TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME ile ÇİZGE ZAMAN SERİLERİNİN ANALİZİ

YÜKSEK LİSANS TEZİ

Mustafa Mert KESKİN

Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Ahmet Murat ÖZBAYOĞLU

NİSAN 2022

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Mustafa Mert Keskin

İmza

ÖZET

Yüksek Lisans

DERİN ÖĞRENME ile ÇİZGE ZAMAN SERİLERİNİN ANALİZİ

Mustafa Mert KESKIN

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Ahmet Murat ÖZBAYOĞLU

Tarih: Nisan 2022

Zaman serileri bir nesnenin zamansal değişimini anlamak için kullanılır. Finans, enerji sektörü, trafik gibi çeşitli alanlarda zaman serileri ile karşılaşılmaktadır. Ayrıca anomali tespiti, davranış tanıma gibi birçok problem, zaman serisi problemi olarak modellenebilir. Bu yüzden gerçek hayatta sıkça karşılaşılan zaman serilerinin analizi büyük öneme sahiptir. Dolayısıyla, zaman serisi problemleri yaygın bir şekilde araştırılmakta ve çözülmeye çalışılmaktadır. Çizgeler ise nesneler arası ilişkileri analiz etmek için kullanılır. Bazı zor problemler, problem verisi çizge olarak analiz edildiği zaman daha iyi anlaşılabilir. Bu yüzden çizge problemleri de literatürde önemli bir yer sahiptir.

Zaman serileri ve çizgeler problemin farklı yönlerini anlama imkânı sunarlar. Bu yüzden, literatürde zaman serilerini çizgeler ile birleştirerek daha iyi modelleme yapan çalışmalar mevcuttur. Bu yöntem finans alanında bazı çalışmalarda uygulanmaktadır. Bu tez çalışmasında, finansal tahmin problemi için derin öğrenme yöntemleri ile çizge serisi analizi yapılmıştır. Bunun için öncelikle DOW 30 borsası bir çizge olarak temsil edilmiştir. Sonrasında farklı zaman anlarındaki çizgeler sıralanarak çizge serisi oluşturulmuştur. Elde edilen seri ile yapay sinir ağı eğitilerek

hisselerin değişim miktarı tahmini yapılmıştır. Tahmin edilen değişim miktarına göre ana paradan günlük al/sat stratejisi uygulanarak yatırım yapılmıştır. Bunun sonucunda yıllık getiri yüzde olarak hesaplanmıştır. Araştırma sonucunda, sadece zaman serisi kullanılarak geliştirilen derin öğrenme modellerine kıyasla daha yüksek ortalama yıllık getiri kazanılmış olup çizge serisi kullanmanın finansal tahmini ciddi ölçüde iyileştirdiği bir başka deyişle zaman serisi ile yakalanamayacak çıkarımların yapılabildiği sonucuna varılmıştır.

Tez çalışmasında, birden çok yöntemle çizge serisi oluşturulmuştur. Farklı çizgelerle eğitilen derin öğrenme modelleri ile benzer ortalama yıllık getiri elde edilmiştir. Böylece, çizge serisi elde etme yönteminin güçlü (robust) ve kararlı (stable) bir yöntem olduğu gösterilmiştir. Ayrıca eğitilen derin öğrenme modellerinin çıktılarından en çok artan hisse tahminin yapan bir kolektif model eğitilmiştir. Nihai model ile ortalama yıllık %26,68 kazanç elde edilmiştir. Bu yöntemin literatürdeki temel yöntemlerin yanı sıra çeşitli açgözlü (greedy) algoritmadan da daha yüksek getiri sağladığı gösterilmiştir. Sonuç olarak geliştirilen kolektif model, gerçek hayatta günlük al/sat stratejisi için kullanılabilecek bir yöntem olarak önerilmiştir.

Anahtar Kelimeler: Zaman serisi, Çizge teori, Yapay öğrenme, Yapay sinir ağları, Derin öğrenme, Kolektif öğrenme

ABSTRACT

Master of Science

ANALYSİS OF GRAPH TIME SERIES WITH DEEP LEARNING

Mustafa Mert KESKIN

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Assoc. Prof. Ahmet Murat ÖZBAYOĞLU

Date: April 2022

Time series are used to understand the temporal variation of an object. Time series are encountered in various fields such as finance, energy and traffic. Moreover, many problems such as anomaly detection, behavior recognition can be modeled as time series problems. Therefore, the analysis of time series which are frequently encountered in real life is of great importance and time series problems are widely researched and tried to be solved. Graphs are used to analyze the relationships between objects. Some difficult problems can be better understood when the problem data is analyzed as graphs. Hence, graph problems have an important place in the literature.

Time series and graphs provide an opportunity to understand different aspects of the problem. Therefore, there are studies in the literature that make better modeling by combining time series and graphs. This method is applied in some studies in the field of finance. In this thesis, graph series analysis was performed with deep learning methods for financial forecasting problems. For this purpose, DOW 30 stock market is represented as a graph. Then, graphs at different timestamps were ordered and graph series was formed. The amount of change in the shares was predicted by training artificial neural networks with obtained series. According to the predicted amount of change, the principal was invested by applying a daily buy/sell strategy.

As a result of this, the annual return was calculated as a percentage. As a result of the study, higher average annual returns were obtained compared to deep learning models using only time series and it was concluded that using graph series significantly improved financial forecasting, in other words, inferences that could not be captured with time series could be made.

In the thesis study, graph series was created with multiple methods. A similar average annual return was obtained with deep learning models trained with different graphs. Thus, it has been shown that the method of obtaining a series of graphs is a robust and stable method. In addition, an ensemble model that predicts the stock that increases the most from the outputs of the trained deep learning models is trained. An average annual return of 26.68 % was achieved with the final model. It has been shown this method provides more profit than various greedy algorithms in the literature as well as the basic methods in the literature. As a result, the developed ensemble model is proposed as a method that can be used for daily buy/sell strategy in real life.

Keywords: Time series, Graph theory, Machine learning, Artifical neural network, Deep learning, Ensemble learning

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Doç. Dr. Ahmet Murat Özbayoğlu‘na, kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Bölümü öğretim üyelerine ve destekleriyle her zaman yanımda olan aileme ve arkadaşlarıma çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

KISALTMALAR

AE : Özkodlayıcı (Autoencoder)
ANN : Yapay Sinir Ağları (Artificial Neural Networks)
CNN : Evrişimsel Sinir Ağı (Convolutional Neural Network)
DL : Derin Öğrenme (Deep Learning)
DMLP : Derin Çok Katmanlı Perceptron (Deep Multilayer Perceptron)
DNN : Derin Sinir Ağları (Deep Neural Networks)
FCNN : Tamamen Bağlı Sinir Ağı (Fully Connected Neural Network)
FN : Yanlış Negatif (False Negative)
FP : Yanlış Pozitif (False Positive)
GA : Genetik algoritma (Genetic Algortihm)
GCN : Çizge Evrişimsel Ağı (Graph Convolutional Network)
GNN : Çizge Sinir Ağları (Graph Neural Networks)
LSTM : Uzun Ömürlü Kısa Dönemli Bellek **(**Long Term Short Memory)
MA : Hareketli Ortalama (Moving Average)
MAE : Ortalama Mutlak Hata (Mean Absolute Error)
ML : Makine Öğrenme (Machine Learning)
MLP : Çok Katmanlı Perceptron (Multilayer Perceptron)
MSE : Ortalama Kare Hata (Mean Square Error)
PCA : Temel Bileşenler Analizi (Principal Component Analysis)
ReLU : Doğrusal Doğrultmaç Ünitesi (Rectified Linear Unit)
RMSE : Ortalama Hata Kare Kare Kökü (Root Mean Square Error)
RNN : Özyinelemeli Sinir Ağı (Recurrent Neural Network)
TN : Doğru Negatif (True Negative)
TP : Doğru Pozitif (True Positive)
WT : Wavelet Dönüşümü (Wavelet Tranformation)

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
Σ	Toplam
μ	Ortalama değer
σ	Standart Sapma
∇	Gradyan
J	Hata Fonksiyonu
w_t	Modelin ağırlığı
x_t	Zaman serisi olarak verilen öznitelik vektörünün t anındaki değeri
$y_{t:}$	Zaman serisinin t anındaki değeri

1. GİRİŞ

Çizge tabanlı algoritmalar gerçek hayatta birçok problemi daha kolay bir şekilde ifade edebilmektedir. Günümüzde makine öğrenmesi ve derin öğrenme yöntemlerinin yaygınlaşmasıyla birlikte çizge zaman serisini öğrenebilen derin öğrenme metotlarının önemi de ciddi miktarda artmaktadır. Çizge tabanlı algoritmalar, uzunca bir süredir bilgisayar biliminin bünyesinde yer alan algoritmalardır. Ancak yapay zeka (Artificial Intelligence, AI) ve derin öğrenme (Deep Learning, DL) modelleri ile bir araya gelmeleri bilgisayar gücünün yetersizliği ve bazı algoritmaların daha geç keşfedilmesinden dolayı uzun sürmüştür.

Zaman serileri analizi çok uzun yıllardır araştırılmakta olan bir konu olmakla birlikte birçok klasik yöntem geliştirilmiştir. Bu yöntemlerden bazıları tez içerisinde verilecektir. AI ve DL modellerinin ilerlemesiyle bilgisayarların işlem gücünün kullanılmasıyla birlikte birçok kendi kendine öğrenebilen ve tahmin yapabilen modeller geliştirilmiştir. Bu yöntemler birçok alandaki zaman serileri problemlerine uygulanabilmektedir. Örnek olarak yaygın bir hastalığın aktif vaka sayısını, borsa üzerindeki bir hisse senedinin gelecekteki değerini, marketlerin gelecekte hangi ürünü ne kadar satması gerektiğini tahmin edebilen modeller bunlara örnek olarak verilebilir.

Araştırmanın amacı derin öğrenme (DL) ile çizge zaman serilerinin analizini yapabilen algoritmalar geliştirmek, deneysel olarak sonuçlar almak ve bu sonuçları yorumlamaktır.

1.1 Problem ve Motivasyon

Bu çalışmada zaman serilerinin çizge tabanlı algoritmalar ile modellenmesi amaçlanmaktadır. Gerçek hayatta birçok zaman serisi uygulaması bulunduğundan dolayı bir alt konu olarak borsa üzerinde çizge tabanlı algoritmaların kullanılması planlanmıştır.

Çizge tabanlı algoritmaların son zamanlarda borsa üzerinde uygulanabilir olduğunu gösteren çalışmalar bu çalışma için önemli bir motivasyon oluşturmaktadır. Çizge tabanlı algoritmaların borsa üzerinde zaman serisi yaklaşımlarıyla birleştirilip borsa üzerinde tahmin yapabilen modeller geliştirmek bu tezin hedefi olarak belirlenmiştir.

1.2 Önerilen Çözümler ve Katkıları

Literatürde borsanın gelecekteki değişimlerini tahmin edebilen model tasarlamak zor bir konu olarak görülmektedir. Bunun sebepleri borsa üzerindeki hareketlerin çok fazla sayıda parametreye bağlı olmasıdır. Aynı zamanda borsa üzerinde birden çok hisse senedi bulunmaktadır. Bunlardan hangilerinin ne zaman artacağını modellemek daha zor bir konu olarak görülebilir.

Çizge tabanlı algoritmalar kullanarak birden çok hisse senedinin zaman içerisindeki hareketlerinin gözlemlenebileceği hakkında fikir veren çalışmalar yapılmıştır. Bu çalışmalar literatür taraması kısmında açıklanacaktır. Literatürde yapılan bu çalışmaların üstüne çıkarak ve çizge tabanlı derin öğrenme modelleri kullanarak birden çok hisse senedini modelleyen yaklaşımlar kullanmak bu çalışmanın önerdiği çözüm yolu olarak sunulmaktadır.

Borsa üzerinde tahmin yapabilen bir model geliştirmek hem akademik açıdan hem de yatırım yapmak isteyenler açısından önemlidir. Yapay zeka algoritmalarının gelişmesiyle birlikte birçok çalışma finans ve borsa üzerinde modellemeler yapmıştır fakat bu algoritmaların büyük bir çoğunluğu çizge yöntemlerini kullanmamıştır. Birden çok hisse senedinin birbiriyle olan ilişkilerini kullanmak gelecekte iyi tahmin yapabilen bir model geliştirmek için önemlidir. Bu modellemelerin detayları ilerleyen bölümlerde açıklanacaktır. Bu çalışma genel olarak literatüre şu katkıları sağlamıştır:

- Çizge tabanlı derin öğrenme metotları kullanarak borsa hisse senetleri üzerinde modelleme yapılabileceği gösterilmiştir.
- Çizge tabanlı modellerin zaman serisi analizi için kullanılabileceği ve bunun nasıl yapılacağı gösterilmiştir.
- Çizge tabanlı modellerin klasik bazı DL metotlarından daha başarılı sonuçlar verdiği gösterilmiştir.

- Çizge tabanlı modellerin etkili olabilmesi için gerekli olan çizge taban gösterimleri analiz edilmiş ve sonuçlar çıkarılmıştır.
- Borsa üzerinde geliştirilen DL yöntemlerini iyileştiren ek bazı yöntemler tanıtılmıştır ve sonuçları elde edilmiştir.

1.3 Tez Taslağı

Bu tez şu şekilde organize edilmiştir:

- 2.bölüm olan kuramsal çerçeve 5 ana bölümden oluşmaktadır. Bu bölümde bu çalışma için gerekli olan temel bilgiler anlatılmaktadır. Bunlar:
  - Çizge
  - Finans ve zaman serileri
  - Ön-işleme
  - Makine öğrenmesi
  - Kolektif öğrenme
- 3.bölüm tez için gerekli olan literatür taraması bilgisini anlatmaktadır. Bu bölümde borsada kullanılan veri setleri, makine öğrenmesi yöntemleri, derin öğrenme modelleri ve çizge tabanlı öğrenme modelleri detaylı bir şekilde anlatılmaktadır.
- 4.bölüm bu tez kapsamında zaman serilerinin çizge ile ifade edilerek derin öğrenme modelleri ile eğitilmesini anlatmaktadır. Burada kullanılan temel yöntemler, veri setinin oluşturulması, yapılan ön işlemeler, borsanın çizgeye dönüştürülmesi ve derin öğrenme modelleri uygulanarak eğitilmesi gibi içerikler detaylı bir şekilde anlatılmaktadır.
- 5.bölümde bu tez kapsamında yapılan çalışmalar ve sonuçları karşılaştırılmalı olacak şekilde anlatılmaktadır.
- 6.bölümde yapılan çalışma özetlenmektedir ve gelecekte yapılabilecek çalışmalardan bahsedilmektedir.

2. KURAMSAL ÇERÇEVE

Tezin konusu çizge tabanlı zaman serisi öğrenebilen modeller geliştirmek ve bu modellerin performanslarını klasik DL metotları ile karşılaştırmak olarak belirlenmiştir. Zaman serileri çok genel kavramlar olduğundan dolayı bir alt konu olarak finans seçilmiş ve geliştirilen modellerin finans üzerinde denenmesi amaçlanmıştır.

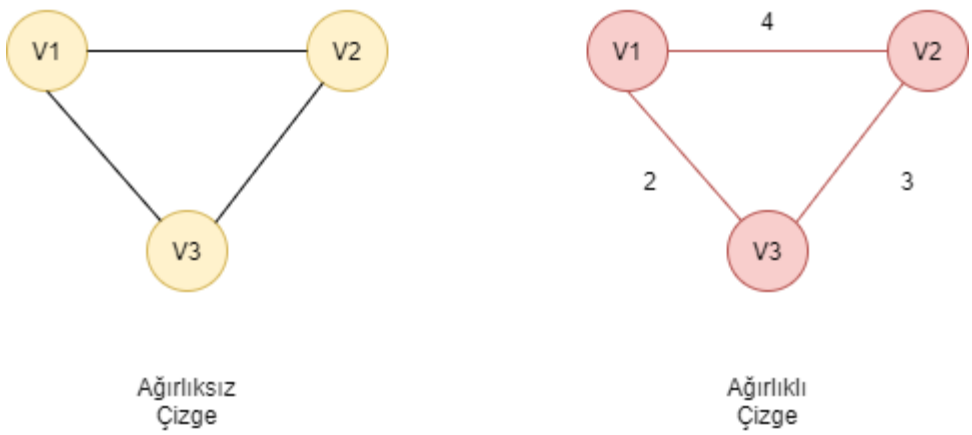
2.1 Çizge

Çizge (Graph) matematiksel anlamda düğümlerden ve kenarlardan oluşan bir kümedir. Matematiksel olarak $G = (V, E)$ şeklinde tanımlanabilir. Eğer A düğümüyle B düğümü arasında bir kenar tanımlanmışsa $e = (A, B)$ bu iki düğüm birbirinin komşusu (adjacent) olarak tanımlanır. Çizgeler iki gruba ayrılmaktadır

- Ağırlıklı çizgeler (Weighted Graph)
- Ağırlıksız çizgeler (Unweighted Graph)

Ağırlıksız çizgelerde iki düğüm arasında bir kenar varsa bu iki düğümün birbiriyle bir bağlantısı yani ilişkisi vardır. Eğer iki düğüm arasında bir ilişki yoksa bu iki düğüm arasında bir kenar oluşmayacaktır. Ağırlıklı çizgelerdeki kenarların bir ağırlığı olmaktadır. Bu değer iki düğüm arasındaki ilişkinin gücünü göstermektedir.

Şekil 2.1'de ağırlıksız ve ağırlıklı çizgeler farkı şekil üzerinde gösterilmiştir. Burada V1, V2 ve V3 düğümlerinin kenar değerleri ağırlıklı çizgede sırasıyla 2, 3 ve 4 iken ağırlıksız çizgede herhangi bir değere sahip değildir. Burada ağırlıklı çizge düğümler arasındaki ilişkiyi daha detaylı bir şekilde yansıtmaktadır. Bazı problemlerde bu ağırlıkların önemi bulunurken bazı problemlerde bu ağırlıklara gerek olmayabilir. Bu çalışma alanından alana ve uygulanmak istenen probleme göre şekillenebilir.

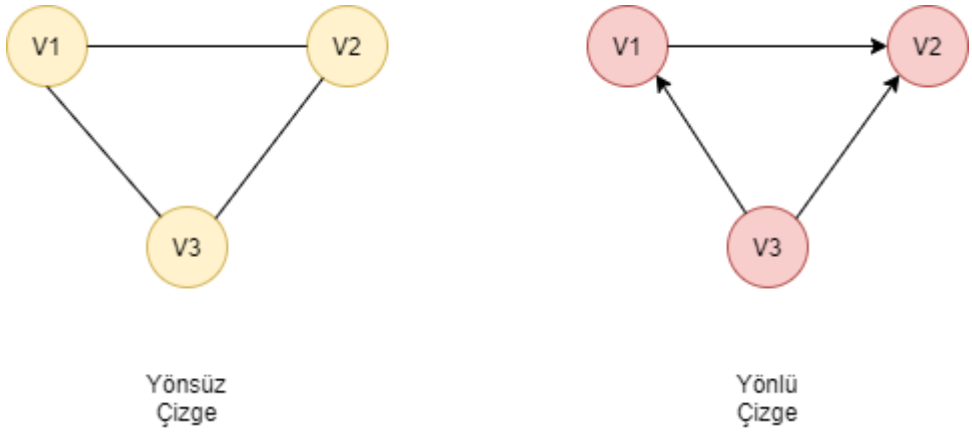


Şekil 2.1: Ağırlıklı çizge ve ağırlıksız çizgenin karşılaştırılmalı örneği.

Çizgeler ağırlıkları haricinde kenarların yönlerine göre de ikiye ayrılmaktadır:

- Yönlü çizge (Directed Graph)
- Yönsüz çizge (Undirected Graph)

Yönsüz çizgelerde kenarlar herhangi bir yöne sahip değildir. Matematiksel olarak u düğümüyle v düğümü arasında oluşan kenar (u, v) ile v düğümüyle u düğümü arasında oluşan kenar (v, u) aynı şeyi ifade etmektedir. Yönlü çizgelerde ise bu iki kenar aynı şeyi ifade etmemektedir. Bu iki kenar arasında her zaman aynı ilişki olmayabilir.



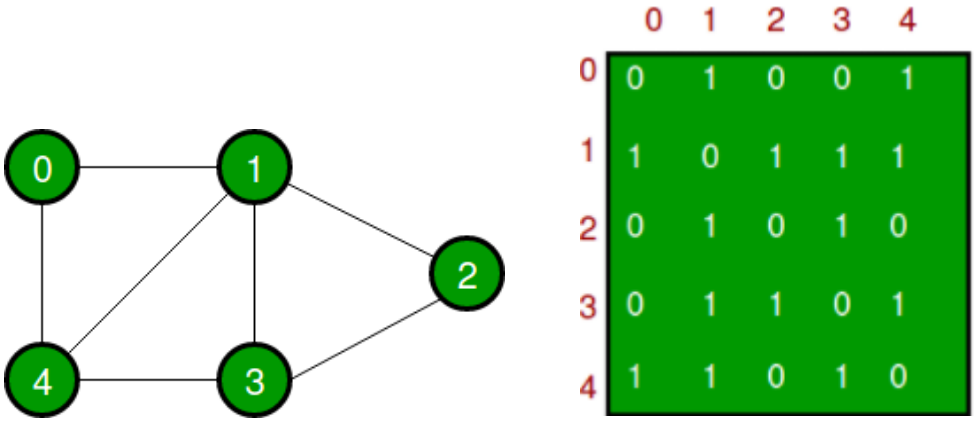
Şekil 2.2: Yönsüz çizge ve yönlü çizgenin şekil üzerinde gösterimi.

Şekil 2.2'de yönsüz ve yönlü çizge arasındaki fark gösterilmiştir. Bu şekildeki V1 ve V2 düğümleri arasındaki ilişki incelendiğinde yönlü çizgede V3'ten V1'e giden bir kenar görünürken V1'den V3'e giden bir kenar görülmemektedir. Burada çift taraflı bir ilişkinin olmadığı ortaya çıkmıştır. Yönsüz çizge incelendiğinde bu şekilde bir

fark görülmemektedir. Yönsüz çizgelerde bu yön bilgisinin bir değeri yoktur. Yönsüz ve yönlü çizgelerin kullanımı problemden probleme değişmektedir.

Bir çizge farklı veri yapılarıyla temsil edilebilir ve üzerinde işlemler yapılabilir. Bu çalışmada komşuluk matrisi gösterimi kullanılmıştır. N bir çizgedeki düğüm sayılarını ifade etmektedir. D bir matris olmak üzere NxN boyutundadır ve D_{ij} bu matrisin her bir elemanın değerini vermektedir. Burada D_{ij} çizge üzerindeki i ve j düğümleri arasındaki kenarların ağırlığını göstermektedir. Ağırlıksız bir çizge üzerinde kenarları olan düğümler için D_{ij} değerleri 1'e eşit olmaktadır. Kenarları olmayan düğümler arasında Dij ise 0' a eşit olmaktadır. Ağırlıklı bir çizge üzerinde Dij değerleri herhangi bir değer alabilir ve bu değerler problemden probleme değişebilir.

Şekil 2.3 üzerinde örnek olarak verilmiş bir çizge ve onun komşuluk matrisi görülmektedir [1]. Yönsüz çizgelerin komşuluk matrisinde Dij ve Dji aynı değere sahip olacaktır. Yönlü çizgelerde ise bu değerler birbirine eşit olmak zorunda değildir. Bir çizge eğer basit yapıda bir çizge ise D_{ii} sıfıra eşit olacaktır çünkü bir düğümden kendine bir kenar oluşamaz. Komşuluk matrisi sadece bir çizgenin anlaşılmasını sağlamakla kalmaz aynı zamanda doğrudan bir çizgeyi matris olarak kullanarak matematiksel kolaylıklar da sağlayacaktır.



Şekil 2.3**:** Komşuluk matrisinin şekil üzerinden örneği.

2.2 Finans ve Zaman Serisi

Zaman serilerinin matematiksel olarak geçmişi çok eskilere dayanmaktadır. Geçmişten günümüze kadar insanlar gelecek üzerine tahminleme yapma ve yorumlama üzerinde türlü çalışmalar yapmıştır. Özellikle bilgisayarların hesaplama gücünün artması ile birlikte bazı zaman serisi problemleri ortaya çıkmaya başlamıştır [2].

Zaman serileri zaman içerisinde değişime uğrayan, belirli bir değer kazanan ya da kaybeden ifadeler olabilir. Bu matematiksel ifadeler olabileceği gibi sosyal hayattan bazı ifadeler de olabilir. Bir sonraki günün hava durumunu, borsada bir hisse senedinin artacağı ya da azalacağını veya dünyadaki nüfusun gelecekteki değerini tahmin etmek basitçe bazı zaman serilerine örnek olarak verilebilir.

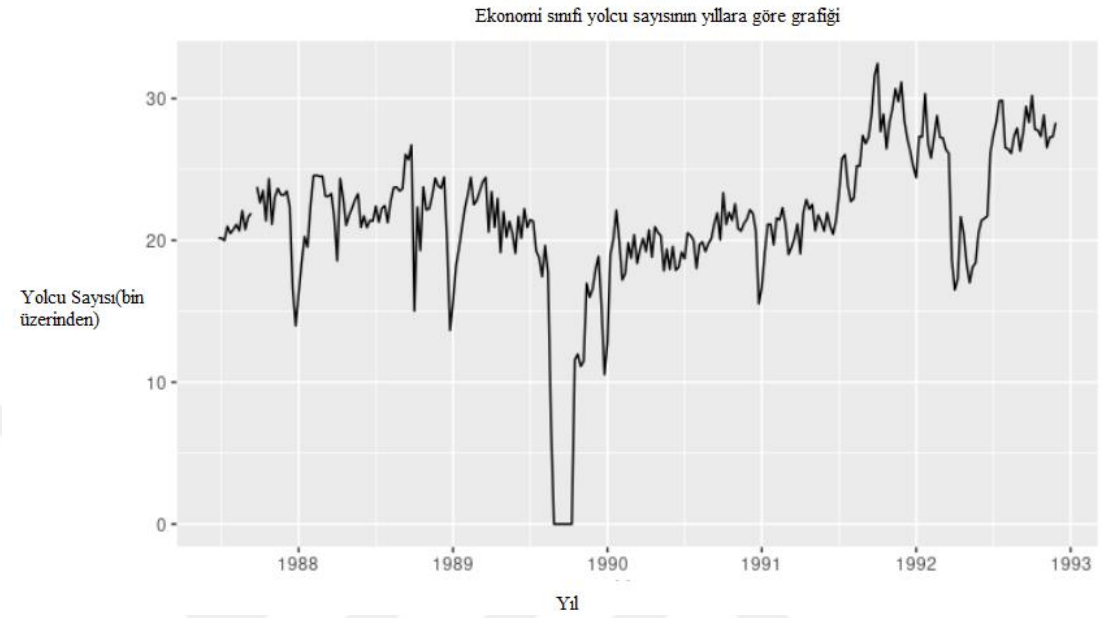
Zaman serileri matematiksel olarak x(t) gibi fonksiyonlar ile gösterilebilir. Burada t zamanı gösterirken, x(t) ise o zaman diliminde gerçekleşen olayın sonucunu göstermektedir. x(t)'nin genel karakteristiği hakkında bilgi edinmek için birçok teknik bulunmaktadır. Bu tekniklerin bazıları Hyndman ve Athanasopoulos'un kitabında [2] açıklanmıştır:

- Zamana göre grafikleri bastırma
- Mevsimlik grafikler
- Gecikme grafikleri (Lag plot)
- Otokorelasyon
- Zaman serilerinin eğilimleri, mevsimsel etkileri ve döngüsel davranışları (Trend, Seasonality, Cyclic)
- ARIMA ve ETS modelleri

Hyndman ve Athanasopoulos bu belirtilen yöntemler haricinde birçok yöntemi kitabında açıklamıştır. Bunlar genel olarak birçok zaman serisi üzerinde uygulanabilecek yöntemler olmakla birlikte bu çalışmada bazı ön işleme teknikleri kullanılmıştır.

Şekil 2.4'te örnek bir zaman serisi bulunmaktadır [2]. Bu zaman serisinde Avustralya'da 1988 ile 1993 arasında Ansett Hava Yolları'nın ekonomi sınıfında yolculuk yapan insan sayısının bir grafiği gösterilmiştir. Bu grafikte ilginç olan bir nokta 1989 yılının bir zaman diliminde hiç yolcu taşınmamıştır. Bu durumun endüstriyel bir anlaşmazlıktan kaynaklandığı bilinmektedir [2]. Bu grafikte ekonomi sınıfında yolculuk yapan insan sayısı bazı zaman dilimlerinde artarken bazı zaman dilimlerinde ise azalmaya başlamıştır. Buradan bazı sorunların olduğu ya da ekonomi

sınıfına olan talebin azaldığı gibi yorumlar yapılabilir. Bu bölümün devamında zaman serilerinin bazı alt bölümleri anlatılacak ve zaman serisi problemlerinin önemli bir türü olan finanstan bahsedilecektir.



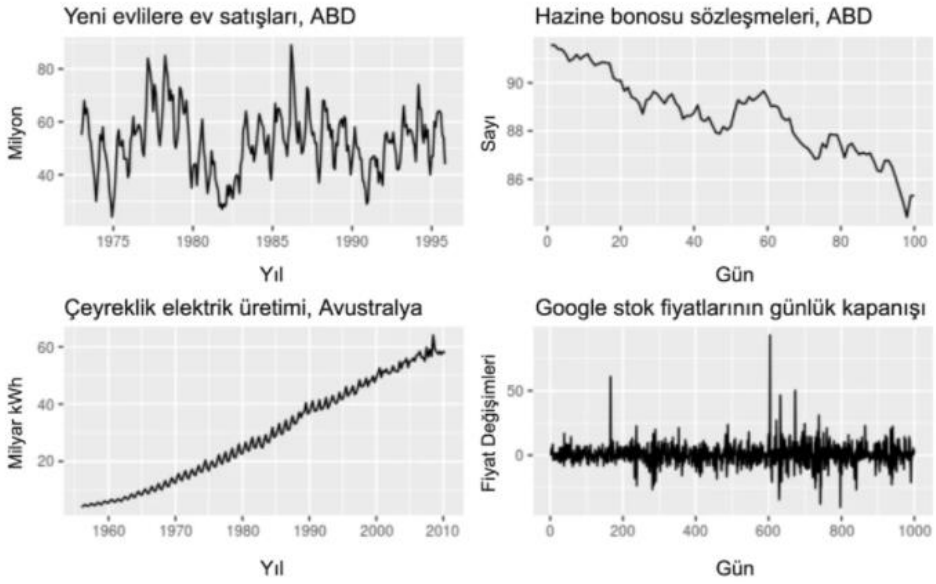
Şekil 2.4: Ekonomi sınıfı yolcu sayısının yıllara göre değişimi.

2.2.1 Zaman serileri üzerinde bazı örüntüler

Zaman serileri üzerinde genel olarak bazı örüntüler olabilir. Bu örüntüler çok genel örüntüler olduğundan dolayı zaman serisi hakkında önemli bilgiler sağlamaktadır. Bu bölümde bu zaman serilerinin genel özellikleri hakkında ve en çok gözlemlenen bazı örüntülerden bahsedilecektir.

2.2.1.1 Trend

Bir zaman serisinde uzun süreli bir artış ya da azalış varsa bu zaman serisinde bir eğilim (trend) gerçekleşmektedir. Bu eğilim doğrusal olmak zorunda değildir. Bu eğilimler bazı zaman dilimlerinde yön değiştirebilir. Örneğin artmakta olan bir zaman serisi azalma yönüne doğru geçebilir. Şekil 2.5'te trend olarak verilebilecek 2 adet zaman serisi bulunmaktadır [2]. Hazine bonosu sözleşmesinde ilerleyen zamanda bir azalma gözlemlenmektedir. Bu da azalan bir trend olduğunu göstermektedir. 3 aylık elektrik üretiminde ise yıllar içerisinde bir artış gözlemlenmiştir. Bu da artmakta olan bir trende örnek olarak verilebilir.



Şekil 2.5: Farklı desenleri gösteren 4 zaman serisi.

2.2.1.2 Mevsimsel

Bir zaman serisinin belirli zaman dilimlerinde bazı etkiler oluşuyorsa bu zaman serisinde mevsimsellik özelliği bulunmaktadır. Bu etkiler mevsimsel, aylık, günlük ve hatta saatlik, dakikalıkta olabilir. Mevsimsel etkiler her zaman bilinmektedir ve zaman serisi üzerinde sabit bir etki oluşturmaktadır. Yani zaman serisinin o periyodunda o etki hep oluşmaktadır. Şekil 2.5'te mevsimsellik özelliğine örnek olarak verilebilecek 2 adet zaman serisi bulunmaktadır. Yeni evlilere ev satışı ve 3 aylık elektrik üretimi grafiklerinde mevsimsel etkiler gözlemlenmektedir. Belirli dönemlerde zaman serileri artış ve azalış hareketini sürekli olarak gerçekleştirmektedir.

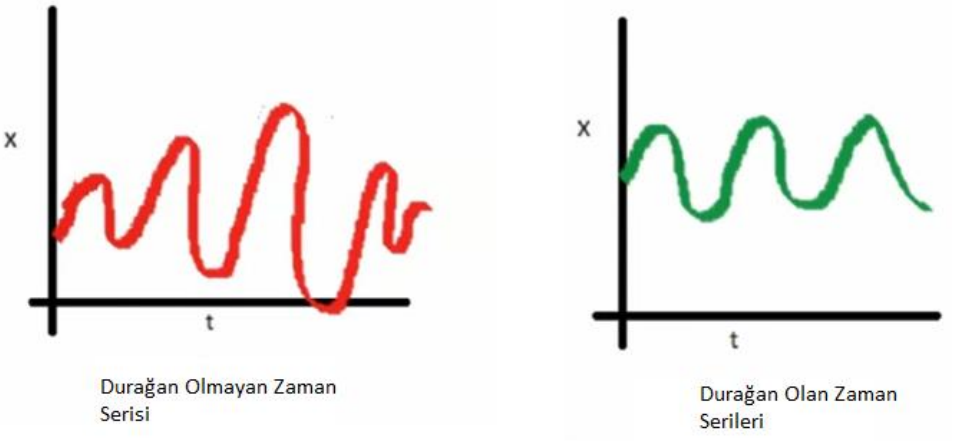
2.2.1.3 Döngüsel

Zaman serilerinde mevsimsel olmayan ve nedensel bir etkiden kaynaklanan dalgalanmalar gerçekleşebilir. Bu dalgalanmalar anlık olarak artış ya da azalış olarak görülmektedir. Bu dalgalanmalar genellikle ekonomik sebeplerden gerçekleşebilir ve iş döngüleriyle ilişkilidir. Şekil 2.5'te döngüsel yapıda bulunmayan zaman serisi 3 aylık elektrik üretimidir. Burada zaman serisinin döngüsel bir özelliğe sahip olduğunu gösteren bir kanıt bulunmamaktadır. Diğer grafiklerde bazı periyotlarda mevsimsel olmayan yükseliş ve alçalışlar bulunmaktadır. Dolayısıyla da bu zaman serileri döngüsel yapıdadır.

2.2.1.4 Durağan ve durağan olmayan zaman serileri

Bir zaman serisinin durağan (stationary) olması için gözlemlendiği zamana bağlı olmaması gerekmektedir. Bu yüzden dolayı trend ve mevsimsel olan zaman serileri durağan değildir. Bu seriler zamanla azalmakta, artmakta ya da mevsimsel zamanlara bağlılık göstermektedir.

Şekil 2.6'da durağan ve durağan olmayan zaman serilerine örnek olabilecek 2 adet grafik bulunmaktadır [3]. Bu grafiklerden kırmızı olan grafiğin zaman içerisinde varyansı ve ortalaması değişmiştir. Bu da zamana bağlı bir şekilde hareket ettiğini ve durağan bir sinyal olmadığını gösterir. Yeşil olan grafikte ise zaman ilerledikçe ortalama ve varyansın sabit kaldığı gözlemlenmiştir. Yeşil olan grafiğin durağan olan bir sinyale ait olduğu söylenebilir.



Şekil 2.6: Durağan ve durağan olmayan zaman serisinin örnek grafikleri.

2.2.1.5 Korelasyon

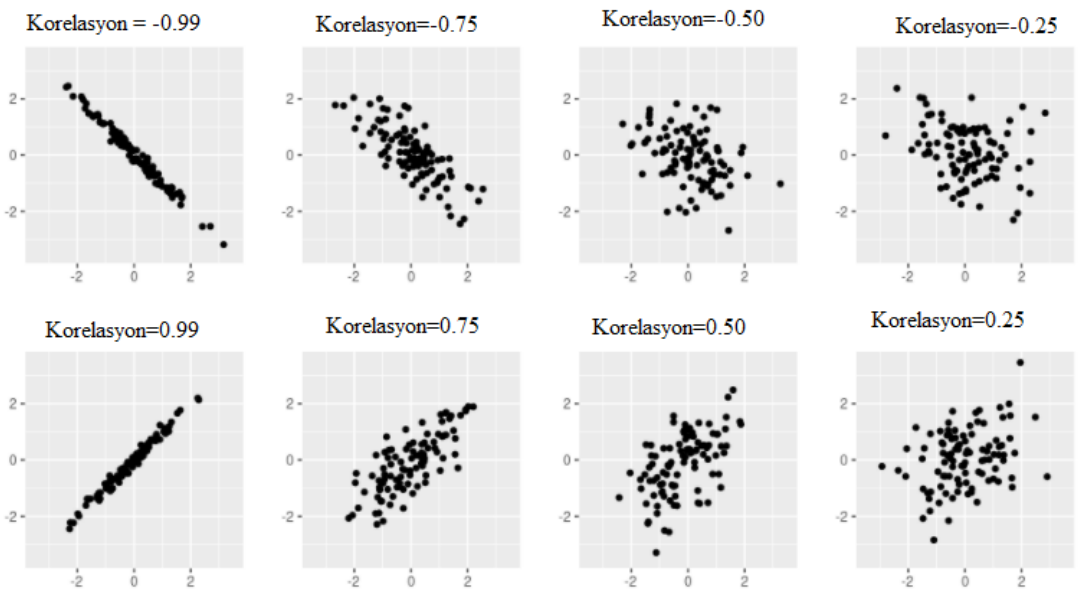
Korelasyon katsayısı iki rassal değişken arasındaki doğrusal ilişkiyi hesaplarken kullanılmaktadır.

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y} \tag{2.1}$$

Denklem (2.1)'de korelasyon formülünün hesaplanması gösterilmiştir. Burada μ_x = E(X) yani bir rassal değişkenin beklenen değerini göstermektedir. σ_x ise rassal değişkenin standart sapmasını göstermektedir. Korelasyon katsayısı -1 ile 1 arasında değer almaktadır. Tam artan bir doğrusal ilişki halinde korelasyon 1, tam azalan yönlü doğrusal bir ilişki olması durumunda korelasyon -1 değerini göstermektedir.

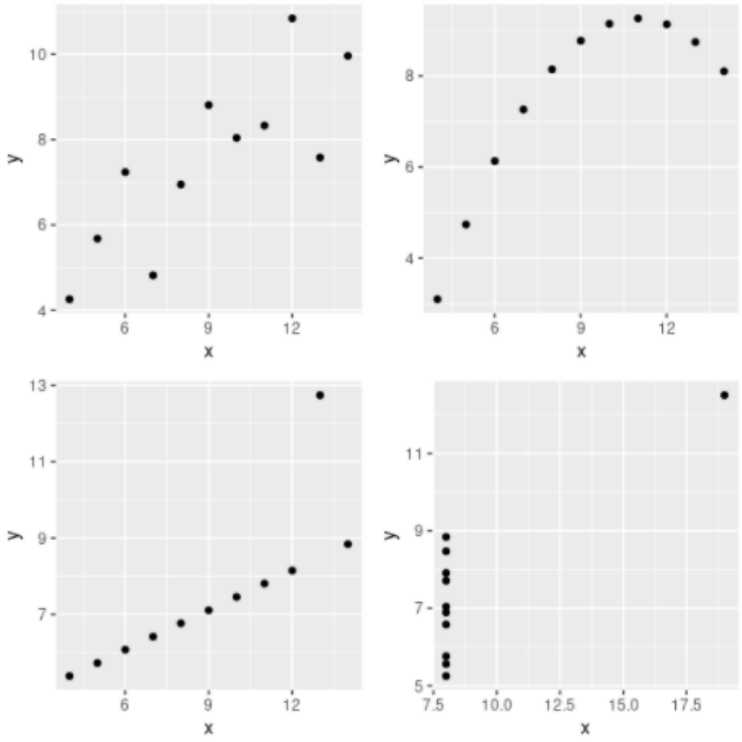
Korelasyon katsayısı -1 ve 1'e ne kadar yakınsa ilişkinin doğrusallığı o kadar güçlü olacaktır.

Şekil 2.7'de örnek veri setleri üzerinde korelasyon dağılımları gösterilmiştir. Burada her veri setinde farklı seviyede bir korelasyon olduğu gözlemlenmiştir [2]. Bu şekilde yüksek korelasyon katsayısına sahip olan verilerin üzerindeki değişkenlerin ilişkileri çok rahat bir şekilde gözlemlenmektedir. Aynı zamanda korelasyon değerinin sıfıra yakın olduğu veriler için dağılımların rastgele bir şekilde gerçekleştiği gözlemlenmektedir. Yani doğrusal bir ilişki gözlemlenmiştir.



Şekil 2.7: Örnek veri setleri üzerinde korelasyon dağılım grafikleri.

Şekil 2.8'de 0,82 korelasyon katsayısına sahip birden çok veri seti gösterilmektedir. Bu veri setleri aynı korelasyon değerlerine sahip olmalarına rağmen çok farklı ilişkilere sahiptirler [2]. Korelasyon katsayısı bir veri üzerinde bu şekilde yanıltıcı etkilere sahip olabilir. Aynı zamanda korelasyon katsayısı yalnız doğrusal olan ilişkileri ölçmektedir. Bazen verilerde doğrusal olmayan ilişkiler olmasına rağmen bu katsayı düşük değerler verebilir. Bu yüzden korelasyon katsayısı her zaman veri üzerindeki ilişkileri yansıtmayabilir.



Şekil 2.8: 0,82 korelasyon katsayı değerlerine sahip olan farklı veri setleri.

2.2.2 Finans ve borsa

Borsa hisse senedi, menkul kıymetler, döviz, vadeli işlemler, opsiyonlu sözleşmelerin işlem gördüğü halka açık sermayenin güvenle yapıldığı piyasalara denir. Bu piyasaların halka açık olan kısımlarında alım ya da satım yapılarak bu paylara ortak olunabilir. Bu çalışmada borsalar piyasada şirketlerin hisse senedini matematiksel olarak bir zaman serisi olacak şekilde temsil etmektedir. Bu zaman serileri çeşitli analizlere tabi tutulabilir. Dünya üzerinde en etkili olan bazı borsalar sırasıyla şunlardır:

- FTSE-100
- Dow Jones
- NASDAQ
- CAC 40
- DAX
- Nikkei 225
- BBC Global 30

2.3 Ön-işleme

Makine öğrenmesi ve derin öğrenme modelleri eğitirken ön işleme teknikleri gerekebilir. Verilere eğer herhangi bir ön işleme tekniği uygulanmazsa bazı durumlarda modellerin performansları ciddi bir şekilde etkilenebilir. Veri setindeki özniteliklerin istatiksel olarak birbirinden bağımsız olmaları ya da iyi bir skalaya yayılmaları modellerin öğrenim sürecini kolaylaştırmaktadır. Bunun için yapılabilecek bazı işlemler ilerleyen bölümlerde açıklanmıştır.

2.3.1 Öznitelik ölçeklendirme

Öznitelik ölçeklendirme literatürde en çok kullanılan ve en basit yöntemlerden bir tanesidir. Bu yöntem sayesinde özniteliklerin değerleri 0-1 arasına inmektedir.

$$y = \frac{x - min(x)}{max(x) - min(x)} \tag{2.2}$$

Denklem (2.2)'de öznitelik ölçeklendirmenin formülü verilmiştir. Burada bir rastgele değişkenden rastgele değişkenin en küçük değeri çıkarılarak elde edilen yeni değer, en büyük ve en küçük değerin farkına bölünerek 0-1 arasına çekilmiş olan yeni öznitelik değeri hesaplanmıştır.

2.3.2 Z-skoru

Z-skoru, öznitelik uzayının her boyutunu μ=0 ve σ = 1 olacak şekilde değiştirir. Böylelikle çok geniş bir uzayda dağılan verilerin yerini sıkıştırılmış bir veri grubu alır. Öznitelik ölçeklendirme yönteminden önemli bir farkı özniteliklerin değeri belirli bir aralıkta sınırlı kalmamış olmaktadır. Denklem (2.3) Z-skorun hesaplanmasını göstermektedir.

$$\frac{X - \mu}{\sigma} \tag{2.3}$$

2.4 Makine Öğrenmesi

Günümüzde yapay zekanın önemli alt türlerinden birisi olan makine öğrenmesi (Machine Learning, ML) birçok problemin çözümünde kullanılmaktadır. ML bilgisayarların klasik çözüm metotlarıyla ya da insan gözlemiyle çözülemeyecek olan

problemleri matematiksel olarak öğrenebilen bazı modeller kurarak çözmesine denir. ML gerçek hayatta problemi çözmeye çalışırken bazı öznitelikler çıkararak bir çözüm yöntemi ortaya koyar. Bu öznitelikler problemi tasarlayan kişiler tarafından seçilebileceği gibi ML bunların bazılarını kendisi de seçebilir. Bu öznitelikler ile yapılmak istenilen görevi öğrenebilen bir model geliştirmek ML literatürünün genel amacı olarak gösterilebilir. Bu öğrenme sürecini tanımlayan birçok farklı yöntem bulunmaktadır.

Makine öğrenmesi hakkında detaylı bilgiler ve öğrenme yöntemleri, optimizasyon teknikleri ilerleyen bölümlerde açıklanmaktadır.

2.4.1 Öğrenme yöntemleri

Makine öğrenmesinde birden çok öğrenme yöntemi bulunmaktadır. Bu öğrenme yöntemlerinden bazıları gözetimli, yarı gözetimli ve gözetmensiz öğrenmedir. Bu öğrenme çeşitlerinin detayları alt bölümlerde açıklanacaktır.

2.4.1.1 Gözetimli öğrenme

Gözetimli öğrenme yönteminde (Supervised Learning) bir veri girdi değerlerine ve çıktı değerlerine sahip olmaktadır. Bu girdi ve karşılık gelen çıktı değerleri analiz edilerek bir f(x) = y oluşturulmaktadır. Burada f(x) = y fonksiyonu gerçek veriyi en iyi yansıtacak şekilde tasarlanmaktadır. Belirli bir miktarda hata ile birlikte bu fonksiyon oluşturulabilir. Gözetimli öğrenmenin temel iki yöntemi sırasıyla sınıflandırma ve regresyon yöntemleridir.

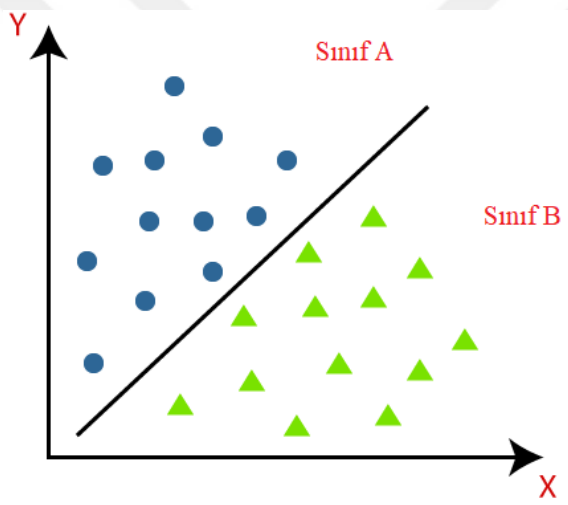
Sınıflandırma probleminde verilen özniteliklerin ya da girdilerin hangi sınıfa ait olduğu tespit edilmeye çalışılmaktadır. Bu problemi çözmek için ML tanım kümesi $f : R^N \rightarrow \{1,\ldots,k\}$ olan bir fonksiyon oluşturmaktadır. Böylelikle fonksiyonun çıktısı k adet sınıftan bir tanesi ile eşleştirilmektedir. Bu fonksiyonun farklı türdeki versiyonları da bulunabilir. Örneğin k adet sınıftan hangi sınıfa ait olduğunun olasılığını veren bir fonksiyon da oluşturulabilir. Sınıflandırma problemine verilebilecek bazı örnekler şu şekildedir:

- Oluşturulan bir görüntü veri seti üzerinde verinin hangi canlı türüne ait olduğunu tahmin etme
- Bir hisse senedinin gelecekte değer kazanıp kazanmayacağını tahmin etme

- Bir video üzerinde anomali olup olmadığı veya hangi türden anomaliler oluştuğunu tahmin etme
- Bir futbol maçını hangi takımın kazanacağını tahmin etme

Şekil 2.9 üzerinde bir sınıflandırma problem örneği gösterilmiştir [4]. Burada özniteliklerin 2 boyutlu uzaydaki dağılımı gösterilmiştir. Sınıflandırma problemlerinde genellikle ayrıştırıcı yüzeyler verilerin özniteliklerini birbirinden ayırarak iki sınıf arasında sınıflandırma yapar. Sınıflandırma çeşitleri aşağıdaki gibidir:

- İkili sınıflandırma (Binary classification)
- Çok sınıflı sınıflandırma (Multi-Class classification)



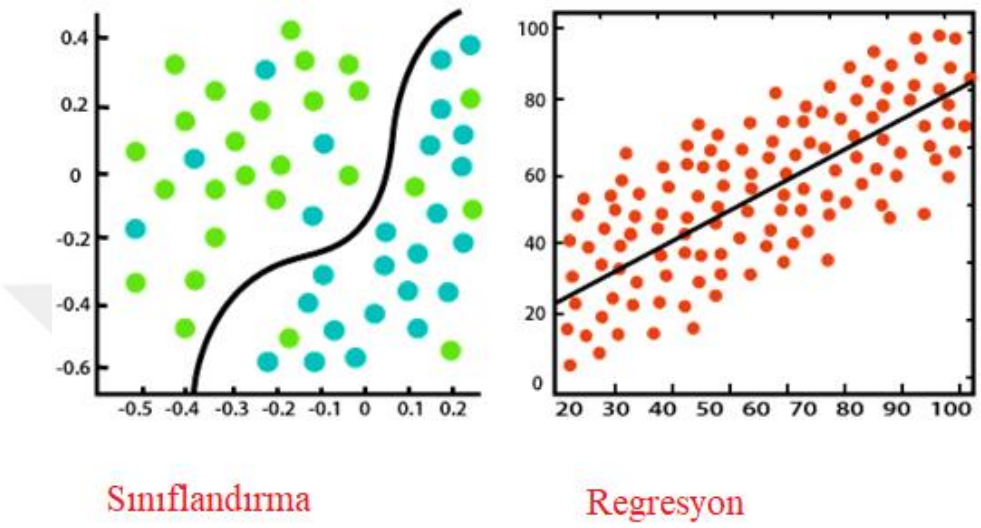
Şekil 2.9: Sınıflandırma probleminin 2 boyutlu düzlem üzerinde örneği.

Regresyon probleminde bir bilgisayar programı verilen öznitelikler için sayısal değerler tahmin etmeye çalışmaktadır. Bu problemi çözmek için ML tanım kümesi f : $R^N \rightarrow R$ olan bir fonksiyon oluşturmaktadır. Regresyon problemi sınıflandırma problemine çok benzerdir fakat çıktı kümesi tüm reel sayı kümesini kapsayabilir. Bir ürünün gelecekteki fiyat tahminini modelleme regresyon problemi olarak düşünülebilir. Bir ürünün fiyatının gelecekte artıp azalacağının tahmini ise sınıflandırma problemi olarak düşünülebilir. Regresyon problemine verilebilecek bazı örnekler şunlardır:

- Bir hisse senedinin gelecekteki değeri
- Bir bölgedeki hava sıcaklığı
- Bir ülkede tüketilecek olan su miktarının tahmini

- Bir bölgedeki insan sayısının zamanla değişimi

Şekil 2.10 üzerinde regresyon ve sınıflandırma problemlerinin farkları gösterilmiştir [5]. Regresyon problemi veri kümesi üzerinde çıktısı reel sayılar olacak şekilde bir fonksiyon oluştururken sınıflandırma problemi ise bir ayrıştırıcı yüzey oluşturarak verilerin sınıfını ayrı ayrı bulmaya çalışmaktadır.



Şekil 2.10: Regresyon ve sınıflandırma probleminin şekil üzerinde örneği.

2.4.1.2 Gözetimsiz öğrenme

Bu öğrenme yönteminde etiketlenmemiş veri ile modelleme yapılmaktadır. Bu öğrenme yöntemi eğer uzmanlar verinin nasıl etiketleneceği hakkında bilgi sahibi değilse ciddi bir avantaj sağlamaktadır. Kümeleme yöntemi en çok kullanılan gözetimsiz öğrenme yöntemidir. Verilerin uzayda belirli bir yerel bölgede toplanarak oluşturduğu farklı öbekleri gösteren bu yöntem en temel yöntemlerden bir tanesidir. Bu tezin kapsamında gözetimsiz öğrenme yöntem kullanılmadığından dolayı detaya girilmeyecektir.

2.4.2 Hata fonksiyonları

Gözetimli öğrenme yönteminde oluşan fonksiyonun başarısını ölçmek için bazı hata fonksiyonları kullanılmaktadır. Bu fonksiyonların her biri farklı bir şekilde değerlendirme yapmaktadır ve problemden probleme modelleme yapılırken hangisinin kullanılacağı değişebilir. Bu fonksiyonlardan en yaygın olanları ilerleyen bölümlerde açıklanmaktadır.

2.4.2.1 Ortalama kare hata(MSE)

Ortalama kare hata (Mean square error) makine öğrenme ve derin öğrenme alanında en çok kullanılan hata fonksiyonlarından biridir. Ortalama kare hatanın formülü Denklem (2.4)'te verilmiştir [6]. Burada hata fonksiyonu hesaplanırken tahmin edilen değer ile gerçek değerin arasındaki farkın karesi alınmaktadır. Bu şekilde her tahmin için bir ceza katsayısı üretilmektedir. Bu değerler toplandıktan sonra elde edilen değerlerin ortalaması alınarak ortalama kare hata hesaplanmaktadır.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} \tag{2.4}$$

MSE'nin önemli özelliklerinden bir tanesi sapma miktarı yüksek olan hataları daha çok cezalandırmasıdır. Bu yüzden dolayı gürültülü verileri daha rahat öğrenebilir. Eğer model gürültülü verileri iyi tahmin edemiyorsa o veriler için daha çok ceza katsayısı atanacağından dolayı yüksek bir MSE değeri hesaplanacaktır. MSE türevi kolay alınabilen bir fonksiyon olduğundan dolayı gradyanların hesaplanmasında kolaylık sağlamaktadır.

2.4.2.2 Ortalama mutlak hata(MAE)

Ortalama mutlak hata (Mean Absolute Error, MAE) makine öğrenmesinde kullanılan popüler hata fonksiyonlarından bir tanesidir. Mutlak değer matematiksel olarak uzaklık kavramını temsil etmektedir. Bundan dolayı da tahminde gerçekleşen hata yönsüz olarak doğrudan uzaklık kadar etkilenmektedir. Veri içerisinde gürültü olan veriler ile normal veriler birbirleriyle eş değer tutulmaktadır. Denklem (2.5)'te MAE fonksiyonunun hesaplanması gösterilmektedir [6].

$$\text{MAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{2.5}$$

2.4.2.3 Çapraz entropi(CE)

Çapraz entropi (Cross entropy, CE) hata fonksiyonu sınıflandırma problemlerinde kullanılmaktadır. Bir verinin hangi sınıfa ait olduğunun olasılıksal değerini hesaba katarak bir hata fonksiyonu oluşturur ve bir değerlendirme yapar. İkili sınıflandırma

için çapraz entropi fonksiyonu denklem (2.6)'da verilmiştir [7]. Burada ML modeli sınıfı 1 olan veriye olasılıksal olarak 1 değerini verirse bu hata fonksiyonu sıfır olmaktadır. Aynı şekilde sınıfı 0 olan veriye 0 olasılık değerini atıyorsa hata fonksiyonu yine sıfır olacaktır. Eğer atanan olasılık değeri gerçek y değerinden sapmaya başlarsa hata fonksiyonu daha yüksek değerler üretmeye başlayacaktır. ML algoritmaları sınıflandırma problemlerinde CE fonksiyonunu kullanmaktadır.

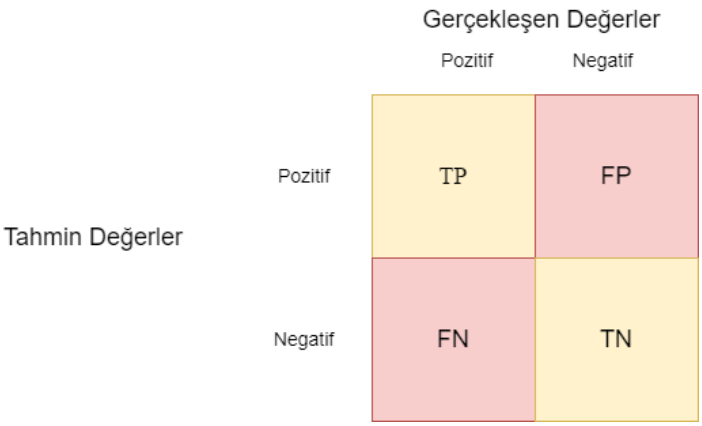
$$L(\theta) = -\sum_{i=1}^{k} y_i \, log(\hat{y}_i) \tag{2.6}$$

2.4.3 Metrikler

Makine öğrenme ve derin öğrenmede hata fonksiyonları kullanarak modeller eğitildikten sonra sınıflandırma problemlerinde ek olarak bazı metrikler kullanılmaktadır. Bunlar modellerin gerçekten ne kadar başarılı sonuçlar aldığını göstermek için gerekli olabilir. Bu metriklerin en yaygın olanları alt bölümlerde açıklanacaktır.

2.4.3.1 Karmaşıklık matrisi

Makine öğrenme ve derin öğrenmede sınıflandırma problemlerinin değerlendirilmesi esnasında kullanılan metriklerden bir tanesi karmaşıklık matrisidir (Confusion matrix). Bu matris tahmin edilen sınıflar ve gerçek değerler arasındaki ölçümleri ortaya koyarak ne kadar iyi bir tahmin yapıldığını göstermektedir. Şekil 2.11'de görüldüğü gibi karmaşıklık matrisinde 4 adet değer bulunmaktadır. Bunlar TP (Gerçek pozitif), FP (Yanlış pozitif), FN (Yanlış negatif) ve TN (Gerçek negatif)'dir.



Şekil 2.11: Karmaşıklık matrisi.

- TP: Gerçekte pozitif olan ve modelin pozitif olarak tahmin ettiği veriler
- FP: Gerçekte negatif olan ve modelin pozitif olarak tahmin ettiği veriler
- TN: Gerçekte negatif olan ve modelin negatif olarak tahmin ettiği veriler
- FN: Gerçekte pozitif olan ve modelin negatif olarak tahmin ettiği veriler

Bu değerlerden TP ve TN değerlerinin olabildiğince yüksek olması modelin performansının yüksek olduğunu göstermektedir. Modelin yaptığı hatayı analiz etmek için FP ve FN değerlerine bakılabilir.

2.4.3.2 Doğruluk

Sınıflandırma problemlerinde önemli metriklerden bir tanesi doğruluk (accuracy) metriğidir. Doğru sınıflandırılan veri sayısının toplam veri sayısına bölünmesi sonucunda doğruluk değeri hesaplamaktadır. Doğruluk değerinin hesaplanması denklem (2.7)'de gösterilmiştir. Doğruluk metriği sınıf sayısının dengeli dağıldığı durumlarda modelin performansını güzel yansıtan bir metrik türüdür. Örnek olarak, bir hayvan türü sınıflandırmaya çalışan model için %55 kedi %45 kedi olmayan nesneli veri verildiğinde eğer modelin doğruluk skoru yüksekse model dengeli bir tahmin yapmaya çalışmaktadır. Aksi takdirde modelin performansı çok ciddi bir miktarda düşecektir. Doğruluk metriği dengeli olmayan sınıflarda iyi bir değerlendirme metriği değildir. Örneğin %90 kedi ve %10 kedi olmayan nesneli bir veride model eğer her nesneye kedi şeklinde cevap vermeyi öğrenmiş ise %90 doğruluk skoru elde edecektir.

$$Doğruluk = \frac{TP + FP}{TP + FP + FN + TN} \tag{2.7}$$

2.4.3.3 Kesinlik

Sınıflandırma problemlerinde modelin performansını ölçmek için kullanılan bir diğer metrik kesinlik (precision) skorudur. Kesinlik modelin pozitif olarak çıktı verdiği değerlerden kaç tanesinin gerçekten pozitif olduğunu ölçer. Denklem (2.8)'de kesinlik skorunun nasıl hesaplandığı gösterilmiştir. Kesinlik skoru [0, 1] arasında bir değer üretmektedir ve kesinlik değeri 1'e yaklaştıkça modelin başarısı artmaktadır.

$$Kesinlik = \frac{TP}{TP + FP} \tag{2.8}$$

Bir örnek üzerinden açıklanacak olursa 100 veriden 10 tanesi kedi sınıfından kalan 90 veri ise farklı objelerden oluşmaktadır. Eğer model her veriye kedi şeklinde yanıt üretiyorsa kesinlik skoru 5/100=0,05 değerini üretecektir. Bu da modelin başarısının kötü olduğunu göstermektedir.

2.4.3.4 Hassasiyet

Sınıflandırma problemlerinde model performansını ölçen bir başka metrik hassasiyet (recall) skorudur. Hassasiyet gerçekte pozitif olan verilerden kaç tanesinin model tarafından pozitif olarak cevaplandığını gösteren metriktir. Denklem (2.9)'da hassasiyet metriğinin hesaplanma formülü gösterilmiştir. Hassasiyet skoru da [0, 1] aralığında değer almaktadır ve skorun değeri 1'e yaklaştıkça modelin performansının daha yüksek olduğunu göstermektedir.

$$Hassasiyet = \frac{TP}{TP + FN} \tag{2.9}$$

Bir örnek vererek açıklanacak olursa 10 adet kedi ve 90 adet kedi olmayan bir veri üzerinde model TP = 2, FP = 8, FN = 8, TN = 82 olacak şekilde bir tahmin yapıyorsa burada hassasiyet skoru 2/10 = 0,2 olacaktır. Modelin kesinlik skoru 84/100 = 0,84 olmasına rağmen model pozitif sınıflardan yalnızca %20'sini tahmin edebilmektedir. Bu da modelin performansının kötü olduğunu göstermektedir.

2.4.4 Model eğitimi ve değerlendirme

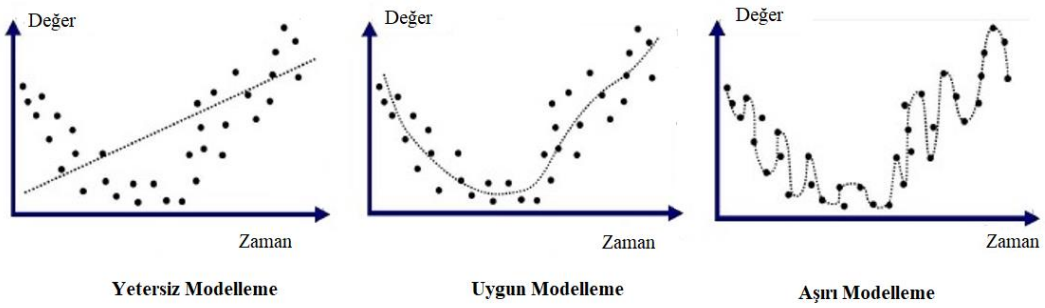
Makine öğrenmesi problemlerinde modellerin seçimi, optimizasyonu, parametre ayarlama gibi bazı önemli noktalar bulunmaktadır. Makine öğrenmesinde hangi modellerin seçileceği, bu modellerin hangi özelliklere sahip olacağı, hangi yöntemle modellerin eğitileceği gibi önemli problemler bulunmaktadır. Bu problemler modellerin alacağı performansları ciddi bir şekilde değiştirecektir. Bu temel kavramlar bu bölümün alt başlıklarında açıklanacaktır.

2.4.4.1 Model genelleştirme

Makine öğrenmesi problemlerinde veriler çok boyutlu uzayda birer nokta olarak ifade edilmektedir. Burada modeller uzayda karar sınır hiper düzlemleri bulmaya çalışmaktadır. Modeller bu hiper düzlemlere bakarak verilerin hangi sınıflara ait olduğuna karar vermektedir. Eğer problem regresyon problemi ise bu hiper düzlemler çeşitli şekillerde kullanılarak bir gerçel sayı değeri üretilir ve model tahminini gerçekleştirir. Modeller bu hiper düzlemleri öğrenirken modelin ne kadar karmaşık olduğu bu karar sınırlarının yapısını etkilemektedir.

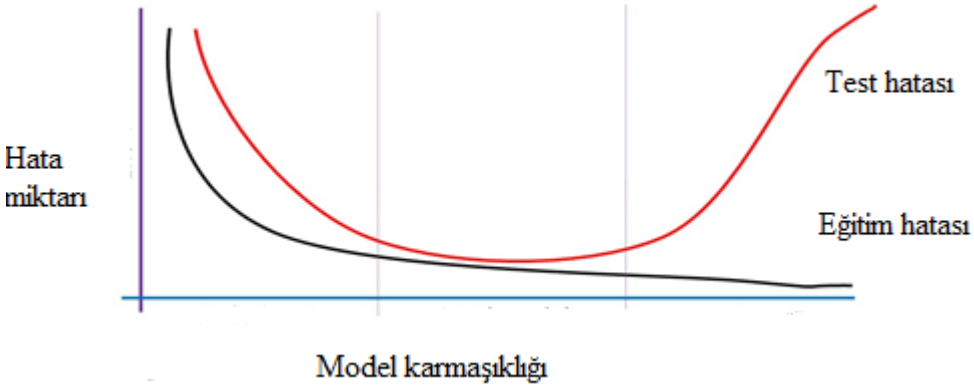
Modellerin ifade ettiği fonksiyonların ya da karar düzlemlerin karmaşıklık derecesi makine öğrenmede önemlidir. Bir model bir veriyi olması gerekenden fazla güçlü bir fonksiyonla ya da karar düzleminde ifade etmeye çalışıyorsa o veriyi ezberlemeye başlar ve bu soruna aşırı modelleme (overfitting) denir. Eğer bir model bir veriyi yeterince iyi modelleyemiyorsa ya da model problemi yeterince iyi öğrenemiyorsa bu duruma yetersiz modelleme (underfitting) denir.

Şekil 2.12'de bir veri üzerinde eğitilmiş bazı modellerin sonuç grafikleri gösterilmektedir [8]. Burada yetersiz modellenmiş, uygun modellenmiş ve aşırı modellenmiş 3 model bulunmaktadır. Yetersiz modelleme yapılan örnekte oluşan fonksiyon aşırı basit kalmış bundan dolayı bu probleme iyi bir tahmin üretememiştir. Aşırı modelleme yapılan örnek model gürültü olan verileri dahi modellemiş ve aşırı karmaşık bir fonksiyonla ifade etmiştir. Bu tarz bir durumda hata fonksiyonu sıfır olmasına rağmen gerçek hayata uygulandığında model performansı aşırı bir şekilde düşebilir. Uygun modelleme yapılan örnekte ise makine öğrenmesi bu iki durumun arasında bulunan bir modelleme yapmaktadır. Makine öğrenmesinde uygun modelleme yapmaya çalışmak önemli noktalardan bir tanesidir.



Şekil 2.12: Modellerin farklı şekilde eğitilme sonucunda elde edilen sonuçlar.

Şekil 2.13'de model karmaşıklığına karşı hata miktarının grafiği gösterilmiştir [9]. Bu grafikten de görüldüğü üzere model karmaşıklığı çok düşük iken eğitim verisi üzerindeki hata ve test verisi üzerindeki hata yüksek gelmektedir. Model karmaşıklığının çok yüksek olduğu yerlerde ise eğitim hatası çok düşük gelirken test hatası aşırı yüksek gelmektedir. Bu iki durum yetersiz modelleme ve aşırı modellemeden kaynaklanmaktadır. En uygun modelleme bu iki modelin karmaşıklık seviyesinin ortasında bir model ile veriyi eğitmektir.



Şekil 2.13: Model karmaşıklığına karşı hata miktarını gösteren grafik.

2.4.4.2 Erken durdurma

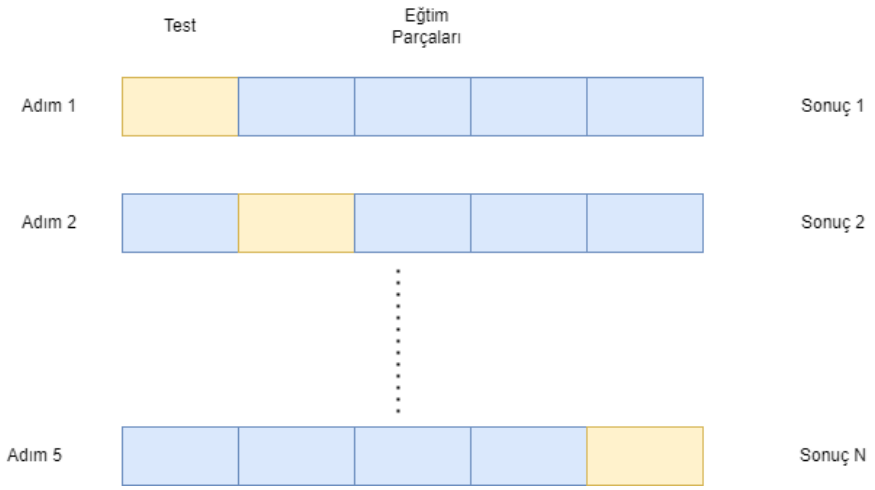
Makine öğrenmesi modeli eğitirken karşılaşılan yetersiz modelleme ya da aşırı modelleme durumunun önüne geçebilmek için kullanılabilecek yöntemlerden bir tanesi erken durdurma (early stopping) yöntemidir. Bu yöntem esnasında modelin bir tur hatayı azaltacak şekilde optimize edilmesi 1 epok (epoch) eğitilme şeklinde tanımlanmaktadır. Her epok sonrasında model eğitim hatasını biraz daha düşürmektedir. Burada önemli olan nokta modelin kaç epok eğitilecek olmasıdır. Erken durdurma metodu eğitim hatası ve test hatası azalana kadar modeli eğitir. Test

hatasının artmaya başladığı epokta modelin eğitiminin durdurulması gerekmektedir. Böylelikle aşırı modellemenin önüne geçilmiş olur. Burada eğitim verisinin ve test verisinin uzunluğunun seçiminde kesin bir yöntem bulunmamaktadır fakat genellikle eğitim verisinin uzunluğu test verisinin uzunluğunun belirli katları olacak şekilde seçilmektedir.

2.4.4.3 Çapraz geçerleme

Veriyi doğrudan iki parçaya ayırıp eğitim ve test şeklinde bölmek yüksek veri sayısında ve eşit dağılım gerçekleşen durumlarda mantıklı bir yöntem olmaktadır fakat veri sayısının az olduğu durumlarda bu yöntemi kullanmak mantıklı olmamaktadır. Bundan dolayı çapraz geçerleme yöntemi kullanılmaktadır. Çapraz geçerleme yöntemi veriyi k adet eş parçaya (fold) ayırdıktan sonra her bir parçayı test için kullanırken kalan büyük parçayı da eğitim için kullanmaktadır. Böylelikle parametre seçimi ve eğitim esnasında verinin farklı parçaları daha dengeli bir şekilde kullanılmış olur.

Şekil 2.14'te 5 fold çapraz geçerleme yönteminin örneği gösterilmiştir. Her adımda verinin bir parçası test için ayrılmıştır ve kalan parça eğitim için kullanılmıştır. 5 fold çapraz doğrulama sonucunda elde edilen her bir sonucun ortalaması alınarak yeni eğitim ve test skoru elde edilmektedir. Bu yöntemde k çok küçük seçilirse verinin eğitim ve test parçaları yeterince iyi bir şekilde bölünmeyeceğinden dolayı iyi bir eğitim veya değerlendirme gerçekleşmeyecektir. Bu yöntemde k en fazla veri sayısı kadar seçilebilir fakat bu durumda model eğitimleri çok uzun sürebilir. Bu yüzden dolayı k ideal olarak 5-10 arasında seçilebilir.



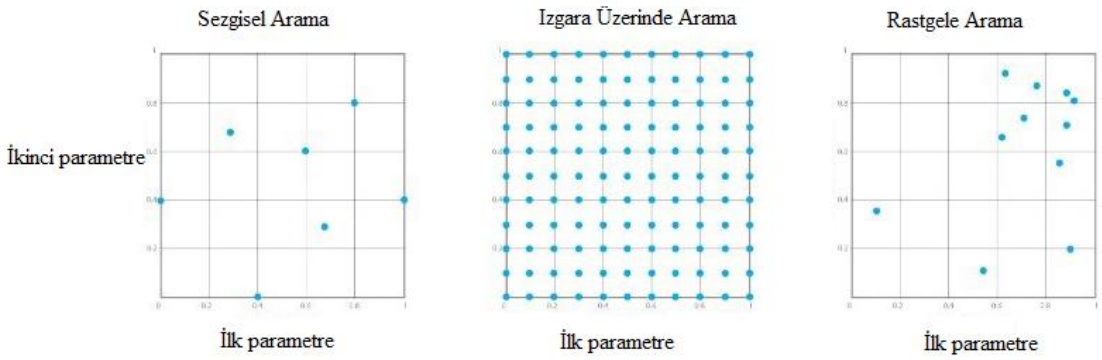
Şekil 2.14: 5-fold çapraz geçerleme ile modelin eğitilme örneği.

2.4.4.4 Hiper parametre seçim yöntemleri

Makine öğrenmesinde ve derin öğrenme yöntemlerinde bir modelin eğitimi esnasında çok sayıda farklı parametre bulunmaktadır. Bunlar genellikle modellerin karmaşıklığını belirleyen ve eğitim esnasında seçilen parametreler olmaktadır. Hiper parametre seçiminde genel amaç bu parametreleri uzayda aramaktır. Bu arama işlemi verinin eğitim ve değerlendirme (validation) parçalarının hata oranını en aza indiren parametreleri bulmaya çalışmaktır. Bazı hiper parametre seçim yöntemleri şunlardır:

- Sezgisel yöntemlerle hiper parametre seçimi
- Izgara üzerinde arama yöntemiyle parametre seçimi (Grid search)
- Rastgele arama yöntemiyle parametre seçimi (Random search)

Şekil 2.15 hiper parametre arama yöntemlerinin 2 boyutlu uzayda nasıl arama yaptıklarını göstermektedir [10]. Burada sezgisel arama yönteminde önceden elde edilen bilgiler sayesinde tahmini olarak bazı parametreler belirleyerek modeller eğitilmektedir. Bu yöntemde ön bilgi aşırı derecede önem taşımaktadır aksi takdirde modelin performansı aşırı derecede kötüleşebilir.



Şekil 2.15: Hiper parametre arama yöntemlerinin 2 boyutlu uzayda gösterimleri.

Izgara üzerinde arama yönteminde ise hiper parametrelerin aranacağı aralıklar belirlenmektedir. Bu aralıklar belirlendikten sonra bu hiper parametrelerin olası tüm kombinasyonları denenerek eğitim ve değerlendirme verisine bakılarak en iyi parametre seçimi yapılır. Bu arama yöntemi olası en iyiyi bulabilme şansına sahip bir yöntem olmasına rağmen seçilen parametreler tekrardan gözden geçirilmelidir.

Rastgele arama yönteminde olası tüm kombinasyonlara bakmak yerine bazı alt kümeler rastgele seçilerek uzayda bir arama yapılır. Bu algoritma ızgara üzerinde

arama yöntemine göre çok hızlı bir yöntem olmakla birlikte her zaman başarılı sonuç vermeyen bir yöntemdir.

2.4.5 Optimizasyon algoritmaları

Makine öğrenmesi problemlerinde verilen girdi X için y değerleri ile eşleşen bir f(X) fonksiyonu bulmak amaçlanmaktadır. Bu fonksiyonlar bulunurken optimal ağırlık değerlerinin hesaplanması gerekmektedir. Bu optimal ağırlık değerleri f(X) fonksiyonun çıktısını gerçek değerlere en çok yaklaştıran değerlerdir yani hatayı minimum değere düşüren ağırlıklardır. Teoride türev alarak f(X) fonksiyonunun ağırlık değerleri hesaplanmaktadır fakat pratikte f(X) fonksiyonu çok karmaşık bir fonksiyon olacağından dolayı türev alarak ağırlık değerlerini hesaplamak imkansız hale gelmektedir. Bundan dolayı iteratif bazı yöntemler ile hatanın minimum olduğu yer ağırlık uzayında aranmaktadır.

2.4.5.1 Gradyan iniş

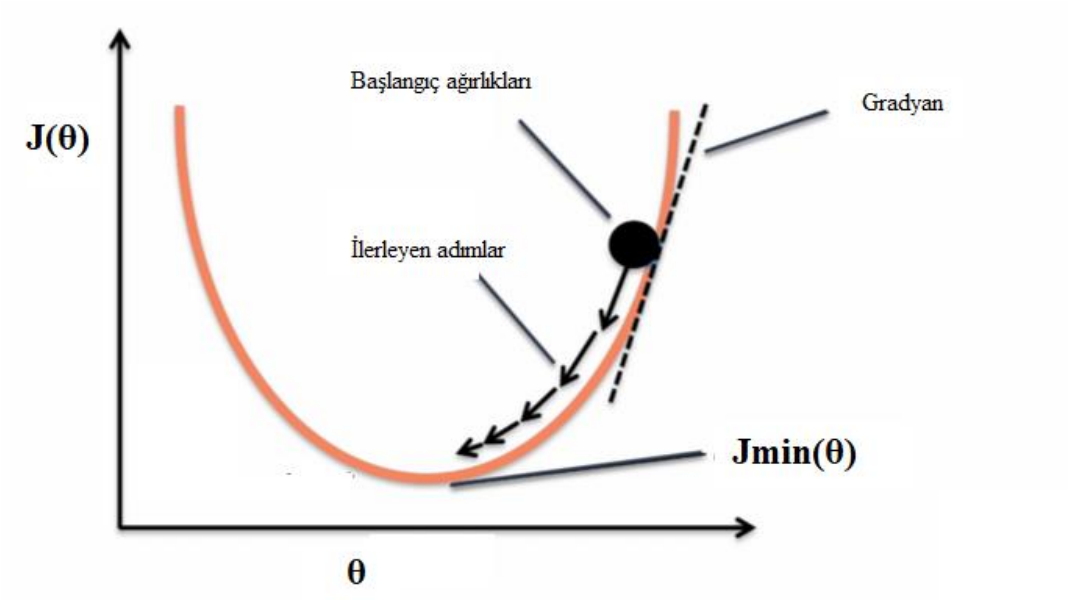
Gradyan iniş (gradient descent) algoritması hatayı iteratif olarak minimum yapmaya çalışan bir algoritmadır. Gradyan iniş algoritmasının optimize etmeye çalıştığı fonksiyonun hem türevi alınabilen bir fonksiyon hem de konveks bir fonksiyon olması gerekmektedir. J(θ) belirlenen ağırlıklar için oluşturulan hata fonksiyonunu temsil etmektedir. Burada J(θ)'nın ağırlıklara göre türevleri gradyan vektörünü oluşturmaktadır.

Şekil 2.16'da gradyan vektörünün temsili gösterimi yapılmıştır [11]. Burada her bir ağırlık değeri θ_i değeri için gradyan değeri hesaplanmaktadır ve bu gradyanların hepsi bir vektör üzerinde birleştirilmektedir. Gradyan vektörü hatanın en çok değişim gösterdiği noktaları vermektedir. Gradyan iniş algoritması J(θ) hata fonksiyonunu minimize edecek şekilde gradyanların tersi yönünde bir hareket yapmaktadır. Bu hareket temelde bir bilyenin kaseye atıldığında yaptığı salınım hareketine benzer şekilde gerçekleşmektedir. Bilye en sonunda kasenin tabanına ulaşacağı gibi hata gradyanlarda ağırlık değerlerini hata fonksiyonun minimum olduğu noktaya doğru iteklemektedir. Gradyan iniş algoritmasında değişim miktarını belirleyen parametre öğrenme oranıdır (learning rate).

$$\nabla_\theta J = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \\ \vdots \\ \frac{\partial J}{\partial \theta_n} \end{bmatrix}$$

Şekil 2.16: Gradyan vektörü.

Şekil 2.17'de gradyan iniş algoritmasının grafik üzerinde gösterimi verilmiştir [12]. Burada ML modeline rastgele ağırlıklar atanarak model eğitilmeye başlatılır. Her adımda J(θ) fonksiyonunun gradyanları hesaplanır ve gradyanların ters yönünde bir ağırlık güncellemesi yapılarak yeni ağırlıklar hesaplanır. Bu ağırlıklar belirli bir öğrenme oranı ile güncellenmektedir. Eğer öğrenme oranı çok düşük olursa eğitim süresi aşırı uzun olacaktır. Eğer öğrenme oranı çok yüksek seçilirse de minimum noktayı kaçırma ihtimali olacak ve salınım hareketi yapılmaya başlanacaktır. Bundan dolayı ilk başta yüksek bir öğrenme oranı ile başlayıp ilerleyen adımlarda bu oranın düşürülmesi gerekmektedir.



Şekil 2.17: Gradyan iniş algoritmasını grafik üzerinde gösterimi.

Yığın gradyan iniş (Batch Gradient Descent) metodu eğitim esnasında tüm verinin gradyanlarını tek bir adımda hesapladıktan sonra ağırlıkları güncellemektedir. Denklem (2.10) ağırlıkların yığın gradyan iniş yöntemiyle hesaplanmasını göstermektedir. Bu yöntemde ağırlıklar güncellenirken tüm veri kullanıldığından dolayı gradyanlar istatiksel olarak daha az sapma ile hesaplanmaktadır ve bundan dolayı hata fonksiyonu eğitim esnasında düşük varyans ile değiştiğinden dolayı daha yumuşak bir şekilde minimum noktayı bulacaktır. Bu yöntemde veri sayısı çok yüksek olacağından dolayı bilgisayarın hafızasının yetmemesi gibi bir sorun oluşacaktır.

$$\theta = \theta - \alpha \nabla_{\theta} J(\theta) \tag{2.10}$$

Stokastik gradyan iniş (stochastic gradient descent) metodu eğitim esnasında verileri birer birer kullanarak ağırlıkları güncellemektedir. Denklem (2.11) ağırlıkların stokastik gradyan iniş yöntemiyle ile hesaplanmasını göstermektedir. Stokastik gradyan iniş metodu her adımda bir verinin gradyanını alarak ağırlıkları güncellediğinden dolayı hata fonksiyonunu azalışı yüksek varyans ile gerçekleşecektir. Eğitim esnasında öğrenme oranı düşük tutulursa bu yöntem de yığın gradyan iniş yöntemi gibi yerel optimal noktaları bulabilmektedir. Stokastik gradyan iniş yönteminde ağırlıklar hesaplanırken verilerin birer birer kullanılmasından dolayı hafıza problemi yaşanmamaktadır. Bu yöntemin bir dezavantajı her bir veri için ayrı ayrı ağırlıklar güncellendiğinden dolayı daha fazla işlem yapılmaktadır ve bundan dolayı algoritmanın çalışma süresi ciddi miktarda artacaktır.

$$\theta = \theta - \alpha \nabla_{\theta} J(\theta, x^{(i)}; y^{(i)}) \tag{2.11}$$

Mini-yığın gradyan iniş(mini-batch gradient descent) literatürde en yaygın kullanılan gradyan iniş yöntemidir. Mini-yığın gradyan iniş verileri belirli sayıda parçalara ayırarak eğitme işlemi yapmaktadır. Böylelikle önceki 2 yöntemin çözemediği hafıza ve işlem gücü sorununu çözmektedir. Denklem (2.12) ağırlıkların mini-yığın gradyan iniş yöntemi ile hesaplanmasını göstermektedir. Burada her mini-yığındaki gradyanlar hesaplandıktan sonra ağırlıklar güncellenmektedir. Böylelikle eğitim süreci hem daha hızlı hem daha yumuşak bir inişle sürdürülmektedir.

$$\theta = \theta - \alpha \nabla_{\theta} J(\theta, x^{(i:i+n)}; y^{(i:i+n)}) \tag{2.12}$$

2.4.5.2 Momentum

Gradyan iniş yöntemlerindeki öğrenme oranını seçmek her zaman kolay olmayabilir. Öğrenme oranının aşırı küçük olması öğrenme sürecinin çok uzamasına, oranın çok büyük olması ise optimal noktanın yakalanamamasına sebep olmaktadır. Bu sorunun çözüm yöntemi olarak ise öğrenme sürecinde momentum kullanılmaktadır. Momentumun formülü denklem (2.13)'te gösterilmiştir. Burada v_t modelin parametrelerinin t. adımdaki değişim miktarıdır. Bu denklemde v_t v_{t-1} 'e ve hata fonksiyonundaki değişime öğrenme oranı kadar bağlıdır. Böylelikle gradyan inişi hata oranının yüksek olduğu yerlerde daha büyük adımlar ile gerçekleştirilirken hata oranının azaldığı yerlerde küçülmeye başlar.

$$\vartheta_t = \gamma\vartheta_{t-1} + \alpha\nabla_\theta J(\theta)$$
$$\theta = \theta - \vartheta_t \tag{2.13}$$

2.5 Derin Öğrenme

Derin öğrenme (Deep Learning, DL) insan nöronlarının çalışma mantığından esinlenerek ortaya çıkarılmıştır. 1943'te Warren McCulloch ve Walter Pitts beyindeki nöronların çalışması hakkında bir makale yazdılar [13]. 1950 yıllarında teknolojinin ilerlemesiyle birlikte varsayımsal yapay sinir ağları (Artificial Neural Network, ANN) simüle edilmeye başlandı [14]. 1962 yılında Widrow ve Hoff bir perceptron modeli için öğrenme algoritması geliştirdi [15]. 1960'tan sonra yapay sinir ağının çözümü için birçok bilim insanı çalışmalar yaptı ve teorik olarak çözüm yöntemi geliştirildi[16]. Yapay sinir ağının geri yayılım algoritması ve optimizasyon teknikleri bulunmasına rağmen bu algoritmalar bilgisayar gücünün ve veri sayısının artmasıyla birlikte 1990'lı yıllardan sonra popülerlik kazanmaya başladı [16]. ANN modelinin üstüne daha fazla katman ve nöron ekleyerek elde edilen model derin sinir ağlarıdır (Deep Neural Networks, DNN). DNN modelleri ANN modelinin çözemediği daha karmaşık problemleri çözmektedir. ANN ve DNN modelleri genellikle veri sayısının çok yüksek olduğu durumda ML modellerinden daha iyi performans göstermektedir[16]. Bu bölümde bu derin öğrenme modelleri ve zaman serisi problemlerinde kullanılabilecek olan bu türevleri anlatılacaktır.

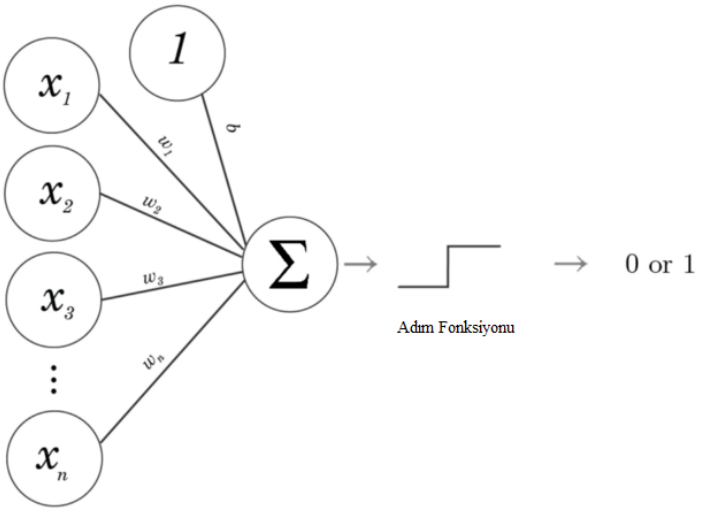
2.5.1 Yapay sinir ağları

Yapay sinir ağları insan beyninin çalışma sistemini modelleyen yöntemdir. İnsan beynindeki nöronların çalışma prensibi matematiksel olarak modellenmiştir. ANN modelleri gerçek hayatta birçok problemi çözümleyebilme kapasitesine sahiptir. Fonksiyon gücü olarak birçok modelin gösteremediği karmaşık fonksiyonları ifade edebilir. Bu fonksiyonların iyi çalışabilmesi için özellikle yüksek sayıda veri gerekmektedir. ANN modellerine farklı sayıda katmanlar ve nöronlar eklenerek çözümlenmesi zor olan problemler çözümlenebilmektedir. Bu bölümde ANN modellerinin temelleri anlatılacaktır.

2.5.1.1 Perceptron

Perceptron yapay sinir ağının bir hücreli modellenmiş halidir. Bu model öznitelik vektörünü girdi olarak aldıktan sonra bu özniteliklerin ağırlık vektörü ile skaler çarpımını hesaplamaktadır. Bu skaler çarpımın sonucu bir aktivasyon fonksiyonundan geçirilmektedir. Bu şekilde lineer olmayan bir f(x) fonksiyonu oluşturarak veriler eşlenmeye çalışılmaktadır. Bir perceptron hücresine giren özniteliklere ağırlıklar atanarak (lineer regresyon algoritması gibi) etiketlenen veri tahmin edilmeye çalışılmaktadır. Aktivasyon fonksiyonu perceptronun lineer olmayan bileşeni olarak eklenmektedir. Bu şekilde perceptron hücreleri daha karmaşık problemleri çözebilmektedir.

Şekil 2.18'de perceptron modelinin örnek bir çizimi gösterilmiştir [17]. Bu model çıktı olarak 0 ya da 1 üretmektedir. Burada her öznitelik X_i belirli bir ağırlık değeri W_i ile çarpıldıktan sonra aktivasyon fonksiyonu olan adım fonksiyonundan geçirilmektedir. Adım fonksiyonu belirli bir eşik değerinin altındaki değerleri sönümleyerek 0 üretmektedir. Eşik değerinin üstünde kalan değerler ise 1 olarak üretilmektedir. Bu modelin ileri doğru beslemesi (forward progapation) bu şekilde gerçekleşmektedir.



Şekil 2.18: Perceptron modeli.

Perceptron modelinde ağırlık değerlerinin hesaplanması için gradyan iniş tekniği kullanılmaktadır. Her ağırlık değeri için türev alınarak uzayda optimal değerler aranmaktadır. Perceptron modeli lineer olmayan bileşen içermesine rağmen her problemi çözebilecek kapasiteye sahip değildir.

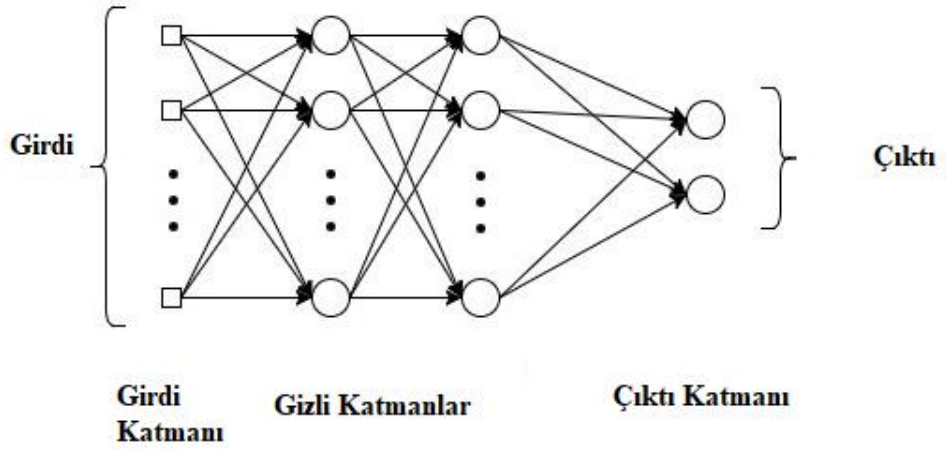
2.5.1.2 Çok katmanlı perceptron

Perceptron modeli basit problemleri çözebilmektedir çünkü karmaşık bir beyin yapısındaki tek bir nöron hücresi gibi davranmaktadır. Bu nöronların çok sayıda birleşmesiyle birlikte çok katmanlı perceptron yapısı ortaya çıkmaktadır. Perceptron hücrelerinin alt alta dizilmesi ile birden çok hücreli bir yapay sinir ağı ortaya çıkmaktadır. Bu şekilde oluşan yapıya bir katman (layer) adı verilir. Birden çok katmanın birleşmesiyle oluşan yapıya çok katmanlı perceptron (Multilayer perceptron, MLP) denir. Burada bir girdi vektörü ilk katmandan girdi olarak verilir ve bir sonraki katmandaki nöronlara bu bilgiler aktarılır. İlk katmana girdi katmanı (input layer), ara katmanlara gizli katman (hidden layer) ve modelin çıkışındaki katmana ise çıktı katmanı (output layer) ismi verilmektedir.

MLP modeli öznitelik uzayını farklı bir uzaya taşıyarak karmaşık ve lineer olmayan problemleri lineer hale getirmektedir. Bu şekilde ilerleyen katmanlarda problemi çözebilmektedir. Girdilerin uzayda çözümlenebilir bir hale getirilmesi için katman sayısı ve nöron sayısı büyük bir önem taşımaktadır. DNN modelleri ikiden daha fazla gizli katman içermektedir. Bu şekilde daha karmaşık problemleri çözebilme yeteneği

oluşmaktadır. DNN modellerinin başarılı olması için veri sayısının yüksek olması gerekmektedir.

Şekil 2.19'da MLP modelinin örneği gösterilmiştir [18]. Bu model bir girdi, 2 gizli ve bir tane çıktı katmanından oluşmaktadır. Çıktı katmanında ise temsili olarak 2 nöron bulunmaktadır. Çıktı katmanındaki nöron sayısı problemin tipine göre değişiklik gösterebilir. Örneğin bir regresyon probleminde çıktı katmanında bir nöron kullanılabilir. Bir sınıflandırma probleminde ise tespit edilmek istenen sınıf sayısı kadar nöron kullanılabilir. Bu konunun devamı aktivasyon fonksiyonlarında anlatılacaktır.



Şekil 2.19: MLP modeli.

2.5.2 Aktivasyon fonksiyonu

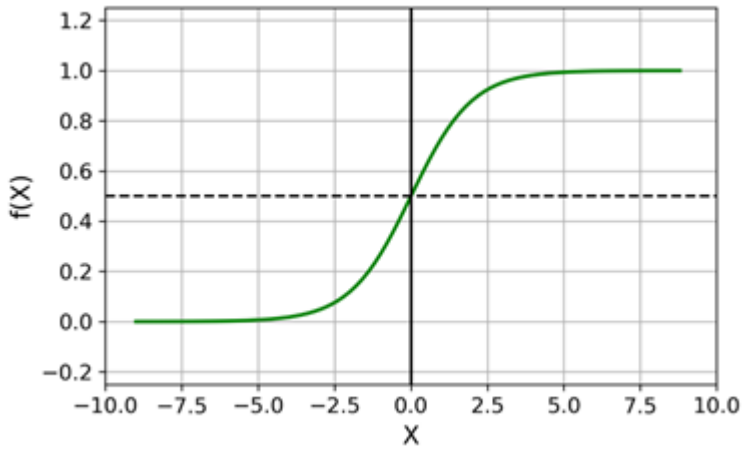
Aktivasyon fonksiyonları ANN ve DNN modellerinde her bir katman arasında kullanılan fonksiyonlardır. Aktivasyon fonksiyonu bir nöronun ürettiği değerlerin bir sonraki nörona nasıl aktarılacağına karar veren bir fonksiyondur. Biyolojik olarak beyin hücrelerinin çalışma mantığından esinlenerek tasarlanmış olan bu fonksiyonlar eğer üretilen değerler belirli bir değerden düşükse o nöronun çıktısını sönümler. Aynı zamanda bu fonksiyonlar lineer olmayan fonksiyonlar olduğundan dolayı modeli lineer olmayan bir modele dönüştürmektedir. Lineer olan bir ANN ve DNN modelinin zayıf noktalarından birisi elde edilen çıktı vektörünün girdi vektörünün bir afin kombinasyon (affine combination) şeklinde olmasıdır. Bu modelin ürettiği çıktının gerdiği uzay basit bir lineer regresyon modelinin gerdiği uzay ile aynı olmaktadır. Bundan dolayı ANN ve DNN modellerinde aktivasyon fonksiyonları

kullanılmaktadır. Literatürde en çok kullanılan aktivasyon fonksiyonları sırayla açıklanmıştır.

2.5.2.1 Sigmoid fonksiyonu

Sigmoid fonksiyonu popüler aktivasyon fonksiyonlarından bir tanesidir. Denklem (2.14)'de sigmoid fonksiyonunun formülü gösterilmiştir. Aynı zamanda Şekil 2.20'de Sigmoid fonksiyonun grafiği verilmiştir [19]. Bu fonksiyon 0-1 ile arasında değer üretmektedir ve özellikle belirli bir eşik değerinden büyük olan değerleri doğrudan 0 ve 1'e eşleştirmektedir. Bu şekilde nöronun çıktısı 0-1'e yakın sayılardan oluşmaktadır.

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.14}$$



Şekil 2.20: Sigmoid fonksiyonun gösterimi.

Sigmoid fonksiyonun bazı avantajları:

- 0-1 ile arasında değerler ürettiğinden dolayı her nöronun çıktısını normalize etmektedir.
- 0-1 ile arasındaki değerler olasılık içeren problemler açısından mantıklı seçimler olabilir.
- Gradyanlar daha yumuşak bir şekilde hesaplanmaktadır ve bu fonksiyon herhangi bir noktada türevlenebilir.

Bu fonksiyonun aynı şekilde bazı dezavantajları da bulunmaktadır. Bunlar:

- Bu fonksiyon bazı üssel ifadeler içermektedir. Bu yüzden bilgisayarda hesaplanış hızı yavaştır.
- Sigmoid fonksiyonunun türevinde büyük ve küçük değerlerin karşılık geldiği değerler çok küçük olmaktadır. Bundan dolayı gradyanlar hesaplanırken belirli değerlerden büyük ve küçük gradyanlar sıfırlanmaktadır. Bu da gradyan iniş algoritmasının bazı değerlerde ciddi zayıflamasına sebep olmaktadır.

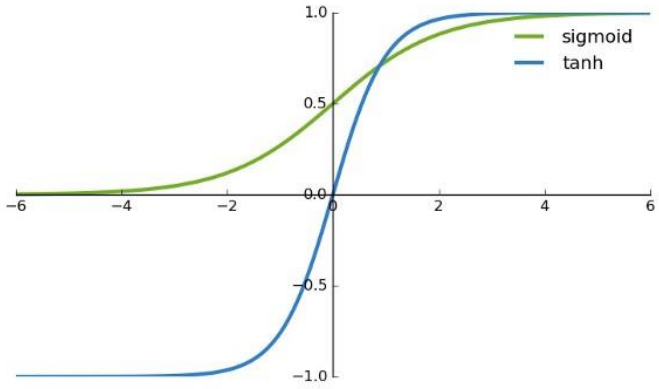
2.5.2.2 Tanh fonksiyonu

Tanh aktivasyon fonksiyonu sigmoid aktivasyon fonksiyonun biraz değiştirilmiş bir halidir. Denklem (2.15)'te tanh fonksiyonun formülü verilmiştir.

$$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1 \tag{2.15}$$

Tanh aktivasyon fonksiyonu sigmoid aktivasyon fonksiyonun taşıdığı özellikleri taşımaktadır ve aynı şekilde türevlenebilir bir fonksiyondur. Sigmoid fonksiyonundan en büyük farkı Şekil 2.21'de görüldüğü üzere tanh fonksiyonun çıktısı [-1,1] arasında olmaktadır [19]. Tanh fonksiyonu gelen değerleri -1 ve 1'e yakınlaştırdığı gibi özellikle 0 civarındaki değerleri de 0 yaklaştırarak sigmoid fonksiyonundan farklı eşleştirme yapmaktadır. Tanh fonksiyonun türevi sigmoid fonksiyonun türevine benzemektedir. Bu yüzden gradyanlar küçük ve büyük değerlerde aynı şekilde sıfırlanmaktadır.

Sınıflandırma problemlerinde DNN modellerinin ara katmanlarında çoğunlukla tanh fonksiyonu çıktı katmanında ise sigmoid fonksiyonu kullanılmaktadır.



Şekil 2.21: Sigmoid ve tanh fonksiyonu.

2.5.2.3 ReLU

Doğrusal doğrultmaç ünitesi (Rectified Linear Unit, ReLU) fonksiyonu doğrusal olmayan fonksiyonun doğrusal hali olarak düşünülebilir. Denklem (2.16)'da ReLU fonksiyonu gösterilmiştir. Bu formüle göre girdinin 0'dan düşük olduğu yerlerde ReLU fonksiyonu çıktı olarak 0 üretmektedir. Eğer fonksiyona giren değerler sıfırdan büyükse bu fonksiyon çıktı olarak girdinin kendisini üretmektedir.

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.16}$$

ReLU aktivasyon fonksiyonu diğer aktivasyon fonksiyonlarından daha yaygın olarak kullanılmaktadır. Sigmoid ve tanh fonksiyonlarıyla karşılaştırıldığında bazı avantajları şunlardır:

- ReLU fonksiyonun pozitif olan parçasında gradyan değerlerinin satürasyon sorunu ortadan kalkmaktadır. ReLU fonksiyonun türevi sabit olduğundan dolayı küçük ya da büyük gradyanlar sönümlenmeyecektir.
- ReLU fonksiyonu hesaplanırken sigmoid ve tanh fonksiyonundan daha hızlı bir şekilde hesaplanmaktadır. Bundan dolayı çok derin yapay sinir ağlarında hesaplama hızını ciddi bir şekilde artırmaktadır.

ReLU fonksiyonun bu avantajlarına rağmen bazı dezavantajları da bulunmaktadır:

- ReLU fonksiyonuna sıfırdan küçük değerler girerse bu değerler sıfırlanmaktadır. Bu durum aslında modelin ileri yayılma (forward propation) kısmında bir sorun oluşturmamaktadır. Burada ağırlıklar hesaplanarak hangi değerlerin sönümlenebileceği ayarlanabilir fakat geri besleme (backpropation) kısmında negatif gradyanlar yine sıfırlanacağı için model bu kısımlarda yine eğitilemeyecektir.
- ReLU fonksiyonu sıfır merkezli bir fonksiyon değildir (Sigmoid fonksiyonu gibi).

2.5.2.4 Softmax

Softmax aktivasyon fonksiyonu çoklu sınıf problemlerinde kullanılmakta olan bir aktivasyon fonksiyonudur. Bir sınıflandırma probleminde K adet sınıf varsa ANN çıktısında da K adet nöron bulunmaktadır. Bu K adet sınıfın hangi olasılıklar ile hangi sınıfa ait olduğunun belirlenebilmesi için nöronların çıktılarının [0,1] aralığına çekilmesi ve aynı zamanda bu nöronların çıktılarının toplamının 1'e eşit olması gerekmektedir. Softmax fonksiyonu bu özellikleri sağlayabilen bir fonksiyondur. Softmax fonksiyonun denklem (2.17)'de gösterilmiştir.

$$f(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} x_j} \tag{2.17}$$

Softmax fonksiyonu sınıflandırma problemlerinde çıktı katmanında kullanılmaktadır. Çıktıdaki nöronların her birinin değerini normalize ederek 0-1 arasına çekmektedir. Bu şekilde her bir sınıf değeri için olasılık üretmektedir. Softmax fonksiyonu isminde de anlaşılabileceği gibi argmax fonksiyonunun yumuşak halidir. Çıktıdaki değerlerin bir sınıfa ait olma olasılığı yerine diğer sınıflardaki değerleri de önemseyecek bir şekilde ölçeklendirme yapmaktadır.

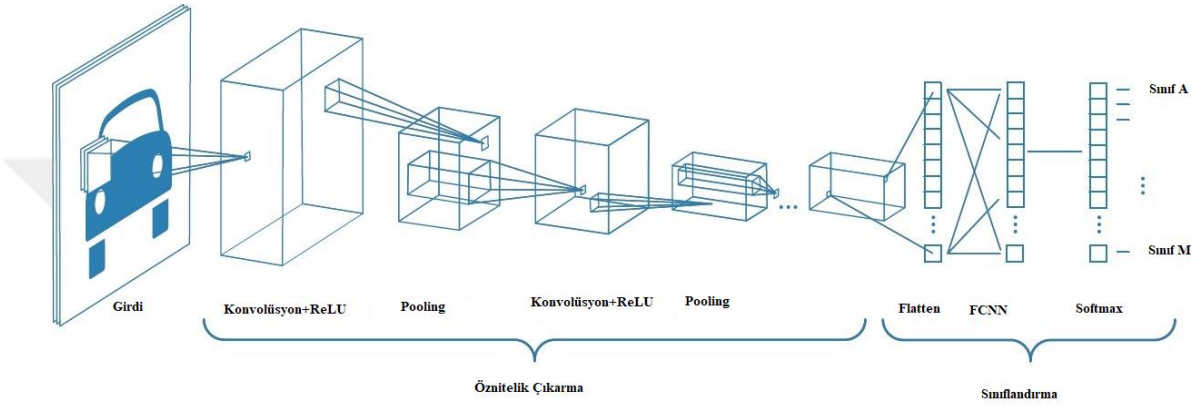
2.5.3 Evrişimsel sinir ağı

Evrişimsel sinir ağı (Convolutional Neural Network, CNN) insanın sinyalleri algılama mantığı düşünülerek geliştirilmiş olan bir modeldir. İnsanlar sinyalleri algıladığı zaman bazı filtrelerden geçirerek o sinyallere belirli anlamlar kazandırmaktadır. Örnek olarak insanın gözlemlediği bir cisim için yorum yapması bu şekilde gerçekleşmektedir. İnsan gözlemlediği cisimleri bazı filtrelerden geçirdikten sonra hangi nesne olduğunu algılar. Beyin bir nesnenin şekil, renk gibi temel bileşenlerini algıladıktan sonra daha detaylı bilgilere odaklanır.

CNN modeli herhangi bir boyuttaki sinyale uygulanabilir fakat genellikle 2 boyutlu görüntü odaklı problemlerde daha yaygın olarak kullanılmaktadır. Bu tez kapsamında CNN modeli 2 boyut üzerinde uygulandığından dolayı açıklamalar 2 boyut üzerinden yapılmaktadır.

Şekil 2.22'de evrişimsel sinir ağının bir örneği verilmiştir. CNN modellerinin temel yapısı bu örnekteki gibidir fakat katman sayısı filtre sayısı ve kullanılan aktivasyon fonksiyonları çeşitlilik gösterebilir [20]. Bir CNN modeli ilk olarak girdiyi matris

olarak almaktadır. Bu matris ilk olarak evrişim işleminden (konvolüsyon) geçirilmektedir. Bu şekilde resimdeki önemli bilgiler gelecek katmana aktarılmaktadır. Daha sonra elde edilen matris havuz katmanından geçirilerek işlenecek olan verinin boyutu düşürülmektedir. Bu şekilde veri belirli katmanlardan tekrar tekrar geçirildikten sonra tek bir vektöre dönüştürülmektedir. Bu oluşan vektör girdi vektöründen elde edilen öznitelik vektörüdür. Bu öznitelik vektörü tamamen bağlı sinir ağına (Fully Connected Neural Networks, FCNN) aktarılarak problem çözümlenmektedir.



Şekil 2.22: Evrişimsel sinir ağı örneği.

CNN modelleri ve türevleri genellikle büyük veri setleri üzerinde çalışmaktadır. CNN modelleri kendi kendine öznitelik çıkarımı yapabildiğinden dolayı fazla sayıda veriye ihtiyaç duymaktadır. Örneğin bir nesne sınıflandırma görevinde ilk katmanlar genellikle obje hakkında az detay içeren bilgileri öğrenmektedir. Derin katmanlarda ise nesne hakkında daha detaylı bilgiler öğrenilmektedir. Bir nesnenin çok sayıda detayı olabileceğinden dolayı çok fazla sayıda katmana ve veriye ihtiyaç duyulmaktadır. Günümüz teknolojisinin ilerlemesiyle birlikte veri toplama işlemi kolaylaşmıştır ve CNN'lerin eğitilebilmesi için gerekli olan sayıda veriler elde edilmeye başlanmıştır. CNN modelinin katmanları ve detayı alt bölümlerde açıklanmaktadır.

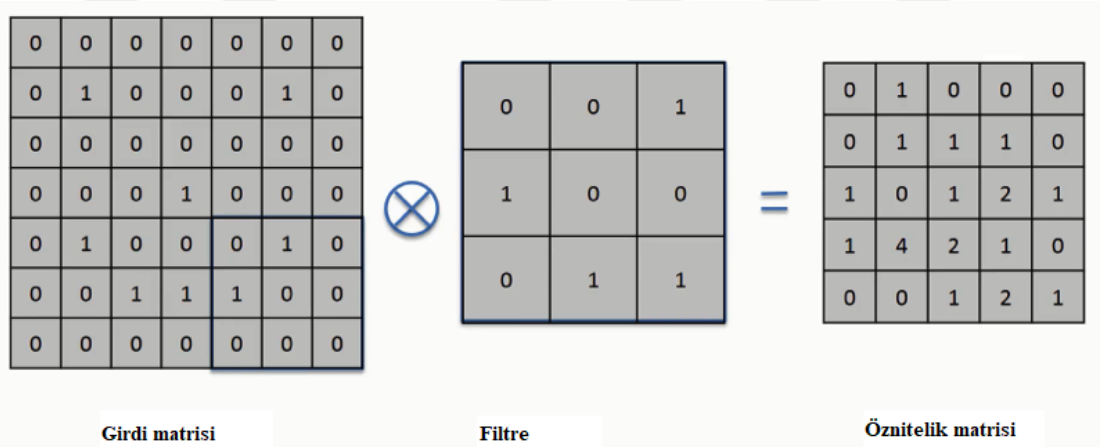
2.5.3.1 Konvolüsyon katmanı

CNN modelinde konvolüsyon katmanı girdi olarak verilen matris ya da vektöre filtreleme işlemi uygulamaktadır. Bu şekilde gelecekteki katmanlara gerekli bilgiler aktarılmaktadır. Örnek olarak CNN modelinde bazı filtrelerin yüzün göz bölgesini, bazılarının burun, dudak, kulak gibi farklı organları tanımaya çalışması verilebilir.

Bu şekilde her filtre girdi olarak verilen resim üzerinden farklı bilgiler çıkararak işlenmesini kolaylaştırmaktadır.

Şekil 2.23'de konvolüsyon işleminin nasıl uygulandığı 2 boyutlu bir matris üzerinde gösterilmiştir [21]. Burada filtre girdi matrisinin her bir elemanın orta noktasına denk gelecek şekilde yerleştirilir ve o bölge üzerinde matrislerin skaler (nokta) çarpımı yapıldıktan sonra elde edilen değer çıktı matrisi olan öznitelik matrisinin karşılık gelen indeksine yazılır. Bu işlem tüm matris üzerinde uygulanır ve en son öznitelik matrisi oluşur. CNN üzerinde bu konvolüsyon filtrelerinin bazı özellikleri şunlardır:

- Filtre boyutu (DL literatüründe genellikle 3x3 kullanılır)
- Filtrenin kayma miktarı (Stride)



Şekil 2.23: Konvolüsyon işleminin örneği.

Filtrenin kayma miktarı bir sonraki katmana aktarılacak olan matrisin boyutunu etkilemektedir. Örneğin, bu miktar 2 ise matrisin boyutu yarı yarıya düşecektir. CNN modelinin en temel amacı bu filtrelerin içindeki değerlerin ne olacağını hesaplamaktır. Bu mantıkla birlikte klasik bilgisayarda görü problemlerinin ötesine geçilmiş bulunmaktadır. Model en son eğitildikten sonra her filtre ne yapacağını öğrenmiş olduğundan dolayı tüm model istenilen problemi çözebilmektedir. Bu yüzden dolayı filtre sayısı, boyutu ve bu filtrelerin kayma miktarı da önemli parametreler olmaktadır.

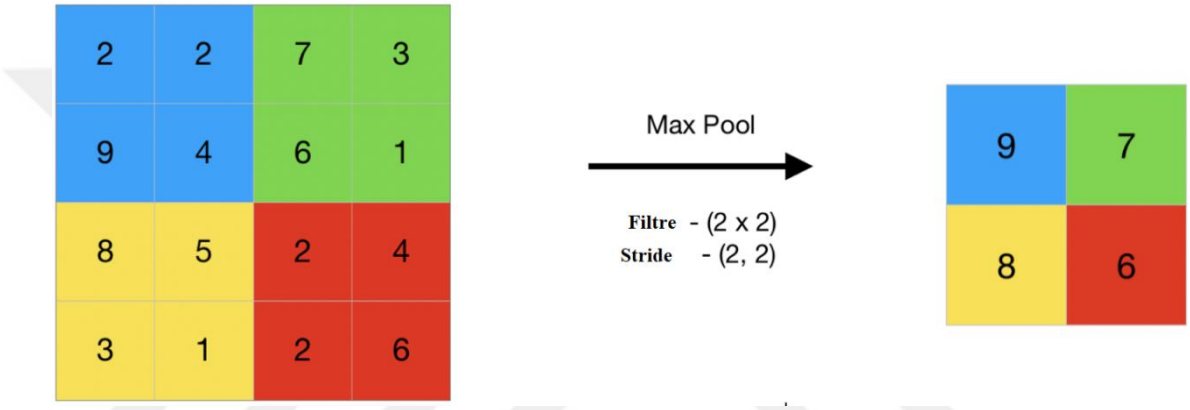
2.5.3.2 Havuz katmanı

CNN modelinde bir diğer kullanılan katman ise havuz katmanıdır (pooling layer). Bu katman çok büyük matrisler ile işlem yapmak zor olduğundan dolayı matrislerin

boyutlarını indirmektedir. Bunun için birçok farklı yöntem olabilir. Bunlardan en çok kullanılanlar şunlardır:

- Maksimum havuz katmanı (Max pooling)
- Ortalama havuz katmanı (Average pooling)
- Minimum havuz katmanı (Minimum pooling)

CNN modelinde popüler olarak kullanılan havuz katmanı filtresi maksimum havuz katmanıdır. Şekil 2.24'te maksimum havuz katmanın filtre boyutu 2x2 ve kayma miktarı (2, 2) için bir örneği gösterilmiştir [22].



Şekil 2.24: Maksimum havuz katmanın şekil üzerinde örneği.

2.5.3.3 Tamamen bağlı katman

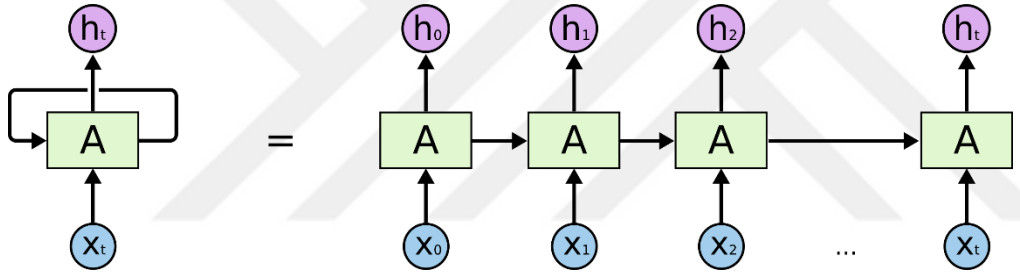
CNN modelinde konvolüsyon işlemlerinden ve havuz katmanından geçirilmiş olan girdi bir öznitelik vektörüne dönüştürüldükten sonra FCNN 'e verilir. Bu katmana da tamamen bağlı katman denmektedir. FCNN katmanı asıl öğrenme işleminin gerçekleştirildiği katmandır. Burada problemin tipine göre çıktı nöronları belirlenir ve daha sonra model geri yayınım tekniğiyle optimize edilerek verilen regresyon ya da sınıflandırma problemi çözülür.

2.5.4 Özyinelemeli sinir ağı

MLP ve CNN modelleri gibi modellerin temel eksik noktalarından bir tanesi hafıza hücrelerinin bulunmamasıdır. Bu yüzden bu modeller bazı problemleri modelleyememektedir. Örneğin bir filmde gerçekleşen olayları sınıflandırmaya çalıştığımız bir problemde CNN modelleri yetersiz gelecektir çünkü geçmiş zamanda olan olaylar ile yeni gerçekleşen olayları birleştiremeyecek ve

anlamlandıramayacaktır. Bu hafıza sorunun çözümü için özyinelemeli sinir ağı (Recurrent Neural Network, RNN) modeli tasarlanmıştır. RNN modeli geçmişten gelen bilgiyi bir sonraki nörona aktararak bir modelleme yaptığından dolayı geçmişteki bilgiyi unutmamaktadır.

Şekil 2.25'te bir RNN hücresi ve açılmış hali şekil üzerinde gösterilmiştir. Burada bir RNN hücresi olan A gösterilmiştir [23]. Bu hücre her bir zaman diliminde X_{t-1} şeklinde bir girdi almaktadır. X_t girdi bir önceki gelen durum bilgisi ile h_{t-1} bilgisi birleştirilerek h_t üretilmektedir. Bu h_t t anındaki modelin tahmini olarak düşünülebilir. Bu tahmin istenirse bir ağırlık matrisi V ile çarpılarak ve bir skaler değeri c değeri eklenerek ek bir işlem ile de üretilebilir ve pratikte daha çok bu şekilde kullanılmaktadır. Burada önemli olan nokta bir önceki gizli durum (hidden state) bir sonraki adımda kullanıldığından dolayı bu model geçmişteki bilgiyi geleceğe taşıyarak bir modelleme yapmaktadır.



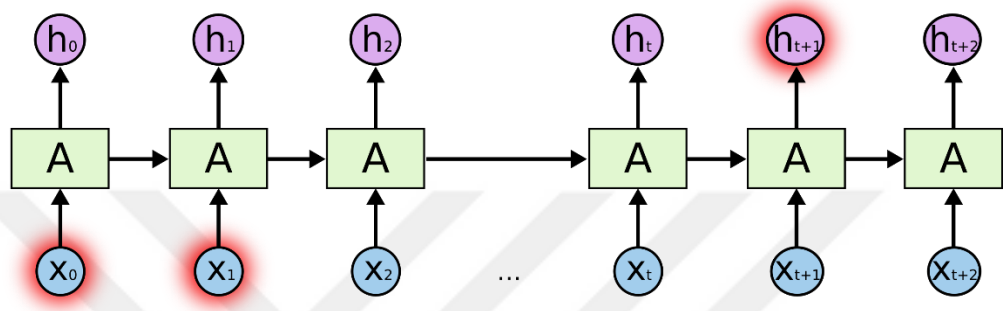
Şekil 2.25: RNN hücresi ve açılmış hali.

$$h_t = \tanh(b + Wh_{t-1} + UX_t) \tag{2.18}$$

Denklem (2.18) h_t değerinin hesaplanma formülünü göstermektedir. W ve U matrisleri ağırlık matrisleri h_t değerinin düzgün bir şekilde tahmin edilebilmesini sağlamaktadır. Denklem (2.19)'da o_t tahmin değerlerinin hesaplamasını gösteren bir formül verilmiştir. Bu formülde V matrisi üretilen h_t değerlerine belirli ağırlıklar atayarak çıktı olarak tahmini vermektedir. Bu iki denklemde bulunan b ve c ağırlık değerleri tahminleri düzenleyen değerlerdir (bias). RNN modeli bu şekilde modelleme yapmaktadır ve ana amaç W, U, b ve c ağırlık değerlerini bulmaya çalışmaktır.

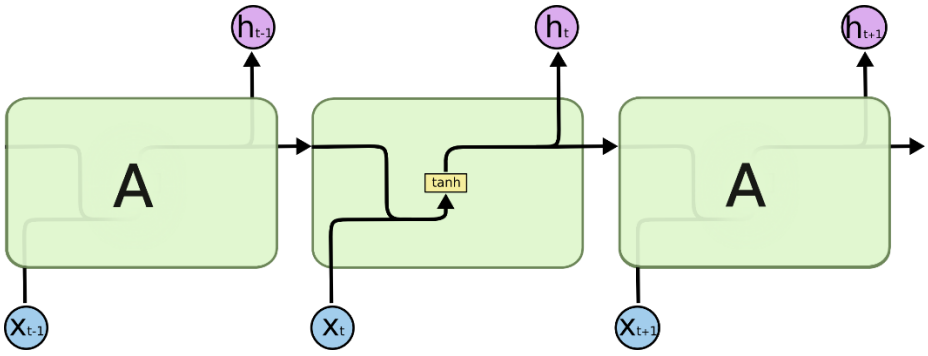
$$o_t = c + V\,h_t \tag{2.19}$$

Şekil 2.26'da RNN hücresinin uzun süreli geçmişe bağlılık (long-term dependency) sorunu gösterilmiştir [23]. RNN hücresindeki bu sorun modellemeyi kötü yönde etkilemektedir. Sıralı ya da zaman serisi problemlerinde her zaman çok geçmişte kalan bilgiler önemli olmayabilir. Örnek olarak bir paragrafın içerisindeki cümle ile 2 paragraf önceki cümlenin bir ilişkisi bulunmayabilir fakat RNN modeli doğrudan geçmişteki tüm bilgileri işlemeye çalışmaktadır. Bundan dolayı da bazı problemlerde başarısız bir modelleme yapmaktadır.

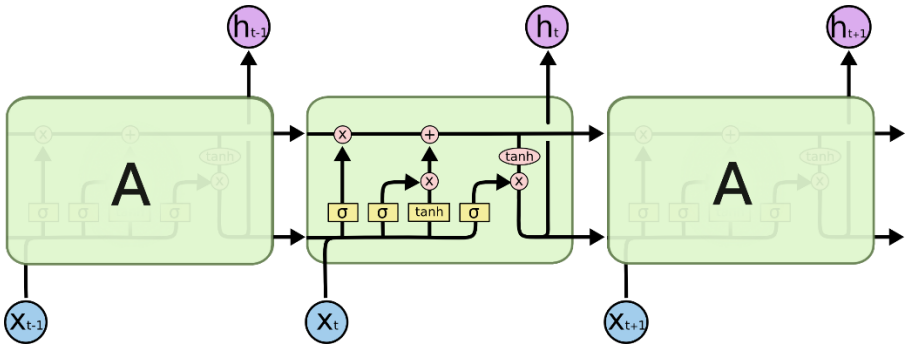


Şekil 2.26: RNN hücresi ve uzun süreli geçmişe bağlılık sorunu.

Bu sorunu çözmek için LSTM (Long Short Term Memory) modeli geliştirilmiştir. Şekil 2.27'de klasik bir RNN hücresinin yapısı gösterilmiştir [23]. RNN hücresi bir önceki durumdan gelen bilgiyi doğrudan sonraki nörona iletmiştir. Şekil 2.28 ise LSTM hücresinin yapısını göstermektedir [23]. Bu modelde önceden gelen bu durum bilgisi doğrudan verilmemektedir. Model uzun süreli geçmişten gelen bilgiyi kullanmak zorunda değildir. Bu da LSTM modelinin RNN modelinden daha iyi bir modelleme yapabilmesini sağlamaktadır [23].

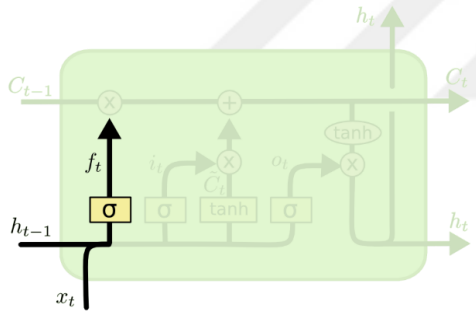


Şekil 2.27: RNN hücre yapısı.



Şekil 2.28: LSTM hücre yapısı.

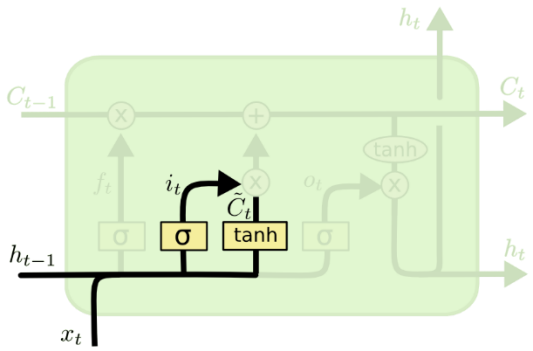
Şekil 2.29'da LSTM unutma kapısı gösterilmiş ve denklemi verilmiştir. Burada unutma kapısı önceden gelen h_{t-1}, girdi x_t ve W_f 'e bağlı olarak C_{t-1} hücre bilgisinin gelecekteki nöronlara ne kadar iletileceğini belirlemektedir [23]. Bu sigmoid fonksiyonun çıktısı (f_t) eğer 0 civarında ise bir önceki hücrenin durumu olan C_{t-1} sıradaki hücreye iletilmeyecektir ya da az iletilecektir. Eğer bu fonksiyonun değeri 1'e yaklaşırsa C_{t-1} 'in değeri sıradaki hücreye iletilecektir. Bu şekilde bu unutma kapısı sayesinde istenilen durumlarda önceki hücrelerin durum bilgisi yeni hücreye iletilmektedir.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Şekil 2.29: LSTM unutma kapısı (forget gate).

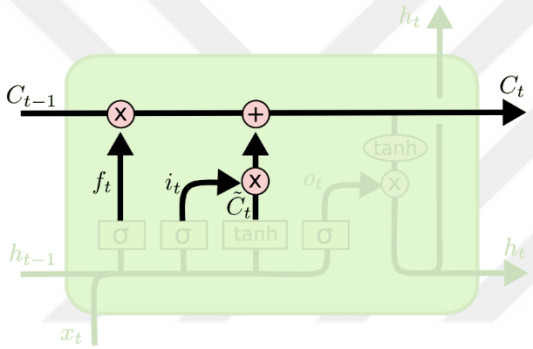
Şekil 2.30 LSTM girdi kapısının yapısını göstermektedir [23]. Burada i_t sigmoid fonksiyonun çıktısını göstermektedir. Tanh aktivasyon fonksiyonun çıktısında aday (candidate) hücre durumu oluşmaktadır. Bu iki durum birleşerek bir durum güncellemesi gerçekleştirilecektir.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Şekil 2.30: LSTM girdi kapısı (input gate).

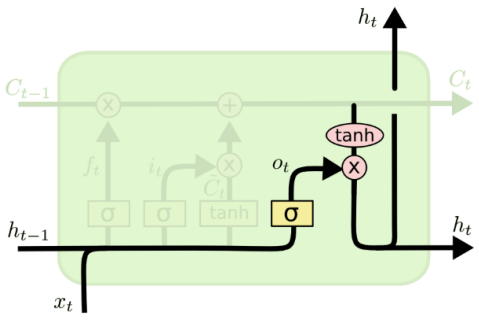
Şekil 2.31 LSTM hücre bilgisinin güncellenme şekli gösterilmiştir [23]. Burada önceden gelen hücre bilgisi f_t ve yeni oluşturulan hücre bilgisi i_t değerleri ile çarpılarak C_t yeni hücrenin değeri belirlenmektedir.



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

Şekil 2.31: LSTM hücre bilgisinin güncellenmesi.

Şekil 2.32 LSTM hücresinin gizli durum bilgisinin güncellenmesini göstermektedir [23]. Burada hücre bilgisi C_t 'nin ne kadarlık bir kısmının taşınacağı hesaplanmaktadır. Burada o_t (çıktı kapısı) bu hücre bilgisinin ne kadarlık bir kısmının geçeceğini ve nasıl geçeceğini hesaplamaktadır. Bu şekilde gelecekteki hücreye bir gizli durum bilgisi iletilmektedir. LSTM modeli eğitilirken temel amaç bu hesaplamaları yapabilecek olan ağırlık matrislerinin optimize edilmesidir.



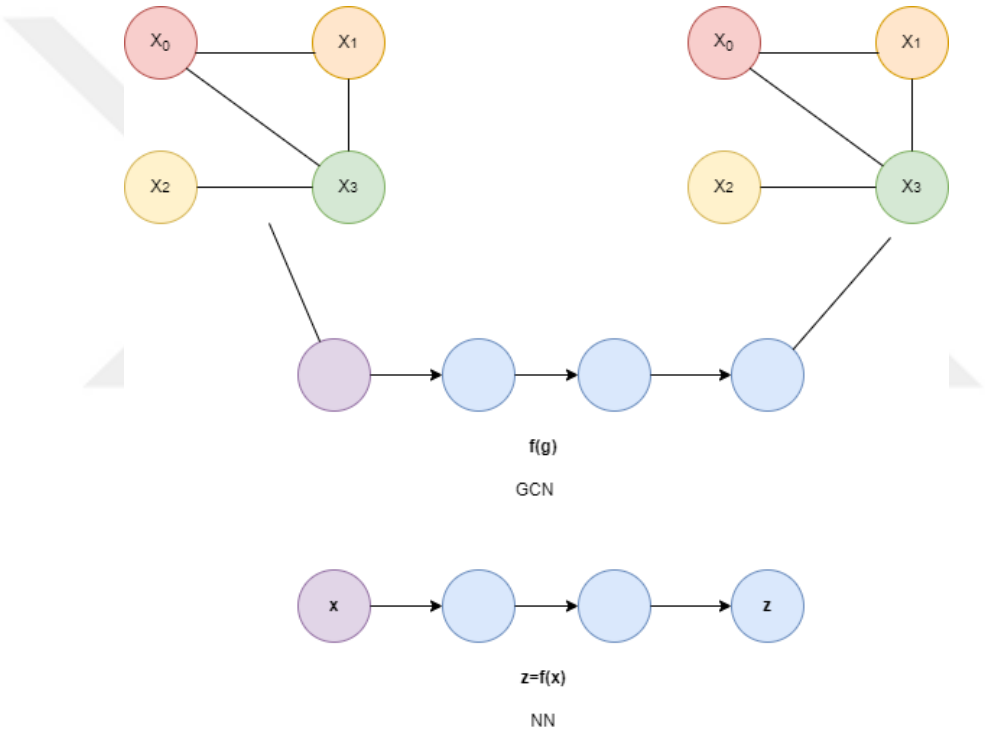
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Şekil 2.32: LSTM hücresinin gizli durum bilgisinin güncellenmesi.

2.5.5 Çizge evrişimsel ağı

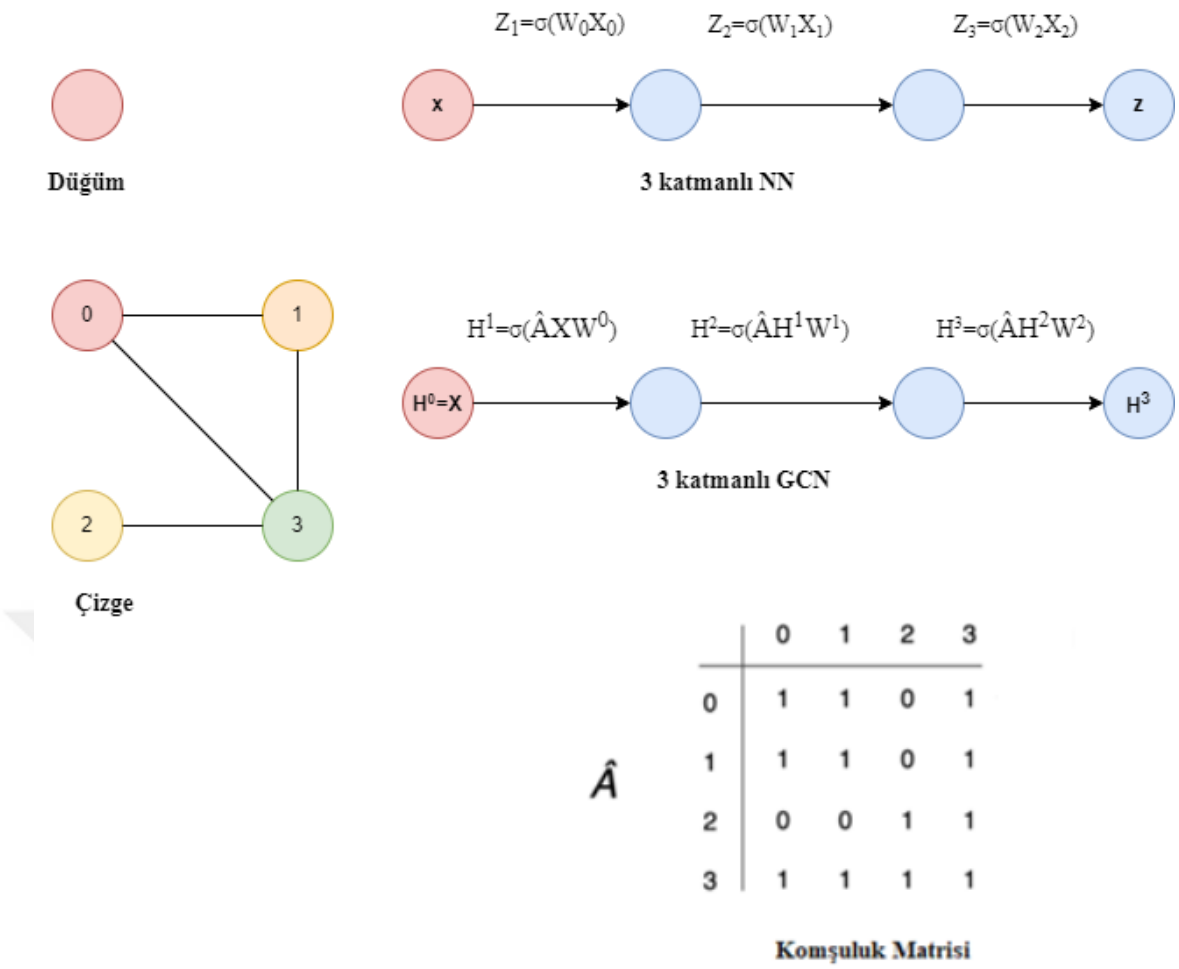
Çizge sinir ağları (Graph Neural Networks, GNN) modelleri klasik ANN modellerinden farklı olarak Öklid uzayının dışında modelleme yapabilmektedir. Bu modellerin bir türü olan çizge evrişimsel ağı (Graph Convolutional Network, GCN) bu tez kapsamında açıklanacaktır.

Şekil 2.33 klasik sinir ağları (Neural Networks, NN) ile GCN modelinin arasındaki farkı göstermektedir [24]. NN modelleri tek bir düğüm için hesaplama yaparak bir z elde etmektedir. GCN modellerinde girdi olarak birden çok düğüm için x_i öznitelikleri verilmektedir ve çıktı olarak ise her bir düğüm için z_i değeri elde edilmektedir.



Şekil 2.33: NN ve GCN modeli farkını gösteren bir şekil.

Şekil 2.34 NN ve GCN modellerinin ileri yayılımını göstermektedir [24]. Burada GCN modeli komşuluk matrisi Â ve X öznitelik matrisi ile hesaplama yapmaktadır. NN modeli ise daha önceden bahsedildiği gibi ileri yayılım yapmaktadır. GCN modelinde her katmanda girdi matrisi komşuluk matrisi ile çarpılarak bir sonraki katmanın değerleri hesaplanmaktadır.

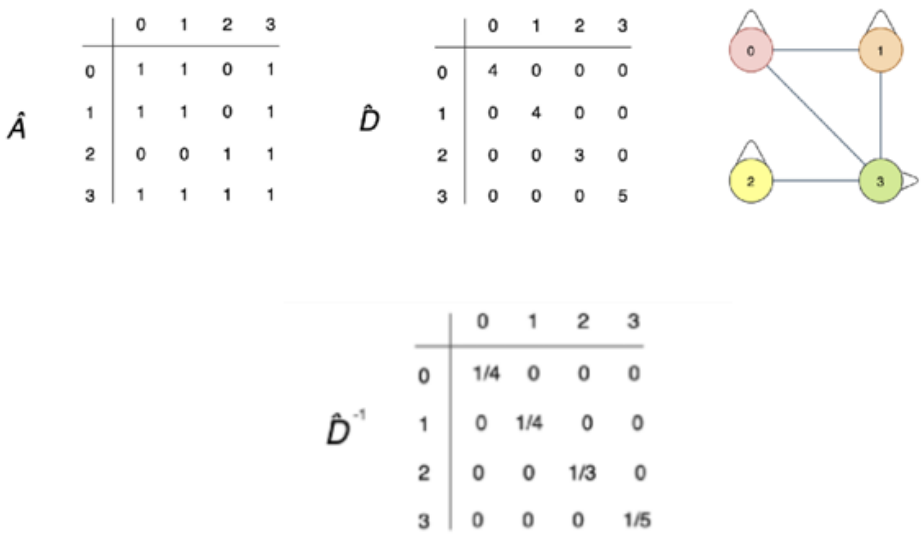


Şekil 2.34: NN ve GCN modelinin ileri yayılımı.

GCN modelinde Â matrisi herhangi bir normalizasyon işleminden geçirilmemiştir. Bu sorunu çözmek için denklem (2.20)'de gösterilen işlem uygulanmıştır. Burada düğümlerin dereceleri (degree) hesaplanarak bir matris oluşturulmaktadır. Bu matris $\hat{\mathbf{D}}$ matrisi olarak isimlendirilmektedir. Bu matrisin tersi $\hat{\mathbf{D}}^{-1}$ normalizasyon için kullanılmaktadır.

$$H^{(l+1)} = \sigma(\hat{D}^{-1}\hat{A}H^{(l)}W^{l}) \tag{2.20}$$

Şekil 2.35'de Â matrisi üzerinden $\hat{\mathbf{D}}$ ve $\hat{\mathbf{D}}^{-1}$ matrislerinin hesaplanması gösterilmiştir [24]. Her düğümden çıkan kenar sayısı hesaplanarak $\hat{\mathbf{D}}$ matrisi oluşturulmaktadır. Burada önemli olan noktalardan birisi kendine döngü (self-loop) bulunan düğümler için derece matrisindeki eleman 2 sefer artırılmaktadır. Bu işlemler sonunda $\hat{\mathbf{D}}$ matrisinin tersi alınmaktadır. Bu matris bir köşegen (diagonal) matris olduğundan dolayı tersi basit bir şekilde hesaplanmaktadır.



Şekil 2.35: GCN derece matrisi ve tersinin hesaplanması.

GCN modelleri de diğer modeller gibi ağırlık matrislerini (W) öğrenmeye çalışmaktadır. Bu şekilde bir çizge ve öznitelik matrisleri ilişkilendirilerek sınıflandırma ya da regresyon problemi çözümlendirilmektedir.

2.6 Kolektif Öğrenme

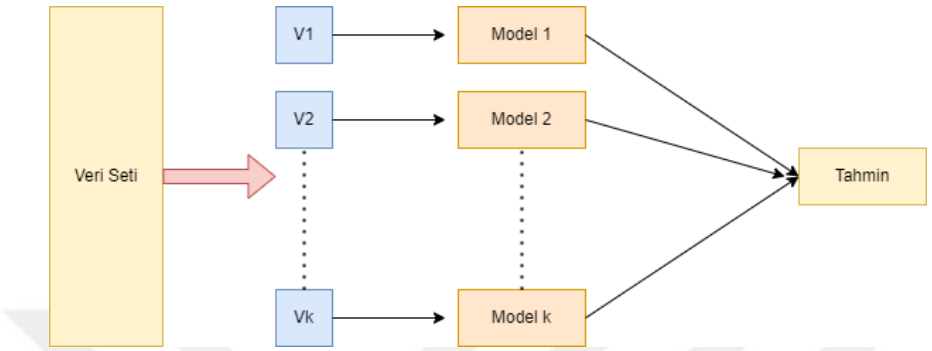
Kolektif öğrenme (Ensemble Learning, EL) zayıf öğrenicilerin (weak learner) birleştirilmesiyle elde edilen öğrenme yöntemidir. ML modellerinin gerçek hayatta belirli kapasiteleri bulunmaktadır ve bu modeller her problemde yüksek başarı getiremeyebilir. Bu modeller bir araya getirilirse daha iyi başarım elde edilebilir. Modellerin kombinasyonları genellikle iki amaçla yapılmaktadır. Bunlardan ilki modellerin birleşimiyle hataları azaltmaktır diğeri ise modellerin problemi genelleştirerek çözümlemesidir.

Kolektif öğrenme genellikle zayıf öğrenicilerin verdiği kararların birleştirilmesiyle oluşmaktadır. Kolektif öğrenmenin türleri ilerleyen bölümde verilmiştir.

2.6.1 Torbalama

Torbalama (bagging) yönteminde bir veri seti K adet alt setlere ayrıldıktan sonra her bir alt veri seti için bir model eğitilmektedir. Daha sonra bu modeller birleştirilerek kolektif öğrenme tahmini üretilmektedir. Şekil 2.36'da torbalama yöntemi şekil üzerinde gösterilmiştir. Burada K adet aynı model K farklı alt veri seti ile eğitildikten

sonra tahmin yapmaktadır. Bu yöntemde modellerin farklı şeyler öğrenebilmesi ve çeşitlilik sağlayabilmesi veri setlerinin farklı kısımlarının eğitilmesinden kaynaklanmaktadır. Modelin en son kısmında eğer bir sınıflandırma problemi çözülüyorsa genellikle topluluk oylaması (majority voting) yapılır. Bir regresyon problemi çözümünde ise tahminlerin ortalaması alınmaktadır.



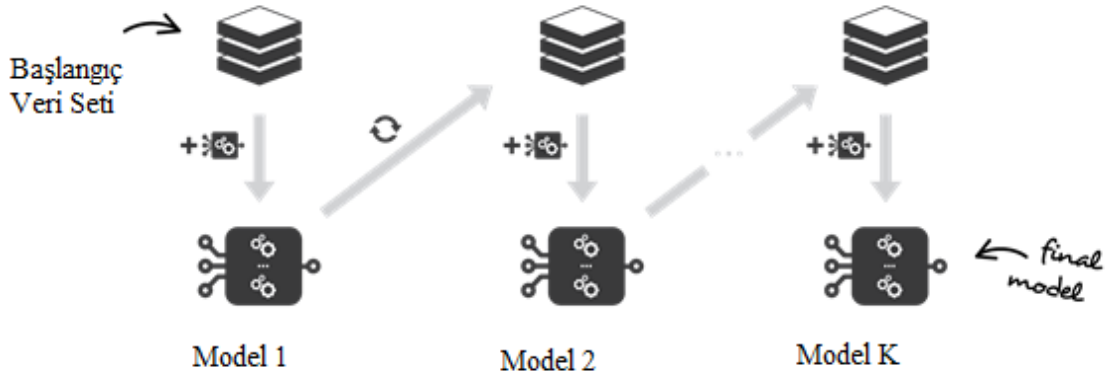
Şekil 2.36: Torbalama yöntemi.

Popüler torbalama yöntemleri şunlardır:

- Bootstrapping
- Rastgele Orman (Random forest)

2.6.2 Yükseltme

Yükseltme (Boosting) yöntemi başka bir kolektif öğrenme yöntemidir. Bu yöntem zayıf öğreniciler ile bir problemi öğrenmeye çalışarak modelin sapma (variance) hatasını düşük tutmayı hedeflemektedir. Bu yöntem ilk başta zayıf bir öğrencinin veriyi öğrenmesi ile başlar. Daha sonra sıralı bir şekilde öğrenilen bölgenin üzerinden başka modeller eğitilerek yöntem devam ettirilmektedir. Şekil 2.37 yükseltme yönteminin çalışma şeklini göstermektedir [25].



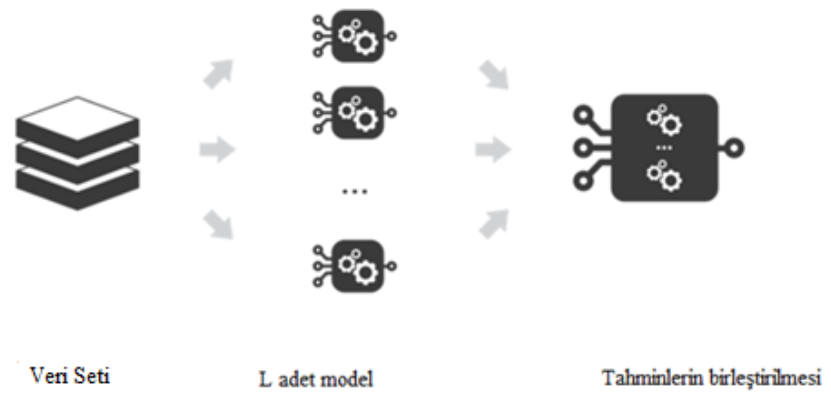
Şekil 2.37: Yükseltme yöntemi.

Popüler yükseltme yöntemleri şunlardır:

- Adaboost
- Gradyan yükseltme (Gradient boosting)
- XGBoost

2.6.3 Yığma

Yığma (Stacking) yöntemi bir veri setinin K farklı model ile öğrenilmesini ve tahminlerin diğer yöntemlerdeki gibi birleştirilmesini amaçlayan yöntemdir. Yığma yöntemi torbalama yöntemine benzer bir yapıdadır fakat bu yöntem bir veri setini K farklı model ile eğitirken torbalama yönteminde ise K farklı veri setinde K adet aynı model eğitilmektedir. Bu da çeşitliliği sağlayan farklı bir yöntem olduğunu göstermektedir. Şekil 2.38'de yığma yöntemi gösterilmiştir [25].



Şekil 2.38: Yığma yöntemi.

3. LİTERATÜR TARAMASI

Bu bölümde literatür taraması yapılarak finans üzerinde kullanılan temel yöntemler, veri setleri, zaman serisi ve çizge tabanlı yöntemler açıklanmıştır. Bu tezin kapsamında zaman serisi olarak finans ve borsa ele alınmıştır ve bundan dolayı zaman serilerinin daha çok borsa üstünde olan kısımları araştırılmıştır. Literatürde yapılan çalışmalar alt bölümlerde açıklanmaktadır.

3.1 Veri Setleri

Bu bölümde literatürde en çok kullanılan finansal veri setleri araştırılmıştır. Literatürde çok sayıda finansal veri bulunmaktadır. Bunlardan en popüler olan ve çalışmalarda kullanılan bazı veri setleri verilmiştir. Bunlar:

- DOW30
- S&P500
- FTSE100
- Kripto para (Bitcoin vb)
- Borsa İstanbul 100

Bu veriler farklı çözünürlükte değerlendirilebilir. Örnek olarak dakikalık, saatlik, günlük, analizlere tabi tutulabilir.

3.2 Çizge Ağları

Çizge ağları genel olarak birçok farklı problemde kullanılmaktadır. Fizik, kimya, biyoloji, çizge üretme, öneri sistemleri, borsa, resim sınıflandırma bunlardan bazılarıdır [26–32].

Çizge ağları modellemeleri genellikle 2 yöntemle yapılmaktadır. Bunlardan ilkinde düğüm bazlı analizler ve modellemeler yapılmaktadır. Örnek olarak düğüm

sınıflandırma ya da kenarlar arasındaki bağlantıların tahmini verilebilir. Diğer modelleme yönteminde ise bir çizgenin türü tahmin edilmektedir [33].

Çizge ağları zaman serilerinde ve uzay zamansal problemlerde de kullanılmaktadır. Trafiğin zamansal olarak karakterini modelleyen bir çalışmada bir şehrin yollarına yerleştirilen sensörlerden alınan ortalama hıza göre tahmin yapılmaktadır. Bu sensörlerden gelen veriler bir çizge olarak gösterildikten sonra farklı çizge modelleri ile eğitilmektedir. Bu çalışma belirli bir zaman dilimi sonrasında trafiğin ortalama akış hızını modellemiştir [34].

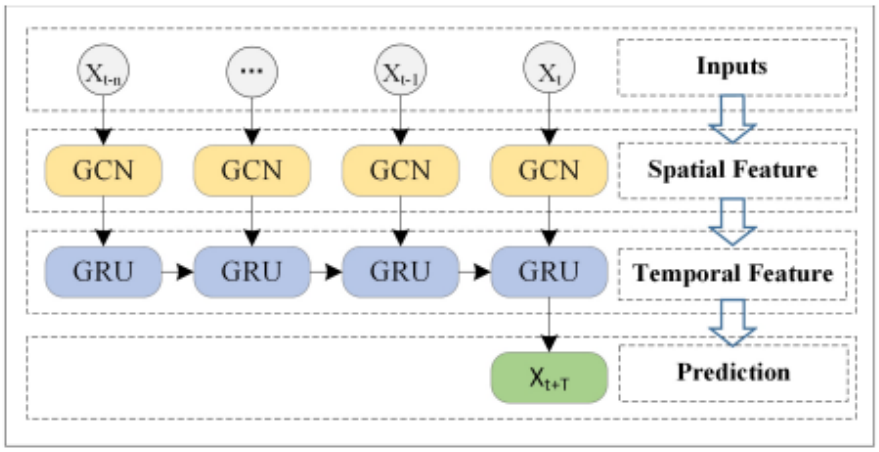
Yapılan bir başka çalışmada çok değişkenli zaman serileri çizge ağları modelleri kullanılarak modellenmiş ve temel bazı modeller ile kıyaslanmıştır. Bu çalışmada GNN modelinin DCRNN, STGCN, Graph Wavenet gibi modellerden daha başarılı sonuçlar verdiği gözlemlenmiştir [35].

Literatürde çizge ağlarını derin öğrenme yöntemleri ile birleştiren ve modelleyen birçok yöntem bulunmaktadır. Bu derin çizge ağları modelleri literatürde birçok farklı alt türe ayrılabilir[33]. Bunlardan temel bazıları şunlardır:

- Çizge Özyinelemeli Sinir ağları (Graph Recurrent Neural Networks, GRNN)
- Çizge evrimsel ağları (Graph Convolutional Networks, GCN)
- Çizge Özkodlayıcılar (Graph Autoencoders, GA)
- Çizge Pekiştirmeli Öğrenme (Graph Reinforcement Learning, GRL)

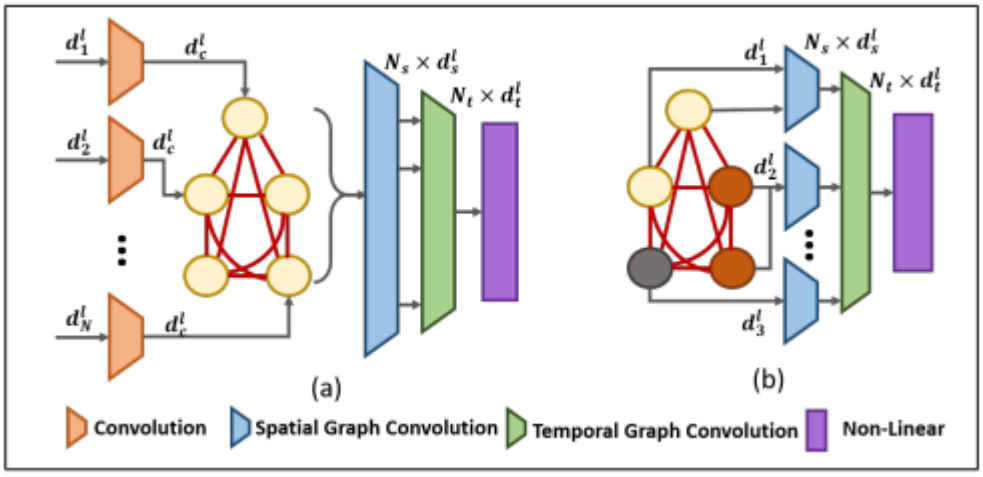
Her bir çizge ağı farklı problemlerde kullanılmaktadır. Bunların içerisinde en yaygın olan GCN modelidir. GCN modeli çizgeyi bir konvolüsyon işleminden geçirdiğinden dolayı birçok problemde başarılı sonuçlar vermektedir [33], [36].

GCN modeli uzay zamansal seriler problemlerinde de kullanılmaktadır. Bu problemlerin bir türü olarak trafiğin akış hızının ya da belirli karakteristik özelliklerinin belirlenmesi örnek olarak verilebilir. Yapılan bir çalışmada GCN modeli GRU hücreleri ile birleştirilerek T-GCN modeli tasarlanmıştır. T-GCN modeli hem uzaysal hem de zamansal çözümlemeler yaparak trafiğin akış hızını modellemiştir. SZ-Taxi, Los-Loop veri setleri üzerinde çalıştırılan bu model ARIMA, SVR, GCN, GRU modellerinden daha başarılı sonuçlar vermiştir [37]. Şekil 3.1 T-GCN modelini göstermektedir.



Şekil 3.1: T-GCN modeli trafiğin akış hızının modellenmesi.

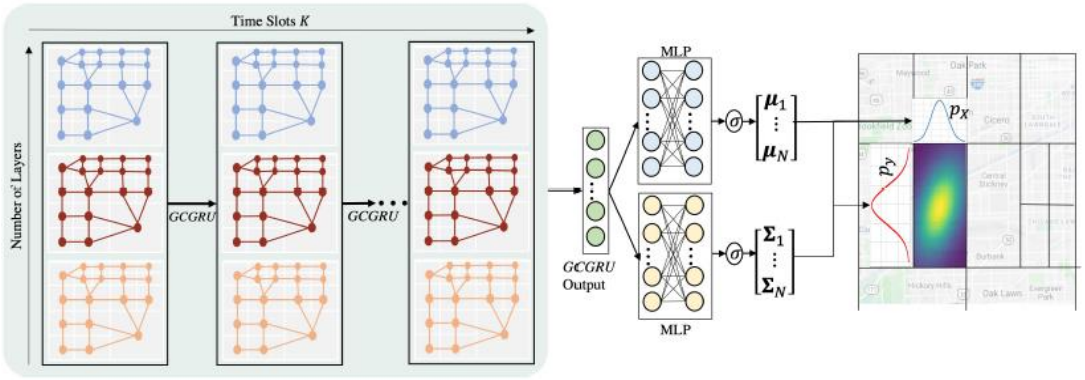
Başka yapılan bir çalışmada GCN insanın hareketlerini uzay zamansal bir problemi çözecek şekilde modellemektedir [38]. Bu yapılan çalışmada insanların yaptığı hareketleri içeren videolar üzerinde uzay zamansal GCN (ST-GCN) modeli eğitilmiş ve S-RNN modeli ile performansı kıyaslanmıştır. GCN modeli S-RNN modelinden daha iyi sonuçlar vermektedir. Şekil 3.2 ST-GCN modelini göstermektedir.



Şekil 3.2: ST-GCN modeli ile insan hareketleri tespiti.

Çizge ağları kullanılarak suç tespiti de yapılmaktadır. Yapılan bir çalışmada bir şehir bölgelere ayrılarak o bölgede ne zaman bir suç gerçekleşeceği tahmin edilmeye çalışılmıştır. Bu suç tespiti yapılırken GCGRU (Graph Convolutional Gated Recurrent Unit) modeli kullanılmıştır. Şekil 3.3 GCGRU modelini göstermektedir.

Bu çalışmada bir şehir R adet bölgeye ayrıldıktan sonra her bölge bir düğüm olarak gösterilmiştir. Her bir bölge arasındaki uzaklık ise kenar olarak gösterilmiştir. Bu şekilde her bir bölgede gelecekte suç işlenip işlenmeyeceği tahmin edilmektedir. Bu çalışmada GCGRU modelinin performansı ConvLSTM, ARIMA, Rastgele Karar Ağaçları, SVR, GPR ile kıyaslanmıştır. Çizge tabanlı bu model diğer modellerden daha başarılı sonuçlar vermektedir [39].



Şekil 3.3: GCGRU Modeli ile Chicago suç veri setinin eğitimi.

GCN modelleri sosyal medyada söylenti tahmin etme [40], bisiklet yollarındaki trafik akışı [41], kara para aklamanın önlenmesi [42], bir video içerisindeki objelerin ilişkilerinin tahmin edilmesi [43], trafiğin akış hızını tahmin etme gibi konularda kullanılmaktadır ve başarılı sonuçlar vermektedir.

Çizge ağları çok sayıda problemin analizinde ve modellenmesinde kullanılmaktadır. Bundan dolayı bu tez kapsamında bazı problemler örnek olarak verilmiştir ve açıklanmıştır. Bu tez içerisinde yapılan çalışmada borsa bir problem türü olarak belirlenmiş ve kullanılmıştır. Çizge ağları ile borsada yapılan çalışmalar ilerleyen bölümlerde anlatılacaktır.

3.3 Borsada Kullanılan ML Yöntemleri

Finansal zaman serileri ve borsa tahmini yapılırken kullanılan alanlardan bir tanesi de makine öğrenmesidir. Bu alanda yapılan birçok farklı çalışma bulunmaktadır. Bu bölümde literatürde borsa üzerinde yapılan ML çalışmalarından bahsedilecektir.

ANN modelleri borsa üzerinde hisse senedi tahminlerinde kullanılmakta olan yöntemlerdir. Bu ANN modellerinin detaylı incelemesi ve karşılaştırılması Bahrammirzaee tarafından açıklanmıştır [44].

D.Zhang 2004 yılında veri madenciliği yöntemleri, ANN, genetik algoritma (Genetic Algorithm, GA) ve kural tabanlı tahmin etme yöntemlerinin literatürde nasıl kullanıldıklarını incelediği detaylı bir çalışma yapmıştır [45].

Mochn literatürde kullanılan bulanık mantık, olasılıksal ve NN tabanlı modeller ile yapılan finansal tabanlı uygulamaları incelemiştir [46]. LeBaron ajan tabanlı (agent based) finansal hesaplamalar üstüne bir kitap bölümü yazmıştır [47]. Chalup finans uygulamaları anlattığı kitabın bir bölümünde çekirdek (kernel) yöntemi, temel bileşenler analizini (Principal Component Analysis, PCA) ve destek vektör makinesi (Support Vector Machine, SVM) gibi yöntemleri anlatmıştır [48].

SVM ve SVR modelleri borsa tahmini yapılırken kullanılan klasik ML yöntemlerinden bir tanesidir. SVM ve SVR yöntemini farklı şekillerde kullanan çalışmalar bulunmaktadır [49]. Yapılan bir çalışmada teknik olarak kullanılabilecek olan bazı indikatörler ile SVM ve SVR modeli eğitilmiştir [50]. Bu çalışmada tasarlanan SVR modeli temel modellerden daha iyi çalışmaktadır.

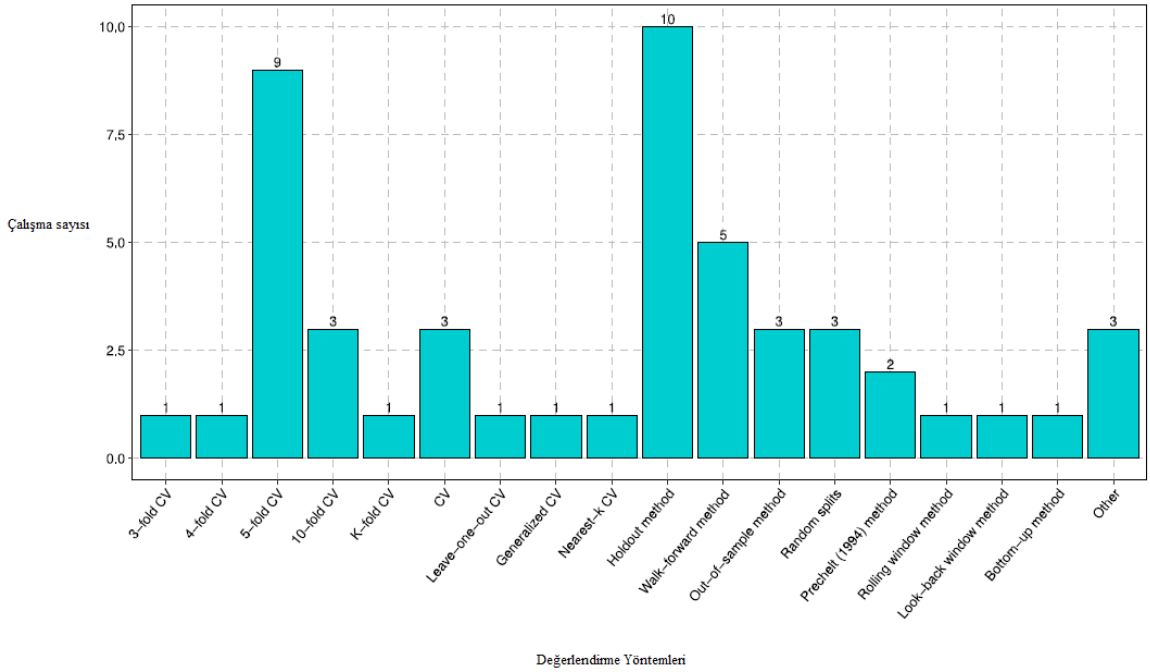
Bulanık mantık borsa tahminleri yaparken kullanılan bir başka yöntemdir [49]. Bulanık mantık kullanarak tahmin yapan çok sayıda çalışma bulunmaktadır. Her çalışma farklı bir yöntem ile bulanık mantığı birleştirdikten sonra tahmin yapmaktadır [49].

Genetik algoritma (GA) borsada kullanılmakta olan optimizasyon yöntemlerinden bir tanesidir. Yapılan bir çalışmada GA kullanılarak portfolyo oluşturulmaktadır. Bu portfolyo 5 farklı hisse senedine yatırım yaparak oluşacak olan riski minimize etmektedir. Her bir hisse senedine yapılacak olan yatırımın ağırlıkları GA ile bulunmuştur [51].

Wavelet, KNN, kümeleme yöntemi, PCA, Karar Ağaçları, Rastgele Karar Ağaçları, XGboost borsada yaygınca kullanılmakta olan temel ML modelleridir [52–58].

Bu değerlendirme yöntemlerinin literatürdeki dağılımları

Şekil 3.4 üzerinde gösterilmiştir. Literatürde en çok kullanılan değerlendirme yöntemleri holdout ve k-fold değerlendirme yöntemleridir.



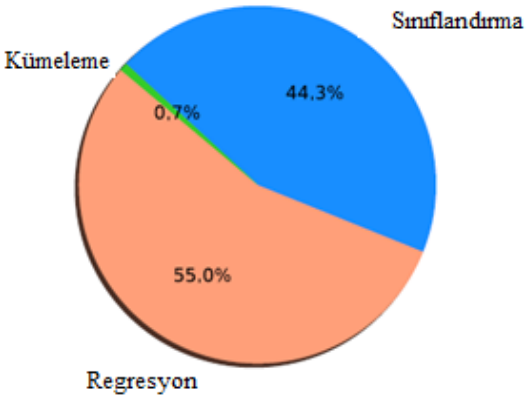
Şekil 3.4: ML modellerinin borsa üzerindeki değerlendirme yöntemlerinin literatürdeki dağılımları.

Borsada ML modelleri eğitilirken ve değerlendirilirken kullanılan veri ayırma yöntemleri şunlardır [59]:

- K-fold CV
- Leave-one-out CV
- Generalized CV
- Nearest-k CV
- Hold-out metodu
- Kayan pencere yaklaşımı (Rolling Window Method)
- Rastgele ayırma (Random Split)

Şekil 3.5 borsa çalışmalarının öğrenme yöntemlerine göre dağılımlarını göstermektedir. Literatürde çalışılan borsa tahmin modellerinin %55'e yakın bir kısmı regresyon, %44,3 gibi bir kısmı sınıflandırma, yaklaşık %0,7 kümeleme yapmaktadır. Borsa tahmin modelleri geliştirilirken farklı çözünürlükte veriler kullanılmaktadır [59]. Genellikle 1 dakikalık, 5 dakikalık, 1 saatlik, günlük, haftalık,

aylık, çeyrek yıllık ve yıllık tahminler yapan modeller geliştirilmektedir. Literatürde en çok günlük veri ve aylık veri üzerinden tahmin yapılmaktadır [59].



Şekil 3.5: Literatürde yapılan borsa çalışmalarının öğrenme yöntemlerine göre dağılımları.

3.4 Borsada Kullanılan Derin Öğrenme Modelleri

Bilimin ve bilgisayar gücünün ilerlemesi ile birlikte derin öğrenme modelleri geliştirilmiş ve zaman serisi alanında uygulanmaya başlanmıştır. Bu çalışma kapsamında finans ve borsada kullanılan derin öğrenme modelleri literatüründen bahsedilecektir.

Literatürde birçok algoritma borsadaki hisse senetlerinin gelecekteki değerini tahmin etmeye çalışmaktadır. Bir hisse senedinin gelecekteki değeri tahmin edilerek yatırım yapma stratejisi ortaya çıkarılabilir. Bunun için DL literatüründe en çok kullanılan yöntemlerden bir tanesi ise LSTM'dir [60]. Marketin yapısını gösteren bazı indikatörler kullanılarak RNN modeli eğitilmektedir [61]. Bir diğer çalışmada ise teknik bazı indikatörler birleştirilerek Wavelet dönüşümü (Wavelet Transform), LSTM ve Özkodlayıcı'dan (Autoencoder) geçirilmiştir [62]. CNN ve LSTM modeli birlikte kullanılarak bir model geliştirilmiştir. Bu modelde CNN hisse senedi seçimini yaparken LSTM ise hisse senedinin değerini tahmin etmektedir [63].

Literatürde borsa tahmini yapılırken kullanılan bir diğer yöntem ise bir borsanın indeks değerinin gelecekteki değerini tahmin etmektir. S&P 500 indeksinin tahminin yapıldığı bir çalışmada LSTM modeli S&P 500 indeksinin değerini öğrenerek gelecekteki değerini tahmin etmeye çalışmaktadır [64]. Mourelatos LSTM, GA ve

SVR yöntemlerini Yunan Kıymetler Borsası İndeksi (Greek Stock Exchange Index) tahmini yapmak için kullanmıştır [65]. Yong derin çok katmanlı perceptron (Deep Multilayer Perceptron) modelini ve hisse senedi indeksinin açılış, kapanış, en düşük ve en yüksek değerlerini kullanarak Singapore Hisse Senedi indeks değerini tahmin etmiştir [66].

Literatürde bulunan bir diğer borsa tahmini yapma yöntemi ise sınıflandırma problemi çözmektir. DL modelleri ile zaman serisinin hangi anında satın alma (buy), satma (sell) ve tutma (hold) işlemlerini yapacağına karar verilebilir. Bir çalışma göreceli güç indeksini (Relative Strength Index) genetik algoritma kullanarak optimize ettikten sonra DMLP metodu kullanarak bir eğitim işlemi gerçekleştirilmiştir [67]. Sirignano insanların belirlediği limit emirleri kitaplarının akışını (limit order book flow) kullanarak LSTM modeli eğitmiştir. Bu LSTM modeli hisse senetlerinin hareketlerini tahmin etmektedir [68]. Tsantekidis ‘ın yaptığı çalışma da limit emirleri verisini kullanarak LSTM modeli ile trend tahmini yapmaktadır [69].

Borsayı sınıflandırma problemi olarak düşünme ve çözme konusunda en çok kullanılan yöntemlerden bir tanesi CNN modelidir. CNN modelleri 2 boyutlu problemlerde daha çok kullanılmaktadır. Bundan dolayı borsadaki hisse senetlerinin farklı şekilde CNN modellere verilmesi gerekmektedir. Bir çalışmada hisse senedinin değeri 2 boyutlu bir resme dönüştürülmüş ve CNN modeli ile eğitilip sonuç alınmıştır [70]. Bir diğer çalışmada hisse senetleri için candlestick akış diyagramları kullanılarak 2 boyutlu resimler elde edilmiştir. Daha sonrasında konvolüsyonel AE yöntemi bu resimleri kullanarak bir fon oluşturmak için eğitilmiştir [71]. Tsantekidis yaptığı bir çalışmada limit emirleri kitabının son 100 kaydını kullanarak 2 boyutlu bir CNN eğiterek hisse senedi değeri tahmini yapmaktadır [72]. Başka bir çalışmada birbirleriyle korelasyona sahip öznitelikler bir araya getirilerek CNN modeline verilmiştir [73]. Literatürde klasik derin öğrenme metotlarını kullanan başka çalışmalar da bulunmaktadır ve bu çalışmalar daha detaylı bir şekilde Özbayoğlu’nun incelemesinde bahsedilmektedir [60].

Borsa ve hisse senedi tahminindeki önemli olan noktalardan bir tanesi de değerlendirme metrikleridir. Problemin türüne göre çok farklı şekilde değerlendirme kriterleri bulunabilir. Bunlardan bazıları şunlardır:

- MSE
- RMSE(Root Mean Square Error)
- R-square
- Doğruluk
- Kesinlik
- Hassasiyet
- AUC-ROC eğrisi
- F1 skor
- Birikimli kazanç(Cumulative gain)
- Toplam kazanç, ortalama kazanç

Bu yöntemlerin çoğu modelin performansını ya da doğru alım-satım noktalarının doğruluğunu ölçmektedir. Burada önemli olan bir nokta bu skorlar ne kadar yüksek olursa olsun en son elde edilen kazanç finansal yatırımlar için ciddi önem taşımaktadır.

3.5 Borsada Kullanılan Çizge Tabanlı Öğrenme Modelleri

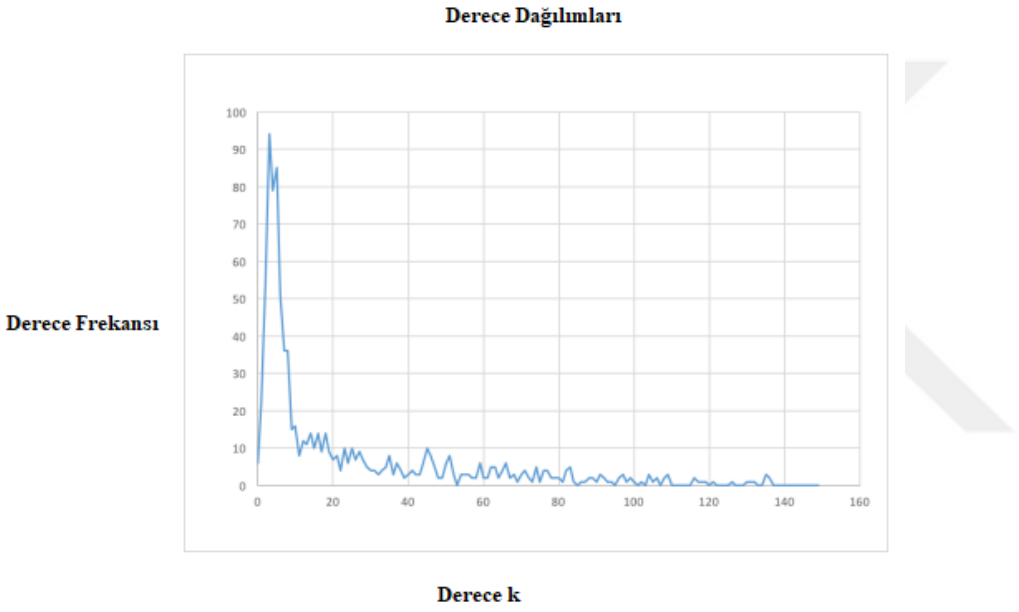
Çizge tabanlı öğrenme yöntemleri günümüzde veri sayısının artmasıyla birlikte yavaş yavaş popülerlik kazanmaya başlamıştır. Çizge tabanlı öğrenme yöntemleri herhangi bir zaman serisi ya da borsa problemini çözerken önemli avantajlar sağlamaktadır. Bu yöntemler birbirinden farklı hisse senetleri ya da zaman serilerinin zaman göre ilişkilerini bir çizge olarak ifade edebildiklerinden dolayı normal DL metodlarından daha iyi çalışması beklenmektedir.

Literatürde çizge gösterimi ile borsadaki hisse senetlerini analiz eden yaklaşımlar bulunmaktadır [74–81]. Bu yaklaşımlar hisse senetlerini çizge olarak oluşturmakta ve basitçe analiz etmektedir. Literatürde çizgelerin SVM, MLP, CNN ve LSTM ile çözülmeye çalışıldığı bazı çalışmalar da bulunmaktadır [82–84]. GCN modelini borsa üzerinde çizge yapısı ile birlikte kullanan bazı çalışmalar bulunmaktadır [85–[90] .

Bu tez iki önemli çalışmadan faydalanılarak yapılmıştır. İlk çalışmada S&P 500 veri seti üzerindeki farklı sektörlerde bulunan hisse senetleri bir çizgeye dönüştürülerek en güvenilir hisse senedi bulunmaya çalışılmıştır [49]. Bu çalışmada Spearman korelasyon katsayısı kullanılarak yönsüz çizge oluşturulmuştur. Bu çizgenin her bir

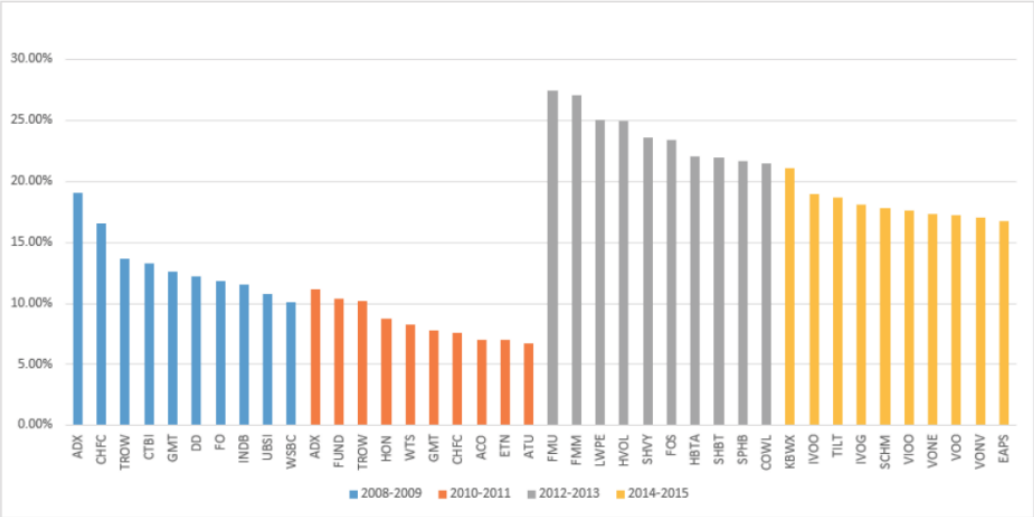
düğümünün dereceleri hesaplandıktan sonra en merkezcil düğümler bulunmuş ve bunlar en güvenilir hisse senetleri ve sektörler olarak belirlenmiştir. Bu çalışmanın gelecekte yapmayı önerdiği bir yöntem ise hisse senetlerinin yönlü çizge olarak tasarlanması ve bu şekilde bir analiz yapılmasıdır.

Şekil 3.6 ve Şekil 3.7 çizgeler üzerinde gerçekleştirilen analizin sonuçlarıdır [91]. Burada yüksek dereceye sahip düğüm sayısı az bulunmaktadır. Bunun sebebi bir çizge üzerinde güvenilir ve merkezcil olan düğüm sayısının az olmasından kaynaklanmaktadır. Yani bir çizge üzerinde belirli düğümler daha önemli olacak ve borsayı etkileyecektir.



Şekil 3.6: Çizgelerin derece dağılımları.

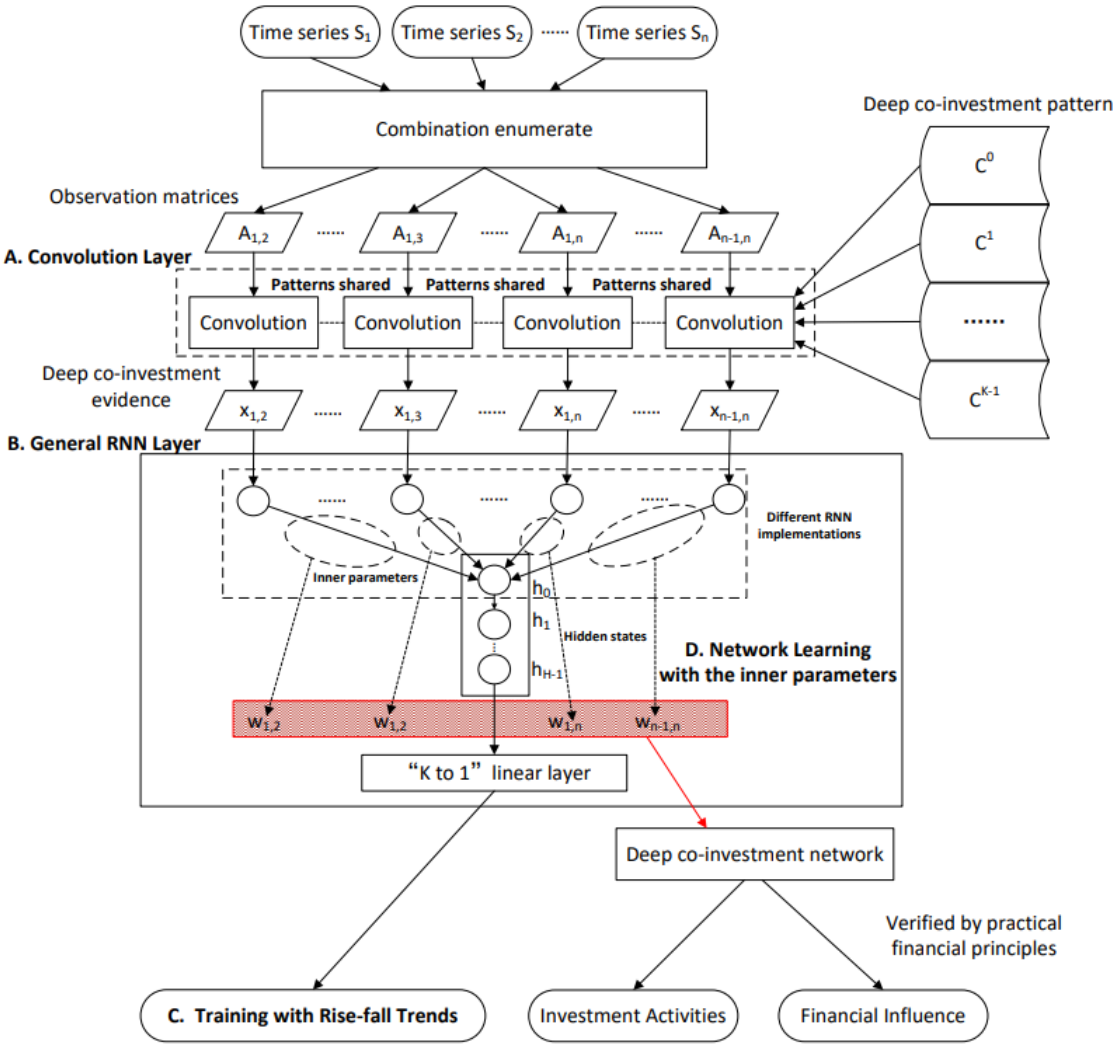
Bu tez kapsamında önemli bulunan ve çalışmanın tasarlanmasında esinlenilen diğer makale de ise derin öğrenme yöntemi kullanılarak birden çok hisseden en güvenilir ve kazandırabilecek olan hisseler tahmin edilmektedir. Bu çalışmada zaman serilerinin arasındaki ilişkiyi çıkaracak olan filtreler eğitilmeye çalışılmaktadır. Bu filtreler genel korelasyon gibi bazı yöntemler ile hesaplanırken bu çalışmada bunları öğrenen bir derin öğrenme yöntemi geliştirilmiştir [92].



Şekil 3.7: Çizgelerde zamana göre en merkezcil düğümleri grafiği.

Bu çalışmadaki asıl varsayım ise birlikte değişen gizli örüntülerin (co-varying latent patterns) tespit edilmesi olarak düşünülmüştür [92]. Eğer 2 hisse senedi aynı grup insan tarafından alınıyorsa onların öznitelikleri birbirleriyle benzer olacaktır. Deep co-investment network (DeepCNL) bu örüntüleri tahmin edebilmek amacıyla $C = \{C^0, C^1 \ldots C^K\}$ şeklinde K adet filtre tasarlamıştır. Bu şekilde benzer yatırımlar yapan insanların bulundukları hisse senetleri ortaya çıkacaktır.

DeepCNL modeli borsada bulunan her bir hisse senedi S_i ve S_j çiftini birleştirerek bir matris A_{ij} oluşturmaktadır. Bu A_{ij} matrisi $C = \{C^0, C^1 \ldots C^K\}$ filtreleri ile konvolüsyon işleminden geçirildikten sonra $x_{ij}(t)$ vektörü olarak tanımlanmaktadır. Bu vektör borsa indeksinin değerini öğrenmesi amacıyla bir RNN ve lineer katmandan geçirilmektedir. RNN katmanın çıktısında bulunan gizli durum vektörü bir lineer katmanda geçirildikten sonra borsa indeksinin trend tahmini yapmaktadır. Bu model eğitilirken RNN katmanında her durum için bir ağırlık atanmaktadır. Bu atanan ağırlıklar bir çizgenin kenarları olarak düşünülerek bir çizge ortaya çıkarılıyor. Bu çizgenin analizleri sonucunda en çok dereceye sahip olan hisse senetleri borsa indeksini en çok etkileyen hisse senetleri olarak tahmin ediliyor. Bu model daha sonrasında oluşturulan bazı temel çizge yapıları ile kıyaslanıyor. Bu çalışmada en son olarak DeepCNL modeli üzerinden belirlenen hisse senetlerinin temel modellerin belirlediği hisse senetlerinden daha etkili olduğu gözlemlenmektedir. Şekil 3.8 DeepCNL modelini göstermektedir.



Şekil 3.8: DeepCNL modelinin yapısı.

Bu tez kapsamında literatürde belirlenen bu yöntemlerden faydalanarak bir çizge yönteminin ortaya çıkarılması ve borsa üzerinde derin öğrenme ile zaman serisi analizi yapılarak stratejiler oluşturulması planlanmıştır.

4. YÖNTEM

Bu tezin ana amacı zaman serileri üzerinde çizge tabanlı öğrenme modelleri çalıştırmak ve bu modellerin klasik DL ve ML modellerinden daha iyi çalıştığını göstermektir. Bu kapsamda bir zaman serisi olan hisse senetleri ve borsa üzerinde bir modelleme yapılmıştır.

Bu araştırmada borsa üzerindeki hisse senetleri nasıl bir çizgeye dönüştürülür ve bu çizgeler nasıl modellenir bu gösterilmiştir. Bu tez çalışmasında farklı çizge oluşturma yöntemleri kullanılmış ve çizge tabanlı öğrenme yöntemleriyle birleştirilerek eğitilmiştir. Bu çizge tabanlı öğrenme yöntemleri önceki çalışmaların üstüne çıkılarak yapılmıştır ve yeni bir model ortaya çıkarılmıştır. Ayrıca oluşturulan bu çizge tabanlı modeller temel bazı modeller ve açgözlü algoritmalar ile karşılaştırılmaktadır. Bu eğitilen modeller kolektif öğrenme modelleri ile eğitilerek iyileştirilmiştir.

Borsa üzerinde çok sayıda araştırma yapılmakla birlikte her bir çalışma farklı veri setlerinin farklı zamanlarında yapılmıştır. Bundan dolayı bu çalışmada kendi içerisinde bazı varsayımlar üzerinden ilerlemektedir. Bu tasarlanan model ve problemin detayları alt bölümlerde açıklanacaktır.

4.1 Problem Tanımı

Bir zaman serisi olan hisse senetlerinin değerleri zaman içerisinde belirli faktörlere bağlı olarak değişmektedir. Bu faktörleri bulmak zaman zaman çok zor olabilir çünkü hisse senetleri çok fazla sayıda parametreden etkilenmektedir. Örnek olarak bir şirketin güvenilirliği zaman içerisinde sürekli değişebilir ya da rekabet ettiği firmaların yaptığı hamleler de o şirketin hisse senedi değerini değiştirebilir. Borsadaki bu zaman serileri için olası tüm parametreleri gerçek hayatta bulmak çok zordur. Bundan dolayı borsadaki hisse senetlerinin değerleri bazı varsayımlar altında hesaplanmaktadır. Bunlardan en önemlisi borsadaki hisse senetlerinin değerlerinin geçmişe bağlı bir davranış göstermesi olacaktır. Örneğin artış eğilimindeki bir hisse senedinin gelecekteki davranışı benzer olacaktır ya da bunun gibi çok

gözlemlenemeyen örüntülerin olduğu düşünülmektedir. Bir diğer varsayım ise hisse senetlerinin birbirlerine olan etkileridir. Örnek olarak A hissenin artışı B hissenin gelecekte artmasına sebep olabilir ya da ters yönde bir etkiye de sebep olabilir.

Bu tezde belirlenen varsayımlar altında bir borsadaki farklı hisse senetleri üzerinde kazanç getiren algoritmalar ve stratejiler geliştirilmesi amaçlanmıştır. Bu tez kapsamında oluşturulacak olan ana yöntem çizge zaman serisinin işlenerek kazanç getiren bir algoritma tasarlanmasıdır. Çizge zaman serilerinin derin öğrenme ile çözümlenmesi ve bu yöntemin genelleştirilerek gelecekteki çalışmalara ön bilgi olarak sunulması amaçlanmaktadır.

Bu amaç gerçekleştirilirken uygulanacak olan yöntemler ilerleyen bölümlerde anlatılacaktır.

4.2 Veri Setinin Oluşturulması

Bu tezdeki algoritmaların eğitileceği ve test edileceği veri seti olarak DOW 30 seçilmiştir. Bu veri seti finance.yahoo.com üzerinden elde edilmiştir. Burada 2019 yılı için belirlenmiş olan 30 firma için bir veri seti oluşturulmuştur. Bu veri seti 2010 yılından başlayarak 2020 yılına kadar günlük kapanış değerleri üzerinden oluşturulmuştur. Bu veride yalnızca borsanın açık olduğu günler bulunmaktadır çünkü borsa kapalıyken bir işlem yapılamamaktadır. Bu veri setinin ilk 2 yılı eğitim için kullanıldığından toplamda 8 yıllık hisse senedi verisi için tüm modellerin test edilmesi planlanmıştır.

4.3 Problem Değerlendirme Ölçütlerinin ve Çözüm Hedeflerinin Belirlenmesi

Bu tezin kapsamında DOW 30 veri seti için bir simülasyon oluşturulacaktır. Bu simülasyona göre 2012'den 2019 yılının sonuna kadar toplam 8 yıllık bir yatırım strateji oluşturulacaktır. Bu yatırım stratejisi çizge zaman serisini öğrenen derin öğrenme modellerinin sonuçlarından çıkarılacaktır. Bunun yanı sıra bazı temel modeller ve stratejiler asıl amaçlanan modelin başarısını ölçmek için kullanılacaktır. Bu problemin temelinde bu 8 yıl boyunca ortalamada iyi kazanç elde etmek bulunmaktadır. İyi kazancı tanımlamak için ise bazı temel stratejiler bulunmaktadır. Bunlar:

- DOW30'un ortalama kazancı

- Basit sezgisel yöntemler(Örnek olarak bir önceki yıl artan hisse senedine yatırım yapmak)
- Temel kabul edilebilecek derin öğrenme yöntemleri(MLP vb)

Bu belirlenen stratejiler en basit seviyede yapılabilecek stratejiler olduğundan dolayı tasarlanacak olan modelin bu stratejilerden daha iyi çalışması beklenmektedir. Bunun haricinde bazı temel ölçütler tasarlanacak olan modellerin genel başarısını ölçmektedir. Bunlar:

- Doğruluk
- Kesinlik
- Hassasiyet
- PoS skoru

Bu problemde önemli olan bir diğer nokta ise bu 8 yıllık simülasyonun her bir yılına 50.000 dolarlık para miktarı ile girildiği varsayılmaktadır. Her bir alım satım işleminin ücreti 1 dolar olarak varsayılmıştır. Yani her yıl birbirinden ayrı bir simülasyon yapılmaktadır ve her birine 50.000 dolar para ile girilmektedir. Yıl sonunda elde edilen kazanç ise o yıla ait kazanç olmaktadır. Burada yıllık kazançlar ve toplamda 8 yılında sonunda elde edilen ortalama kazanç problemi değerlendirilirken kullanılacak olan asıl ölçüt olacaktır.

Bu problemin basitleştirilmesi amacıyla yapılacak olan işlemler birer günlük olacaktır. Yani t-1 anında al komutu ile alınan bir hisse senedi alındığından bir sonraki gün t anında satılacaktır. Dolayısıyla günlük alış ve satışlardan kazanç sağlanmaya çalışılacaktır. Burada herhangi bir tutma komutu bulunmamaktadır. Bu problemin analizini bir miktar kolaylaştırmak amacıyla ve günlük tahminin daha çok kazanç getirdiği bilindiğinden dolayı işlemler bu şekilde tasarlanmıştır. Bu varsayımın bir dezavantajı gürültü seviyesi yüksek bir veri üzerinde çalışılacak olmasıdır.

4.4 Ön işleme

ML literatüründe model eğitirken yapılması gereken önemli noktalardan bir tanesi ön işlemedir. Ön işleme için literatürde birçok yöntem olabilir ve bu yöntemler problemden probleme değişebilir. Bu bölümde DOW 30 hisse senetleri üzerinde yapılan ön işlemeler anlatılacaktır.

4.4.1 Normalizasyon

ML literatüründe modellerin eğitilebilmesi için düzgün bir normalizasyon yapılması gerekmektedir. Örneğin bu çalışmadaki DOW 30 hisse senedi değerleri zaman içerisinde değişmektedir. Verilerin bulunduğu aralık çok değişken olabileceği gibi her bir hisse senedi farklı bir sayı aralığında değer almaktadır. Bu problemlerin önüne geçebilmek için iki gün arasındaki artış oranı(değişim) tahmin edilmeye çalışılmıştır. Yani veri setinde etiket olarak hisse senetlerinin 2 gün arasındaki artış oranı kullanılmıştır. Denklem (4.1) bu artış oranın formülünü göstermektedir. Bu formülde X(t) bir hisse senedinin t anındaki değerini göstermektedir. Y(t) ise 2 gün arasındaki artış oranını göstermektedir.

$$Y(t+1) = \frac{X(t+1) - X(t)}{X(t)} \tag{4.1}$$

Bu tez kapsamında yapılan çalışmada bu normalizasyon kullanılarak model eğitilmiş ve strateji geliştirilmiştir. Bu normalizasyon hem modellerin problemi öğrenmesini kolaylaştırmıştır hem de veriler arasındaki ölçekleme sorununu çözmüştür.

4.4.2 Öznitelik çıkarımı

Bu çalışmada DL modelleri ve referans modeller eğitilirken bazı öznitelikler kullanılmıştır. Bu öznitelikler şunlardır:

- Son 5 gün gerçekleşen değişim miktarları(artış oranları)
- Son 5 gün gerçekleşen değişim miktarlarının minimum, ortalama ve maksimum değerleri
- Son 5 gün içerisinde gerçekleşen artış sayısı

Bu öznitelikler hisse senedinin yakın zamandaki artış azalış miktarları ve istatiksel olarak hissenin son zamanlardaki karakteristiğini inceleyerek bir tahmin yapmayı amaçlamaktadır. Bu şekilde gelecek hisse senedinin hangi yöne ne kadar değişeceğini tahmin edebilecektir. Bu özniteliklerin nasıl kullanıldığı gelecek bölümlerde anlatılacaktır.

4.5 Borsanın Çizgeye Dönüştürülmesi

Bu tez içerisinde yapılan çalışmalardan birisi borsa üzerindeki hisse senetlerinin birbirleriyle olan ilişkilerini bir çizge üzerinde göstermektir. Bu çizge her zaman dilimi *t* için ayrı ayrı oluşturulmaktadır. Bu tez kapsamında çizgenin boyutu 30x30 olacaktır ve bu çizge komşuluk matrisi kullanılarak gösterilmektedir. $G_t(i,j)$ bu çizgenin t anında i ve j numaralı hisse senetlerinin arasındaki kenarın ağırlık değerini göstermektedir. Bu çizgenin nasıl oluşturulduğu ilerleyen bölümlerde anlatılacaktır.

4.5.1 Benzerlik metrikleri

İki zaman serisinin belirli bir pencere boyutu(window size) içerisindeki benzerliklerini ölçecek olan bazı fonksiyonlar bulunmaktadır. Bu fonksiyonların ortak özelliği zaman serilerinin hareketleri veya birbirlerine benzerlikleri hakkında bilgi vermesidir. Bu tez içerisindeki benzerlik metrikleri belirli bir pencere boyutu olan N üzerinden incelenmiştir. Böylelikle zaman serisinin tamamı üzerinde işlem yapılmamış ve geçici kısımları üzerinden bir benzerlik değeri üretilmiştir. Bu değer her t anında zaman serisinin t'den t-N'e kadar olan parçası üzerinde uygulanmaktadır. Bu işlemin nasıl uygulandığı çizgelerin oluşturulduğu bölümde daha detaylı anlatılacaktır.

4.5.1.1 Pearson korelasyon

Denklem **(**2.1)'de Pearson korelasyonun nasıl hesaplandığı gösterilmiştir. Bu Pearson korelasyon formülü belirli bir pencere boyutu üzerinde uygulanarak $\rho_{x,y}$ değeri hesaplanmaktadır. Bu çalışma kapsamında iki farklı hisse senedinin belirli bölgeleri alınarak Pearson korelasyon katsayıları elde edilmektedir. Burada hisse senetlerinin alınan bölgeleri aynı zamanda diliminde olabileceği gibi farklı zaman dilimlerinde farklı pencere boyutunda da olabilir.

$$C(t_i, t_j) = \frac{\sum_{k=1}^{n}(X_{t_i-k} - \bar{x})(X_{t_j-k} - \bar{y})}{\sqrt{\sum_{k=1}^{n}(X_{t_i-k} - \bar{x})^2 \sum_{k=1}^{n}(X_{t_j-k} - \bar{y})^2}} \quad (4.2)$$

Denklem (4.2)'de iki zaman serisinin herhangi bir t_i ve t_j anları için N boyutundaki bir pencere boyutu için Pearson korelasyon katsayısının hesaplanması gösterilmiştir. C korelasyon matrisinin $C(t_i, t_j)$ elemanı i ve j indeksli hisse senetleri (X_{t_i}, X_{t_j}) için

t_i ve t_j anlarında hesaplanan katsayıyı göstermektedir. Bu değer burada doğrudan Pearson korelasyon formülüne eşitlenmektedir.

4.5.1.2 Spearman korelasyon

Spearman korelasyon fonksiyonu Pearson korelasyon formülüne benzemektedir fakat fonksiyona giren seriler bir R(X) fonksiyonundan geçirilmektedir. Bu fonksiyon verileri değerlerine göre bir sıralamadan geçirerek her birine bir derece vermektedir. Spearman korelasyon hesaplanırken bu adımdan sonrası Pearson korelasyon formülü ile birebir aynıdır.

$$C(t_i, t_j) = \rho_{R(x),R(y)} = \frac{cov(R(x),R(y))}{\sigma_{R(x)}\,\sigma_{R(y)}} \tag{4.3}$$

Denklem (4.3)'te Spearman korelasyonun hesaplanması gösterilmiştir. Bu formül basit ve sade olması amacıyla kısaltılmıştır. Bu fonksiyondaki R(x) i numaralı hisse senedinin t_i anındaki değeri ve geri kalan N boyutlu pencereyi gösterirken R(y) j numaralı hisse senedinin t_j anındaki değeri ve geri kalan N boyutlu pencereyi göstermektedir. Bu şekilde Pearson korelasyon katsayısı gibi Spearman korelasyon katsayısı hesaplanmaktadır. Spearman korelasyon Pearson korelasyondan farklı olarak monoton ilişkileri de ölçmektedir. Bundan dolayı Pearson korelasyon formülünde korelasyon değeri düşük olan değerler bu formülde daha yüksek çıkabilir. Lineer bir ilişkide iki değişken arasındaki değişim oransal iken monoton bir ilişkide ise sabit bir oranla artmak zorunda değildir. Bu şekilde lineer olmayan bir veri üzerinde de korelasyon katsayısı yüksek gelebilmektedir.

4.5.1.3 Öklid uzaklığı

İki zaman serisi arasındaki benzerlik hesaplanırken kullanılan yöntemlerden bir tanesi de Öklid uzaklığıdır. Denklem (4.4) iki hisse senedinin belirli bir pencere boyutu üzerinden Öklid uzaklıklarını ölçmektedir. Bu şekilde 2 zaman serisi arasındaki benzerlik hesaplanmaktadır.

$$D(X_{t_i}, X_{t_j}) = \sqrt{\sum_{k=1}^{N}(X_{t_i-k} - X_{t_j-k})^2} \tag{4.4}$$

Bu benzerlik uzaklık türünden olduğundan dolayı eğer iki zaman serisi birbirinden çok uzak ve farklı ise yüksek değer üretmektedir. Bu benzerliğin ters orantılı

olabilmesi ve normalize edilebilmesi amacıyla denklem (4.5) kullanılmıştır. Bu oluşan değer $C(t_i, t_j)$'ye eşit olmaktadır. Bu şekilde korelasyona benzer bir başka metrik oluşturulmuştur.

$$C(t_i, t_j) = \frac{\sqrt{D(X_{t_i}, X_{t_j})}}{40} \tag{4.5}$$

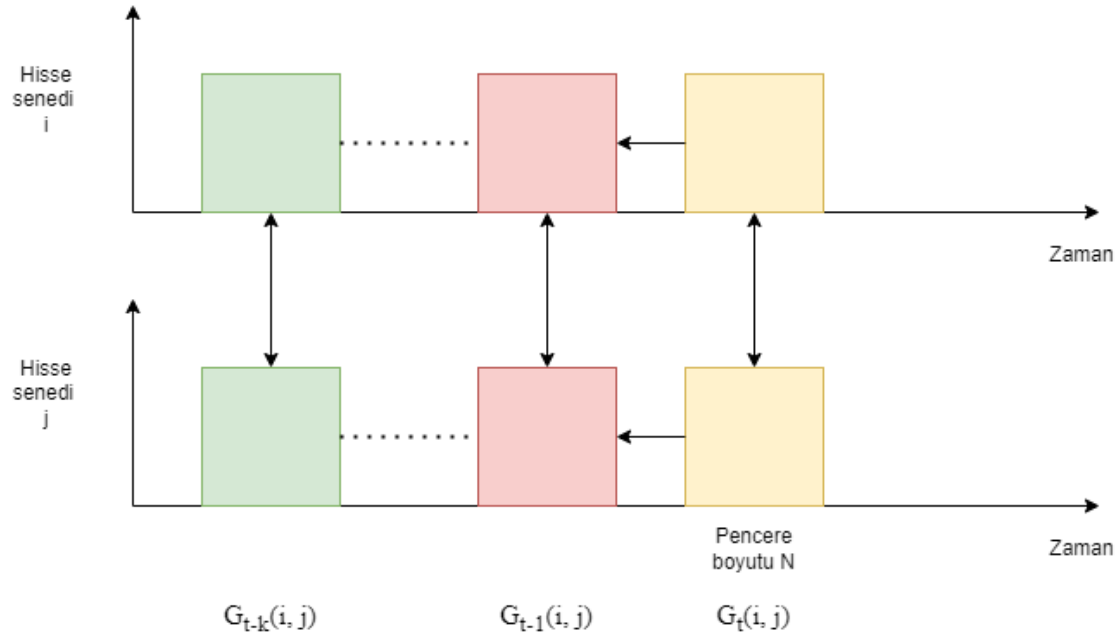
4.5.2 Yönsüz çizge

DOW 30 veri setinde 30 adet hisse bulunmaktadır. Bu hisselerin arasındaki ilişki her zaman dilimi t için boyutu 30x30 olan matris olarak gösterilebilir. G_t bu çizgenin matematiksel olarak gösterimini temsil etmektedir. $G_t(i, j)$ i ve j numaralı hisse senetleri arasındaki kenarın t anındaki değerini göstermektedir. Bu yönsüz çizgede bir kenar için şu eşitlik geçerli olmaktadır:

$$G_t(i, j) = C(t_i, t_j)$$

Şekil 4.1'de 2 hisse senedi arasında oluşan kenarın kayan pencere yaklaşımı ile hesaplanması gösterilmiştir. Bu pencerenin boyutu N olarak kabul edilmektedir. Bu yaklaşımda çizgenin bir kenarı hesaplanırken her zaman dilimi t anında iki karşılıklı pencere üst üste konarak ve bir benzerlik metriği kullanılarak i ile j düğümleri arasındaki kenarın değeri hesaplanmaktadır. Bu değer tüm i ve j çiftleri için her t anında hesaplanmaktadır. Bu şekilde G_t hesaplanmış olmaktadır. Kayan pencere yaklaşımına göre tüm t değerleri için çizge oluşturulmaktadır. Böylelikle sonuçta bir benzerlik metriği ile oluşturulmuş 30x30xT kadarlık bir tensör depolanmaktadır. Bu tensörün her bir 30x30'luk elemanı yönsüz çizgeyi temsil etmektedir. Bu yönsüz çizge aynı zamanda simetrik bir matris ile gösterilmektedir. D Dow 30 borsasındaki hisselerin kümesini göstermektedir. Denklem (4.6) G_t matrisinin bir simetrik matris olduğunu göstermiştir çünkü i düğümünden j düğümüne giden kenarın hesaplanması ile j düğümünden i düğümüne giden kenarın hesaplanması aynıdır. $G_t(i, j)$ hesaplanırken oluşturulan pencere ile $G_t(j, i)$ hesaplanırken oluşturulan pencere aynı bölgede bulunduğundan dolayı bu çizge bir yönsüz çizge olmaktadır. Bu 2 kenarda aynı bilgiyi sunmaktadır.

$$G_t(i, j) = G_t(j, i) \; \forall (i, j) \in D \tag{4.6}$$



Şekil 4.1: 2 hisse senedinin arasında oluşan kenarın kayan pencere yaklaşımı ile hesaplanması.

Oluşturulan bu yönsüz çizgenin asıl amacı hisse senetlerinin arasındaki korelasyonu ya da benzerliği gösteren bir veri yapısı oluşturmaktır. Bu şekilde bu çizge üzerinden hangi hisselerin daha değerli olabileceği, merkezcil olabileceği ya da birlikte aynı yönde, ters yönde hareketler yapabileceği öngörülebilir ve DL modelleri kullanılarak modelleme yapılabilir.

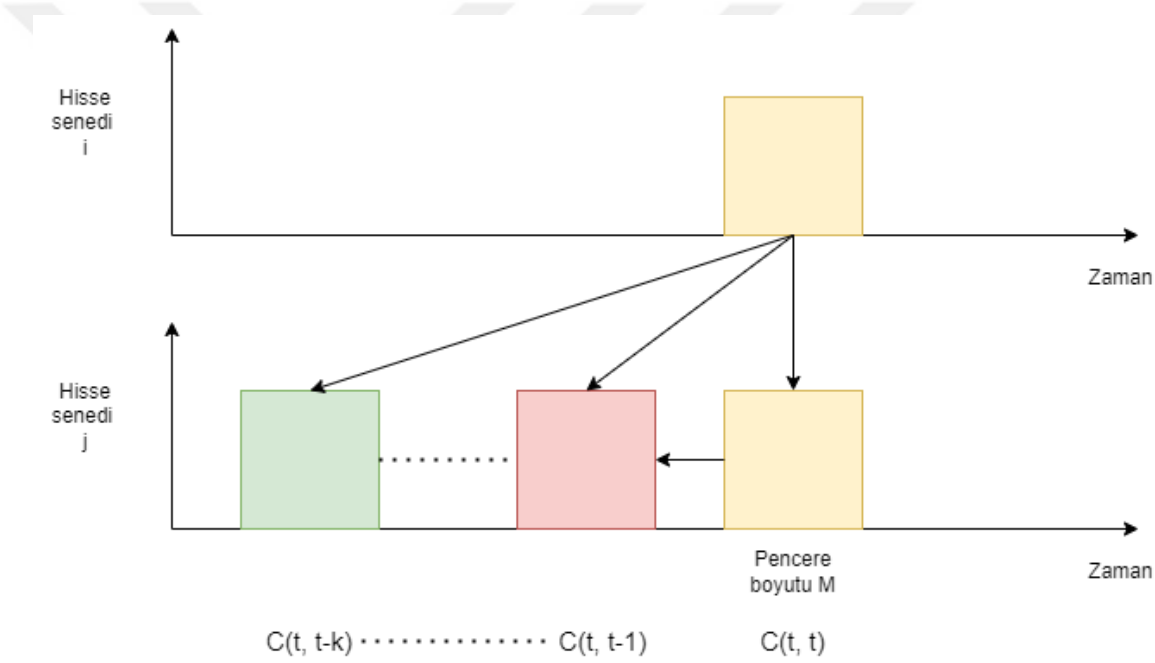
4.5.3 Yönlü çizge

Yönsüz çizge iki hisse senedi arasındaki ilişkiyi bir kenarda göstermektedir fakat bu gösterimin eksik olan bir kısmı iki kenarında aynı bilgiyi içermesidir. Bu eksiği kapatmak amacıyla ve daha iyi bir çizge gösterimi tasarlanması amacıyla borsayı yönlü çizge olarak göstermek bu tez kapsamında amaçlanmıştır. Yönlü çizgede A hissesinden B hissesine giden bir kenarın ağırlığı B hissesinden A hissesine giden bir kenarın ağırlığıyla aynı olmamaktadır. Burada A hissesinin B hissesini takip etme ağırlığı ile B hissesinin A hissesini takip etme ağırlığı aynı olmamaktadır.

$$G_t(i,j) = \text{sign}(C(t_k)) * max(|C(t-k,t)|) \;\; 0 \leq k \leq N-1 \qquad (4.7)$$

Denklem 4.7'de yönlü çizge G_t'nin hesaplanması gösterilmiştir. Burada hisse senedi i ve j arasındaki korelasyon M gün üzerinden hesaplanmaktadır. Yani pencere boyutu M ile gösterilmektedir. Yönlü çizge oluşturulurken j hisse senedi için pencerenin

konumu t anında sabit kalırken i hisse senedi için pencerenin konumu sürekli 1 kaydırılarak korelasyon hesaplanmaktadır. Bu pencere her kaydırıldığında yeni bir korelasyon değeri hesaplanmaktadır. Bu pencere toplamda N kere kaydırıldıktan sonra çıkan değerlerin mutlak değeri alınarak en yüksek benzerliği ya da korelasyonu gösteren değer o kenara verilmektedir. Bu yöntem i hissenin k gün önceki davranışı ile j hissenin güncel olan davranışını kıyasladığından dolayı eğer i hissesinin önceki günlerdeki değeri j hissesini takip ediyorsa ya da tersi yönde hareket yapıyorsa o kenara yüksek bir değer verecektir. Eğer aralarında herhangi bir ilişki yoksa düşük bir değer verecektir. Şekil 4.2'de yönlü çizgenin bir kenarının hesaplanması gösterilmiştir.



Şekil 4.2: 2 hisse senedi için yönlü çizge üzerindeki bir kenarın hesaplanmasını gösteren şekil.

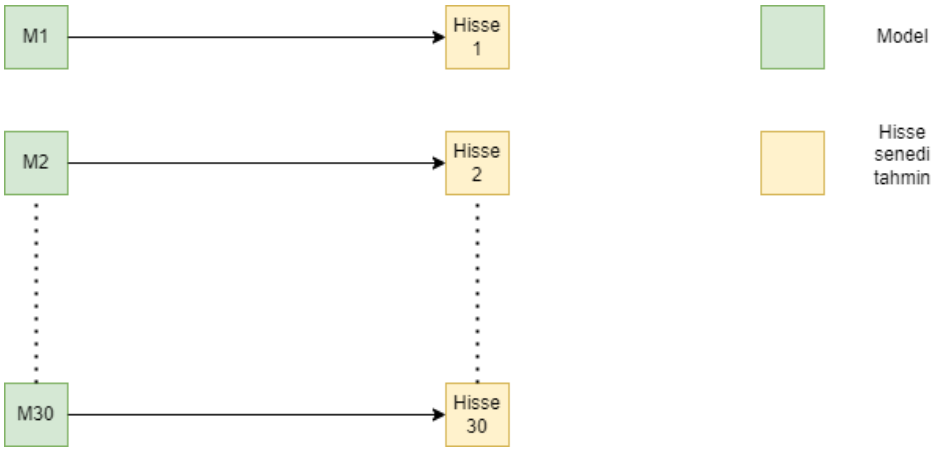
$$G_t(i,j) = G_t(j,j) \; \exists (i,j) \in D \qquad (4.8)$$

Denklem (4.8) G_t çizgesinin yönlü bir çizge olduğunu göstermektedir. Bunun sebebi ise çizgenin kenar değeri hesaplanırken kayan pencere yöntemi ve maksimum işleminin gerçekleşmesidir. Bu şekilde çizgenin simetrik yapısı bozulmaktadır. Yani i düğümden çıkan kenar ile j düğümünden çıkan kenarlar aynı değeri taşımamaktadır.

Denklem (4.7) hesaplanan korelasyon değerlerinden maksimum olanı seçmektedir. Bunun iki önemli sebebi bulunmaktadır. İlki DNN modellerinin maksimum işlemi kullanıldığında daha iyi çalışmasıdır. İkinci sebebi ise bir hisse senedinin diğer hisse senedini ne kadar oranda takip ettiği ya da benzer bir yapıda ilerlediğini gösteren bir kenar elde etmek için maksimum işleminin teorik olarak da mantıklı olmasıdır. Eğer minimum alma yöntemi uygulanmış olsaydı yüksek korelasyona sahip kenarlar yok edilmiş olacaktı. Ortalama almak teoride mantıklı gibi durmasına rağmen maksimum alınması daha iyi sonuçlar elde edilmesini sağlamaktadır. Bu çalışmada N = 5, 10 ve 20 seçilerek bir çizge üretilmiştir. M pencere boyutu ise N değerine eşit olacak şekilde güncellenmiştir. Pencere boyutunun 5 olarak seçildiği durumda eğer N 5'ten küçük bir değer seçilirse yeterince geçmişe gidilemez. Eğer 5'ten büyük bir değer olarak seçilirse de çok geçmişe gideceğinden dolayı geçmiş ile gelecek arasındaki ilişki zayıflamaya ya da anlamsız olmaya başlayacaktır. Bundan dolayı N ve M değerlerinin birbirine yakın seçilmesi mantıklı bulunmuştur.

4.6 Derin Öğrenme Modelleri ve Uygulanması

Bu tez kapsamında geliştirilen DNN modelleri ve temel kıyaslama modelleri etiket olarak DOW 30 borsasındaki hisse senetlerini birer birer almaktadır. Yani her bir model bir hisse senedinin gelecekteki değişim oranını öğrenmeye çalışmaktadır. Bu şekilde her hisse senedi için bir model eğitilmektedir. Dolayısıyla bir model türü tasarlanırken toplamda 30 ayrı model oluşturulmaktadır. Bu çalışma kapsamında tüm modeller bu şekilde tasarlanmıştır. Her bir model eğitilirken MSE kullanılarak bir optimizasyon yapılmaktadır. Şekil 4.3 tasarlanan bir model türünün eğitim ve tahmin sürecini göstermektedir. Burada çoklu etiketleme yapılmamıştır. Bunun sebebi 30 hisse senedini aynı anda tahmin etmenin daha zor bir problem olmasıdır. Her bir hisse senedinin değeri ayrı ayrı tahmin edildikten sonra istenen şekilde birleştirilerek bir strateji oluşturulabilir.



Şekil 4.3: Model eğitimini ve tahmini gösteren şekil.

4.6.1 Temel modeller

Bu modeller yapılan çalışma kapsamında kıyaslama amacıyla tasarlanmış olan modellerdir. Bu tez kapsamında çizge tabanlı modellerin daha iyi performans göstermesi beklenmektedir. Bunun kıyaslanabilmesi amacıyla bazı basit modeller kullanılması gerekmektedir. Bu bölümde bu amaçla kullanılan modeller verilmiştir.

4.6.1.1 MLP

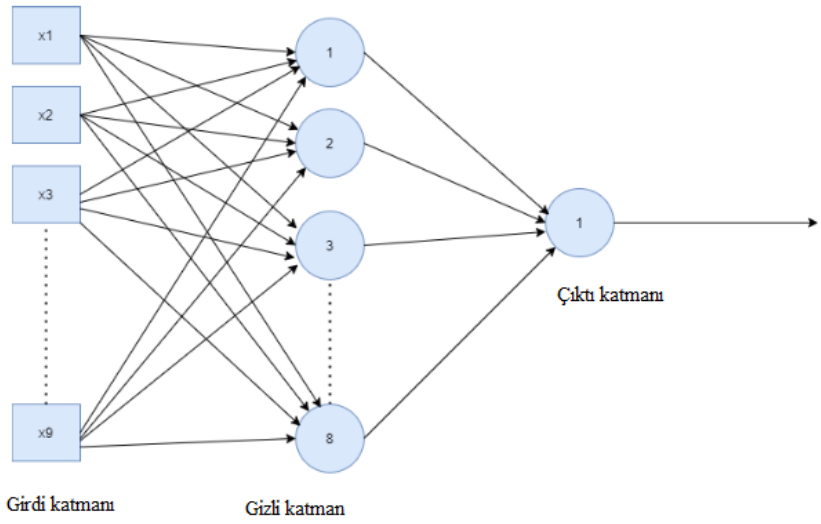
Bu çalışmada temel kıyaslama modellerinden bir tanesi MLP modeli kullanılarak tasarlanmıştır. RNN ve LSTM modellerini borsa gibi karmaşık bir sinyal üzerinde eğitmek zor olduğundan dolayı MLP modeli tercih edilmiştir. Bu MLP modeli sade bir model olarak tasarlanmıştır. Borsa üzerindeki bir hisse senedini tahmin ederken yalnızca o hisse senedinin özniteliklerini girdi olarak almaktadır ve o hisse senedinin değişim oranını etiket olarak kabul edilmektedir. Bu şekilde ayrı ayrı 30 tane model eğitilmektedir.

MLP modeli eğitilirken kullanılan öznitelikler şunlardır:

- Son 5 gün gerçekleşen değişim miktarları(artış oranları)
- Son 5 gün gerçekleşen değişim miktarlarının minimum, ortalama ve maksimum değerleri
- Son 5 gün içerisinde gerçekleşen artış sayısı

4x1 hisse senedinin geçmişi ile ilgili bilgi veren istatiksel öznitelik vektörü ile 5x1 geçmişteki değişimleri tutan bilgi öznitelik vektörü birleştirilerek 9x1'lik bir öznitelik vektörüne dönüştürülmektedir. MLP modeli 9x1'lik bu öznitelik vektörünü

ilk katmanda girdi olarak almaktadır. Bu vektör 8 nöron hücresi bulunan bir gizli katmandan geçirilmektedir. Oluşan 8x1'lik vektörde son olarak çıktı nöronuna bağlanmaktadır. Şekil 4.4 MLP modelinin yapısını göstermektedir. Bu MLP modeli 30 hisse senedinin bulunduğu DOW 30 borsasında yalnızca bir hisse senedi üzerinden eğitilmektedir. Dolayısıyla diğer hisse senetlerinin oluşturduğu hareketleri öğrenecek bir model tasarımı değildir fakat tez kapsamında asıl tasarlanan modeller için önemli bir model olarak görülebilir.

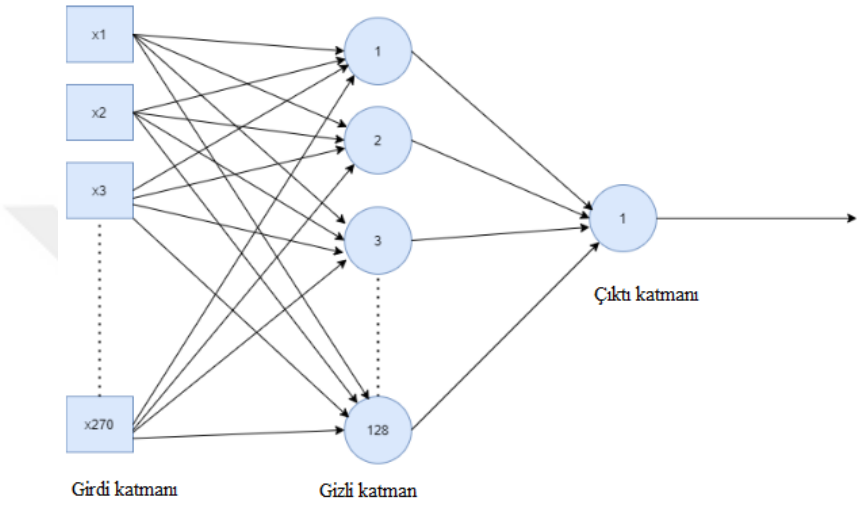


Şekil 4.4: MLP modeli.

4.6.1.2 MLP-Multi

MLP modelinin temel eksik noktası yalnızca 1 hisse senedinin geçmişine bakarak karar vermesidir. Hisse senetleri arasındaki ilişkiyi öğrenebilen bir yapıya sahip olmayan MLP modelinin geliştirilmiş haline MLP-Multi model ismi verilmiştir. MLP-Multi modeli 30x5 hisse senetlerinin geçmişini tutan bir matris ve 30x4 bu hisse senetlerine karşılık gelen öznitelikleri almaktadır. Toplamda boyutu 30x9 olan bir matris boyutu 270x1 olan bir vektöre dönüştürülmektedir. Bu vektör MLP modeline girdi olarak verilmektedir. Bu model bir önceki modelden farklı olarak her hisse senedi için öznitelik vektörünü aldığından dolayı hisseler arasındaki ilişkiyi öğrenebilecektir.

Şekil 4.5 üzerinde MLP-Multi modeli gösterilmiştir. Bu modele 270x1 boyutunda öznitelik vektörü girdi olarak verilmiştir. Bu vektör 128 nörona sahip bir gizli katmandan geçirildikten sonra çıktı nöronuna bağlanmıştır. En son oluşan çıktı tahmin olarak üretilmektedir. MLP-Multi birden çok hisse senedini birlikte öğrenerek bir hisse senedi tahmini yapmaya çalışmaktadır. Modelin hisse senedi arasındaki korelasyonları öğrenebileceği düşünüldüğünden dolayı MLP modelden daha iyi performans vermesi beklenmektedir.



Şekil 4.5: MLP-Multi modeli.

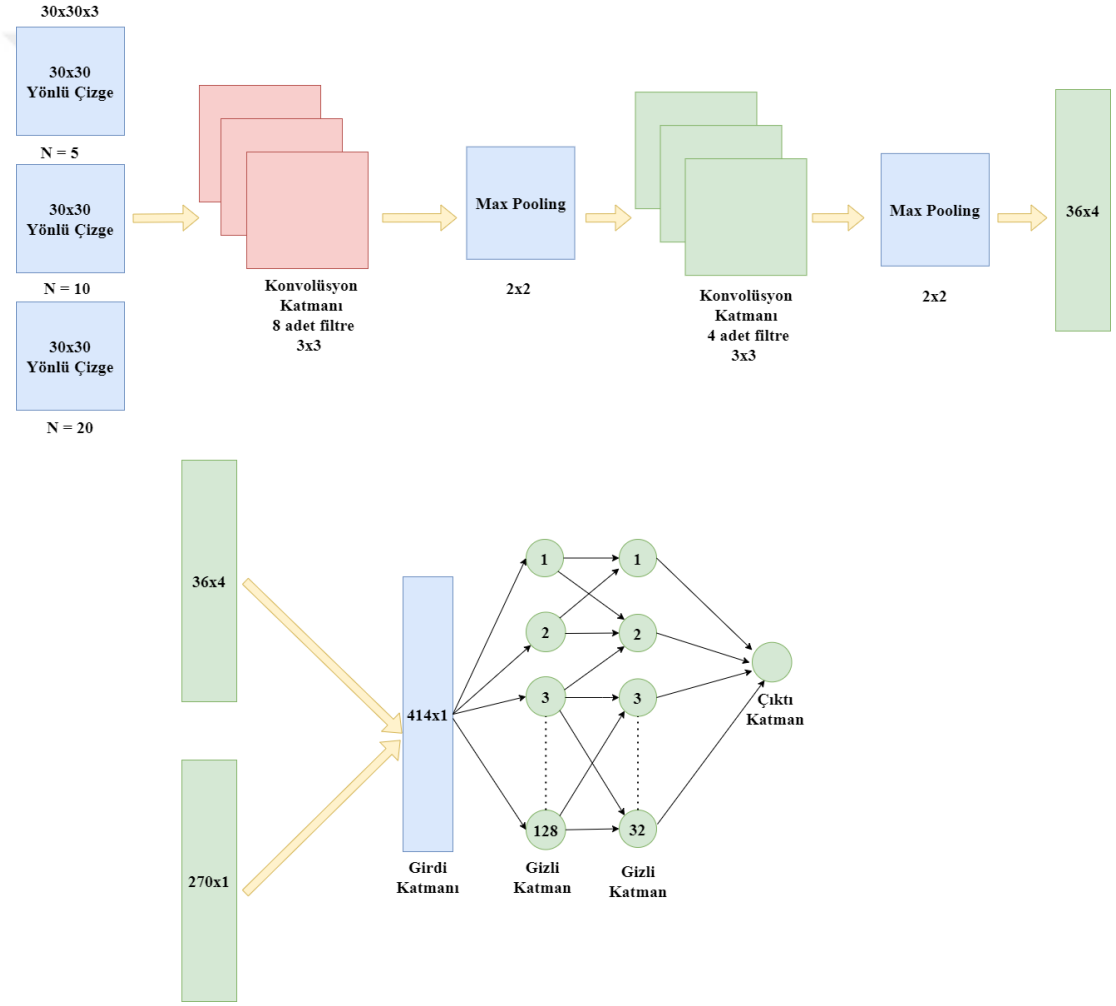
4.6.2 Çizge tabanlı modeller

Yönlü ve yönsüz çizge komşuluk matrisi yardımıyla gösterilmektedir. Bu matrislerin boyutu daha önceden de bahsedildiği üzere 30x30'dur. Bu çizgelerin öğrenilebilmesi amacıyla CNN modeli geliştirilmiştir. Buradaki konvolüsyon katmanları çizgeyi girdi olarak aldıktan sonra sırayla filtrelerden ve havuz katmanlarından geçirmektedir. En son oluşan çıktı vektörü bir önceki bölümde bahsedilen öznitelik vektörü ile birleştirilerek bir MLP modelinden geçirilmektedir. Bu MLP modeli hisse senedinin gelecekteki değişim oranını öğrenmektedir.

Şekil 4.6 yönlü çizge için tasarlanan modeli göstermektedir. Bu modelin yapısı sırasıyla şu şekildedir

- Pencere boyutu 5, 10 ve 20 için tasarlanan 30x30'luk 3 tane yönlü çizge birleştirilerek 3x30x30'luk matrisler elde edilmektedir.

- Bu 3x30x30'luk matrisler 3x3'lük filtrelerden oluşan 2 adet konvolüsyon katmanından geçirilmektedir.
- Çizgelerin çözünürlüklerinin düşürülmesi amacıyla 2 adet havuz katmanı kullanılmaktadır.
- Konvolüsyon katmanın çıktısında elde edilen 36x4 boyutundaki vektör düz bir hale getirilerek 144x1'lik bir vektöre dönüştürülmektedir.
- 144x1 boyutundaki konvolüsyon çıktısı olan vektör ile 270x1'lik öznitelik vektörü tek bir vektör olarak birleştirildikten sonra 414x1'lik bu vektör 2 gizli katmanlı ve 128 nöronlu olan MLP modeline verilmektedir.
- Çıktı olarak bir hisse senedinin tahmini alınmaktadır.



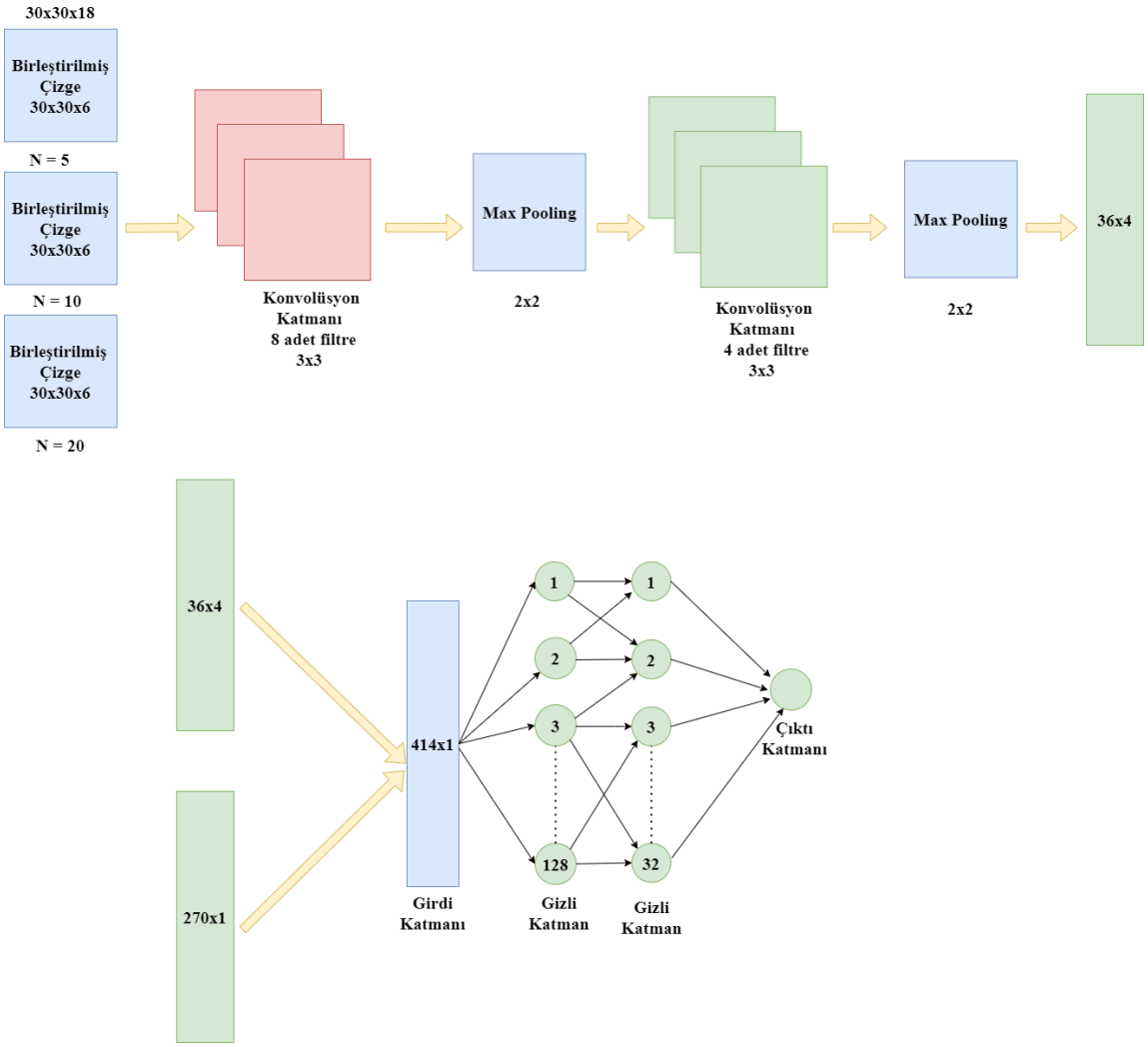
Şekil 4.6: Yönlü çizge için CNN modeli.

Yönsüz çizge için tasarlanan CNN modeli bu modele benzemektedir fakat konvolüsyon katmanındaki filtre sayısı farklıdır. Son katmanda filtre sayısı olarak 1 kullanılmıştır bundan dolayı çıktıda 36x1'lik bir vektör elde edilmektedir. Bu vektör

270x1 boyutundaki öznitelik vektörü ile birleştirildiğinde elde edilen 306x1'lik vektörü 1 katmanlı bir MLP modeli ile eğitilmektedir.

Bu yönlü ve yönsüz çizge modelleri Pearson, Spearman ve Öklid için ayrı ayrı tasarlanmıştır. Bundan dolayı toplam 3x30x30 bu matrislerden yönlü ve yönsüz çizgelerde 6 adet bulunmaktadır. Bu tez içerisinde bu modellerin her biri ayrı ayrı eğitilerek toplamda 6 adet çizgeden sonuç alınmıştır. Burada ilk tasarlanan modeller Pearson korelasyon çizgeleri ile tasarlandığından çizgeler arasında temel model olarak kabul edilmektedir.

Farklı çizgeler tasarlanmasının temel amacı modelin farklı gösterimler ile farklı şeyler öğrenebildiğini gözlemlemek ve bunlardan bazılarının daha başarılı olduğunu göstermektir. Bu mantığa dayanarak tüm çizgeler birleştirilerek başka bir CNN modeli eğitilmektedir. Şekil 4.7 birleştirilmiş olan çizge modelinin tasarımını göstermektedir. CNN model yapısı yönlü çizge modelinin birebir aynısı olarak tasarlanmıştır. Girdi matrisinin boyutu tüm çizgeler verildiğinden dolayı 18x30x30 olmaktadır. Bu modelin ayrı ayrı tasarlanan çizge modellerinden daha başarılı olması beklenmektedir.



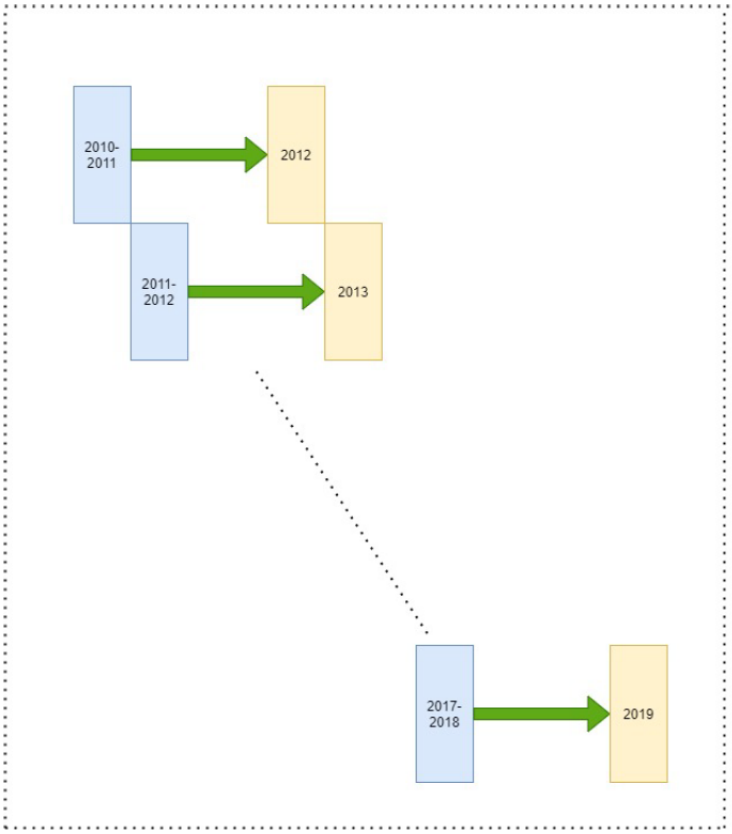
Şekil 4.7: Birleştirilmiş çizge öğrenme modeli.

4.6.3 Eğitim stratejisi

Şekil 4.8 modellerin eğitim stratejisini göstermektedir. Eğitim ve test esnasında kaydırmalı çapraz doğrulama yöntemi kullanılmıştır. Borsanın son zamanlardaki trendlerden daha çok etkilendiği düşünüldüğünden dolayı 2 sene eğitim ve 1 sene test stratejisi kullanılmıştır. Bu şekilde 2012 yılından başlayarak 2020 yılına kadar toplam 8 yıllık tahmin alınmaktadır. Eğitimler gerçekleştirilirken MSE hatası optimize edilmektedir ve gradyan iniş algoritması kullanılmıştır.

Derin öğrenme modellerinin problemlerinden bir tanesi de ağırlıkların rastgele bir noktada başlayıp yerel minimumda takılmasıdır. Bu sorunun üstesinden gelmek amacıyla her bir model 30 kere farklı rastgele ağırlık ile başlatılıp eğitilmektedir ve sonuç olarak ortalaması alınmaktadır. Modellerin hiper parametreleri optimize

edilirken ızgara tabanlı arama yöntemi türevinde bir deneme yanılma yöntemi kullanılmıştır.



Şekil 4.8: Modellerin eğitim stratejisi.

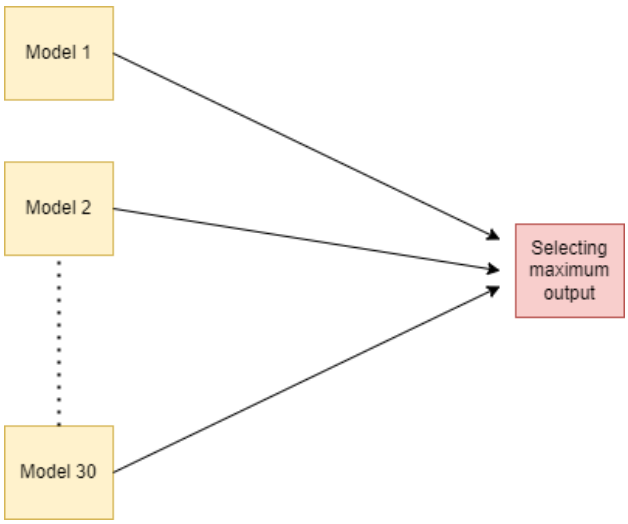
4.7 Yatırım Stratejisinin İyileştirilmesi ve Kolektif Öğrenme Modeli

Bu tez kapsamında tasarlanan temel ve çizge modelleri geliştirmek amacıyla bazı stratejiler ve modeller geliştirilmiştir. Bu strateji ve modellerin amacı tasarlanan modellerin zayıf yönlerini iyileştirerek daha başarılı modeller geliştirmektir. Yapılan çalışma kapsamında her bir model birer hisse senedi için eğitilmiş ve sonuçlar alınmıştır fakat bir hisse senedi için tahmin yapmak borsa açısından yeterli değildir. Eğitilen bu 30 model üzerinden mantıklı olan hisse senetlerinin seçilmesi büyük bir önem taşımaktadır. Bu bölümde eğitilen modellere farklı stratejiler ve kolektif öğrenme yöntemi uygulanarak daha başarılı modeller elde edilmesi planlanmıştır.

4.7.1 Sezgisel strateji

Bir önceki bölümde tasarlanan MLP, MLP-Multi ve çizge tabanlı modeller her bir hisse senedi için ayrı ayrı eğitilmiştir. Bundan dolayı her zaman dilimi t anında 30 ayrı tahmin bulunmaktadır. Yani t anında bir yatırımcı bu 30 hisse senedinden istediğine yatırım yapabilmektedir. Bu yatırımcı t anında 30 hisse senedinden birini seçtikten sonra o hisse senedini satın almaktadır ve t+1 anında geri satmaktadır. Eğer hiçbir hisse senedini almazsa bir sonraki gün bir seçim yapacaktır. Modellerin t anındaki tahmini bir *p* vektörüne yazıldıktan sonra bu vektördeki maksimum artış oranına sahip olan hisse senedine yatırım yapılması stratejisi sezgisel strateji olarak düşünülmüştür.

Şekil 4.9 sezgisel stratejinin uygulanışını göstermektedir. 30 ayrı modelden tahmin değerleri alındıktan sonra maksimum olan değer tahmin olarak verilmektedir ve yatırım stratejisi buna göre uygulanmaktadır. Sezgisel stratejinin amacı her modelin tahmininden en yüksek artış değerine sahip olanı seçmektir. Bu şekilde modellerin yaptığı hata en aza indirgenerek muhtemel yükselecek olan bir hisse senedinin seçilmesi amaçlanmıştır. Bu yüzden sezgisel stratejinin ayrı ayrı her hisse senedine yapılan yatırımdan daha fazla kazanç getirmesi beklenmektedir.



Şekil 4.9: Sezgisel strateji.

4.7.2 Temel stratejiler

Temel stratejiler uygulanan sezgisel strateji ve tasarlanacak olan kolektif öğrenme yönteminin kıyaslanması amacıyla oluşturulmuş olan stratejilerdir. Bu tarz kıyaslama metrikleri her ML probleminde bulunmaktadır. Tasarlanacak olan yeni modellerin

temel modelleri ya da stratejileri geçebilmesi gerekmektedir. Borsa için temel stratejiler insan algısıyla yapılabilecek olan en basit stratejilerdir. Bu stratejiler basit olmasına karşın zaman zaman iyi kazandırabilir fakat risk faktörü genellikle yüksek olan yöntemlerdir.

S1 stratejisi DOW 30 borsasında bir önceki yıl en çok yükselen hisse senedine bir sonraki yılda yatırım yapmaktadır. S1 stratejisi aç gözlü bir stratejidir. S2 stratejisi ise bir önceki sene en çok artan 3 hisse senedine yatırım yapmaktadır. S2 stratejisi S1 stratejisine göre daha güvenilir bir stratejidir. 3 farklı hisse senedinin 3'nün de aynı anda azalma olasılığı 1 tanesinin azalma olasılığından çok düşüktür. Bu yüzden uzun vadede kaybettirme olasılığı daha düşüktür.

Bir farklı stratejide DOW 30 indeksine yatırım yapmaktır. Yani 30 adet hisse senedinin yıllık olarak değişim miktarının ortalaması en güvenilir yöntemlerden bir tanesidir. Geliştirilecek olan modelin bu stratejiden daha çok kazandırması amaçlanmıştır.

4.7.3 Kolektif öğrenme ile strateji geliştirme

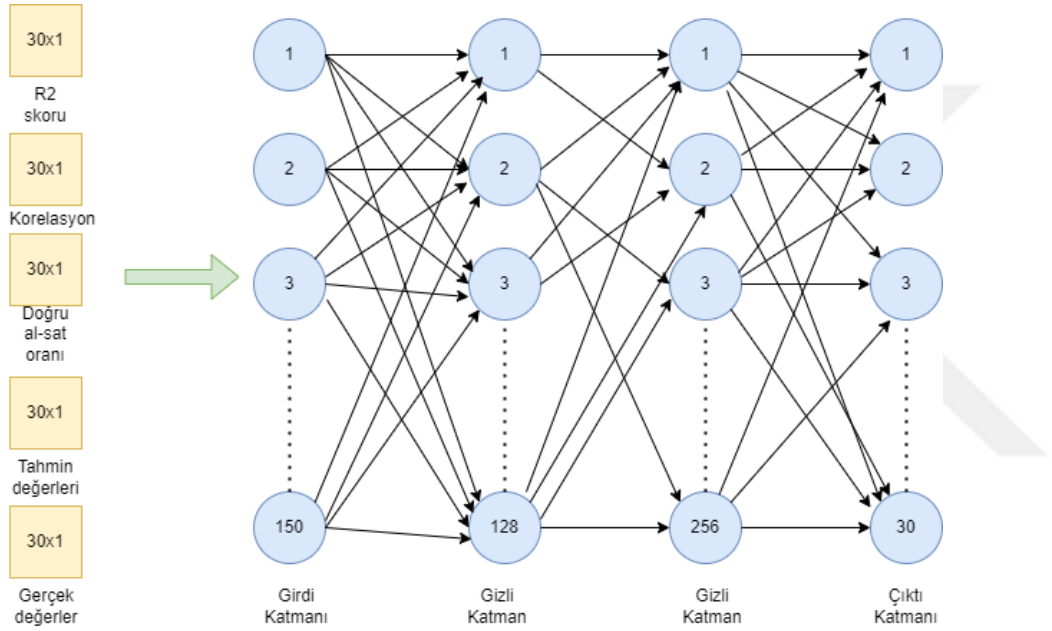
Bu çalışmada kolektif öğrenme yöntemi ile eğitilen modellerin iyileştirilmesi amaçlanmaktadır. Her bir model türü için geliştirilen 30 farklı hisse senedi tahmini modelinin yaptığı hataların öğrenilmesi amacıyla bir NN modeli geliştirilmiştir. Bu model diğer modellerin yaptığı hatalara odaklanarak daha düzgün tahminler yapmayı amaçlamıştır. Bu tahminler sayesinde daha fazla kazanç sağlanması hedeflenmiştir.

Bir model türünün 30 farklı hisse senedi için tahminleri 30x1 boyutunda *p* vektöründe tutulmaktadır. NN modeli aşağıdaki öznitelikleri kullanarak eğitilmektedir.

- Modelin tahmini ile gerçek değerler arasındaki R2 skoru(30x1)
- Modelin tahmini ile gerçek değerler arasındaki korelasyon(30x1)
- Modelin belirli bir pencere boyutunda yaptığı doğru yönlü tahmin oranı(30x1)
- Modelin t-1 günü için tahmin ettiği 30x1 vektör ve bu tahminlere karşılık gelen gerçek değerler (30x1) vektörü

Yukarıda verilen öznitelikler pencere boyutu W = 10 üzerinden hesaplanmıştır. R2, korelasyon ve doğru yönlü tahmin modellerin son zamanda gösterdiği performansı

ortaya koymaktadır. Modelin tahmin ettiği 30x1'lik vektör ile gerçek değerler vektörü de modellerin son zamanlardaki başarısına bakarak hangi hisse senedinin daha çok artacağını belirlemede önemli olan özniteliklerdir. NN modeli tüm bu öznitelikleri 150x1 boyutunda bir girdi vektörü olarak almaktadır. NN modeli etiket değeri olarak en çok artan hisse senedinin numarasını almaktadır. Bu şekilde eğitilecek olan NN modeli hangi hisse senedinin tahminin daha çok artacağını öğreneceğinden dolayı modelin performansı artacaktır. Eğer tasarlanan modeller bazı hisse senetleri değerlerinin belirli zaman aralıklarında iyi tahmin edemiyorsa NN modeli bunu öğrenecektir.



Şekil 4.10: Kolektif öğrenme modeli.

Şekil 4.10 tasarlanan NN modelini göstermektedir. Bu model 150x1 boyutundaki öznitelik vektörünü aldıktan sonra çıktı olarak 30x1'lik tahmin vektörünü üretmektedir. Bu çıktı vektörü t anında hangi hisse senedinin en çok artacağını tahmin etmektedir. Bu yüzden dolayı etiket olarak t anında 30 hisse senedinden hangisinin en çok artacağı one-hot vektörü olarak verilmektedir. Örneğin 0 numaralı hisse senedi t+1 anında en çok artacak ise [1000….0] şeklinde 30x1'lik bir vektör etiket olarak verilmektedir.

150x1 öznitelik vektörü sırasıyla şu katmanlardan geçmektedir.

- 128 nörona sahip gizli katman ve ReLu aktivasyon fonksiyonu
- 256 nörona sahip gizli katman ve ReLu aktivasyon fonksiyonu

- 30 nörona sahip çıktı katmanı ve softmax aktivasyon fonksiyonu

Kolektif öğrenme modelinin çıktısı softmax aktivasyon fonksiyonundan geçirildikten sonra seçilmesi gereken hisse senetlerinin olasılık değerleri üretilmektedir. Bu değerlerin toplamı 1 olacaktır. Model her t anında olasılık değeri en yüksek olan hisse senedinden pay aldıktan sonra t+1 anında satmaktadır. Bu model optimize edilirken kategorik çapraz entropi(categorical cross entropy) hata fonksiyonunu ve gradyan iniş algoritmasını kullanmaktadır. Bu model yalnızca yönlü ve yönsüz çizge üzerinde uygulanmıştır. Temel karşılaştırma modelleri olan MLP ve MLP-Multi için eğitilmemiştir

.

5. DENEY SONUÇLARI VE DEĞERLENDİRME

Bu çalışma kapsamında tasarlanan modeller eğitildikten sonra bu modellerin performansları bazı metrikler üzerinden hesaplanmış ve birbirleriyle kıyaslanmıştır. Bu metrikler hesaplanırken modeller toplamda 30 kere eğitilmiştir ve ortalaması alınmıştır. Çalışmanın deneysel sonuçları ilerleyen bölümlerde açıklanmaktadır.

5.1 Çizge Modellerin ve Temel Modellerin Performansları

Bu çalışmada ilk olarak MLP, MLP-Multi, Yönlü Çizge ve Yönsüz Çizge modelleri eğitilmiştir. Yönlü Çizge ve Yönsüz Çizge Pearson korelasyon formülü kullanılarak üretilen çizgeleri temsil etmektedir. Her bir model 30 farklı hisse senedi için ayrı ayrı eğitilmiştir. Modeller borsada bir hisse senedi için bir sonraki gün gerçekleşecek olan değişim miktarının yönünü tahmin etmektedir. Yani bir sonraki gün için alma işlemi mi gerçekleşecek yoksa satma işlemi mi gerçekleşecek modellenmiştir. Her yıl için yapılan tahminlerin sonuçları ayrı ayrı hesaplanmıştır. Bu 4 model için doğruluk, kesinlik ve hassasiyet değerleri hesaplanmıştır. Bu modellere ek olarak rastegele tahmin yaparak al-sat yapan bir strateji Rastgele Strateji (Random Strategy, RS) oluşturulmuştur. Bir diğer model olarak ise hareketli ortalama (Moving Average, MA) oluşturulmuştur. MA modelinin pencere boyutu 25 olarak seçilmiştir. RS ve MA modelleri basitçe performansların karşılaştırılması amacıyla eklenmiştir.

Eğitilen bu 4 model için yıllık ortalama doğruluk değerleri yaklaşık olarak 0,50 civarındadır. Ayrıca RS ve MA modellerinin doğruluk değerleri de 0,50 civarında gelmektedir. Tüm yıllar için doğruluk değeri yaklaşık 0,50 gelirken kesinlik skoru yıllık trendlere bağlı bir şekilde gerçekleşmektedir. Burada kesinlik skoru modelin hisse senedi değerini artacak şekilde yaptığı tahminlerden gerçekte ne kadarlık bir kısmının artacak olduğunu göstermektedir. Bu deneyde eğer bir sonraki gün artış gerçekleşecek ise o gün alım yapılıp bir sonraki gün satılmaktadır. Artış gerçekleşmeyecek günlerde ise herhangi bir işlem yapılmamaktadır. Bundan dolayı kesinlik skorunun yüksek olması önemlidir. Çizelge 5.1 4 farklı model için kesinlik skorlarını göstermektedir. Bu tabloda bazı yıllarda kesinlik skorunun daha yüksek

olduğu gözlemlenmektedir. Bu yıllar genellikle artış trendlerinin olduğu yıllardır. Kesinlik skorunun düşük olduğu yıllarda ise genellikle borsa durağan dönemini yaşamaktadır.

Çizelge 5.1:4 farklı model için kesinlik skorları.

	MLP	MLP-Multi	Yönsüz Çizge	Yönlü Çizge	RS	MA
2019	0,55	0,55	0,55	0,55	0,56	0,53
2018	0,52	0,52	0,52	0,52	0,51	0,51
2017	0,54	0,54	0,54	0,54	0,54	0,53
2016	0,52	0,52	0,52	0,52	0,53	0,52
2015	0,49	0,49	0,49	0,49	0,48	0,47
2014	0,52	0,53	0,53	0,53	0,52	0,51
2013	0,54	0,54	0,54	0,54	0,55	0,53
2012	0,51	0,51	0,51	0,51	0,51	0,50

Çizelge 5.2:4 farklı model için hassasiyet skorları.

	MLP	MLP-Multi	Yönsüz Çizge	Yönlü Çizge	RS	MA
2019	0,50	0,51	0,50	0,51	0,50	0,63
2018	0,50	0,51	0,51	0,51	0,51	0,57
2017	0,51	0,50	0,50	0,49	0,50	0,68
2016	0,50	0,50	0,50	0,51	0,49	0,60
2015	0,50	0,51	0,51	0,50	0,52	0,50
2014	0,50	0,51	0,51	0,51	0,50	0,63
2013	0,50	0,50	0,51	0,51	0,50	0,69
2012	0,50	0,50	0,51	0,51	0,50	0,63

Çizelge 5.2 hassasiyet skorlarını göstermektedir. Hassasiyet skorları modellerin borsada artış gösteren günlerden ne kadarlık bir kesiminin doğru olarak bulunabildiğini göstermiştir. Artış gösteren günlerden olabildiğince büyük bir çoğunluğunu bulmak borsada kazanç sağlamak için önemlidir. Bu tabloda genellikle tüm modeller benzer oranda bir tespit yapmıştır. MA modelinin hassasiyet skorları diğer modellerden daha yüksek gelmektedir. Bu da MA modelinin artış gerçekleşecek olan gün sayısını daha fazla yakaladığını göstermektedir.

Çizelge 5.3 6 farklı model için yıllık kazanç değerlerini göstermektedir. Bu kazançlar her bir modelin her bir hisse senedi için eğitilmesi ve yatırım yapılması sonucunda elde edilen kazançlardır. Eğer modeller bir sonraki gün için artar şeklinde bir tahmin yaparsa yatırımcı t anında alıp t+1 anında o hisse senedini satacaktır. Her bir hisse senedi için bu şekilde yıllık bir simülasyon yapıldıktan sonra 30 hisse senedi için elde edilen kazançların ortalaması alınarak Çizelge 5.3 oluşturulmuştur. 6 modelin yıllık ortalama kazançları birbirlerine yakın gelmektedir. Yönlü çizge modeli diğer modellerden biraz daha iyi performans göstermiştir. Bu 6 model de DOW 30 indeks değerinden düşük bir miktarda kazanç getirmektedir. MA modelinin hassasiyet skoru yüksek olmasına rağmen aynı oranda kazanç getirmektedir. Bunun sebebi kesinlik skorunun kazancı doğrudan etkilenmesinden kaynaklanmaktadır.

Çizelge 5.3 gözlemlenen önemli bir başka gözlem ise elde edilen kazanç değerleri ile kesinlik skorları ile benzerlik göstermektedir. Yüksek kesinlik değerine sahip yıllarda (2019, 2017, 2013 gibi) elde edilen kazançlar düşük kesinlik değerine sahip yıllara oranla çok daha yüksek olmaktadır. Bu da kesinlik skorunun kazancı etkilediğini göstermektedir.

Çizelge 5.3: 4 farklı modellerin yıllık yüzdelik kazanç değerleri.

	RS	MA	MLP	MLP-Multi	Yönsüz Çizge	Yönlü Çizge	DOW30 indeks
2019	11,31	7,63	11,60	11,54	11,71	12,27	24,2
2018	-0,70	-5,38	0,53	0,15	0,25	0,74	1,0
2017	10,21	16,97	11,33	11,17	11,58	11,73	25,0
2016	10,23	9,34	6,46	7,28	6,46	6,81	16,0
2015	-3,01	-0,98	1,41	1,96	1,49	1,16	4,3
2014	5,66	4,34	7,61	7,07	7,11	7,08	15,4
2013	15,13	15,51	15,07	15,07	15,41	15,33	29,2
2012	7,26	9,21	7,81	8,10	7,91	7,77	15,1
Ortalama	7,01	7,08	7,72	7,68	7,74	7,86	16,2

Çizelge 5.3 üzerinden gözlemlendiği üzere 6 modelin 8 yıllık ortalama kazancı DOW 30 indeksinin getirdiği ortalama kazancın ciddi miktarda altında gelmektedir. Bu elde

edilen sonuçlar kötü durmasına rağmen sezgisel strateji kullanılarak iyileştirilmiştir. Sezgisel strateji yalnızca eğitilen modeller üzerinde kullanılmıştır. MLP, MLP-Multi ,Yönsüz Çizge ve Yönlü Çizge üzerinde sezgisel stratejiler uygulanmaktadır.

$$PoS = \frac{Başarılı\ gerçekleşen\ işlem\ sayısı}{Toplamda\ gerçekleşen\ işlem\ sayısı} \quad (5.1)$$

Denklem (5.1) sezgisel strateji için başarılı işlem oranını gösteren PoS(percent of successful transaction) skorun hesaplamasını göstermektedir. Sezgisel strateji her bir t anı için 30 hisse senedi arasından en çok artış göstereni bulup ona yatırım yapmaktadır. Bundan dolayı her gün mutlaka bir hisse senedi seçilmektedir. Bu yüzden toplamda gerçekleşen işlem sayısı bir yılın işlem yapılan gün sayısına eşit olmaktadır. Başarılı gerçekleşen işlem sayısı ise her t anı için seçilen hisse senedinin bir sonraki gün artmasına göre hesaplanmaktadır. PoS skoru aslında kesinlik skoruna çok benzemektedir. Bu tez kapsamında kavramsal olarak sezgisel strateji için ayrıca isimlendirilmiştir.

Çizelge 5.4:4 farklı model için PoS skorları.

	MLP	MLP-Multi	Yönsüz Çizge	Yönlü Çizge
2019	0,54	0,55	0,55	0,55
2018	0,53	0,51	0,52	0,51
2017	0,54	0,53	0,54	0,55
2016	0,52	0,52	0,52	0,52
2015	0,50	0,49	0,49	0,50
2014	0,52	0,52	0,53	0,54
2013	0,54	0,55	0,55	0,54
2012	0,51	0,52	0,52	0,51

Çizelge 5.4 üzerinde 4 model için hesaplanan yıllık PoS skorları verilmiştir. Çizelge 5.5 tasarlanan 4 modelin sezgisel strateji kullanarak elde edilen yıllık ortalama kazanç değerlerini göstermektedir. PoS skorun %55 civarında olduğu 2013, 2017 ve 2019 yılları için elde edilen kazançlar diğer yıllara göre daha yüksek olmaktadır. 2015 ve 2018 yılları için elde edilen kazançlar diğer yıllara göre çok düşük

gelmektedir ve bu yılların PoS skorları da düşük gelmektedir. PoS skor bu şekilde kazançlar arasında bir ilişki vermesine rağmen 2015 ve 2012 Yönlü Çizge modelinde elde edilen yıllık kazançlar sırasıyla 4,81 ve 18,98'dir. PoS skoru bu model için 2012 ve 2015 yılında 0,50 civarında bir skor vermektedir. Bu kazançlar incelendiğinde PoS skorunun doğrudan kazancın miktarını etkilemediği gözlemlenmektedir. PoS skoru doğru gerçekleşen işlem sayısını göstermektedir.

Çizelge 5.5 üzerinden de görüldüğü üzere MLP modeli %14,70, MLP-Multi modeli yaklaşık %15,84, Yönsüz Çizge %17,63, Yönlü Çizge %18,30 yüzdelik ortalama kazançlar sağlamaktadır. MLP modelinin getirdiği yıllık ortalama kazanç oranı diğer 3 modelden düşüktür. Bunun sebebi olarak MLP modelinin yalnızca tek bir hisse senedinin bilgilerini öğrenmeye çalışmasından kaynaklanmaktadır. MLP-Multi çizge modellerinden daha kötü çalışmakla birlikte MLP modelini performans olarak geçmiştir. MLP-Multi modeli 30 farklı hisse senedinin değerlerini alarak işlediğinden dolayı MLP modelinden daha başarılı bir sonuç elde etmektedir. MLP-Multi modeli MLP modelinden daha başarılı olmasına rağmen DOW 30 indeks değerinin getirdiği kazancı geçememektedir.

Çizge modelleri MLP ve MLP Multi modellerinden yıllık ortalama daha fazla kazanç sağlamaktadır. Bu modellerin getirdiği ortalama kazançlar aynı zamanda DOW 30 indeksinin ortalama kazancından daha yüksektir. Dolayısıyla çizge modellerin performanslarının temel modeller ve DOW 30 indeksinden daha yüksek olduğu gözlemlenmektedir. Bu da hisse senetlerinin çizge olarak gösteriminin etkili bir yöntem olduğunu göstermektedir. Yönlü Çizge modeli Yönsüz çizgenin modelinden biraz daha fazla oranda kazanç sağlamaktadır. Bunun sebebi de Yönlü Çizge simetrik olmayan bir matris ile gösterilmektedir ve daha çok bilgi içermektedir. Bu yüzden dolayı da ortalama kazançta tüm modellerden daha yüksek bir kazanç sağlamaktadır.

Şekil 5.1 üzerinde 4 modelin 8 yıllık ortalama kazanç grafiği gösterilmiştir. Bu grafik elde edilirken modellerin her bir yıl için getirdiği kazanç günlere göre hesaplanarak çizdirilmiştir. Elde edilen 8 yıl için kazançların sıralı bir şekilde günlere göre ortalaması alındığında yani bu kazançların toplamı 8'e bölündüğünde elde edilen grafik Şekil 5.1'de gösterilmiştir. Bu grafiğe bakılarak Yönlü Çizge modelinin tüm yıl boyunca diğer modellerden daha çok kazanç sağladığı görülmektedir. Yönsüz çizge modeli ile MLP-Multi modeli yılın belirli bir zaman diliminde birbirlerine

yakın olmasına rağmen Yönsüz Çizge MLP-Multi modelinden daha fazla kazanç sağlamaktadır. MLP modeli genel olarak diğer modellerden daha az kazanç sağlayarak grafikte en aşağıda kalan model olmaktadır.

Çizelge 5.5: 4 farklı modelin sezgisel strateji kullanılarak iyileştirilmesi sonucu elde edilen yıllık kazanç değerleri.

	MLP	MLP-Multi	Yönsüz Çizge	Yönlü Çizge	DOW30 indeks
2019	18,32	17,93	22,31	26,26	24,2
2018	2,98	-3,91	0,97	2,58	1,0
2017	21,40	22,81	25,26	22,43	25,0
2016	10,16	16,11	15,04	15,67	16,0
2015	0,79	6,48	2,25	4,81	4,3
2014	14,84	14,53	19,49	19,83	15,4
2013	28,72	35,07	34,23	35,89	29,2
2012	20,41	17,70	21,50	18,98	15,1
Ortalama	14,70	15,84	17,63	18,30	16,2

Şekil 5.1**:** 8 yıllık ortalama kazanç grafiğinin her model için yıllık gösterimi

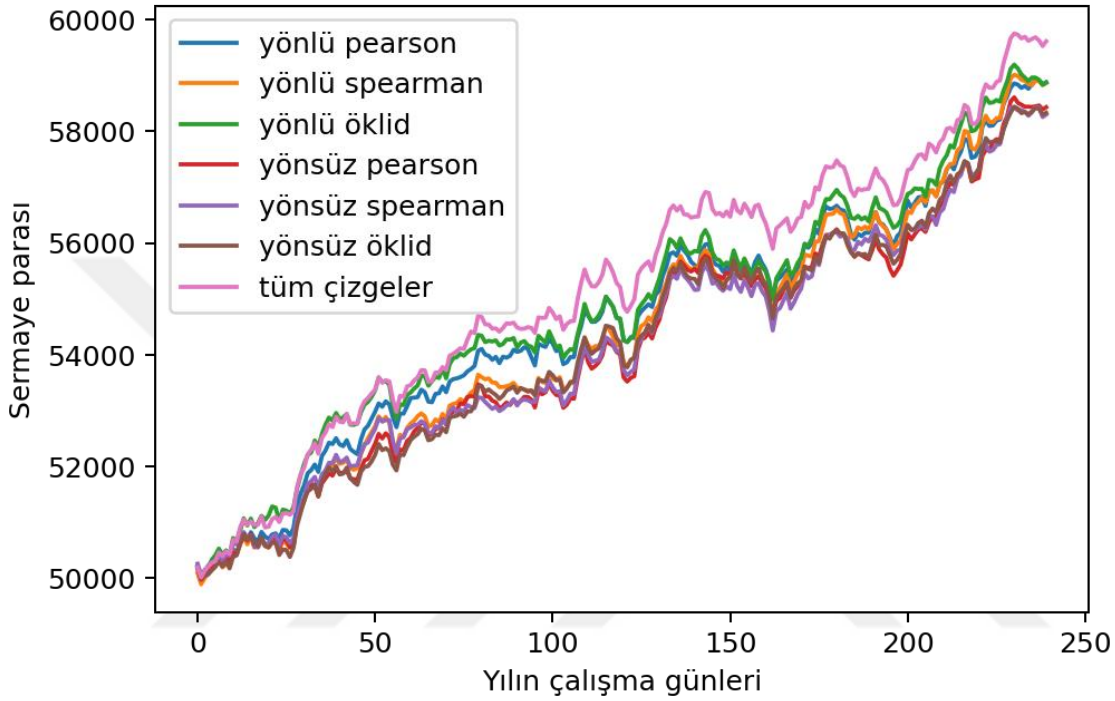
Çizelge 5.6 üzerinde Yönlü Çizge, Yönsüz Çizge, S1, S2 ve DOW 30 indeksinin getirdiği kazançlar sıralanmıştır. Yönlü Çizge ve Yönsüz çizge ortalama kazançta DOW 30 indeksinin ve S2 stratejisinin getirdiği ortalama kazancı geçmesine rağmen S1 stratejisinin ortalama kazancını geçememiştir. Bunun sebebi S1 stratejisinin bir önceki sene yıllık en çok artış gösteren hisse senedini seçmesinden kaynaklanmaktadır. Bir önceki sene çok yüksek artış gösteren bir hisse senedi bir sonraki sene de çok yüksek bir artış göstermektedir. Genellikle bir A firması büyük bir ivme ile gelişiyorsa ve güven kazanıyorsa bu firma belirli senelerde değerini koruyacak ya da yüksek bir artış sağlayacaktır. Bundan dolayı S1 stratejisi 2013, 2017 yıllarında ortalamanın çok üzerinde kazanç sağlamıştır. S1 stratejisi bu avantajına rağmen 2014, 2015 ve 2016 yıllarında DOW 30 indeksinin çok altında kazanç sağlamıştır. Bundan dolayı S1 stratejisi çok güvenilir bir strateji değildir. S2 stratejisi S1'den daha güvenilir olmasına rağmen azalan bir borsada kazanç getirmeyi başaramamaktadır. Bundan dolayı çizge modellerinin ortalama kazançları S1 stratejisinden daha düşük olmasına rağmen S1 stratejisine göre daha az riskli davranmaktadırlar.

Çizelge 5.6: Yönlü ve Yönsüz Çizge ve S1, S2 stratejileri yıllık kazanç çizelgesi.

	Yönsüz Çizge	Yönlü Çizge	S1	S2	DOW30 indeks
2019	22,31	26,26	23,6	0,48	24,2
2018	0,97	2,58	10,8	-3,3	1,0
2017	25,26	22,43	72,7	45,8	25,0
2016	15,04	15,67	1,5	-2,6	16,0
2015	2,25	4,81	-2,4	3,1	4,3
2014	19,49	19,83	-2,6	9,7	15,4
2013	34,23	35,89	47,9	38,9	29,2
2012	21,50	18,98	31,2	30,0	15,1
Ortalama	17,63	18,30	22,8	15,3	16,2

5.2 Tüm Çizge Modellerinin Performansları

Bu çalışmada ilk amaçlanan çizge modelleri Pearson korelasyon formülü kullanılarak elde edilmiştir. Pearson korelasyona ek olarak Spearman ve Öklid uzaklığı kullanılarak elde edilen çizge türleri de bu tez kapsamında eğitilmiştir ve ayrıca tüm çizgelerin birleştirilmiş hali de bir model üzerinde eğitilmiştir.



Şekil 5.2: Tüm çizge modellerinin 8 yıllık ortalama kazanç grafiği.

Çizelge 5.7 eğitilen tüm çizge modellerinin yıllara göre getirdiği kazançları ve 8 yıllık ortalama kazancı göstermektedir. Yönlü Sperman modeli bir çizge türü için en çok kazanç sağlayan çizge gösterimi ve modeli olmuştur. Spearman çizge gösterimi Pearson çizge gösteriminden daha başarılı bir modelleme yapmaktadır. Bunun sebebi daha önceden bahsedildiği gibi Spearman korelasyonun Pearson korelasyondan daha iyi ölçüm yapabildiği durumlar olmasından kaynaklanmaktadır. Pearson, Spearman ve Öklid çizge gösterimleri arasında performans açısından küçük farklar bulunmasına rağmen bu yöntemler ortalamada benzer kazanç sağlamaktadır. Bundan dolayı da bu çizge gösterim yöntemleri kararlı(stable) ve güçlü(robust) olarak gösterilebilir.

Yönlü çizge modelleri farklı çizge türleri içinde yönsüz çizge modellerine göre daha fazla kazanç sağlamaktadır. Bu yönlü çizge modellerinin daha başarılı sonuçlar verdiğini ve bu sonuçların daha kararlı olduğunu göstermektedir.

Tüm çizge türlerini birleştirilerek eğitilen bir model ise yıllık ortalama %20,04 kazanç sağlamıştır. Bu model Yönlü Spearman çizge modelinin performansını da geçmeyi başarmıştır. Dolayısıyla tüm çizge yöntemlerinin birleştirilip eğitilmesi ayrı ayrı eğitilmesinden daha iyi performans göstermektedir.

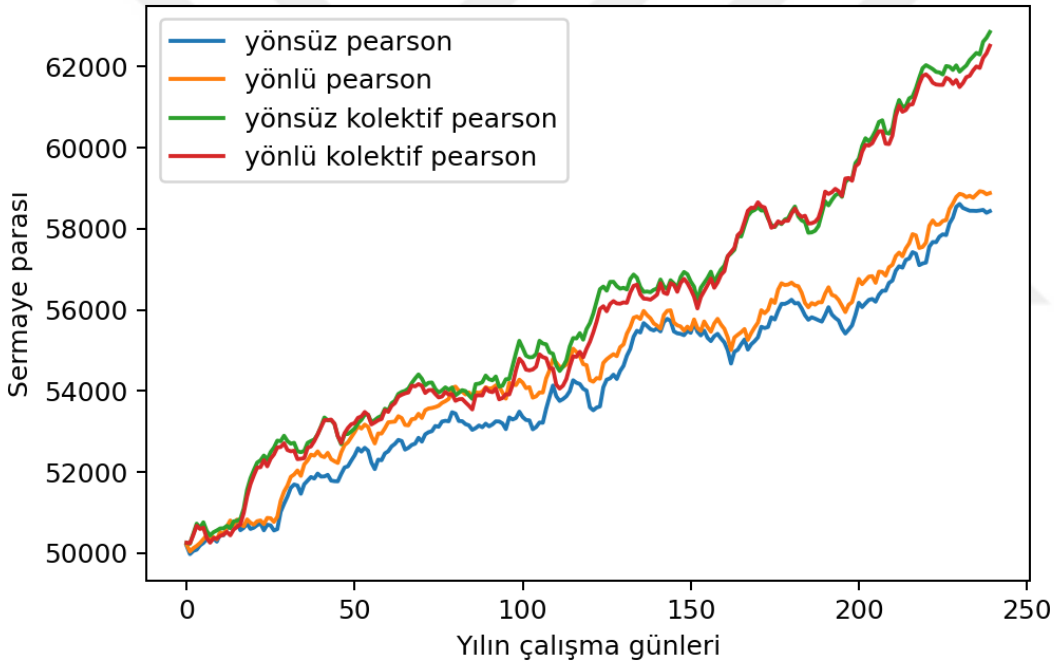
Şekil 5.2 çizge modellerinin 8 yıllık getirdiği ortalama kazancın grafiğini yılın günlerine göre göstermektedir. Bu grafikten görüleceği üzere tüm çizgelerin birleştirilmiş hali neredeyse tüm yıl boyunca diğer modellerden daha çok kazandırmıştır. Yılın ilk başında çok bir fark oluşmamasına rağmen ilerleyen zamanlarda yılın orta periyodunda gerçekleşen işlemler birikimli olacak şekilde bir kazanç sağlamaktadır. Yılın en son kısmında da tüm çizge modelleri en çok kazancı sağlamaktadır.

Çizelge 5.7: Tüm çizge türleri için eğitilen modellerin performansları ve tüm çizgelerin birleştirilmesi ile eğitilen modelin performansı.

	Yönsüz Pearson	Yönlü Pearson	Yönsüz Spearman	Yönlü Spearman	Yönsüz Öklid	Yönlü Öklid	Tüm Çizgeler
2019	22,31	26,26	22,98	32,51	27,61	30,29	25,84
2018	0,97	2,58	-0,27	7,07	3.20	2,40	3,30
2017	25,26	22,43	28,47	24,18	26,53	26,08	26,60
2016	15,04	15,67	17,11	18,50	17,80	17,05	18,21
2015	2,25	4,81	0,87	4,52	2,02	3,58	10,52
2014	19,49	19,83	15,02	10,41	18,59	14,09	13,15
2013	34,23	35,89	35,29	33,71	28,43	34,05	42,48
2012	21,50	18,98	20,93	19,12	14,50	18,37	20,27
Ortalama	17,63	18,30	17,58	18,72	17,33	18,23	20,04

5.3 Kolektif Öğrenme Modelleri Performansları

Kolektif öğrenme modelleri çizge modellerinin performansını iyileştirmek amacıyla tasarlanmış olan yöntemlerdir. Bir önceki bölümde gözlemlendiği üzere çizge modellerinin performansları iyi olmasına rağmen S1 stratejisinin getirdiği kazancı geçememektedir. Kolektif öğrenme yöntemiyle birlikte modellerin performanslarının iyileştirilmesi amaçlanmış ve tasarlanan modellerden daha iyi sonuçlar elde edilmesi beklenmektedir.

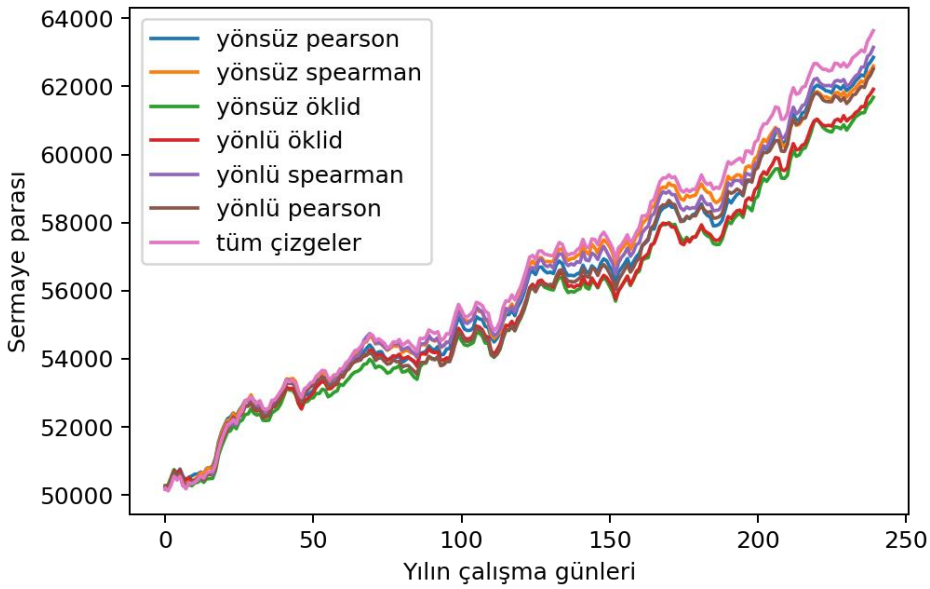


Şekil 5.3: Pearson çizge modelleri ve bu modellerin kolektif öğrenme yöntemiyle geliştirilmiş hallerinin 8 yıllık ortalama kazanç grafiği.

Çizelge 5.8 üzerinde kolektif öğrenme modellerinin Pearson Yönlü çizgeye uygulanması sonucunda elde edilen sonuçlar gösterilmiştir. Kolektif Yönsüz Pearson Çizge modelinin ortalama kazancı %25,12 ve Kolektif Yönlü Pearson Çizge modelinin ortalama kazancı ise %24,41 olmaktadır. Kolektif öğrenme modelleri Pearson çizge modelinin başarısını ciddi miktarda artırmaktadır. Kolektif öğrenme modelleri ile eğitilen Pearson çizge modellerinin başarısı S1 stratejisini geçmektedir. Böylelikle bu çizge yöntemleri hem daha az riskli olmakla beraber hem de S1 gibi

temel bir stratejinin üstünde kazanç getirmektedir. Şekil 5.3 üzerinde Kolektif öğrenme yöntemlerinin getirdiği kazanç açık bir şekilde gözükmektedir. Yılın özellikle belirli bir kısmından sonra birikimli olarak daha fazla kazanç sağlayan bu yöntemler çizge modellerini ciddi miktarda iyileştirmektedir.

Kolektif öğrenme modelleri tüm çizge modelleri üzerinde eğitilerek performansları Çizelge 5.9 üzerinde gösterilmiştir. Tüm çizge türlerinin performansları kendi içerisinde ciddi miktarda artmıştır. Bundan dolayı da kolektif öğrenme yönteminin çizge modelleri başarılı bir şekilde iyileştirdiği ve temel stratejilerden biri olan S1 stratejisini geçmesini sağladığı gözlemlenmiştir. Her bir çizge türü için eğitilen kolektif öğrenme yöntemlerinden en başarılı olan Yönsüz Pearson modeli ve Yönlü Spearman modelidir. Bu 2 modelin performansı diğer çizge modellerinin üstünde kalmaktadır. Tüm çizge modellerinin birleştirilmesiyle elde edilen kolektif öğrenme modeli tüm modellerin getirdiği kazançtan daha fazla bir kazanç sağlamaktadır. Bu da tüm çizgeleri birlikte eğitmenin modellerin performansını ciddi bir şekilde artırdığını ortaya çıkarmıştır. Bunun sebebi her bir çizge modeli belirli koşullarda başarılı olmaktadır ve hepsinin birleştirilmesi sonucunda daha iyi bir model elde edilmektedir. Şekil 5.4 kolektif öğrenme yöntemiyle elde edilen kazanç grafiğini göstermektedir. Grafikteki modeller zaman zaman birbirlerine performans olarak yakınlaşmış olsalar da tüm çizgelerin birlikte eğitildiği kolektif öğrenme yöntemi genel olarak üstte kalmaktadır.



Şekil 5.4: Kolektif öğrenme modellerinin tüm çizge türleri için eğitilmesi sonucunda elde edilen yıllık ortalama kazanç grafiği.

Çizelge 5.8: Pearson çizge modelleri ve bu modellerin kolektif öğrenme yöntemiyle geliştirilmesiyle elde edilen sonuçlar tablosu.

	Yönsüz Pearson	Yönlü Pearson	Kolektif Yönsüz Pearson	Kolektif Yönlü Pearson	S1	DOW30 indeks
2019	22,31	26,26	32,26	31,34	23,6	24,2
2018	0,97	2,58	2,03	-2.33	10,8	1,0
2017	25,26	22,43	25,25	29,58	72,7	25,0
2016	15,04	15,67	28,76	26,96	1,5	16,0
2015	2,25	4,81	9,79	13,82	-2,4	4,3
2014	19,49	19,83	26,20	24,91	-2,6	15,4
2013	34,23	35,89	46,64	37,33	47,9	29,2
2012	21,50	18,98	30,42	33,73	31,2	15,1
Ortalama	17,63	18,30	25,12	24,41	22,8	16,2

Çizelge 5.9: Kolektif öğrenme modellerinin tüm çizge türleri için eğitilmesi sonucu elde edilen kazanç tablosu.

	Yönsüz Pearson	Yönlü Pearson	Yönsüz Spearman	Yönlü Spearman	Yönsüz Öklid	Yönlü Öklid	Tüm Çizgeler
2019	32,26	31,34	28,65	26,29	26,54	27,25	33,82
2018	2,03	-2.33	-3,99	-3,79	-4,36	-4,38	3,62
2017	25,25	29,58	28,49	25,92	28,48	26,03	26,08
2016	28,76	26,96	27,54	35,26	29,73	30,66	35,34
2015	9,79	13,82	14,65	14,39	8,42	11,24	13,54
2014	26,20	24,91	29,05	28,98	25,51	28,94	27,04
2013	46,64	37,33	44,10	47,37	38,67	43,94	49,50
2012	30,42	33,73	28,68	31,76	29,86	21,84	24,52
Ortalama	25,12	24,41	24,39	25,77	22,85	23,19	26,68

6. SONUÇ

Bu tez kapsamında MLP, MLP-Multi modelleri temel kıyaslama yöntemi olması amacıyla eğitilmiştir. Bunlara ek olarak S1, S2 ve DOW 30 indeksi temel stratejiler olarak kullanılmıştır. Bu tezin temel amacı çizgelerin zaman serisi şeklinde derin öğrenme yöntemleriyle eğitilmesi ve borsada bu temel stratejilerden daha başarılı modeller elde etmektir. Yapılan çalışmalarda çizge tabanlı yöntemlerin MLP, MLP-Multi, S2 ve DOW 30 indeksinden daha fazla kazanç sağladığı gösterilmiştir. Çizge tabanlı yöntemler S1 stratejisinin kazancını geçememesine rağmen daha az riskli davranışlarda bulunmasından dolayı gerçek hayatta tercih edilebilir.

Yönlü çizge gösterimi yönsüz çizge gösterimine göre daha başarılı sonuçlar vermektedir. Bunun sebebi yönlü çizge gösteriminin daha fazla bilgi içermesi olarak açıklanabilir. Yönlü çizge gösterimi yapılan birçok deneyde daha başarılı sonuçlar vermektedir. Dolayısıyla yönlü çizge gösterimi gerçek hayatta kullanılabilecek bir çizge oluşturma ve işleme yöntemidir.

Çizge tabanlı yöntemlerde tüm çizge türlerinin birleştirilmesiyle elde edilen model en başarılı model olmaktadır. Bu model birden çok çizgenin zamana göre davranışını analiz ettiğinden dolayı diğer modellerden daha iyi çalışmaktadır. Bir yatırımcı için bu tarz bir model kullanmak kazançlı olabilir.

Çizge tabanlı modellerin bir önemli avantajı borsanın azalan trende sahip olduğu yıllarda bile zarar etmemeleridir. Bundan dolayı yatırımcıların kullanabileceği modellerdir. MLP ve MLP-Multi modelleri performans olarak hem DOW 30 indeksinin altında bir kazanç sağlamakla birlikte hem de bazı yıllarda yatırımcıyı zarara sokmaktadır. Bundan dolayı çizge tabanlı modeller daha güvenilir modeller olmaktadır.

Çizge tabanlı modellerin iyileştirilmesi amacıyla kolektif öğrenme yöntemleri kullanılmıştır. Kolektif öğrenme yöntemleri çizge tabanlı modellerin performanslarını ciddi miktarda artırmıştır. Bu artış oranı S1 stratejisinin getirdiği

kazancı da geçmektedir. Bundan dolayı bir yatırımcı hem daha güvenilir olan bu yöntemi kullanırken hem de daha fazla kar elde etme imkanı bulmaktadır.

Bu çalışmada bir zaman serisi olan borsa üzerinde çizge tabanlı derin öğrenme yöntemleri kullanılmış ve başarı gösterilmiştir. Bu tez kapsamında yalnızca DOW 30 hisse senetleri kullanılmıştır. İlerleyen çalışmalarda farklı borsa türleri de kullanılarak çizge tabanlı bu modellerin performansları incelenebilir. Bu modellerin eğitilmesi ciddi bir işlem gücü gerektirdiğinden dolayı farklı veri setleri üzerinde çalıştırılmamıştır. Ayrıca her veri setinin kendine özgü bazı ek özellikleri bulunabilir ve modellere göre küçük değişiklikler ile uygulanması gerekebilir.

Bu çalışmada oluşturulan yönlü çizge modelleri maksimum korelasyonu seçme mantığı kullanılarak tasarlanmıştır. Gt hesaplanırken 30x30xMx3xT olacak şekilde bir çizge tensörü oluşturularak eğitim yapılabilir. Yani her t anında maksimum korelasyonu seçmek yerine kaydırılarak elde edilen tüm korelasyon kombinasyoları farklı bir boyut üzerinden verilebilir. Bu şekilde daha kapsamlı bir çizge üretilebilir.

Bu çalışmada çizgeden çıkarılan özniteliklerin modelin performansını iyileştirdiği gözlemlenmektedir fakat bu çizgeden çıkarılan öznitelikler açıklanamamıştır. Çizgedeki özniteliklerin anlamlandırılması ve çizge modelindeki hangi kenarların ya da düğümlerin ne kadar etkili olduğunun analiz edilmesi gelecekte yapılacak olan çalışmalar arasında gösterilebilir.

Bu tezde yapılan çalışmaya gelecekte uygulanabilecek ek bir yöntem ise verilerin etiketlenme şekli günlük olarak tasarlanmıştır. Düşük frekanslarda borsa gibi problemlerde gürültü miktarı artmaktadır. Bundan dolayı zaman serisinin kendisi ya da farklı hisse senetleri arasındaki ilişkiyi öğrenmek zor olabilir. Bundan dolayı farklı bir etiketleme tekniği ile bu tarz bir çizge yapısı gelecekte denenebilir.

KAYNAKLAR

[1] "geeksforgeeks." https://www.geeksforgeeks.org/graph-and-its-representations/ (accessed Nov. 08, 2021).

[2] **R. J. Hyndman and G. Athanasopoulos**, *Forecasting: principles and practice*. OTexts, 2018.

[3] "Stationary." https://hilalgozutok.medium.com/zaman-serisi-tahmini-time-series-forecasting-dd541fdd41af (accessed Dec. 24, 2021).

[4] "Classification Image." https://www.javatpoint.com/classification-algorithm-in-machine-learning (accessed Jan. 03, 2022).

[5] "Regression Image." https://www.javatpoint.com/regression-vs-classification-in-machine-learning (accessed Jan. 03, 2022).

[6] "Error types-1." https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23 (accessed Jan. 04, 2022).

[7] "Cross Entropy Error Figure." https://www.kaggle.com/viveknimsarkar/hands-on-guide-to-loss-functions (accessed Jan. 07, 2022).

[8] "Model Geçerleme." https://medium.com/data-science-tr/overfitting-underfitting-cross-validation-b47dfda0cf4e (accessed Jan. 08, 2022).

[9] "Overfitting and Underfitting." https://medium.com/kodluyoruz/makine-%C3%B6%C4%9Frenmesinde-overfitting-underfitting-best-fitting-kavramlar%C4%B1-9f80a5d42719 (accessed Jan. 09, 2022).

[10] "Grid Search ." https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-model-dogrulama-ve-hiper-parametre-secim-yontemleri-823812d95f3 (accessed Jan. 09, 2022).

[11] "Gradient descent vector." https://towardsdatascience.com/wondering-why-do-you-subtract-gradient-in-a-gradient-descent-algorithm-9b5aabdf8150 (accessed Jan. 09, 2022).

[12] "Gradient Descent ." https://blog.clairvoyantsoft.com/the-ascent-of-gradient-descent-23356390836f (accessed Jan. 09, 2022).

[13] **W. S. McCulloch and W. Pitts,** "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[14] **F. Rosenblatt,** "Perceptron simulation experiments," *Proceedings of the IRE*, vol. 48, no. 3, pp. 301–309, 1960.

[15] **B. Widrow and M. E. Hoff,** "Adaptive switching circuits," 1960.

[16] **I. Goodfellow, Y. Bengio, and A. Courville,** *Deep Learning*. MIT Press, 2016.

[17] “Perceptron.” https://ai.plainenglish.io/the-rise-and-fall-of-the-perceptron-c04ae53ea465 (accessed Apr. 25, 2022).

[18] “MLP figure”, Accessed: Jan. 22, 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/12/mlp-multilayer-perceptron-simple-overview/

[19] “Aktivasyon fonksiyonları”, Accessed: Jan. 22, 2022. [Online]. Available: https://medium.com/analytics-vidhya/activation-functions-all-you-need-to-know-355a850d025e

[20] “CNN”, Accessed: Jan. 22, 2022. [Online]. Available: https://purnasaigudikandula.medium.com/a-beginner-intro-to-convolutional-neural-networks-684c5620c2ce

[21] “CNN Conv layer.” https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-1-convolution-operation (accessed Jan. 22, 2022).

[22] “CNN Pooling Layer.” https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/ (accessed Jan. 23, 2022).

[23] “RNN-LSTM model.” https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed Jan. 23, 2022).

[24] “GCN.” https://jonathan-hui.medium.com/graph-convolutional-networks-gcn-pooling-839184205692 (accessed Jan. 23, 2022).

[25] “Ensemble Learning.” https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205 (accessed Jan. 24, 2022).

[26] **D. Matsunaga, T. Suzumura, and T. Takahashi,** “Exploring graph neural networks for stock market predictions with rolling window analysis,” *arXiv preprint arXiv:1909.10660*, 2019.

[27] **R. van den Berg, T. N. Kipf, and M. Welling,** “Graph convolutional matrix completion,” *arXiv preprint arXiv:1706.02263*, 2017.

[28] **V. Garcia and J. Bruna,** “Few-shot learning with graph neural networks,” *arXiv preprint arXiv:1711.04043*, 2017.

[29] **A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann,** “Netgan: Generating graphs via random walks,” in *International Conference on Machine Learning*, 2018, pp. 610–619.

[30] **A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur,** “Protein interface prediction using graph convolutional networks,” *Adv Neural Inf Process Syst*, vol. 30, 2017.

[31] **S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley,** “Molecular graph convolutions: moving beyond fingerprints,” *J Comput Aided Mol Des*, vol. 30, no. 8, pp. 595–608, 2016.

[32] **A. Sanchez-Gonzalez *et al.*,** “Graph networks as learnable physics engines for inference and control,” in *International Conference on Machine Learning*, 2018, pp. 4470–4479.

[33] **Z. Zhang, P. Cui, and W. Zhu,** “Deep learning on graphs: A survey,” *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[34] **K.-H. N. Bui, J. Cho, and H. Yi,** "Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues," *Applied Intelligence*, pp. 1–12, 2021.

[35] **Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang,** "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 753–763.

[36] **J. Zhou *et al.*,** "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[37] **L. Zhao *et al.*,** "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[38] **P. Ghosh, Y. Yao, L. Davis, and A. Divakaran,** "Stacked spatio-temporal graph convolutional networks for action segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 576–585.

[39] **S. F. Tekin and S. S. Kozat,** "Crime Prediction with Graph Neural Networks and Multivariate Normal Distributions," *arXiv preprint arXiv:2111.14733*, 2021.

[40] **T. Bian *et al.*,** "Rumor detection on social media with bi-directional graph convolutional networks," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, vol. 34, no. 01, pp. 549–556.

[41] **D. Chai, L. Wang, and Q. Yang,** "Bike flow prediction with multi-graph convolutional networks," in *Proceedings of the 26th ACM SIGSPATIAL international conference on advances in geographic information systems*, 2018, pp. 397–400.

[42] **M. Weber *et al.*,** "Anti-money laundering in bitcoin Experimenting with graph convolutional networks for financial forensics," *arXiv preprint arXiv1908.02591*, 2019.

[43] **C. Liu, Y. Jin, K. Xu, G. Gong, and Y. Mu,** "Beyond Short-Term Snippet: Video Relation Detection With Spatio-Temporal Global Context," Jun. 2020.

[44] **A. Bahrammirzaee,** "A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems," *Neural Computing and Applications*, vol. 19, no. 8, pp. 1165–1195, 2010.

[45] **D. Zhang and L. Zhou,** "Discovering golden nuggets: data mining in financial application," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 4, pp. 513–522, 2004.

[46] **A. Mochón, D. Quintana, Y. Sáez, and P. Isasi,** "Soft computing techniques applied to finance," *Applied Intelligence*, vol. 29, no. 2, pp. 111–115, 2008.

[47] **L. Tesfatsion and K. L. Judd,** *Handbook of computational economics: agent-based computational economics*. Elsevier, 2006.

[48] **S. K. Chalup and A. Mitschele,** "Kernel methods in finance," in *Handbook on information technology in finance*, Springer, 2008, pp. 655–687.

[49] **T. J. Strader, J. J. Rozycki, T. H. Root, and Y.-H. J. Huang,** "Machine learning stock market prediction studies: Review and research directions," *Journal of International Technology and Information Management*, vol. 28, no. 4, pp. 63–83, 2020.

[50] **M. Sedighi, H. Jahangirnia, M. Gharakhani, and S. Farahani Fard,** "A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine," *Data (Basel)*, vol. 4, no. 2, p. 75, 2019.

[51] **S. Sefiane and M. Benbouziane,** "Portfolio selection using genetic algorithm," 2012.

[52] **J. Nobre and R. F. Neves,** "Combining principal component analysis, discrete wavelet transform and XGBoost to trade in the financial markets," *Expert Systems with Applications*, vol. 125, pp. 181–194, 2019.

[53] **K. Alkhatib, H. Najadat, I. Hmeidi, and M. K. A. Shatnawi,** "Stock price prediction using k-nearest neighbor (kNN) algorithm," *International Journal of Business, Humanities and Technology*, vol. 3, no. 3, pp. 32–44, 2013.

[54] **M. Kumar and M. Thenmozhi,** "Forecasting stock index movement: A comparison of support vector machines and random forest," 2006.

[55] **M. R. Machado, S. Karray, and I. T. de Sousa,** "LightGBM: An effective decision tree gradient boosting method to predict customer loyalty in the finance industry," in *2019 14th International Conference on Computer Science & Education (ICCSE)*, 2019, pp. 1111–1116.

[56] **B. Fünfgeld and M. Wang,** "Attitudes and behaviour in everyday finance: evidence from Switzerland," *International Journal of Bank Marketing*, 2009.

[57] **X. Wang,** "On the effects of dimension reduction techniques on some high-dimensional problems in finance," *Operations Research*, vol. 54, no. 6, pp. 1063–1078, 2006.

[58] **M. Gallegati and W. Semmler,** "Wavelet applications in economics and finance," 2014.

[59] **M. M. Kumbure, C. Lohrmann, P. Luukka, and J. Porras,** "Machine learning techniques and data for stock market forecasting: a literature review," *Expert Systems with Applications*, p. 116659, 2022.

[60] **A. M. Ozbayoglu, M. U. Gudelek, and O. B. Sezer,** "Deep learning for financial applications: A survey," *Applied Soft Computing*, vol. 93, p. 106384, 2020.

[61] **S. Karaoglu, U. Arpaci, and S. Ayvaz,** "A deep learning approach for optimization of systematic signal detection in financial trading systems with big data," *International Journal of Intelligent Systems and Applications in Engineering, SpecialIssue (SpecialIssue)*, pp. 31–36, 2017.

[62] **W. Bao, J. Yue, and Y. Rao,** "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS One*, vol. 12, no. 7, p. e0180944, 2017.

[63] **S. Liu, C. Zhang, and J. Ma,** "CNN-LSTM neural network model for quantitative strategy analysis in stock markets," in *international conference on neural information processing*, 2017, pp. 198–206.

[64] **T. Fischer and C. Krauss,** "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[65] **M. Mourelatos, C. Alexakos,** T. Amorgianiotis, and S. Likothanassis, "Financial indices modelling and trading utilizing deep learning techniques: the ATHENS SE FTSE/ASE large cap use case," in *2018 Innovations in Intelligent Systems and Applications (INISTA)*, 2018, pp. 1–7.

[66] **B. X. Yong, M. R. A. Rahim, and A. S. Abdullah,** "A stock market trading system using deep neural network," in *Asian Simulation Conference*, 2017, pp. 356–364.

[67] **O. B. Sezer, M. Ozbayoglu, and E. Dogdu,** "A deep neural-network based stock trading system based on evolutionary optimized technical analysis parameters," *Procedia Comput Sci*, vol. 114, pp. 473–480, 2017.

[68] **J. Sirignano and R. Cont,** "Universal features of price formation in financial markets: perspectives from deep learning," *Quantitative Finance*, vol. 19, no. 9, pp. 1449–1459, 2019.

[69] **A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis,** "Using deep learning to detect price change indications in financial markets," in *2017 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 2511–2515.

[70] **M. U. Gudelek, S. A. Boluk, and A. M. Ozbayoglu,** "A deep learning based stock trading model with 2-D CNN trend detection," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2017, pp. 1–8.

[71] **G. Hu *et al.*,** "Deep stock representation learning: From candlestick charts to investment decisions," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2018, pp. 2706–2710.

[72] **A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis,** "Forecasting stock prices from the limit order book using convolutional neural networks," in *2017 IEEE 19th Conference on Business Informatics (CBI)*, 2017, vol. 1, pp. 7–12.

[73] **H. Gunduz, Y. Yaslan, and Z. Cataltepe,** "Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations," *Knowledge-Based Systems*, vol. 137, pp. 138–148, 2017.

[74] **E. J. Ruiz, V. Hristidis, C. Castillo, A. Gionis, and A. Jaimes,** "Correlating financial time series with micro-blogging activity," in *Proceedings of the fifth ACM international conference on Web search and data mining*, 2012, pp. 513–522.

[75] **D. Xiao and J. Wang,** "Graph based and multifractal analysis of financial time series model by continuum percolation," *International Journal of Nonlinear Sciences and Numerical Simulation*, vol. 15, no. 5, pp. 265–277, 2014.

[76] **L. Wang, Z. Wang, S. Zhao, and S. Tan,** "Stock market trend prediction using dynamical Bayesian factor graph," *Expert Systems with Applications*, vol. 42, no. 15–16, pp. 6267–6275, 2015.

[77] **Y. Long,** "Visibility graph network analysis of gold price time series," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 16, pp. 3374–3384, 2013.

[78] **J. Long, Z. Chen, W. He, T. Wu, and J. Ren,** "An integrated framework of deep learning and knowledge graph for prediction of stock price trend: An application in Chinese stock exchange market," *Applied Soft Computing*, vol. 91, p. 106205, 2020.

[79] **R. Flanagan and L. Lacasa,** "Irreversibility of financial time series: a graph-theoretical approach," *Physics Letters A*, vol. 380, no. 20, pp. 1689–1697, 2016.

[80] **D. F. Ahelegbey, L. Carvalho, and E. Kolaczyk,** "A Bayesian Covariance Graph And Latent Position Model For Multivariate Financial Time Series," *Available at SSRN 3090236*, 2020.

[81] **Y. Xiu, G. Wang, and W. K. V. Chan,** "Crash Diagnosis and Price Rebound Prediction in NYSE Composite Index Based on Visibility Graph and Time-Evolving Stock Correlation Network," *Entropy*, vol. 23, no. 12, p. 1612, 2021.

[82] **H. Liu and B. Song,** "Stock price trend prediction model based on deep residual network and stock price graph," in *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, 2018, vol. 2, pp. 328–331.

[83] **S. Jeon, B. Hong, and V. Chang,** "Pattern graph tracking-based stock price prediction using big data," *Future Generation Computer Systems*, vol. 80, pp. 171–187, 2018.

[84] **C. K.-S. Leung, R. K. MacKinnon, and Y. Wang,** "A machine learning approach for stock price prediction," in *Proceedings of the 18th International Database Engineering & Applications Symposium*, 2014, pp. 274–277.

[85] **Y. Chen, Z. Wei, and X. Huang,** "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1655–1658.

[86] **D. Loe, S. Chang, and J. Chau,** "Stock Market Movement Prediction Using Graph Convolutional Networks".

[87] **W. Chen, M. Jiang, W.-G. Zhang, and Z. Chen,** "A novel graph convolutional feature based convolutional neural network for stock trend prediction," *Information Sciences*, vol. 556, pp. 67–94, 2021.

[88] **X. Ying, C. Xu, J. Gao, J. Wang, and Z. Li,** "Time-aware graph relational attention network for stock recommendation," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 2281–2284.

[89] **P. Patil, C.-S. M. Wu, K. Potika, and M. Orang,** "Stock market prediction using ensemble of graph theory, machine learning and deep learning models," in *Proceedings of the 3rd International Conference on Software Engineering and Information Management*, 2020, pp. 85–92.

[90] **J. M.-T. Wu, Z. Li, G. Srivastava, M.-H. Tasi, and J. C.-W. Lin,** "A graph-based convolutional neural network stock price prediction with leading

indicators," *Software: Practice and Experience*, vol. 51, no. 3, pp. 628–644, 2021.

[91] **J. R. Abrams, J. Celaya-Alcalá, D. Baldwin, R. Gonda, and Z. Chen,** "Analysis of equity markets: A graph theory approach," *Society for Industrial and Applied Mathematics. doi*, vol. 10, 2016.

[92] **Y. Wang, C. Zhang, S. Wang, S. Y. Philip, L. Bai, and L. Cui,** "Deep co-investment network learning for financial assets," in *2018 IEEE International Conference on Big Knowledge (ICBK)*, 2018, pp. 41–48.