğini ele almak amacıyla tasarlanan AE-NFE yöntemi, %99.95 doğruluk, %82.65 geri çağırma, %99.98 özgüllük, %93.10 hassasiyet, %87.56 F_1 puanı, %99.41 AUC ve %87.70 MCC ile etkileyici sonuçlar sunmaktadır. Bu sonuçlar, hileli faaliyetlerin olağanüstü hassasiyet ve doğrulukla tanımlanmasında yaklaşımımızın etkililiğini vurgulamaktadır. Mevcut çalışmalarla karşılaştırıldığında, AE-NFE yöntemimiz güçlü bir rakip olarak öne çıkmaktadır. Örneğin, (Zou et al., 2019), üst örnekleme, gürültü gideren otokodlayıcı sinir ağları ve derin tam bağlantılı sinir ağlarını birleştirir, ancak genellikle hassasiyet, F_1 puanı, AUC ve MCC dâhil birçok ölçümde AE-NFE yöntemimizi aşmaz. Benzer şekilde, (Tingfei et al., 2020), VAE kullanarak yenilikçi bir yaklaşım sunar, ancak birçok ölçümde AE-NFE yöntemimizi aşmaz. Ayrıca, (Misra et al., 2020), otokodlayıcı temelli özellik çıkarma ile iki aşamalı bir yöntem sunar, ancak

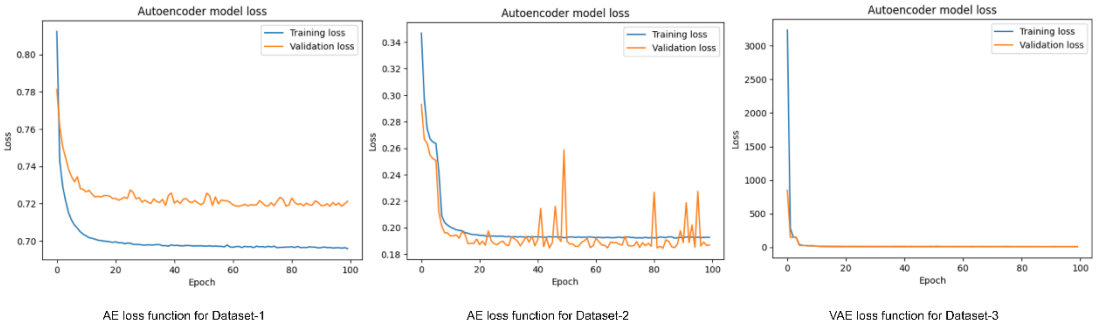
yaklaşımımız genellikle daha yüksek doğruluk ve geri çağırma sergiler. Dahası, (T.-H. Lin & Jiang, 2021), AE'yi olasılıksal RF ile birleştirir ve bazı ölçümlerde rekabetçi sonuçlar elde eder. Ancak AE-NFE, daha yüksek hassasiyet ve AUC seviyelerini sürdürür. Öte yandan, (Zioviris et al., 2022), çok aşamalı bir derin öğrenme modeli kullanır, ancak doğruluk, hassasiyet ve MCC açısından yetersiz kalır. Ayrıca, (Du et al., 2023), LightGBM ile otokodlayıcıları kullanır, ancak AE-NFE'nin genel performansını aşmaz. Dahası, (Y. Ding et al., 2023), VAE-GAN tabanlı geliştirilmiş bir üst örnekleme tekniği tanıtır, ancak çoğu ölçümde yaklaşımımızı tutarlı bir şekilde aşmaz. Bu karşılaştırmalar, AE-NFE yönteminin, sınıf dengesizliği sorununu ele alabilme ve dolandırıcılık tespit modellerinin doğruluğunu artırabilme konusundaki etkililiğini vurgulamaktadır. Özetle, AE-NFE yöntemi, var olan en iyi yöntemlere kıyasla üstün veya rekabetçi sonuçlar sunan etkileyici ve çok yönlü bir çözüm olarak ortaya çıkar. Yüksek hassasiyet ve doğruluk sağlama yeteneği, sınıf dengesizliğini ele alma ve farklı performans metriklerinde rekabet avantajını sürdürme yeteneğini vurgulamaktadır.

8.3. SMOTE ile Karşılaştırma

Deneylerimizde üç NFE tabanlı yaklaşımın (AE-NFE, VAE-NFE ve CAE-NFE) hepsi yalnızca SMOTE'tan daha iyi performans göstermiştir. SMOTE, üst örnekleme için yaygın olarak kullanılsa da hileli işlemlerin karmaşık özelliklerini ve karmaşıklıklarını tam olarak yakalayamayabilir. Bunun aksine, NFE tabanlı yaklaşımlar, dolandırıcılık örneklerinin gizli temsillerini zenginleştirir ve bunları normal örneklerden daha iyi ayırt etmelerini sağlar, bu da dolandırıcılık tespit performansını artırır. Bu, sınıf dengesizliği ile başa çıkmak ve dolandırıcılık tespit modellerinin doğruluğunu artırmak için otokodlayıcı tabanlı NFE'nin SMOTE ile birleştirilmesinin etkililiğini vurgular.

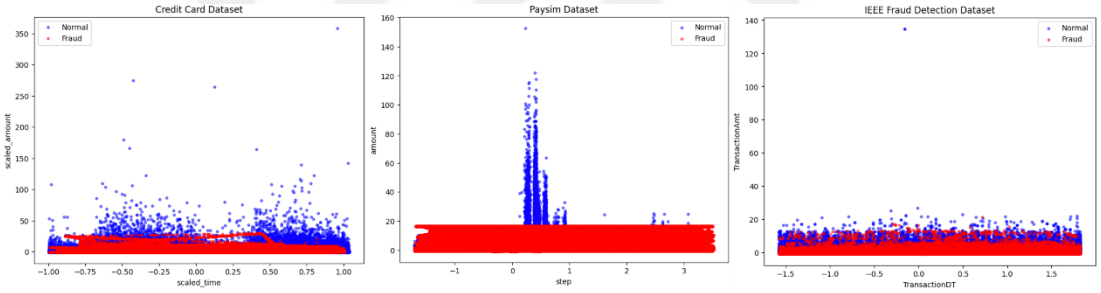
8.4. Analiz Görselleştirme

Şekil 8.1, farklı yaklaşımların kayıp fonksiyonlarını göstermektedir. Gözlemlediğimiz üzere, AE-NFE yaklaşımı diğer yöntemlere göre tutarlı olarak daha düşük kayıp değerlerine ulaşır, bu da daha iyi model yakınsama ve yeniden oluşturma doğruluğunu gösterir. Bu sonuçlar, değerlendirme metriklerindeki üstün performansıyla uyumlu olmaktadır.



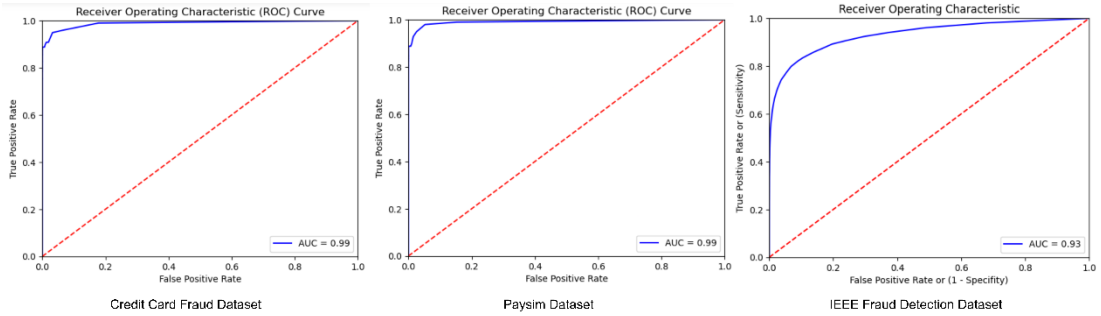
Şekil 8.1: Üç veri kümesi için AE modellerinin kayıp fonksiyonları

Şekil 8.2'de, oversampling sonucu oluşan görüntüleri sunuyoruz. Bu görüntüler, SMOTE ve NFE tabanlı yaklaşımların sentetik örnekler oluşturarak azınlık sınıfını (dolandırıcılık durumları) nasıl artırdığını göstermektedir. NFE tabanlı yaklaşımların, orijinal dolandırıcılık durumlarına daha yakından benzeyen sentetik örnekler ürettiğini ve dolandırıcılık altında yatan karakteristikleri daha etkili bir şekilde yakaladığını gözlemliyoruz. Bu görsel onay, NFE tabanlı yaklaşımların temsilci dolandırıcılık durumları ile veri kümesini zenginleştirme yeteneğini vurgulamaktadır.



Şekil 8.2: Üst örnekleme sonrasında üç farklı veri kümesinde normal (mavi) ve anormallik (kırmızı) durumlarının dağılımları

Şekil 8.3, ROC eğrilerini gösteren grafikler, farklı yaklaşımların ayırt etme yeteneklerini (normal/fraud) sunar. Gözlemlediğimiz üzere, AE-NFE sürekli olarak daha yüksek bir AUC-ROC değeri gösterir, bu da normal ve dolandırıcılık içeren işlemleri ayırt etme konusundaki genel performansın daha iyi olduğunu gösterir. Bu, AE-NFE'nin dolandırıcılık tespiti görevlerindeki üstünlüğünü daha da desteklemektedir.



Şekil 8.3: Farklı Yaklaşımların ROC Eğrisi Analizi

Bu çalışmada sunulan deneysel sonuçlar, kodlama ve gürültü faktörü yaklaşımı kullanılarak oluşturulan sentetik dolandırıcılık verilerinin, SMOTE ile birleştirilerek sınıf dengesizliği ile başa çıkmak için etkili bir yöntem olduğunu önermektedir. Yaklaşımımız, SMOTE ile karşılaştırıldığında daha iyi performans gösterdi, aynı zamanda diğer test edilen otokodlayıcı tabanlı yöntemleri de geride bıraktı.

8.5.Sonuç ve Gelecek Çalışmalar

Sonuç olarak, çalışmamız otokodlayıcı tabanlı NFE'nin SMOTE ile entegrasyonunun sınıf dengesizliğini etkili bir şekilde ele almak için büyük bir potansiyele sahip olduğunu göstermiştir. AE-NFE yaklaşımı birçok veri kümesi üzerinde üstün performans sergilemiş ve gerçek dünya uygulamaları için uygunluğunu göstermiştir. Ancak, farklı veri kümeleri üzerinde performans değişkenliği gözlemlenmiştir. Bu, veri kümesi özel karakteristiklerin dikkate alınması gerektiğini ve VAE-NFE ve CAE-NFE gibi alternatif otokodlayıcı varyantlarını incelemek için ihtiyaç olduğunu vurgulamaktadır. Ayrıca, gürültü faktörü parametrelerinin daha iyi ayarlanması ve otokodlayıcı model mimarilerinin optimize edilmesi için daha derin bir incelemeyi gerektirir.

Bu yaklaşımların daha büyük ve daha çeşitli veri kümeleri üzerindeki genelleme yeteneklerini dikkate alarak gelecekteki araştırmanın kapsamını genişletmenin gerekliliğini vurguluyoruz. Bu adım, bu yaklaşımların farklı dolandırıcılık tespiti senaryolarında etkinliğini doğrulamak ve güvenilir dolandırıcılık tespiti metodolojilerinin daha da gelişmesine katkıda bulunmak için kritik bir öneme sahiptir. Dolandırıcılıkların devam ettiği bir dünyada, araştırmamız dolandırıcılık tespit sistemlerinin doğruluğunun ve dayanıklılığının artırılmasının yanı sıra farklı alanlardaki yeni tehditlere karşı bir kalkan olarak hizmet etmek için otokodlayıcılar ve

üst örnekleme iş birliğini kullanan yenilikçi stratejilere acil bir ihtiyacı vurgulamaktadır.

Bu çalışmanın katkıları, finansal suçlar ve dolandırıcılık tespiti alanında yeni yöntemlerin geliştirilmesi ve mevcut modellerin güçlendirilmesi konusunda araştırmacılara önemli bir rehberlik sunmaktadır.

- Geleneksel ML ve DL yaklaşımları ile çok aşamalı derin öğrenme modellerinin oluşturulmasını içeren kapsamlı deneyler yapılmıştır.
- Çok aşamalı modellerin kullanılmasının, genellikle daha yüksek performans sağladığı gösterilmiştir.
- Finansal suçlar aracılığıyla gerçekleştirilen KPA konusunda CIP için yenilikçi bir çözüm olarak otokodlayıcı tabanlı gürültü faktörü kodlama yöntemi önerilmiştir.
- Önerilen üst örnekleme yöntemi, sınıf dengesizliği sorununu çözmek adına SMOTE ile entegre edilmiştir, bu da dolandırıcılık tespit modellerinin daha kesin ve güvenilir hale gelmesine katkı sağlamıştır.
- Çeşitli performans metrikleri aracılığıyla yapılan kapsamlı deneyler, önerilen gürültü faktörü kodlama yönteminin geleneksel üst örnekleme yöntemlerinden daha etkili olduğunu göstermektedir.
- Yüksek ACC, R, P, F_1, AUC-ROC ve MCC değerleri ve karşılaştırmalar ile literatürde üst örnekleme alanına katkı yapılmıştır.
- Yüksek hassasiyet ve doğruluk sağlama yeteneği, bu yöntemin gerçek dünya uygulamalarında etkili bir şekilde kullanılabilirliğini vurgulamaktadır.
- CIP alanında üç farklı otokodlayıcı varyantını kullanarak yapılan deneyler aracılığıyla bu varyantların etkinlik dereceleri değerlendirilmiştir.
- Farklı otokodlayıcı varyantlarının, veri kümesi özelliklerine bağlı olarak farklı etkililik dereceleri sergileyebileceği vurgulanmıştır.

KAYNAKÇA

Abdallah, A., Maarof, M. A., & Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications*, *68*, 90–113.

Alarfaj, F. K., Malik, I., Khan, H. U., Almusallam, N., Ramzan, M., & Ahmed, M. (2022). Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms. *IEEE Access*, *10*, 39700–39715.

Al-Hashedi, K. G., & Magalingam, P. (2021). Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019. *Computer Science Review*, *40*, 100402. https://doi.org/10.1016/j.cosrev.2021.100402

Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., & Aljaaf, A. J. (2020). *A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science* (pp. 3–21). https://doi.org/10.1007/978-3-030-22475-2_1

Arabzadeh, N., Zarrinkalam, F., Jovanovic, J., Al-Obeidat, F., & Bagheri, E. (2020). Neural embedding-based specificity metrics for pre-retrieval query performance prediction. *Information Processing & Management*, *57*(4), 102248. https://doi.org/10.1016/j.ipm.2020.102248

Awan, F. M., Saleem, Y., Minerva, R., & Crespi, N. (2020). A Comparative Analysis of Machine/Deep Learning Models for Parking Space Availability Prediction. *Sensors*, *20*(1), 322. https://doi.org/10.3390/s20010322

Bahnsen, A. C., Stojanovic, A., Aouada, D., & Ottersten, B. (2013). Cost sensitive credit card fraud detection using Bayes minimum risk. *2013 12th International Conference on Machine Learning and Applications*, *1*, 333–338.

Barto, A. G. (1997). Reinforcement Learning. In *Neural Systems for Control* (pp. 7–30). Elsevier. https://doi.org/10.1016/B978-012526430-3/50003-9

Benson Edwin Raj, S., & Annie Portia, A. (2011). Analysis on credit card fraud detection methods. *2011 International Conference on Computer, Communication and Electrical Technology (ICCCET)*, 152–156. https://doi.org/10.1109/ICCCET.2011.5762457

Bhalekar, P. D., & Shaikh, M. Z. (2019). Machine learning: Survey, types and challenges. *Int. Res. J. Eng. Technol.(IRJET)*, *6*(3), 8131–8136.

Bhatla, T. P., Prabhu, V., & Dua, A. (2003). Understanding credit card frauds. *Cards Business Review*, *1*(6), 1–15.

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems*, *50*(3), 602–613.

Biau, G., & Scornet, E. (2016). A random forest guided tour. *TEST*, *25*(2), 197–227. https://doi.org/10.1007/s11749-016-0481-7

Bielecki, A. (2019). *Foundations of Artificial Neural Networks* (pp. 15–28). https://doi.org/10.1007/978-3-319-90140-4_3

Bin Sulaiman, R., Schetinin, V., & Sant, P. (2022). Review of Machine Learning Approach on Credit Card Fraud Detection. *Human-Centric Intelligent Systems*, *2*(1–2), 55–68. https://doi.org/10.1007/s44230-022-00004-0

Block, H. D. (1962). The Perceptron: A Model for Brain Functioning. I. *Reviews of Modern Physics*, *34*(1), 123–135. https://doi.org/10.1103/RevModPhys.34.123

Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science*, *17*(3), 235–255.

Boughorbel, S., Jarray, F., & El-Anbari, M. (2017). Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLOS ONE*, *12*(6), e0177678. https://doi.org/10.1371/journal.pone.0177678

Brink, H., Richards, J., & Fetherolf, M. (2016). *Real-world machine learning*. Simon and Schuster.

Çakır, M. Y., & Şirin, Y. (2023). Enhanced autoencoder-based fraud detection: a novel approach with noise factor encoding and SMOTE. *Knowledge and Information Systems*. https://doi.org/10.1007/s10115-023-02016-z

Cano, A., Zafra, A., & Ventura, S. (2013). Weighted Data Gravitation Classification for Standard and Imbalanced Data. *IEEE Transactions on Cybernetics*, *43*(6), 1672–1687. https://doi.org/10.1109/TSMCB.2012.2227470

Cao, K., Brbic, M., & Leskovec, J. (2021). Open-world semi-supervised learning. *ArXiv Preprint ArXiv:2102.03526*.

Carbonell, J. G., Michalski, R. S., & Mitchell, T. M. (1983). AN OVERVIEW OF MACHINE LEARNING. In *Machine Learning* (pp. 3–23). Elsevier. https://doi.org/10.1016/B978-0-08-051054-5.50005-4

Carcillo, F., Le Borgne, Y.-A., Caelen, O., & Bontempi, G. (2018). Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *International Journal of Data Science and Analytics*, *5*(4), 285–300.

Carcillo, F., Le Borgne, Y.-A., Caelen, O., Kessaci, Y., Oblé, F., & Bontempi, G. (2021). Combining unsupervised and supervised learning in credit card fraud detection. *Information Sciences*, *557*, 317–331.

Cassella, S. D. (2018). Toward a new model of money laundering. *Journal of Money Laundering Control*, *21*(4), 494–497. https://doi.org/10.1108/JMLC-09-2017-0045

Chawla, N. V, Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chen, Y., & Zhang, R. (2021). Research on Credit Card Default Prediction Based on k-Means SMOTE and BP Neural Network. *Complexity*, *2021*, 1–13. https://doi.org/10.1155/2021/6618841

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, *21*(1), 6. https://doi.org/10.1186/s12864-019-6413-7

Chitimira, H., & Animashaun, O. (2023). The adequacy of the legal framework for combating money laundering and terrorist financing in Nigeria. *Journal of Money Laundering Control*, *26*(7), 110–126. https://doi.org/10.1108/JMLC-12-2022-0171

Corral, G., Armengol, E., Fornells, A., & Golobardes, E. (2007). *Data Security Analysis Using Unsupervised Learning and Explanations* (pp. 112–119). https://doi.org/10.1007/978-3-540-74972-1_16

Cunningham, P., Cord, M., & Delany, S. J. (n.d.). Supervised Learning. In *Machine Learning Techniques for Multimedia* (pp. 21–49). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-75171-7_2

Dablain, D., Krawczyk, B., & Chawla, N. V. (2022). DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*.

Dai, Q., Liu, J., & Shi, Y. (2023). Class-overlap undersampling based on Schur decomposition for Class-imbalance problems. *Expert Systems with Applications*, *221*, 119735. https://doi.org/10.1016/j.eswa.2023.119735

Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. *2015 IEEE Symposium Series on Computational Intelligence*, 159–166.

Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., & Bontempi, G. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Systems with Applications*, *41*(10), 4915–4928. https://doi.org/10.1016/j.eswa.2014.02.026

Dang, N. C., Moreno-García, M. N., & De la Prieta, F. (2020). Sentiment Analysis Based on Deep Learning: A Comparative Study. *Electronics*, *9*(3), 483. https://doi.org/10.3390/electronics9030483

Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*, 233–240. https://doi.org/10.1145/1143844.1143874

de Ville, B. (2013). Decision trees. *WIREs Computational Statistics*, *5*(6), 448–455. https://doi.org/10.1002/wics.1278

Dietterich, T. G. (2000). *Ensemble Methods in Machine Learning* (pp. 1–15). https://doi.org/10.1007/3-540-45014-9_1

Ding, S., Li, H., Su, C., Yu, J., & Jin, F. (2013). Evolutionary artificial neural networks: a review. *Artificial Intelligence Review*, *39*(3), 251–260. https://doi.org/10.1007/s10462-011-9270-6

Ding, Y., Kang, W., Feng, J., Peng, B., & Yang, A. (2023). Credit card fraud detection based on improved Variational Autoencoder Generative Adversarial Network. *IEEE Access*.

Du, H., Lv, L., Guo, A., & Wang, H. (2023). AutoEncoder and LightGBM for Credit Card Fraud Detection Problems. *Symmetry*, *15*(4), 870.

Duda, R. O., Hart, P. E., & others. (1973). *Pattern classification and scene analysis* (Vol. 3). Wiley New York.

Duman, E., & Ozcelik, M. H. (2011). Detecting credit card fraud by genetic algorithm and scatter search. *Expert Systems with Applications*, *38*(10), 13057–13063.

Erickson, B. J., & Kitamura, F. (2021). Magician's Corner: 9. Performance Metrics for Machine Learning Models. *Radiology: Artificial Intelligence*, *3*(3), e200126. https://doi.org/10.1148/ryai.2021200126

Esenogho, E., Mienye, I. D., Swart, T. G., Aruleba, K., & Obaido, G. (2022). A neural network ensemble with feature engineering for Improved Credit Card Fraud Detection. *IEEE Access*, *10*, 16400–16407.

Eshghi, A., & Kargari, M. (2019). Introducing a method for combining supervised and semi-supervised methods in fraud detection. *2019 15th Iran International Industrial Engineering Conference (IIIEC)*, 23–30.

Fanai, H., & Abbasimehr, H. (2023). A novel combined approach based on deep Autoencoder and deep classifiers for credit card fraud detection. *Expert Systems with Applications*, 119562.

Farias, F., Ludermir, T., & Bastos-Filho, C. (2020). Similarity based stratified splitting: an approach to train better classifiers. *ArXiv Preprint ArXiv:2010.06099*.

Fekri, M. N., Patel, H., Grolinger, K., & Sharma, V. (2021). Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network. *Applied Energy*, *282*, 116177. https://doi.org/10.1016/j.apenergy.2020.116177

Feng, S., Keung, J., Yu, X., Xiao, Y., & Zhang, M. (2021). Investigation on the stability of SMOTE-based oversampling techniques in software defect prediction. *Information and Software Technology*, *139*, 106662. https://doi.org/10.1016/j.infsof.2021.106662

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Cost-Sensitive Learning. In *Learning from Imbalanced Data Sets* (pp. 63–78). Springer International Publishing. https://doi.org/10.1007/978-3-319-98074-4_4

Fernando, K. R. M., & Tsokos, C. P. (2021). Dynamically weighted balanced loss: class imbalanced learning and confidence calibration of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(7), 2940–2951.

Fithriasari, K., Hariastuti, I., & Wening, K. S. (2020). Handling imbalance data in classification model with nominal predictors. *(IJCSAM) International Journal of Computing Science and Applied Mathematics*, *6*(1), 33–37.

Fonseca, J., Douzas, G., & Bacao, F. (2021). Improving Imbalanced Land Cover Classification with K-Means SMOTE: Detecting and Oversampling Distinctive Minority Spectral Signatures. *Information*, *12*(7), 266. https://doi.org/10.3390/info12070266

Gao, J., Zhou, Z., Ai, J., Xia, B., & Coggeshall, S. (2019). Predicting credit card transaction fraud using machine learning algorithms. *Journal of Intelligent Learning Systems and Applications*, *11*(3), 33–63.

Ghahramani, Z. (2004). *Unsupervised Learning* (pp. 72–112). https://doi.org/10.1007/978-3-540-28650-9_5

Gillies, M., Fiebrink, R., Tanaka, A., Garcia, J., Bevilacqua, F., Heloir, A., Nunnari, F., Mackay, W., Amershi, S., Lee, B., d'Alessandro, N., Tilmanne, J., Kulesza, T., & Caramiaux, B. (2016). Human-Centred Machine Learning. *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, 3558–3565. https://doi.org/10.1145/2851581.2856492

Goecks, L. S., Korzenowski, A. L., Gonçalves Terra Neto, P., de Souza, D. L., & Mareth, T. (2022). Anti-money laundering and financial fraud detection: A

systematic literature review. *Intelligent Systems in Accounting, Finance and Management*, *29*(2), 71–85. https://doi.org/10.1002/isaf.1509

Gosavi, A. (2004). Reinforcement learning for long-run average cost. *European Journal of Operational Research*, *155*(3), 654–674. https://doi.org/10.1016/S0377-2217(02)00874-3

Gray, D., Bowes, D., Davey, N., Yi Sun, & Christianson, B. (2011). Further thoughts on precision. *15th Annual Conference on Evaluation & Assessment in Software Engineering (EASE 2011)*, 129–133. https://doi.org/10.1049/ic.2011.0016

Gu, Q., Zhu, L., & Cai, Z. (2009). *Evaluation Measures of the Classification Performance of Imbalanced Data Sets* (pp. 461–471). https://doi.org/10.1007/978-3-642-04962-0_53

Guo, X., Yin, Y., Dong, C., Yang, G., & Zhou, G. (2008). On the Class Imbalance Problem. *2008 Fourth International Conference on Natural Computation*, 192–201. https://doi.org/10.1109/ICNC.2008.871

Habibpour, M., Gharoun, H., Mehdipour, M., Tajally, A., Asgharnezhad, H., Shamsi, A., Khosravi, A., & Nahavandi, S. (2023). Uncertainty-aware credit card fraud detection using deep learning. *Engineering Applications of Artificial Intelligence*, *123*, 106248.

Haglin, J. M., Jimenez, G., & Eltorai, A. E. M. (2019). Artificial neural networks in medicine. *Health and Technology*, *9*(1), 1–6. https://doi.org/10.1007/s12553-018-0244-4

Haibo He, Yang Bai, Garcia, E. A., & Shutao Li. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 1322–1328. https://doi.org/10.1109/IJCNN.2008.4633969

Hastie, T., Tibshirani, R., & Friedman, J. (2009a). *Overview of Supervised Learning* (pp. 9–41). https://doi.org/10.1007/978-0-387-84858-7_2

Hastie, T., Tibshirani, R., & Friedman, J. (2009b). *Unsupervised Learning* (pp. 485–585). https://doi.org/10.1007/978-0-387-84858-7_14

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Interceptd. (2022). *Using Machine Learning Fraud Detection*.

Jakobi, A. P. (2018). Governing illicit finance in transnational security spaces: the FATF and anti-money laundering. *Crime, Law and Social Change*, *69*(2), 173–190. https://doi.org/10.1007/s10611-017-9750-y

Jan, B., Farman, H., Khan, M., Imran, M., Islam, I. U., Ahmad, A., Ali, S., & Jeon, G. (2019). Deep learning in big data Analytics: A comparative study. *Computers & Electrical Engineering*, *75*, 275–287. https://doi.org/10.1016/j.compeleceng.2017.12.009

Jayantilal, S., Jorge, S. F., & Ferreira, A. (2017). Portuguese Anti-money Laundering Policy: a Game Theory Approach. *European Journal on Criminal Policy and Research*, *23*(4), 559–574. https://doi.org/10.1007/s10610-017-9347-0

Jiang, T., Gradus, J. L., & Rosellini, A. J. (2020). Supervised Machine Learning: A Brief Primer. *Behavior Therapy*, *51*(5), 675–687. https://doi.org/10.1016/j.beth.2020.05.002

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, *4*, 237–285. https://doi.org/10.1613/jair.301

Kalaycıoğlu, O., Akhanlı, S. E., Menteşe, E. Y., Kalaycıoğlu, M., & Kalaycıoğlu, S. (2023). Using machine learning algorithms to identify predictors of social vulnerability in the event of a hazard: Istanbul case study. Natural Hazards and Earth System Sciences, 23(6), 2133–2156. https://doi.org/10.5194/nhess-23-2133-2023

Khan, M. Y., Qayoom, A., Nizami, M. S., Siddiqui, M. S., Wasi, S., & Raazi, S. M. K.-R. (2021). Automated Prediction of Good Dictionary EXamples (GDEX): A Comprehensive Experiment with Distant Supervision, Machine Learning, and Word Embedding-Based Deep Learning Techniques. *Complexity*, *2021*, 1–18. https://doi.org/10.1155/2021/2553199

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *ArXiv Preprint ArXiv:1312.6114*.

Kingsford, C., & Salzberg, S. L. (2008). What are decision trees? *Nature Biotechnology*, *26*(9), 1011–1013. https://doi.org/10.1038/nbt0908-1011

Korejo, M. S., Rajamanickam, R., & Md. Said, M. H. (2021). The concept of money laundering: a quest for legal definition. *Journal of Money Laundering Control*, *24*(4), 725–736. https://doi.org/10.1108/JMLC-05-2020-0045

Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review*, *39*(4), 261–283. https://doi.org/10.1007/s10462-011-9272-4

Kumar, K., & Thakur, G. S. M. (2012). Advanced Applications of Neural Networks and Artificial Intelligence: A Review. *International Journal of Information Technology and Computer Science*, *4*(6), 57–68. https://doi.org/10.5815/ijitcs.2012.06.08

Laakom, F., Raitoharju, J., Iosifidis, A., & Gabbouj, M. (2022). Reducing redundancy in the bottleneck representation of the autoencoders. *ArXiv Preprint ArXiv:2202.04629*.

Learning, S.-S. (2006). Semi-Supervised Learning. *CSZ2006. Html*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Leevy, J. L., Johnson, J. M., Hancock, J., & Khoshgoftaar, T. M. (2023). Threshold optimization and random undersampling for imbalanced credit card data. *Journal of Big Data*, *10*(1), 58. https://doi.org/10.1186/s40537-023-00738-z

Li, Y. (2017). Deep reinforcement learning: An overview. *ArXiv Preprint ArXiv:1701.07274*.

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2022). A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Lin, T.-H., & Jiang, J.-R. (2021). Credit card fraud detection with autoencoder and probabilistic random forest. *Mathematics*, *9*(21), 2683.

Lin, W.-C., Tsai, C.-F., Hu, Y.-H., & Jhang, J.-S. (2017). Clustering-based undersampling in class-imbalanced data. *Information Sciences*, *409–410*, 17–26. https://doi.org/10.1016/j.ins.2017.05.008

Liu, A., Ghosh, J., & Martin, C. (2007). Generative Oversampling for Mining Imbalanced Datasets. *DMIN*, *7*, 66–72.

Liu, B. (2011). Supervised Learning. In *Web Data Mining* (pp. 63–132). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-19460-3_3

Liu, S. M., Chen, J.-H., & Liu, Z. (2023). An empirical study of dynamic selection and random under-sampling for the class imbalance problem. Expert Systems with Applications, 221, 119703. https://doi.org/10.1016/j.eswa.2023.119703

Liu, Y., Wang, Y., & Zhang, J. (2012). *New Machine Learning Algorithm: Random Forest* (pp. 246–252). https://doi.org/10.1007/978-3-642-34062-8_32

Liu, Y., Zhou, Y., Wen, S., & Tang, C. (2014). A Strategy on Selecting Performance Metrics for Classifier Evaluation. *International Journal of Mobile Computing and Multimedia Communications*, *6*(4), 20–35. https://doi.org/10.4018/IJMCMC.2014100102

Luque, A., Carrasco, A., Martín, A., & de las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, *91*, 216–231. https://doi.org/10.1016/j.patcog.2019.02.023

Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002). Credit card fraud detection using Bayesian and neural networks. *Proceedings of the 1st International Naiso Congress on Neuro Fuzzy Technologies*, *261*, 270.

Magee, J. F. (1964). *Decision trees for decision making*. Harvard Business Review Brighton, MA, USA.

Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, *9*(1), 381–386.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://doi.org/10.1007/BF02478259

Mehrotra, K., Mohan, C. K., & Ranka, S. (1997). *Elements of artificial neural networks*. MIT press.

Michie, D., Spiegelhalter, D. J., & Taylor, C. C. (1994). *Machine learning, neural and statistical classification*.

Mishra, V., & Rath, S. K. (2021). Detection of breast cancer tumours based on feature reduction and classification of thermograms. *Quantitative InfraRed Thermography Journal*, *18*(5), 300–313. https://doi.org/10.1080/17686733.2020.1768497

Misra, S., Thakur, S., Ghosh, M., & Saha, S. K. (2020). An autoencoder based model for detecting fraudulent credit card transaction. *Procedia Computer Science*, *167*, 254–262.

Mohammed, R., Rawashdeh, J., & Abdullah, M. (2020). Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results. *2020 11th International Conference on Information and Communication Systems (ICICS)*, 243–248. https://doi.org/10.1109/ICICS49469.2020.239556

Moustakis, V. S., & Herrmann, J. (1997). Where do machine learning and human-computer interaction meet? *Applied Artificial Intelligence*, *11*(7–8), 595–609. https://doi.org/10.1080/088395197117948

Mqadi, N. M., Naicker, N., & Adeliyi, T. (2021). Solving Misclassification of the Credit Card Imbalance Problem Using Near Miss. *Mathematical Problems in Engineering*, *2021*, 1–16. https://doi.org/10.1155/2021/7194728

Nasteski, V. (2017). An overview of the supervised machine learning methods. *HORIZONS.B*, *4*, 51–62. https://doi.org/10.20544/HORIZONS.B.04.1.17.P05

Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, *7*. https://doi.org/10.3389/fnbot.2013.00021

Nations, U. (2022). *Money Laundering*.

Nilsson, N. J. (1965). *Learning machines.*

Ogbeide, H., Thomson, M. E., Gonul, M. S., Pollock, A. C., Bhowmick, S., & Bello, A. U. (2023). The anti-money laundering risk assessment: A probabilistic approach. *Journal of Business Research*, *162*, 113820. https://doi.org/10.1016/j.jbusres.2023.113820

Oliveira, G. P. de, Fonseca, A., & Rodrigues, P. C. (2022). Diabetes diagnosis based on hard and soft voting classifiers combining statistical learning models. *Brazilian Journal of Biometrics, 40*(4), 415–427. https://doi.org/10.28951/bjb.v40i4.605

Omar, S., Ngadi, A., & Jebur, H. H. (2013). Machine learning techniques for anomaly detection: an overview. *International Journal of Computer Applications*, *79*(2).

Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012). *How Many Trees in a Random Forest?* (pp. 154–168). https://doi.org/10.1007/978-3-642-31537-4_13

Ouali, Y., Hudelot, C., & Tami, M. (2020). An overview of deep semi-supervised learning. *ArXiv Preprint ArXiv:2006.05278*.

Özçift, A. (2011). Random forests ensemble classifier trained with data resampling strategy to improve cardiac arrhythmia diagnosis. *Computers in Biology and Medicine*, *41*(5), 265–271. https://doi.org/10.1016/j.compbiomed.2011.03.001

Patel, H., Singh Rajput, D., Thippa Reddy, G., Iwendi, C., Kashif Bashir, A., & Jo, O. (2020). A review on classification of imbalanced data for wireless sensor networks. *International Journal of Distributed Sensor Networks*, *16*(4), 1550147720916404.

Peng, J., Jury, E. C., Dönnes, P., & Ciurtin, C. (2021). Machine Learning Techniques for Personalised Medicine Approaches in Immune-Mediated Chronic Inflammatory Diseases: Applications and Challenges. *Frontiers in Pharmacology*, *12*. https://doi.org/10.3389/fphar.2021.720694

Pise, N. N., & Kulkarni, P. (2008). A Survey of Semi-Supervised Learning Methods. *2008 International Conference on Computational Intelligence and Security*, 30–34. https://doi.org/10.1109/CIS.2008.204

Provost, F., & Fawcett, T. (2001). Robust classification for imprecise environments. *Machine Learning*, *42*(3), 203–231.

Qi, Y. (2012). Random Forest for Bioinformatics. In *Ensemble Machine Learning* (pp. 307–323). Springer New York. https://doi.org/10.1007/978-1-4419-9326-7_11

Rafrastara, F. A., Supriyanto, C., Paramita, C., Astuti, Y. P., & Ahmed, F. (2023). Performance Improvement of Random Forest Algorithm for Malware Detection on Imbalanced Dataset using Random Under-Sampling Method. *Jurnal Informatika: Jurnal Pengembangan IT*, *8*(2), 113–118.

Randhawa, K., Loo, C. K., Seera, M., Lim, C. P., & Nandi, A. K. (2018). Credit card fraud detection using AdaBoost and majority voting. *IEEE Access*, *6*, 14277–14284.

Rathore, S. S., Chouhan, S. S., Jain, D. K., & Vachhani, A. G. (2022). Generative Oversampling Methods for Handling Imbalanced Data in Software Fault Prediction. *IEEE Transactions on Reliability*, *71*(2), 747–762.

Richhariya, P., & Singh, P. K. (2014). Evaluating and emerging payment card fraud challenges and resolution. *International Journal of Computer Applications*, *107*(14).

Rifai, S., Mesnil, G., Vincent, P., Muller, X., Bengio, Y., Dauphin, Y., & Glorot, X. (2011). Higher order contractive auto-encoder. *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part II 22*, 645–660.

Rigatti, S. J. (2017). Random Forest. *Journal of Insurance Medicine*, *47*(1), 31–39. https://doi.org/10.17849/insm-47-01-31-39.1

Rong, S., & Bao-wen, Z. (2018). The research of regression model in machine learning field. *MATEC Web of Conferences*, *176*, 01033. https://doi.org/10.1051/matecconf/201817601033

Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., & Beling, P. (2018). Deep learning detecting fraud in credit card transactions. *2018 Systems and Information Engineering Design Symposium (SIEDS)*, 129–134.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Ruta, D., & Gabrys, B. (2005). Classifier selection for majority voting. *Information Fusion*, *6*(1), 63–81. https://doi.org/10.1016/j.inffus.2004.04.008

Sailusha, R., Gnaneswar, V., Ramesh, R., & Rao, G. R. (2020). Credit card fraud detection using machine learning. *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 1264–1270.

Salazar, A., Safont, G., & Vergara, L. (2018). Semi-supervised learning for imbalanced classification of credit card transaction. *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–7.

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*(3), 210–229. https://doi.org/10.1147/rd.33.0210

Sánchez, D., Vila, M. A., Cerda, L., & Serrano, J.-M. (2009). Association rules applied to credit card fraud detection. *Expert Systems with Applications*, *36*(2), 3630–3640.

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Sarker, I. H. (2022). AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems. *SN Computer Science*, *3*(2), 158. https://doi.org/10.1007/s42979-022-01043-x

Schapire, R. E. (2013). Explaining AdaBoost. In *Empirical Inference* (pp. 37–52). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-41136-6_5

Schonlau, M., & Zou, R. Y. (2020). The random forest algorithm for statistical learning. *The Stata Journal: Promoting Communications on Statistics and Stata*, *20*(1), 3–29. https://doi.org/10.1177/1536867X20909688

Sen, P. C., Hajra, M., & Ghosh, M. (2020). *Supervised Classification Algorithms in Machine Learning: A Survey and Review* (pp. 99–111). https://doi.org/10.1007/978-981-13-7403-6_11

Sinayobye, O., Musabe, R., Uwitonze, A., & Ngenzi, A. (2023). A Credit Card Fraud Detection Model Using Machine Learning Methods with a Hybrid of Undersampling and Oversampling for Handling Imbalanced Datasets for High Scores. *Applied Machine Learning and Data Analytics: 5th International*

Conference, AMLDA 2022, Reynosa, Tamaulipas, Mexico, December 22–23, 2022, Revised Selected Papers, 142–155.

Sofaer, H. R., Hoeting, J. A., & Jarnevich, C. S. (2019). The area under the precision-recall curve as a performance metric for rare binary events. *Methods in Ecology and Evolution*, *10*(4), 565–577. https://doi.org/10.1111/2041-210X.13140

Soltanzadeh, P., Feizi-Derakhshi, M. R., & Hashemzadeh, M. (2023). Addressing the class-imbalance and class-overlap problems by a metaheuristic-based under-sampling approach. *Pattern Recognition*, *143*, 109721. https://doi.org/10.1016/j.patcog.2023.109721

Srivastava, A., Kundu, A., Sural, S., & Majumdar, A. K. (2008). Credit Card Fraud Detection Using Hidden Markov Model. *IEEE Transactions on Dependable and Secure Computing*, *5*(1), 37–48. https://doi.org/10.1109/TDSC.2007.70228

Strelcenia, E., & Prakoonwit, S. (2023a). A Survey on GAN Techniques for Data Augmentation to Address the Imbalanced Data Issues in Credit Card Fraud Detection. *Machine Learning and Knowledge Extraction*, *5*(1), 304–329.

Strelcenia, E., & Prakoonwit, S. (2023b). Improving Classification Performance in Credit Card Fraud Detection by Using New Data Augmentation. *AI*, *4*(1), 172–198.

Sullivan, K. (2015). What Is Money Laundering? In *Anti–Money Laundering in a Nutshell* (pp. 1–13). Apress. https://doi.org/10.1007/978-1-4302-6161-2_1

Takiddin, A., Ismail, M., Zafar, U., & Serpedin, E. (2022). Deep autoencoder-based anomaly detection of electricity theft cyberattacks in smart grids. *IEEE Systems Journal*, *16*(3), 4106–4117.

Teichmann, F. M. (2019). Recent trends in money laundering and terrorism financing. *Journal of Financial Regulation and Compliance*, *27*(1), 2–12. https://doi.org/10.1108/JFRC-03-2018-0042

Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., & Kuruwitaarachchi, N. (2019). Real-time credit card fraud detection using machine learning. *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 488–493.

Tingfei, H., Guangquan, C., & Kuihua, H. (2020). Using variational auto encoding in credit card fraud detection. *IEEE Access*, *8*, 149841–149853.

Usama, M., Qadir, J., Raza, A., Arif, H., Yau, K. A., Elkhatib, Y., Hussain, A., & Al-Fuqaha, A. (2019). Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges. *IEEE Access*, *7*, 65579–65615. https://doi.org/10.1109/ACCESS.2019.2916648

Wang, C., Wang, J., Li, C., Ho, D., Cheng, J., Yan, T., Meng, L., & Meng, M. Q.-H. (2019). Safe and Robust Mobile Robot Navigation in Uneven Indoor Environments. *Sensors*, *19*(13), 2993. https://doi.org/10.3390/s19132993

Wang, J.-B., Zou, C.-A., & Fu, G.-H. (2021). AWSMOTE: An SVM-Based Adaptive Weighted SMOTE for Class-Imbalance Learning. *Scientific Programming*, *2021*, 1–18. https://doi.org/10.1155/2021/9947621

Wang, Q., Luo, Z., Huang, J., Feng, Y., & Liu, Z. (2017). A Novel Ensemble Method for Imbalanced Data Learning: Bagging of Extrapolation-SMOTE SVM. *Computational Intelligence and Neuroscience*, *2017*, 1–11. https://doi.org/10.1155/2017/1827016

Wang, W., Huang, Y., Wang, Y., & Wang, L. (2014). Generalized autoencoder: A neural network framework for dimensionality reduction. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 490–497.

Wang, Z., Zhang, J., & Verma, N. (2015). Realizing Low-Energy Classification Systems by Implementing Matrix Multiplication Directly Within an ADC. *IEEE Transactions on Biomedical Circuits and Systems*, 1–1. https://doi.org/10.1109/TBCAS.2015.2500101

Wei, J., Chu, X., Sun, X., Xu, K., Deng, H., Chen, J., Wei, Z., & Lei, M. (2019). Machine learning in materials science. *InfoMat*, *1*(3), 338–358. https://doi.org/10.1002/inf2.12028

Wei, Y.-C., Lai, Y.-X., & Wu, M.-E. (2023). An evaluation of deep learning models for chargeback Fraud detection in online games. *Cluster Computing*, *26*(2), 927–943.

Weston, D. J., Hand, D. J., Adams, N. M., Whitrow, C., & Juszczak, P. (2008). Plastic card fraud detection using peer group analysis. *Advances in Data Analysis and Classification*, *2*(1), 45–62.

Wiering, M., & van Otterlo, M. (Eds.). (2012). *Reinforcement Learning* (Vol. 12). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-27645-3

Wongvorachan, T., He, S., & Bulut, O. (2023). A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining. *Information*, *14*(1), 54.

Wu, G., & Chang, E. Y. (2005). KBA: kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 786–795. https://doi.org/10.1109/TKDE.2005.95

Xuan, S., Liu, G., Li, Z., Zheng, L., Wang, S., & Jiang, C. (2018). Random forest for credit card fraud detection. *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 1–6.

Yadav, D., & Pal, S. (2020). Prediction of Heart Disease Using Feature Selection and Random Forest Ensemble Method. *International Journal of Pharmaceutical Research*, *12*(04). https://doi.org/10.31838/ijpr/2020.12.04.013

Yang, G., Liu, X., & Li, B. (2023). Anti-money laundering supervision by intelligent algorithm. *Computers & Security*, *132*, 103344. https://doi.org/10.1016/j.cose.2023.103344

Yaqoob, A., Musheer Aziz, R., & verma, N. K. (2023). Applications and Techniques of Machine Learning in Cancer Classification: A Systematic Review. *Human-Centric Intelligent Systems*. https://doi.org/10.1007/s44230-023-00041-3

Yee, O. S., Sagadevan, S., & Malim, N. (2018). Credit card fraud detection using machine learning as data mining technique. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *10*(1–4), 23–27.

Yi, H., Jiang, Q., Yan, X., & Wang, B. (2020). Imbalanced classification based on minority clustering synthetic minority oversampling technique with wind turbine fault detection application. *IEEE Transactions on Industrial Informatics*, *17*(9), 5867–5875.

Zaabi, K. Al, & Tubaishat, A. (2015). Security Awarness Program for Customers Using Online Banking. *GSTF Journal on Computing (JoC)*, *4*(3), 19. https://doi.org/10.7603/s40601-014-0019-3

Zakariah, M., AlQahtani, S. A., & Al-Rakhami, M. S. (2023). Machine Learning-Based Adaptive Synthetic Sampling Technique for Intrusion Detection. *Applied Sciences*, *13*(11), 6504.

Zanoni, M., Arcelli Fontana, F., & Stella, F. (2015). On applying machine learning techniques for design pattern detection. *Journal of Systems and Software*, *103*, 102–117. https://doi.org/10.1016/j.jss.2015.01.037

Zhang, T., Lin, W., Vogelmann, A. M., Zhang, M., Xie, S., Qin, Y., & Golaz, J. (2021). Improving Convection Trigger Functions in Deep Convective Parameterization Schemes Using Machine Learning. *Journal of Advances in Modeling Earth Systems*, *13*(5). https://doi.org/10.1029/2020MS002365

Zhu, H. (2020). Big Data and Artificial Intelligence Modeling for Drug Discovery. *Annual Review of Pharmacology and Toxicology*, *60*(1), 573–589. https://doi.org/10.1146/annurev-pharmtox-010919-023324

Zhu, T., Lin, Y., & Liu, Y. (2017). Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, *72*, 327–340. https://doi.org/10.1016/j.patcog.2017.07.024

Zioviris, G., Kolomvatsos, K., & Stamoulis, G. (2022). Credit card fraud detection using a deep learning multistage model. *The Journal of Supercomputing*, 1–26.

Zou, J., Han, Y., & So, S.-S. (2008). *Overview of Artificial Neural Networks* (pp. 14–22). https://doi.org/10.1007/978-1-60327-101-1_2

Zou, J., Zhang, J., & Jiang, P. (2019). Credit card fraud detection using autoencoder neural network. *ArXiv Preprint ArXiv:1908.11553*.

ÖZGEÇMİŞ

Adı-Soyadı **:** Mert Yılmaz ÇAKIR

Uyruğu **:**

Doğum Tarihi ve Yeri **:**

ÖĞRENİM DURUMU

Lisans: 2015, İstanbul Sabahattin Zaim Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar Mühendisliği

Yüksek Lisans: 2018, İstanbul Sabahattin Zaim Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Bilimi ve Mühendisliği

İŞ DENEYİMİ

Yıl	**Yer**	**Görev**
2016	Erişim Sağlayıcıları Birliği	Software Developer
2015	İZÜ Bilgi İşlem	Network & System
2014	Matriks Bilgi Dağıtım Hizmetleri	ETL Developer

YAYINLAR

Çakır, M. Y., & Şirin, Y. (2023). Enhanced autoencoder-based fraud detection: a novel approach with noise factor encoding and SMOTE. *Knowledge and Information Systems*. https://doi.org/10.1007/s10115-023-02016-z

Çakır, M. Y., & Şirin, Y. (2018). Comparison of windowing techniques for speech recognition system. *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 1–4, IEEE.

Çakır, M. Y., & Şirin, Y. (2018). Speaker independent Turkish speech recognition optimization with energy derivates on feature vectors. *2018 26th Signal Processing and Communications Applications Conference (SIU)*, 1–4, IEEE.

Çakır, M. Y. (2017). *Gerçek zamanlı yüksek kalitede ses tanıma*. İstanbul Sabahattin Zaim Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı.

Çakır, M. Y., Kutlugün, M. A., & Kiani, F. (2017). Derin sinir ağları ile konuşma tespiti ve cinsiyet tahmini. *22.Türkiye'de İnternet Konferansı.* Beşiktaş, İstanbul: Bahçeşehir Üniversitesi.

Çakır, M. Y., Kutlugün, M. A., & Kiani, F. (2017). Yapay sinir ağları ve K-En yakın komşu algoritmalarının birlikte çalşma tekniği (Ensemble) ile metin türü tanıma. *22. Türkiye'de İnternet Konferansı*. Beşiktaş, İstanbul: Bahçeşehir Üniversitesi.