T.C. DOĞUŞ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ZAMAN SERİLERİ ANALİZİ VE DERİN ÖĞRENME MODELLERİ KULLANARAK AMERİKAN DOLARI/TÜRK LİRASI DÖVİZ KURU İÇİN HİBRİD TAHMİN MODELİ

YÜKSEK LİSANS TEZİ

HARUN YAŞAR
201695006

Danışman:
DR. ÖĞR. ÜYESİ ZEYNEP HİLAL KİLİMCİ

İstanbul, 2019

	YÜKSEK LİSANS TEZ SINAV TUTANAĞI	Doküman No	FR.1.26
		Yürürlük Tarihi	1.11.2017
		Revizyon Tarihi	1.11.2017
		Revizyon No	1
		Sayfa	1 / 1

SOSYAL BİLİMLER / FEN BİLİMLERİ ENSTİTÜSÜ

Tarih : 19./11../2019.

Anabilim/Anasanat Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Öğrencinin Adı Soyadı : HARUN YASAR

Öğrenci No : 2016.95.006

Tez Danışmanının Adı Soyadı : ZEYNEP HİLAL KILINÇ

İkinci Tez Danışmanının Adı Soyadı : —

Tezin Başlığı : Zaman Serileri Analizi ve Derin Öğrenme Modelleri Kullanarak Amerikan Dolarının/Türk Lirası Döviz Kuru için Hibrid Tahmin Modeli

Doğuş Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin 32.Maddesi uyarınca yapılan değerlendirmeler sonunda;

☒ **tezin kabul edilmesine** ☐ **tezde düzeltme verilmesine** ☐ **tezin reddedilmesine**

oy birliği / oy çokluğu ile karar verilmiştir.Gereği için arz olunur.

Danışman Üye

Dr. Öğr. Üyesi Zeynep Hilal Kılınç

Üye Prof. Dr. Selim Akyokuş

Üye Dr. Öğr. Üyesi Ramazan DURAK

Üye

Anabilim/Anasanat Dalı Başkanı Onayı:

Dr. Öğr. Üyesi. Yasemin Karpat

(Form No: FR-001, Revizyon Tarihi:30.09.2016, Revizyon No:00)

YEMİN METNİ

Yüksek lisans tezi olarak sunduğum "Zaman Serileri Analizi ve Derin Öğrenme Modelleri Kullanarak Amerikan Doları/Türk Lirası Döviz Kuru için Tahmin Modeli" adlı çalışmanın, tarafımdan, akademik kurallara ve etik değerlere uygun olarak yazıldığını ve yararlandığım eserlerin kaynakçada gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve bunu onurumla doğrularım.

Harun Yaşar

ÖNSÖZ

Öncelikle, tüm zorluklara rağmen desteğini ve yardımlarını esirgemeyen tez danışmanım Dr. Öğr. Üyesi Zeynep Hilal KİLİMCİ'ye, yüksek lisans programına kabul sürecinde ve sonrasında kıymetli vaktini hiç tereddüt etmeden benimle paylaşan Prof. Dr. Selim AKYOKUŞ'a desteklerinden ötürü sonsuz teşekkürler.

Hayatımın büyük bir çoğunluğunda olduğu gibi yüksek lisans süresi boyunca da varlığını ve desteğini benden esirgemeyen hayat arkadaşım, eşim Yeşim Aytekin Yaşar'a şükranlarımı sunarım.

İstanbul, 2019 Harun Yaşar

ÖZET

Borsa tahminlemesinin yanı sıra döviz kuru tahminlemesi de yatırımcılar, araştırmacılar ve analistler için önemli bir çalışma konusu olmuştur. Bu tez kapsamında finansal duygu analizi ve zaman serisi analizi yapılarak döviz kuru yönünü tahminleyen hibrid bir model oluşturulması amaçlanmıştır. Bu amaçla, önerilen hibrid model, metin verilerinin finansal duygu analizi için elde edilmesi ve modellenmesi, sayısal verilerin zaman serisi analizi için elde edilmesi ve modellenmesi ve iki modelin harmanlanması şeklinde üç aşamalı olarak inşa edilmiştir.

Yapılan literatür araştırması ile, sosyal medya platformlarını finansal duygu analizi amacıyla kaynak olarak kullanan ve bunu sayısal veriler kullanarak zaman serisi analizi yöntemleriyle harmanlayan literatürdeki ilk çalışma olduğu düşünülmektedir. Dahası, Amerikan doları/Türk lirası kurunun yönünün tahminlenmesinin hem finansal duygu analizi yaparak hem de hybrid bir model kullanarak gerçekleştiren literatürdeki ilk çalışma niteliğindedir.

Çalışmanın literatüre katkısı beş aşamada özetlenebilir: İlk aşamada, finansal duygu analizini gerçekleştirebilmek için Twitter ortamında toplanan verilerin ayrıştırılması, kelimelerin sözlükteki doğru hallerinin bulunması, kelimelerin köklerinin bulunması, kelimelerin normalizasyonu, kullanılmayan karakterlerin ve kelimelerin temizlenmesi gibi yöntemlerle temizlenip modellenmeye hazır hale getirilmiştir. Modellemeye hazır olan veri kümesi Word2vec, GloVe, fastText gibi kelime yerleştirme yöntemleri ve CNN, RNN, LSTM gibi derin öğrenme modelleriyle hem ayrı ayrı hem de kombinasyonları kullanılarak sınıflandırılmıştır. Bu çalışma, finansal duygu analizinin gerçekleştirilmesinde kullanılan kelime yerleştirme ve derin öğrenme modellerinin kombinasyonlarını elde edilerek analiz edilmesi açısından da anlaşıldığı kadarıyla literatürdeki ilk girişimdir. İkincisi, Amerikan Doları/Türk Lirası döviz kuru gerçek verileriyle Basit üssel yumuşatma, Holt-Winters, Holt's Linear ve ARIMA modelleri kullanılarak zaman serisi analizi gerçekleştirilmiştir. Üçüncü olarak, birbirinden farklı yapıda olan iki tahmin modelinden alınan sonuçların döviz kuru yönüne olan etkisi gösterilmiştir. Dördüncüsü ise, önerilen yaklaşımın performansını kanıtlamak amacıyla, finansal duygu analizi için 01 Ocak 2018 ile 31 Aralık 2018 zaman aralığındaki Türkçe ve İngilizce Twitter veri kümeleri, zaman serisi analizi için ise, yine aynı zaman aralığındaki gerçek kur verileri kullanılmıştır. Sonuç olarak, önerilen modelin

performansı literatür çalışmalarıyla kıyaslandığında kayda değer ölçüde üstünlük göstermektedir.

Anahtar Kelimeler: Derin öğrenme modelleri, döviz kuru tahmini, finansal duygu analizi, zaman serisi analizi.

ABSTRACT

Exchange rate forecasting, as well as stock market forecasting, has been an important topic for investors, researchers and analysts. In this study, it is aimed to create a hybrid model that predicts the exchange rate trend by performing financial emotion analysis and time series analysis. For this purpose, the proposed hybrid model was constructed in three stages: obtaining and modelling text data for financial sentiment analysis, obtaining and modelling numerical data for time series analysis, and blending two models.

It is thought that this is the first study in the literature that uses social media platforms as a source for financial emotion analysis and blends it with time series analysis methods using numerical data. Moreover, it is the first study in the literature that performs the US Dollar/Turkish Lira exchange rate prediction of the trend by performing a financial sentiment analysis and using a hybrid model.

The contribution of the study to the literature can be summarized in five sections: In the section one, in order to perform financial sentiment analysis, the data collected from Twitter has been prepared and modelled by using a couple of pre-processing stages such as parsing the documents, lemmatization, stemming, removing the unused characters and the words, normalizing the words. The data set, which is ready for modelling, is classified using word embedding methods such as Word2vec, GloVe, fastText, and deep learning models such as CNN, RNN, LSTM, both individually and in combination. This study is the first attempt as it is understood from the literature in terms of performing the financial sentiment analysis by using the combinations of deep learning models with word embedding methods. In the second section, time series analysis was performed by using Simple Exponential Smoothing, Holt-Winters, Holt's Linear, and ARIMA models along with real US Dollar/Turkish Lira exchange rates. In the third section, two different prediction models were combined with a weighted majority algorithm to form a hybrid model. In the fourth section, in order to prove the performance of the proposed model, the real Twitter and the exchange rate data sets from January 1, 2018 to December 12, 2018 were used to achieve financial sentiment analysis and time series analysis respectively. In the fifth section, as a result, the performance of the proposed model is significantly superior to that of the literature.

Keywords: Deep learning model, exchange rate prediction, financial sentiment analysis, time series analysis.

İÇİNDEKİLER

Sayfa No.

TABLO LİSTESİ

Sayfa No.

ŞEKİL LİSTESİ

Sayfa No.

KISALTMALAR

ACF : Autocorrelation Function
AR : Autoregression
API : Application Programming Interface
ARIMA : Autoregressive Integrated Moving Average
ARMA : Autoregressive–Moving-Average Model
CBOW : Continuous Bag of Words
CNN : Convolutional Neural Network
CPU : Central Processing Unit
GPU : Graphics Processing Unit
HTML : Hypertext Markup Language
I : Integration
IoT : Internet of Things
LDA : Latent Dirichlet Allocation
LSA : Latent Semantic Analysis
LSTM : Long Short-Term Memory
MA : Moving Average
NLTK : Natural Language Toolkit
NoSQL : Not only SQL
PACF : Partial Autocorrelation Function
RNN : Recurrent Neural Network
SES : Simple Exponential Smoothing
TF-IDF : Term Frequency–Inverse Document Frequency
XML : Extensible Markup Language
VAR : Vector Autoregression

1. GİRİŞ

Günümüzde sosyal ağlar günlük hayatımızın içerisinde önemli bir yer edindi ve insanlar yani kullanıcılar artık boş vakitlerini Twitter, Facebook ya da Instagram gibi sosyal platformlar kullanarak değerlendirmekte. Bu alışkanlıklar her ne kadar olumsuz olarak görünse de milyonların sosyal medya araçlarını kullanması araştırmacıların en önemli araştırma alanı ve veri kaynağı haline dönüşmesi göz ardı edilemez. Bunun en büyük sebebi ise verinin bu platformlarda resim, yazı, video gibi farklı çeşitlerde, çok hızlı bir şekilde milyonlarca kişi tarafından büyük hacimlerde aynı anda üretilmesi denilebilir. Twitter'ı diğer platformlardan ayıran özelliği ise yazı tabanlı verinin büyüklüğü ve günlük 500 milyondan fazla tweetin gönderiliyor olmasıdır. Buna ek olarak her gün her saniyede ortalama 6000 tweetin gönderilmesi, her ay yaklaşık 326 milyon kullanıcının aktif olması Twitter'ı dünya çapında en popüler sosyal platformlardan biri haline getiriyor. Bu da, bu kaynağın her an taze, büyük ve çeşitli kalmasını sağlamakta. Diğer bir yanı ise sadece kullanıcıların kendi ürettikleri içerik ile değil farklı kullanıcılar ile etkileşim halinde olması üretilen verinin çeşitliliği ile Twitter'ı gerçek bir veri kaynağı olarak tercih edilebilir kılıyor. Rakiplerine, benzerlerine ya da diğer platformlara nazaran araştırmacılara ya da meraklılarına farklı API'ler ile kolay veri erişim yolları sağlaması ise en belirgin özelliği olarak kabul edilebilir. Dahası kullanıcıların olaylara Twitter'ın sunmuş olduğu hashtag, mention, trend topic gibi özellikleri kullanarak anlık etkileşimler sunuyor olması da markalar, araştırmacılar ve meraklıları için tercih edilebilir bir platform haline geliyor. Özellikle markaların pazar beklentileri, pazarın rekabetçi yapısı nedeniyle ve firmaların fark yaratma çabası birleştiğinde, markaların kullanıcı odaklı olmaları ve müşteri geribildirimlerinin izlenebiliyor olması önemli bir hale geliyor. Pazarı yakından takip etme çabasında olan bu markalar ise gerek kendi müşterilerini gerekse potansiyel müşterilerini takip etme eğiliminde oluyorlar. Bu nedenledir ki kendine değer katma eğiliminde olan bu markalar Twitter'ı bir veri kaynağı ve bir avantaj faktörü olarak görerek verinin nasıl analiz edileceğinin uzun süre yollarını aradılar (Okazaki & Matsuo, 2009; Pang & Lee, 2008).

Twitter, ticari faaliyette bulunan kuruluşlar için geniş kitlelere ulaşmalarını sağlayan farklı ve ücretli özellikler sunarak platformun gerçek bir veri kaynağı olarak kullanılmasının önünü açmış oldu. İnsanlar yani kullanıcılar bir konu hakkındaki yorumlarımızı daha çok pozitif ve negatif dengeler üzerinde sınıflandırdığımız için

yorumların bu yönünün analizini yapan duygu analizi ve diğer adı ile de bilinen görüş madenciliği önem kazanmış oldu. Gönderilen olumlu tweetlerin hızla yayılması ve binlerce insan tarafından görülmesi ne kadar iyi ise olumsuz bir yorumun aniden tüm Twitter'da yayılması da bir o kadar önemlidir. Bu sebeple Twitter'ın izlenmesi, mevcut müşterilerin ya da potansiyel müşterilerin takip edilmesi hayati bir önem kazanmıştır. Farklı araştırma metotları ve özellikle doğal dil işleme yöntemleri ile duygu analizi yapıp etkileşimlerin değerini artık ölçebiliyoruz. Twitter'ın izlenmesi, şirketlerin kitleleri anlamalarını, rakipleri hakkında görüş elde etmelerini ve aktif oldukları sektörler hakkında trendleri takip etmelerini sağlamakta. Sadece bu özellikler ile sınırlı değil elbette; gündemin takip edilmesi, haber kanyağı olarak kullanılması ve hatta bu araştırmaya konu olan döviz kuru takibinin yapılması dahi mümkün bir hale gelebiliyor. Bu araştırmada ise Twitter kullanıcılarının gündeme ve gündemin etki ettiği döviz kurlarına dair etkileşimlerinin ölçülmesi ve duygu analizinin yapılması olarak ifade edilebilir.

Döviz kurları, sadece dış dünya ile yakından iletişim halinde olan şirketler için değil aynı zamanda herhangi bir ülke için en önemli yatırım araçlarından biri olarak kabul edilebilir. Ülkeler ve çok uluslu firmalar dış dünya ile olan bağlantılarını sağlamak için en önemli iktisadi değişkenlerden biri olan döviz kuru değişkenini kullanmakta ve bu durum döviz kurlarını ve döviz piyasasını dünyanın en büyük finansal piyasalarından biri haline getirmektedir.

Bu sebepledir ki döviz kurları piyasalarda ve ekonomide oluşabilecek bir çok gelişmeden olumlu ya da olumsuz ve hızlı bir şekilde etkilenebilir. Bu dış faktörler göz önüne alındığında döviz kurlarının ve piyasanın gelecekteki seviyesini kontrol etmek neredeyse imkansızlaşır.

Tahmin edilemezlik gerçeğine rağmen, döviz tahmininin doğru bir şekilde yapılması yatırımcılar ve işletmeler için önemli bir hale gelmiştir. Daha önce de bahsedildiği gibi döviz kurları, göz önünde bulundurulması gereken çeşitli dış etkenlerden çok kolay bir şekilde etkilendiğinden ve genellikle karmaşık ve değişken yapısı nedeniyle döviz tahminlemesinde yüksek doğruluklar ile sonuçlar almayı güçleştirmektedir.

Doğrusal olmayan yapısı döviz kuru tahminlemesini çekici ve çok aktif bir araştırma alanı olmaya itmiştir. Her ne kadar birçok zaman serisi analizi ve makine öğrenmesi yöntemleri ile farklı sonuçlar elde edilse de oluşturulan modellerin zamanla daha karmaşık hale gelmesi ve döviz kurlarının değişimin ardında asıl faktörlerin doğru analiz edilemeyişi bir çok yanlış algıyı da beraberinde getirmektedir.

Bu dinamik, doğrusal olmayan ve hızlı bir değişkenlik içerisinde olan döviz kuru piyasasının etkilendiği dış etkenlerin yorumlanması kısmında birçok makine öğrenmesi algoritması ile duygu analizinden faydalanılmaktadır. Buna rağmen doğru bir döviz kuru tahmini yapmak oldukça güçtür. Fakat biz burada, teknolojinin gelişmesiyle birlikte yeni ve güvenilir derin öğrenme modelleri ile biraz daha doğru tahminler yapabilmeyi amaçlıyoruz.

Her ne kadar piyasanın etkisini ve değişimini önceden tahmin etmek pek mümkün olmasa da klasik finansal yöntemlerin dışında zaman serileri analizi, makine öğrenmesi ve derin öğrenme gibi farklı yöntemler deneyerek kesin sonuçlar elde edilmese de mevcut koşullar ışığında çeşitli kaynaklardan yararlanarak mevcut veriler üzerinde yeni yöntemler denenebilir.

Teknolojinin büyük bir ivme ile gelişmesiyle birlikte birçok alanda olduğu gibi özellikle bilgisayar bilimleri alanında zamana ayak uyduran hatta zamanın ötesine geçen gelişmeleri gözlemlemek olağan hale geldi. Bilgisayar bileşenlerine kolay erişim, bulut teknolojisinin varlığı, firmaların bu ilerlemeyi bir avantaj haline getirip yeni teknolojiler üzerine yaptığı çalışmalar mevcut ivmenin katlanarak büyümesine yardımcı olmakta. Bu gelişmelerin son on yılda makine öğrenmesi ve derin öğrenme gibi disiplinlerin gelişiminde büyük rol oynadığı gerçeği yadsınamaz. Yıllar geçtikçe önemini arttıran makine öğrenmesi ve derin öğrenme disiplinlerinin etki ettiği farklı yöntemlerin günümüz ihtiyaçlarına çözümler sunması farklı araştırmaların da önüne açmakta.

Yüksek doğruluğa sahip makine öğrenmesi ve derin öğrenme algoritmalarının yanında bulut teknolojilerinin gelişimi büyük veri analizinin kolaylaşmasına yol açmış son zamanlarda popülaritesini arttırmakta olan duygu analizinden yüksek derecede faydalanılmasına neden olmuştur. Duygu analizi mevcut kaynakta bulunan öznel bilgilerin ve görüşlerin bir kapsam içerisinde incelenmesidir. Duygu analizi amacıyla yapılan araştırmalarda doğal dil işleme yöntemlerinden faydalanıldığı gibi makine

öğrenmesi, derin öğrenme ve sınıflandırma algoritmalarından da faydalanılmaktadır. Bir insanın kaynakları dokümanları inceleyerek kendi başına uzun süre zaman harcayarak yapabileceği bu türdeki analizleri belirttiğimiz yöntemler vasıtası ile saniyeler içerisinde yapılabilir. Biz ise bu araştırmada bir duygu analizi yapılabilmesi için gerekli olan aşamaları, ön işleme sürecinde kaynak verilerin ayrıştırılması, kelimelerin sözlükteki doğru hallerinin bulunması, kelimelerin köklerinin bulunması, kelimelerin normalizasyonu, kullanılmayan karakterlerin ve kelimelerin temizlenmesi gibi yöntemleri kullanarak belirleyeceğiz. Twitter verisi ile çalışacağımız bu araştırmada verinin içerisinde bulunan platforma özel olan etiket (hashtag) ve adını anma (mention) belirteçlerinden de arındırmış olacağız. Türkçe veriler için Zemberek projesinden ve İngilizce metinler için ise TextBlob kütüphanesinin ön işlemeler kabiliyetlerinden faydalanacağız. Bu aşamaların ardından verinin duygu analizi için hazır olması beklenecek ve uygun olan yöntemlere başvurulacak. Bu noktada TextBlob kütüphanesinin sağlamış olduğu önceden eğitilmiş olan metin sınıflandırma yöntemini, Türkçe metinler için ise önceden etiketlenmiş Kaggle veri kümesini Naif Bayes makine öğrenmesi sınıflandırıcısını TextBlob ile eğiterek kullanacağız. Böyle yaparak Twitter'dan toplanan veri kümesi İngilizce ve Türkçe olarak ayırılarak positif ve negatif olmak üzere iki farklı sınıf olarak etiketlenmiş olacak.

Bir önceki adımda uygulanan yöntemler ile temizlenen dokümanlar kelimelere ayrılıp bir vektör uzaya yerleştirilerek ayrıştırılan kelimelerin bu vektör uzayındaki bağlamsal benzerlikleri kelime yerleştirme yöntemleri kullanılarak hesaplanacak. Kelime yerleştirme, benzer anlamları olan kelimelerin benzer bir gösterime sahip olmasını sağlamak amacıyla uygulanan bir tür hesaplama yöntemidir. Doğal dil işleme yöntemleri ile karşılaşılan sorunları dokümanları bir temsil yolu ile ifade ederek vektör uzayındaki gerçek sayılar ile eşleştirilerek temsil edilir. Neredeyse 20 yıllık bir geçmişi olan bu yöntem ilk defa Bengio vd., (2003) tarafından yayınlanmıştır. Günümüzde en popüler kelime yerleştirme yöntemleri Word2vec, GloVe ve fastText yöntemleridir.

Word2vec hesaplama açısından oldukça elverişli iki katmanlı bir yapay sinir ağı tahmin modelidir. Uygulama detayında ise CBOW ve skip gram modellerini uygular. Vektör uzayındaki kelimelerin birbirlerine olan uzaklık ve yakınlık değerleri ile kapsam olarak benzerliklerini hesaplar.

GloVe ise bir tahmin modelinden daha çok bir kelimenin bir bağlamda ne sıklıkla göründüğünü hesaplayan bir kelime yerleştirme yöntemidir. Fakat GloVe herhangi bir kelimenin kullanılma sıklığını her bir doküman için hesapladığından bu hesabı tüm dokümanlara uygulayan farklı bir yönteme de ihtiyaç duyar. Genelde bunu TF-IDF yöntemini kullanarak hesaplamak gerekir (Ramos, 2003). Çünkü TF-IDF her ne kadar GloVe gibi bir hesaplama yöntemi olsa da sadece bir kelimenin tüm dokümanlarda kaç kere ortaya çıktığını hesaplar. Bu hesaplama ile bir kelimenin bir kitaplık içindeki bir doküman için ne kadar önemli olduğunu tespit edebilmemizi sağlar.

fastText de Word2vec ve GloVe gibi kelimelerin ve kelime öbeklerinin sınıflandırılması amacıyla kullanılan ve Word2vec modelinin uzantısı olan bir kelime yerleştirme kütüphanesidir. Öte yandan fastText, her bir kelimenin yanı sıra her bir tam kelimenin n-gram vektörlerini de denetimli ve denetimsiz olarak sağlar. Böyle yaparak kelimeler için vektör uzayları yaratmak yerine, her bir kelimeyi n-gram karakter olarak gösterir.

Bu üç farklı yöntemin asıl kullanım amacı ise kelimelerin sayısal gösterimlerinin oluşturularak makine öğrenmesi algoritmalarına birer girdi olarak verebilmektir. Kelimelerin ve kelime öbeklerinin sayısal bir şekilde temsil edilmesi işlemi önemli ve gereklidir çünkü çoğu makine öğrenmesi ve derin öğrenme algoritmaları girdileri sayısal değerler olarak beklemektedir.

Son yıllarda sıklıkla duyduğumuz makine öğrenmesi ve derin öğrenme kavramları yapay zeka uygulamaları ile birlikte hayatımızın birçok noktasına tesir etmekte (Mohapatra, 2019). Yapay zeka, makinelerin veya programların tipik olarak insan zekası gerektiren işleri yapabildiği hali olmakla birlikte makinelerin deneyimlerden öğrenmesini, yeni girdilere uyum sağlamasını ve insan benzeri işler yapmasını insan müdahalesine gerek olmadan mümkün kılar.

Derin öğrenme, makine öğrenmesi ve onun özel bir disiplini olup yapay sinir ağları ile beynin yapısını, işlevini ve öğrenme şeklini taklit eden yöntemlerle ilgili bir makine öğrenmesi alt alanıdır. Zaman zaman derin öğrenme ve makine öğrenmesi terimleri birbirlerinin yerine kullanılıyor olsa da her ikisi de birbirinden farklı kavramlardır. Derin öğrenme, örnekler ile öğrenme şeklinde bir yapıya sahip olduğundan büyük miktarda etiketlenmiş veriye ihtiyaç duyar. Derin öğrenme algoritmalarının birçoğu yapay sinir

ağları mimarilerini kullanması nedeniyle oluşturulan modellere genellikle derin sinir ağları denilmektedir.

İlerleyen teknoloji ile birlikte donanım gücünün artması derin öğrenme modellerinin popülaritesini arttırmıştır. Doğruluğu yüksek modeller elde etmek için artık CPU gücünün yanında GPU gücünden de faydalanabiliyor olmak makine öğrenmesinin bir alt kümesi olan derin öğrenme disiplinin popülerleşmesinde önemli bir etkisi olmuştur. Donanımsal avantajlarının yanı sıra kolay ölçeklenebilirliğine de ek olarak, derin öğrenme modellerinin sıkça bahsettiği bir başka yarar da, özellik öğrenimi olarak da adlandırılan ham verilerden otomatik özellik çıkarma işlemini gerçekleştirebiliyor olmalarıdır.

Tez kapsamında ise derin öğrenme algoritmalarının bu özelliklerinden yararlanarak CNN, RNN ve LSTM gibi popüler derin öğrenme modellerini kullanarak Twitter'dan toplanmış ve daha sonra pozitif/negatif olarak etiketlenmiş olan veri üzerinde kelime yerleştirme modellerinden elde ettiğimiz kelime vektör uzaylarını girdi olarak kullanıp doğruluğu ve performansı arttırılmış duygu analizi yapabilen modelleri oluşturmayı amaçlıyoruz.

Günümüzde çalışmalar gösteriyor ki nesnelerin interneti (IoT) büyük bir ivme ile gelişmekte ve önemli bir hal almakta. Nesnelerin interneti cihazlarının sayıca yükselişi ile akıllı evlerin, akıllı iş yerlerinin ve akıllı şehirlerin varlığı zaman sırasına dayalı çok büyük ve hızlı veri üretilmesine neden olmakta. Buna ek olarak sürekli izleme ve veri toplama yaygın hale geldikçe hem istatistiki hem de makine öğrenmesi teknikleri ile zaman serileri analizinin kolay bir şekilde yapılmasını mümkün hale getiriyor. Geçmişin analiz edilmesi, güncelin gözlenmesi ve geleceğin tahmin edilebilmesi amacı ile sorulan "geçmiş geleceği nasıl etkiliyor" sorusu zaman serileri analizinin önemine önem katmaktadır.

Zaman serileri analizi, zaman sıralı gelen düzenli verilerden anlamlı bir bilgi çıkarımı yapmak amacıyla özellikle kurumsal iş ölçümlerini takip etmek, endüstriyel süreçleri izlemek, bütçeleme yapmak, iş tahmini yapmak ya da finansal bir analiz yapmak amacıyla istatistiki analiz modellerinin uygulanmasıdır. Böyle yaparak geçmiş analiz edilebilir, güncel izlenebilir ve gelecek hakkında fikir sahibi olunabilir. Özellikle öngörü modelleri ile iş tahmini yapmak isteyen firmalar zaman serileri analizine büyük önem

vermektedir. Örneğin bir doğal gaz servis sağlayıcısı mevcut müşterilerine daha iyi ve kaliteli hizmet vermek adına müşterilerinin geçmiş kullanım detaylarını analiz ederek gelecek yıllar için daha güvenilir metre küp kullanım tahminleri ile satış tahminleri yapabilir. Böylece kullanıcının eski kullanım alışkanlıklar zaman serileri üzerinden analiz edilirken gelecek için de bir tahmin modeli oluşturulabilir.

Bir zaman serisi, art arda gelen sayısal veri noktalarının bir dizisi olduğundan ve zaman içinde değişen herhangi bir değişken olarak alınabildiğinden gelecekteki hareketliliği tahmin etmek için geçmiş değerler ve bunlarla ilişkili modeller hakkındaki bilgileri kullanarak bir zaman serisi tahmini yapılabilir. Böylece geçmişi iyi bir şekilde anlayıp geleceği tahmin etmek için bir avantaj elde edilebilir.

Bu tez çalışmasında ise belirli bir döviz kuru üzerinden bir yıllık süre boyunca açılış ve kapanış fiyatları üzerinden bir dizi analiz gerçekleştirilecek. 01 Ocak 2018 ile 31 Aralık 2018 tarihleri arasında gerçekleşen tüm açılış ve kapanış fiyatlarının bir listesi Merkez Bankası kaynağı kullanılarak alınacak ve bir zaman sırası ile listelenecek. Bu, döviz kuru için bir yıllık günlük kapanış ve satış fiyatı zaman serisi olacaktır. Böylece bir finansal analiz yapmak için belirli bir dönemi kullanarak döviz kuru üzerinden fiyat tahmin edebilmek için tarihsel verileri kullanacağız.

Basit üssel yumuşatma, Holt-Winters, Holt's Linear ve ARIMA modelleri kullanılarak zaman serisi üzerindeki veri noktalarında bir eğilime yol açan temel faktörlerin neler olduğunu anlamaya çalışacağız ve ileriye dönük bir döviz kuru tahmini yapmaya çalışacağız.

1.1. Tezin Amacı

Uzun yıllar boyunca yeterli bilgisayar gücüne sahip olamadığımızdan sahip olduğumuz verileri beklendiği kadar iyi işleyemiyorduk. Günümüzde ise borsa tahmini ve döviz kuru tahmini, bilgisayar gücünün gelişimi ile birlikte eskiye nazaran daha tahmin edilebilir bir seviyeye gelmiştir. Her ne kadar verilerin işleme hızı artıp insanların yapabildiğinden daha karmaşık analizleri bilgisayarla yaptırabiliyor olsak da zaman zaman bilgisayarların dahi kolaylıkla üstesinden gelemediği zorluklar hala var olmakta.

Bu sebeple zaman serisi tahmini, borsa tahmini, döviz kuru tahmini, hava durumu tahmini ve benzer birçok alanda uygulanan, çok yeni ve popüler bir alandır. Zaman serisi analizi bir makine öğrenme alt dalı olarak nitelendiriliyor olsa da verilerin dinamik ve geçici yapısı sadece istatistiki yöntemler ile tahminleme yapmakta yetersiz kalmaktadır. İstatistiki hesaplamaların yanında makine öğrenmesi, derin öğrenme ve hatta yapay sinir ağları gibi disiplinlerden yardımlar almak oluşabilecek herhangi bir engelin çözümlenmesi adına büyük bir katkısı olacağı aşikardır.

Bu tez çalışmasında ise amaç, giderek popüler bir hale gelen zaman serileri analizi ile geleceğin ön görülmesi yaklaşımına bir yenisini eklemek. Özellikle geçmişi analiz ederek farklı yaklaşımların ortak bir uygulaması ile döviz kuru üzerinden bir gelecek tahminlemesinin doğruluğu araştırılmaktadır.

1.2. Tezin Kapsamı

Zaman serileri analiziyle, kelime yerleştirme yöntemlerini birer girdi olarak kullanan derin öğrenme modellerinin kombinasyonlarından elde edilen yeni bir aday modelin performansının araştırıldığı bu çalışmada, finansal duygu analizi ve derin öğrenme metotlarının birlikte çalışabilme olasılıkları incelenmektedir. Tezin gerçek amacına ek olarak kullanılan yöntemlerin münferit olarak performansları da değerlendirilecek.

Tez kapsamında, giriş bölümünü izleyen ikinci bölümde finansal duygu analizi için gerekli olan verinin Twitter sosyal medya aracılığı ile toplanıp analiz için hazırlanması aşamalarını kapsamaktadır. Twitter'dan toplanan kullanıcı verilerinin üzerinde doğru ve güvenilir bir finansal duygu analizinin çalıştırılabilmesi için bir dizi ön işlem metotlarından yararlanılmıştır. Finansal duygu analizi için hazır olan Twitter kullanıcı verisi naif Bayes sınıflandırıcısının İngilizce ve Türkçe veriler için ayrı ayrı eğitilmesi ile verinin pozitif ve negatif olarak sınıflandırılması amaçlanmıştır.

İkinci bölümde, GloVe, fastText ve Word2vec gibi popüler kelime yerleştirme metotlarını kullanarak Twitter'dan elde edilen ve duygu analizine sokulmuş verilerden her bir kelime yerleştirme yöntemi ile birer vektör uzayı elde edip bir sonraki bölümde kullanılacak olan derin öğrenme algoritmalarına girdi elde edilecek. Bu bölümde şu ana kadar yapılacak olan çalışmaya ek olarak da her bir kelime yerleştirme metodunu basit bir yapay sinir ağı üzerinde çalıştırarak olası bir duygu analizinin doğruluk derecesi incelenecek.

Üçüncü bölümde ise bir önceki bölümden elde ettiğimiz GloVe, fastText ve Word2vec vektör uzayı çıktılarını CNN, RNN ve LSTM gibi derin öğrenme algoritmaları üzerinde birer girdi olarak kullanıp farklı kombinasyonlarının performans değerleri ölçümlenecek. Bu bölümde test edilen tüm kombinasyonlar içerisinden doğruluk derecesi en yüksek olan uygulamanın bir sonraki bölümden elde edilecek olan karma model için aday olacak.

Dördüncü ve son bölümde ise aşamalı olarak elde ettiğimiz birbirinden farklı yapıda olan iki tahmin modelini ağırlıklı çoğunluk oylaması yöntemiyle birleştirilip karma bir model oluşturulacak. Önerilen yeni karma modelin performansı literatür çalışmalarıyla kıyaslanacak ve kayda değer üstünlüğe sahip olduğu analiz edilmeye çalışılacak.

1.3. Tezin Yönetimi

Bu tez çalışması hazırlanırken makalelerden, raporlardan, tezlerden ve diğer projelerden yararlanılmıştır. Çalışma boyunca, veri işleme işlemleri için Pandas, Numpy, scikit-learn gibi kütüphanelerden, Twitter verilerinin Türkçe ve İngilizce olarak toplanması için bir web crawler scriptinden, doğal dil işleme işlemleri için TextBlob ve Zemberek kütüphanelerinden, metin sınıflandırma için Naif Bayes sınıflandırıcısından, finansal duygu analizi esnasından derin öğrenme yöntemlerinin uygulanması için Keras kütüphanesinden faydalanılmıştır. Zaman serileri analizi için ise statsmodels kütüphanesinden yararlanılmıştır.

Elde edilen verilen sırasıyla temizlenerek üzerinde finansal duygu analizi çalıştırılmış elde edilen modellerin başarı oranını arttırmak için farklı deneyler gerçekleştirilmiştir. Farkı yapılarda elde edilen modellerin başarı oranı ağırlıklı hesaplanarak ortaya karma bir model çıkarılmıştır.

2. İLGİLİ ÇALIŞMALAR

Yapılan literatür araştırmasında görülüyor ki bu tez kapsamında ele alınan konuların her biri kendi içerisinde farklı araştırmalara konu olmakta. Twitter sosyal medya platformunun göz ardı edilemez önemli istatistikleri sadece şirketlerin gözünde değil akademik çalışma amacında olan araştırmacıların gözünde de bir öneme sahip olduğu kolayca söylenebilir. Bu ana kadar yapılan çalışmaların, özellikle duygu analizinin büyük veri kümesi ihtiyacının genellikle Twitter sosyal platformundan saylandığı görülmektedir. Stenqvist & Lönnö (2017) yaptığı araştırmada her ne kadar direkt bir döviz kuru tahmin modeli olmasa da Bitcoin fiyatlarındaki dalgalanma ile Twitter verisi üzerinde bir korelasyon olup olmadığı araştırılmıştır. Bu çalışmada herhangi bir derin öğrenme modeli ya da bir zaman serisi analizi kullanılmamış olup Twitter veri kümesi üzerinde gerçek zamanlı duygu analizi gerçekleştiren VADER adında bir yazılım kullanılmıştır.

Ozcan (2016) da yaptığı çalışmasında Twitter'da çok konuşulan konular (trending topics) kullanarak bir veri kümesi oluşturup üzerinde çalışmıştır. Bu veri kümesi kullanarak duyarlılık tabanlı bir konu kümeleme modeli oluşturmayı amaçlamıştır. Yapılan çalışmada tüm tweetlerin tek bir dokümanda toplanabilmesi için Dirichlet sürecinden yararlanılmış daha sonra bir konu tabanlı duygu skoru oluşturulmuştur. Bu aşamadan sonra döviz kurlarının tahmin edilebilmesi için biri karşılaştırma yapmak amacıyla bazal değerlerin elde edileceği bir AR modeli, bir diğeri de oluşturulan tweet serisi için inşa edilen bir VAR modeli olmak üzere iki farklı modelden yararlanılmıştır.

Duygu analizinin Naif Bayes algoritması ile yapılması açısından bakıldığında bu araştırmaya en yakın çalışma Komariah & Sin (2015) yapmış olduğu çalışma olarak söylenebilir. Fakat bu çalışmada sadece veri kümesinin toplanması, bu veri kümesinin kullanılarak bir sözlük yaratılması, modelin oluşturulması ve duygu analizinin yapılması gibi adımlardan oluşmaktadır. Eğitim verilerinden oluşturulan sözlük kullanılarak veriler pozitif ve negatif olarak ayrılmıştır. Daha sonra duygu analizi uygulanan veri kümesinin sonuçları Endonezya Rupisi'nin Amerika Doları karşısındaki değişiminin Twitter veri kümesi ile yapılan duygu analizi ile arasında bir korelasyonunun olup olmadığı gözlenmiştir.

Ozturk & Ciftci (2014) çalışmasında ise yine Twitter verisinden faydalanılarak bir duygu analizi yapmış ve bu çalışmaya da benzer olarak sayısal değerler Türkiye Cumhuriyeti Merkez Bankası'ndan döviz kurları olarak belli bir tarih aralığında toplanmıştır. Fakat yapılan bu çalışmada, bu tez kapsamında kullanılan fiyat verisi yerine getiri verisi kullanılmış, para biriminin yönünü belirleyen hareket gözlemlenmeye çalışılmıştır. Aynı zamanda duygu analizi kapsamında veriler alış ve satış olarak etiketlenmiştir. Çalışma Twitter verisinin döviz kurunun değişim yönünün tahmin edilmesi amacıyla kullanabileceğini iddia etmiştir.

Yasir vd. (2019) yaptığı çalışmada ise Twitter veri kümesi kullanılarak bir duygu analizi yapılmış ve döviz kuru üzerindeki etkisi araştırılmıştır. Bu çalışmada, yerel ve küresel olayların duyarlılığını içeren bir derin öğrenme modeli sadece genel etkiyi hesaplayan bir duygu analizi ile değil pozitif ve negatif tweetlerin pozitif yüzdesi hesaplanarak birleştirilmiştir. Bu tez kapsamında sayısal veri olarak kullanılan Türk Lirası ve Amerikan Doları yerine hem döviz kuru hem de petrol ve altın fiyatları da incelenmiştir. Deney sonuçları incelendiğinde bu tez kapsamında sunulacak olan hibrid modelin gözle görülür üstünlük sağladığı tespit edilmiştir.

Her ne kadar Twitter'ı bir veri kümesi kaynağı olarak kullanan araştırmalar olsa da bu tez kapsamında sunulacak olan modele tam anlamıyla benzer bir araştırmaya rastlanmamıştır. Zaman serileri analizi kapsamında bakıldığında literatür zaman zaman sadece yumuşatma tekniklerinin ya da ARIMA gibi modellerin kullanıldığı görülmüştür. Bu çalışmalarda genellikle sadece zaman serisi üzerinde yumuşatma teknikleri ya da ARIMA modeli uygulanmıştır (Codru & Eva, 1983; Rout vd., 2014). Bu çalışmalarda herhangi bir kelime yerleştirme yönteminden ya da derin öğrenme modellerinden yararlanılmamıştır. Twitter veri kümesinin kullanılmadığı bu tip çalışmalarda yer yer makine öğrenmesi algoritmalarından yararlanılıp sadece regresyon modelleri ile döviz kuru tahmini yapılmaya çalışılmış ve oluşturulan modelin performansı yapay sinir ağları ile öngörme performansını arttırabilmek için girdi özelliklerinin arasındaki karmaşık ilişkilerin modellenmesi sağlanmıştır (Rojas & Herman, 2018). Çalışmaların deney sonuçları incelendiğinde bu tez kapsamında sunulacak olan derin öğrenme modelleri ile performansı iyileştirilmiş modelin gerisinde kalmaktadır (Palikuca & Seidl, 2016).

Çalışma konusuna zaman serileri analizini katmış olan deneylerde ise daha çok farkı zaman serisi analizi yöntemleri denenmiş fakat bunun yanında bir hibrid model

amaçlanmamış ya da herhangi bir kelime yerleştirme yönteminden ya da derin öğrenme modellerinden yararlanılmamıştır (Varenius, 2017).

3. DUYGU ANALİZİ

Duygu analizi bir diğer adıyla görüş madenciliği, veri madenciliğinin bir parçası olarak algılanabilir ve metin analizi olarak da adlandırılabilir. Duygu analizi, temelde, metinlerin içerisinde gizlenmiş olan görüşlerin ve duyguların anlaşılması, bu görüşlerin uygun yöntemler ile olumlu, olumsuz ya da tarafsız olarak etiketlenmesini amaçlar.

Duygu analizi ya da bir diğer adıyla görüş madenciliği metin odaklı olduğundan çoğunlukla sosyal medya içerikleri, anket çalışmaları, ürünlere yazılan yorumların üzerinde çalıştırılır. Bu bölümde ise tez kapsamında sosyal medyanın gözlemlenerek belirli konularda görüş çıkarımı yapabilmek için kamuoyu ve tepkileri hakkında genel bir bilgi edineceğiz.

Görüş madenciliğinin uygun bir şekilde yapılabilmesi için elde edilen verilen temizlenmesi ve analiz için hazır halde getirilmesi gereklidir. Bu aşamalar için genellikle verinin işlenmesi için pandas, kelimelerin sözlükteki doğru hallerinin bulunması, kelimelerin köklerinin bulunması gibi doğal dil işleme yöntemlerinin uygulanması için NLTK, veri kümesinin eğitilmesi ve duygu analizinin gerçekleştirilebilmesi için TextBlob gibi kütüphanelerden yararlanılmıştır.

Verinin eğitilmesi aşamasında Naif Bayes sınıflandırıcısı TextBlob ile birlikte kullanılmıştır.

3.1. Duygu Analizi İçin Ön Çalışma

Asıl amacın uygulanabilmesi için, yani finansal duygu analizinin gerçekleştirilebilmesi için amaca uygun bir veri kümesinin, tez sonunda oluşturulacak karma modelin ihtiyacına uygun girdi şeklinde hazırlanması gerekmektedir. Bu sebeple incelemek üzere önceden belirlediğimiz tarih aralığına ait olan kamuoyu verisinin kaynağından İngilizce ve Türkçe olarak iki farklı dilde toplanması gereklidir.

Özellike finansal duygu analizi için çalışma ile doğrudan ilgili olan verilerin toplanması için belirli konulara yönelmek gerekmektedir. Bu şekilde kamuoyunun tepkisi doğru bir şekilde hedeflenir.

Veri kaynağı olarak kullanacağımız Twitter'ın sunmuş olduğu etiket (hashtag) özelliği ile kamuoyunun görüşlerinin belli konularda kategorize edilmesi işlemini bir nebze olsun kolaylaştırmaktadır. Twitter etiketi ise, kullanıcıların oluşturabildiği ve

kullanıcıların daha güvenilir arama sonuçları elde etmesi için kullanılan, bir konuyu ya da bir temayı tanımlamak için kullanılan ve önüne pound işareti konularak belirtilen kelimeler ve kelime öbekleridir. Finansal duygu analizi için hedef aldığımız verilere ulaşmak için ise Twitter'ın sunmuş olduğu gelişmiş arama özelliği ile #usdtl, #usdtry, #usd/tl, #usd/try, #dolartl, #dolartry, #dolar/tl, #dolar/try, #dollartl, #dollartry, #dollar/tl, #dollar/try etiketleri izlenmiş ve veriler bu konular üzerinden toplanmıştır.

3.2. Veri Kümesi Kaynağı Olarak Twitter

2019 istatistiki verilerine göre Twitter, her gün üretilen 500 milyon içerik 139 milyon aktif günlük ve 326 milyon aylık kullanıcı sayısı ile dünya üzerindeki en popüler sosyal platformlardan biri olmaya devam etmektedir (Cooper, 2019). Demografik yapısı itibari ile özellikle markaların ve işletmelerin hedef kitlesi halini de almaktadır. Twitter, kullanıcılarına teknolojik alt yapısı ile gerçek zamana yakın içerikler sunmaktadır. Yani üretilen içerikler anlık olarak Twitter üzerinde profiliniz aracılığı ile gönderilir ve bu içerikler takipçileriniz tarafından hemen izlenebilir. Bu, özellikle işletmeler için harika bir özelliktir. Twitter'daki bu iletişim ağı çok hızlı bir şekilde etkileşim içerisinde olduğundan şirketler bu iletişim zincirinin içerisinde olmayı benimsemektedirler.

Bu anlatılanlar her ne kadar sadece işletmelerin göz bebeği olarak görülüyor olsa da Twitter, verinin çeşitliliği, hızla üretilmesi ve büyüklüğü nedeniyle akademik çalışmalar için de edilen bir veri kaynağı haline gelmektedir.

Teknik özellikler açısından bakıldığında, araştırmacıların Twitter'ı bir veri kaynağı olarak tercih etmelerinin sebebi olarak, Twitter alt yapısının sunmuş olduğu farklı uygulama programlama ara yüzleri kabul edilebilir. Her ne kadar her uygulama ara yüzünün kendine göre sınırları olsa da alternatif yöntemler ile hedef kitleden araştırmalar için yeterli miktarda veri toplanabilmektedir.

Tez kapsamında daha önce belirlenmiş olan ilgili etiketlerin okunması ve Twitter'ın sunmuş olduğu ileri düzey arama özelliği ile kamuoyunun gerçek zamanlı olarak ürettiği içeriklerin toplanması adımları öncelikli olarak gerçekleştirilecektir.

3.3. Verilerin Twitter Üzerinden Toplanması

Twitter sosyal platformu kullanarak toplanacak olan veriler, zaman serileri analizi yaparken kullanılacak olan sayısal veriler ile uyumluluk göstermesi açısından 01 Ocak 2018 ile 31 Aralık 2018 arasında daha önceden belirlenmiş olan etiketlere yazılan içeriklerin toplanması işlemini kapsamaktadır.

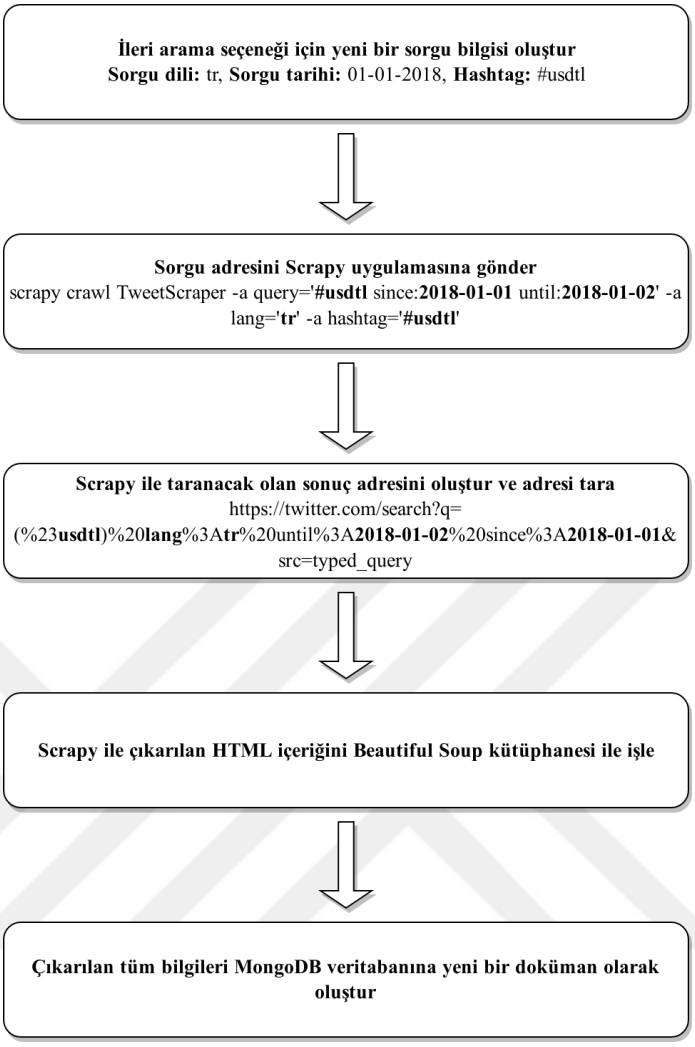
Veriler toplanırken programlama dili olarak Python 3.6.8, içerikleri tarayan bir gezginin oluşturulması için Scrapy uygulama çatısı ve elde edilen içeriğin işlenmesi için Beautiful Soup kütüphanesi ve işlenen verinin hızlı bir şekilde depolanması için MongoDB doküman veri tabanı tercih edilmiştir.

Scrapy, web sitelerinden ihtiyacımız olan verileri toplamamız için bize bir takım özellikler sunan açık kaynaklı bir uygulama çerçevesidir. Bu uygulama vasıtası için her bir gün içerisinde üretilmiş olan etiket içeriklerinin ziyaret edilip içerik çıkarımı yapılması için kullanılmıştır. Ziyaret edilecek olan sayfalar öncelikli olarak oluşturulup daha sonra Twitter ileri arama ara yüzü aracılığı ile el de edilmiştir.

Beautiful Soup, web sitelerinin içeriklerinin oluşturulduğu HTML ve XML gibi biçimlendirme dillerinin anlamlı ve okunaklı bir hale getirmesini sağlayan bir Python kütüphanesidir. Twitter üzerinden içerik üretilirken HTML ve XML gibi biçimlendirme dilleri sebebiyle fazladan oluştan doküman elemanlarından kaçınabilmek için kullanılmıştır.

MongoDB, yüksek hacimli veri depolama için kullanılan, belge odaklı bir NoSQL veri tabanıdır. Belge odaklı yapısı ile birlikte Scrapy ve Beautiful Soup vasıtası ile toplanan içeriğin her birini bir doküman olarak hızlı bir şekilde saklamak için kullanılmıştır.

Yukarıda belirtilen tüm bu araçların ahenk içerisinde çalışabilmesi için oluşturulan uygulama akışı Şekil 3.1'de gösterilmektedir.



Şekil 3.1 Veri toplama uygulaması akış şeması

Dokümanlar oluşturulurken herhangi bir tekrarın önüne geçebilmek için içerik sahibinin kimlik bilgisi ve içeriğin kimlik bilgisi her defasında MongoDB depolama alanı içerisinde aranarak herhangi bir eşleşmenin olup olmadığı denetlenmiştir. Eşleşen bir kayıt bulunduğunda yeni bir doküman oluşumu engellenmiştir.

TextBlob kütüphanesi yardımı ile her bir dokümanın tekrardan incelenerek hangi dile ait olduğu da anlaşılmaya çalışılmıştır. Dil ataması yapılamayan dokümanlar ise "diğer diller" olarak işaretlenmiştir.

Örnek dil tespit etme uygulaması Şekil 3.2'de gösterilmektedir.

```
text = TextBlob("Borsada Afrin konusunda İngiltere olmak üzere Nato'dan gelen
destek haberleri pozitif algıyı etkilerken dolarda çekilme destekleyici.")
text.detect_language()
```

Şekil 3.2 Dil tespiti yapan örnek uygulama

Tüm arama sonunda oluşturulan doküman sayıları Tablo 3.1'de gösterilmektedir.

Tablo 3.1 Twitter üzerinden toplanan toplam doküman istatistikleri

Toplam veri kümesi	Türkçe	İngilizce	Diğer diller
111516	91197	11653	8666

İçeriklerin etiketlere göre dağılımı Tablo 3.2'de gösterilmektedir.

Tablo 3.2 Twitter dokümanlarına ait olan etiketlerin dağılımı

Etiket	Toplam	Türkçe	İngilizce	Diğer
#usdtl	3623	917	25	2681
#usdtry	59607	47985	7957	3665
#usd/tl	2633	2343	132	158
#usd/try	7075	3413	2976	686
#dolartl	10105	8715	201	1189
#dolartry	10	10	0	0
#dolar/tl	27999	27696	19	284
#dolar/try	86	81	4	1
#dollartl	5	2	3	0
#dollartry	0	0	0	0
#dollar/tl	58	28	28	2
#dollar/try	315	7	308	0
Toplam	111516	91197	11653	8666

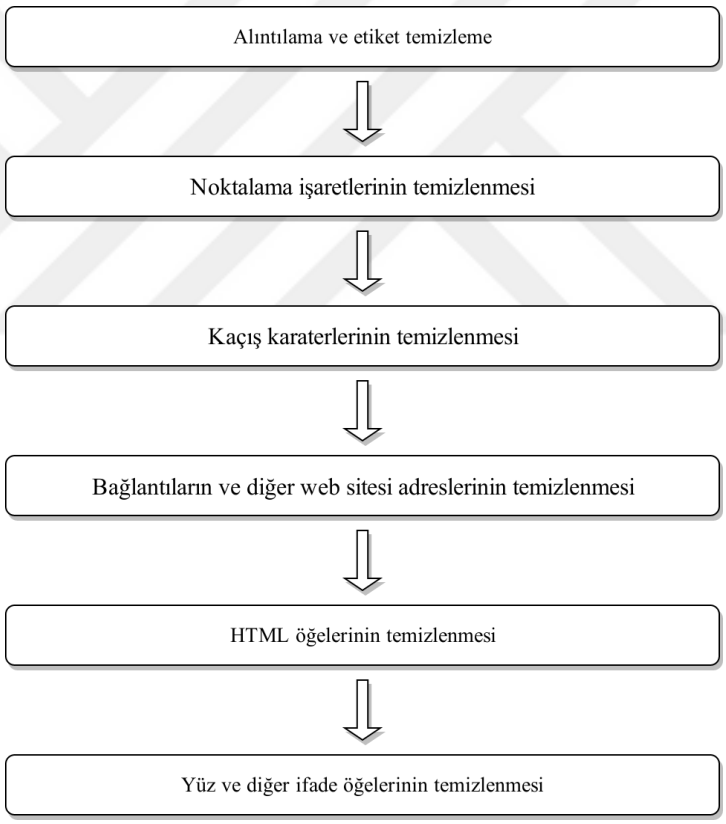
3.4. Twitter Verileri İçin Ön İşlemler

Dünya üzerinde farklı cihazlardan üretilen verilerin tamamının %80'i yapılandırılmamış veriden oluşmaktadır. Yapılandırılmamış veri olarak bahsedilen yapı genelde her şeyi kapsamaktadır. Yani önceden belirlenmiş bir modele ya da şemaya uymayan verinin her biri yapılandırılmamış veri olarak adlandırılabilir. İnsanlar ve makineler tarafından oluşturulan veriler olarak ikiye ayrılan bu tipteki verilerin insanlar tarafından üretilen hali üzerinde duracağız. İnsanlar tarafından oluşturulan yapılandırılmamış içerikler olarak metin verileri, eposta içerikleri, web siteleri, sosyal medya içerikleri, ses dosyaları, görüntü dosyaları, fotoğraflar olarak sayılabilir.

Yapılandırılmamış veriler farklı boyutlarda ve farklı tiplerde üretildiği için zaman zaman kirli veriler ile karşılaşılmaktadır. Daha öncede izah edildiği gibi herhangi bir kurala bağlı olmadan üretilen bu verilere, üretildiği hali ile işlemek pek doğru olmayan sonuçlar elde etmemize sebep olacaktır.

Sonuç olarak bu tip yapılandırılmamış olarak adlandırılan verilerin öncelikli olarak kirli halinden arındırılıp temizlenmesi gerekmektedir. Tez kapsamında bu bölümde Twitter üzerinden toplanan verilerin hangi yöntemler ile temizlendiğinden bahsedilmektedir.

Twitter veri kümesi temizleme aşamaları Şekil 3.3 ile gösterilmiştir.

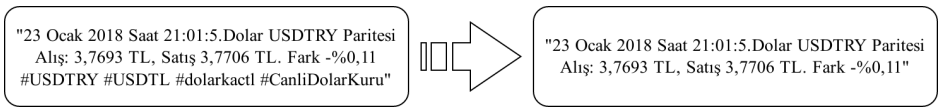


Şekil 3.3 Veri kümesi temizleme aşamaları

3.4.1. Alıntılama ve etiket temizleme

Alıntılama ve etiket karakterlerinin, bu karakterlere ait olan kelimelerin ve kelime öbeklerinin temizlenme işlemi gerçekleştirilmiştir.

Örnek uygulama Şekil 3.4 ile gösterilmektedir.



Şekil 3.4 Alıntılama ve etiket temizleme

3.4.2. Noktalama işareti temizleme

Metinlerin içerisinde bulunan noktalama işaretleri Python'ın sunmuş olduğu *punctuation* eklentisi ile silinmiştir. Bu eklentide '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~' gibi karakterler tutulmaktadır.

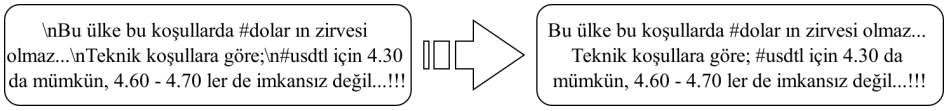
Örnek uygulama Şekil 3.5 ile gösterilmektedir.

Şekil 3.5 Noktalama işareti temizleme

3.4.3. Kaçış karakteri temizleme

Kaçış karakterleri \a, \b, \t, \n, \v, \f, \r gibi karakterleri kapsamaktadır. Doğal dil işleme yöntemlerinin doğru uygulanabilmesi için bu karakterlerin de temizlenmesi gerekmektedir.

Örnek uygulama Şekil 3.6 ile gösterilmektedir.

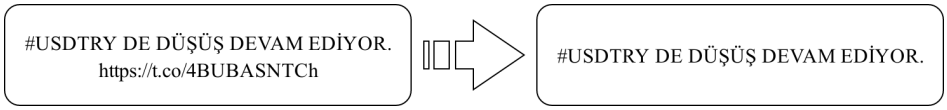


Şekil 3.6 Kaçış karakteri temizleme

3.4.4. Bağlantı ve web adresi temizleme

Twitter paylaşılan içeriklerde eğer bir web adresi ya da bir bağlantı varsa kendi sağladığı bir adres kısaltma servisi ile her bir bağlantı adresini belirli bir formata çevirmektedir. Bu özellik de içeriklerin içerisinden bağlantı ve web adresi gibi istenmeyen bölümlerin çıkarılmasını kolaylaştırmaktadır.

Örnek uygulama Şekil 3.7 ile gösterilmektedir.



Şekil 3.7 Bağlantı ve web adresi temizleme

3.4.5. HTML öğesi temizleme

Twitter verilerinde bulunan HTML öğelerinin temizlenmesi için Python *html* eklentisinden yararlanılmıştır. Eklentinin sunmuş olduğu unescape metodu ile '&' ve benzeri karakterler '&' gibi karşılıkları olan metinlere dönüştürülmüştür. Daha sonra da noktalama işaret temizleme aşaması tekrar çalıştırılacak bu oluşan yeni karakterler temizlenmiştir.

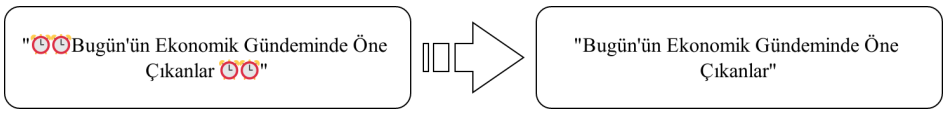
Örnek uygulama Şekil 3.8'de gösterilmektedir.

Şekil 3.8 HTML öğesi temizleme

3.4.6 Yüz ve diğer ifadeleri temizleme

Twitter yüz ve diğer ifadelerin kullanılmasına izin vermektedir. Duygu belirtebilen bu ifadelerin duygu analizi aşamasında çalışmaya olumsuz etki etmemesi için temizlenmesi gerekmektedir.

Örnek uygulama Şekil 3.9'da gösterilmektedir.



Şekil 3.9 Yüz ve diğer ifadeleri temizleme

3.5. Twitter Verilerinin Duygu Analizi İçin Etiketlenmesi

TextBlob'un sunmuş olduğu etiketlenmiş veri kümesini kullanarak İngilizce metinler üzerinde duygu analizi çalıştırmak mümkün iken Türkçe dili için herhangi bir etiketli veri kümesi olmadığından bu bölümde ise öncelikli olarak elde etmiş olduğumuz Twitter Türkçe veri kümesinin eğitilmesi amaçlanmıştır.

Kaggle üzerinden paylaşılmış olan etiketli Hepsiburada verisi alınarak daha önceki bölümlerde bahsedilen veri temizleme ve ön işleme işlemleri uygulanmıştır (Sahin, 2019). Zemberek kütüphanesini ile dokümanlar temel yazım denetleyicisi, kelime önerisi ve gürültülü metin temizleme amacı için normalleştirme metodu, morfolojik analiz, belirsizlik ve sözcük üretimi için ise morfoloji metodundan yararlanılmıştır. Gereksiz kelimelerin (stopwords) kaldırılması amacıyla Türkçe için Zemberek'den İngilizce için ise NLTK'in gereksiz kelime öbekleri kullanılmıştır.

Bu adımda ise 1'den 5'e kadar puanlanmış olan yorumların; 1 ve 2 puan olanların negatif, 4 ve 5 puan olanların ise pozitif olarak işaretlenmesi sağlanmıştır.

Etikenlenmiş veriler Tablo 3.3 ile gösterilmektedir.

Tablo 3.3 Tüm işaretlenmiş olan Türkçe veriler

Pozitif yorumlar	**Negatif yorumlar**
5000	5000

Türkçe veri kümesinin eğitilmesi için TextBlob'un *NaiveBayesClassifier* sınıfı kullanılmıştır.

Naif Bayes algoritması özellikle duygu analizi ve benzeri sınıflandırma problemlerinde kullanılan, uygulaması basit ve hızlı, olasılıkları ve koşullu olasılıkları hesaplamak için Bayes Teorisi'ni kullanan bir sınıflandırıcıdır (Zhang & Li, 2007).

Naif Bayes algoritması, belirli özellikleri temel alan farklı nesneleri sınıflandırmak için kullanılan bir sınıflandırıcı modelidir. Naif Bayes sınıflandırıcısı, bir sınıftaki belirli bir özelliğin varlığının diğer herhangi bir özelliğin varlığı ile ilgisiz olduğunu varsayar. Bu sebeple tahmincilerin bağımsız olması beklenmektedir.

Bayes teorisi ile kelime frekanslarını kullanarak dokümanlar üzerinde bir olasılık hesabı yapabilmek için kullanılacak.

$$p(C|x) = \frac{p(C)p(x|C)}{p(x)} \tag{3.1}$$

Denklem 3.1'de gösterilen Bayes teroisi formülü şu şekilde açıklanabilir P(c|x), öngörülen (x, özellikler) verilen sınıfın (c, hedef) sonraki olasılığıdır, P(c) sınıfın önceki olasılığıdır, P(x|c), verilen sınıfın tahmin edicisi olma olasılığıdır, P(x) ise tahmin edicinin öncelikli olasılığıdır.

NTLK'in Naif Bayes sınıfını uygulayan TexBlob bir etiketin olasılığını bulmak için ilk olarak Bayes Teorisi'ni P(sınıf) ve P(özellikler|sınıf) cinsinden P(sınıf|özellikler) olarak ifade eder.

Denklem 3.1'de gösterilen formül Denklem 3.2'de gösterildiği gibi ifade edilebilir:

$$P(sınıf|özellikler) = \frac{P(sınıf) * P(özellikler|sınıf)}{P(özellikler)} \tag{3.2}$$

Denklem 3.2'deki ifade algoritma tarafından tüm özelliklerin "naif" olduğu varsayımını yapar ve sınıf Denklem 3.3 tarafından şu şekilde ifade edilir:

$$P(sınıf|özellikler) = \frac{P(sınıf) * P(f1|sınıf) *\ldots* P(fn|sınıf)}{P(özellikler)} \tag{3.3}$$

P(özellikler)'in hesaplanması yerine, algoritma sadece her sınıf bir sayaç hesaplar ve tüm bu sonuçları normalleştirir ve sonra hepsini birbiri ile Denklem 3.4'de gösterilen formül yardımı ile toplar:

$$P(sınıf|özellikler) = \frac{(sınıf) * P(f1|sınıf) *\ldots* P(fn|sınıf)}{SUM[sınıf](P(sınıf) * P(f1|sınıf) *\ldots* P(fn|sınıf))} \tag{3.4}$$

Daha sonra pozitif ve negatif olarak etiketlenen veriler %80 eğitim ve %20 test verisi olmak üzere iki ayrılmıştır.

Eğitim ve test kümesi olarak ayrılan veriler Tablo 3.4 ile gösterilmektedir.

Tablo 3.4 Türkçe sınıflandırma için veri kümesi

Eğitim veri kümesi	Test veri kümesi
8000	2000

Ve daha sonra eğitim verisi NaiveBayesClassifier sınıflandırıcısı ile eğitilir.

Uygulama örneği Şekil 3.10 ile gösterilmiştir.

```
tr_model = NaiveBayesClassifier(egitim_veri_seti)
```

Şekil 3.10 Türkçe sınıflandırıcı eğitim uygulaması

Bu işlem sonrası elde edilen Türkçe sınıflandırıcı modelinden *%81* doğruluk derecesi elde edilmiştir. İngilizce için kullanılan önceden eğitimli modelin ise *%76* gibi bir doğruluk oranı vardır.

3.6. Twitter Verileri Kullanarak Duygu Analizinin Uygulanması

Önceki bölümlerde temizlenip pozitifi ve negatif olarak etiketlendikten sonra eğitilen veri kümeleri ile üretilen İngilizce ve Türkçe sınıflandırıcı modelleri duygu analizi yapılmak üzere tüm Twitter dokümanları üzerinde çalıştırılmıştır.

Türkçe için Kaggle veri kümesi eğitilerek oluşturduğumuz modeli kullanırken İngilizce için ise NLTK'in önceden eğitilmiş *NaiveBayesClassifier* sınıfını kullanacağız.

Türkçe Twitter dokümanları için çalıştırılan duygu analiz uygulaması Şekil 3.11'de gösterilirken, İngilizce Twitter dokümanları için çalıştırılan duygu analizi uygulaması Şekil 3.12'de gösterilmiştir.

```
siniflandirici = NaiveBayesClassifier(ingilizce_veri_seti)
metin = "becoming increasingly popular offers lucrative option token based like
mining inflation model return 80 fee holders"
siniflandirici.classify(metin) # neg
```

Şekil 3.11 İngilizce sınıflandırıcı ile duygu tespiti

```
siniflandirici = NaiveBayesClassifier(turkce_veri_seti)
metin = "yoluna devam ediyor formasyon hedefi olan yükselişe gözünü dikmiş
durumda"
siniflandirici.classify(metin) # pos
```

Şekil 3.12 Türkçe sınıflandırıcı ile duygu tespiti

Yukarıda belirtilen etiketleme işleminden sonra duygu analizi sonuçları Tablo 3.5 ile gösterilmiştir.

Tablo 3.5 Duygu analizi uygulanmış tüm verilerin dağılımı

	Türkçe	**İngilizce**
Pozitif	75947	4585
Negatif	15244	6968
Diğer	6	100
Toplam	91197	11653

4. KELİME YERLEŞTİRME TEKNİKLERİ

Kelime yerleştirme teknikleri ya da diğer bilenen adıyla kelime vektörleri makine öğrenmesi için çok önemli olan, dokümanlar içerisindeki benzer anlamı olan kelimelerin ve kelime öbeklerinin benzer bir gösterim ile sunulmasını sağlayan bir tür temsil yöntemidir. Bu yöntemler ile kelimeler, kelime öbekleri her bir yöntemin uyguladığı hesaplama teknikleri ile kayan nokta sayısal değerler ile bir vektör uzayında temsil edilir. Yani bir kelime yerleştirme tekniği bir dokümandaki kelimelerin her birinin bir vektör uzayını çıkarıp tüm sayılar değerlerini bu vektör uzayının üzerine taşır ve bir dizi sayısal veri üretir. Kısaca kelime yerleştirme, metinleri sayısal verilere dönüştüren yöntemlerdir.

Makine öğrenmesi algoritmaları, derin öğrenme algoritmaları ya da yapay sinir ağları sayılar ile çalıştığından kelimelerin sayısal sunumları ile bu yöntemlere girdiler oluşturulabilir. Aslında kısaca denilebilir ki, kelime yerleştirme teknikleri insanın anlayabildiği dokümanların bilgisayarlar ve programlar tarafından kolayca anlaşılabilmesi için kullanılmaktadır.

Bunun yanında aynı gibi görünen cümlelerin aslında farklı anlamlara sahip olduğu durumlarda kelimelerin ve kelime öbeklerinin vektör uzayında benzerlik hesapları ile gerçek anlamlarının hesaplanarak tespit edilmesi doğal dil işleme problemlerine de güvenilir çözümler sunabilmektedir.

4.1. GloVe

Pennington, Socher, & Manning (2014) yaptıkları çalışma sonrası sundukları GloVe yani diğer adı ile Global Vector açık kaynak olarak geliştirilmiş olan bir denetimsiz öğrenme algoritmasıdır. GloVe, bir tahmin modelinden daha çok bir kelimenin bir bağlamda ne sıklıkla göründüğünü hesaplayan bir kelime yerleştirme yöntemidir. GloVe herhangi bir kelimenin kullanılma sıklığını her bir doküman için ayrı hesaplar. Matris çarpanlara ayırma ve yerel bağlam penceresi yöntemleri olmak üzere iki farklı modelin özelliklerini birleştirerek her bir kelime ya da kelime öbeği için vektör uzayları oluşturur.

Çoğu kelime vektör metotları kelimelerin birbirlerine olan uzaklıklarını hesaplayarak benzerlikler bulmakta. TF-IDF ya da LSA gibi yöntemler bir metin madenciliği yöntemi olan kosinüs benzerliği hesaplama yöntemini kullanarak benzerlikler ve vektör uzayı çıkarmaktadır (Landauer, Foltz, & Laham, 1998). Basitçe denilebilir ki oluşturulan bu vektörlerin birbirlerine olan benzerlikleri, birbirleri olan ilişkisi bir açı değeri ile belirtilir. GloVe ise daha önce bahsedildiği gibi Word2vec'in dayandığı Mikolov vd. (2013) skip-gram modeli ile matris faktörleştirme yöntemlerinden biri olan Deerwester vd. (1990) LSA yöntemini birleştirmektedir. Kelimelerin birlikte oluşma olasılıkları yerine kelimelerin eşzamanlı olma olasılıklarının oranını, içerdiği bilgiyi ve bu bilgiyi vektör farklılıkları hesaplayarak oluşturmayı amaçlayan bir yöntemdir.

GloVe öncelikle kelime-kelime birlikte oluşma matrisi oluşturup ilk olarak bir kelimenin bulunduğu kapsam içerisinde meydana gelme olasılığını hesaplar. Hesaplama formülü Denklem 4.1'de gösterilmiştir.

$$Xi = \Sigma_k X_{ik} \tag{4.1}$$

X kelime-kelime birlikte oluşma matrisi

X_{ij} bir kelimenin bir kapsamda oluşma sayısı

i kelime

j kapsam

Kelimenin kapsam içerisinde görünme ihtimali Denklem 4.2'de gösterildiği gibi hesaplanır:

$$P_{ij} = P(j|i) = \frac{X_{ij}}{x_i} \tag{4.2}$$

Denklem 4.3 ile vektörler kullanarak birlikle oluşma oranlarının tahmin edilir:

$$F(w_i, w_j, \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.3}$$

F i, j ve k değişkenlerini bir girdi olarak kullanan bir işlev

w girdi olarak kullanılan kelime yerleştirme vektörü

$\tilde{w}$ çıktı olarak kullanılan kelime yerleştirme vektörü

F'in vektör uzayındaki P_{ik}/P_{jk} olasılığını hesaplaması için vektör uzaylarının farkı Denklem 4.4 ile hesaplanır.

$$F(w_i - w_j, \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.4}$$

Denklem 4.5 ile doğrusal bir ilişki oluşturmak için Denklem 4.4'deki parametreler iç çarpımdan yararlanılır:

$$F((w_i - w_j)^T \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.5}$$

Daha sonra Denklem 4.6 ve Denklem 4.7'de gösterilen yollar ile sadeleştirilir.

$$w_i^T w_k = log(P_{ik}) = log(X_{ik}) - log(X_i) \tag{4.6}$$

$$w_i^T wk + b_i + \tilde{b}_k = log(X_{ik}) \tag{4.7}$$

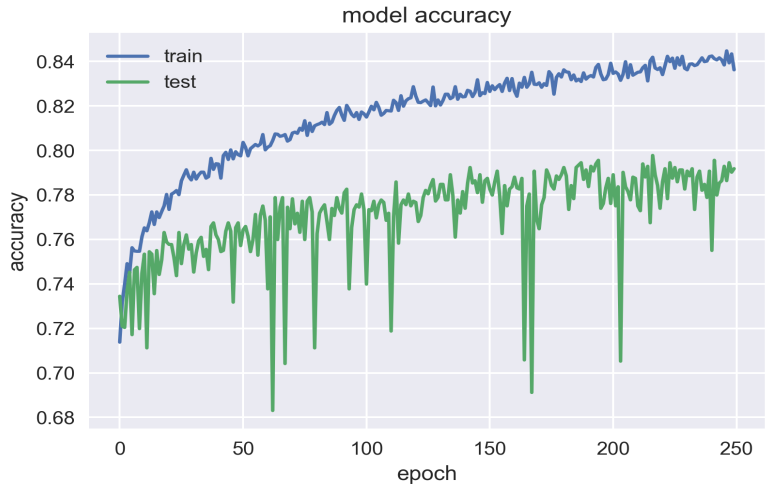
4.1.1. GloVe uygulaması için ön hazırlık

Bir GloVe modeli oluşturmak için Python Glove kütüphanesinden yararlanılmış. Bu uygulama gerçek GloVe uygulamasına bir ara yüz oluşturmaktadır. Bu kütüphane vasıtası ile GloVe herhangi bir Python kodu içerisinden çağırabilir duruma gelmektedir.

Tez kapsamında önceki bölümlerde temizlenen Twitter verisi öncelikle TextBlob kütüphanesinin belirtkeleme işlevi kullanılarak birer kelime dizi haline getirilmiştir. Her bir Twitter doküman bu işlev ile kelimeler dizi haline getirildikten hemen sonra %80 eğitim ve %20 test verisi olarak ayrıştırılmıştır.

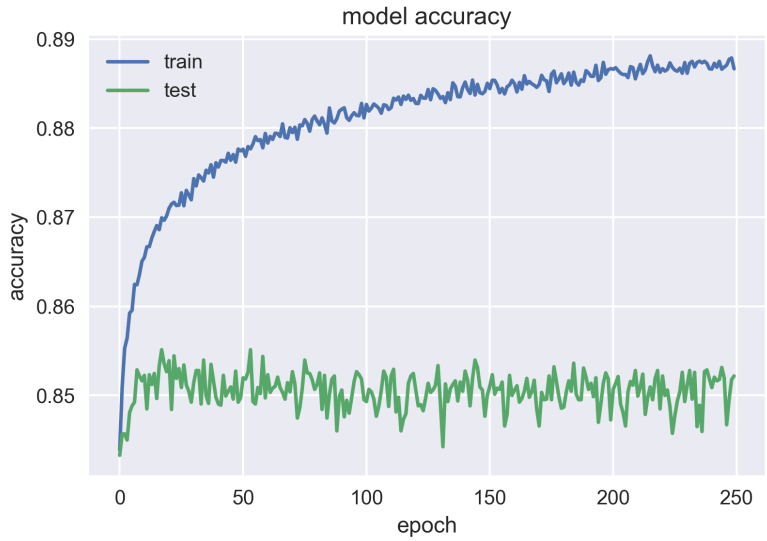
4.1.2. GloVe uygulaması

GloVe'dan elde etmek istediğimiz model bir önceki bölümde oluşturduğumuz eğitim verisi ile eğitilmiştir. Elde edilen model ile bir özellik çıkarımı yapılmış *sklearn* kütüphanesinin sunduğu *TfidfVectorizer* sınıfı kullanılarak eğitim verisi üzerinden bir özellik çıkarımı yapılmıştır. Özellik çıkarım işlemi sonrası oluşan kitaplık bir sinir ağı vasıtası ile test verisi kullanılarak test edilmiştir. Modelin doğruluğunu ölçmek *sklearn* kütüphanesinin *metric* eklentisinden yararlanışmış. Modelin doğruluk bilgileri Türkçe için Tablo 4.1'de İngilizce için ise Tablo 4.2'de gösterilmiştir. Doğruluk ve kayıp grafiği ise İngilizce için Şekil 4.1'de ve Şekil 4.2'de, Türkçe için ise Şekil 4.3'de ve Şekil 4.4'de gösterilmiştir.

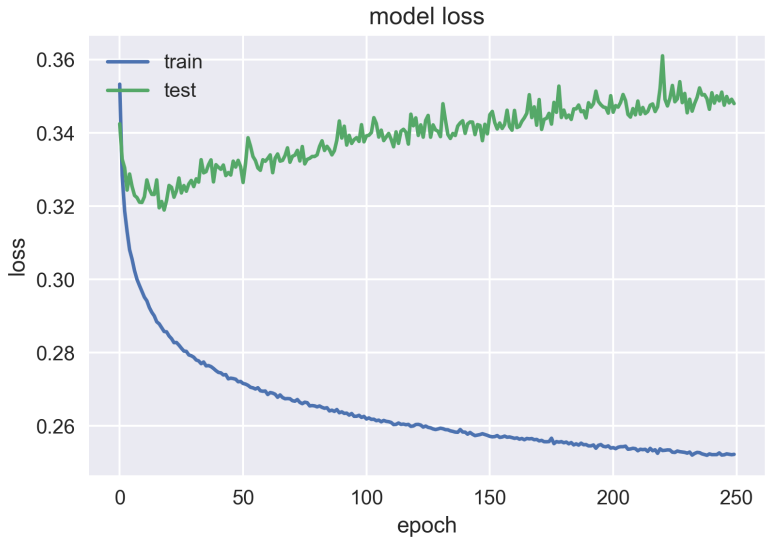


Şekil 4.1 GloVe İngilizce doğruluk grafiği

Şekil 4.2 GloVe İngilizce kayıp grafiği



Şekil 4.3 GloVe Türkçe doğruluk grafiği



Şekil 4.4 GloVe Türkçe kayıp grafiği

Tablo 4.1 Türkçe GloVe modelinin doğruluk bilgileri

GloVe - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
85.10%	91.35%	88.32%	94.59%

Çapraz Doğrulama

Doğruluk	Standart Sapma
85.24%	0.24%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
85.42%	0.27%

Tablo 4.2 İngilizce GloVe modelinin doğruluk bilgileri

GloVe - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
78.36%	73.71%	70.17%	77.63%

Çapraz Doğrulama

Doğruluk	Standart Sapma
79.59%	1.29%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
79.43%	1.64%

4.2. Word2vec

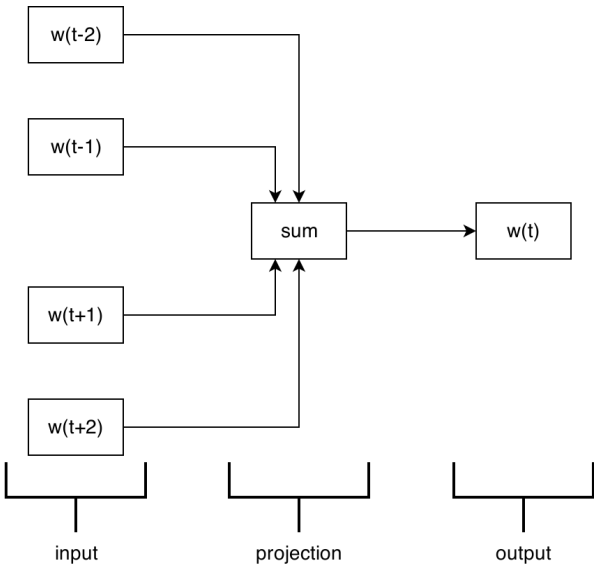
Word2vec, kelime yerleştirme tekniklerinden en popüler olanıdır. Hesaplama açısından oldukça elverişli, zengin kütüphane tercihleri ile uygulaması kolay, iki katmanlı bir yapay sinir ağı kullanan frekansa dayalı olmaktan ziyade bir tahmin modelidir.

Günümüzde, GloVe kelime yerleştirme tekniğinin de kullandığı LSA ve LDA gibi bilinen yöntemlerden, kelimelerin doğru temsil edilebilmesi ve tahmin edilebilmesi için yararlanıldı (Blei, Ng, & Jordan, 2003). Word2vec, bu yöntemleri kullanmak yerine sinir ağları tarafından öğrenilen kelimelerin dağılım temsillerini sağlamakta (Mikolov vd., 2013). Birbirine benzer kelimelerin birlikte ortaya çıkma ihtimalini bir kelime için ağırlıklı olarak tahmin etmek için uygulama detayında CBOW ve skip grams modellerinden yararlanır.

Mikolov vd. (2013), Word2vec ile iki farklı model sunmaktadır. Bu iki farklı model ile hesaplama karmaşıklığı azaltılmaya çalışılmıştır.

CBOW modeli, giriş düğümlerinden, gizli düğümlere (eğer varsa) ve çıkış düğümlerine doğru sadece bir yönde ilerleyen Bengio vd. (2003) ileri beslemeli dil modeline benzemektedir.

Giriş seviyesinde ortadaki kelimeyi doğru bir şekilde sınıflandıran gelecek ve geçmiş olarak dört adet kelime ile doğrusal log sınıflandırıcı oluşturulur. Word2vec CBOW modeli Şekil 4.5'de gösterilmiştir.



Şekil 4.5 Word2vec CBOW modeli

CBOW modelinin eğitim karmaşıklığı Denklem 4.8 ile şu şekilde ifade edilebilir:

$$Q = N * D + D * log_2(V) \tag{4.8}$$

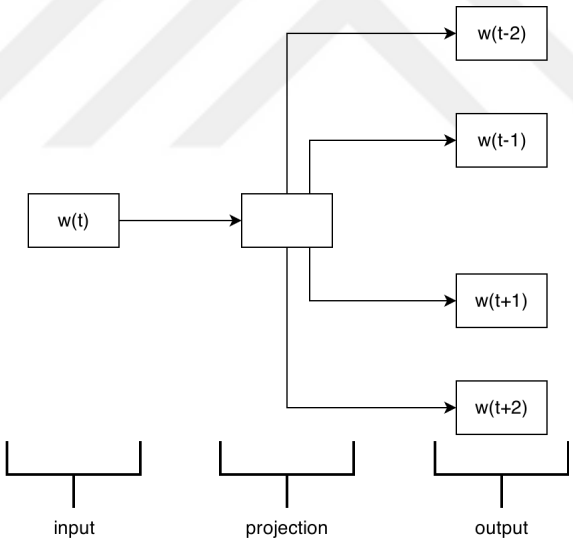
Q her bir ilerideki model mimari için tanımlanır

N giriş katmanındaki kodlanmış bir önceki kelimeler

D vektör uzayının boyutu

V sözlük - kelime büyüklüğü

Devamlı skip-gram modeli ise bir önceki CBOW modeli ile benzerlik göstermektedir. Fakat mevcut kelimeyi içeriğe dayalı olarak tahmin etmek yerine her bir mevcut sözcüğü, sürekli iz düşüm katmanına sahip bir doğrusal log sınıflandırıcısına girdi olarak vererek mevcut sözcükten önce ve sonraki belirli bir aralıktaki sözcükleri tahmin eder. Word2vec devamlı skip-gram modeli Şekil 4.6'da gösterilmiştir.



Şekil 4.6 Word2vec Continuous skip-gram modeli

Devamlı skip-gram modelinin eğitim karmaşıklığı Denklem 4.9 ile şu şekilde ifade edilebilir:

$$Q = C * (D + D * log_2(V)) \tag{4.9}$$

Q her bir ilerideki model mimari için tanımlanır

C her bir kelimenin arasındaki maksimum mesafe

D vektör uzayının boyutu

V sözlük - kelime büyüklüğü

4.2.1 Word2vec uygulaması için ön hazırlık

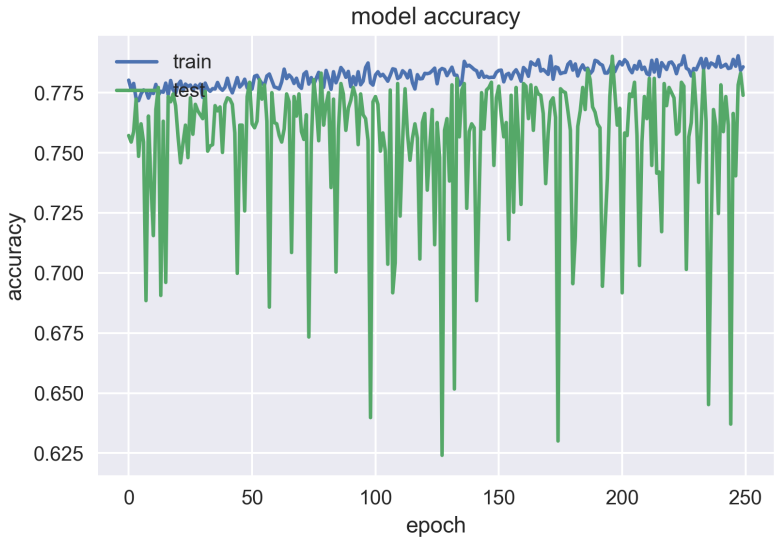
Bir Word2vec modeli oluşturmak için Gensim'in uyguladığı Word2vec modelinden yararlanılmış. Bu uygulama gerçek Word2vec uygulamasına bir arayüz oluşturmaktadır. Böylece Word2vec ile bir Python uygulaması geliştirmek imkanlı hale gelmiştir.

Tez kapsamında önceki bölümlerde temizlenen Twitter verisi öncelikle TextBlob kütüphanesinin belirtkeleme işlevi kullanılarak birer kelime dizi haline getirilmiştir. Her bir Twitter dokümanı bu işlev ile kelimeler dizisi haline getirildikten hemen sonra %80 eğitim ve %20 test verisi olarak ayrıştırılmış ve uygulama için hazır hale getirilmiştir.

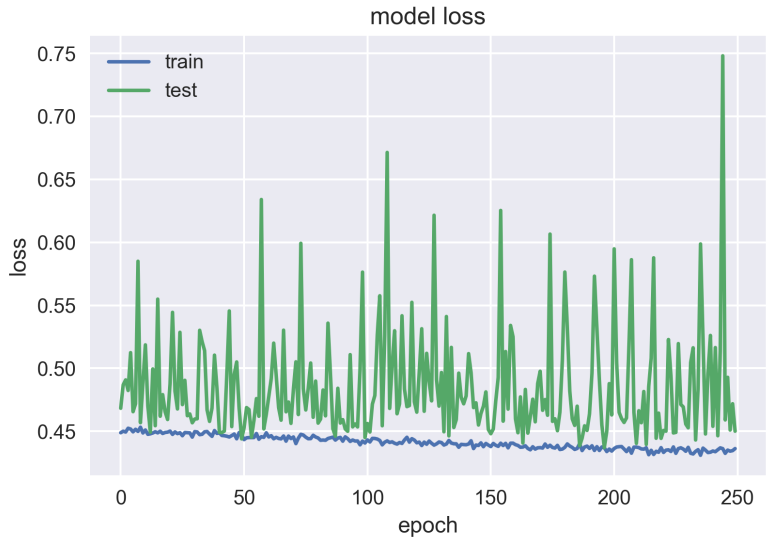
4.2.2. Word2vec uygulaması

Word2vec ile oluşturmak istediğimiz model bir önceki bölümde hazırladığımız eğitim verisi ile eğitilmiştir. Elde edilen modelin ürettiği kelime vektörleri *sklearn* kütüphanesinin sunduğu *TfidfVectorizer* sınıfı kullanılarak bir özellik çıkarımı yapılmıştır. Özellik çıkarım işlemi sonrası oluşan kitaplık bir sinir ağı vasıtası ile test verisi kullanılarak test edilmiştir. Modelin doğruluğunu ölçmek *sklearn* kütüphanesinin *metric* eklentisinden yararlanışmış.

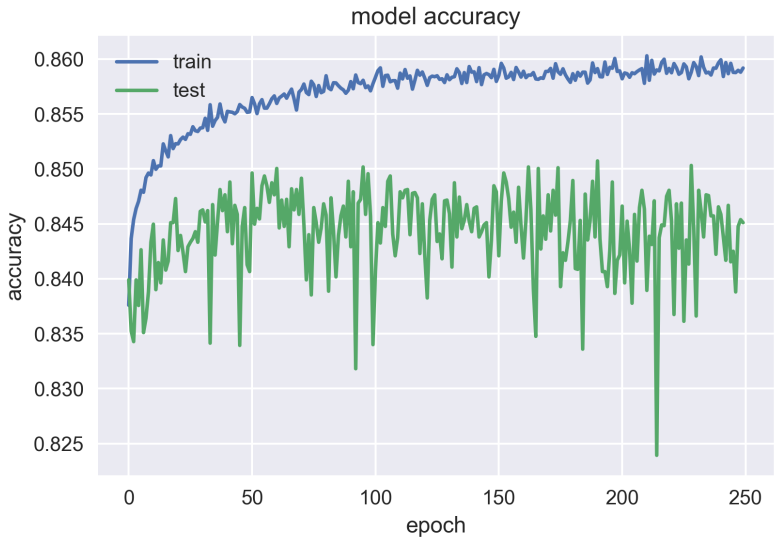
Modelin doğruluk bilgileri Türkçe için Tablo 4.3'de İngilizce için ise Tablo 4.4'de gösterilmiştir. Doğruluk ve kayıt grafiği ise İngilizce için Şekil 4.7'de ve Şekil 4.8'de, Türkçe için ise Şekil 4.9'da ve Şekil 4.10'da gösterilmiştir.



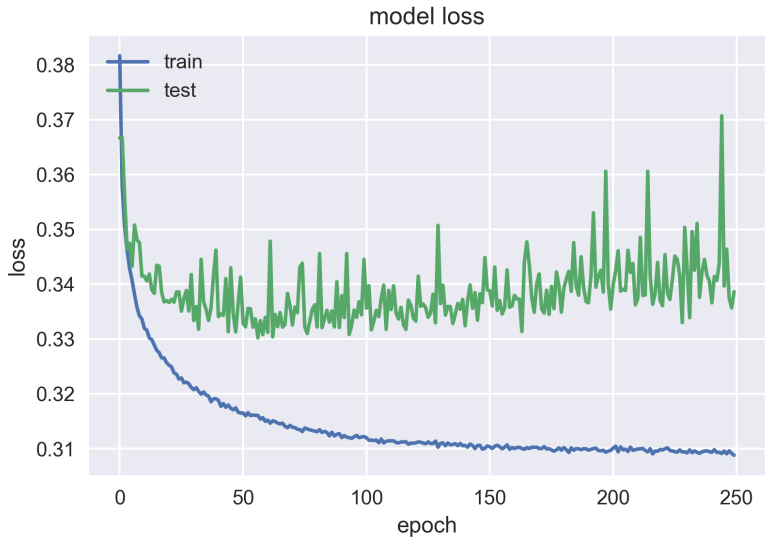
Şekil 4.7 Word2vec İngilizce doğruluk grafiği



Şekil 4.8 Word2vec İngilizce kayıp grafiği



Şekil 4.9 Word2vec Türkçe doğruluk grafiği



Şekil 4.10 Word2vec Türkçe kayıp grafiği

Tablo 4.3 Türkçe Word2vec modelinin doğruluk bilgileri

Word2vec - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
84.59%	91.04%	88.24%	94.03%

Çapraz Doğrulama

Doğruluk	Standart Sapma
83.40%	1.04%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
72.09%	0.79%

Tablo 4.4 İngilizce Word2vec modelinin doğruluk bilgileri

Word2vec - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
73.65%	68.30%	64.44%	72.65%

Çapraz Doğrulama

Doğruluk	Standart Sapma
73.09%	0.68%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
72.25%	0.84%

4.3. fastText

fastText, Word2vec modelinin bir uzantısı olarak Facebook tarafından 2016 yılında geliştirilmiş açık kaynaklı, hızlı ve etkili bir kelime yerleştirme çözümüdür.

fastText, kelimelerin birbirinden bağımsız olduğunu düşünmek yerine, bir kelimenin temsilini hesaplarken tüm karakter olasılıklarını n-gram vektörler yaratarak hesaplar. Ve bunu yaparken vektör gösterimlerini denetimli ve denetimsiz olarak sağlar. fastText, 294 farklı dilin önceden eğitilmiş kelime yerlşetirme verilerini de sunmaktadır.

fastText, etiket tahmini ve duyarlılık analizi olmak üzere iki farklı işlev sunmaktadır (Bojanowski vd., 2017). fastText'in kullandığı model, ortadaki kelimenin bir sınıf ile değiştirilmesi yönünden Mikolov vd. (2013) CBOW modeli ile benzerlik gösterir. Fakat fastText modelinde tanımlı sınıflar üzerinden tanımlanmış olan olasılık dağılımının hesaplanması için, N sayıda doküman için sınıflar üzerinde oluşan negatif log olasığını en aza indiren *softmax*'den yararlanılır. fastText'in hesaplama yöntemi Denklem 4.10 ile ifade edilmiştir.

$$-\frac{1}{N}\sum_{n=1}^{N} y_n log(f(BAx_n)) \tag{4.10}$$

f olasılık dağılımını hesaplayan işlem

N doküman sayısı

x_n n'inci doküman için oluşturulan özellik çantası

y_n sınıf

A matris ağırlığı

B matris ağırlığı

4.3.1. fastText uygulaması için ön hazırlık

Bir fastText modeli oluşturabilmek için verinin ilk olarak beklenilen biçime getirilmesi gerekmektedir. Denetimsiz bir öğrenme modeli amaçlandığından tüm Twitter dokümanlarını bir metin dosyasına aktarılacaktır. Çünkü fastText'in denetimsiz eğitim işlevi girdi olarak bir metin dosyası kullanmaktadır. Bu işlem hem Türkçe hem de İngilizce dokümanlar için uygulanıp hedeflenen yeni modeller için eğitim verisi olarak hazırlanmış olacak.

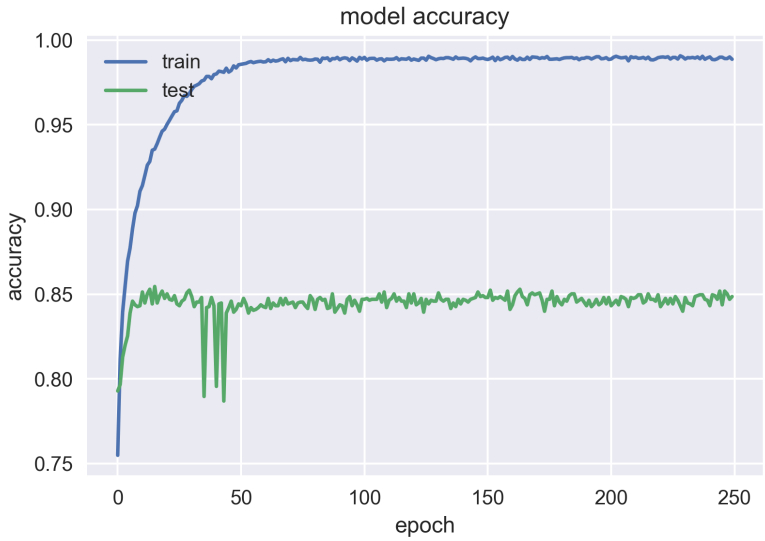
Tüm Twitter dokümanları bir metin dosyasına kayıt edilerek fastText ile eğitilmek üzere uygun hale getirilmiştir.

4.3.2. fastText uygulaması

Bir önce adımda oluşturulan iki farklı metin dosyası eğitim verisi olarak kullanılmak üzere fastText'in sunmuş olduğu denetimsiz veri eğitebilen işlevine gönderilmiş ve her bir dil için (Türkçe ve İngilizce) iki farklı model elde edilmiştir.

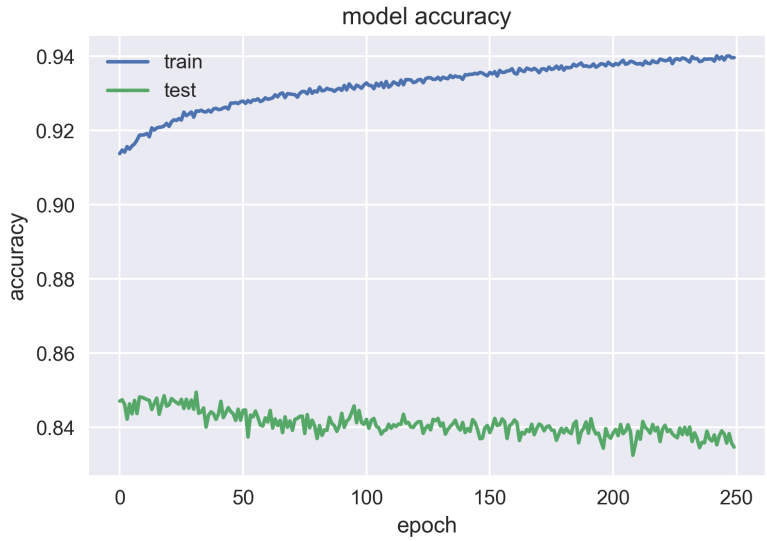
Daha sonra Twitter dokümanları Keras'ın sunmuş olduğu *Tokenizer*, *sequence* ve *np_utils* eklentileri ile eğitim ve test verileri vektörleştirilmiştir ve bir yapay sinir ağı modeline her dil için bir önceki bölümde oluşturulmuş kelime vektörleri girdi olarak kullanılarak ölçev hesabı yapılmıştır.

Modelin doğruluk bilgileri Türkçe için Tablo 4.5'de İngilizce için ise Tablo 4.6'da gösterilmiştir. Doğruluk ve kayıp grafiği ise İngilizce için Şekil 4.11'de ve Şekil 4.12'de, Türkçe için ise Şekil 4.13'de ve Şekil 4.14'de gösterilmiştir.

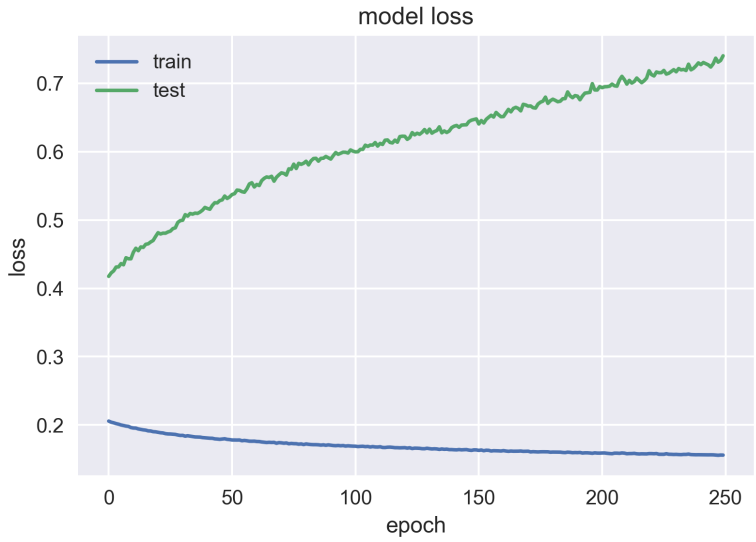


Şekil 4.11 fastText İngilizce doğruluk grafiği

Şekil 4.12 fastText İngilizce kayıp grafiği



Şekil 4.13 fastText Türkçe doğruluk grafiği



Şekil 4.14 fastText Türkçe kayıp grafiği

Tablo 4.5 Türkçe fastText modelinin doğruluk bilgileri

fastText - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
83.69%	90.15%	90.82%	89.49%

Çapraz Doğrulama

Doğruluk	Standart Sapma
84.34%	0.36%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
92.88%	0.23%

Tablo 4.6 İngilizce fastText modelinin doğruluk bilgileri

fastText - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
85.29%	80.98%	81.81%	80.18%

Çapraz Doğrulama

Doğruluk	Standart Sapma
85.75%	0.42%

Tekrarlanan Bekleme

Doğruluk	Standart Sapma
85.39%	0.77%

5. DERİN ÖĞRENME TEKNİKLERİ

Birçok yapay zeka uygulamasının arkasında derin öğrenme modellerinin uygulanıyor olması hiç kimseye garip gelmiyor olmalı. Bulut teknolojisinin gelişmesi, CPU ve özellikle GPU gücünün inanılmaz derecede hızlanması, veri miktarının gün geçtikçe artması derin öğrenme yöntemlerinin neredeyse her alanda uygulanmasının önünü açmıştır. Bu yöntemler büyük miktarda verilere ve hesaplama gücüne ihtiyaç duyduğundan ve hesaplamaların zaman kısıdına duyarlı olarak paralel bir şekilde işlenebilmesi için daha çok GPU gücünden yararlanılmaktadır. Çünkü GPU'lar CPU'lara nazaran daha fazla çekirdek sayısına sahip olduğundan CPU'lara nazaran daha performansı ve hızlı çalışmaktadırlar.

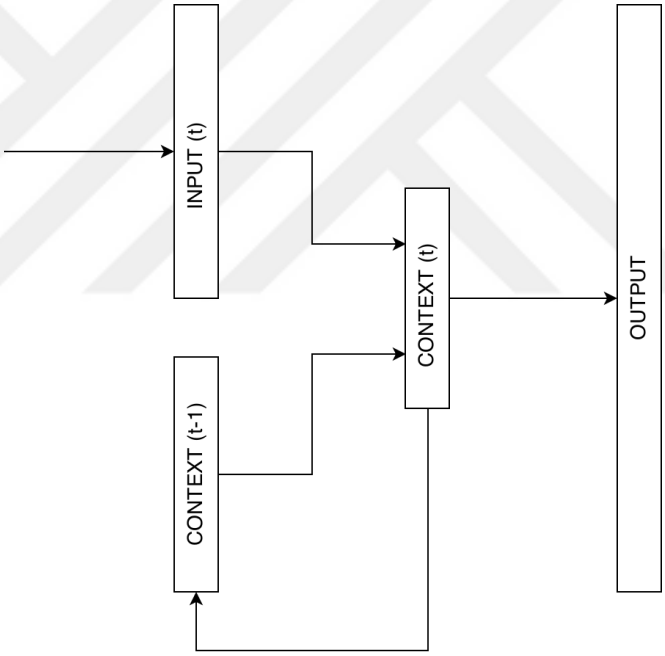
Derin öğrenme, temelde insan beyninin işleme şeklini yani nöron yapısını diğer bir değişle beynin sinir yapısını taklit ederek yeni bir yöntem oluşturma çabası olarak algılanabilir. Bu sebeple derin öğrenme daha çok yapay sinir ağlarının gelişimi sonrası popüler olmuştur denilebilir. Çünkü yukarıda bahsedildiği gibi kullanılan veriler bu güçlü teknolojik destek sayesinde bir veya birden fazla katman içeren bir sinir ağı ile eğitilmektedirler. Derin öğrenme ile birlikte, makine öğrenmesinin bir alt sınıfı olması sebebiyle makine öğrenmesi modellerini için el yordamı ile yapılan daha çok insan müdahalesi ile oluşturulan özellik çıkarım işlemleri ya da parametre tanımlama adımları modelin kendisine yaptırılmaktadır. Bu nedenle derin öğrenme büyük miktarda verileri işleyen ve kendi kendine öğrenerek bir hesaplama modeli çıkaran bir sistem olarak görülebilir.

Bu tez kapsamında Bölüm 5.1'de RNN'den, bölüm 5.2'de CNN'den ve bölüm 5.3'de LSTM algoritmalarından bahsederek her bir kelime yerleştirme yönteminin derin öğrenme algoritmalarına birer girdi olarak gönderilerek nasıl uygulandığını inceleyeceğiz.

5.1. RNN

RNN olarak da bilinen tekrarlayan sinir ağları, sıralı yani bir dizi bilgiden yararlanan, ağ üzerindeki bir önceki katmandan gelen çıktıları girdi olarak kullanan bir sinir ağı modelidir (Lipton, Berkowitz, & Elkan, 2015).

Denilebilir ki, tekrarlayan bir sinir ağı dizilerden gelen bir elemanı bir seferde işler ve bundan önce de hangi eleman geldiyse onun değerini bir sonraki adımda hatırlayabilecek şekilde bir hafızada tutar. Yani bu tip sinir ağları gelinen noktaya kadar yapılmış tüm hesapları hafızasında tutar ve bir sonra adımda bunu hatırlayarak bir bilgi çıkarma işlemi yapar. Şekil 5.1'de gösterilen modelin alacağı herhangi bir karar ya da bir çıkarım o ona kadar verilmiş tüm kararların tamamından etkilenebilir (Mikolov vd., 2010).



Şekil 5.1 Basit bir tekrarlayan sinir ağı modeli

Basit bir tekrarlayan ağ kısa süreli belleği içeren aktivasyon geri bildirimlerine sahiptir. Giriş katmanı, gizli katman ve çıktı katmanı için öğrenme yoluyla otomatik adaptasyonu mümkün kılacak bir takım ağırlık matrisleri, durum geçişi ve çıkış fonksiyonlarıdır (Kilimci & Akyokus, 2019). *t* yani durum/gizli katman sadece bir giriş katmanı ile değil aynı zamanda ileri yayılımdan gelen aktivasyon yani *t-1* ile de beslenir (Bodén, 2001).

Basit tekrarlayan sinir ağı modeli adım adım sadeleştirme aşamaları ile Denklem 5.1, Denklem 5.2 ve Denklem 5.3 ile gösterilmiştir (Mikolov vd., 2010):

$$x(t) = w(t) + s(t-1) \tag{5.1}$$

$$s_j(t) = f\left(\sum_i x_i(t) u_{ji}\right) \tag{5.2}$$

$$y_k(t) = g\left(\sum_j s_j(t) v_{kj}\right) \tag{5.3}$$

$f(z)$ bir sigmoid fonksiyonu olduğunda formül Denklem 5.4'deki şekilde ifade edilir:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{5.4}$$

$g(z)$ bir softmax fonksiyonu olduğunda ise formül Denklem 5.5'deki gibi olur:

$$g(z_m) = \frac{e^{z_m}}{\Sigma_k e^{z_k}} \tag{5.5}$$

x girdi katmanı

s gizli katman ya da durum katmanı

y çıktı katmanı

$x(t)$ t zamanda ağa verilen girdi

$y(t)$ çıkış verisi

w kelime ifade eden vektör

v sözlüğün boyutu

5.1.1. RNN ve GloVe

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni RNN modeli için hazır hale getirilmiştir. GloVe kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. GloVe modelinden üretilen vektörler 300 birimlik çıktı vektör uzayı boyutsallığı olan RNN modeline gönderilmiş ve aktivasyon katmanında *relu* ve *softmax*'den yararlanılmıştır.

Ortaya çıkan RNN ve GloVe modeli İngilizce için Şekil 5.2'de, Türkçe için ise Şekil 5.3'de özetlenmiştir.

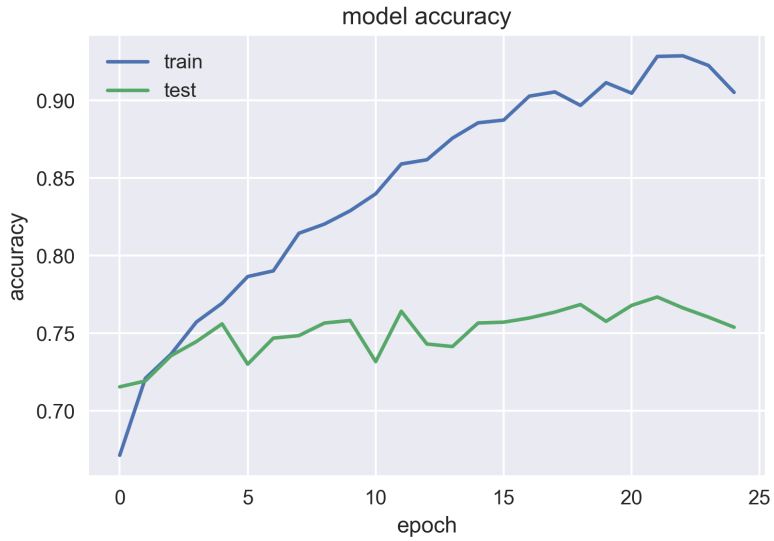
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           2298800
_________________________________________________________________
simple_rnn_2 (SimpleRNN)     (None, 50, 300)           150300
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 300)               0
_________________________________________________________________
dense_3 (Dense)              (None, 8)                 2408
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 18
=================================================================
Total params: 2,451,526
Trainable params: 152,726
Non-trainable params: 2,298,800
_________________________________________________________________
```

Şekil 5.2 İngilizce RNN + GloVe modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           18465400
_________________________________________________________________
simple_rnn_2 (SimpleRNN)     (None, 50, 300)           150300
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 300)               0
_________________________________________________________________
dense_3 (Dense)              (None, 8)                 2408
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 18
=================================================================
Total params: 18,618,126
Trainable params: 152,726
Non-trainable params: 18,465,400
_________________________________________________________________
```

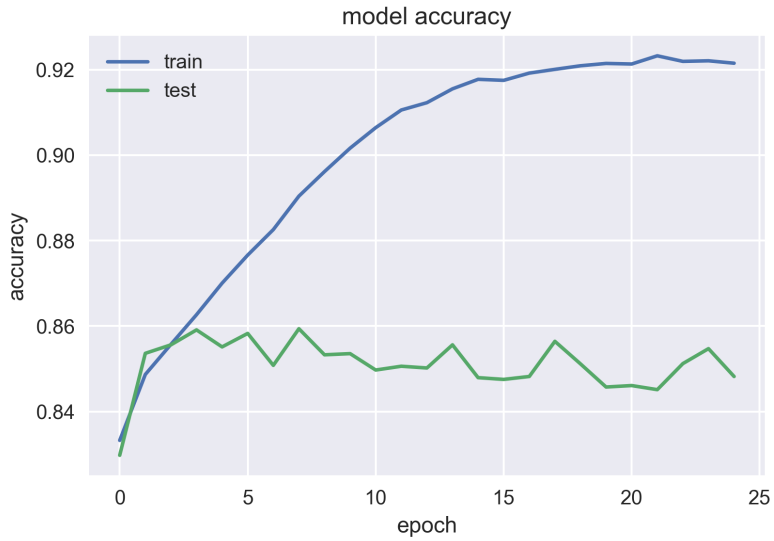
Şekil 5.3 Türkçe RNN + GloVe modelinin özeti

Şekil 5.4'de İngilizce dokümanlar için doğruluk grafiği, Şekil 5.5'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.6'da Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.7'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.1'de İngilizce dokümanların kullanılmasıyla oluşturulan RNN modeli için, Tablo 5.2'de ise Türkçe dokümanlar için oluşturulan RNN modeli için doğruluk bilgileri gösterilmiştir.

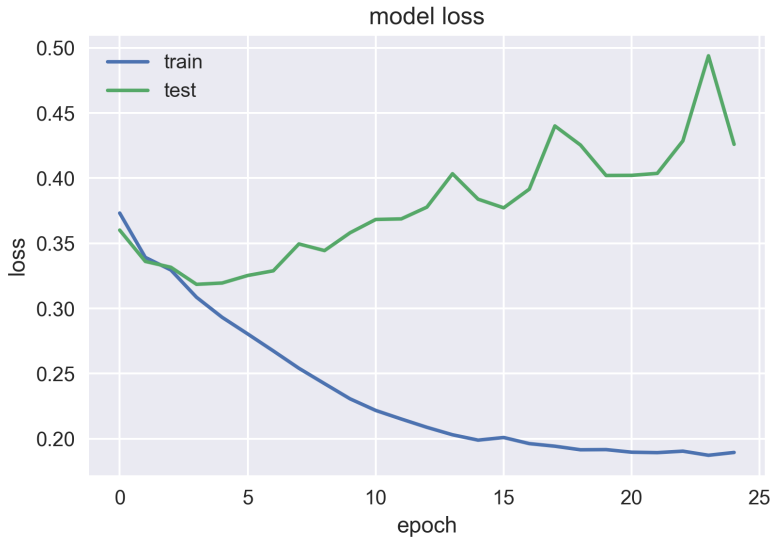


Şekil 5.4 RNN + GloVe İngilizce doğruluk grafiği

Şekil 5.5 RNN + GloVe İngilizce kayıp grafiği



Şekil 5.6 RNN + GloVe Türkçe doğruluk grafiği



Şekil 5.7 RNN + GloVe Türkçe kayıp grafiği

Tablo 5.1 Türkçe RNN + GloVe modelinin doğruluk bilgileri

RNN + GloVe - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
85.31%	91.42%	89.12%	93.85%

Çapraz Doğrulama

Doğruluk	Standart Sapma
85.33%	0.01%

Tablo 5.2 İngilizce RNN + GloVe modelinin doğruluk bilgiler

RNN + GloVe - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
76.68%	64.04%	80.54%	53.16%

Çapraz Doğrulama

Doğruluk	Standart Sapma
77.65%	0.01%

5.1.2. RNN ve Word2vec

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni RNN modeli için hazır hale getirilmiştir. Word2vec kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Daha önce pozitifi ve negatif olarak işaretlenmiş olan veriler kategorik değişkenlere dönüştürülmüştür.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. Word2vec modelinden üretilen vektörler 300 birimlik çıktı vektör uzayı boyutsallığı olan RNN modeline gönderilmiş ve aktivasyon katmanında *relu* ve *softmax*'den yararlanılmıştır.

Ortaya çıkan RNN ve Word2vec modeli İngilizce için Şekil 5.8'de, Türkçe için ise Şekil 5.9'da özetlenmiştir.

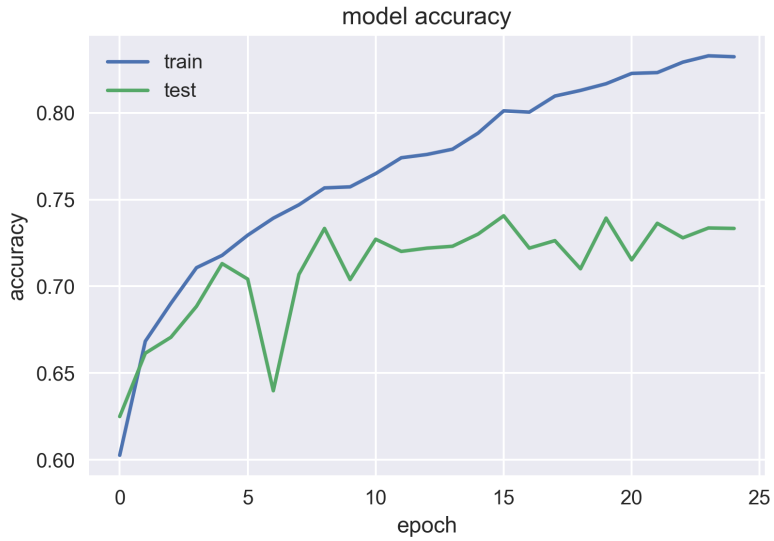
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 44, 200)           282800
_________________________________________________________________
simple_rnn_4 (SimpleRNN)     (None, 44, 300)           150300
_________________________________________________________________
global_max_pooling1d_4 (Glob (None, 300)               0
_________________________________________________________________
dense_7 (Dense)              (None, 8)                 2408
_________________________________________________________________
dense_8 (Dense)              (None, 2)                 18
=================================================================
Total params: 435,526
Trainable params: 152,726
Non-trainable params: 282,800
_________________________________________________________________
```

Şekil 5.8 İngilizce RNN + Word2vec modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 47, 200)           2208200
_________________________________________________________________
simple_rnn_3 (SimpleRNN)     (None, 300)               150300
_________________________________________________________________
dense_5 (Dense)              (None, 32)                9632
_________________________________________________________________
dense_6 (Dense)              (None, 2)                 66
=================================================================
Total params: 2,368,198
Trainable params: 159,998
Non-trainable params: 2,208,200
_________________________________________________________________
```

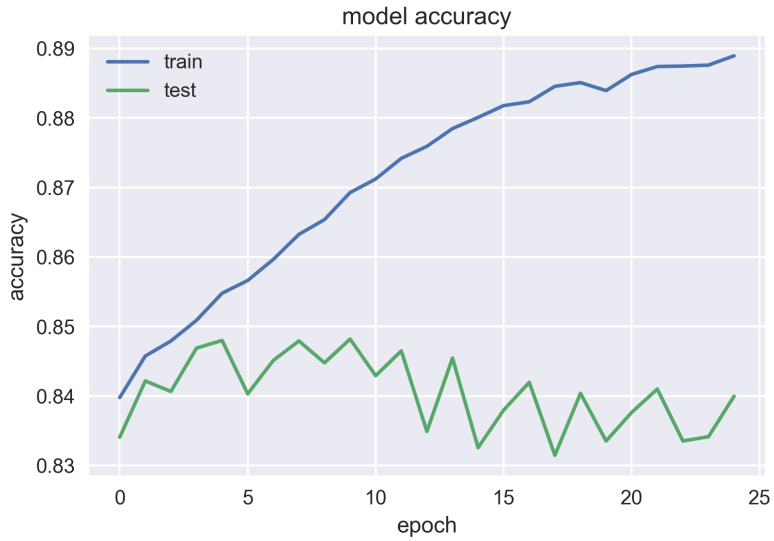
Şekil 5.9 Türkçe RNN + Word2vec modelinin özeti

Şekil 5.10'da İngilizce dokümanlar için doğruluk grafiği, Şekil 5.11'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.12'de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.13'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.3'de İngilizce dokümanların kullanılmasıyla oluşturulan RNN modeli için, Tablo 5.4'de ise Türkçe dokümanlar için oluşturulan RNN modeli için doğruluk bilgileri gösterilmiştir.

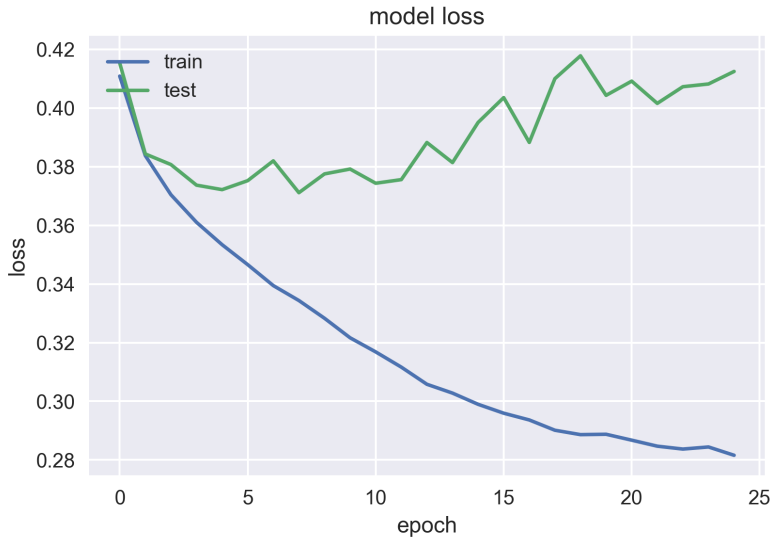


Şekil 5.10 RNN + Word2vec İngilizce doğruluk grafiği

Şekil 5.11 RNN + Word2vec İngilizce kayıp grafiği



Şekil 5.12 RNN + Word2vec Türkçe doğruluk grafiği



Şekil 5.13 RNN + Word2vec Türkçe kayıp grafiği

Tablo 5.3 Türkçe RNN + Word2vec modelinin doğruluk bilgileri

RNN + Word2vec - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
84.45%	91.21%	86.27%	96.75%

Çapraz Doğrulama

Doğruluk	Standart Sapma
84.44%	0.01%

Tablo 5.4 İngilizce RNN + Word2vec modelinin doğruluk bilgiler

RNN + Word2vec - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
75.34%	68.65%	68.20%	69.10%

Çapraz Doğrulama

Doğruluk	Standart Sapma
75.50%	0.02%

5.1.3. RNN ve fastText

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni RNN modeli için hazır hale getirilmiştir. fastText kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Daha önce pozitifi ve negatif olarak işaretlenmiş olan veriler kategorik değişkenlere dönüştürülmüştür. Aynı zamanda eğitim sırasında her güncelleme esnasında girdi birimlerinin bir oranının rastgele olarak ayarlanmasıyla aşırı uyumun önüne geçilmeye çalışılmıştır.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. fastText modelinden üretilen vektörler 300 birimlik çıktı vektör uzayı boyutsallığı olan RNN modeline gönderilmiş ve aktivasyon katmanında *linear* işlevinden yararlanılmıştır.

Ortaya çıkan RNN ve fastText modeli İngilizce için Şekil 5.14'de, Türkçe için ise Şekil 5.15'de özetlenmiştir.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           536800
_________________________________________________________________
simple_rnn_3 (SimpleRNN)     (None, 50, 300)           150300
_________________________________________________________________
dropout_3 (Dropout)          (None, 50, 300)           0
_________________________________________________________________
simple_rnn_4 (SimpleRNN)     (None, 50)                17550
_________________________________________________________________
dropout_4 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 102
=================================================================
Total params: 704,752
Trainable params: 167,952
Non-trainable params: 536,800
_________________________________________________________________
```

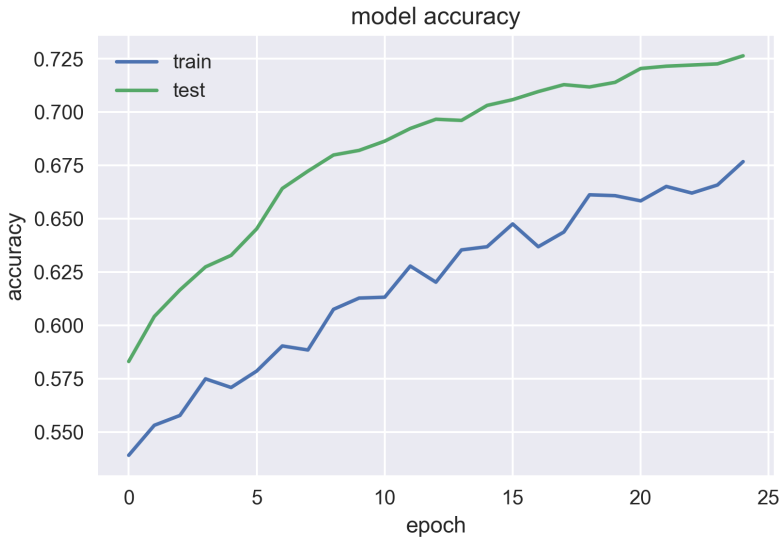
Şekil 5.14 İngilizce RNN + fastText modelinin özeti

Layer (type)	Output Shape	Param #
embedding_3 (Embedding)	(None, 50, 200)	4185600
simple_rnn_5 (SimpleRNN)	(None, 50, 300)	150300
dropout_5 (Dropout)	(None, 50, 300)	0
simple_rnn_6 (SimpleRNN)	(None, 50)	17550
dropout_6 (Dropout)	(None, 50)	0
dense_3 (Dense)	(None, 2)	102

Total params: 4,353,552
Trainable params: 167,952
Non-trainable params: 4,185,600

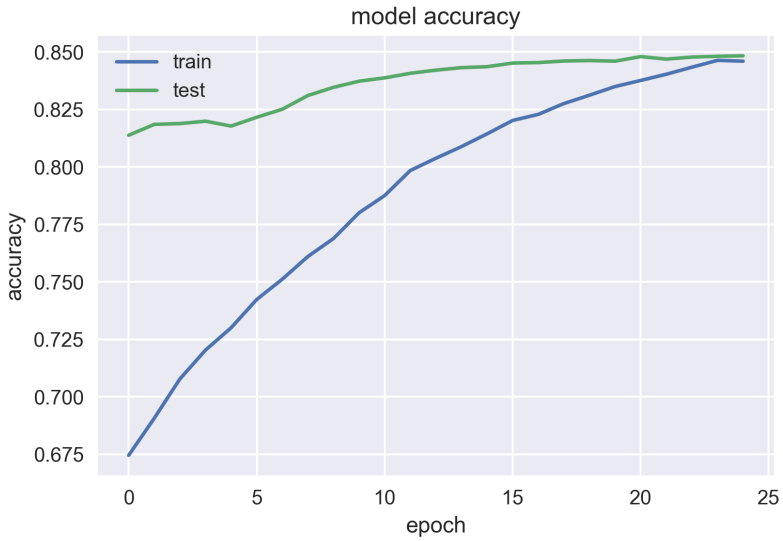
Şekil 5.15 Türkçe RNN + fastText modelinin özeti

Şekil 5.16'da İngilizce dokümanlar için doğruluk grafiği, Şekil 5.17'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.18'de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.1.19'da ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.5'de İngilizce dokümanların kullanılmasıyla oluşturulan RNN modeli için, Tablo 5.6'da ise Türkçe dokümanlar için oluşturulan RNN modeli için doğruluk bilgileri gösterilmiştir.

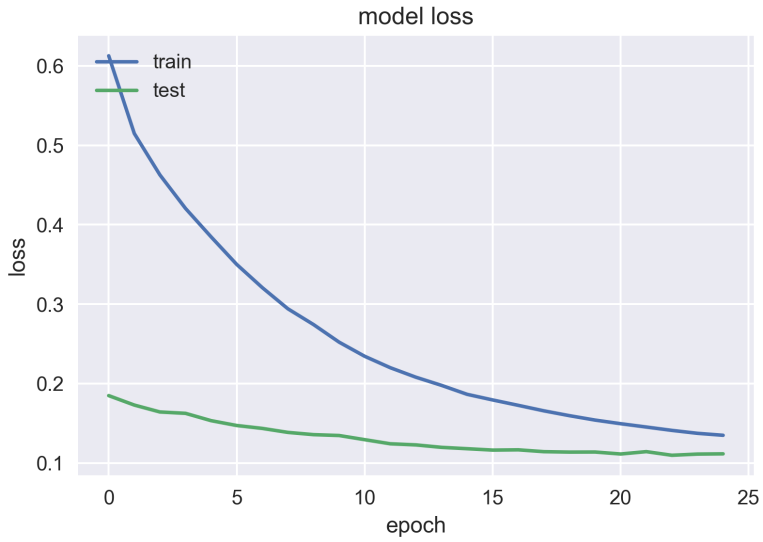


Şekil 5.16 RNN + fastText İngilizce doğruluk grafiği

Şekil 5.17 RNN + fastText İngilizce kayıp grafiği



Şekil 5.18 RNN + fastText Türkçe doğruluk grafiği



Şekil 5.19 RNN + fastText Türkçe kayıp grafiği

Tablo 5.5 Türkçe RNN + fastText modelinin doğruluk bilgileri

RNN + fastText - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
84.05%	91.23%	84.27%	99.44%

Çapraz Doğrulama

Doğruluk	Standart Sapma
83.84%	0.00%

Tablo 5.6 İngilizce RNN + fastText modelinin doğruluk bilgiler

RNN + fastText - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
68.71%	48.47%	68.00%	37.65%

Çapraz Doğrulama

Doğruluk	Standart Sapma
67.69%	0.02%

5.2. CNN

CNN, bir giriş katmanından, bir çıkış katmanından ve bir veya daha fazla evrişimli katmandan oluşan bir sinir ağı modelidir.

Bir CNN modeli, evrişimli, havuzlama ve tam bağlı olmak üzere üç farklı tipte katmanın bir araya gelmesiyle oluşturulur.

Evrişimli katman, eşlemelerin sayısı ve boyutu, çekirdek boyutları, atlama faktörlerini ve bağlantı tablosu tarafından parametrelenen ve bu yüklü hesaplama işinden sorumlu olan katmandır. Girdi bu katmana ulaştığında her filtreyi girdinin uzamsal boyutsallığı boyunca bükerek iki boyutlu bir aktivasyon eşleşmesi üretir (Cires vd., 2003; O'Shea & Nash, 2015).

Evrişimli katman, katmandan çıkan verinin karmaşıklığının azaltılması ve optimize edilebilmesi için hiperparametre, derinlik, adım ve sıfır dolgusu kullanır (Cires vd., 2003; O'Shea & Nash, 2015).

Belirli bir hacimde kaç tane nöron olduğu Denklem 5.6'da gösterilen formül ile belirlenir:

$$\frac{W - K + 2P}{S} + 1 \tag{5.6}$$

W çıkış hacminin uzamsal boyutu

K evrişimli katmandaki nöronların çekirdek sayısı

S atlama faktörleri

P sıfır doldurma miktarı

Havuz katmanı ise Denklem 5.7'de gösterilen yöntem ile hesaplama karmaşıklığını azaltmak amacıyla aşırı uyumluluğu kontrol altına alarak modelin boyutluluğunu ve çok parametreliliğini daha da azaltmaya çalışır.

$$\frac{(W - K)}{S} + 1 \tag{5.7}$$

Havuz katmanı, giriş katmanındaki her bir aktivasyon eşleşmesi üzerinde max pooling işlevini kullanarak mekansal olarak yeniden boyutlandırır.

Tam bağlı katman, önceki katmanlara ait olan tüm aktivasyonlara bağlantılara sahip olan nöronları içerir. Evrişimli ve havuzlama katmanında yapılan birçok hesaplamadan sonra sinir ağındaki karar tamamen birbirine bağlı katmanlar verilir.

5.2.1. CNN ve GloVe

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni CNN modeli için hazır hale getirilmiştir. GloVe kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür. Çıkış filtrelerinin sayısı 300 olan iki adet katlamalı katman ile eğitilmiş olup ve *relu* aktivasyon işlevi çağırılmıştır. Aynı zamanda eğitim sırasında her güncelleme esnasında girdi birimlerinin bir oranının rastgele olarak ayarlanmasıyla aşırı uyumun önüne geçilmeye çalışılmıştır.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. GloVe modelinden üretilen vektörler CNN modeline gönderilmiş ve kullanılan iki aktivasyon katmanında ise sırasıyla *relu* ve *sigmoid* kullanılmıştır.

Ortaya çıkan CNN ve GloVe modeli İngilizce için Şekil 5.20'de, Türkçe için ise Şekil 5.21'de özetlenmiştir.

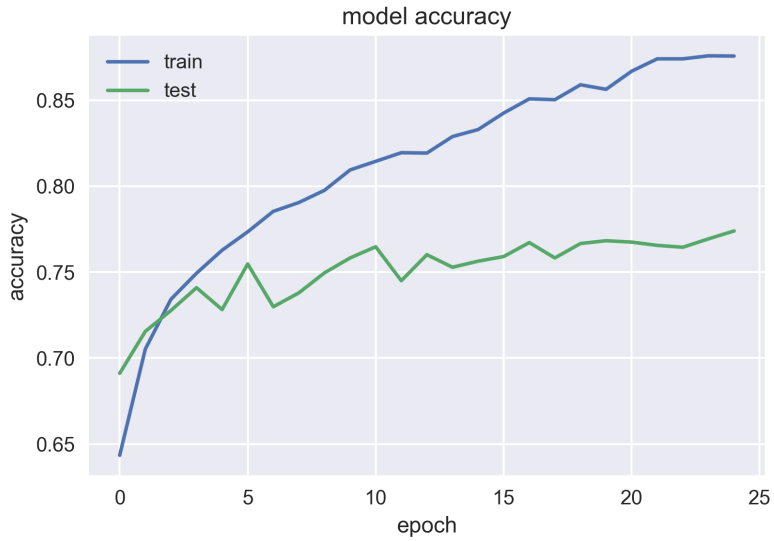
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           2298800
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 50, 64)            89664
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 25, 64)            0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 25, 64)            28736
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 64)                0
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66
=================================================================
Total params: 2,419,346
Trainable params: 120,546
Non-trainable params: 2,298,800
_________________________________________________________________
```

Şekil 5.20 İngilizce CNN + GloVe modelinin özeti

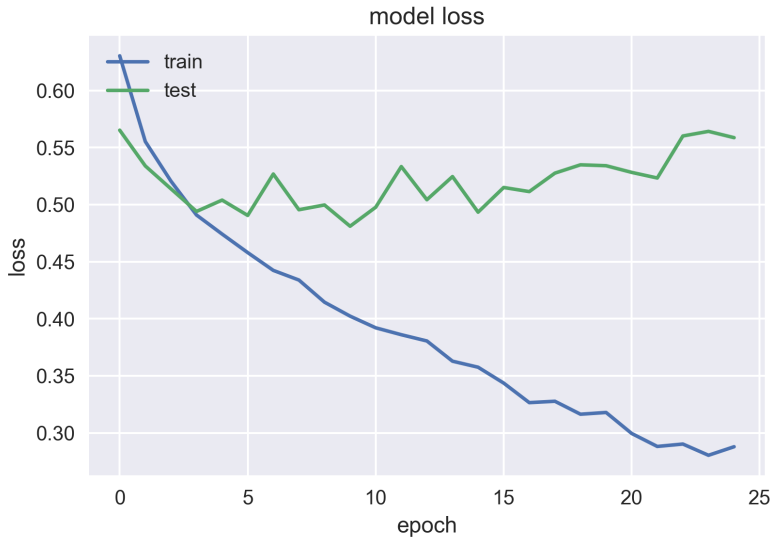
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           18465400
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 50, 64)            89664
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 25, 64)            0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 25, 64)            28736
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 64)                0
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66
=================================================================
Total params: 18,585,946
Trainable params: 120,546
Non-trainable params: 18,465,400
_________________________________________________________________
```

Şekil 5.21 Türkçe CNN + GloVe modelinin özeti

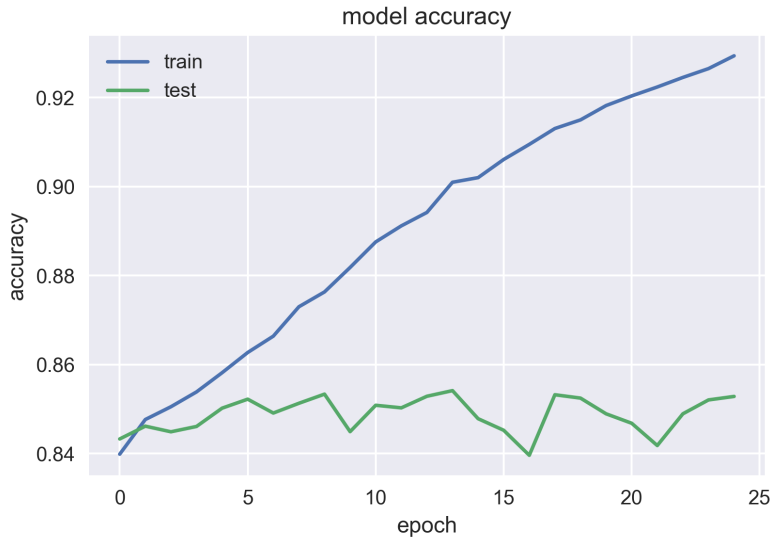
Şekil 5.22’de İngilizce dokümanlar için doğruluk grafiği, Şekil 5.23’de İngilizce dokümanlar için kayıp grafiği, Şekil 5.24’de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.25’de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.7’de İngilizce dokümanların kullanılmasıyla oluşturulan CNN modeli için, Tablo 5.8’de ise Türkçe dokümanlar için oluşturulan CNN modeli için doğruluk bilgileri gösterilmiştir.



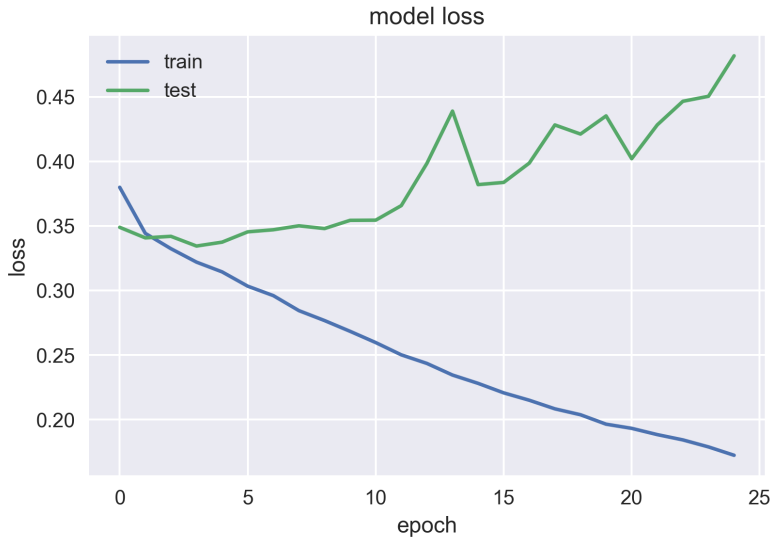
Şekil 5.22 CNN + GloVe İngilizce doğruluk grafiği



Şekil 5.23 CNN + GloVe İngilizce kayıp grafiği



Şekil 5.24 CNN + GloVe Türkçe doğruluk grafiği



Şekil 5.25 CNN + GloVe Türkçe kayıp grafiği

Tablo 5.7 Türkçe CNN + GloVe modelinin doğruluk bilgileri

CNN + GloVe - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
84.88%	91.17%	88.91%	93.54%

Çapraz Doğrulama

Doğruluk	Standart Sapma
85.00%	0.00%

Tablo 5.8 İngilizce CNN + GloVe modelinin doğruluk bilgiler

CNN + GloVe - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
77.85%	70.08%	73.87%	66.67%

Çapraz Doğrulama

Doğruluk	Standart Sapma
76.01%	0.02%

5.2.2. CNN ve Word2vec

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni CNN modeli için hazır hale getirilmiştir. Word2vec kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür. Model iki adet katlamalı katman ile eğitilmiş olup ve *relu* aktivasyon işlevi çağırılmıştır. Aynı zamanda eğitim sırasında her güncelleme esnasında girdi birimlerinin bir oranının rastgele olarak ayarlanmasıyla aşırı uyumun önüne geçilmeye çalışılmıştır.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. Word2vec modelinden üretilen vektörler CNN modeline gönderilmiş ve kullanılan iki aktivasyon katmanında ise sırasıyla *relu* ve *sigmoid* kullanılmıştır.

Ortaya çıkan CNN ve Word2vec modeli İngilizce için Şekil 5.26'da, Türkçe için ise Şekil 5.27'de özetlenmiştir.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 50, 200)           282800
_________________________________________________________________
conv1d_7 (Conv1D)            (None, 50, 32)            44832
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 25, 32)            0
_________________________________________________________________
conv1d_8 (Conv1D)            (None, 25, 64)            14400
_________________________________________________________________
global_max_pooling1d_4 (Glob (None, 64)                0
_________________________________________________________________
dropout_4 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_7 (Dense)              (None, 16)                1040
_________________________________________________________________
dense_8 (Dense)              (None, 2)                 34
=================================================================
Total params: 343,106
Trainable params: 60,306
Non-trainable params: 282,800
_________________________________________________________________
```

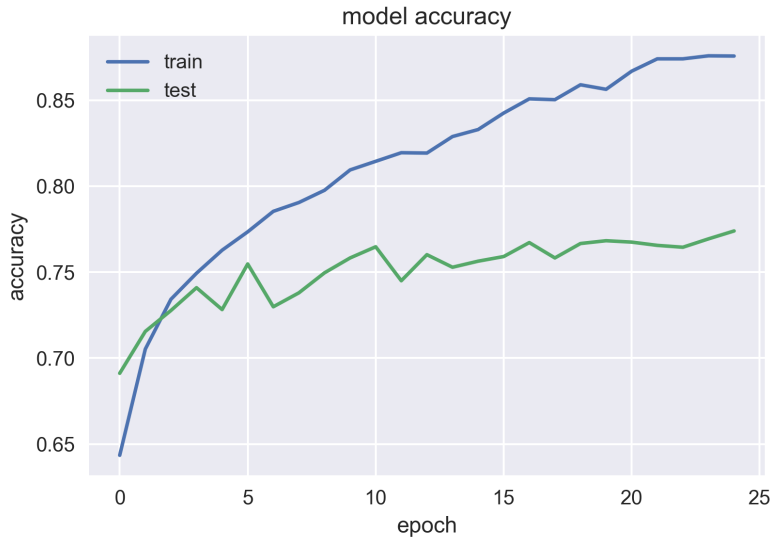
Şekil 5.26 İngilizce CNN + Word2vec modelinin özeti

Layer (type)	Output Shape	Param #
embedding_2 (Embedding)	(None, 200, 200)	2208200
conv1d_3 (Conv1D)	(None, 200, 64)	89664
max_pooling1d_2 (MaxPooling1	(None, 100, 64)	0
conv1d_4 (Conv1D)	(None, 100, 64)	28736
global_max_pooling1d_2 (Glob	(None, 64)	0
dropout_2 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 32)	2080
dense_4 (Dense)	(None, 2)	66

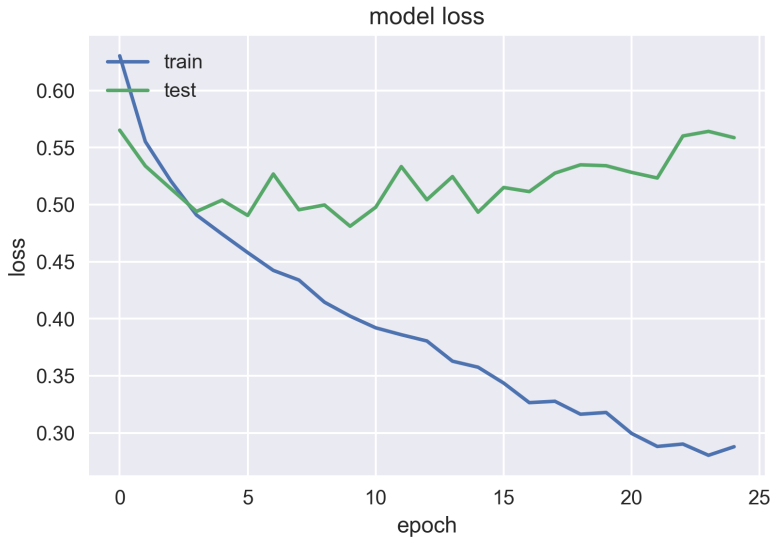
Total params: 2,328,746
Trainable params: 120,546
Non-trainable params: 2,208,200

Şekil 5.27 Türkçe CNN + Word2vec modelinin özeti

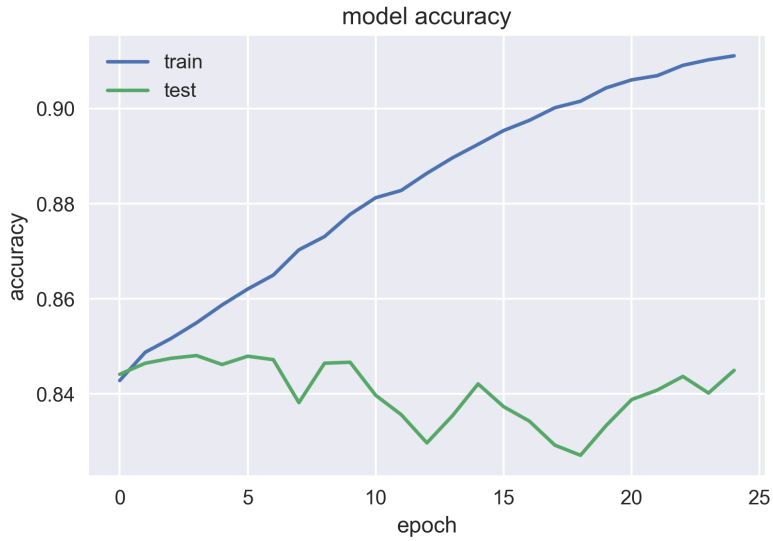
Şekil 5.28'de İngilizce dokümanlar için doğruluk grafiği, Şekil 5.29'da İngilizce dokümanlar için kayıp grafiği, Şekil 5.30'da Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.31'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.9'da İngilizce dokümanların kullanılmasıyla oluşturulan CNN modeli için, Tablo 5.10'da ise Türkçe dokümanlar için oluşturulan CNN modeli için doğruluk bilgileri gösterilmiştir.



Şekil 5.28 CNN + Word2vec İngilizce doğruluk grafiği



Şekil 5.29 CNN + Word2vec İngilizce kayıp grafiği



Şekil 5.30 CNN + Word2vec Türkçe doğruluk grafiği

Şekil 5.31 CNN + Word2vec Türkçe kayıp grafiği

Tablo 5.9 Türkçe CNN + Word2vec modelinin doğruluk bilgileri

CNN + Word2vec - Türkçe			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
84.60%	91.23%	86.87%	96.06%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
83.94%	0.01%

Tablo 5.10 İngilizce CNN + Word2vec modelinin doğruluk bilgiler

CNN + Word2vec - İngilizce			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
70.99%	56.46%	68.18%	48.17%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
71.47%	0.04%

5.2.3. CNN ve fastText

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni CNN modeli için hazır hale getirilmiştir. fastText kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Daha önce pozitifi ve negatif olarak işaretlenmiş olan veriler kategorik değişkenlere dönüştürülmüştür. Model iki adet katlamalı katman ile eğitilmiş olup ve *relu* aktivasyon işlevi çağırılmıştır. Aynı zamanda eğitim sırasında her güncelleme esnasında girdi birimlerinin bir oranının rastgele olarak ayarlanmasıyla aşırı uyumun önüne geçilmeye çalışılmıştır.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. fastText modelinden üretilen vektörler CNN modeline gönderilmiş, aktivasyon katmanında *relu* ve *sigmoid* işlevlerinden yararlanılmıştır.

Ortaya çıkan CNN ve fastText modeli İngilizce için Şekil 5.32'de, Türkçe için ise Şekil 5.33'de özetlenmiştir.

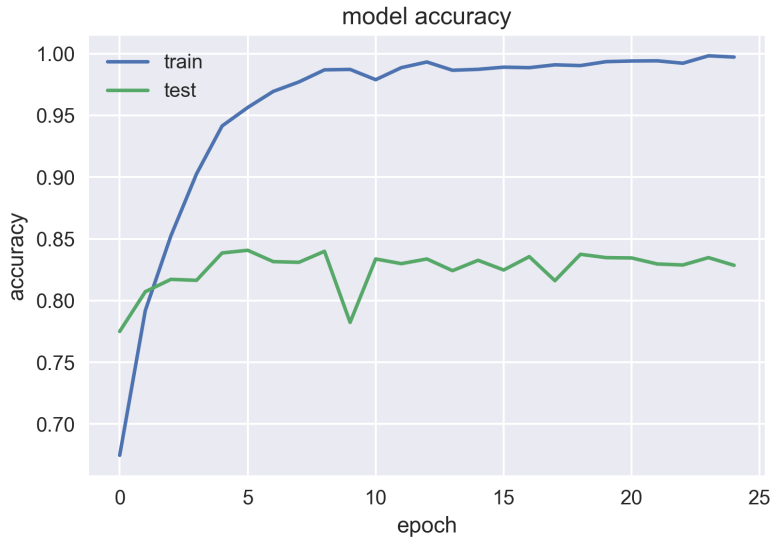
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           536800
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 50, 64)            89664
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 25, 64)            0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 25, 64)            28736
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 64)                0
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66
=================================================================
Total params: 657,346
Trainable params: 120,546
Non-trainable params: 536,800
_________________________________________________________________
```

Şekil 5.32 İngilizce CNN + fastText modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           4185600
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 50, 64)            89664
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 25, 64)            0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 25, 64)            28736
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 64)                0
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66
=================================================================
Total params: 4,306,146
Trainable params: 120,546
Non-trainable params: 4,185,600
_________________________________________________________________
```

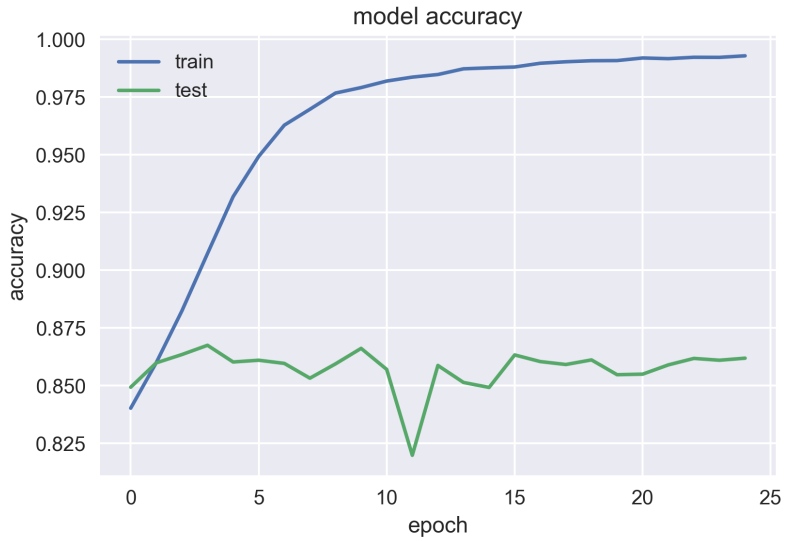
Şekil 5.33 Türkçe CNN + fastText modelinin özeti

Şekil 5.34'de İngilizce dokümanlar için doğruluk grafiği, Şekil 5.35'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.36'da Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.37'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.11'de İngilizce dokümanların kullanılmasıyla oluşturulan CNN modeli için, Tablo 5.12'de ise Türkçe dokümanlar için oluşturulan CNN modeli için doğruluk bilgileri gösterilmiştir.



Şekil 5.34 CNN + fastText İngilizce doğruluk grafiği

Şekil 5.35 CNN + fastText İngilizce kayıp grafiği



Şekil 5.36 CNN + fastText Türkçe doğruluk grafiği

Şekil 5.37 CNN + fastText Türkçe kayıp grafiği

Tablo 5.11 Türkçe CNN + fastText modelinin doğruluk bilgileri

CNN + fastText - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
84.54%	91.45%	84.89%	99.11%

Çapraz Doğrulama

Doğruluk	Standart Sapma
84.48%	0.01%

Tablo 5.12 İngilizce CNN + fastText modelinin doğruluk bilgiler

CNN + fastText - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
77.91%	72.10%	71.24%	72.98%

Çapraz Doğrulama

Doğruluk	Standart Sapma
76.50%	0.01%

5.3. LSTM

Uzun kısa süreli bellek, uzun süreli bağımlılıkları anlayabilen tekrarlı sinir ağı modeli üzerine geliştirilmiş standart beslemeli sinir ağlarının aksine, geri bildirim bağlantıları bulunan bir derin öğrenme modelidir.

LSTM bir RNN gibi çalışıyor olsa da hafıza yinelenen gizli katmanda blokları adı verilen özel bileşenler ile geçit adı verilen birimlerle bilgi akışını kontrol eder ve ağın durumunu kayıt eden hafıza hücreleri barındırır (Sak, Senior, & Beaufays, 2014). Hafıza blokları bir ya da daha fazla hafıza hücrelerinden oluşan ve bu hücrelerin paylaştığı toplamsal ve çarpımsal geçit birimlerinden oluşan temel birimidir (Gers & Schmidhuber, 2001). Her LSTM birimi, hangi bilgi bölümlerinin hatırlanacağını, unutulacağını ve bir sonraki adıma geçeceğini kontrol etmek için giriş, unutma ve çıkış geçitleri içerir (Kilimci & Akyokus, 2019).

Kısaca denilebilir ki bir LSTM ağı, giriş katmanından gelen veri dizisinden ağ aktivasyonlarını hesaplayarak çıkış katmanına veri dizisine iletilmek üzere bir eşleme hesaplar.

Bu hesapları yapan yinelemeli denklem Denklem 5.8, Denklem 5.9, Denklem 5.10, Denklem 5.11, Denklem 5.12 ve Denklem 5.13 adımları ile şu şekilde ifade edilebilir:

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_{t-1} + b_i) \tag{5.8}$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f) \tag{5.9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1} + b_c) \tag{5.10}$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o) \tag{5.11}$$

$$m_t = o_t \odot h(c_t) \tag{5.12}$$

$$y_t = \phi(W_{ym}m_t + b_y) \tag{5.13}$$

W ağırlık matrislerini belirten terim

W_{ix} giriş geçidinden girişe kadar olan ağırlık matrisi

W_{ic}, W_{fc}, W_{oc} geçit katmanlarının hücre durumuna bakmasına izin veren anlamına köşegen ağırlıklı matrisler

b önyargıyı belirten terim

b_i giriş geçidi önyargı vektörü

σ sigmoid işlevi

i, f, o, c giriş geçidi, ihmal geçidi, çıktı geçidi, hücre aktivasyon vektörü

m hücre çıktı aktivasyon vektörü

$\odot$ vektörlerin parça bazlı ürünü

g, h hücre girdisi, hücre çıktısı

ϕ tahn işlevi ya da sigmoid işlevi

5.3.1. LSTM ve GloVe

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni LSTM modeli için hazır hale getirilmiştir. GloVe kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür. Model bir adet iki yönlü ve 300 birimlik çıktı vektör uzayı boyutsallığı olan bir LSTM ağı ile eğitilmiştir.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. GloVe modelinden üretilen vektörler LSTM modeline gönderilmiş ve *softmax* aktivasyon işlevinden yararlanılmıştır.

Ortaya çıkan LSTM ve GloVe modeli İngilizce için Şekil 5.38'de, Türkçe için ise Şekil 5.39'da özetlenmiştir.

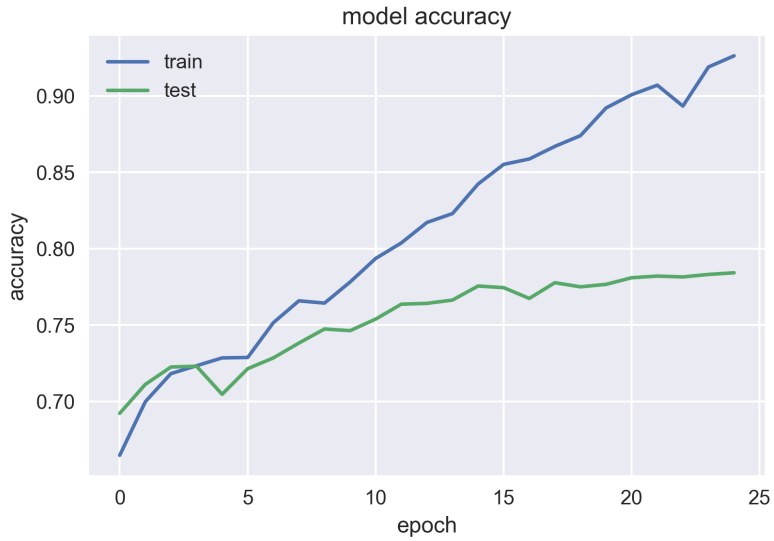
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           2298800
_________________________________________________________________
bidirectional_2 (Bidirection (None, 50, 600)           1202400
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 600)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 1202
=================================================================
Total params: 3,502,402
Trainable params: 1,203,602
Non-trainable params: 2,298,800
_________________________________________________________________
```

Şekil 5.38 İngilizce LSTM + GloVe modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           18465400
_________________________________________________________________
bidirectional_2 (Bidirection (None, 50, 600)           1202400
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 600)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 1202
=================================================================
Total params: 19,669,002
Trainable params: 1,203,602
Non-trainable params: 18,465,400
_________________________________________________________________
```

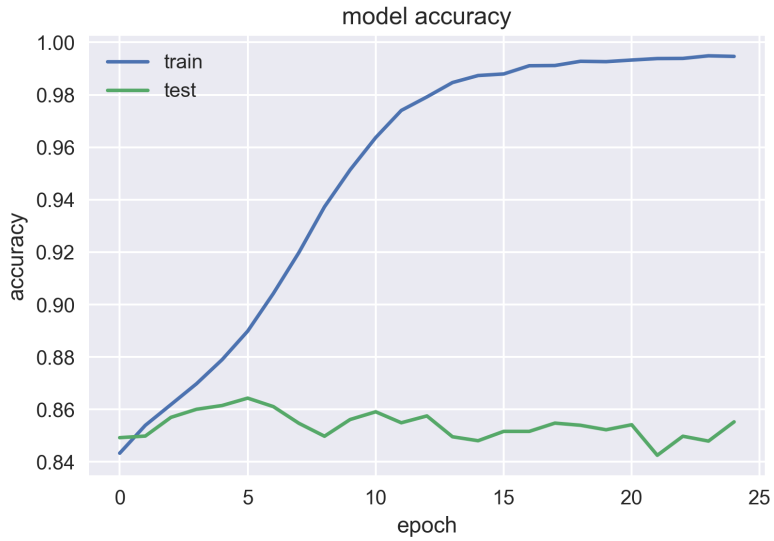
Şekil 5.39 Türkçe LSTM + GloVe modelinin özeti

Şekil 5.40'da İngilizce dokümanlar için doğruluk grafiği, Şekil 5.41'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.42'de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.43'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.13'de İngilizce dokümanların kullanılmasıyla oluşturulan LSTM modeli için, Tablo 5.14'de ise Türkçe dokümanlar için oluşturulan LSTM modeli için doğruluk bilgileri gösterilmiştir.



Şekil 5.40 LSTM + GloVe İngilizce doğruluk grafiği

Şekil 5.41 LSTM + GloVe İngilizce kayıp grafiği



Şekil 5.42 LSTM + GloVe Türkçe doğruluk grafiği

Şekil 5.43 LSTM + GloVe Türkçe kayıp grafiği

Tablo 5.13 Türkçe LSTM + GloVe modelinin doğruluk bilgileri

LSTM + GloVe - Türkçe

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
86.03%	91.85%	89.45%	94.39%

Çapraz Doğrulama

Doğruluk	Standart Sapma
85.97%	0.01%

Tablo 5.14 İngilizce LSTM + GloVe modelinin doğruluk bilgiler

LSTM + GloVe - İngilizce

Doğruluk	F1 Değerlendirme	Kesinlik	Hassasiyet
79.01%	72.83%	73.70%	71.98%

Çapraz Doğrulama

Doğruluk	Standart Sapma
78.46%	0.02%

5.3.2. LSTM ve Word2vec

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni LSTM modeli için hazır hale getirilmiştir. Word2vec kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür. Model bir adet iki yönlü ve 300 birimlik çıktı vektör uzayı boyutsallığı olan bir LSTM ağı ile eğitilmiştir.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. Word2vec modelinden üretilen vektörler LSTM modeline gönderilmiş ve *softmax* aktivasyon işlevinden yararlanılmıştır.

Ortaya çıkan LSTM ve Word2vec modeli İngilizce için Şekil 5.44'de, Türkçe için ise Şekil 5.45'da özetlenmiştir.

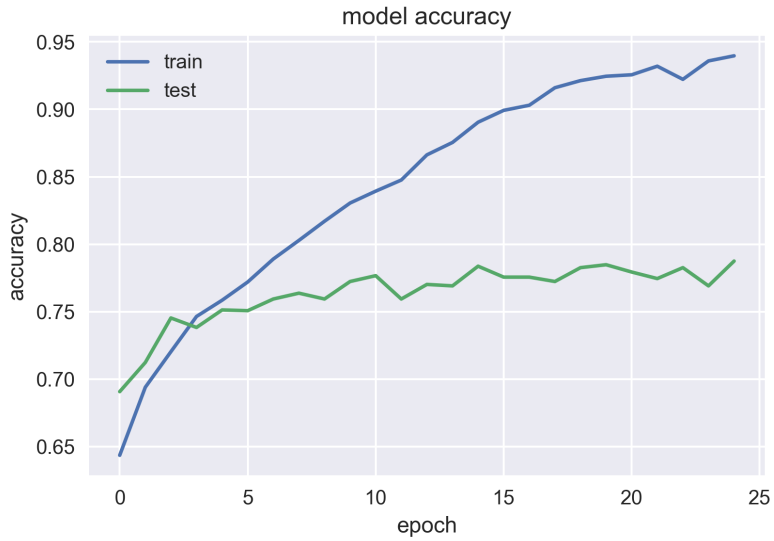
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 55, 200)           282800
_________________________________________________________________
bidirectional_4 (Bidirection (None, 55, 600)           1202400
_________________________________________________________________
global_max_pooling1d_4 (Glob (None, 600)               0
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 1202
=================================================================
Total params: 1,486,402
Trainable params: 1,203,602
Non-trainable params: 282,800
_________________________________________________________________
```

Şekil 5.44 İngilizce LSTM + Word2vec modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 49, 200)           2208200
_________________________________________________________________
bidirectional_3 (Bidirection (None, 49, 600)           1202400
_________________________________________________________________
global_max_pooling1d_3 (Glob (None, 600)               0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 1202
=================================================================
Total params: 3,411,802
Trainable params: 1,203,602
Non-trainable params: 2,208,200
_________________________________________________________________
```

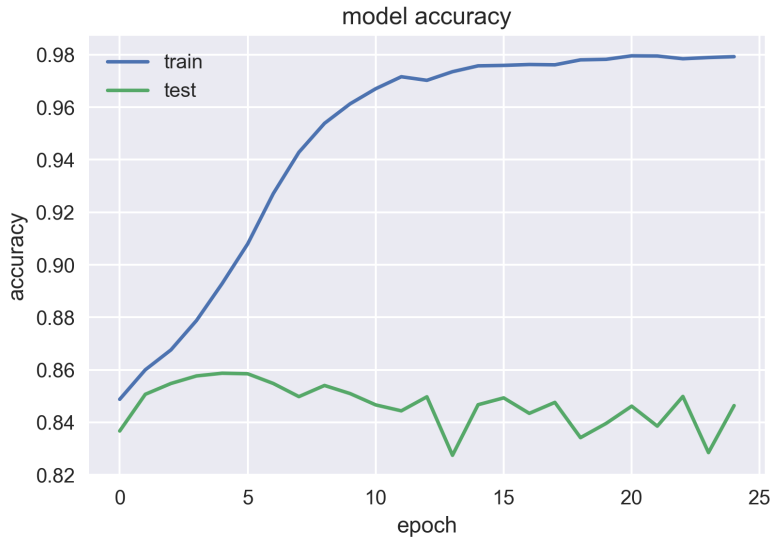
Şekil 5.45 Türkçe LSTM + Word2vec modelinin özeti

Şekil 5.46'da İngilizce dokümanlar için doğruluk grafiği, Şekil 5.47'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.48'de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.49'da ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.15'de İngilizce dokümanların kullanılmasıyla oluşturulan LSTM modeli için, Tablo 5.16'de ise Türkçe dokümanlar için oluşturulan LSTM modeli için doğruluk bilgileri gösterilmiştir.

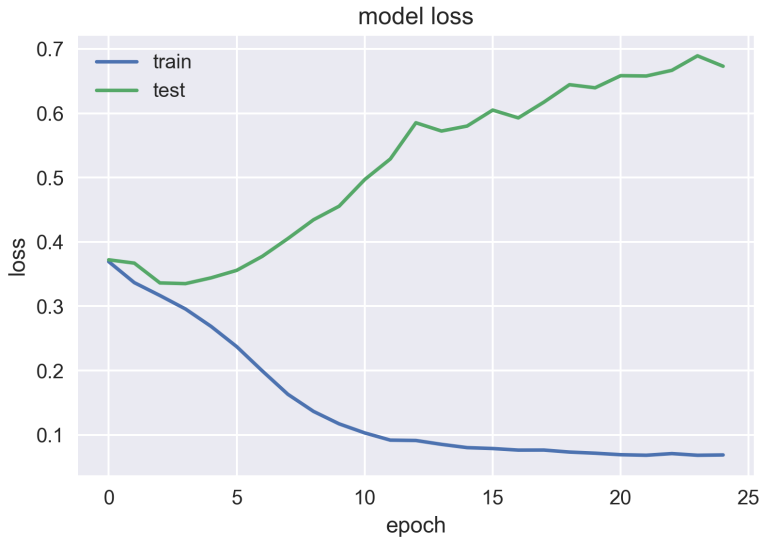


Şekil 5.46 LSTM + Word2vec İngilizce doğruluk grafiği

Şekil 5.47 LSTM + Word2vec İngilizce kayıp grafiği



Şekil 5.48 LSTM + Word2vec Türkçe doğruluk grafiği



Şekil 5.49 LSTM + Word2vec Türkçe kayıp grafiği

Tablo 5.15 Türkçe LSTM + Word2vec modelinin doğruluk bilgileri

LSTM + Word2vec - Türkçe			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
83.80%	90.49%	88.63%	92.43%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
83.43%	0.01%

Tablo 5.16 İngilizce LSTM + Word2vec modelinin doğruluk bilgiler

LSTM + Word2vec - İngilizce			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
74.34%	58.73%	79.03%	46.73%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
76.11%	0.01%

5.3.3. LSTM ve fastText

Üzerinde duygu analizi çalıştırılacak olan ham veri önce çağırılmış daha sonra parçalara ayrılmış ve oluşturulacak olan yeni LSTM modeli için hazır hale getirilmiştir. fastText kullanılarak oluşturulan kelime vektörleri aynı boyutlara getirilecek şekilde diziler haline dönüştürülmüştür. Özellik sayısı 200 olarak belirlenmiş ve dizi boyutu 200 olacak şekilde doldurulmuştur. Aynı zamanda duygu analizi yapabilmek için sınıfların bulunduğu tam sayı içeren vektör ikili sınıf matrisine dönüştürülmüştür. Model bir adet iki yönlü ve 300 birimlik çıktı vektör uzayı boyutsallığı olan bir LSTM ağı ile eğitilmiştir.

Veriler daha sonra %80 eğitim ve %20 test olmak üzere ayrılmıştır. fastText modelinden üretilen vektörler LSTM modeline gönderilmiş ve *softmax* aktivasyon işlevinden yararlanılmıştır.

Ortaya çıkan LSTM ve fastText modeli İngilizce için Şekil 5.50'de, Türkçe için ise Şekil 5.51'de özetlenmiştir.

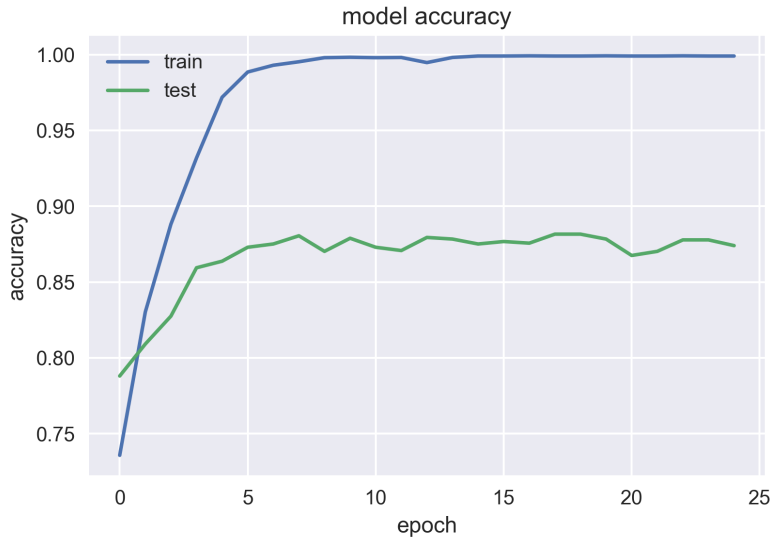
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 50, 200)           536800
_________________________________________________________________
bidirectional_4 (Bidirection (None, 50, 600)           1202400
_________________________________________________________________
global_max_pooling1d_4 (Glob (None, 600)               0
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 1202
=================================================================
Total params: 1,740,402
Trainable params: 1,203,602
Non-trainable params: 536,800
_________________________________________________________________
```

Şekil 5.50 İngilizce LSTM + fastText modelinin özeti

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 50, 200)           4185600
_________________________________________________________________
bidirectional_2 (Bidirection (None, 50, 600)           1202400
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 600)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 1202
=================================================================
Total params: 5,389,202
Trainable params: 1,203,602
Non-trainable params: 4,185,600
_________________________________________________________________
```

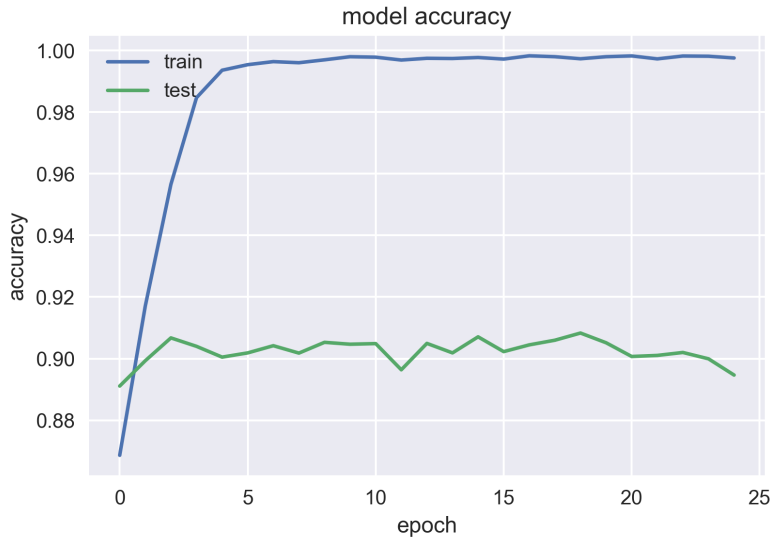
Şekil 5.51 Türkçe LSTM + fastText modelinin özeti

Şekil 5.52'de İngilizce dokümanlar için doğruluk grafiği, Şekil 5.54'de İngilizce dokümanlar için kayıp grafiği, Şekil 5.54'de Türkçe dokümanlar için doğruluk grafiği ve Şekil 5.55'de ise Türkçe dokümanlar için kayıp grafiği gösterilmiştir. Tablo 5.17'de İngilizce dokümanların kullanılmasıyla oluşturulan LSTM modeli için, Tablo 5.18'de ise Türkçe dokümanlar için oluşturulan LSTM modeli için doğruluk bilgileri gösterilmiştir.



Şekil 5.52 LSTM + fastText İngilizce doğruluk grafiği

Şekil 5.53 LSTM + fastText İngilizce kayıp grafiği



Şekil 5.54 LSTM + fastText Türkçe doğruluk grafiği

Şekil 5.55 LSTM + fastText Türkçe kayıp grafiği

Tablo 5.17 Türkçe LSTM + fastText modelinin doğruluk bilgileri

LSTM + fastText - Türkçe			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
83.75%	90.61%	87.50%	93.94%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
83.42%	1.13%

Tablo 5.18 İngilizce LSTM + fastText modelinin doğruluk bilgiler

LSTM + fastText - İngilizce			
Doğruluk	**F1 Değerlendirme**	**Kesinlik**	**Hassasiyet**
76.11%	67.98%	71.38%	64.89%

Çapraz Doğrulama	
Doğruluk	**Standart Sapma**
75.00%	0.01%

6. ZAMAN SERİLERİ ANALİZİ

Zaman serileri analizi, zaman sıralı gelen düzenli verilerden anlamlı bir bilgi çıkarımı yapmak amacıyla geçmiş analiz edildiği ve öngörü modelleri ile gelecek hakkında fikir sahibi olabilmek amacıyla bir takım istatistiki analiz modellerinin uygulanmasıdır.

Zaman sırasıyla elde edilen verilere zaman serisi denir. Yani zaman serisi verileri, verilerin belirli bir zaman dilimlerinde veya aralıklarla toplandığı anlamına gelir.

Sürekli ve ayrık bir zaman dilimi ile toplanan zaman serileri Denklem 6.1'de gösterilmiştir.

$$\{Y_t\} \text{ ya da } \{Y_1,\ Y_2, \dots Y_t\} \tag{6.1}$$

Zaman serileri analizinin asıl amacı; zamanın X değerindeki, Y değişkeni üzerinde gerçekleşen değişiminde bir etkisinin olup olmadığını anlamaktır. Denklem 6.2 ile bir zaman serisi için gerekli olan iki değişkenli regresyon modeli şöyle ifade edilebilir:

$$Y_t = \beta_0 + \beta X_t + u_t \tag{6.2}$$

Y değişken

t zaman birimi

Y_t Y'nin t zamanda almış olduğu değer

6.1. Döviz Kuru Veri Kümesi Kaynağı Olarak Merkez Bankası

Türkiye Cumhuriyeti Merkez Bankası beş farklı görev ve sorumluluğu bulunan temel olarak ülkemizdeki para ve kur politikalarının yönetilmesinden sorumlu olan kurumdur.

Ekonomik kararlar alınırken etkisi olmayacak ölçüde enflasyon oranı vasıtasıyla fiyat istikrarı sağlamak, para ve döviz piyasaları ile ilgili düzenleyici önlemler alarak finansal istikrar amaçlamak, altın ve döviz rezervlerinin korunması, ülkenin menfaatleri adına doğru bir şekilde kullanılması ve döviz kuru rejimin hükümet ile ortak bir şekilde yönetmek, para basmak, fonların ve menkul kıymetlerin güvenli ve hızlı bir şekilde

aktarılması için güvenilir ödeme sistemleri kurmak gibi görevleri vardır (Merkez Bankası Görev ve Sorumlulukları, 2019).

Merkez Bankası'nın para ve döviz piyasaları ile alakalı olan görevlerinden biri de günlük döviz kurlarının paylaşılmasıdır. Merkez Bankası o günün kapanış kurunu 15:30'da belirler ve sitesinden paylaşır. Merkez Bankası Türk Lirası'nın 1 ABD Doları karşısındaki değerini, öncelikle 10:00 – 15:00 arası birer saat arayla 1 ABD doları karşılığında listelenmeye kabul görülen alış-satış fiyat ortalamasını alır, ortaya çıkan sonuç ile bir orta değer hesabı yapılır ve daha sonra bu sonuç üzerinden ıskonto ve prim uygulanarak son alış ve satış kurları belirlenir (Gösterge Niteliğindeki Kurlar, 2019).

6.2. Döviz Kuru Veri Kümesinin Toplanması

Türkiye Cumhuriyet Merkez Bankası geriye dönük döviz kurlarını arşivlemekte ve web sayfası aracılığıyla paylaşmaktadır. Mevcut ara yüz vasıtasıyla 01-01-2018 ve 31-12-2018 tarih aralığına ait tüm döviz kurları bir Python uygulaması aracılıyla ziyaret edilerek toplanmıştır (Gösterge Niteliğindeki Merkez Bankası Kurları, 2019).

Uygulama öncelikle 01 Ocak 2018 tarihinden 31 Aralık 2018 tarihine kadar olan aralığı tek tek Merkez Bankası'nın kur sayfasını ziyaret ederek alır ve içeriği derleyerek Amerikan Doları'na ait olan ve o gün 15:30 itibari olan paylaşılmış olan tüm değerlerini MongoDB doküman veri tabanına kayıt eder.

Türkiye Cumhuriyeti Merkez Bankası, hafta sonları ve resmi tatil olan günlerde veri paylaşmamaktadır. Bu sebeple tüm kurlar alındıktan ve veri tabanına kayıt edildikten sonra kayıp verilerin tamamlanması için bir dizi işlem uygulanmıştır. Bu hazırlık aşaması bir sonraki bölümde tez kapsamında detaylandırılmıştır.

6.3. Döviz Kuru Veri Kümesinin Zaman Serisi Analizi İçin Hazırlanması

Bir önceki bölümde yapılan veri kümesi toplama uygulama sonrasında görülmüştür ki X sayıda kayıp satır, yani X gün kayıp döviz kuru görüntülenmiştir. Toplanan veri kümesi Tablo 6.1'de gösterilmiştir.

Tablo 6.1 Merkez Bankası'ndan alınan toplam veri kümesi değerleri

Toplanan Gün Sayısı	Kayıp Gün Sayısı	Toplam
249	116	365

Çoğu makine öğrenmesi modeli eksik değerlerle çalışmaya elverişli olmadığı için ve bu modellerin etkili ve uygun bir şekilde kullanılabilmesi için zaman serisinin sürekli olması gerekir. Bu sorunun önüne geçmek için kayıp değerlerin uygun veriler ile doldurulması ya da kayıp verilerin olduğu satırların silinmesi gerekmektedir. Tez kapsamında kayıp verilerin bulunduğu satırların silinmesi yerine uygun verilerin bulunması sağlanacaktır. Veri kümesi kayıp veriler ile birlikte Şekil 6.1'de gösterilmiştir.

Şekil 6.1 Kayıp verilerle birlikte Merkez Bankası veri kümesi grafiği

Kayıp verilerin ileri doldurma ya da geri doldurma kullanarak doldurmak önyargıya ve modelin varsayım yapmasına neden olacaktır. Bu sebeple kayıp verilerin olduğu tarihler Twitter üzerinden o güne ait olan paylaşılmış döviz kuru bilgileri saat 15:30 itibari ile karşılaştırılmış ve en uygun modelin ikinci dereceden ara değer hesaplama modeli olduğu görüşmüştür.

Eğer bir zaman serileri analizi için yönelim, sezonsallık ve uzun vadeli döngüler tahmin edilebiliyorsa iki bilinen değer arasında bilinmeyen bir değerin hesaplanması işlemi nispeten kolaydır (VandeBogert, 2019).

Verilen üç farklı veri noktası $(x_0,\ y_0), (x_1,\ y_1), (x_2,\ y_2)$ için ikinci dereceden ara değer hesaplama modeli Denklem 6.3 ve Denklem 6.4 ile gösterilen denklemi karşılayan bir denklem bulunmalı (Atkinson, 2019; Mathonline, 2019):

$$P_2(x_i) = y_i \tag{6.3}$$

$$i = 0, 1, 2 \tag{6.4}$$

Hesaplanması istenen denklem, Denklem 6.5'de kat sayıları ile birlikte gösterilmiştir.

$$P_2(x) = a_0 + a_1 x + a_2 x^2 \tag{6.5}$$

Bu denklemi noktaların yerine koyulması ile Denkle 6.6'daki halini alır:

$$P_2(x) = y_0 L_0(x) + y_1 L_1(x) + y_2 L_2(x) \tag{6.6}$$

Ve her bir nokta sırasıyla birinci, ikinci ve üçüncü nokta olmak üzere Denklem 6.7'de, Denklem 6.8'de ve Denklem 6.9'da gösterilen hesaplamalar ile bulunur.

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \tag{6.7}$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \tag{6.8}$$

$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \tag{6.9}$$

Tez kapsamında, zaman serileri analizi için kullanılacak olan veri kümesinde kayıp veriler ikinci dereceden ara değer hesaplama modeli doldurulmuş ve analiz için hazır hale getirilmiştir.

6.4. Basit üssel yumuşatma

Basit üssel yumuşatma, kesin bir eğilimi olmayan veya sezonsallığı bulunmaya veriler üzerinde tahmin yapmak üzere kullanılan, bu verilerden en yeni ve en eski

gözlemlere dayanarak üssel olarak azalan ağırlıklar atayan bir üssel yumuşatma metodudur (Hyndman & Athanasopoulos, 2019).

Basit üssel yumuşatma üssel olarak azalan ağırlıklar ile ağırlıklı hareketli bir ortalama kullandığından kısa vadeli tahminler için elverişlidir ve bu tekniği kullanan uzun vadeli tahminler oldukça güvenilmez olabilir.

Denklem 6.10 ile basit üssel yumuşatma modeli şu şekilde formüle edilebilir (Glen, 2018):

$$S_t = \alpha y_{t-1} + (1-\alpha) S_{t-1} \tag{6.10}$$

α yumuşatma sabiti (0 ile 1 arasında bir değer)

t zaman aralığı

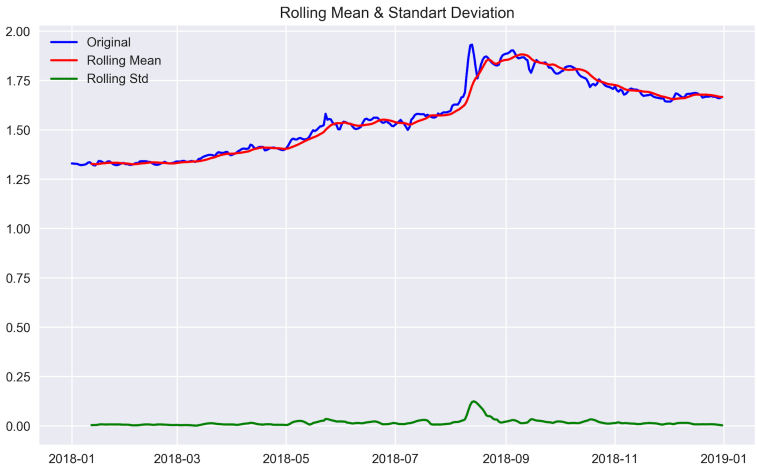
α sıfıra yakın olduğunda yumuşatma işlemi daha yavaş gerçekleşir. α için en iyi değer en küçük ortalama kare hatasının en küçük olduğu halidir.

6.4.1. Basit üssel yumuşatma uygulaması

Merkez Bankası'dan edinilen veri kümesi ve ikinci dereceden ara değer hesaplama modeli ile doldurulmuş şekli ile birlikte bir eğilim tahmini yapılmak üzere log işlevine sokulur. Veri kümesi Şekil 6.2'de gösterilmiştir.

Şekil 6.2 Logaritmik veri üzerinden eğilim tahmini

Daha sonra veri kümesi yuvarlama hesaplamasından ve Dickey-Fuller yönteminden yararlanarak durağanlık testine sokulur. Test sonucu Şekil 6.3'de gösterilmiştir.



Şekil 6.3 Logaritmik veri üzerinde durağanlık testi

Veri kümesine basit üssel yumuşatma modeli dört farklı yumuşatma seviyesi uygulanmıştır. Aralarından en iyisi seçilmiştir. Uygulama grafiği Şekil 6.4'de, yumuşatma sonuçları ise Tablo 6.2'de gösterilmiştir.

Şekil 6.4 Basit üssel yumuşatma uygulamasından sonra oluşan grafik

Tablo 6.2 Farklı yumuşatma seviyeleri basit üssel yumuşatma sonuçları

yumuşatma seviyesi	**0.2**	**0.4**	**0.6**	**0.8**
r2_score	-7.4057	3.9354	2.9672	-2.7397
mean_absolute_percentage_error	4.1383	3.0774	2.7288	2.6432
median_absolute_error	0.0740	0.0552	0.0486	0.0469
mean_absolute_error	0.0693	0.05150	0.0456	0.0442
mean_squared_error	0.0054	0.0032	0.0025	0.0024
mean_squared_log_error	0.0007	0.0004	0.0003	0.0003
accuracy	93.0658	94.8496	95.4342	**95.5774**

6.5. Holt's Linear Trend Modeli

Holt's Linear Trend modeli eğilimin gelecekte sabit, sürekli olarak artıyor veya azalıyor olduğunu varsayan, basit üssel yumuşatma modelinin genişletilmiş halidir. Sezonsallık bulunmayan ama eğilim olan veriler üzerinde uygulanabilir (Holt, 1957).

Holt's Linear Trend modeli Denklem 6.11, Denklem 6.12 ve Denklem 6.13'de gösterilmiştir.

$$\hat{y}_{t+h|t} = \ell_t + hb_t \tag{6.11}$$

$$\ell_t = \alpha_{y_t} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \tag{6.12}$$

$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \tag{6.13}$$

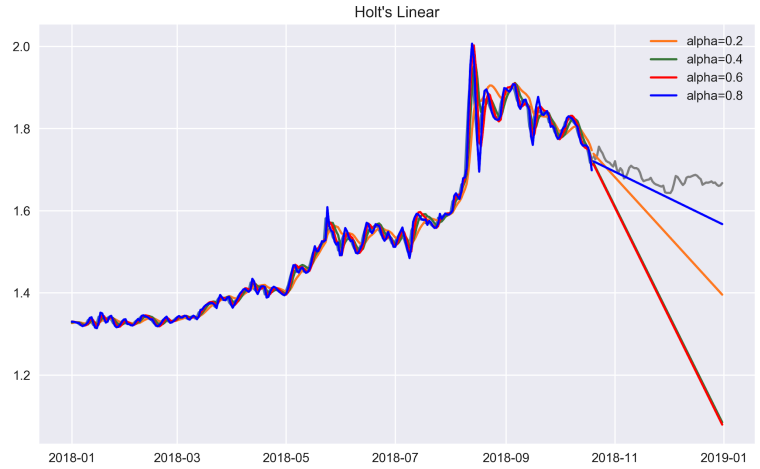
ℓ_t seviye

b_t eğilim

α, β^* yumuşatma parametreleri

6.5.1. Holt's linear trend modeli uygulaması

Veri kümesine Holt's Linear Trend modeli dört farklı yumuşatma eğimi ve dört farklı yumuşatma eğimi ile uygulanmıştır. Aralarından en iyisi seçilmiştir. Uygulama grafiği Şekil 6.5'de, yumuşatma sonuçları ise Tablo 6.3'de gösterilmiştir.



Şekil 6.5 Holt's Linear Trend uygulama sonrası oluşan grafik

Tablo 6.3 Farklı yumuşatma seviyeleri ile oluşan Holt's Linear sonuçları

yumuşatma seviyesi	**0.2**	**0.4**	**0.6**	**0.8**
yumuşatma eğimi	0.1	0.2	0.4	0.8
r2_score	-31.0277	-166.7145	-170.8184	-3.0738
mean_absolute_percentage_error	7.0545	16.9976	17.2445	2.4576
median_absolute_error	0.0950	0.2620	0.2662	0.0278
mean_absolute_error	0.1181	0.2846	0.2888	0.0412
mean_squared_error	0.0207	0.1087	0.1114	0.0026
mean_squared_log_error	0.0031	0.0182	0.0186	0.0003
accuracy	88.1871	71.5300	71.1151	**95.8710**

6.6. Holt-Winters Sezonluk Modeli

Diğer bir adıyla da üçlü üssel yumuşatma olan bu metot üç kere üssel yumuşatma uygulanarak elde edilir. Hatta, Holt's Linear Trend modeline sezonsal bir bileşen eklemesi sebebiyle bu modeli Holt's Linear Trend modelinin genişletilmiş hali denilebilir (Winters, 1960).

Bu model ile farklı sezonsallık türleri çarpımsal ve toplamsal olarak iki farklı metot uygular. Eğer sezonsal değişimler zaman serisi boyunca aşağı yukarı sabit ise toplamsal model, sezonsal değişimler zaman serisi boyunca orantılı olarak değişiyorsa çarpımsal model tercih edilir.

Holt-Winters toplamsal metodu Denklem 6.14, Denklem 6.15, Denklem 6.16 ve Denklem 6.17 adımları ile şu şekilde ifade edilebilir:

$$\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)} \tag{6.14}$$

$$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \tag{6.15}$$

$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \tag{6.16}$$

$$s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} \tag{6.17}$$

Holt-Winters çarpımsal metodu Denklem 6.18, Denklem 6.19, Denklem 6.20 ve Denkle 6.21 adımları ile şu şekilde ifade edilebilir:

$$\hat{y}_{t+h|t} = (\ell_t + hb_t)s_{t+h-m(k+1)} \tag{6.18}$$

$$\ell_t = \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \tag{6.19}$$

$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \tag{6.20}$$

$$s_t = \gamma\frac{y_t}{(\ell_{t-1} - b_{t-1})} + (1-\gamma)s_{t-m} \tag{6.21}$$

ℓ_t seviye

b_t eğilim

s_t sezonsallık

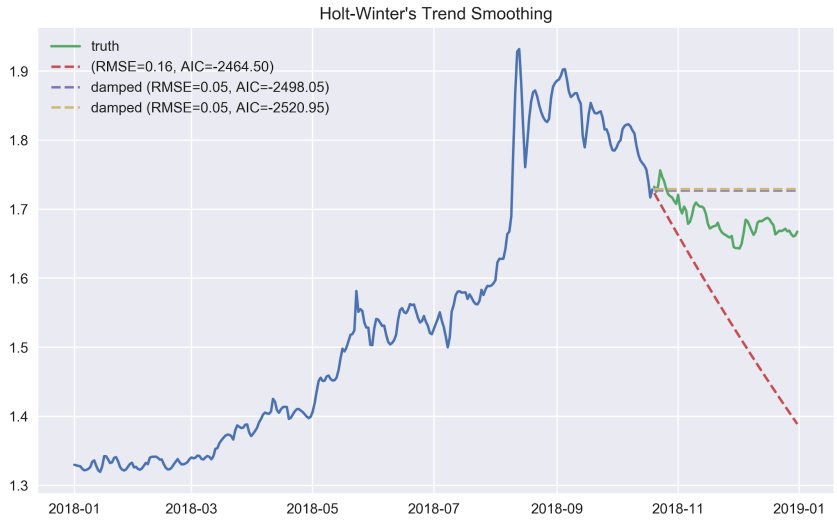
α, β^*, γ yumuşatma parametreleri

m sezonsallık sıklığı

6.6.1. Holt-Winters sezonluk modeli uygulaması

Holt-Winters modeli çarpımsal ve toplamsal metotları ile 0.1, 0.2 ve 0.4 olmak üzere üç farklı yumuşatma eğimi değeri ile denenmiştir. Yumuşatma eğim değeri 0.2 ve 0.4 olan modellere aynı zamanda eğilimin gelecekte sabit bir değere yakınlaşması için bir parametre ekleyen damp yöntemi uygulanıp sonuçlar incelenmiştir.

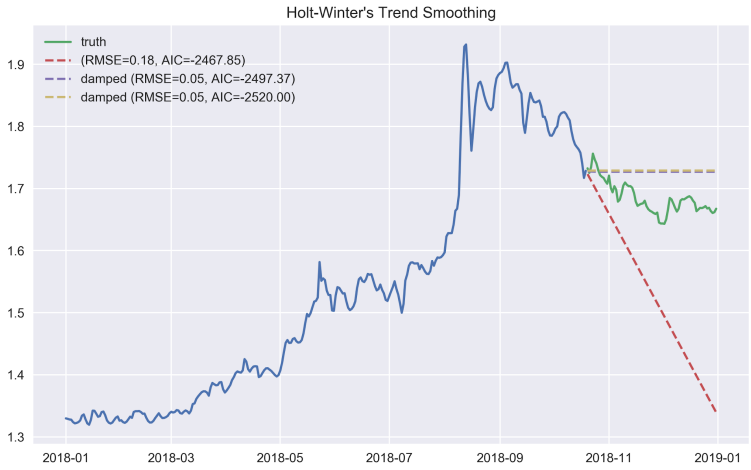
Şekil 6.6'da çarpımsal metot grafiği ve Tablo 6.4'de çarpımsal metot uygulama sonuçları, Şekil 6.7'de toplamsal metot grafiği ve Tablo 6.5'de de toplamsal metot uygulama sonuçları gösterilmiştir.



Şekil 6.6 Holt-Winters çarpımsal uygulanmasından sonra oluşan grafik

Tablo 6.4 Holt-Winters çarpımsal uygulamasıda sonra oluşan sonuçlar

yumuşatma seviyesi	**0.1**	**0.2**	**0.4**
r2_score	-37.0198	-2.7133	-2.9642
mean_absolute_percentage_error	8.0524	2.6325	2.7279
median_absolute_error	0.1154	0.0467	0.0486
mean_absolute_error	0.1349	0.0440	0.0456
mean_squared_error	0.0246	0.0024	0.0025
mean_squared_log_error	0.0037	0.0003	0.0003
accuracy	86.5060	**95.5954**	95.4356



Şekil 6.7 Holt-Winters toplamsal uygulaması sonra oluşan grafik

Tablo 6.5 Holt-Winters toplamsal uygulanması sonrasında oluşan sonuçlar

yumuşatma seviyesi	**0.1**	**0.2**	**0.4**
r2_score	-49.3854	-2.7000	-2.9487
mean_absolute_percentage_error	9.1836	2.6273	2.7221
median_absolute_error	0.1312	0.0466	0.0485
mean_absolute_error	0.1538	0.0439	0.0455
mean_squared_error	0.0326	0.0023	0.0025
mean_squared_log_error	0.0050	0.0003	0.0003
accuracy	84.6145	**95.6041**	95.4453

6.7. ARIMA

Zaman serileri analizi ve tahmini yaparken en çok kullanılan otoregresif hareketli ortalama (ARMA) modelinin genelleştirilmiş hali olan otoregresif entegre hareketli ortalama (ARIMA) modelidir. Bahsi geçen bu her iki model de zaman serilerini daha iyi anlamak için veya zaman serilerindeki gelecekteki noktaları tahmin etmek için kullanılmaktadır.

ARIMA modeli otoregresif (AR) ve hareketli ortalama (MA) modellerini birleştirip entegrasyon (I) denilen yeni bir ön işleme aşaması ile zaman serilerini durağan yapmaya çalışmaktadır (Box vd., 2016). Otoregresif entegre hareketli ortalama modelinde, bir değişkenin gelecekteki değerinin, geçmiş gözlemlerin ve rastgele hataların doğrusal bir fonksiyonu olduğu kabul edilir (Zhang G. P., 2003).

ARIMA modeli, zaman serisinin gecikmeli değerlerinin ağırlıklı toplamı olarak gösterilen *p* değeri yani *AR*, zaman serisinin gecikmeli tahmin edilen hatalarının ağırlıklı toplamı olarak gösterilen *q* yani *MA* ve zaman serisinin gecikmeli tahmin edilen hatalarının ağırlıklı toplamı olarak gösterilen *d* yani *I* olmak üzere üç farklı adımdan oluşmaktadır.

Yukarıdaki bilgilere dayanarak ARIMA modelinin notasyonu Denklem 6.22 ile şu şekilde ifade edilebilir:

$$ARIMA(p, d, q) \tag{6.22}$$

Denklem 6.23 ile otoregresif modelini ifade eden *AR(p)* notasyonu şu şekilde ifade edilebilir:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t \tag{6.23}$$

Denklem 6.24 ile hareketli ortalama modelini ifade eden *MA(q)* notasyonu ise şu şekilde ifade edilebilir:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \, \varepsilon_{t-i} \tag{6.24}$$

c model tarafından tahmin edilen kesme parametresi

φ model tarafından tahmin edilen gecikme kat sayısı

θ model tarafından tahmin edilen gecikme kat sayısı

μ X_t'nin beklediği değer

X gecikme

ε rastgele tanımlanan hata parametreleri

6.7.1. ARIMA modeli için veri kümesinin hazırlanması

İlk olarak veri kümesinin nasıl göründüğünü anlamak için 365 günlük döviz kuru verisini bir grafik vasıtasıyla Şekil 6.8'de gösterilmiştir. Grafiğe bakıldığında veri kümesinin durağan olmadığı görülmektedir.

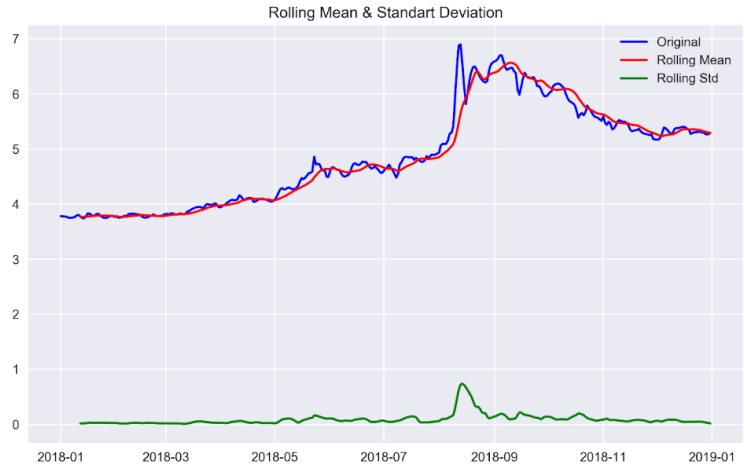
Şekil 6.8 Merkez Bankası döviz kuru grafiği

Bir veri kümesine zaman serileri analizi uygulanabilmesi için veri kümesinin durağan olması gerekmektedir. Yani ortalama ve standart sapma verilerinin zaman bağlı olarak değişmemesi gerekmektedir. Bir veri kümesine durağanlık testi uygulamanın iki farklı yolu vardır. Biri yuvarlama hesaplaması ve diğeri de Dickey-Fuller yöntemidir.

Başka yollar uygulanarak tespit edilmesi zor olan eğilimleri bulmak için kullanılan basit bir yuvarlama hesaplaması son *n* sayıdaki değerin ağırlıksız ortalamasıdır. Dickey-

Fuller test zaman serileri analizi uygulanacak olan bir veri kümesinden sonuç alınıp alınamayacağını test eden ve durağan olup olmadığını gösteren bir istatistiki yöntemdir.

Bu kısımda Merkez Bankası'dan toplanan döviz kuru verisinin durağan olup olmadığını anlamak için veri kümesine yuvarlama hesaplaması ve Dickey-Fuller metotları uygulanmış ve test edilmiştir. Yuvarlama hesaplaması sonuçları Şekil 6.9'da, Dickey-Fuller sonuçları ise Tablo 6.6'da gösterilmiştir.



Şekil 6.9 Yuvarlama hesaplaması ve Dickey Fuller test grafiği

Tablo 6.6 Yuvarlama hesaplaması ve Dickey Fuller test sonuçları

Test İstatistikleri	-1.210298		
p-değerleri	0.669150		
Gecikme sayısı	4		
Gözlem sayısı	360		
Kritik değerler	**1%**	**5%**	**10%**
	3.4486	-2.8696	-2.5710

Veri kümesinin durağan olabilmesi için birkaç farklı yöntem daha uygulanıp durağanlık testlerine sokulmuştur. Bir değişkenin zaman içinde nasıl değiştiğini görmek için; özellikle bir eğilimin olup olmadığını görmek için verilerin özelliklerinin incelenmesi için veri kümesinin logu alınmıştır. Logu alınmış olan veri kümesi Şekil 6.10'da gösterilmiştir.

Şekil 6.10 Logu alınmış veri kümesinin oluşturduğu grafik

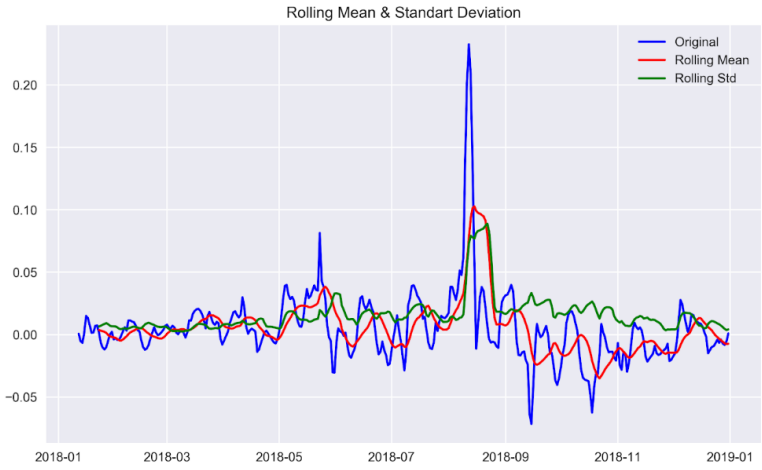
Hareketli ortalama yöntemini kullanarak veri kümesi tekrar durağanlık testine sokulmuştur. Sonuçlara bakarak denilebilir ki ortalama değeri sabit olmadığından veri kümesi durağan değildir. Sonucu gösteren veriler Şekil 6.11'de gösterilmiştir.

Şekil 6.11 Hareketli ortalama grafiği

Hareketli ortalama ile gerçek döviz kuru değerleri arasındaki fark alınarak durağanlık testi tekrar uygulanmıştır. Örnek veri kümesi Tablo 6.7'de, hareketli ortalama grafiği Şekil 6.12'de ve Dickey-Fuller sonuçları Tablo 6.8'de gösterilmiştir. Sonuçlara bakıldığında veri kümesinin durağan olduğu görülmektedir.

Tablo 6.7 Hareketli ortalama sonrası veri kümesi örneği

Gün	Değer
2018-01-12	0.000644
2018-01-13	-0.005215
2018-01-14	-0.006606
2018-01-15	0.001361
2018-01-16	0.014865
2018-01-17	0.013168
2018-01-18	0.007263
2018-01-19	0.001386
2018-01-20	0.001673
2018-01-21	0.007024

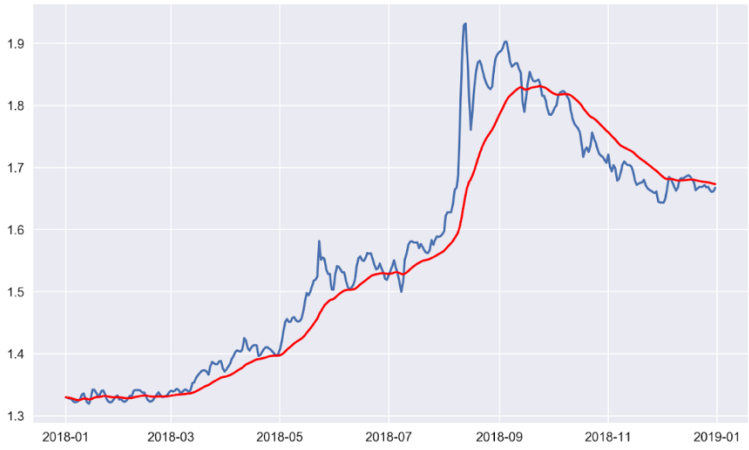


Şekil 6.12 Hareketli ortalama durağanlık grafiği

Tablo 6.8 Hareketli ortalama için durağanlık testi sonuçları

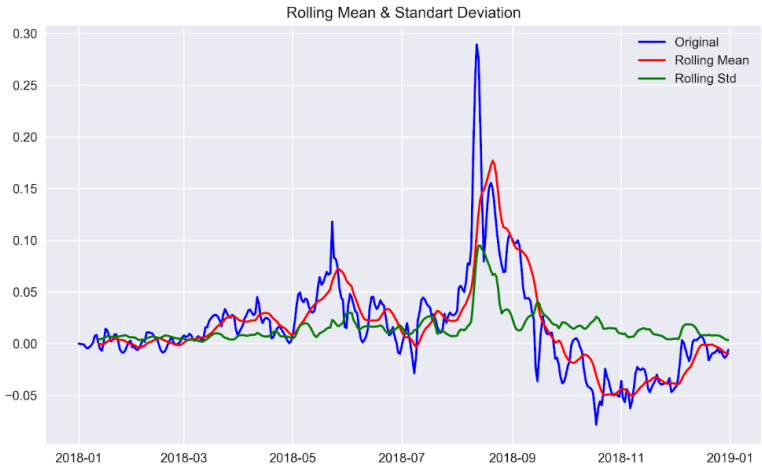
Test İstatistikleri	-4.668428		
p-değerleri	0.000096		
Gecikme sayısı	4		
Gözlem sayısı	349		
Kritik değerler	**1%**	**5%**	**10%**
	-3.4492	-2.8698	-2.5712

Dickey-Fuller test ile *p* değerlerinin 0 ile 1 arasında olması ve mümkün olduğunca düşüş olması beklenir. Aynı zamanda test istatistiklerinin değeri kritik değerlere yakın olmalıdır. Bu sebeple zaman serisi içerisinde bulunun eğilimi görmek için veri kümesi üzerinden bir ağırlıklı ortalama hesabı yapılmıştır. Sonuca ait olan grafik Şekil 6.13'de gösterilmiştir.



Şekil 6.13 Ağırlıklı ortalama hesabı sonrası oluşan sonuç grafiği

Daha sonra ağırlıklı ortalama hesabının sonucu ile güncel döviz kuru değerlerinin bir farklı alınarak yuvarlama hesaplaması ve Dickey-Fuller test aracılığı ile durağanlık testi uygulanmıştır. Sonuç grafiği Şekil 6.14'de test sonuçları ise Tablo 6.9'da gösterilmiştir.



Şekil 6.14 Ağırlıklı ortalama hesabı durağanlık grafiği

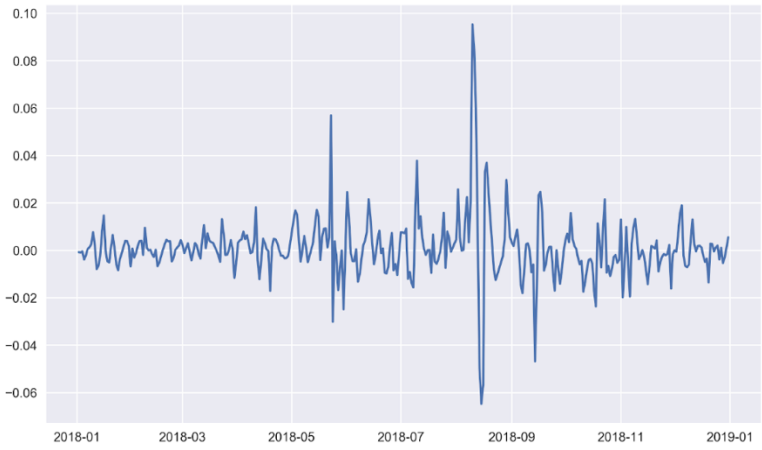
Tablo 6.9 Ağırlıklı ortalama için durağanlık testi sonuçları

Test İstatistikleri	-2.459007		
p-değerleri	0.125769		
Gecikme sayısı	4		
Gözlem sayısı	360		
Kritik değerler	**1%**	**5%**	**10%**
	-3.4486	-2.8696	-2.5710

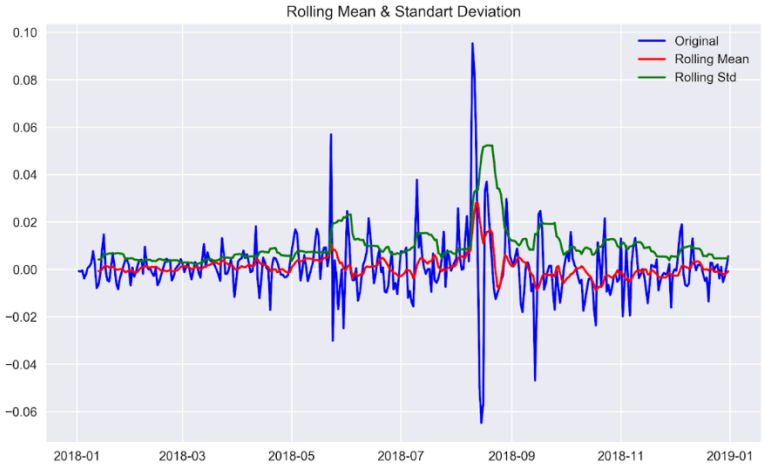
Bu sonuçlara bakıldığında denilebilir ki veri kümesi artık durağandır ve ARIMA modelinin uygulanması için hazır durumdadır. Tez kapsamında bir sonraki bölümde hazırlanan veri kümesine ARIMA modeli uygulanacaktır.

6.7.2. ARIMA modeli uygulaması

Bir önceki bölümde durağanlık testi ile ARIMA modelinin uygulanması için hazırlanan veri kümesi öncelikle oluşturulacak olan öngörü modeli için logu alınmış olan veri kümesi üzerinde kaydırma uygulanmıştır. Daha sonra kaydırma uygulanmış veri kümesi durağanlık testine sokulmuştur. Kaydırılmış veri kümesinin grafiği Şekil 6.15'de, durağanlık testi grafiği Şekil 6.16'da ve sonuçlar ise Tablo 6.10'da gösterilmiştir.



Şekil 6.15 Kaydırılmış veri grafiği

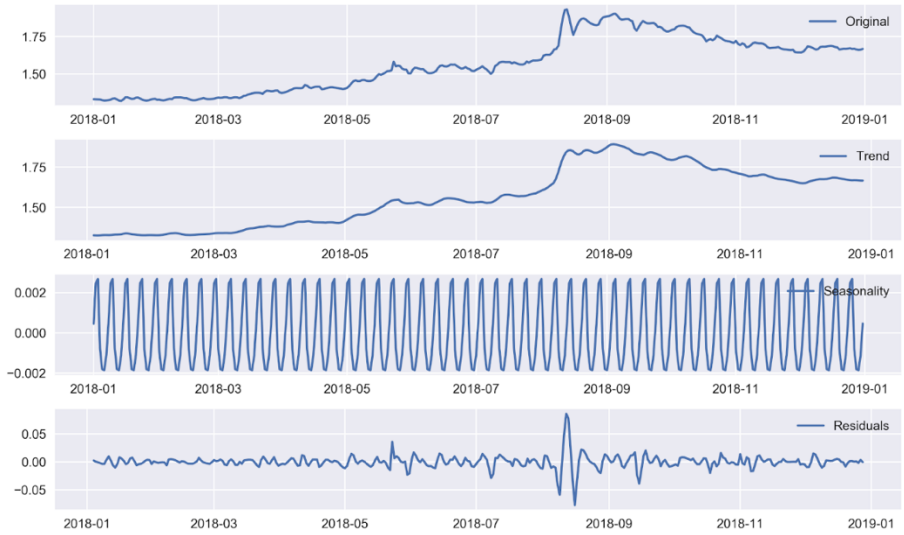


Şekil 6.16 Kaydırılmış veri durağanlık grafiği

Tablo 6.10 Kaydırılmış veri için durağanlık testi sonuçları

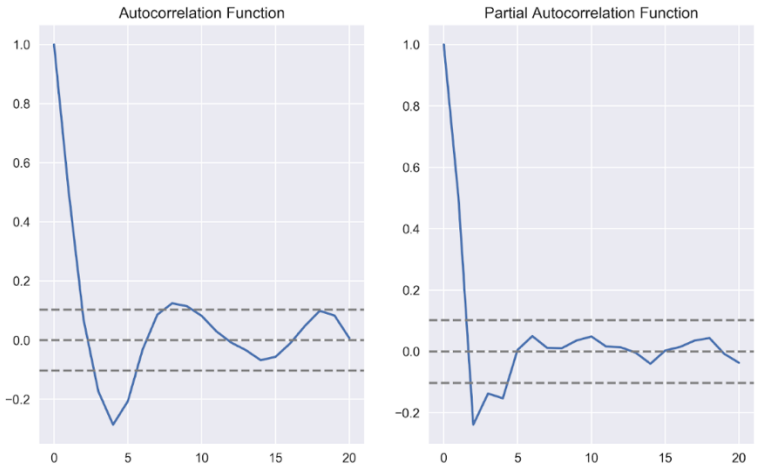
Test İstatistikleri	-1.156141		
p-değerleri	3.263467		
Gecikme sayısı	3		
Gözlem sayısı	360		
Kritik değerler	1%	5%	10%
	-3.4486	-2.8696	-2.5710

Mevcut zaman serisi, ayırma işlevi kullanılarak birden çok zaman serisine; sezon, eğilim ve rastgele olarak ayrılmıştır. Sırasıyla gerçek veri, ortalama sezonsallık, tespit edilen eğilim ve geriye kalan rastgele gürültü zaman serilerini gösteren grafikler Şekil 6.17'de gösterilmiştir.

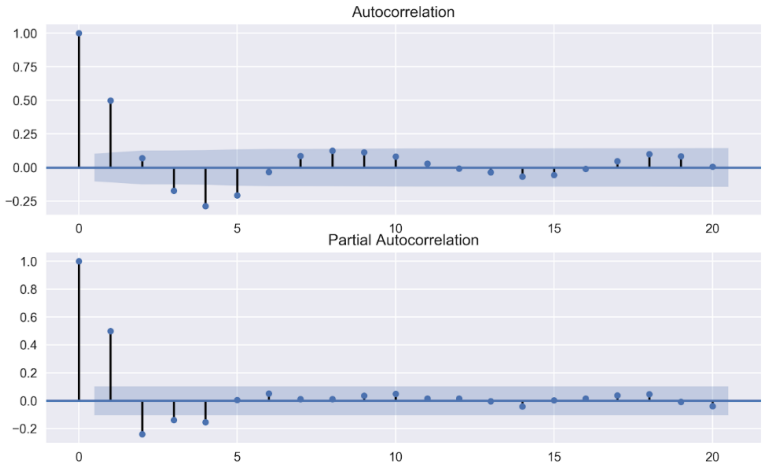


Şekil 6.17 Hareketli ortalamalar ile hesaplanan sezonsal ayrışma grafiği

Daha sonra bir otokorelasyon (ACF) grafiği ile, zaman serisinin sahip olduğu gecikmeler ile kendi kendine olan korelasyonu ve kısmi bir otokorelasyon (PACF) grafiği ile de bir dizi ile tüm düşük dereceli gecikmelerdeki korelasyonlarla açıklanamayan bir dizi arasındaki korelasyon miktarı gösterilmiştir. Böyle yaparak zaman serisi ile gecikme arasında bir ilişki olup olmadığını anlamaya çalışılmıştır. Otokorelasyon (ACF) ve kısmi otokorelasyon (PACF) grafikleri Şekil 6.18'de ve Şekil 6.19'da gösterilmiştir.



Şekil 6.18 Otokorelasyon ve kısmi otokorelasyon grafiği



Şekil 6.19 Otokorelasyon ve kısmi otokorelasyon grafiği

Mavi bölgenin üzerinde sadece bir yükselme var ve bu da 1 numaralı gecikmede bir korelasyon olduğu anlamına geliyor.

ARIMA modeli (2, 1, 2) farklılık sırasını seçilerek uygulanmıştır. Oluşturulan ARIMA modelinin özeti ve sonuçları Şekilde 6.20'de, ARIMA modelinin sonuç grafiği ile artık kareler toplamı (RSS) bilgisi ise Şekil 6.21'de gösterilmiştir.

ARIMA Model Results

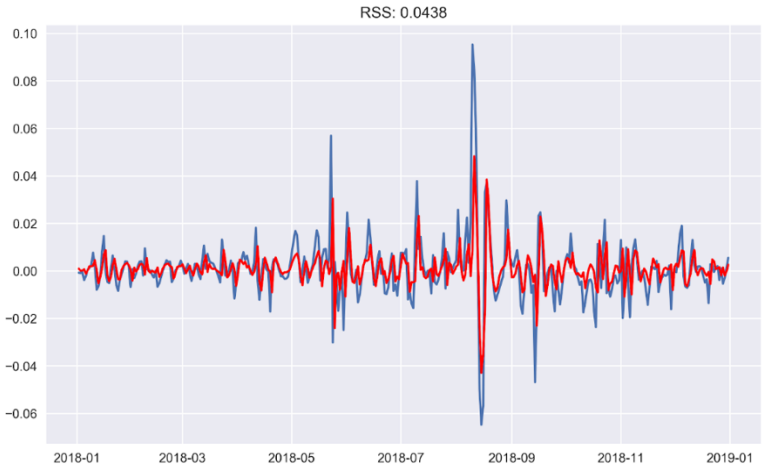
Dep. Variable:	D.value	No. Observations:	364
Model:	ARIMA(2, 1, 2)	Log Likelihood	1125.851
Method:	css-mle	S.D. of innovations	0.011
Date:	Sun, 03 Nov 2019	AIC	-2239.701
Time:	14:58:27	BIC	-2216.319
Sample:	01-02-2018 - 12-31-2018	HQIC	-2230.408

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0009	0.001	1.311	0.191	-0.000	0.002
ar.L1.D.value	1.1271	0.099	11.332	0.000	0.932	1.322
ar.L2.D.value	-0.5681	0.085	-6.698	0.000	-0.734	-0.402
ma.L1.D.value	-0.5763	0.118	-4.896	0.000	-0.807	-0.346
ma.L2.D.value	0.1246	0.112	1.114	0.266	-0.095	0.344

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	0.9921	-0.8810j	1.3268	-0.1156
AR.2	0.9921	+0.8810j	1.3268	0.1156
MA.1	2.3126	-1.6362j	2.8329	-0.0980
MA.2	2.3126	+1.6362j	2.8329	0.0980

Şekil 6.20 ARIMA modeli özeti ve sonuçları



Şekil 6.21 ARIMA modeli artık kareler toplamı grafiği

Daha sonra zaman serisi 365 günlük veri kümesi 291'i eğitim ve 74'ü test verisi olmak üzere ayrılmıştır ikiye ayrılmıştır. Bir önceki adımda elde edilen ARIMA modeli eğitim verisi ile eğitilip hemen ardından test verisi ile deneyimlenmiştir. ARIMA modelinin sonuçları ve test verisi ile elde edilen sonuçlar sırasıyla Şekil 6.22'de ve Şekil 6.23'de gösterilmiştir.

ARIMA Model Results

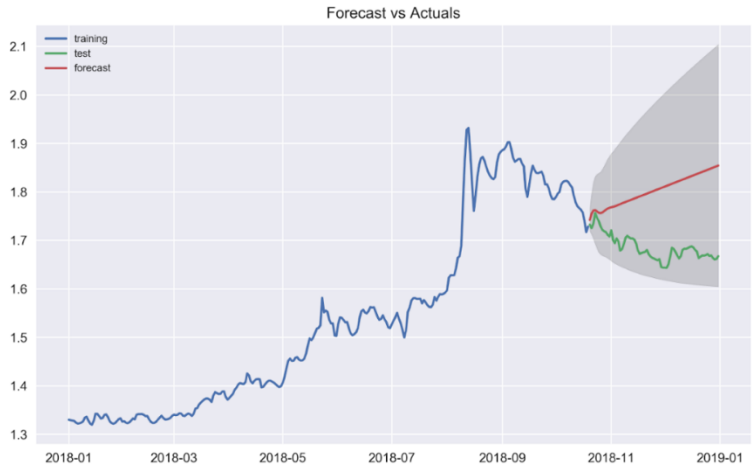
Dep. Variable:	D.value	No. Observations:	291
Model:	ARIMA(2, 1, 2)	Log Likelihood	885.145
Method:	css-mle	S.D. of innovations	0.012
Date:	Sat, 02 Nov 2019	AIC	-1758.289
Time:	17:21:18	BIC	-1736.249
Sample:	01-02-2018 - 10-19-2018	HQIC	-1749.460

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0014	0.001	1.667	0.097	-0.000	0.003
ar.L1.D.value	1.1371	0.108	10.531	0.000	0.925	1.349
ar.L2.D.value	-0.6039	0.081	-7.447	0.000	-0.763	-0.445
ma.L1.D.value	-0.5602	0.125	-4.471	0.000	-0.806	-0.315
ma.L2.D.value	0.1571	0.114	1.379	0.169	-0.066	0.380

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	0.9415	-0.8772j	1.2868	-0.1194
AR.2	0.9415	+0.8772j	1.2868	0.1194
MA.1	1.7831	-1.7851j	2.5231	-0.1251
MA.2	1.7831	+1.7851j	2.5231	0.1251

Şekil 6.22 ARIMA modeli eğitim verisi ile elde edilen özeti ve sonuçları



Şekil 6.23 ARIMA modeli test verisi ile elde edilen sonuçlar

Elde edilen sonuçlardan sonra ARIMA modelinin güvenirliğini test etmek için doğruluk bilgileri hesaplanmıştır. Sonuçlar Tablo 6.11'de paylaşılmıştır.

Tablo 6.11 ARIMA modeli test sonuçları ve doğruluk verileri

Mean Absolute Percentage Error	0.0700
Mean Error	0.1172
Mean Absolute Error	0.1172
Mean Percentage Error	0.0700
Root Mean Squared Error	0.1282
Lag Autocorrelation of Error	0.9459
Correlation between the Actual and the Forecast	-0.7365
Min-Max Error	0.0646
Accuracy	**92.9971**

7. SONUÇ VE DEĞERLENDİRME

7.1. Değerlendirme

Bu tez kapsamında Twitter aracılığı ile; kullanıcıların günlük ekonomik gelişmelere ve bunun yansıması olan döviz kuru görüşlerini bildiren veriler toplanmış uygulanan doğal dil işleme teknikleri ve kelime yerleştirme yöntemleri ile birlikte finansal duygu analizi gerçekleştirilmiştir. Türkiye Cumhuriyeti Merkez Bankası'nın sunmuş olduğu ve herkese açık olan ara yüz vasıtası ile günlük döviz kurları belirli bir tarih aralığında toplanmış ve üzerinde sayısal analizler zaman serileri analizi yöntemleri ile gerçekleştirilmiştir.

Twitter sosyal platformundan toplanan Türkçe ve İngilizce veri kümeleri zaman serisi analizi esnasında tutarlılık göstermesi ve oluşturulacak olan modelin performansının arttırılması için sayısal veriler ile aynı olan tarih aralığında pozitif ve negatif olarak etiketlenmiştir. Etiketleme işlemi İngilizce veri kümesi için önceden eğitimli bir model aracılığı ile etiketlenmiş olup Türkçe veri kümesi ile Kaggle üzerinden paylaşılmış olan etiketli veri kümesi Naif Bayes kullanılarak eğitilmiş bir model ile etiketlenmiştir. Duygu analizinin sonunda doğru ve güvenilir sonuçlar elde edebilmek için veri kümesi veri ön işleme kapsamında bir dizi temizleme işlemine sokulmuştur. Word2vec, GloVe ve fastText gibi kelime yerleştirme yöntemlerinden elde edilen her bir kelime vektörleri LSTM, RNN ve CNN gibi derin öğrenme modellerine girdi olarak verilmiştir. Her bir modelin eğitilmesi sırasında veri kümesi %80 eğitim ve %20 test veri kümesi olarak ayrılmıştır. Deney sonuçları incelenmiş ve aralarından en yüksek doğruluğa sahip olan model seçilmiştir. Çapraz doğrulama yöntemi ile deneyimlenen sonuçlara göre en yüksek performansın Türkçe için LSTM ve GloVe, İngilizce için ise LSTM ve GloVe olduğu gözlemlenmiştir.

Türkiye Cumhuriyeti Merkez Bankası'nın resmi tatillere denk gelen günlerde ve hafta sonu tatillerinde döviz kuru verisi paylaşmamasından ötürü bir yıllık süreyi kapsayacak şekilde toplanan döviz kuru verileri içerisinde oluşan kayıp veriler güvenilir bir yöntem ile doldurulmuş ve eksik olan günler Twitter'da o gün paylaşılan güvenilir kur kaynakları ile kıyaslanmıştır. Böylece sayısal veri analizi üzerinden devamlılık ve analizin tutarlılığı sağlanmıştır. Zaman serisi analizi birkaç farklı model ile deneylenmiş

ve aralarında en iyi performansa sahip olan seçilmiştir. İstatistiklerde bir tahmin yönteminin tahmin doğruluğunu ölçebildiğimiz tüm yöntemler ile hesaplar yapılmış ve en iyi performansın Holt-Winters toplamsal metoduna ait olduğu gözlemlenmiştir.

Yapılan çalışma sonrasında görülmüştür ki toplanan Twitter verisi her ne kadar doğal dil işleme yöntemleri ile temizlenmiş olsa da kullanıcıların ima, manüpilatif ve ironi yaklaşımlı yorumları duygu analizini güçleştirmektedir. Aynı zaman Twitter'dan toplanan verinin yazım yanlışları ile dolu olması verinin kirlilik oranını arttırmakta ve temizleme işlemi sonrası geriye doğru sonuçlar alınabilecek veri kümesinin kalmamasına sebep olmaktadır. Çok büyük veri kümesinin etiketlenmesinde yaşanan zorluklarla birlikte duygu analizinde yüksek doğruluk oranlarının alınamadığı da gözlemlenmiştir.

Derin öğrenme algoritmalarının çok fazla veriye ihtiyaç duyması ve toplanan verinin işlenmesi için gerekli olan donanım ihtiyacını da beraber getirmekte. Gözlemler sonrası seçilen LSTM derin öğrenme modelinin dezavantajlarından bir tanesi ise hesaplama zamanıdır. LSTM yapısı itibari ile veri kümesi büyüdükçe hesaplama süresi de artmaktadır. Bu da beraberinde maliyeti getirmektedir.

7.2. Sonuç

Bu tez kapsamında yapılan araştırmalara ve deneyler sonucunda elde edilen sonuçlara göre zaman serileri yöntemleriyle Amerikan Doları/Türk Lirası kur tahminlemesinde en iyi sonucu veren yöntem olan Holt-Winters toplamsal modeli ve duygu analizinde hem Türkçe hem de İngilizce için en başarılı sonuçların elde edildiği LSTM ve GloVe modeli ile hibrid modeli inşa edilmiştir. Bu deney sonucunda önerilen hibrid modelin Amerikan Doları/Türk Lirası kur tahminlemesinde kayda değer sonuçlar verdiğini göstermiştir.

KAYNAKÇA

Atkinson, K. E. (2019). *Interpolation.* Retrieved from Kendall E. Atkinson, Professor Emeritus: http://homepage.math.uiowa.edu/~atkinson/ftp/ENA_Materials/Overheads/sec_4-1.pdf

Bengio vd. (2003). A Neural Probabilistic Language Model. *Journal of Machine Learning Research 3*, 1137-1155.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 993-1022.

Bodén, M. (2001). A Guide to Recurrent Neural Networks and Backpropagation.

Bojanowski vd. (2017). Enriching Word Vectors with Subword Information.

Box vd. (2016). *Time Series Analysis: Forecasting and Control.*

Ciftci, K., & Ozturk, S. S. (2015). A Sentiment Analysis of Twitter Content as a Predictor of Exchange Rate Movements.

Cireşan vd. (2011). Flexible, High Performance ConvolutionalNeural Networks for Image Classification.

Cooper, P. (2019, October 30). *25 Twitter Stats All Marketers Need to Know in 2020*. Retrieved from Hootsuite: https://blog.hootsuite.com/twitter-statistics/

Deerwester vd. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 391-407.

Gösterge Niteliğindeki Kurlar. (2019). Retrieved from Türkiye Cumhuriyeti Merkez Bankası: https://www.tcmb.gov.tr/wps/wcm/connect/TR/TCMB+TR/Main+Menu/Temel+Faaliyetler/Doviz+Efektif/Doviz+ve+Efektif+Piyasalari/Gosterge+Niteligindeki+Kurlar

Gösterge Niteliğindeki Merkez Bankası Kurları. (2019). Retrieved from Türkiye Cumhuriyeti Merkez Bankası: https://www.tcmb.gov.tr/kurlar/kurlar_tr.html

Gers, F. A., & Schmidhuber, E. (2001). LSTM recurrent networks learn simple context-free and context-sensitive languages.

Glen, S. (2018). *Statistics How To.* Retrieved from Exponential Smoothing: Definition of Simple, Double and Triple: https://www.statisticshowto.datasciencecentral.com/exponential-smoothing/

Holt, C. C. (1957). Forecasting seasonals and trends by exponentially weighted moving averages.

Hyndman, R. J., & Athanasopoulos, G. (2019). *Simple exponential smoothing.* Retrieved from Forecasting: Principles and Practice: https://otexts.com/fpp2/ses.html

Kilimci, Z. H., & Akyokus, S. (2019). The Evaluation of Word Embedding Models and Deep Learning Algorithms for Turkish Text Classification.

Komariah, K., & Sin, B.-K. (2015). Naïve Bayes Approach for Predicting Foreign Exchange Rate Fluctuation Based On Twitter Sentiment Analysis.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An Introduction to Latent Semantic Analysis. *Discourse Processes*, 259-284.

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A Critical Review of Recurrent Neural Networksfor Sequence Learning.

Maria, F. C., & Dezsi, E. (2011). Exhange-rates Forecasting: Exponential Smoothing Techniques and ARIMA Models.

Merkez Bankası Görev ve Sorumlulukları. (2019). Retrieved from Türkiye Cumhuriyeti Merkez Bankası: https://www.tcmb.gov.tr/wps/wcm/connect/TR/TCMB+TR/Main+Menu/Banka+Hakkinda/Genel+Bakis

Mikolov vd. (2010). Recurrent neural network based language model.

Mikolov vd. (2013). Distributed Representations of Words and Phrasesand their Compositionality.

Mohapatra, B. (2019). Machine learning applications to smart city.

Okazaki, M., & Matsuo, Y. (2009). Semantic Twitter: Analyzing Tweets for Real-Time Event Notification., (pp. 63-74).

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks.

Ostertagova, E., & Ostertag, O. (2011). The Simple Exponential Smoothing Model. *Modelling of Mechanical and Mechatronic Systems 2011.*

Ozcan, F. (2016). Exchange Rate Prediction from Twitter's Trending Topics.

Palikuca, A., & Seidl, T. (2016). Predicting High Frequency Exchange Rates using Machine Learning.

Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends® in Information Retrieval*, 1-135.

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Association for Computational Linguistics*, 1532-1543.

Quadratic Polynomial Interpolation. (2019). Retrieved from Mathonline: http://mathonline.wikidot.com/deleted:quadratic-polynomial-interpolation

Ramos, J. (2003). Using TF-IDF to Determine Word Relevance in Document Queries.

Rojas, C. G., & Herman, M. (2018). *Foreign Exchange Forecasting via Machine Learning.*

Rout vd. (2013). Forecasting of currency exchange rates using an adaptive ARMA model with differential evolution based training. *Journal of King Saud University – Computer and Information Sciences*, 7-18.

Sahin, M. (2019, July 16). *HepsiBuradaYorumlar*. Retrieved from Kaggle: https://www.kaggle.com/murats/hepsiburadayorumlar

Sak, H., Senior, A., & Beaufays, F. (2014). Long Short-Term Memory Recurrent Neural Network Architecturesfor Large Scale Acoustic Modeling.

Stenqvist, E., & Lönnö, J. (2017). Predicting Bitcoin price fluctuation with Twitter sentiment analysis.

VandeBogert, K. (2019). *Method of Quadratic Interpolation.* Retrieved from Academic Webpage of Keller VandeBogert: http://people.math.sc.edu/kellerlv/Quadratic_Interpolation.pdf

Varenius, M. (2017). Real currency exchange rate prediction. - A time series analysis.

Winters, P. R. (1960). Forecasting Sales by Exponentially Weighted Moving Averages. *Management Science*, 231-362.

Yasir vd. (2019). An Intelligent Event-Sentiment-Based Daily Foreign Exchange Rate Forecasting System.

Zhang, G. P. (2003). Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 159-175.

Zhang, H., & Li, D. (2007). Naïve Bayes Text Classifier.

ÖZGEÇMİŞ

1986'da Adapazarı'nda doğdu. 2004 yılında Bozüyük Mustafa Şeker Anadolu Lisesi'nden, 2011 yılında ise Anadolu Üniversitesi Kamu Yönetimi bölümünden mezun oldu. Türkiye'de farklı sektörlerde faaliyet gösteren şirketlerde yazılım geliştirici olarak çalışıp altı yıllık tecrübe elde ettikten sonra 2016 yılında Doğuş Üniversitesi Bilgisayar Mühendisliği yüksek lisans programına kabul edilmiştir. 2017 yılında İngiltere'ye taşınmıştır ve şu an Oxford Üniversitesi'nde yazılım mühendisi olarak çalışmaktadır.