
GAZİ ÜNİVERSİTESİ
Gazi
1926

DERİN ÖĞRENME İLE DÖVİZ KURLARI DEĞİŞİMLERİNİN ZAMAN SERİSİ ANALİZİ

Büşra AKSEL

YÜKSEK LİSANS TEZİ
İSTATİSTİK ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2023

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Büşra AKSEL

13 / 06 / 2023

DERİN ÖĞRENME İLE DÖVİZ KURLARI DEĞİŞİMLERİNİN ZAMAN SERİSİ ANALİZİ

(Yüksek Lisans Tezi)

Büşra AKSEL

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2023

ÖZET

Finansal piyasalar içerisinde döviz kurları ülkelerin ekonomik gelişiminde önemli bir rol oynar. Döviz kurlarındaki hareketlilik araştırmacılar içinde ilgi çekici bir alan olmuştur. Döviz kurlarındaki değişim doğrusal ve durağan olmadığından doğru bir şekilde tahmin etmek oldukça zordur. Finansal verilerin fiyat tahmini için farklı tahmin teknikleri geliştirilmiştir. Zaman serisi modellemesi ve tahmini, finansal veri analizi sürecinde önemli bir rol oynar. Günümüzde teknolojinin gelişimiyle birlikte ortaya çıkan derin öğrenme modelleri görüntü sınıflandırma, ses tanıma, otomatik tercüme, hisse senede fiyat tahmini gibi birçok farklı alanda sıklıkla kullanılmaktadır. Bu çalışmada, ele alınan döviz kurları üzerinden fiyat hareketlerini tahmin etmede Genelleştirilmiş Otoregresif Koşullu Değişken Varyans (GARCH), Tekrarlayan Sinir Ağları (Recurrent Neural Networks- RNN) ve Uzun Kısa Vadeli Bellek (Long Short Term Memory- LSTM) modelleri kullanılmıştır. Çalışmada kullanılan USD/TRY ve EUR/TRY veri setleri Yahoo Finance sitesinden elde edilmiştir ve döviz kurlarının kapanış fiyat değerleri kullanılmıştır. Veri setleri 3 Ocak 2015'ten 31 Aralık 2020'ye kadar 4176 veri içermektedir. Modelin tahmin performansını değerlendirmek için istatistiksel ölçütler kullanılmıştır. USD/TRY ve EUR/TRY veri setleri üzerinden elde edilen GARCH, RNN ve LSTM modellerinin tahminleri performans değerlendirme metrikleriyle ölçülmüş ve kullanılan modellerin performansları karşılaştırılmıştır.

Bilim Kodu	:	20517
Anahtar Kelimeler	:	Zaman serileri, yapay zekâ, derin öğrenme
Sayfa Adedi	:	81
Danışman	:	Prof. Dr. H. Hasan ÖRKCÜ
İkinci Danışman	:	Prof. Dr. Reşat KASAP

TIME SERIES ANALYSIS OF EXCHANGE RATE CHANGES WITH DEEP LEARNING

(M. Sc. Thesis)

Büşra AKSEL

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2023

ABSTRACT

Exchange rates in financial markets play an important role in the economic development of countries. The mobility in exchange rates has been an interesting area for researchers. Since the change in exchange rates is not linear and stationary, it is quite difficult to accurately predict. Different estimation techniques have been developed for price estimation of financial data. Time series modeling and forecasting plays an important role in the financial data analysis process. Today, deep learning models that emerged with the development of technology are frequently used in many different areas such as image classification, voice recognition, automatic translation, stock price prediction. In this study, Generalized Autoregressive Conditional Variable Variance (GARCH), Recurrent Neural Networks (RNN) and Long Short Term Memory (LSTM) models were used to predict price movements over the exchange rates discussed. The USD/TRY and EUR/TRY data sets used in the study were obtained from the Yahoo Finance site and the closing price values of the exchange rates were used. The datasets contain 4176 data from January 3, 2015 to December 31, 2020. Statistical measures were used to evaluate the prediction performance of the model. Estimates of GARCH, RNN and LSTM models obtained from USD/TRY and EUR/TRY datasets were measured with performance evaluation metrics and the performances of the models used were compared.

Science Code	:	20517
Key Words	:	Time series, artificial intelligence, deep learning
Page Number	:	81
Supervisor	:	Prof. Dr. H. Hasan ÖRKCÜ
Co-Supervisor	:	Prof. Dr. Reşat KASAP

TEŞEKKÜR

Tez çalışmam boyunca kıymetli katkıları ve destekleri ile bana yol gösteren değerli danışman hocam sayın Prof. Dr. H. Hasan ÖRKCÜ'ye ve kıymetli bilgi birikimleri ile desteğini esirgemeyen değerli eş danışman hocam sayın Prof. Dr. Reşat KASAP'a teşekkür ve saygılarımı sunarım. Ayrıca çalışmam boyunca destek olan değerli çalışma arkadaşlarıma teşekkür ederim.

Hayatım boyunca gösterdikleri sevgi, verdikleri emek ve destekleriyle beni yalnız bırakmayan canım annem Yeternaz AKSEL'e, sevgili babam Nuri AKSEL'e ve aileme tüm kalbimle teşekkür ederim. Son olarak her zaman yanımda olan ve bana inanan arkadaşlarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
μ	Ortalama
σ^2	Varyans
γ_k	Otokovaryans fonksiyonu
ρ_k	Otokorelasyon fonksiyonu
ε_t	Hata terimi
Ψ_{t-i}	Bilgi kümesi
b	Yan değeri
η	Öğrenme Oranı
β	Momentum terimi
W	Ağırlık katsayısı

Kısaltmalar	**Açıklamalar**
ACF	Otokorelasyon Fonksiyonu
ADF	Genişletilmiş Dickey Fuller
AdaGrad	Uyarlanabilir Gradyan Algoritması
Adam	Adaptif Moment Tahmini
AIC	Akaike Bilgi Kriteri
AR	Otoregresif
ARMA	Otoregresif Hareketli Ortalama
ARIMA	Otoregresif Entegre Hareketli Ortalama
ARCH	Otoregresif Koşullu Değişen Varyans
ANN	Yapay Sinir Ağları
BIC	Bayes Bilgi Kriteri
CNN	Evrişimli Sinir Ağları
EGARCH	Üstel Genelleştirilmiş Otoregresif Koşullu Değişen Varyans

Kısaltmalar	Açıklamalar
EKK	En Küçük Kareler
ERNN	Elman Yinelemeli Sinir Ağı
GAN	Üretken Çekişmeli Ağlar
GARCH	Genelleştirilmiş Otoregresif Koşullu Değişen Varyans
HMAE	Değişen Varyans Ayarlı Ortalama Mutlak Hata
HMSE	Değişen Varyans Ayarlı Hata Kareler Ortalaması
LSTM	Uzun Kısa Vadeli Bellek
MA	Hareketli Ortalamalar
MAE	Ortalama Mutlak Hata
MAPE	Ortalama Mutlak Hata Yüzdesi
MCC	Matthews Korelasyon Katsayısı
MLP	Çok Katmanlı Algılayıcı
MSE	Hata Kareler Ortalaması
PACF	Kısmi otokorelasyon fonksiyonu
ReLU	Doğrultulmuş Doğrusal Birimler
RMSE	Hata Kareler Ortalamasının Karekökü
RMSProp	Kareler Ortalaması Karekökünün Yayılımı
RNN	Tekrarlayan Sinir Ağı
SGD	Stokastik Gradyan İniş
SLP	Tek Katmanlı Algılayıcı
SVM	Destek Vektör Makineleri
TIC	Theil Eşitsizlik Katsayısı
TGARCH	Eşik Değerleri Genelleştirilmiş Otoregresif Koşullu Değişen Varyans

1. GİRİŞ

Finansal piyasalar ülkelerin ekonomileri için büyük öneme sahiptir. Finansal piyasalar içerisindeki döviz kurları, ülke ekonomilerinin temel bileşenidir ve ülkelerin ekonomik gelişimini etkilemektedir. Finansal zaman serisi analizinin temel amacı, tarihsel bilgiler arasında ilişki kurmak veri yapısını analiz ederek uygun bir model geliştirmek ve gelecekte belirli bir döneme ait davranışını tahmin etmektir (Escudero, Alcocer ve Paredes, 2021). Bir döviz kurunun geçmiş fiyat değerleri, gelecekte hangi noktada olabileceğine dair bilgi verir. Bu sebeple döviz kurunu zamanında kavramak ve fiyatını tahmin etmek yatırımcılar, şirketler, araştırmacılar ve akademisyenler için önemli ve ilgi çekici bir alandır (Wang, Wang, Li ve Wang, 2021). Finansal zaman serileri birtakım özellikler taşımaktadır. Genel olarak döviz piyasalarının durağan olmadığı ve değişen varyansa sahip olduğu görülmektedir. Döviz piyasalarında meydana gelen ani değişimler ve beklenmedik durumlar, oynaklığa yol açabilmektedir. Bu özellikleri sebebiyle finansal zaman serilerini tahmin etmek oldukça zor bir problem haline gelmiştir (Gao, 2021).

Finansal piyasalardaki bu hareketlilikler finansal zaman serilerinin modellenmesine ilişkin çeşitli teknikleri de beraberinde getirmiştir. Klasik istatistiksel zaman serisi modellerinin yanı sıra günümüzde teknolojinin gelişmesiyle beraber çeşitli yapay zekâ modelleri ortaya çıkmıştır. Son dönemdeki gelişmeler ile piyasa işlemleri ve finansal verilerin analizi gibi karmaşık problemlerinde çözümünde derin öğrenme modelleri sıklıkla tercih edilmektedir (Cao, Li ve Li, 2019; Escudero vd., 2021; Gao, 2021; Lu, Li, Li, Sun ve Wang, 2020).

Literatürde yer alan zaman serisi ve derin öğrenme modelleri ile ilgili çalışmalardan bazıları aşağıda verilmektedir:

Kızılsu ve diğerleri (2001), çalışmalarında yıllık Gayrisafi Milli Hasıla (GSMH) büyüme hızı, aylık dönem sonu İMKB-100 bileşik endeksi ve Esbank günlük repo oranları ekonometrik serilerine ait verileri Otoregresif Koşullu Değişen Varyans (ARCH) ve Genelleştirilmiş Otoregresif Koşullu Değişen Varyans (GARCH) modelleri ve türevlerini kullanarak ile analiz etmişlerdir. Veriler için gerekli dönüşümler uygulanarak uygun modeller bulunmuş ve analizler yapılmıştır (Kızılsu, Aksoy ve Kasap, 2001).

Songül (2010), çalışmasında dolar ve euro döviz kurlarını için ARCH/GARCH modellemesi yapmıştır. Çalışma 23.02.2001 ile 31.12.2009 tarihleri arasında 2221 veriyi kapsamaktadır. Döviz kuru getiri serilerinin betimleyici istatistikleri sunulmuştur ve yapılan analizler sonucunda en uygun modeller dolar ve euro getiri serisi için belirlenmiştir. ARCH/GARCH modelleri kullanılarak kurların fiyat tahminleri gerçekleştirilmiştir. Yapılan analizler sonucunda en uygun modeller dolar getiri serisi için AR(2)-EGARCH(1, 1, 1), Euro getirisinin ise AR(2)-TARCH(1, 1, 1) modelleri olarak elde edilmiştir (Songül, 2010).

Kim ve Won (2018), yaptıkları çalışmada KOSPI 200 hisse senedi volatilitesini tahmin etmek için derin öğrenme ve GARCH modellerini kullanmışlardır. Modellerin performanslarını karşılaştırmak için Ortalama Mutlak Hata (MAE), Hata Kareler Ortalaması (MSE), Değişen Varyans Ayarlı Ortalama Mutlak Hata (HMAE) ve Değişen Varyans Ayarlı Hata Kareler Ortalaması (HMSE) metriklerini kullanmışlardır. Analiz sonucunda GARCH modelleri arasında en iyi model EGARCH modeli olarak elde edilmiştir. Uzun Kısa Vadeli Bellek (Long Short Term Memory- LSTM) ve GARCH modellerinin kıyaslamasında ise LSTM modelinin daha iyi performans sergilediği gözlemlenmiştir (Kim ve Won, 2018).

Laily ve diğerleri (2018), BRI hisse senedinin günlük kapanış fiyatlarını kullanarak ARCH/GARCH modeli ile Elman Yinelemeli Sinir Ağı (ERNN) modeli arasındaki en iyi modeli belirlemeye çalışmışlardır. Çalışmada kullanılan veri seti 1 Ocak 2015'ten 17 Şubat 2017 yılları arasındaki günlük kapanış fiyat değerlerini içermektedir. Değerlendirme kriteri olarak MSE kullanılmıştır ve analiz sonuçlarına göre GARCH modeli ERNN modeline göre daha iyi performans sergilemiştir (Laily, Warsito ve Maruddani, 2018).

Topaloğlu (2019), çalışmasında dolar, euro ve sterlin döviz kuru serilerini GARCH kullanarak modelleme yapmıştır. Çalışmada 05.01.2000 ile 13.09.2018 dönemine ait döviz kuru verileri kullanılmıştır. Çalışmada serilerin tanımlayıcı istatistikleri incelenmiş ve farklı modeller denenerek en uygun model araştırılmıştır. Döviz kuru serileri için modellerin belirlenmesinde Theil Eşitsizlik Katsayısı (TIC), Hata Kareler Ortalamasının Karekökü (RMSE) ve MAE katsayıları hesaplanmıştır. Yapılan analizler sonucunda dolar döviz kuru serisi için GARCH(1, 2), euro döviz kuru serisi için EGARCH(1, 2) ve sterlin döviz kuru serisi için IGARCH(1, 1) modeli en uygun model olarak belirlenmiştir (Topaloğlu, 2019).

Alpay (2020), çalışmasında USD/TRY fiyat tahmini için LSTM mimarisini kullanılmıştır. Çalışmada kullanılan veri seti 01.01.2000 ile 31.12.2017 tarihleri arasındadır. Veri seti, eğitim ve test verisi olarak ikiye ayrılmış ve eğitim verisi olarak, 01.01.2000 ile 31.12.2016 tarihleri arası 01.01.2017 ile 31.12.2017 tarihleri arası test verisi olarak kullanılmıştır. Çalışmada kurulan derin öğrenme modeli 4 LSTM ve 1 yoğunluk katmanından oluşmaktadır. Her LSTM katmanı 50 nöron içermektedir ve modelin aşırı uydurmasını engellemek için katmanlar arasında 0,2 oranında seyreltme uygulanmıştır. Uygulamada 1, 10, 100 devir sayısı değerlerinde 256 ve 512 yığın sayısı değerleri kullanılarak eğitim işlemi yapılmıştır. Sonuçlara göre devir sayısının artması modeli daha başarılı yapmıştır (Alpay, 2020).

Kaya ve diğerleri (2020), Covid-19 öncesi ve sonrasındaki Bitcoin fiyat değişimlerini zaman serisi analizi, makine öğrenmesi ve derin öğrenme yöntemleriyle değerlendirmesini yapmışlardır. Çalışmada koronavirüs öncesini kapsayan 5 Şubat 2018 ile 27 Ekim 2019 tarihleri arasında BTC/USD kurundaki haftalık kapanış fiyatları baz alınarak bir veri kümesi oluşturulmuştur. Koronavirüs pandemisinin sonrasını kapsayan 5 Şubat 2018 ile 28 Haziran 2020 tarihleri arasındaki BTC/USD haftalık kapanış fiyatları baz alınarak ikinci bir veri kümesi oluşturulmuştur. Bu veri kümelerine Otoregresif bütünleşik hareketli ortalama süreci (Autoregressive integrated moving average- ARIMA), Destek Vektör Makineleri (Support Vector Machines-SVM) ve LSTM modelleri uygulamıştır. Modellerin tahmin başarısını ölçmek amacıyla karmaşıklık matrisi, kesinlik, hassasiyet, F1 skor, RMSE, MAPE, Matthews Korelasyon Katsayısı (MCC) ve doğruluk ölçütleri kullanılmıştır. Yapılan çalışmada pandemi öncesi verilerde SVM, pandemi sonrası verilerle ise ARIMA en başarılı sonuçları vermiştir (Kaya, Akba, Medeni ve Medeni, 2020).

Dobrovolny ve diğerleri (2020), Tekrarlayan Sinir Ağlarından olan LSTM kullanarak FOREX için fiyat tahmin işlemi gerçekleştirmiştir. Günlük FOREX verilerine dayanarak tahmin işlemini gerçekleştirmek amacıyla en iyi zaman zaman bloğunu bulmak ve önermek için test etmektedir. Eğitim veri seti, 01.04.1971'den 09.05.2019'a kadar günlük döviz kuru verilerini içermektedir. Girdi veri setini 10 ila 60 gün arasında dilimlere ayırıp tahminler yapılmıştır. Tahmin doğrulaması MSE ve MAE ile hesaplanmıştır. En iyi performans gösteren ağ 30 günlük süre ve 100 devir sayısı için eğitilmiştir (Dobrovolny, Soukal, Lim, Selamat ve Krejcar, 2020).

Lu ve diğerleri (2020), hisse senedi fiyat tahmin için Çok Katmanlı Algılayıcı (Multilayer Perceptron- MLP), Evrişimli Sinir Ağları (Convolutional Neural Networks- CNN), Tekrarlayan Sinir Ağı (Recurrent Neural Network- RNN), LSTM, CNN-RNN ve CNN-LSTM modellerini kullanmışlardır. Bu araştırmada kullanılan veriler, 7127 işlem günü dahil olmak üzere 1 Temmuz 1991'den 31 Ağustos 2020'ye kadar olan günlük hisse senedi fiyatları ile ilgilidir. Modellerin tahmin sonuçlarını karşılaştırmak için MAE, RMSE ve R^2 değerlendirme kriterleri olarak kullanılmıştır (Lu vd., 2020).

Gao (2021), çalışmasında ARIMA ve derin öğrenme modelleri kullanarak Dow Johns Endüstriyel Ortalamasında listelenen 30 hisse senedi için fiyat tahmin etmede farklı zaman serisi modellerinin performansını karşılaştırmaktadır. ARIMA, Yapay Sinir Ağı (Artificial Neural Network- ANN), RNN ve LSTM ve Seq2seq modelleri ile tahminler gerçekleştirilmiştir. Çalışmada kullanılan zaman serisi modelleri için ele alınan veri seti Ocak 2016'dan Aralık 2017'ye kadardır. Modellerin tahminlerini karşılaştırmak için MSE istatistiği kullanılmıştır (Gao, 2021).

Escudero ve diğerleri (2021), çalışmalarında farklı modeller kullanarak EUR/USD döviz kuruna ilişkin fiyat tahmini yapmışlardır. Bu amaçla ARIMA, Elman tipinde Tekrarlayan Sinir Ağı (ERNN) ve LSTM modelleri kullanılmıştır. Analiz edilen dönem 2 Ocak 1998 'den 31 Aralık 2019'a kadardır ve toplam 5737 gözlem içermektedir. Farklı zaman adımları kullanarak tahmin işlemi yapılmıştır. Kullanılan üç tahmin modelini karşılaştırmak için performans değerlendirme metrikleri, model doğruluklarını karşılaştırmak için ise Theil'in U katsayısı kullanılmıştır. Yapılan çalışmasının sonucunda ARIMA'nın sabit sayılar ürettiği, kısa vadede LSTM'nin, uzun vadede ise ERNN'nin en iyi tahminleri sağladığı görülmüştür (Escudero vd., 2021).

Taş ve diğerleri (2021), S&P 500 endeksinin derin öğrenme ve yapay sinir ağı yöntemleri kullanarak fiyat tahmini çalışması yapmıştır. Uygulamada 12.08.2000 ile 13.08.2020 tarihleri arasındaki günlük fiyat verileri kullanılmıştır. İlk 19 yılı yani veri setinin %95'i eğitim, son 1 yılı yani veri setinin %5'i test olacak şekilde ayrılmıştır. LSTM ve MLP yöntemleri kullanarak tahmin işlemi gerçekleştirilmiştir. LSTM mimarisi 300 nöron içeren iki katmandan oluşmaktadır. Yığın sayısı 120, devir sayısı 1000 ve seyreltme değeri 0,2 olarak tercih edilmiştir. Sığ MLP mimarisinde ilk katman 15 nöron, ikinci katman 30 olarak iki katmanlı bir şekilde tasarlanmıştır ve devir sayısı 100 olarak belirlenmiştir. Çalışmada

tahmin hatalarını belirlemek için RMSE ve MSE değerleri kullanılmıştır. Elde edilen sonuçlara göre kullanılan iki yönteminde birbirine yakın sonuçlar verdiği gözlemlenmiştir (Taş, Gülüm ve Tulum, 2021).

Abar (2021), çalışmasında BİST100 endeksinin açılış verilerini kullanarak ARCH/GARCH ve LSTM modellerinin karşılaştırmasını yapmıştır. Çalışmada kullanılan veriler 2 Ocak 1995 ve 15 Eylül 2021 dönemi arasında 6673 güne aittir. Çalışmada verilerin %90'ı eğitim verisi, %10'u ise test verisi olarak ikiye ayrılmıştır. Zaman serisi modeli, ortalama denklemi için ARIMA(0, 0, 5) ve varyans denklemi için GARCH(2, 1) olarak kurulmuştur. Derin öğrenme modeli için; 128, 64 ve 32 nöron içeren üç LSTM katmanı ve 1 nöron içeren yoğunluk (dense) katmanı kullanılarak 4 katmanlı bir model kurulmuştur. Çalışmadan kayıp fonksiyonu olarak MSE, ve 0,000001 öğrenme oranıyla optimizasyon algoritması olarak Adam tercih edilmiştir. Girdi için zaman aralığı deneyler sonucu 2, devir sayısı 25, yığın sayısı 50 olarak belirlenmiştir. Çalışmada kurulan modellerden elde edilen öngörüleri karşılaştırmak amacıyla RMSE ve R^2 istatistikleri kullanılmıştır. Bulguların değerlendirilmesi sonucunda kurulan derin öğrenme modelinin daha başarılı olduğu görülmektedir (Hayri vd., 2021).

Wang ve diğerleri (2021), yaptıkları çalışmada MLP, CNN, RNN, LSTM, CNN-LSTM ve CNN-TLSTM modellerini kullanarak USD/CNY döviz kurunun fiyat tahminini gerçekleştirmişlerdir. Çalışmada kullanılan veri seti 2 Ocak 2006'dan 30 Ekim 2020'ye kadar günlük kapanış fiyat değerleri ele alınmıştır. Veri seti 2 Ocak 2006'dan 31 Aralık 2019'a kadar eğitim verisi, 1 Ocak 2020'den 30 Ekim 2020'ye kadar Çalışmada kullanılan modellerin performanslarını değerlendirmek ve karşılaştırmak için MAPE, MSE ve R^2 istatistikleri kullanılmıştır. Yapılan çalışma sonucunda CNN-TLSTM modeli daha iyi performans sergilemiştir (Wang vd., 2021).

Cheng ve diğerleri (2021), çalışmalarında alım satım stratejileri yapmak amacıyla hisse senedi fiyat hareketini öngörmek için hibrit bir makine öğrenimi modeli önermektedir. Çalışmada RNN, LSTM ve önerilen derin sinir ağı modelleri beş farklı hisse senetleri verileri üzerinde uygulanmıştır. Elde edilen sonuçlara bir ticaret stratejisine karar vermek için kullanılmıştır. Çalışmada değerlendirme ölçütleri olarak doğruluk, kesinlik, duyarlılık ve F1 değerleri hesaplanmıştır. Elde edilen sonuçlara göre beş farklı hisse senedinden dördü için LSTM modelinin daha iyi performans göstermiştir (Cheng, Huang, Hsieh ve Wu, 2021).

Liu ve diğerleri (2022), hisse senetleri volatilite modellemesi için GARCH ve LSTM yöntemlerini kullanmışlardır. Çalışmada iki modeli MAE ve MSE değerlendirme ölçütleri olarak kullanarak değerlendirmişlerdir. Elde edilen sonuçlara göre LSTM modeli GARCH modeline göre daha iyi performans göstermiştir (Liu, Jiang ve Lin, 2022).

Cengiz ve Yıldırım (2022) tarafından yapılan çalışmada, USD/TRY kurunun modellenmesi ve tahmin edilmesi için çeşitli ARIMA/GARCH yaklaşımları ele alınmıştır. Çalışma Ocak 2016'dan Aralık 2018'e kadar olan günlük veriler kullanılmıştır. Veri setinin yaklaşık olarak %80'i eğitim %20'si ise test verisi olarak ayrılmıştır. Çalışmada modellerin tahmin doğruluğunu değerlendirmek için MAE, MAPE, RMSE ve TIC ölçütleri kullanılmıştır (Yıldırım ve Cengiz, 2022).

Bu çalışmada, döviz kurlarının kapanış fiyat değerlerinin tahmininde istatistiksel zaman serisi yöntemi ve derin öğrenme yöntemleri uygulanmıştır. Döviz kuru verilerini modellemek için çeşitli ekonometrik modellerden GARCH modelleri ile derin öğrenme modellerinden RNN ve LSTM modelleri kullanılmıştır. Döviz kuru serileri modellenerek tahmin işlemi gerçekleştirilmiştir. Modeller belirlendikten sonra performanslarını değerlendirmek amacıyla MAE, MAPE, MSE ve RMSE ölçümleri kullanılmıştır. Elde edilen sonuçlar doğrultusunda kullanılan modellerin performansları değerlendirilerek en başarılı yöntemin tespit edilmesi amaçlanmıştır.

Çalışmanın ikinci bölümünde zaman serisi analizi, üçüncü bölümünde derin öğrenme kavramları ve mimarileri, dördüncü bölümde ise performans değerlendirme ölçümleri üzerinde durulacaktır. Beşinci bölümde, çeşitli döviz kuru serilerinin yapısı incelenerek modellenecek ve tahmin işlemi yapılacaktır. Altıncı bölümde ise yapılan analizlerden elde edilen sonuçlar özetlenerek genel bir değerlendirme yapılmıştır.

2. ZAMAN SERİLERİ ANALİZİ

Zaman serisi analizi, zaman içinde toplanan verileri analiz etmek için kullanılan istatistiksel bir tekniktir. Zaman serisi analizinde, düzenli aralıklarla bir dizi veri noktası toplanır ve ardından verilerdeki kalıpları, eğilimleri ve ilişkileri belirlemek için analiz edilir.

Finans, ekonomi, mühendislik, sosyal bilimler, doğa bilimleri gibi çeşitli alanlarda yaygın olarak kullanılmaktadır. Geçmiş kalıplara dayalı olarak gelecekteki eğilimler hakkında tahminlerde bulunmaya ve verilerin zaman içinde nasıl değiştiğine dair analizler yapmak için kullanılabilir (Box vd. 2016: 1-2).

2.1. Zaman Serileri Analizinde Kullanılan Temel Kavramlar

2.1.1. Zaman serisi bileşenleri

Zaman serisi verileri, zaman içinde düzenli aralıklarla kaydedilen bir dizi gözlemlerdir (Box vd., 2016: 1). Bu gözlemler, her biri verilerin farklı bir modelini veya özelliğini temsil eden çeşitli bileşenlere ayrıştırılabilir. Bir zaman serisinin dört ana bileşeni trend, mevsimsel bileşen, döngüsel bileşen, rasgele bileşendir.

Trend

Verilerin zaman içindeki uzun vadeli davranışını ve verilerin hareket ettiği genel eğilimi temsil eder. Serinin zaman içinde arttığını, azaldığını veya durağan olduğunu gösterir. Eğilimler doğrusal, doğrusal olmayan veya döngüsel olabilir.

Mevsimsel bileşen

Verilerde belirli bir zaman diliminde meydana gelen periyodik dalgalanmaları ifade eder. Örneğin kış aylarında insanların daha sık hastalanmaları sebebiyle soğuk algınlığı ilaçları satışlarının kışın artması yazın ise azalması mevsimsel dalgalanmalardır. Mevsimsel bileşen periyodik olup aylık, üç aylık veya yıllık olabilir.

Döngüsel bileşen

Döngüsel bileşen, mevsimsel bileşenden daha uzun bir zaman diliminde meydana gelen zaman serilerindeki dalgalanmaları temsil eder. Mevsimsel bileşenden farklıdır çünkü sabit zaman aralıklarında gerçekleşmez. Döngüsel modellere tipik olarak durgunluklar ve patlamalar gibi ekonomik faktörler neden olur.

Rastgele bileşen

Zaman serilerinin diğer bileşenleri olan trend, mevsimsellik veya döngüsel bileşenlerle açıklanamayan değişimleri ifade eder. Verilerde meydana gelen ölçüm hatası veya ölçülmemiş değişkenler gibi faktörlerin neden olabileceği öngörülemeyen dalgalanmaları temsil eder.

Bir zaman serisinin farklı bileşenlerini anlamak verilerin altında yatan yapıyı anlamamıza ve geleceğe yönelik daha iyi tahminler yapılmasına yardımcı olur (Baş, 2021: 5; Brockwell ve Davis, 2016: 12-20).

2.1.2. Durağanlık

Zaman serisi modellerinin geliştirilmesindeki önemli bir özellik, bir tür istatistiksel denge varsayımı olan durağanlık varsayımıdır. Durağan modellerin spesifik bir istatistiksel denge formunda oldukları varsayılır. Serinin ortalamasının, varyansının ve otokorelasyonunun zaman içinde değişmediği anlamına gelir. Birçok istatistiksel model ve teknik durağanlık varsayımına dayandığından zaman serisi analizinde oldukça önemli bir kavramdır (Box vd., 2016: 7).

Geleneksel bir zaman serisi modelinin durağan olmaması durumunda, En Küçük Kareler (EKK) tahmin edicisi yansızlık ve tutarlılık özelliklerini korumaktadır ancak tahminler etkin değildir. Bunun sonucu olarak da parametre tahminleri istatistiki olarak anlamlı olmayabilir (Engle, 2001).

Durağanlık iki şekilde tanımlanabilir. Bir serinin tamamen durağan olması için olasılık yapısının zaman içinde değişmemesi gerekir. Gerçek dünya problemlerinde güçlü

durağanlığın sağlanabilmesi oldukça güç olduğundan, durağanlık kavramı genellikle zayıf durağanlık biçiminde ele alınmaktadır (Cryer ve Chan, 2008: 16-17).

Eğer bir seri;

i. Ortalaması zamana göre sabittir:

$$E(Y_t) = \mu \tag{2.1}$$

ii. Varyansı zamana göre sabit ve sonludur:

$$Var(Y_t) = E(Y_t - \mu)^2 = \sigma^2 < \infty \tag{2.2}$$

iii. Otokovaryansı gecikme mesafesine bağımlıdır:

$$\gamma_k = Kov(Y_t, Y_{t+k}) = E[(Y_t - \mu)(Y_{t+k} - \mu)] \tag{2.3}$$

yukarıda tanımlanan özellikleri taşıyorsa bu seriye durağan seri denilebilir (Box vd., 2016: 24-25).

Bununla birlikte, endüstride veya ekonomide karşılaşılan pek çok seri (örneğin, hisse senedi fiyatları ve döviz kurları) durağan olmayan davranış sergiler ve özellikle sabit bir ortalamaya göre değişmez. Durağan olmayan zaman serileri, zamanla değişen istatistiksel özelliklere sahiptir. Bu, serinin ortalama, varyans veya diğer özelliklerindeki değişiklikleri içerebilir. Birçok istatistiksel yöntemin varsayımları geçerli olmayabileceğinden, durağan olmayan zaman serilerinin analizi ve modellenmesi daha zor olabilir. Bu nedenle, istatistiksel modeller veya yöntemler uygulanmadan önce genellikle durağan olmayan bir zaman serisinin durağan hale dönüştürülmesi gerekir (Box vd., 2016: 10).

Yaygın dönüşümler arasında fark alma, mevsimsel fark alma, logaritmik dönüşümler yer alır. Bir zaman serisi durağan hale getirildikten sonra, çeşitli istatistiksel modeller ve teknikler kullanılarak analiz edilebilir. Bir serinin durağanlık yapısını ortaya koymak için zaman serisi grafiği çizmek, ortalama ve varyans gibi özet istatistikleri hesaplamak, otokorelasyon fonksiyonu oluşturmak ve birim kök testleri kullanmak gibi durağanlığı test etmenin birkaç yolu vardır (Hyndman ve Athanasopoulos, 2018: 221).

2.1.3. Otokorelasyon fonksiyonu (Autocorrelation function- ACF)

Otokorelasyon, bir zaman serisi ile gecikmeli değerleri arasındaki benzerlik derecesini ölçen istatistiksel bir araçtır. Başka bir deyişle, bir zaman serisinin zaman içinde kendisiyle ne kadar ilişkili olduğunu ölçer (Box vd., 2016: 29-30).

Otokorelasyon, zaman serisi analizinde önemli bir kavramdır, çünkü zaman serisinin altında yatan yapıya ilişkin bilgi sağlayabilir. Bu sayede model oluşturma ve tahmin amacıyla kullanılacak uygun zaman serisi modelini belirlemeye yardımcı olabilir (Kaya, 2019). Ayrıca serinin durağan olup olmadığı konusunda da bilgi verebilmektedir. Yüksek otokorelasyona sahip bir zaman serisi, serinin durağan olmadığını gösterebilir ve seriyi durağan hale getirmek için fark alınması gerekebilir. Öte yandan, düşük otokorelasyonlu bir zaman serisi, serinin beyaz gürültü olduğunu gösterebilir ve daha fazla modelleme gerektirmeyebilir (Box vd. 2016: 28-29).

Otokorelasyon grafikleri incelenerek durağanlık hakkında bilgi sahibi olunabilir. Gecikme sayılarına (k) karşılık otokorelasyon değerlerinin yer aldığı grafiğe otokorelasyon fonksiyonu grafiği ya da korelogram adı verilir (Kadılar ve Öncel Çekim, 2020: 26). Otokorelasyon değerlerinin hızlı bir şekilde sönmesi, zaman serisinin durağan olduğu anlamına gelirken yavaş bir şekilde azalması serinin durağan olmadığı anlamına gelir (Box vd., 2016: 353).

Otokorelasyon fonksiyonu:

$$\rho_k = \frac{Kov(Y_t, Y_{t+k})}{\sqrt{Var(Y_t)Var(Y_{t+k})}} \tag{2.4}$$

şeklinde ifade edilir (Tsay, 2005: 25-26).

Otokorelasyon katsayıları -1 ile +1 arasında değer almaktadır ve serinin geçmiş dönem değerleri ile arasında bağımlılığın derecesini göstermektedir. Otokorelasyon katsayısının yüksek olması değişkenin geçmiş dönem değerlerine bağımlı olduğunu göstermektedir (Kaya, 2019).

2.1.4. Kısmi otokorelasyon fonksiyonu (Partial autocorrelation function- PACF)

Kısmi otokorelasyon fonksiyonu ara gecikmelerin etkisini ortadan kaldırdıktan sonra bir zaman serisi ile gecikmeli değerleri arasındaki ilişkiyi belirlemek için kullanılan istatistiksel bir araçtır.

Otokorelasyon fonksiyonu, bir zaman serisi ile tüm gecikmeli değerleri arasındaki korelasyonu gösterirken, kısmi otokorelasyon fonksiyonu, gecikmeli değerlerin etkisi ihmal edildikten sonra bir zaman serisi ile gecikmeli değeri arasındaki korelasyonu gösterir (Kaya, 2019).

Genel olarak ACF grafiği, bir zaman serisinin genel korelasyon yapısını gösterirken, PACF grafiği, ara gecikmelerin etkisini ortadan kaldırdıktan sonra, zaman serisi ile gecikmeli değerleri arasındaki doğrudan ilişkiyi gösterir. Birlikte, tahmin ve modelleme amaçları için kullanılacak uygun zaman serisi modelini belirlemeye yardımcı olabilirler.

2.1.5. Birim kök testleri

Durağanlığı test etmede en geçerli yöntemlerden biri olan birim kök testleri bir serinin durağan olup olmadığını ya da durağanlık derecesini belirlemek amacıyla kullanılır (Box vd., 2016: 352).

Birim kök, bir zaman serisinin durağan olmadığını gösteren özelliğidir. Zaman serisi en birim köke sahipse seri durağan değildir. Bütünleşme sırası ise seri durağan bir seri olana dek alınan fark sayısına eşittir (Cryer ve Chan, 2008: 128-129).

Uygulamalarda Dickey-Fuller, Genişletilmiş Dickey-Fuller (Augmented Dickey-Fuller-ADF), Phillips-Perron (PP), Kwiatkowski, Phillips, Schmidt ve Shin (KPSS) gibi birim kök testleri kullanılmaktadır ancak en çok kullanılan birim kök testi ADF testidir. Bu testlerin her birinin farklı varsayımları vardır ve belirli zaman serisi verileri türleri için daha uygun olabilir (Box vd., 2016: 353-359; Karadağ, 2022).

ADF testi bir zaman serisinin birim köke sahip olup olmadığını test eden ve yaygın olarak kullanılan bir birim kök testidir. Serinin durağan olmadığı yani birim kök içerdiği hipotezine

karşılık durağan olduğu yani birim kök içermediği alternatif hipotezinin sınanması ile ilgilidir ve serilerin durağan olup olmadığı test edilebilmektedir (Cryer ve Chan, 2008: 128-129).

Zaman serileri analizinde yanlış sonuca varmamak için birim kökleri test etmek önemlidir. Durağan olmayan zaman serileri, iki değişken arasındaki ilişkinin önemli gibi göründüğü ancak aslında şansa bağlı olduğu sahte sonuçlarına yol açabilir. Birim kökleri test ederek, zaman serisi analizinin geçerli ve güvenilir olduğundan emin olunabilir.

2.2. Box-Jenkins Yöntemi (ARIMA Modelleri)

Zaman serileri analizinde kullanılan birçok yöntem vardır. Literatürde en bilinen ve en çok kullanılan yöntemlerden biri Box-Jenkins yöntemi zaman serilerinin modellenmesinde ve geleceğe yönelik yapılan kestirimlerde kullanılan istatistiksel yöntemlerinden biridir. Box ve Jenkins tarafından 1970'li yıllarda geliştirilen bu yöntem otoregresif entegre hareketli ortalama yöntemi olarak da literatürde geçmektedir. Durağan süreçlerde ya da durağan olmayan fakat çeşitli istatistiksel yöntemlerle durağanlaştırılabilen serilerde bu yöntemi uygulanabilir. Box-Jenkins yöntemlerinde temel esas, zaman serisine en uygun ve en az parametre içeren doğrusal modeli bulmaktır (Box vd., 2016; Kaya, 2019; Kaynar ve Taştan, 2009).

2.2.1. Otoregresif süreç (Autoregressive- AR)

Yule (1921) tarafından tanıtılan otoregresif süreçler bir zaman serisinin kendi gecikmeli değerlerinin bir fonksiyonu şeklinde ifade edilebilir.

Genel AR(p) süreci:

$$\varepsilon_t \sim WN(0, \sigma_\varepsilon^2) \tag{2.5}$$

$$Y_t = \delta + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t \tag{2.6}$$

şeklinde tanımlanmaktadır. Eş. 2.6'da tanımlanan Y_t serisi kendi geçmiş değerlerinin ve inovasyon, şok veya hata olarak adlandırılan ε_t'nin doğrusal bir kombinasyonudur. Burada

δ sabit terimi, ϕ ise (-1, 1) arasında değer alan bilinmeyen parametreleri temsil etmektedir. $\varepsilon_t, Y_{t-1}, Y_{t-2}, \ldots, Y_{t-p}$'den bağımsız, sıfır ortalamalı ve σ_ε^2 varyansa sahip bir beyaz gürültü (white noise) sürecidir ve Eş. 2.5'te tanımlanmıştır (Box vd., 2016: 52-55; Cryer ve Chan: 2008, 66-77; Karadağ, 2022; Kaya, 2019).

2.2.2. Hareketli ortalama süreci (Moving average- MA)

Hareketli ortalama süreç, zaman serisi herhangi bir dönemdeki hata terimi ile gecikmeli hata terimlerinin doğrusal bir bileşimi olarak ifade edilebilir. Hareketli ortalama süreci geçmiş dönem hatalar ile açıklanmaktadır.

Genel MA(q) süreci:

$$Y_t = \mu + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q} \tag{2.7}$$

şeklinde tanımlanmaktadır. Eş. 2.7'de yer alan denklemde μ sabit terimi, θ bilinmeyen parametreleri temsil etmektedir. ε_t, sıfır ortalamaya ve sabit bir varyansa sahiptir (Box vd., 2016: 68-72; Brockwell ve Davis, 2016: 43-44; Cryer ve Chan, 2008: 57-65; Karadağ, 2022; Kaya, 2019).

2.2.3. Otoregresif hareketli ortalama süreci (Autoregressive moving average- ARMA)

Bazı uygulamalarda, verilerin dinamik yapısını yeterince iyi tanımlamak için birçok parametreye sahip yüksek dereceli bir modele ihtiyaç duyulabilir. Bu durumda AR veya MA modelleri kullanışsız hale gelir. Bu zorluğun üstesinden gelmek için ARMA modelleri kullanılır. ARMA(p, q) modeli AR(p) ve MA(q) modellerinin birleşimidir.

Genel ARMA(p, q) süreci:

$$Y_t = \delta + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_p\varepsilon_{t-p} \tag{2.8}$$

şeklinde tanımlanmaktadır.

ARMA sürecinde, bir zaman serisinin herhangi bir gözlem değeri, gecikmeli gözlem değerinin ve hata teriminin doğrusal bileşimi olarak ifade edilebilir (Box vd., 2016: 75-80; Brockwell ve Davis, 2016: 47-49; Cryer ve Chan, 2008: 77-80; Karadağ, 2022; Kaya, 2019).

2.2.4. Otoregresif bütünleşik hareketli ortalama süreci (Autoregressive integrated moving average- ARIMA)

Çoğu zaman uygulamalarda karşılaşılan seriler, özellikle ekonometrik veriler durağan olmayan serilerdir. Durağan zaman serilerinde AR, MA ve ARMA modelleri kullanılırken durağan olmayan zaman seriler için ARIMA modeli geliştirilmiştir. Durağan olmayan serilerde fark alma işlemi uygulanarak serilerin durağanlığı sağlanır ve ARIMA modelleri kullanılır.

Bir zaman serisinin gözlem değerleri serinin ortalama değeri etrafında durağan değilse, serinin farkları alınarak durağanlık sağlanır. Uygulamalarda seri durağan hale gelene kadar fark alma işlemi yapılmaktadır. Farkı almış ve durağan hale getirilmiş seri üzerinde model kurulmaktadır.

Genel ARIMA(p, d, q) süreci:

$$(1-B)^d(1-\phi_1 B-\phi_2 B^2-\cdots-\phi_p B^p)Y_t=(1-\theta_1 B-\theta_2 B^2-\cdots-\theta_q B^q)\varepsilon_t \qquad (2.9)$$

AR ve MA süreçlerinin birer kombinasyonu şeklinde ifade edilebilen bu yöntem genel olarak ARIMA(p, d, q) şeklinde gösterilmektedir. Burada, p otoregresyon parametresinin derecesini, d durağanlığı elde etmek için gerekli fark alma derecesini ve q hareketli ortalama parametresinin derecesini göstermektedir (Box vd., 2016: 88-98; Brockwell ve Davis, 2016: 158-160; Cryer ve Chan, 2008: 92-98; Karadağ, 2022; Kaya, 2019).

2.3. Finansal Zaman Serileri Analizi

Finansal zaman serisi analizi, tipik olarak, farklı finansal değişkenler arasındaki eğilimleri, kalıpları ve ilişkileri belirlemek için tarihsel finansal verilerin incelemesidir. Bu, hisse senedi

fiyatları, faiz oranları veya döviz kurları gibi zaman sırasına göre endekslenmiş bir veri noktaları dizisi olan zaman serisi verilerinin analiz edilmesini içerir (Kaya, 2019).

Finansal zaman serisi analizinde kullanılan yaygın teknikler arasında regresyon analizi, zaman serisi yöntemleri ve makine öğrenimi algoritmaları bulunur. Finansal zaman serisi analizinin bazı yaygın uygulamaları, gelecekteki piyasa trendlerini ve varlık fiyatlarını tahmin etmeyi, risk ve oynaklığı modellemeyi, piyasa eğilimlerini ve döngülerini belirlemeyi ve geçmiş piyasa verilerine dayalı yatırım stratejileri geliştirmeyi içerir (Engle, 2001).

Finansal zaman serisi analizindeki temel zorluklardan bazıları, tahminlerin doğruluğunu etkileyebilecek ve ele alınması için özel teknikler gerektirebilecek, eksik veriler veya durağan olmama gibi veri düzensizlikleriyle uğraşmayı içerir. Ek olarak, finansal piyasalar karmaşık olabilir ve ani değişikliklere tabi olabilir, bu da gelecekteki sonuçları doğru bir şekilde tahmin etmeyi zorlaştırır (Cao vd., 2019).

2.3.1. Oynaklık

Oynaklık, bir finansal aracın veya varlığın fiyatının zaman içindeki değişimin veya dalgalanmanın derecesini ifade eder. Bir zaman serisinde oynaklık kavramı pozitif ya da negatif yönlü olarak ortalama değerden sapma genişliğinin ölçüsü olarak ifade edilebilir. Oynaklık, yatırımların risk ve getiri profilini etkilemesi nedeniyle finansal piyasalarda önemli bir kavramdır. Yüksek oynaklık, daha büyük potansiyel kazançlara veya kayıplara yol açabilirken, düşük oynaklık daha fazla istikrar ancak daha düşük getiri sunabilir.

Finansal zaman serileri olan döviz piyasalarında meydana gelen şoklar ve beklenmedik durumlar oynaklık veya volatilite olarak adlandırılan ani değişimlere yol açabilmektedir. Yüksek oynaklık, bir varlığın fiyatının kısa bir süre içinde hızla ve öngörülemez bir şekilde değişebileceği anlamına gelirken, düşük oynaklık, varlığın fiyatının nispeten istikrarlı olduğu anlamına gelir. Bu açıdan yatırımcılar için risk unsuru olan fiyat serilerinde oynaklık, yatırım kararlarını önemli ölçüde etkilediği için ölçülmesi gereklidir (Topaloğlu, 2019).

Finansal zaman serisi analizini diğer zaman serisi analizlerinden ayıran bir belirsizlik unsuru içerir. Günümüzde finansal serilerin değişkenlik seviyelerinin artması sonucunda zaman

serisi analizinde kullanılan yöntemler serilerdeki oynaklığı, değişen varyans sorunundan dolayı tam olarak açıklayamamaktadır. Finansal serilerde sıklıkla karşılaşılan bu sorunun üstesinden gelmek için geliştirilmiş bazı modellere bulunmaktadır (Tsay, 2005: 1).

2.4. Koşullu Değişen Varyans Modelleri (Conditional Heteroscedastic Models)

Geleneksel zaman serilerinde oynaklığın bir ölçüsü olan varyansın, zamana bağlı olarak değişmediği varsayılmaktadır. Ancak finansal zaman serilerinin varyansları genellikle zamana bağlı bir şekilde değişkenlik gösterdiğinden dolayı geleneksel modeller finansal zaman serilerini tam olarak açıklayamadıklarından bazı modeller geliştirilmiştir (Box vd., 2016: 361-362).

Engle (1982) finansal varlıkların özelliklerinin daha iyi anlaşılması ve zaman içinde değişen varyansın tahmin edilebilmesi için ARCH modelini tanıtmıştır (Engle, 1982). Daha sonra Bollerslev (1986), ARCH modelinin otoregresif hareketli ortalama modeline dönüştürülmüş hali olan ve geçmiş dönem hata karelerinin ağırlıklandırılmasına dayanan GARCH modelini geliştirmiştir (Bollerslev, 1986; Brockwell ve Davis, 2016: 196; Engle, 1982).

2.4.1. ARCH/GARCH etkisinin test edilmesi

ARCH modeli kullanılmadan önce seride ARCH etkisinin olup olmadığının kontrol edilmesi gerekmektedir. Literatürde ARCH etkilerinin varlığını kontrol etmek üzere önerilen ve en çok kullanılan test Engle'ın ARCH LM testidir. Zaman serisi verilerinde belirli sayıda gecikmeye kadar ARCH etkilerine dair kanıt olup olmadığını inceleyen bir Lagrange çarpanı testidir (Box vd., 2016: 367-368).

$$H_0 = \alpha_1 = \alpha_2 = \cdots = \alpha_p = 0$$
$$H_1 = En\ az\ bir\ \alpha_j \neq 0 \qquad j = 1,2,\dots,p \tag{2.10}$$

$LM = T * R^2$ şeklinde hesaplanır ve test istatistiği, p serbestlik dereceli χ^2dağılımına sahiptir. Test istatistiğinin, belirli bir anlamlılık düzeyinde ki-kare dağılımındaki kritik değerden büyük olması, yokluk hipotezi olan H_0 reddedilerek ARCH etkileri olduğuna dair kanıt olduğunu gösterir (Engle, 1982: 999-1000).

2.4.2. Otoregresif koşullu değişen varyans modeli (Autoregresif conditional heteroskedasticity- ARCH)

Geleneksel zaman serileri yöntemleri sabit varyans varsayımı altında çalışmaktadır. Buna karşı, finansal zaman serilerinde varyansın zamana bağlı olarak değişkenlik göstermesi yani değişen varyans söz konusudur. Bu zaman serilerinde oynaklık olmasından dolayı bilinen zaman serisi yöntemleri modellemede ve tahminde başarılı olamamaktadır (Brockwell ve Davis, 2016: 196).

1980'lerin başında Robert Engle, bir zaman serilerinin hata terimlerinin varyansının geçmiş dönemlerdeki hata terimlerine bağlı olduğunu ileri sürmüş ve değişen varyansını modellemek için ARCH modelini önermiştir (Engle, 1982).

ARCH modelleri, hisse senedi fiyatlarının, döviz kurlarının ve diğer finansal zaman serilerinin oynaklığını modellemek için yaygın olarak kullanılmaktadır. Çeşitli veri türlerinin oynaklığını modellemek için ekonometri, sanayi ve meteoroloji gibi diğer alanlarda da uygulanmıştır (Box vd., 2016: 362-363).

ARCH modelinde koşullu varyans, hata teriminin varyansının önceki dönem hata terimlerinin karelerinin bir fonksiyonu olarak karakterize edilir. Varyanstaki değişimleri dikkate alarak geçmiş dönemlerdeki bilgilere dayanarak tahmin yapabilir bu da ARCH modelinin basit bir doğrusal regresyon modeli tarafından yakalanmayan serilerin volatilitesindeki kalıpları yakalayabileceği anlamına gelir. Dolayısıyla durağan olmayan zaman serilerine daha uygun bir modelleme ve tahmin yapılabilir (Songül, 2010).

ARCH modeli;

$$y_t|\Psi_{t-i} \sim N(0, h_t) \tag{2.11}$$

$$h_t = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \alpha_2 \varepsilon_{t-2}^2 + \cdots + \alpha_p \varepsilon_{t-p}^2 \tag{2.12}$$

şeklinde tanımlanabilir. Eş. 2.11'de tanımlanan y_t , Ψ_{t-i} bilgi kümesine bağlı olarak sıfır ortalama ve h_t koşullu varyans ile normal dağılıma sahiptir.

Genel olarak bir ARCH (p) modeli;

$$h_t = \alpha_0 + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 \tag{2.13}$$

şeklinde ifade edilmektedir. Eş. 2.13'de verilen modelde h_t koşullu varyans, p ARCH sürecinin derecesini ve α bilinmeyen parametrelerin vektörünü gösterir.

ARCH modelinde;

$$\alpha_0 > 0 \;\; ve \;\; \alpha_i \geq 0, \qquad i = 1, \ldots, p \tag{2.14}$$

$$\sum_{i=1}^{p} \alpha_i < 1 \tag{2.15}$$

olmak üzere bazı kısıtlar mevuttur. Eş. 2.14'de verilen kısıtlar, koşullu varyansın pozitif olması için uygulanan parametre kısıtlamalarıdır. ARCH modelinde, ε_t'nin koşullu varyansı ε_{t-i}^2 lerin gerçekleşen değerlerine bağlıdır. Eş. 2.12, ARCH modelinde $\varepsilon_{t-1}^2, \varepsilon_{t-2}^2, \ldots, \varepsilon_{t-p}^2$ değerleri negatif olamayacağından bütün ε_t değerleri için koşullu varyans denklemi de negatif değer alamayacağı gösterir. Ek olarak Eş. 2.15'te verilen kısıt, modelin kararlılığının sağlanabilmesi için α parametrelerinin sabit terim hariç her birinin veya toplamlarının birden küçük olması gerekliliğidir. Bu kısıt sağlanmadığı taktirde model sonsuz bir varyansa sahip olacaktır (Engle, 1982).

2.4.3. ARCH modelinin zayıflıkları

ARCH modelinin avantajlarının yanı sıra bazı zayıf yönleri de bulunmaktadır;

i. Model, geçmiş dönem şokların karesine bağlı olduğu için pozitif ve negatif şokların oynaklık üzerinde aynı etkiye sahip olduğunu varsayar. Ancak uygulamalarda, bir finansal varlığın fiyatının pozitif ve negatif şoklara farklı tepkiler verdiği bilinmektedir.

ii. Modeldeki parametreler katı kısıtlara tabidir.

iii. ARCH modeli, bir finansal zaman serisindeki değişimlerin meydana gelmesine neyin sebep olduğunu açıklamaz, varyasyonların kaynağını anlamak için herhangi bir yeni katkı sağlamaz. Yalnızca koşullu varyansın davranışı açıklamaya yardımcı olabilecek mekanik bir yol önerir.

iv. ARCH modelleri finansal serilerdeki büyük şoklara yavaş tepki verdiğinden muhtemelen oynaklığı olduğundan daha büyük öngörebilmektedir. (Tsay, 2005: 106)

Varyans dinamiklerinin daha karmaşık spesifikasyonlarına izin veren GARCH ve Üstel GARCH (EGARCH) dahil olmak üzere ARCH modellerinin çeşitli varyasyonları vardır (Brockwell ve Davis, 2016).

2.4.4. Genelleştirilmiş otoregresif koşullu değişen varyans modeli (Generalized autoregressive conditional heteroskedasticity- GARCH)

Tim Bollerslev (1986), Engle tarafından önerilen ARCH sürecini genişleterek GARCH modelini geliştirmiştir. ARCH modeline koşullu varyansların geçmiş değerlerinin eklenmesiyle genelleştirilmiştir. GARCH model sınıfı, finans alanında kullanılan son derece popüler bir modeldir (Bollerslev, 1986).

ARCH modeli basit olmasına rağmen, genellikle bir finansal serilerde volatiliteyi yeterince açıklamak için birçok parametreye ihtiyaç duyar (Tsay, 2005: 113). ARCH modelinin ampirik uygulamalarında, koşullu varyans denkleminde genellikle modeli verilere uydururken, nispeten uzun bir gecikme tercih edilmesinden dolayı negatif varyanslı parametre tahminleriyle ilgili sorunlardan kaçınmak amacıyla tipik olarak sabit bir gecikme yapısı empoze edilir.

ARCH model sınıfını hem daha uzun bir belleğe hem de daha esnek bir gecikme yapısına izin verecek şekilde genişletmek amacıyla GARCH modeli önerilmiştir. ARCH sürecinin GARCH sürecine genişletilmesi, standart zaman serisi AR sürecinin genel ARMA sürecine genişletilmesiyle büyük benzerlikler taşır (Bollerslev, 1986).

GARCH modeli şu şekilde tanımlanır;

$$y_t|\Psi_{t-i} \sim N(0, h_t) \tag{2.16}$$

$$h_t = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \alpha_2 \varepsilon_{t-2}^2 + \cdots + \alpha_p \varepsilon_{t-p}^2 + \beta_1 h_{t-1} + \beta_2 h_{t-2} + \cdots + \beta_q h_{t-q} \tag{2.17}$$

$$h_t = \alpha_0 + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j h_{t-j} \tag{2.18}$$

Eş. 2.16'da tanımlanan y_t serisi Ψ_{t-i} bilgi kümesine bağlı olarak sıfır koşullu ortalama ve h_t koşullu varyans ile normal dağılıma sahiptir. ARCH modeli, bir zaman serisinin koşullu varyansın hata terimlerinin geçmiş değerlerine bağlı olduğunu varsayarken, GARCH modellerinde koşullu varyans hem geçmiş değerlere hem de geçmiş koşullu varyanslara bağlıdır. Bu, bir tür uyarlanabilir öğrenme mekanizmasına karşılık gelir (Bollerslev, 1986). Bu model aynı zamanda geçmiş hata karelerinin ağırlıklı ortalamasıdır, ağırlıklar gecikmeyle birlikte azalır, ancak hiçbir zaman tamamen sıfıra gitmez (Engle, 1982).

GARCH(p, q) modelindeki koşullu varyansın tanımlı olabilmesi için;

$$\alpha_0 > 0, \ \alpha_i \geq 0, \ i = 1, \dots, p,$$

$$\beta_j \geq 0, \qquad j = 0,1, \dots, q \tag{2.19}$$

$$p > 0, \ q \geq 0,$$

$$\sum_{i=1}^{\max(p,q)} (\alpha_i + \beta_i) < 1 \tag{2.20}$$

Koşullu varyansların negatif olmaması gerektiğinden, bir GARCH modelindeki katsayılar genellikle negatif olmayacak şekilde kısıtlanır. Dolayısıyla Eş. 2.19'da verilen kısıtların sağlanması gerekmektedir. Eş. 2.2'de verilen kısıt, GARCH(p, q) modelinin zayıf durağanlığı için gerekli ve yeterli şartıdır. Ayrıca ε_t'nin koşullu varyansının zaman içinde değiştiğini ve koşulsuz varyansının sonlu olduğunu ifade eder (Brockwell ve Davis, 2016: 366; Cryer ve Chan, 2008: 296). Ayrıca $q = 0$ için süreç ARCH(p) sürecine indirgenir. $p = q = 0$ olması durumunda ε_t, beyaz gürültü süreci olur (Bollerslev, 1986).

2.4.5. GARCH modeli uzantıları

ARCH ve GARCH modelleri volatilitenin davranışını yakalarken, finansal verilerde yaygın olarak gözlemlenen diğer bazı özellikleri hesaba katmazlar. Finansal varlıkların volatilitesi pozitif ve negatif şoklara karşı asimetrik tepkiler verebilmektedir. Ancak ARCH ve GARCH modellerinde pozitif ve negatif şoklara karşı volatilitenin simetrik tepki verdiğini varsayılır. GARCH modellerinde volatilite, şokların işaretlerine değil büyüklüğüne bağlıdır, yani şokların volatilite üzerindeki etkisini ayrıştıramamaktadır. Dolayısıyla GARCH süreci varyans yapısındaki asimetriyi yakalamakta başarısız kalmaktadır (Box vd., 2016).

Temel ARCH ve GARCH modellerinin bir diğer sınırlaması ise, parametrelerin negatif olmamasını sağlamak için uygulanan negatif olmama kısıtlamalarından kaynaklanır. Ayrıca, bu kısıtlamaları, GARCH modellerini tahmin etmede zorluklar yaratır (Nelson, 1991).

Volatilite dinamiklerini modellemede daha fazla esnekliğe izin veren temel GARCH modelinin Üstel GARCH (Exponential Generalized Autoregressive Conditional Heteroskedastic, EGARCH), Eşik Değerli GARCH (Threshold Generalized Autoregressive Conditional Heteroskedastic, TGARCH) gibi birkaç uzantısı vardır. Bu uzantılar, finansal verilerin belirli özelliklerini yakalamada yararlı olabilir ve oynaklık tahminlerinin doğruluğunu artırabilir (Kızılsu vd., 2001; Tsay, 2005: 124-131).

Üstel genelleştirilmiş otoregresif koşullu değişen varyans modeli (Exponential generalized autoregressive conditional heteroskedastic- EGARCH)

Finansal zaman serilerinin analizinde GARCH modelinin bazı zayıflıklarının üstesinden gelmek için Nelson (1991) EGARCH modelini önermektedir (Tsay, 2005: 124).

GARCH modeli başarılı bir şekilde finansal varlıklarda meydana gelen volatilite kümelenmelerini başarılı bir şekilde yakalamaktadır. Bununla birlikte, GARCH modeli varyans yapısındaki asimetrik tepkileri yakalamakta yeterli değildir. Nelson (1991) volatilite yapısındaki asimetriyi hesaba katarak, koşullu varyansın, gecikmeli hata terimlerinin hem büyüklükleri hem de işaretleri dikkate alarak modellendiği Üstel GARCH (EGARCH) modelini geliştirmiştir (Nelson, 1991).

Genel EGARCH(p, q) modeli şu şekildedir;

$$z_t = \frac{\varepsilon_t}{\sqrt{h_t}} \tag{2.21}$$

$$g(z_t) = \theta z_t + \gamma[|z_t| - E|z_t|] \tag{2.22}$$

$$lnh_t = \alpha_0 + \sum_{i=1}^{p} \alpha_i g(z_{t-i}) + \sum_{j=1}^{q} \beta_j lnh_{t-j} \tag{2.23}$$

Eş. 2.23'te yer alan α_0, α_i, β_j model parametreleridir, Eş. 2.21'de tanımlanan z_t ise ortalaması sıfır, varyansı bir olan standartlaştırılmış bir değişkendir. EGARCH modeli, koşullu varyansının logaritması ile geçmiş bilgiler arasındaki ilişkiyi açıkladığından, model koşullu varyansın negatif olmamasını sağlamak için parametreler üzerinde herhangi bir kısıtlama gerektirmez (Nelson, 1991).

EGARCH modeli, bu veri kümelerindeki oynaklığın karmaşık dinamiklerini yakalayabildiğinden, finans ve ekonomide hisse senedi getirilerini, döviz kurlarını ve diğer finansal zaman serisi verilerini modellemek için sıklıkla kullanılır (Tsay, 2005: 126-127).

2.5. Model Seçim Kriterleri

Zaman serisi model seçiminde en iyi olan modeli belirlenmeye yardımcı olan kullanılan birkaç model seçim kriteri geliştirilmiştir. Akaike bilgi kriteri (Akaike Information Criterion- AIC), Bayes bilgi kriteri (Bayes Information Criterion- BIC) literatürde en sık kullanılan bilgi kriterleridir. Bu kriterler veriye tam olarak uyan en basit modelin seçilmesi gerektiğini belirten tutumluluk ilkesine dayanmaktadır. En düşük AIC veya BIC değerine sahip olan model seçilir (Box vd., 2016: 190-194).

Akaike Bilgi Kriteri (Akaike Information Criterion- AIC)

Akaike (1973) tarafından tasarlanan bilgi kriteri şu şekilde tanımlanır:

$$AIC = \frac{-2\ln(L) + 2r}{n} \approx \ln(\hat{\sigma}_a^2) + \frac{2r}{n} \tag{2.24}$$

Eş. 2.24'te verilen formülde $\hat{\sigma}_a^2$, σ_a^2'nin maksimum olabilirlik tahminidir. n gözlem sayısını r ise modelde tahmin edilen parametre sayısını göstermektedir. AIC, çok sayıda parametreye sahip modelleri cezalandırır ve bu nedenle verilere uyan daha basit modelleri tercih eder.

Bayes Bilgi Kriteri (Bayes Information Criterion- BIC*)*

Schwarz bilgi kriteri olarak da bilinen BIC, Gideon E. Schwarz (1978) tarafından geliştirilen bilgi kriteri şu şekilde tanımlanır:

$$BIC = \ln(\hat{\sigma}_a^2) + \frac{r}{n}\ln(n) \tag{2.25}$$

BIC model karmaşıklığına yönelik cezayı ayarlar. AIC'ye benzer, ancak birçok parametreye sahip modellere daha ağır bir ceza verir. BIC değerleri ne kadar düşükse, model o kadar iyidir (Box vd., 2016: 193-194).

3. DERİN ÖĞRENME VE TEMEL KAVRAMLAR

3.1. Yapay Zekâ

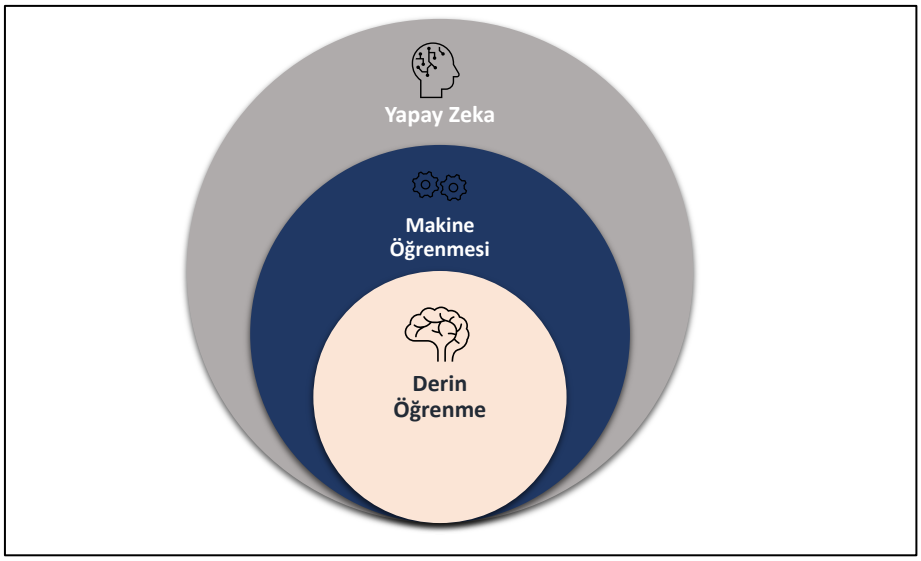
1950'li yıllarda yeni gelişmeye başlamış olan yapay zekâ veri bilimcilerin ve araştırmacıların bugün de cevabını aradığı "bilgisayarlar düşünebilir mi?" sorusuna cevap ararlarken ortaya çıkmıştır. Yapay zekâ, bir bilgisayarın genellikle insana özgü nitelikler olduğu varsayılan düşünsel faaliyetleri; akıl yürütme, anlam çıkartma, genelleme ve geçmiş deneyimlerden öğrenme gibi süreçlere ilişkin görevleri yerine getirme yeteneği olarak tanımlanmaktadır (Chollet, 2020: 4).

Yapay zekâ, insanlar gibi düşünebilen, akıl yürütebilen ve öğrenebilen makineler oluşturma fikrine dayanmaktadır. Genel olarak, görsel algı, konuşma, karar verme gibi insana ait bilişsel yetenekler gerektiren görevleri gerçekleştirmek üzere programlanmış makinelerde insan zekâsının simülasyonu olarak da ifade edilebilir.

Robotik, doğal dil işleme, uzman sistemler, makine öğrenmesi ve bilgisayar görüşü dahil olmak üzere çeşitli uygulamalarda kullanılan yapay zekânın amacı, insan müdahalesi veya denetimi olmadan karmaşık görevleri yerine getirebilen ve sorunları çözebilen makineler oluşturmaktır. Yapay zekânın gelişimi devam eden bir süreçtir ve hayatımızın birçok yönünü iyileştirme ve endüstrilerde devrim yaratma potansiyeline sahiptir (Nabiyev, 2021: 64-68).

Yapay zekâ bilgisayarların verilerden öğrenmesini ve zaman içinde performanslarını iyileştirmesini sağlayan makine öğrenmesi algoritmaları tarafından desteklenmektedir. Makine öğrenmesinin bir alt kümesi olan derin öğrenme, insan beyninin yapısını ve işlevini taklit etmek için sinir ağlarını kullanır ve bilgisayarların karmaşık verileri analiz etmesine ve daha doğru tahminler yapmasına olanak tanır (Chollet, 2020: 5-9).

Yapay zekâ, makine öğrenmesi ve derin öğrenme kavramlarının birbiri ile olan ilişkisinin şeması Şekil 3.1'de verilmiştir.



Şekil 3.1. Yapay zekâ, makine öğrenmesi ve derin öğrenme (Chollet, 2020: 4)

3.2. Makine Öğrenmesi

Makine öğrenmesi, bilgisayar sistemlerinin açıkça programlanmadan verilerden öğrenmesini ve verilere dayalı tahminler veya kararlar almasını sağlayan algoritmalar ve istatistiksel modeller geliştirmeye odaklanan yapay zekânın bir alt kümesidir. Kalıpları belirlemek ve yeni verilere dayalı olarak doğru tahminler veya kararlar verebilecek modeller geliştirmek için büyük miktarda veriyi analiz etmeyi içerir. Makine öğrenimi, doğal dil işleme, bilgisayar görüşü, öneri sistemleri ve otonom araçlar dahil olmak üzere çeşitli uygulama alanlarına sahiptir (Chollet, 2020: 5-8).

Makine öğrenmesi, karar verme süreçlerinde kullanmak üzere tasarlanmış "veriden öğrenme" sürecidir ve mevcut veriler üzerinden öğrenme süreci gerçekleşir. Genel olarak denetimli öğrenme, denetimsiz öğrenme ve pekiştirmeli öğrenme olmak üzere üç farklı makine öğrenmesi türünden söz edilebilir (Zaccone ve Karim, 2018: 2-4).

3.2.1. Denetimli öğrenme

Geçmişte edinilen bilgileri gelecekteki olayları tahmin etmek amacıyla denetimli makine öğrenmesi algoritmaları kullanılır. Denetimli öğrenme, algoritmanın bir veri seti üzerinde eğitildiği ve her veri noktasının belirli bir çıktı veya yanıtla ilişkilendirildiği bir makine öğrenmesi türüdür.

Denetimli öğrenmede eğitim verisi olarak adlandırılan bir veri seti kullanılır. Bu yaklaşımda veri seti hem girdi hem de çıktı verilerine sahiptir ve algoritma, girdi verilerini çıktı verilerine eşlemek için eğitilir. Algoritma, bir dizi girdi ve bunlara karşılık gelen doğru çıktılarla sağlanır ve amaç, yeni girdiler için çıktıyı doğru bir şekilde tahmin edebilmesi için girdi ve çıktı verileri arasındaki ilişkiyi öğrenmektir. Verilere dayalı bilinçli kararlar almak için kullanılabilecek tahmine dayalı modeller oluşturmada güçlü bir araçtır. Denetimli öğrenme sınıflandırma ve regresyon uygulamalarında kullanılabilir.

3.2.2. Denetimsiz öğrenme

Makine öğrenmesinin başka bir türü olan denetimsiz öğrenmede, denetimli öğrenmeden farklı olarak ele alınan veri kümesinde herhangi bir çıktı değişkeni bulunmamaktadır. Başka bir deyişle denetimsiz öğrenmenin algoritmanın ulaşması gereken belirli bir sonucu veya hedefi yoktur. Verilerdeki kalıpları, kümeleri ve ilişkileri bulmayı amaçlayan denetimsiz öğrenme daha fazla bilgi edinmek ve analiz yapabilmek için verilerin altında yatan yapıyı anlamaya çalışır.

Denetimsiz öğrenme tekniklerinin bazı örnekleri arasında kümeleme, birliktelik analizi ve boyut azaltma yer alır. Denetimsiz öğrenme veri madenciliği, örüntü tanıma ve büyük miktarda verinin analiz edilmesi ve düzenlenmesi gereken diğer uygulamalarda yaygın olarak kullanılır.

3.2.3. Pekiştirmeli öğrenme

Belirli bir soruna çözüm bulmak için deneme yanılma yöntemini kullanan pekiştirmeli öğrenme tekniği, amaca yönelik nasıl karar vereceğini öğrenen makine öğrenmesi türlerinden biridir.

Etkileşimli bir ortam oluşturmak için çeşitli eylemler üreterek hataların cezalandırılıp ve başarılı durumları ödüllendirilmesi esasına dayanan pekiştirmeli öğrenme yönteminde performansı en üst düzeye çıkarılması ve maksimum performansa götüren eylemi optimize etmek amaçlanmaktadır (Özkan, 2021: 67-72; Zaccone ve Karim, 2018: 3-8).

3.3. Derin Öğrenme

Son yıllarda veri hacminin artması ve veri setindeki değişkenlerin fazlalığı ve değişkenler arasındaki karmaşık ilişkilerin çözümlenmesine ilişkin yapılan çalışmalar yapay sinir ağları temelli derin öğrenme yaklaşımına olan ilgiyi arttırmıştır.

Derin öğrenme yirminci yüzyılın ortalarında ortaya çıkan bir yapay zekâ yaklaşımıdır. Derin öğrenmenin tarihi 1943 yılında Warren McCulloch ve Walter Pitts tarafından düşünce sürecini taklit etmek amacıyla matematiğe ve algoritmalara dayalı sinir ağları için bir işlem sistemi oluşturmalarına uzanmaktadır. Karmaşık sorunları modellemek ve çözmek için yapay sinir ağlarını kullanan makine öğrenmesinin bir alt alanı olan derin öğrenmenin gelişimi yapay zekâ ve sinir ağları çalışmalarına paralel olarak ilerlemiştir (Zhang, Lipton, Li ve Smola, 2021: 34-38).

Derin öğrenme, insan beyninin yapısından ve işlevinden ilham alır ve bilgileri işlemek için yapay sinir ağları adı verilen birbirine bağlı düğümlerden oluşan katmanlı ağlar kullanır. Bu ağlar kalıpları tanımayı verileri sınıflandırmayı ve tahminlerde bulunmayı öğrenebilir (Campesato, 2020: 19-21). Derin öğrenmenin arkasındaki temel fikir, ağa daha fazla katman ekleyerek verilerdeki karmaşık kalıpları ve özellikleri tanımada daha iyi hale gelmesi ve giderek daha karmaşık tahminler ve kararlar almayı öğrenebilmesidir.

Derin öğrenme; nesne tespiti, ses işleme-tanıma, görüntü işleme-sınıflandırma, yüz tanıma, doğal dil işleme, hava durumu tahmini, finansal veri tahmini gibi birçok alanda yaygın olarak kullanılmaktadır. Bilim ve teknolojinin birçok alanında önemli ilerlemelere yol açmıştır (Chollet, 2020: 9-12).

Derin öğrenme modelleri veri kaynağından alınan bilgileri insan müdahalesi olmaksızın analiz eder. Klasik makine öğrenmesi modellerinden farklı olarak derin öğrenmenin veriden öğrenebilmesi iki durumla özetlenebilir. İlk olarak, karmaşık problemleri çözmek için artan bir şekilde katmanlar oluşturulabilir. İkincisi ise tüm katmanların beraber öğrenebilmesi, bir katmanın kendinden önceki ve sonraki katmanların ihtiyaçlarına göre kendisini güncelleyebilmesidir. Bunların birleşimi derin öğrenme modellerini makine öğrenmesi modellerinden daha başarılı yapar (Chollet, 2020: 19-20).

3.3.1. Yapay sinir ağları (Artificial neural network- ANN)

Yapay sinir ağları beynin temel özelliği olan öğrenme kabiliyetinden ilham alarak bilgi edinme, yorumlama ve geliştirme özelliklerini taklit eden biyolojik sinir ağlarını matematiksel olarak modellenmesini sağlayan yapılardır (Zaccone ve Karim, 2018: 11-14).

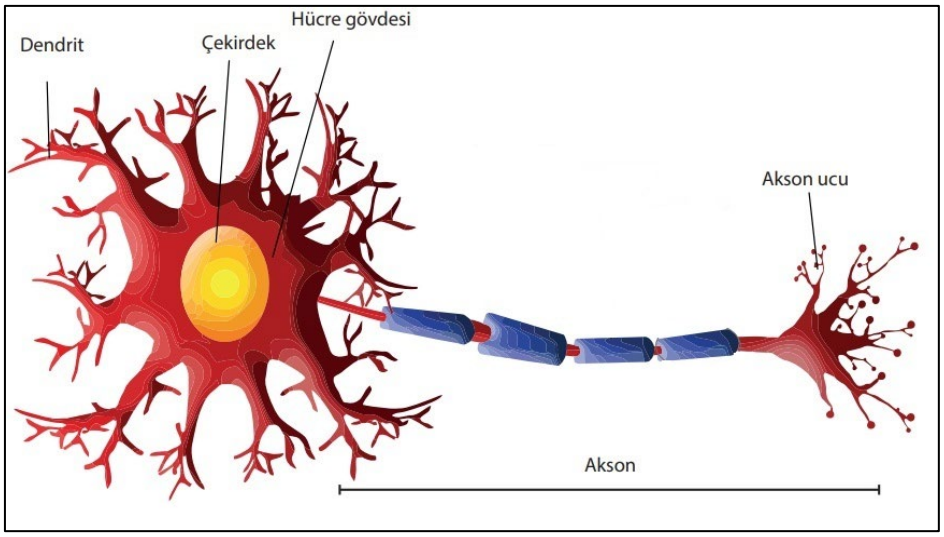
Veriden öğrenme ve genelleme yapabilme yeteneğiyle araştırmacılar tarafından farklı alanlarda kullanılan yapay sinir ağları, yapay zekânın geliştirilmesinde kilit bir teknolojidir. Büyük veri kümelerinden karmaşık kalıpları öğrenebilen derin öğrenme modellerinde de kullanılırlar.

Yapay sinir ağları, sinir sisteminin temel yapı taşları olan biyolojik sinir ağlarının yapısından ve işlevinden esinlenilen bir hesaplama modelidir. Bilgileri işleyen ve ağ üzerinden ileten birbirine bağlı nöron adı verilen yapılardan oluşur (Nabiyev, 2021: 577).

Biyolojik sinir hücresi

Sinir hücresi, nöron olarak da bilinen, sinir sisteminde elektriksel ve kimyasal sinyaller yoluyla bilgi ileten özel bir hücredir. Sinir hücreleri, geniş bir hücre gövdesi ve bu gövdeden çıkan dendrit denilen kısa uzantılar ve akson adı verilen uzun uzantılar ile sinir hücresinin sinir ucu arasında kalan ince uzantılar olan sinapslardan oluşur.

Hücre gövdesi; hücreyi denetler ve hücrenin yönetimini sağlar. Dendritler üzerinden gelen sinyalleri toplar. Dendrit; hücre gövdesinden çıkan ve geniş alana yayılan ağaç dalları biçiminde uzantılardır. Dendritler, sinyalleri iletim hatları olarak kullanılan aksonlar boyunca diğer nöronlardan alır ve hücre gövdesine taşır. Akson; kendisine gelen bir sonraki hücreye iletilmesini sağlar (Zaccone ve Karim, 2018: 11-12). Şekil 3.2'de biyolojik bir sinir hücresi gösterilmiştir.



Şekil 3.2. Biyolojik sinir hücresi (Yeni Biyoloji, 2022)

Akson sonları ile dendritler arasında, sinaps adı verilen küçük boşluklar bulunur. Her sinir hücresi, diğer nöronlarla sinaps yardımı ile iletişim kurar. Aksonla dendritlerin bağlantı noktası olan bu sinapslar, bilgilerin uzun süre saklandığı bilgi saklama yerleri olarak düşünülmektedir. Uzun süre sakladıklarından dolayı, uzun süreli hafıza olarak bilinirler. Dolayısıyla sinapslar, nöronun kendi sinyalini komşu nörona ilettiği bağlantı noktasıdır.

Nöronlar arasındaki bağlantı sayısının artmasıyla beynin büyümesi gerçekleşir. Nöronlar arasındaki bağlantılar arttıkça, beyin daha ayrıntılı işlemler yapabilir ve daha çok öğrenir. Beynin büyümesi ise yeni nöronların oluşumuyla değil, nöronlar arasındaki bağlantı sayısının artmasıyla gerçekleşir. Nöronlar arasındaki bağlantılar arttıkça, beyin daha ayrıntılı işlemler yapabilir ve daha çok öğrenir (Nabiyev, 2021: 578-579).

Yapay sinir hücresi

Biyolojik sinir hücresi modeline dayanan yapay sinir ağı modeli McCulloch ve Pits tarafından geliştirilmiştir. Oluşturulan modelde biyolojik sinir hücresi ile yapay sinir hücresi arasında bir ilişki kurulmuştur. Yapay sinir ağları ise yapay sinir hücrelerinin aralarında bağ kurmasıyla oluşur (Öztürk ve Şahin, 2018).

Nöron, bir sinir ağının temel yapı taşıdır. Diğer nöronlardan girdi alan, onu işleyen ve ağdaki diğer nöronlara bir sinyal gönderen matematiksel bir fonksiyondur. Girdiler ağırlıklandırılır ve birleştirilir. Nöronun çıktısını belirlemek için doğrusal olmayan bir aktivasyon

fonksiyonu uygulanır. Her bir nöronun ağırlıkları ve sapmaları, ağın performansını optimize etmek için eğitim sırasında ayarlanır (Nabiyev, 2021: 582-583).

Yapay sinir ağı yaklaşımının matematiksel gösterimi;

$$y = f(\sum_{i=1}^{N} w_i x_i + b) \tag{3.1}$$

şeklinde ifade edilir (Haykin, 2001: 158).

Burada x_i girdi değerleri, w_i ağırlıkları ve b yan değerini göstermektedir. Bir yapay sinir ağı modelinde girdi değerleri ile ağırlıklandırma katsayıları çarpılır ve yan değerleri eklenerek toplanır. Sonuç olarak çıktı değeri aktivasyon fonksiyonu ile işlenerek elde edilir.

Aktivasyon fonksiyonları, yapay sinir ağlarında çıktı değerini hesaplamak için kullanılır. Fonksiyon seçiminde genel bir kural bulunmayıp bu seçim verilere ve verilerin dönüşüm türlerine göre değişkenlik gösterir. Bununla birlikte aktivasyon fonksiyonlarının grafiğini incelemek karar vermede yardımcı olur (Nabiyev, 2021: 583-584).

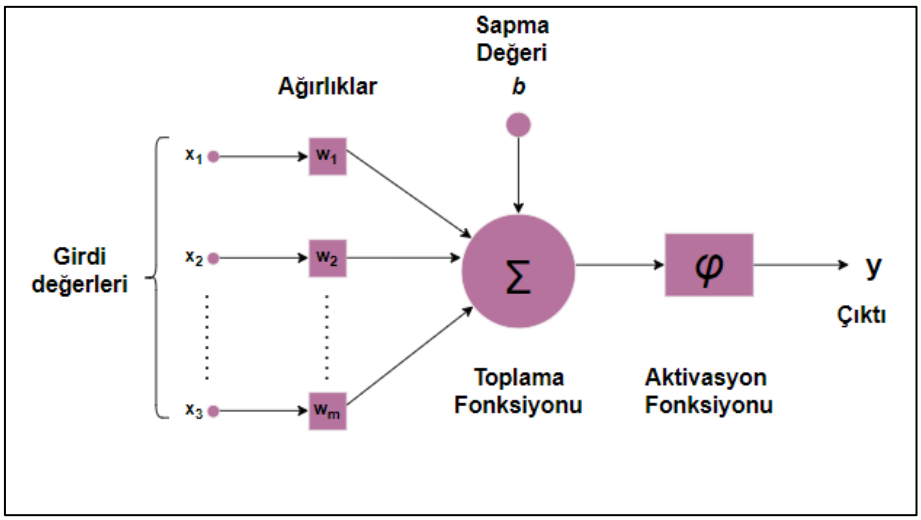
3.4. Aktivasyon Fonksiyonları

Aktivasyon işlemi, sinir ağının verilerdeki karmaşık kalıpları öğrenmesine yardımcı olan bir işlemdir. Yapay sinir ağı tasarımının önemli bir parçasıdır ve oluşturulan sinir ağının performansı üzerinde güçlü bir etkiye sahiptir. Aktivasyon fonksiyonu, bir nöronun aktive edilip edilmeyeceğine karar verir. Başka bir deyişle matematiksel işlemler kullanarak tahmin sürecinde nöronun ağa girişinin önemli olup olmadığına karar vereceği anlamına gelir (Baheti, 2021).

Genellikle yapay sinir ağı modellerinde kullanılan aktivasyon fonksiyonları doğrusal olmayan fonksiyonlardır. Yalnızca doğrusal aktivasyon fonksiyonlarından oluşan bir ağın eğitilmesi çok kolaydır, ancak karmaşık işlevleri öğrenemez. Karmaşık yapıdaki gerçek dünya problemlerini çözmek için doğrusal olmayan aktivasyon fonksiyonları kullanılır. Bu sayede oluşturulan yapay sinir ağı modeli veri kümeleri üzerindeki karmaşık yapıları daha iyi bir şekilde öğrenerek daha başarılı sonuçlar verir (Brownlee, 2019).

Doğrusal olmayan problemleri çözmek için aktivasyon fonksiyonu kullanılmayan bir sinir ağı öğrenme gücü açısından sınırlıdır. Her bir derin öğrenme katmanı için aktivasyon fonksiyonu seçimi dikkatli bir şekilde yapılmalıdır. Gizli katmanlarda kullanılacak aktivasyon fonksiyonu, ağın eğitim veri setini ne kadar iyi öğrendiğini kontrol eder. Çıktı katmanında kullanılacak aktivasyon fonksiyonu ise modelin yapabileceği tahminlerin türünü tanımlar (Brownlee, 2021).

Aktivasyon sürecinin matematiksel bir görselleştirilmesi Şekil 3.3'te verilmiştir.



Şekil 3.3. Yapay sinir ağında aktivasyon süreci (Nabiyev, 2021: 591)

Bir sinir ağında, aktivasyon fonksiyonu, düğümden gelen toplam ağırlıklı girdiyi, o girdi için düğümün veya çıktının aktivasyonuna dönüştürmekten sorumludur. Toplama fonksiyonundan elde edilen y değerinde o nöronun yapısına göre nöronun aktif olup olmayacağına karar verir.

En sık kullanılan aktivasyon fonksiyonları arasında sigmoid, tanh, ReLU, swish ve softmax bulunur. Bu fonksiyonlar verideki karmaşık kalıpları ve ilişkileri öğrenmesine izin veren ağa doğrusal olmama özelliğini getirme yeteneklerine göre seçilir. Aktivasyon fonksiyonu seçimi, sinir ağının performansı üzerinde önemli bir etkiye sahiptir ve belirli bir görev için uygun işlevi seçmek, derin öğrenme sürecinin önemli bir parçasıdır (Campesato, 2020: 81-82). Aktivasyon fonksiyonlarının grafikleri ve denklemleri Çizelge 3.1.'de verilmiştir.

3.4.1. Sigmoid fonksiyonu

Sigmoid aktivasyon fonksiyonu, sinir ağlarında kullanılan popüler bir aktivasyon fonksiyonudur. Herhangi bir giriş değerini 0 ile 1 arasında bir değere eşleyen doğrusal olmayan matematiksel bir işlevdir.

Girilen gerçek bir sayının çıktısını olasılığa dönüştürülmesi gereken problemler için kullanışlıdır. Sigmoid fonksiyonu türevlenebilir bir fonksiyondur ve düzgün bir gradyan sağlar bu sayede çıkış değerlerinde sıçramaları önler. İkili sınıflandırma problemlerinde tercih edilen bir aktivasyon fonksiyonudur (Buduma, 2015: 17-18).

3.4.2. Tanh (Hiperbolik tanjant) fonksiyonu

Yaygın olarak "tanh" olarak kısaltılan hiperbolik tanjant aktivasyon fonksiyonu, sinir ağlarında yaygın olarak kullanılan bir aktivasyon fonksiyonudur. Sigmoid fonksiyonuna benzer bir yapıya sahip olan tanh fonksiyonu girdi değerlerini (-1, 1) aralığında bir çıkış değerine eşleyen matematiksel bir işlevdir. Sigmoid fonksiyonuna kıyasla daha geniş bir çıktı değerleri aralığına sahiptir. Sigmoid fonksiyonuna göre avantajı türevinin daha dik olmasıdır yani daha çok değer alabilmesidir. Bu aktivasyon fonksiyonu genellikle gizli katmanlarda tercih edilir. Verilerin merkezlenmesine yardımcı olduğundan sonraki katman için öğrenme işlemi çok daha kolay gerçekleşir (Campesato, 2020: 86-87).

3.4.3. ReLU (Doğrultulmuş doğrusal birimler) fonksiyonu

Genel olarak, ReLU aktivasyon fonksiyonu, birçok farklı sinir ağı türünde yaygın olarak kullanılan popüler ve etkili bir aktivasyon işlevidir. ReLU fonksiyonunun en büyük avantajı sıfırdan büyük tüm girdiler için ağın daha hızlı eğitilmesini sağlar. Sigmoid ve tanh fonksiyonlarında neredeyse tüm nöronlar aktive olduğundan dolayı aktivasyon işlemi yoğundur ve işlem yükü fazladır. ReLU fonksiyonu ise tüm nöronların aynı anda aktive etmez, bu sayede işlem yükü diğer fonksiyonlara göre daha azdır (Campesato, 2020: 84-85).

ReLU popüler hale gelmeden önce yaygın olarak kullanılan sigmoid ve tanh gibi diğer aktivasyon fonksiyonlarına göre çeşitli avantajlara sahiptir. Ana avantajlarından biri, hızlı bir hesaplama süresine sahip basit bir matematiksel fonksiyon olduğu için hesaplama

açısından verimli olmasıdır. Ayrıca ReLU, gizli katmanlarda diğer aktivasyon fonksiyonları kullanılırken ortaya çıkabilen kaybolan gradyan sorununu çözmeye yardımcı olur.

Gradyanların kaybolması sorunu önceki katmanların önemli bilgileri öğrenmesini engeller. Bu problem, geri yayılım sırasında gradyanlar çok küçük hale geldiğinde öğrenme sürecinin yavaşlamasına ve hatta durmasına neden olabilir. ReLU işlevi, daha hızlı öğrenme ve daha iyi yakınsama ile sonuçlanabilecek işlevin doygunluğundan kaçınarak bu sorunu hafifletmeye yardımcı olabilir (Zhang vd., 2021: 133-134).

3.4.5. Swish (Self gated/Kendinden geçitli) fonksiyonu

2017 yılında Google araştırmacıları tarafından geliştirilen kendinden geçitli bir aktivasyon fonksiyonudur. Basitliği ve bazı derin öğrenme görevlerinde iyi performansı nedeniyle popülerlik kazanmıştır (El-Amir ve Hamdy, 2020: 304).

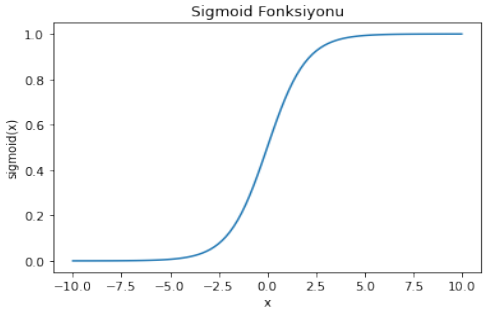
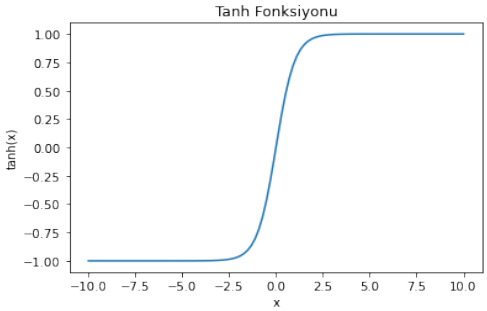
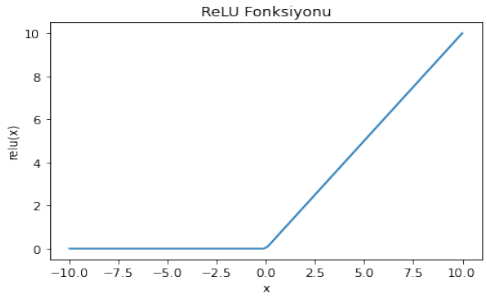
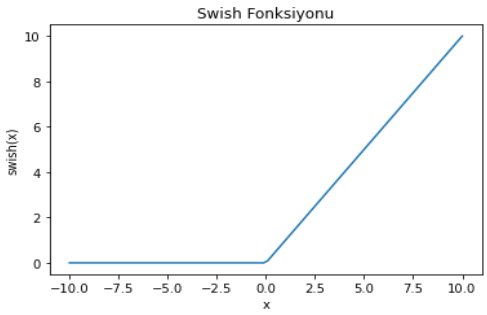
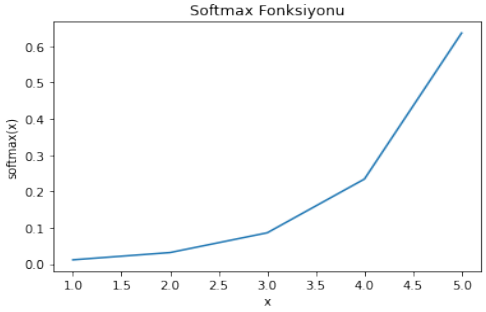
Swish aktivasyon fonksiyonu, görüntü sınıflandırması ve makine çevirisi gibi çeşitli zorlu alanlara uygulanan derin ağlarda sürekli olarak ReLU ile eşleşen veya ondan daha iyi performans gösteren fonksiyondur. Swish fonksiyonu yalnızsa sinir ağı 40 ve üstü katman olduğunda uygulanabilir. Çok derin ağlarda Swish aktivasyon fonksiyonu ReLU'dan çok daha yüksek bir performans gösterir. Bununla birlikte etkinliği, sinir ağında kullanılan belirli göreve, mimariye ve hiperparametrelere bağlı olabilir.

Ayrıca Swish aktivasyon fonksiyonu, dizi tahmini ve olasılık problemlerinde yaygın olarak kullanılan LSTM sinir ağlarında kullanılır. Swish, küçük gradyanların güncellenmesine izin verdiği için kaybolan gradyan problemini önleyebilir (Fatima, 2023).

3.4.6. Softmax fonksiyonu

Sigmoid fonksiyonuna benzer bir yapıya sahip olan softmax aktivasyon fonksiyonu özellikle derin öğrenme modellerinin çıkış katmanında tercih edilir. Girdi değerinin belirli sınıfa ait olma olasılığı (0, 1) aralığında değerleri üreterek belirlenmesini sağlar. Çoklu sınıflandırma problemlerinde yaygın olarak kullanılan matematiksel bir fonksiyondur (El-Amir ve Hamdy, 2020: 302).

Çizelge 3.1. Aktivasyon fonksiyonları ve grafikleri

Aktivasyon Fonksiyon Grafiği	Aktivasyon Fonksiyonu	Aralık
	$Sigmoid(x) = \sigma(x) = \frac{1}{1 + exp\ (-x)}$	(0, 1)
	$Tanh(x) = \frac{exp(x) - exp\ (-x)}{exp\ (x) + exp\ (-x)}$	(-1, 1)
	$ReLU(x) = \max(0, x)$	[0, ∞)
	$Swish(x) = xsigmoid(\beta x) = \frac{1}{1 + \exp\ (-\beta x)}$	(-∞, ∞)
	$Softmax(x_i) = \frac{exp(x_i)}{\sum_j exp\ (x_j)}$	(0, 1)

3.5. Derin Sinir Ağları

Derin sinir ağları, birbirine bağlı yapay nöronların çoklu katmanlarından oluşan yapay sinir ağlarıdır. Bu katmanlar, girdi verilerinin karmaşık modellenmesine izin vererek ağın verilerdeki kalıpları öğrenmesini ve tanımlamasını sağlar (Zaccone ve Karim, 2018: 18).

Bir başka deyişle derin sinir ağları, yapay sinir ağlarının yapısal olarak daha derinleşmiş ve genişlemiş hali olarak tanımlanabilir. Derin sinir ağlarındaki "derin" kelimesinden kastedilen, her biri girdi verileri üzerinde farklı türde bir hesaplama gerçekleştiren birbirini takip eden katmanları ifade etmesidir. Bu, daha basit bir modelle tespit edilmesi zor olan karmaşık kalıpları ve ilişkileri öğrenmelerini sağlar. Modeldeki katman sayısı modelin derinliğini oluşturmaktadır (Chollet, 2020: 8-9).

3.5.1. Derin sinir ağlarının mimari yapısı

Derin öğrenme, kalıpları öğrenmek ve büyük miktarda veriyi işleyerek tahminler yapmak için bir modeli eğitmeyi içeren bir tür yapay zekâdır. Bu, birbirine bağlı katmanlar halinde düzenlenmiş nöronların bir koleksiyonu olan bir sinir ağı kullanılarak elde edilir.

Derin sinir ağları, nöronların bilgiyi işleyen ve ileten birbirleriyle bağlantılı katman adı verilen yapılardan oluşmaktadır. Katmanlar derin sinir ağlarının yapı taşlarıdır. Belirli bir bilgi türünü işlemekten sorumludur ve birlikte, ağın verideki kalıpları öğrenmesine ve tanımasına yardımcı olur (Chollet, 2020: 58-59). Tipik bir derin öğrenme sinir ağında üç ana katman türü vardır: giriş katmanı, gizli katmanlar ve çıkış katmanı.

Giriş Katmanı: Sinir ağındaki girdi verilerini alan ve bir sonraki katmana ileten ilk katmandır.

Gizli katman: Giriş katmanından gelen bilgileri işleyerek bir sonraki gizli katmana veya çıktı katmanına iletir. Gizli katmanlar birkaç katman içerebilir ve gizli katmanların sayısı problemin karmaşıklığına bağlıdır. Bu katmanlar, girdi verilerinin işlenmesinden ve daha kullanışlı bir forma dönüştürülmesinden sorumludur.

Çıkış Katmanı: Girdi verilerine dayalı olarak çıktıyı veya tahmini üreten derin öğrenme modelinin son katmanıdır (Özkan, 2021: 111-114).

Tek katmanlı algılayıcı (Single layer perceptron- SLP)

1958'de Frank Rosenbluth tarafından önerilen algılayıcılar, yapay sinir ağlarının ilk modellerindendir. Tek katmanlı algılayıcı en ilkel yapay sinir ağı olarak değerlendirilir. Sadece iki katman bulunur; giriş ve çıkış katmanı. Ancak giriş katmanında herhangi bir hesaplama işlem yapılmadığından dolayı bu katman sayılmaz (El-Amir ve Hamdy, 2020: 62).

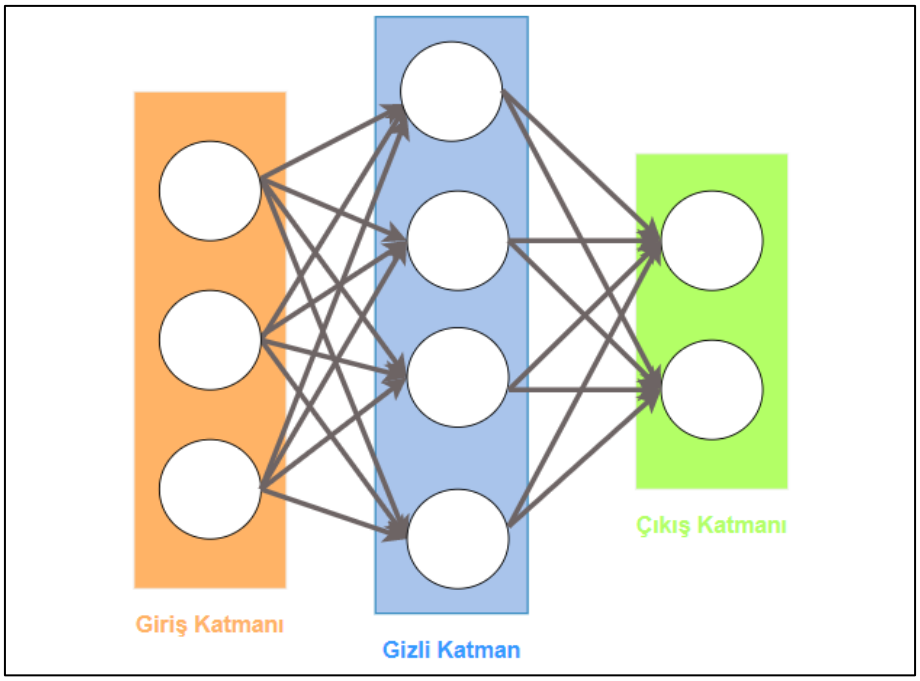
Tek katmanlı algılayıcıların karmaşık fonksiyonları modelleme yetenekleri sınırlıdır ve sadece doğrusal olarak ayrılabilen kalıpları öğrenebilirler. Bununla birlikte, görüntü tanıma, metin sınıflandırma ve spam filtreleme gibi basit sınıflandırma görevlerinde hala yaygın olarak kullanılmaktadırlar.

Tek katmanlı algılayıcıların karmaşık problemleri çözme yetenekleri sınırlı olduğundan çok katmanlı algılayıcılar (multilayer perceptron) geliştirilmiştir (Fernández, Soria, Martín ve Serrano, 2006).

Çok katmanlı algılayıcı (Multilayer perceptron- MLP)

Çok katmanlı algılayıcı, birbirine bağlı nöronların birden çok katmanından oluşan ve büyük miktarda veriyi işleyebilen bir tür yapay sinir ağıdır.

Çok katmanlı algılayıcı giriş katmanı (input layer), bir veya birden fazla gizli katman (hidden layer) ve çıkış katmanı (output layer) olmak üzere üç katmandan oluşmaktadır. Ağdaki her nöron bir önceki katmandan girdi alır, işler ve son çıktı üretilene kadar bir sonraki katmana iletir (Campesato, 2020: 110).



Şekil 3.4. Çok katmanlı algılayıcı (Nabiyev, 2021: 596)

Şekil 3.4'te çok katmanlı algılayıcının yapısı görselleştirilmiştir. Bu katmanlı gösterim, sinir ağı diye adlandırılan ve birbirini takip eden katmanları olan model sayesinde öğrenilmektedir.

Tek katmanlı algılayıcıdan farklı olarak giriş ve çıkış katmanları arasında gizli katman bulundurur. Tek katmanlı algılayıcının yeterli olmadığı karmaşık ve doğrusal olmayan problemlerin çözümünü yapabilmektedir. Çözülen problemin niteliğine göre, gizli katmanların sayısı ile bu katmanlarda bulunan nöron sayıları değişebilmektedir (Bento, 2021).

Çok katmanlı algılayıcıların ana avantajlarından biri verilerdeki karmaşık kalıpları ve ilişkileri öğrenme yetenekleridir, bu da onları birçok gerçek dünya problemini çözmek için uygun hale getirir. Ancak, en iyi performansı elde etmek için hiperparametrelerin dikkatli bir şekilde ayarlanmasını gerektirebilir (Nabiyev, 2021: 599).

Yapay sinir ağı modelleri, bir ağın öğrenmesi gereken giriş çıkış bilgilerine ve yapay sinirler arası ilişkilerle ifade edilen mimarilerine göre farklılıklar göstermektedir. Farklı öğrenme türleri, yapay sinir ağı modelini belirlemektedir (Nabiyev, 2021: 584-585).

3.5.2. Yapay sinir ağlarının öğrenmesi

Ağın öğrenme metodu genel olarak iki aşamada gerçekleşmektedir. İlk aşamada ileri doğru yayılım hesaplanır ikinci aşamada ise geriye doğru yayılım hesaplanmaktadır.

İleriye doğru yayılım (Feed forward propagation)

İleriye doğru yayılım, bir çıktı üretmek için girdi verilerini bir sinir ağı aracılığıyla iletme sürecini ifade eder. Başka bir deyişle, sinir ağının bir girdiden bir çıktı üretme yöntemidir. Bu işlem sırasında, girdi verileri, her katmanın verileri dönüştürmek için bir dizi matematiksel işlem gerçekleştirdiği ağın birkaç katmanından geçirilir. Her katmanın çıktısı, son çıktı üretilene kadar bir sonraki katmanın girdisi olur (Doğan, 2021: 19).

İleriye doğru yayılım algoritmasında gelen bilgiler yalnızca ileri doğru işlenmektedir yani bilgi giriş katmanından gizli katmanlara ve çıkış katmanına doğru ağ üzerinde ileri doğru hareket eder. İleri beslemeli yapay sinir ağlarında, girdi değerlerinin ağa sunulmasıyla ilk bilgiyi sağlanır, rastgele ağırlık değerlerinin atanmasıyla başlayan hesaplamaların gizli katmanlara ve en son olarak çıktı katmanına aktarılarak çıktı değeri üretilmesiyle sonlanan süreci ifade eder. Elde edilen çıktı değeri ile gerçek değer karşılaştırılarak hata değeri hesaplanır (Özkan, 2021: 116-117).

Geriye doğru yayılım (Backpropagation)

Geriye doğru yayılım, yapay sinir ağlarını eğitmek için yaygın olarak kullanılan bir algoritmadır. Tahmin edilen çıktı ile gerçek çıktı arasındaki farkı en aza indirmek için sinir ağının ağırlıklarının ayarlanmasını içeren bir denetimli öğrenme biçimidir. Bu algoritma hataları çıkış katmanından giriş katmanına doğru azaltmaya çalışmasından dolayı geri yayılım ismini almıştır.

Geriye doğru yayılım, eğitim sırasında bir sinir ağı modelinin parametrelerini güncellemek için makine öğrenmesinde kullanılan bir yöntemdir. Bu süreçte, modelin çıktısı gerçek hedef değerlerle karşılaştırılır ve hata, ağın katmanları boyunca geriye doğru yayılır. Bu, genel hatayı azaltmak ve tahminlerin doğruluğunu artırmak için ağın nöronlar arasındaki bağlantıların ağırlıklarını ve sapmalarını ayarlamasına izin verir. Geriye doğru yayılım,

Evrişimli Sinir Ağları ve Tekrarlayan Sinir Ağları gibi derin öğrenme modelleri dahil olmak üzere birçok popüler sinir ağı mimarisinin önemli bir bileşenidir (El-Amir ve Hamdy, 2020: 314-315).

Geriye doğru yayılım algoritması ileri geçiş ve geri geçiş olmak üzere iki aşamadan oluşur. İleri geçişte, girdi sinir ağı aracılığıyla beslenir ve her katmanın çıktısı hesaplanır. Son katmanın çıktısı, gerçek çıktıyla karşılaştırılır ve ikisi arasındaki hata hesaplanır. Geriye geçişte, hata çıkış katmanından başlayarak ağ boyunca geri yayılır. Her katmanın ağırlıklarına göre hatanın gradyanı, hesabın zincir kuralı kullanılarak hesaplanır ve ağırlıklar, bir gradyan iniş algoritması kullanılarak gradyanın ters yönünde güncellenir (Nabiyev, 2021: 598).

3.6. Optimizasyon Yöntemleri

Derin öğrenme uygulamalarında, eğitim aşamasında kayıp fonksiyonunu minimuma indirgemek için optimizasyon teknikleri kullanılmaktadır. Optimizasyon, ağın ürettiği çıkış değeri ile gerçek değer arasındaki farkı yani hatayı en küçük yapmak için kullanılan yöntemlerdir. Optimizasyon algoritmaları yapay sinir ağlarının oluşturulmasında önemli araçlardan birdir (Zhang vd., 2021: 435).

Kayıp fonksiyonu, gerçek y değeri ile tahmin edilen $\hat{y}$ arasındaki farkı veya hatayı ölçer. Bu, hatayı en aza indirmek ve yerel ve yerel/genel minimumu bulmak için parametreleri ayarlayabilmesi için modele geri bildirim sağlayarak derin öğrenme modelinin etkinliğini artırır. Kayıp fonksiyonu sıfıra yakın veya sıfırda olana kadar hareket ederek sürekli yinelenir. Sıfır olduğu noktada model öğrenmeyi durduracaktır (IBM, 2023).

Optimize ediciler, tahmini çıktı ile gerçek çıktı arasındaki hatayı en aza indirmek için bir sinir ağındaki ağırlıkları ve sapmaları ayarlamak için makine öğreniminde kullanılan algoritmalardır. Bu algoritmalar, hata en aza indirilene kadar her yinelemeden sonra sinir ağındaki ağırlıkları ve sapmaları güncelleyerek çalışır.

Her optimize edici için eğitim sürecinde kayıp fonksiyonunun hesaplanma şekliyle ilgili farklı varsayımlarda bulunurlar. Hangi tür gradyan inişin kullanılacağını seçerken dikkate alınması gereken avantajları ve dezavantajları vardır (El-Amir ve Hamdy, 2020: 305-311).

Yapay sinir ağlarının optimizasyonu için çeşitli algoritmalar mevcuttur. Optimizasyon algoritmalarında öğrenme oranının ayarlanması modelin eğitiminde kritik bir rol oynamaktadır. Ancak, her algoritma ile modeldeki öğrenme oranının tam olarak ayarlamak mümkün değildir. Bu problemi çözmek için gradyan yöntemlerinin çeşitli varyasyonları önerilmiştir.

Derin öğrenme modellerinin doğruluğunu ve hızını artırmak için gerekli olan optimize edicilerin seçimi ve veri türüne ve modelden istenen performansa bağlıdır. Popüler optimize ediciler arasında Gradyan İniş, Stokastik Gradyan İniş, Adagrad, RMSProp ve Adam bulunur (Campesato, 2020: 108).

3.6.1. Gradyan iniş

Gradyan iniş yöntemi yapay sinir ağlarının optimizasyonu için en temel ve en çok kullanılan optimizasyon algoritmalarından biridir.

Türevlenebilir bir fonksiyonun yerel minimum değerini bulmak için kullanılan bir optimizasyon algoritması olan gradyan inişinin ardındaki fikir, kayıp fonksiyonunu en aza indirgemek modelin parametrelerini yinelemeli olarak ayarlamaktır (Kwiatkowski, 2022). Eşitliği aşağıdaki gibi verilmektedir:

$$w_t = w_t - \eta \frac{\partial L}{\partial w_t} \tag{3.2}$$

Burada η öğrenme oranını göstermektedir. Bu öğrenme oranı ile gradyan çarpılarak w_t değeri güncellenmektedir.

Güncellemenin büyüklüğü, her yinelemede parametrelerin ne kadar ayarlanacağını belirleyen bir öğrenme oranı parametresi tarafından kontrol edilir. Algoritma, yakınsayana veya bir durdurma kriteri karşılanana kadar parametreleri güncellemeye devam eder. Gradyan iniş, etkili verimli ve ölçeklenebilir olduğu için derin öğrenmede güçlü ve yaygın olarak kullanılan bir algoritmadır (El-Amir ve Hamdy, 2020: 305-308).

Derin öğrenmenin önemli bir parametresi olan öğrenme oranı (learning rate), algoritmanın eğitim sırasında model parametrelerini güncellediği adım boyutunu belirler. Bu parametre, modelin performansı üzerinde güçlü bir etkiye sahiptir.

Öğrenme oranı çok küçükse, algoritmanın yakınsamak için birçok yinelemeden geçmesi gerekecektir. Bu da daha fazla zaman ve hesaplama gerektirdiğinden genel verimliliği kötü etkileyebilir. Öğrenme oranı çok büyükse, algoritma optimum noktaya yakınsamayabilir hatta optimal değeri atlayabilir. Dolayısıyla kullanılan optimizasyon algoritmasında doğru öğrenme oranını bulmak, eğitim sırasında iyi performans ve modelin hızlı bir şekilde yakınsaması için kritik öneme sahiptir (Buduma, 2015: 25-26).

Gradyan iniş algoritması, çoğu amaç için en iyi sonucu verir ancak, bazı dezavantajları bulunmaktadır. Konveks fonksiyonlar için kolaylıkla global minimumu bulabilirken, konveks olmayan fonksiyonlar için gradyan boyunca ne kadar uzağa gidileceğini bilmez, global minimumu bulmakta zorlanabilir ve yerel minimumlara takılabilir (IBM, 2023).

3.6.2. Stokastik gradyan iniş

Stokastik gradyan iniş, makine öğrenimi ve derin öğrenmede kullanılan popüler bir optimizasyon algoritmasıdır. Amaç fonksiyonunun gradyanını hesaplamak için eğitim verilerinin rasgele seçilmiş alt kümelerini kullanılan bu algoritma gradyan inişinin bir çeşididir.

Stokastik gradyan iniş, gradyan inişteki kayıp fonksiyonunun türevini hesaplamak için tüm veri kümesini bir seferde yüklemek için çok fazla bellek gerektirmesi sorununu çözmek için geliştirilmiştir (El-Amir ve Hamdy, 2020: 309-310). Eşitliği aşağıdaki gibi verilmektedir:

$$w_t = w_{t-1} - \eta \frac{\partial L}{\partial w_{t-1}} \tag{3.3}$$

Stokastik terimi, algoritmanın dayandığı rastgelelik anlamına gelir. Stokastik gradyan inişinde, her yineleme için tüm veri setini almak yerine veri gruplarını rastgele seçilir. Veri gruplarının rastgele seçilmesi, global bir minimuma ulaşmaya yardımcı olur ve yerel bir minimumda takılıp kalmayı önler (Gupta, 2021).

Stokastik gradyan iniş algoritmasının ana avantajı, özellikle büyük veri kümeleriyle uğraşırken standart gradyan iniş algoritmasından daha hızlı yakınsamasıdır. Bunun nedeni verileri küçük gruplar halinde işlemesidir, bu da gradyanı hesaplamanın hesaplama maliyetini azaltır (Zaccone ve Karim, 2018: 17).

Genel olarak, stokastik gradyan iniş, özellikle sinir ağlarını eğitmek için en yaygın kullanılan algoritma olduğu derin öğrenmede, modeli optimize etmek için yaygın olarak kullanılan etkili bir algoritmadır. Bununla birlikte, alt kümelerin rasgele örneklemesi nedeniyle stokastik gradyan iniş, gradyan tahmininde yüksek varyansa sahip olabilir ve bu da algoritmanın gürültülü bir şekilde yakınsamasına yol açabilir. Bu sorun, öğrenme hızını ayarlayarak, Adam veya RMSProp gibi daha karmaşık optimizasyon teknikleri kullanarak çözülebilir (El-Amir ve Hamdy, 2020: 309-310).

3.6.3. Momentum

Momentum, eğitim sırasında bir sinir ağının ağırlıklarını güncellemek için derin öğrenmede kullanılan popüler bir optimizasyon algoritmasıdır. Derin öğrenme için yaygın olarak kullanılan bir optimizasyon algoritması olan stokastik gradyan iniş (SGD) optimize edicinin bir uzantısıdır (Brownlee, 2021).

Momentum optimize edici, SGD optimize edicinin güncelleme kuralına, optimizasyon sürecinin yakınsamasını hızlandırmaya yardımcı olan bir momentum terimi ekler (Zhang vd., 2021: 475). Eşitliği aşağıdaki gibi verilmektedir:

$$w_t = w_{t-1} - \eta V_t \tag{3.4}$$

$$V_t = \beta V_{t-1} + (1-\beta)\frac{\partial L}{\partial w_{t-1}} \tag{3.5}$$

β önceki momentumun mevcut güncellemeye katkısını kontrol eden momentum katsayısı, 0 ile 1 arasında bir değer almaktadır ve genelde 0,9 değeri tercih edilir. Ancak belirli soruna ve veri kümesine göre ayarlanabilir.

SGD'de optimum nokta aranırken çok fazla salınım olmaktadır. Momentum terimi, yerel minimumları aşmasına ve küresel minimuma daha hızlı ilerlemesine yardımcı olur. Ayrıca

gradyan güncellemelerindeki salınımları ve gürültüyü azaltmaya olarak daha sorunsuz bir optimizasyon sürecine yol açar (Goodfellow, Bengio ve Courville, 2018: 296-299).

Genel olarak, momentum derin öğrenme modellerinin yakınsama hızını ve performansını önemli ölçüde artırabilen güçlü bir optimizasyon algoritmasıdır.

3.6.4. AdaGrad

Uyarlanabilir gradyan algoritması veya kısaca AdaGrad, eğitim sürecinde modelin öğrenme oranını ayarlamak için derin öğrenmede kullanılan bir optimizasyon algoritmasıdır. Öğrenme oranını her parametrenin gradyanına uyarlayan stokastik gradyan iniş algoritmasının bir uzantısıdır (El-Amir ve Hamdy, 2020: 349). Eşitliği aşağıdaki gibi verilmektedir:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{S_t + \epsilon}} \cdot \frac{\partial L}{\partial w_{t-1}} \tag{3.6}$$

$$S_t = S_{t-1} + \left[\frac{\partial L}{\partial w_{t-1}}\right]^2 \tag{3.7}$$

Burada S_t geçmiş gradyanların karelerinin toplamını ifade etmektedir. Burada ϵ işlemi sıfıra bölünme hatasından kaçınmak için genelde çok küçük bir sayı olarak alınmaktadır.

Bir algoritma için öğrenme oranının ayarlanma zorluğundan kaynaklanan parametre güncellemesinin yavaşlığı veya minimum değere ulaşılamama gibi sorunları çözmek için her boyut için farklı öğrenme oranı seçilmesi önerilebilir. Ancak derin sinir ağlarında çok sayıda veri olduğundan dolayı bu yaklaşım kullanışlı olmaz (Gylberth, 2018).

Optimizasyon sürecini hızlandırmak ve optimizasyon algoritmasının kapasitesini geliştirmek için tasarlanan Adagrad algoritmasının arkasındaki fikir, her parametre için farklı öğrenme oranı kullanarak her bir parametre için farklı güncelleme yapmaktadır. Bu sayede öğrenme oranını manuel olarak ayarlama ihtiyacını ortadan kaldırarak otomatik bir şekilde gerçekleştirir (Çarkacı, 2020).

Stokastik gradyan iniş algoritmasında ise öğrenme oranı sabittir ve eğitim süreci boyunca sabit kalır. Bu yaklaşım, özellikle girdi verileri seyrek veya gürültülü olduğunda yavaş yakınsamaya yol açabilir. Bazen verilerde pek görünmeyen seyrek özellikler bir problemin çözümü için çok yararlı olabilir. Bununla birlikte, gradyan iniş veya stokastik gradyan iniş algoritmalarında, öğrenme oranı her yinelemede tüm özelliklere eşit önem verir. Öğrenme oranı aynı olduğu için, seyrek olmayan özniteliklerin toplam katkısı, seyrek özniteliklerden çok daha fazla olacaktır. Bu nedenle, seyrek özelliklerden kritik bilgileri kaybedilir. Adagrad, öğrenme oranını önceki eğimlere göre ayarlayarak bu sorunu giderir (El-Amir ve Hamdy, 2020: 349-350).

3.6.5. RMSProp

Kareler Ortalaması Karekökünün Yayılımı anlamına gelen RMSProp derin öğrenmede yaygın olarak kullanılan bir optimizasyon algoritmasıdır. Derin sinir ağlarını eğitirken meydana gelebilecek kaybolan ve patlayan gradyan problemini çözmek için tasarlanmış gradyan iniş ve AdaGrad optimizasyon algoritmasının gelişmiş bir uzantısıdır (El-Amir ve Hamdy, 2020: 351).

RMSProp, AdaGrad algoritmasının eksikliklerinin üstesinden gelmek için geliştirilmiştir. AdaGrad algoritması güncellemede tüm geçmiş gradyanları kullandığından dolayı öğrenme oranını agresif bir şekilde azaltır. Sonuç olarak, bir süre sonra azalan öğrenme oranı nedeniyle çok küçük güncellemeler almaya başlayacaktır. Dolayısıyla belirli bir sayıda eğitim adımından sonra kaybı iyileştirmez, eğitim durdurulana kadar kaybın sabit kalmasına neden olur. Eşitliği aşağıdaki gibi verilmektedir:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{S_t + \epsilon}} \cdot \frac{\partial L}{\partial w_{t-1}} \tag{3.8}$$

$$S_t = \beta S_{t-1} + (1-\beta)\left[\frac{\partial L}{\partial w_{t-1}}\right]^2 \tag{3.9}$$

RMSProp algoritmasında gradyanların karelerini almak yerine momentumlu gradyanların karelerini almaktadır, yani karesi alınmış gradyanların hareketli bir ortalamasını kullanır.

Burada β geçmiş gradyanların işleme ne kadar katılacağını ayarlamak için kullanılır, 0 ile 1 arasında bir değer almaktadır ve genelde 0,9 değeri tercih edilir (Ruder, 2016).

RMSProp, modeldeki her parametre için o parametrenin karesi alınmış gradyanının hareketli ortalamasını kullanarak uyarlanabilir bir öğrenme oranı hesaplar. Bu, gürültülü gradyan güncellemelerinin etkisini azaltmaya yardımcı olur ve optimizasyon sürecini daha kararlı hale getirir. Bu sayede öğrenme oranını çok hızlı düşürmez (Chandra, 2021).

Algoritma aynı zamanda yakınsama oranını hızlandırmaya yardımcı olan geçmiş güncellemelerden gelen bilgileri dahil etmek için bir momentum terimi kullanır. RMSProp algoritması, yerel minimuma ulaşılana kadar eğitim süreci boyunca kaybı sürekli olarak azaltabilir (Machine Learning Journey, 2021).

Genel olarak, RMSProp, derin sinir ağlarını eğitmek için etkili ve popüler bir optimizasyon algoritmasıdır. Özellikle büyük ve seyrek özellikteki veri setlerine sahip modeller ile optimize edilmesi gereken birçok parametreye sahip modeller için uygundur.

3.6.6. Adaptif moment tahmini (Adaptive moment estimation- Adam)

Adaptif Moment Tahmini, kısaca Adam optimizasyon algoritması son zamanlarda derin öğrenme çalışmalarında kullanılan en popüler algoritmadır. Uygulama kolaylığı, çok az bellek gereksinimi ve hesaplama açısından verimli olmasından dolay en çok tercih edilen algoritmadır.

"Adam" adı, optimize edicinin eğitim sürecinde yakınsamayı hızlandırmak için uyarlanabilir bir öğrenme hızı ve momentum parametreleri kullandığı gerçeğini yansıtan uyarlanabilir moment tahmini anlamına gelir. Bu optimizasyon algoritması, eğitim sırasında ağ ağırlıklarını güncellemek için stokastik gradyan inişinin bir başka uzantısıdır. Aşağıdaki eşitlikler kullanılarak hesaplamalar yapılmaktadır:

$$V_t = \beta_1 V_{t-1} + (1 - \beta_1) \frac{\partial L}{\partial w_{t-1}} \tag{3.10}$$

$$S_t = \beta_2 S_{t-1} + (1-\beta_2)\left(\frac{\partial L}{\partial w_{t-1}}\right)^2 \quad (3.11)$$

$$V_t' = \frac{V_t}{1-\beta_1^t} \quad (3.12)$$

$$S_t' = \frac{S_t}{1-\beta_2^t} \quad (3.13)$$

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{S_t' + \epsilon}}.V_t' \quad (3.14)$$

Adam algoritması sinir ağlarını optimize etmek için AdaGrad ve RMSProp algoritmalarının en iyi özelliklerini birleştirir ve problemlerin çözümünde çok daha yüksek performans gösterir.

Adam algoritması ilk olarak gradyanın (V_t) ve birinci ve ikinci momentin tahminleri olan kare gradyanın (S_t) üstel hareketli ortalamalarını günceller. Hiperparametreler β_1 ve β_2 0 ile 1 aralığında değer alır. β_1 için varsayılan değer 0,9 β_2 için varsayılan değer ise 0,999'dur. . η öğrenme oranını, V_t geçmiş gradyanların karelerinin üstel olarak ağırlıklandırılmış ortalamalarını, S_t momentumdaki geçmiş gradyanların üstel olarak ağırlıklandırılmış ortalamalarını ifade etmektedir.

Stokastik gradyan inişinde tek bir öğrenme oranını korumanın aksine, Adam algoritmasında öğrenme oranını her bir ağ ağırlığı için ayrı ayrı günceller. Farklı parametreler için bireysel öğrenme oranlarını hesaplanır ve hem momentumun hem de gradyanın ikinci momentinin bir tahminini elde etmek için üstel ağırlıklı hareketli ortalamaları kullanır (Zhang vd., 2021: 497-501).

Adam optimize edicisi, özellikle büyük veri kümeleri ve karmaşık modeller için hızlı ve verimli bir şekilde yakınsama yeteneği nedeniyle derin öğrenme uygulamalarında yaygın olarak kullanılmaktadır (Khandelwal, 2019).

3.7. Derin Öğrenme Mimarileri

Derin öğrenme mimarisi, giderek daha karmaşık hale gelen modellerin öğrenilmesini sağlayan yapay sinir ağlarının yapısını ifade eder. Derin öğrenme mimarileri, her biri girdi verilerinin farklı yönlerini öğrenen ve işleyen çok sayıda birbirine bağlı yapay nöron katmanından oluşur. Derin öğrenme mimarisi, sinir ağlarını öğrenmek ve tahminler yapmak için büyük miktarda veriyle eğitmeyi içeren makine öğreniminin bir alt kümesi olan derin öğrenme için kullanılan bir sinir ağının tasarımını ve yapısını ifade eder (Zaccone & Karim, 2018: 9-11).

Bir derin öğrenme modelinin mimarisi, belirli soruna ve mevcut verilere bağlı olarak değişebilir, ancak genellikle birbirine bağlı nöronların birden çok katmanından oluşur. Girdi katmanı verileri alır ve daha sonra her biri veriler üzerinde matematiksel işlemler gerçekleştiren bir dizi nöron içeren bir veya daha fazla gizli katmandan geçirilir. Nihai çıktı katmanı, işlenen verilere dayalı olarak tahminleri veya sınıflandırmaları üretir.

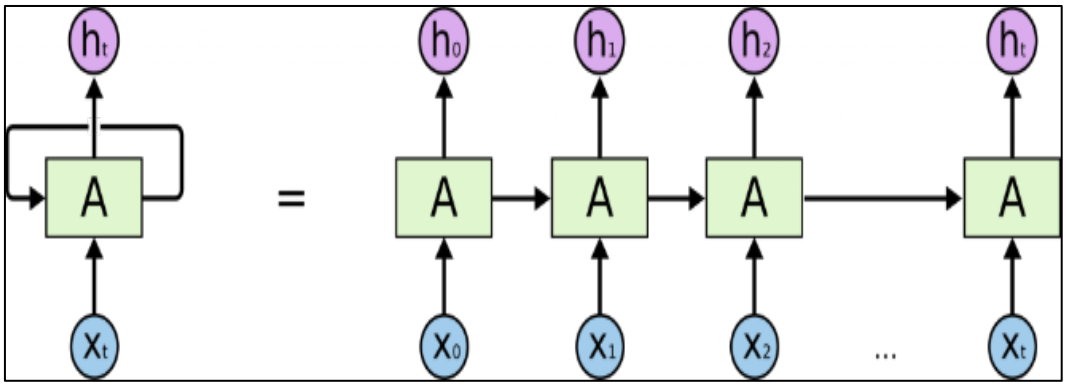
Bazı popüler derin öğrenme mimarileri arasında, her biri benzersiz yapıya ve tasarıma sahip Evrişimli Sinir Ağları, Tekrarlayan Sinir Ağları ve Üretken Çekişmeli Ağlar (Generative Adversarial Networks- GAN) bulunur.

Genel olarak, bir derin öğrenme modelinin mimarisi, performansında ve verilerden öğrenme yeteneğinde kritik bir rol oynar ve bu da onu herhangi bir derin öğrenme projesi için önemli bir konu haline getirir. Farklı problemlerde kullanılmak üzere çeşitli derin öğrenme mimarileri mevcuttur (Campesato, 2020: 101-103).

3.7.1. Tekrarlayan Sinir Ağları (Recurrent Neural Network- RNN)

İnsanoğlu sıfırdan düşünmeye başlamaz. İnsan zihni bir bilgiyi kavrayabilmesi ve o bilgiye yanıt oluşturabilmesi için geçmişte öğrenilen bilgiler ile yeni öğrenilmiş bilgileri ilişkilendirerek kullanma yeteneğine sahiptir. Geleneksel sinir ağları geçmiş bilgileri görmezden gelir, girdilerin ve çıktıların birbirinden bağımsız olduğunu varsayar. Tekrarlayan Sinir Ağları ise geleneksel sinir ağlarının aksine, önceki ögelere bağlıdır ve bilginin hatırlanmasını sağlayan bir döngüye sahiptir (Zaccone ve Karim, 2018: 236).

Tekrarlayan Sinir Ağları ardışık verileri işlemek için kullanılan bir derin öğrenme mimarisidir. Mevcut girdi ve çıktıyı etkilemek için önceki girdilerden elde edilen bilgileri saklamaya yarayan bir belleğe sahiptir ve bu hafıza özelliği ile diğer sinir ağlarından ayırt edilirler (IBM, 2023).



Şekil 3.5. Tekrarlayan Sinir Ağı yapısı (Zaccone ve Karim, 2018: 236)

Tekrarlayan Sinir Ağı yapısı Şekil 3.5'te verilmiştir. RNN mimarilerinde bilginin bir zaman adımından diğerine geçmesine izin veren döngülere sahip olmaları, önceki zaman adımlarının çıktısının mevcut zaman adımına girdi olarak geri beslenmesine izin verir. Bu geri bildirim döngüsü, RNN'leri doğal dil işleme, kelime ve cümle tahmini, hava durumu tahmini, konuşma tanıma, dil çevirileri, duygu analizi ve borsa tahmini gibi zaman serisi verileri veya sıralı veriler için çok uygun hale getirir (Campesato, 2020: 127-128).

$$h_t = f(W.x_t + U.h_{t-1}) \tag{3.15}$$

Yinelemeli ağlarda girdiler geçmiş ve şimdiki bilgilerin birleştirilmesi ile çıktı üretirler. Burada, x_t, t anındaki giriş değeri, W ve U değerleri girdi ile katman arasındaki ağırlıkları, h_{t-1} son anın çıkış değerini, h_t, t anındaki gizli katmanın sonucunu temsil etmektedir. x_t girdi değerleri W ağırlığı ile çarpılır ve t-1 anında bellekte tutulan h_{t-1} değeri U ağırlığı ile çarpılır sonrasında bu iki değer toplanır. Elde edilen değer sigmoid, tanh gibi f aktivasyon fonksiyonuna sokulur ve h_t değeri hesaplanır (Biswal, 2021).

RNN güçlü ve popüler derin öğrenme mimarilerinden biridir. Bununla birlikte, RNN mevcut problemi çözmek için uzun zaman önceki bilgilerden ziyade genellikle son bilgilere bakar.

Eğer ilgili bilgi ile ihtiyaç duyulan yer arasındaki boşluk daha büyükse, RNN'ler geçmiş bilgileri kullanmayı ve bağlamayı öğrenemez (Zaccone ve Karim, 2018: 243-244).

RNN yapısının bir diğer dezavantajı ise gradyanların yok olması veya patlaması sorunudur. Daha derin katmanlar için gradyanlar, çok katmanlı ağdaki birçok aktivasyon fonksiyonu gradyanının çarpımı olarak hesaplanır. Ağırlıklar küçükse, gradyan değerleri o kadar küçülür ki öğrenme ya çok yavaşlar ya da tamamen durur. Bu durum kaybolan gradyanlar olarak adlandırılır. Ağırlıkların büyük olması ise patlayan gradyan olarak adlandırılan, ağın çok büyük değerler üretmesine yol açarak doğru sonuçtan uzaklaşmasına neden olan bir problemdir.

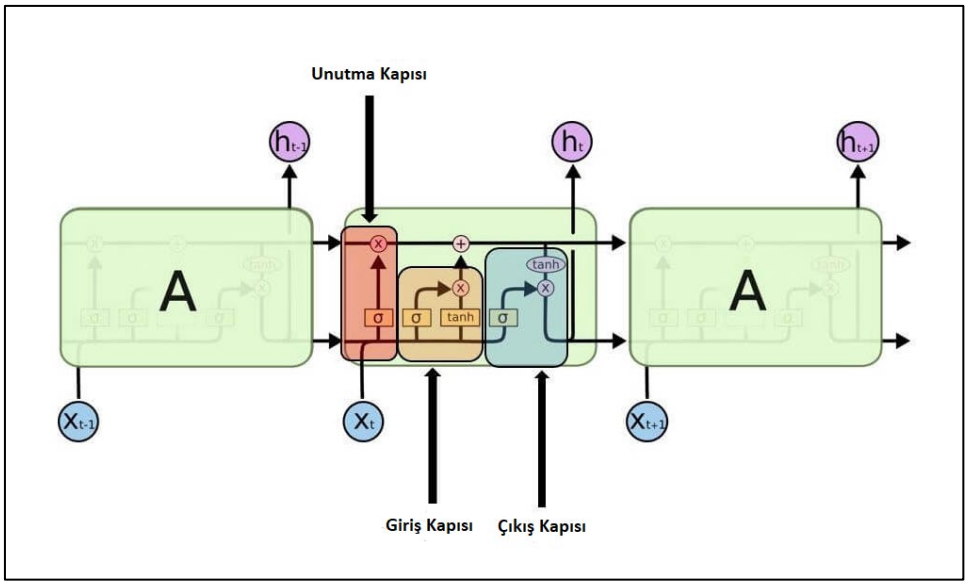
Tekrarlayan Sinir Ağlarının en popüler varyantlarından biri, uzun diziler üzerinde eğitim yapılırken oluşabilen ve performansı doğrudan etkileyen kaybolan-patlayan gradyan problemini çözmek için tasarlanmış LSTM ağlarıdır (Zaccone ve Karim, 2018: 246-248).

3.7.2. Uzun Kısa Vadeli Bellek (Long Short-Term Memory- LSTM)

Uzun Kısa Vadeli Bellek (LSTM), Sepp Hochreiter ve Juergen Schmidhuber tarafından1997 yılında geleneksel Tekrarlayan Sinir Ağlarının yok olan gradyan sorununun üstesinden gelmek için tasarlanmış, derin öğrenmede kullanılan özel bir tür Tekrarlayan Sinir Ağı mimarisidir (Zaccone ve Karim, 2018: 249).

Tekrarlayan Sinir Ağları gelişmiş bir sürümü olan LSTM kısa vadeli bellek problemini çözmek için geliştirilmiştir ve bilgileri daha uzun süre hatırlama yeteneğine sahiptir. Teoride RNN'ler uzun vadeli bağımlılıkları öğrenme yeteneğine sahiptir ancak pratikte uzun dizilerde iyi sonuç vermediği görülmüştür.

Tekrarlayan Sinir Ağlarının kısa vadeli bir hafızası olmasından dolayı geçmiş bilgileri hatırlamakta güçlük çeker ve doğal dil işleme, zaman serisi tahminleri gibi bazı problemlerde iyi sonuç vermez (El-Amir ve Hamdy, 2020: 433).



Şekil 3.6. Uzun Kısa Vadeli Bellek yapısı (Akköse, 2021)

Şekil 3.6'da verilen LSTM yapısı, RNN ile benzer bir yapıya sahiptir ancak RNN'den farklı olarak özel bir şekilde etkileşime giren yapılar bulunmaktadır. LSTM mimarisi, bilgileri uzun bir süre boyunca depolayabilen birden çok bellek hücresinden oluşur. Her bellek hücresi, hücreye giren ve çıkan bilgi akışını kontrol eden bir kapı mekanizması ile donatılmıştır. Bu kapı mekanizmaları, giriş kapısı, unutma kapısı ve çıkış kapısını içerir. Bu kapıları kullanarak, LSTM'ler bilgileri seçerek hatırlayabilir ve unutabilir, bu da onları uzun vadeli bağımlılıklarla sıralı verileri işlemede daha etkili hale getirir.

Bellek hücresi (Cell state)

Anlamlı bilgileri hücreler boyunca taşıyan bir iletişim hattı ve ağın hafızası olarak açıklanabilir. Bu sayede kısa süreli bellek problemi çözülerek eski veriler ağ zinciri boyunca taşınabilir. Bellek hücresinin bu yolculuğu boyunca taşıması gereken bilgiler ise kapılar aracılığı ile belirlenir. Bu kapılar hangi bilginin gerekli veya gereksiz olduğunu belirleyebilir.

Unutma kapısı (Forget gate)

Hangi bilgilerin unutulacağı veya tutulacağı kararına varan kapıdır. Bir önceki hücreden gelen bilgiler (h_t) ve mevcut bilgiler (x_t) sigmoid aktivasyon fonksiyonuna sokulur ve

sonucuna göre karar verilir. Elde edilen sonuçta 0 olan bilgiler unutulur ve 1 olan bilgiler bellek hücresi ile taşınmaya devam eder.

Giriş kapısı (Input gate)

Giriş kapısı, bellek hücresine hangi bilgilerin eklendiğini kontrol eder ve bellek hücresi güncellemesi yapar. Önceki ve mevcut bilginin sigmoid işlemi sonucuna göre güncelleme yapıp yapılmayacağı kararına varılır. 0 olan bilgi önemsiz ve 1 olan bilgi önemli olarak kabul edilir. Ayrıca ağı düzenleme işlemi için veriyi -1 ve 1 aralığına sıkıştıran tanh aktivasyon fonksiyonu da kullanılır. Daha sonrasında sigmoid ve tanh fonksiyonu çıkışları çarpılır ve hangi bilginin güncelleneceği kararına varılır.

Çıkış kapısı (Output gate)

Çıkış kapısı bir sonraki katmana gönderilecek bilgilere karar verir. Çıkış kapısı ise bir sonraki hücrenin girişini (h_{t+1}) belirler. Öncelikle önceki bilgiyi ve mevcut girişin bilgisini sigmoid fonksiyondan geçirilir. Daha sonra bellek hücresi üzerinde var olan bilgiyi tanh fonksiyonundan geçirilir. Son olarak iki sonucu çarparak hangi bilgilerin bir sonraki hücre için giriş (h_{t+1}) olacağına karar verilir. Mevcut hücre için kapı işlemleri tamamlandığında bir sonraki hücreye gidecek olan bellek hücresi ve hücrenin giriş bilgisi olarak tanımlanan gizli durum (h_t) bilgilerine karar verilmiş olur (Akköse, 2021).

Üç kontrol kapısından ve bellek hücresinden yararlanan bu mekanizma, LSTM'lerin bilgileri daha uzun süreler boyunca tutmasını sağlayarak, onları uzun vadeli bağımlılıklara sahip veri dizilerini işlemeyi gerektiren görevler için uygun hale getirir. LSTM'ler, doğal dil işleme, konuşma tanıma, görüntü alt yazısı ve zaman serisi tahmini dahil olmak üzere çok çeşitli uygulamalarda kullanılmıştır.

LSTM ağındaki işlemler aşağıda belirtilen adımlar şeklinde yürütülür:

Adım 1

İlk adımda unutma kapısı ile ilgili işlemler yürütülür. Hücre durumunda artık kullanışlı olmayan bilgiler, unutma kapısı ile kaldırılır. Son anın çıkış değeri ve mevcut zamanın giriş

değeri, unutma kapısına girilir ağırlık matrisleriyle çarpılır ve yanlılık değeri eklenir. Unutma kapısının çıkış değeri, aşağıdaki formülde gösterildiği gibi hesaplamadan sonra elde edilir:

$$f_t = \sigma(W_f \,.\, [h_{t-1}, x_t] + b_f) \tag{3.16}$$

Burada σ sigmoid aktivasyon fonksiyonu temsil etmektedir. W_f unutma kapısının ağırlığı, b_f unutma kapısının yanlılığı, x_t ise t anındaki giriş değeri, h_{t-1} son anın çıkış değerini simgeler.

f_t 'nin değer aralığı (0,1) arasındadır. Unutma kapısında sigmoid aktivasyon fonksiyonu o an için neyi unutacağına ve neyi hatırlayacağına karar verilmesini sağlar. Elde edilen sonuç, ikili çıktı veren sigmoid aktivasyon fonksiyonundan geçirilir. Belirli bir hücre durumu için çıktı 0 ise, bilgi parçası unutulur aksi halde çıktı 1 ise bilgi ileride kullanılmak üzere saklanır.

Adım 2

Bu aşamada yeni belleğe hangi bilgilerin yüklenmesi gerektiğine karar verilir. Hücre durumuna faydalı bilgilerin eklenmesi giriş kapısı tarafından yapılır. Son zamanın çıkış değeri ve şimdiki zamanın giriş değeri giriş kapısına girilir ve giriş kapısının çıkış değeri ve aday hücre durumu, aşağıdaki formüller ile hesaplamadan sonra elde edilir.

$$i_t = \sigma(W_i \,.\, [h_{t-1}, x_t] + b_i) \tag{3.17}$$

$$\tilde{C}_t = \tanh(W_c \,.\, [h_{t-1}, x_t] + b_c) \tag{3.18}$$

Burada i_t'nin değer aralığı (0, 1) arasındadır ve W_i giriş kapısının ağırlığı, b_i giriş kapısının yanlılığı(sapması), W_c aday giriş kapısının ağırlığı, b_c aday giriş kapısının yanlılığını simgeler.

Bu adımda, bellekte hangi yeni bilgilerin depolanacağına karar verilmektedir. Bu katman iki bölümden oluşmaktadır. Birinci bölümde giriş kapısı adı verilen ve hangi değerlerin güncelleştirileceğine karar veren bir sigmoid katmanı yer alır. İkinci bölümde ise bir tanh

katmanı çıktısı olarak $\tilde{C}_t$ yeni aday anılarının bir vektörünü oluşturur. Sonraki adımda, güncellemeyi oluşturmak üzere bu ikisi birleştirilir.

Adım 3

Bu adımda c_t belleğinin güncellenmesine sıra gelmiştir. c_t'nin değer aralığı (0, 1) arasındadır. Birinci ve ikinci adımda elde edilen çıktıların toplanmasıyla c_t durumu elde edilir. Elde edilen sonuç bir sonraki c_{t+1} durumu için giriş olarak değerlendirilir. Mevcut hücre durumu aşağıdaki gibi güncellenir:

$$c_t = f_t * c_{t-1} + i_t * \tilde{C}_t \tag{3.19}$$

Adım 4

Bu adımda mevcut c_t bellek durumundan h_t çıkışı düzenlenerek bir sonraki bellek durumuna geçiş için son hazırlık yapılır. Bu katmandan yapılacak çıkışlar için bazı düzenlemeler yapılır. Bellek durumunun hangi kısımlarının çıktısının verileceğine sigmoid aktivasyon fonksiyonu ile karar verilir.

$$o_t = \sigma(W_o \,.\, [h_{t-1}, x_t] + b_o) \tag{3.20}$$

o_t'nin değer aralığı (0, 1) arasında olup W_o çıkış kapısının ağırlığını ve b_o çıkış kapısının yanlılığını simgeler.

Adım 5

LSTM çıkış değeri aşağıda gösterildiği gibi çıkış kapısından elde edilen çıkış değeri ve hücre durumu hesaplanarak elde edilir.

$$h_t = o_t * \tanh(c_t) \tag{3.21}$$

Bellek durumu tanh fonksiyonu ile (-1, 1) aralığına dönüştürülür. Sigmoid geçidinin çıktısı ile tanh çıktıları çarpılır ve böylece sadece karar verilen kısımlar çıkışa hazırlanır (Özkan, 2021: 173-175).

4. PERFORMANS DEĞERLENDİRME ÖLÇÜMLERİ

Tahmine dayalı modellerin performansını değerlendirmek amacıyla kullanılan bazı ölçümler bulunmaktadır. Değerlendirme ölçümlerinden MAE, MAPE, MSE ve RMSE modellerin doğruluğunu değerlendirmek için kullanılan yaygın olarak kullanılan performans ölçümleridir. Bu ölçümler, farklı modellerin performansını karşılaştırmak veya tek bir modelin kalitesini değerlendirmek için kullanılabilir.

4.1. Ortalama Mutlak Hata (Mean Absolute Error- MAE)

Veri kümesindeki gerçek ve tahmin edilen değerler arasındaki mutlak farkın ortalamasını temsil eder. Model tahminindeki hataların ortalama büyüklüğünü ölçer. Tahmin edilen ve gerçek değerler arasındaki mutlak farkların ortalaması alınarak hesaplanır:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \widehat{y_t}| \tag{4.1}$$

4.2. Ortalama Mutlak Hata Yüzdesi (Mean Absolute Percentage Error- MAPE)

Bu ölçüm, MAE'ye benzer, ancak tahmin edilen ve gerçek değerler arasındaki ortalama yüzde farkını ölçer. Öngörülen ve gerçekleşen değerler arasındaki mutlak yüzde farkların ortalaması alınarak hesaplanır:

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}\left|\frac{y_t - \widehat{y_t}}{y_t}\right| \tag{4.2}$$

4.3. Hata Kareler Ortalaması (Mean Squared Error- MSE)

Modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkın karesinin ortalamasını temsil eder. Tahmin hatalarının varyansını ölçer:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_t - \widehat{y_t})^2 \tag{4.3}$$

4.4. Hata Kareler Ortalamasının Karekökü (Root Mean Squared Error- RMSE)

Modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkı ölçer, yani tahmin hatalarının standart sapmasıdır. Tahmin edilen ve gerçekleşen değerler arasındaki farkların karelerinin ortalamasının karekökü alınarak hesaplanır:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{4.4}$$

Modellerin performansını ölçmek için kullanılan bu istatistikler gerçek değerler ile tahmin değerleri arasındaki fark olarak ifade edilebilen hata değerlerine dayalı ölçümlerdir. Analiz sonucunda hesaplanan bu ölçütlerde daha düşük hata değerine sahip model tercih edilir, modelin diğer modellere göre daha başarılı olduğu anlamına gelir (Chugh, 2022; Verly Lopes, Bobadilha ve Peres Vieira Bedette, 2021; Zeroual, Harrou, Dairi ve Sun, 2020).

5. UYGULAMA

Çalışmada istatistiksel zaman serisi modelleri ve derin öğrenme modelleri kullanılarak döviz kurlarının fiyat tahmini yapılmıştır. Uygulama Yahoo Finance internet sitesi üzerinden elde edilen USD/TRY ve EURO/TRY döviz kurlarına ait tarihsel iş günü verilerini içermektedir (Yahoo Finance, 2023). Döviz kurları için veri seti olarak 3 Ocak 2005 ile 31 Aralık 2020 tarihleri arasındaki kapanış değerleri alınmıştır. Veri seti toplam 4174 gözlem içermektedir. Veri setini %90 eğitim verisi ve %10 test verisi olmak üzere ikiye ayrılarak istatistiksel zaman serisi modelleri ve derin öğrenme modelleri kullanılarak analiz yapılmıştır.

Uygulamada kodlama dili olarak zaman serisi analizi için R programlama ve derin öğrenme modellerini için Python kullanılmıştır.

USD/TRY ve EUR/TRY döviz kurlarına ait geçmiş veriler, günlük bazda olup açılış, en düşük, en yüksek, kapanış ve düzenlenmiş kapanış değerlerini içermektedir. Döviz kurlarına ait tarihsel veriler Çizelge 5.1 ve Çizelge 5.2'de verilmiştir. Çalışma kapsamında sadece kapanış değerleri kullanılmıştır.

Çizelge 5.1. USD/TRY kuruna ait veri seti

Tarih	Açılış	Yüksek	Düşük	Kapanış	Düz. Kapanış
3/01/2005	1,34550	1,34550	1,33680	1,34400	1,34400
4/01/2005	1,33680	1,35100	1,33680	1,35100	1,35100
5/01/2005	1,35450	1,38650	1,35450	1,37400	1,37400
6/01/2005	1,37800	1,40580	1,37800	1,39250	1,39250
7/01/2005	1,38700	1,39400	1,36150	1,39400	1,39400
...	...	...	...	...	...
25/12/2020	7,57760	7,58715	7,53880	7,57410	7,57410
28/12/2020	7,57982	7,57982	7,40330	7,56950	7,56950
29/12/2020	7,44550	7,47330	7,34350	7,43940	7,43940
30/12/2020	7,38830	7,39280	7,30550	7,38880	7,38880
31/12/2020	7,37373	7,48725	7,34308	7,37373	7,37373

Çizelge 5.2. EUR/TRY kuruna ait veri seti

Tarih	Açılış	Yüksek	Düşük	Kapanış	Düz. Kapanış
3/01/2005	1,81250	1,81300	1,80140	1,80900	1,80900
4/01/2005	1,78000	1,80420	1,78000	1,80050	1,80050
5/01/2005	1,76300	1,83790	1,76300	1,82100	1,82100
6/01/2005	1,82700	1,85650	1,82370	1,83350	1,83350
7/01/2005	1,83000	1,83000	1,79490	1,81900	1,81900
...	...	...	...	...	...
25/12/2020	9,22228	9,25715	9,21691	9,22228	9,22228
28/12/2020	9,25200	9,25495	9,04151	9,24027	9,24027
29/12/2020	9,08603	9,14732	8,99853	9,08455	9,08455
30/12/2020	9,05423	9,08000	8,95553	9,04856	9,04856
31/12/2020	9,06843	9,19777	9,03118	9,06038	9,06038

5.1. Döviz Kurlarına Ait Veri Seti için Zaman Serisi Analizi

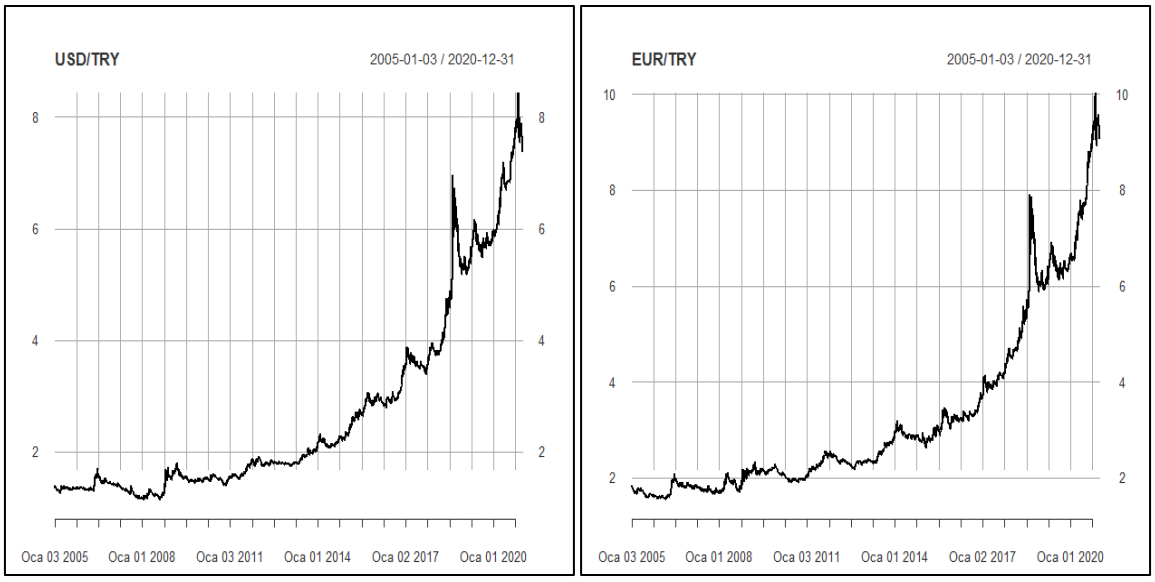
Uygulamanın ilk aşamasında döviz kurlarına ait serilerin tanımlayıcı istatistikleri hesaplanmıştır ve Çizelge 5.3'te yer verilmiştir. USD serisi için gerçekleştirilen analizler sonucunda basıklık katsayısı 0,9423, çarpıklık katsayısı ise 1,4165'tir, EURO serisi için ise basıklık katsayısı 1,5460, çarpıklık katsayısı ise 1,5498'dir, her iki döviz kuru serisinin sağa çarpık olduğu söylenebilmektedir. Ayrıca serilerin normal dağılıp dağılmadığını kontrol etmek için Jarque-Bera testine bakıldığında, USD serisi için test istatistiği değeri 1551,8869, EURO serisi için test istatistiği değeri 2088,9088 olarak elde edilmiştir. Sonuç olarak serilerin normal dağılıma uyduğunu ifade eden yokluk hipotezi %5 anlamlılık düzeyinde reddedilmiştir ($p - değeri = 0{,}0001 < \alpha = 0{,}05$). Elde edilen sonuçların doğrultusunda döviz kurlarına ait serilerin normal dağılmadığı tespit edilmiştir.

Finansal zaman serileri yapıları gereği çoğu zaman normal dağılımdan farklılık gösterir ve bu farklılıklar finansal serilerde ani ve büyük hareketler olduğunu ifade eder. Veri setlerine ilişkin tanımlayıcı istatistikler incelendiğinde USD ve EURO döviz kuru serilerinin durağan olmadığına ve değişen varyans yapısına sahip olabileceklerine ilişkin göstergeler sergilemiştir.

Çizelge 5.3. Tanımlayıcı istatistikler

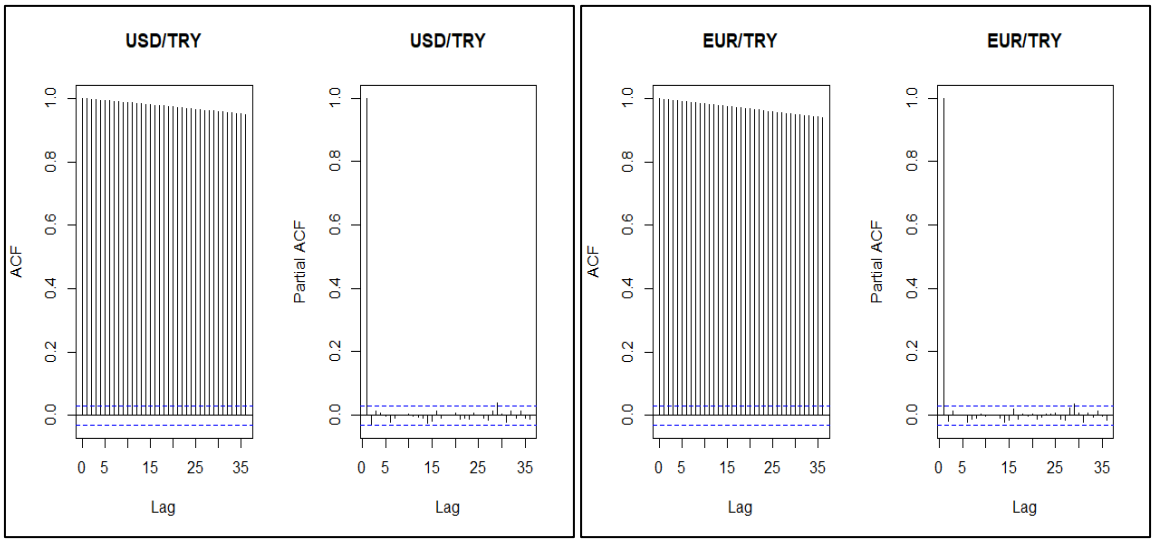
	Ortalama	Maksimum	Minimum	Varyans	Std. Sapma	Çarpıklık	Basıklık
USD	2,6838	8,4405	1,1423	2,9664	1,7223	1,4165	0,9423
EUR	3,2456	10,0310	1,5470	3,4565	1,8592	1,5498	1,5460

Zaman serisi modelinin belirlenmesi ve tahmin aşamasına geçmeden önce serilerin durağan olup olmadığının araştırılması gerekmektedir. Durağanlığı araştırmak amacıyla ilk olarak USD/TRY ve EUR/TRY döviz kuru serilerine ait zaman serisi grafikleri çizdirilmiş ve Şekil 5.1'de gösterilmiştir.



Şekil 5.1. Döviz kurlarının zaman serisi grafiği

Döviz kurlarının zaman serisi grafikleri incelendiğinde her iki serininde durağan olmadığı ve yükselen bir trende sahip olduğu görülmektedir. Serilerin otokorelasyona sahip olup olmadığını analiz etmek amacıyla şekil 5.2'de verilen ACF grafikleri incelenmiştir.



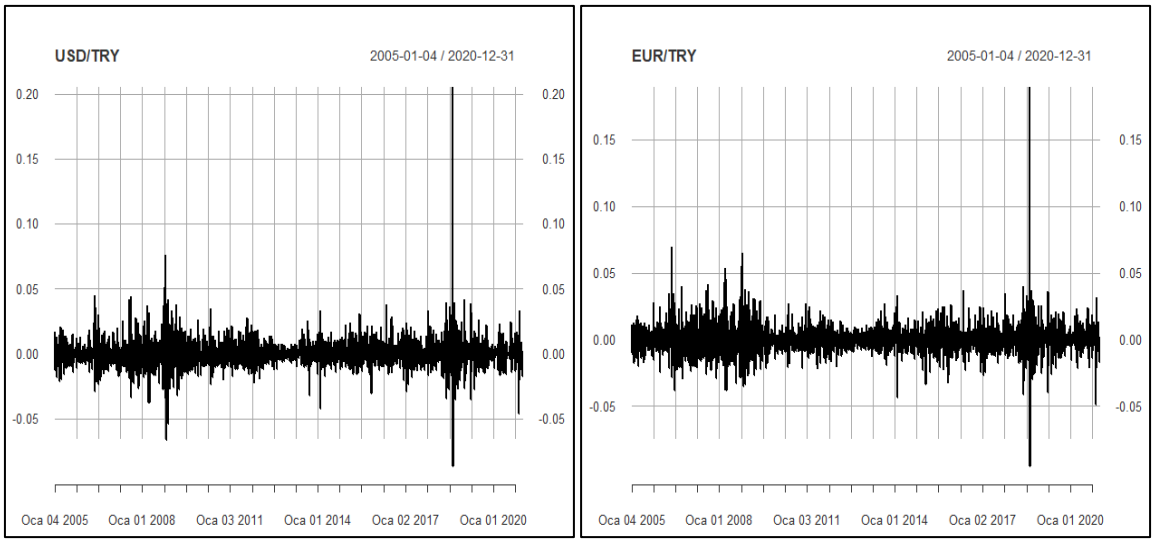
Şekil 5.2. Döviz kurlarına ilişkin ACF ve PACF grafikleri

ACF grafiklerine bakarak serilerin durağan olmadığı görülmektedir. ACF grafikleri hızlı bir şekilde sıfıra gitmediği için döviz kuru serileri durağan olmadığı yorumu yapılabilir. Son olarak grafiklerde gözlemlenen yapı ADF birim kök testi ile araştırılmıştır ve sonuçlar Çizelge 5.4'te gösterilmiştir.

Çizelge 5.4. Düzeyde seriler için ADF birim kök testi sonuçları

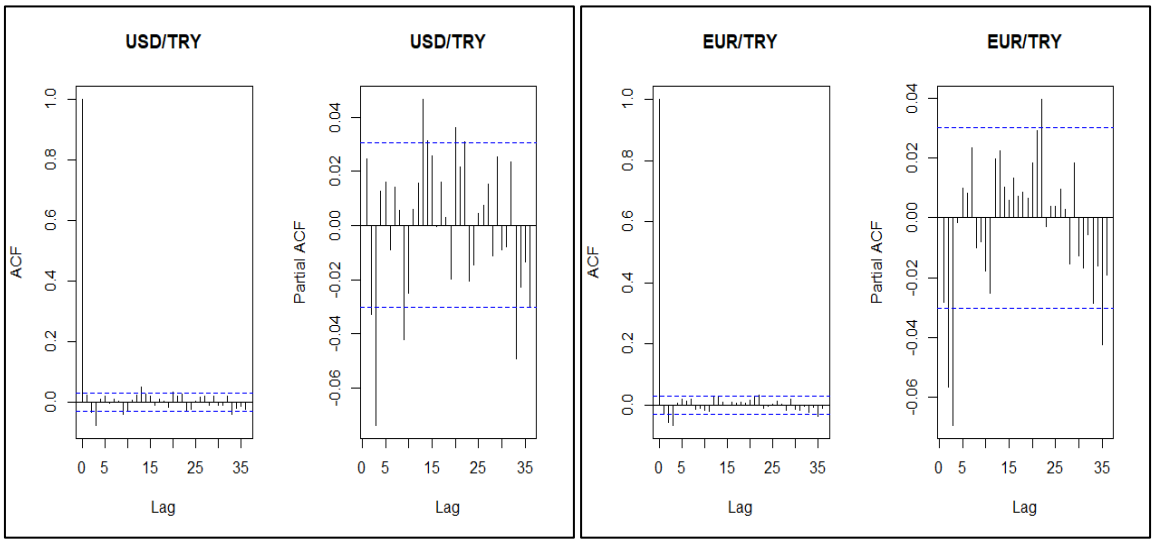
Test Augmented Dickey-Fuller Testi	USD/TRY	
	Test İstatistiği	p-değeri
	-0,9184	0,9511
	EURO/TRY	
	Test İstatistiği	p-değeri
	-0,2161	0,9999

ADF testi sonuçlarına göre serilerin birim kök içerdiğini yani durağan olmadığını iddia eden yokluk hipotezi reddedilememiştir ($p - değeri > \alpha = 0{,}05$). Serilerin durağan olmadığına karar verilmiştir. Bu aşamada serilerin ortalamada ve varyansta durağanlaştırılması amacıyla serilerin logaritması alındıktan sonra farkı alınmıştır. Logaritması ve farkı alınan döviz kurlarına ilişkin zaman serilerine ait grafikleri Şekil 5.3'de gösterilmiştir.



Şekil 5.3. Logaritması ve farkı alınmış zaman serilerine ait grafikleri

Logaritması ve farkı alınan döviz kurlarının zaman serisi grafikleri incelendiğinde her iki serininde durağan olabileceği önsel olarak görülmektedir. Şekil 5.3'de verilen grafikler incelendiğinde sıfır ortalama etrafında salınım yaptığı görülmektedir. Serilerin otokorelasyona sahip olup olmadığını analiz etmek amacıyla ACF grafikleri incelenmiştir.



Şekil 5.4. Logaritması ve farkı alınmış zaman serilerinin ACF ve PACF grafikleri

Şekil 5.4'de verilen logaritması ve farkı alınmış zaman serilerinin ACF grafiklerinin hızlı bir şekilde sıfıra gittiği görülmüştür. Grafiklere bakarak serilerin durağan olduğu görülmektedir. Son olarak grafiklerde gözlemlenen yapı ADF birim kök testi ile

araştırılmıştır ve sonuçlar Çizelge 5.5'te gösterilmiştir. Test sonucuna göre serilerin durağan olduğu görülmüştür ($p - değeri < \alpha = 0{,}05$).

Çizelge 5.5. Logaritmik farkı alınmış seriler için ADF birim kök testi sonuçları

Test Augmented Dickey-Fuller Testi	USD/TRY	
	Test İstatistiği	p-değeri
	-14,481	0,01
	EURO/TRY	
	Test İstatistiği	p-değeri
	-15,345	0,01

Döviz kuru serilerine ait temel özellikler belirlendikten sonra ARCH/GARCH modellerini uygulamak için serilerde ARCH etkisinin varlığının araştırılması gerekmektedir. Model kurma aşaması için öncelikle ortalama denkleminin belirlenmesi gerekir. ACF ve PACF grafiklerine göre USD serisi için ARIMA(0, 1, 1), EURO serisi için ARIMA(0, 1, 1) modeli en uygun model olarak seçilmiştir.

Modellemenin son aşamasında tanı analizi yapılmıştır. Tanı analizi modelin kestirim artıklarına dayalı olarak yapılır. USD ve EURO serilerinin model artıkları incelendiğinde artıkların normal dağılıma sahip olmadığı ve otokorelasyonlu olduğu sonucu elde edilmiştir. Sonrasında serilere ait hata terimlerinde volatilitenin varlığı ARCH-LM testi ile test edilmiştir sonuçlar Çizelge 5.6'da gösterilmiştir.

Çizelge 5.6. ARCH testi sonuçları

	Test İstatistiği	Serbestlik Derecesi	p-değeri
USD/TRY	13,179	1	0,0002
EUR/TRY	11,976	1	0,0005

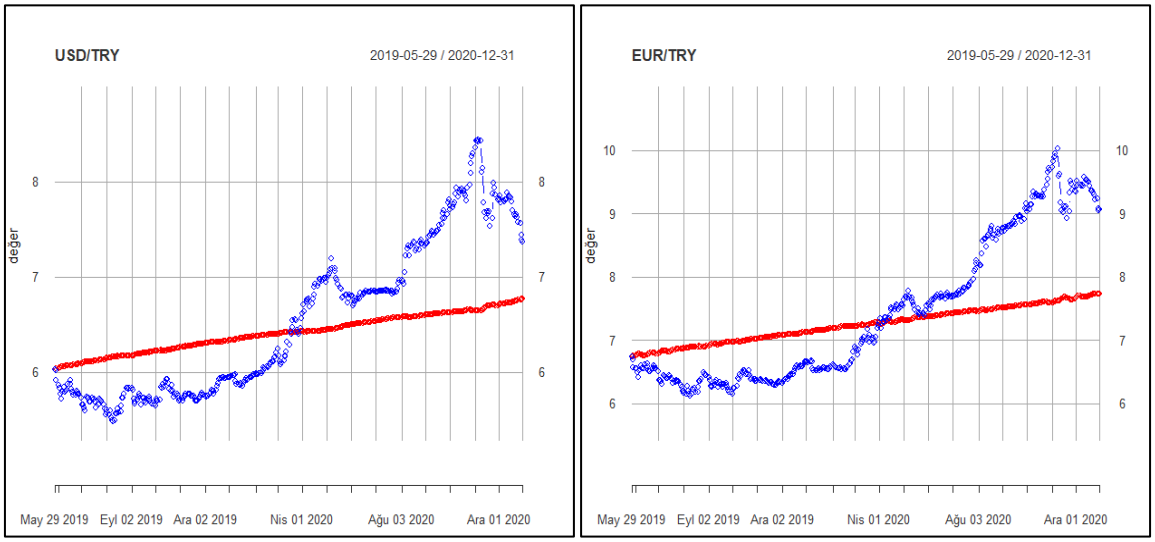
Serilerde ARCH etkisinin olmadığını iddia eden yokluk hipotezi reddedilmiştir ($p - değeri < \alpha = 0{,}05$). Sonuç olarak döviz kuru serilerinde değişen varyans söz konusudur. Bu doğrultuda USD/TRY ve EUR/TRY döviz kuru serileri için simetrik ve asimetrik modeller ile oynaklık tahmini yapılmıştır. En uygun modelin seçilmesi için çeşitli

derecelerde ARCH, GARCH ve EGARCH modeli denenmiştir. AIC değerlerine bakılarak USD/TRY döviz kuru serisi için en uygun modelin EGARCH(0, 1), EUR/TRY serisi için ise EGARCH(0, 1) modeli olduğuna karar verilmiştir ve kestirim (forecast) değerleri elde edilmiştir. Asimetri etkisini gösteren γ parametresi istatistiksel olarak anlamlı olduğundan pozitif ve negatif şokların oynaklık üzerine etkisinin asimetrik olduğu söylenebilmektedir ve tahmin hatalarını en aza indirebilmek üzere asimetrik bilgiyi de modele dahil ederek kestirim değerleri elde edilmiştir.

Kurulan modellerin performansını değerlendirmek için MAE, MAPE, MSE ve RMSE ölçütleri kullanılmıştır. Test verileri üzerinden elde edilen sonuçlar Çizelge 5.7'de verilmiştir.

Çizelge 5.7. Model performans değerlendirme ölçütleri

	MSE	RMSE	MAE	MAPE
USD/TRY	0,4284	0,6545	0,5583	8,2520
EUR/TRY	0,8228	0,9071	0,7317	9,3662



Şekil 5.5. GARCH modeli ile kestirim değerleri

Analizler sonucunda elde edilen kestirim değerleri Şekil 5.5'te görselleştirilmiştir. Her iki zaman serisi için GARCH modelleri tek düze değerler ürettiği görülmüştür.

5.2. Derin Öğrenme ile Döviz Kurlarının Analizi

Veri Önişleme

Verinin modellenebilmesi için uygun hale getirilmesi gerekmektedir. Literatürde ondalık ölçekleme, min-max normalleştirmesi ve z-skor standartlaştırma gibi farklı teknikler bulunmaktadır. Bu çalışmada sayısal değerler için min-max normalleştirme işlemi tercih edilmiştir. Bu yöntem verideki en küçük ve en büyük değerleri göz önüne alarak veriyi 0,1 aralığında yer alacak şekilde dönüşüm yapar.

$$X^* = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{5.1}$$

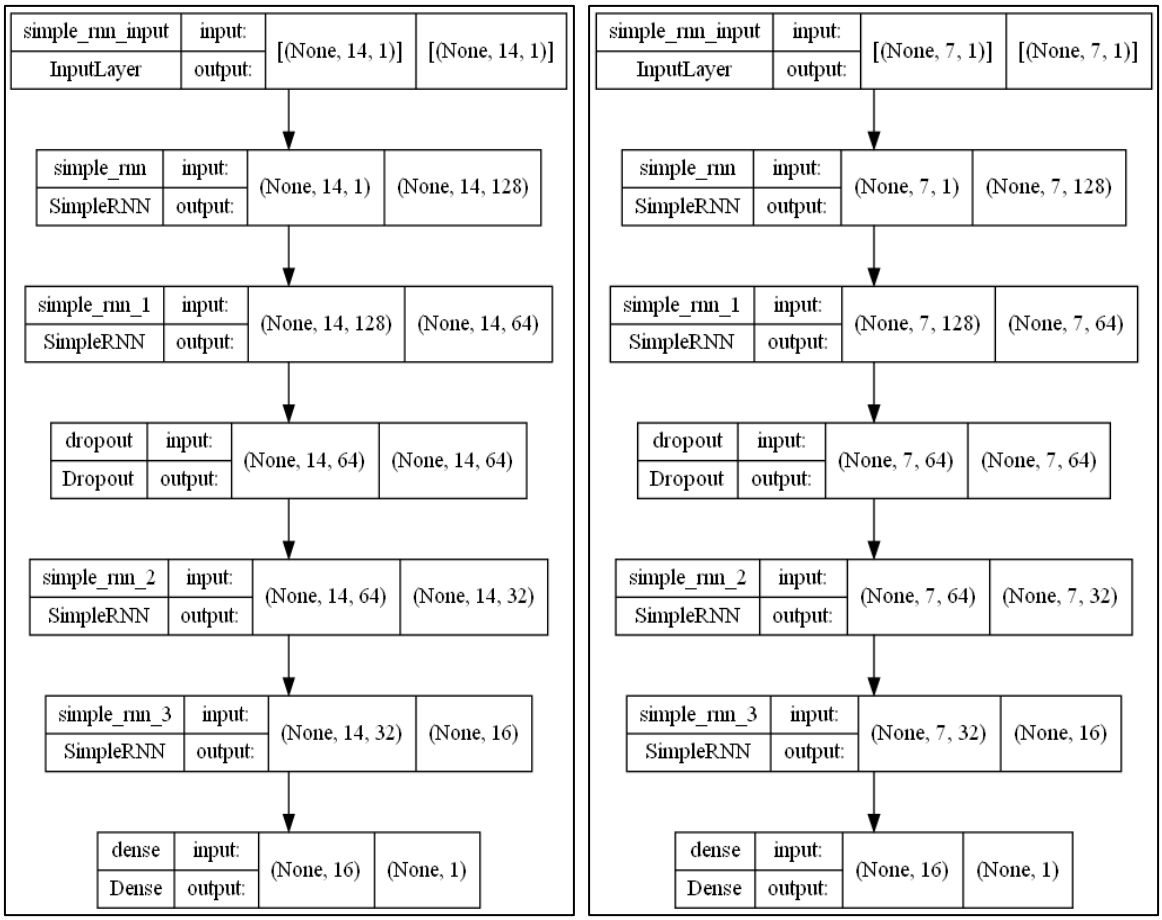
Veriler modelleme için uygun hale getirildikten sonra eğitim verisi ve test verisi olarak veri seti ikiye bölünür. Eğitim verisi üzerinde modelin eğitimi yapılır ve bu model kullanılarak modelin performansı ölçülür (Özkan, 2021: 88-91; Zaccone ve Karim, 2018: 224).

Derin öğrenmede model oluşturulurken katı kurallar bulunmamaktadır. Model oluşturma deneysel olarak parametreler ve katman yapılarının farklı kombinasyonları üzerinde denemeler yapılarak oluşturulur. Bu şekilde RNN ve LSTM yöntemleri ile farklı modeller oluşturularak en etkili parametreye sahip modeller aşağıdaki şekilde elde edilmiştir.

5.2.1. RNN ile döviz kurlarının analizi

Model mimarisi ve ağın eğitilmesi

Model kurulurken eğitim ve test sırasında kullanılan zaman adımı 1, 3, 7, 14, 21 ve 30 olacak şekilde deneysel olarak seçilerek eğitim ve test işlemi gerçekleştirilmiş olup performans karşılaştırılmaları yapılmıştır. USD/TRY veri setinde RNN modeli için en başarılı zaman adımı 14, EUR/TRY veri seti için ise 7 olarak elde edilmiştir. Burada zaman adımı, modelin bir sonraki günün fiyatını tahmin etmesi için kullanacağı günlük verilerin boyutunu belirtmektedir.



Şekil 5.6. RNN model mimarileri

USD ve EURO döviz kuru verileri için oluşturulan RNN model mimarileri Şekil 5.6'da gösterilmektedir. Uygulamada 4 katmanlı bir derin öğrenme modeli kullanılmıştır. İlk katmanda 128 nöron, ikinci katmanda 64 nöron, üçüncü katmanda 32 nöron, dördüncü katmanda 16 nöron ve son olarak 1 nöron içeren yoğunluk katmanı bulunmaktadır. Yığın sayısı (batch size) 32, 64, 128, 256 ve 512 olarak farklı yığın sayıları denenmiştir. USD veri seti için en başarılı sonuç 32, EURO veri seti için 512 olarak elde edilmiştir. Her denemede devir sayısı (epoch) 100 olarak çalıştırılmıştır.

Eğitim sırasında modelin ezber yapmaması için 0,2 oranında seyreltme(dropout) katmanı kullanılmıştır. Katmanlarda aktivasyon fonksiyonu olarak 'ReLU', kayıp fonksiyon olarak MSE ölçütü uygulanmaktadır. Optimizasyon algoritması olarak ise Adam kullanılmıştır ve algoritmanın öğrenme oranı 0,001 olarak ayarlanmıştır. Döviz kurlarına ilişkin RNN modeli için hiperparametreler seçimi Çizelge 5.8'de özetlenmiştir.

Çizelge 5.8. Döviz kurlarına ilişkin RNN modeli parametreleri

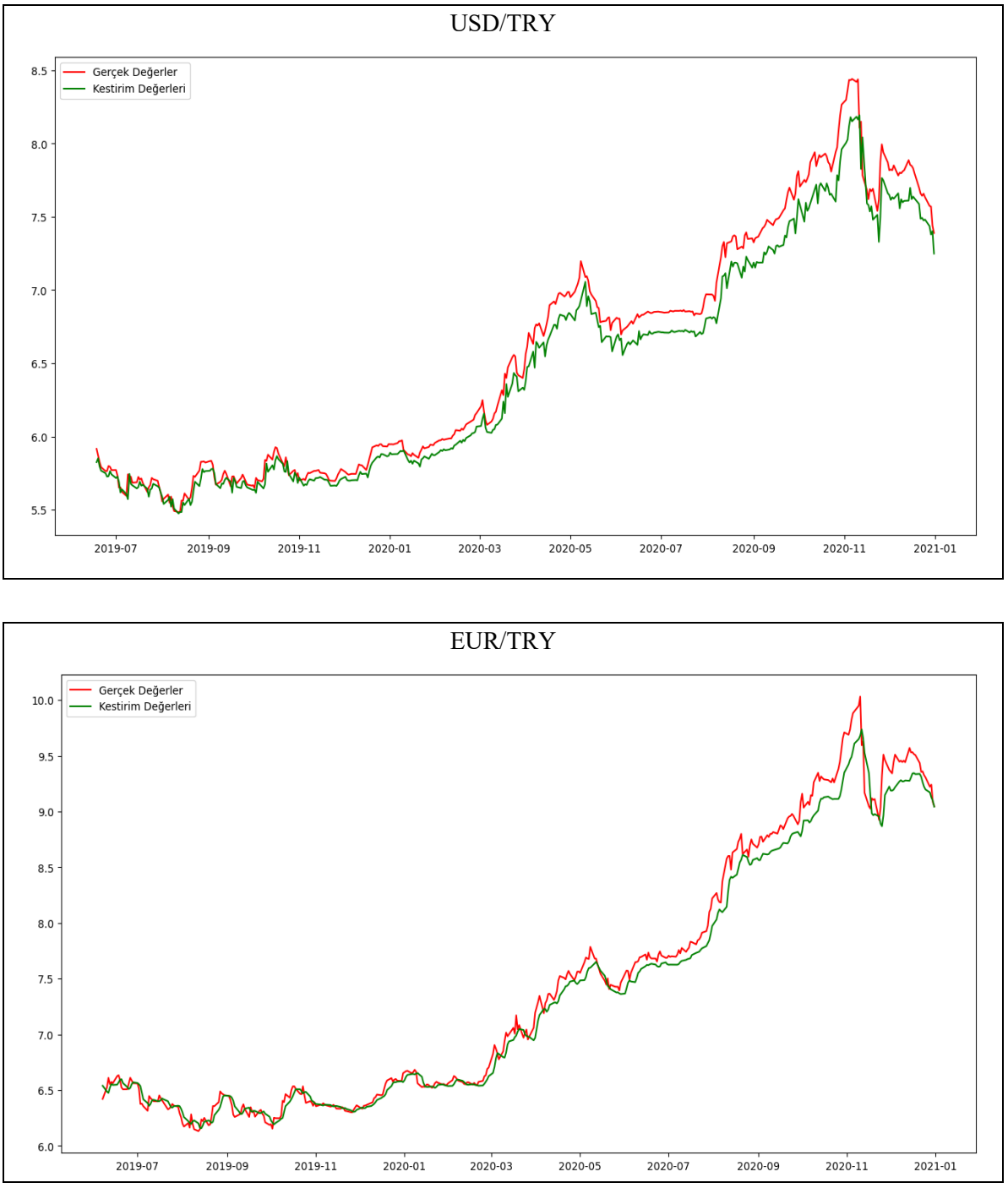
	USD/TRY	EUR/TRY
Hiperparametreler	Değerler	Değerler
Aktivasyon Fonksiyonu	ReLU	ReLU
Optimizasyon Algoritması	Adam	Adam
Öğrenme Oranı	0,001	0,001
Zaman Adımı	14	7
Yığın Sayısı	32	512
Devir Sayısı	100	100

Kurulan derin öğrenme modellerinin performansını değerlendirmek için MAE, MAPE, MSE ve RMSE ölçütleri kullanılmıştır. Test verileri üzerinden elde edilen sonuçlar Çizelge 5.9'da verilmektedir.

Çizelge 5.9. RNN model performans değerlendirme ölçütleri

	MSE	RMSE	MAE	MAPE
USD/TRY	0,0196	0,1400	0,1157	0,0167
EUR/TRY	0,0171	0,1307	0,0955	0,0119

Sonuçlar incelendiğinde RNN modeli performans ölçüm değerlerine göre EUR/TRY veri setinde daha iyi performans sergilediği görülmektedir.



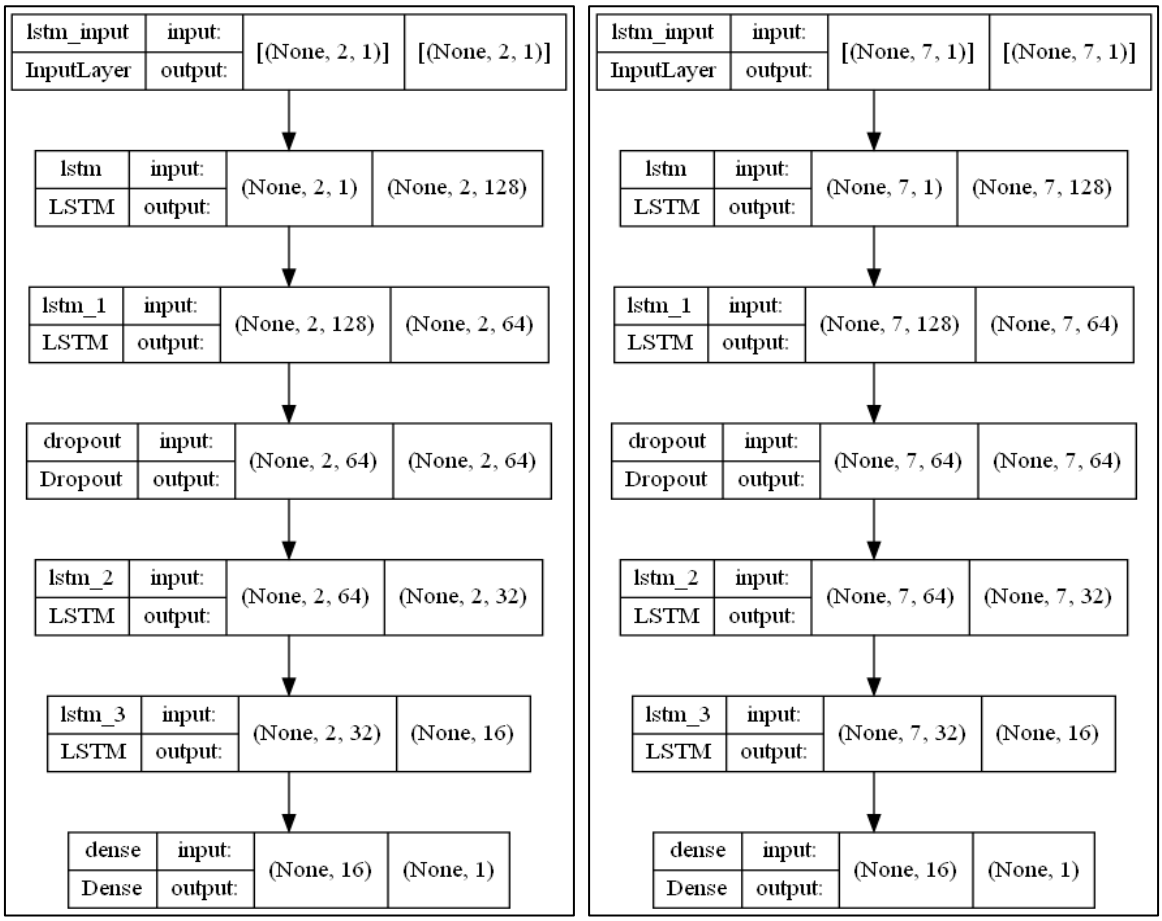
Şekil 5.7. RNN modeli kestirim değerleri

Analizler sonucunda RNN modeli ile elde edilen kestirim değerleri Şekil 5.7'de görselleştirilmiştir. Her iki zaman serisi için RNN modellerinin gerçek değerlere yakın değerler ürettiği görülmüştür.

5.2.2. LSTM ile döviz kurlarının analizi

Model mimarisi ve ağın eğitilmesi

Model kurulurken eğitim ve test sırasında kullanılan zaman adımı 1, 3, 7, 14, 21 ve 30 olacak şekilde deneysel olarak seçilerek eğitim ve test işlemi gerçekleştirilmiş olup performans karşılaştırılmaları yapılmıştır. LSTM modelinde USD/TRY veri seti için en başarılı zaman adımı 2, EUR/TRY veri seti için ise 7 olarak elde edilmiştir. Burada zaman adımı, modelin bir sonraki günün fiyatını tahmin etmesi için kullanacağı günlük verilerin boyutunu belirtmektedir.



Şekil 5.8. LSTM model mimarileri

USD ve EURO döviz kuru verileri için oluşturulan LSTM model mimarileri Şekil 5.8'de gösterilmektedir. Uygulamada 4 katmanlı bir derin öğrenme modeli kullanılmıştır. İlk katmanda 128 nöron, ikinci katmanda 64 nöron, üçüncü katmanda 32 nöron, dördüncü katmanda 16 nöron ve son olarak 1 nöron içeren yoğunluk katmanı bulunmaktadır. Yığın

sayısı (batch size) 32, 64, 128, 256 ve 512 olarak farklı yığın sayıları denenmiştir. USD veri seti için en başarılı sonuç 128, EURO veri seti için 512 olarak elde edilmiştir. Her denemede devir sayısı (epoch) 100 olarak çalıştırılmıştır.

Eğitim sırasında modelin ezber yapmaması için 0,2 oranında seyreltme(dropout) katmanı kullanılmıştır. Katmanlarda aktivasyon fonksiyonu olarak 'ReLU', kayıp fonksiyon olarak MSE ölçütü uygulanmaktadır. Optimizasyon algoritması olarak ise Adam kullanılmıştır ve algoritmanın öğrenme oranı 0,001 olarak ayarlanmıştır. Döviz kurlarına ilişkin LSTM modeli için hiperparametreler seçimi Çizelge 5.10'da özetlenmiştir.

Çizelge 5.10. Döviz kurlarına ilişkin LSTM modeli parametreleri

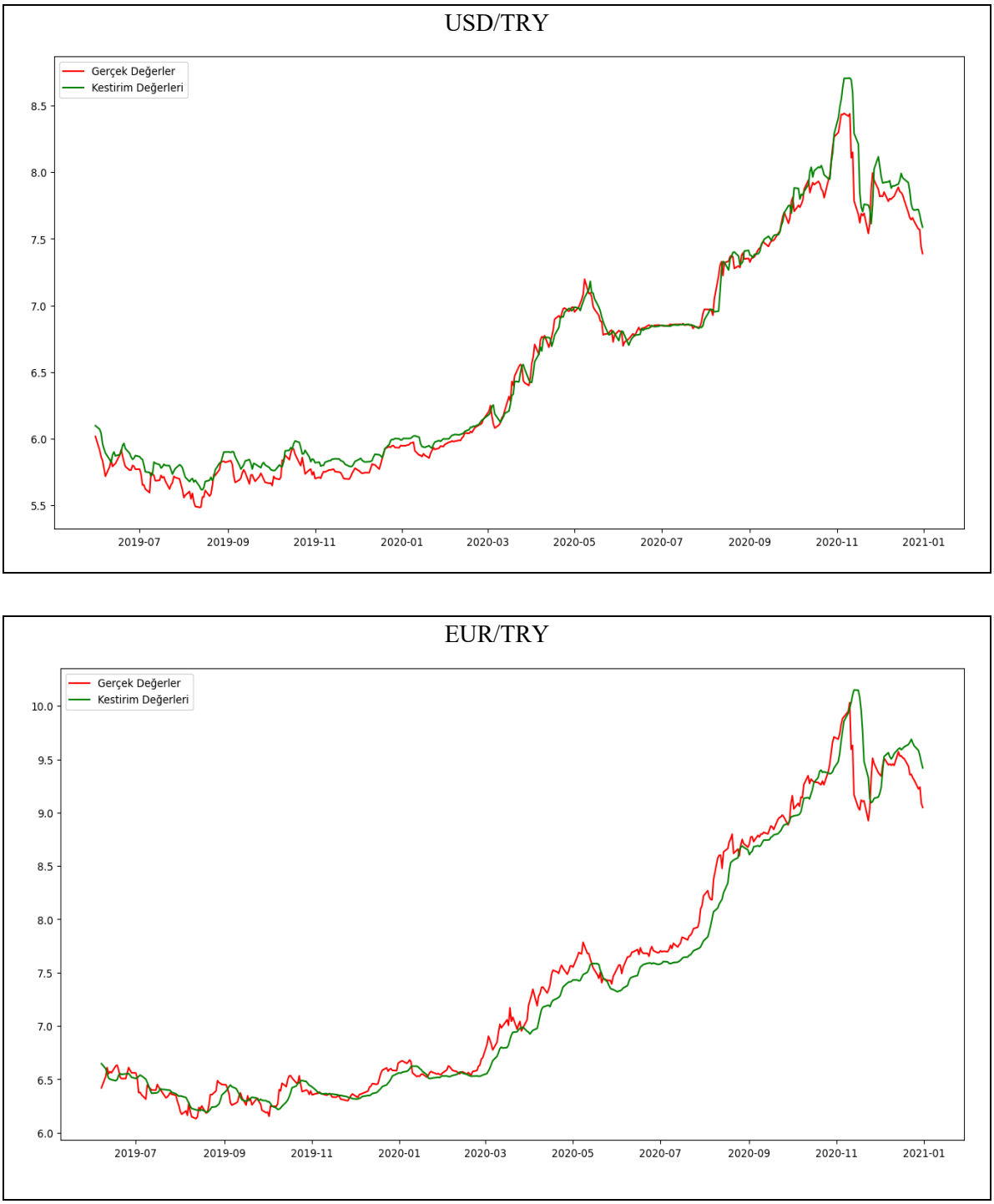
	USD/TRY	EUR/TRY
Hiperparametreler	Değerler	Değerler
Aktivasyon Fonksiyonu	ReLU	ReLU
Optimizasyon Algoritması	Adam	Adam
Öğrenme Oranı	0,001	0,001
Zaman Adımı	2	7
Yığın Sayısı	128	512
Devir Sayısı	100	100

Kurulan derin öğrenme modellerinin performansını değerlendirmek için MAE, MAPE, MSE ve RMSE ölçütleri kullanılmıştır. Elde edilen sonuçlar Çizelge 5.11'de verilmektedir.

Çizelge 5.11. LSTM modeli performans değerlendirme ölçütleri

	MSE	RMSE	MAE	MAPE
USD/TRY	0,0110	0,1049	0,0766	0,0117
EUR/TRY	0,0355	0,1883	0,1309	0,0169

Sonuçlar incelendiğinde LSTM modeli performans ölçüm değerlerine göre USD/TRY veri setinde daha iyi performans sergilediği görülmektedir.



Şekil 5.9. LSTM modeli ile kestirim değerleri

Analizler sonucunda LSTM modeli ile elde edilen kestirim değerleri Şekil 5.9'da görselleştirilmiştir. Her iki zaman serisi için LSTM modellerinin gerçek değerlere yakın değerler ürettiği görülmüştür.

Çizelge 5.12. Modellerin performans değerlendirme ölçütleri

USD/TRY		MSE	RMSE	MAE	MAPE
	GARCH	0,4284	0,6545	0,5583	8,2520
	RNN	0,0355	0,1883	0,1309	0,0169
	LSTM	0,0110	0,1049	0,0766	0,0117
EUR/TRY		MSE	RMSE	MAE	MAPE
	GARCH	0,8228	0,9071	0,7317	9,3662
	RNN	0,0171	0,1307	0,0955	0,0119
	LSTM	0,0355	0,1883	0,1309	0,0169

Döviz kuru kapanış değerleri için GARCH zaman serisi modelleri ve derin öğrenme modelleri olan RNN ve LSTM kullanılarak ile tahmin işlemi gerçekleştirilmiştir. Yapılan analizler doğrultusunda elde edilen sonuçlar Çizelge 5.12'de özetlenmiştir. USD/TRY döviz kuru veri setinde RNN modelinin diğer modellere göre daha başarılı performans sergilediği görülmektedir. EUR/TRY döviz kuru veri setinden elde edilen sonuçlara bakıldığında ise LSTM modelinin diğer modellere göre daha başarılı performans sergilediği görülmektedir. Yapılan analizler doğrultusunda model performans ölçüm değerlerine göre derin öğrenme modellerinin klasik zaman serisi modelinden daha iyi performans sergilediği görülmektedir.

6. SONUÇLAR VE ÖNERİLER

Geleceği tahmin etmek her zaman ilgi çekici bir konu olmuştur. Alınacak kararlarda ve yapılacak yatırımlarda geleceğin öngörüsü alınacak kararları şekillendirebilir. Finansal verilerde kendine özgü pek çok özellik içerdiğinden tahmin işlemi kolay değildir ve başarılı tahminler elde etmek amacıyla farklı yöntemler geliştirilmiştir.

Günümüzde teknolojinin gelişimiyle birlikte klasik yöntemlere alternatif olarak farklı derin öğrenme yöntemleri ortaya çıkmıştır. Çalışmada klasik zaman serisi yöntem ve derin öğrenme yöntemleri uygulanmış ve karşılaştırılmaları yapılmıştır.

Çalışmada 3 Ocak 2005 ile 31 Aralık 2020 dönemi arasındaki USD/TRY ve EUR/TRY döviz kurlarının günlük kapanış değerleri kullanılmıştır. Kullanılan veri setleri %90 eğitim ve %10 test verisi olmak üzere bölünerek uygulama yapılmıştır. Döviz kurlarının kapanış fiyatlarını tahmin etmek amacıyla verilere uygun GARCH, RNN ve LSTM modelleri elde edilmiştir. Elde edilen bu modeller için kestirim değerleri hesaplanmış ve MAE, MAPE, MSE ve RMSE performans değerlendirme ölçütleri kullanılarak karşılaştırılmıştır.

Yapılan analizler sonucunda çalışmada kullanılan USD veri seti için kurulan LSTM modelinin diğer modellere göre daha başarılı olduğu söylenebilir. EURO veri seti için ise RNN modelinin diğer modellere göre daha başarılı olduğu görülmektedir. Elde edilen sonuçlar doğrultusunda döviz kuru kapanış değerlerinin tahmininde derin öğrenme modellerinin GARCH modeline göre daha iyi performans sergilediği gözlemlenmiştir.

Klasik zaman serisi modellerin temeli birçok önemli varsayıma dayanmaktadır. Bu varsayımlar model seçim aşamasında farklı kısıtlamalar getirmektedir. Derin öğrenme modellerinde ise model kurma aşamasında katı kurallar söz konusu değildir. Dolayısıyla seçilen hiperparametrelere göre modellerin performansı değişkenlik gösterir.

Çalışmada ulaşılan bulgular klasik zaman serisi yöntemlerinin varsayımlara ve katı kısıtlara tabi olmasından dolayı derin öğrenme yöntemlerine göre başarılı sonuç sergileyemediğini göstermiştir ve kullanılan yöntemlerin avantaj ve dezavantajları yapılan analizlerle de desteklenmektedir.

Derin öğrenme modelleri nöron sayısı, katman sayısı, yığın sayısı, devir sayısı, optimizasyon algoritması, aktivasyon fonksiyonu gibi birçok hiperparametreye sahiptir. Modellerin kurulum aşamasında hiperparametre seçimi, sonuca dair büyük bir etkiye sahip olduğundan oldukça önemlidir. Çalışmada, derin öğrenme modellerinin zaman adımı ve yığın sayısına göre farklı kombinasyon denemeleri yapılmıştır ve yapılan simülasyon çalışmalarına göre sonuçların bu kombinasyonlara göre değiştiği gözlemlenmiştir. Ancak derin öğrenme modellerinde en başarılı sonucu veren hiperparametre seçiminin yapılabilmesi için çok sayıda kombinasyon olduğundan yalnızca en başarılı modellerin sonuçları verilmiştir.

Finansal serilere ilişki yapılan bu çalışma, derin öğrenme yöntemlerinde tercih edilen hiperparametrelere göre farklı sonuçlar elde etmesi bu alanda çalışma yapan araştırmacılara hiperparametre tercihlerinin sonuçlar üzerinde etkisi olacağına dikkat çekmiştir. Gelecekteki çalışmalar için hiperparametre tercihlerine önem verilmesi gerektiğinin vurgulanmasının yararlı olabileceği düşünülmektedir.

Sonraki çalışmalarda derin öğrenme modelleri için hiperparametre optimizasyonu çalışmaları yapılarak en iyi sonucu veren parametrelere kısa zamanda ulaşılması amaçlanmaktadır. Bu çalışma, farklı döviz kurlarına ilişkin oynaklık modellemelerinin yapılması ve farklı modellerin kullanılmasıyla sonraki çalışmalarca geliştirilebilir. Ayrıca literatürde sıklıkla kullanılan optimizasyon algoritmalarının yanı sıra farklı optimizasyon algoritmaları kullanılarak daha iyi sonuçları elde edip edilemeyeceği ayrı bir çalışma konusu olarak ele alınması planlanmaktadır.

KAYNAKLAR

Alpay, Ö. (2020). LSTM Mimarisi Kullanarak USD/TRY Fiyat Tahmini. *European Journal of Science and Technology*, 452-456.

Baş, E. (2021). *Zaman Serileri ve Öngörü Yöntemleri—R Uygulamalı*. Ankara: Nobel Akademik Yayıncılık.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307-327.

Box, G. E. P., Jenkins, Gwilym M., Reinsel, Gregory C., and Ljung, Greta M. (2016). *Time Series Analysis* (Fifth). Canada.: John Wiley and Sons, Inc., Hoboken, New Jersey.

Brockwell, P. J., and Davis, R. A. (2016). *Introduction to Time Series and Forecasting*. Cham: Springer International Publishing.

Buduma, N. (2015). *Fundamentals of Deep Learning*. ABD: O'Reilly Media.

Campesato, O. (2020). *Artificial intelligence, machine learning, and deep learning*. Dulles, Virginia Boston, Massachusetts New Delhi, India: Mercury Learning and Information.

Cao, J., Li, Z., and Li, J. (2019). Financial time series forecasting model based on CEEMDAN and LSTM. *Physica A: Statistical Mechanics and Its Applications*, *519*, 127-139.

Cheng, L.-C., Huang, Y.-H., Hsieh, M.-H., and Wu, M.-E. (2021). A Novel Trading Strategy Framework Based on Reinforcement Deep Learning for Financial Market Predictions. *Mathematics*, *9*(23), 3094.

Chollet, F. (2020). *Python ile Derin Öğrenme*. Ankara: Buzdağı Yayınevi.

Cryer, J. D., and Chan, K. (2008). *Time series analysis: With applications in R*. New York: Springer.

Dobrovolny, M., Soukal, I., Lim, K. C., Selamat, A., and Krejcar, O. (2020). *Forecasting of FOREX Price Trend using Recurrent Neural Network—Long short-term memory*, Hradec Economic Days, Çekya, 95-103.

Doğan, C. (2021). *İstatistiksel ve Makine Öğrenme İle Derin Sinir Ağlarında Hiper-Parametre Seçimi İçin Melez Yaklaşım*. Yüksek Lisans Tezi, Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

El-Amir, H., and Hamdy, M. (2020). *Deep Learning Pipeline: Building a Deep Learning Model with TensorFlow*. Berkeley, CA: Apress.

Engle, R. (2001). GARCH 101: The Use of ARCH/GARCH Models in Applied Econometrics. *Journal of Economic Perspectives*, *15*(4), 157-168.

Engle, R. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, *50*(4), 987.

Escudero, P., Alcocer, W., and Paredes, J. (2021). Recurrent Neural Networks and ARIMA Models for Euro/Dollar Exchange Rate Forecasting. *Applied Sciences*, *11*(12), 5658.

Fernández, C., Soria, E., Martín, J. D., and Serrano, A. J. (2006). Neural networks for animal science applications: Two case studies. *Expert Systems with Applications*, *31*(2), 444-450.

Gao, Z. (2021). Stock Price Prediction With ARIMA and Deep Learning Models. *2021 IEEE 6th International Conference on Big Data Analytics (ICBDA)*, 61-68.

Goodfellow, I., Bengio, Y., and Courville, A. (2018). *Deep Learning*. Ankara: Buzdağı Yayınevi.

Haykin, S. (2001). *Neural Networks—A Comprehensive Foundations*. Canada: Prentice Hall International.

Hayri, A., Yakut, E., Tekeli, E., Altındağ, İ., Rençber, Ö. F., Akay, Ö. ve Mete, S. (2021). *Veri Madenciliğinde Kullanılan Regresyon Modelleri ve R ile Uygulamalı Örnekler*. Ankara, Nobel Akademik Yayıncılık.

Hyndman, R. J., and Athanasopoulos, G. (2018). *Forecasting: Principles And Practice*. Melbourne, Australia : OTEXTS.

İnternet: Akköse, O. (2021). *Uzun-Kısa Vadeli Bellek(LSTM)*. Web: https://medium.com/deep-learning-turkiye/uzun-k%C4%B1sa-vadeli-bellek-lstm-b018c07174a3, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Baheti, P. (2021). *Activation Functions in Neural Networks [12 Types and Use Cases]*. Web: https://www.v7labs.com/blog/neural-networks-activation-functions, Son Erişim Tarihi: 16 Mayıs 2023

İnternet: Bento, C. (2021). *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis*. Web: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141, Son Erişim Tarihi: 18 Mayıs 2023

İnternet: Biswal, A. (2021). *Recurrent Neural Network (RNN) Tutorial: Types and Examples [Updated] | Simplilearn*. Web: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn#how_does_recurrent_neural_networks_work, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Brownlee, J. (2019). *A Gentle Introduction to the Rectified Linear Unit (ReLU)*. Web: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks, Son Erişim Tarihi: 16 Mayıs 2023

İnternet: Brownlee, J. (2021). *Gradient Descent With Momentum from Scratch*. Web: https://machinelearningmastery.com/gradient-descent-with-momentum-from-scratch, Son Erişim Tarihi: 18 Mayıs 2023

İnternet: Chandra, A. L. (2021). *Learning Parameters Part 5: AdaGrad, RMSProp, and Adam*. Web: https://towardsdatascience.com/learning-parameters-part-5-65a2f3583f7d, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Chugh, A. (2022). *MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared—Which Metric is Better?* Web: https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-of-determination-adjusted-r-squared-which-metric-is-better-cd0326a5697e, Son Erişim Tarihi: 23 Mayıs 2023

İnternet: Çarkacı, N. (2020). *Derin Öğrenme Uygulamalarında En Sık kullanılan Hiper-parametreler*. Web: https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Evrim Ağacı (2022). *Sinir hücreleri neden iyileşmez? Gelecekte bu mümkün olabilir mi? | Soru and Cevap*. Web: https://evrimagaci.org/soru/sinir-hucreleri-neden-iyilesmez-gelecekte-bu-mumkun-olabilir-mi-27286, Son Erişim Tarihi: 16 Mayıs 2023

İnternet: Gupta, A. (2021). *A Comprehensive Guide on Optimizers in Deep Learning*. Web: https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Gylberth, R. (2018). *An Introduction to AdaGrad*. Web: https://medium.com/konvergen/an-introduction-to-adagrad-f130ae871827, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Fatima, E. (2023). *What is the Swish activation function?*. Web: https://www.educative.io/answers/what-is-the-swish-activation-function, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: IBM (2023). *What are Recurrent Neural Networks?*. Web: https://www.ibm.com/topics/recurrent-neural-networks, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: IBM (2023). *What is Gradient Descent?*. Web: https://www.ibm.com/topics/gradient-descent, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Khandelwal, R. (2019). *Overview of different Optimizers for neural networks*. Web: https://medium.datadriveninvestor.com/overview-of-different-optimizers-for-neural-networks-e0ed119440c3, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Kwiatkowski, R. (2022). *Gradient Descent Algorithm—A deep dive*. Web: https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Machine Learning Journey (2021). *RMSProp optimizer explained*. Web: https://machinelearningjourney.com/index.php/2021/01/06/rmsprop, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Ruder, S. (2016). *An Overview of Gradient Descent Optimization Algorithms*. Web: https://www.ruder.io/optimizing-gradient-descent, Son Erişim Tarihi: 17 Mayıs 2023

İnternet: Yahoo Finance (2023), Web: https://finance.yahoo.com/quote/TRY%3DX/history?period1=1104710400&period2=1609372800&interval=1d&filter=history&frequency=1d&includeAdjustedClose=true, Son Erişim Tarihi: 26 Mayıs 2023

İnternet: Yeni Biyoloji (2017), *Sinir Hücresinin (Nöronun) Yapısı, Görevleri ve Nöron Çeşitleri*. Web: https://www.yenibiyoloji.com/sinir-hucresinin-noronun-yapisi-gorevleri-ve-noron-cesitleri-1556, Son Erişim Tarihi: 16 Mayıs 2023

Kadılar, C., and Öncel Çekim, H. (2020). *SPSS ve R Uygulamalı Zaman Serileri Analizine Giriş*. Ankara: Seçkin Yayıncılık.

Karadağ, K. (2022). *Hibrit Derin Öğrenme Modelleri İle Hisse Senedi Fiyat Tahmini*. Yüksek Lisans Tezi, Trakya Üniversitesi, Sosyal Bilimler Enstitüsü, Edirne.

Kaya, E. (2019). *Zaman Serileri Analizinde Box-Jenkins Yöntemi İle Savunma Saniyi Verileri Üzerine Bir Uygulama*. Yüksek Lisans Tezi, Karamanoğlu Mehmetbey Üniversitesi, Sosyal Bilimler Enstitüsü, Karaman.

Kaya, U., Akba, F., Medeni, İ., and Medeni, T. (2020). Covid-19 Öncesi ve Sonrasındaki Bitcoin Fiyat Değişimlerinin Makine Öğrenmesi, Zaman Serileri Analizi ve Derin Öğrenme Yöntemleriyle Değerlendirilmesi. *Bilişim Teknolojileri Dergisi*, *13*(3), 341-355.

Kaynar, O., and Taştan, S. (2009). Zaman Serisi Analizinde Mlp Yapay Sinir Ağları Ve Arıma Modelinin Karşılaştırılması. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, (33), 161-172.

Kızılsu, S. S., Aksoy, S., and Kasap, R. (2001). Bazı makroekonomik zaman dizilerinde değişen varyanslılığın incelenmesi. *Gazi Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi*, *3*(1), 1-18.

Kim, H. Y., and Won, C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications*, *103*, 25-37.

Liu, R., Jiang, Y., and Lin, J. (2022). Forecasting the Volatility of Specifc Risk for Stocks with LSTM. *Procedia Computer Science*, *202*, 111-114.

Lu, W., Li, J., Li, Y., Sun, A., and Wang, J. (2020). A CNN-LSTM-Based Model to Forecast Stock Prices. *Complexity*, *2020*, 1-10.

Nabiyev, V. (2021). *Yapay Zeka Derin Öğrenme – Stratejili Oyunlar Örüntü Tanıma – Doğal Dil İşleme*. Ankara, Seçkin Yayıncılık.

Nelson, D. B. (1991). Conditional Heteroskedasticity in Asset Returns: A New Approach. *Econometrica*, *59*(2), 347.

Nur Laily, V. O., Warsito, B., and I Maruddani, D. A. (2018). Comparison of ARCH / GARCH model and Elman Recurrent Neural Network on data return of closing price stock. *Journal of Physics: Conference Series*, *1025*, 012103.

Özkan, Y. (2021). *Uygulamalı Derin Öğrenme*. İstanbul: Papatya Yayıncılık Eğitim.

Öztürk, K., and Şahin, M. E. (2018). *Yapay Sinir Ağları ve Yapay Zekâ'ya Genel Bir Bakış*. *6*(2), 25-36.

Songül, H. (2010). *Otoregresif Koşullu Değişen Varyans Modelleri: Döviz Kurları Üzerine Uygulama*. Uzmanlık Tezi, Türkiye Cumhuriyet Merkez Bankası Araştırma ve Para Politikası Genel Müdürlüğü, Ankara.

Taş, A. İ., Gülüm, P., and Tulum, G. (2021). Finansal Piyasalarda Hisse Fiyatlarının Derin Öğrenme ve Yapay Sinir Ağı Yöntemleri ile Tahmin Edilmesi; SandP 500 Endeksi Örneği. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, *9*(3), 446-460.

Topaloğlu, E. E. (2019). Döviz Kuru Getiri Oynaklığı Modellemesi: Dolar, Euro ve Sterlin Serileri Üzerine Garch, Egarch ve Igarch Modelleri İle Bir İnceleme. *MANAS Sosyal Araştırmalar Dergisi*, *8*(4), 3352-3378.

Tsay, R. S. (2005). *Analysis of Financial Time Series*. Canada: A John Wiley and Sons, Inc., Publication.

Verly Lopes, D. J., Bobadilha, G. D. S., and Peres Vieira Bedette, A. (2021). Analysis of Lumber Prices Time Series Using Long Short-Term Memory Artificial Neural Networks. *Forests*, *12*(4), 428.

Wang, J., Wang, X., Li, J., and Wang, H. (2021). A Prediction Model of CNN-TLSTM for USD/CNY Exchange Rate Prediction. *IEEE Access*, *9*, 73346-73354.

Yıldırım, E., and Cengiz, M. A. (2022). Modelling and Forecasting of Usd/Try Exchange Rate Using ARMA-GARCH Approach. *İstatistik Araştırma Dergisi*, 12 (2), 1-13

Zaccone, G., and Karim, M. R. (2018). *Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python*. Birmingham: Packt Publishing.

Zeroual, A., Harrou, F., Dairi, A., and Sun, Y. (2020). Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study. *Chaos, Solitons and Fractals*, *140*, 110121.

Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. (2021). *Dive into Deep Learning*. United States, Corwin.

Gazili olmak ayrıcalıktır