BİR TEKSTİL FABRİKASININ ELEKTRİK TÜKETİM DEĞERLERİNİN DERİN ÖĞRENME İLE TAHMİNLENMESİ

Pamukkale Üniversitesi
Sosyal Bilimler Enstitüsü
Yüksek Lisans Tezi
Yönetim Bilişim Sistemleri Anabilim Dalı
Yönetim Bilişim Sistemleri Yüksek Lisans Programı

Hakan YURDOĞLU

Danışman: Dr. Öğr. Üyesi Ömer GÜLEÇ

Haziran 2023
DENİZLİ

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atıfta bulunulduğunu beyan ederim.

Hakan YURDOĞLU

ÖNSÖZ

Çalışma süreci boyunca bilgi birikim ve tecrübelerini aktararak gelişimime katkı sağlayan, her konuda yol gösterici, motive edici ve cesaret kaynağı olan, birlikte çalışmaktan ve öğrencisi olmaktan büyük keyif aldığım danışman hocam Sayın Dr. Öğr. Üyesi Ömer GÜLEÇ'e, yüksek lisans mülakatına katıldığım andan bu tez çalışmasına kadar geçen tüm süreçte desteğini hissettiğim ve derslerini keyifle takip ettiğim Pamukkale Üniversitesi Yönetim Bilişim Sistemleri bölüm başkanı Sayın Prof. Dr. Selçuk Burak HAŞILOĞLU'na, çalışmanın şekillenmesinde rehberlik eden Sayın Doç. Dr. Umut UYAR'a, tez jürisi olarak katkıda bulunan Sayın Dr. Öğr. Üyesi Cihat ÇETİNKAYA'ya, yüksek lisans eğitimi alma imkanı tanıyan ve bu tez çalışmasında kullanılan verileri sağlayan Menderes Tekstil A.Ş.'ne, iş hayatımda her zaman desteğini hissettiğim ve gelişimime büyük katkıda bulunan Menderes Tekstil Bilgi Teknolojileri Direktörü Sayın Ayfer DOĞRU'ya, hem iş hayatında hem de bu tez çalışmasında bilgi ve tecrübeleriyle bana değer katan iş arkadaşım Onur DOĞAN'a, başta annem ve babam olmak üzere beni her koşulda destekleyen aileme, özel hayatım ve iş hayatımda olduğu gibi eğitim hayatımda da desteğini her zaman hissettiğim eşim Özge YURDOĞLU'na sonsuz teşekkür ederim.

Bu tez çalışması, Menderes Tekstil A.Ş. Ar-Ge Merkezi tarafından desteklenen bir proje olup geliştirilen modelin Menderes Tekstil A.Ş.'nin tüm fabrikalarında kullanılması hedeflenmektedir.

ÖZET

BİR TEKSTİL FABRİKASININ ELEKTRİK TÜKETİM DEĞERLERİNİN DERİN ÖĞRENME İLE TAHMİNLENMESİ

YURDOĞLU, Hakan

Yüksek Lisans Tezi

Yönetim Bilişim Sistemleri ABD

Yönetim Bilişim Sistemleri Yüksek Lisans Programı

Danışman: Dr. Öğr. Üyesi Ömer GÜLEÇ

Haziran 2023, 64 Sayfa

Endüstriyel süreçlerin devamlılığını sağlayan en önemli kaynaklardan biri elektrik enerjisidir. Elektrik enerjisinin oldukça maliyetli bir kaynak olması nedeniyle tüketiminin en aza indirilmesi işletmeler için önemli bir husustur. İşletmelerin üretim süreçlerinde tüketilen kaynak değerlerinin tahmin edilebilmesi ile maliyetlerin azaltılması sağlanabilmektedir. Son dönemlerde, Makine Öğrenmesi ve Derin Öğrenme kavramları, herhangi bir alanda gelecek tahmini için kullanılan güçlü Yapay Zekâ alt alanlarıdır. Bu nedenle bu tez çalışmasında, tekstil endüstrisi makinelerinin bekleme durumunda aşırı kaynak tüketimini önlemek amacıyla Derin Öğrenme destekli bir elektrik tahmin modeli tasarlanmıştır. Bu yöntem, Uzun-Kısa Süreli Bellek (LSTM) ve kayan pencere tekniği sayesinde tekstil makinelerindeki elektrik tüketiminin dinamik eşik değerlerini tahminlemektedir. LSTM modeli kullanılarak elde edilen elektrik eşik değerleri, Tekrarlayan Sinir Ağları (RNN) ve Kapılı Tekrarlayan Birimler (GRU) gibi diğer Derin Öğrenme yöntemlerinin yanı sıra geleneksel bir yöntem olan Otomatik Regresif Entegre Hareketli Ortalama (ARIMA) ile karşılaştırılmış, elde edilen sonuçların gerçek zamanlı elektrik tüketim verilerine ne kadar yaklaştığı analiz edilmiştir. Bu tez çalışmasında geliştirilen model, elektrik tüketim seviyelerini başarılı bir şekilde tahmin etmekte, tüketim seviyeleri eşiğe ulaştığında Programlanabilir Mantık Denetleyicisi (PLC) ünitesine durma sinyali göndermekte ve bu sayede aşırı kaynak tüketimini engellemektedir.

Anahtar Kelimeler: Derin Öğrenme, LSTM, Tekstil Endüstrisi, Elektrik Tüketimi

ABSTRACT

ESTIMATING THE ELECTRIC CONSUMPTION VALUES OF A TEXTILE FACTORY WITH DEEP LEARNING

YURDOĞLU, Hakan

Master Thesis

Department of Management Information Systems

Management Information Systems Master Programme

Advisor: Assist. Prof. Dr. Ömer GÜLEÇ

June 2023, 64 Pages

Electrical energy is one of the most important sources that ensure the continuity of industrial processes. Since electrical energy is a very costly resource, minimizing electricity consumption is an important issue for enterprises. It is possible to reduce costs by estimating the resource values consumed in the production processes of the enterprises. Recently, Machine Learning and Deep Learning concepts are powerful Artificial Intelligence subdomains used for future prediction in any field. Therefore, in this thesis, a Deep Learning supported electrical forecasting model is designed to prevent excessive resource consumption of textile industry machines in their standby state. The proposed method estimates dynamic threshold values of electricity consumption in textile machines using the Long-Short-Term Memory (LSTM) and Sliding Window technique. The threshold values obtained with the LSTM model were compared with other Deep Learning methods such Recurrent Neural Networks (RNN) and Gated Repetitive Units (GRU) and Automated Regressive Integrated Moving Average (ARIMA) as a traditional method, then the results has been analyzed how close they are to real-time electricity consumption data at standby. The proposed model in this thesis successfully predicts electricity consumption levels and sends an interrupt signal to the Programmable Logic Controller (PLC) unit when the consumption levels reach the threshold, thus preventing excessive resource consumption.

Keywords: Deep Learning, LSTM, Textile Industry, Electricity Consumption

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGE VE KISALTMALAR DİZİNİ

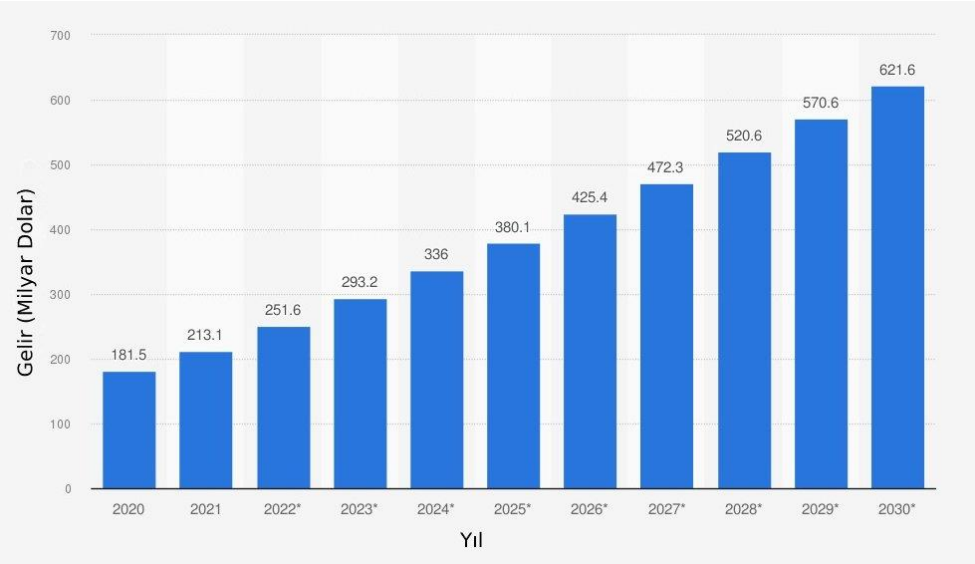
Adagrad	Adaptive Gradient
Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
ARIMA	Autoregressive Integrated Moving Average
CNN	Convolutional Neural Network
DL	Deep Learning
GRU	Gated Recurrent Unit
IIoT	Industrial Internet of Things
IoT	Internet of Things
LSTM	Long-Short Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
PLC	Programmable Logic Controler
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RMSprop	Root Mean Square Propagation
RNN	Recurrent Neural Networks
SGD	Stochastic Gradient Descent
YSA	Yapay Sinir Ağları

GİRİŞ

Günümüzde teknolojinin gelişmesi ile birlikte iletişim kavramı da değişmektedir. Tarihi süreçte iletişim önce insan ile insan arasındayken, bu kavram günümüzde insan ile makine arasında bir köprü vazifesi görmektedir. Sensör teknolojilerinin gelişmesi ve boyutlarının çok daha küçülmesi, insan ile makine arasındaki iletişimi kolaylaştırmış, insanların günlük sorunlarını çözebilen ve hayatı kolaylaştıran uygulamaların geliştirilmesini sağlamıştır. Bu iletişim yaklaşımı, günlük hayatta el yordamı ile çalıştırdığımız makinelerin ve hatta sürekli kullandığımız her nesnenin "akıllı" hale getirilmesini sağlamıştır. Bu uygulamalar sayesinde insan, her nerede olursa olsun İnternet ya da yerel bir ağ üzerinden "akıllı" nesnelere komut verebilmekte, karşılığında da komutu yerine getiren nesnelerden bilgi edinebilmektedir. Çevremizde bulunan her nesnenin, sensörler sayesinde akıllı hale getirilmesi, bununla birlikte bu nesnelerin İnternet üzerinden insanlar ile ve hatta kendi aralarında da iletişime geçebilmesi Nesnelerin İnterneti (IoT) kavramını ortaya çıkarmıştır.

IoT, mevcut her nesneyi ekosisteme bağlamak için modern bir ağ hizmeti olarak ele alınabilir. Bu nesneler, bir hizmet talep edebilen veya bir hizmet sunabilen canlı veya cansız varlıklar olabilir. Bu küresel bağlanabilirlik, nesnelerin içine sensörler, aktüatörler, mikrodenetleyiciler vb. elektronik ve bilgisayar alanındaki gelişmelerle mümkün hale gelmiştir. Bu durum, IoT'nin son birkaç yılda birçok alanda somutlaşmasını görünür hale getirmiştir. IoT kavramı basitçe sensörler, iletişim ağları, veri ve uygulamalar/servislerden oluşan bir mimariye sahiptir (Gupta, 2020: 8).

Gün geçtikçe IoT ağına dahil olan cihaz sayısı katlanarak artmaktadır. 2022 yılında 13 milyar cihazın IoT ekosisteminde yer aldığı, 2025 yılında 21,5 milyar cihazın, 2030 yılında da 30 milyara yakın cihazın IoT ekosistemine dahil olacağı ön görülmektedir (Statista, 2023). Dünya genelinde büyük bir pazar payına sahip olan IoT teknolojisi için elde edilen yıllık gelir Şekil 1 ile verilen grafiğe göre 2022 yılında 251,6 milyar dolar iken 2023 yılında bu sayının 293,2 milyar dolar, 2025 itibariyle 380,1 milyar dolar ve 2030 yılında 621,6 milyar dolar olması beklenmektedir.



Şekil 1. IoT Teknolojisinden Elde Edilen Yıllık Gelir (Statista, 2023)

IoT kavramı, farklı alanlarda uygulanabilen geniş bir çalışma sahasına sahip olmakla birlikte, nesnelerin bulundukları ortamdan elde ettikleri bilgiler sayesinde oldukça büyük bir veri akışının ortaya çıkmasını sağlamaktadır. Mobil telefonlardan giyilebilir teknolojilere, bilgisayarlardan IP kameralara, televizyonlardan ve diğer tüm akıllı cihazlara kadar çoğu akıllı cihaz artık günlük hayatın vazgeçilmez bir parçası olmuştur. Bunun yanı sıra, endüstriyel uygulamalarda akıllı fabrikalar, süreçleri yöneten robotlar, insansız üretim gibi kavramlar daha da popülerleşmeye başlamıştır. Gerek günlük hayatta gerek endüstriyel uygulamalardaki tüm süreçlerde rol alan akıllı nesneler, oldukça yoğun bir veri akışına neden olmaktadır.

Her IoT sistemi bir veri yaşam döngüsüne sahiptir. Bu yaşam döngüsü veri üretme, veri toplama, veri filtreleme, veri ön işleme, arşivleme ve veri saklama gibi kısımlardan oluşur. Bu döngünün son aşamasında veri analizi ve sorgulama işlemleri yer almaktadır (Shirvanian vd., 2022: 6). Buna bağlı olarak, son yıllarda Endüstri 4.0 ile IoT uygulamaları sayesinde toplanan büyük verilerin analiz edilmesi ve faydalı bilgiye dönüştürülmesi oldukça önem kazanmıştır. Endüstriyel alanlarda kullanılan IoT uygulamalarına Endüstriyel Nesnelerin İnterneti (Industrial Internet of Things - IIoT) adı verilmektedir (Gang vd., 2022: 9683). IIoT, endüstriyel süreçlerde yer alan birçok farklı seri port, sensör, endüstriyel İnternet ve 5G ile desteklenmektedir (Changchun vd., 2022: 4050). Ancak IoT ile IIoT arasında temel farklılıklar bulunmaktadır. IIoT, IoT'nin aksine sürdürülebilirlik yönetimi, risk yönetimi, varlık yönetimi gibi birçok gerekliliğe sahiptir.

Bunun yanı sıra, birçok cihazdan, alandan ve yapıdan elde edilen çok büyük boyutta gerçek zamanlı verinin alınması IIoT'nin avantajlarından biridir (Milic vd., 2023).

Benzer şekilde son dönemlerde göstermiş olduğu hızlı gelişim sayesinde oldukça popülerleşen, günlük hayatta kullandığımız uygulamalara entegrasyonu sayesinde de günlük alışkanlıklarımızı farklı bir noktaya taşıyan yapay zekâ kavramı, veriden bilgiye uzanan yolda aktif rol oynamaktadırlar. Yapay zekâ ve IoT teknolojileri, Endüstri 4.0 kavramının en temel bileşenleridir (Hansen ve Bogh, 2021: 363). Bu nedenle, IoT cihazlarından elde edilen veriler sayesinde IoT için geliştirilen uygulamaların yapay zekâ destekli olması endüstriyel süreçleri daha üst bir seviyeye çıkarmaktadır. Bugün olduğu gibi uzun yıllar boyunca, IoT ve IIoT kavramları makine öğrenmesi ve yapay zekâ algoritmaları ile ayrı düşünülemez durumda olacaktır (Milic vd., 2023).

Tez kapsamında, bir tekstil işletmesinde yer alan bez terbiye makinelerinin 2021 ile 2023 yılları arasındaki elektrik tüketim değerlerini Derin Öğrenme yöntemleri ile işleyen bir yapay zekâ uygulaması geliştirilmiştir. Bu uygulama, elektrik tüketim verilerini öğrenerek bir sonraki zaman dilimi için tüketim eşik değeri belirlemektedir. Tekstil işletmesinde yer alan ve Endüstriyel Nesnelerin İnterneti (IIoT) kapsamında çalışan bez terbiye makinelerinde tüketilen elektrik enerjisi miktarı, tez kapsamında geliştirilen uygulama tarafından tahmin edilen eşik değerine ulaştığında, uygulama tarafından yine uygulamanın entegre olduğu Programlanabilir Mantık Kontrolörü (Programmable Logic Controler - PLC) devresine bir uyarı sinyali gönderilmektedir. Geliştirilen bu uygulama ile, kaynakların değerli ve kaynak kullanımının kritik olduğu tekstil işletmesinin üretim süreçlerinde aşırı elektrik enerjisi kullanımının önüne geçilmesi hedeflenmiştir. Tez kapsamında geliştirilen bu uygulama, makinelerden alınan gerçek zamanlı veriler üzerinde benzetim ortamlarında geliştirilmiş, modelin eğitimi ile testi yine bu veriler ile sağlanmış, modelin tahminleri gerçek zamanlı veriler ile karşılaştırılmış, modelin hataları giderilmiş ve gerçeğe daha yakın tahminler elde edilmesi için optimize edilmiştir. Tez kapsamında kullanılan LSTM Derin Öğrenme yöntemi ile elde edilen sonuçlar RNN ve GRU gibi farklı Derin Öğrenme yöntemleri ile karşılaştırılmış, bu sayede de seçilen yöntemin etkinliği ve uygunluğu belirlenmiştir.

Tezin organizasyonu ise şu şekildedir; yapay zekâ, makine öğrenmesi ve derin öğrenme kavramları Bölüm 1'de anlatılmaktadır. Bölüm 2'de literatürde yer alan endüstri için geliştirilmiş ve Endüstriyel Nesnelerin İnterneti (IIoT) kavramı ile birlikte kullanılan yapay zekâ çalışmalarına yer verilmiştir. Ayrıca tez kapsamında bir tekstil işletmesinin

elektrik tüketim değerlerini derin öğrenme metodu ile tahminlenmesi için geliştirilen modele yer verilmiştir. Sonuç bölümünde ise elde edilen sonuçlar değerlendirilmiştir.

BİRİNCİ BÖLÜM
YAPAY ZEKÂ

IoT kavramı içinde yer alan makinelerin ve endüstriyel süreçlerin daha "akıllı" hale gelmesini sağlamak, bu kavramlara bir de zekâyı dahil etmek ile mümkündür. Yıllar önce ortaya çıkan, makinelerin de insanlar gibi öğrenmesinin, düşünmesinin ve kararlar almasının mümkün olup olmadığı düşüncesi, günümüz yapay zekâ kavramının temellerini oluşturmaktadır. Bu fikir, Alan Turing, John von Neumann ve Norbert Wiener gibi önemli isimlerin, bilgisayar programlarına zekâ eklemek, kendi kendisine öğrenebilmesini ve çevresini kontrol etmesini sağlamak amacıyla insan beyninin çalışma prensibini makinelere uyarlamaya yöneltmiştir.

Modern yapay zekâ çalışmalarının ön dönemi olan İkinci Dünya Savaşı sırasında Alan Turing "Bombe" adını verdiği kod kırma makinesini icat ederek savaşın kaderini değiştirecek bir hamle yapmıştır (Acar, 2020). Modern yapay zekâ çalışmaları ise İkinci Dünya Savaşı'ndan sonra hız kazanmıştır. Alan Turing, 1947 yılında bir konferansında akıllı makinelerin yapılabileceğini belirtmiş (McCarthy, 2007: 1175), 1950 yılında ise "Bilgi İşlem Makineleri ve Zekâ" adlı bir makalede (Turing, 1950: 440) "Makineler düşünebilir mi?" sorusuna yanıt aramış ve bu sayede modern yapay zekâ kavramının temellerini atmıştır. Bu çalışmada Turing, zeki sistemlerinin nasıl olması gerektiği hakkında bilgi vermiştir. Daha sonra bu akıma birçok bilim insanı dahil olmuştur.

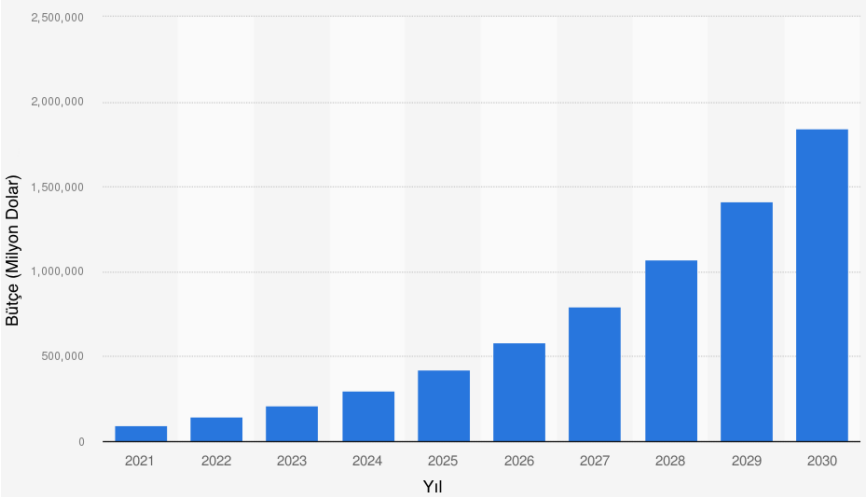
1955 yılında ise John McCarthy, Marvin L. Minsky, Nathaniel Rochester ve Claude E. Shannon, Yapay Zekâ (Artificial Intelligence) terimini üzerinde çalışmalar yapmışlardır. Yapay zekânın fikir babası Alan Turing olarak bilinse de terim olarak "yapay zekâ" ilk olarak 1956'da Hannover Dartmouth College'da yapılan bir konferansta John McCarthy tarafından kullanılmıştır (Lewis, 2014). McCarthy'ye göre Yapay Zekâ, makineleri zeki yapmaya çalışan, makinelerin insanların dilini anlaması için uğraşan ve insanlar gibi sorunlara ve amaçlara sahip olmasını sağlayan bir mühendislik ve bilim dalıdır (González García vd., 2019: 5). Yapay zekânın terim olarak kullanıldığı bu etkinlikten sonraki yıllarda Aziz (1961), Benzeşim (1963), Eliza (1965), Bilgin (1970) ve Stajyer (1979) gibi önemli yapay zekâ programları geliştirilmiştir ve çalışmalar hız kazanmıştır (Kutlusoy, 2019).

Yapay zekâ çalışmaları yıllara göre artış göstermesine rağmen birçok düşüş döneminden de geçmiştir. Bu dönemler "yapay zekâ kışı" olarak adlandırılmış olsa da en

önemlisi 1970'lerin sonunda yaşanan düşüş dönemidir (Floridi, 2020: 1). Daha sonra ise 1997 yılında IBM "Deep Blue" isimli bir yapay zekanın dünya satranç şampiyonu Garry Kasparov'u satranç maçında yenilgiye uğratmıştır ve bu gelişme dünya genelinde büyük ses getirmiştir (Goodfellow vd., 2016).

Son yıllarda yapay zekâ uygulamaları, işletmecilik, tıp, otomotiv, eğitim, finans, sağlık vb. alanlarda, mühendisler, bilgi teknolojileri uzmanları, analistler gibi meslek grupları kullanılmaktadır (Jarek ve Mazurek, 2019: 50). Günlük hayatımızda birçok yapay zekâ teknolojisi kullanılmakta ve bunlar insan hayatını kolaylaştırmaktadır. Uygulamaların artmasıyla insan gücüne daha az ihtiyaç duyulacağı düşünülmektedir (Yeğin, 2020: 493).

Şekil 2 ile verilen grafiğe göre 2022'de yapay zekâ alanı için dünya genelinde pazar payı 142 milyar dolar iken bu payın 2025 itibariyle 420 milyar dolar, 2030 itibariyle de 1,8 trilyon dolar olması beklenmektedir.

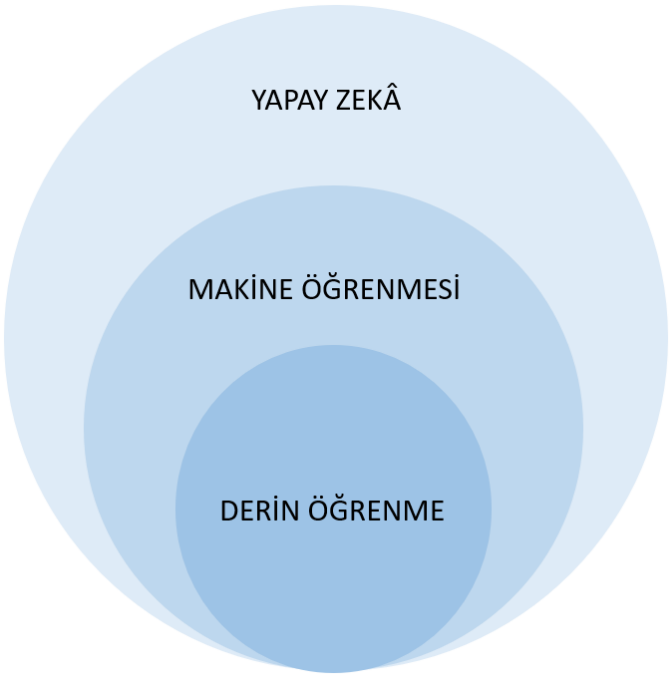


Şekil 2. Yapay Zekâ Alanının Yıllara Göre Pazar Payı (Statista, 2023)

Pazar payı dikkate alındığında uzun yıllar güçlü, kalıcı ve başarılı olmayı hedefleyen şirket, kuruluş ve devletler yapay zekâ sistemlerini geliştirmek ve kullanmak zorundadır. Bunun yanı sıra endüstriyel süreçler de yapay zekâ destekli olarak tasarlanmalı ya da yenilenmelidir. Yapay zekâ yatırımındaki artış, yapay zekâ sistemlerine olan ihtiyaçtan doğmaktadır. Dünya çapındaki şirketler, bu ihtiyaç karşısında projeler geliştirerek yatırımlar yapmaktadır. Bilişim sektörünün sürekli genişlemesi göz önüne alındığında, önümüzdeki yıllarda yapay zekâ çalışmalarının daha fazla olacağı görülmektedir.

Kurum ve kuruluşlar stratejik planlarını yaparken büyük veriyi işleyerek daha doğru ve etkili kararlar alabilirler (Aydın, 2019). Günümüzde yapay zekâ, video ve ses işlemeden görüntü oluşturmaya, oyunlardan örüntü tanımaya, doğal dil işlemeden e-ticarete kadar uzanan geniş bir çalışma alanına sahiptir. Son dönemlerde, kullanıcı deneyiminden öğrenebilen ve kullanıcıya istatistiki öneriler sunmak yerine bir sonraki adımı tahminleyebilen yapay zekâ destekli sistemler geliştirilmektedir. Bu tür sistemlerde, kullanıcı bir tercih yaptığında yapay zekâ bir sonraki tercihini tahmin edebilmekte, kullanıcıya ilgili alan içinde rehber olabilmektedir.

Yapay zekâ kavramı geniş bir alana sahip olmakla birlikte, alt kolları da oldukça güçlü birer çalışma alanı haline gelmiştir. Günümüzde sıklıkla birbirine karıştırılan Yapay zekâ, makine öğrenmesi, derin öğrenme ve yapay öğrenme kavramları aslında temelde birbirleri ile ilişkili olsa da farklılıklara sahiptir. Derin öğrenme makine öğrenmesinin, makine öğrenmesi ise yapay zekanın bir alt dalıdır. Şekil 3'te yapay zekâ, makine öğrenmesi ve derin öğrenme arasındaki ilişki gösterilmektedir.



Şekil 3. Yapay Zekâ Makine Öğrenmesi ve Derin Öğrenmenin Arasındaki İlişki

1.1. Makine Öğrenmesi

Yapay zekâ'nın ve bilgisayar bilimlerinin önemli bir alt alanı olan Makine Öğrenmesi (Machine Learning) son dönemin en popüler alanlarından biri haline gelmiştir. Bu kavram, makinelerin insanlar gibi öğrenme süreçlerini içermektedir.

Öğrenme süreci, verinin analiz edilmesi ile başlayan bir süreçtir. Bu süreç daha sonra verinin işlenmesi ve son olarak da daha önceden analiz edilen veriden karar verme aşamalarına kadar uzanmaktadır (González García vd., 2019: 5). Bir başka deyişle makine öğrenmesi, veri modellerini ve veri yapılarını analiz etmeye ve yorumlamaya odaklanmaktadır. Makine öğrenmesi, insan müdahalesi veya açık programlama talimatları olmadan doğru bir model oluşturarak sistemin otomatik olarak öğrenmesini sağlar. Makine öğrenmesi algoritmaları, özellikle IoT'nin sınıflandırma, optimizasyon, kümeleme gibi problemlerini çözmek için kullanılmaktadır.

Makine öğrenmesi algoritmaları 3 sınıfa ayrılmaktadır (Cifuentes vd. 2020: 4215). Bunlar;

1. Denetimli Öğrenme (Supervised Learning)
2. Denetimsiz Öğrenme (Unsupervised Learning)
3. Pekiştirmeli Öğrenme (Reinforcement Learning)

Denetimli öğrenmede giriş ve çıkış verileri etiketlenmiş ve belirlenmiştir. Bu teknikte model, öğrenme sürecinde insan yardımı ile etiketlenmiş verilerin fonksiyonundan öğrenme işlemi gerçekleştirir. Bu tür bir öğrenme süreci, yüksek doğruluklu veri modelleri üretir. Ancak bu modelde büyük veri kümeleri için etiketleme işlemi maliyetli ve zordur (Saba Raoof ve Durai, 2022).

Regresyon analizi (Regression Analysis), destek vektör makineleri (Support Vector Machines), karar ağaçları (Decision Trees) ve rastgele orman (Random Forest) algoritmaları denetimli öğrenme yöntemlerinden bazılarıdır. Regresyon analizi, bağımlı ve bağımsız değişkenler arasındaki ilişkiyi göstermektedir. Eğer veri analizinde bir ya da birden fazla bağımsız değişken kullanılırsa, buna çoklu regresyon analizi adı verilmektedir. Destek vektör makineleri ise farklı veri kümelerini ayırmak ve tahmin etmek için kullanılan matematiksel bir modeldir. Karar ağaçları tabanlı sınıflandırma algoritmaları ise kategorileri tahmin etmek için kullanılır. Bu modelde, tahminin ve ilişkinin yüksek derecesini bulmak için entropi ve bilgi kazancı hesaplanır. Düşük entropi ve yüksek bilgi kazancı etkin bir karar ağacı oluşturur. Rastgele orman algoritması ise bir ya da daha fazla karar ağacı algoritmasının bir araya gelmesiyle oluşur. Bazen bu algoritmalar sınıflandırıcı ve regresyon analizi aracı gibi davranırlar. Rastgele orman algoritması, her iterasyonda karar ağaçlarının raporlarını toplar. Sonuç olarak, rastgele

orman algoritması karar ağaçlarından elde edilen sonuçlarından hepsinden faydalanmaktadır (Samadi ve Seitz, 2022: 110).

Denetimsiz öğrenme modelinde verilerin etiketlenmesine ve kullanıcının modeli izlemesine gerek yoktur. Bu tür öğrenmede model veri kümesinden kendi veri desenini ve bilgisini keşfetmeye çalışır. Daha çok kümeleme işlemleri için kullanılır. Denetimsiz öğrenme daha az karmaşıklık içerdiğinden uygulanabilirliği yüksektir. Diğer taraftan, bu yöntem için giriş bilgisi olmadığı için çıkış verilerinin ve modelin doğruluğu düşüktür (Ahmad vd., 2022: 4730).

Temel Bileşenler Analizi (Principal Component Analysiz), k-Ortalamalar Kümelemesi (k-Means Clustering), Kendi-kendine Organize Haritalar (Self Organizing Maps) denetimsiz öğrenme algoritmalarından bazılarıdır. Temel bileşenler analizi, büyük değişken kümelerinin boyutunu orijinal kümedeki hemen hemen her bilgiyi içeren daha küçük bir kümeye indirger. Bu model, IoT sistemlerinde veri boyutu azaltmak için sensör seviyesinde çalışan ve veri doğrulamayı sağlayan kullanışlı bir modeldir. k-Ortalamalar Kümesi, kümelerde ya da sınıflarda farklı verileri sınıflandırmak için kullanılır. Başlangıç olarak k-rastgele orta noktalar (centroid) seçilir. Veri kümesinde yer alan diğer noktalar en yakın orta noktanın kümesine dahil olur.

Pekiştirmeli öğrenme modelinde ise modelin çevre ile interaktif bir ilişkisi bulunmaktadır. Bu öğrenme modeli ödül ve ceza kavramlarını içerdiği için model, deneme yanılma ile öğrenerek en yüksek ödülü almak için çalışır.

1.2. Derin Öğrenme

Derin öğrenme, yapay sinir ağları tabanlı öğrenme algoritmaları kullanarak, bilgisayarların veri üzerinde karmaşık yapılar ve desenler algılamasını ve bunları kullanmasını sağlayan bir makine öğrenmesi yöntemidir (Patchaiammal ve Sundar, 2022: 205). Bir başka deyişle derin öğrenme, insan beyninin yapısını taklit ederek karmaşık sorunları çözebilen bir dizi algoritmadır (Forootan vd., 2022: 4832).

1943 yılında Warren McCulloh ve Walter Pitts, makinelere zekâ kavramını eklemek için insanlarda bulunun sinir sisteminin bir benzerini yapay sinir ağları adıyla bilgisayar üzerinde tasarlamışlardır. 1950'li yıllarda IBM'den Nathanial Rochester IBM 704 bilgisayarlarında yapay sinir ağlarını modellemiştir. 1958 yılında Frank Rosenblatt, günümüzde de kullanılan Perceptron kavramını ortaya çıkarmıştır. 1959 yılında Bernard

Widrow ve Marcian Hoff, ADALINE ve MADALINE modellerini tasarlamışlardır. MADALINE, gerçek dünya problemlerine uygulanan ilk yapay sinir ağı modelidir. 1982 yılında yapay sinir ağına olan ilgi tekrar artmış, John Hopfield çok yönlü sinir ağları geliştirmiştir. Bu çalışmaya kadar yapılan yapay sinir ağları çalışmalarında sinirler arası bağlantılar tek yönlü olarak tasarlanmıştır. 1982 ve 1985 yıllarında çeşitli konferanslar düzenlenmiş olsa da bunlardan en büyüğü 1987 yılında 1800 katılımcıyla IEEE tarafından gerçekleştirilmiştir. 1997 yılında ise derin öğrenme alanında LSTM modeli geliştirmiştir. 1998 yılında ise Yann LeCun, veriden öğrenme kısmında önemli bir adım olarak doküman tanımaya uygulanan GRU modelini tasarlamıştır (Dong vd. 2021).

Birçok derin öğrenme algoritması yapay sinir ağı mimarisine sahiptir. Buna bağlı olarak derin öğrenmeye derin yapay sinir ağları da denmektedir. Bu öğrenme biçiminin "derin" olması, yapay sinir ağının zamanla oldukça fazla seviyeye sahip olmasından kaynaklanmaktadır. Yani, yordamların üretkenliği ağın derinliğine doğrudan bağlıdır. Derin öğrenme, en uygun sonucu bulmak için verileri düşük düzeyden yüksek düzeye çeviren temel doğrusal olmayan modüllerden oluşur.

Derin öğrenmenin, verinin yüksek-seviye karakteristiklerini ayıklaması, verinin etiketli ya da etiketsiz olmasına bağlı olmadan veriyi işlemesi, farklı hedefleri yerine getirebilmek için eğitilebilmesi gibi birçok avantajı bulunmaktadır. Derin öğrenme algoritmaları genellikle zaman serilerinde oldukça başarılı tahminlerde bulunmaktadır.

Derin öğrenme, Yapay Sinir Ağları, CNN ve RNN olmak üzere üç ana tiple çoklu sinir ağ mimarisi olarak bilinen bir mimari oluşturur (Patchaiammal ve Sundar, 2022: 208). Literatürde en çok kullanılan derin öğrenme algoritmaları LSTM, RNN, GRU, CNN ve YSA'lardır.

1.2.1. Tekrarlayan Sinir Ağı (Recurrent Neural Network - RNN)

RNN, sinir ağlarının sıralı olduğu durumlarda yararlı bir modeldir. RNN, her bir girişe yeni katmanların eklendiği durumlarda önceki katmanın sonuçlarını hatırlar. Bir RNN modeli 1'den n'e kadar giriş serisi içerir ve ağ katmanı üzerinden bir geri propogasyon gösterir (Sundar ve Patchaiammal, 2022: 225). RNN modeli, zaman serisi üzerinde iyi çalışan bir modeldir. Veri dizilerinin geçmiş durumlarını hatırlayamayan geleneksel YSA'ların aksine geniş ölçüde giriş veri dizisini hesaplamak için gizli katmanlar içerir. RNN modeli, çoklu katmandan oluşurken her bir katman da çoklu düğümlere sahiptir (Zaheer vd. 2023: 590).

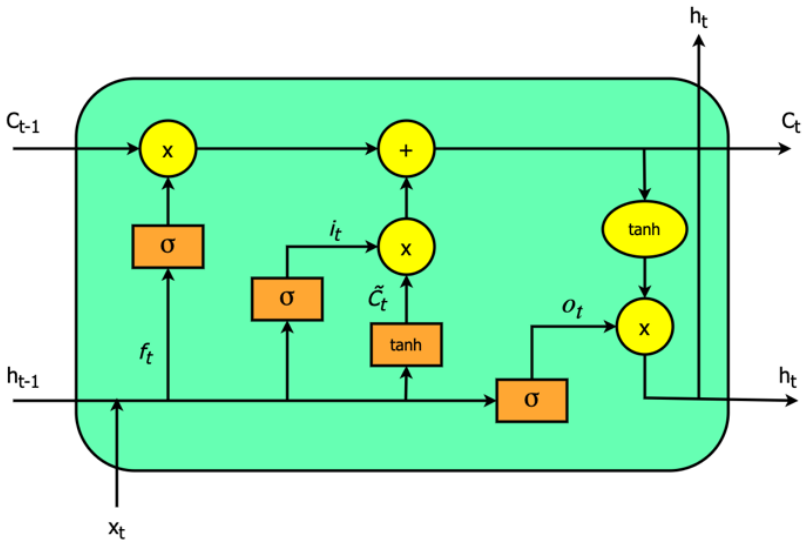
RNN modeli, makine çevrisi, insan genetik kodunun düzenlenmesi, elektrik yük tahmini gibi birçok farklı alanda kullanılır. Genel olarak RNN, geri propogasyon süreci zamanla gerçekleşecek şekilde tasarlanır ve çok büyük dizilere sahip modellerde böyle bir özellik önceki verileri unutmalarına neden olur. Bu durumun üstesinden gelmek için LSTM modeli geliştirilmiştir (Forootan vd. 2022: 4832).

1.2.2. Uzun-Kısa Süreli Hafıza (Long-Short Term Memory - LSTM)

1997 yılında Sepp Hochreiter ve Jürgen Schmidhuber RNN'in sahip olduğu gradyan yok oluşu (gradient disappearence) ve gradyan patlaması (gradient explosion) gibi iki dezavantajını ortadan kaldırmak için RNN modeline giriş ve çıkış kapılarını ekleyerek LSTM modelini tasarlamışlardır (Hochreiter ve Schmidhuber, 1997: 1741). Diğer bir deyişle LSTM, geleneksel ileri-beslemeli yapay sinir ağının sadece giriş düğümünden veri aldığı ve verinin sadece giriş katmanından gizli katmana daha sonra da çıkış katmanına ilerlediği RNN ailesine ait bir yapay sinir ağıdır (Agga vd., 2022).

LSTM'in ana fikri, bellek hücresinin varlığı nedeniyle uzun bir süre sonra bile belleğin durumunu korumasıdır. Bellek durumu, bellekteki veri akışını düzenleyen kapılardan oluşur. Bellek durumu, geçit birimlerinin önemine dayalı olarak önceki durumların bilgi değerlerini değiştirmek için tüm LSTM hücrelerinde mevcuttur (Greff vd., 2016: 2222).

Şekil 4'te gösterildiği gibi standart bir LSTM hücresi, giriş katmanı (input layer), tekrar katmanı (recurrent layer) ve çıkış katmanı (output layer) olmak üzere 3 katmandan oluşur (Aparna, 2018: 1676). Bunun yanı sıra bu katmanlarda 3 adet kapı bulunmaktadır.



Şekil 4. LSTM Hücresi

Bu kapılar şu şekilde özetlenebilir (Rani vd., 2022);

1) Unutma Kapısı (Forget Gate): Hücre durumundan silinmesi gereken bilgi hakkında karar veren kapıdır. Önceki hücre çıkışını (O_{t-1}) ve mevcut hücre girişini (I_t) göz önüne alarak, 0 ile 1 arasında bir değer elde etmek için her bir gizli katmana sigmoid fonksiyonu uygulanır. Eğer çıkış "1" değerini veriyorsa, sinir ağı bu bilgiyi saklayacaktır aksi halde bilgi silinecektir.
2) Güncelleme Kapısı (Update Gate): Aynı değerler olan önceki hücre çıkışı (O_{t-1}) ve mevcut hücre girişi (I_t) verisini, sigmoid ve $tanh$ fonksiyonlarından oluşan aktivasyon katmanına gönderir. Bu katmanda bu iki çıkış değeri ile çarpım gerçekleştirilir ve r sonucu elde edilir. Son olarak, unutma kapısının çıkışı ile elde edile r sonucunun toplaması yapılır. Elde edilen yeni bilgi durum güncellemesi için kullanılır.
3) Çıkış Kapısı (Output Gate): Hücre içinde elde edilen bilginin bir sonraki hücreye gönderilip gönderilmeyeceğinin kararının verilmesinden sorumludur.

Şekil 4'te hücrede kullanılan tüm denklemler Denklem [1 – 6]'da verilmiştir. Denkleme σ sigmoid fonksiyonunu, b kapılar için öngerilim değerini ve W ağırlıkları ifade etmektedir.

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

$$f_t = \sigma\left(W_f.[h_{t-1}, x_t] + b_f\right) \qquad (2)$$

$$o_t = \sigma\left(W_o.[h_{t-1}, x_t] + b_o\right) \qquad (3)$$

$$\tilde{C}_t = tanh\,(W_C.[h_{t-1}, x_t] + b_C) \qquad (4)$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \tilde{C}_t \qquad (5)$$

$$h_t = o_t \otimes tanh\,(C_t) \qquad (6)$$

1.2.3. Kapılı Tekrarlayan Birim (Gated Recurrent Unit - GRU)

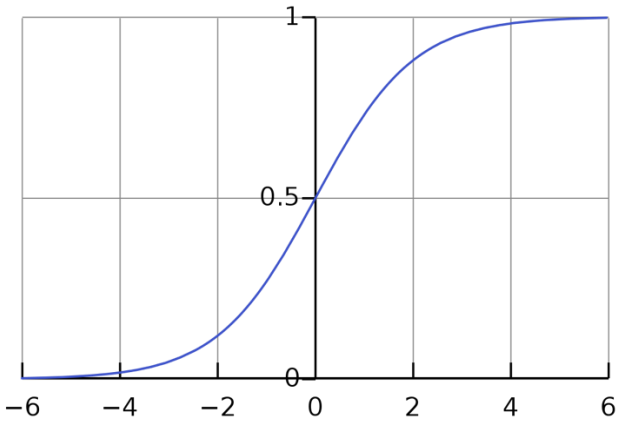
GRU, LSTM gibi kısa süreli hafıza problemlerini çözmek için tasarlanmış bir RNN çeşididir. GRU bir hücre durumuna sahip olmamakla birlikte, bilgileri taşıyan gizli bir duruma sahiptir. GRU ayrıca, sıfırlama (reset) (r^t) ve güncelleme (update) (z^t) kapısı olmak üzere iki kapıdan oluşmaktadır. Güncelleme kapısı, LSTM'deki unutma ve giriş kapılarına benzer davranışa sahiptir. Bu kısım, hangi bilginin reddedileceği, hangi bilginin ekleneceğine karar vermekle sorumludur. Sıfırlama kapısı ise önceki verinin ne kadarının unutulacağının belirlenmesinden sorumludur. GRU'nun LSTM'e göre daha az kapıya sahip olmasından dolayı, öğrenme süreci daha kısadır.

1.2.4. Aktivasyon Fonksiyonları

Bir yapay sinir ağının en önemli bileşenlerinden biri olan aktivasyon fonksiyonu, iletme ya da iletmeme işlemini yapan insan sinir ağındaki iletime benzemektedir. Aktivasyon fonksiyonları, giriş ve çıkış arasındaki doğrusal olmayan bağıntıyı oluşturmak için kullanılır (Dong vd., 2021). Bu doğrusal olmayan bağıntı, insan beynini taklit eden birçok sinir düğümü ve birçok katmanı bir araya getirir. Sigmoid, hiperbolik tanjant ve ReLU gibi birçok farklı aktivasyon fonksiyonu vardır.

Sigmoid Fonksiyonu (σ): Sonucu 0 ya da 1 olan fonksiyondur. Bilgiyi unutmak ya da hatırlamak için LSTM kullanırken, LSTM hücreleri arasında bilgi geçişi için de kullanılır (Sundar ve Patchaiammal, 2022: 228). Sigmoid fonksiyonunun denklemi Eşitlik 7, grafiği ise Şekil 5 ile verilmiştir.

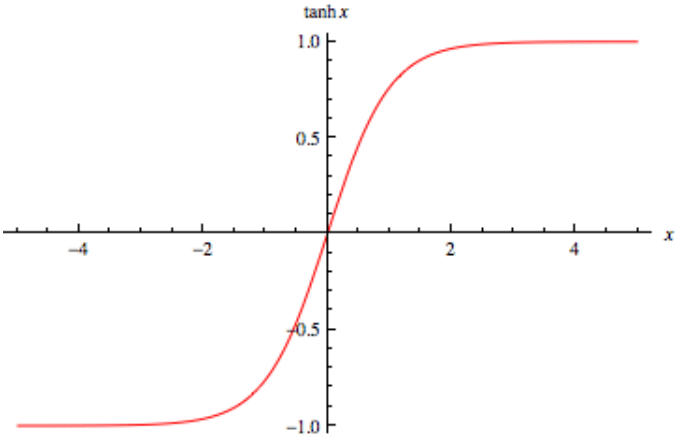
$$S(x) = \frac{1}{1 + e^{-x}} \qquad (7)$$



Şekil 5. Sigmoid Fonksiyonu (Han ve Moraga, 1995)

Hiperbolik Tanjant ($tanh$) Fonksiyonu: Sigmoid fonksiyonun aksine, $tanh$ pozitif ve negatif değerler üretir. Hiperbolik tanjant fonksiyonunun denklemi Eşitlik 8, grafiği ise Şekil 6 ile verilmiştir.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (8)$$



Şekil 6. Hiperbolik Tanjant (tanh) Fonksiyonu (Blickle vd., 1998)

Doğrultulmuş Doğrusal Birim Fonksiyonu (Rectified Linear Unit - ReLU): Giriş değeri 0'dan küçükse 0, aksi halde giriş değerini üretir. Bu fonksiyonda, yapay sinirler aynı anda aktive olmaz. Gizli katmanlar için idealdir. Geri propagasyon hatalarını engeller. Bir sinir hücresinin çalışma prensibine en yakın fonksiyondur (Patchaiammal ve Sundar, 2022: 210). ReLU aktivasyon fonksiyonunun denklemi Eşitlik 9 ile verilmiştir.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \qquad (9)$$

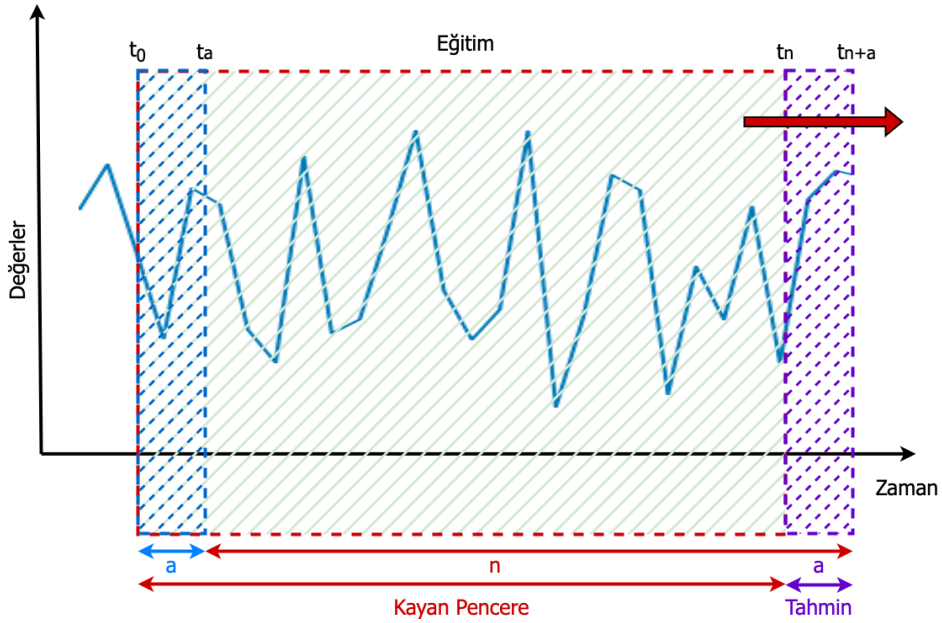
1.2.5. Aşırı Öğrenme (Overfitting) ve Yetersiz Öğrenme (Underfitting)

Makine öğrenmesi algoritmaları tahmin yapabilmek için verinin büyük bir bölümünü öğrenme sürecinde kullanır. Buna bağlı olarak, verinin kalan kısmı test sürecinde kullanılır. Yapay öğrenmede, modelin tahminlerinin gerçek verilere yakın olması modelin oldukça iyi tasarlandığını gösterir. İyi tasarlanmış bir modelde öğrenme sürecindeki veri ile yapılan başarılı tahminler, test sürecinde farklı veriler kullanıldığında da devam edecektir. Ancak, model öğrenme sürecindeki verilere hızlıca uyum sağlar. Eğer model, öğrenme verilerinde yüksek başarıyı sağlarken farklı test verilerinde bu başarıyı yakalayamıyorsa, tasarlanan model öğrenmek yerine ezberlemeye başlamış demektir. Bu duruma aşırı öğrenme ya da ezberleme (overfitting) denir. Aşırı öğrenme, makine öğrenmesi algoritmalarında sıkça karşılaşılan ve çözülmesi gereken bir sorundur. Bu sorun, modelin genellikle çok sayıda katmana sahip olması, modelin karmaşık yapıda olması, veride çok sayıda özellik olması ya da öğrenme sürecinde yeterli verinin kullanılmamasından kaynaklanır (Zhang vd., 2019: 2).

Yetersiz öğrenme (underfitting) ise makine öğrenme algoritmalarında modelin öğrenme verilerden geçerli ya da uygun tahminler yapamaması anlamına gelmektedir. Bu durumda model bir türlü öğrenemez ve dolayısıyla doğru tahminlerde bulunamaz. Bu sorun genellikle modelin öğrenme sürecinde kullanılan verilerin yetersiz olmasından, modelin çok basit düzeyde tasarlanmış olmasından ya da veri ön işlemenin düzgün yapılmamasından kaynaklanmaktadır (Koehrsen 2018: 6).

1.2.6. Kayan Pencere Tekniği (Sliding Window Technique)

Bir zaman serisinin x boyutlu bir parçasını makine öğrenmesi metodunun öğrenme sürecinde kullanılması ile elde edilen y boyutlu tahmin değerleri ile verinin başlangıcından $x - y$ kadar veriyi çıkararak bu iki veri parçacığını öğrenme sürecine dahil eden, veri boyutu sabit kalacak şekilde bu süreci iteratif olarak tekrarlayan yönteme kayan pencere tekniği adı verilmektedir.



Şekil 7. Kayan Pencere Tekniği

Şekil 7'de gösterilen kayan pencere tekniğine göre, t_0 ile t_n arasında n boyutlu bir zaman serisi verisi yer almaktadır. İlk iterasyonda n boyutlu bu veri makine öğrenmesi metodunu öğrenme sürecinde kullanılarak t_n ile t_{n+a} arasında a boyutlu bir tahmin verisi elde etmektedir. Kayan pencere tekniğine göre ikinci iterasyonda, t_0 ile t_a arasında a boyutlu veri, n boyutlu veriden çıkarılarak bu veriye t_n ile t_{n+a} arasında yer alan a boyutlu veri eklenmiştir. Bu durumda, n boyutlu veri bloğunun boyutu (aynı zamanda kayan pencerenin boyutu) korunmuş olmakla beraber bu veri bloğu her iterasyonda sağa doğru hareket etmektedir. Böylece zaman serisi verisi üzerinde eski veriler kullanılarak yeni tahmin verileri üretilmektedir ve bu tahminler daha sonraki tahminler için de kullanılmaktadır.

İKİNCİ BÖLÜM
BİR TEKSTİL İŞLETMESİNİN ELEKTRİK TÜKETİM DEĞERLERİNİN DERİN ÖĞRENME YÖNTEMİ İLE TAHMİNLENMESİ

2.1. Endüstride Enerji Tüketimi ve Yapay Zekâ

Endüstriyel makinelerin akıllanması, yapay zekâ ile IoT alanlarının Endüstri 4.0 çatısı altında ayrılmaz bir bütün hale gelmeleriyle mümkün olmaktadır. Makine öğrenmesi ve derin öğrenme özellikle IoT ekosisteminde sıklıkla tercih edilen bir yapay zekâ alanlarıdır. Özellikle akıllı evler ve şehirler, enerji sistemleri, görüntü tanıma, akıllı sağlık sistemleri, akıllı çevresel uygulamalar ve ulaşım gibi alanlarda yapay zekâ algoritmaları sıklıkla kullanılmaktadır.

Yucesan vd. (2021), regresyon, zaman serileri, makine öğrenmesi yöntemleri kullanarak Türkiye'nin günlük doğal gaz tüketim tahmini modeli geliştirmiştir. Malakouti vd. (2022), rastgele ağaçlar ve LSTM kullanarak rüzgâr türbini gücü tahmini yapmak için bir makine öğrenmesi modeli geliştirmiştir. Benzer olarak, Ikeda ve Nagai (2021), makine öğrenmesi kullanarak binaların merkezi enerji ve depolama sistemleri için bir model geliştirmişlerdir. Yang vd. (2022) elektrik tüketim ücretlerinin tahmini için makine öğrenmesi algoritmaları üzerine çalışma yapmışlardır. Oprea vd. (2021) ise denetimsiz öğrenme metodu kullanarak elektrik tüketim değerlerindeki anomalilerin tespitini yapan bir makine öğrenmesi sistemi geliştirmişlerdir. Walker vd. (2020) binalarda saatlik elektrik talep değerlerini tahmin edebilmek için yapay sinir ağları, destek vektör makineleri ve rastgele ağaçlar içeren bir makine öğrenmesi modeli geliştirmiştir. Çalışmada, 47 resmi kurum binasından elde edilen 2 senelik veriler kullanılmıştır. Elde edilen sonuçlara göre, düşük hata ve yüksek doğruluk göz önüne alındığında rastgele ağaçların kullanıldığı modelin daha iyi bir performans gösterdiği görülmüştür. Grimaldo vd. (2020), kNN yöntemini görsel analiz ile birleştirdikleri bir enerji tedarik tahmin modeli geliştirmişlerdir. Derin öğrenme finans sektöründe, hisse senedi fiyatları, döviz kuru ve kredi riski gibi tahminler için kullanılmaktadır. Kitaoka vd. (2021), derin öğrenme algoritmaları kullanılarak Bitcoin fiyatlarının tahmini yapılmıştır.

Tekstil endüstrisi de gün be gün yapay zekâ teknolojilerinden yararlanmaktadır. Buna bağlı olarak, Kumar ve Bai (2023) yaptıkları çalışmada kumaşta kullanılan

desenleri sınıflandıran ve kumaş üzerindeki hataları saptayan bir LSTM modeli geliştirmişlerdir. Yasir vd. (2022), tekstil endüstrisindeki talep tahmininin iç ve dış göstergelerin önemini araştırmak için talep tahmini yaparken, genelleştirilmiş en küçük kareler yöntemi, tek-katmanlı perceptron, doğrusal regresyon, destek vektör makineleri ve LSTM modellerini kullanmışlardır.

2.2. Yöntem ve Amaç

Günümüzde enerji verimliliği, üretim işletmeleri için çok önemli bir konu haline gelmiştir. İşletmeler maliyetlerini azaltıp bu sayede karlılığı artırmayı hedeflemektedirler. Maliyeti azaltabilmenin yöntemlerinden birisi de enerji tasarrufu yapmaktır. Enerji tasarrufunu sağlayabilmek için çeşitli yöntemler ve teknolojiler kullanılmaktadır. Derin öğrenme, enerji tüketimlerinin tahminlerinin kolaylıkla yapılabildiği yöntemlerden başında gelmektedir. Bu sayede, işletme yöneticileri enerji tüketimlerini kontrol edip yönetebilmektedir.

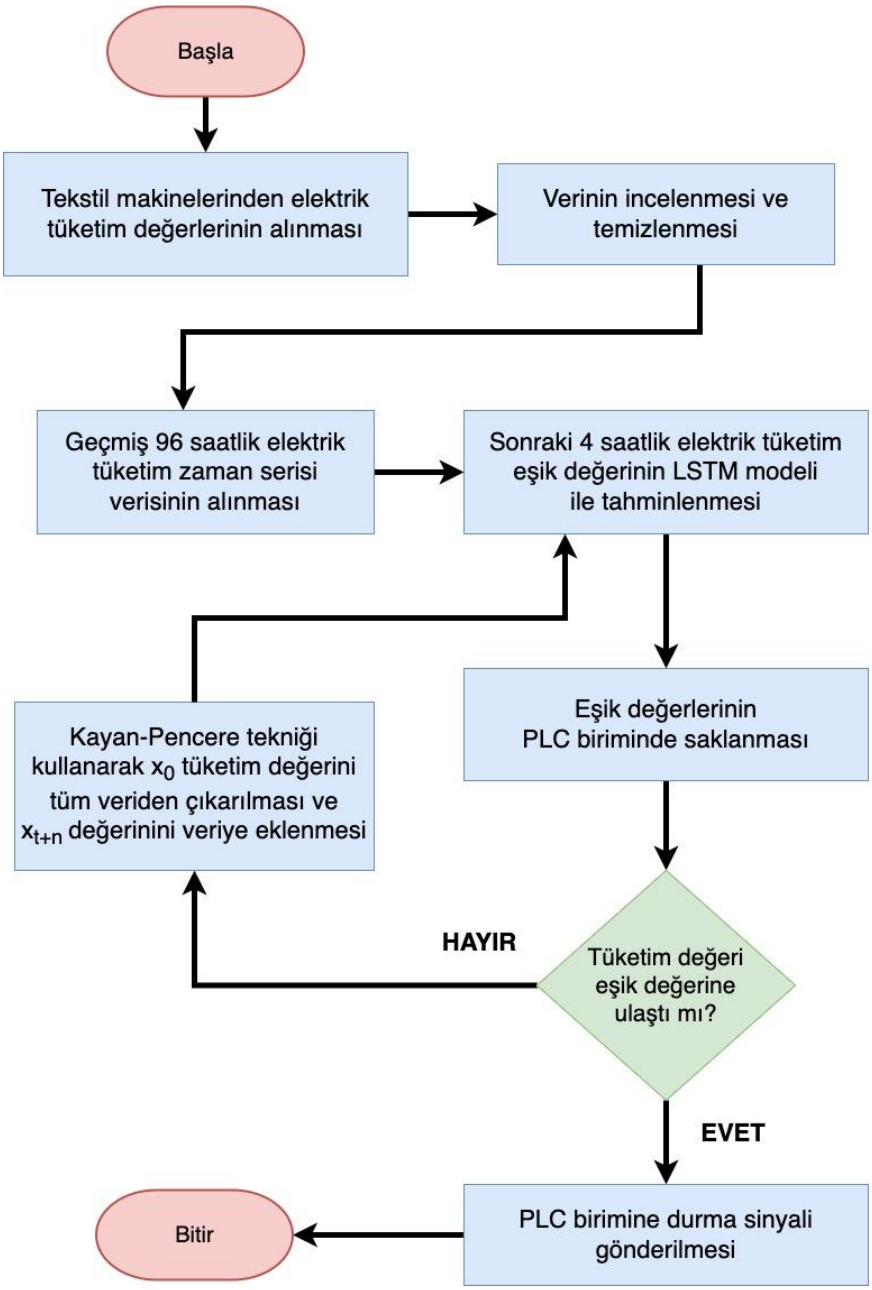
Derin öğrenme ile tahmin yapabilmek için öncelikle işletmenin verilerine ihtiyaç duyulmaktadır. Bu veri, zamanla büyüyerek daha doğru sonuçlar elde edilebilecek büyük veri (big data) haline gelecektir. Elde edilen bu veriyi analiz edip istenilen sonuca varmak için öncelikle verinin anlaşılması gerekmektedir. Tekstil işletmelerinde kullanılan makinelerin enerji tüketimleri de diğer sektörlerde olduğu gibi derin öğrenme yöntemleriyle tahmin edilebilmektedir. Derin öğrenme ile tahminleme yapılarak enerji tasarrufu sağlanabilmekte, üretimde verimlilik artmakta, makine arızaları önceden tespit edilebilmekte ve bakım süreçleri optimize edilebilmektedir. Tekstil işletmelerinde doğru analizlerin yapılabilmesi için her makineden ayrı ayrı veri toplanarak farklı veri setleri oluşturulmalıdır. Daha sonra bu veriler, bir takım veri işleme süreçlerinden geçirilerek derin öğrenme algoritmalarına uygun hale getirilmelidir. En doğru sonuca ulaşabilmenin önemli kurallarından birisi de verinin doğruluğundan ve veri toplama cihazlarının kalibre edilmiş olmasından emin olmaktır. Toplanan yanlış veri ya da yapılan yanlış ölçümler, elde edilecek tahminler ile gerçek veriler arasında büyük farklara neden olacaktır.

Bir yapay zekâ uygulamasında model oluşturulmadan önce problemin doğru belirlenmesi en kritik adımlardan birisidir. Çalışmanın odaklanacağı konu, çalışmanın hedefleri, çalışmanın önemi, çalışmada kullanılacak yöntemler ve veri işleme aşamaları, problemin belirlenmesi sürecinde şekillenmeye başlamaktadır. Bu sayede geliştirilecek model daha etkili ve daha verimli olacaktır.

Hedeflerin belirlenmesi, girdilere karşılık olarak beklenen çıktıları daha doğru bir şekilde belirleyecektir. Ulaşılmaya çalışılan çıktılar kesin olarak belirlendiğinde, modelin doğru çalışması sağlanacak ve bu sayede çalışmanın doğru şekilde sonuçlanması için bir temel oluşturmuş olacaktır. Bunun yanı sıra, çalışmanın öneminin, amaçlarının ve hangi soruna çözüm getireceğinin anlaşılması da çalışmayı başarıya götüren diğer önemli konulardır.

Derin öğrenme projelerinde model, veri setindeki özniteliklerin öğrenilmesi ve sonraki adımlarda tahmin edilmesi için önemlidir. Derin öğrenme modelleri genellikle çok katmanlıdır ve her katman, girdi olarak aldığı öznitelikleri daha yüksek seviyede temsil eden öznitelikler haline getirir (LeCun vd., 2015: 436). Derin öğrenme modellerinde eğitim verilerinin çeşitliliğini, verinin kalitesi modelin doğruluğu ile performansını etkilemektedir. Bu modellerde öğrenme kapasitesi yüksek olduğundan gürültülü ya da doğru olmayan veriler modelin yanlış öğrenmesine hatta öğrenememesine sebep olabilmektedir (Goodfellow vd., 2016: 175). Model seçimi, derin öğrenme projelerinde en önemli adımlardan birisidir. Modelin, veri setinin boyutuna, özniteliklerine, etiketlenmesine ve tahmin edilmesi gereken probleme uygun olması gerekmektedir. Farklı problemlerde problemin çözümünü sağlayacak uygun modeller seçilmelidir. Doğru model seçimi ve uygun hiper parametrelerin belirlenmesi önemli bir başarı ölçütüdür (Chollet, 2018: 71).

Bu tez çalışmasında, bir tekstil işletmesinde yer alan tekstil makinelerinin bekleme anındaki elektrik enerji tüketimleri derin öğrenme metotları kullanılarak, bir sonraki zaman dilimi için tüketim eşik değeri belirlenmesi hedeflenmiştir. Oluşturulan modele ait akış şeması Şekil 8'de gösterilmektedir.



Şekil 8. Elektrik Tüketim Tahmini Derin Öğrenme Modeli Akış Şeması

Tekstil işletmesinde yer alan tekstil makinelerinin elektrik tüketim değerleri, her makine üzerinde Şekil 9 ile gösterilen Entes MPR63 marka ve modelli enerji analizörü ile elde edilmektedir.

Şekil 9. Entes MPR63 Enerji Analizörü

Enerji analizöründen alınan elektrik sinyal değerleri, Şekil 10'da gösterilen Siemens S7-1200 model PLC'ler ile anlamlandırılmaktadır. Bu PLC'ler elektrik verisini, ağ anahtarı sayesinde CAT6 kablosu üzerinden Profinet haberleşme protokolü ile işletmenin ana ağına iletmektedir.

Şekil 10. Siemens S7-1200 PLC

Daha sonra .NET programlama teknolojisi ile yazılmış olan uygulama aracılığı ile toplanan bu veriler, sunucuya aktarılmaktadır. Sunucu üzerinde bulunan MSSQL veri tabanlarında ise tüm makinelere ait elektrik enerji tüketim değerlerini depolanmaktadır. Derin öğrenme modelinde kullanılmak üzere, veri tabanlarında depolanan verileri barındıran tablolardan bekleme birim elektrik tüketimi, tarih, saat ve bekleme alanlarının barındırdığı veriler .xlsx formatında modele aktarılmıştır.

Bu tez çalışmasında geliştirilen derin öğrenme modeli, Google Colab üzerinde Python dili ile geliştirilmiştir. Ayrıca, büyük ve çok boyutlu dizileri ve matrisleri işlemek ve bilimsel hesaplamalar yapmak için Numpy kütüphanesi, grafik oluşturma, veri görselleştirme ve grafikleri kaydetme amacıyla Matplotlib kütüphanesi, veri okuma ve yazma, veri temizleme ve düzenleme, veri yapılarını düzenleme, veri seçme ve filtreleme, veri manipülasyonu gibi işlemleri yapmak için Pandas kütüphanesi, makine öğrenimi algoritmalarını uygulamak, veri ön işleme, model eğitimi ve değerlendirme, performans ölçütlerinin hesaplanması, model seçimi ve doğrulama gibi işlemler için Sklearn (scikit-

learn) kütüphanesi, derin öğrenme modelleri oluşturma, model eğitimi ve değerlendirme, model katmanlarını oluşturma gibi işlemler için Tensorflow kütüphanesi kullanılmıştır. Bunun yanı sıra, Tensorflow kütüphanesinin bir bileşeni olarak çalışan Keras kütüphanesi ise derin öğrenme modellerini daha hızlı ve kolay bir şekilde oluşturmak, eğitmek ve değerlendirmek için kullanılmıştır.

Tez kapsamında geliştirilen derin öğrenme modeli, 4 saatlik elektrik verilerinden 24 tanesini çerçeveleyerek bir sonraki 4 saatlik elektrik tüketimi tahmin etmektedir. Bir başka deyişle, 24 adet 4 saatlik veri alınmış, makinelerin toplam 96 saatlik elektrik tüketimine göre bir sonraki 4 saat için bir tahmin değerinin bulunması amaçlanmıştır. Çalışmada kullanılan veri seti Ocak 2021 ile Mart 2023 tarihleri arasında toplanan verilerden oluşmaktadır.

Tekstil işletmesinde yer alan tekstil makinesine ait elektrik tüketimlerini içeren veri seti bir zaman serisi veri seti olduğu için, zaman serisi analizi ve tahminlemesinde sıkça kullanılan LSTM, RNN ve GRU derin öğrenme algoritmaları kullanılmış ve sonuçları alınmıştır (Rajagukguk vd., 2020: 6623). Bu sayede, bu algoritmalar arasında karşılaştırma yapılarak hangi algoritmanın bu model için uygun olduğu belirlenmiştir.

2.3. Problemin Tanımı

Problemin tanımının yapılması, geliştirilecek modelin başarısı için en önemli faktörlerden birisidir. Bu sayede, çalışmanın hedefine ulaşması için gerekli olan yol haritası da ortaya çıkmaktadır (Géron, 2019: 36). Problemin tanımlanması, veri toplama süreci ile doğrudan bağlantılıdır. Bu yüzden, problemin tanımlanmasından sonra, çalışmanın amacı belirlenmelidir ve böylece uygun verilerin toplanma aşamasına geçilmelidir. Veri toplama süreci, çalışmanın ulaşmak istediği hedeflere uygun olarak gerçekleştirilmelidir. Ayrıca toplanan veri, probleme ait özellikleri ve problemin yapısını yansıtmalıdır. Daha sonra toplanan veriye uygun model seçimi yapılabilir (Chollet, 2018: 72).

Diğer tüm işletmelerde olduğu gibi tekstil işletmelerinde de kaynak tüketimi, verimliliği artırmak için ele alınması gereken en önemli konulardan birisidir. Günümüzde kaynakların azalması, kaynaklara erişimin maliyetli olması, çevre dostu yenilenebilir kaynak arayışları, kaynakların tüketiminde daha dikkatli olunması gerektiğini göstermektedir. Buna bağlı olarak, tekstil işletmelerinin en önemli enerji kaynaklarının

başında elektrik enerjisi gelmektedir. Bu nedenle, elektrik tüketiminin izlenmesi ve azaltılması tekstil işletmesine doğrudan katkı sağlamaktadır.

Araştırma kapsamında geliştirilen modelde, tekstil makinelerine ait bekleme anındaki elektrik tüketim verileri toplanarak analiz edilmekte ve geliştirilen derin öğrenme modeli ile gelecekteki tüketimler ile ilgili tahminler yapılmaktadır. Çalışmada kullanılan elektrik verileri, işletmenin terbiye fabrikasında yer alan ön terbiye makineleri, baskı işlemi makineleri, şablon makineleri, caratsch makineleri, apre makineleri ve ramözler, şardon makineleri, düz boya makineleri ile kalite kontrol makineleri olarak sınıflandırılmış 160 adet makine içindeki ön terbiye makineleri arasında yer alan ve 3 adet merserize makinesinden biri olan 117 numaralı merserize makinesinden elde edilmiştir. İşletmenin veri toplama sistemlerini kuran ve uygulayan Elektrik Elektronik Mühendisi'nin uzman görüşüne başvurulmuş ve 117 numaralı merserize makinesine ait veri toplama sisteminin kalibrasyon ve doğrulama işlemlerinin başarıyla tamamlandığı tespit edilmiştir. Bu nedenle, en güvenilir verilerin bu makineden elde edildiği belirlenmiştir. Çalışmada verileri kullanılan 117 numaralı merserize makinesinin panoramik görünümü Şekil 11 ile gösterilmektedir.

Şekil 11. Merzerize Makinesi Panoramik Görünümü

Çalışmanın yapıldığı tekstil işletmesine ait boya terbiye fabrikasında birçok bez merzerize işleminden geçtiği için, bu makinelerin maliyetlerin en aza indirilmesi karlılık açısından tekstil işletmeleri için önemli bir rol oynamaktadır. Merserize makinesi elektrik

tüketiminin yanı sıra sıcak su, soğuk su ve doğal gaz tüketimi de yapmaktadır. Merserize makinesine bezin ilk girişinin sağlandığı kısım Şekil 12'de gösterilmiştir.

Şekil 12. Merserize Makinesi Bez Giriş Bölümü

Merserize makinesinde merserize işlemi görecek olan bezler tükendiğinde, operatör bez değişimi yapmaktadır. Bu sırada merserize makinesi beklemeye alınmaktadır. Merserize işlemi sırasında bazen de bez tipi değişiklikleri gerekmektedir. Bu durumda da yine operatör tarafından makine beklemeye alınmaktadır. Bunun yanı sıra makinelerde planlı bakımlar yapılmaktadır. Bu durumda, makine bazen tamamen kapatılırken, bazen bu işlem makine bekleme halindeyken yapılmaktadır. Ayrıca operatörlerin vardiya değişimlerinde makine bekleme pozisyonuna alınabilmektedir. Tüm bu bekleme anlarında merserize makinesi elektrik enerjisi tüketimine devam etmektedir.

Makinenin çalışırken tükettiği elektrik enerji miktarları bezin cinsi, ağırlığı, kalınlığı, eni, pamuk karışımı gibi birçok parametreden etkilense de bekleme anında bez niteliği ile ilgili parametrelerin elektrik tüketimine bir etkisi yoktur. Bu sebeple, bekleme anındaki tüketimler ile ilgili tahmin yapılması olası fazla tüketimlerin önüne geçecektir. Bu sayede, işletmenin kaynak tüketimleri azaltılacaktır. Ayrıca beklenmeyen bir tüketim olduğu tahmin edilirse, makinede herhangi bir sorun olduğu ön görülüp gerekli bakımların yapılması sağlanabilir ve böylelikle hem makinenin zarar görmesi engellenmiş olacak hem de maliyetler en aza indirilecektir.

Elektrik enerjisi, makinelerde çok tüketilen maliyetli bir enerji türü olduğu için bu tez çalışmasında elektrik tüketim değerleri kullanılmıştır. Endüstride farklı alanlarda faaliyet gösteren işletmelerdeki tüm makineler elektrik tükettiği için tez kapsamında geliştirilen bu model farklı makineler için kullanılabilecektir.

2.4. Veri Ön İşleme Süreci

Verinin ön işleme sürecinden geçmesi için veri seti içerisindeki alanların özelliklerini ve birbirleriyle olan ilişkileri bilmek gerekmektedir. Bu sayede verileri tanıyıp potansiyel aykırı değerler görülebilmektedir ve veri setindeki eksik veriler tespit edilebilmektedir. Veri setini anlamak, uygun model ve parametre seçimi açısından oldukça önemlidir. Ayrıca sonuçları doğru yorumlamak için de veriyi anlamak gerekir.

Makinelerdeki sensörlerden toplanan veriler PLC aracılığı ile ilgili veri tabanı tablosuna yazılmaktadır. Veri tabanı tablosunda hem çalışma hem de bekleme anındaki birim sıcak su, soğuk su, buhar, doğalgaz ve elektrik tüketimleri yer almaktadır. Veri, makinelerden saatlik olarak gelmekte ve ilgili saatteki bekleme ve çalışma dakikaları veri tabanına kaydedilmektedir. Eğer bekleme anında tüketim değeri yoksa, makine kapalı konumda olduğu anlamına gelmektedir. Bu değer eğer sıfırdan büyükse ya da tüketim değerleri varsa makine bekleme durumundadır.

Analiz ve çalışmada kullanılan veri tabanı alanları, bekleme birim elektrik tüketimi, yıl ay gün formatında tarih, saat ve bekleme dakikasıdır. Elektrik tüketiminin birimi kilovat saat (KWh) olup çalışmada kullanılan veri seti Ocak 2021 ile Mart 2023 tarihleri arasında toplanan verilerden oluşmaktadır.

2.5. Verinin Hazırlaması

Makine öğrenmesi ve derin öğrenme projelerinde verinin hazırlanması projenin en önemli adımlarından birisidir. Verinin kalitesi, doğruluğu ve çeşitliliği modelin doğru çalışmasını ve performansını doğrudan etkilemektedir. Veri setindeki ön işleme veri hazırlama aşamasında yapılır. Eksik ve gereksiz verileri tespit etmek, veriyi standartlaştırmak ve modele girdi olarak vermek bu aşamada gerçekleşmektedir (Bengio vd., 2013: 77).

Veri hazırlama aşamasında veri setinin eğitim ve test verisi olarak doğru bir şekilde bölünmesi modelin performansı için önemlidir (Goodfellow vd., 2016: 232). Veri

hazırlama aşaması modelin performans ve doğruluğunu büyük ölçüde etkilediği için verinin işlenmesi ve doğru oranda bölünmesi modelin genelinde büyük rol oynamaktadır. Bu çalışmada, Microsoft Excel tablosu üzerinde yer alan ham veri, Google Colab çevrimiçi düzenleyicisine taşınmaktadır. Veri, amaca uygun şekilde düzenlenip doğru sonuca ulaşılmayı sağlayacak hale getirilmiştir. Bu durum ham verinin bilgi haline dönüşmesi yolculuğunda önemli bir rol oynamaktadır.

Verilerin kaydedildiği "xlsx" formatı, veriyi tablolar halinde saklayan ve matematiksel işlemler yapmaya olanak sağlayan bir yapıdır. "CSV" (Comma-Seperated Values) formatı ise verilerin virgüllerle ayrılarak saklandığı basit bir metin dosyası formatıdır. Excel'de sütunlar fiziksel olarak ayrılmışken CSV dosyalarında sütunlar virgüllerle ayrılmıştır. Büyük veri kullanılan projelerde Excel tabloları yerine CSV dosyaları tercih edilmektedir. Bunun en önemli sebebi, CSV dosyalarının Excel dosyalarına göre çok daha az depolama alanında aynı miktarda veriyi tutabilmesidir. Yani CSV dosyaları daha küçük boyutlara sahiptir. Ayrıca Excel dosyalarında tabloların karmaşık yapısı ve formül kullanılabilmesi sebebiyle veri işleme süreleri daha yavaştır.

Büyük veri projelerinde veri analitiği yapılırken CSV formatlı dosyalar veri analizi araçlarına daha kolay entegre edilebilir. Farklı sistemler arası uyumluluk ve veri alışverişi CSV dosyalarında daha kolay yapılabilmektedir. Bu nedenle Excel formatında alınan veriler, CSV formatına dönüştürülmüştür.

117 numaralı merserize makinesinden alınan saatlik birim elektrik tüketim verileri incelendiğinde, her saatte mutlaka bir bekleme olmadığı görülmüştür. Genellikle dörder saatlik aralıklarda mutlaka bez değişimi, vardiya değişimi ya da makine arızası gibi durumlar sebebiyle duruşların olduğu gözlemlenmiştir. Bu yüzden veri hazırlanırken 00-03, 04-07, 08-11, 12-15, 16-19, 20-23 saatleri gruplanarak birim elektrik tüketimleri ile durma dakikası çarpılarak toplamları alınmış ve bulunan sonuç toplam durma zamanına bölünerek 4 saatlik ortalama birim tüketim hesaplanarak veri seti düzenlenmiştir. Böylelikle bir güne ait toplam 6 adet veri elde edilmiştir.

Veriler gruplandıktan sonra 4 saatlik zaman diliminde bekleme değeri boş (null) olan verilerin olup olmadığı kontrol edilmiştir. Boş alanlar, veri setinin yapısını korumak amacıyla veri setinin tamamının medyan değeri ile doldurulmuştur. Böylelikle hesaplama yaparken ortaya çıkabilecek boş değer sorunları giderilmiştir. Bu çalışmada veri setinin orta noktasını işaret eden medyan değerinin tercih edilme sebebi, aykırı değerlerin etkisini

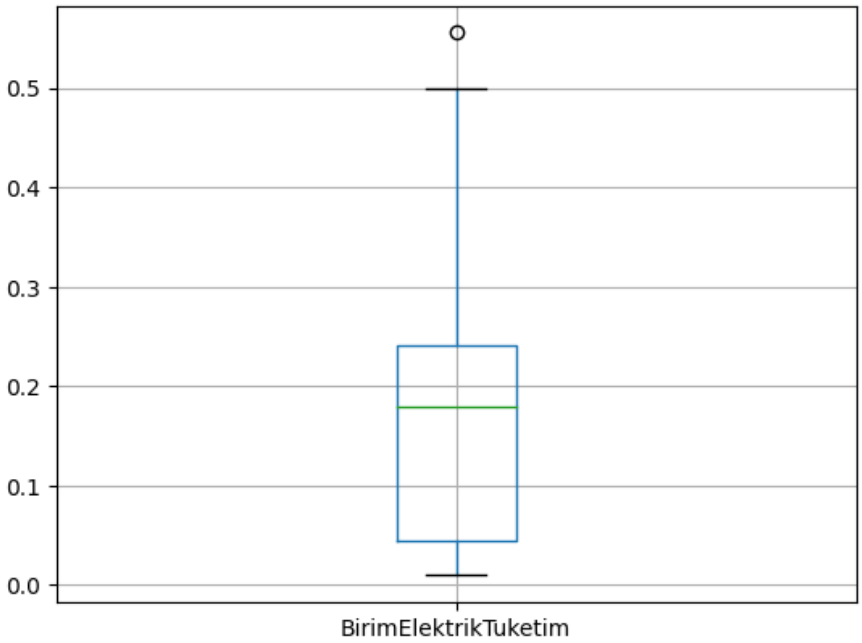
sınırlamaktır. Aykırı değerlerin aritmetik ortalama gibi diğer merkezi eğilim ölçü birimlerine etkisi medyana göre daha fazladır. Bu nedenle medyan kullanımı veri setinin bütünlüğünü koruması ve analizlerin güvenilirliğini artırmaktadır (Khan vd., 2019).

Farklı sütunlarda gelen yıl, ay, gün ve saat bilgileri analize uygun olacak şekilde YYYY-AA-GG SS:DD:SS formatına dönüştürülerek DateTime tipinde veri setine eklenmiştir. 813 güne ait, 4311 satır ve 4 sütundan oluşan veri setinin görüntüsünün başlangıç ve bitiş değerleri Tablo 1'de gösterilmiştir.

Tablo 1. Düzenlenmiş Veri Seti

Tarih	Toplam Elektrik Tüketimi	Bekleme Dakikası	Birim Elektrik Tüketimi
01.01.2021 00:00	3,99989	238,90000	0,01674
01.01.2021 04:00	3,99989	236,40000	0,01692
01.01.2021 08:00	2,99987	237,20000	0,01265
01.01.2021 12:00	3,99994	235,80000	0,01696
01.01.2021 16:00	2,99991	239,20000	0,01254
…	…	…	…
23.03.2023 20:00	13,00000	42,80000	0,30374
24.03.2023 00:00	24,00000	90,00000	0,26667
24.03.2023 04:00	30,00000	125,00000	0,24000
24.03.2023 08:00	25,00000	222,50000	0,11236
24.03.2023 12:00	17,00000	48,10000	0,35343

Veri setinin dağılımı ve merkezi eğilimi Şekil 13'te gösterilmiştir. Veri setine ait kutu grafiğinde, kutunun içinde yer alan yatay çizgi orta değer yani medyanı göstermektedir. Kutunun alt ve üst kenarları ise alt çeyrek ve üst çeyrek değerlerini, yani veri setinin üst yüzde 25'i ile alt yüzde 25'ini kapsayan kısmı göstermektedir.



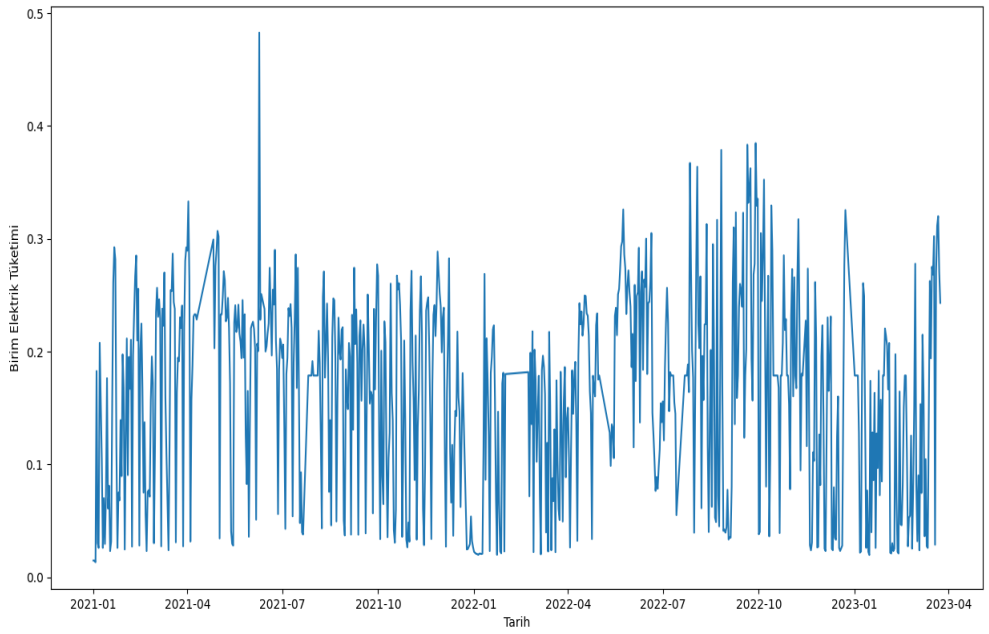
Şekil 13. Veri Seti Kutu Grafiği

Veri seti bir zaman serisi olduğu için sıra numarası değeri olarak tarih sütunu belirlenmiştir ve bu değerler işaretlenmiştir. Veri seti analize hazırladıktan sonra temizlenerek CSV formatına dönüştürülmüştür. Verileri daha iyi analiz edebilmek için veri setinde yer alan birim elektrik tüketimi sütununun günlük ve haftalık ortalamaları alınmıştır. Tablo 2 günlük ortalamaları ortalamaların başlangıç ve bitiş değerlerini içermektedir.

Tablo 2. Günlük Birim Elektrik Tüketim Ortalamaları

Tarih	Birim Elektrik Tüketim
01.01.2021	0,01475
02.01.2021	0,01456
03.01.2021	0,01332
04.01.2021	0,18267
05.01.2021	0,03023
06.01.2021	0,02599
…	…
19.03.2023	0,02880
20.03.2023	0,27090
21.03.2023	0,31138
22.03.2023	0,31988
23.03.2023	0,27077
24.03.2023	0,24311

Günlük birim elektrik tüketim ortalamalarına ait değerlerin grafiği Şekil 14'te gösterilmiştir.



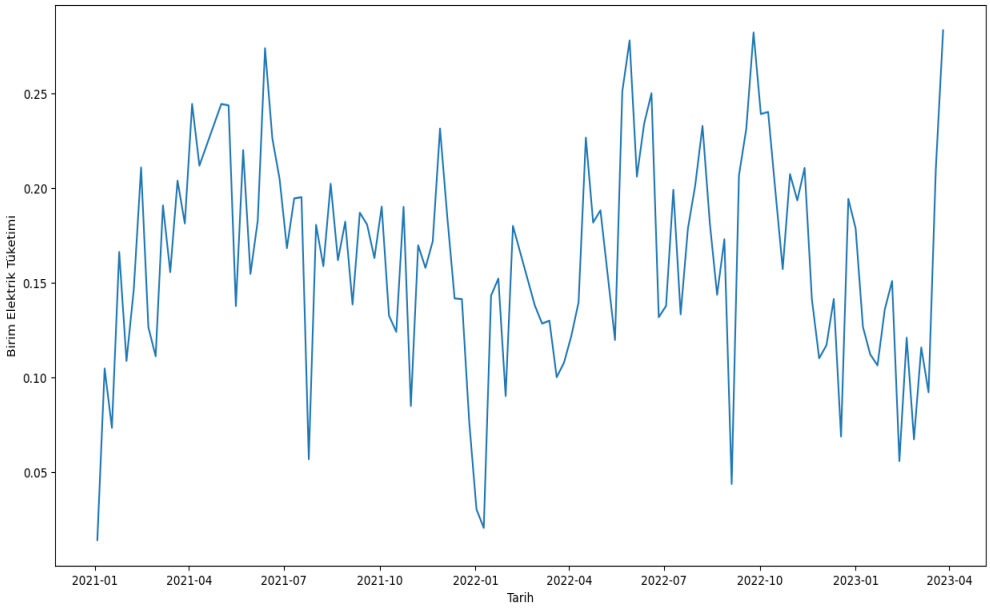
Şekil 14. Günlük Ortalama Elektrik Tüketimleri

Veri setine ait değerlerin haftalık olarak da ortalamaları alınarak daha geniş aralıktaki değerlere ait başlangıç ve bitiş değerleri Tablo 3'te verilmiştir.

Tablo 3. Haftalık Birim Elektrik Tüketim Ortalamaları

Tarih	Birim Elektrik Tüketim
03.01.2021	0,01421
10.01.2021	0,10472
17.01.2021	0,07343
24.01.2021	0,16620
31.01.2021	0,10874
07.02.2021	0,14658
…	…
19.02.2023	0,12097
26.02.2023	0,06737
05.03.2023	0,11580
12.03.2023	0,09221
19.03.2023	0,21037
26.03.2023	0,28321

Haftalık birim elektrik tüketim ortalamalarına ait değerlerin grafiği Şekil 15'te gösterilmiştir.



Şekil 15. Haftalık Ortalama Elektrik Tüketimleri

Günlük ve haftalık birim elektrik tüketim ortalamalarına ait değerlerin grafikleri incelendiğinde verilerin düzenli bir artış veya azalış eğilimi göstermediği ve aykırı değerlerin veri setinde yer aldığı görülmektedir.

2.6. Parametrelerin Belirlenmesi

Tez kapsamında geliştirilen LSTM, GRU ve RNN algoritmalarının her biri için derin öğrenme modeli oluşturulurken ilk olarak parametreler belirlenmiştir.

Parametrelerin işlevleri ve etkilediği alanlar şu şekilde açıklanabilir;

1. **Units:** Derin öğrenme modellerinde "units" parametresi bir katmanda yer alan hücre sayısını (nöron sayısı) belirlemek amacıyla kullanılmaktadır. Daha fazla nöron modelin daha fazla özellik öğrenmesine ve daha karmaşık ilişkileri modellemesine olanak sağlamaktadır. Çok fazla nöron kullanılırsa modelin hızı da etkilenecektir. Bu yüzden birim sayısı seçilirken modelin karmaşıklığı göz önünde bulundurulmalıdır. Veri seti çok karmaşık değilse birim sayısını çok yüksek tutmaya gerek olmayacaktır. Birim sayısı arttıkça model ezberleme eğilimine de yönelebilmektedir. Birim sayısı ile ilgili kesin bir kural olmamakla birlikte Goodfellow vd. (2016: 219), doğrusal olmayan problemlerde genel olarak birim sayısını giriş boyutunun 10 katı

olarak önermektedir. Modelde units değeri için optimum sayı olarak 100 belirlenip kullanılmıştır.

2. **SEED:** Rastgele sayı üretip bir parametreye atama yapıldığında aynı sonuçları tekrarlamak için kullanılmaktadır. Başlangıç ağırlıkları ve değerleri tutularak modelin eğitimi tekrarlandığında aynı sonuçlar bu parametre sayesinde alınabilmektedir. Farklı SEED değerleri kullanılarak model farklı başlangıç noktalarından başlatılarak sonuçlar arası karşılaştırma yapılabilir. Modelde SEED değeri rastgele belirlenerek 123 olarak tanımlanmıştır.
3. **Learning_rate:** Derin öğrenme modellerindeki en önemli parametrelerden birisidir. Bu değer ne kadar yüksek olursa modeldeki ağırlıklar daha hızlı güncellenir ve eğitim süreci daha da hızlanır. Fakat burada aşırı uyum riski ortaya çıkabilmektedir. Bu parametrenin değerinin çok düşük tutulması ise modelin yavaşlamasına sebep olacaktır. Bu yüzden iyi performans ve hızlı yakınsama arasında bir denge kurulması gerekmektedir. Öğrenme hızı genelde deneme yanılma yoluyla belirlenmektedir (Goodfellow vd., 2016: 293). Modelde 0,01, 0,008, 0,005, 0,001 gibi çeşitli değerler denenerek en iyi değerin 0,005 olduğu görülmüştür.
4. **Layer:** Derin öğrenme modellerinde çeşitli katman türleri kullanılmaktadır. Zaman serisi modellerde RNN, GRU (Gated Recurrent Unit) ve LSTM gibi özyinelemeli katmanlar sıklıkla tercih edilmektedir. RNN, GRU ve LSTM zaman serisi modellerinde zaman bağımlılıklarını etkin bir şekilde modellemek için geliştirilmiş katman türleridir (Rajagukguk vd., 2020: 6623). LSTM diğer katmanlara göre daha uzun vadeli bağımlılıkları daha iyi yakalayabilmektedir (Cho vd., 2014). LSTM'de hafıza hücrelerinde giriş ve çıkış kontrol mekanizmaları yer almaktadır. Bu nedenle bilginin saklanması ve aktarılması konusunda avantaj sağlamaktadır (Greff vd. 2016: 2225). Modelde LSTM, GRU ve RNN katmanları kullanılarak performans ve hata karşılaştırmaları yapılmıştır.
5. **activation_function:** Aktivasyon fonksiyonunu belirlemektedir. Aktivasyon fonksiyonları derin öğrenme modellerindeki matematiksel fonksiyonlardır. Katmanlar arası bilgi akışını kontrol eden doğrusal

olmayan işlevi belirler. Aktivasyon fonksiyonları girdi verilerini dönüştürür ve çıktıları belirler. Böylece doğrusal olmama durumu sağlanmış olacaktır. Sigmoid fonksiyonu, giriş değerlerini [0, 1] aralığına sıkıştırır. Bu fonksiyon, özellikle ikili sınıflandırma problemlerinde kullanılır. $Tanh$ fonksiyonu, giriş değerlerini [-1, 1] aralığına sıkıştırır. Sigmoid fonksiyonuna benzer şekilde kullanılan bir aktivasyon fonksiyonudur (Goodfellow vd., 2016: 208). ReLU fonksiyonu, negatif giriş değerlerini sıfır yaparken, pozitif giriş değerlerini olduğu gibi bırakır. ReLU, son zamanlarda popülerlik kazanan ve derin öğrenme modellerinde yaygın olarak kullanılan bir aktivasyon fonksiyonudur (Nair ve Hinton, 2010: 811). Modelde daha performanslı olduğu için 'ReLU' kullanılmıştır.

6. **dropout_rate:** Modelin eğitimi sırasında birimin devre dışı bırakılma olasılığını dropout (bırakma veya atma) oranı belirler. Bu sayede, her birim eğitimde bağımsız bir şekilde çalışır ve diğer birimlerin çıktılarına bağımlı hale gelmez. Dropout, ağın aşırı uyumu azaltarak daha genelleştirilebilir ve daha iyi bir performans elde etmesini sağlar. Optimum dropout oranı, modelin karmaşıklığına, veri setinin büyüklüğüne ve diğer hiperparametrelerin değerlerine bağlı olarak değişebilir. Genel olarak, dropout oranı %20 ile %50 arasında seçilir. Ancak, en iyi oranın deneysel olarak belirlenmesi önerilmektedir (Srivastava vd., 2014: 1933). Modelde, droput_rate için 0,05, 0,2 ve 0,1 gibi farklı değerler denenerek, optimum değer 0,1 olarak belirlenmiştir.
7. **epoch:** Eğitim sürecindeki toplam iterasyon sayısını belirler. Eğitim sürecindeki tüm aşamaların bir kez aktarıldığı bir geçişi temsil etmektedir. Epoch sayısı arttıkça model daha fazla eğitim verisiyle karşılaşır ve daha iyi bir genelleme yapabilir. Epoch sayısı gereğinden fazla artırılırsa aşırı öğrenme (overfitting) gerçekleşebilir. Bu yüzden epoch veri setinin özelliğine, problemin amacına, kullanılan algoritmalara ve modelin karmaşıklığına göre optimum sayıda olmalıdır. Modelde, epoch sayısı olarak 10, 12, 15, 20 ve 50 değerleri denenmiş ve yapılan denemeler sonucunda en uygun epoch sayısı 15 olarak belirlenmiştir.

8. **batch_size:** Eğitim verilerinin kümelere bölünme boyutunu belirlemektedir. Batch_size büyüdükçe daha fazla bellek kullanılmaktadır ve işlem yükü de büyümektedir. Aynı zamanda daha fazla paralel işi aynı anda yürümekte ve eğitim süreci hızlanmaktadır. Ancak batch_size değerinin artmasıyla aşırı öğrenme riski de ortaya çıkabilecektir. Ayrıca eğitim süreci hızlansa da hesaplama süreci uzun sürebilecektir. Bununla birlikte hafıza sorunları da yaşanabilecektir. Bazı durumlarda daha küçük batch_size boyutu daha fazla varyasyon oluşturarak daha iyi genelleme yapmayı sağlayabilmektedir. Batch_size ayarlanırken donanım kaynakları, veri setinin büyüklüğü ve modelin karmaşıklığına bakılmaktadır. Modelde batch_size için 32, 64, 128 ve 256 gibi farklı değerler denenerek en iyi sonucu elde eden değerin 128 olduğu görülmüştür.
9. **loss_function:** Hedef değişken ile tahminler arasındaki kaybı ölçmek için kullanılan fonksiyonu belirlemektedir. Bu parametrenin en iyi değeri, kayıp fonksiyonunu minimize edecek şekilde güncellenerek bulunmaktadır. Kayıp fonksiyonu düştükçe modelin performansı artmaktadır.

   Yaygın kullanılan kayıp fonksiyonlarından bazıları şunlardır (Chollet, 2018: 83, Nielsen, 2015: 67):

   a. Ortalama Kare Hatası (Mean Squared Error – MSE): Regresyon problemlerinde kullanılmaktadır. Tahmin edilen değerler ile gerçek değerler arasındaki hata karelerinin ortalamasını hesaplar. Büyük farkların daha yüksek, küçük farkların daha düşük bir hata olarak değerlendirildiği bir ölçüdür. MSE, Denklem 10'daki gibi hesaplanmaktadır.

$$MSE = \frac{1}{n} \times \sum (y - \hat{y})^2 \qquad (10)$$

Bu denklemde, n veri noktalarının sayısını, y gerçek değeri ve $\hat{y}$ tahmin edilen değeri göstermektedir.

b. Kök Ortalama Kare Hatası (Root Mean Squared Error – RMSE): MSE'nin karekökü alınarak elde edilmektedir. MSE'ye göre avantajı, hata ölçüsünün orijinal ölçek biriminde olmasıdır. RMSE, Denklem 11'deki gibi hesaplanmaktadır.

$$RMSE = \sqrt{\frac{1}{n} \times \sum (y - \hat{y})^2} \qquad (11)$$

Bu denklemde, n veri noktalarının sayısını, y gerçek değeri ve $\hat{y}$ tahmin edilen değeri göstermektedir.

c. Ortalama Mutlak Hata (Mean Absolute Error – MAE): Regresyon problemlerinde kullanılmaktadır. Gerçek değerlerle tahmin edilen değerler arasındaki mutlak farkların ortalamasını hesaplamaktadır. Her bir farkın mutlak değeri alındığı için, pozitif ve negatif hataların etkisi aynıdır. Bu sebeple işaret bilgisi dikkate alınmamaktadır. MAE değeri ne kadar düşükse modelin tahminleri gerçeğe o kadar yakın olacaktır. Aykırı değerlerin minimize edilmeye çalışılan modellerde tercih edilmektedir. MAE, Denklem 12'deki gibi hesaplanmaktadır.

$$MAE = \frac{1}{n} \times \sum |y - \hat{y}| \qquad (12)$$

Bu denklemde, n veri noktalarının sayısını, y gerçek değeri ve $\hat{y}$ tahmin edilen değeri göstermektedir. Modelde kayıp fonksiyon göstergesi olarak MAE kullanılmıştır.

10. optimizer: Derin öğrenme modelinde optimizasyon (eniyileme) işlemini yapmaktadır. Modelin kayıp fonksiyonunu minimize etmek ve en iyi performansı elde etmek için parametrelerin güncellenmesini sağlamaktadır. Yaygın kullanılan optimizer (eniyileyici) çeşitleri şunlardır (Nielsen, 2015: 67, Chollet, 2018: 83):

a. Stokastik Gradyan İnişi (Stochastic Gradient Descent - SGD): Temel optimizasyon algoritmasıdır. Her adımda rastgele seçilen örneklerden gradyan hesaplanarak parametreler güncellenmektedir.

b. Uyarlanabilir Moment Tahmini (Adaptive Moment Estimation - Adam): Gradyanların hareketli ortalamalarını ve ikinci momentlerini hesaplayarak parametre güncellemesi yapmaktadır.
c. Kök Ortalama Kare Yayılımı (Root Mean Square Propagation - RMSprop): Gradyan karelerinin hareketli ortalamasını kullanarak parametreleri günceller.
d. Uyarlanabilir Gradyan (Adaptive Gradient - Adagrad): Her parametrenin ayrı ayrı öğrenme hızını hesaplar ve bu hızı adaptif bir şekilde ayarlar.

Adam optimizer, hızlı yakınsama, bellek verimliliği, adaptif öğrenme hızı, düzlemsel kısıtlama ve uygun hiperparametre seçimi gibi avantajlara sahiptir. Bu yüzden çalışmada Adam optimizer kullanılmıştır (Kingma ve Ba, 2014: 10).

11. **return_sequence:** LSTM katmanlarının çıktılarının veya sonuçlarının tüm zaman adımlarında dönmesini sağlar. Bunu sağlamak için çalışmada bu parametre "true" olarak ayarlanmıştır.
12. **target:** Tahmin yapılacak hedef değişkeni belirler. Çalışmada "BirimElektrikTuketim" olarak belirlenmiştir.
13. **n_hours:** Geçmişteki verilere dayanarak bir sonraki 4 saatlik dönemi tahmin etmek için kullanılacak saat sayısını belirler. Çalışmada 24 olarak verilmiş, her 4 saatlik verinin ortalaması alındığı için 96 öncesinden itibaren veriler ele alınmıştır.
14. **train_size:** Veri setinin eğitim için kullanılacak bölümünün yüzdesini belirler. Çalışmada, eğitim ve test veri setleri %80-%20, %70-%30 ve %50-%50 olarak ayrılacak denemeler yapılmış ve en uygun sonuca ulaşılan oran olduğu için eğitime %80 ayrılmış ve 0,8 olarak parametre ayarlanmıştır.

Modelde kullanılan tüm parametrelere ait değerler Tablo 4'te gösterilmiştir.

Tablo 4. Model Parametreleri

Parametre	Açıklama	Değer
units	Hücre Sayısı	100
SEED		123
learning_rate	Öğrenme Oranı	0,005
Layer	Katman Türü	LSTM, GRU, RNN
activation_function	Aktivasyon Fonksiyonu	ReLU
dropout_rate	Bırakma (Atma) Oranı	0,1
epoch	Tekrar Sayısı	15
batch_size	Küme Bölünme Boyutu	128
loss_function	Kayıp Fonksiyonu	MAE
optimizer	Optimizasyon Yöntemi	Adam
Önceki Veri		24
Gelecek Tahmin		1
train_size	Eğitim Boyutu	80%
test_size	Test Boyutu	20%

Parametre değerleri belirlenirken deneme yanılma yöntemi kullanılmış ve performans, süre ve doğruluk açılarından en uygun parametreler bulunmuştur. Ancak, Hiperparametre Optimizasyonu (Hyperparameter Tuning) tercih edilmemiştir, çünkü zaman alıcı olabileceği, kaynak ve hesaplama gücü gerektirebileceği ve aşırı uyum sorunu yaşanabileceği göz önünde bulundurulmuştur. Ayrıca Adam optimizer, hiperparametreleri güncelleyerek eğitim sürecini optimize etmektedir.

2.7. Normalizasyon

Veri setindeki veri özellikleri arasındaki ölçek farklılıklarını ortadan kaldırmak için normalizasyon yöntemleri kullanılır. Normalizasyonda amaç, analiz ve modelleme işlemlerinde daha tutarlı sonuçlar elde etmektir. Böylelikle farklı özellikler arasındaki farklı ölçekleri dengelemek, aykırı değerlerin etkisini azaltmak ve bazı algoritmaların performansını artırmak istenmektedir (Nguyen, 2018).

Çalışmada, veri özelliklerini (sütunları) normalize etmek için Min-Max ölçeklendirme yöntemi kullanılmıştır. İşlemler aşağıdaki adımlarla gerçekleştirilir:

- MinMaxScaler sınıfı, sklearn.preprocessing modülünden çağrılmıştır.
- feature_range=(0, 1) parametresiyle bir scaler örneği oluşturulmuştur. Bu, ölçeklendirilmiş değerlerin 0 ile 1 aralığında olmasını sağlamaktadır.

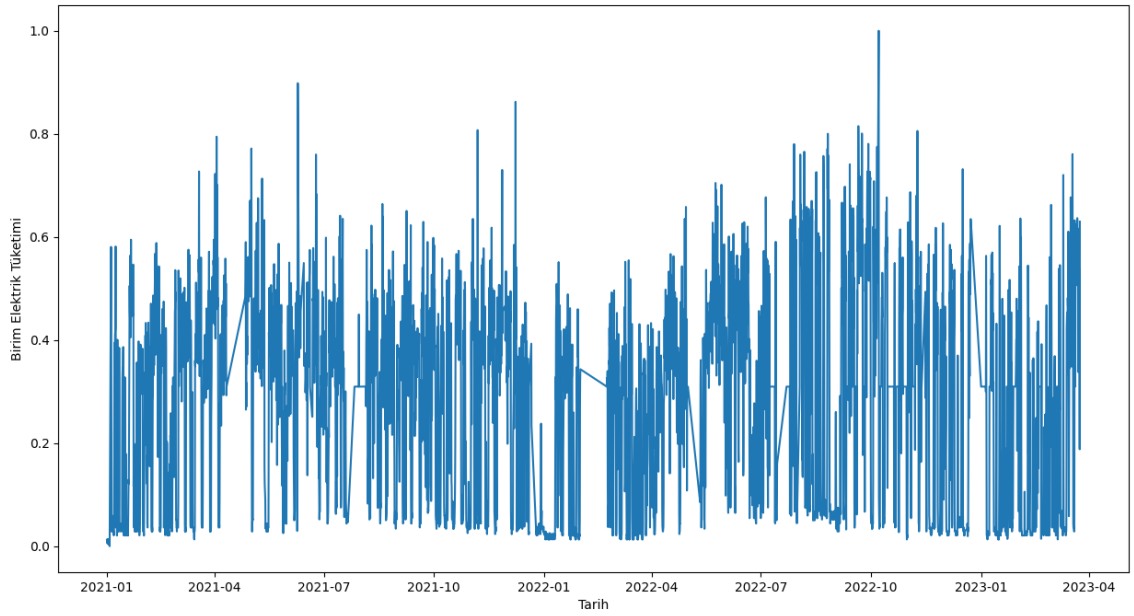
- scaler nesnesi, fit_transform yöntemiyle seçilen sütunların (cols_to_analyze) yani analiz için seçilen sütunun değerlerini normalize eder. fit_transform yöntemi, önce veriye uyarlanır (fit) ve ardından veriyi dönüştürür (transform).
- Ölçeklendirilmiş verileri içeren scaled_df adlı bir DataFrame olarak oluşturulur. Bu, orijinal veri çerçevesinin aynı indeksini ve sütunlarını korur.

Normalizasyon, verilerin farklı ölçeklerde olması durumunda bir modelin yanlış sonuçlar vermemesini sağlamak için sıklıkla kullanılır. Örneğin, bir özelliğin değerleri 0 ile 1000 arasında iken diğer bir özelliğin değerleri 0 ile 1 arasında ise, bu farklı ölçekler modelin doğruluğunu etkileyebilir. Bu nedenle, tüm özelliklerin aynı ölçekte olması istenmektedir. Tablo 5'te normalize edilmiş veri setinin bir bölümü yer almaktadır.

Tablo 5. Normalize Edilmiş Veri Setinin Görünümü

Tarih	Birim Elektrik Tüketimi
01.01.2021 00:00	0,01289
01.01.2021 04:00	0,01321
01.01.2021 08:00	0,00538
01.01.2021 12:00	0,01329
01.01.2021 16:00	0,00519
01.01.2021 20:00	0,00547
…	…
23.03.2023 16:00	0,49149
23.03.2023 20:00	0,53867
24.03.2023 00:00	0,47075
24.03.2023 04:00	0,42190
24.03.2023 08:00	0,18806
24.03.2023 12:00	0,62970

Normalize edilmiş veri setine ait değişimlerin görselleştirilmesi için çizilen grafik Şekil 16'da yer almaktadır. Grafikte y eksenine ait değerlerin 0 ile 1 arasında değiştiği görülmektedir.



Şekil 16. Normalize Edilmiş Veri Setinin Grafiği

Normalize edilmiş veri setine ait grafik incelendiğinde, uç değerlerin orijinal veri setinde olduğu gibi hala mevcut olduğu gözlemlenmiştir. Bu durum, veri setinin ölçeklendirme işlemine tabi tutulduğunu ve modelin doğruluğu için uygun bir veri setinin oluşturulduğunu göstermektedir. İlerleyen aşamalarda, tahminleme yapıldıktan sonra tahmin edilen değerlerin ölçeklendirmesi tersine çevrilecektir ve tahmin edilen değerler orijinal ölçeklerine geri dönüştürülmüş olacaktır.

2.8. Verilerin Yeniden Çerçevelenmesi

Veri setini denetimli öğrenme modeline dönüştürmek için "series_to_supervised" adlı bir fonksiyonu tanımlanmıştır. Derin öğrenme modelleri genellikle denetimli öğrenme üzerine kuruludur ve giriş verilerini (özelliklerini) ve hedef çıktılarını (etiketleri) eşleştirmek için bu tür bir dönüşüm gerekmektedir.

Fonksiyonun parametreleri şunlardır:

data: Dönüştürülecek veri serisi.

n_in: Geçmiş veri sayısı, yani giriş dizisinin ne kadar geriye gideceği.

n_out: Gelecek veri sayısı, yani çıktı dizisinin kaç adım sonrasını tahmin edeceği.

dropnan: NaN değerlere sahip satırların düşürülüp düşürülmeyeceğini belirten bir bayrak.

Fonksiyonun çalışma adımları şunlardır:

1. Giriş veri serisi data bir DataFrame'e dönüştürülmüştür.
2. cols ve names adında boş liste oluşturulmuştur.
3. Döngü kullanarak, geçmiş veri dizileri (var(t-n) şeklinde) cols listesine eklenmiştir ve names listesine sütun adları eklenmiştir.
4. Bir başka döngü ile gelecek veri dizileri (var(t) veya var(t+n) şeklinde) cols listesine eklenmiştir ve names listesine sütun adları eklenmiştir.
5. cols listesindeki tüm veri dizileri birleştirilerek agg adında yeni bir DataFrame oluşturulmuştur.
6. Sütun adları names listesindeki değerlerle güncellenmiştir.
7. dropnan parametresi True olarak ayarlanmış ve NaN değerlere sahip satırlar düşürülmüştür.
8. Son olarak, oluşturulan dönüştürülmüş veri DataFrame'i (agg) döndürülmüştür.

Bu dönüşüm, derin öğrenme modellerinde zaman serilerinin giriş-çıktı eşleştirmesi gerektiği durumlarda kullanılır. Geçmiş zaman adımlarının giriş verileri olduğu ve gelecek zaman adımlarının hedef çıktılar olduğu bir yapı oluşturur. Böylece model, geçmiş verilere dayanarak gelecekteki değerleri tahmin edebilecektir.

"series_to_supervised" adlı fonksiyon oluşturulmuş ve çağrılmaya hazır hale getirilmiştir. Bu fonksiyonun çağrılması ile veri serisinin önceki saatlere geriye doğru dayanan bir şekilde yeniden çerçevelenmesini gerçekleşmektedir. Ayrıca, hedef değişkenin belirtilen boyutta tanımlanması ve gereksiz sütunların düşürülmesi işlemleri de yapılmaktadır.

İlk olarak, “series_to_supervised” fonksiyonu kullanılarak veri serisi, n_hours parametresiyle belirtilen sayıda saat geriye gidecek şekilde yeniden yapılandırılmıştır. Burada scaled_df veri seti to_numpy() metoduyla Numpy dizisine dönüştürülerek fonksiyona verilir. Bu işlem, her satırın geçmiş saatlere ait değerlerle birlikte gelecekteki değerleri içerecek şekilde bir çerçeve oluşturur. Bir başka deyişle, değişkenin 24 birim önceki değeri, 23 birim önceki değeri olarak başlayıp mevcut değerine kadar ayrı sütunlar oluşturulmuş ve çerçeveleme yapılmıştır.

Çerçeveleme bu aşamada yapıldığından dolayı model için oldukça önemlidir. Çerçeveleme sonrası, başlangıç ve bitiş değerleri Tablo 6’da yer alan, 4287×25

boyutunda bir reframed_df veri çerçevesi oluşmuştur. Bunun ardından, reframed_df adında bir DataFrame oluşturulmuştur ve series_to_supervised fonksiyonundan dönen sonuç bu DataFrame'e atanmıştır. scaled_df veri seti to_numpy() metoduyla Numpy dizisine dönüştürülerek fonksiyona girdi parametresi olarak verilmiştir. len([target]) ifadesi ile hedef değişkenin boyutu belirlenmiştir. Burada, sadece 1 boyutlu hedef değişken alındığı için len([target]) ifadesi 1'dir.

Son olarak, reframed_df DataFrame'inin indeksi, df veri setinin indeksi n_hours kadar atlandıktan sonra başlayacak şekilde güncellenmiştir. Çünkü, geçmiş saatlere dayalı çerçevenin oluşabilmesi ve veri setinin tam olarak doldurulabilmesi için ilk n_hours kadar verinin atlanması gerekmektedir. Bu sebeple, veri seti 4311 satır veri içermesine rağmen, çerçevelenirken ilk 24 veri çıkarıldığı için çerçevelenmiş veri setinde 4287 satır vardır.

Bu aşamada kayan pencere yöntemi ile çerçeveleme uygulanmıştır. Çerçevelenen veri setinde ilk 24 sütun girdi, 25'inci sütun çıktı değişkeni olarak ayrıştırılacaktır. Girdi değişkeni x, çıktı değişkeni y olarak etiketlenmiştir. İlk olarak, x değişkeni, reframed_df DataFrame'inin son sütunu hariç tüm sütunlarını içerecek şekilde tanımlanmıştır. Yani, girdi değişkenleri, çerçeveleme işlemi sonucu oluşan DataFrame'in son sütunu dışındaki tüm sütunlardan oluşmuştur.

y değişkeni ise reframed_df DataFrame'inin sadece son sütununu içerecek şekilde tanımlanmıştır. Yani, çıktı değişkeni, çerçeveleme işlemi sonucu oluşan DataFrame'in son sütunudur. Böylece çerçeveleme işlemi sonrası elde edilen DataFrame'den girdi ve çıktı değişkenlerini ayırarak derin öğrenme modelinin kullanabileceği şekilde veri seti hazırlanmıştır. Girdi değişkeni (x) DataFrame'in tüm sütunlarını, son sütunu hariç alırken, çıktı değişkeni (y) sadece son sütunu almıştır. x değişkeninin boyutu 4287×24, y değişkeninin boyutu ise 4287×1'dir.

Tablo 6. Çerçevelenmiş Veri Seti

Tarih	var1(t-24)	var1(t-23)	var1(t-22)	var1(t-21)	var1(t-20)	var1(t-19)	var1(t-18)	var1(t-17)	var1(t-16)	var1(t-15)	var1(t-14)	var1(t-13)	var1(t-12)	var1(t-11)	var1(t-10)	var1(t-9)	var1(t-8)	var1(t-7)	var1(t-6)	var1(t-5)	var1(t-4)	var1(t-3)	var1(t-2)	var1(t-1)	var1(t)
06.01.2021 00:00	0,01289	0,01321	0,00538	0,01329	0,00519	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616
06.01.2021 04:00	0,01321	0,00538	0,01329	0,00519	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626
06.01.2021 08:00	0,00538	0,01329	0,00519	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857
06.01.2021 12:00	0,01329	0,00519	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855
06.01.2021 16:00	0,00519	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088
06.01.2021 20:00	0,00547	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088	0,02855
07.01.2021 00:00	0,01323	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088	0,02855	0,24706
07.01.2021 04:00	0,00538	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088	0,02855	0,24706	0,39446
07.01.2021 08:00	0,01319	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088	0,02855	0,24706	0,39446	0,34716
07.01.2021 12:00	0,00306	0,01304	0,00541	0,01324	0	0,58061	0,46724	0,19108	0,02853	0,04405	0,05951	0,02852	0,03628	0,02861	0,02859	0,03616	0,03626	0,02857	0,02855	0,02088	0,02855	0,24706	0,39446	0,34716	0,36156
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
23.03.2023 00:00	0,04367	0,03655	0,0361	0,03651	0,02844	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813
23.03.2023 04:00	0,03655	0,0361	0,03651	0,02844	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512
23.03.2023 08:00	0,0361	0,03651	0,02844	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036
23.03.2023 12:00	0,03651	0,02844	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582
23.03.2023 16:00	0,02844	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149
23.03.2023 20:00	0,02857	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149	0,53867
24.03.2023 00:00	0,26769	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149	0,53867	0,47075
24.03.2023 04:00	0,63228	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149	0,53867	0,47075	0,4219
24.03.2023 08:00	0,30901	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149	0,53867	0,47075	0,4219	0,18806
24.03.2023 12:00	0,59405	0,47117	0,59683	0,57102	0,54091	0,56022	0,53519	0,5175	0,59112	0,62552	0,5064	0,63651	0,5751	0,52173	0,54418	0,33813	0,39512	0,49036	0,61582	0,49149	0,53867	0,47075	0,4219	0,18806	0,6297

2.9. Eğitim ve Test Verilerinin Ayrıştırılması

Eğitim ve test verilerinin ayrıştırılması aşamasında, train_test_split fonksiyonu kullanılarak x ve y veri setleri, belirtilen train_size oranına göre yani yüzde 80 eğitim seti ve yüzde 20 test seti olacak şekilde bölünmüştür. shuffle=False parametresi ise sıralı veri olduğu için verilerin karıştırılmamasını sağlamak için kullanılmıştır. Zaman serili veri setlerinde karıştırma işlemi yapılmamalıdır. Bu yüzden, bu parametre çok önemlidir.

x_train_df, x_test_df, y_train_df, y_test_df değişkenleri, eğitim ve test setlerinin DataFrame formatında tutulması için oluşturulmuştur. Daha sonra, veri seti eğitim ve test setlerine ayrıştırılmıştır ve veriyi LSTM, GRU ve RNN gibi algoritmalarla uyumlu hale getirmek için yeniden şekillendirme (reshape) işlemi yapılmıştır. Bu algoritmaların uyumlu bir şekilde çalışabilmesi için veri seti 2 boyutlu halden 3 boyutlu hale getirilmiştir. x_train_df ve x_test_df DataFrame'leri Numpy dizilerine (to_numpy()) dönüştürülerek 3 boyutlu hale burada getirilmiştir. reshape işlemi, LSTM, GRU ve RNN gibi algoritmaların girdi olarak 3 boyutlu veri gerektirdiği için yapılmıştır. x_train_df ve x_test_df Numpy dizileri, (örnek sayısı, zaman adımları, özellik sayısı) şeklinde bir çerçeveleme yapar. x_train_df.shape[0] satır sayısını, n_hours zaman adımlarını ve scaled_df.shape[1] özellik sayısını içerecek şekilde yeniden şekillendirilmiştir.

x_train veri setinin boyutu 3429 satır, 24 sütun ve bu satır ve sütunlardan 1 tane olacak şekilde 3429×24×1 olarak şekillendirilmiştir. Benzer şekilde, x_test veri setinin boyutu 858 satır ve 24 sütun ve bu satır ve sütunlardan 1 tane olacak şekilde 858×24×1 olarak şekillendirilmiştir. Bu kısımdaki en önemli durumlardan birisi veri setinin sıralı bir şekilde bölünmüş olması ve veri setinin 3 boyutlu bir yapıya dönüştürülmesidir.

2.10. Derin Öğrenme Modelinin Oluşturulması

Eğitim ve test verileri ayrıştırılıp veri seti algoritmalara uygun hale getirildikten sonra artık derin öğrenme modelinin kurulması ve eğitimin gerçekleştirilmesi aşamasına geçilebilecektir. LSTM algoritması kullanılarak aşağıdaki aşamalar gerçekleştirilmiştir:

1. Sequential sınıfı kullanılarak bir Keras modeli oluşturulmuştur. Sequential sınıfı, ardışık katmanlardan oluşan bir sinir ağı modeli tanımlamak için kullanılır.
2. model.add yöntemiyle katmanlar sırasıyla eklenmiştir. İlk olarak, LSTM katmanı eklenmiştir. LSTM hücrelerinin sayısını belirtmek için units

parametresi, girdi verisinin boyutunu tanımlamak için input_shape kullanılmıştır. x_train.shape[1] zaman adımlarını, x_train.shape[2] ise özellik sayısını temsil etmektedir. LSTM katmanının çıktılarını döndürmesini ve bir sonraki LSTM katmanına aktarmasını sağlamak için return_sequences=True şeklinde atama yapılmıştır. Daha sonra, Dropout katmanı eklenmiştir. dropout_rate parametresi, ağdaki bağlantıların rastgele olarak atılma olasılığını belirlemektedir. Modelde aşırı öğrenmeyi (overfitting) engellemek için kullanılmıştır.

3. Daha sonra, ikinci bir LSTM katmanı ve bir Dropout katmanı daha eklenmiştir. Bu, modelin daha karmaşık ilişkileri öğrenmesine ve daha iyi bir performans elde etmesine olanak tanır.
4. Son olarak, tek bir çıktıya sahip bir Dense katmanı eklenmiştir. Bu katman, tahminlerin yapılmasını sağlamaktadır. Dense katmanı, bir modelin öğrenmesi gereken karmaşık ilişkileri ifade etme yeteneğine sahiptir. Genellikle modelin orta ve çıktı katmanlarında bulunur.
5. model.compile yöntemiyle model derlenmiştir. loss parametresi, kayıp fonksiyonunu belirtmek için kullanılmıştır. optimizer parametresi, modelin eniyileme algoritmasını belirlemektedir.
6. model.fit yöntemiyle model eğitilmiştir. x_train ve y_train_df.to_numpy() girdileri parametre olarak verilmiştir. Ayrıca eğitim döngüsünün kaç kez tamamlanacağını belirtmek için epochs parametresi, her güncelleme adımında kullanılacak örnek sayısını belirlemek için batch_size parametresi parametresi verilmiştir. verbose parametresi, eğitim ilerlemesinin yazdırılmasını sağlamaktadır. shuffle=False parametresi, verilerin karıştırılmamasını sağlamaktadır, çünkü zaman serisi verilerinde karıştırma işlemi yapılmamalıdır.

Modelde kullanılan epoch ve batch_size gibi parametreler modelin eğitilme süresini doğrudan etkilemektedir. Doğru kurgulanan bir modelde epoch'lar çalıştıkça kayıp fonksiyon değerinin düşmesi beklenmektedir.

Yukarıda işlem adımları verilen LSTM algoritması kullanılarak, oluşturulmuş modele benzer yapıda RNN ve GRU modelleri oluşturularak sonuçlar karşılaştırılmıştır. Bunu sağlamak için sadece 2. ve 3. adımlarda, LSTM katmanı eklemek yerine RNN ve

GRU katmanları ekleyerek model oluşturmak yeterli olacaktır. Diğer adımlar birçok derin öğrenme modelinde benzer şekilde oluşturulmaktadır.

RNN katmanı ve algoritması kullanılarak bu adımlar gerçekleştirilmek istendiğinde, ilk olarak model.add komutu ile SimpleRNN katmanı eklenmiştir. İlk SimpleRNN katmanında, girdi şekli (input shape) ve dönüş dizileri (return sequences) belirtilmiştir. Girdi şekli, verinin boyutunu ve özellik sayısını belirtmektedir. İkinci SimpleRNN katmanı, yalnızca aktivasyon fonksiyonu belirtmektedir. Bu katman, önceki katmandan gelen çıktıları alır ve kendi içinde gizli hücrelerle işlemler gerçekleştirmektedir.

GRU katmanı ve algoritması kullanılarak model yeniden oluşturulduğunda ise, RNN'de olduğu gibi ilk GRU katmanı, girdi şekli (input shape) ve dönüş dizileri (return sequences) belirtilmiştir. İkinci GRU katmanında, yalnızca aktivasyon fonksiyonu belirtilmiştir.

Model oluşturulurken kullanılan parametreler için farklı değerler denenmiş olup performans, süre ve sonuçların doğruluğu göz önünde bulundurularak optimum parametre değerleri belirlenmiştir. Böylece model son halini almıştır.

2.11. Bulgular

LSTM ile oluşturulan modelin özeti Tablo 7'de yer almaktadır. Modelde toplam 121301 parametre kullanılmıştır.

Tablo 7. LSTM Model Özeti

Katman Tipi	Çıktı Şekli	Parametre Sayısı
lstm (LSTM)	(None, 24, 100)	40800
dropout (Dropout)	(None, 24, 100)	0
lstm_1 (LSTM)	(None, 100)	80400
dropout_1 (Dropout)	(None, 100)	0
dense (Dense)	(None, 1)	101
Toplam Parametre		121301

GRU ile oluşturulan modellerin özeti Tablo 8'de yer almaktadır. Modelde toplam 91601 parametre kullanılmıştır. LSTM ile oluşturulan modele göre daha az parametre sayısına sahiptir.

Tablo 8. GRU Model Özeti

Katman Tipi	Çıktı Şekli	Parametre Sayısı
gru (GRU)	(None, 24, 100)	30900
dropout (Dropout)	(None, 24, 100)	0
gru_1 (GRU)	(None, 100)	60600
dropout_1 (Dropout)	(None, 100)	0
dense (Dense)	(None, 1)	101
Toplam Parametre		91601

RNN ile oluşturulan modellerin özeti Tablo 9'da yer almaktadır. Modelde toplam 30401 parametre kullanılmıştır. 3 model arasında en az parametreye sahip olan model RNN ile oluşturulan modeldir.

Tablo 9. RNN Model Özeti

Katman Tipi	Çıktı Şekli	Parametre Sayısı
simple_rnn (SimpleRNN)	(None, 24, 100)	10200
dropout (Dropout)	(None, 24, 100)	0
simple_rnn_1 (SimpleRNN)	(None, 100)	20100
dropout_1 (Dropout)	(None, 100)	0
dense (Dense)	(None, 1)	101
Toplam Parametre		30401

Model özetindeki toplam parametre sayısı, modelin karmaşıklığını ve öğrenebileceği bilgi miktarını belirlemektedir. Eğer model daha fazla parametreye sahipse, daha fazla öğrenilebilir bilgi taşıyabilmektedir ve daha fazla ilişki yakalayabilmektedir. Bu durum aynı zamanda daha fazla hesaplama gücü gerektirir ve daha fazla veriyle eğitilmesi gerekmektedir. Toplam parametre sayısı, modelin performansı ve doğruluğu üzerinde etkilidir, ancak tek başına bir gösterge değildir. Ayrıca aşırı öğrenme riski de göz önünde bulundurulmalıdır.

Model özeti ile ilgili bilgiler incelendikten sonra modelin hedef değişken ile tahminler arasındaki kayıp ölçümünün değerini veren Ortalama Mutlak Hata (MAE) değerine ait grafikler oluşturulmuştur. Böylece derin öğrenme modelinin eğitimi sırasında kaydedilen eğitim ve test kayıpları görselleştirilmiştir.

Eğitim süreci boyunca kaydedilen eğitim kayıpları loss değişkenine ve test kayıpları val_loss atanmıştır. history nesnesi, eğitim sırasında kayıp metriğinin geçmiş değerlerini içermektedir. Test kaybı, modelin ayrı bir test veri kümesi üzerindeki performansını ölçmektedir. X eksenine epoch değeri, y eksenine kayıp değerleri atanmıştır.

Loss değeri her eğitim epoch'u sonrasında modelin eğitim veri kümesine ne kadar iyi uyum sağladığını ölçmektedir. Daha düşük bir eğitim kaybı, modelin eğitim verilerine daha iyi uyum sağladığını ve daha iyi performans gösterdiğini göstermektedir. Eğitim kaybının zamanla azalması beklenmektedir, çünkü modelin, verileri daha iyi anlaması ve tahminleri daha doğru yapmayı öğrenmesi hedeflenmektedir. val_loss değeri ise modelin test veri kümesi üzerindeki performansını ölçmektedir. Test verileri, modelin daha önce görmediği ve bilmediği veriler olduğu için ve genellikle modelin genelleme yeteneğini değerlendirmek için kullanılmaktadır. Daha düşük bir test kaybı, modelin genel olarak daha iyi performans gösterdiğini ve yeni verilere de iyi uyum sağladığını göstermektedir. Test kaybının zamanla azalması veya aynı kalması beklenmektedir.

Model rastgele başlatıldığı için eğitim ve test kayıpları başlangıçta yüksek seviyede olabilir. Bunun nedeni, modelin henüz verileri anlamaya başlamamasıdır. Ancak eğitim süreci boyunca hem eğitim kayıplarının hem de test kayıplarının azalmaya başlaması beklenmektedir. Bu da tahminlerin daha doğru elde edildiği aynı zamanda da modelin eğitim verilerine uyum sağlamaya başladığı anlamına gelmektedir.

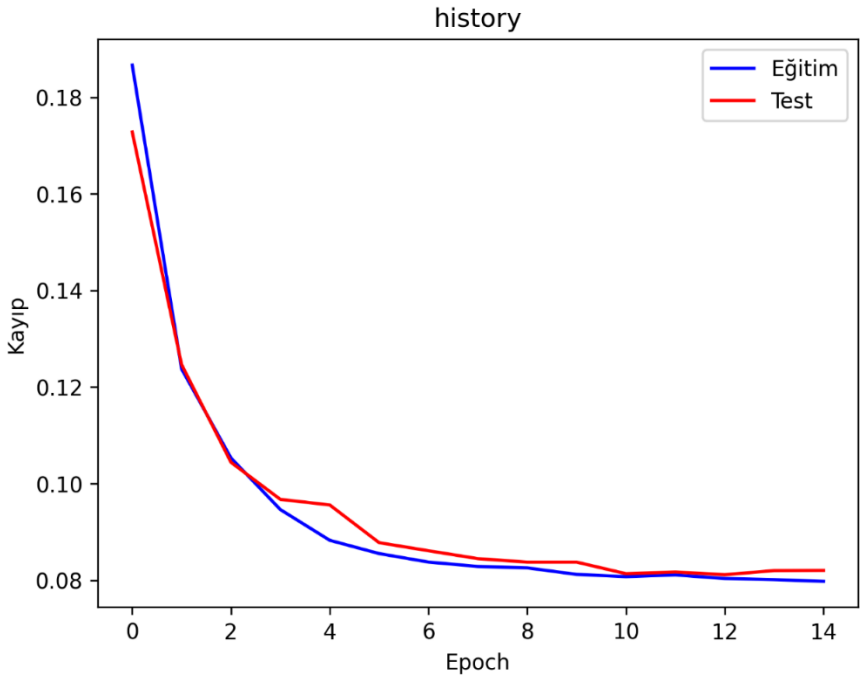
Eğitim kaybı ve test kaybı arasındaki farkın büyümesi, modelin aşırı öğrenmeye (overfitting) başladığını göstermektedir. Kayıp değerlerinin değişimine ait grafik incelenerek, modelin eğitim sürecinin nasıl ilerlediği ve eğitim/test kayıplarının nasıl değiştiği gözlemlenebilmektedir. İyi bir modelde genellikle eğitim ve test kayıpları azalırken, aşırı öğrenmeye yol açabilecek eğilimlerin olmaması önemlidir. Tablo 10’da LSTM, GRU ve RNN modellerinin eğitim ve test sırasında kayıplarına ait değerler karşılaştırılarak gösterilmiştir.

Tablo 10. LSTM, GRU ve RNN Modellerine ait Kayıp Değerleri

			loss			val_loss
epoch	**LSTM**	**GRU**	**RNN**	**LSTM**	**GRU**	**RNN**
1	0,18666	0,10897	0,12963	0,17280	0,10523	0,09079
2	0,12373	0,08836	0,08762	0,12465	0,08829	0,08214
3	0,10531	0,08482	0,08394	0,10441	0,08735	0,08087
4	0,09462	0,08432	0,08435	0,09674	0,08529	0,08464
5	0,08826	0,08684	0,08389	0,09558	0,08761	0,08088
6	0,08551	0,08365	0,08352	0,08777	0,08367	0,08416
7	0,08375	0,08474	0,08204	0,08610	0,08640	0,08584
8	0,08284	0,08251	0,08132	0,08446	0,08123	0,08751
9	0,08255	0,08341	0,08158	0,08374	0,08406	0,08529
10	0,08120	0,08059	0,08088	0,08374	0,08032	0,08800
11	0,08074	0,07927	0,08186	0,08135	0,08208	0,08407
12	0,08109	0,08157	0,08031	0,08170	0,08199	0,08626
13	0,08034	0,08119	0,07980	0,08114	0,08063	0,09114
14	0,08012	0,08071	0,08032	0,08199	0,08154	0,08196
15	0,07976	0,08296	0,08127	0,08202	0,08242	0,08531

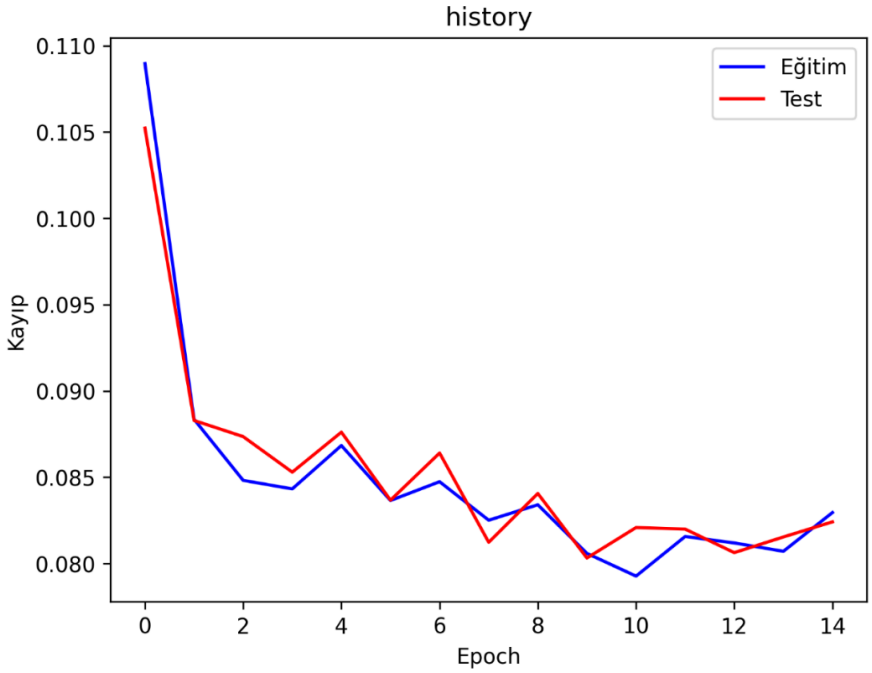
Tablo 10 incelendiğinde LSTM modeline ait loss ve val_loss değeri ilk epoch'larda diğer modellerden daha yüksek çıkmasına rağmen son epoch'lara bakıldığında daha düşük değerlere ulaştığı görülmüştür.

LSTM modelinin kayıp değerlerinden oluşturulan grafiği içeren Şekil 17'de kayıp değerlerinin sürekli olarak bir düşüş eğilimi gösterdiği net bir şekilde görülmektedir.



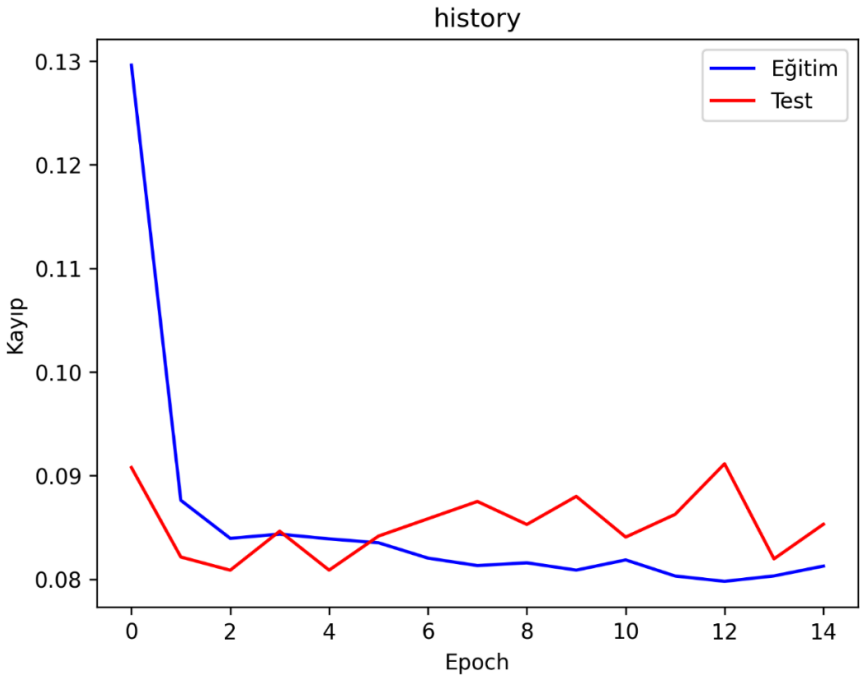
Şekil 17. LSTM Modelinin Eğitimi Sırasında Kayıp Değerlerinin Değişimi

GRU katmanı ile oluşturulan modelde, eğitim ve test veri setlerine ait hedef değerler ile tahmin değerler arasındaki farkların değişimleri birbirine yakın olsa da Şekil 18 incelendiğinde sürekli azalan bir grafik görülememiş bazı epoch sayılarında artışlar görülmüştür.



Şekil 18. GRU Modelinin Eğitimi Sırasında Kayıp Değerlerinin Değişimi

RNN katmanı ile oluşturulan modelde, eğitim ve test veri setlerine ait hedef değerler ile tahmin değerler arasındaki farkların değişimlerinin birbirine yakın olmadığı Şekil 19'daki grafikte görülmektedir. Ayrıca test veri setine ait kayıp değerlerinde bazı epoch sayılarında artışlar görülmektedir.



Şekil 19. RNN Modelinin Eğitimi Sırasında Kayıp Değerlerinin Değişimi

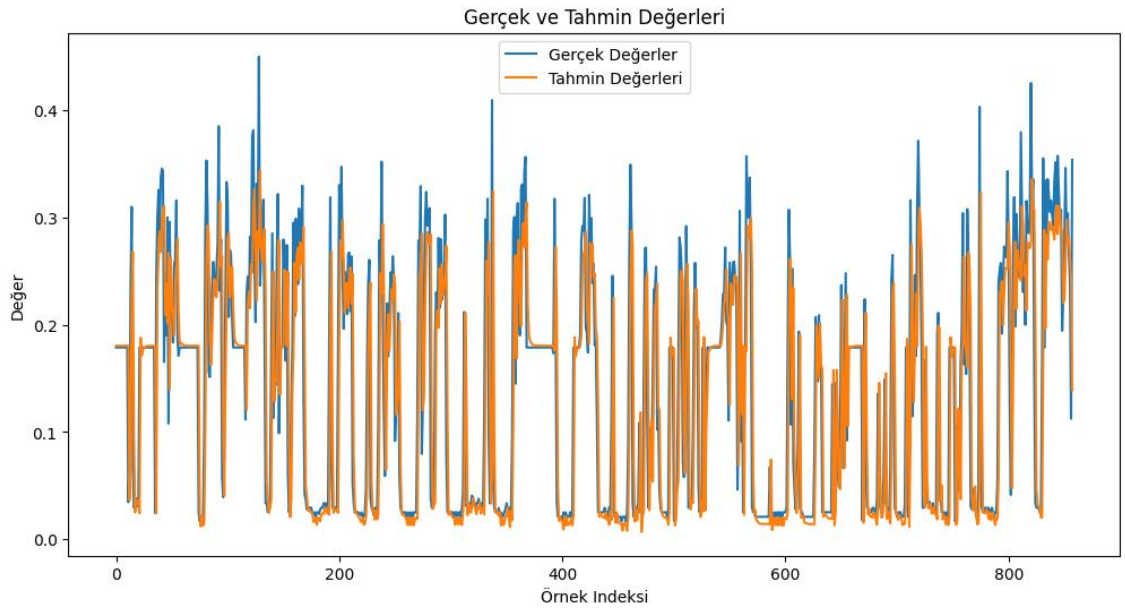
LSTM, GRU ve RNN katmanları ile oluşturulan modellere ait kayıp değerleri karşılaştırıldığında en iyi sonucun LSTM modelinden elde edildiği görülmektedir. Bunun nedeni hem eğitim hem de test veri setindeki hedef değerler ile tahmin değerleri arasındaki MAE kayıp değerlerinin azalmasıdır. Ayrıca eğitim ve test veri setlerindeki kayıp değerleri birbirine çok yakındır.

En iyi modele karar verildikten sonra, tahmin değerleri orijinal ölçeklerine geri dönüştürülerek, tahmin edilen değerler, gerçek değerler ile karşılaştırılabilecektir. Böylece test veri setine ait tahmin değerlerine gerçek ölçeğinde ulaşılabilecektir. Başka bir deyişle, modelin test veri setindeki giriş verilerine dayanarak hedef değişkenin tahmin edilen değerlerini döndürmesi sağlanmıştır. Tahmin edilen değerlerin ölçeklendirmesi tersine çevrilmiştir. Bu işleme “inverse scaling” denir. Böylece tahmin edilen değerler orijinal ölçeklerine geri dönüştürülmüş olacaktır.

Gerçek değerler, y_test_df veri seti kullanılarak ölçeklendirmeden çıkarılarak orijinal ölçeklerine geri dönüştürülmüştür. Aynı işlemler, eğitim veri seti (x_train, y_train_df) için de yapılmıştır. Böylece, eğitim veri setindeki tahmin değerlerini gerçek değerlerle karşılaştırılmıştır.

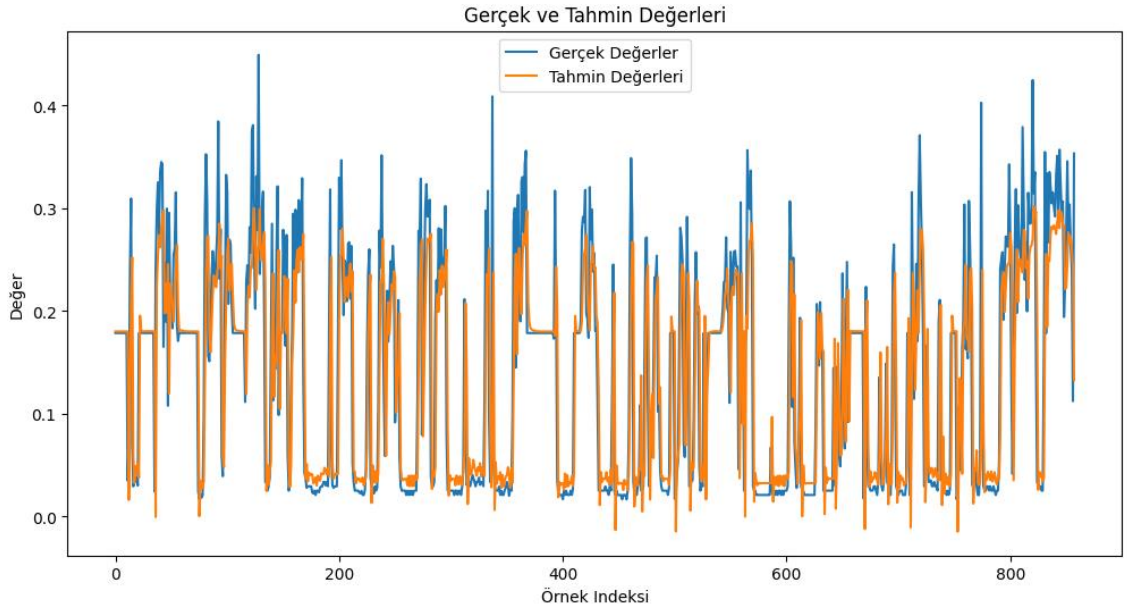
Modelinin test veri seti üzerindeki gerçek ve tahmin edilen değerleri arasındaki ilişkiyi görselleştirmek için grafik oluşturulmuştur. LSTM modeline ait grafiği içeren

Şekil 20, gerçek değerleri (y_true_test) ve tahmin edilen değerleri (inv_y_pred_test) birbirine karşı konumlandırarak göstermektedir.



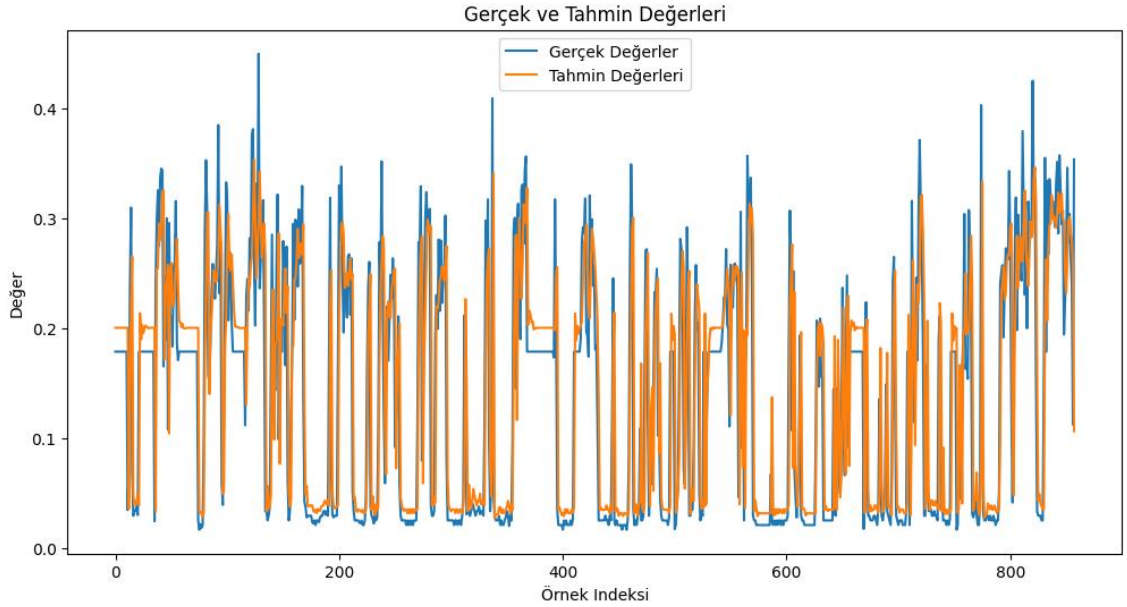
Şekil 20. LSTM Modeli Test Veri Setine ait Gerçek Değerler ve Tahmin Değerleri

GRU ile oluşturulan modele ait grafiği içeren Şekil 21'de özellikle düşük ve yüksek tüketimlerde LSTM modeline göre daha uzak tahminlerin yapıldığı görülmüştür.



Şekil 21. GRU Modeli Test Veri Setine ait Gerçek Değerler ve Tahmin Değerleri

RNN ile oluşturulan modele ait grafiği içeren Şekil 22'de ise hem düşük hem ortalamaya yakın hem de yüksek tüketimlerin gerçek değerleri ile tahmin değerleri arasında farklılıklar olduğu görülmüştür.



Şekil 22. RNN Modeli Test Veri Setine ait Gerçek Değerler ve Tahmin Değerleri

LSTM, GRU ve RNN modellerine ait gerçek değer ve tahmin değerlerinin karşılaştırıldığı grafiklere bakıldığında en iyi tahminlemeyi LSTM modelinin yaptığı anlaşılmaktadır.

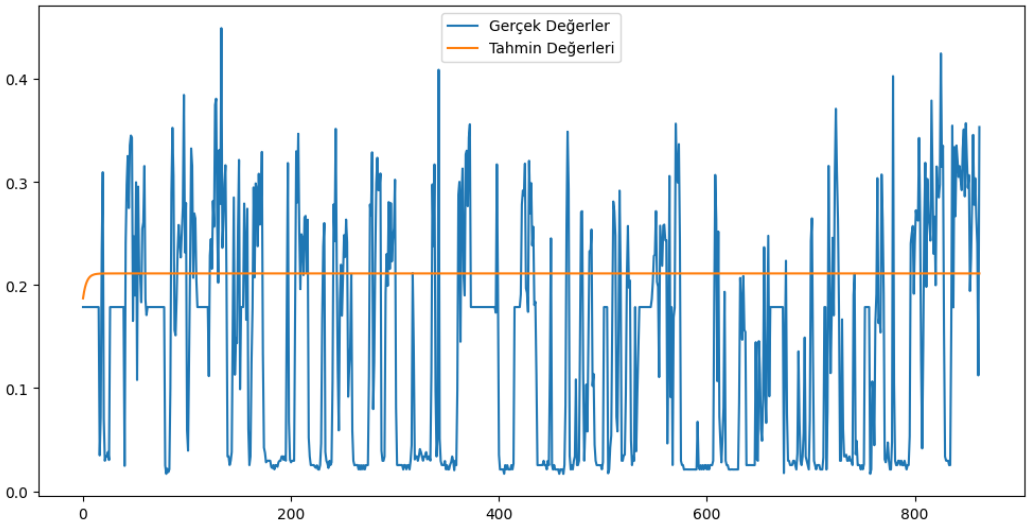
Zaman serisi tahminlemesi için kullanılan çeşitli geleneksel istatistik yöntemleri de bulunmaktadır. ARIMA (Otomatik Regresif Entegre Hareketli Ortalama) da bunlardan birisidir (Kalpakis vd., 2021: 273). ARIMA, otoregresif (AR), entegre (I) ve hareketli ortalama (MA) bileşenlerini içeren istatistiksel bir modeldir. Bu model, veri setindeki trend ve mevsimsellik gibi yapıları dikkate alabilir ve bu yapıları modele dahil edebilir. Ayrıca, daha basit zaman serisi modelleri için uygun olabilir ve özellikle düşük boyutlu veri setlerinde iyi performans gösterebilmektedir. Bunun dışında, gelecekteki tahminleri yaparken genellikle mevcut veri noktalarının kombinasyonunu kullanır ve bu nedenle daha çok ortalama değerlerin tahminlerini üretebilir. ARIMA, durağanlık ve lineer ilişkileri varsayar, bu nedenle veri setindeki karmaşık yapıları yakalayamayabilir. ARIMA'nın veri uyumu, daha küçük verilerle çalışabilme, hesaplama gücü ve süresi olarak hızlı ve basit olması, parametre ve tahminlerin yorumlanma kolaylığı, karmaşık ve gizli bağımlılıkların olmadığı verilerdeki performansı gibi bazı avantajları bulunmaktadır (Wang vd., 2013: 248). Bu nedenle ARIMA daha basit veri setleri ve daha yorumlanabilir tahminler için tercih edilebilirken, derin öğrenme modelleri daha büyük veri setleri ve karmaşık yapılar için daha uygun olabilir. Karşılaştırma yaparken, veri setinin

özelliklerini, hedefleri ve mevcut kaynakları göz önünde bulundurmak önemlidir (Siami-Namini vd., 2018: 1395).

LSTM, GRU ve RNN katmanlarıyla oluşturulan modellere ait sonuçlar, geleneksel istatistik yöntemlerinin kullanıldığı tahminleme yöntemleriyle karşılaştırılmak istenmiş ve bu sebeple ARIMA modeli oluşturulmuştur. ARIMA modelinin oluşturulma aşamaları şu şekildedir.

- İlk olarak, veri seti eğitim ve test verilerine ayrılıyor. Veri setinin yüzde 80'i eğitim için kullanılırken, yüzde 20'si test için kullanılmıştır.
- Daha sonra ARIMA modeli oluşturulmuştur. ARIMA(train_data, order=(1, 1, 1)) ifadesi ile ARIMA modelinin parametreleri belirtilmiştir. Buradaki (1, 1, 1) değerleri sırasıyla p, d ve q değerlerini temsil etmektedir. p, otoregresif (AR) terimlerinin sayısını, d, veri serisinin farklarının alındığı dereceyi ve q, hareketli ortalama (MA) terimlerinin sayısını belirlemektedir. Bu değerler, modelin performansını etkileyen önemli parametrelerdir ve genellikle deneme yanılma yöntemiyle belirlenmektedir (Reisen, 1994: 341). Modelde (6,1,4), (3,1,2), (1,1,1) gibi çeşitli değerler denenerek en iyi değerin (1,1,1) olduğu görülmüştür.
- Bir sonraki adımda model.fit() ifadesi ile model eğitilmiştir. model_fit = Eğitim süreci, modelin içindeki mevcut verilere uyum sağlamayı ve model parametrelerini ayarlamayı içermektedir.
- Son olarak tahmin yapılmıştır. model_fit.predict(start=len(train_data), end=len(train_data) + len(test_data) - 1) ifadesi, eğitilmiş ARIMA modelini kullanarak gelecekteki tahminleri yapmaktadır. Tahminler, eğitim veri setinin sonundan başlayarak test veri seti boyunca yapılmaktadır.

ARIMA modelinde test veri seti ile oluşturulan tahminlere ait grafik çizilmiş ve Şekil 23’te gösterilmiştir.



Şekil 23. ARIMA Modeli Test Veri Setine ait Gerçek Değerler ve Tahmin Değerleri

ARIMA modeli, otoregresif (AR) ve hareketli ortalama (MA) bileşenlerini içerdiği için, tahminleri mevcut veri noktalarının bir kombinasyonu olarak hesaplamaktadır. Bu nedenle, modelin tahminleri genellikle veri setinin ortalama değerine yakın bir seviyede sabitlenebilmektedir. Grafikte tahmin değerlerinin ortalamaya yakın bir değere sabitlenmesi ve yükselme ya da düşme eğilimi göstermemesi, modelin bu özelliği ile ilgilidir. Bu durumun bir diğer sebebi modelin veri setindeki değişkenlikleri yeterince hesaba katamamasıdır. Sonuç olarak, veri setine ait tahminler ARIMA modeli ile yapılmış ve LSTM, GRU ve RNN gibi derin öğrenme modellerinin ARIMA modeline göre daha başarılı olduğu görülmüştür.

LSTM, GRU, RNN ve ARIMA modellerine ait test veri setindeki gerçek ve tahmin edilen değerler arasındaki farkın yüzde olarak ifade edilen bir ölçümü hesaplayarak modellerin performanslarını nicel olarak karşılaştırmak amaçlanmıştır. Hesaplama adımları şunlardır:

- İlk olarak calculate_percentage_difference adında bir fonksiyon tanımlanmış ve y_true (gerçek değerler) ve y_pred (tahmin edilen değerler) parametrelerini alması sağlanmıştır.
- İki veri setine ait her bir satırdaki veriyi çiftler halinde alarak farkları bulunup mutlak değeri alınmış ve differences adında tanımlanmış olan listeye eklenmiştir. Döngü ile tüm değerler için bu işlem yapılmış ve differences listesi oluşturulmuştur.

- Oluşan differences listesine ait fark değerlerinin ve gerçek değerlerin toplamı ayrı olarak alınmıştır.
- Fark değerlerinin gerçek değerlere oranı yüzde olarak hesaplanmış ve bu değer 100'den çıkartılarak uyum oranı hesaplanmıştır.
- Elde edilen sonuç percentage_difference değişkenine atanmış ve fonksiyondan bu değer döndürülmüştür.

Modellere ait hesaplanan değerler Tablo 11'de gösterilmektedir.

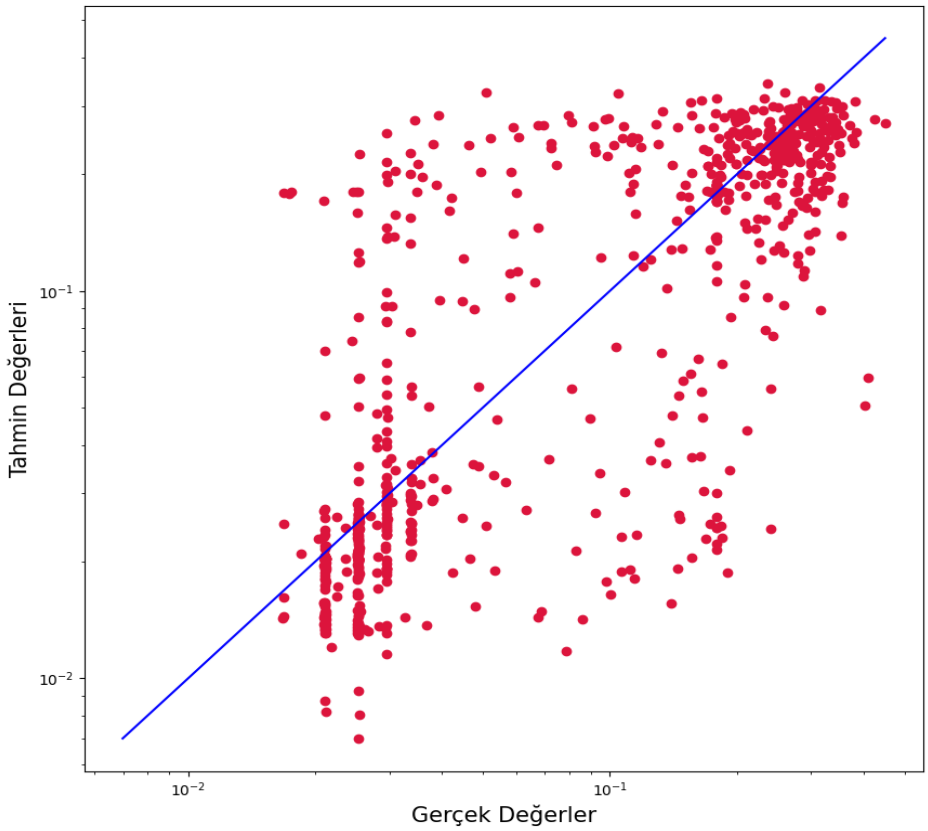
Tablo 11. Gerçek ve Tahmin Edilen Değerler Arasındaki Uyum Oranları

Model	Uyum Oranı (%)
LSTM	67,47
GRU	67,32
RNN	66,17
ARIMA	19,48

Daha yüksek bir yüzde değeri, daha doğru tahminler yapıldığını ve daha iyi bir performans elde edildiğini göstermektedir. Tablodaki sonuçlara bakıldığında en yüksek uyum oranını LSTM modelinin yakaladığı görülmektedir. ARIMA modelinin ise derin öğrenme modellerine kıyasla oldukça düşük bir uyum oranına sahip olduğu görülmektedir.

LSTM, GRU, RNN ve ARIMA modellerine ait gerçek ve tahmin edilen değerlerin uyum oranları hesaplanarak karşılaştırıldığında, en iyi performansı gösteren ve en doğru değerleri tahminleyebilen modelin LSTM modeli olduğu görülmektedir.

LSTM modelinde gerçek değerler ve tahmin değerlerinin arasındaki ilişkinin görselleştirilmesi için Şekil 24'te gösterilen dağılım grafiği çizilmiştir. Bu grafiğe göre, gerçek değerler ve tahmin değerleri arasındaki uyum, noktaların dağılımından anlaşılmaktadır. Değerlere ait noktalar, grafiği ortadan bölen doğrunun etrafında simetrik olarak yer almaktadır.



Şekil 24. Gerçek Değerler ve Tahmin Değerleri

Şekil 24'ün analizi, modelin performansının görsel bir şekilde anlaşılmasını sağlamakta ve gerçek değerler ile tahmin değerleri arasındaki ilişkiyi değerlendirmek için önemli bir araç sunmaktadır. Bu analiz, modelin güvenilirliğini değerlendirme, iyileştirme ve gelecekteki tahminler için bir kılavuz olarak kullanılabilmektedir.

SONUÇ

Endüstriyel süreçlerde tüketilen kaynak miktarı maliyeti doğrudan etkilemektedir. İşletmelerde Derin Öğrenme modellerinin kullanılması ile enerji tüketimlerinin en aza indirilmesi sağlanabildiği gibi, kaynak tüketim maliyetleri de düşürülmektedir. Bunun yanı sıra işletmeler bu sayede olası tüketim değerlerini tahminleyerek gerekli önemleri almaktadır. Ayrıca süreçlerde aktif görev alan makinelerin bakım onarım ve iyileştirme faaliyetleri önceden planlanabilmektedir. Bu çalışma, LSTM ile derin öğrenme modelinin bekleme anındaki elektrik tüketimi tahmininde başarılı olduğunu göstermiştir. Geçmiş elektrik tüketim verileri kullanılarak geleceğe yönelik doğru tahminler yapılmıştır.

Bu tahminler sayesinde, işletme enerji tasarrufu sağlayabilecek ve gereksiz enerji israfının önüne geçilecektir. Bu durum, maliyetlerin azalmasına katkı sağlayacaktır. Elektrik tüketimi tahmini, işletmelere operasyonel verimliliklerini artırma imkânı da sunmaktadır. Tahminler sayesinde, işletmeler enerji taleplerini önceden planlayarak kaynakları daha etkili bir şekilde yönetebilme imkanına sahip olacaktır. Bu da işletmeye operasyonel maliyetlerinin düşürülmesi konusunda katkı sağlayacaktır. Sağlanan maliyet tasarrufu işletmeye rakiplerine karşı rekabet avantajı sağlayacaktır. Elektrik tüketimlerinin azaltılmasının bir diğer faydası da doğal kaynakların tüketimlerinin azaltılması sayesinde çevreye olan duyarlılığın da artması olacaktır. Ayrıca işletme, beklenmedik tüketimleri önceden tahmin ederek olası makine arızalarının önüne geçebilecek ve önceden bakım yapma şansına sahip olacaktır. Bunların dışında, büyük verilerle derin öğrenme modelleri üzerinde analiz ve çalışmalar yapmak işletmeye veri odaklı çalışma alışkanlığı kazandırarak veriye dayalı kararlar vermesini ve stratejilerini buna göre belirlemesini sağlayacaktır.

Bu çalışma, makinelerin enerji tüketimi üzerine odaklanan daha önceki çalışmalardan farklı bir bakış açısı sunmaktadır. Genellikle makinelerin aktif çalışma anında enerji tüketimi incelenirken, beklerken enerji tüketimi durumu göz ardı edilmektedir. Bu çalışma, beklerken tüketilen enerjinin önemini vurgulayarak, büyük maliyetlere neden olabilecek bir potansiyele dikkat çekmektedir. Bekleme anının uzun sürebileceği durumlar göz önüne alındığında, enerji tüketiminin sadece aktif çalışma anında değil, aynı zamanda beklerken de incelenmesi önem kazanmaktadır. Makinenin bekleme modunda ne kadar süre geçirdiği, enerji maliyetlerinin belirlenmesi ve işletme verimliliğinin artırılması açısından kritik bir faktördür. Bu çalışma, bekleme anındaki

enerji tüketimi üzerine yapılan analizler sonucunda belirli önlemlerin alınmasının, büyük ölçekli maliyetlerin önüne geçebileceğini göstermektedir. Özellikle beklemenin gerektiği durumlarda, enerji tüketiminin en iyilenmesi ve verimli bir şekilde yönetilmesi, işletmelerin maliyetlerini düşürebilir ve sürdürülebilirlik hedeflerine katkı sağlayabilir.

Bu çalışmada, bir tekstil işletmesinin terbiye fabrikasındaki merserize makinesine ait bekleme anındaki elektrik tüketim verileri LSTM algoritması kullanılarak derin öğrenme modeli oluşturup geçmiş verilerden geleceğe yönelik tahminler yapılmıştır. Çalışma sonucunda çeşitli performans göstergelerinden faydalanılarak modelin başarıya ulaştığı görülmüştür. Sonuç olarak, bu çalışma bekleme anındaki enerji tüketimine odaklanarak mevcut literatüre yeni bir bakış açısı sunmuştur. Beklemenin göz ardı edilen bir alan olduğunu ve uzun bekleme sürelerinin büyük maliyetlere yol açabileceği vurgulanmıştır. Bu çalışmanın sonuçları, enerji verimliliği alanında yapılan çalışmalara katkıda bulunmanın yanı sıra, işletmelerin enerji maliyetlerini düşürme ve sürdürülebilirlik hedeflerini gerçekleştirme potansiyeline sahiptir.

Gelecekte, bekleme elektrik verileri ile yapılan çalışmaya benzer şekilde işletmelerdeki makinelerde tüketilen su, doğalgaz ve buhar gibi diğer enerji kaynakları ile çalışmalar yapılabilir. Beklemenin enerji tüketimi üzerindeki etkisi genellikle göz ardı edildiği için bu alanda daha fazla araştırmanın yapılması gerekmektedir. Ayrıca, kurumsal kaynak planlama yazılımları aracılığı ile iş emri gibi bilgiler ile tüketim verileri eşleştirilerek üretilen ürüne ait özellikler de belirlenerek çalışma anındaki enerji tüketimleri de tahminlenebilir. Tekstil işletmelerinde, bezin niteliğine, ağırlığına ve sürtünme katsayısına göre enerji tüketimleri analiz edilip enerji tasarrufu açısından makineye en uygun bez tahminlemeleri yapılabilir.

KAYNAKLAR

Acar, O. (2020). *Yapay zeka fırsat mı yoksa tehdit mi?,* Kriter Yayınevi, İstanbul.

Agga, A., Abbou, A., Labbadi, M., El Houm, Y., Ali, I. H. O. (2022). "CNN-LSTM: An efficient hybrid deep learning architecture for predicting short-term photovoltaic power production", *Electric Power Systems Research*, 208, 107908.

Ahmad, R., Wazirali, R., & Abu-Ain, T. (2022). "Machine learning for wireless sensor networks security: An overview of challenges and issues", *Sensors*, 22(13), 4730.

Aparna, S. (2018, June). "Long short term memory and rolling window technique for modeling power demand prediction." *In 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, 1675-1678.

Aydın, A. (2019). *Devlet erkinin yönetim paradigmasının yapay zeka bağlamında dönüşümü*. G. Telli (Ed.), Yapay Zeka ve Gelecek içinde, Doğu Kitapevi, İstanbul.

Bengio, Y., Courville, A., & Vincent, P. (2013). "Representation Learning: A Review and New Perspectives." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798-1828.

Blickle, T., Lakatos, B. G., Mihálykó, C., & Ulbert, Z. (1998). "The hyperbolic tangent distribution family", *Powder technology*, 97(2), 100-108.

Changchun L., Haihua Z., Dunbing T., Qingwei N., Shipei L., Yi Z. & Xuan L. (2022). "A transfer learning CNN-LSTM network-based production progress prediction approach in IIoT-enabled manufacturing", *International Journal of Production Research*, 61(12), 4045-4068.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation", *arXiv preprint arXiv:1406.1078*.

Chollet, F. (2018). *Deep Learning with Python*, Manning Publications, New York.

Cifuentes, J., Marulanda, G., Bello, A., & Reneses, J. (2020). "Air temperature forecasting using machine learning techniques: a review", *Energies*, 13(16), 4215.

Dong, S., Wang, P., & Abbas, K. (2021). "A survey on deep learning and its applications", *Computer Science Review*, 40, 100379.

Floridi, L. (2020). "AI and its new winter: From myths to realities", *Philosophy & Technology*, 33, 1-3.

Forootan, M. M., Larki, I., Zahedi, R., & Ahmadi, A. (2022). "Machine learning and deep learning in energy systems: A review", *Sustainability*, 14(8), 4832.

Gang, Q., Muhammad, A., Khan, Z. U., Khan, M. S., Ahmed, F., & Ahmad, J. (2022). "Machine Learning-Based Prediction of Node Localization Accuracy in IIoT-Based MI-UWSNs and Design of a TD Coil for Omnidirectional Communication", *Sustainability*, 14(15), 9683.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, Inc, Sebastopol, California.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press, Cambridge.

González García, C., Núñez Valdéz, E. R., García Díaz, V., Pelayo García-Bustelo, B. C., & Cueva Lovelle, J. M. (2019). "A review of artificial intelligence in the internet of things", *International Journal Of Interactive Multimedia And Artificial Intelligence,* 5.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2016). "LSTM: A search space odyssey", *IEEE transactions on neural networks and learning systems,* 28(10), 2222-2232.

Grimaldo, A. I., & Novak, J. (2020). "Combining machine learning with visual analytics for explainable forecasting of energy demand in prosumer scenarios", *Procedia Computer Science,* 175, 525-532.

Gupta, H. (2020). "Foundation of IoT: an overview", *Internet of Things (IoT) Concepts and Applications,* 3-24.

Han, J., & Moraga, C. (1995, June). "The influence of the sigmoid function parameters on the speed of backpropagation learning", *In From Natural to Artificial Neural Computation: International Workshop on Artificial Neural Networks Malaga-Torremolinos, Spain,* 195-201.

Hansen, E. B., & Bøgh, S. (2021). "Artificial intelligence and internet of things in small and medium-sized enterprises: A survey", *Journal of Manufacturing Systems*, 58, 362-372.

Hochreiter, S., & Schmidhuber, J. (1997). "Long short-term memory", *Neural computation,* 9(8), 1735-1780.

Ikeda, S., & Nagai, T. (2021). "A novel optimization method combining metaheuristics and machine learning for daily optimal operations in building energy and storage systems", *Applied Energy,* 289, 116716.

Jarek, K. & Mazurek, G. (2019). "Marketing and artificial ıntelligence", *Central European Business Review,* 8(2), 46-55.

Kalpakis, K., Gada, D., & Puttagunta, V. (2001, November). "Distance measures for effective clustering of ARIMA time-series", *In Proceedings 2001 IEEE international conference on data mining, IEEE,* 273-280).

Khan, Z., Khan, S. M., Dey, K., & Chowdhury, M. (2019). "Development and evaluation of recurrent neural network-based models for hourly traffic volume and annual average daily traffic prediction", *Transportation Research Record*, 2673(7), 489-503.

Kingma, D. P., & Ba, J. (2014). "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980.

Kitaoka, H., Kashiwagi, Y., & Terano, T. (2021). "Forecasting Bitcoin prices using deep learning", *Expert Systems with Applications,* 181, 115054.

Koehrsen, W. (2018). "Overfitting vs. underfitting: A complete example", *Towards Data Science,* 1-12.

Kumar, K. S., & Bai, M. R. (2023). "LSTM based texture classification and defect detection in a fabric", *Measurement: Sensors*, 26, 100603.

Kutlusoy, Z. (2019). *Felsefe açısından yapay zeka*. G. Telli (Ed.), Yapay zeka ve gelecek içinde, Doğu Kitapevi, İstanbul.

LeCun, Y., Bengio, Y., Hinton, G. (2015). "Deep learning". *Nature*, 521(7553), 436-444.

Lewis, T. (2014). *A Brief History of Artificial Intelligence*, LiveScience Retrieved.

Malakouti, S. M., Ghiasi, A. R., Ghavifekr, A. A., & Emami, P. (2022). "Predicting wind power generation using machine learning and CNN-LSTM approaches", *Wind Engineering*, 46(6), 1853-1869.

McCarthy, J. (2007). "From here to human-level AI", *Artificial Intelligence*, 171(18), 1174-1182.

Milić, S. D., Đurović, Ž., & Stojanović, M. D. (2023). "Data science and machine learning in the IIoT concepts of power plants", *International Journal of Electrical Power & Energy Systems*, 145, 108711.

Nair, V., & Hinton, G. E. (2010). "Rectified linear units improve restricted boltzmann machines", *In Proceedings of the 27th international conference on machine learning (ICML-10)*.

Nguyen, T. L. (2018). "Scaling, Standardizing, or Normalizing with Scikit-Learn. Towards Data Science". https://towardsdatascience.com/scale-standardize-or-normalize-with-scikit-learn-6ccc7d176a02 (15.05.2023)

Nielsen, M. A. (2015). *Neural networks and deep learning*, Determination press, San Francisco, CA, USA.

Oprea, S. V., Bâra, A., Puican, F. C., & Radu, I. C. (2021). "Anomaly detection with machine learning algorithms and big data in electricity consumption", *Sustainability*, 13(19), 10963.

Patchaiammal, P., & Sundar, G. (2022, February). "Feature Extraction by Rework Image Recognition (RIR) Learning Model", *In International Conference on Computing, Communication, Electrical and Biomedical Systems, Cham: Springer International Publishing*, 199-213.

Rajagukguk, R. A., Ramadhan, R. A. A., & Lee, H. J. (2020). "A review on deep learning models for forecasting time series data of solar irradiance and photovoltaic power", *Energies*, 13 (24), 6623.

Rani, S., Maheswar, R., Kanagachidambaresan, G. R., Ahuja, S., & Gupta, D. (Eds.). (2022). *Machine Learning Paradigm for Internet of Things Applications*, John Wiley & Sons, New Jersey.

Reisen, V. A. (1994). "Estimation of the fractional difference parameter in the ARIMA (p, d, q) model using the smoothed periodogram", *Journal of Time Series Analysis*, 15(3), 335-350.

Saba Raoof, S., & Durai, M. A. (2022). "A Comprehensive Review on Smart Health Care: Applications, Paradigms, and Challenges with Case Studies", *Contrast Media & Molecular Imaging*, 2022.

Samadi, R., & Seitz, J. (2022, November). "Machine Learning Routing Protocol in Mobile IoT based on Software-Defined Networking", *In 2022 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN, IEEE*, 108-111.

Shirvanian, N., Shams, M., & Rahmani, A. M. (2022). "Internet of Things data management: A systematic literature review, vision, and future trends", *International Journal of Communication Systems*, 35(14), e5267.

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

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). "Dropout: A simple way to prevent neural networks from overfitting", *Journal of Machine Learning Research*, 15(1), 1929-1958.

Sundar, G., and Patchaiammal, P. (2022, February). "Comprehensive Deep Recurrent Artificial Neural Network (CDRANN): Evolutionary Model for Future Prediction", *In International Conference on Computing, Communication, Electrical and Biomedical Systems, Cham: Springer International Publishing*, 217-234.

Thormundsson B. (2023), Statista, https://www.statista.com/statistics/1365145/artificial-intelligence-market-size/ (18.05.2023).

Turing, A. M. (1950). "Computing machinery and intelligence", *Mind a Quarterly Review of Psychology and Philosophy*, 433-460.

Vailshery L. S. (2022), Statista, https://www.statista.com/statistics/1194709/iot-revenue-worldwide/ (15.03.2023).

Vailshery L. S. (2023), Statista, https://www.statista.com/statistics/1194682/iot-connected-devices-vertically/ (15.03.2023).

Walker, S., Khan, W., Katic, K., Maassen, W., & Zeiler, W. (2020). "Accuracy of different machine learning algorithms and added-value of predicting aggregated-level energy performance of commercial buildings", *Energy and Buildings*, 209, 109705.

Wang, L., Zou, H., Su, J., Li, L., & Chaudhry, S. (2013). "An ARIMA-ANN hybrid model for time series forecasting", *Systems Research and Behavioral Science*, 30(3), 244-259.

Yang, W., Sun, S., Hao, Y., & Wang, S. (2022). "A novel machine learning-based electricity price forecasting model based on optimal model selection strategy", *Energy*, 238, 121989.

Yasir, M., Ansari, Y., Latif, K., Maqsood, H., Habib, A., Moon, J., & Rho, S. (2022). "Machine learning–assisted efficient demand forecasting using endogenous and exogenous indicators for the textile industry", *International Journal of Logistics Research and Applications*, 1-20.

Yeğin, T. (2020). "The place and future of artificial intelligence in marketing strategies", *Ekev Akademi Dergisi*, 24(81), 489-506.

Yucesan, M., Pekel, E., Celik, E., Gul, M., & Serin, F. (2021). "Forecasting daily natural gas consumption with regression, time series and machine learning based methods" *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, 1-16.

Zaheer, S., Anjum, N., Hussain, S., Algarni, A. D., Iqbal, J., Bourouis, S., & Ullah, S. S. (2023). "A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model", *Mathematics*, 11(3), 590.

Zhang, H., Zhang, L., & Jiang, Y. (2019, October). "Overfitting and underfitting analysis for deep learning based end-to-end communication systems." *In 2019 11th*

international conference on wireless communications and signal processing (WCSP) IEEE, 1-6.