T.C.
YALOVA ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

BLOK ZİNCİR AĞLAR ÜZERİNDE YASAL OLMAYAN PARA TRANSFERLERİNİN GRAF TEORİSİ VE YAPAY ZEKÂ YÖNTEMLERİ İLE TESPİT EDİLMESİ

DOKTORA TEZİ

EMİNE CENGİZ

DANIŞMAN: PROF. DR. MURAT GÖK

YALOVA
EKİM 2025

T.C.
YALOVA ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

BLOK ZİNCİR AĞLAR ÜZERİNDE YASAL OLMAYAN PARA TRANSFERLERİNİN GRAF TEORİSİ VE YAPAY ZEKÂ YÖNTEMLERİ İLE TESPİT EDİLMESİ

DOKTORA TEZİ

EMİNE CENGİZ
205305003

DANIŞMAN: PROF. DR. MURAT GÖK

YALOVA
EKİM 2025

ETİK BEYAN

Yalova Üniversitesi Lisansüstü Eğitim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım "Blok Zincir Ağlar Üzerinde Yasal Olmayan Para Transferlerinin Graf Teorisi ve Yapay Zekâ Yöntemleri ile Tespit Edilmesi" başlıklı bu tez çalışmasında; tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi, tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu, tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi, kullanılan verilerde herhangi bir değişiklik yapmadığımı, bu tezde sunduğum çalışmanın özgün olduğunu bildirir, aksinin tespiti halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi taahhüt ve beyan ederim.

Emine CENGİZ

ÖNSÖZ

Doktora eğitimim boyunca benden yardımlarını esirgemeyen, akademik anlamda bana her zaman destek olan ve doğru yolu gösteren sayın danışman hocam Prof. Dr. Murat GÖK'e teşekkür ederim.

Her zaman maddi ve manevi desteklerini esirgemeyen, beni hiçbir zaman yalnız bırakmayan, hayattaki en büyük gurur kaynağım olan aileme; sevgileri, özverili destekleri, anlayışları ve sürekli teşvikleri için teşekkür ederim.

Ekim – 2025 Emine CENGİZ

İÇİNDEKİLER

KISALTMALAR LİSTESİ

AM	: Dikkat Mekanizması (Attention Mechanism)
AML	: Kara Para Aklamayı Önleme (Anti Money Laundering)
BiLSTM	: Çift Yönlü Uzun Kısa Vadeli Bellek(Bidirectional Long Short Term Memory)
CNN	: Evrişimsel Sinir Ağları (Convolutional Neural Network)
DL	: Derin Öğrenme (Deep Learning)
FN	: Yanlış Negatif (False Negative)
FP	: Yanlış Pozitif (False Positive)
GAT	: Graf Dikkat Ağı (Graph Attention Network)
GCN	: Graf Evrişimli Ağ (Graph Convolutional Network)
GIN	: Graf İzomorfizm Ağı (Graph Isomorphism Network)
GNN	: Graf Sinir Ağı (Graph Neural Network)
k-NN	: k-En Yakın Komşu (k-Nearest Neighbor)
LEs	: Lyapunov Üstelleri (Lyapunov Exponents)
LightGBM	: Light Gradyan Artırma Makinesi (Light Gradient Boosting Machine)
LR	: Lojistik Regresyon (Logistic Regression)
LSTM	: Uzun Kısa Süreli Bellek (Long Short Term Memory)
MGU	: Minimal Kapılı Birim (Minimal Gated Unit)
ML	: Makine Öğrenmesi (Machine Learning)
MLP	: Çok Katmanlı Algılayıcı (Multilayer Perceptron)
NB	: Naive Bayes
RF	: Rastgele Orman (Random Forest)
RNN	: Yinelenen Sinir Ağları (Recurrent Neural Networks)
SMOTE	: Sentetik Azınlık Aşırı Örnekleme Tekniği (Synthetic Minority Oversampling Technique)
SVM	: Destek Vektör Makineleri (Support Vector Machine)
TN	: Doğru Negatif (True Negative)
TP	: Doğru Pozitif (True Positive)
XGBoost	: Aşırı Gradyan Artırma (Extreme Gradient Boosting)

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

BLOK ZİNCİR AĞLAR ÜZERİNDE YASAL OLMAYAN PARA TRANSFERLERİNİN GRAF TEORİSİ VE YAPAY ZEKÂ YÖNTEMLERİ İLE TESPİT EDİLMESİ

ÖZET

Blok zinciri tabanlı finansal sistemlerde kara para aklama, kripto paraların anonimlik ve merkeziyetsizlik özellikleri nedeniyle ekonomik istikrar için ciddi bir tehdit oluşturmaktadır. Geleneksel kural tabanlı tespit yöntemleri, yüksek yanlış pozitif oranları üretmekte ve değişen kara para aklama stratejilerine uyum sağlamakta yetersiz kalmaktadır. Bu tez, blok zinciri işlem ağlarında kara para aklama faaliyetlerini tespit etmek amacıyla, kaotik zaman serisi analizi ile makine öğrenmesi ve derin öğrenme yaklaşımlarını kullanarak yenilikçi bir model önermektedir.

Öncelikle, blok zinciri işlem graflarından elde edilen yapısal özniteliklerin sayısı artırılmış ve kaotik zaman serilerine dönüştürülmüştür. Bu serilerin dinamik davranışı Lyapunov Üstelleri hesaplanarak tanımlanmış, elde edilen kaotik öznitelikler yasa dışı işlemleri temsil eden ayırt edici öznitelikler olarak kullanılmıştır. Bu öznitelikler iki modelde değerlendirilmiştir: Birincisi doğrudan makine öğrenmesi algoritmaları ile sınıflandırma, ikincisi yeniden oluşturulan işlem graflarının Graf Evrişimli Ağlar ile sınıflandırılması. Ek olarak, Evrişimsel Sinir Ağları, Çift Yönlü Uzun Kısa Süreli Bellek ve Dikkat Mekanizması birleşiminden oluşan hibrit bir derin öğrenme modeli önerilmiştir.

Elliptic veri seti üzerinde gerçekleştirilen modeller, önerilen kaotik öznitelik tabanlı yaklaşımın, geleneksel makine öğrenmesi ve derin öğrnme yöntemlerine kıyasla tespit doğruluğunu artırdığını ve yanlış pozitif oranlarını azalttığını göstermektedir. Bulgular, kaos teorisinin blok zinciri işlem analiziyle birleştirilmesinin, merkeziyetsiz finans ortamlarında kara para aklama tespit sistemlerinin etkinliğini artırmada güçlü bir potansiyel sunduğunu ortaya koymaktadır.

Anahtar Kelimeler: Kara para, Kaotik öznitelik, Lyapunov üstel, Sınıflandırma

DETECTION OF ILLEGAL MONEY TRANSFERS ON BLOCKCHAIN NETWORKS WITH GRAPH THEORY AND ARTIFICIAL INTELLIGENCE METHODS

ABSTRACT

Money laundering in blockchain based financial systems poses a significant threat to economic stability because of the anonymity and decentralization of cryptocurrencies. Traditional rule based detection methods generate high false positive rates and struggle to adapt to evolving money laundering strategies. This thesis proposes an innovative model that combines chaotic time series analysis with machine learning and deep learning approaches to detect money laundering activities in blockchain transaction networks.

First, the number of structural features extracted from blockchain transaction graphs was increased and transformed into a chaotic time series. The dynamic behavior of these series was characterized by calculating the Lyapunov exponents, and the resulting chaotic features were employed as distinctive indicators of illicit transactions. These features were evaluated through two models. The first model applies machine learning algorithms directly for classification, while the second performs classification using Graph Convolutional Networks on the reconstructed transaction graphs. In addition, a hybrid deep learning model was proposed, combining Convolutional Neural Networks, Bidirectional Long Short Term Memory, and an Attention Mechanism.

Experiments conducted on the Elliptic dataset demonstrated that the proposed chaos based feature extraction approach improved the detection accuracy and reduced the false positive rates compared with conventional machine learning and deep learning methods. The findings highlight the strong potential of integrating chaos theory with blockchain transaction analysis to enhance the effectiveness of anti money laundering systems in decentralized finance environments.

Keywords: Money laundering, Chaotic features, Lyapunov exponent, Classification

1. GİRİŞ

Küresel ekonomik sistemin güvenliğini tehdit eden başlıca unsurlardan biri olan kara para aklama, yasa dışı yollarla elde edilen gelirlerin meşru kaynaklardan gelmiş gibi gösterilmesi sürecidir (Korejo ve diğ., 2021). Bu faaliyetler yalnızca suç gelirlerinin izini gizlemekle kalmamakta, aynı zamanda finansal sistemlerin güvenilirliğini sarsmakta, ekonomik istikrarı bozmakta ve kamu gelirlerinde ciddi kayıplara yol açmaktadır (Marasi ve Ferretti, 2024). Bu nedenle, kara para aklamayla mücadele ulusal ve uluslararası düzeyde öncelikli gündem maddelerinden biri haline gelmiştir. Son yıllarda, finansal kuruluşlar ve devlet kurumları kara para aklamayı önlemek için daha katı düzenlemeler ve denetim mekanizmaları getirmiştir. Özellikle bankalar, sigorta şirketleri ve ödeme hizmeti sağlayıcıları, müşteri kimlik doğrulama ve şüpheli işlem bildirimi gibi önlemlerle kara para aklama faaliyetlerini engellemeye çalışmaktadır. Ayrıca, finansal kurumlar arasındaki uluslararası iş birliğinin artırılması, yasadışı finansal işlemlerin tespit edilmesini kolaylaştırmaktadır.

Kara para aklama ile mücadelede teknolojik gelişmelerin rolü de büyük önem taşımaktadır. Özellikle son yıllarda ortaya çıkan ve hızla yaygınlaşan blok zinciri teknolojisi, finansal işlemlerin şeffaflığı ve izlenebilirliği açısından önemli fırsatlar sunmaktadır. Blok zinciri, dijital bilgilerin dağıtılmış ve değiştirilemez bir defterde saklandığı bir teknolojidir (Bacon ve Tarr, 2024). Bu teknoloji, verilerin güvenli ve şeffaf bir şekilde kaydedilmesini sağlar. Her bir bilgi bloğu, önceki blokla kriptografik olarak bağlanarak zincir oluşturur. Bu durum verilerin değiştirilmesini veya silinmesini neredeyse imkânsız hale getirir. Blok zinciri, merkezi bir otoriteye ihtiyaç duymadan işlem yapmayı mümkün kıldığı için özellikle kripto para birimlerinde yaygın olarak kullanılır. Ancak, blok zinciri ağlarındaki anonimlik ve merkeziyetsizlik özellikleri, yasa dışı faaliyetler için de bir zemin oluşturabilir (Zhang ve diğ., 2020). Bu bağlamda, blok zinciri ağlarındaki kara para aklama, dijital para birimlerinin yasa dışı gelirlerin kaynağını gizleme sürecidir. Suçlular, elde ettikleri yasadışı gelirleri farklı dijital cüzdanlar ve karmaşık işlem zincirleri üzerinden aktararak bu fonları yasal kaynaklardan gelmiş gibi gösterebilirler. Kısaca blok zinciri teknolojisi, kullanıcı kimliklerinin anonim kalabilmesi nedeniyle suçlular tarafından kötüye kullanılabilir. Bu nedenle, blok zinciri ağlarındaki kara para aklama faaliyetlerini tespit etmek ve önlemek için veri analiz yöntemlerinin geliştirilmesi kritik öneme sahiptir (Kotagiri, 2024).

Kara para aklamayı tespit etmek için kullanılan en temel yöntemlerden biri, kural tabanlı sistemlerdir. Bu sistemler, belirli olayların gerçekleşip gerçekleşmediğini veya belirli eşik

değerlerin aşılıp aşılmadığını kontrol eden koşullardan oluşur. Ancak, bu yaklaşımın bazı dezavantajları vardır. En önemli dezavantajlarından biri, yüksek oranlarda yanlış pozitif sonuçlar üretmesi ve bu kuralların uzmanlar tarafından manuel olarak oluşturulmasıdır. (Jullum ve diğ., 2020). Bu sorunlar, kural tabanlı sistemlerin değişen ve karmaşık kara para aklama işlemlerine yeterince yanıt verememesine neden olur. Ayrıca, kural tabanlı sistemler yeni ve gelişen kara para aklama tekniklerine karşı esneklik gösterememektedir. Suçlular, mevcut kuralların etrafından dolaşmak için sürekli olarak yeni yöntemler geliştirmektedir. Bu nedenle, kural tabanlı sistemlerin tek başına yeterli olmadığı ve daha karmaşık veri analizi yöntemlerine ihtiyaç duyulduğu görülmektedir.

Makine Öğrenmesi (Machine Learning, ML) yöntemleri geçmiş verilerden karmaşık modeller çıkararak kural tabanlı sistemlerin zorluklarını aşar ve yüksek yanlış pozitif oranlarını azaltabilir (Lorenz ve diğ., 2020). ML (Weber ve diğ., 2019; Bakry ve diğ., 2024; Lokanan, 2024) ve graf analiz yöntemleri (Chai ve diğ., 2022; Zhong ve diğ., 2022; Lo ve diğ., 2023; Li ve diğ., 2024; Liu ve diğ., 2024) kara para aklamayı tespit etmek için önemli araçlar olarak öne çıkmaktadır. Bu yöntemler, büyük veri kümelerindeki karmaşık desenleri ve anormallikleri belirleyerek, kara para aklama faaliyetlerinin tespitinde bir araç olarak kullanılabilir. Ancak, mevcut ML ve graf tabanlı yaklaşımlar genellikle sistemin dinamik davranışlarını tam olarak modelleyememekte ve kara para aklamanın yapısını yeterince yansıtamamaktadır. Bu çalışma, söz konusu eksiklikleri gidermek amacıyla kara para aklama işlemlerini kaotik zaman serileri ve graf temelli analiz yöntemleriyle inceleyen yenilikçi bir yaklaşım sunmaktadır.

Bu çalışmanın temel amacı**,** blok zinciri üzerindeki işlem özniteliklerini değerlendirerek, kara para aklama faaliyetlerinin tespitinde daha yüksek performans sağlayan bir sınıflandırma modeli geliştirmektir. Bu çalışmanın başlıca katkıları aşağıdaki şekilde özetlenmiştir:

1. Kara para aklama tespitinde graf yapıların kaotik zaman serileri olarak analiz edilmesi önerilmiştir. Grafın zaman serisine dönüştürülmesi sırasında öznitelik sayısının artırılması çalışmanın özgün değerlerinden biridir.
2. Lyapunov Üstelleri (LEs) gibi dinamik sistem öznitelikleri sınıflandırma sürecine dahil edilmiştir. Faz uzayında farklı gömme boyutları kullanarak, sistemin dinamik yapısındaki olası değişiklikleri ve belirsizlikleri daha iyi yakalamak amaçlanmış ve bu yaklaşımın model performansına katkıları incelenmiştir.
3. Elde edilen bu üstellere dayanan öznitelik vektörleri kullanılarak ilk olarak ML yöntemleri ile sınıflandırma yapılmıştır.

4. Daha sonra aynı üstellere dayanan öznitelik vektörleri ile işlemleri temsil eden graf yapısı tekrar oluşturulmuş ve Graf Evrişimli Ağ (Graph Convolutional Network, GCN) yöntemi kullanılarak sınıflandırma yapılmıştır. Sonuçlar, önerilen yöntemin kara para aklama faaliyetlerini tespit etmede etkili olduğunu göstermektedir. Ayrıca bu yöntem kara para aklamayı önleme (Anti Money Laundering, AML) konusunda graf yapısına ait kaotik yapıya dayalı sınıflandırma konusunda ilk olma niteliği taşımaktadır.
5. Ayrıca kaotik yapıdan bağımsız olarak Evrişimsel Sinir Ağları (Convolutional Neural Network, CNN), Uzun Kısa Süreli Bellek (Long Short Term Memory, LSTM), Çift Yönlü Uzun Kısa Vadeli Bellek (Bidirectional Long Short Term Memory, BiLSTM), Dikkat Mekanizması (Attention Mechanism, AM) yöntemleri ve bunların hibrit modelleri test edildikten sonra, CNN-BiLSTM-AM modeli önerilmiştir.

1.1. Tezin Amacı

Bu tezin amacı, blok zinciri tabanlı finansal işlemlerdeki kara para aklama faaliyetlerini tespit etmek için hibrit bir yöntem geliştirmektir. Bu amaç doğrultusunda, öncelikle blok zinciri ağı üzerindeki işlem graflarından elde edilen öznitelikler artırılarak bu öznitelikler kaotik zaman serilerine dönüştürülecektir. Zaman serilerinin, LEs hesaplanarak dinamik yapı tanımlanacak ve ortaya çıkan kaotik öznitelikler ML modellerine girdi olarak verilecektir. Ardından bu özniteliklere dayanarak bir işlem grafı yeniden oluşturulacak ve GCN gibi Derin Öğrenme (Deep Learning, DL) yöntemleri ile işlemlerin yasallığı sınıflandırılacaktır. Ayrıca, farklı DL mimarileri ve bunların hibrit modelleri karşılaştırılarak en yüksek tespit başarısını sağlayan model önerilecektir. Bu yaklaşım sayesinde, kara para aklama tespitinde doğruluk artırılacak ve yanlış pozitif oranları azaltılacaktır.

1.2 Literatür Taraması

Bu bölümde ML algoritmaları, Graf Sinir Ağı (Graph Neural Network-GNN) ve DL modelleri kullanılarak yapılan araştırmalar ve yöntemler açıklanmıştır.

GNN, graf verilerini işlemek ve analiz etmek için kullanılan bir tür sinir ağıdır (Zhou ve diğ., 2020). Geleneksel sinir ağları genellikle sabit yapılı verilerle çalışırken, GNN'ler, düğümler ve kenarlardan oluşan dinamik yapıları işleyebilir. GNN'ler, düğüm ve kenarların özelliklerini öğrenmek için komşuluk bilgisinden faydalanarak, graftaki ilişkileri ve yapıları modellemelerine olanak tanır. GNN'lerin Graf Dikkat Ağı (Graph Attention Network, GAT), GCN, GraphSAGE ve Graf İzomorfizm Ağı (Graph Isomorphism Network, GIN) gibi birçok

çeşidi bulunmaktadır. GCN, graf tabanlı veri yapılarında desenleri tanımak ve tahmin etmek için kullanılan bir DL modelidir. GCN, CNN temel alınarak graf veri yapılarına uyarlanmış bir yapıdır. İlk olarak, Weber ve diğ., (2019) çalışmalarında Elliptic veri setini önermiş ve yasa dışı Bitcoin işlemlerini tespit etmek için bu veri setini %70 eğitim ve %30 test olarak bölmüştür. Veri setini değerlendirmek ve ikili sınıflandırma yapmak için GCN, Lojistik Regresyon (Logistic Regression, LR), Rastgele Orman (Random Forest, RF) ve Çok Katmanlı Algılayıcı (Multilayer Perceptron, MLP) yöntemlerini kullanmışlardır. Sonuçlar kesinlik, duyarlık ve F1-skoru açısından karşılaştırılmıştır. RF yönteminde, %95 kesinlik, %67 duyarlık ve %78 F1-skoru elde etmişlerdir. Elbaghdadi ve diğ., (2022), k-En Yakın Komşu (k-Nearest Neighbor, k-NN) algoritmasını kullanarak yasadışı işlemleri tespit etmek için bir model geliştirdi. Alarab ve diğ., (2020) Elliptic veri setinde yasa dışı işlemleri tahmin etmek için MLP ve GCN'ye dayalı yeni bir yaklaşım sunmuşlardır. Önerilen yöntemde, GCN'den türetilen gizli temsiller ve doğrusal bir katmanın kullanımının, Weber ve diğ., (2019)'nın çalışmasına göre modelin performansını artırdığı sonucuna varmışlardır. GAT, graf verileri üzerinde çalışan bir sinir ağıdır. GAT, her düğümün komşularına farklı ağırlıklar atayarak komşularının özelliklerini toplar ve bu süreçte dikkat mekanizması kullanır (Veličković ve diğ., 2018). GAT'lar ilk kez Bitcoin anomali tespiti için Pocher ve diğ., (2023) tarafından uygulanmıştır. Çalışmada ML ve graf analizine dayalı yöntemleri incelemişlerdir. GCN'nin, RF'den daha iyi performans gösterirken, GAT'dan daha düşük performans gösterdiği sonucuna varmışlardır. Liu ve diğ., (2024) gelişen dikkat ağı temelli bir blok zinciri anomali tespit yöntemi önerdiler. Bu yöntem farklı zaman dilimlerinde alt ağların düğüm öğrenme ağırlıklarını dinamik olarak güncelleyen bir yapıdır. Her işlemi kenarlar kullanarak modellemek için dinamik özellikli graf ağı oluşturma ile farklı bir yöntem sundular. Bu yöntemde graf temsili öğrenme için daha öğrenilebilir işlem özellikleri sağlamayı amaçladılar. GraphSAGE düğüm öznitelikleri ve komşu bilgilerini kullanarak düğüm temsillerini öğrenen bir graf sinir ağı yöntemidir. Chen ve diğ., (2023) çalışmalarında GraphSAGE yöntemini kullanarak anomali tespit modeli önerdiler. Model, hedef ve komşu düğümler arasındaki benzerlikleri hesaplar ve bu benzerliklerin zamanla nasıl değiştiğini inceler. GraphSAGE, hedef düğümün komşu bilgilerini kullanarak benzerlik ile denetimli öğrenmeyi gerçekleştirir. Komşu düğümlerin özelliklerini bir araya getirmek için GraphSAGE'yi kullanmışlardır. Derin Graf Infomax (Deep Graph Infomax, DGI) ve GIN graf verilerini modellemek için kullanılan iki farklı GNN yöntemidir. Lo ve diğ., (2023), DGI ve GIN tabanlı Inspection-L adlı kendiliğinden denetimli (self-supervised) bir GNN çerçevesi önermişlerdir. DGI, graf yapısına sahip verilerde düğüm temsilleri elde etmek için kullanılan genel bir gözetimsiz öğrenme yaklaşımıdır. Önerilen Inspection-L yöntemi, yasa dışı işlemleri

tespit etmek için RF ve DGI'nin bir arada kullanımına dayanmaktadır. Ayrıca önerilen model, komşuluk bilgisini yakalayarak her düğümün tüm graf yapısına erişmesini sağlar. DGI, önerilen GIN ile birlikte düğüm yerleşimlerini öğrenmek için kullanmışlardır. Düğüm yerleşimleri daha sonra eğitim ve test setlerinde RF algoritması kullanılarak değerlendirmişlerdir.

DL, kara para aklama gibi karmaşık ve çok boyutlu problemlerin çözümünde etkili bir araçtır. Kara para aklama işlemlerinin mekânsal-zamansal karmaşıklığını ele almak amacıyla Wan ve Li (2024), MDGC-LSTM adlı DL tabanlı bir tahmin modeli önermişlerdir. Model, işlem verilerindeki çok katmanlı ilişkileri yakalamak için Dinamik Graf Evrişim Ağı ve LSTM kombinasyonunu kullanmaktadır. Bu model, Elliptic veri setinde mevcut en iyi modellere kıyasla %25 daha yüksek Macro-F1 skoruna ulaşmıştır. Yang ve diğ., (2023) AML sorununu çözmek için iki yöntem önermişlerdir. İlk yöntem, sezgisel kuralları ve LSTM+GCN algoritmasını birleştirerek, etiketlenmemiş verilerdeki farklı AML anomalilerini denetimli sınıflandırma yoluyla tespit eder. Yöntemde, sezgisel kurallar kara para aklama desenlerini filtreler. LSTM+GCN algoritması ise çok etiketli bir sınıflandırma modeli olarak yasa dışı kategorileri belirler. İkinci yöntemde ise, topluluk öğrenimini kullanarak, sezgisel kuralların tespit edemediği AML anomali işlem verilerini tanımlar. Bu iki yöntemin kombinasyonunda, kara para aklama işlemlerinin daha doğru bir şekilde tespit edildiği sonucunu elde etmişlerdir. Xia ve diğ., (2022) çalışmalarında mekânsal-zamansal model olan MGC-LSTM'i önermişlerdir. Modelde, farklı işlemler arasındaki karmaşık korelasyonu ve zaman içindeki bağımlılığı öğrenmek için GCN ve LSTM yöntemlerini birleştirmişlerdir. LSTM, zaman içindeki kara para aklama verilerinin bağımlılığını öğrenirken, GCN farklı kara para aklama işlemlerinin karmaşık mekânsal bağımlılığını öğrenmek için kullanmışlardır. Modelde LSTM'in çıktısı GCN'in girdisi olarak kullanılır. Çalışmalarında, kara para aklama tespiti için etiketlenmemiş verilerde farklı anomalileri tanımlamak için yeni bir yaklaşım sunmuşlardır. Xiao ve diğ. (2023) CTDM adlı GCN ağ parametrelerini ayarlamak için yeni bir yöntem sunmuşlardır. Bu yöntem, EvolveGCN'yi Minimal Kapılı Birim (Minimal Gated Unit, MGU) ile kullanmaktadır. MGU, LSTM hücrelerine alternatif olarak geliştirilmiştir. MGU, LSTM hücrelerine göre daha az giriş kapısı kullanmaktadır. Ayrıca MGU, bellek hücresine ve unutma kapısına doğrudan ek bir lojistik değer uygulayarak daha basit bir yapı sunmaktadır. CTDM, GCN parametrelerini iyileştirmek için MGU'yu kullanmakta ve MGU sayesinde daha az öğrenme parametresine ihtiyaç duymaktadır. Karim ve diğ., (2024) yarı denetimli graf öğrenme tekniklerini kullanarak AML tespitine yönelik yaklaşım önermişlerdir. Çalışmada, graf gömme modelleri düğüm gömmelerini oluşturmak amacıyla eğitilmiştir. Ardından, elde edilen bu gömmeler, ikili

sınıflandırıcıların eğitimi için kullanmışlardır. Ayrıca aynı amaç için SkipGCN, FastGCN ve EvolveGCN modellerini eğitilerek düğüm sınıflandırması yapmışlardır. Guo ve diğ. (2023), kara para aklama işlemlerinin topolojik yapısını ve özniteliklerini yakalamak için LB-GLAT'yi önermişlerdir. LB-GLAT, bir işlem grafiği ve ters işlem grafiği kullanarak blok zincir işlemlerinin hedefini başarıyla tanımlamak için tasarlanmıştır. Çalışmalarında %97 doğruluk, %93 hassasiyet, %84 geri çağırma, %88,87 F1-skoru elde etmişlerdir. Wang ve diğ. (2025), GCN'nin sınırlamalarını aşmak için RMGANets yöntemini önermişlerdir. Bu yöntem, çok ilişkili dikkatli graf farkındalığını içeren pekiştirmeli öğrenme tabanlı bir yaklaşımdır. Çalışmalarında, işlemler, adresler, kullanıcılar ve nakit akışlarını düğüm olarak içeren büyük birçok ilişkili graf oluşturmuşlardır. Pekiştirmeli öğrenmeyi, maskeleme işlemleri nedeniyle eksik olan düğüm bilgilerini tamamlamak ve farklı türdeki düğümler arasındaki farkları vurgulamak amacıyla kullanmışlardır.

Etiket eksikliği, eğitim veri setindeki tüm örneklerin ya da bazı örneklerin sınıf etiketlerine sahip olmamasını ifade eder. Lorenz ve diğ. (2020) Elliptic veri setindeki etiket eksikliğini incelemişlerdir. Veri setinde Bitcoin işlemlerinin %21'i yasal, %2'si ise yasa dışı olarak etiketlenmiştir. Geriye kalan işlemler, etiket bilgisi içermemektedir. Denetimsiz öğrenmenin performansını artırmak amacıyla Aktif Öğrenme (Active Learning) yöntemini kullanarak etiket sayısını artırmışlardır. Sınıflandırma işlemlerinde XGBoost (%76), LR (%45) ve RF (%83) algoritmalarını kullanmışlardır. Yazarlar bu soruna farklı bir yaklaşım getirmiş olsalar da, üstün sonuçlar elde etmeyi hedeflememişlerdir. Monte Carlo Dropout (MC-dropout), ML ve özellikle DL modellerinde belirsizlik tahmini yapmak için kullanılan bir yöntemdir. Bu yöntem, öğrenme sırasında bir ağı düzenlemek için kullanılan dropout adlı teknikle ilişkilidir. Alarab ve diğ., (2021) ve Alarab ve Prakoonwit (2023), çalışmalarında MC-Dropout yöntemini uygulamışlardır. Alarab ve Prakoonwit (2023) çalışmada, Bitcoin veri setini analiz etmek için bir aktif öğrenme çerçevesini önermişlerdir. Belirsizlikleri hesaplamak için MC-Dropout ve Monte Carlo Tabanlı Saldırı (MC-AA) yöntemlerini kullanmışlardır. Daha sonra LSTM ve GCN modellerini birleştirerek bir Zaman Temelli GCN (temporal-GCN) önermişlerdir. Çalışmalarının sonuçları, %97,7 doğruluk elde etmişlerdir. Jatoth ve diğ., (2021), sınıflandırma probleminde öznitelik seçimini analiz etmişlerdir. Öznitelik seçiminin amacı, daha iyi bir model elde etmek için daha uygun öznitelikleri seçmektir. Yasal ve yasa dışı işlemleri sınıflandırmak için topluluk öğrenme modelleri (ensemble learning) ve klasik denetimli öğrenme yöntemlerini uygulamışlardır. Çalışma sonunda topluluk öğrenme modellerinde daha iyi sonuçlar elde etmişlerdir.

AML çalışmalarının bir incelemesi olarak, kara para aklama yöntemlerinin giderek artan karmaşıklığını ve sanal para işlemlerinin anonim yapısının mevcut yöntemlere oluşturduğu önemli zorlukları ortaya koymaktadır. Bu sorun, etiketli verilerin eksikliğiyle daha karmaşık hale gelmekte ve bu durum, denetimli öğrenme tekniklerinin etkinliğini sınırlamaktadır (Wan ve Li, 2024). Daha önceki araştırmalar, GCN, LSTM, hibrit ve graf tabanlı yaklaşımlar da dahil olmak üzere çeşitli yöntemlerin potansiyelini araştırmış olsa da, bu modeller genellikle ölçeklenebilirlik, yorumlanabilirlik ve kripto para işlemlerinin dinamik doğasını etkili bir şekilde ele alma konularında zorluklarla karşılaşmaktadır.

ML yöntemlerinde öznitelik çıkarımı manuel olarak gerçekleştirilirken, CNN ve LSTM modellerinde giriş verisinin öznitelikleri otomatik olarak öğrenilmektedir (Aggarwal ve diğ., 2022). Öznitelik çıkarımı, önerilen modelin performansını artırmada ve kripto para işlemlerinin dinamik ve karmaşık doğasını etkili bir şekilde analiz etmede kritik bir rol oynar. Özellikle kripto para işlemleri genellikle düzensiz, yüksek boyutlu ve zaman serisi verisi içeren yapılardır. Bu tür verilerin doğrudan analiz edilmesi, hem öğrenme sürecini hem de sınıflandırma performansını olumsuz etkileyebilir. Öznitelik çıkarımı, bu tür karmaşık ve yüksek boyutlu verilerden anlamlı ve özet bilgiler elde edilmesini sağlar. Kute ve diğ. (2024), sentetik finansal işlem verisi üzerinde CNN modelini kullanmışlardır. Önerilen model, şüpheli işlemleri diğer ML yöntemlerine (RF, SVM, XGBoost) kıyasla daha düşük yanlış negatif oranları ve daha yüksek doğrulukla tespit etmiştir. Modelin açıklanabilirliğini sağlamak için Shapley Additive exPlanations (SHAP) yöntemi uygulanmış ve her bir özniteliğin model tahminlerine olan etkisi görselleştirilerek AML uzmanları tarafından doğrulanmıştır. Rani ve diğ. (2024), kara para aklama faaliyetlerini tespit etmek ve önlemek amacıyla LSTM tabanlı bir işlem ağı ve davranış analizi modeli önermişlerdir. Model, müşteri risk profiline ve şüpheli faaliyetleri raporlamaya odaklanmaktadır. Finansal verileri zaman serisi formatında analiz ederek, kara para aklama ile ilişkili olağandışı işlem desenlerini belirlemektedir.

BiLSTM, zaman bağımlılıklarını yakalamak ve anlam açısından daha kapsamlı bir yerel bağlam oluşturmak için kullanılan çift yönlü bir LSTM modelidir. AM ise, bir sinir ağının ardışık verilerdeki önemli bilgilere odaklanmasını sağlayan ve farklı gizli birimlere farklı ağırlıklar atayan bir yapıdır (Wang ve diğ., 2016). AM, önemli özniteliklere öncelik vererek modelin kritik bilgilere odaklanmasını sağlar. Bu yaklaşım, öğrenme sürecinde gereksiz bilgileri azaltarak daha doğru ve hızlı sonuçlar elde edilmesine olanak tanır. Ayrıca, öznitelik çıkarımı modelin yorumlanabilirliğini artırarak karar verme sürecini etkileyen faktörleri ön plana çıkarır. Tang ve diğ. (2024), Ethereum ağında kimlik avı dolandırıcılığı yapan hesapları

tespit etmek için BiLSTM ve AM tabanlı BiLSTM4DPS modelini önermişlerdir. Model, işlem miktarları ve işlem sayıları gibi temel öznitelikleri çıkararak hesap davranışlarını analiz etmektedir. Hesap işlem kayıtlarını ardışık dizilere dönüştürerek işlemlerin zamansal ve saklı desenlerini belirlemektedir. AM ile maskeleme teknikleri entegre edilerek dolandırıcılık yapan hesapları sınıflandıran bir model oluşturulmuştur. Wang ve diğ. (2024) blok zinciri işlemlerinde anormal davranışları tespit etmek için bir yöntem geliştirmişlerdir. Bu yöntem, CNN modeline çoklu katman yapısını entegre ederek farklı ölçeklerde öznitelik öğrenmeyi sağlamaktadır. Ayrıca, modelin anormallik tespit yeteneğini artırmak ve öğrenilen öznitelikler arasındaki ilişkileri daha iyi anlamak için kendine dikkat mekanizması (self-AM) kullanmışlardır. DL, AML tespiti için önemli bir potansiyel sunmasına rağmen, mevcut modellerin çoğu kripto para işlemlerinin dinamik ve karmaşık doğasını etkili bir şekilde ele almakta zorlanmaktadır. Bu çalışmada, CNN ile yerel özniteliklerin çıkarılması ve BiLSTM ile ardışık özniteliklerin çıkarılması birleştirilmiştir. Daha sonra, AM kullanılarak örnek girişteki her bir özniteliğin önemi dinamik olarak değerlendirilmiştir.

1.3 Hipotez

Bu tezde öne sürülen temel varsayım, blok zinciri tabanlı finansal işlem graflarından elde edilen yapısal özniteliklerin sayısının artırılması, bu özniteliklerin kaotik zaman serilerine dönüştürülmesi ve LEs ile tanımlanmasının, kara para aklama faaliyetlerine özgü dinamik desenleri geleneksel istatistiksel ve ML tabanlı yaklaşımlara kıyasla daha belirgin hale getireceğidir. Bu yöntemle elde edilen kaotik tabanlı özniteliklerin, işlem ağlarındaki karmaşık ve gizlenmiş ilişkileri ortaya çıkararak modelin öğrenme kapasitesini artırması beklenmektedir.

İkinci varsayım ise, bu kaotik tabanlı öznitelik kümesinin hem ML hem de DL modelleriyle işlenmesi sonucunda, blok zinciri tabanlı işlemler içerisinde yer alan şüpheli aktivitelerin tespitinde doğruluğun artacağı ve yanlış pozitif oranlarının azalacağıdır. Böylece, geliştirilen hibrit yaklaşımın kural tabanlı ve klasik veri analizi yöntemlerine göre daha yüksek performans sergilemesi ve kara para aklama tespitinde güvenilirliği artırması öngörülmektedir.

1.4 Tez İçeriği

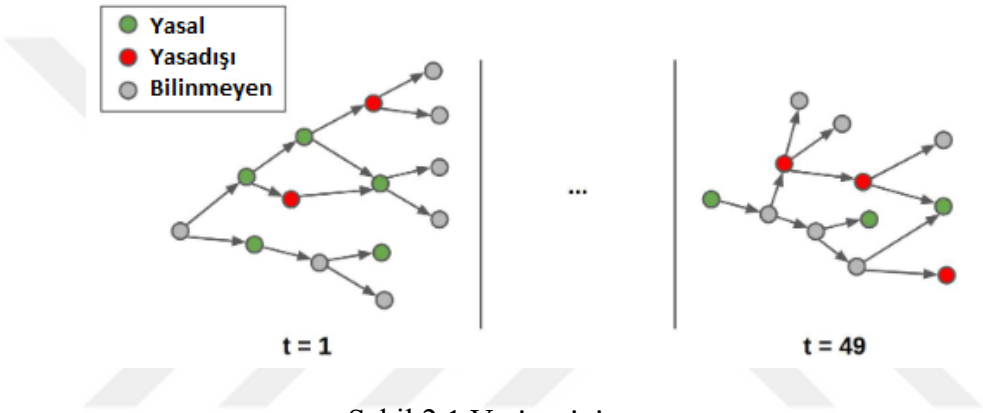
Tezin 2. bölümünde çalışmada kullanılan veri seti tanıtılmış; öznitelik artırımı, kaotik temelli öznitelik çıkarımı, faz uzayının elde edilmesi ve LEs hesaplanması detaylandırılmıştır. Ayrıca, farklı sınıflandırma yaklaşımları olarak ML, GCN ve DL modelleri açıklanmıştır. 3. bölümde bu tez çalışması kapsamında geliştirilen modeller anlatılmıştır. 4. bölümde deneysel çalışmalar ve performans değerlendirmeleri sunulmuş, elde edilen bulgular ayrıntılı biçimde

paylaşılmıştır. Ayrıca, önerilen yaklaşımın sonuçları literatürde yapılan çalışmalar karşılaştırmalı olarak sunulmaktadır. 5. bölümde tartışma yapılmış, 6.bölümde ise sonuçlar ve gelecekte yapılabilecek çalışmalar hakkında önerilerde bulunulmuştur.

2. MATERYAL VE METOT

2.1. Veri Seti

Bu çalışmada kullanılacak olan Elliptic veri seti bir bitcoin grafıdır (Weber ve diğ., 2019). Bu veri seti, herhangi bir kripto para biriminde mevcut olan en büyük etiketli veri setlerinden biri olarak kabul edilmektedir (Alarab ve diğ., 2020). Şekil 2.1'de yapısı verilen Elliptic veri seti, farklı zaman adımlarında Bitcoin blok zincirinden örneklenmiş 49 graf yapısı içermektedir. 1'den 49'a kadar uzanan zaman adımları yaklaşık iki haftalık eşit aralıklarla eşit yerleştirilmiştir. Her bir zaman adımı, blok zincirinde aralarında üç saatten daha kısa bir süre bulunan işlemlerin tek bir bağlantılı bileşenini içerir, farklı zaman adımlarını birbirine bağlayan kenarlar yoktur.



Şekil 2.1 Veri setinin yapısı

Elliptic veri seti, bir işlemden diğerine giden Bitcoin para biriminin akışını temsil eden 203.769 düğüm işlemi ve 234.355 yönlendirilmiş kenar içerir. Her işlem, "yasal", "yasadışı" veya "bilinmeyen" olmak üzere üç sınıfa ayrılır. Toplam işlem sayısının %21'i (42.019) yasal, %2'si (4.545) yasa dışı ve kalan %77'si (157.205) bilinmeyen sınıf olarak etiketlenmiştir. Veri seti 166 adet özelliğe sahiptir. İlk 94'ü zaman adımı, girdi/çıktı sayısı ve işlem ücreti gibi işlemin kendisiyle ilgili bilgilerle ilgilidir. Kalan özellikler ise her bir işlemin maksimum, minimum, standart sapma ve korelasyon katsayılarını vererek, işlemin doğrudan komşuları hakkında toplu bilgilerle ilgilidir.

2.2 Öznitelik Artırımı

Kaotik sistem, başlangıç sartlarına hassas bağlılık gösteren ve ölçülemeyecek karmaşıklıkta sistemlerdir (Strogatz, 1994). Kaos bilimine göre, kaotik bir sistemin başlangıç şartlarındaki ölçülemez derecede küçük bir değişiklik, sistemin gelecekteki durumunda ölçülemez ve çok büyük değişikliklere neden olabilir. Kaotik terimi, insanın hesaplama kapasitesini aşan, son derece karmaşık ancak kendi iç düzenine sahip süreçleri ifade etmektedir. Kaotik hareketin,

rastgele her durumu alamadığı, belli bir olasılıklar kümesi içerisinde hareket etmek zorunda olduğunu bilinir. Yani kaos, aslında oldukça karmaşık bir "düzen"dir.

Özellikle dikkat edilmesi gereken önemli bir konu da, kaosun rastgelelik olmadığıdır. Kaotik sistemler, bilinçsizce de olsa tüm girdileri değerlendirip ona göre nihai bir davranış ortaya koyarlar. Sistemin zamanla değişen parametrelerini gösteren ve sistemin zaman içinde nasıl bir davranış gösterdiğinin bir yansıması olan bu tip verilere "zaman serileri" adı verilir.

Değişkenlerin çok sayıda olması, ortamı kaotik yapan temel etkendir. Bir sistemin kaotik olup olmadığını anlamak için elimizde ilk olması gereken şey, sistemin davranışına dair olabildiği kadar uzun süreyle kaydedilmiş bir değişkenler kaydıdır. Bu amaçla çalışmada ilk 94 öznitelik değerine Polinom Özellikleri (Polynomial Features) ve Etkileşim Özellikleri (Interaction Features) kullanılarak öznitelik sayısı artırılmıştır. Daha sonra artılaran bu özniteliklerden zaman serileri elde edilmiştir.

Polinom ve etkileşim özellikleri, veri işleme ve ML modellerinde kullanılan öznitelik mühendisliği teknikleridir. Polinom özellikleri, mevcut özniteliklerin polinom terimlerine genişletilmesiyle oluşturulur. Etkileşim özellikleri ise, birden fazla özniteliğin çarpımıyla oluşturulan yeni özniteliklerdir (Albon, 2018; Nguyen ve diğ., 2021).

Örneğin, veri kümesi üç öznitelikten oluşuyorsa:

$$x = [x_1, x_2, x_3] \tag{2.1}$$

2. dereceden polinom özellikleri:

$$x = [x_1, x_2, x_3, x_1^2, x_2^2, x_3^2, x_1x_2, x_1x_3, x_2x_3] \tag{2.2}$$

2. dereceden etkileşim özellikleri:

$$x = [x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3] \tag{2.3}$$

Polinom özellikleri sayısı Denklem (2.4) ve etkileşim özellikleri sayısı Denklem (2.5) kullanılarak hesaplanmıştır.

$$\binom{n+d}{d} = \frac{(n+d)!}{d!.n!} \tag{2.4}$$

$$n + \binom{n}{d} = n + \frac{n!}{d!.(n-d)!} \tag{2.5}$$

Burada;

n: özniteliklerin sayısı;

d: polinomun derecesi.

Bu çalışmada, n=93 ve d=2 değerleri kullanıldığında polinom özelliği ile oluşturulan yeni özellik sayısı şu şekildedir:

C(93+2,2) = C(95,2) = 4465.

Etkileşim özelliği ile oluşturulan yeni özellik sayısı şu şekildedir:

93+C(93,2) = 93+4278 = 4371.

Polinom ve etkileşim özellikleri arasındaki temel fark, polinom özelliklerin hem bireysel özniteliklerin kendi kendilerine olan kombinasyonlarını hem de farklı özniteliklerin birbiriyle olan kombinasyonlarını içermesidir. Etkileşim özellikleri ise sadece farklı özniteliklerin birbiriyle olan çarpımlarını içerir ve bireysel özniteliklerin kendi kendilerine olan kombinasyonlarını içermez.

2.3 Kaotik Temelli Öznitelik Çıkarımı

Gerçek dünyada gözlemlenen kaos genellikle kaotik zaman serisi verileri şeklindedir. Bu nedenle, kaotik zaman serilerinin analizi kaos araştırmalarında kritik bir öneme sahiptir (Zhang ve diğ., 2004). Rastgele görünen kaotik zaman serisi verileri faz uzayına yerleştirildiğinde kaos çekicisi gözlemlenebilir. Kaotik zaman serileri zaman boyutunda bazı rastgele davranışlar gösterirken, faz uzayı yapısında belirli bir düzenlilik sergilerler. Bu nedenle, faz uzayında kaotik zaman serileri analiz edilebilir.

2.3.1 Faz uzayının elde edilmesi

Faz uzayı, bir sistemin parametrelerinin grafikte tek bir nokta olarak temsil edilmesidir. Faz uzayının boyutu, bu parametrelerin sayısı ile belirlenir. Zaman serilerinde faz uzayı, verilerin zaman gecikmesiyle oluşturulan kopyalarının birbirlerine göre olan davranışlarını gösteren bir diyagramdır. Bu çalışmada verisetinde verilen öznitelikler arasındaki ilişkiyi ve kaotik yapıyı açıklamak için, öznitelik vektörleri zaman serisi gibi ele alınarak faz uzayına dönüştürülmüştür. Faz uzayı vektörleri Denklem (2.6) yardımıyla oluşturulur.

$$X(i) = \{x(n), x(n+\tau), \ldots\ldots, x(n+(m-1)\tau)\}, \quad i = 1,2,3, \ldots, M \tag{2.6}$$

$$x = \{x_n, n = 1,2, \ldots, N\} \tag{2.7}$$

Burada n zaman indeksini ve N örnek sayısını gösterir. Bu zaman serisine göre elde edilecek faz uzayı vektörü $X(i)$ şu şekilde oluşturulur:

$X(1) = [x_1 \quad x_{1+\tau} \dots\dots\dots x_{(1+(m-1)\tau)}]$;

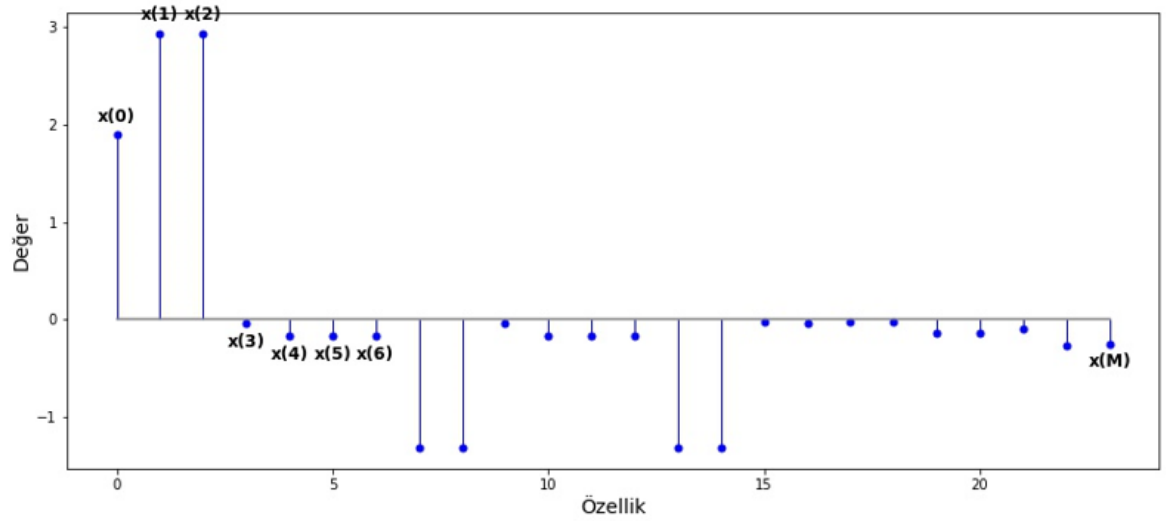
$X(2) = [x_2 \quad x_{2+\tau} \dots\dots\dots x_{(2+(m-1)\tau)}]$;

$X(3) = [x_3 \quad x_{3+\tau} \dots\dots\dots x_{(3+(m-1)\tau)}]$;

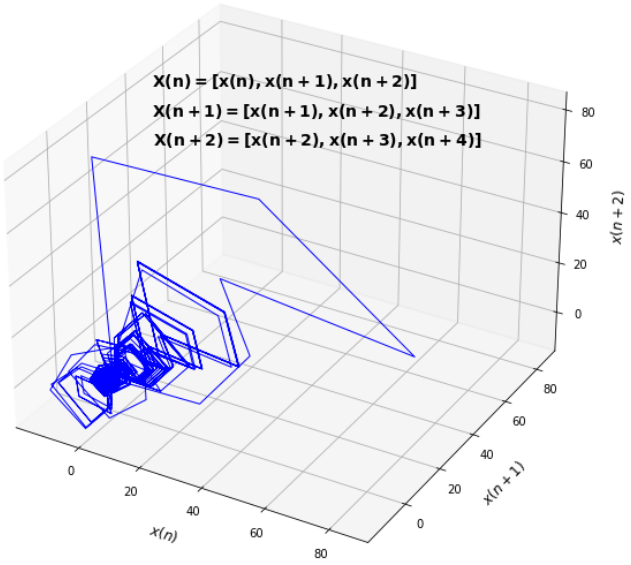
……

$X(M) = [x_M \quad x_{M+\tau} \dots\dots\dots x_{(M+(m-1)\tau)}]$.

$x_{(n)}$ zaman serisi, m gömme boyutu, τ zaman gecikmesi ve $M = N - (m-1)\tau$ gömülü noktaların sayısıdır. Zaman gecikmesi her bir vektörün komşu elemanları arasındaki sabit zaman farkını belirtir. Şekil 2.2 zaman gecikmeli vektör yörüngesine sahip özniteliklerin bir kısmını, Şekil 2.3 ise m gömme boyutu için faz uzayını gösterir.



Şekil 2.2 Öznitelik vektörleri için sinyalin zaman serisi



Şekil 2.3 Zaman serisinin faz uzayı

2.3.2 Lyapunov üstelleri

LEs, sistemin kaotik bileşenler içerip içermediğini anlamamıza yarayan matematiksel bir analiz yöntemidir. LEs, bir sistemin olası durumlarını gösteren çekerler üzerinde, başlangıçta yakın komşu olan iki rastgele noktanın birbirlerinden ayrılma derecesinin sayısal bir ifadesidir (Altuntaş ve diğ., 2020). Eğer bu komşu noktalar hızla birbirlerinden ayrılıyorlarsa, hesaplanan LEs pozitif bir değerde olacaktır. Bu da sistemin kaotik bir yapıda olduğunu gösterir (Wolf ve diğ., 1985). Denklem (2.8)'de $s(n)$ referans noktası, ona en yakın komşu $s(m)$ ve $d(s(n), s(m))$ başlangıçtaki en yakın komşular arasındaki uzaklıktır. Sonraki iki nokta arasındaki Öklid uzaklığı $d(s(n+1), s(m+1))$ dir. Her bir nokta için bu Öklid uzaklığı N defa tekrarlanarak LEs hesaplanıp üstellerin ortalaması alınarak yörüngelerdeki yakınlaşma ve uzaklaşmayı ifade eden λ , Denklem (2.8) ile elde edilir.

$$\lambda = \frac{1}{N}\sum_{n=1}^{N} ln \frac{d(s(n+1),s(m+1))}{d(s(n),s(m))} \tag{2.8}$$

Faz uzayında her bir boyuttaki gelişmeyi bir λ temsil ettiğinden, m boyutlu bir sisteme ait Lyapunov üsteli spektrumu, λ_1 en büyük olmak üzere $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$ şeklinde elde edilir. Kaotik bir sistem en az bir pozitif Lyapunov üsteline sahip olacağı için sistemde en büyük Lyapunov üsteli $\lambda_1>0$ ise davranış kaotik, $\lambda_1<0$ ise davranış kararlıdır (Abarbanel ve diğ., 1993). Lyapunov üstel hesaplamalarını yapmak için TISEAN paket programı (Hegger ve diğ., 1999) kullanıldı.

2.4 Sınıflandırma Yöntemleri

2.4.1 Makine öğrenmesi yöntemleri

ML, bilgisayarların veri ile deneyim kazanarak ve örüntüleri tanıyarak belirli görevleri otomatik olarak öğrenmesini sağlayan bir yapay zekâ dalıdır (Gök, 2024). Bu süreçte algoritmalar, büyük miktarda veriyi analiz eder, anlamlı ilişkiler kurar ve gelecekteki veriler üzerinde tahminlerde bulunur (Mohri ve diğ., 2018).

- *Destek Vektör Makineleri (Support Vector Machine, SVM) :* SVM, Cortes ve Vapnik (1995) tarafından önerilen sınıflandırma ve regresyon analizinde kullanılan bir algoritmadır. SVM, veri noktalarını sınıflandırmak için en iyi hiper düzlemi bulur. Bu hiper düzlem, farklı sınıfları birbirinden en iyi şekilde ayıran çizgidir ve sınıflar arasındaki mesafeyi maksimize eder.

- *k-En Yakın Komşu :* k-NN algoritması, uygulama kolaylığı ve basitliği nedeniyle yaygın olarak kullanılan bir algoritmadır (Peterson, 2009). k-NN, yeni bir veri noktasının sınıfını veya değerini belirlemek için, en yakın k komşusunun sınıflarını veya değerlerini kullanır. Bu işlemde, yeni veri noktası ile diğer veri noktaları arasındaki mesafeler hesaplanır ve en yakın k komşu belirlenir. Mesafe metrikleri olarak Öklid, Manhattan ve Minkowski mesafeleri kullanılabilir. Denklem (2.9)'da Minkowski mesafesi verilmiştir.

$$D(p,q) = (\sum_{i=1}^{n} \mid p_i - q_i \mid^m)^{\frac{1}{m}} \tag{2.9}$$

Burada:

p ve q, n boyutlu vektörlerdeki iki noktayı temsil eder.

m, Minkowski mesafesinin derecesidir. Örneğin, m=1 olduğunda Manhattan mesafesini, m=2 olduğunda ise Öklidyen mesafesi elde edilir.

- *Rastgele Orman :* RF, bir dizi karar ağacından oluşur. Eğitim veri setine aşırı uyum sağlayan bir karar ağacının aksine daha genelleştirilmiş bir modeldir (Schonlau ve Zou, 2020). Her ağaç eğitilirken, düğümler belirli öznitelikler üzerinden bölünür ve en iyi bölünme noktası rastgele seçilir (Couronné ve diğ., 2018). Bu yaklaşım, modelin genelleştirme yeteneğini artırır ve aşırı öğrenmeyi azaltır. RF'nin, eğitimi hızlıdır (Raiter, 2021). Çünkü her bir karar ağacı birbirinden bağımsız olarak eğitilir. Bu özellik sayesinde, büyük veri setleri üzerinde etkilidir.

- *Naive Bayes (NB) :* NB, olasılığa dayalı bir algoritmadır. Bu algoritma Bayes teoreminin temel prensiplerine dayanır (Alkhalili ve diğ., 2021). Sınıflandırma için kullanıldığında, Bayes Teoremi Denklem (2.10) ile ifade edilir:

$$P(y \mid X) = \frac{P(X \mid y).P(y)}{P(X)} \tag{2.10}$$

Burada,

P(y|X) : Verilen bir X özelliğine sahip bir veri noktasının sınıfının y olma olasılığı.

P(X|y) : Belirli bir sınıfa y ait olan X özniteliklerine sahip bir veri noktasının olasılığı.

P(y) : Herhangi bir veri noktasının rastgele bir sınıfa ait olma olasılığı.

P(X) : X özniteliklerine sahip bir veri noktasının olasılığı.

NB algoritması, veri setindeki öznitelikler arasındaki ilişkileri ve her bir özelliğin sınıf etiketiyle ilişkisini kullanarak yeni bir örneğin sınıfını tahmin eder. Bu tahmin süreci, öznitelikler arasındaki bağımsızlık varsayımı nedeniyle basitleştirilir.

- *Logistik Regresyon :* LR, bir bağımlı değişkenin (sınıf etiketi) olasılığını bağımsız değişkenlerin (özniteliklerin) doğrusal kombinasyonuna dayalı olarak modellemeyi amaçlayan sınıflandırma tekniğidir (Healy, 2006). Bu algoritma, gerçek değerleri 0 ile 1 arasında sınırlayan sigmoid fonksiyonunu kullanır. LR'nin temel varsayımı, $\beta_0+\beta_{1X}$ şeklindeki doğrusal fonksiyondur. Bu fonksiyon, sigmoid fonksiyonu ile dönüştürülür. Bu nedenle β_0, β_1, β_2 … ve β_k X değerleri ne olursa olsun, y (hedef değişken) her zaman 0 ve 1 değerlerini alır. LR modelleri, X'in değeri verildiğinde y'nin 1 olma olasılığını Denklem (2.11) ile tahmin eder:

$$P(Y = 1 \mid X = x) = \frac{e^{B_0+B_1x}}{1+e^{B_0+B_1x}} \tag{2.11}$$

- *Çok Katmanlı Algılayıcı* : Yapay sinir ağlarının en temel ve yaygın formlarından biridir (Kruse ve diğ., 2022). MLP, birden fazla birbirine bağlı nörondan oluşan katmanlı bir yapıya sahiptir. Bu yapı genellikle giriş katmanı, bir veya daha fazla sayıda gizli katman ve çıktı katmanından oluşur. Modelin ilk verileri aldığı bölüm giriş katmanıdır. Bu katmanın ardından gelen gizli katmanlar, her biri belirli ağırlıklar ve bias değerleri ile donatılmış nöronlardan oluşur ve modelin verilerdeki karmaşık desenleri öğrenmesini sağlar. Son olarak, tahminlerin veya sınıflandırma sonuçlarının üretildiği bölüm ise çıktı katmanıdır**.**

- *Aşırı Gradyan Artırma (Extreme Gradient Boosting, XGBoost):* XGBoost, bir topluluk ağacı algoritmasıdır (Chen ve Guestrin, 2016). Bu algoritma, Friedman'ın gradyan artırma

algoritmasına dayanmaktadır (Friedman, 2001). XGBoost, karar ağaçlarını verimli bir şekilde uygular. Ağaç oluştururken maksimum derinlik değerini kullanır. Aşırı büyüme durumunda budama yaparak aşırı uyumun önüne geçer. Gradient Boosting algoritmasıyla karşılaştırıldığında, XGBoost ikinci dereceden fonksiyonları kullanarak kayıp fonksiyonlarını hesaplar, bu da daha iyi sonuçlar elde etmeyi sağlar. Ayrıca, paralel çalışma özelliği sayesinde diğer algoritmalara göre daha hızlı sonuç alınabilir.

- *Light Gradyan Artırma Makinesi (Light Gradient Boosting Machine, LightGBM):* LightGBM, Microsoft tarafından geliştirilen, hızlı ve dağınık bir gradyan artırma yöntemidir. LightGBM, diğer gradyan artırma çözümlerine kıyasla daha hızlı eğitim süreleri ve daha yüksek performansa sahiptir (Ke ve diğ., 2017). Bu, büyük veri kümeleri üzerinde etkili bir şekilde çalışmasını sağlar. LightGBM, histogram tabanlı çalışan bir algoritmadır. Düşük bellek kullanımıyla büyük veri kümeleri üzerinde yüksek performans sağlar.

2.4.2 Graf evrişimli ağ yöntemi

GCN, graf verileri üzerinde işlem yapmak için kullanılan bir DL modelidir. Temelde, komşuluk ilişkilerini kullanarak düğümler arasındaki ilişkileri modellemek için tasarlanmıştır (Kipf ve Welling, 2017). CNN'deki evrişim katmanları, GCN'lerdeki "evrişim" ile aynı süreçtir. Giriş nöronları, filtreler veya çekirdekler adı verilen ağırlıklarla çarpılır. Filtreler görüntü üzerinde kayan bir pencere görevi görerek CNN'nin yakındaki hücrelerden bilgi öğrenmesine olanak tanır. Komşu düğümleri analiz ederek öznitelikleri öğrenen GCN'ler de benzer davranışlar sergiler. CNN'ler ve GCN'ler arasındaki temel fark, CNN'lerin düzenli (Öklidyen) sıralı veriler üzerinde çalışacak şekilde olmasıdır. GCN'ler ise farklı sayıda düğüm bağlantısına ve sırasız düğümlere sahip CNN'lerin genelleştirilmiş bir versiyonudur (Khemani ve diğ., 2024).

GCN'ler özellikle sosyal ağlar, öneri sistemleri ve daha fazlası gibi graf olarak temsil edilen verileri içeren görevler için uygundur (Wu ve diğ., 2019). GCN'lerin arkasındaki temel fikir graf verileri üzerinde evrişim işlemlerini gerçekleştirmektir. Bu, bir düğümün özniteliklerini ve komşu düğümlerin özniteliklerini dikkate alarak bir graf içindeki düğümler arasında bilgiyi yakalayıp iletmelerini sağlar. GCN'ler tipik olarak her biri graftaki düğüm temsillerini iyileştirmek için evrişim ve toplama adımları gerçekleştiren birkaç katmandan oluşur. GCN'ler, bu katmanları yinelemeli olarak uygulayarak graf verileri içindeki karmaşık modelleri ve bağımlılıkları yakalayabilir. GCN'ler görüntü sınıflandırma (Monti ve diğ., 2017), trafik (Cui ve diğ., 2020), öneri sistemleri (Fan ve diğ., 2019) ve görsel soru yanıtlama (Teney ve diğ., 2017) gibi birçok problem için kullanılmıştır.

GCN'lerdeki her katman, komşu düğümlerin özniteliklerini dikkate alarak graftaki her düğümün öznitelik vektörünü günceller. Bir G grafı, V ile gösterilen düğümlerden (node veya vertex) ve E ile gösterilen kenarlardan (edge) oluşur (G=(V,E)).

N düğüm sayısı olmak olmak üzere G grafının;

Komşuluk matrisi : $A \in R^{N\times N}$,

Derece Matrisi : $D_{ii} = \sum_j A_{ij}$,

Öznitelik Matrisi : $X \in R^{N\times C}$ (C bir öznitelik vektörünün boyutu)

Ağırlık matrisi : W olarak tanımlanır.

GCN'lerin katman bazında ileri yayılım işlemi Denklem (2.12)'de gösterilmiştir

$$X_{l+1} = \sigma(\hat{A}X_l W_l) \tag{2.12}$$

Burada $\hat{A}$, A nın şu şekilde tanımlanan normalizasyonudur:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}, \tag{2.13}$$

$$\tilde{A} = A + I,$$

$$\tilde{D} = diag\left(\sum_j \tilde{A}_{ij}\right).$$

σ, aktivasyon fonksiyonudur.

X_l ve X_{l+1} sırasıyla katman l'in giriş ve çıkış matrisleridir. Her iki matris için de satır sayıları aynı olup graftaki düğüm sayısına karşılık gelir; sütun sayıları ise giriş ve çıkış öznitelik uzayının boyutlarına bağlı olarak farklı olabilir.

2.4.3 Evrişimsel sinir ağları

CNN, DL alanında yaygın olarak kullanılan bir yapay sinir ağı türüdür (Watanobe ve diğ., 2023). Temel olarak, CNN, verinin yapısını ve özniteliklerini anlamak için konvolüsyon ve havuzlama gibi özel katmanları kullanır (Baysal ve Taşkın, 2024).

CNN'in temel adımları, genellikle birbirini takip eden konvolüsyon, aktivasyon, havuzlama ve tam bağlantılı katmanlardan oluşur. Konvolüsyon katmanı, bir filtrenin veri üzerinde kaydırma işlemi yaparak öznitelik haritalarının oluşturulmasını sağlar. Bu öznitelik haritaları, giriş veri içerisindeki farklı özniteliklerin temsilini sağlar. Aktivasyon katmanı, her öznitelik haritasındaki piksellerin üzerinden geçerken belirli bir aktivasyon fonksiyonunu uygular. Bu aktivasyon fonksiyonu, ağırlıklara dayalı olarak her öznitelik haritasındaki önemli öznitelikleri

vurgular veya bastırır. Havuzlama katmanı, öznitelik haritalarındaki boyutu azaltma amacıyla kullanılır (Cihan ve diğ., 2023). Genellikle maksimum veya ortalama değerleri kullanarak veriyi örnekler. Bu sayede, modelin işlemesi gereken parametre sayısı azaltılarak hesaplama maliyeti düşürülür ve aşırı öğrenme (overfitting) riski azaltılır. Tam bağlantılı katmanlar, öznitelik haritalarının çıktılarını düzleştirerek ve ardından geleneksel bir yapay sinir ağı mimarisine aktararak sınıflandırma veya çıktı tahmini yaparlar. Bu katmanlar, CNN'nin önceki katmanlarda öğrendiği öznitelikleri kullanarak nihai kararları verir.

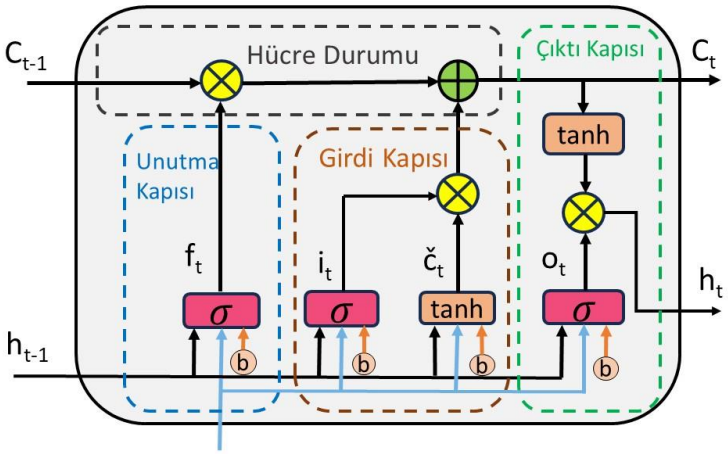
2.4.4 Çift yönlü uzun kısa vadeli bellek

Yinelenen Sinir Ağları (Recurrent Neural Networks (RNN)), verilerin önceki bilgilerine bağlı olarak çalışan bir DL modelidir (Pascanu ve diğ., 2013). RNN'lerin temel yapı taşı, genellikle "hücre" olarak adlandırılan birimlerdir. Bu hücreler, hem mevcut girdiyi hem de önceki zaman adımına ait çıktıyı işleyerek içsel durumlarını koruma yeteneğine sahiptir. RNN mimarisi, her zaman adımında bu hücrelerin çıktısını bir sonraki zaman adımının girdisi olarak kullanarak zaman bağımlılığını modelleyebilir. RNN'lerin bir diğer önemli unsuru, geriye yayılım (backpropagation) algoritmasıdır. Bu algoritma, hata sinyallerini hesaplayarak ağın ağırlıklarını güncellemek ve parametrelerini optimize etmek amacıyla kullanılır.

RNN'lerin bir diğer varyasyonu, LSTM ağlarıdır (Sherstinsky, 2020). LSTM, sıralı verileri girdi olarak alır ve önceki zaman adımlarında öğrenilen bilgileri hafızasında tutarak, sonraki adımlara yönelik tahminlerde bulunur. Bu yapısı sayesinde, LSTM modelleri karmaşık ve bağımlılığı yüksek zaman serisi verilerini etkili bir şekilde işleyebilir. Ayrıca, düzenlileştirme teknikleri ile birleştirildiğinde, daha sınırlı sayıda eğitim verisiyle dahi başarılı sonuçlar üretme kapasitesine sahiptir. LSTM hücresi, unutma, girdi, çıktı kapılarından ve hücre durumundan oluşan bir mekanizmadan oluşur. Şekil 2.4 'de LSTM yapısı gösterilmiştir.

Unutma Kapısı (Forget Gate) : Hangi bilginin tutulacağı veya unutulacağına karar verir. Bir önceki gizli katmandan gelen bilgiler ve güncel bilgiler sigmoid fonksiyonundan geçer. X_t girişinden ve bir önceki çıkış h_{t-1}'den gelen bilgiler sigmoid fonksiyonundan geçirilir. Sigmoid fonksiyonu 0 ile 1 arasında değer üretir. Elde edilen değer 0'a ne kadar yakınsa bilgiler o kadar unutulacak demektir. Denklem (2.14)'de unutma kapısı (f_t) verilmiştir.

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \tag{2.14}$$



Şekil 2.4 LSTM yapısı

Girdi Kapısı (Input Gate) : Hücre durumu'nu güncellemek için kullanılır. Öncelikle sigmoid fonksiyonu uygulanır ve hangi bilginin tutulacağına karar verilir. Daha sonra ağı düzenlemek için tanh fonksiyonu yardımıyla -1, 1 arasına indirgenir. Elde edilen iki değer çarpılarak hücre durumu güncellenir. Bu yeni bellek C_{t-1} belleğine eklenerek C_t elde edilir. Denklem (2.15)'de sigmoid; Denklem (2.16)'da ise tanh fonksiyonu adımları verilmiştir.

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \tag{2.15}$$

$$\check{C}_t = \tanh(W_C . [h_{t-1}, x_t] + b_C \tag{2.16}$$

Hücre Durumu (Cell State) : Hücre durumu'nun hücre içerisindeki en önemli görevi bilgiyi taşımaktır. Taşınması gereken verileri alır ve hücre sonuna, oradan da diğer hücrelere taşır. İlk olarak unutma kapısı'dan gelen sonuç ile bir önceki katmanın sonucu çarpılır. Daha sonra girdi kapısı'ndan gelen değer ile toplanır. Yeni hücre durumu (C_t) Denklem (2.17)'de verilmiştir.

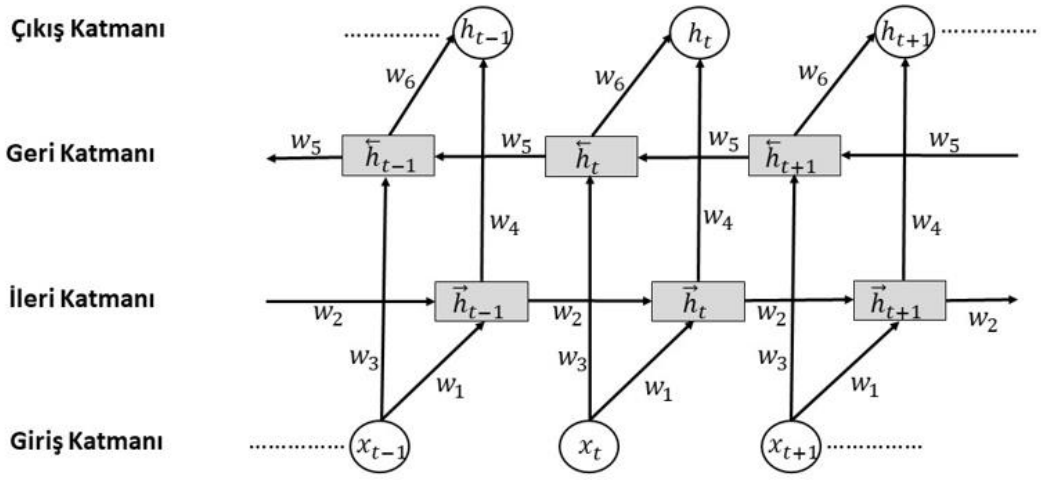
$$C_t = f_t * C_{t-1} + i_t * \check{C}_t \tag{2.17}$$

Çıktı Kapısı (Output Gate) : Bir sonraki katmana gönderilecek değere karar verir. Bu değer, tahmin için kullanılır. Öncelikle bir önceki değer ile şu anki girdi üçüncü sigmoid fonksiyonua aktarılır. Hücre durumun'dan gelen değer tanh fonksiyonundan geçer ve iki değer çarpılır. Elde edilen değer sonraki katmana "bir önceki değer" olarak gider. Denklem (2.18)'de çıkış değeri (h_t) ve Denklem (2.19)'da çıkış hücre durumu (O_t) verilmiştir.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o \tag{2.18}$$

$$h_t = o_t * \tanh(C_t) \tag{2.19}$$

BiLSTM, LSTM algoritmasının geliştirilmiş bir versiyonudur (Schuster ve Paliwal, 1997). LSTM mimarisi tek yönlüyken BiLSTM çift yönlüdür. BiLSTM, biri girdiyi ileri yönde, diğeri ise geri yönde işleyen iki LSTM katmanını içerir. LSTM geçmiş veri bilgilerini, BİLSTM ise geçmiş ve gelecek veri bilgilerini kullanarak tahmin yapabilir. BiLSTM ağ yapısı Giriş Katmanı, İleri Katmanı, Geri Katmanı ve Çıkış Katmanı olmak üzere dört katmandan oluşur. Şekil 2.5'de BiLSTM yapısı gösterilmiştir.



Şekil 2.5 BiLSTM yapısı

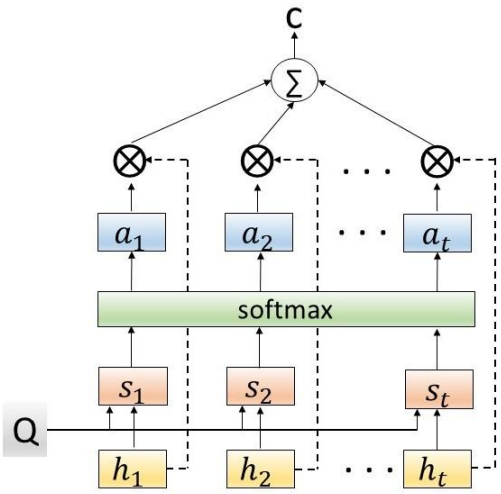
BiLSTM'de giriş veri dizisi (x_{t-1}, x_t , x_{t+1}) hem ileri LSTM hem de geri LSTM'de giriş olarak kullanılır. Her *t* adımında aynı anda ileri LSTM gizli durumu ($\vec{h}_t$) ve geri LSTM gizli durumu ($\overleftarrow{h}_t$) hesaplanır. BiLSTM'in çıktısı h_t , hem $\vec{h}_t$ hem de $\overleftarrow{h}_t$ bilgileri kullanılarak hesaplanır. Denklem (2.20-2.22) $\vec{h}_t$, $\overleftarrow{h}_t$ ve h_t değerleri hesaplanmaktadır.

$$\vec{h}_t = f((w_1 x_t + w_2 \vec{h}_{t-1})) \tag{2.20}$$

$$\overleftarrow{h}_t = f((w_3 x_t + w_5 \overleftarrow{h}_{t+1})) \tag{2.21}$$

$$h_t = g((w_4 \vec{h}_t + w_6 \overleftarrow{h}_t)) \tag{2.22}$$

Dikkat Mekanizması: AM, özellikle doğal dil işleme ve görüntü işleme gibi alanlarda kullanılan DL modellerinde, belirli bir girdiye daha fazla veya daha az dikkat göstermelerini sağlayan bir mekanizmayı ifade eder (Niu ve diğ., 2021). Bu, modelin belirli bir görevi daha iyi yapabilmesini sağlayabilir ve aynı zamanda daha az hesaplama kaynağı kullanmasına yardımcı olabilir. Şekil 2.6'da gösterildiği gibi AM'nin hesaplama süreci üç aşamadan oluşmaktadır.



Şekil 2.6 AM yapısı

Dikkat Puanı (Attention Score) **:** Dikkat puanı, girdi ve çıktı özellikleri arasındaki benzerliği veya ilişkiyi ölçer. Bu puan, modelin hangi girdi özelliklerine odaklanması gerektiğini belirlemesine yardımcı olur. Bu nedenle, dikkat puanı, çıktıyı üretirken hangi girdi özelliklerinin model için daha önemli olduğunu vurgular. Dikkat puanı, Denklem (2.23) kullanılarak hesaplanır.

$$s_t = \tanh(W_h h_t b_h) \tag{2.23}$$

Burada W_h ,h_t ve b_h sırasıyla AM'nin ağırlığı, giriş vektörü ve AM'nın sapmasıdır.

Dikkat puanları, modelin hangi girdi öğelerine öncelik vereceğini belirleyerek modelin kaynaklarına odaklanmasını sağlar. Yüksek dikkat puanına sahip girdi öğeleri, modelin daha fazla odaklanmasını gerektirirken, düşük dikkat puanlarına sahip olanlar daha az önemli olarak kabul edilir. Bu şekilde, model, dikkat puanlarını kullanarak dikkatini önemli bilgilere odaklayabilir ve gürültülü, önemsiz veya gereksiz bilgilere daha az dikkat edebilir.

Dikkat Ağırlığı (Attention Weight) : Belirli bir girdi öğesinin veya bileşeninin ne kadar önemli olduğunu gösteren Dikkat ağırlığı değeri Denklem (2.24) kullanılarak hesaplanır.

$$\mathrm{a_t} = \mathrm{softmax}\big(\mathrm{s(h_t, Q)}\big) = \frac{\exp(\mathrm{s(h_t,Q)})}{\sum_{\mathrm{t}} \exp(\mathrm{s(h_t,Q)})'} \tag{2.24}$$

Dikkat ağırlıkları, genellikle softmax fonksiyonu kullanılarak hesaplanır. Her bir ağırlık 0 ile 1 arasında bir değer alır ve tüm ağırlıkların toplamı 1'e eşittir. Bu, dikkat mekanizmasının bir girdinin farklı bileşenlerine öncelikler atamasına ve daha önemli bileşenlere daha büyük ağırlıklar vermesine olanak tanır.

Bağlam Vektörü (Context vector) : Bağlam vektörü, modelin dikkatini en yüksek ağırlığa sahip girdi öğelerine odaklamasına ve önemli bilgileri daha etkili bir şekilde yakalamasına yardımcı olur. Bağlam vektörü, Denklem (2.25)'de gösterildiği gibi, girdi öğeleri tarafından ağırlıklandırılan dikkat ağırlıklarının toplanmasıyla elde edilir.

$$c = \sum_t a_t h_t \tag{2.25}$$

2.5 Hiperparametre Optimizasyonu

Hiperparametreler, modelin öğrenme süreci boyunca önceden ayarlanmış olan ve eğitilen modelin doğrudan öğrenmediği parametrelerdir. ML algoritmalarında hiperparametre seçimi, modelin performansını optimize etmek için önemli bir adımdır (Yang ve Shami, 2020). Hiperparametre seçimi genellikle çeşitli arama stratejileri ile gerçekleştirilir. Bu stratejiler, en uygun hiperparametre kombinasyonlarını bulmaya yönelik optimizasyon teknikleridir. En yaygın yöntemler, Grid (Izgara) Arama, Rastgele Arama ve Bayes Optimizasyonu yöntemidir.

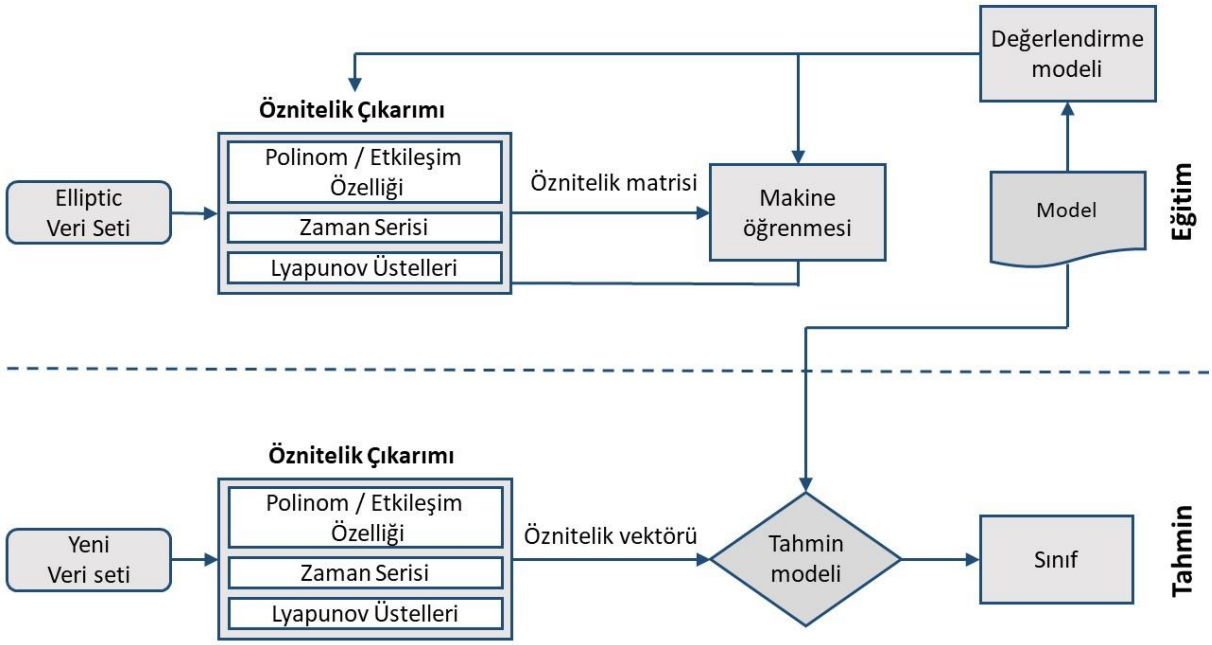
Grid arama yöntemi, hiperparametreler için belirli değer aralıkları oluşturur ve bu aralıklar içindeki tüm olası kombinasyonları dener (Bergstra ve diğ., 2011). Amaç, her kombinasyonu test ederek modelin en iyi performansı gösterdiği hiperparametreleri bulmaktır. Bu yöntem, tüm kombinasyonları denediği için kapsamlıdır. Ancak zaman ve kaynak açısından maliyetlidir. Rastgele arama ise belirtilen hiperparametre aralıklarından rastgele örnekler seçmeyi ve bu hiperparametre kombinasyonlarını test ederek her kombinasyon için modelin performansını değerlendirmeyi içerir (Bergstra ve Bengio, 2012). Bayes optimizasyon yöntemi ise rastgele ve grid arama optimizasyon tekniklerinden farklı olarak, geçmiş denemelerden elde edilen bilgileri kullanarak yeni denemeleri daha bilinçli bir şekilde seçmeyi amaçlar. Böylece, hiperparametre arama sürecini daha verimli hale getirir ve gereksiz hesaplama maliyetlerini azaltır.

3. GELİŞTİRİLEN MODELLER

Bu bölümde, blok zinciri işlem ağlarında yasa dışı para transferlerinin tespitine yönelik geliştirilen yöntemler açıklanmıştır. Çalışmada, işlem graflarından elde edilen öznitelikler kaotik zaman serilerine dönüştürülmüş ve LEs hesaplanarak yeni kaotik öznitelikler elde edilmiştir. Bu öznitelikler hem ML algoritmaları hem de graf temelli DL modeli ile sınıflandırma sürecinde kullanılmıştır. Ayrıca, graf verisinden elde edilen öznitelikler kullanılarak hibrit bir derin öğrenme modeli geliştirilmiş ve performansı diğer yaklaşımlarla karşılaştırılmıştır.

3.1. Makine Öğrenmesi Temelli Model

Bu bölümde, kara para aklama tespitinde kullanılan ML yaklaşımın genel işleyişi açıklanmıştır. ML yöntemleri kullanılarak geliştirilen kara para aklama tespit modeline ait akış şeması Şekil 3.1'de gösterilmiştir.



Şekil 3.1 ML algoritmaları için önerilen modelin akış şeması

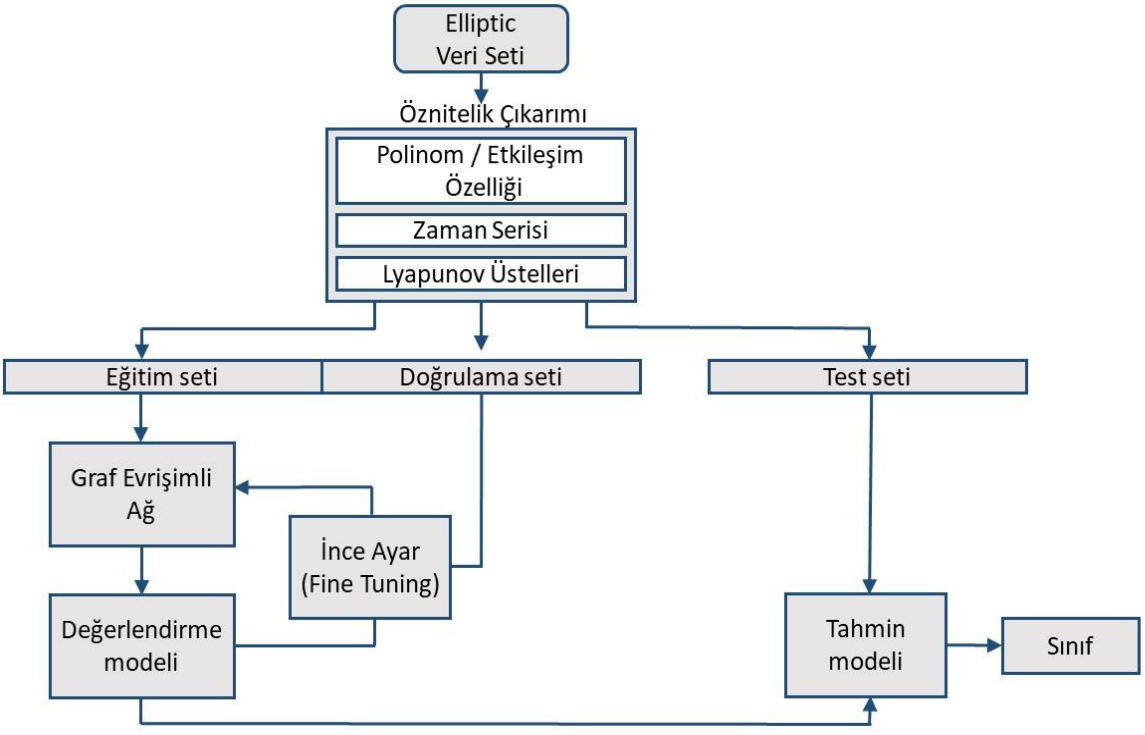
ML algoritmaları için önerilen model yaklaşımı, iki temel aşamadan oluşmaktadır: Eğitim ve tahmin. Eğitim aşamasında, ilk olarak Elliptic veri setindeki işlem düğümlerine ait öznitelikler zaman serilerine dönüştürülmüştür. Bu dönüşüm sürecinde, düğümlerin polinom ve etkileşim özellikleri kullanılarak öznitelik sayısı artırılmıştır. Elde edilen zaman serileri, dinamik sistem analizine olanak tanımak amacıyla faz uzayına aktarılmış ve bu uzayda sistemin kaotik

özelliklerini karakterize etmek için LEs hesaplanmıştır. Deneysel gözlemler doğrultusunda, faz uzayı gömme boyutu olarak $\lambda = 7$ ve $\lambda = 9$ olarak seçilmiş, daha yüksek λ değerlerinde sistemin kaotik yapısında bozulmalar tespit edilmiştir. Bu nedenle, yalnızca bu iki gömme boyutu değerlendirmeye alınmıştır. Elde edilen öznitelikler birleştirilerek öznitelik matrisi oluşturulmuş ve bu matris, çeşitli ML algoritmaları ile eğitilmiştir. Eğitim ve test verilerinin rastgele seçilmesi nedeniyle performans sonuçlarındaki değişkenliği azaltmak amacıyla k-kat çapraz doğrulama yöntemi kullanılmıştır.

Tahmin aşamasında, yeni veri seti üzerinde benzer bir öznitelik çıkarımı süreci izlenmiştir. Polinom ve etkileşim özellikleri, zaman serisi dönüşümleri ve LEs hesaplanarak öznitelik vektörü oluşturulmuştur. Bu vektör, daha önce eğitilen tahmin modeline giriş olarak verilmiş ve her düğüm sınıfı belirlenerek işlem yapısının yasal ya da yasa dışı olup olmadığı sınıflandırılmıştır.

3.2 Graf Evrişimli Ağ Temelli Model

Bu bölümde, kara para aklama tespitinde kullanılan DL tabanlı yöntemlerden biri olan GCN yaklaşımının genel işleyişi açıklanmıştır. GCN yöntemi kullanılarak geliştirilen kara para aklama tespit modeline ait akış şeması Şekil 3.2'de gösterilmiştir.



Şekil 3.2 GCN yönteminin akış şeması

GCN yönteminin uygulanmasının ilk aşamasında düğüm öznitelikleri çıkarılmıştır. Bu süreç öznitelik artırımı, zaman serilerinin elde edilmesi ve LEs hesaplanması olmak üzere üç bölüme ayrılmıştır. Bu süreçte elde edilen öznitelikler, her düğüm için bir temsil vektörü oluşturmakta ve bu vektörler, sonraki aşamalarda GCN modeli için giriş olarak kullanılmıştır. İkinci aşamada veri seti üç alt kümeye bölünmüştür: eğitim, doğrulama ve test setleri. Eğitim seti**,** modelin düğümlerine ait öznitelik ilişkilerini öğrenmesi için kullanılmıştır. Doğrulama seti, modelin aşırı öğrenmeyi önlemesi ve hiperparametre ayarlarını yapmak için kullanılmıştır. Test seti ise, modelin eğitilmeyen veriler üzerindeki genelleme yeteneğini bağımsız olarak değerlendirmek için ayrılmıştır.

GCN modeli, eğitim setinden elde edilen öznitelik vektörleri ve graf yapısı kullanılarak eğitilmiştir. Modelin doğrulama performansına göre hiperparametreler, rastgele arama yöntemiyle ayarlanmış ve bu sayede modelin hem doğruluk oranı hem de genelleme yeteneği artırılmıştır. Bu süreç, klasik ince ayar (fine tuning) yaklaşımına dayalıdır.

Eğitim tamamlandıktan sonra, tahmin modeli, test verisi üzerinde uygulanmış ve her bir düğümün ait olduğu sınıf (yasal veya yasadışı) belirlenmiştir. Bu sınıflandırma süreci, blok zinciri üzerindeki şüpheli işlemlerin tespitinde modelin etkinliğini doğrudan yansıtmaktadır.

Bu yapı sayesinde, hem düğüm özniteliklerinin zamansal ve kaotik doğası hem de ağ üzerindeki topolojik ilişkiler birlikte değerlendirilerek, yüksek doğruluk oranına sahip bir kara para aklama tespit modeli geliştirilmiştir.

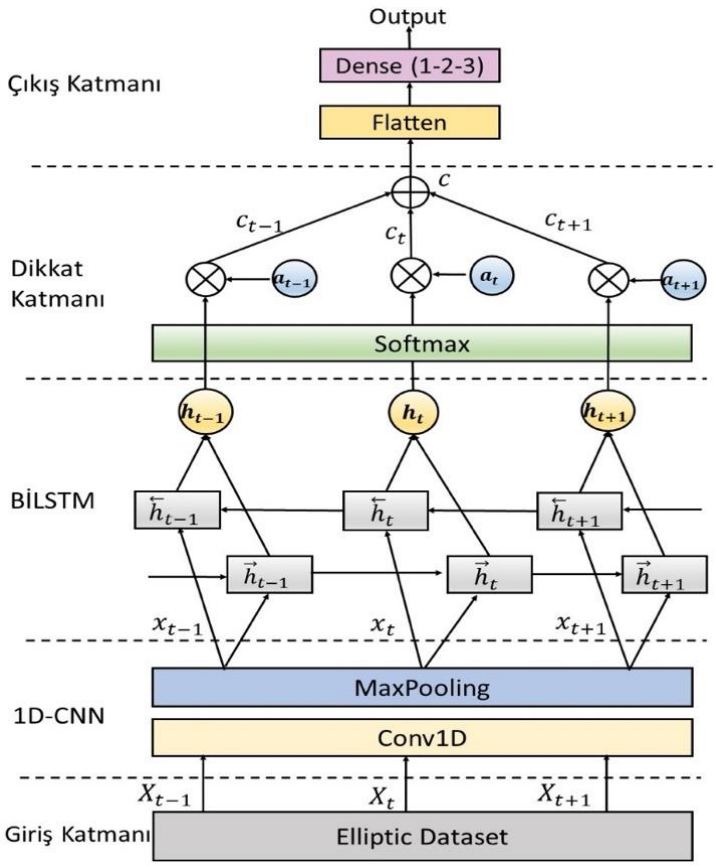
3.3 CNN-BiLSTM-AM Temelli Hibrit Model

Bu tez çalışmasında Elliptic veri setinde yasal ve yasal olmayan işlemleri tespit etmek için CNN, BiLSTM ve AM'nin özelliklerini birleştiren CNN-BiLSTM-AM hibrit modeli geliştirilmiştir. CNN-BiLSTM-AM modelini yapısı Şekil 3.3'de gösterilmiştir. Bu model giriş katmanı, CNN katmanı, BiLSTM katmanı, AM katmanı ve çıkış katmanından oluşmaktadır.

CNN-BİSTM-AM yönteminin her bir bloğu şu şekildedir:

Giriş Katmanı, Elliptic veri setinden elde edilen kripto para işlemlerini temsil eder. İşlemler zaman serisi olarak ifade edilir ve belirli bir zaman dilimi boyunca X_{t-1}, X ve X_{t+1} olarak gösterilir.

1D-CNN, zaman serisi verilerinden öznitelik çıkarmak için kullanılır. Girdi verisine kaydırmalı filtreler uygulanarak anlamlı öznitelikler elde edilir. Bu aşamadan sonra, çıkarılan özniteliklerin daha özet ve temsil gücü yüksek biçimde elde edilmesi için MaxPooling işlemi gerçekleştirilir.



Şekil 3.3 CNN-BİLSTM-AM yapısı

BiLSTM, zaman serisi verilerindeki sıralı bağımlılıkları yakalamak için kullanılır. Her bir zaman adımında $(t-1, t, t+1)$ elde edilen çıktılar, hem geçmiş hem de gelecek bilgilere dayalı olarak işlenir. $\vec{h}$ *ve* $\overleftarrow{h}_t$, sırasıyla ileri ve geri yönlerde işlenen gizli durumları temsil eder. Bu çift yönlü işleme sayesinde, verilerin her iki yöndeki bağlamının yakalanması amaçlanmıştır.

Dikkat Katmanı, BiLSTM bloğunun sonuna eklenmiştir. BiLSTM çıktıları üzerinde dikkat mekanizması uygulanarak her bir gizli duruma farklı önem dereceleri atanır. Bu mekanizma, belirli zaman adımlarındaki gizli durumlara ağırlık katsayıları (a_{t-1}, a_t, a_{t+1}) atar ve bu ağırlıklar yardımıyla en bilgilendirici kısımlar ön plana çıkarılır. Sonuç olarak, işlemler arasındaki en kritik bilgileri özetleyen bağlam vektörü (c) elde edilir.

Çıkış Katmanı'nda, AM tarafından oluşturulan bağlam vektörleri düzleştirilerek tek bir vektöre dönüştürülür. Bu vektör, son sınıflandırma işlemini gerçekleştiren bir veya birden fazla tam bağlı katmana (fully connected layers) aktarılır. Çıktı katmanı, işlem verilerini ilgili sınıflara ayırmayı sağlar.

4. DENEYSEL ÇALIŞMALAR

4.1. Performans Değerlendirilmesi

Eğitim seti kullanılarak oluşturulan bir modelin performansını değerlendirmek için eğitim aşamasında kullanılmayan ve modelin daha önce hiç karşılaşmadığı verilerden oluşan bir test seti kullanılır. Test seti, modelin gerçek dünya verileri karşısında ne kadar başarılı olduğunu ölçmek amacıyla kullanılır. Model, test setindeki verilerin etiketlerini doğru bir şekilde tahmin edebiliyorsa, bu durum modelin genelleme yeteneğinin yüksek olduğunu ve eğitim verileri dışındaki verilere de uyum sağlayabildiğini gösterir. Modelin başarısını detaylı bir şekilde analiz etmek için çeşitli performans metrikleri kullanılır. Bu metrikler, modelin eğitim sürecinde öğrendiği bilgileri test verileri üzerinde ne kadar etkili bir şekilde uygulayabildiğini ölçer. Sınıflandırma modelini değerlendirmek için performans metrikleri karmaşıklık matrisinden elde edilir.

Karmaşıklık matrisi (confusion matrix), sınıflandırma modellerinin performansını değerlendirmek için kullanılan bir görselleştirme tekniğidir. Bu matris, gerçek ve tahmin edilen sınıflar arasındaki ilişkiyi gösterir ve doğru pozitif (True Positive, TP), yanlış pozitif (False Positive, FP), doğru negatif (True Negative, TN) ve yanlış negatif (False Negative, FN) terimlerini içerir. TP, doğru bir şekilde sınıflandırılmış pozitif örneklerin sayısını; FP, yanlış bir şekilde sınıflandırılmış pozitif örneklerin sayısını; TN, doğru bir şekilde sınıflandırılmış negatif örneklerin sayısını; FN, yanlış bir şekilde sınıflandırılmış negatif örneklerin sayısını ifade eder. Karmaşıklık matrisi, modelin hangi sınıflarda daha başarılı olduğunu ve hangi hataları yaptığını görselleştirerek, performans metriklerinin hesaplanmasına temel oluşturur. Şekil 4.1'de örnek bir karmaşıklık matrisi verilmiştir.

		Tahmin Edilen Sınıf	
		Pozitif	**Negatif**
Gerçek Sınıf	**Pozitif**	**Doğru Pozitif (TP)**	**Yanlış Negatif (FN)**
	Negatif	**Yanlış Pozitif (FP)**	**Doğru Negatif (TN)**

Şekil 4.1 Karmaşıklık matrisi

Doğruluk (Accuracy): Modelin tüm tahminler içinde ne kadarının doğru olduğunu gösterir. Doğru sınıflandırılmış pozitif ve negatif örneklerin sayısının toplam sınıflandırılmış örnek sayısına bölünmesiyle hesaplanır. Doğruluk değeri Denklem (4.1)'de gösterilmiştir.

$$Doğruluk = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Kesinlik (Precision): Kesinlik, bir sınıflandırma modelinin pozitif tahminlerinin ne kadarının doğru olduğunu ölçen bir performans metriğidir. Doğru pozitiflerin sayısının, doğru pozitifler ve yanlış pozitiflerin toplamına bölünmesiyle hesaplanır. Kesinlik değeri Denklem (4.2)'de gösterilmiştir.

$$Kesinlik = \frac{TP}{TP + FP} \tag{4.2}$$

Duyarlık (Recall): Duyarlık, bir sınıflandırma modelinin gerçek pozitifleri ne kadar iyi tahmin ettiğini ölçen bir performans metriğidir. Doğru pozitif tahminlerin gerçek pozitif örneklerin toplamına bölünmesiyle hesaplanır. Duyarlık değeri Denklem (4.3)'de gösterilmiştir.

$$Duyarlık = \frac{TP}{TP + FN} \tag{4.3}$$

F1-skor (F1-score): F1-skor, kesinlik ve duyarlık ölçümlerinin harmonik ortalamasıdır. F1-skoru, kesinlik ve duyarlık arasındaki dengeyi anlamak için kullanılır. F1-skor değeri Denklem (4.4)'de gösterilmiştir.

$$F1 = 2 * \frac{Kesinlik * Duyarlık}{Kesinlik + Duyarlık} \tag{4.4}$$

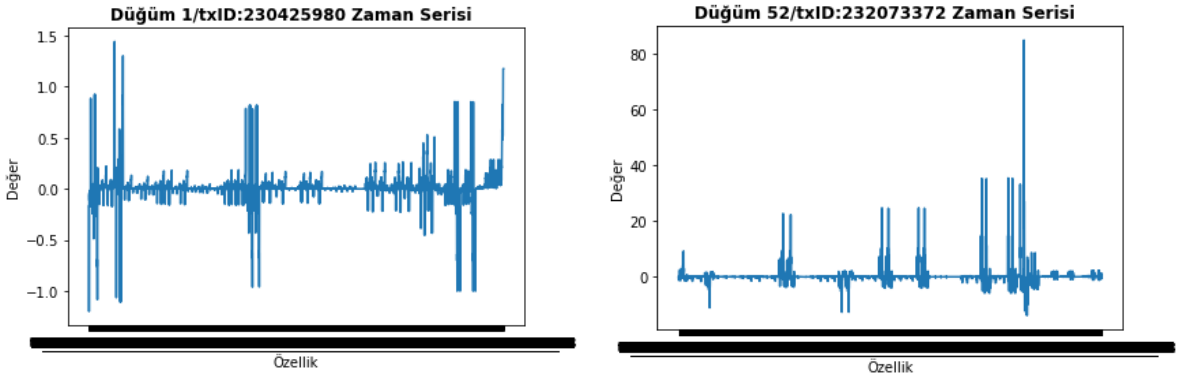
ROC Eğrisi ve Eğri Altında Kalan Alan: Receiver Operating Characteristic (ROC) eğrisi ve ROC eğrisi altında kalan alan (Area Under the Curve - AUC), sınıflandırma problemlerinde kullanılan değerlendirme metrikleridir.

PR Eğrisi ve Eğri Altında Kalan Alan: Kesinlik-duyarlık (precision-recall (PR)) eğrisi farklı eşik değerlerinde kesinlik ile duyarlık arasındaki dengeyi gösterir. Bu eğri, özellikle dengesiz veri kümeleri için oldukça yararlıdır. PR eğrisinin altındaki alan (PR-AUC), modelin pozitif sınıfları doğru bir şekilde tanımlama yeteneğini ölçer.

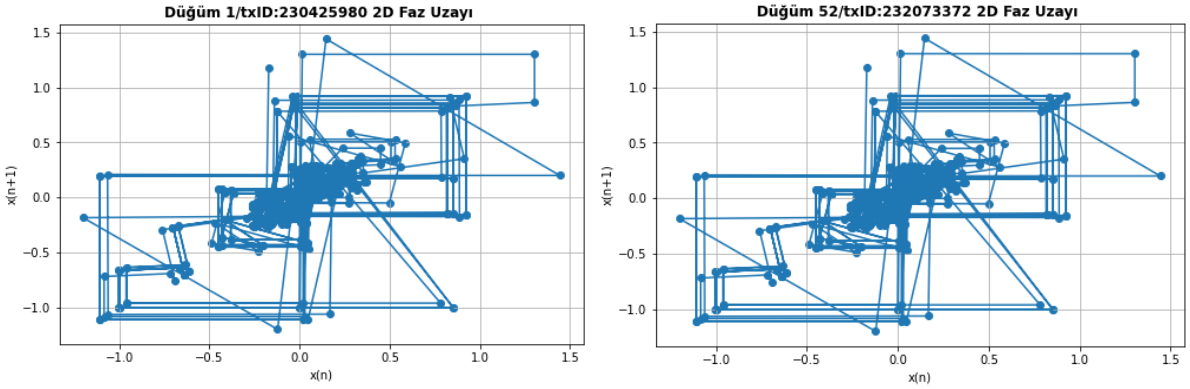
ROC eğrisi modelin genel ayırt edici kapasitesini değerlendirirken, PR eğrisi özellikle sınıf dengesizliği bulunan durumlarda performansın değerlendirilmesinde avantaj sağlar.

4.2 Faz Uzayı ve Lyapunov Üstellerinin Elde Edilmesi

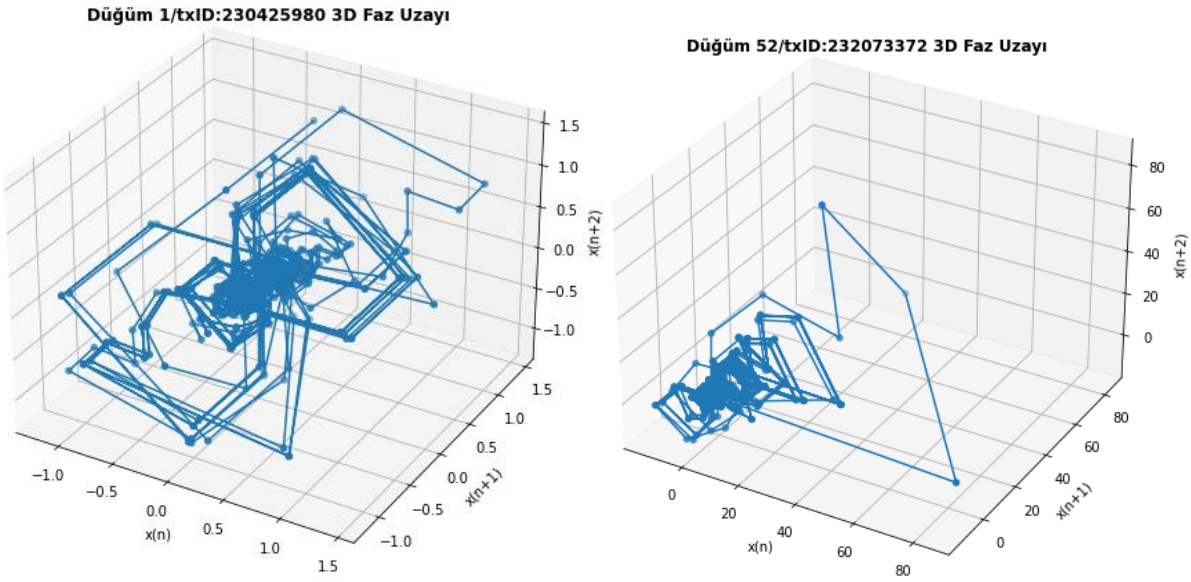
Sistemdeki değişkenlerin fazla sayıda olması, ortamın kaotik olmasına neden olan temel unsurdur. Ayrıca, kaotik sistemler genellikle dağılmayı engellemek için sürekli bir enerji girişine de ihtiyaç duyarlar. Bir sistemin kaotik olup olmadığını belirleyebilmek için, sistemin davranışını uzun süre boyunca izleyen bir değişkenler kaydına sahip olmamız gerekir. Bu doğrultuda, çalışmada 93 öznitelik değerine polinom ve etkileşim özellikleri yöntemi uygulanarak, öznitelik sayısı artırılmıştır. Graf yapısında bulunan düğümlere ait öznitelikler zaman serilerine dönüştürülmüştür. Zaman serilerinin kaotik özelliklerini analiz edebilmek için, bu serileri faz uzayına aktarıldı ve LEs elde edilmiştir. Bu değerler düğümlere ait yeni öznitelik vektörleri olarak kullanılmıştır. Şekil 4.2 düğümlere ait özniteliklerin zaman serilerini, Şekil 4.3 bu zaman serilerin elde edilen 2 boyutlu faz uzaylarını, Şekil 4.4 ise 3 boyutlu faz uzaylarını gösterilmiştir.



Şekil 4.2 Zaman serileri



Şekil 4.3 2 boyutlu faz uzayı



Şekil 4.4 3 boyutlu faz uzayı

4.3 Makine Öğrenmesi Yöntemleri ile Elde Edilen Bulgular

Elliptic veri setine uygulanan ML algoritmaları için grid arama yöntemi kullanılarak polinom özellikleri için elde edilen en iyi parametreler Tablo 4.1'de, etkileşim özellikleri için elde edilen en iyi parametreler Tablo 4.2'de gösterilmiştir. Polinom ve etkileşim özelliği ile özniteliği artırılan veri setine ait performans sonuçları sırasıyla Tablo 4.3 ve Tablo 4.4'de gösterilmiştir.

Tablo 4.3 ve Tablo 4.4'de, farklı ML modellerinin (SVM, k-NN, RF, NB, LR, MLP, XGBoost ve LightGBM) iki farklı öznitelik mühendisliği yöntemi (polinom ve etkileşim özellikleri) ile değerlendirilmesi sonucunda elde edilen performans metrikleri verilmiştir. Bu metrikler arasında doğruluk, kesinlik, duyarlık ve F1-skoru yer almakta olup, LEs değerine göre sonuçlar incelenmiştir. Tablo 4.3 ve Tablo 4.4'de modellerin yüksek doğruluk oranlarına sahip olduğu ve %90'ın üzerinde bir performans sergilediği gösterilmiştir.

Polinom özellikleri kullanıldığında k-NN ve LightGBM yöntemleri, özellikle kesinlik ve F1-skoru açısından diğer modellere göre daha iyi performans göstermiştir. Örneğin, k-NN modeli %86,5 kesinlik ve %86,1 F1-skoru ile bu metriklerde öne çıkarken, LightGBM modeli %86,6 kesinlik ve %86,2 F1-skoru ile benzer bir başarı elde etmiştir. Buna ek olarak, XGBoost sınıflandırıcısı Lyapunov üstel değeri **7** olduğunda %86,8 F1-skoru ile dikkat çekmiş, ancak üstel değeri 9 olduğunda diğer modellerle benzer seviyede performans göstermiştir. SVM, RF, NB, LR ve MLP modellerinde ise tüm metriklerde genel olarak dengeli ve tutarlı sonuçlar elde edilmiştir.

Tablo 4.1 Polinom özelliği için en iyi sınıflandırma modeli parametre değerleri

Yöntem	**Hiperparametreler için arama alanı**	**En iyi parametre değerleri polinom özellikleri / LEs 7**	**En iyi parametre değerleri polinom özellikleri / LEs 9**
SVM	'C': np.logspace(-3, 3, 7) 'kernel': ['linear', 'rbf']	'C': 0.001 'kernel': linear	'C': 0.001 'kernel': linear
k-NN	'n_neighbors': np.arange (1, 50)	'n_neighbors': 47	'n_neighbors': 35
RF	'n_estimators': [200, 500], 'max_features': ['auto', 'sqrt','log2'], 'max_depth': [4, 5, 6, 7, 8]	'n_estimators': 500 'max_features': 'auto', 'max_depth': 8,	'n_estimators': 200 'max_features': 'auto', 'max_depth': 4,
NB	'var_smoothing': np.logspace(0, -9, num = 100)	var_smoothing': 1.0	var_smoothing': 1.0
LR	'C': np.logspace(-3, 3, 7), 'max_iter': [100, 200, 300], 'penalty': ['l1', 'l2'], 'random_state': [42]	'C': 0.001 'max_iter': 100, 'penalty': l1 'random_state': [42]	'C': 0.001 'max_iter': 100, 'penalty': l1 'random_state': [42]
MLP	'solver': ['lbfgs'], 'max_iter': [1000, 1100, 1200, 1300, 1400, 1500, 1600, 1700, 1800, 1900, 2000], 'alpha': 10.0 **-np.arange(1, 10), 'hidden_layer_sizes': np.arange(10,15), 'random_state': [0, 1, 2, 3, 4, 5, 6,7, 8, 9]	solver': 'lbfgs' 'max_iter': 1900, 'alpha': 0.001, 'hidden_layer_sizes': 11, 'random_state': 1,	'solver':'lbfgs' 'max_iter': 1000, 'alpha': 0.1, 'hidden_layer_sizes': 10, 'random_state': 0,
XGBoost	'n_estimators': [400, 700, 1000], 'max_depth': [15, 20, 25], 'reg_lambda': [1.1, 1.2, 1.3], 'reg_alpha': [1.1, 1.2, 1.3], 'subsample': [0.7, 0.8, 0.9], 'colsample_bytree': [0.7, 0.8]	'n_estimators': 400, 'max_depth': 20, 'reg_lambda': 1.2, 'reg_alpha': 1.1, 'subsample': 0.9, 'colsample_bytree': 0.8	'n_estimators': 400, 'max_depth': 20, 'reg_lambda': 1.2, 'reg_alpha': 1.1, 'subsample': 0.9, 'colsample_bytree': 0.8
LightGBM	'n_estimators': [200, 600, 1000], 'colsample_bytree': [0.5, 0.6], 'max_depth': [10, 15, 20], 'num_leaves':[50,100,200], 'reg_alpha':[1.1,1.2, 1.3], 'reg_lambda': [1.1, 1.2, 1.3], 'min_split_gain':[0.4,0.5], 'subsample': [0.6, 0.7, 0.8], 'subsample_freq': [15]	'n_estimators': 200, 'colsample_bytree': 0.5 'max_depth': 10, 'num_leaves': 50, 'reg_alpha': 1.3, 'reg_lambda': 1.2, 'min_split_gain': 0.5, 'subsample': 0.8, 'subsample_freq': 15	'n_estimators': 200, 'colsample_bytree': 0.5 'max_depth': 10, 'num_leaves': 50, 'reg_alpha': 1.3, 'reg_lambda': 1.2, 'min_split_gain': 0.5, 'subsample': 0.8, 'subsample_freq': 15

Tablo 4.2 Etkileşim özelliği için en iyi sınıflandırma modeli parametre değerleri

Yöntem	Hiperparametreler için arama alanı	En iyi parametre değerleri etkileşim özellikleri / LEs 7	En iyi parametre değerleri etkileşim özellikleri / LEs 9
SVM	'C': np.logspace(-3, 3, 7) 'kernel': ['linear', 'rbf']	'C': 0.001 'kernel': linear	'C': 0.001 'kernel': linear
k-NN	'n_neighbors': np.arange (1, 50)	'n_neighbors': 34	'n_neighbors': 33
RF	'n_estimators': [200, 500], 'max_features': ['auto', 'sqrt','log2'], 'max_depth': [4, 5, 6, 7, 8]	'n_estimators': 200 'max_features': 'auto', 'max_depth': 4,	'n_estimators': 200 'max_features': 'auto', 'max_depth': 4,
NB	'var_smoothing': np.logspace(0, -9, num = 100)	var_smoothing': 1.0	var_smoothing': 1.0
LR	'C': np.logspace(-3, 3, 7), 'max_iter': [100, 200, 300], 'penalty': ['l1', 'l2'], 'random_state': [42]	'C': 0.001 'max_iter': 100, 'penalty': l1 'random_state': [42]	'C': 0.001 'max_iter': 100, 'penalty': l1 'random_state': [42]
MLP	'solver': ['lbfgs'], 'max_iter': [1000, 1100, 1200, 1300, 1400, 1500, 1600, 1700, 1800, 1900, 2000], 'alpha': 10.0 **-np.arange(1, 10), 'hidden_layer_sizes': np.arange(10,15), 'random_state': [0, 1, 2, 3, 4, 5, 6,7, 8, 9]	'solver': 'lbfgs' 'max_iter': 1000, 'alpha': 0.1, 'hidden_layer_sizes': 10 'random_state': 0,	'solver': 'lbfgs' 'max_iter': 1000, 'alpha': 0.1, 'hidden_layer_sizes': 10, 'random_state': 0,
XGBoost	'n_estimators': [400, 700, 1000], 'max_depth': [15, 20, 25], 'reg_lambda': [1.1, 1.2, 1.3], 'reg_alpha': [1.1, 1.2, 1.3], 'subsample': [0.7, 0.8, 0.9], 'colsample_bytree': [0.7, 0.8]	'n_estimators': 400, 'max_depth': 25, 'reg_lambda': 1.1, 'reg_alpha': 1.1, 'subsample': 0.9, 'colsample_bytree':0.7	'n_estimators': 400, 'max_depth': 25, 'reg_lambda': 1.1, 'reg_alpha': 1.1, 'subsample': 0.9, 'colsample_bytree': 0.8
LightGBM	'n_estimators': [200, 600, 1000], 'colsample_bytree': [0.5, 0.6], 'max_depth': [10, 15, 20], 'num_leaves':[50,100,200], 'reg_alpha':[1.1,1.2, 1.3], 'reg_lambda': [1.1, 1.2, 1.3], 'min_split_gain':[0.4,0.5], 'subsample': [0.6, 0.7, 0.8], 'subsample_freq': [15]	'n_estimators': 200 'colsample_bytree':0.5 'max_depth': 15 'num_leaves': 50 'reg_alpha': 1.3 'reg_lambda': 1.3, 'min_split_gain': 0.5, 'subsample':0.8, 'subsample_freq': 15	'n_estimators': 200, 'colsample_bytree':0.5 'max_depth': 15, 'num_leaves': 50, 'reg_alpha': 1.3, 'reg_lambda': 1.2, 'min_split_gain': 0.5, 'subsample': 0.8, 'subsample_freq': 15

Tablo 4.3 Polinom özelliği ile özniteliği artırılan veri seti üzerinde ML yöntemlerinin performans sonuçları

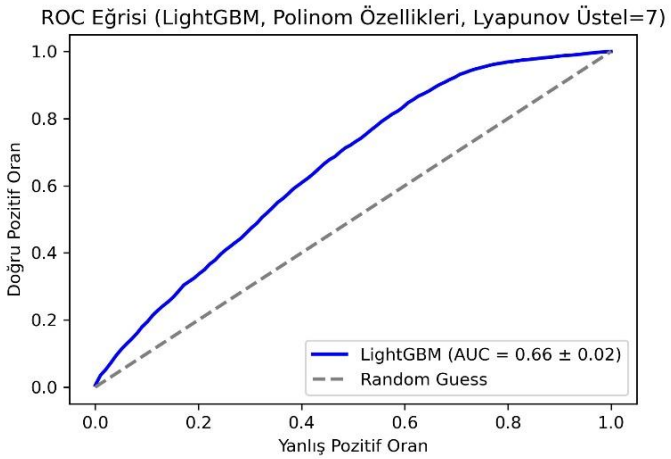
Yöntem	Lyapunov Üstel Değeri	Doğruluk (%)	Kesinlik (%)	Duyarlık (%)	F1-skor (%)	AUC	PR-AUC
SVM	7	90,2	81,4	90,2	85,6	0.54	0.91
	9	90,2	81,4	90,2	85,6	0.55	0.91
k-NN	7	90,2	86,5	90,2	86,1	0.65	0.93
	9	90,2	82,4	90,2	85,6	0.59	0.92
RF	7	90,2	82,9	90,2	85,6	0.65	0.94
	9	90,2	81,4	90,2	85,6	0.59	0.92
NB	7	90,2	81,4	90,2	85,6	0.59	0.92
	9	90,2	81,4	90,2	85,6	0.54	0.91
LR	7	90,2	81,4	90,2	85,6	0.54	0.92
	9	90,2	81,4	90,2	85,6	0.49	0.90
MLP	7	90,2	81,4	90,2	85,6	0.63	0.93
	9	90,2	81,4	90,2	85,6	0.59	0.92
XGBoost	7	89,7	85,9	89,7	86,8	0.63	0.93
	9	90,2	81,4	90,2	85,6	0.56	0.92
LightGBM	7	90,2	86,6	90,2	86,2	0.66	0.94
	9	90,1	81,3	90,1	85,5	0.58	0.92

Tablo 4.4 Etkileşim özelliği ile özniteliği artırılan veri seti üzerinde ML yöntemlerinin performans sonuçları

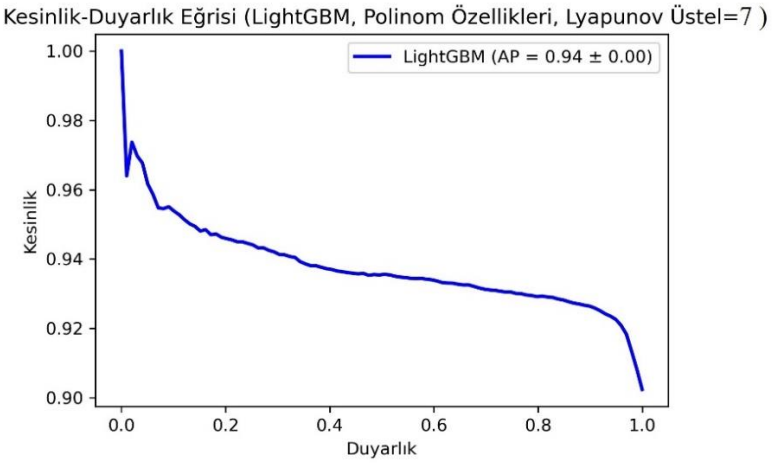
Yöntem	Lyapunov Üstel Değeri	Doğruluk (%)	Kesinlik (%)	Duyarlık (%)	F1-skor (%)	AUC	PR-AUC
SVM	7	90,2	81,4	90,2	85,6	0.49	0.91
	9	90,2	81,4	90,2	85,6	0.50	0.91
k-NN	7	90,2	85,6	90,2	85,6	0.59	0.92
	9	90,2	81,4	90,2	85,6	0.58	0.92
RF	7	90,2	81,4	90,2	85,6	0.59	0.92
	9	90,2	81,4	90,2	85,6	0.58	0.92
NB	7	90,2	81,4	90,2	85,6	0.55	0.91
	9	90,2	81,4	90,2	85,6	0.54	0.91
LR	7	90,2	81,4	90,2	85,6	0.50	0.91
	9	90,2	81,4	90,2	85,6	0.53	0.91
MLP	7	90,2	81,4	90,2	85,6	0.58	0.92
	9	90,2	81,4	90,2	85,6	0.59	0.92
XGBoost	7	89,6	83,7	89,6	85,7	0.56	0.92
	9	90,2	81,4	90,2	85,6	0.57	0.92
LightGBM	7	90,2	84,6	90,2	85,6	0.59	0.92
	9	90,2	81,4	90,2	85,5	0.58	0.92

Etkileşim özellikleri ile yapılan analizde de doğruluk oranları genelde %90'ın üzerinde olup polinom özelliklerine benzer bir başarı gözlemlenmiştir. Ancak bu öznitelik mühendisliği yönteminde de k-NN ve LightGBM kesinlik değerinde daha güçlü bir performans sergilemiştir. Diğer modeller (SVM, RF, NB, LR, MLP) etkileşim özellikleri ile polinom özelliklerine benzer şekilde dengeli bir performans göstermiştir.

Hem polinom hem de etkileşim yöntemlerine göre özniteliği artırılan veriler üzerinde pozitif sınıfın performansını gösteren kesinlik ve duyarlık metrikleri açısından sınıflandırıcıların performansındaki farklılıklar çok küçüktür. LightGBM modeli doğruluk, kesinlik, duyarlık, AUC ve PR-AUC değerlerinde diğer modellere göre daha yüksek performans sergileyerek genel olarak en iyi sonuçları vermiştir. Şekil 4.5 ve Şekil 4.6'da ML yöntemlerinden LightGBM'a (polinom özelliği, LEs=7) ait ROC ve PR eğrileri gösterilmiştir. Ek A.1 ve Ek A.2'da diğer ML modellerine ait ROC ve PR eğrileri verilmiştir.



Şekil 4.5 LightGBM yöntemine ait ROC eğrisi



Şekil 4.6 LightGBM yöntemine ait PR eğrisi

4.4 Graf Evrişimli Ağ Yöntemi ile Elde Edilen Bulgular

GCN yöntemi ile gerçekleştirilen deneysel çalışmalarda elde edilen bulgular bu bölümde verilmiştir. Modelin eğitimi sürecinde, rastgele arama yöntemi kullanılmış ve hiperparametreler aşağıdaki aralıklar içerisinde belirlenmiştir

- Katman sayısı (number of layers): [2, 3, 4]
- Gizli boyut (hidden dimension): [64, 128, 256]
- Öğrenme oranı (learning rate): [0.01, 0.001, 0.0001]
- Ağırlık azaltma (weight decay): [1e-4, 1e-5, 1e-6]
- Devir (Epoch) sayısı: [100, 300, 500]

En iyi performansı sağlayan hiperparametre kombinasyonu Tablo 4.5'de verilmiştir.

Tablo 4.5 GCN model parametreleri

	LEs	**Öznitelik mühendisliği**	**Katman sayısı**	**Gizli boyut**	**Öğrenme oranı**	**Ağırlık azaltma**	**Devir sayısı**
Model_1	7	Polinom özellikleri	2	64	0,01	1e-5	300
Model_2	9	Polinom özellikleri	2	64	0,01	1e-5	300
Model_3	7	Etkileşim özellikleri	2	64	0,01	1e-5	300
Model_4	9	Etkileşim özellikleri	3	128	0,01	1e-6	300

GCN yönteminde kullanılan ortak parametre değerleri Tablo 4.6'da verilmiştir.

Tablo 4.6 GCN ortak parametre değerleri

Parametre Sınıfı	**Parametre Adı**	**Parametre Değeri**
Model parametresi	Aktivasyon fonksiyonu	ReLU
	Seyreltme (dropout)	0.5
Sınıflandırıcı parametresi	Veri sınıfı sayısı (n-classes)	2
	Kayıp fonksiyonu (loss function)	Focal Loss
	Optimizasyon algoritması	Adam
Eğitim parametresi	Eğitim seti oranı	80%
	Doğrulama seti oranı	10%
	Test seti oranı	10%

Model, verilerdeki karmaşık ilişkileri ve etkileşimleri yakalamak için tasarlanmış graf evrişim katmanlarından oluşmaktadır. Bu katmanlar, modelin derinliği ile hesaplama verimliliği arasında bir denge sağlamaktadır. Öğrenme oranı, bir modelin yeni bilgileri ne kadar hızlı öğrendiğini belirler (Jeon ve diğ., 2018). Modelde, karmaşık ilişkileri daha iyi yakalayabilmek için ReLU (Rectified Linear Unit) aktivasyon fonksiyonu kullanılmıştır. Focal Loss, sınıf dengesizliği durumlarında veya modelin kolay sınıflara aşırı odaklanmasını önlemek için kullanılan bir kayıp fonksiyonudur (Lin ve diğ., 2017). Bu fonksiyonun temel amacı, kolay sınıflara verilen ödülü azaltarak zor sınıflara daha fazla ağırlık vermek ve böylece modelin zor örneklerden daha fazla öğrenmesini sağlamaktır (Wang ve diğ., 2021). Özellikle, pozitif ve negatif örnekler arasındaki dengenin bozuk olduğu sınıflandırma problemlerinde etkilidir.

Focal Loss denklemi Denklem (4.5)'de gösterilmiştir.

$$FL(p_t) = -a_t(1-p_t)^{\gamma}\log(p_t) \tag{4.5}$$

Burada:

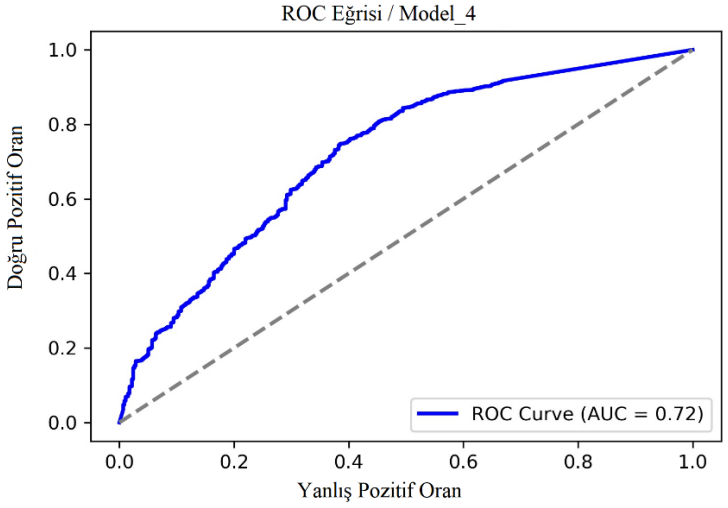
- p_t: Doğru sınıfın olasılığıdır.
- γ: Pozitif ve negatif örnekler arasındaki dengesizliği dengelemek için kullanılan ağırlık faktörüdür.
- a_t: Kolay örneklerin etkisini azaltarak zor örneklere daha fazla odaklanmayı sağlayan parametredir.

Model eğitimi için, büyük veri kümelerinin verimli bir şekilde işleyebilmesi ve adaptif öğrenme oranı yetenekleri nedeniyle Adam optimizasyon algoritması kullanılmıştır. (Kingma, 2014). Adam, AdaGrad ve RMSProp algoritmalarının avantajlarını birleştirerek, güçlü ve hızlı yakınsama (convergence) sağlayan bir yöntemdir. Ayrıca, ağırlık azaltma parametresi ile bir düzenleme (regularization) tekniği uygulanarak modelin karmaşıklığı kontrol edilmiş ve aşırı öğrenme riski azaltılmıştır. Tablo 4.7'de, GCN modelinin performans sonuçları verilmiştir.

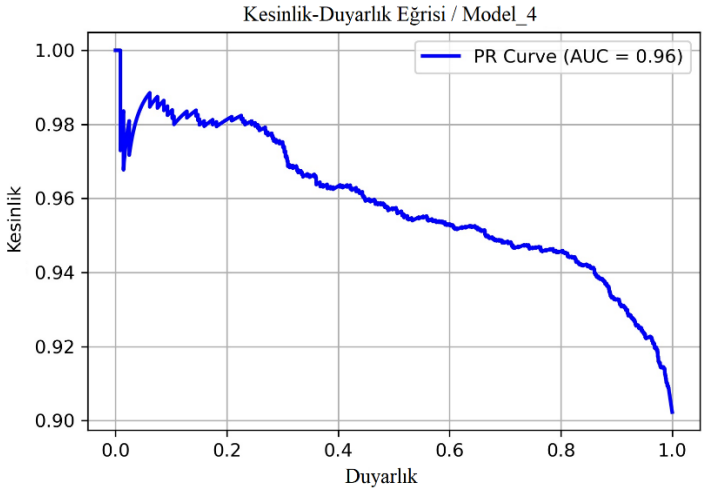
Tablo 4.7 GCN modelinin performans sonuçları

Önerilen Model	Doğruluk (%)	Kesinlik (%)	Duyarlık (%)	F1-skor (%)	AUC	PR-AUC
Model_1	85,3	92,4	91,7	92,1	0.72	0.95
Model_2	84,9	92,5	90,6	91,5	0.73	0.95
Model_3	83,7	92,3	89,3	90,8	0.72	0.95
Model_4	86,2	92,5	92,1	92,3	0.72	0.96

Önerilen modeller arasında Model_4**,** %92,5 kesinlik, %92,1 duyarlılık, %92,3 F1-skoru ve %86,2 doğruluk ile en yüksek performans elde edilmiştir. Bu sonuçlar, LEs değeri dokuz ve etkileşim özellikleri yönteminin kullanılmasının**,** yanlış pozitif ve yanlış negatif oranlarını en aza indirirken modelin kara para aklama faaliyetlerini tespit etme yeteneğini önemli ölçüde artırdığını göstermektedir. Model_1**,** LEs değeri yedi ve polinom özellikleri teknikleri kullanılarak %85,3 doğruluk ve %92,1 F1-skoru elde edilmiş ve Model_4'e göre biraz daha düşük bir performans sergilemiştir. Model_2, daha yüksek boyutluluğa sahip olmasına (LEs = 9) ve polinom özellikleri kullanmasına rağmen %84,9 doğruluk ve %91,5 F1-skoru ile sınırlı bir iyileşme göstermiştir. Model_3**,** LEs değeri yedi ile etkileşim özellikleri tekniklerini kullanarak %83,7 doğruluk ve %90,8 F1 skoru elde edilmiştir. Bu sonuçlar, tüm modellerin iyi performans gösterdiğini ancak Model_4'te kullanılan LEs değeri dokuz ve etkileşim özellikleri tekniklerinin, yasa dışı işlemleri daha yüksek doğrulukla sınıflandırmada en etkili yaklaşım olduğunu ortaya koymaktadır. Şekil 4.7 ve Şekil 4.8'de Model_4 ait ROC ve PR eğrileri gösterilmiştir. Ek A.3 ve Ek A.4'de diğer GCN modellerine ait ROC ve PR eğrileri verilmiştir. Tüm modellerin 0.72 ile 0.73 aralığında AUC değerlerine sahip olduğu gözlemlenmiştir; bu da modellerin iyi ayrıştırma performansı sergilediğini göstermektedir. AUC değerlerindeki küçük farklılıklar, modellerin sınıflandırma yeteneklerinin göreceli olarak benzer olduğunu göstermektedir.



Şekil 4.7 Model_4 yöntemine ait ROC eğrisi



Şekil 4.8 Model_4 yöntemine ait PR eğrisi

4.5 CNN-BiLSTM-AM Yöntemi ile Elde Edilen Bulgular

CNN-BiLSTM-AM yöntemi ile gerçekleştirilen deneysel çalışmalara ilişkin bulgular bu bölümde sunulmuştur. Modelin eğitiminde kullanılan parametreler Tablo 4.8'de verilmiştir.

Tablo 4.8 CNN-BiLSTM-AM yönteminin parametreleri

Parametreler	**Değer**
Conv1D filtreleri (filters)	64
Conv1D çekirdek boyutu (kernel size)	1
Conv1D aktivasyon fonksiyonu	Relu
Conv1D padding	Same
Maxpooling havuz boyutu (pool_size)	2
BiLSTM birim sayısı (units)	64
BiLSTM aktivasyon fonksiyonu	Relu
Dikkat (Attention)	-
Dense_1 nöron sayısı	64
Dense_2 nöron sayısı	32
Dense_3 nöron sayısı	1

Tablo 4.8 (Devam) CNN-BiLSTM-AM yönteminin parametreleri

Eğitim Parametreleri	**Değer**
Optimizasyon (Optimizer) algoritması	RMSprop
Devir (Epoch) Sayısı	100
Batch boyutu (Batch_size)	64
Kayıp fonksiyonu (Loss function)	binary-crossentropy

Veri setinde toplam işlem sayısının %21'i (42.019) yasal, %2'si (4.545) yasadışı, geri kalan %77'si (157.205) ise bilinmeyen olarak etiketlenmiştir. Veri setindeki dengesizliği ele almak için Sentetik Azınlık Aşırı Örnekleme Tekniği (Synthetic Minority Oversampling Technique, SMOTE) uygulandı. SMOTE, azınlık sınıfına ait örnekleri çoğaltarak veri setini dengeler. SMOTE işlemi şu adımları içerir:

- Azınlık sınıfına ait bir örnek seçilir.
- Rastgele bir başka örnek seçilir ve bu iki örnek arasında bir vektör oluşturulur.
- Bu vektör kullanılarak yeni örnekler oluşturulur ve orijinal örneğe benzer sentetik örnekler üretilir.
- Yeni sentetik örnekler, azınlık sınıfının bir parçası olarak veri setine eklenir.

SMOTE yöntemini k-kat çapraz doğrulama sürecinde kullanıldı. Veri seti her katmanda dengelendi ve her bir katmanda ayrı ayrı k-kat çapraz doğrulama gerçekleştirildi. Bu sayede, modelin her katmanda daha dengeli bir veri setine dayanmasını sağladık. Ayrıca CNN-BİLSTM-AM yönteminin doğruluğunu test etmek için her bir model bağımsız olarak da test edilmiştir. Elde edilen sonuçlar Tablo 4.9'da verilmiştir.

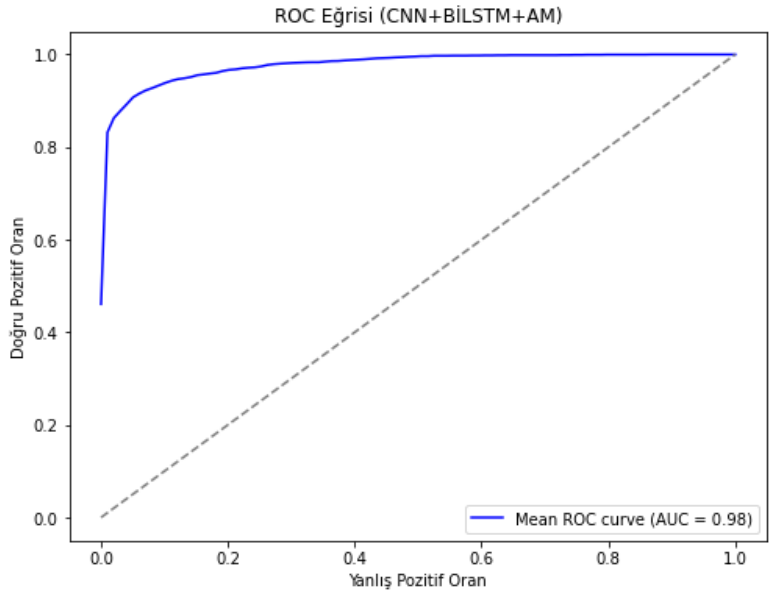
Tablo 4.9 CNN-BİLSTM-AM modelinin ve bağımsız yöntemlerin performans karşılaştırması

Yöntem	**Doğruluk (%)**	**Kesinlik (%)**	**Duyarlık (%)**	**F1-skor (%)**	**AUC**	**PR-AUC**
CNN	96,5	97,6	91,0	83,7	0,99	0.94
LSTM	91,8	65,0	84,7	68,0	0,97	0.93
BİLSTM	94,3	86,4	90,1	78,1	0,99	0.93

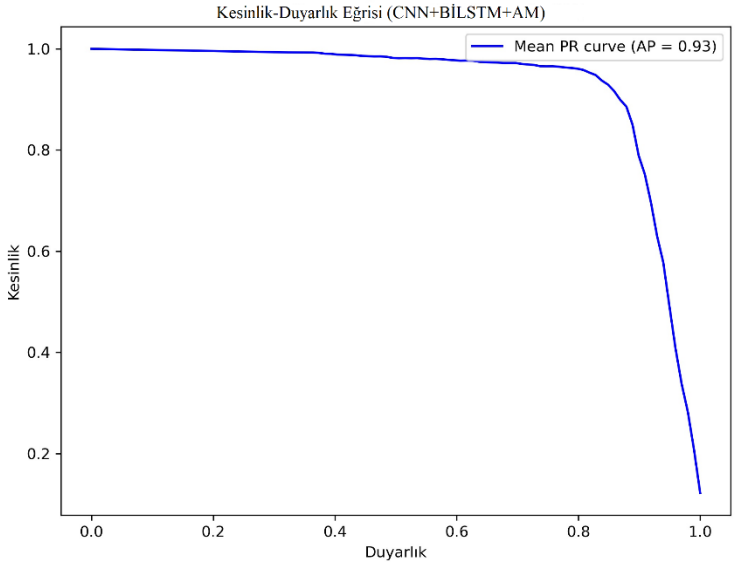
Tablo 4.9 (Devam) CNN-BİLSTM-AM modelinin ve bağımsız yöntemlerin performans karşılaştırması

CNN+LSTM	95,4	73,3	84,7	78,5	0,97	0.92
CNN+BİLSTM	96,4	98,7	64,4	77,9	0,99	0.95
CNN+AM	94,7	71,2	82,4	75,8	0,92	0.74
BİLSTM+AM	92,4	50,7	66,6	56,5	0,97	0.93
CNN-BiLSTM-AM	99,0	98,0	98,0	97,9	0,98	0.93

Tablo 4.9'da, CNN-BiLSTM-AM modelin diğer bağımsız DL yöntemleriyle karşılaştırmasını gösterilmiştir. CNN, verinin mekânsal özelliklerini öğrenmede etkiliyken, BiLSTM, zamansal bağımlılıkları anlamada güçlüdür. Bu iki modelin birlikte kullanılması, verinin hem yerel özelliklerini hem de zaman serisi ilişkilerini aynı anda öğrenebilen bir model ortaya çıkarmaktadır. AM, modelin yalnızca önemli özniteliklere odaklanmasını sağlayarak verinin karmaşıklığını yönetir ve sınıflandırma doğruluğunu artırır. Bu özellik, özellikle kripto para işlemlerinin dinamik doğasında önemli bir avantaj sağlar. Bu çalışmada önerilen model, CNN, BiLSTM ve hibrit yöntemlerin her birinin performansını artıran bir yapıya sahiptir. Şekil 4.9 ve Şekil 4.10'da CNN-BİLSTM-AM yöntemine ait ROC ve PR eğrileri gösterilmiştir. Ek A.5 ve Ek A.6'da diğer bağımsız modellere ait ROC ve PR eğrileri verilmiştir.



Şekil 4.9 CNN-BİLSTM-AM yöntemine ait ROC eğrisi



Şekil 4.10 CNN-BİLSTM-AM yöntemine ait PR eğrisi

Bu tez kapsamında önerilen üç farklı model (ML tabanlı, GCN tabanlı ve hibrit CNN-BiLSTM-AM modeli), Elliptic veri seti üzerinde test edilmiş ve elde edilen performans metrikleri literatürdeki çalışmalarla Tablo 4.10'da karşılaştırılmıştır.

Tablo 4.10 Elliptic veri setine uygulanan modellerin performans sonuçları

Referans	**Metot**	**Doğruluk (%)**	**Kesinlik (%)**	**Duyarlık (%)**	**F1-skor (%)**
Weber ve diğ. (2019)	LR	-	40,4	59,3	48,1
	RF		95,6	67,0	78,8
	MLP		69,4	61,7	65,3
	GCN		81,2	51,2	62,8
	Skip-GCN		81,2	62,3	70,5
	EvolveGCN		85,0	62,4	72,0
Alarab ve diğ. (2020)	GCN + MLP	97,4	89,9	67,8	77,3

Tablo 4.10 (Devam) Elliptic veri setine uygulanan modellerin performans sonuçları

Lorenz ve diğ. (2020)	XGBoost LR RF	-	-	-	76,0 45,0 83,0
Elbaghdadi ve diğ. (2021)	k-NN	90,0	78,0	56,0	-
Pocher ve diğ. (2023)	GCN (tx) GAT (tx)	-	90,6 89,7	79,0 60,5	84,4 72,3
Chen ve diğ. (2023)	GraphSAGE	91,3	61,8	80,2	66,1
Lo ve diğ. (2023)	Inspection-L	-	97,2	72,1	82,8
Yang ve diğ. (2023)	LSTM+GCN	-	85,4	71,2	77,6
Xiao ve diğ. (2023)	CTDM	-	85,1	87,4	83,5
Guo ve diğ. (2023)	LB-GLAT	97,76	93,17	84,94	88,87
Alarab ve diğ. (2021)	MC-dropout	96,6	-	-	72,9
Xia ve diğ. (2022)	MGC-LSTM	-	96,6	84,4	90,2
Alarab ve Prakoonwit (2023)	Temporal-GCN	97,7	92,7	71,3	80,6

Tablo 4.10 (Devam) Elliptic veri setine uygulanan modellerin performans sonuçları

Jatoth ve diğ. (2021)	SVM	97,0	92,0	79,0	84,0
	k-NN	96,0	82,0	79,0	80,0
	LR	93,0	71,0	79,0	75,0
	KararAğaçları	96,0	85,0	83,0	83,0
	MLP	96,0	83,0	61,0	65,0
	Ensemble (Boosting)	98,0	98,0	82,0	89,0
	Ensemble (Stacking)	97,0	95,0	81,0	86,0
	Ensemble (Stacking-CFS)	98,0	98,0	92,0	92,0
	Ensemble (Stacking-CFS)	98,0	98,0	93,0	95,0
Karim ve diğ. (2024)	Skip-GCN	-	-	-	91,6
	FastGCN				92,5
	EvolveGCN				93,4
Liu ve diğ. (2024)	Evolved graph attention network	-	43,2	95,3	59,5
Wang ve diğ. (2025)	RMGANets	97,9	96,8	91,3	93,8
Önerilen ML modeli	LightGBM (Lyap7)	90,2	86,6	90,2	86,2
Önerilen GCN modeli	GCN Model_4	86,2	92,5	92,1	92,3
Önerilen hibrit DL modeli	**CNN-BiLSTM-AM**	**97,9**	**99,0**	**98,0**	**98,0**

Weber ve diğ. (2019) çalışmasında RF modeli ile %95,6 kesinlik ve %78,8 F1-skor elde etmiş, bu sonuç ML yöntemlerinin güçlü bir temel sunduğunu göstermiştir. Ancak aynı çalışmada LR yalnızca %40,4 kesinlik sağlamış, bu da ML yöntemlerinin veri karmaşıklığına karşı değişken performans gösterebildiğini ortaya koymuştur. Benzer şekilde Jatoth ve diğ. (2021), topluluk öğrenmesi (ensemble learning) yöntemleriyle %98 doğruluk ve %95 F1-skoru gibi oldukça yüksek başarılar elde etmiş, ancak bu yöntemler genellikle yüksek hesaplama maliyeti ve düşük yorumlanabilirlik sorunları barındırmaktadır. LEs'lerine dayalı kaotik öznitelikler kullanılarak geliştirilen LightGBM modelinde %90,2 doğruluk, %86,6 kesinlik, %90,2 duyarlık ve %86,2 F1-skoru elde edilmiştir. Yöntemimiz, LightGBM'in optimize edilmiş yaprak tabanlı gradyan artırma algoritmasından yararlanarak yüksek doğruluk, hızlı eğitim süresi ve iyi genelleme performansı sağlamaktadır. Bu yapı, özellikle büyük veri kümeleri üzerinde çalışan veya gerçek zamanlı ortamlar için geliştirilen AML sistemlerinde önemli avantajlar sunmaktadır. Daha karmaşık ve kaynak yoğun topluluk teknikleriyle karşılaştırıldığında, LightGBM daha verimli hesaplama imkânı sağlamakta ve aşırı uyum riskini azaltmaktadır.

GCN ve türevlerine dayalı yöntemler (Skip-GCN, EvolveGCN, Temporal-GCN) birçok çalışmada test edilmiştir. Weber ve diğ. (2019)'un orijinal GCN modeli %81 kesinlik ve %62,8 F1-skor ile sınırlı bir performans sergilemiştir. Ancak Alarab ve Prakoonwit (2023) tarafından önerilen Temporal-GCN modeli %97,7 doğruluk ve %80,6 F1-skor ile belirgin bir gelişim göstermiştir. Guo ve diğ. (2023) ise LB-GLAT modeliyle %97,7 doğruluk ve %88,8 F1-skor elde ederek graf temsillerinin AML tespitinde güçlü bir potansiyel taşıdığını kanıtlamıştır. Bu çalışmada önerilen GCN tabanlı Model_4 ise %92,3 F1-skoru ile literatürdeki pek çok GCN modelini geride bırakmıştır. Özellikle GraphSAGE (%66,1 F1-skor) ve Temporal-GCN (%80,6 F1-skor) gibi yöntemlere kıyasla belirgin bir iyileşme sağlanmıştır. Bu durum, önerilen modelin doğru bir komşuluk örnekleme ve öznitelik aktarımı ile graf tabanlı yaklaşımlarda performans dengesini geliştirebildiğini göstermektedir.

Zamansal bağımlılıkları yakalamada başarılı olan LSTM tabanlı modeller, özellikle Xia ve diğ. (2022)'nin MGC-LSTM çalışmasında %96,9 kesinlik ve %90,2 F1-skor ile öne çıkmıştır. Hibrit yöntemlerde ise Wang ve diğ. (2025) tarafından geliştirilen RMGANets modeli %97,9 doğruluk ve %93,8 F1-skor elde etmiştir. Bu sonuç, derin öğrenme tabanlı hibrit mimarilerin kara para aklama tespitinde güçlü bir alternatif sunduğunu ortaya koymaktadır. Bu çalışmada önerilen CNN-BiLSTM-AM hibrit mimarisi, %97,9 doğruluk ve %98,0 F1-skoru ile tabloda yer alan tüm yöntemlerden daha iyi sonuçlar vermiştir. Bu sonuç, konvolüsyonel katmanların uzamsal örüntüleri yakalama gücünü, BiLSTM'in zamansal bağımlılıkları modelleme

kapasitesini ve AM'nin öne çıkan özniteliklere odaklanma yeteneğini bir araya getirmenin etkinliğini ortaya koymaktadır.

Genel olarak, bu üç model arasında CNN-BiLSTM-AM mimarisi en yüksek başarıyı sağlamış, GCN Model_4 graf tabanlı yöntemler içinde güçlü bir alternatif olarak öne çıkmış, LightGBM ise geleneksel yöntemlerin doğru öznitelik mühendisliği ile hâlâ güçlü bir seçenek olabileceğini göstermiştir. Bu bulgular, literatürdeki sonuçlarla birlikte değerlendirildiğinde, hibrit yaklaşımların özellikle karmaşık ve heterojen veri yapılarında üstünlük sağladığını, ancak graf tabanlı ve geleneksel yöntemlerin de farklı senaryolarda avantajlar sunabileceğini ortaya koymuştur.

5. TARTIŞMA

Bu çalışmada önerilen yaklaşım; blok zinciri işlem graf yapısından artırılmış öznitelik çıkarımı, bu özniteliklerin kaotik zaman serilerine dönüştürülmesi ve LEs ile dinamiklerinin incelenmesi ve elde edilen kaotik tabanlı özniteliklerin hem geleneksel ML hem de DL modelleriyle işlenmesi bileşenlerini bir araya getirir. Hipotezlerimiz doğrultusunda gerçekleştirilen deneyler, LEs temelli özniteliklerin ağdaki gizli dinamikleri ortaya çıkarmada katkı sağladığını ve bu özniteliklerin özellikle GCN ve hibrit DL mimarilerinin performansını artırdığını göstermektedir. Özet olarak, GCN tabanlı Model_4 için elde edilen %92,5 kesinlik, %92,1 duyarlık, %92,3 F1-skor ve %86,2 doğruluk ile CNN-BiLSTM-AM hibrit modelinin yaklaşık %98 civarı doğruluk/kesinlik/duyarlık/F1-skor değerleri, kaotik özniteliklerin ve AM'nin sınıflandırma başarısına kayda değer katkı verdiğini ortaya koymaktadır.

LEs hesaplarının analizi, işlem serilerinin doğrusal olmayan, hassas başlangıç koşullarına duyarlı davranışlar içerdiğine işaret etmiştir; özellikle pozitif LEs değerleri sistemde kaotik davranış varlığını desteklemektedir. Bu sonuç, işlem ağındaki küçük değişikliklerin zaman içinde farklı izler oluşturabileceği ve dolayısıyla klasik kural yaklaşımlarının bu dinamikleri yakalayamayacağı hipotezini destekler. Kaotik özniteliklerin ML ve DL sınıflandırıcılara dâhil edilmesi, modellerin veri içindeki zamanla değişen ve doğrusal olmayan ilişkileri öğrenmesini kolaylaştırmış, özellikle GCN mimarilerinde topolojik bilgi ile dinamik özelliklerin birleşimi daha iyi ayrıştırma performansı sağlamıştır.

Öznitelik mühendisliği kapsamında uygulanan polinom ve etkileşim özelikleri, veri içindeki yüksek dereceli ilişkilere modelin erişimini artırmış; bu da özellikle ML tabanlı sınıflandırıcılarda doğruluk artışı ve yanlış pozitiflerde azalma ile sonuçlanmıştır. AM entegrasyonu ise CNN-BiLSTM mimarisinde sıralı veri içerisindeki anlamlı zaman aralıklarına odaklanmayı sağlayarak hem hassasiyeti hem de genel F1-skorunu iyileştirmiştir. SMOTE ile sınıf dengesinin iyileştirilmesi, eğitim sırasında dengesizlik kaynaklı sapmaları azaltmış ve modellerin yasadışı sınıfı öğrenmesini kolaylaştırmıştır.

Sonuç olarak, çalışmanın bulguları Bölüm 1.3 de verilen Hipotezler ile uyumludur: Kaotik tabanlı özniteliklerin çıkarımı, blok zinciri işlem ağlarının dinamik örüntülerini daha belirgin hale getirmiş ve bu öznitelikler GCN ile hibrit DL modellerinin öğrenme kapasitesini artırarak AML tespit performansını iyileştirmiştir.

6. SONUÇLAR VE ÖNERİLER

Kripto paralar, finans dünyasında devrim yaratan ve küresel ölçekte kabul gören teknolojik yenilikler olarak öne çıkmaktadır. Ancak bu yenilikler, yasa dışı faaliyetlerin artmasına da zemin hazırlamıştır. Kara para aklama, terörizmin finansmanı ve diğer suç faaliyetleri, kripto paraların anonim ve sınır ötesi doğası nedeniyle daha karmaşık ve yaygın hale gelmiştir. Bu durum, kripto para ekosisteminin şeffaflığını ve güvenilirliğini tehdit ederken, finansal suçlarla mücadele için yeni ve etkili çözümlerin geliştirilmesini zorunlu kılmaktadır. Alınabilecek önlemler arasında şeffaflığı ve kimlik tespitini artırmak amacıyla düzenlemelerin ve uyum önlemlerinin güçlendirilmesi yer almaktadır. Blok zincir analitiği ve yapay zeka tabanlı algoritmalar, kara para aklama faaliyetlerini tespit etmek için kullanılabilecek güçlü araçlardır.

Bu tez çalışması, blok zinciri tabanlı finansal sistemlerde kara para aklama tespitine yönelik hibrit ve yenilikçi bir yaklaşım önermektedir. Çalışmanın temel katkısı, blok zinciri işlem graflarından elde edilen yapısal özniteliklerin kaotik zaman serilerine dönüştürülmesi ve LEs ile tanımlanmasıdır. Bu yöntem, geleneksel öznitelik çıkarım süreçlerinin ötesine geçerek, işlem ağlarının dinamik ve karmaşık yapısını daha ayrıntılı bir biçimde modellemeyi mümkün kılmıştır.

Elde edilen kaotik tabanlı öznitelikler, hem ML algoritmaları hem de DL tabanlı yöntemlerle değerlendirilmiştir. Bulgular, bu özniteliklerin sınıflandırma performansını anlamlı ölçüde artırdığını ve özellikle yanlış pozitif oranlarını azalttığını göstermiştir. GCN ve diğer DL modelleri ile yapılan karşılaştırmalar, önerilen hibrit yaklaşımın işlem verilerindeki karmaşık ilişkileri daha etkili biçimde yakaladığını ortaya koymuştur. Ek olarak, CNN, BiLSTM ve AM birleştiren hibrit modelin, Elliptic veri seti üzerinde gerçekleştirilen deneylerde literatürdeki mevcut yöntemlere kıyasla daha yüksek doğruluk sağladığı görülmüştür.

Bu çalışma, kripto para ekosisteminde kara para aklamayı tespit etmek için pratik ve ölçeklenebilir bir yöntem sunmaktadır. Önerilen model, yalnızca akademik bir katkı değil, aynı zamanda finansal kurumlar ve düzenleyici otoriteler için uygulanabilir bir çözüm niteliği taşımaktadır. Ayrıca, blok zinciri tabanlı işlem verilerine uyarlanabilir yapısı sayesinde, finansal sistemlerde şeffaflığın artmasına ve güvenliğin güçlenmesine katkı sağlamaktadır.

Gelecek çalışmalar açısından, önerilen yöntemin farklı blok zinciri platformlarına uygulanması, gerçek zamanlı tespit senaryolarına uyarlanması ve daha geniş ölçekli veri kümeleri üzerinde uygulanması önerilmektedir.

Sonuç olarak, bu tez, blok zinciri tabanlı finansal sistemlerde yasa dışı faaliyetlerin tespitinde kaos teorisi ile yapay zeka yöntemlerini bir araya getiren özgün bir yaklaşım önermekte ve literatüre önemli bir katkı sağlamaktadır. Elde edilen bulgular, yalnızca akademik açıdan değil, aynı zamanda finansal suçlarla mücadelede kullanılabilecek güçlü ve yenilikçi araçların geliştirilmesi bakımından da değer taşımaktadır.

KAYNAKLAR

Abarbanel, H. D. I., Brown, R., Sidorowich, J. J., Tsimring, L. Sh. (1993). The analysis of observed chaotic data in physical systems. *Reviews of Modern Physics*, 65(4), 1331-1392.

Aggarwal, K., Mijwil, M. M., Al-Mistarehi, A. H., Alomari, S., Gök, M., Alaabdin, A. M. Z., & Abdulrhman, S. H., (2022). Has the future started? The current growth of artificial intelligence, machine learning, and deep learning. *Iraqi Journal for Computer Science and Mathematics*, 3(1), 115-123.

Alarab, I., Prakoonwit, S., Nacer, M. I. (2020). *Competence of graph convolutional networks for anti-money laundering in bitcoin blockchain*. In: Proceedings of the 2020 5th international conference on machine learning technologies, 23–27.

Alarab, I., Prakoonwit, S., Nacer, M. I. (2021). Illustrative discussion of mc-dropout in general dataset: Uncertainty estimation in bitcoin. *Neural Processing Letters*, 53(2), 1001-1011.

Alarab, I., Prakoonwit, S. (2023). Graph-Based LSTM for Anti-money Laundering: Experimenting Temporal Graph Convolutional Network with Bitcoin Data. *Neural Processing Letters*, 55(1), 689-707.

Albon, C. (2018). *Machine learning with python cookbook: Practical solutions from preprocessing to deep learning*. O'Reilly Media, Inc.

Alkhalili, M., Qutqut, M. H., Almasalha, F. (2021). Investigation of applying machine learning for watch-list filtering in anti-money laundering. *IEEE Access*, 9, 18481-18496

Altuntas, V., Gok, M., Kocal, O. H. (2020). Response of Lyapunov exponents to diffusion state of biological networks. *International Journal of Applied Mathematics and Computer Science*, 30(4), 689-702.

Bacon, L., Tarr, J. A. (2024). Distributed ledger technology and blockchain: Insurance. *The Global Insurance Market and Change*, 95-126.

Bakry, A. N., Alsharkawy, A. S., Farag, M. S., & Raslan, K. R. (2024). Automatic suppression of false positive alerts in anti-money laundering systems using machine learning. *The Journal of Supercomputing*, 80(5), 6264-6284.

Baysal, K., Taşkin, D. (2024). Blocking harmful images with a deep learning based next generation firewall. *Sigma Journal of Engineering and Natural Sciences*, 42(4), 1133-1147.

Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24.

Bergstra, J., Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

Chai, Z., You, S., Yang, Y., Pu, S., Xu, J., Cai, H., & Jiang, W. (2022, July). *Can abnormality be detected by graph neural networks?*. Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22), 1945-1951.

Chen, C., Li, Q., Chen, L., Liang, Y., Huang, H. (2023). An improved GraphSAGE to detect power system anomaly based on time-neighbor feature. *Energy Reports*, 9, 930-937.

Chen, T., Guestrin, C. (2016). *Xgboost: A scalable tree boosting system*. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 785-794.

Cihan, M., Uzbaş, B., Ceylan, M. (2023). Fusion and CNN based classification of liver focal lesions using magnetic resonance imaging phases. *Sigma Journal of Engineering and Natural Sciences*, 41(1), 119-129.

Cortes, C., Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20, 273-297.

Couronné, R., Probst, P., Boulesteix, A. L. (2018). Random forest versus logistic regression: a large-scale benchmark experiment, *BMC bioinformatics*, 19, 1-14.

Cui, Z., Henrickson, K., Ke, R., Wang, Y. (2020). Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 21(11), 4883-4894.

Elbaghdadi, A., Mezroui, S., El Oualkadi, A. (2021). K-nearest neighbors algorithm (knn): an approach to detect illicit transaction in the bitcoin network. *In Integration Challenges for Analytics, Business Intelligence, and Data Mining* (161-178). IGI Global Scientific Publishing.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., Yin, D. (2019). *Graph Neural Networks for Social Recommendation*. The World Wide Web Conference, 417-426.

Friedman, J. H. (2001). *Greedy function approximation: a gradient boosting machine*. Annals of statistics, 1189-1232.

Gök, M. (2024). *Makine Öğrenmesi Algoritmaları*. Nobel Yayınevi.

Guo, C., Zhang, S., Zhang, P., Alkubati, M., Song, J. (2023). LB-GLAT: Long-Term Bi- Graph Layer Attention Convolutional Network for Anti-Money Laundering in Transactional Blockchain. *Mathematics*, 11(18):3927.

Healy, L. M. (2006). Logistic regression: An overview, *Eastern Michighan College of Technology*.

Hegger, R., Kantz, H., Schreiber, T. (1999). Practical implementation of nonlinear time series methods: The TISEAN package. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 9(2), 413-435.

Jeon, Y., Ra, I., Park, Y., Lee, S. (2018). Machine Learning Optimization of Parameters for Noise Estimation. *Journal of Universal Computer Science*, 24(9), 1271-1281.

Jatoth, C., Jain, R., Fiore, U., Chatharasupalli, S. (2021). Improved classification of blockchain transactions using feature engineering and ensemble learning. *Future Internet*, 14(1), 16.

Jullum, M., Løland, A., Huseby, R. B., Ånonsen, G., Lorentzen, J. (2020). Detecting money laundering transactions with machine learning. *Journal of Money Laundering Control*, 23(1), 173-186.

Karim, Md. R., Hermsen, F., Chala, S. A., De Perthuis, P., Mandal, A. (2024). Scalable Semi-Supervised Graph Learning Techniques for Anti Money Laundering. *IEEE Access*, 12, 50012-50029

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Khemani, B., Patil, S., Kotecha, K., Tanwar, S. (2024). A review of graph neural networks: Concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1), 18.

Kingma, D. P. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kipf, T. N., Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks, arXiv:1609.02907, DOI: 10.48550/arXiv.1609.02907

Korejo, M. S., Rajamanickam, R., Md., S. M. H. (2021). The concept of money laundering: A quest for legal definition. *Journal of Money Laundering Control*, 24(4), 725-736.

Kotagiri, A. (2024). AML Detection and Reporting with Intelligent Automation and Machine learning. *International Machine learning journal and Computer Engineering*, 7(7), 1-17.

Kruse, R., Mostaghim, S., Borgelt, C., Braune, C., Steinbrecher, M. (2022). Multi-layer perceptrons. *In Computational intelligence: a methodological introduction*, 53-124.

Kute, D. V., Pradhan, B., Shukla, N., & Alamri, A., (2024). Explainable deep learning model for predicting money laundering transactions. *International Journal on Smart Sensing and Intelligent Systems*, 17(1).

Li, J., Zhang, C., Zhang, J., & Shao, Y. (2024). Research on Blockchain Transaction Privacy Protection Methods Based on Deep Learning. *Future Internet*, 16(4), 113.

Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. (2017, October*). Focal loss for dense object detection*. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy.

Liu, C., Xu, Y., & Sun, Z. (2024). Directed dynamic attribute graph anomaly detection based on evolved graph attention for blockchain. *Knowledge and Information Systems*, 66(2), 989-1010.

Lo, W. W., Kulatilleke, G. K., Sarhan, M., Layeghy, S., & Portmann, M. (2023). Inspection-L: self-supervised GNN node embeddings for money laundering detection in bitcoin. *Applied Intelligence*, 53(16), 19406-19417.

Lorenz, J., Silva, M. I., Aparício, D., Ascensão, J. T., Bizarro, P. (2020, October). *Machine learning methods to detect money laundering in the bitcoin blockchain in the presence of*

label scarcity, Proceedings of the First ACM International Conference on AI in Finance, 1-8.

Lokanan, M. E. (2024). Predicting money laundering using machine learning and artificial neural networks algorithms in banks. *Journal of Applied Security Research*, 19(1), 20-44.

Marasi, S., Ferretti, S. (2024). *Anti-Money Laundering in Cryptocurrencies Through Graph Neural Networks: A Comparative Study*, 2024 IEEE 21st Consumer Communications & Networking Conference (CCNC), 272-277.

Mohri, M., Rostamizadeh, A., Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M. M. (2017). *Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs*. In Proceedings of the IEEE conference on computer vision and pattern recognition, 5115-5124.

Nguyen, H., Vu, T., Vo, T. P., Thai, H. T. (2021). *Efficient machine learning models for prediction of concrete strengths*. Construction and Building Materials, 266, 120950.

Niu, Z., Zhong, G., Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, 452, 48–62.

Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y. (2013). How to construct deep recurrent neural networks. arXiv preprint arXiv:13126026.

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

Pocher, N., Zichichi, M., Merizzi, F., Shafiq, MZ., Ferretti, S., (2023) Detecting anomalous cryptocurrency transactions: An AML/CFT application of machine learning-based forensics, *Electronic Markets*, 33(1), 37.

Raiter, O. (2021). Applying supervised machine learning algorithms for fraud detection in anti-money laundering. *Journal of Modern Issues in Business Research*, 1(1), 14-26.

Rani, K., Deepak, B., Hemanthh, P., Mohanasudhan, B., & Sathishkumar, G., (2024, March). *Spatio-Temporal Network Based Bank Transactional Behaviour Analysis to Detect Suspicious Activities*, In 2024 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (AIMLA) (pp. 1-6). IEEE.

Schonlau, M., Zou, R. Y. (2020). The random forest algorithm for statistical learning. *The Stata Journal*, 20(1), 3-29.

Schuster, M., Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11), 2673-2681.

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Strogatz, S.H. (1994). *Nonlinear Dynamics and Chaos With Applications ιo Physics, Biology, Chemistry and Engineering*. Perseus Books Publishing, Massachusetts.

Tang, M., Ye, M., Chen, W., & Zhou, D., (2024). BiLSTM4DPS: An attention-based BiLSTM approach for detecting phishing scams in ethereum, *Expert Systems with Applications*, 256, 124941.

Teney, D., Liu, L., van Den Hengel, A. (2017). *Graph-structured representations for visual question answering*. In Proceedings of the IEEE conference on computer vision and pattern recognition, 1-9.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903. DOI: 10.48550/arXiv.1710.10903

Wan, F., Li, P. (2024). A Novel Money Laundering Prediction Model Based on a Dynamic Graph Convolutional Neural Network and Long Short-Term. *Symmetry*, 16(3), 378.

Wang, Q., Tsai, W. T., Du, B. (2025). RMGANets: reinforcement learning-enhanced multi-relational attention graph-aware network for anti-money laundering detection. *Complex & Intelligent Systems*, 11(1), 5.

Wang, S., Tang, H., & Chai, L. (2021, October). *Class Imbalance in Facial Expression Recognition by GCN with Focal Loss*. In 2021 China Automation Congress (CAC). 3270-3275, IEEE.

Wang, Y., Huang, M., Zhu, X., Zhao, L., (2016). *Attention-based LSTM for aspect-level sentiment classification*. In: Proceedings of the 2016 conference on empirical methods in natural language processing, p. 606–615.

Wang, Z., Ni, A., Tian, Z., Wang, Z., & Gong, Y., (2024) Research on blockchain abnormal transaction detection technology combining CNN and transformer structure, *Computers and Electrical Engineering*, 116, 109194.

Watanobe, Y., Rahman, MM., Amin, MFI., Kabir, R. (2023). Identifying algorithm in program code based on structural features using CNN classification model. *Applied Intelligence*, 53(10), 12210–12236.

Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., Leiserson, C. E. (2019). Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics. arXiv preprint arXiv:1908.02591. DOI: 10.48550/arXiv.1908.02591.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K. (2019). *Simplifying Graph Convolutional Networks*. Proceedings of the 36th International Conference on Machine Learning, 6861-6871.

Wolf, A., Swift, J. B., Swinney, H. L., Vastano, J. A. (1985). Determining Lyapunov exponents from a time series. *Physica D: Nonlinear Phenomena*, 16(3), 285-317.

Xia, P., Ni, Z., Xiao, H., Zhu, X., Peng, P. (2022). A Novel Spatiotemporal Prediction Approach Based on Graph Convolution Neural Networks and Long Short-Term Memory for Money Laundering Fraud. *Arabian Journal for Science and Engineering*, 47(2), 1921-1937.

Xiao, L., Han, D., Li, D., Liang, W., Yang, C., Li, K.-C., Castiglione, A. (2023). CTDM: Cryptocurrency abnormal transaction detection method with spatio-temporal and global representation. *Soft Computing*, 27(16), 11647-11660.

Yang, G., Liu, X., Li, B. (2023). Anti-money laundering supervision by intelligent algorithm. *Computers & Security*, 132, 103344.

Yang, L., Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415, 295-316.

Zhang, J., Lam, K. C., Yan, W. J., Gao, H., Li, Y. (2004). Time series prediction using Lyapunov exponents in embedding phase space. *Computers & Electrical Engineering*, 30(1), 1-15.

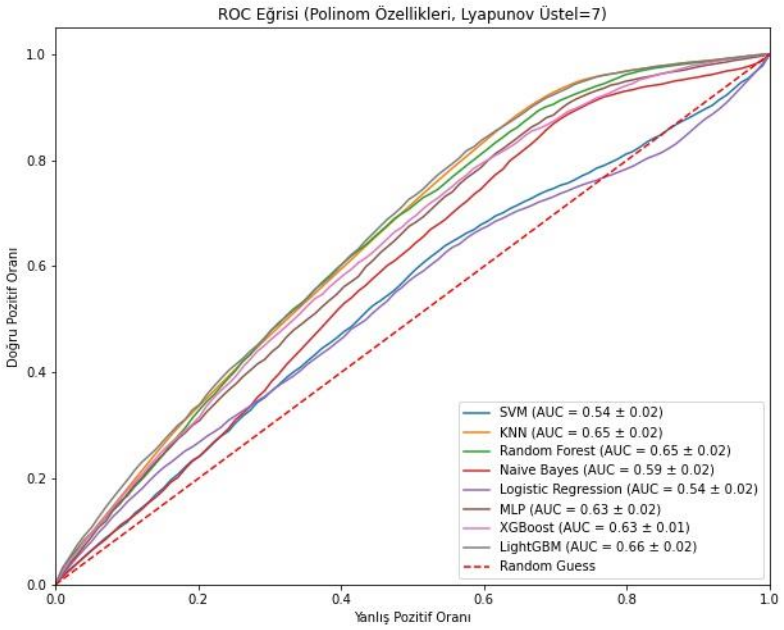
Zhang, J., Zhong, S., Wang, T., Chao, H. C., Wang, J. (2020). Blockchain-based systems and applications: a survey. *Journal of Internet Technology*, 21(1), 1-14.

Zhong, F., Liu, Y., Liu, L., Zhang, G., & Duan, S. (2022). DEDGCN: Dual Evolving Dynamic Graph Convolutional Network. *Security and Communication Networks*, 2022(1), 6945397.

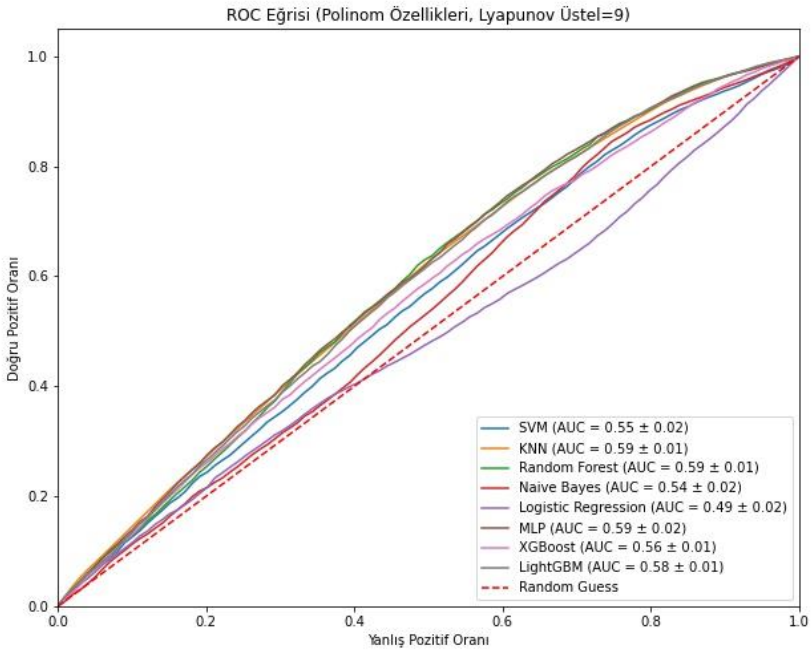
Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI open*, 1, 57-81.

EKLER

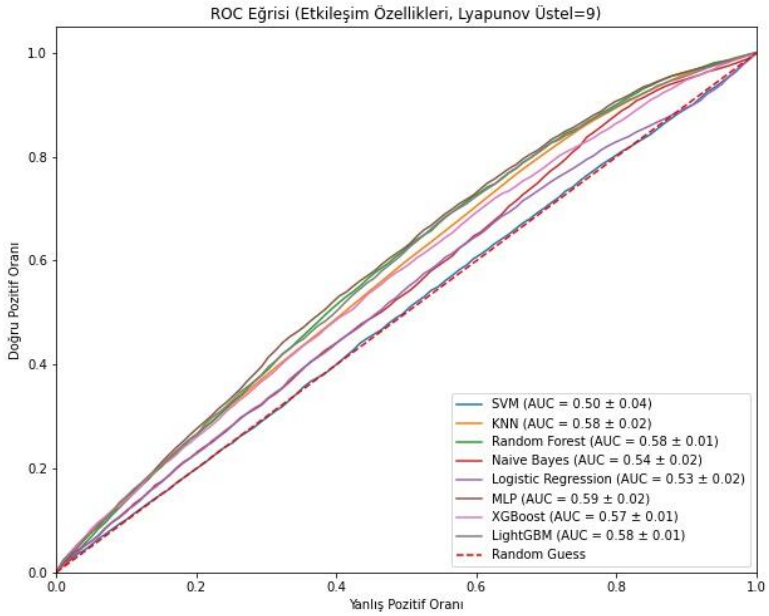
EK A.1


ROC Eğrisi (Polinom Özellikleri, Lyapunov Üstel=7)
Doğru Pozitif Oranı
Yanlış Pozitif Oranı
SVM (AUC = 0.54 ± 0.02)
KNN (AUC = 0.65 ± 0.02)
Random Forest (AUC = 0.65 ± 0.02)
Naive Bayes (AUC = 0.59 ± 0.02)
Logistic Regression (AUC = 0.54 ± 0.02)
MLP (AUC = 0.63 ± 0.02)
XGBoost (AUC = 0.63 ± 0.01)
LightGBM (AUC = 0.66 ± 0.02)
Random Guess

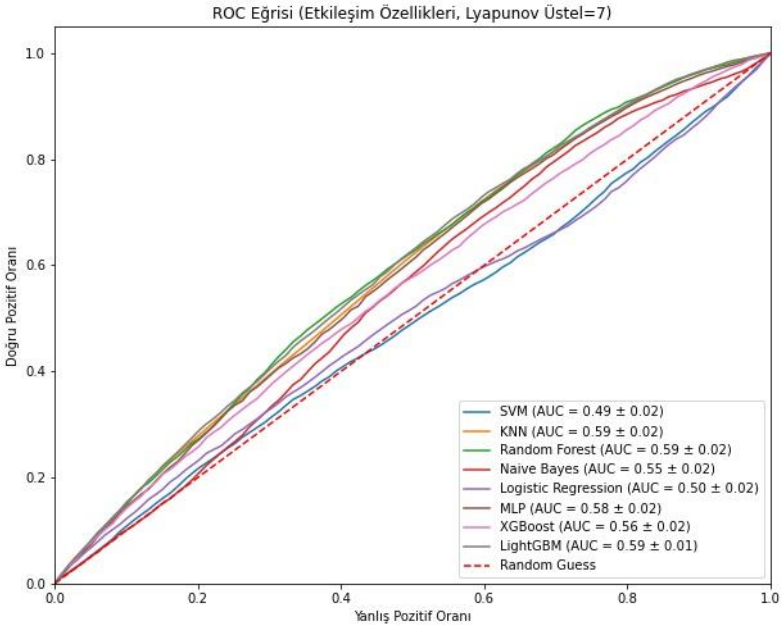
(a)


ROC Eğrisi (Polinom Özellikleri, Lyapunov Üstel=9)
Doğru Pozitif Oranı
Yanlış Pozitif Oranı
SVM (AUC = 0.55 ± 0.02)
KNN (AUC = 0.59 ± 0.01)
Random Forest (AUC = 0.59 ± 0.01)
Naive Bayes (AUC = 0.54 ± 0.02)
Logistic Regression (AUC = 0.49 ± 0.02)
MLP (AUC = 0.59 ± 0.02)
XGBoost (AUC = 0.56 ± 0.01)
LightGBM (AUC = 0.58 ± 0.01)
Random Guess

(b)



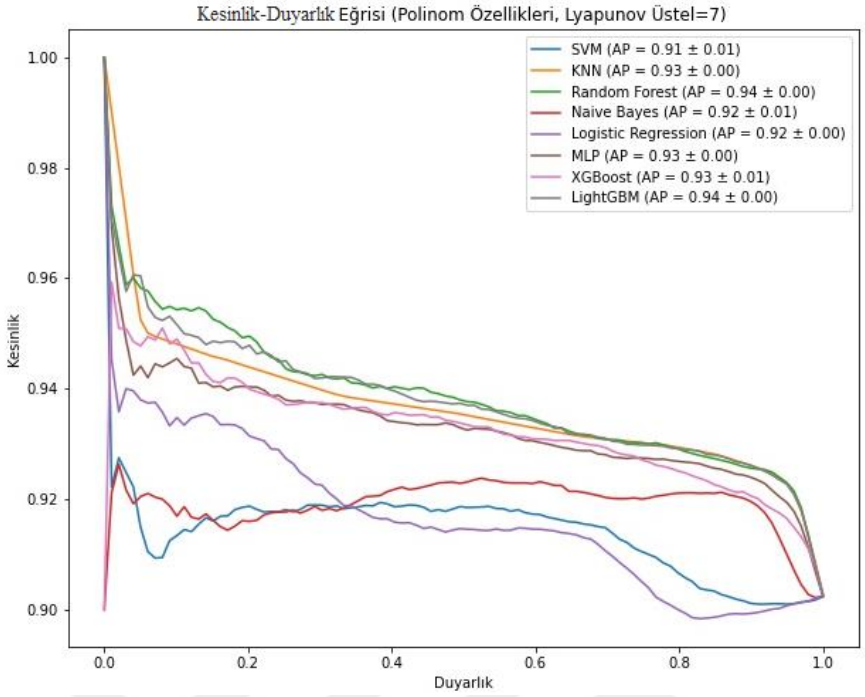
(c)



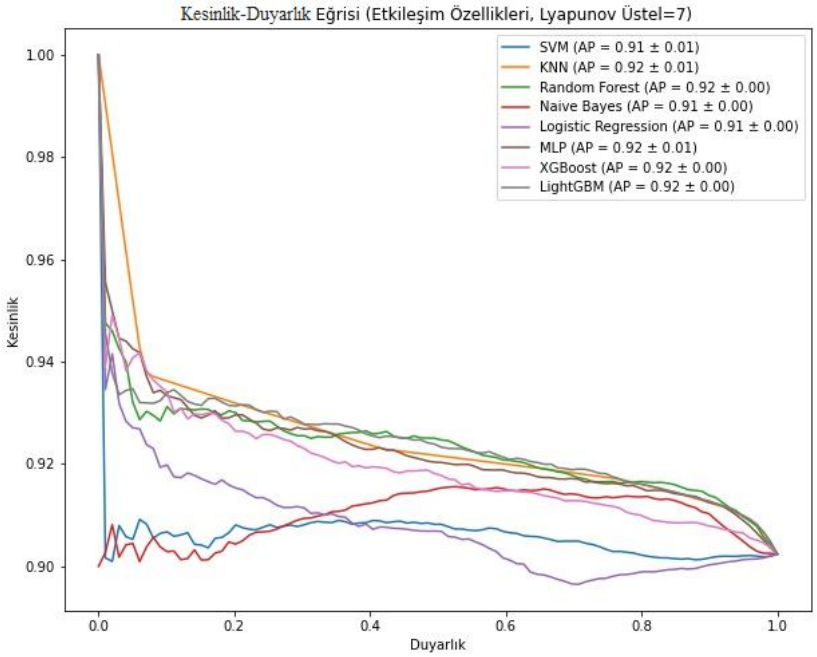
(d)

Şekil A.1 Makine öğrenmesi modellerine ait ROC eğrileri

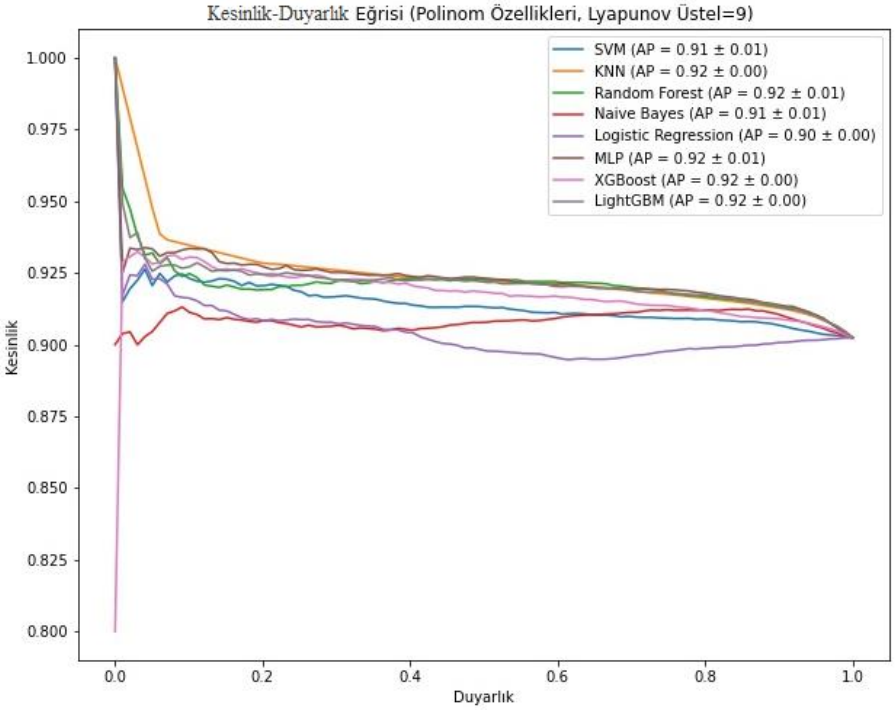
EK A.2


Kesinlik-Duyarlık Eğrisi (Polinom Özellikleri, Lyapunov Üstel=7)
SVM (AP = 0.91 ± 0.01)
KNN (AP = 0.93 ± 0.00)
Random Forest (AP = 0.94 ± 0.00)
Naive Bayes (AP = 0.92 ± 0.01)
Logistic Regression (AP = 0.92 ± 0.00)
MLP (AP = 0.93 ± 0.00)
XGBoost (AP = 0.93 ± 0.01)
LightGBM (AP = 0.94 ± 0.00)
Kesinlik
Duyarlık
1.00
0.98
0.96
0.94
0.92
0.90
0.0
0.2
0.4
0.6
0.8
1.0

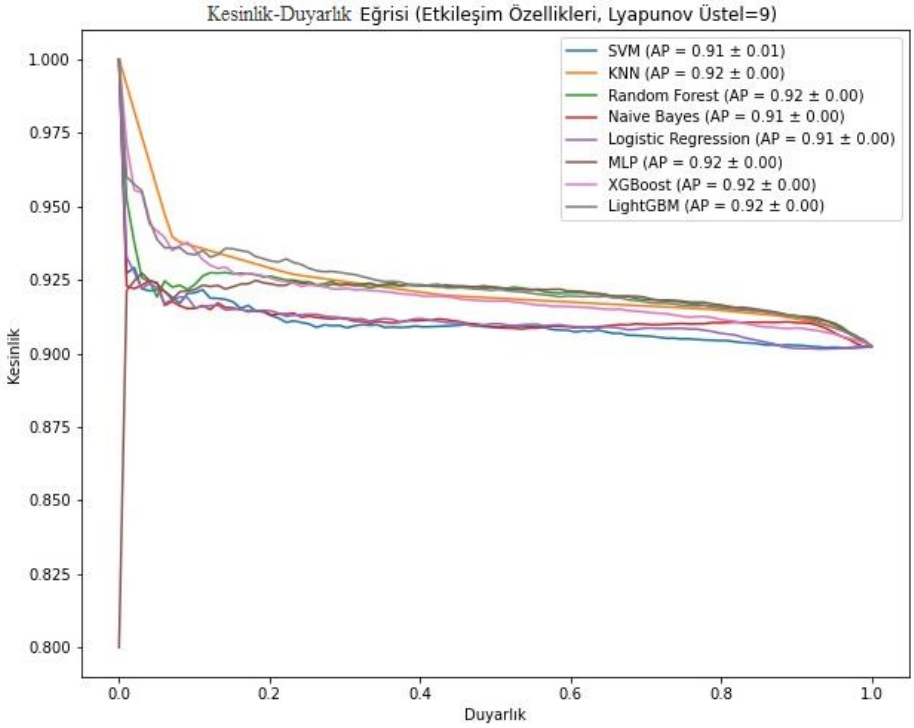
(a)


Kesinlik-Duyarlık Eğrisi (Etkileşim Özellikleri, Lyapunov Üstel=7)
SVM (AP = 0.91 ± 0.01)
KNN (AP = 0.92 ± 0.01)
Random Forest (AP = 0.92 ± 0.00)
Naive Bayes (AP = 0.91 ± 0.00)
Logistic Regression (AP = 0.91 ± 0.00)
MLP (AP = 0.92 ± 0.01)
XGBoost (AP = 0.92 ± 0.00)
LightGBM (AP = 0.92 ± 0.00)
Kesinlik
Duyarlık
1.00
0.98
0.96
0.94
0.92
0.90
0.0
0.2
0.4
0.6
0.8
1.0

(b)



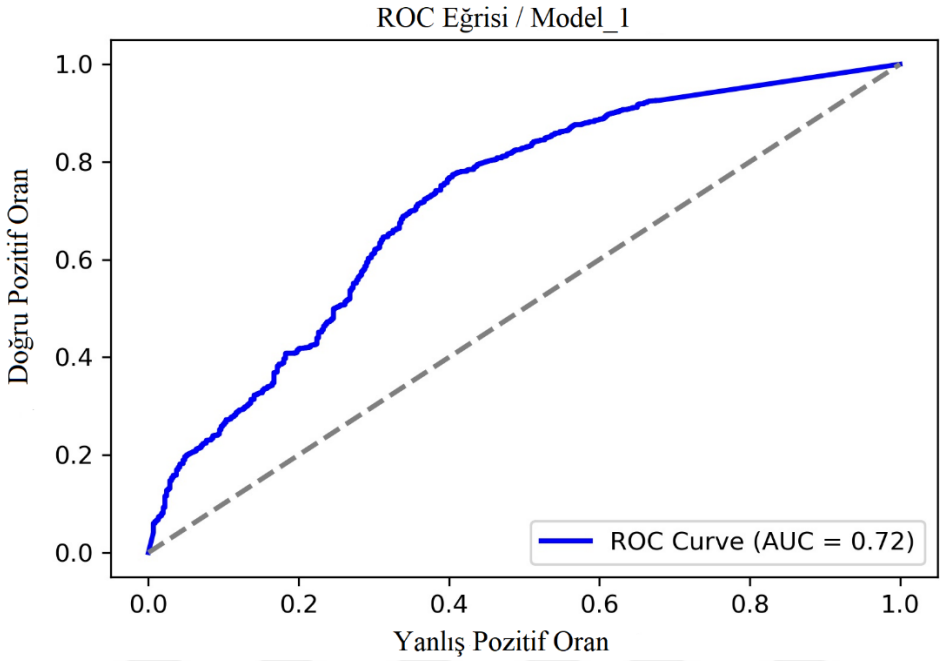
(c)



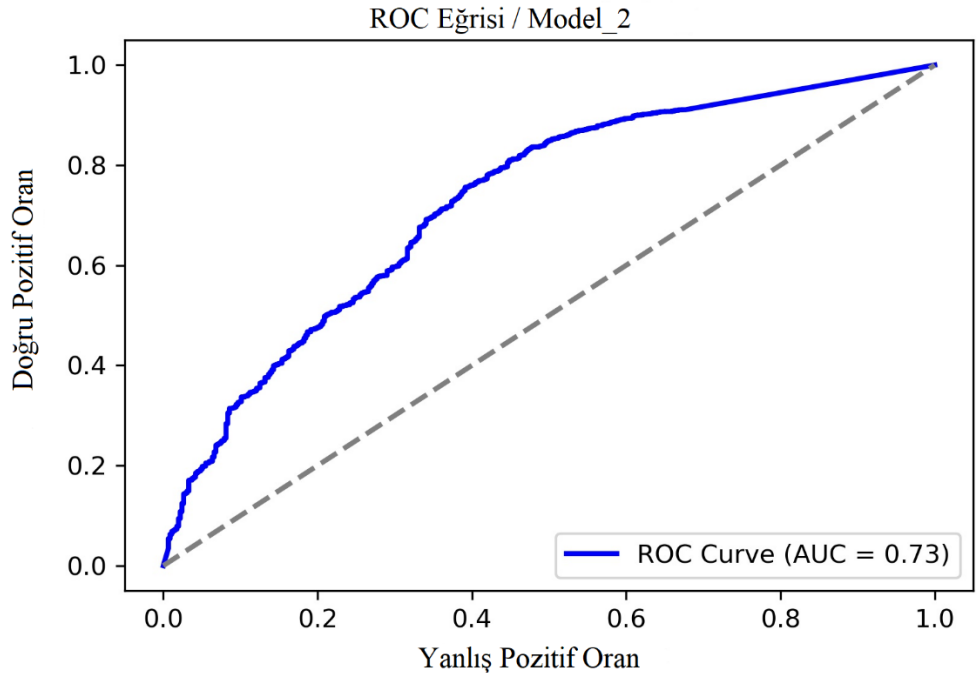
(d)

Şekil A.2 Makine öğrenmesi modellerine ait PR eğrileri

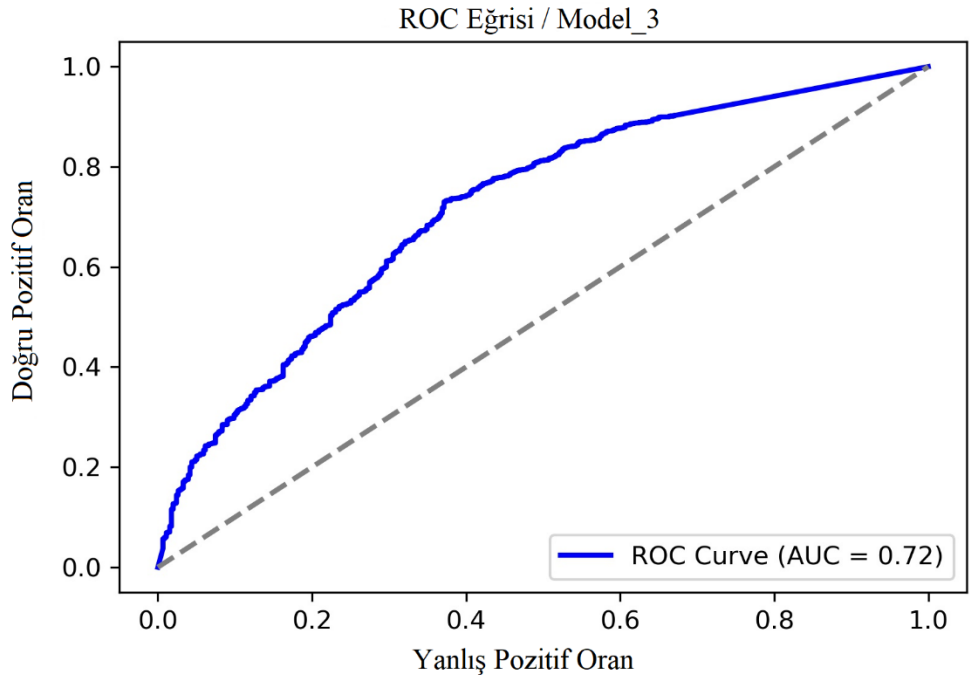
EK A.3


ROC Eğrisi / Model_1
Doğru Pozitif Oran
Yanlış Pozitif Oran
ROC Curve (AUC = 0.72)
0.0
0.2
0.4
0.6
0.8
1.0

(a)


ROC Eğrisi / Model_2
Doğru Pozitif Oran
Yanlış Pozitif Oran
ROC Curve (AUC = 0.73)
0.0
0.2
0.4
0.6
0.8
1.0

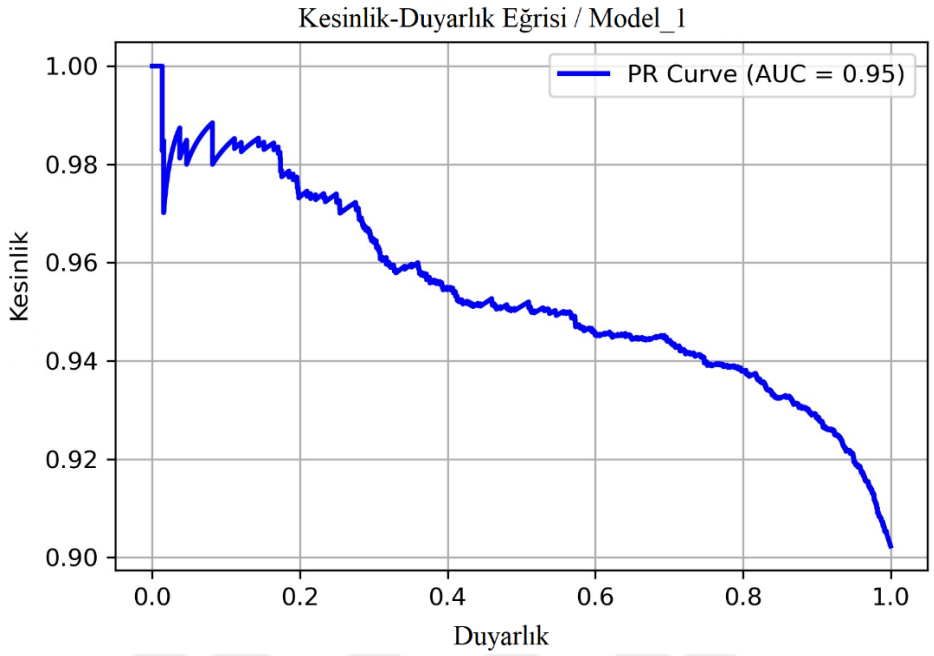
(b)



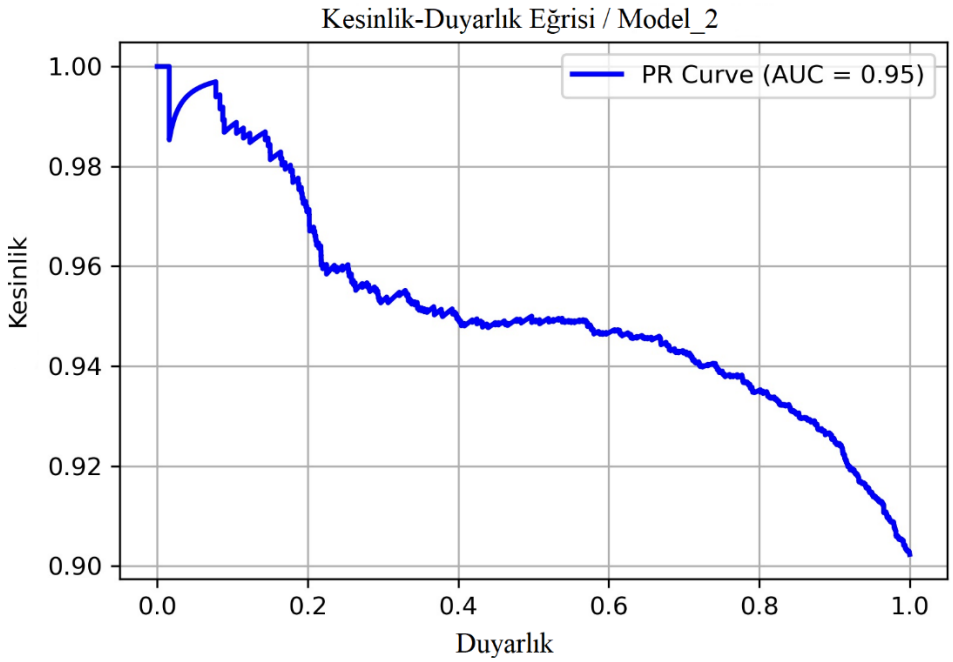
(c)

Şekil A.3 Graf evrişimli ağ modellerine ait ROC eğrileri

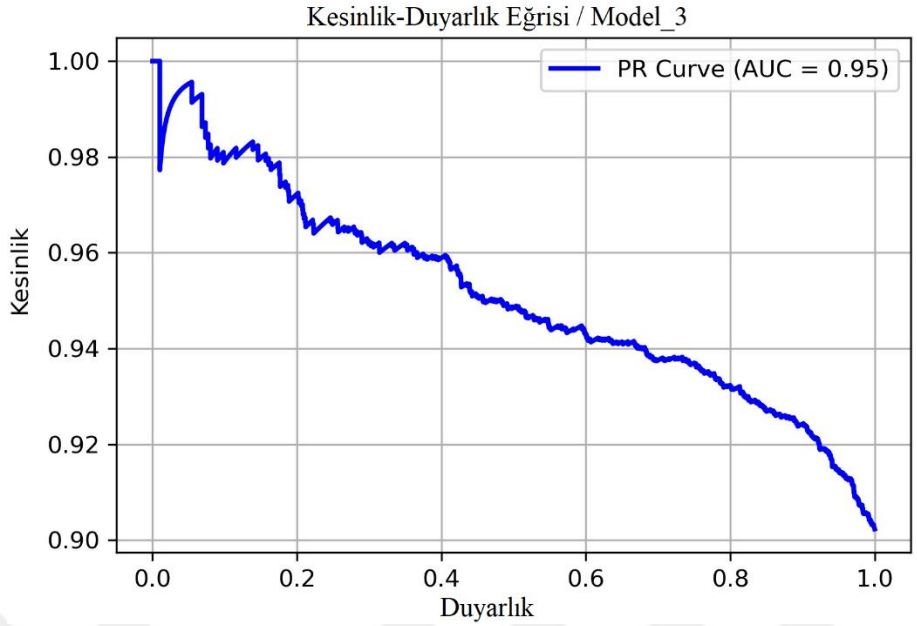
EK A.4


Kesinlik-Duyarlık Eğrisi / Model_1
PR Curve (AUC = 0.95)
Kesinlik
Duyarlık
1.00
0.98
0.96
0.94
0.92
0.90
0.0
0.2
0.4
0.6
0.8
1.0

(a)


Kesinlik-Duyarlık Eğrisi / Model_2
PR Curve (AUC = 0.95)
Kesinlik
Duyarlık
1.00
0.98
0.96
0.94
0.92
0.90
0.0
0.2
0.4
0.6
0.8
1.0

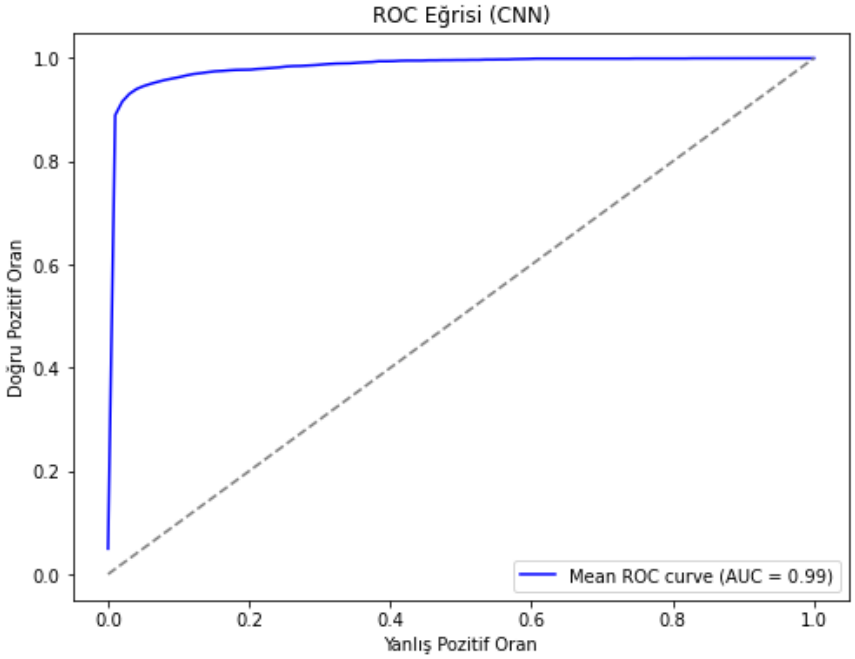
(b)



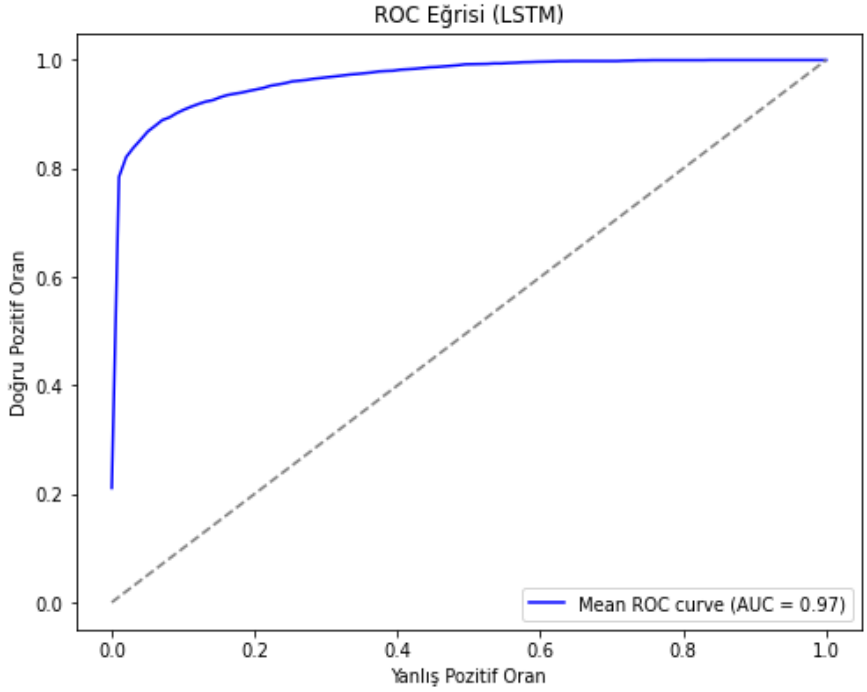
(c)

Şekil A.4 Graf evrişimli ağ modellerine ait PR eğrileri

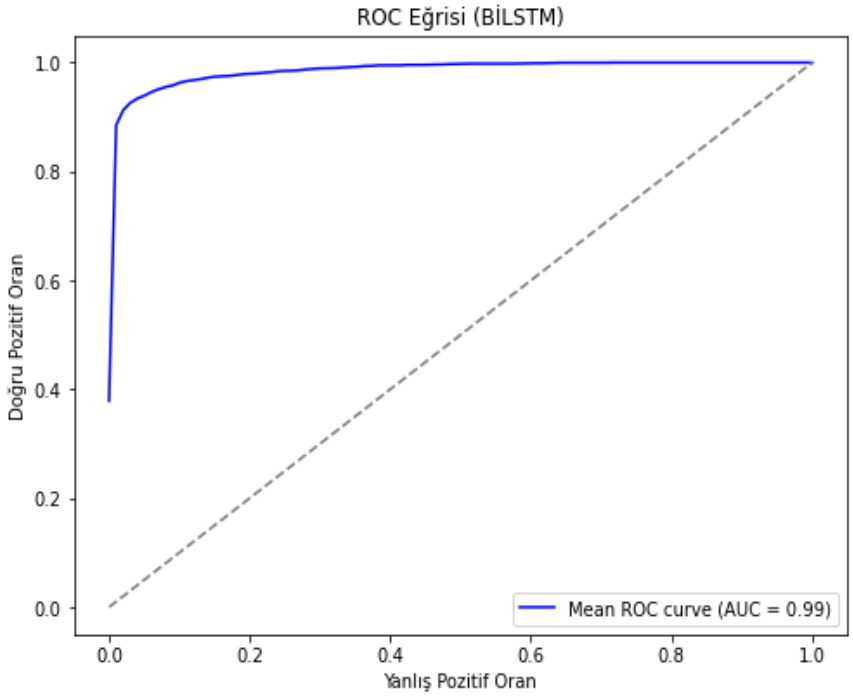
EK A.5


ROC Eğrisi (CNN)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.99)
0.0
0.2
0.4
0.6
0.8
1.0

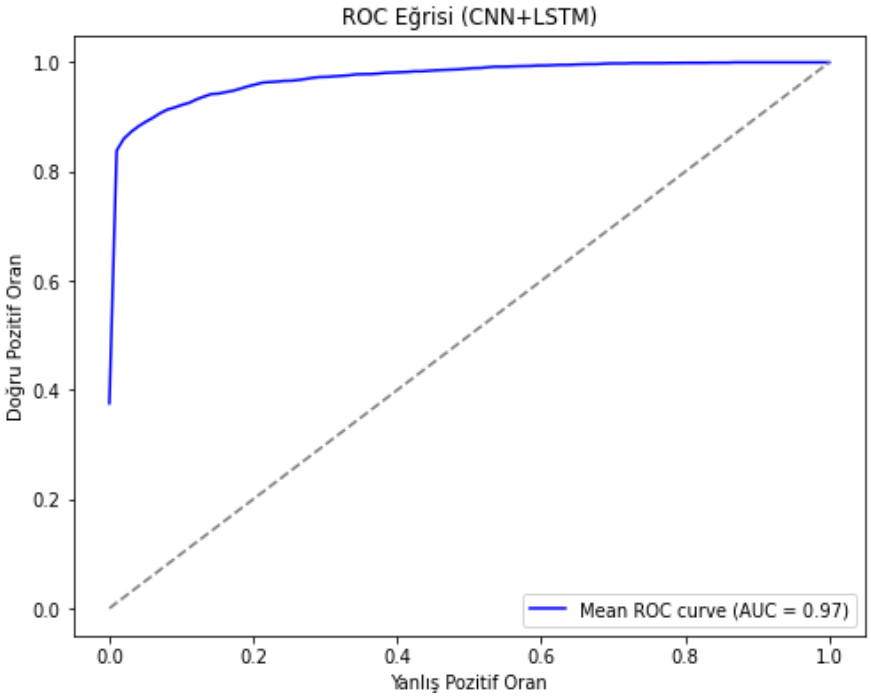
(a)


ROC Eğrisi (LSTM)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.97)
0.0
0.2
0.4
0.6
0.8
1.0

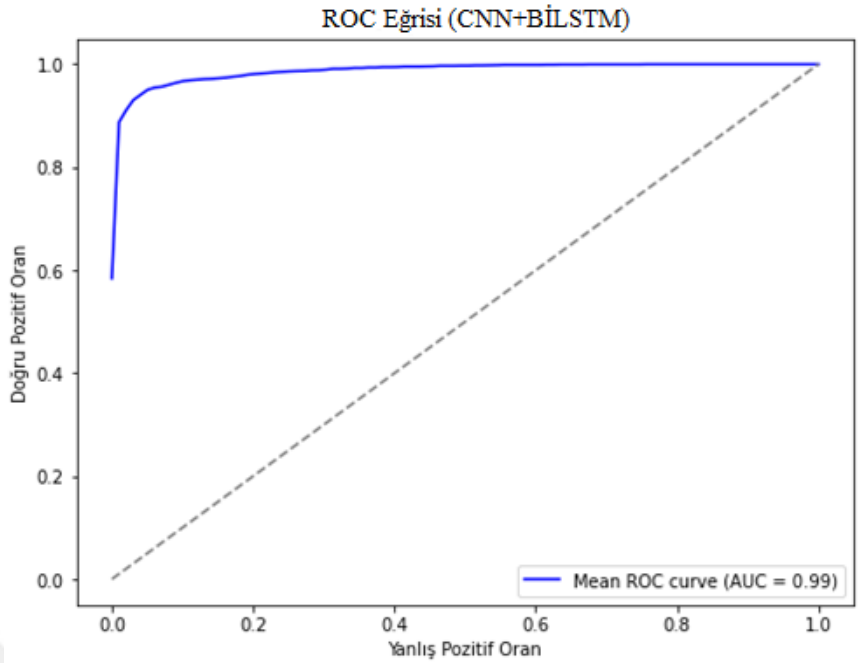
(b)


ROC Eğrisi (BİLSTM)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.99)
0.0
0.2
0.4
0.6
0.8
1.0

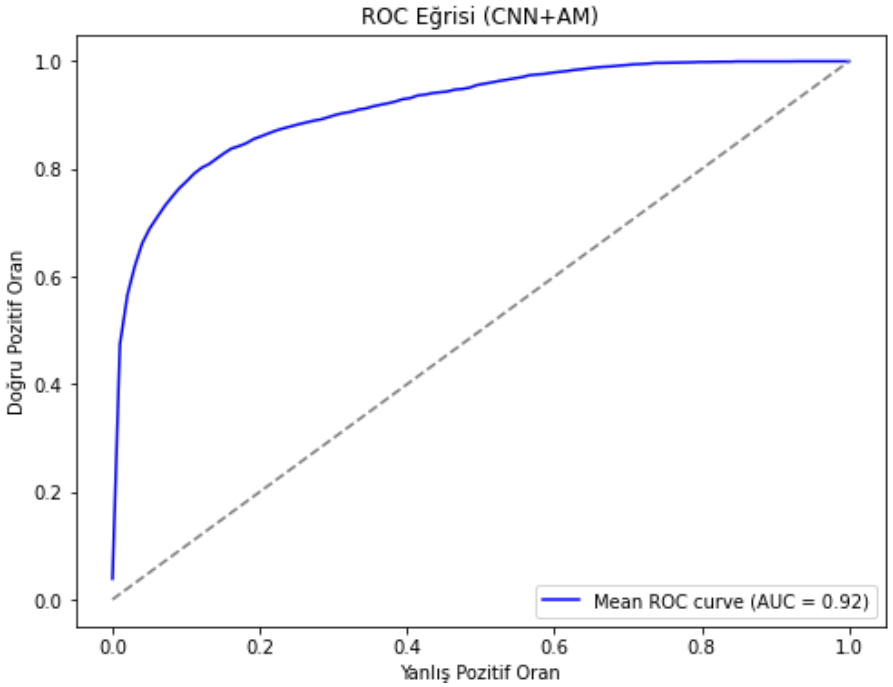
(c)


ROC Eğrisi (CNN+LSTM)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.97)
0.0
0.2
0.4
0.6
0.8
1.0

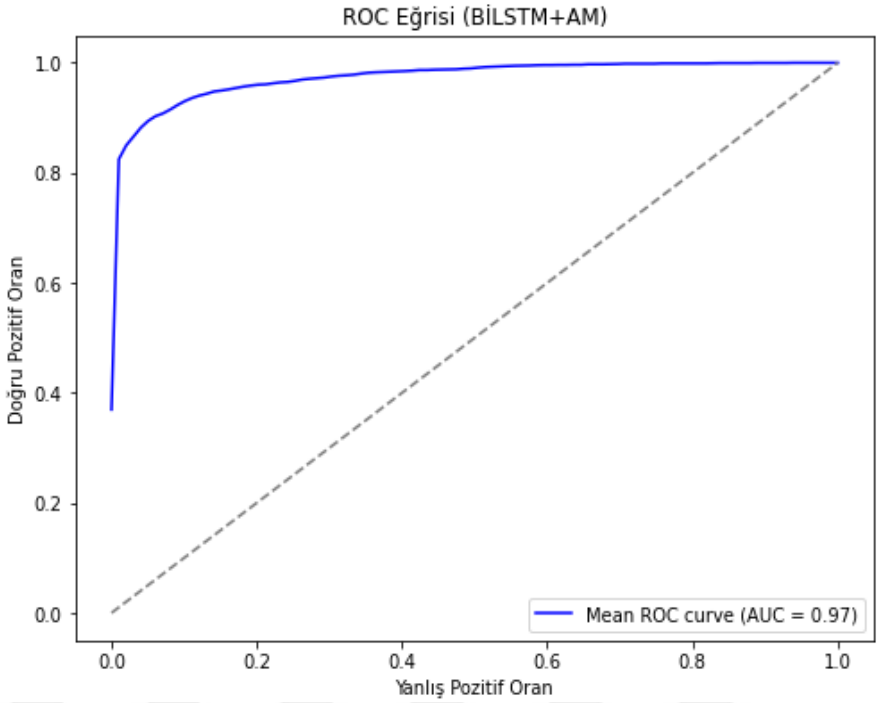
(d)


ROC Eğrisi (CNN+BİLSTM)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.99)
0.0
0.2
0.4
0.6
0.8
1.0

(e)


ROC Eğrisi (CNN+AM)
Doğru Pozitif Oran
Yanlış Pozitif Oran
Mean ROC curve (AUC = 0.92)
0.0
0.2
0.4
0.6
0.8
1.0

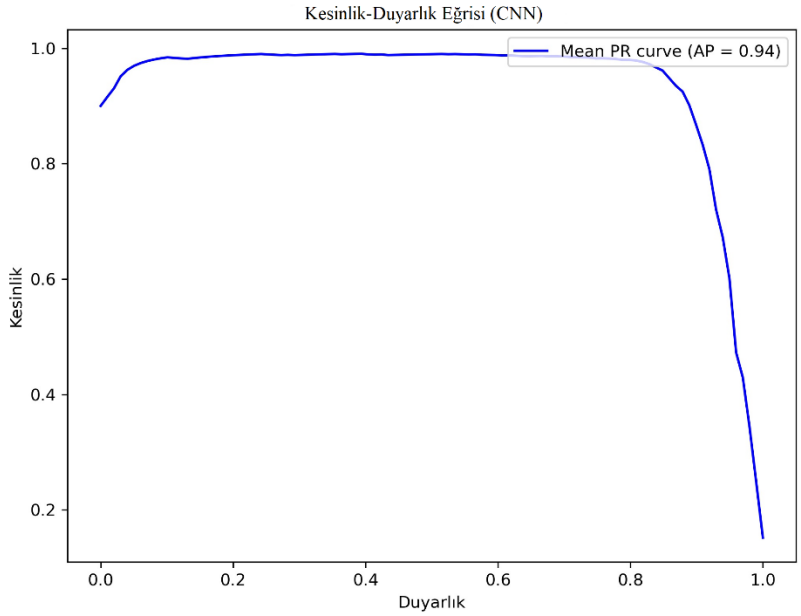
(f)



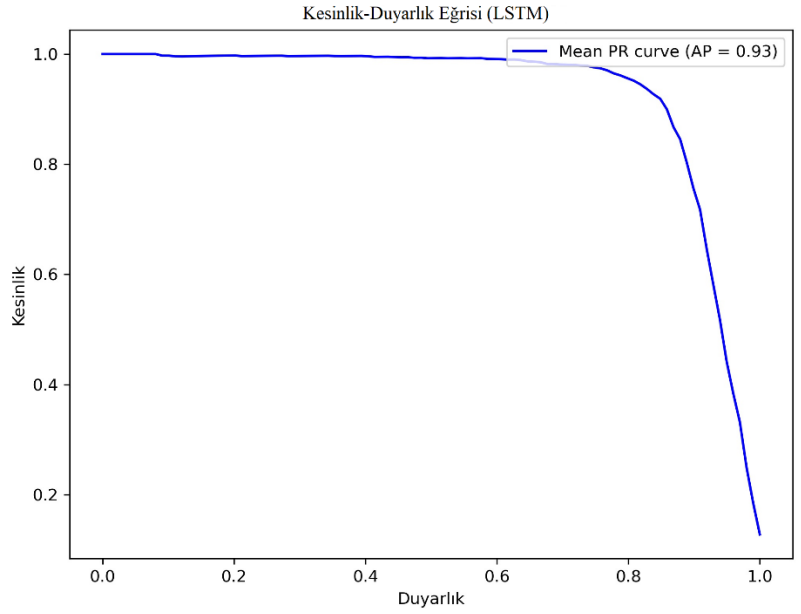
(g)

Şekil A.5 Derin öğrenme modellerine ait ROC eğrileri

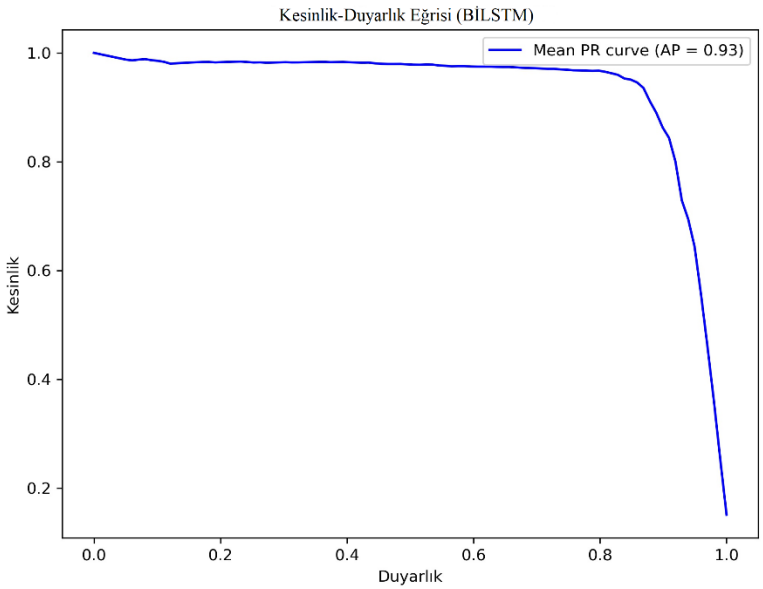
EK A.6


Kesinlik-Duyarlık Eğrisi (CNN)
Mean PR curve (AP = 0.94)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

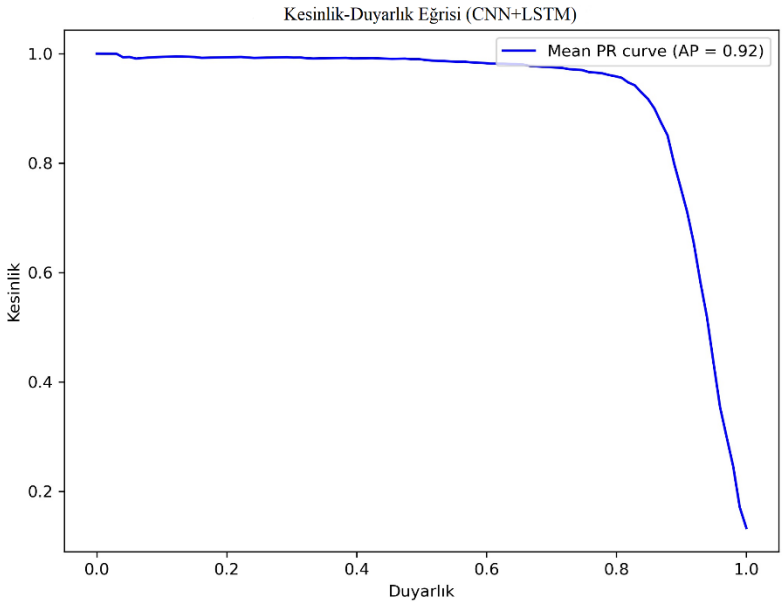
(a)


Kesinlik-Duyarlık Eğrisi (LSTM)
Mean PR curve (AP = 0.93)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

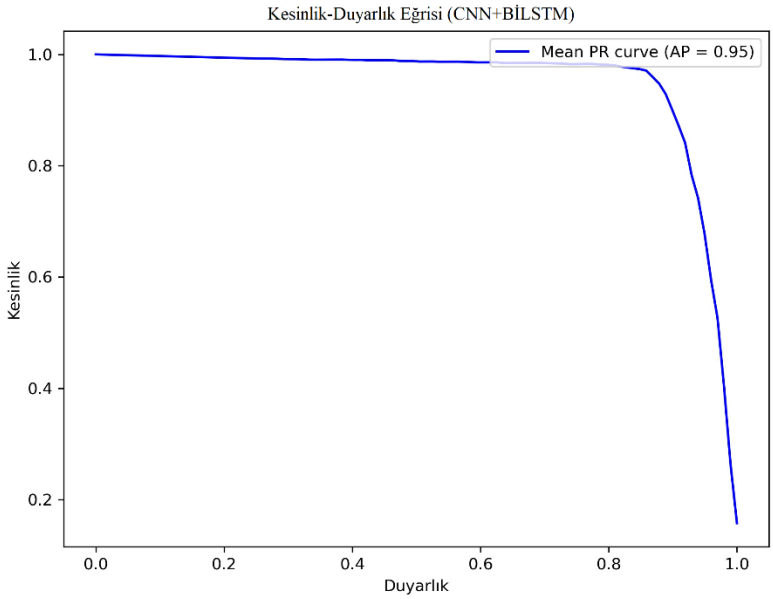
(b)


Kesinlik-Duyarlık Eğrisi (BİLSTM)
Mean PR curve (AP = 0.93)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

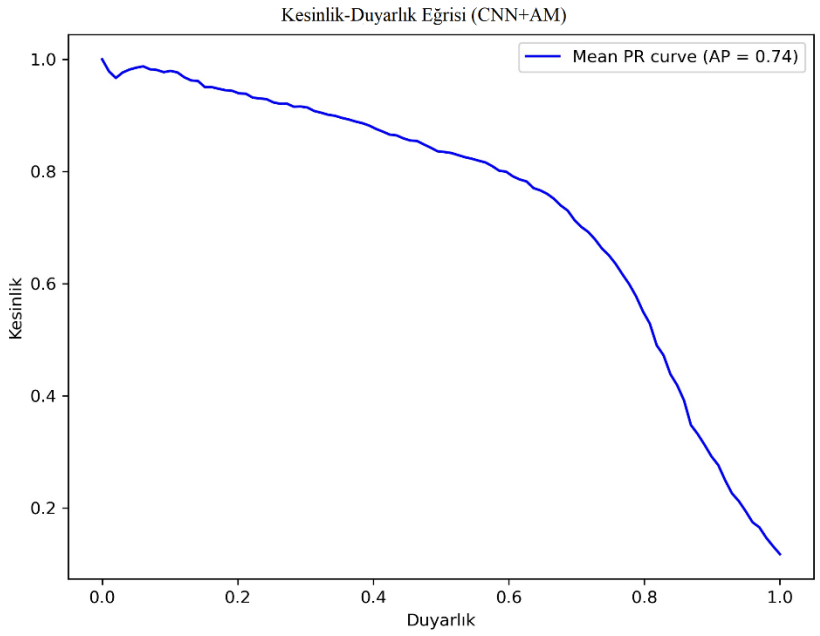
(c)


Kesinlik-Duyarlık Eğrisi (CNN+LSTM)
Mean PR curve (AP = 0.92)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

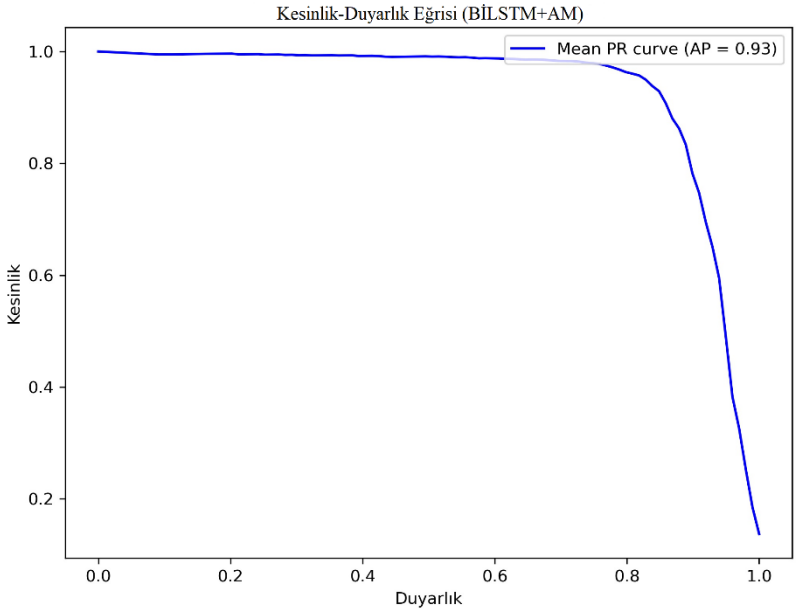
(d)


Kesinlik-Duyarlık Eğrisi (CNN+BİLSTM)
Mean PR curve (AP = 0.95)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

(e)


Kesinlik-Duyarlık Eğrisi (CNN+AM)
Mean PR curve (AP = 0.74)
Kesinlik
Duyarlık
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

(f)



(g)

Şekil A.6 Derin öğrenme modellerine ait PR eğrileri

ÖZGEÇMİŞ

2010 yılında Mevlana Üniversitesi Bilgisayar Mühendisliği Bölümüne girdi ve 2014 yılında bölüm birincisi olarak mezun oldu. 2020 yılında Eskişehir Osmangazi Üniversitesi Bilgisayar Mühendisliği Bölümünde yüksek lisans eğitimini tamamladı. 2020 yılından beri Yalova Üniversitesi Bilgisayar Mühendisliği Bölümünde Araştırma Görevlisi olarak çalışmaktadır.

TEZDEN TÜRETİLEN YAYIN VE ESERLER

Proje:

Blok Zincir Ağları Üzerinde Yasal Olmayan Para Transferlerinin Graf Teorisi ve Yapay Zekâ Yöntemleri ile Tespit Edilmesi (2025) (123E312 numaralı Tübitak 1002/A).

Makale:

Cengiz, E., Gök, M., (2025). Anti Money Laundering in Bitcoin Network Using Chaotic Time Series and Graph Convolution Network. *Journal of Universal Computer Science,* 31(11), 1175-1195. (SCI-E). DOI: 10.3897/jucs.135907

Cengiz, E., & Gök, M. (2025). Prediction of Money Laundering with Machine Learning Methods Using Chaotic Features. *The Computer Journal*, (SCI-E). DOI: 10.1093/comjnl/bxaf108

Cengiz, E., Gök, M. (2026). A Hybrid Deep Learning Framework with Attention Mechanism for Anti Money Laundering in Cryptocurrency Transactions. *Sigma Journal of Engineering and Natural Sciences,* 44(3). (ESCI / Kabul).

Bildiri:

Cengiz, E., Gök, M. (2024). *Prediction of Money Laundering in the Bitcoin Network Using Chaotic Based Features*, International Conference on Data Science and Applications (ICONDATA), Pristina, Kosovo.

DİĞER YAYIN VE ESERLER

Makale:

Al-Kateb, G., Cengiz, E., Gök, M. (2025). BioGPT: A Generative Transformer-Based Framework for Personalized Genomic Medicine and Rare Disease Diagnosis. *Mesopotamian Journal of Artificial Intelligence in Healthcare*, 2025, 154-164.

Cengiz, E., Yaylak, F., & Gülbandılar, E. (2022). Investıgatıon of Polyps in Endoscopy Images By Usıng Deep Learnıng Algorıthm. *Eskişehir Osmangazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi*, 30(3), 441-453.

Cengiz, E., & Harman, G. (2022). Dengesiz Ml-Tabanlı Nıds Veri Setlerinin Sınıflandırma Performanslarının Karşılaştırılması. *Avrupa Bilim Ve Teknoloji Dergisi*, (41), 349-356.

Cengiz, E., Gök, M. (2023). Reinforcement Learning Applications in Cyber Security: A Review. *Sakarya University Journal of Science*, 27(2), 481-503.

Gök, M., Cengiz, E., Mijwil, MM. (2024). Unveiling Protein-Ligand Interactions: Regression Methods for Binding Affinity Prediction. *International Journal on Engineering Applications*, 12(6), 508-513.

Harman, G., Cengiz, E. (2024). Deep Learnıng Based Network Intrusıon Detection. *Mühendislik Bilimleri ve Tasarım Dergisi*, 12(3), 517-530.

Mijwil, MM., Sadıkoğlu, E., Cengiz, E., Candan, H. (2022). Siber Güvenlikte Yapay Zekanın Rolü ve Önemi: Bir Derleme. *Veri Bilimi*, 5(2), 97-105.

Bildiri:

Gülbandılar, E., Cengiz, E., Yaylak, F., (2020). *Endoskopi Görüntülerindeki Polipli Hücrelerin Derin Öğrenme Algoritması Kullanılarak İncelenmesi,* Sağlıkta Yapay Zeka Kongresi, İzmir, Türkiye.

Cengiz, E., Gök, M., (2021). *The Effect of Deep Learning Model Parameters on Model Performance,* International Conference on Interdisciplinary Applications of Artificial Intelligence (ICIDAAI), Çevrimiçi.

Cengiz, E., Gök, M., (2021). *Reinforcement Learning for Cyber Security,* International Conference on Data Science and Applications (ICONDATA), Çevrimiçi.