T.C.
MERSİN ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME
ANABİLİM DALI

YAPAY SİNİR AĞLARININ KRİZ ANLARINDA BALTIK KURU YÜK ENDEKSİNİ TAHMİNLEME PERFORMANSI

Yüksek Lisans Tezi

Hazırlayan
Mehmet ONGUN

Danışman
Dr. Öğr. Üyesi Ender GÜRGEN

KASIM- 2023, MERSİN

T.C.
MERSİN ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME
ANABİLİM DALI

YAPAY SİNİR AĞLARININ KRİZ ANLARINDA BALTIK KURU YÜK ENDEKSİNİ TAHMİNLEME PERFORMANSI

Yüksek Lisans Tezi

Hazırlayan
Mehmet ONGUN
ORCID ID: 0009-0002-7943-6917

Danışman
Dr. Öğr. Üyesi Ender GÜRGEN
ORCID ID:000-0002-1654-3005

KASIM- 2023, MERSİN

ÖZET

YAPAY SİNİR AĞLARININ KRİZ ANLARINDA BALTIK KURU YÜK ENDEKSİNİ TAHMİNLEME PERFORMANSI

Bu çalışmada kuru yük ve konteyner taşımacılığında navlun fiyatlandırma da önemli bir etken olan Baltık Kuru Yük Endeksi'nin (BDI) bugüne kadar yaşanmış olan Covid-19 pandemisi ve 2008 global krizi zamanlarındaki davranışını üç Yapay Sinir Ağ modeli ile incelenip metrik değerlerine bakılarak hangi modelin kriz zamanlarında öngörü için daha uygun olduğu hakkında bulgulara yer verilecektir. Kıyaslama için seçilen modeller 1D-CNN, CRNN, LSTM modellerdir. Tüm modeller eğitilirken değer kaybına göre erken durdurma sağlanacak şekilde 200 iterasyonla eğitime sokulmuştur. Her üç modelde iki farklı veri seti için iki farklı bölünme için 10 kez eğitilmiştir ve çıkan sonuçlar doğrultusunda tahminleme için en iyi model önerilmiştir.

Anahtar Kelimeler: Yapay Sinir Ağları, Derin Öğrenme, ANN, LSTM, CRNN,1D-CNN, BDI, Tahminleme

Danışman: Dr. Öğr. Üyesi, Ender Gürgen, İşletme Anabilim Dalı, Mersin Üniversitesi, Mersin

ABSTRACT

THE PERFORMANCE OF ARTIFICIAL NEURAL NETWORKS IN PREDICTING THE BALTIC DRY INDEX IN MOMENTS OF CRISIS

Baltic Dry Index (BDI) is an important factor in dry cargo and container transportation pricing. In this study, the behavior of BDI is analyzed in Covid-19 pandemic and 2008 financial crisis using 1D-CNN, CRNN, LSTM neural network models. The analysis consists of assessment for metric values and conclusion where it's decided which model is better for forecasting of the future values. All of the models trained with 200 epochs and early stopping is applied according to value loss. Each models are trained with Covid-19 and 2008 financial crisis datasets and two different train test data splitting methods.

Keywords: Artifical Neural Networks, Deep Learning, Baltic Dry Index, ANN, LSTM, CRNN,1D-CNN, BDI, Forecasting

Advisor: Dr. Öğr. Üyesi, Ender Gürgen, İşletme Anabilim Dalı, Mersin Üniversitesi, Mersin

,

TEŞEKKÜR

Tezimin hazırlanmasında her konuda yardımcı olan danışmanım Dr. Öğrenim Üyesi Ender Gürgen'e ve bu süreçte desteklerini esirgemeyen aile üyelerime en içten teşekkürlerimi sunarım.

İÇİNDEKİLER

TABLOLAR DİZİNİ

ŞEKİLLER DİZİNİ

SİMGELER VE KISALTMALAR

Kısaltma/Simge	Tanım
1D-CNN	1 Boyutlu Evrişimli Sinir Ağı
ADAM	Adaptif Moment Tahmini
AIC	Akakike Bilgi Ölçütü (Akaike Information Criatirion)
AR	Otoregresif Model
ARCH	Otoregresif Koşullu Değişken Varyans
ARIMA	Otoregresif Entegre Hareketli Ortalama Model
ARMA	Otoregresif Hareketli Ortalama Model
BDI	Baltık Kuru Yük Endeksi (Baltic Dry Index)
BIC	Bayesian Bilgi Ölçütü (Bayesian Information Criterion)
CNN	Evrişimli Sinir Ağları (Convolutional Neural Network)
DNN	Derin Sinir Ağları (Deep Neural Networks)
GARCH	Otoregresif Koşullu Değişen Varyans Model
LSE TTC	Lloyd's Shipping Economist Tramp Trip Charter Endeksi
LSTM	Uzun Kısa Süreli Bellek (Long Short-Term Memory)
MAD	Ortalama Mutlak Hata
MAXAD	Maksimum Mutlak Sapma
RMSE	Hata Kareler Ortalamasının Karekökü (Root-Mean-Square Deviation)
RNN	Yenilemeli Sinir Ağları (Recurrent Neural Network)
STAR	Yumuşak Geçişli Otoregresif
VARX	Vektör Otoregresif Model
VECM	. Vektör Hata Düzeltme Modeli
YSA	Yapay Sinir Ağları

GİRİŞ

Müşteri ihtiyaçlarının değişmesi ve yeni taleplerin oluşması nedeniyle konteyner taşımacılığı 1960`lardan itibaren uluslararası ekonomi ve ticaretin ihtiyaçlarına derinden etki etmiştir. Son yıllarda konteyner nakliye pazarı olgunlaşarak globalleşmiş ve rekabet son derece artmıştır. (Hsu & Ho,2021) Varil petrol fiyatlarının yükselmesi, pandemi ve politik anlaşmazlıklardan ötürü navlun fiyatlarında ani iniş-çıkışlar gerçekleşmiştir. Bu durum ithalat ve ihracat yapan şirketlerde navlun fiyatlarını tahminleme ihtiyacını ortaya çıkarmıştır. Literatürde sıkça kullanılan doğrusal tahminleme yöntemleri volatilitesi yüksek borsa ve endeks verilerini tahminlemede yetersiz kalmaktadır. Bu araştırmanın temel amacı Baltık Kuru Yük Endeksinin (BDI) kriz anında öngörülmesi için zaman serilerinin tahminlemesinde kullanılmaya başlayan ve çalışma içerisinde yer alan Yapay Sinir Ağı (YSA) modellerinden, hangisinin daha başarılı olacağının karşılaştırılarak bulgularla tespit edilmesini hedeflemiştir.

Geçtiğimiz son otuz yılda, birçok araştırma geleceği tahmin etme ve daha doğru kararlar vermek üzerine yönelmiştir. Bu araştırmalar tahminleme yöntemlerinde birçok gelişime neden olmuştur. Zaman serisi tahmini için yaygın olarak kullanılan yöntemler şu şekilde özetlenebilir. Doğrusal zaman serileriyle oldukça iyi sonuç veren Box-Jenkins yöntemi bunlardan biridir. Bu teknik doğrusal ve durağan veri setleri ya da durağan olmayıp, durağan veri setlerine dönüşümü yapılabilen veri setlerinin tahminlemesinde oldukça etkilidir. Doğrusal yöntemler geliştirilmesi kolay, anlaşılması ve yorumlanması nispeten basittir. Ancak bu yöntemler veriler arasındaki doğrusal olmayan ilişkileri yakalayamadığı için ciddi sınırlamalar mevcuttur. Ne yazık ki iş dünyasındaki gerçek-zamanlı veriler lineer davranış göstermemektedir. Bu doğrusal olmayan verilerin tahminlenebilmesi için farklı tekniklere gereksinim duyulmuştur ve bu ihtiyaçlar doğrultusunda çift doğrusal, eşik otoregresif (TAR) model, yumuşak geçişli otoregresif (STAR) model ve otoregresif koşullu değişken varyans (ARCH) gibi modeller geliştirilmiştir. Geliştirilen modeller belirli problem ve veriler için çok kullanışlı ve doğrusal modellere göre daha iyi sonuçlar verse de farklı problemler için yeterli etkiyi sağlayamamıştır. Birçok metodoloji istatistiksel yöntemlere dayanırken YSA`lar yeni ideal yöntem olarak ortaya çıkmıştır. Model tabanlı doğrusal olmayan metotların aksine YSA'ları parametrik olmayan veri bazlı karar verme yaklaşımıyla, bir probleme ilişkin ön varsayımlara ihtiyaç duymadan doğrusal olmayan veri yapıları arasındaki ilişkileri yakalayabilirler. YSA'lar tahminleme için daha genel ve esnek bir analiz aracıdır. Yalnızca doğrusal olmayan verilerin değil aynı zamanda doğrusal olan verilerin modellenmesi içinde kullanılabilir. YSA modelleri, zaman serileri için en az Box-Jenkins yöntemiyle sağlanan tahminler kadar iyi performans gösterir (Marquez, 1992; Tealab, 2018; Hamzaçebi vd., 2009; Abbas, 2005; Zhan vd., 2001).

Navlun fiyatları lojistik sektöründe sadece rekabet için değil, aynı zamanda doğru nakliye fiyatlandırmaları için de önemli bir faktördür. Ekonomik krizler, varil petrol fiyatı, ve pandemi gibi global etmenler navlun fiyatlarında çok büyük dalgalanmalara yol açabilmektedir.

Global rekabetin gittikçe arttığı bu durumlarda doğru fiyat tahminlemesi ve bütçeleme, firmalar için rekabet gücü oluşturmaktadır. Bu çalışmada Baltık Kuru Yük Endeksinin kriz zamanlarındaki davranışlarının tahminlenmesinde YSA'ların performansı gözlenmiştir. Bunun doğrultusunda nasıl geliştirmeler yapılabileceği tartışılmış ve önceki çalışmalarla kıyaslaması yapılmıştır. İlk olarak BDI tahminlemesi için literatürde yapılmış çalışmalar incelenmiş ve sonuçlarından bahsedilmiştir.

Literatürde bulunan YSA tahminleme çalışmaları incelenerek kullanılacak YSA modelleri seçilip tasarlanmıştır. Literatürlerdeki birçok borsa tahminleme çalışmasında LSTM ve 1D-CNN modelin doğrusal tahminleme yöntemlerine göre daha iyi tahminleme yaptığının kanıtlandığı görülmüş ve incelenmek istenen verilerin karakteristikleri incelendiğinde doğrusal hiçbir ilişki saptanmamıştır. Bu nedenle YSA ile tahminleme yoluna gidilmiştir. Bu adımın sonucunda LSTM ,1D-CNN modelleri ve bu modellerin birleşimi olan CRNN modeli seçilmiştir. Modeller kararlaştırıldıktan sonra Python programlama dili ve kütüphaneleri kullanılarak kurulan YSA ve doğrusal tahminleme modelleri simüle edilmiştir. Her modelde doğru parametrelerin bulunması için defalarca iterasyonlar yapılarak deneme yanılma yöntemiyle en iyi parametre ve kısıtlar seçilmiştir. Çıktılar analiz edilmiş ve gelecek çalışmalar için öneriler yapılmıştır. Çalışmanın özgün katkısı olarak BDI sadece kriz anlarında tahminlenmiş ve tahminlenmesi için LSTM ,1D-CNN ve CRNN modelleri kullanılmıştır. Çalışmanın kısıtları şu şekildedir;

- YSA'larla çalışılırken en iyi model seçimi deneme yanılma yoluyla bulunmaktadır. Seçilen model en iyi model varsayım olarak kabul edilmektedir. Bu çalışmada seçilen modeller parametreleri değiştirilerek iterasyonlar sonucu optimum model olarak seçilmiştir.
- YSA modelleri black-box modellerdir ve verinin karakteristiğine bağlıdır. Veri işlenirken ağırlığını kendisi belirlemektedir. Bu ağırlıklandırmanın doğru olduğu kabul edilmektedir.

Çalışmanın birinci bölümünde kavramsal çerçeve anlatılmıştır. İkinci bölümünde, BDI tahminlemesi için yapılmış geçmiş çalışmalara yer verilmiştir. Üçüncü bölümde kullanılan verilerin bilgisi ve kurulan modellerin açıklaması yer almaktadır. Dördüncü bölümde tahminlemelerin metrik tablolarına bakılarak bulguların ortaya konması ve son olarak beşinci bölümde çalışma bulgularının özeti ve gelecek çalışmalara öneriler yer almaktadır.

.

1.KAVRAMSAL ÇERÇEVE

1.1.Navlun ve BDI

Geçtiğimiz son 50 yılla birlikte ekonominin globalleşme oranı artarak ve evrimleşerek ekonomiyi destekleyen tedarik zincirini de beraberinde getirdi. Bu evrime ayak uydurabilmek için daha etkili tedarik zinciri ağları gelişti. Bu sebeple ürünler ve hizmetler aynı coğrafik bölgelerin ötesine geçerek global ticaret ağlarını oluşturdu (Crainic & Laporte, 1997) . Oluşan global ticaretin %80'ninden fazlasının taşınmasını sağlayan deniz ticareti, global ekonominin ve ticaretin belkemiği olmuştur. (UNCTAD,2021). Birçok endüstri gibi taşıma endüstrisi de global ekonomik krizlerden fazlasıyla etkilenmektedir. Son zamanlarda Covid-19 pandemisi ve Süveyş Kanalı'nın tıkanması gibi olaylar navlun fiyatlarında ciddi dalgalanmalara sebep olmuştur. BDI Navlun oranlarının finansal riskten korunmasına yarayan, vadeli navlun sözleşmeleri veya endeks bağlantılı konteyner sözleşmeleri gibi araçlar konteyner taşıma endüstrisin de hala geniş bir kullanıma ulaşamamıştır. Bu nedenle nakliyeciler ve taşıyıcılar navlun fiyatlarındaki bu tarz dalgalanmaları tahmin edebilir ve doğru zamanda doğru kararları vererek birçok maliyetten kurtulabilirler. Bu tarz piyasalar için genellikle endeks ya da grafik benzeri kılavuzlar kullanılarak tahminleme yapılmaya çalışılır. Baltık Kuru Yük Endeksi global ölçekte deniz yoluyla taşınan kuru yük navlun değerlerini zamana bağlı verilerle oluşturan, Baltık Borsası (Baltic Stock Market) tarafından yayınlanan tüm dünyaca kabul gören ve literatürde tahminleme için en çok kullanılan endekstir. BDI volatilitesi yüksek bir endeks olup global ekonomik olaylardan etkilenmektedir. Bunun en büyük örneği 2008 yılı global ekonomik krizidir. Bu kriz esnasında 20 Mayıs 2008 tarihinde BDI 11,793 dolar değerini görerek tarihinin en yüksek zirvesini görmüştür ve devamındaki 6 ay içerisinde keskin bir şekilde yaklaşık olarak 700 dolar seviyesine kadar gerilemiştir (Munim & Schramm, 2017; Zeren & Kahramaner, 2019). Literatürde navlun fiyatları ve endeksler üzerine birçok tahminleme çalışması yapılmıştır.

Londra Threadneedle caddesinde bulunan Virginia ve Baltic kahve evindeki tüccarlar ve gemi kaptanları arasındaki gündelik konuşmaları sırasında kurulan Baltık Borsası , 1744 yılına dayanan çok uzun bir geçmişe sahiptir(Lin & Sim, 2013) . Baltık borsası, 1985 yılında ağırlıklı olarak tahıl, kömür, demir cevheri, bakır ve diğer birincil ürünler gibi ham mallardan oluşan kuru yük kargolarının, sevkiyat oranlarının genel bir göstergesi olarak BDI'yı faaliyete geçirdi (Lin & Sim, 2013) .Baltık Borsasının yayınladığı başka indekslerde bulunmaktadır. Bunlar:

Baltic Panamax Endeksi (BPI) 1998 yılında; Baltic Capesize Endeksi (BCI) 1999 yılında; the Baltic Handymax Index (BHMI) 2000 yılında ve son olarak Baltic Supramax Endeksi (BSI) 2005 yılında faaliyete geçmiştir.(Geman & Smith, 2012).

BDI`nin yansıra taşıma endüstrisinde kullanılan birçok endeks bulunmaktadır. Bunlardan en yaygın olanlarının isimleri ve yayınlayan kuruluşlar şunlardır:

- Lloyd's Shipping Economist Tramp Trip Charter Endeksi (LSE TTC) (Informa Maritime & Transport, 1978- günümüze)
- Maritime Research Inc. Freight Endeksi (Maritime Research Inc., 1970- günümüze)
- SSY Capesize Endeksi, Atlantik ve Pasifik (SSY, 1985 - günümüze)
- JE Hyde Handy Endeksi, Supermaxand Handymax (Icap Hyde, 1990- günümüze) (Veenstra, A., & van Dalen, J. ,2008)

Lloyd's Shipping Economist Tramp Trip Charter Endeksi, beş farklı gemi boyut ve klasına göre alt endeksler oluşturmuştur. Ana endeks bu endekslerin ağırlıklı ortalamasıyla oluşturulmuştur. Maritime Research Inc. Freight Endeksi emtia türüne göre (tahıl nakliyesi ve diğer ürünler) iki faklı alt endeksi bulunmaktadır. Bu iki alt endeks seçilen ticaret rotasının ana ürünlerine göre oluşturulmuştur. SSY Capesize ve JE Hyde endeksleri seçilen on adet rotanın ortalama kur fiyatlarına göre oluşturulmuştur. Bu ortalamalar rotaların önem sırasına göre belirlenen bir ağırlık sistemine göre birleştirilir. Daha çok kargo ve ana ürüne sahip veya daha çok yük taşıyan ticaret yoluna daha yüksek ağırlık verilir (Veenstra & van Dalen, 2008).

BDI, emtia analistleri tarafından dünya çapında ekonomik büyümenin durumu hakkında erken bir gösterge olarak anılmaktadır. Bunun nedeni emtia talebiyle doğrudan ilgili olması ve ülkelerin birçok stratejik sektörüyle bağlantılı olmasından kaynaklanmaktadır (inşaat, araba imalatı vb.) (Geman & Smith, 2012). Bu nedenle literatürde en çok çalışma yapılan endekslerin başında gelmektedir.

1.2.Tahminleme

Tahminleme, karar vermenin önemli bir parçasıdır ve gelecekteki bilinmeyen olaylar için kararlarımızı bu tahminlere dayandırarak veririz. Gelecek olaylar belirsiz olduğundan dolayı bu tahminlemeler genellikle mükemmel değildir. Öngörmenin temel amacı tahminleme hatasını minimize etmektir. Tahminleme yöntemleri genel olarak nitel ve nicel olmak üzere sınıflandırılır. Matematiksel model ve istatistiğe dayalı varsayımlar nicel tahminleme yöntemleridir. İstatistiksel modeller otoregresif (AR), otoregresif hareketli ortalama, durağan olmayan doğrusal stokastik modeller (ARIMA), Bayesian yaklaşımı ve gri tahminleme modellerini içerir (Abraham & Ledolter, 1983; Chang, 2014; Duru, 2007).

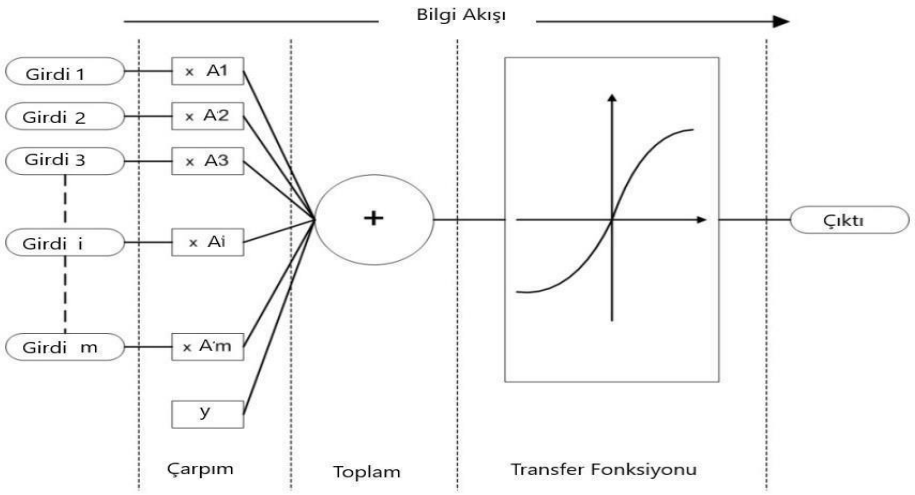
Zaman serilerinin ortak özellikleri, matematiksel fonksiyonlarla deterministik olarak modellenebilen trend ve mevsimsel değişimlerdir. Öte yandan çoğu zaman serisinde birbirine yakın olan değerler arasında korelasyon eğilimi bulunmaktadır. Zaman serisi analizinin temel amacı bu

korelasyonu uygun istatistiksel model ve tanımlayıcı yöntemler aracılığıyla açıklamaktır. Veriye uygun bir model bulunması, verinin gelecekteki değerlerinin tahminlenmesi ve simüle edilmesini mümkün kılmaktadır (Cowpertwait & Metcalfe, 2009). İstatiksel tahminleme metotları için en hassas nokta model veri uyumunun sağlanmasıdır. Gerekli düzeltmelerin yapılarak veriler arasındaki korelasyonun giderilmesi elzemdir. Doğrusal olmayan veri setleri için istatiksel modellerin uygulanması oldukça zordur. Doğrusal olmayan veriler için geliştirilen modeller, veri karakteristiğine ve kısıtlamalara uymadığı takdirde istenilen performansı verememektedir. Bu noktada YSA'lar veri bazlı yaklaşımı doğrusal olmayan ve doğrusal veri setleri için kolaylıklar sağlamıştır.

YSA'ların tahminlemede kullanılması birçok avantajı ve dikkat edilmesi gereken birçok kriteri de yanında getirmiştir. Makine öğrenmesi kabiliyetine sahip olması ve büyük verilerle çalışabilir olması güçlü donanım gereksinimlerine ihtiyaç duyulmasına sebep olmuştur. En büyük avantajlarından birisi tamamlanmamış verilerle de çalışabilmesidir. Eğitim verisinin seçimi kritik bir noktadır. YSA literatürde black-box uygulaması olarak görülmektedir. YSA'lar çözüm sunarken neden ve nasıl olduğuna dair kanıt verememektedir ve bu güvenilirliklerini düşürmektedir. (Mijwel, 2018)

1.3.Yapay Sinir Ağları

İnsan beyni organik materyallerden ve ıslak kimyadan oluşmuş dünyanın en iyi bilgisayarlarından biri olarak düşünülebilir. Ancak insan beyni, tüm verilerin ciddiyetle işlenmesi gereken karmaşık çarpma işlemleri gibi problemleri çözmekte problem yaşamaktadır. Öte yandan insan beyni, varlığın kanıtı olan, insanların becerikli olduğu bilişsel, algısal ve kontrol görevlerini başarıyla yapan büyük nöral ağlar sağlamaktadır. YSA'lar memelilerin sinir sisteminden ilham alınarak geliştirilen matematiksel modellerdir. YSA'lar bilgiyi öğrenme ve sürdürme yeteneğine sahiptir ve verileri işleme birimi olarak tanımlanabilir. Birçok ara bağlantı ile birbirine bağlanan katmanların vektör ağırlıkları ve matrisleriyle uygulaması yapılır. YSA modellerinde üç basit kural bulunmaktadır. Bunlar çarpma, toplama ve aktivasyondur. Şekil 1.1 de yapay nöronun çalışma prensibi şematize edilmiştir.



Şekil 1.1. Yapay Nöron Çalışma Prensibi (Krenker vd., 2011)

-Doğrusal Aktivasyon Fonksiyonu

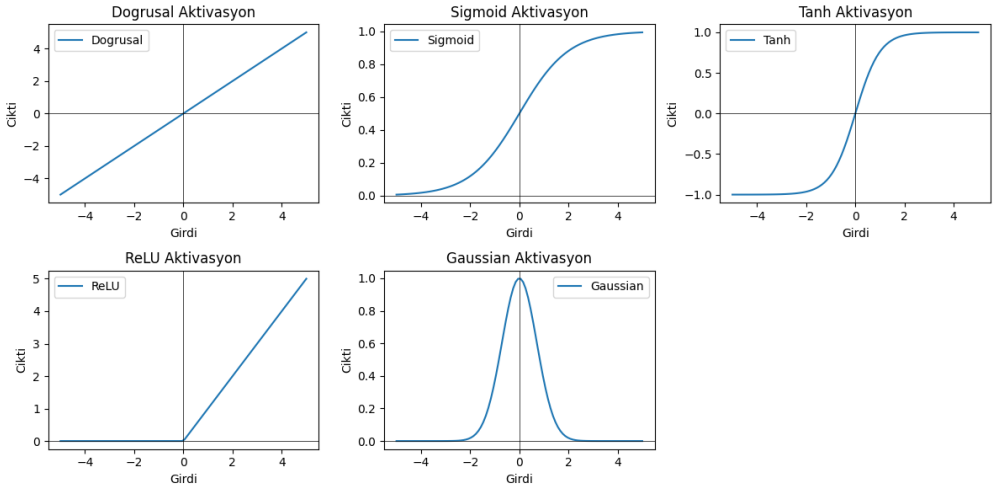
-Sigmoid Aktivasyon Fonksiyonu

-Tahn Aktivasyon Fonksiyonu

- ReLU Aktivasyon Fonksiyonu

-Gaussian Aktivasyon Fonksiyonu

Sekil.1.2'de sık kullanılan aktivasyon fonksiyonlarının grafikleri göstermiştir. Doğrusal aktivasyon fonksiyonu gerçek sayı aralığında sadece pozitif sayı üretmektedir. Sigmoid aktivasyon fonksiyon değerleri 0-1 arasında pozitif sayılara dönüştürür. Tahn aktivasyon fonksiyonu hiperbolik tanjant fonksiyonudur. Sigmoid fonksiyona benzerlerdir ancak farklı olarak orijine göre simetrisi vardır. ReLu doğrusal olmayan ve YSA`da çokça kullanılan bir fonksiyondur. ReLU fonksiyonun en büyük faydası aynı anda belirli sayıda nöron aktifleştirmesidir. Bunun nedeni bazı durumlarda gradyan fonksiyonun doğası gereği sıfıra eşit olmasıdır. Gaussian fonksiyonu aktivasyon aralığında daha sıkı kontrole ihtiyaç duyulduğunda kullanılır (Sibi vd., 2013; Sharma vd., 2017).



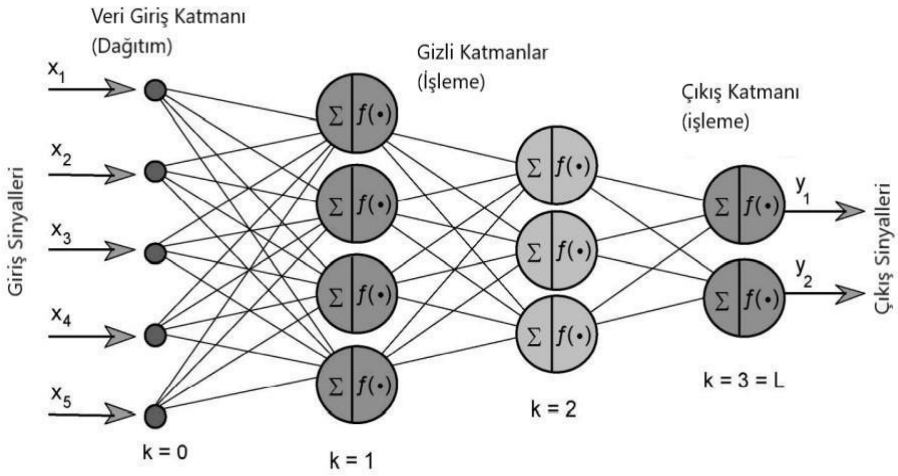
Sekil 1.2. Yaygın Kullanılan Aktivasyon Fonksiyonları

En yaygın kullanılan aktivasyon fonksiyonu sigmoid ve ReLu fonksiyondur. Tablo 1.1'de sık kullanılan aktivasyon fonksiyonlarının formülleri verilmiştir. Çalışma prensibi ve kuralları basit olmasına rağmen yapay nöronlar asıl potansiyelini ve hesaplama gücünü başka nöronlarla bağlantı kurduğu zaman ortaya çıkarmaktadır. Birden fazla yapay nörondan oluşan YSA'lar matematiksel modelin kompleks yapısını fazlasıyla arttırmaktadır (Krenker vd., 2011; Priddy, & Keller, 2005).

Tablo 1.1 Sık kullanılan aktivasyon fonksiyonlarının formülleri

Doğrusal	$f(x) = ax$
Sigmoid	$f(x) = 1 / (1 + exp(-x))$
Tahn	$f(x) = 2sigmoid(2x)-1$
ReLu	$f(x) = max(0,x)$
Gaussian	$f(x) = exp(-(x - \mu)^2 / (2\sigma^2))$

Yapay nöronların birleşmesiyle oluşan YSA yapıları ileri beslemeli ya da geri beslemeli yapay sinir ağlarını oluşturur. İleri beslemeli YSA'lar Şekil 1.3'te gösterilmektedir. Görselin sol tarafında veri giriş katmanının girdileri, devamında iç içe bağlı olan gizli katman nöronları, çıkış katmanı ve çıktılar bulunmaktadır. Ağdaki her katman direk olarak bir sonraki katmanı beslemektedir. Böylece tüm katmanlar ileriye doğru veri akışını sağlayarak YSA mimarisini oluşturur ve ileri beslemeli adını alır.



Şekil 1.3. İleri Beslemeli Yapay Sinir Ağı İşleyişi (Priddy & Keller, 2005)

Nöronların transfer fonksiyonları, ağın ileri besleme davranışını etkilememektedir. Bir modelde nöronlarda istenen transfer fonksiyonu kullanılabilir. Ayrıca önceki katmandaki bir nöron sonraki katmandaki herhangi bir nöronu besleyebilmektedir. Eğer bir nöron kendinden önceki bir katmanı ya da kendi katmanındaki bir nöronu beslerse bu yapıya Geri Beslemeli YSA denmektedir (Hopfield, 1988; Abraham, 2005; Mehrotra vd., 1997).

YSA uygulaması belirli network tasarım parametre bilgileri gerektirmektedir. Bu parametrelerin önceliğinin belirlenmesi için tanımlanmış bir kural yoktur. Herhangi bir problem için belirli bir ağ tasarımı yoktur. Problem için birden fazla model kurgulanıp en uygun olan seçilir. YSA mimarisi seçerken dikkat edilmesi gereken parametreler şunlardır:

-Girdi nöronu sayısı

-Çıktı nöronu sayısı

-Gizli katman sayısı

-Her gizli katmandaki nöron sayısı

-Başlangıç bağlantı ağırlıkları

-Başlangıç nöron yanlılığı

-Öğrenme oranı

-Momentum oranı

Girdi ve çıktı nöronu sayısının problemde verilen girdi çeşidiyle aynı olması beklenir. Gizli katman sayısı, girdi kombinasyonlarını arttır ve yüksek ya da düşük değerli sonuçların oluşmasını sağlar.

Birden fazla gizli katmanın olması kombinasyonun kombinasyonlarının oluşmasını sağlar. Her katmandaki nöronlar önceki katmandaki tüm nöronlara bağlandığı için gizli katmanlardaki nöron sayısı da önemlidir. Başlangıçta verilen bağlantı ağırlıkları, çözüm için bir başlangıç noktası niteliğindedir. YSA'nın eğitimi başlangıç ağırlık değerlerine bağlıdır. Optimum başlangıç ağırlıklarının bulunması için YSA'yı farklı başlangıç ağırlıklarıyla birkaç kez eğitmek önemlidir. Nöron yanlılık değerleri, o nöronu besleyen kombinasyonların önemini belirtir. Yanlılık değerleri eğitim esnasında değiştirilebilir. Öğrenme oranı kullanıcının belirlediği bir orandır. Bağlantı ağırlıkları ve yanlılıklar değiştirilerek daha doğru bir sonuca gidilmeye çalışılabilir (Cowpertwait, & Metcalfe, 2009; Dumitru & Maria, 2013).

1.3.1.Derin Sinir Ağları (DNN) ve Derin Öğrenme

Son zamanlarda makine öğrenmesi, gelişmiş öğrenme algoritmaları, etkili ön işleme teknikleri ve yüksek performanslı bilgi işleme merkezlerinin artmasıyla kayda değer bir gelişme kaydetmiştir. Bu gelişmelerden en önemlisi YSA'ların, daha gelişmiş öğrenme yeteneklerine sahip daha derin mimarilere evrilerek derin öğrenme olarak adlandırılan DNN yapılarına dönüşmesidir (Janiesch vd., 2021). DNN'ler çok sayıda niteliği işleme yeteneğiyle yapılandırılmamış verileri işlerken daha güçlü performans ve esneklik gösterir. Derin öğrenme algoritmaları veriyi birden fazla katmandan geçirir. Her katman verinin özelliklerini ortaya çıkarma yeteneğine sahiptir ve bu bilgiyi diğer katmanlara aktarabilir. Giriş katmanları düşük seviye özellik bilgilerini ortaya çıkarır ve sonraki katmanlara bu bilgiyi işleyip anlamlı çıktılar vermesi için iletir (Mathew vd., 2021). Derin öğrenme modern toplumun yapısına birçok açıdan destek vermektedir. Internet araştırmaları, içerik filtreleme, sosyal ağlar, e-ticaret sitelerindeki öneriler vb. alanlarda kullanılmaktadır. Özellikle günümüzde telefon ve kamera gibi tüketici ürünlerinde daha fazla yer almaya başlamaktadır. Akıllı telefonlarda, obje tanımlama, konuşmayı yazıya çevirme, kullanıcının ilgisine göre ürünleri, haberleri ve arama sonuçlarını gösterme gibi birçok alanda kullanımı giderek artmaktadır. Ayrıca tarım, hastalık saptama ve ekonomi tahminlemeleri gibi bilimin her alanında kullanımı yaygınlaşmaktadır (LeCun vd, 2015).

1.3.2.Evrişimli Sinir Ağları (CNN- Convolutional Neural Network)

CNN'ler biyolojik olarak memelilerin basit hesaplamalı görsel kortekslerinden ilham alınarak oluşturulmuş ileri beslemeli bir YSA'dır. Diğer bir deyişle memeli canlıların görme ve görüntüyü işleme şekli taklit edilerek oluşturulmuştur. Görüntü ve video tanımlama problemlerinin çözülmesinde en sık kullanılan modellerdir (Eren vd., 2019).

CNN'ler kendi kendine öğrenme yoluyla nöronlarını optimize eder ve bu bakımdan geleneksel YSA'lara oldukça benzemektedir. Her bir nöron tüm YSA da olduğu gibi girdi verisini işleyerek bir sonraki katmana iletir (O'Shea & Nash, 2015).

CNN modelleri için popüler aktivasyon fonksiyonları sigmoid, Tahn ve ReLudur. Çalışmalarda genel olarak basit ve hesaplaması kolay olmasından ötürü ReLu fonksiyonu tercih edilmektedir. Veriler

aktivasyon katmanından geçtikten sonra havuzlama katmanına geçirilirler. Burada gelen bilgiler kümelendirilir. Bu işlemin amacı girdideki temsili daha güçlü ve küçük değişimlerden daha ez etkilenir hale getirmektedir (Zhao & Wang, 2017).

1.3.3.Yenilemeli Sinir ağları (RNN) ve Uzun Kısa Süreli Bellek (LSTM-Long short-term memory)

Yenilemeli sinir ağları genellikle el yazısı, genomlar, endüstri ortamında üretilen (stok-market, sensör) yazı veya sayısal zaman serileri gibi veri dizilerindeki kalıpları saptamak için kullanılmaktadır. Daha yüksek seviyelerde RNN'ler dil modelleme, yazı yaratma ve çeviri işlemleri ve konuşma tanımlama gibi uygulamalarda kullanılmaktadır. Verileri döngü olmadan katmanlarından ileri doğru ileten CNN'lerin aksine RNN'ler nöron çıktılarını döngüsel şekilde kendi içerisine iletebilir (Schmidt, 2019).

RNN içerisindeki gizli katmanlar, sıralı verileri erken safhada okurken, yakalanan bilgileri depolamak için dahili hafıza işlevi görür. Bu ağlara yenilemeli denmesinin sebebi gelecekte öngörülemeyen değerlerin tahmininde depolanan verilerin karakteristiğine göre tüm veri setlerinin işlenmesidir. Tipik bir RNN de karşılaşılan en büyük zorluk ağın yalnızca birkaç adım öncesini hatırlayabilmesidir. Daha uzun süreli veriler için bu hatırlama yetisi uygun değildir. Bu zorluk "hafıza hattı" ile tanıtılan LSTM yenilemeli ağı ile çözülmüştür (Siami-Namini vd., 2018).

LSTM özel çarpımsal ünitelerdeki sabit hata karoselleri boyunca zararsız olduğu yerlerde, sabit hata akışını zorlayarak ağdaki ağırlıkları yenileme işlemini keser. Bu doğrusal olmayan üniteler sabit hata akışını düzenlemek için ağdaki kapıları açıp kapatmayı öğrenir.

Bu nedenle, LSTM alışıla gelmiş RNN algoritmasını hızlandırarak çok uzun süre gecikme ile gelen uzun süreli bilgiyi tahmin eder (Sagheer & Kotb, 2019).

2.LİTERATÜR

Fan Yonghui, Xing Yuwei ve Yang Hualong; Baltık Kuru Yük Endeksini tahminlemek için genelleştirilmiş otoregresif koşullu değişen varyans modelini (GARCH) kullanmışlardır. Çalışmaları sonucunda GARCH modelinin katsayılarının %5 anlamlılık seviyesinin altında olmasından dolayı tahminleme için uygun olduğunu belirtmişlerdir (Fan vd., 2014). Cullinane, Kevin; Baltık Kuru Yük Endeksi bağlı olan BIFFEX (The Baltic International Freight Futures Exchange) spekülasyonlarını Box-Jenkins yaklaşımıyla ARMA modelleri kullanılarak tahminleme yapmıştır. Çalışmalarında hareketli ortalama ve HOLDS modelleri, ortalama mutlak hata, varyans ve Theil'in eşitsizlik katsayısı değerleri karşılaştırılarak Box-Jenkins yaklaşımın daha iyi sonuç verdiğini belirtmiştir (Cullinane, 1992). Cullinane, Mason ve Cape; BIF (Baltic Innovation Fund) için Box-Jenkis ve AR modellerinin karşılaştırmasını yapılmış, kök ortalama kare hata (RMSE),ortalama mutlak hata (MAD) , maksimum mutlak sapma (MAXAD) ve Theil'in eşitsizlik katsayısı kullanarak ARIMA modelinin kısa süreli tahminlemede iyi olduğunu ve parametreler arttıkça Box-Jenkins yaklaşımının daha iyi sonuç verdiği

belirtilmiştir(Cullinane vd., 1999).Tsioumas, Papadimitriou, Smirlis ve Shaher; BDI için dışsal değişkenli vektör otoregresif model (VARX) ile ARIMA modelini karşılaştırmışlardır. Çalışmaları sonucunda tüm senaryolarda VARX modeli daha az tahminleme hatası ve daha yüksek R-Kare değeri verdiğini ve daha iyi tahminleme yaptığını ispatlamışlardır (Tsioumas vd., 2017).Batchelor, Alizadeh, ve Visvikis; navlun spot ve vadeli fiyatlarının tahminlemesini yapmışlardır. Vektör hata düzeltme modeli (VECM) ile spot fiyatları için en iyi tahminleme modeli olduğunu belirtmişlerdir. Ancak vadeli fiyat tahmini için ARIMA veya VAR modelinin daha iyi sonuç verdiğini gözlemlemişlerdir (Batchelor vd.,2007). Munim ve Schramm; Uzak Doğu ve Kuzey Avrupa ticaret hattının navlun oranın tahminlemesini çalışmışlardır. Model olarak ARIMA modeli ile otoregresif koşullu değişen varyans modelini (ARCH) birleştirerek oluşturdukları kendi modellerini kısa dönemli tahminleme konusunda önceki modellere göre daha düşük tahminleme hatası verdiğini belirtmişlerdir. Çalışmalarında ARIMARCH (3,1,0,3), ARIMARCH (3,1,3,5), ARIMA (3,2,0) ve ARIMA (3,2,0) modellerinin haftalık ve aylık tahminlemelerinin istatistiksel hata değerlerini karşılaştırmışlardır. Tüm tahminlemelerde gecikme değerini 10 olarak almışlardır (Munim & Schramm, 2017). Yang ve Mehmed; Dinamik YSA modellerinden doğrusal olmayan otoregresif dinamik network (NARNET) ve dış veri ile doğrusal olmayan otoregresif dinamik network (NARXNET) kullanarak Baltık Panama Endeksini (BPI) ve Baltık Vadeli Değerlendirmesi verilerini (BFA) dış veri olarak kullanarak ve tahminleyerek karşılaştırmasını yapmışlardır. Çalışmaları sonuncuda NARXNET modelinin her koşulda daha iyi performans verdiğini ve vadeli navlun anlaşmalarının (FFA) tahminlemeyi güçlendirdiğini fark etmişlerdir. Çalışmalarında bir, iki, üç ve altı aylık periyotlar için tahminleme yapmışlardır. Karşılaştırma parametresi olarak ortalama kare hata (MSE) metriğini kullanmışlardır. Gizli katmandaki nöron sayısını 10,20,30 ve 40 yaparak dört zaman dilimi için performans kıyaslaması yapmışlardır (Yang& Mehmed, 2019). Hirata ve Matsuda; Derin makine öğrenmesinde kullanılan yapay yenilemeli sinir ağı (RNN) modellerinden uzun kısa-süreli hafıza(LSTM) mimarisini kullanarak Shangay Konteyner Navlun Endeks'ini(SCFI) tahminlemiş ve SARIMA modeli ile karşılaştırmasını yapmışlardır. Çalışmaları sonucunda LSTM metodunun bazı ticaret yollarında (Güney Amerika ve Amerika'nın doğu kıyıları) tahminleme hatasının %85 oranında düşürdüğünü ancak SARIMA modelinin doğu ve batı Japonya ticaret yolları için daha isabetli tahminleme yaptığını belirtmişlerdir (Hirata & Matsuda, 2022). Lyridis, Zacharioudakis,Mitrou ve Mylonas; YSA'larını VLCC petrol tankerlerinin navlun oranlarını tahminlemek için kullanmışlardır. YSA'nın iyi bir mimari ve eğitimle yüksek volatiliteye sahip verilerde iyi bir tahminleme aracı olacağını belirtmişlerdir. Değişkenlerin diferansiyel formda olması yüksek volatilite seviyesinde iyi performans sergilerken, ortalama volatilitedeki normal formdaki değişkenler daha iyi performans göstermiştir. Karşılaştırma yapılan modellerde 1,3,6,9, ve 12 gecikme kullanılmıstır.1 gecikme seviyesi dışında YSA, NAIVE modeline göre daha az ortalama hata vermiştir. YSA tasarımında her katmandaki nöron sayısını 6,12,16 ve 18 nöron olacak şekilde eğitmişler ve en iyi ortalama hata veren nöron sayısını kullanmışlardır. Kullanılan girdi değişkenleri 6 ila 13

arasında değişkenlik göstermiştir. Kısa dönemli tahminlerde değişken sayısının az olması daha iyi performans vermiştir (Lyridis vd., 2004). Li, Parsons ve Michael ; YSA kullanarak kirli tanker spot oranı, tanker arzı ve tanker talebi verileri ile tanker navlun oranlarını tahminlemişlerdir. Çalışmaları sonucunda YSA'lar klasik zaman serisi tahminlerine göre özellikle uzun vadeli tahminler için çok daha iyi performans sergilediğini belirtmişlerdir. Bu sağlam performansın yapı taşı olarak iyi bir yapay sinir ağı ve düzgün hazırlanmış bir ağ eğitimi olmuştur. Modellerini kurarken şu adımlar izlemişlerdir; girdi verisinin sayısının tespit edilmesi, bir gizli katman kullanarak bu katmandaki nöron sayısının tespit edilmesi ve bir hata toleransı belirleyerek süreç boyunca sabit tutulması. Çeşitli gizli katman nöron sayılarını test edip eğiterek MSE değeri en iyi olan sayıyı seçmişlerdir. Seçilen gizli katman nöron sayısı sabit tutularak ve girdi veri sayısı değiştirilerek MSE değerine göre optimum girdi veri sayısı bulmayı hedeflemişlerdir. En optimal çözümü veren model bulunana kadar bu işlemler tekrarlanmıştır (Li vd., 1997). Şahin, Gürgen, Ünver ve Altin; çalışmalarında BDI'nın tahminlemesini yapay sinir ağları ile yapmışlardır. Ham petrol fiyatını da (COP) girdi olarak kullanmış ve 3 farklı model geliştirmişlerdir. İlk modelde tek girdi katmanı kullanmışlardır. İkinci modellerinde BDI'nın farklı zaman dilimlerindeki verilerini kullanarak girdi katmanındaki nöron sayısını ikiye çıkarmışlardır. Üçüncü modellerinde girdi katmanındaki veri sayısını ikide sabit tutup BDI ve COP verilerini kullanarak tahminleme yapmışlardır. Tüm modeller için gizli katmandaki nöron sayısını 1 ile 15 arasında değiştirerek MAPE ve R^2 değerlerini karşılaştırmışlardır. İlk model için 8, ikinci model için 12 ve üçüncü model için 14 nöron en iyi MAPE değerini vermiştir. Sonuç olarak YSA'nın %95 güven aralığında BDI için etkili bir tahminleme yöntemi olduğunu ortaya koymuştur (Şahin vd., 2018). Zhang, Xue, ve Stanley; BDI üzerindeki ekonometrik tahminleme modelleri ve YSA algoritmalarıyla tahminlemeleri karşılaştırmışlardır. Bu karşılaştırılmaları günlük, haftalık ve aylık olmak üzere test etmişlerdir. Çalışma sonucunda günlük tahminlemelerde ARIMA ve GARCH modeli YSA'na göre daha iyi performans vermiştir. YSA'lar haftalık ve aylık tahminlemelerde büyük bir üstünlük sağlamıştır. Ayrıca YSA'ların girdi verileri ve tahminleme ufku konusunda hassaslığına dikkat çekmişlerdir. YSA eğitimi için kullanılan birden fazla eğitim setinin tahminlemeleri farklılık göstermektedir. Bu nedenle hiçbir model her durum için en iyi seçenek değildir (Zhang vd., 2018).

3.MATERYAL VE METOD

3.1.MATERYAL

Çalışmada Baltık Kuru Yük Endeksinin hesaplamaya dahil edilen 2008 global ekonomik krizi ve COVID-19 global pandemi krizi anlarındaki veriler kullanılarak tahminleme çalışması yapılmıştır.

2008 global ekonomik krizi için alınacak örneklem:

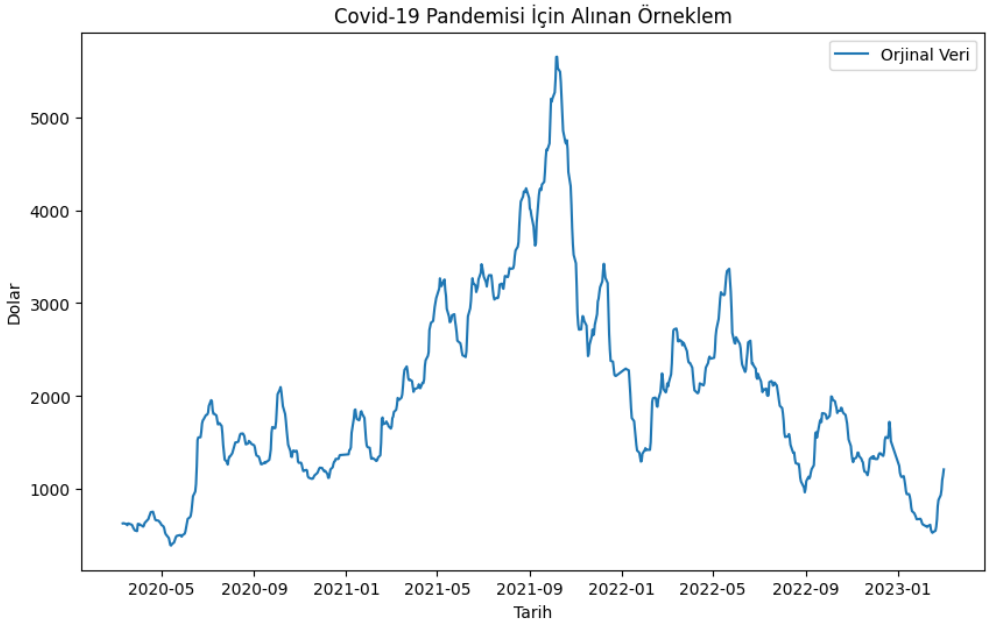
- Baltık Kuru Yük Endeksi 3 Aralık 2007 – 06 Ocak 2009 tarihleri arasıdır.

COVID-19 global pandemi krizi için alınacak örneklem:

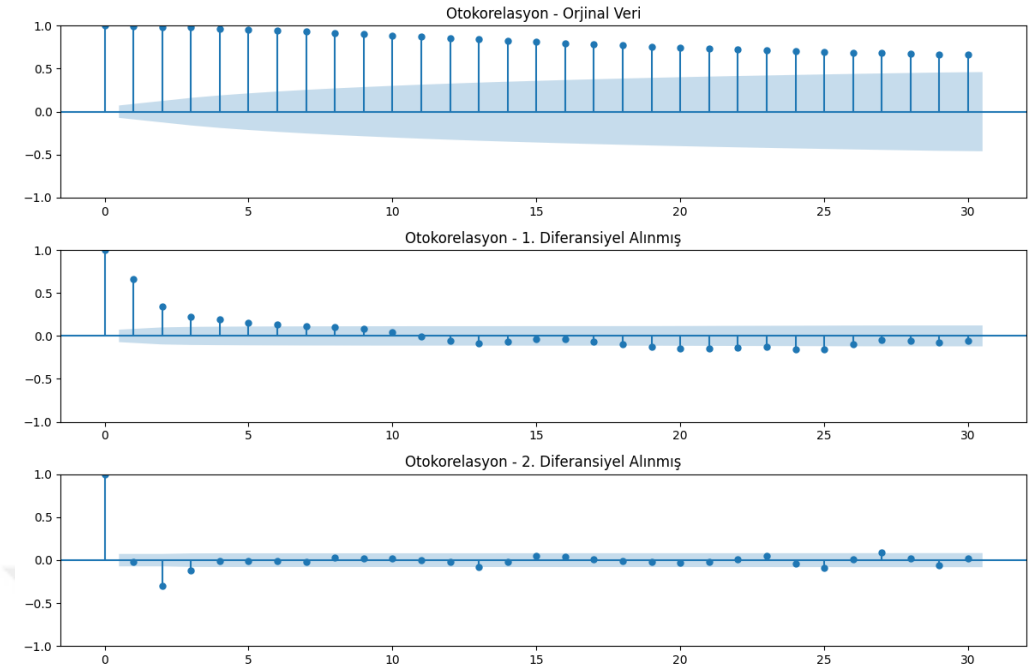
- Baltık Kuru Yük Endeksi 11 Mart 2020 – 03 Mart 2023 tarihleri arasıdır

YSA eğitimleri için kullanılacak veri setinin yüzde sekseni (%80) ve yüzde altmışı (%60) ile iki şekilde yapılmıştır. Eğitim setinin yüzde yirmi (%20) oranında bölünerek model validasyonu için kullanılmıştır ve kalan kısmın tahminlemesi yapılmıştır.

Şekil 3.1 ve Şekil 3.4'te iki veri grubunun grafikleri verilmiştir. Grafiklerde görüldüğü üzere her iki veri grafiğinde de kriz süresince sabit bir trend, mevsimlik ya da doğrusal bir artış-azalış görülmemektedir. Veri ilk bakışta doğrusal tahminleme yöntemleri için uygun olmadığını grafiğinde görülmektedir.



Sekil 3.1. Covid-19 Pandemi Verisi Grafiği

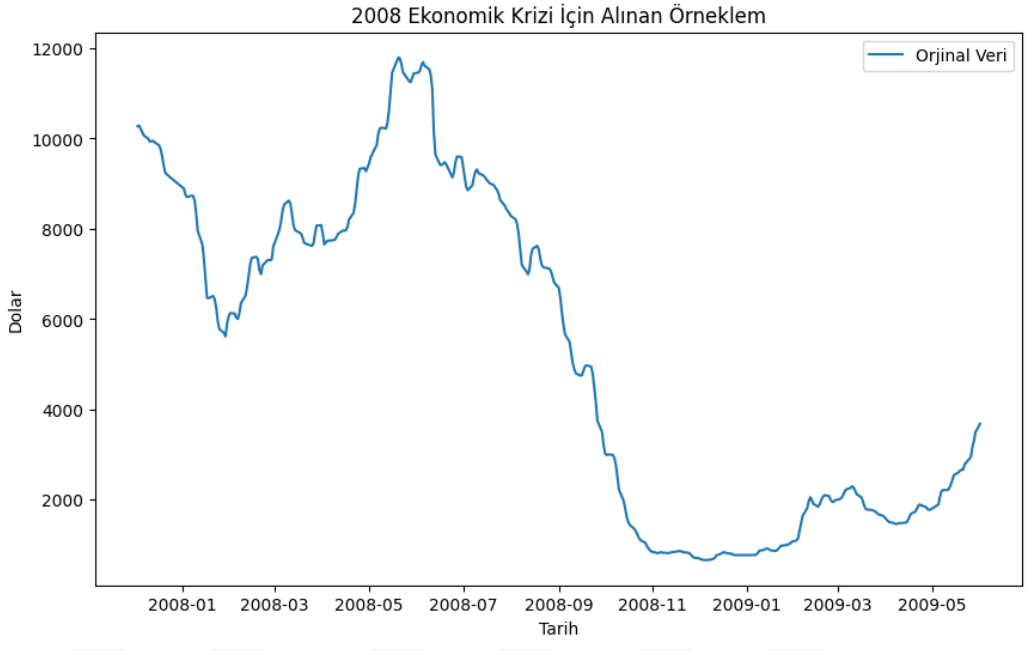


Şekil 3.2. Covıd-19 Pandemi Verisi Korelasyon Grafikleri.

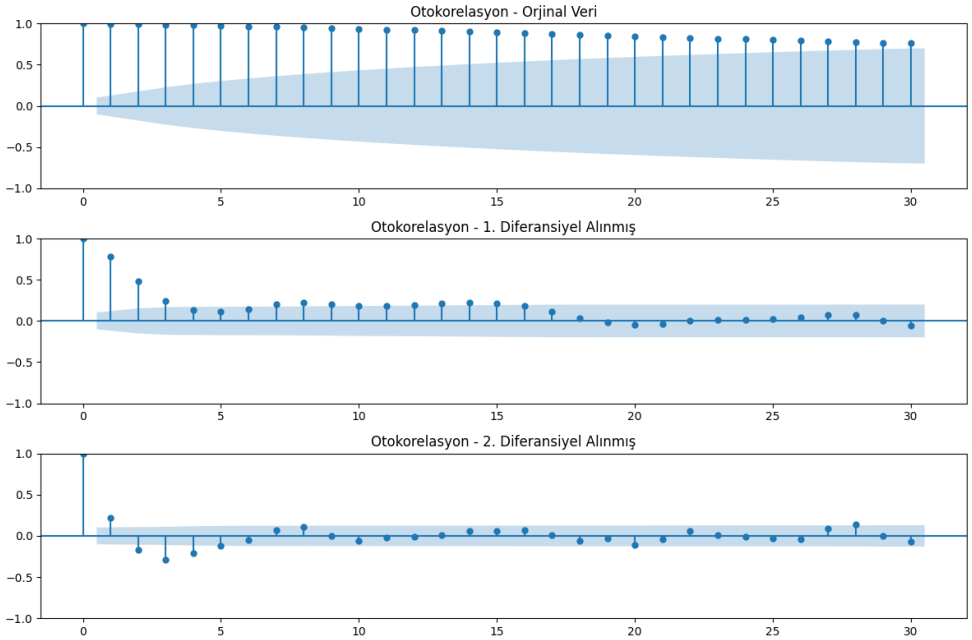
Şekil.3.2 ve Şekil.3.4'te iki veri grubunun korelasyon grafikleri görülmektedir. İki orijinal veride de çok fazla pozitif korelasyon bulunmaktadır. Tablo 3.1 ve Tablo 3.2 de orijinal veri ve korelasyon seviyeleri için ADF istatistiği sonucu, p-değerleri ve ADF kritik değerleri verilmiştir. Tablolardaki ADF değerlerinden iki veri grubunun da durağan olmadığı anlaşılmaktadır. Korelasyondan kurtulmak ve durağanlığı sağlamak için diferansiyelleri alınmıştır. İki seviye diferansiyel alınmış korelasyon kaybolmasına rağmen da ADF değerleri kritik değerlerin altındadır ve p-değeri çok aşırı küçük çıkmaktadır.

Tablo.3.1. Covid-19 P-değeri ve ADF İstatistikleri

	İstatistik	p-değeri	Kritik Degerler
Orjinal Veri	-2,271	0,181	{'1%': -3.439, '5%': -2.865, '10%': -2.568}
Birinci Diferansiyel	-10,301	3,37E-18	{'1%': -3.4392, '5%': -2.865, '10%': -2.568}
İkinci Diferansiyel	-15,451	2,78E-28	{'1%': -3.439, '5%': -2.865, '10%': -2.568}



Sekil 3.3. 2008 Ekonomik Krizi Verisi Grafiği



Şekil 3.4. 2008 Ekonomik Krizi Verisi Korelasyon Grafikleri.

Tablo.3.2 2008 Ekonomi Krizi P-değeri ve ADF İstatistikleri

	İstatistik	p-değeri	Kritik Degerler
Orjinal Veri	-1,514	0,526	{'1%': -3.448, '5%': -2.869, '10%': -2.571}
Birinci Diferansiyel	-9,032	5,39E-15	{'1%': -3.448, '5%': -2.869, '10%': -2.570}
İkinci Diferansiyel	-9,034	5,34E-15	{'1%': -3.448, '5%': -2.8696, '10%': -2.571}

Verilerin durağanlaştırılmasına ve korelasyonun yok edilmesine rağmen denenen hiçbir (ARIMA, VARX, GARCH) doğrusal modelleri iyi öngörü yapamamıştır. Bu nedenle tahminleme için son dönemlerde oldukça popüler YSA modelleri tercih edilmiştir.

3.2.METOD

Çalışmanın modelleri bilgisayar ortamında hazırlanmıştır. Verilerin tahminlenmesi için 1D-CNN, LSTM, CRNN modelleri Python programlama dilinde kodlanmıştır. Kod içerisindeki Drop-out ve L2 regülasyonları verilerin aşırı uyumlu olmasından kaçınmak için kullanılır. L2 regülasyonu girdi ağırlıklarını belirlenen ceza oranına göre düzenler (Bakır vd., 2023). Drop-out katmanları çok katmanlı YSA'larda ağ içerisinde ezber yapan düğümleri yok ederek aşırı öğrenmeyi engeller (Doğan & Türkoğlu, 2019).

Modeller için optimizasyon algoritması olarak Adaptif Moment Tahmini (ADAM) seçilmiştir. ADAM algoritması, eğitim verilerine dayanan ağ ağırlıklarını güncellemek için klasik stokastik gradyan tabanlı yöntemler yerine kullanılabilecek bir optimizasyon algoritmasıdır (Çiğdem & Çırak, 2019).. Optimizasyon algoritmaları değer kaybı fonksiyonunu minimize ederek efektif ögrenmeyi sağlar. Soydaner(2020), yaptığı çalışmasında ADAM algoritması ve türevlerinin birçok veri setinde veri kaybı değerini daha iyi minimize ettiğini belirtmiştir. Zaheer ve Shaziya (2019), yayınladığı çalışmada kullandığı dört veri setinin 3'ünde istenen 1 değerini, en erken eğitim iterasyonunda ADAM algoritması üretmiştir. Çalışmada önceki çalışmalar incelenmiş ve ADAM algoritmasının kullanılmasına karar verilmiştir. Tüm modeller için kayıp fonksiyonu (loss function) için ortalama hata karesi seçilmiştir.

Her iki kriz datası için %80, %60 eğitim verisi olacak şekilde bölünüp geriye kalan kısmının tahminlemesi yapılmıştır. Eğitim verisinin %20 si validasyon verisi olarak alınarak, erken durma (early stop) ayarı için kullanılmıştır. Böylece modellerin fazla iterasyonlarla aşırı uyumlu olmasından kaçınılmıştır. Her dönem için on iterasyon yapılarak ortalama mutlak hata değerleri (RMSE), Akaike bilgi kriteri(AIC), Bayesian bilgi kriteri (BIC) değerleri not edilerek ortalaması alınmıştır. Bilgi kriterleri, modellerdeki tahmin edilen toplam değişken sayısı, varyans ve kovaryans matrisleri, kullanılan gözlem sayısı gibi değişkenleri formüle ederek iki modelin uyumluluk seviyesinin karşılaştırılmasını sağlar (Bilgili, F. 2001).

3.2.1.Genel Kod Akışı

Tüm modellerin Python kodları benzer kod akışına sahiptir. Kod akışı aşağıdaki gibidir:

1-Bulut hesabına bağlanılması.

2-Gerekli kütüphanelerin yüklenmesi

3-Bulut hesabında bulunan veri dosyasının alınması

4-Verilerin artan şekilde tarihe göre sıralanması

5-Verinin istenen oranlarda bölünmesi

6-Bölünen verinin ölçeklendirilmesi

7-Verilerin kaç adım öncesi referans alınacağının belirlenerek girdi-çıktı formatına dönüştürülmesi.

8-YSA modelinin kurulması.

9-Erken duruş ayarının yapılması

10-Modelin eğitilmesi

11-Tahminleme yapılacak test verisinin tahminlemesinin yapılması

12-Model metriklerinin (RME, AIC, BIC) hesaplanması

13-Eğitim, tahmin ve test verilerinin ters dönüşümle ölçeklenmeden arındırılması

14-Verilerin grafiğe dökülmesi

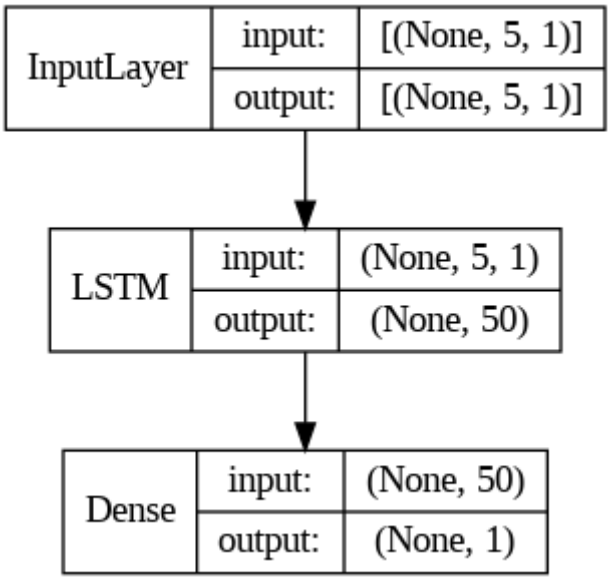
15-Modelin şemasının çıkarılması

3.2.2 Kurulan Modeller

Çalışma için üç farkı model kullanılmıştır. Bunlar; LSTM ,1D-CNN ve iki modelin birleşiminden oluşan CRNN modelidir. Tüm modeller Python yazılım dili kullanılarak modellenmiş ve kod içerisindeki parametreler değiştirilerek en iyi parametre kombinasyonu yakalanmaya çalışılmıştır.

3.2.2.1.LSTM Model

Kurulan LSTM modelin mimarisi Şekil 3.5'te gösterilmiştir. Model kurulurken Python Tensorflow Keras LSTM kütüphanesi kullanılmıştır. Aşağıdaki Şekil 3.6'da kullanılan diğer kütüphaneler gösterilmiştir.



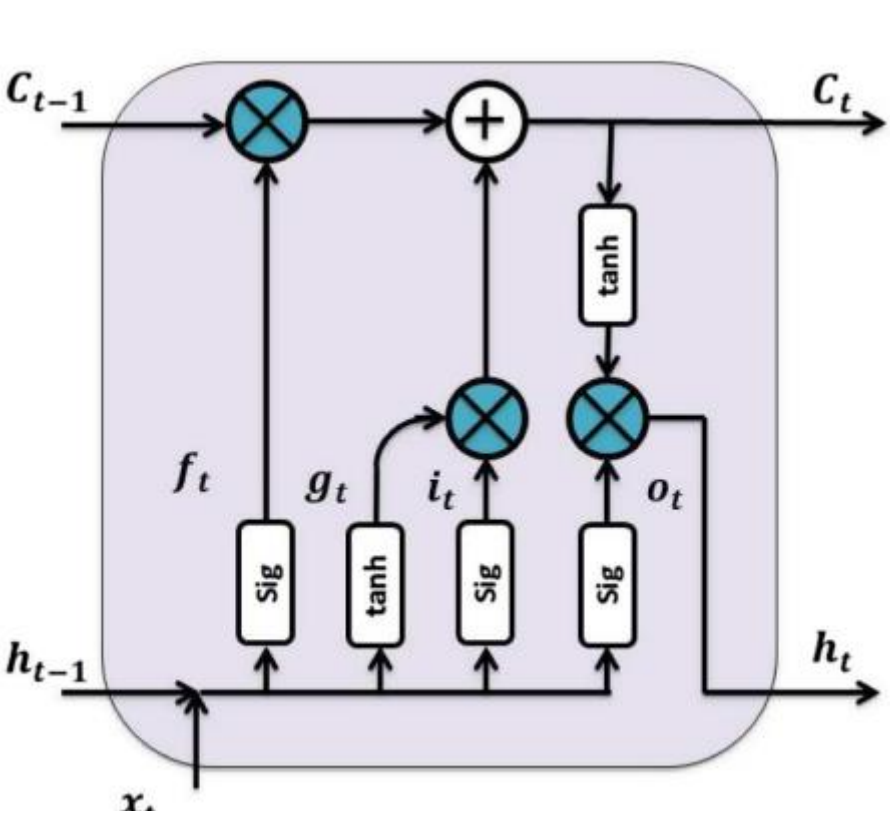
Şekil 3.5. LSTM Model Mimarisi.

Şekil 3.5'teki model mimarisinde görüldüğü üzere girdi katmanı verilerin geriye bakma değeri 5 alınarak düzenlendiği için verilerin 5 er 5 er gelmesini ve tek özellik barındırdığını belirtmektedir. Katmanlardaki "None" yazıları tek seferde yapılacak iş sayısının girilmemesinden dolayı yazmaktadır.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
from tensorflow.keras.regularizers import L2
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import plot_model
```

Şekil 3.6. LSTM Python Kodu İçin Kullanılan Kütüphaneler

Bu değerin boş bırakılması herhangi bir değişkenin iş boyutunun kabul edileceği anlamına gelerek esneklik sağlamaktadır. Giriş katmanındaki 5 ve 1 sayıları tek seferde 5 verinin işlenerek kendilerinden sonraki 1 veriyi tahminleme yapacağını belirtir. Her seferinde giriş katmanından veriler 5 er 5 girmektedir. LSTM katmanını beynin özelleşmiş bir bölümü gibi düşünebiliriz. Bu özel bölge gönderdiğimiz verinin zamanla nasıl bir davranış sergilediğini çözümler. Başlangıç zamanından itibaren ilk 5 veri ile 6. veri arasındaki ilişkiyi anlamlandırmaya çalışır ve bu döngü tüm veriler işlenene kadar devam eder. Bu işlenme sırasında her 5 veriye rastgele bir ağırlık verilir. Ve LSTM hücresi üç farklı kapıya sahiptir. Bunlar girdi kapısı, unutma kapısı ve çıktı kapısıdır. Şekil 3.7'de LSTM hücresi şematize edilmiştir.



Şekil 3.7. LSTM Hücresi (Smagulova, K., & James, A. P. ,2019)

Şekil 3.7'de görüldüğü üzere çarpım ve artı işaretleri bir sonraki hesaplamada hangi değerlerin çarpılıp toplanacağını söylemektedir. Şekildeki t-1 bir önceki adımdaki değerleri, t ise şu anki adımın değerlerini sembolize etmektedir. "C" hücrenin durumunu (cell state) temsil eder ve LSTM modelin anahtar elemanıdır. Bilgi geçişi, hücreler arasında hücre durumu aracılığı ile sağlanır. "h" hücrenin gizli durumu (hidden state) temsil eder. Gizli durum, o ana kadar ki modelin gelen veriler arasındaki öğrenilen ilişki katsayısı olarak düşünülebilir. "f" unutma kapısını (forget gate) temsil eder. T anında

gelen veri ve bir önceki hücreden gelen gizli durumdaki bilgi ile modelin rastgele belirlemiş olduğu katsayı matrisinden gelen katsayı "W" değerinin vektör çarpımı yapılır. Vektörel çarpım sonucunda ortaya çıkan her iki değere yanlılık (bias) değeri eklenir. İki değer sigmoid fonksiyondan geçer ve eğer sonuç 0'a yakın ise verinin hatırlanma oranı düşük, 1'e yakın ise yüksek olur. Tüm yanlılık ve katsayı matrisi değerleri iterasyonlarla model tarafından belirlenir. İstenildiği taktirde bu değerlerin hangi değerden başlatılacağı ayarlanabilir. "g" değeri aday durumunu (candite state) belirtir ve t anındaki hücre durumuna eklenecek olan yeni bilgiyi temsil eder. "i" girdi kapısını(input gate) temsil eder ve yeni eklenecek verinin ne kadarının hücre durumuna ekleneceğini belirler. Bu iki değer çarpılarak C(t-1) değerine eklenir ve yeni hücre durumu hesaplanmış olur. Son olarak "o" çıktı kapısını (output gate) temsil eder. Bu değer yeni hesaplanan C(t) değerinin ne kadarının yeni h(t) değerine ekleneceğini belirler. Her hücre girdileri bağımsız olarak işler ve hücre durumu ile gizli durumu diğer hücreye aktarır. Böylece bellekte tutulan bilgi kaybolmadan veriler arasındaki ilişki daha hassas yakalanır. Tablo.3.3'te tüm hücre elamanlarının hangi formülerle hesaplandığı ve bir iterasyon için başlangıç değerleri verilerek yaklaşık değerleri hesaplamaları yapılmıştır. Bu süreç verilerin hepsi için uygulanır ve veri ağırlıkları belirlenerek bir sonraki eğitim iterasyonu için referans sağlar. 50 LSTM hücresi bu adımlarla girdi dizilerini işler ve bunları daha üst düzey bir temsile dönüştürür. Tüm hücrelerden gelen 50 çıktı L2 regülasyonundan geçirilerek yoğunluk katmanında son tahmini üretmek için doğrusal bir dönüşüm gerçekleştirir.

Tablo 3.3. LSTM Hücresi Formülleri ve Hesaplamaları (ElSaid vd., 2016)

Başlangıç Değerleri	Formüller	Yaklaşık Değerler
$Wf=0.8,\ bf=0.2, x(t)=0.6,\ h(t-1)=0.4$	$f(t)=\sigma(Wf\cdot[h(t-1),x(t)]+bf)$	$\sigma(0.8\cdot[0.4,0.6]+0.2)\approx 0.73$
$Wi=0.4\ bi=0.1$	$i(t)=\sigma(Wi\cdot[h(t-1),x(t)]+bi)$	$\sigma(0.4\cdot[0.4,0.6]+0.1)\approx 0.62$
$Wc=0.3\ bc=0.5$	$g(t)=\tanh(Wc\cdot[h(t-1),x(t)]+bc)$	$\tanh(0.3\cdot[0.4,0.6]+0.5\approx 0.66$
$C(t-1)=0.2$	$C(t)=f(t)\cdot C(t-1)+i(t)\cdot g(t)$	$0.73\cdot 0.2+0.62\cdot 0.66\approx 0.5552$
$Wo=0.9\ bo=0.6$	$o(t)=\sigma(Wo\cdot[h(t-1),x(t)]+bo)$	$\sigma(0.9\cdot[0.4,0.6]+0.6)\approx 0.82$
	$h(t)=o(t)\cdot\tanh(C(t))$	$0.82\cdot 0.5552\approx 0.455184$

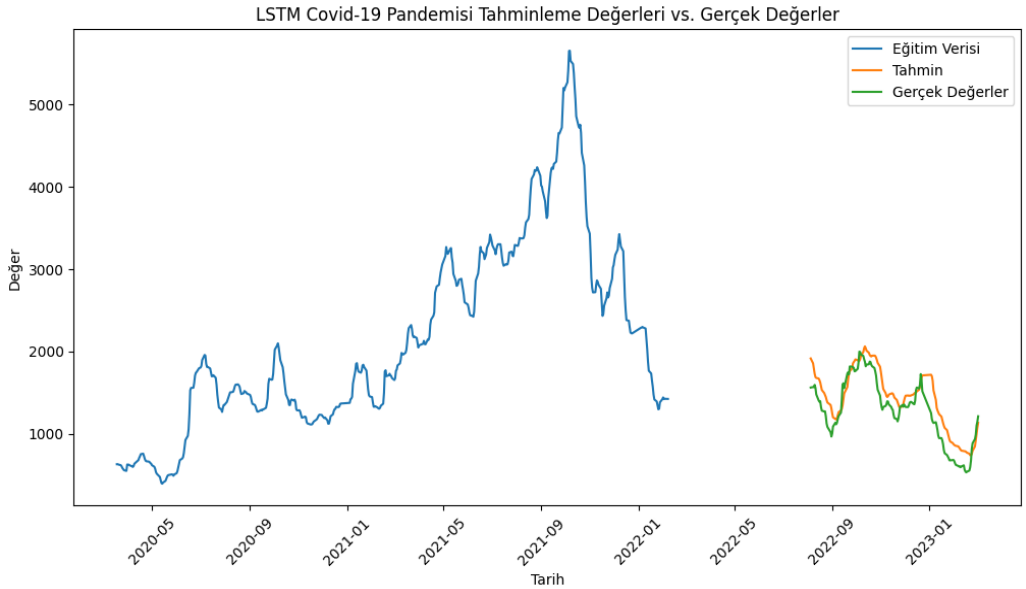
Şekil 3.8'te LSTM modelin Python kodu verilmiştir. Model 200 eğitim döngüsü ile erken durma değeri 5 olacak şekilde ayarlanmış ve değer kaybı kontrol edilerek eğitimi yapılmıştır. Değer kaybı, tahminleme değeri ile validasyon verisi arasındaki ortalama hata karesi

değeridir. Erken durma, eğer değer kaybı 5 kere arka arkaya iyileşmezse eğitimi durduracaktır. Bu sayede fazla öğrenme engellenerek mükemmel uyumdan kaçınılmıştır

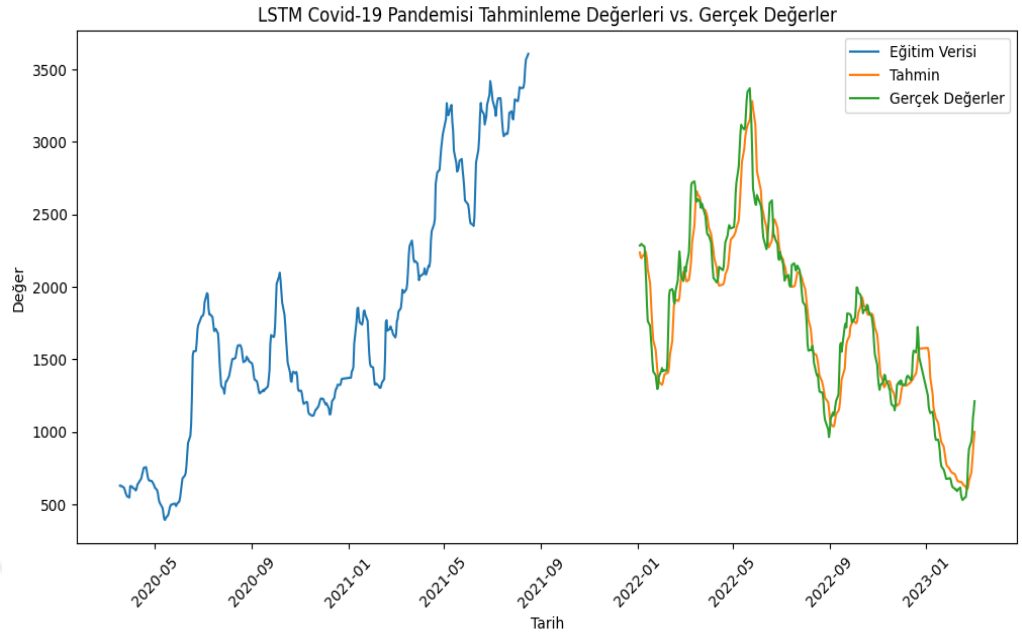
```
model = Sequential()
model.add(LSTM(50, input_shape=(look_back, 1), kernel_regularizer=L2(0.01)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
early_stopping = EarlyStopping(monitor='val_loss', patience=5, verbose=1)

model.fit(X_train, y_train, epochs=200, batch_size=32,
          validation_data=(X_val, y_val), callbacks=[early_stopping], verbose=1)
```

Şekil 3.8. LSTM Python Model Kodu.

Şekil 3.9 ve Şekil 3.10 eğitilen modelin Covid-19 verileri için bir tahminleme iterasyonunun grafiğini göstermektedir. Grafik arasında oluşan boşluk validasyon verisinin grafiğe eklenmemiş olmasından kaynaklanmaktadır.



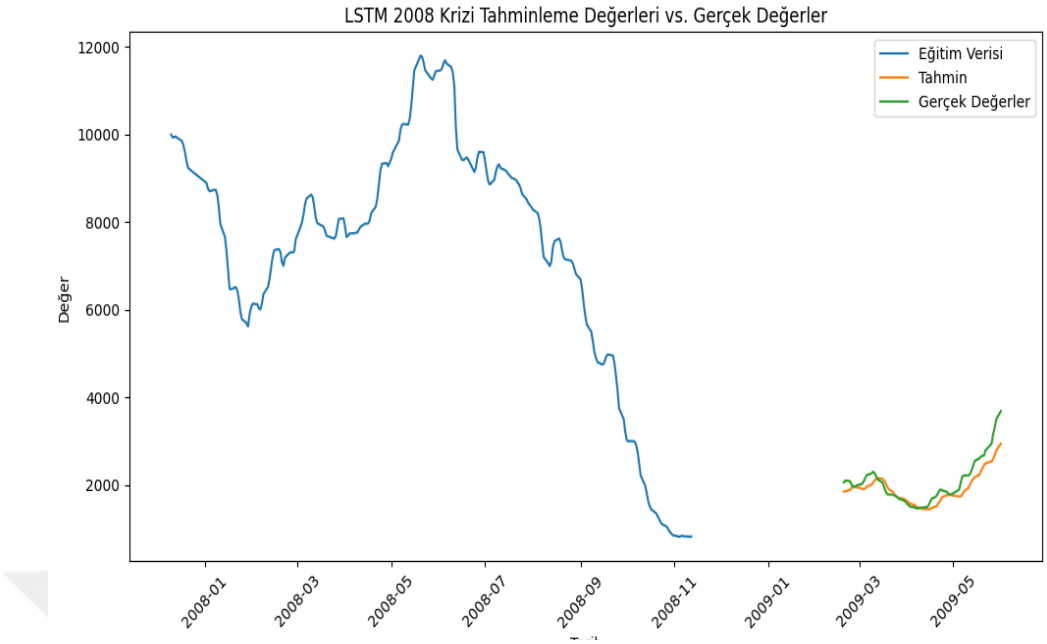
Şekil 3.9. LSTM Covid-19 pandemisi %80/20 Tahminleme Sonucu Grafiği



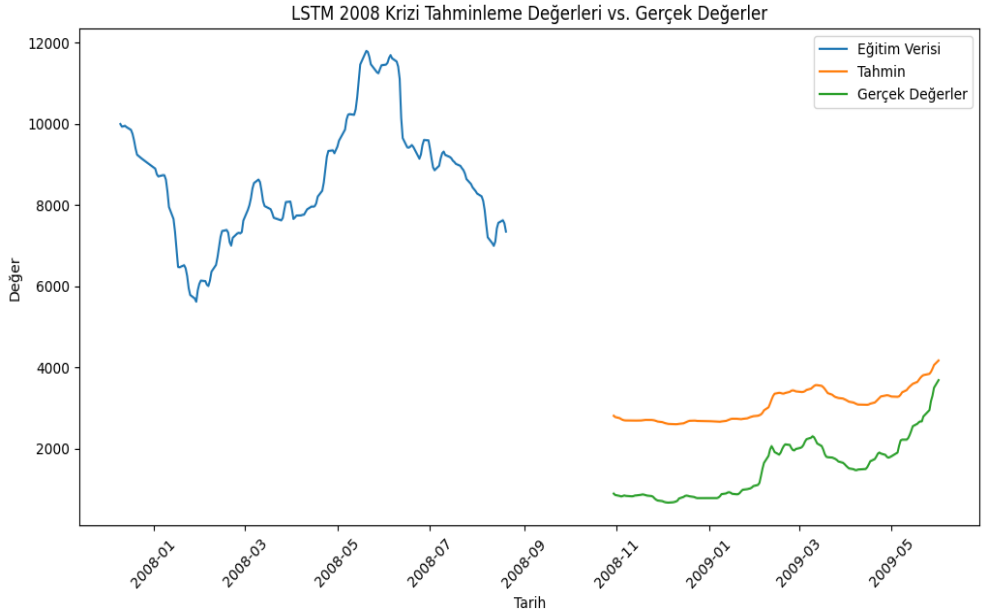
Şekil 3.10. LSTM Covid-10 %60/40 Tahminleme Sonucu Grafiği

Her iki grafikte görüldüğü üzere LSTM modeli, mükemmel uyum olmadan veriler arasındaki ilişkiyi yakalamış ve iyi tahminleme sonuçları vermiştir.

Şekil 3.11 ve Şekil 3.12 LSTM modelinin 2008 ekonomik krizi verisi için bir tahminleme iterasyonunun grafiğini göstermektedir. %80 bölünmede LSTM modelinin veri ilişkisini iyi yakaladığı görülmüştür. %60 veri bölünmesinde ise veri setindeki sert düşüş eğitim verisi dışında kaldığı için tahminle gerçeklesen değerler arasında yaklaşık 1000 dolarlık fark gözlemlenmektedir. Ancak eğrilerin birbirine benzer olması ilişkiyi yakalama konusunda iyi bir kanıt olmaktadır.



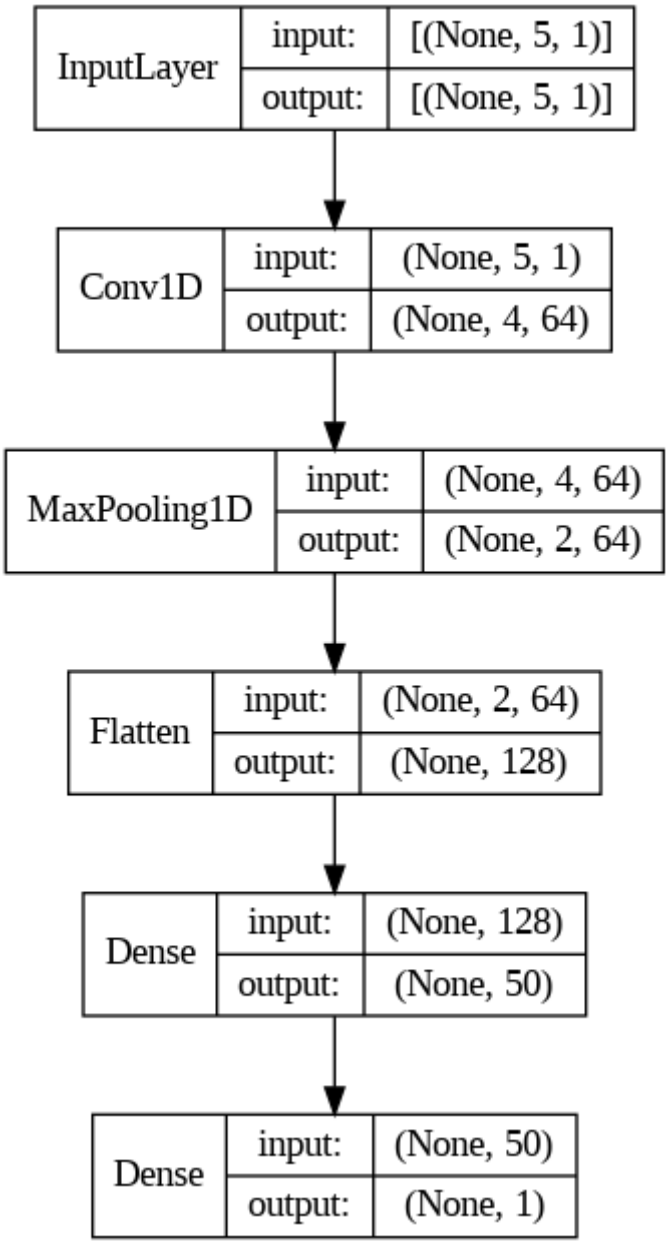
Şekil 3.11. LSTM 2008 Krizi %80/20 Tahminleme Sonucu



Şekil 3.12. LSTM 2008 Krizi %60/40 Tahminleme Sonucu

3.2.2.2.1D-CNN Model

Şekil 3.13 te 1D-CNN modelinin şeması verilmiştir. 1D-CNN model için Tensorflow ve Keras kütüphanelerinden yararlanılmıştır. Şekil 3.13'teki model mimarisinde görüldüğü üzere girdi katmanı verilerin geriye bakma değeri 5 alınarak düzenlendiği için verilerin 5 er 5 er gelmesini ve tek özellik barındırdığını belirtmektedir.



Şekil 3.13. 1D-CNN Modelin Mimarisi

.

Evrişim katmanı gelen veriler "kernel_size" olarak isimlendiren parametreye göre ardışık verileri inceler. Modelimiz de bu değer 2 olarak belirlenmiştir. Filtre gelen veri üzerinde kayarak 4x1 seklinde bir değer matrisi oluşturur. Bu matris sırasıyla (1, 2), (2, 3), (3, 4), (4, 5) elaman çiftleri arasında Kernel ağırlığı ile çarpılıp bu değerlerin toplanmasıyla oluşturulur. Eğer bu değer ReLu fonksiyonundan geçerse kendi değerini aksi takdirde 0 değerini alır. Kernel ağırlıkları başta rastgele olarak verilir ve iterasyonlarla optimum değer bulunur.

Tablo.3.4 1D-CNN Katman İşlemleri.

Girdi Değerleri	Evrişim Dönüşümü	Havuzlama Katmanı	Düzleştirme Katmanı
x1	(x1*w1)+(x2*-w2)=t1	max(t1,t2) =z1	z1,z2
x2	(x2*w1)+(x3*-w2)=t2	max(t3,t4)=z2	
x3	(x3*w1)+(x4*-w2)=t3		
x4	(x4*w1)+(x5*-w2)=t4		
x5			

Her 64 node 4 boyutlu bir çıktı dizisi üretir. Havuzlama katmanı, havuzlama işlemi gerçekleştirerek bu 4x1 boyutlu matrislerin boyutunu azaltır. 2 komşu değerden en yükseğini alır ve 2x1 boyutlu bir matris çıktı verir. Düzleştirme katmanı 64 birimli tamamen bağlı bir katmandır ve dizilerdeki 2 boyutu kaldırarak doğrusallaşmayı sağlar. Tablo 3.1'de katmanların nasıl işlediği gösterilmiştir. 64 adet 2x1 matris verisi olduğundan dolayı yoğunlaşma katmanı 128 noddan oluşur. Yoğunlaştırma katmanları tüm nodlardan gelen değerleri aktivasyon fonksiyonundan geçirir ve doğrusal bir dönüşüm yaparak son tahminlemeyi yapar. 1D-CNN model için Tensorflow ve Keras kütüphanelerinden yararlanılmıştır. Şekil 3.14 de kullanılan diğer kütüphaneler gösterilmiştir.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.utils import plot_model
from keras.callbacks import TensorBoard
import tensorfllayers import Conv1D, MaxPooling1D, LSTM, Dense, Flatten, Dropout
from keras.optimizers import Adamow as tf
from keras.
from keras.callbacks import EarlyStopping
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.regularizers import l2
from datetime import datetime, timedelta
from sklearn.preprocessing import MinMaxScaler
```

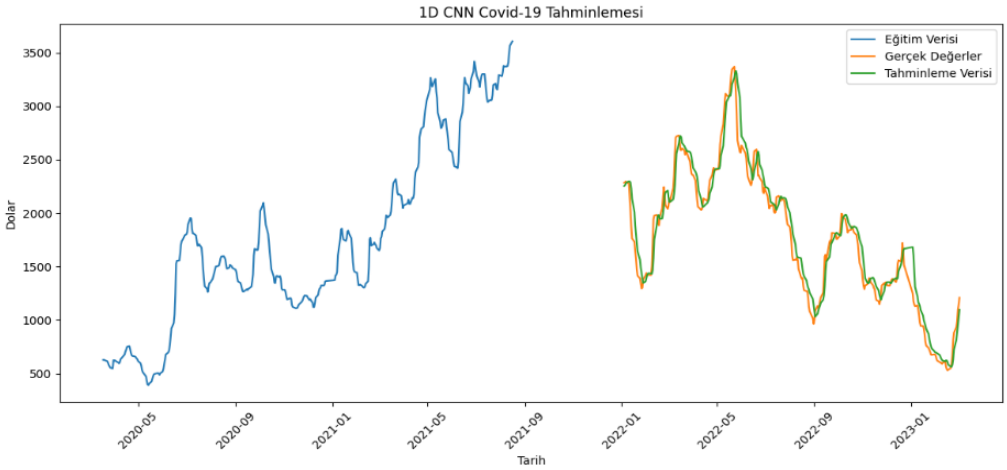
Şekil 3.14. 1D-CNN Model İçin Kullanılan Python Kütüphaneleri.

Şekil 3.15'te 1D-CNN modelinin Python kodu verilmiştir. İlk yoğunlaşma katmanı aşırı uyumdan kaçınmak için L2 regülasyonuyla regüle edilmiştir.

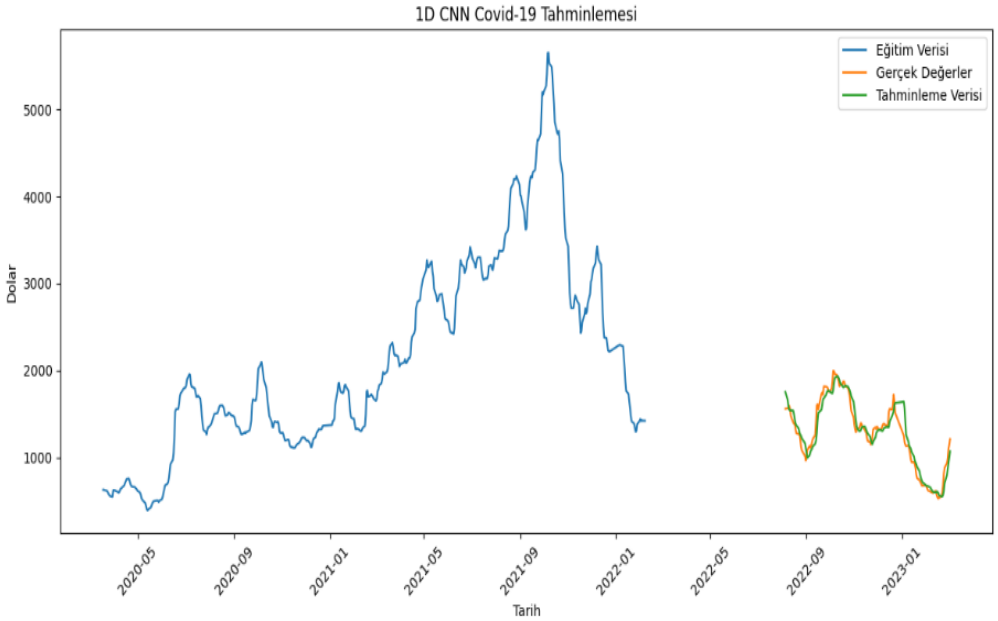
```
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(look_back, 1)))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(50, activation='relu', kernel_regularizer=l2(0.01)))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')

early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

model.fit(X_train, y_train, epochs=300, batch_size=16, verbose=1, validation_data=(X_val, y_val), callbacks=[early_stopping])
```

Şekil 3.15. 1D-CNN Modelinin Python Kodu.

Model, 200 eğitim döngüsü ve erken durdurma değeri 5 olacak şekilde değer kaybı kontrol edilerek eğitimi yapılmıştır. Erken durma koşulu LSTM modelindeki ile aynı şartlarda gerçekleşmektedir. Şekil 3.16 ve Şekil 3.17 1D-CNN modelin Covid-19 dönemi için tahminleme sonuçlarının tek bir iterasyon için grafiğini göstermektedir.



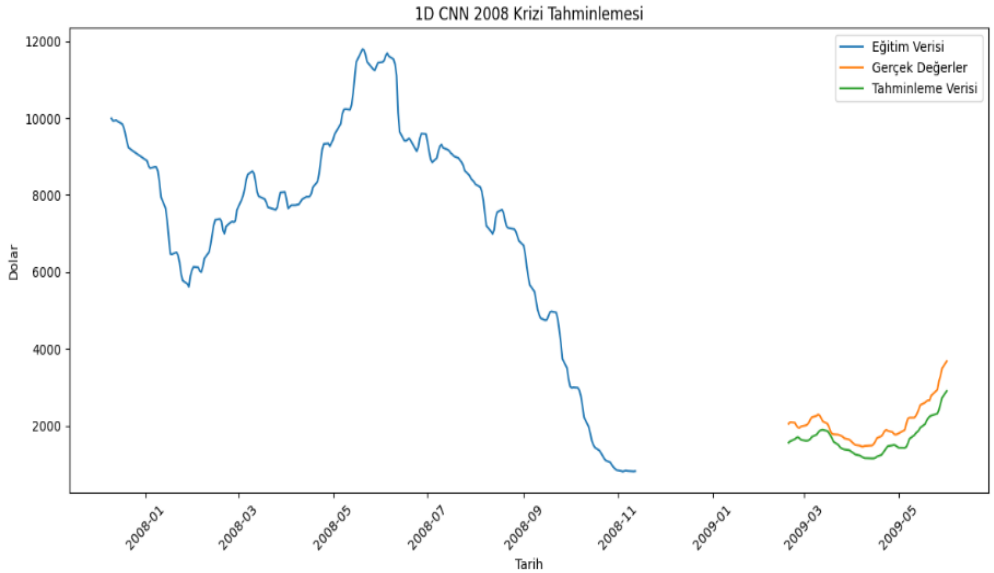
Şekil 3.16. 1D-CNN Covid-19 Pandemisi %60/40 Tahminleme Sonucu Grafiği



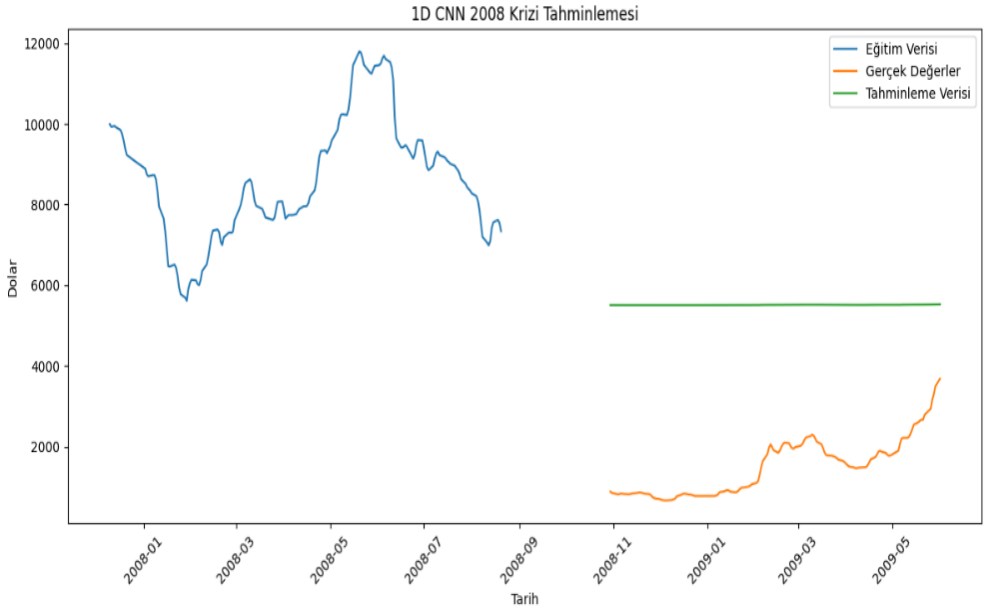
Şekil 3.17. 1D-CNN Covid-19 Pandemisi %80/20 Tahminleme Sonucu Grafiği

Her iki grafikte de 1D-CNN modelin, Covid-19 pandemisi için iyi tahmin yaptığı görülmektedir. İki veri bölünmesinde de veriler arasındaki ilişki yakalanmıştır. Grafikler arasındaki boşluklar validasyon verisinden kaynaklanmaktadır.

Şekil 3.17 ve Şekil 3.2 1D-CNN modelin 2008 ekonomik krizi dönemi için tek bir iterasyon grafiğini göstermektedir. Doğrulama verisi grafikte gösterilmemiştir.



Şekil 3.18. 1D-CNN 2008 Krizi %80/20 Tahminleme Sonucu

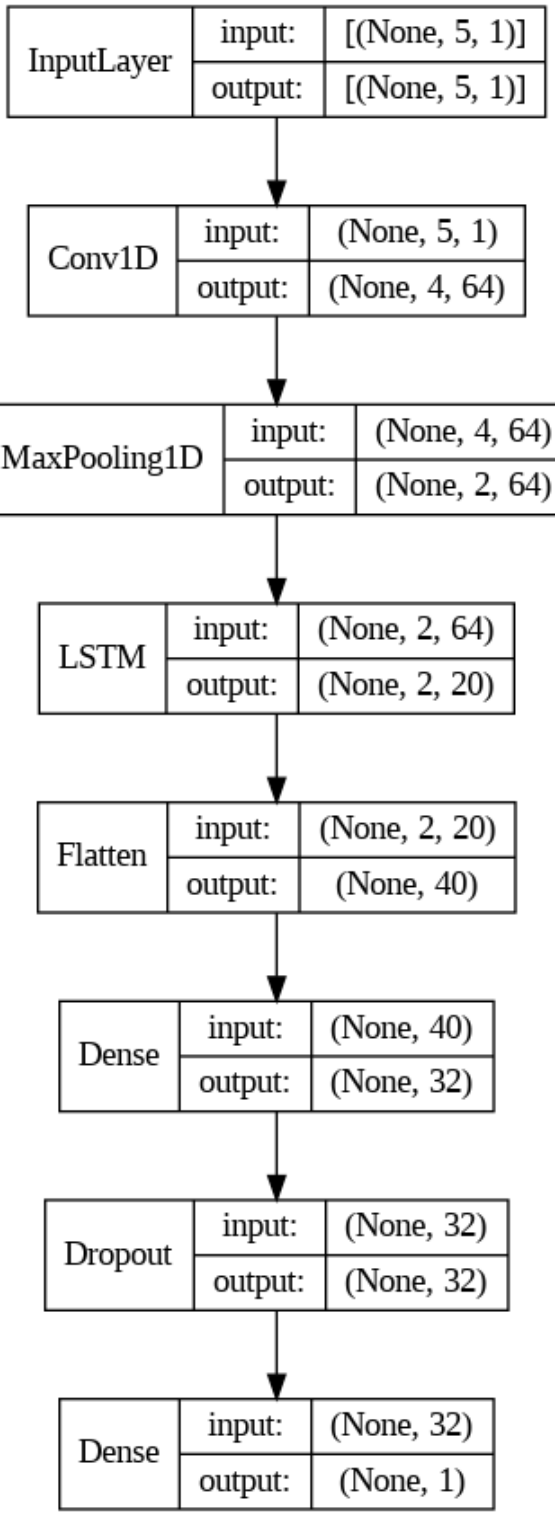


Şekil 3.19. 1D-CNN 2008 Krizi %60/40 Tahminleme Sonucu

1D-CNN modeli her ne kadar %80 veri bölünmesinde iyi bir grafik çıkarmış olsa da %60'lık bölünmede istenen performansı verememiştir ve veriler arasındaki ilişkiyi yakalayamamıştır.

3.2.2.3.CRNN Model

Şekil 3.20'de Model CRNN modelinin mimarisi verilmiştir. CRNN modeli RNN ve CNN modellerin birleşmesiyle oluşmaktadır.



Şekil 3.20. CRNN Model Mimarisi.

CRNN modeli için Tensorflow ve Keras kütüphaneleri kullanılmıştır. Şekil 3.21 de kod için kullanılan diğer kütüphaneler gösterilmektedir.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.utils import plot_model
from keras.callbacks import TensorBoard
import tensorfllayers import Conv1D, MaxPooling1D, LSTM, Dense, Flatten, Dropout
from keras.optimizers import Adamow as tf
from keras.callbacks import EarlyStopping
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.regularizers import l2
from datetime import datetime, timedelta
from sklearn.preprocessing import MinMaxScaler
```

Şekil 3.21. CRNN Modeli İçin Kullanılan Python Kütüphaneleri.

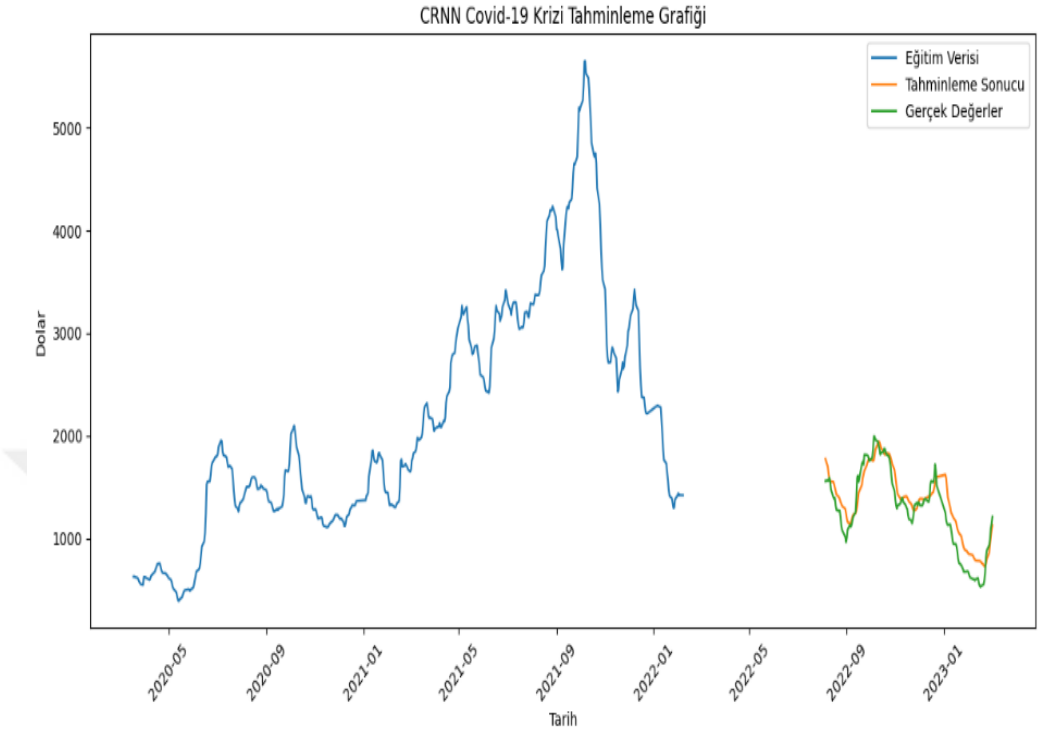
Şekil 3.20 deki model mimarisinde görüldüğü üzere, ilk olarak veriler girdi katmanından evrişim katmanına geçer. Evrişim katmanından çıkan 2x1 matris veriler 20 nodlu LSTM katmanında işlenir. Çıkan 2 boyutlu düzleştirilerek yoğunluk katmanına aktarılır. Burada L2 regülasyonu yapılarak aşırı uyumdan kaçınılmıştır ve tahminleme yapılmak üzere tek çıktıya dönüştürülmüştür.

```
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(look_back, 1)))
model.add(MaxPooling1D(pool_size=2))
model.add(LSTM(20, activation='relu', return_sequences=True))
model.add(Flatten())
model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.01)))  # L2 regularization
model.add(Dropout(0.5))  # Dropout layer for regularization
model.add(Dense(1))  # Output layer with 1 neuron for regression

model.compile(optimizer='adam', loss='mean_squared_error')

early_stopping = EarlyStopping(monitor='val_loss', patience=10, verbose=1, restore_best_weights=True)

# Train the model with updated data splits
model.fit(X_train, y_train, epochs=200, batch_size=16,
          validation_data=(X_val, y_val), callbacks=[early_stopping], verbose=1)
```

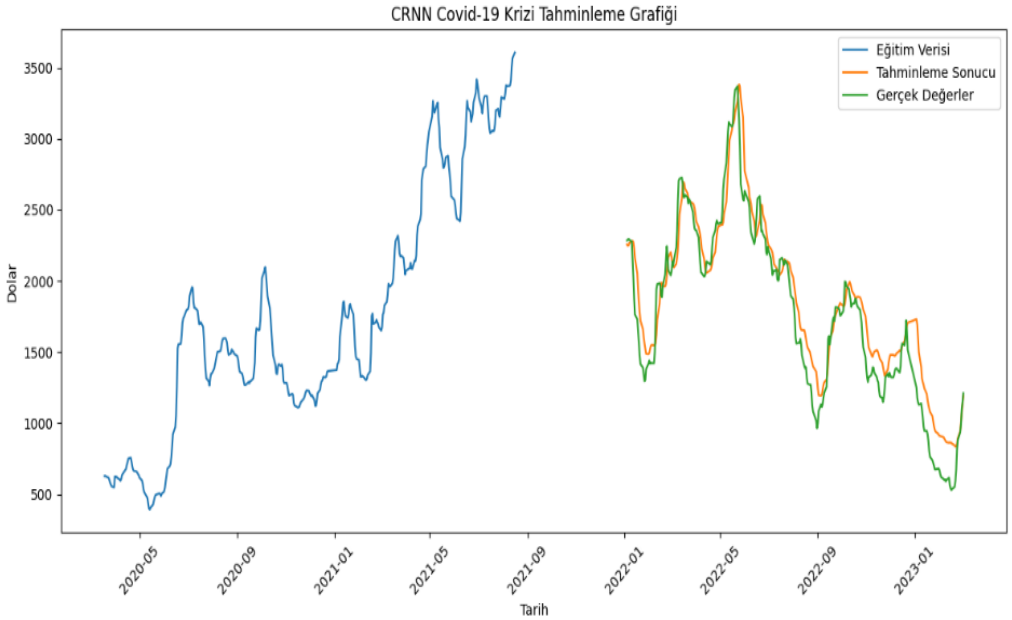
Şekil 3.22. CRNN Python Modeli

Şekil 3.22'de CRNN Python modeli kodu verilmiştir. Model 200 eğitim döngüsü planlanarak, erken stop değeri 10 olacak şekilde ve değer kaybı kontrol edilerek eğitimi yapılmıştır.

Şekil 3.23 ve Şekil 3.24 te eğitilen CRNN modelinin Covid-19 pandemisi için tahmin grafikleri gösterilmektedir.

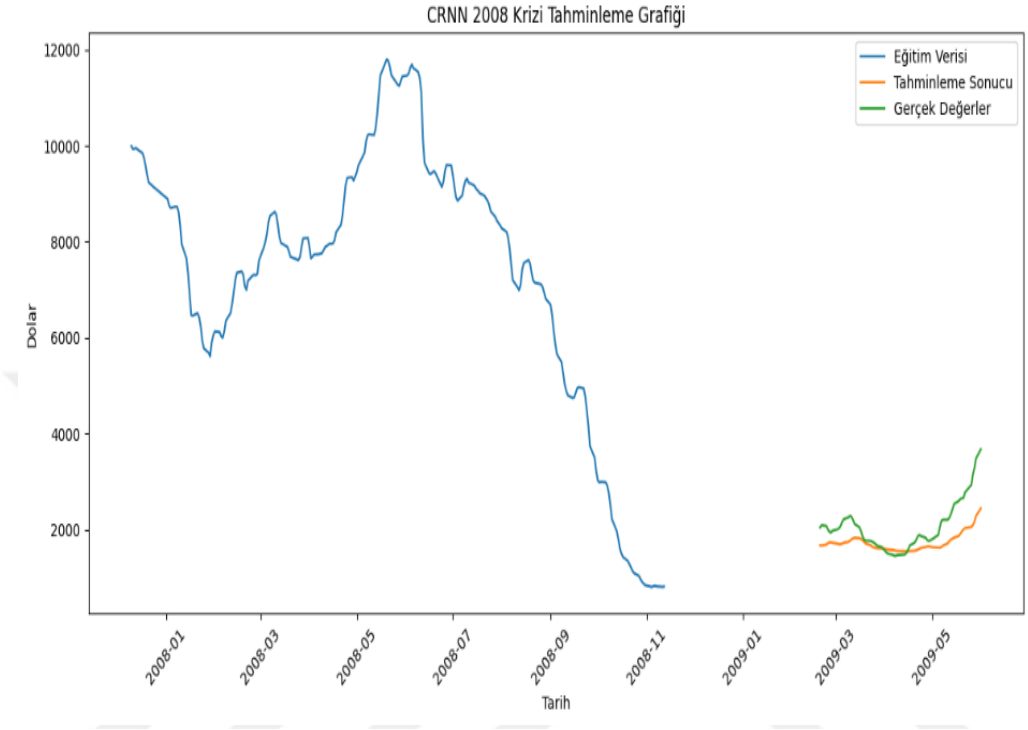


Şekil 3.23. CRNN Covid-19 Pandemisi %80/20 Tahminleme Sonucu Grafiği

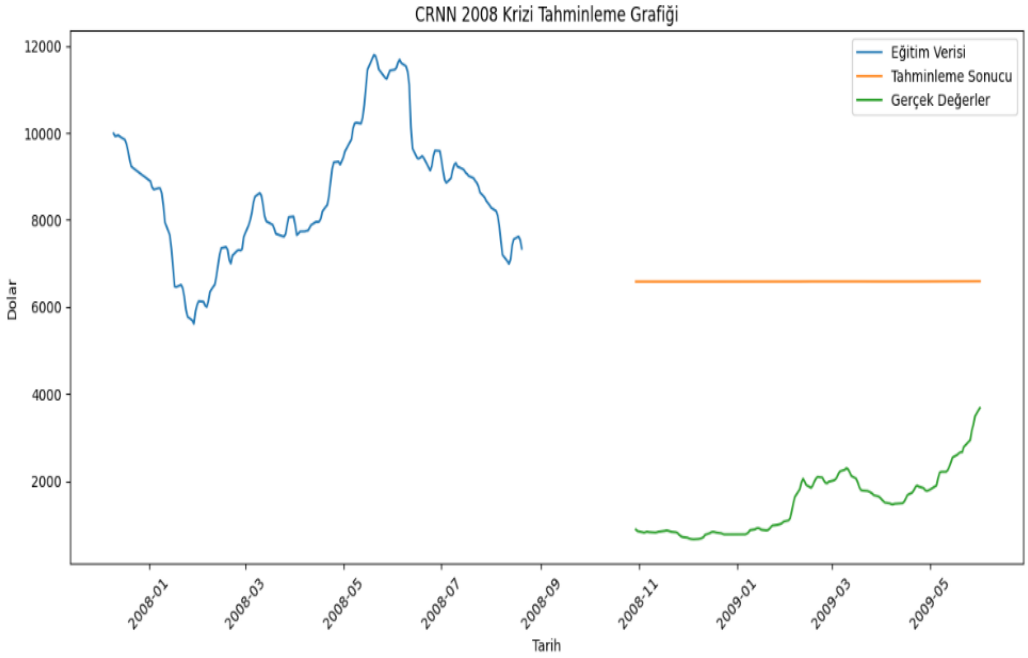


Şekil 3.24. CRNN Covid-19 Pandemisi %60/40 Tahminleme Sonucu Grafiği

CRNN modeli de her Covid-19 pandemi verisindeki her bölünme için kabul edilebilir tahminleme yapmıştır. Gerçek değerler ve tahmin verisi arasındaki makasın diğer modellere göre daha fazla olduğu göze çarpmaktadır. Şekil 3.25 ve Şekil 3.26'da 2008 ekonomik krizi için tahminleme sonuçları görülmektedir.



Şekil 3.25. CRNN 2008 Krizi %80/20 Tahminleme Sonucu Grafiği



Şekil 3.26. CRNN 2008 Krizi %60/40 Tahminleme Sonucu Grafiği

CRNN modeli %60 veri bölünmesinde veriler arasındaki ilişkiyi yakalayamamıştır. %80'li bölünmede ise belli tarihlerde gerek değerlere yakın tahmin verebilmiştir.

4.BULGULAR

Oluşturulan 3 YSA modeli ADAM optimizasyon algoritması ve değer kaybı fonksiyonu olarak ortalama hata karesi ile eğitime sokulmuştur. Tüm modeller her iki veri grubu ve veri bölünmesi için 10 iterasyon yapılarak sonuçları metrik sonuçları kaydedilmiştir. Her iterasyonda modeller yaklaşık olarak 30-90 aralığında iterasyon ile eğitilmiş ve erken durma koşulu çalışana kadar eğitim sürdürülmüştür. Modellerin eğitimi Google Colabs üzerinde ücretsiz sağlanan işlemciler üzerinde yapılmıştır. Veri setlerine göre RMSE değerleri Tablo 4.1'de verilmiştir. Tablo 4.2'de tüm iterasyonlar için AIC ve BIC değerleri verilmiştir.

Covid-19 pandemisi özelinde ortalama RMSE değeri en düşük, her iki veri bölünmesinde de 1D-CNN modelinde çıkmıştır. %80'e % 20 bölünmede ikinci olarak LSTM , %60'a %40 bölünmede ise ikinci olarak CRNN modeli iyi performans göstermektedir. Covid-19 dönemi için en iyi tahminleme yöntemi 1D-CNN olarak görünmektedir. Tablo 4.2'de görüldüğü üzere 1D-CNN her iki veri bölünmesinde de en düşük AIC ve BIC değerlerine sahiptir.

Daha kısa bir donemi içeren 2008 kriz dönemi tahminlemesinde %80'e %20 bölünmesinde en düşük RMSE değerini CRNN modeli vermiştir ve 1D-CNN modeliyle ortalama RMSE değerleri çok yakın çıkmıştır. Ancak CRNN modeli %60 a %40 veri bölünmesinde Şekil 3.26'da görüldüğü üzere veriler arasındaki ilişkiyi yakalamakta yetersiz kalmıştır. Bu bölünmede en iyi sonucu LSTM modeli vermektedir. Ancak her iki bölünme içim AIC ve BIC değerleri 1D-CNN modelinin veri için daha iyi bir model uyumu sağladığını söylemektedir.

Her iki veri grubu içinde 1D-CNN modelinin iyi tahminleme yaptığını RMSE AIC, BIC değerlerine bakarak söylenebilir.

. Tüm modellerde eğitim verisinin artması RMSE değerini düşürmektedir. YSA ile çalışırken eğitim setinin önemi ve büyüklüğü görülmektedir. 2008 %60'a %40 bölünme dışında veri setleri için en iyi sonucu 1D-CNN modeli vermiştir. BDI için ilerideki kriz durumlarında birinci tercih olarak 1D-CNN, ikinci tercih olarak LSTM modeli önerilmiştir. 1D-CNN ve LSTM modelin birleşiminden oluşan CRNN modeli iyi tahminleme yapamamıştır. Model karmaşıklığının artması tahminlemede daha iyi sonuç vereceğinin garantisini vermemektedir. Modeller ayrı ayrı iyi performans verirken birleşimlerinde oluşan modelin tahminleme gücü azalmıştır

Tablo 4.1 Covid-19 Pandemisi ve 2008 Ekonomik Krizi RMSE Değerleri

		İterasyon	1	2	3	4	5	6	7	8	9	10	Ortalama RMSE
Covid 19 Pandemisi	1D-CNN	60/40	0,046627	0,04339	0,043884	0,046402	0,044668	0,043222	0,039677	0,047390	0,043649	0,040978	0,043989
		80/20	0,020489	0,021094	0,020655	0,020051	0,020664	0,023050	0,020073	0,021791	0,020756	0,021922	0,021055
	CRNN	60/40	0,050000	0,060000	0,050000	0,060000	0,050000	0,050000	0,050000	0,050000	0,050000	0,050000	0,052000
		80/20	0,040000	0,030000	0,030000	0,030000	0,040000	0,030000	0,040000	0,040000	0,040000	0,040000	0,036000
	LSTM	60/40	0,057435	0,053556	0,054225	0,051739	0,052554	0,054110	0,054344	0,059672	0,060163	0,053555	0,055135
		80/20	0,038003	0,027797	0,031978	0,030191	0,030204	0,030834	0,035957	0,035270	0,033836	0,030895	0,032496

		İterasyon	1	2	3	4	5	6	7	8	9	10	Ortalama RMSE
2008 Ekonomik Krizi	1D-CNN	60/40	0,670140	0,648507	0,649157	0,642230	0,652202	0,634336	0,648853	0,645220	0,637687	0,670883	0,649921
		80/20	0,036310	0,033760	0,037813	0,032980	0,042117	0,041161	0,034856	0,034065	0,036966	0,042709	0,037274
	CRNN	60/40	0,820000	0,790000	0,850000	0,840000	0,840000	0,850000	0,820000	0,900000	0,620000	0,790000	0,812000
		80/20	0,040000	0,030000	0,030000	0,040000	0,040000	0,030000	0,040000	0,030000	0,050000	0,030000	0,036000
	LSTM	60/40	0,283051	0,273423	0,257786	0,275225	0,292798	0,276401	0,307335	0,244547	0,278652	0,328225	0,281744
		80/20	0,022821	0,021287	0,161595	0,147620	0,020277	0,141423	0,148562	0,142601	0,111346	0,125269	0,104280

Tablo 4.2. Covid-19 ve 2008 Ekonomik Krizi Pandemisi AIC ve BIC Değerleri

		İterasyon	1	2	3	4	5	6	7	8	9	10	Ortalama
Covid 19 Pandemisi	1D-CNN	60/40	AIC	11601,84	11559,96	11566,55	11599,03	11576,85	11557,70	11507,90	11611,28	11563,42	11526,68	11567,12
			BIC	36187,39	36145,52	36152,10	36184,58	36162,40	36143,25	36093,45	36196,83	36148,98	36112,23	
		80/20	AIC	12274,07	12282,39	12276,38	12267,89	12276,51	12307,76	12268,21	12291,69	12277,78	12293,41	12281,61
			BIC	32104,39	32112,71	32106,70	32098,21	32106,82	32138,08	32098,53	32122,01	32108,09	32123,73	
	CRNN	60/40	AIC	14969,65	14990,39	14945,20	14996,43	14949,31	14973,16	14961,27	14915,66	14979,34	14974,50	14965,49
			BIC	45594,15	45614,89	45569,70	45620,92	45573,81	45597,66	45585,77	45540,16	45603,83	45599,00	
		80/20	AIC	15738,21	15708,19	15664,18	15708,03	15720,77	15699,98	15723,91	15734,63	15756,97	15727,96	
			BIC	40439,45	40409,43	40365,42	40409,26	40422,01	40401,22	40425,14	40435,86	40458,20	40429,19	40419,52
	LSTM	60/40	AIC	19239,16	19198,47	19205,70	19178,38	19187,48	19204,46	19206,97	19261,40	19266,17	19198,45	19214,66
			BIC	57629,07	57588,37	57595,60	57568,28	57577,38	57594,36	57596,87	57651,30	57656,08	57588,36	
		80/20	AIC	19966,75	19877,31	19917,38	19900,94	19901,07	19906,97	19950,92	19945,41	19933,54	19907,53	19920,78
			BIC	50931,44	50842,00	50882,07	50865,63	50865,76	50871,65	50915,61	50910,10	50898,23	50872,22	
2008 Ekonomik Krizi			İterasyon	1	2	3	4	5	6	7	8	9	10	Ortalama
	1D-CNN	60/40	AIC	13271,52	13262,14	13262,42	13259,36	13263,76	13255,82	13262,29	13260,69	13257,33	13271,84	13262,72
			BIC	33101,84	33092,46	33092,74	33089,68	33094,08	33086,14	33092,61	33091,00	33087,65	33102,16	
		80/20	AIC	12928,44	12918,39	12934,04	12915,16	12948,91	12945,74	12922,80	12919,63	12930,91	12950,84	12931,49
			BIC	27881,31	27871,26	27886,91	27868,04	27901,79	27898,62	27875,67	27872,51	27883,79	27903,71	
	CRNN	60/40	AIC	16616,10	16606,22	16628,79	16624,32	16624,32	16628,08	16615,87	16644,00	16536,44	16605,39	16612,95
			BIC	41317,34	41307,46	41330,02	41325,56	41325,56	41329,32	41317,11	41345,24	41237,67	41306,62	
		80/20	AIC	16229,93	16185,85	16195,12	16217,16	16215,61	16178,71	16235,55	16183,47	16254,57	16195,26	
			BIC	34855,68	34811,60	34820,86	34842,91	34841,35	34804,45	34861,30	34809,22	34880,32	34821,00	34834,87
	LSTM	60/40	AIC	20541,03	20531,13	20514,29	20533,01	20550,71	20534,23	20564,57	20499,21	20536,55	20583,38	20538,81
			BIC	51505,72	51495,82	51478,98	51497,70	51515,40	51498,92	51529,26	51463,90	51501,24	51548,07	
		80/20	AIC	20380,35	20370,75	20650,47	20637,99	20364,04	20632,07	20638,87	20633,22	20599,07	20615,33	20552,22
			BIC	43729,00	43719,39	43999,12	43986,64	43712,68	43980,72	43987,52	43981,86	43947,72	43963,98	

LSTM modeli 2008 ekonomik krizi için %60'a %40 bölünme için en iyi tahminlemeyi yapmıştır. Diğer modeller verideki ani değişimleri yakalayamamıştır. Eğitim verisinin küçük olduğu veri gruplarında LSTM modelinin daha iyi çalıştığı görülmektedir.

SONUÇLAR, TARTIŞMA VE ÖNERİLER

Günümüzde ekonominin globalleşmesiyle birlikte, rekabet bütün sektörlerde artmış ve yurt sınırlarının dışına çıkmıştır. Geleceği görme ve tahminleme bu rekabette her zaman önemli bir güç olacaktır. Bu çalışmada konteyner taşımacılığında önemli bir endeks olan BDI tahminlemesi için 3 farklı YSA modeli kurulmuş ve performansları incelenmiştir. Bu çalışma sonucunda, volatilitenin yüksek olduğu global ekonomik krizlerin, erken safhalarında olan işletmelerin LSTM modelini kullanmaları daha uygun bulunmuştur. Kısa dönemde yüksek volatilite gözlemlenen krizlerde LSTM değerler arasındaki ilişkiyi daha iyi yakalanmıştır. Kriz süreci uzadıkça, başka bir deyişle, eğitim verisi arttıkça 1D-CNN modeli en iyi performansı vermiştir.

YSA modelleri doğrusal modellere göre daha iyi performans vermiştir. Doğrusal modellerin yakalayamadığı ilişkiyi 3 YSA modelinden 2'si etkili bir biçimde yakalamıştır. Her iki modelin RMSE değeri, Tsioumas ve ark. (2017) çalışmalarında kullandığı ARIMA ve VARX modellerinin RMSE değerlerinden çok daha düşük çıkmıştır. Daha düşük RMSE değerlerinin, yüksek volatiliteye sahip kriz anlarında yakalanmış olması modellerin başarısını işaret etmektedir. Bu nedenle YSA'nın borsa ve endeks tahminlemesinde doğrusal tahminleme yöntemlerinden çok daha iyi performans gösterdiğini söyleyebiliriz. Ancak Zhang ve ark. (2018) çalışmalarında belirtildiği gibi eğitim verisinin YSA'lar için oldukça önemli olduğu gözlenmiştir. Eğitim verisinin artması ve volatilite aktivitesinin eğitim setinin içeresinde olması YSA'nın performansını arttırmaktadır. Daha güçlü tahmin sonuçları için veri seti %90'a %10 bölünebilir. Daha küçük node sayılarında erken durma eşiği yükseltilerek daha uzun eğitim süre yapılabilir. Böylece 1D-CNN ve CRNN modellerinin yakalayamadığı %60'a %40 veri bölünmesinde daha etkili sonuçlar elde edilebilir. YSA'lar eğitilirken veri karakteristiği ve uzunluğu önemlidir. Şekil 3.19 ve Şekil 3.26'da görüldüğü üzere 2008 ekonomik krizindeki sert düşüş eğitim verisinde yer almamaktadır ya da çok az bir kısmı yer almaktadır. Bu nedenle 1D-CNN ve CRNN modelleri ilişkiyi yakalayamamıştır. Tahminleme çalışmalarında son zamanlarda gerçekleşen değerlerin tahminlemeye etkisinin, geçmişte gerçekleşen değerlerden daha fazla etki ettiği düşünülür. Bu nedenle veriler en yakın tarihten en uzak tarihe doğru ağırlıklandırılarak modeller eğitilebilir. Bu ağırlıklandırma sayesinde her iki modelin ilişki yakalama gücü artabilir.

Başka bir seçenekte veri sayısını arttırmak olabilir. BDI verisi sadece kriz anlarında ele alındığı için veri sayısı 2 – 3 yıllık günlük veriyi içermektedir. Eğitim verisinin büyüklüğü ve karakteristiği YSA modeller için en önemli etkenlerden biridir. Sadece kriz anlarındaki veriler yerine kriz anlarına kadar olan verilerin tamamı kullanılabilir. Bu durumda daha sonra yaşanan krizler tahminlenirken önceki krizlerin verisiyle beslenen bir model daha iyi sonuç verebilir. Örnek olarak Covid-19 pandemisi ve öncesi verileri ile modeller eğitilip daha küçük bir veri bölünmesiyle, kriz anının tahminlemesi yapılabilir. Böylece 2008 ekonomik krizindeki davranışlarda eğitim verisi içerisinde olacaktır.

Hirata ve Matsuda(2022) çalışmalarında Shangay Konteyner Navlun Endeks'inin LSTM ile tahminlenmesinde aktivasyon fonksiyonu olarak Tanh kullanmışlardır. Ulaştıkları RMSE değerleri sıfırın oldukça üstündedir. Bu çalışmanın ulaştığı RMSE değerleri ise oldukça küçüktür. LSTM tahminlenmesi için ReLu aktivasyon fonksiyonu daha iyi bir tercih olabilir. Bölüm 1.3'te bahsedildiği üzere birçok aktivasyon fonksiyonu bulunmaktadır. Veri karakteristiğine uygun olarak başka aktivasyon fonksiyonları modeller içerisindeki katmanlara uygulanarak daha düşük değer kaybı sağlanabilir. Buna ek olarak bütün modellerde ADAM optimizasyonu yapılmıştır. Başka optimizasyon algoritmaları ile modellerden daha iyi RMSE değerleri bulunmaya çalışılabilir.

Verilerin girdi şeklide YSA'lar ile tahminleme yaparken önemli bir etmendir. Tüm modellerde geriye bakma(lookback) değeri 5 seçilmiştir. Her 5 veriyle 6. veri arasındaki ilişkinin iterasyonlarla araştırılması anlamına gelmektedir. Geriye bakma değerlerini azaltarak veya daha büyük sayılarda tutarak daha fazla ilişki yakalanabilir.

Tarihsel verilerin yanında, aynı dönem içerisindeki başka verilerde eğitime katılarak nedensellikler araştırılabilir. Örneğin varil petrol fiyatları, Dolar/Euro endeksi gibi değişkenler ile BDI fiyatları arasındaki regresyon YSA tahminlerini güçlendirebilir.

YSA'larla çalışılırken birçok model kombinasyonu kullanılarak katman sayısı arttırılarak daha etkili derin ögrenme yapılabilir. Başka YSA modelleri birleştirilerek kurulacak yeni modeller aynı veri setiyle eğitilerek sonuçları karşılaştırılabilir.

YSA birçok parametreyi bünyesinde barındıran bir yapıdır. Daha iyi bir modeli kurmanın anahtarı daha iyi parametreler bulmanın yanında veri setine de bağlıdır Bunu her iki kriz verisi için farklı modellerin daha iyi sonuç vermesinden anlayabiliriz.

BDI'nın iki büyük global kriz zamanında tahminlemesi için, çeşitli parametreleri değiştirilerek bulunan en iyi üç modelin performansı incelenmiş ve önceki çalışmalarla karşılaştırılmıştır. Karşılaştırmalar sonucu YSA'ların kriz zamanlarında BDI tahminleme için iyi bir araç olduğu saptanmıştır. Volatilitenin daha az olduğu ve lojistik faaliyetleri daha kısa süreli etkilediği Covid-19 krizi için 1D-CNN modeli en iyi performansı vermiştir. Volatilitenin aşırı fazla olduğu ve veri seti içerisinde belirgin karakteristikler bulunan (çok ani düşüş-yükseliş vb.) 2008 ekonomik krizinde LSTM veriler arasındaki ilişkiyi daha etkili yakalamıştır.

Gelecekteki çalışmalarda farklı model parametreleri kullanılarak ve yeni bağımsız değişkenler eklenerek nasıl daha iyi sonuç elde edileceği araştırılabilir. Çalışmada kullanılan modellerle, Baltık Borsası'nın yayınladığı alt endeksler ve diğer kuruluşların yayınladığı taşımacılık endeksleri, yerel ve global kriz anlarında tahminlenip performansları ölçülebilir.

KAYNAKLAR

Abbas, O. (2005). Neural networks in business forecasting. *International Journal of Computer* (IJC), 19(1), 114-128.

Abraham, A. (2005). Artificial neural networks. *Handbook of measuring system design.*

Abraham, B., & Ledolter, J. (1983). *Statistical methods for forecasting (Vol. 179)*. New York: Wiley.

Bakır, H., Oktay, S., & Tabaru, E. (2023). Detectıon Of Pneumonıa From X-Ray Images Usıng Deep Learnıng Technıques. *Journal of Scientific Reports-A*, (052), 419-440. https://doi.org/10.59313/jsr-a.1219363

Batchelor, R., Alizadeh, A., & Visvikis, I. (2007). Forecasting spot and forward prices in the international freight market. *International Journal of Forecasting*, 23(1), 101-114. https://doi.org/10.1016/j.ijforecast.2006.07.004

Bilgili, F. (2001). Arıma Ve Var Modellerinin Tahmin Başarılarının Karşılaştırılması. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, (17), 37-53.

Chang, W. Y. (2014). A literature review of wind forecasting methods. *Journal of Power and Energy Engineering*, 2(04), 161. http://dx.doi.org/10.4236/jpee.2014.24023

Cowpertwait, P. S., & Metcalfe, A. V. (2009). *Introductory time series with R*. Springer Science & Business Media.

Crainic, T. G., & Laporte, G. (1997). Planning models for freight transportation. *European journal of operational research*, 97(3), 409-438. https://doi.org/10.1016/S0377-2217(96)00298-6

Cullinane, K P B; Mason, K J; Cape, M (1999). A Comparison of Models for Forecasting the Baltic Freight Index: Box-Jenkins Revisited. *International journal of maritime economics*, 1(2), 15–39. https://doi.org/10.1016/S0377-2217(96)00298-6

Cullinane, Kevin (1992). A short-term adaptive forecasting model for BIFFEX speculation: a Box—Jenkins approach. *Maritime Policy & Management*, 19(2), 91–114. https://doi.org/10.1080/03088839200000018

Çiğdem, A. C. I., & Çırak, A. (2019). Türkçe haber metinlerinin konvolüsyonel sinir ağları ve Word2Vec kullanılarak sınıflandırılması. *Bilişim Teknolojileri Dergisi*, 12(3), 219-228. https://doi.org/10.17671/gazibtd.457917

Doğan, F., & Türkoğlu, İ. (2019). Derin öğrenme modelleri ve uygulama alanlarına ilişkin bir derleme. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 10(2), 409-445. https://doi.org/10.24012/dumf.411130

Dumitru, C., & Maria, V. (2013). Advantages and Disadvantages of Using Neural Networks for Predictions. *Ovidius University Annals, Series Economic Sciences*, 13(1).

Duru, Ö. (2007). *Zaman serileri analizinde ARIMA modelleri ve bir uygulama.* [Yayımlanmamış Yüksek Lisans Tezi], İstanbul Üniversitesi.

E ElSaid, A., Wild, B., Higgins, J., & Desell, T. (2016, October). Using LSTM recurrent neural networks to predict excess vibration events in aircraft engines. *2016 IEEE 12th International Conference on e-Science (e-Science)* (pp. 260-269). IEEE. https://doi.org/10.1109/eScience.2016.7870907

Eren, L., Ince, T., & Kiranyaz, S. (2019). A generic intelligent bearing fault diagnosis system using compact adaptive 1D CNN classifier. *Journal of Signal Processing Systems*, 91, 179-189. http://dx.doi.org/10.1007/s11265-018-1378-3

Fan, Yong Hui; Xing, Yu Wei; Yang, Hua Long (2014). Prediction of Baltic Capesize Freight Index Based on GARCH Model. *Applied Mechanics and Materials*, 488-489(), 1494–1497. https://doi.org/10.4028/www.scientific.net/AMM.488-489.1494

Fusion Media Limited (2022). Baltıc Dry Index. https://www.investing.com/indices/baltic-dry-historical-data?utm_source=google&utm_medium=cpc&utm_campaign=16891691908&utm_content=592966459613&utm_term=aud-1652419780036:dsa-1518303930164_&GL_Ad_ID=592966459613&GL_Campaign_ID=16891691908&gclid=Cj0KCQjwhLKUBhDiARIsAMaTLnGwhmWCgM-b8kxmKO4AeT-LZrlsTedN4nuW_5MbIXMHNKSXpOuT7K4aAuiVEALw_wcB adresinden 10 Eylül 2023 tarihinde alınmıştır .

Geman, H., & Smith, W. O. (2012). Shipping markets and freight rates: an analysis of the Baltic Dry Index. *Journal of Alternative Investments*, 15(1), 98-109. https://doi.org/10.3905/jai.2012.15.1.098

Hagan, M. T., Demuth, H. B., & Beale, M. (1997). *Neural network desig*n. PWS Publishing Co.

Hamzaçebi, C., Akay, D., & Kutay, F. (2009). Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. *Expert systems with applications*, 36(2), 3839-3844. https://doi.org/10.1016/j.eswa.2008.02.042

Hirata, E., & Matsuda, T. (2022). Forecasting Shanghai Container Freight Index: A Deep-Learning-Based Model Experiment. *Journal of Marine Science and Engineering*, 10(5), 593. https://doi.org/10.3390/jmse10050593

Hopfield, J. J. (1988). Artificial neural networks. IEEE Circuits and Devices Magazine, 4(5), 3-10. https://doi.org/10.1103/RevModPhys.71.S431

Hsu, C. L., & Ho, T. C. (2021). Evaluating key factors of container shipping lines from the perspective of high-tech industry shippers. *Journal of Marine Science and Technology*, *29*(1), 3. https://doi.org/10.51400/2709-6998.1002

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, 31(3), 685-695. https://doi.org/10.1007/s12525-021-00475-2

Krenker, A., Bešter, J., & Kos, A. (2011). Introduction to the artificial neural networks. *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, 1-18. https://doi.org/10.5772/15751

Li, Jun; Parsons, Michael G. (1997). Forecasting tanker freight rate using neural networks. *Maritime Policy & Management*, 24(1), 9–30. https://doi.org/10.1080/03088839700000053

Lin, F., & Sim, N. C. (2013). Trade, income and the baltic dry index. *European Economic Review*, 59, 1-18. https://doi.org/10.1016/j.euroecorev.2012.12.004

Lyridis, D V; Zacharioudakis, P; Mitrou, P; Mylonas, A (2004). Forecasting Tanker Market Using Artificial Neural Networks. *Maritime Economics & Logistics*, 6(2), 93–108. https://doi.org/10.1057/palgrave.mel.9100097

Marquez, L. O. (1992). *Function approximation using neural networks: A simulation study*. University of Hawai'i at Manoa.

Mathew, A., Amudha, P., & Sivakumari, S. (2021). Deep learning techniques: an overview. *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020*, 599-608. https://doi.org/10.1007/978-981-15-3383-9_54

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

Mijwel, M. M. (2021). Artificial neural networks advantages and disadvantages. *Mesopotamian Journal of Big Data*, *2021*, 29-31. https://doi.org/10.58496/MJBD/2021/006

Munim, Z. H., & Schramm, H. J. (2017). Forecasting container shipping freight rates for the Far East–northern Europe trade lane. *Maritime Economics & Logistics*, 19(1), 106-125. https://doi.org/10.1057/s41278-016-0051-7

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*. https://doi.org/10.48550/arXiv.1511.08458

Priddy, K. L., & Keller, P. E. (2005). *Artificial neural networks: an introduction* (Vol. 68). SPIE press.

Sagheer, A., & Kotb, M. (2019). Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*, 323, 203-213. https://doi.org/10.1016/j.neucom.2018.09.082

Schmidt, R. M. (2019). Recurrent neural networks (rnns): A gentle introduction and overview. *arXiv preprintarXiv:1912.05911*. https://doi.org/10.48550/arXiv.1912.05911

Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Sci*, 6(12), 310-316.

Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2018, December). A comparison of ARIMA and LSTM in forecasting time series. *2018 17th IEEE international conference on machine learning and applications (ICMLA)* (pp. 1394-1401). IEEE.

Sibi, P., Jones, S. A., & Siddarth, P. (2013). Analysis of different activation functions using back propagation neural networks. *Journal of theoretical and applied information technology*, 47(3), 1264-1268.

Smagulova, K., & James, A. P. (2019). A survey on LSTM memristive neural network architectures and applications. *The European Physical Journal Special Topics*, 228(10), 2313-2324. https://doi.org/10.1140/epjst/e2019-900046-x

Soydaner, D. (2020). A comparison of optimization algorithms for deep learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 34(13), 2052013. https://doi.org/10.1142/S0218001420520138

Şahin, B., Gürgen, S., Ünver, B., & Altin, I. (2018). Forecasting the Baltic Dry Index by using an artificial neural network approach. *Turkish Journal of Electrical Engineering & Computer Sciences*, 26(3), 1673-1684. https://doi.org/10.3906/elk-1706-155

Şipal, Y. Z. (2016). Türkiye'de Ticari Deniz Taşımacılığı Ve Gemi Fiyatlarında Arz-Talep Dengesizliği, Navlun Fiyatlarına Yansıması. *Imuco 2016*, 641.

Tealab, A. (2018). Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal*, 3(2), 334-340. https://doi.org/10.1016/j.fcij.2018.10.003

Tsioumas, V., Papadimitriou, S., Smirlis, Y., & Zahran, S. (2017). A Novel Approach to Forecasting the Bulk Freight Market. *Asian Journal of Shipping and Logistics*, *33*(1), 33-41. https://doi.org/10.1016/j.ajsl.2017.03.00

Veenstra, A., & van Dalen, J. (2008). Price indices for ocean charter contracts. *The Second International Index Measures Congress, Washington, Digital proceedings*.

Yang, Z., & Mehmed, E. E. (2019). Artificial neural networks in freight rate forecasting. *Maritime Economics & Logistics*, 21(3), 390-414. http://dx.doi.org/10.1057/s41278-019-00121-x

Zaheer, R., & Shaziya, H. (2019, January). A study of the optimization algorithms in deep learning. *2019 third international conference on inventive systems and control (ICISC)* (pp. 536-539). IEEE.

Zeren, F., & Kahramaner, H. (2019). Baltık Kuru Yük Endeksi İle İstanbul Navlun Endeksi Arasındaki Etkileşimin İncelenmesi: Ekonometrik Bir Uygulama. *Journal of International Management Educational and Economics Perspectives*, 7(1), 68-79.

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (2001). A simulation study of artificial neural networks for nonlinear time-series forecasting. *Computers & Operations Research*, 28(4), 381-396. https://doi.org/10.1016/S0305-0548(99)00123-9

Zhang, X., Xue, T., & Stanley, H. E. (2018). Comparison of econometric models and artificial neural networks algorithms for the prediction of baltic dry index. IEEE Access, 7, 1647-1657. https://doi.org/10.1109/ACCESS.2018.2884877

Zhao, K., & Wang, C. (2017). Sales forecast in e-commerce using convolutional neural network. *arXiv preprint arXiv:1708.07946*. https://doi.org/10.48550/arXiv.1708.07946

EKLER

EK-1 LSTM model Python kodu

```python
from google.colab import drive
drive.mount('/gdrive')
# %cd /gdrive

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
from tensorflow.keras.regularizers import L2
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import plot_model

data = pd.read_csv('/gdrive/My Drive/Colab Notebooks/Baltic Dry Index Historical Data-2008 crisis daily (1).csv')
data['Date'] = pd.to_datetime(data['Date'])
data = data.sort_values('Date')

target_col = 'Value'
train_size = int(len(data) * 0.80)
val_size = int(train_size * 0.2)
train_size = train_size - val_size
test_size = len(data) - train_size - val_size

train_data = data.iloc[:train_size]
val_data = train_data.iloc[train_size - val_size:train_size]
test_data = data.iloc[train_size + val_size:]

scaler = MinMaxScaler()
train_data_scaled = scaler.fit_transform(train_data[[target_col]])
val_data_scaled = scaler.transform(val_data[[target_col]])
test_data_scaled = scaler.transform(test_data[[target_col]])

def create_dataset(data, look_back):
    X, y = [], []
    for i in range(len(data) - look_back):
        X.append(data[i:(i+look_back), 0])
        y.append(data[i + look_back, 0])
    return np.array(X), np.array(y)

look_back = 5
X_train, y_train = create_dataset(train_data_scaled, look_back)
X_val, y_val = create_dataset(val_data_scaled, look_back)
X_test, y_test = create_dataset(test_data_scaled, look_back)

model = Sequential()
model.add(LSTM(50, input_shape=(look_back, 1), kernel_regularizer=L2(0.01)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
early_stopping = EarlyStopping(monitor='val_loss', patience=5, verbose=1)

model.fit(X_train, y_train, epochs=200, batch_size=32,
          validation_data=(X_val, y_val), callbacks=[early_stopping], verbose=1)

y_train_actual = scaler.inverse_transform(y_train.reshape(-1, 1))
test_predictions = model.predict(X_test)
```

```
70
71
72      test_predictions_inscaled = scaler.inverse_transform(test_predictions)
73
74      n = len(y_test)
75      residuals = y_test - test_predictions.flatten()
76
77
78      ssr = np.sum(residuals**2)
79
80
81      num_parameters = model.count_params()
82
83
84      aic = n * np.log(ssr / n) + 2 * num_parameters
85      bic = n * np.log(ssr / n) + num_parameters * np.log(n)
86
87      print(f'AIC: {aic:.2f}')
88      print(f'BIC: {bic:.2f}')
89      test_rmse = np.sqrt(mean_squared_error(y_test, test_predictions))
90      print(f"Test RMSE: {test_rmse}")
91
92      train_dates = train_data['Date'].values[look_back:]
93      test_dates = test_data['Date'].values[look_back:]
94      df_train_pred = pd.DataFrame({'Date': train_dates, 'Predicted': y_train_actual[:, 0]})
95      df_test_pred = pd.DataFrame({'Date': test_dates, 'Predicted': test_predictions[:, 0]})
96      df_test_real = pd.DataFrame({'Date': test_dates, 'Real': test_data[target_col].values[look_back:]})
97
98
99      plt.figure(figsize=(12, 6))
100     plt.plot(train_dates, y_train_actual, label='Eğitim Verisi')
101     plt.plot(test_dates, test_predictions_inscaled, label='Tahmin')
102     plt.plot(test_dates, test_data[target_col].values[look_back:], label='Gerçek Değerler')
103     plt.xlabel('Tarih')
104     plt.ylabel('Değer')
105     plt.title('LSTM Covid-19 Pandemisi Tahminleme Değerleri vs. Gerçek Değerler')
106     plt.legend()
107     plt.xticks(rotation=45)
108     plt.show()
109
110     plot_model(model, show_shapes=True, to_file='/gdrive/My Drive/Colab Notebooks/lstm_model.png')
```

EK-2 1D-CNN model Python kodu

```
from google.colab import drive
drive.mount('/gdrive')
# %cd /gdrive

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense
from tensorflow.keras.regularizers import l2
from tensorflow.keras.callbacks import EarlyStopping
from keras.utils import plot_model
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from matplotlib.dates import date2num, DateFormatter

data = pd.read_csv('/gdrive/My Drive/Colab Notebooks/Baltic Dry Index Historical Data-2008 crisis daily (1).csv')
data['Date'] = pd.to_datetime(data['Date'])
data = data.sort_values('Date')
data.set_index('Date', inplace=True)

target_col = 'Value'
train_size = int(len(data) * 0.80)
val_size = int(train_size * 0.2)
train_size = train_size - val_size
test_size = len(data) - train_size - val_size

train_data = data.iloc[:train_size]
val_data = train_data.iloc[train_size - val_size:train_size]
test_data = data.iloc[train_size + val_size:]
print(len(data),len(val_data),len(train_data),len(test_data))

scaler = MinMaxScaler()
train_data_scaled = scaler.fit_transform(train_data[[target_col]])
val_data_scaled = scaler.transform(val_data[[target_col]])
test_data_scaled = scaler.transform(test_data[[target_col]])


def create_dataset(data, look_back):
    X, y = [], []
    for i in range(len(data) - look_back):
        X.append(data[i:(i+look_back), 0])
        y.append(data[i + look_back, 0])
    return np.array(X), np.array(y)

look_back = 5


X_train, y_train = create_dataset(train_data_scaled, look_back)
X_val, y_val = create_dataset(val_data_scaled, look_back)
X_test, y_test = create_dataset(test_data_scaled, look_back)


X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
X_val = X_val.reshape(X_val.shape[0], X_val.shape[1], 1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)

model = Sequential()
model.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(look_back, 1)))
```

```
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(50, activation='relu', kernel_regularizer=l2(0.01)))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')

early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

model.fit(X_train, y_train, epochs=300, batch_size=16, verbose=1, validation_data=(X_val, y_val), callbacks=[early_stopping])

y_test_pred = model.predict(X_test)

test_rmse = np.sqrt(mean_squared_error(y_test, y_test_pred))


n = len(y_test)
residuals = y_test - y_test_pred.flatten()  # Residuals


ssr = np.sum(residuals**2)

num_parameters = model.count_params()

aic = n * np.log(ssr / n) + 2 * num_parameters
bic = n * np.log(ssr / n) + num_parameters * np.log(n)

print(f'AIC: {aic:.2f}')
print(f'BIC: {bic:.2f}')
print(f"Test RMSE: {test_rmse}")

y_test_pred_actual = scaler.inverse_transform(y_test_pred)


y_train_actual = scaler.inverse_transform(y_train.reshape(-1, 1))
y_test_actual = scaler.inverse_transform(y_test.reshape(-1, 1))


train_dates = train_data.index[look_back:]
test_dates = test_data.index[look_back:]

df_train_real = pd.DataFrame({'Date': train_dates, 'Real': y_train_actual[:, 0]})
df_test_pred = pd.DataFrame({'Date': test_dates, 'Predicted': y_test_pred_actual[:, 0]})
df_test_real = pd.DataFrame({'Date': test_dates, 'Real': y_test_actual[:, 0]})


plt.figure(figsize=(12, 6))
plt.plot(df_train_real['Date'], df_train_real['Real'], label='Eğitim Verisi')
plt.plot(df_test_real['Date'], df_test_real['Real'], label='Gerçek Değerler')
plt.plot(df_test_pred['Date'], df_test_pred['Predicted'], label='Tahminleme Verisi')
plt.xlabel('Tarih')
plt.ylabel('Dolar')
plt.title('1D CNN 2008 Krizi Tahminlemesi')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

plot_model(model, show_shapes=True, to_file='/gdrive/My Drive/Colab Notebooks/1d-CNN5_model.png')
```

EK-3 CRNN model Python kodu

```
from google.colab import drive
drive.mount('/gdrive')
# %cd /gdrive

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.utils import plot_model
from keras.callbacks import TensorBoard
import tensorfllayers import Conv1D, MaxPooling1D, LSTM, Dense, Flatten, Dropout
from keras.optimizers import Adamow as tf
from keras.callbacks import EarlyStopping
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.regularizers import l2
from datetime import datetime, timedelta
from sklearn.preprocessing import MinMaxScaler

data = pd.read_csv('/gdrive/My Drive/Colab Notebooks/Baltic Dry Index Historical Data-covid19 daily (1).csv')
data['Date'] = pd.to_datetime(data['Date'])
data = data.sort_values('Date')

data['Date'] = pd.to_datetime(data['Date'])
data = data.sort_values('Date')
target_col = 'Value'
train_size = int(len(data) * 0.60)
val_size = int(train_size * 0.2)
train_size = train_size - val_size
test_size = len(data) - train_size - val_size

train_data = data.iloc[:train_size]
val_data = train_data.iloc[train_size - val_size:train_size]
test_data = data.iloc[train_size + val_size:]
print(len(data),len(val_data),len(train_data),len(test_data))

scaler = MinMaxScaler()
train_data_scaled = scaler.fit_transform(train_data[[target_col]])
val_data_scaled = scaler.transform(val_data[[target_col]])
test_data_scaled = scaler.transform(test_data[[target_col]])


def create_dataset(data, look_back):
    X, y = [], []
    for i in range(len(data) - look_back):
        X.append(data[i:(i+look_back), 0])
        y.append(data[i + look_back, 0])
    return np.array(X), np.array(y)

look_back = 5
X_train, y_train = create_dataset(train_data_scaled, look_back)
X_val, y_val = create_dataset(val_data_scaled, look_back)
X_test, y_test = create_dataset(test_data_scaled, look_back)

print(len(data),len(val_data),len(train_data),len(test_data))

model = Sequential()
model.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(look_back, 1)))
model.add(MaxPooling1D(pool_size=2))
model.add(LSTM(20, activation='relu', return_sequences=True))
model.add(Flatten())
```

```
model.add(Dense(32, activation='relu', kernel_regularizer=l2(0.01)))
model.add(Dropout(0.5))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')

early_stopping = EarlyStopping(monitor='val_loss', patience=10, verbose=1, restore_best_weights=True)


model.fit(X_train, y_train, epochs=200, batch_size=16,
          validation_data=(X_val, y_val), callbacks=[early_stopping], verbose=1)

X_test_reshaped = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
predictions = model.predict(X_test_reshaped)


n = len(y_test)
residuals = y_test - predictions.flatten()


ssr = np.sum(residuals**2)


num_parameters = model.count_params()


aic = n * np.log(ssr / n) + 2 * num_parameters
bic = n * np.log(ssr / n) + num_parameters * np.log(n)

print(f'AIC: {aic:.2f}')
print(f'BIC: {bic:.2f}')
test_predictions = model.predict(X_test_reshaped)
test_rmse = np.sqrt(mean_squared_error(y_test, test_predictions))
print(f"Test RMSE: {test_rmse:.2f}")

predictions_actual = scaler.inverse_transform(predictions.reshape(-1, 1))
y_train_actual = scaler.inverse_transform(y_train.reshape(-1, 1))


train_dates = train_data['Date'].values[look_back:]
test_dates = test_data['Date'].values[look_back:]
df_train_real = pd.DataFrame({'Date': train_dates, 'Real': y_train_actual[:, 0]})
df_test_pred = pd.DataFrame({'Date': test_dates, 'Predicted': predictions_actual[:, 0]})
df_test_real = pd.DataFrame({'Date': test_dates, 'Real': test_data[target_col].values[look_back:]})


plt.figure(figsize=(12, 6))
plt.plot(df_train_real['Date'], df_train_real['Real'], label='Eğitim Verisi')
plt.plot(df_test_pred['Date'], df_test_pred['Predicted'], label='Tahminleme Sonucu')
plt.plot(df_test_real['Date'], df_test_real['Real'], label='Gerçek Değerler')
plt.xlabel('Tarih')
plt.ylabel('Dolar')
plt.title('CRNN 2008 Krisi Tahminleme Grafiği')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

plot_model(model, show_shapes=True, to_file='/gdrive/My Drive/Colab Notebooks/crnn_model.png')
```

EK-4 Kullanılan verilerin csv dosya linki

https://drive.google.com/file/d/1XhXmIlYdWTMbzqRFPDOICTFMcpLXvEOB/view?usp=drive_link

https://drive.google.com/file/d/1kXZXbL_X-6A09CDb79DrCi22RD_RPC5o/view?usp=drive_link

BENZERLİK RAPORU ÖZET SAYFASI

YAPAY SİNİR AĞLARININ KRİZ ANLARINDA BALTIK KURU YÜK ENDEKSİNİ TAHMİNLEME PERFORMANSI

ORİJİNALLİK RAPORU

%3	%3	%0	%1
BENZERLİK ENDEKSİ	İNTERNET KAYNAKLARI	YAYINLAR	ÖĞRENCİ ÖDEVLERİ

BİRİNCİL KAYNAKLAR

1	dergipark.org.tr İnternet Kaynağı	%1
2	acikbilim.yok.gov.tr İnternet Kaynağı	<%1
3	Submitted to Berlin School of Business and Innovation Öğrenci Ödevi	<%1
4	link.springer.com İnternet Kaynağı	<%1
5	Submitted to Beykent Universitesi Öğrenci Ödevi	<%1
6	www.emerald.com İnternet Kaynağı	<%1
7	export.arxiv.org İnternet Kaynağı	<%1
8	s-space.snu.ac.kr İnternet Kaynağı	<%1

ÖZGEÇMİŞ

Adı ve Soyadı: Mehmet Ongun

Doğum Tarihi:

Cep Telefonu:

E-postası:

Bildiği Yabancı Diller: İngilizce

Öğrenim Durumu :

Derece	**Anabilim Dalı/ Bölüm/Program**	**Üniversite**	**Yıl**
Lisans	Endüstri Mühendisliği	Çukurova Üniversitesi	2014-2019
Y. Lisans	İşletme	Mersin Üniversitesi	2020-2023

Görevler :

Görev Unvanı		**Görev Yeri**	**Yıl**
Software Engineer		Huawei Technologies Co., Ltd.	12.2022-Halen
Software Developer		Kartaca	6.2022-12.2022
Junior Java Developer		Saha Bilgi Teknolojiler	12.2021-6.2022
Ambar Uzman Yardımcısı		Rönesans Holding	07.2021-7.2021
Endüstri Mühendisi		H-Concept Mobilya	9.2020-2.2020