T.C.
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME İLE FARKLI MODEL VE ÖĞRENME ALGORİTMALARI KULLANILARAK KISA VADELİ ELEKTRİK YÜK TÜKETİMİNİN TAHMİNİ

DOKTORA TEZİ

Mehmet Tahir UÇAR

Elektrik Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Prof. Dr. Asım KAYGUSUZ

TEMMUZ 2025

T.C
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME İLE FARKLI MODEL VE ÖĞRENME ALGORİTMALARI KULLANILARAK KISA VADELİ ELEKTRİK YÜK TÜKETİMİNİN TAHMİNİ

DOKTORA TEZİ

Mehmet Tahir UÇAR
(10739259)

Elektrik Elektronik Mühendisliği Anabilim Dalı

Tez Danışmanı: Prof. Dr. Asım KAYGUSUZ

TEMMUZ 2025

TEŞEKKÜR VE ÖNSÖZ

Bu tez çalışmasının her aşamasında yardım, öneri, bilgi, tecrübe ve desteklerini esirgemeden beni her konuda yönlendiren danışman hocam Sayın Prof. Dr. Asım KAYGUSUZ'a,

Çalışmalarımda ayrıca tüm hayatım boyunca olduğu gibi bu çalışmalarım süresince de benden her türlü desteklerini esirgemeyen eşime, çocuklarıma, abilerime ve aileme,

Ayrıca manevi ve maddi olarak katkılarını gördüğüm Prof. Dr. Mehmet AYBAK ve Prof. Dr. Mehmet AKIN'a, ve desteğini gördüğüm tüm hocalarıma,

teşekkür ederim.

ONUR SÖZÜ

Doktora tezi olarak sunduğum “Derin Öğrenme ile Farklı Model ve Öğrenme Algoritmaları Kullanılarak Kısa Vadeli Elektrik Yük Tüketiminin Tahmini” başlıklı bu çalışmanın bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurmaksızın tarafımdan yazıldığına ve yararlandığım bütün kaynakların hem metin içinde hem de kaynakçada yöntemine uygun biçimde gösterilenlerden oluştuğunu belirtir, bunu onurumla doğrularım.

M. Tahir Uçar

İÇİNDEKİLER

TEŞEKKÜR VE ÖNSÖZ ..... **i**
ONUR SÖZÜ ..... **ii**
İÇİNDEKİLER ..... **iii**
ÇİZELGELER DİZİNİ ..... **vi**
ŞEKİLLER DİZİNİ ..... **vii**
SEMBOLLER VE KISALTMALAR ..... **ix**
SEMBOLLER VE KISALTMALAR DEVAMI ..... **x**
ÖZET ..... **xi**
ABSTRACT ..... **xii**
1. GİRİŞ **1**
1.1 Tezin Amacı 4
2. KONU KAPSAM ve LİTERATÜR ÖZETİ **5**
3. GENEL KAVRAMLAR **15**
3.1 Yapay Zeka 16
3.2 Yapay Sinir Ağları 17
3.2.1 Yapay Sinir Ağı Yapısı 19
3.2.2 Algılayıcı 19
3.2.3 Yapay Sinir Ağlarının Mimari Yapısı 20
3.2.3.1 İleri beslemeli YSA modeli 21
3.2.3.2 Geri beslemeli YSA modeli 21
3.2.4 Yapay Sinir Ağları İçin Aktivasyon Fonksiyonları 22
3.2.4.1 Adım basamak (Step) fonksiyonu 22
3.2.4.2 Doğrusal (Linear) fonksiyon 22
3.2.4.3 Sigmoid fonksiyonu 23
3.2.4.4 Hiperbolik tanjant fonksiyonu 24
3.2.4.5 Relu (Rectified linear unit) fonksiyonu 25
3.2.4.6 Sızıntı (Leaky) relu fonksiyonu 25
3.2.4.7 Softmax fonksiyonu 26
3.2.4.8 Üssel lineer birim (Exponential Linear Unit, ELU) aktivasyon fonksiyonu 26
3.2.4.9 Swish (A self-gated/kendinden geçitli) fonksiyonu 27
3.2.5 Geri Yayılım Algoritması 28
3.3 Makine Öğrenmesi 28
3.3.1 Gözetimli Öğrenme 29
3.3.2 Gözetimsiz Öğrenme 30
3.3.3 Pekiştirmeli Öğrenme 30
3.3.4 Derin Öğrenme 31
3.4 Makine Öğrenmesi İle Küçük Bir Veri Seti Üzerinden Yapılan Regresyon Çalışması 31
3.4.1 Veri Setininin Hazırlanması 31
3.4.2 Program Arayüzünün Oluşturulması 32
3.4.3 Doğrusal Regresyon Yöntemiyle Diyarbakır Nüfus Tahmin Çalışması 32
3.4.4 Doğrusal Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 34
3.4.5 Diğer Regresyon Yöntemleriyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 36
3.4.6 Poligonal Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 37

3.4.7 Destek Vektör Regresyonu (SVR) Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 37
3.4.8 Karar Ağacı Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 38
3.4.9 Rasgele Orman Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması 39
3.4.10 Diyarbakır Nüfus Tahmini Sonuçları 39
3.4.11 Diyarbakır Elektrik Tüketim Tahmini Sonuçları 40
3.4.12 R^2 Sonuçları 44
4. DERİN ÖĞRENME 45
4.1 Derin Öğrenme Mimarileri 48
4.1.1 RNN (Özyinelemeli sinir ağları) 48
4.1.2 LSTM (Uzun kısa vadeli hafıza ağları) 49
4.1.3 GRU (Geçitli tekrarlayan birim) 51
4.1.4 Bi-LSTM (Çift yönlü uzun kısa vadeli hafıza ağları) 53
4.2 Derin Öğrenmede Kullanılan Donanımlar 54
4.3 Derin Öğrenmede Kullanılan CPU, GPU ve TPU 54
4.4 Derin Öğrenmede Kullanılan Aktivasyon Fonksiyonları 56
4.5 Gradyan İniş Optimizasyonu Algoritmaları 57
4.5.1 SGD 57
4.5.2 Adagrad 57
4.5.3 Adadelta 57
4.5.4 RMSProp 58
4.5.5 Adam 58
4.5.6 Nadam 58
4.5.7 Adamax 58
4.5.8 Ftrl 59
4.6 Derin Öğrenmede Kullanılan Araçlar ve Programlama Dilleri 59
4.7 Derin Öğrenmede Kullanılan Kütüphaneler ve Veri Kümeleri 61
4.7.1 Tensorflow 61
4.7.2 Caffe 62
4.7.3 Torch 63
4.7.4 Keras 63
4.7.5 DeepLearning4j 63
4.7.6 Numpy 63
4.7.7 Scikit-learn 64
4.7.8 Pandas 64
4.7.9 Statsmodels 64
4.7.10 Matplotlib 65
4.7.11 Seaborn 65
4.8 Hata Fonksiyonları 66
4.8.1 Kategorik Çapraz Entropi Kaybı 67
4.8.2 İkili Çapraz Entropi Kaybı 67
4.8.3 Seyrek Kategorik Çapraz Entropi Kaybı 67
4.8.4 Ortalama Karesel Hata (MSE) Kaybı 67
4.8.5 Kök Ortalama Karesel Hata (RMSE) Kaybı 67
4.8.6 Ortalama Mutlak Hata (MAE) Kaybı 68
5. MATERYAL VE METOD 69
5.1 Veri Analizi 71
5.1.1 Veri 71

5.1.2 Keşifsel veri analizi 71
5.1.2.1 Tanımlayıcı istatistikler, eksik verilerin tamamlanması ve fazla verilerin çıkarılması 72
5.1.2.2 Zaman grafiği 73
5.1.2.3 Mevsimsel grafikler 75
5.2 Derin Öğrenme Çalışması 77
5.3 Program Arayüzünün Oluşturulması 81
6. ÇALIŞMA 82
6.1 4 Farklı Model Üzerinden 11 Farklı Optimize Edici Yöntem Kullanılarak Tüketim Tahmininin Analiz Edilmesi 82
6.2 4 Farklı Model Üzerinden 11 Farklı Optimize Edici Yöntem Kullanılarak 10 epok - 50 epok ve 100 epok Kullanımıyla Tüketim Tahmininin Analiz Edilmesi 83
6.3 4 Farklı Model Üzerinden 6 Farklı Aktivasyon Fonksiyonu Kullanılarak 11 Farklı Optimize Edici Yöntem Kullanımıyla Tüketim Tahmininin Analiz Edilmesi 85
7. ÇALIŞMANIN SONUÇLARI 87
7.1 4 Farkli Model 11 Farklı Optimize Edici Kullanılarak Yapılan Çalışmanın Sonuçları 87
7.2 4 Farklı Model 11 Farklı Optimize Edici ve 10 epok - 50 epok ve 100 epok Kullanılarak Yapılan Çalışmanın Sonuçları 89
7.3 4 Farklı Model 6 Farklı Aktivasyon Fonksiyonu 11 Farklı Optimize Edici Kullanılarak Yapılan Çalışmanın Sonuçları 91
7.4 Elde Edilen En İyi Sonuçlar 95
8. SONUÇ 99
KAYNAKLAR 102
ÖZGEÇMİŞ 114

ÇİZELGELER DİZİNİ

Çizelge 2.1 Derin öğrenme çalışmalarında kullanılan modellerin avantajları, sınırlamaları ve genellikle kullanıldığı alanlar **10**
Çizelge 3.1 Diyarbakır nüfus ve elektrik tüketim değerleri. **32**
Çizelge 3.2 Gelecek yıllar Diyarbakır tahmini nüfus değerleri..... **34**
Çizelge 3.3 Gelecek yıllar Diyarbakır tahmini elektrik tüketim değerleri **36**
Çizelge 3.4 Diyarbakır'ın mevcut verileri için farklı regresyon yöntemleri ile elektrik tüketim tahmin değerleri. **42**
Çizelge 3.5 Gelecek yıllar Diyarbakır tahmini elektrik tüketim değerleri **43**
Çizelge 3.6 Elektrik tüketim tahmini için R^2 değerleri **44**
Çizelge 4.1 En Çok Kullanım Oranına Göre Derin Öğrenme Programlama Dilleri **600**
Çizelge 4.2 Kullanım Yoğunluğuna Göre Derin Öğrenme Kütüphaneleri **666**
Çizelge 5.1 (a) Düzenlenmemiş ve (b) Düzenlenmiş veriye ait ilk ve son 5 satır. .. **733**
Çizelge 5.2 Düzenlenmiş ham veriye ait ilk 5 satır. **777**
Çizelge 5.3 Giriş değerlerine ait son 24 saatlik veriler. **79**
Çizelge 6.1 Kullanılan yöntemlerin detayları..... **82**
Çizelge 6.2 4 model -11 optimizer ile çalışma sonuçları **83**
Çizelge 6.3 4 model-11 optimizer ile 10-50-100 epok deneme sonuçları..... **84**
Çizelge 6.4 4 model-6 aktivasyon fonksiyonu-11 optimizer ile çalışma sonuçları ... **85**
Çizelge 6.4 (devam) 4 model-6 aktivasyon fonksiyonu-11 optimizer ile çalışma sonuçları **86**
Çizelge 7.1 Yapılan çalışmanın meta-analiz yöntrmi ile aynı veri setini kullanan üç benzer çalışmaya göre kıyaslanması **95**
Çizelge 7.2 50 Epok İle Yapılan Çalışmalara Ait En İyi 10 Sonuç **97**

ŞEKİLLER DİZİNİ

Şekil 2.1 : Google scholar üzerinde makale başlıklarında yapılan "deep learning" arama sonucu. **8**
Şekil 3.1: Bir sinir hücresinin biyolojik gösterimi. **19**
Şekil 3.2: Bir sinir hücresinin matematiksel modeli **20**
Şekil 3.3: Tek gizli katmanlı ve birden çok gizli katmanlı sinir ağı yapısı. **20**
Şekil 3.4: İleri beslemeli sinir ağı yapısı. **21**
Şekil 3.5: Geri beslemeli yapay sinir ağı yapısı **21**
Şekil 3.6: Basamak fonksiyonu ve türevi **22**
Şekil 3.7: Doğrusal fonksiyon ve türevi **23**
Şekil 3.8: Sigmoid fonksiyonu ve türevi **24**
Şekil 3.9: Hiperbolik tanjant fonksiyonu ve türevi **24**
Şekil 3.10: Relu fonksiyonu ve türevi **25**
Şekil 3.11: Leaky relu fonksiyonu ve türevi **26**
Şekil 3.12: Softmax fonksiyonu **26**
Şekil 3.13: Elu fonksiyonu ve türevi **27**
Şekil 3.14: Swish fonksiyonu ve türevi **28**
Şekil 3.15: Makina öğreniminin alt konuları. **29**
Şekil 3.16: Gözetimli öğrenme (sınıflandırma ve regresyon) **30**
Şekil 3.17: Gözetimsiz öğrenme (kümelenmemiş ve kümelenmiş veri) **30**
Şekil 3.18: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon çizgisi (mavi çizgi) **33**
Şekil 3.19: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon test çizgisi (mavi çizgi) **33**
Şekil 3.20: Gerçek değerler (kırmızı noktalar), doğrusal regresyon test çizgisi (mavi çizgi) ve tahmin değerleri (yeşil noktalar) **34**
Şekil 3.21: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon çizgisi (mavi çizgi) **35**
Şekil 3.22: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon test çizgisi (mavi çizgi) **35**
Şekil 3.23: Gerçek değerler (kırmızı noktalar), doğrusal regresyon test çizgisi (mavi çizgi) ve tahmin değerleri (yeşil noktalar) **36**
Şekil 3.24: Poligonal regresyon grafikleri a) 2. dereceden, b) 3. dereceden, c) 4. dereceden **37**
Şekil 3.25: Çekirdek fonksiyonu için farklı tercihler sonucu oluşturulan destek vektör grafikleri. a) linear, b) poly, c) rbf. d) sigmoid **38**
Şekil 3.26: Karar ağacı regresyon grafiği. **39**
Şekil 3.27: Rasgele orman regresyon grafiği. **39**
Şekil 3.28: Doğrusal regresyon ile Diyarbakır nüfus tahmin grafikleri a) mevcut durum, b) tahmin **40**
Şekil 3.29: Doğrusal regresyon ile Diyarbakır elektrik tüketim tahmin grafikleri a) mevcut durum, b) tahmin **41**
Şekil 3.30: Farklı regresyon yöntemleri ile Diyarbakır'ın mevcut verileri için elektrik tüketim tahmin grafiği **42**
Şekil 3.31: Farklı regresyon yöntemleri ile Diyarbakır'ın gelecek yıllar için elektrik tüketim tahmin grafiği **43**
Şekil 4.1: Yapay zekâ – makine öğrenmesi – derin öğrenme ilişkisi. **45**
Şekil 4.2: Makine öğrenmesi – derin öğrenme **46**
Şekil 4.3: Kaydedilmemiş bir tekrarlayan sinir ağı. **49**

Şekil 4.4: Standart bir RNN'de tek bir katman içeren yinelenen modül **49**
Şekil 4.5: LSTM'deki birbiriyle etkileşen dört katman içeren yinelenen modül **50**
Şekil 4.6: LSTM mimarisinin genel yapısı. **51**
Şekil 4.7: GRU modeli **52**
Şekil 4.8: Bi-LSTM modeli **53**
Şekil 4.9: CPU-GPU-TPU saniyedeki kestirim sayısı karşılaştırması **55**
Şekil 5.1 : Yapılan çalışmanın akış diyagramı.....**69**
Şekil 5.2 : Düzenlenmemiş ham veri (a) ve düzenlenmiş veri (b).....**74**
Şekil 5.3 : Yıllara göre aylık tüketim grafiği. **75**
Şekil 5.4 : Aylara göre günlük tüketim grafiği. **76**
Şekil 5.5 : Günlere göre saatlik tüketim grafiği. **76**
Şekil 7.1 : 4 model - 7 optimizer - R^2 grafiği.....**88**
Şekil 7.2 : 4 model - 5 optimizer - R^2 grafiği.....**88**
Şekil 7.3 : LSTM-8 optimizer 1-50-100 epok - R^2 grafiği.....**89**
Şekil 7.4 : GRU - 8 optimizer 1-50-100 epok - R^2 grafiği.....**90**
Şekil 7.5 : SimpleRNN - 8 optimizer 1-50-100 epok - R^2 grafiği.....**90**
Şekil 7.6 : BiLSTM - 8 optimizer 1-50-100 epok - R^2 grafiği.....**91**
Şekil 7.7 : a) LSTM - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği. b) LSTM - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği.....**92**
Şekil 7.8 : a) GRU - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği. b) GRU - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği.....**93**
Şekil 7.9 : a) SimpleRNN - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği b) SimpleRNN - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği **94**
Şekil 7.10 : a) BiLSTM - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği b) BiLSTM - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği.....**94**
Şekil 7.11 : En başarılı modele göre ilk 200 ve 30000 değer için gerçek değer-tahmin grafikleri **96**
Şekil 7.12 : En başarılı modele göre tüm değerler için gerçek değer çizgisi ve tahmin popülasyonu.....**97**
Şekil 7.13 : En başarılı 10 sonuca göre aktivasyon fonksiyonu - R^2 grafiği.....**98**
Şekil 7.14 : En başarılı 10 sonuca göre model ve optimizer - R^2 grafiği.....**98**

SEMBOLLER VE KISALTMALAR

RNN : Özyinelemeli sinir ağı

SimpleRNN : Basit Tekrarlayan Sinir Ağı

LSTM : Uzun Kısa Dönem Bellek

GRU : Geçitli Tekrarlayan Birim

BiLSTM : Çift Yönlü Uzun Kısa Dönem Bellek

EDA : Keşifsel Veri Analizi

RMSE : Kök ortalama karesel hata

MAE : Ortalama mutlak hata

STLF : Kısa vadeli yük tahmini

MTLF : Orta vadeli yük tahmini

LTLF : Uzun vadeli yük tahmini

CNN : Evrişimsel Sinir Ağları

RUL : Kalan ömür

PdM : Kestirimci bakım

ANN : Yapay sinir ağları

YSA : Yapay sinir ağları

DT : Karar ağaçları

DVM : Destek vektör makineleri

SVM : Destek vektör makineleri

SVR : Destek vektör regresyonu

GA : Genetik algoritma

k-NN : k-en yakın komşular

PSO : Parçacık sürü optimizasyonu

KKO : Karınca kolonisi optimizasyonu

GSYH : Gayri safi yurtiçi hâsıla

KBGSYH : Kişi başı gayri safi yurtiçi hâsıla

Deep Learning: Derin Öğrenme

Autoencoder : Otomatik kodlayıcı

RBM : Kısıtlı Boltzmann Makinesi

DBN : Derin İnanç Ağları

GAN : Çekişmeli Üretici Ağlar

DRL : Derin Pekiştirmeli Öğrenme

Transfer Learning: Transfer Öğrenimi

SEMBOLLER VE KISALTMALAR DEVAMI

ReLU : Doğrultulmuş Lineer Ünite

Leaky ReLU : Sızıntı Doğrultulmuş Lineer Ünite

tanh : Hiperbolik Tanjant

Swish : Kendinden Geçitli

TUİK : Türkiye İstatistik Kurumu

EPDK : Enerji Piyasası Denetleme Kurulu

TEDAŞ : Türkiye Elektrik Dağıtım Anonim Şirketi

TWh : Tera Watt Saat

MWh : Mega Watt Saat

MLP : Çok Katmanlı Algılayıcı

DNN : Derin Sinir Ağları

FFNN : İleri Besleme Ağları

CPU : Merkezi İşlem Birimi

GPU : Grafik İşleme Birimi

TPU : Tensör İşleme Birimi

ARIMA : Otoregresif Entegre Hareketli Ortalama

SARIMA : Mevsimsel Otoregresif Entegre Hareketli Ortalama

ESS : Enerji Depolama Sistemleri

BPNN : Geçitli Tekrarlayan Birim

ReLU : Doğrultulmuş Lineer Ünite

ESS : Enerji Depolama Sistemleri

BPNN : Geçitli Tekrarlayan Birim

ARMA : Otoregresif Hareketli Ortalama

ARFIMA : Otoregresif Kesirli Bütünleşik Hareketli Ortalama

QRLSTM : Nicel Regresyon Uzun Kısa Süreli Bellek Sinir Ağı

KDE : Çekirdek Yoğunluk Tahmini

MEC : Çoklu Elektrik Enerjisi Tüketimi Tahmini

TLL : Transfer Öğrenme ve Uzun Kısa Süreli Bellek

TL-MCLSTM: Zaman Konumlu Çok Kanallı Uzun Kısa Süreli Bellek

ConvLSTM : Evrişimli Uzun Kısa Süreli Bellek

QRLSTM : Nicel Regresyon Uzun Kısa Süreli Bellek Sinir Ağı

KDE : Çekirdek Yoğunluk Tahmini

PJME : PJM Doğu Bölgesi

ÖZET

Doktora Tezi

DERİN ÖĞRENME İLE FARKLI MODEL VE ÖĞRENME ALGORİTMALARI KULLANILARAK KISA VADELİ ELEKTRİK YÜK TÜKETİMİNİN TAHMİNİ

MEHMET TAHİR UÇAR

İnönü Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik Elektronik Mühendisliği Anabilim Dalı

113+XVII sayfa

2025

Danışman: Prof. Dr. Asım KAYGUSUZ

Akıllı şebekeleri akıllı kılan önceden elde edilmiş bilgi ve verilere göre yapılan çıkarımlardır. Geçmişte elde edilen zaman verileri özellikle enerji planlamalarında ve öngörülerde önem arz etmektedir. Enerji çalışmaları için planlama ve veri eldesi önemlidir. Fakat zamanla değişen olayları modellemek, veri analizinin zorlu yönlerinden biridir. Bu kıyasla, zamanla değişen elektrik gücü değerlerini tahmin etmekte önemli bir veri analizi problemidir. Verilerdeki örüntüleri belirlemek ve tahmin modelleri geliştirmek için regresyon, makine öğrenimi ve derin öğrenme yöntemleri kullanılmaktadır. Böylece, güç üretim veya tüketimi tahmini için en başarılı modelleri belirlemek ve mümkün olan en yüksek doğruluk seviyesine ulaşmak önem arz etmektedir. Yapılan bu tez çalışmasında uygulanan yöntemler kapsamında öncelikle makine öğrenmesi ile basit bir çalışma yapılmaktadır. Bu ilk çalışmada regresyon analizi yöntemleri kullanılarak Diyarbakır'a ait yıllık nüfus ve elektrik tüketim verileri üzerinden sonraki yıllara ait nüfus ve yıllık elektrik tüketim tahmini yapılmaktadır. Sonrasında ise büyük veri ile derin öğrenme çalışması yapılmaktadır. Teze amaç olan bu çalışmada ilk olarak, hazır bir veri kümesini değerlendirmek ve düzenlemek ve özniteliklerini çıkarmak için Keşifsel Veri Analizi (EDA) kullanılmaktadır. Sonrasında Uzun Kısa Dönem Bellek (LSTM), Geçitli Tekrarlayan Birim (GRU), Basit Tekrarlayan Sinir Ağı (SimpleRNN) ve Çift Yönlü Uzun Kısa Dönem Bellek (BiLSTM) mimarileri kullanılmaktadır. 3 ayrı çalışma mantığının uygulandığı bu çalışmanın ilk kısmında bu dört mimari on bir farklı optimizasyon yöntemi ile kullanılmaktadır. İkinci kısmında, çalışma bir dizi farklı epok sayısı kullanılarak test edilmektedir. Üçüncü kısmında ise, dört mimari, 11 optimizasyon yöntemi ve altı aktivasyon fonksiyonu kullanılarak 264 model üretilmekte ve denemeler bu modeller üzerinden gerçekleştirilmektedir. Sonuçlar kök ortalama karesel hata (RMSE), ortalama mutlak hata (MAE) ve R^2 skor indekslerine göre analiz edilip grafiklere aktarılmaktadır. R^2 skor endeksine göre, çalışmadaki en yüksek değer olarak 0,9979'luk bir başarı oranına ulaşılmaktadır. Son olarak, en başarılı on uygulama listelenmektedir.

Anahtar Kelimeler: Derin öğrenme modelleri, Keşifsel veri analizi, Optimizasyon ve aktivasyon fonksiyonları, Elektrik tüketim tahmini, SimpleRNN, LSTM, GRU, BiLSTM.

ABSTRACT

Phd. Thesis

FORECASTING SHORT-TERM ELECTRICITY LOAD CONSUMPTION USING DIFFERENT MODELS AND LEARNING ALGORITHMS WITH DEEP LEARNING

Mehmet Tahir Uçar

Inonu University
Graduate School of Nature and Applied Sciences
Department of Electrical and Electronics Engineering

113+XVII paper

2025

Supervisor: Prof. Dr.Asım Kaygusuz

What makes smart grids smart are the inferences made according to previously obtained information and data. Time data obtained in the past is especially important in energy planning and predictions. Planning and data acquisition are important for energy studies. However, modeling events that change over time is one of the challenging aspects of data analysis. In comparison, estimating electrical power values that change over time is an important data analysis problem. Regression, machine learning and deep learning methods are used to determine patterns in the data and develop prediction models. Thus, it is important to determine the most successful models for power production or consumption prediction and to reach the highest possible accuracy level. In the study conducted, a simple study is first carried out with machine learning. In this first study, the population and annual electricity consumption estimates for the following years are made using regression analysis methods on the annual population and electricity consumption data of Diyarbakır. Then, a deep learning study is carried out with big data. In this study, which is the aim of the thesis, Exploratory Data Analysis (EDA) is first used to evaluate and organize a ready data set and extract its features. Afterwards, Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Simple Recurrent Neural Network (SimpleRNN) and Bidirectional Long Short Term Memory (BiLSTM) architectures are used. In the first part of this study where 3 different working logics are applied, these four architectures are used with eleven different optimization methods. In the second part, the study is tested using a series of different epoch numbers. In the third part, 264 models are produced using four architectures, eleven optimization methods and six activation functions and the experiments are carried out on these models. The results are analyzed according to the root mean square error (RMSE), mean absolute error (MAE) and R^2 score indices and transferred to the graphics. According to the R^2 score index, a success rate of 0.9979 is reached as the highest value in the study. Finally, the ten most successful applications are listed.

Keywords: Deep learning models, Exploratory data analysis, Optimization and activation functions, Electricity consumption prediction, SimpleRNN, LSTM, GRU, BiLSTM.

1. GİRİŞ

Bilgi iletişim teknolojilerindeki ilerlemeler ve artan veriye bağlı yapay zekâ uygulamalarındaki ilerlemeler sayesinde akıllı şebekeler gittikçe daha modern ve güvenilir hale gelmektedir. Böylece enerji şebekeleri daha sağlıklı inşa edilmekte, arızalar daha iyi tespit edilip çözülmekte, ihtiyaçlar daha sağlıklı belirlenmekte, kesintiler azalmakta, öngörüler artmakta, kaçak kullanım azalmakta ve kalite yükselmektedir.

Artan yenilenebilir enerji kullanımıyla birlikte düzensiz olan elektrik kullanım durumuna düzensiz elektrik üretimi de eklenmiştir. Bu durumun çözümü için yük tahminleri yapılması ve buna uygun üretim planlamaları yapılması gerekmektedir. İşte tam da bu noktada akıllı sayaçlar ve yapay zekâ uygulamaları devreye girmektedir. Akıllı şebekenin önemli atılımlarından biri de akıllı sayaçlardır. Akıllı sayaçlar kullanıcıya ait kullanım, arıza, kesinti, kaçak durumları hakkında kayıt tutan, bu kayıtlara uzaktan müdahale ve kontrol imkânı sağlayan cihazlardır. Ayrıca akıllı sayaçların kaydetme özelliği sayesinde büyük kullanım verileri ortaya çıkmaktadır. Bu noktada da yapay zekâ, makine öğrenmesi ve derin öğrenme çalışmaları devreye girmektedir. Akıllı sayaçlar, üretim bilgileri ve hava tahmin raporlarından elde edilen veri setleri üzerinden yapay sinir ağı modellemeleri kullanılarak üretim tahminleri ve yük tahminleri yapılmakta ve sistemin en öngörülen şekilde ve en az maliyet ile çalışmasına olanak sağlanmaktadır. Bu çözümler hem Entegre Enerji Sistemlerinde hem de akıllı ev gibi küçük ölçekli yapılarda da kullanılabilmektedir.

Akıllı sistemler alanında günümüze varıncaya dek pek çok ilerlemeler kaydedilmiştir. Araştırmacıların çoğu gerçek sinir hücrelerinden esinlenmişlerdir. Farklı alanlardan araştırmacılar örüntü tanıma, ilişkisel bellek, kontrol, optimizasyon, tahmin gibi çeşitli alanlardaki sorunlara çözüm üretebilmek amacıyla yapay sinir ağları tasarlamışlardır. Bu konularda karşılaşılan problemlerle başa çıkabilmek için çok sayıda geleneksel metot öne sürülmüştür. Kullanılan bu metotların bazıları kısıtlı alanlarda iyi sonuçlar üretmesine karşın, etki ettikleri alan dışında yeterli başarı sağlamamışlardır [1]. Yani, etki alanı dışında başarı elde edebilecek esnekliğe ve performansa ulaşamamışlardır. Buna rağmen yapay sinir ağları bu konularda oldukça etkileyici sonuçlar vermektedir [2].

Yapay sinir ağları ve insan beyni günümüz bilgisayarlarından farklı olarak genelleştirme ve öğrenme yeteneğine, doğal içeriğe bağlı bilgi işlemeye, hata toleransı ve düşük enerji tüketimi gibi yetenekler sergilemektedir [3]. Gerçek sinir ağlarından faydalanılarak geliştirilen yapay sinir ağlarının bahsedilen yeteneklerin büyük kısmını karşılayacağı öngörülmektedir.

Derin öğrenme, makine öğrenmesinin bir alt dalı olarak son zamanlarda popülerliğini oldukça artmıştır. Derin öğrenme yöntemi çok büyük ve karmaşık olasılıklı modellerin eğitilmesi ve bu modellere öğrenme yeteneğinin kazandırılması konusunda çok başarılı bir metottur [4]. Özellikle derin yapay sinir ağları kullanılan klasik yaklaşımlara nazaran başarılı sonuçlar üretmektedir. Derin sinir ağları, verileri işleyen birimlerden oluşan çok sayıda doğrusal olmayan katmandan meydana gelmektedir [5]. Olasılıklı model yapısında mevcut verilerin eğitimi esnasında katmanlı denetimsiz ön eğitim yöntemi kullanılmaktadır. Derin ağdaki katmanlar düşük seviyeli özelliklerden yüksek seviyeli özelliklere doğru ilerleyen hiyerarşik bir yapıya sahiptir [6]. Zamanla değişiklik gösteren olayların modellenmesi zorlu bir veri analizi problemidir. Bu olaylardan biri olan elektrik güç tüketiminde ise veriden farklı örüntülerin öğrenilerek bir tüketim tahmin modelinin geliştirilebilmesi için klasik makine öğrenmesi ve özellikle derin öğrenme yöntemlerinden faydalanılmaktadır. Bu çalışmada, hazır veri kümeleri üzerinden bir kısa vadeli tüketim tahmin metodu geliştirilmekte ve farklı modeller kullanılarak derin öğrenme algoritmaları ile tahmin çalışmaları yapılmaktadır.

Yapılan bu tüketim tahmin hesaplarına göre üretim yapan birimlerin üretecekleri veya satabilecekleri enerji miktarını programlamaları ve yük tevzi birimlerinin tahmin sonuçlarına göre sonraki günler için daha kolay programlama yapabilecekleri düşünülmektedir. Böylece talep edilen ve üretilen enerji birbirini karşılayacak ve plansız enerji kesintileri, voltaj dalgalanmaları ve sistem çökmeleri önlenmiş olacaktır.

Ayrıca bu tahmin sonuçları, enerji üretiminde, enerji tüketiminde, enerji satış aşamalarında, serbest piyasa alım satımlarında, ünitelerin bakım programlarında, ünitelerin yakıt kullanımı hesabında, ünitelerin çalışma programı aşamasında ve tüm bu sistemlerin işletim aşamalarında oldukça önemlidir.

Elektrik enerjisi tahmini için yapılan çalışmaları 3 sınıfa ayırabiliriz. Bunları da kısa vadeli yük tahmini (STLF), orta vadeli yük tahmini (MTLF) ve uzun vadeli yük

tahmini (LTLF) olarak sıralayabiliriz. Bu çalışmada kısa vadeli elektrik tüketim tahminleri baz alınmaktadır.

STLF için genellikle geçmiş yük değerleri yeterli olmasına rağmen bazen hava durumu bilgileri de kullanılmaktadır. MTLF için hava durumu bilgileri ile birlikte ekonomik bilgiler de kullanılmaktadır. LTLF ise hava durumu, ekonomi, demografik bilgileri ve bazen de arazi kullanım bilgilerini kullanmaktadır [7].

Çalışmada son yıllarda oldukça ilgi uyandıran bir makine öğrenmesi yaklaşımı olan derin öğrenme kavramı ve bu yöntemin akıllı şebeke alanıyla ilgili kısımlarına da değinilmektedir. Çalışmada özellikle kısa vadeli yük tahmini üzerinde durulmaktadır. Bu alandaki 4 ana model olan SimpleRNN, LSTM, GRU ve BiLSTM kullanılarak çalışmalar yapılmaktadır. Ayrıca aktivasyon fonksiyonu, epok sayısı ve optimizasyon fonksiyonu kullanımıyla ilgili oldukça fazla seçenek denenmektedir. Amaç çalışılmamış, denenmemiş veya pek sık kullanılmayan yöntemlere de vurgu yapıp bu yöntemlerin kullanımının sonuçlara etkisini analiz etmek ve bu alanda çalışacak araştırmacılar için bir ön çalışma sunmaktır. 1. Bölümde genel hatlarıyla konuya giriş yapılmaktadır. 2. Bölümde genişçe bir literatür araştırması yapılmaktadır. 3. Bölümde genel kavramlar işlenmekte, yapay zekâ ve yapay sinir ağlarına değinilmektedir. Yapay zekânın bir alt birimi olan makina öğrenmesinin ne olduğu açıklanmakta, makina öğrenmesinin bir alt başlığı olan derin öğrenme ile farklı olduğu noktalar belirtilmektedir. Ayrıca makine öğrenmesi ile ilgili basit bir örnek sunulmaktadır. 4. Bölümde asıl konumuz olan derin öğrenmede kullanılan metotlar, araçlar, aktivasyon fonksiyonları, optimizasyon yöntemleri, sinir ağı mimarileri, programlama dilleri, veri kümeleri ve kütüphaneler anlatılmaktadır. Ayrıca yük tahmini için kullanılan derin öğrenme metotları anlatılmaktadır. 5. Bölümde önerilen çalışmanın materyal, yöntem ve diğer çalışmalardan ayrılan yönleri sunulmaktadır. 6. Bölümde Çalışma ve Analizler sunulmaktadır. 7. Bölümde ise konu ile ilgili genel olarak çalışmanın bulguları ve sonuçları belirtilip değerlendirme yapılmaktadır. Son olarak tez Bölüm 8 ile sonuçlandırılmaktadır.

Bu tez, yazarın bir makale ve bir kitap bölümünden [8,9] materyaller kullanmaktadır.

1.1 Tezin Amacı

Literatür çalışmalarında görüleceği üzere tüm derin öğrenme modelleri ile çalışmalar mevcuttur. Yani akıllı şebeke ile ilgili yapılacak olan çalışmalarda tüm modellerin kullanılması olasıdır. Nesne tanıma alanlarında CNN ve sıralı veya dizi halindeki verilerin değerlendirilmesinde RNN modelleri daha öne çıkmaktadır. Buna göre eldeki veriler dizi veya zaman serisi olacağından RNN modelleri kullanılması daha uygun olacaktır. Fakat RNN modelleri eski bilgileri hatırlayıp değerlendirmede sorun yaşamaktadır. Bu yüzden çalışmalarda yine bir RNN modeli olan LSTM (hafızalı RNN), GRU ve Bi-LSTM modelleri tercih edilmektedir. LSTM yöntemi özellikle tahmin etme konusunda çok başarılı çalışmalara sahne olduğu için tahmin problemlerinde çokça kullanılmaktadır. Özellikle yük tahmini veya üretilecek enerji tahmininde son yıllarda çokça çalışma yapılmaktadır.

Bu tezdeki çalışmada tahmin doğruluğunu arttırmak amacıyla farklı 4 mimari (LSTM, GRU, SimpleRNN ve BiLSTM), 6 aktivasyon fonksiyonu (relu, tanh, elu, leakyrelu, sigmoid ve softmax) ve 11 optimizasyon fonksiyonu (adam, nadam, rmsprop, rmsprop*, adamax, sgdtrue, sgd, adagrad, ftrl, adadelta ve sgdfalse) kullanılmaktadır. Yapılan her denemede diğerleri sabit tutularak sadece bir değer değiştirilerek 4*6*11 deneme, yani 264 deneme yapılmıştır. Ayrıca LSTM-relu ile yapılan 50 epok denemesi dışında 10 ve 100 epok ile 11 farklı aktivasyon fonksiyonu kullanılarak 4*2*11 deneme, yani 88 deneme daha yapılmıştır. Toplamda 352 farklı varyasyon denenmiştir.

Bu tez çalışmasındaki amaç; kullanılan mimari, epok sayısı, aktivasyon fonksiyonu ve optimizasyon fonksiyonu gibi değişkenleri değiştirerek sonuçları analiz etmektir. Çalışmaların çoğunda tek mimari, tek aktivasyon fonksiyonu ve tek optimizasyon fonksiyonu kullanılmaktadır. Burada yapılan denemelerin büyük kısmı yayınlarda denenmemiş veya en yüksek sonucu veren mimariye tek değinilmiştir. Bu yönüyle yapılan çalışmaların tahmin konusunda kullanılacak değişkenler ve hiperparametreler açısından kaynak oluşturacağı düşünülmektedir.

2. KONU KAPSAM ve LİTERATÜR ÖZETİ

Tahmin alanında farklı alanlarda yıllardan beridir çok sayıda çalışma yapılmaktadır. Bu çalışmaların içinde en büyük kısmını borsa tahminleri oluşturmaktadır [10-13]. Bu arada hisse senedi alım satım işlemleriyle ilgili [14], belirlenen bir endeksin gün içinde hareketiyle ilgili [15], ve hava sıcaklığı ile ilgili tahmin çalışmaları da sıklıkla görülmektedir [16]. Bunların dışında meterolojik durum, hava sıcaklığı, güneş ve rüzgar değerleri ve hava kirliliği gibi konuların tahminiyle ilgili de çok sayıda çalışmalar mevcuttur [17-20].

Elektrik alanında ise kalan ömür (RUL) ve bozulma durumu tahmini gibi çalışmalar yapılmıştır [21-23]. Kestirimci bakım (PdM) alanında yapılan bir çalışmada geniş bir literatür çalışması yapıldığı, uygulanan makine öğrenimi yaklaşımlarının öğrenme stratejileri, avantajları, sınırlamaları ve tipik uygulamalarının tablolar halinde sunulduğu görülmektedir [24]. Akıllı şebekeler alanına bakıldığında ise güneş veya rüzgar potansiyeli tahmini, tüketilen ve üretilen enerji tahmini ve kısa devre arızalarının yerlerinin tahmini gibi çalışmalar mevcuttur [25-27].

Geleneksel ve istatistiksel yöntemler, normal şartlarda yıllardır kullanılmasına ve belli bir başarı seviyesini yakalamış olmasına rağmen değişen hava durumu, farklı sosyolojik ve ekonomik durum, tatil günleri ve düzenli olmayan veriler gibi farklı etkiler başarım oranlarını iyileştirme ihtiyacını doğurmuştur. Bu nedenle, araştırmacılar yeni arayışlara yönelmiş ve yapay zekâ yöntemleri önem kazanmıştır. Yük tahmini için kullanılan başlıca yapay zeka yöntemleri yapay sinir ağları (YSA), bulanık mantık, karar ağaçları (DT), destek vektör makineleri (DVM-SVM), destek vektör regresyonu (SVR), genetik algoritma (GA), k-en yakın komşular (k-NN), parçacık sürü optimizasyonu (PSO) ve karınca kolonisi optimizasyonu (KKO) olarak değerlendirilmektedir [28-30]. Literatürde tahmin çalışmaları için sıklıkla kullanılmış olan regresyon analizi çalışmalarına detaylıca bakacak olursak;

Tuğba Akman ve arkadaşları yük tahminleri için kullanılan zaman serileri analizi, regresyon analizi, yapay sinir ağları, destek vektör makineleri, hibrit ve diğer yaklaşım yöntemlerinin farklılıkları, avantajları ve dezavantajlarını değerlendirmişlerdir [31].

Arzu Arı ve Hasan Önder regresyon yöntemlerinden; doğrusal regresyon, lojistik regresyon, temel bileşenler regresyonu, negatif binom regresyon, probit regresyon, ridge

regresyon, poisson regresyon ve cox regresyon yöntemlerinin hangi durumlarda kullanılabileceği konusunda çalışmalar gerçekleştirmişlerdir [32].

Hüseyin Avni Es ve arkadaşları 1970-2010 yılları arasındaki gayri safi yurtiçi hâsıla (GSYH) , nüfus, ithalat, ihracat, bina yüz ölçümü ve taşıt sayısı değişken verilerini YSA modelinin girdisi olarak kullanarak çoklu doğrusal regresyon tekniği ile Türkiye net enerji talebi tahminini gerçekleştirmişlerdir [33].

Karaca ve Karacan basit ve çoklu regresyon analizi ile ortalama yaşam beklentisi, internet kullanımı ve gayri safi yurt içi milli hasıla değerlerini kullanarak tahminleme çalışması yapmış ve sonuçları karşılaştırmışlardır [34].

Yalçın Alcan ve arkadaşları regresyon analizi, YSA ve hibrit (YSA- regresyon) modeli kullanarak 2012-2016 yıllarına ait elektrik enerjisi tüketim verilerini ve tarih, sıcaklık, hava durumu, üfe-tüfe, abone bilgileri ve bazı ekonomik veriler üzerinden Sinop/Türkiye elektrik enerjisi tüketim tahmini yapmışlardır [35].

Ali Osman Özkan ve arkadaşları 1960-2018 yılları arası nüfus bilgileri, Gayri Safi Yurtiçi Hâsıla ve enerji tüketim verilerini kullanarak WEKA yazılımı yardımıyla regresyon algoritmalarıyla Türkiye'nin 2019 ile 2023 yılları arasına ait uzun dönem enerji tüketim tahmin çalışması yapmışlardır [36].

Gökhan Turan Fransa'da bulunan akıllı bir evin 4 yıllık elektrik enerjisi tüketimi verileri üzerinden Knime aracı ile farklı makine öğrenme algoritmaları olan doğrusal regresyon, basit ağaç regresyon, gradyan destekli ağaç regresyon ve ağaç topluluğu regresyon analizleri yapmıştır [37].

Emine Elif Nebati ve arkadaşları Türkiye'deki elektrik tüketim verileri doğrultusunda, kullanılan beyaz eşya sayısı ve iş yeri sayısı bağımsız değişkenleri üzerinden zaman serisi ve regresyon analizi tekniklerinden faydalanarak talep tahmin sonuçlarını analiz etmişlerdir [38].

Harika Ülkü ve Şükran Yalpır çoklu regresyon analizi ve yapay sinir ağları yöntemleri kullanmışlardır. Çalışmalarında 2009 ile 2018 yılları arası nüfus, eğitim durumu, ortalama hane halkı büyüklüğü, ihracat, ithalat, gayri safi yurtiçi hâsıla, sanayi girdi ve elektrik enerji tüketim miktarları üzerinden tüm illere göre 2030 yılı elektrik enerjisi ihtiyacı için tahminde bulunmuşlardır [39].

Yeşim Er ve Eda Karaca çoklu doğrusal regresyon, destek vektör regresyonu ve rastgele orman regresyonu olarak üç farklı yöntem ile Karadeniz Bölgesi'ndeki 18 ilin ve bölgenin aylık elektrik tüketimini tahmin etmeye çalışmışlardır [40].

Hyojoo Son ve Changwan Kim aylık elektrik tüketimi, 14 hava durumu değişkeni ve beş sosyal değişken olmak üzere toplam 20 etkili değişken üzerinden parçacık sürüsü optimizasyon algoritmaları, destek vektör regresyonu ve bulanık-kaba öznitelik seçimine dayanan yöntem ile aylık elektrik tüketimlerini tahmin etmişlerdir. Üretilen sonuçlar çoklu doğrusal regresyon modelleri, yapay sinir ağı, oto-regresif entegre hareketli ortalama ve literatürde kullanılan diğer yöntemlere ait sonuçlarla kıyaslanmıştır [41].

Oğuz Kaynar ve arkadaşları destek vektör regresyon yöntemiyle Türkiye'nin 1975-2014 yılları arası elektrik tüketim, nüfus, ithalat, ihracat ve GSYH verilerini kullanarak bir tahmin çalışması yapmışlardır [42].

Xiaoou Monica Zhang ve arkadaşları ile Yang, M ve arkadaşları enerji tüketim tahmini için SVM yöntemini kullanmıştır. Konutlar için takvim, hava durumu, kullanım zamanı fiyatı ve enerji tüketimi verilerine göre yapılan tahmin sonuçlarına göre günlük ve saatlik tahmin çalışmalarında SVM yönteminin başarılı olduğu görülmüştür [43,44].

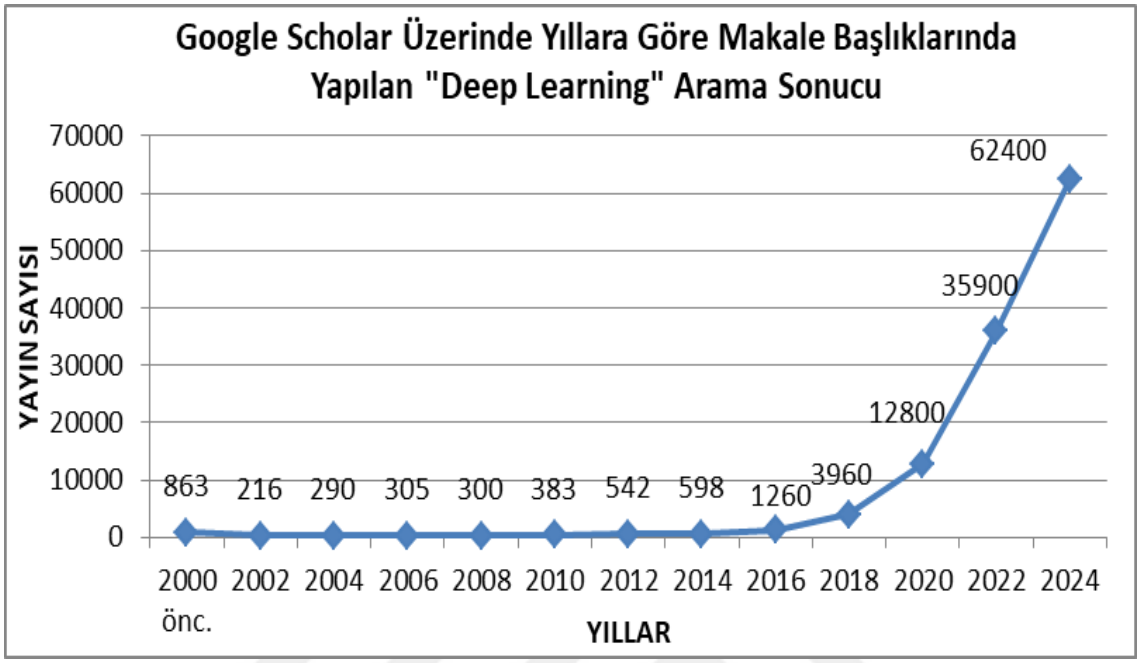
Mehmet Emin Baltaş ve Cuma Akbay Akdeniz Elektrik Dağıtım Bölgesinin tüketimini, Türkiye tüketimi ve GSYH işe ilişkilendirerek nüfus, brüt talep, gayri safi yurt içi hâsıla (GSYH) ve kişi başına gayri safi yurt içi hâsıla (KBGSYH) değişkenleri üzerinden regresyon modeli ile tahmin etmeye çalışmıştır [45].

Literatürde çokça çalışılan makine öğrenmesi yöntemlerinden biri olan regresyon analizi yöntemlerinin başarılı sonuçlar verdiği görüldüğünden öncelikle Bölüm 3.4'te eldeki Diyarbakır iline ait yıllık tüketim verileri üzerinde basitçe ve kısaca regresyon analizi yöntemleri kullanılmaktadır. Bu çalışmanın geniş hali kitap bölümü olarak yayınlanmıştır. Bu mimarilerin en bilinenlerini doğrusal regresyon, polinomal regresyon, destek vektör regresyonu, karar ağaçları ve rasgele orman halinde sıralanmaktadır.

Yapay zekâ, yapay sinir ağları, makine öğrenmesi ve derin öğrenme konularında çok farklı alanlarda ve çok sayıda makale, tez ve bildiri çalışmaları yapılmakta ve başarılı sonuçlar elde edilmektedir. Fakat biz konuyu fazla dağıtmadan derin öğrenme ile ilgili yayınlara bakmaya çalışacağız.

Google Scholar üzerinden makale aramasını tıklanıldığında, kelimelerin tümünü içeren kısmına "Deep Learning (Derin Öğrenme)" yazıp kelimelerin geçtiği yerleri

“makale başlığı” seçip uygun yılları yazıldığında aşağıdaki grafik elde edilmektedir. Grafikten de görüldüğü üzere son yıllarda derin öğrenme konusundaki araştırmalar çok artmıştır.



Şekil 2.1 : Google scholar üzerinde makale başlıklarında yapılan "deep learning" arama sonucu.

Literatür çalışmalarında görüleceği üzere yapay zekâ yöntemleri, tahmini etkileyen farklı özellikleri tutturmada ve daha başarılı tahmin sonuçları üretmede istatistiki yöntemlerden daha başarılı olduğundan araştırmacılar tarafından sıklıkla kullanılmaktadır.

İstatistiksel tahmin modellerinin doğrusal olmayan verilere uymaması, model parametrelerinin duyarlılığı ve model genelleme için düşük yetenek gibi çeşitli nedenlere rağmen zaman serileri için yapay zeka yöntemleri başarı göstermektedir. Fakat yöntemlerin çoğu önceden tanımlanmış doğrusal olmayan şekli benimsemekte ve doğrusal olmayan ilişkiyi simüle edememektedir [46].

Yapay zeka ile ilgili makine öğrenmesi modelleri bilinmesine rağmen yeterli veri olmamasından dolayı pek tercih edilmemekte ve kullanılmamaktaydı. Akıllı sayaçların kullanılması ve kullanımının yayılmasıyla yeterince veriye ulaşıldı. Böylece daha önceki modeller yerine makine öğrenme modelleri tercih edilmeye ve başarılı sonuçlar vermeye başladı.

Bu veriler ve gelişen veri bilimi sayesinde makine öğreniminin bir alt dalı olan derin öğrenme, hızla gelişti ve bazı alanlarda tahmin doğruluğu ve verimliliği bakımından makine öğrenimini geçti [47].

Derin öğrenme alanındaki çalışmaların büyük kısmını nesne tanıma, bilgisayarlı görü, doğal dil işleme ve konuşma tanıma konuları oluşturmaktadır.

Günümüzde en çok çalışılan derin öğrenme çalışmalarını: Özerk Araçlar (Autonomous Vehicles), Doğal Dil İşleme (Natural Language Processing), Ses ve Video Tanıma (Voice and Video Recognition), El Yazısı Karakter Tanıma (Handwritten Character Recognition), Tıbbi Görüntü İşleme (Medical İmage Processing), Büyük Veri (Big Data), İmza Doğrulama (Signature Verification) olarak sıralayabiliriz. Bu popular konular ile ilgili detaylı literatür çalışması mevcuttur [48]. Otonom araçlar, tıbbi görüntü işleme, büyük veri analizi ve karakter tanıma gibi konularda da derin öğrenme gün geçtikçe daha başarılı sonuçlar vermektedir.

Derin öğrenme çalışmalarında farklı problemler ve farklı alanlar için farklı mimarilerin tasarlandığı görülmektedir. Bu mimarilerden en bilinenleri CNN (Evrişimli Sinir Ağları), RNN (Özyinelemeli Sinir Ağları), Uzun Kısa Dönem Hafıza (LSTM), Auto-Encoder (Otomatik Kodlayıcı), RBM (Kısıtlı Boltzmann Makinesi), Transfer Learning (Transfer Öğrenimi), DBN (Derin İnanç Ağları), GAN (Çekişmeli Üretici Ağlar) ve DRL (Derin Pekiştirmeli Öğrenme) olarak görülmektedir.

Yapay zekâ, makine öğrenme ve derin öğrenme uygulamalarıyla ilgili de geniş bir literatür çalışması yapılmıştır [49]. Yapılan bu çalışmadan elde edilen aşağıdaki Çizelge 2.1'de derin öğrenme ağları, bu ağlara ait başlıca sınırlamalar, tipleri ve bu ağların alt uygulama başlıklarına göre yapılmış bazı çalışmalar listelenmiştir.

Derin öğrenmede, zaman serisi verileri için özellikle kendi kendine geri bildirimli nöronları kullanarak zamansal ilişkilere sahip dizileri işleyen tekrarlayan sinir ağları (RNN'ler) kullanılır [50]. Güç tüketimi modellerini uzun vadeli bağımlılıklar yoluyla belirlemeye çalışılsa da RNN'nin tahmin edilen değeri yalnızca son değerle ilişkilidir. RNN modelinin kullanıldığı çalışmalarda gradyan kaybolması veya gradyan patlama sorunu sıklıkla gözlemlenmektedir. Bu da tahmin doğruluğunu azaltmaktadır. Bu sorunları çözmek için, son yıllarda RNN'nin bir alt dalı olan kapılı çevrimsel birim LSTM kullanılmaktadır [47].

Çizelge 2.1 Derin öğrenme çalışmalarında kullanılan modellerin avantajları, sınırlamaları ve genellikle kullanıldığı alanlar [49].

Mimari Tipi	Avantajlar	Sorunlar	Kullanım Alanları
Auto-Encoder (Otomatik Kodlayıcı)	• Öncesinde veri bilgisine ihtiyaç duymaz • Verileri sıkıştırabilir ve çok duyusal verileri birleştirebilir • Regresyon veya sınıflandırma metotları ile birleştirilmesi kolaydır.	• Ön eğitim için fazla miktarda veri gerekir • İlgili bilgilerin hangileri olduğu belirlenemez • Üretken Çekişmeli Ağlara (GAN) kıyasla yeniden yapılandırma işlemlerinde pek başarılı değildir.	• Özellik çıkarma. •Çok duyulu (sensörlü) veri füzyonu. • Arıza teşhisi. • Bozulma süreci tahmini. • Kalan Faydalı Yaşam (RUL) tahmini.
CNN (Evrişimli Sinir Ağları)	• Birçok görevde YSA (Yapay Sinir Ağları)'dan daha iyi performans gösterir (örneğin, görüntü tanıma) • YSA'ya göre daha az karmaşıktır ve hafıza tasarrufu sağlar • Önemli özellikleri insan gözetimi olmadan otomatik olarak algılar	• Hiper parametre ayarı önemsizdir • Kolay uydurulabilir • Yüksek hesaplama maliyeti • Çok fazla eğitim verisine ihtiyaç duyar	• Arıza teşhisi. • Bozulma süreci tahmini. • RUL tahmini. • Ortak arıza tespiti ve RUL tahmini.
RNN (Özyinelemeli Sinir Ağları)	• Zamana bağlı sıralı modeller	• Gradyanların kaybolması ve patlaması problemleri • Aktivasyon fonksiyonu olarak tanh veya relu kullanılıyorsa çok uzun diziler işlenemez	• Arıza teşhisi. • RUL tahmini. • Sağlık göstergelerini yorumlama.
DBN (Derin İnanç Ağları)	• Yukarıdan aşağıya, üretken ağırlıkların öğrenilmesi için katman prosedürü vardır • Ön eğitim sırasında etiketlenmiş verilere gerek yoktur • Sınıflandırmada sağlamlık	• Yüksek hesaplama maliyeti	• Özellik çıkarma. • Hata sınıflandırması. • Erken arıza tanısı ve RUL tahmini.
GAN (Çekişmeli Üretici Ağlar)	• Sınıflandırıcıları yarı denetimli bir şekilde eğitmek için iyi bir yaklaşım • Otomatik kodlayıcılara kıyasla herhangi bir deterministik sapma oluşturmaz • Sınıf dengesizliği sorununu çözmek için kullanılabilir	• Nash dengesi (oyun teorisi) gereksinimi nedeniyle eğitim kararsızdır • Orijinal GAN'ın ayrık veri üretmeyi öğrenmesi zordur	• Sınıf dengesizliği sorunu. • Arıza tanımlama. • RUL tahmini.
Transfer Learning (Transfer Öğrenimi)	• Eğitim süresini kısaltır. • Daha az veri ile hedeflenen görevi yerine getirir. • Simülasyonlardan bilgi öğrenebilir	• Bilgi transferinin mümkün olması için 'uygun' olmalıdır. • Negatif aktarımdan muzdariptir.	• Arıza teşhisi: AdaBN , ters tabanlı alan uyarlaması, temsil uyarlaması, dijital ikiz, parametre aktarımı. • RUL tahmini.
DRL (Derin Pekiştirmeli Öğrenme)	• Fazlasıyla karmaşık problemleri çözümlemek için tercih edilebilir. • Araştırma ve kullanım arasındaki dengeyi korur	• Fazla miktarda veri ve hesaplamaya ihtiyaç duyuyor gerekiyor • Dünyanın Markovyan olduğunu varsayar ki • Boyutsallığın lanetinden muzdarip • Ödül fonksiyonu tasarımı zor	• İşletme ve bakım karar verme. • Arıza teşhisi. • Sağlık göstergesi öğrenimi.

Yapılan zaman serisi tahmin çalışmalarının çoğunda LSTM kendine yer edinmiş ve kısa vadeli yük tahmini çalışmalarında rakip algoritmalardan daha iyi performans gösterme başarısı elde etmiştir [51-53].

Bir kısım çalışmalarda ise LSTM tarzı derin öğrenme tabanlı algoritmaların otoregresif entegre hareketli mesafe (ARIMA) modeli gibi geleneksel tabanlı algoritmalardan daha başarılı olduğu görülmektedir [54,55].

LSTM modeli uygun çözümler sunmaktadır. Bundan dolayı zaman serileri ile ilgili son zamanlarda yapılan derin öğrenme çalışmalarının çoğunda tercih edilmektedir. Bununla birlikte bu başarım yeterli görülmeyip daha yüksek başarım metrikleri için LSTM modeli, tahmin verisi, eğitim metodu, hiperparametre ve diğer değişkenlerle oynanarak çalışmalar geliştirilmiştir. Literatürdeki bu tarz tahmin başarımını arttırmaya yönelik denenmiş ve başarılı olmuş çalışmalara kısaca göz atacak olursak:

Li ve arkadaşları çalışmalarında modern endüstriyel parklara enerji depolama sistemlerinin (ESS'ler) dahil edilmesiyle enerji depolamanın etkisini hesaba katan kısa vadeli bir yük tahmin yöntemi kullanmaktadır. Ayrıca, ESS güç akışını LSTM-RNN algoritmalarına dahil etmek için iki stratejinin sonuçlarını karşılaştırmaktadır. Sonuçlara göre ESS etkisinin son işlenmesinden daha küçük bir hataya sahip olduğu görülmektedir [56].

Lin ve arkadaşları çift aşamalı dikkat temelli uzun kısa süreli bellek (LSTM) ağı ile çalışmaktadır. Başlangıçta bir özellik dikkat temelli kodlayıcı oluşturulmuştur. En alakalı giriş özellikleri uyarlanabilir şekilde seçilmektedir. Sonrasında zamansal dikkat emelli bir kod çözücü geliştirilmiştir. Daha sonra, LSTM yöntemi bu dikkat sonuçlarını bütünleştirmekte ve bir langırt kaybı fonksiyonu üzerinden tahminler yapılmaktadır [57].

Ding ve arkadaşları evrimsel bir çift dikkat temelli LSTM modeli üzerinden denemeler yapmaktadır. Bu denemelerde sekiz tahmin yönteminin tahmin performansı birbirleriyle kıyaslanmaktadır. Sonuçlar dikkat temelli LSTM modeli verimliliği artırdığını göstermektedir [58].

Wang ve arkadaşları LSTM tabanlı bir model ile enerji tüketim tahmini üzerine çalışmaktadırlar. Öncelikle otokorelasyon grafiği ile gizli özellikler tespit edilmekte, korelasyon analizi ve mekanizma analizi ile ikincil değişkenler bulunmaya çalışılmaktadır. Ardından, sıralanmış verileri modellemek için bir LSTM ağı kullanılmaktadır. Çalışılan modelin BPNN, ARMA, ARFIMA'ya göre daha başarılı olduğu görülmektedir [59].

Zhao ve arkadaşları k-ortalamalar algoritması, nicel regresyon uzun kısa süreli bellek sinir ağı (QRLSTM) ve çekirdek yoğunluk tahmini (KDE) dahil olmak üzere birkaç veriye dayalı ve istatistiksel algoritmayı etkili bir şekilde birleştirmektedir. Sonuçlar, kullanılan yöntemin bazı kıyaslama yöntemlerinden net bir şekilde daha başarılı olduğunu göstermektedir [60].

Le ve arkadaşları MEC-TLL çerçevesi olarak adlandırılan Transfer Öğrenimi Ve Uzun Kısa Süreli Bellek (TLL) kullanan bir akıllı binanın Çoklu Elektrik Enerjisi Tüketimi tahmini (MEC) için sağlam bir çerçeve geliştirmektedir. Burada eğitim setindeki günlük yük talebini kümelemek amacıyla önce bir k-ortalama kümeleme algoritması kullanmaktadır. Sonrasında en uygun küme sayısını belirlemek için Silhouette analizi kullanılmaktadır. Bunlara ilaveten, işlem süresini düşürmek için Uzun Kısa Süreli Bellek modellerini transfer öğrenimi için küme tabanlı bir strateji kullanan MEC eğitim algoritmasını geliştirmektedir. Sonuçlar, akıllı binalarda akıllı enerji yönetimi için önerilen yaklaşımın üstün performanslar gösterdiğini ve ekonomik genel giderlere sahip olduğunu göstermektedir [61].

Wang ve arkadaşları uygulamalarında parametrelerin eğitimine rehberlik etmek için MSE yerine langırt kaybını tercih etmektedirler. Bunu yaparak geleneksel LSTM tabanlı nokta tahminini nicelikler biçiminde olasılıklı tahmine dönüştürmüş olmaktadırlar. Sonuçlara göre, kullanılan yöntemin geleneksel yöntemlerden daha başarılı olduğu görülmektedir [62].

Shao ve arkadaşları çok adımlı kısa vadeli güç tüketimini tahmin etmek için çok kanallı uzun kısa süreli bellek (LSTM) olarak adlandırılan ve zaman konumlu (TL-MCLSTM) çok çıkışlı bir stratejide yeni bir derin model sunmaktadır. Kullanılan model: güç tüketimi, zaman konumu ve müşteri davranış kanalı olmak üzere üç kanaldan oluşmaktadır. Kanallardan güç tüketimi, kullanımdaki değişimi ve genel yönelimi ifade eder. Zaman konumu, gün, saat, tatillerden oluşan bilgileri kaydeden müşteri alışkanlıklarının gizli modelini ifade eder. Güç tüketimi ve zaman konumu kanalları, LSTM aracılığıyla ayrı ayrı eğitilmektedir. Güç tüketimi ve zaman konumu kanallarında LSTM'den çıkarılan özellikler, tahmin edilecek kapsamlı özellikler olarak müşteri davranışıyla birleştirilmektedir. Yöntemin etkinliğini ve önceliğini doğrulanmakta ve çoklu çıktı stratejisi ile güç tüketimi, zaman konumu ve müşteri davranış kanalları ile üç kanaldan zengin özellik temsilleri çıkarabileceğini göstermektedir [63].

Nazir ve arkadaşları en başarılı modeli tespit etmek amacıyla Vanilla LSTM, Stacked LSTM ve Convolutional LSTM (ConvLSTM) ve Bidirectional LSTM modellerini incelemektedirler. ConvLSTM mimarisini kısaca tanıtılmakta, ardından deney kurulumu tanımlanmakta, ardından çıkarımlar yapılmaktadır. Sonuçlara göre, ConvLSTM modelinin, çok adımlı LSTM'den daha başarılı olduğunu göstermektedir [64].

Farklı bir model ise kapı tekrarlayan birim ağları (GRU) olarak değerlendirilmektedir. Yapılan bazı çalışmalarda daha fazla doğruluk sağlamada, tahmini iyileştirmekte, daha iyi performans sonuçları üretmede ve daha hızlı olma yönüyle bazı çalışmalarda GRU'nun LSTM'ye göre daha başarılı olduğu görülmektedir [65-68].

Tezde kullanılacak diğer bir yöntem ise çift yönlü LSTM'ler (BiLSTM'ler) dir. Bu yöntemde BiLSTM'ler giriş verilerini iki defa (yani, önce soldan sağa ve sonra sağdan sola) hareket ettirerek ek eğitim ile başarımı arttırırlar. BiLSTM yönteminin normal LSTM yönteminden daha iyi tahminler sunduğunu bazı çalışmalarda açıkça görülmektedir. Hatta yapılan çalışmada BiLSTM modellerinin hem ARIMA hem de LSTM modellerine göre daha başarılı olduğu görülmektedir. BiLSTM modellerinin çalışma sürelerinin LSTM tabanlı modellere göre çok daha ağır olduğu da görülmektedir [69].

Tez çalışmasında kullanılan modellerden olan LSTM-GRU-SimpleRNN-BiLSTM modellerinin dördü kullanılarak yapılan bazı çalışmalar görülmektedir. Fakat benzer çalışmaların çoğunlukla pandemi, üretim tahmini ve borsa alanlarında yapıldığı gözlemlenmektedir. Yapılan benzer bir çalışmada COVID-19'dan etkilenen on büyük ülkede ölümler ve iyileşmeler üzerine bir çalışma yapılmakta ve RMSE, MAE ve R^2_score indeksleri ile performans ölçümleri yapılmaktadır. Tüm senaryolarda iyi performanstan en düşüğe doğru sıralanan modeller Bi-LSTM, LSTM, GRU, SVR ve ARIMA'dır [70]. Yine [71]'de, LSTM, GRU ve BiLSTM modelleri üzerinden karşılaştırmalı analiz yapılarak bir fiyat tahmin sistemi üretilmeye çalışılmıştır. Hisse senedine ait son 5 günlük işlem verilerinin kullanıldığı BiLSTM modelinde çalışmanın %63,54 lük doğruluk değerine ulaştığı ve en başarılı yöntemin bu olduğu görülmektedir.

Tezde çalışılan veri setiyle aynı veri setini kullanan bir çalışmada verilerin düzenlenmediği görülmekle beraber çok katmanlı ve kompleks bir SimpleRNN ve LSTM mimarisi ile R^2 metriği olarak 0,9814 değeri ile başarılı bir çalışma yapıldığı görülmektedir [72]. Yapılan bu tez çalışmasında tek gizli katmana sahip basit bir model kullanıldığı halde 114 farklı modelin daha başarılı sonuçlar verdiği görülmektedir.

Aynı veri setiyle çalışılan başka bir araştırmada eksik veriler düzenlenmeyip sadece sıralanmakta ve kayıp fonksiyonu olarak mse kullanılmaktadır. Çalışmada en düşük hata oranı olarak RNN ile 0,4, LSTM ile 0,31 ve GRU ile 0,24 değerleri elde edilmektedir [73]. Tezdeki değerlerde ise bu metriğin 0,002 değerlerine kadar düştüğü görülmektedir

Aynı veri setiyle sürdürülen üçüncü bir çalışmada veriler düzenlenerek modeller kayıp fonksiyonu mse, mae ve rmse değerleri üzerinden karşılaştırılmaktadır. Bu çalışmada mse kayıp fonksiyonu en düşük hata oranı mse kayıp fonksiyonu cinsinden 0,103 ile LSTM'ye aittir. Diğer modellerde başarı sırasıyla Çok kanallı, CNN, Doğrusal regresyon, SVR, Karar Ağacı, KARAAĞAÇ ve GRUP modelleriyle elde edilmektedir [74].

RNN'nin bir alt dalı olan LSTM, GRU ve BiLSTM mimarileri kullanılarak yapılan çalışmalarla başarımın arttığı literatürden açıkça görülmektedir. Tamda bu nedenden dolayı bu tezdeki çalışma SimpleRNN, LSTM, GRU ve BiLSTM modelleri üzerinden yürütülmekle birlikte başarımı daha da arttırmak için farklı hiperparametre değerleri denenmektedir. Ek olarak literatürde pek görülmeyen birçok hiperparametre de çalışmalarda denenmektedir.

Hazırlanmış bu tez mevcut literatüre aşağıdaki katkıları sağlamayı amaçlamaktadır.

- Yapay zeka alanında kısa bir derleme yapmak.
- Tahmin alanında geniş bir literatür sunmak.
- Akıllı enerji alanında elektrik tüketim tahmininin önemini vurgulamak.
- Güç tüketimi tahmini için bazı derin öğrenme modellerinin kullanımını göstermek.
- Performans ölçütü olarak kullanılan RMSE, MAE ve R^2 ile istatistiksel metriklerin ana hatlarını çizmek.

3. GENEL KAVRAMLAR

Bu bölümde öncelikle yapay zeka, yapay sinir ağları ve makine öğrenmesi ile ilgili temel düzeyde bilinmesi gereken ifadelere değinilmekte ve açıklanmaktadır. Yapay zeka çağı diye adlandırılan bu yüzyılda yapay zekaya niçin ihtiyaç duyduğumuz veya yapay zekayı insanlardan üstün ve tercih nedeni kılan özelliklere değinilmektedir. Aslında yapay zeka çalışmalarında kullanılan yapay sinir ağlarında insan sinir sistemi örnek alınmaktadır. Bu amaçla gerçek sinir hücrelerinin yapısı, algılayıcı yapısı ve matematiksel modeli şekillerle açıklanmaktadır. Yapay sinir ağları mimari yapısı olan ileri ve geri beslemeli yapay sinir ağı modelleri şekillerle açıklanmaktadır.

Yapay sinir ağlarını aktifleştirmek için bir zorunluluk olan aktivasyon fonksiyonları çok çeşitlilik göstermektedir. Burada en sık tercih edilen adım basamak (step) fonksiyonu, doğrusal (linear) fonksiyon, sigmoid fonksiyonu, hiperbolik tanjant (tanh) fonksiyonu, relu (rectified linear unit) fonksiyonu, sızıntı (leaky) relu fonksiyonu, softmax fonksiyonu, üssel lineer birim (exponential linear unit, elu) aktivasyon fonksiyonu ve swish (a self-gated/kendinden geçitli) fonksiyonu şekillerle açıklanmakta ve avantaj veya dezavantajlı kısımlarına değinilmektedir. Bu 9 aktivasyon fonksiyonundan en çok tercih edilen ve başarı gösteren 6 aktivasyon fonksiyonu olan relu, elu, leakyrelu, tanh, sigmoid ve softmax fonksiyonu çalışma boyunca kullanılmakta ve birbirleriyle kıyaslanmaktadır. Bu kısımda geri yayılım algoritması da açıklanmaktadır.

Sonraki kısımda makine öğrenmesi kısaca anlatılmakta ve makine öğrenmesinin alt birimleri olan 4 ana başlık olan gözetimli öğrenme, gözetimsiz öğrenme, pekiştirmeli öğrenme ve derin öğrenmeye kısaca değinilmektedir.

Bu bölümün son kısmında ise bölümde anlatılanlarla ilişkili olarak makine öğrenmesiyle ilgili basit bir çalışma yapılmaktadır. Burada makine öğrenmesi ile küçük bir veri seti üzerinden yapılan regresyon çalışması sunulmaktadır. Yapılan çalışmalar; veri setinin hazırlanması, program arayüzünün oluşturulması ve yöntemlerin denenmesi olarak gerçekleştirilmektedir. Yöntem olarak; doğrusal regresyon yöntemiyle Diyarbakır nüfus tahmin çalışması yapıldıktan sonra başarılı olunmuş fakat doğrusal regresyon yöntemiyle Diyarbakır elektrik tüketim tahmin çalışmasının yeterli başarı sağlamadığı görülmüştür. Bu nedenle diğer regresyon yöntemleri olan poligonal regresyon yöntemi, destek vektör regresyonu (svr) yöntemi, karar ağacı regresyon yöntemi ve rasgele orman regresyon yöntemiyle Diyarbakır elektrik tüketim tahmin çalışması yapılmıştır. Yapılan çalışma

sonunda Diyarbakır nüfus tahmini sonuçları ve Diyarbakır elektrik tüketim tahmini sonuçları çizelge ve grafiklerle sunulmuştur. R^2 sonuçları da bu kısımda değerlendirilmiştir.

3.1 Yapay Zeka

Teknolojik gelişmelerle beraber günümüzde yapay zekâ, akıllı sistem, robot kelimeleri sıkça kullanılmaktadır. Hatta bu kelimeler Google da en çok arananlar listesinde bulunmaktadır.

Öncelikle zekâ ve akıl kavramlarına bakalım. Zekâ: anlama, yargılama, sonuç çıkarma, zihinsel öğrenme, yeni durumlara uyum sağlama anlamlarına gelmektedir. Akıl ise düşünme, kavrama doğru ile yanlışı ayırt etme yeteneğidir. O yüzdendir ki, çocuklar için "zeki fakat akil-baliğ değil" denmektedir. Çünkü öğrenme yaşanmakta fakat doğru ile yanlışı ayırt etme olayı sağlıklı olarak gerçekleşmemektedir.

Yapay zekâ ise adından da anlaşılacağı üzere insanlar veya bilgisayarlar tarafından oluşturulan veya üretilen belirli bir zekâya sahipmiş gibi davranan teknolojik cihazlar veya programlardır. Diğer bir ifadeyle, bazı canlılara özgü olarak bildiğimiz zekâ kavramının cansız varlık ya da cihazlara uyarlanmasıdır. Cep telefonları, internet reklamları, harita programları, "Siri" programı, "Google asistant" programı, yeni tip televizyonlar, robot kollar ve akıllı denilen sistemler yapay zekâ uygulamalarına örnektir.

Bu uygulamaların her alana uygulanması ve uygulamaya konmasında elbette ki çok kısıtlamalar mevcuttur. Genel itibariyle insan zekâsı gibi kapsamlı ve geniş çalışması şu an itibariyle zor görünmektedir. Fakat özel veya kısmi alanlarda bu uygulamaların insan başarısını geçtiği görülmektedir. Bunun uygulamaları doğada da görülmektedir. Mesela bir kuş, sinek veya böceğin insana yetişmesi eşdeğer olması mümkün değildir. Fakat belli bir alanda veya özel olarak bu canlılar insanları geçerler. Kuşların uçması, arının bal yapması, karıncanın kendisinden defalarca ağır yükleri taşıması gibi. Benzer şekilde yapay zekâ ile cihazlar insanların yapacağı bazı işleri çok daha hızlı ve daha yüksek doğrulukla yapabilirler. Matematiksel hesaplar, harita hesaplamaları, bilgisayar oyunları bunlara örnektir. Sonuçta fabrikalarda bile makinalar başarılarıyla insanların yerine geçmektedir.

Hususi alanlarda yapay zekâyı insanlardan başarılı kılan belli nedenler vardır. Bunlar:

1. İnsanların belli sınırları vardır. Bunlar rekor diye de bilinir. Yiyebilecekleri yemek, kaldırabilecekleri yük, gidebilecekleri yol, çıkabilecekleri hız, zıplayabilecekleri yükseklik vb. güç ve imkânları sınırlıdır. Fakat kullanılan teknolojiler için bu değerler her geçen gün aşılmaktadır.

2. İnsanlardaki hafıza unutmaya maruz kalmakta fakat kullanılan cihazlarda böyle bir engel bulunmamaktadır. İnsanın gördüğü bir resmi daha öncekilerle karşılaştırıp hatırlayıp hatırlamayacağı şüphelidir. Fakat doğru programlanmış bir cihaz bunu tam doğrulukla yerine getirecektir.

3. Özellikle hesaplamalar ve problem çözme konularında insanlar yavaş kalmaktadır.

4. İstem dışı yapılan eylemlerde beyin birçok faaliyeti aynı anda yürütebilir iken istemli olarak iki işi aynı anda yapmakta zorlanmaktadır. İki problemi aynı anda çözme, iki ayrı şeyi aynı anda değerlendirme, dinleme, söyleme gibi. Fakat basit bilgisayarlar bile aynı anda birçok sesi kaydetme, çalma, birçok hesaplamayı yapma özelliklerine sahiptir.

3.2 Yapay Sinir Ağları

Yapay zekâ kavramının ortaya çıkmasıyla birlikte birçok farklı yöntem de beraberinde üretilmiştir. Temel olarak zekâ kavramı kullanıldığı için insan zekâsını oluşturan insan beyni dikkate alınmıştır. İnsan beyni ve beynin bağlantılarını oluşturan sinirler model alınmış ve buna benzetilmeye çalışılan yapay sinir ağları ortaya çıkarılmıştır. Sinir sistemi temelde nöron denilen yapılardan oluşmakta olup yapay sinir ağlarının da nöron adı verilen birimlerden oluştuğu görülmektedir.

Cansız varlıklar dünyaya geldikleri andan itibaren doğada kimyasal özellikleriyle bulunmaktadırlar. Sonradan özellik kazanma öğrenme gibi durumlar söz konusu değildir. Bitkiler için de benzer durum söz konusu olup kabiliyetleri veya yaşamları olan programları çekirdeklerinde mevcuttur. Hareketleri o programa göre şekillenmektedir. Hayvanlar da yüzme, yürüme, koşma, ısırma, yemek yeme, avlanma gibi kabiliyetlerinin çoğunu öğrenerek değil de hazır almış ve öğrenmiş olarak dünyaya gelmektedirler. Diğer canlılara göre hayvanlar için zekâ ve öğrenme kavramları bir nebze gerçekleşmektedir. Fakat insandaki gibi bir öğrenme diğer hiçbir canlıda mevcut olmadığından rol model olarak insan ve öğrenme merkezi olan beyin dikkate alınmıştır. Yapay sinir ağlarının, bu

şekilde insan beynini oluşturan biyolojik sinir ağlarına benzeme ve model almasındaki amaç, insandaki öğrenme kabiliyetini makineler üzerinde gerçekleştirme arzusu ve gerçek hayatta makinelere insanlar tarafından yürütülen görevleri yükleyebilme isteğidir. Çünkü görülen canlı cansız varlıklar içinde öğrenme yeteneği mevcut ve en üst düzeyde olan insandır. Yapay sinir ağları bu niteliğiyle görüntü ve ses tanıma, öğrenme gibi sınıflandırma alanlarında başarı göstermekte bu nedenle de gün geçtikçe daha popüler hale gelmektedir.

İlk yapay nöron, nöro psikiyatrist Warren McCulloch ve bilim adamı Walter Pits tarafından 1943 yılında üretilmiştir. Ancak dönemin dar imkânları sebebiyle, bu alanda pek ilerleme kaydedilememiştir. Daha sonra Minsky ve Papert 1969'da bir kitap çıkarmışlar, yapay sinir ağları konusundaki etik tedirginlikleri gidermiş ve bu teknolojiyi tekrar gündeme getirmişlerdir. Fakat bu konuda kayda değer gelişmeler 1990 yıllarından sonra ortaya çıkmıştır [75].

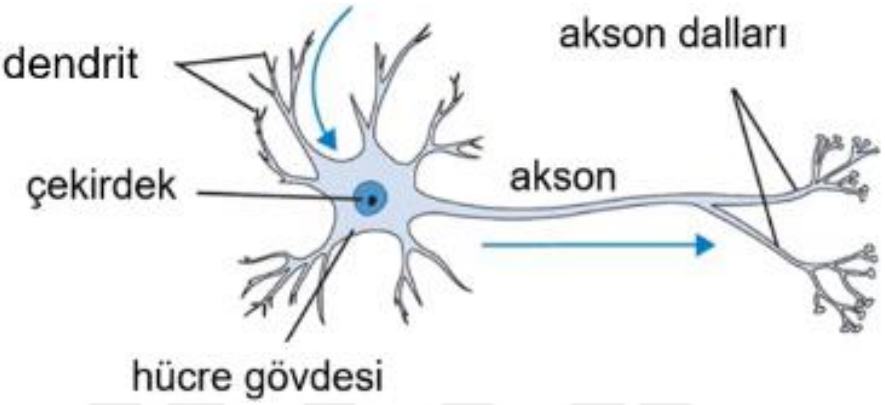
Genel olarak YSA, beyinden esinlenerek oluşturulmuş ve beynin bir fonksiyonu yerine getirme davranışını modellemek üzere geliştirilmiş bir sistem olarak ifade edilebilir. YSA, yapay sinir hücrelerinin aralarında farklı şekillerde bağlanıp irtibat kurmasıyla oluşur. Ayrıca YSA, çoğunlukla katmanlar şeklinde oluşturulur. Elektronik devrelerle donanım veya bilgisayarlarda yazılım olarak oluşturulabilir. Beynin bilgiyi işlemesine benzer olarak YSA, bir öğrenme deneyiminden sonra bilgiyi saklama ve genelleştirme yeteneğine haiz paralel dağılmış bir işlemcidir [76]. Temeli Turing makineleriyle atılmıştır. Yapay zekâ alanında en çok araştırılan başlık "Yapay Sinir Ağları"'dır. Yapay sinir ağları, tamamen insan beyni temel alınarak ve örneklenerek geliştirilmiş bir sistemdir. Bir sinir ağı, bilgiyi kaydetmek ve onu işlevsel hale getirmek için basit kısımlardan oluşan paralel dağıtılmış bir işlemcidir. İnsan beynine iki yönüyle benzemektedir: 1.Bilgi, öğrenme süreci yoluyla ağ üzerinden temin edilir. 2. Sinaptik ağırlık denilen nöronlar arasındaki bağlantı kuvvetlerini, bilgiyi depolamak amacıyla kullanır [77].

Yapay sinir ağları çok fazla alanda kullanılmakla beraber daha çok sınıflandırma, modelleme ve tahmin alanlarında kullanılmaktadır. Başarılı olmuş uygulamalar incelendiğinde, YSA'ların karmaşık, çok boyutlu, gürültülü, kesin olmayan, eksik, kusurlu, hata olasılığı yüksek sensör verilerinin olması ve problemi çözmek için algoritmaların ve matematiksel modelin bulunmadığı, yalnızca örneklerin mevcut olduğu şartlarda çokça tercih edildikleri ortaya çıkmaktadır. Bu maksatla oluşturulan ağlar fonksiyonların birçoğunu gerçekleştirmektedirler. Bu ağlardan olan ilişkilendirme veya örüntü eşleştirme,

sınıflandırma, muhtemel fonksiyon kestirimleri, zaman serileri analizleri, örüntü tanıma, veri sıkıştırma, doğrusal olmayan sinyal işleme, sinyal filtreleme, doğrusal olmayan sistem modelleme, optimizasyon ve kontrol olmak üzere YSA'lar pek çok sektörde farklı uygulama alanlarında kullanılmaktadır [78].

3.2.1 Yapay Sinir Ağı Yapısı

İnsandaki bir sinir hücresinin (nöron) yapısı Şekil 3.1'de gösterilmektedir.

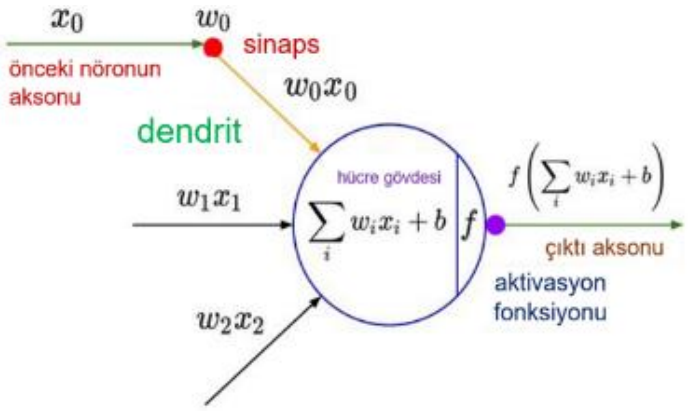


Şekil 3.1: Bir sinir hücresinin biyolojik gösterimi. [79]

Şekil 3.1 ve şekil 3.2'de verilen bazı ifadeler ve sinir hücresi ile ilgili temel terimlerin kısaca açıklamaları burada verilmiştir. Bunlar: Akson (Axon): Çıkış darbelerinin üretildiği elektriksel aktif gövdedir. İletim gövde boyunca tek yönlüdür. Sistemin çıkışıdır. Dentritler (Dendrites): Elektriksel anlamda pasif olan ve diğer hücrelerden gelen işaretleri toplayan kollardır. Sistemin girişidir. Sinaps (Synapse): Hücrelerin aksonlarının diğer dentritlerle irtibatını kurar. Miyelin Tabaka (Myelin Sheath): Yayılım hızını etkileyen yalıtım malzemesidir. Çekirdek (Nucleus): Periyodik olarak işaretlerin akson boyunca üretilmesini sağlar. Aksonda taşınan işaret sinapslara kimyasal taşıyıcılar vasıtasıyla ulaştırılmaktadır. Stoplazma -85 mV ile polarizedir. -40 mV (Na+ içeri): uyarma (+) akıma sebep olur. -90 mV (K+ dışarı): bastırma (-) akıma sebep olur. Yani voltaj değeri bir sınır değeri aşınca hücre uyarılır, diğer koşullarda bastırılır. Σ Sinirsel hesaplama ise bu duruma göre çıkış işareti üretilmesidir [80].

3.2.2 Algılayıcı

Matematiksel olarak bir sinir hücresinin ifade tarzı aşağıda Şekil 3.2'de görülmektedir.



Şekil 3.2: Bir sinir hücresinin matematiksel modeli [79]

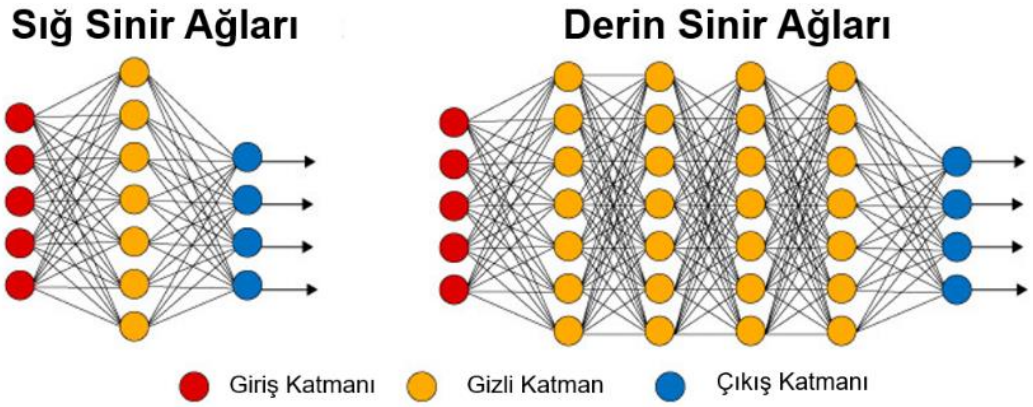
Algılayıcı (perceptron) yapay sinir ağının en küçük parçasıdır. İlk olarak 1957 yılında Frank Rosenblatt tarafından tanımlanmıştır. Doğrusal bir fonksiyon olarak aşağıdaki gibi gösterilmektedir.

$$y = wx + b \tag{3.1}$$

Burada y: x'e bağlıdır ve bağımlı değişken olarak nitelendirilmektedir. Girdiye ait sonucu verir. x: bağımsız değişken veya girdi olarak ifade edilir. W: ağırlık parametresidir. b: bias değeridir. YSA ya da Derin Öğrenme çalışmalarında yapılan ana işlem; yukarıdaki formüle göre çalışmanın en iyi sonucu üreteceği W ve b değerlerini hesaplamaktır [80].

3.2.3 Yapay Sinir Ağlarının Mimari Yapısı

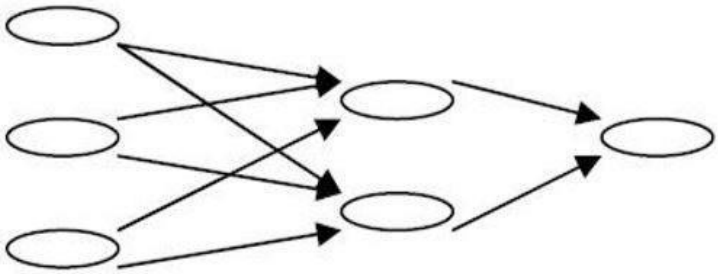
Aşağıdaki Şekil 3.3'te girdi katman, gizli katman ve çıktı katmanından oluşan 3 katmanlı ve tek bir gizli katmana sahip sinir ağı modeli görülmektedir. Hemen yanında da girdi katman, iki adet gizli katman ve çıktı katmanından oluşan 4 tabakalı ileri beslemeli çok katmanlı bir sinir ağı modeli görülmektedir.



Şekil 3.3: Tek gizli katmanlı ve birden çok gizli katmanlı sinir ağı yapısı. [79]

3.2.3.1 İleri beslemeli YSA modeli

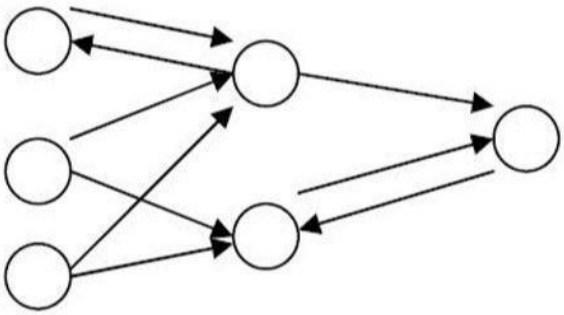
Şekil 3.4'te görüldüğü üzere tek yönlü sinyal akışına izin verir. Bu yapıda hücreler katmanlar halinde dizayn edilmektedir. Ayrıca herhangi bir katmana ait hücrelerin çıkışları ağırlıklar üzerinden daha sonra gelen katmana giriş olarak verilir. Giriş katmanı, dış ortamlardan aldığı bilgileri herhangi bir değişiklik yapmadan ara katmandaki diğer hücrelere ulaştırır. Gizli katmanlarda ve çıktı katmanında bilginin işlenmesi ile çıkış değeri belirlenir [81].



Şekil 3.4: İleri beslemeli sinir ağı yapısı [81].

3.2.3.2 Geri beslemeli YSA modeli

Bu yapıda en az bir hücrenin çıkışı kendisine ya da diğer hücrelere giriş olarak verilir ve geri besleme genellikle bir geciktirme elemanı vasıtasıyla yerine getirilir. Geri besleme, bir katmanda bulunan hücreler arasında olabileceği gibi katmanlar arasındaki hücreler arasında da olabilir. Bu çeşit sinir ağları dinamik hafızaya sahiptirler. Böyle bir ağda nöronların çıkış değerleri yalnız o anki giriş değerleriyle değil önceki giriş değerleriyle de ilgilidir. Bu nedenle, bu ağ modeli özelikle tahmin çalışmaları için uygun görülmektedir. Şekil 3.5'te geri beslemeli ağ modeli görülmektedir. Bu ağ modeli özellikle farklı tipteki zaman serilerinin tahmininde fazlasıyla başarılı olmaktadır [82].



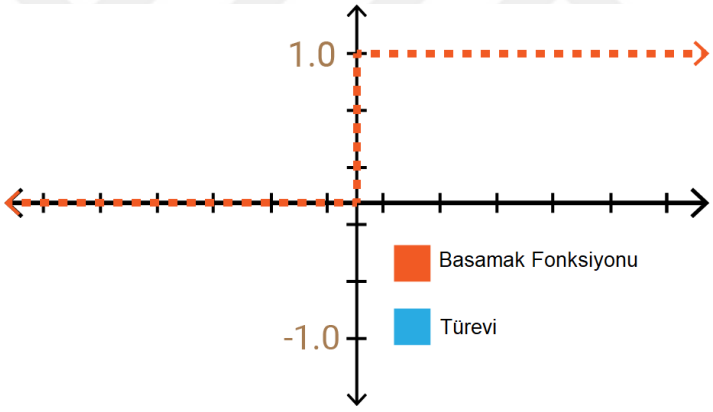
Şekil 3.5: Geri beslemeli yapay sinir ağı yapısı [81].

3.2.4 Yapay Sinir Ağları İçin Aktivasyon Fonksiyonları

Doğal yaşamda karşılaşılan problemlerin çoğu doğrusal değildir. Bu durumda da modellenecek yapay sinir ağları için doğrusal fonksiyonlar yetersiz kalmaktadır. Bu yüzden aktivasyon fonksiyonlarına ihtiyaç duyulmaktadır. Basitçe yapay bir sinir ağında girdiler x, ağırlıklar ise w şeklinde tanımlanmaktadır. Bu durumda ağın çıkışına aktarılacak değere f(x) yani aktivasyon işlemi uygularız. Bu da nihai çıkış olacak veya başka bir katmana giriş olarak uygulanacaktır. Burada önemli aktivasyon fonksiyonlarını sıralayacak olursak:

3.2.4.1 Adım basamak (Step) fonksiyonu

İkili değer alan bir fonksiyondur ve ikili sınıflayıcı olarak kullanılmaktadır. Bu nedenle çoğunlukla çıkış katmanlarında kullanılmaktadır. Gizli katmanlarda türevi öğrenme değeri temsil etmediği için tercih edilmemektedir. O zaman da türevlenebilir bir fonksiyon olarak doğrusal (linear) fonksiyon düşünülmektedir [83].

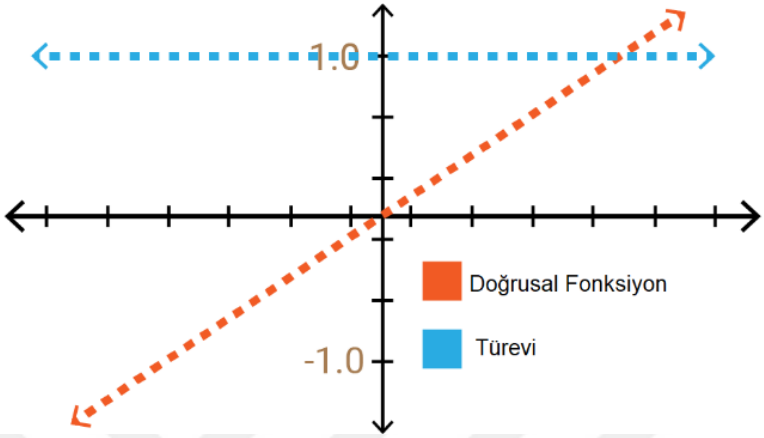


Şekil 3.6: Basamak fonksiyonu ve türevi [83].

3.2.4.2 Doğrusal (Linear) fonksiyon

Bir dizi aktivasyon değeri üretmektedir ve bunlar ikili değerlerden oluşmamaktadır. Kesinlikle birkaç nöronu (sinir hücresi) birbirine bağlamaya izin verir. Burada bu fonksiyonun türevine ihtiyaç duyulmakta fakat türevin sabit olması bizim için sorun teşkil etmektedir. Çünkü nöronlar için geriye yayılım (backpropagation) ile öğrenme olayının gerçekleştirilmesi gerekmektedir. Bu geriye yayılım algoritması ise türev alan bir sistemden oluşmaktadır. A=c.x formülü üzerinden x'i baz alarak türev hesaplandığında c değerine erişilmektedir. Eğer türevi alınmaz ise bu durum x ile bir ilişkinin kalmadığı göstermektedir. Türev sonucunun sürekli sabit bir değer çıkması durumunda ise öğrenme olayı gerçekleşmemektedir [83].

Ayrıca katmanların hepsinde doğrusal fonksiyon kullanıldığında giriş ve çıkış katmanı arasında sürekli aynı doğrusal sonuç elde edilmektedir. Doğrusal fonksiyonların doğrusal bir şekilde birleşiminden yine farklı bir doğrusal fonksiyon elde edilmektedir. Bu ise daha başlangıçta birbirine bağlanacak nöronların işlevsiz kalmasına neden olmaktadır. Böylece çok katmanda çalışma imkânı kalmayacaktır [83].

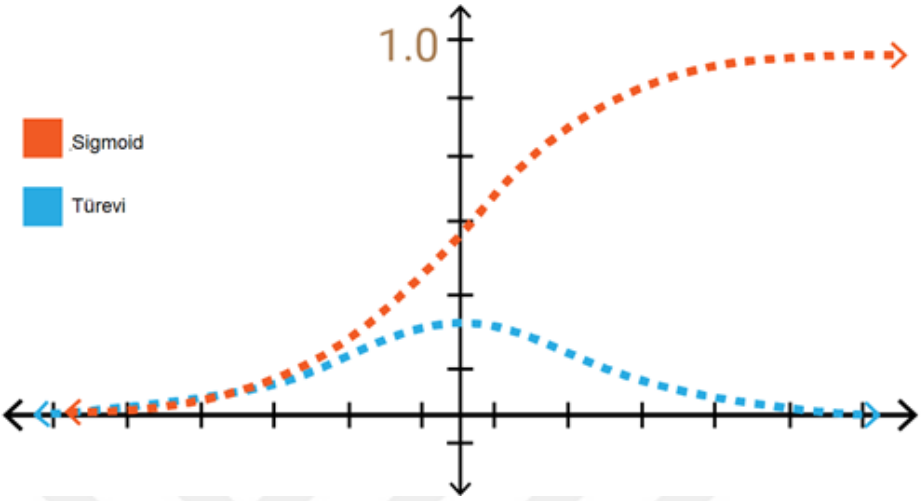


Şekil 3.7: Doğrusal fonksiyon ve türevi [83]

3.2.4.3 Sigmoid fonksiyonu

Çevredeki çoğu problemler doğrusal değildir. Bu ise doğrusal olmayan sigmoid fonksiyonunun kombinasyonları yardımıyla katmanların sıralanabileceğini göstermektedir. İkili olmayan fonksiyonları değerlendirildiğinde ise yine basamak fonksiyonundan farklı olduğu için türevlenebilirdir. Bu durum da öğrenme olayının gerçekleşebileceğini göstermektedir. Şekil 3.7'ye bakıldığında x, -2 ile +2 değer aralığında bulunurken y'nin hızla değiştiği gözlemlenmektedir. x'teki küçük oynamalar y'de büyük değişimlere neden olmaktadır. Bu durum iyi bir sınıflayıcı olduğunu gösterir. Bu fonksiyonun başka bir avantajı da doğrusal fonksiyonlarda gerçekleştiği gibi (-sonsuz, +sonsuz) ile karşılaşılması durumunda hep (0,1) arasında değerler üretmesidir. Böylece aktivasyon değeri aşırı değerler almamaktadır. Sigmoid fonksiyonu aktivasyon fonksiyonları arasında en çok tercih edilenlerden sayılmakla beraber farklı ve daha verimli seçenekleri de bulunmaktadır. Buna ihtiyaç duyulmasının nedeni ise fonksiyona at grafiğin uç kısımlarına dikkatlice bakıldığı zaman görülen, y değerlerinin x'teki değişimlere nazaran çok az tepki vermesidir. Bu durum da bu bölgelerde türev değerlerinin çok küçük olmasına neden olmakta ve 0'a yakınsamaktadır. Bu durum gradyanların ölmesi/kaybolması (vanishing gradient) olarak adlandırılmakta ve bu durumda öğrenme olayı minimum seviyede gerçekleşmektedir. Türev 0 olursa öğrenme gerçekleşmeyecektir. Yavaş bir öğrenme olayı ile karşılaşıldığında

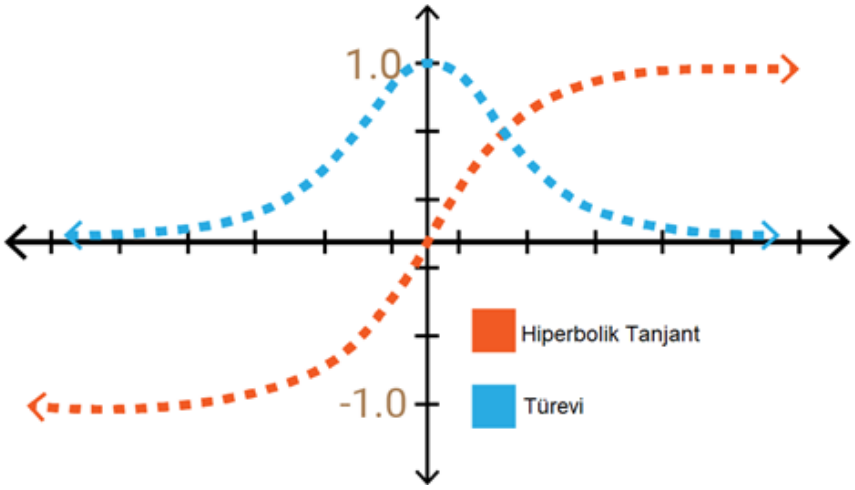
hatayı minimize etmek amacıyla kullanılan optimizasyon algoritması yerel (lokal) minimum değerlere takılabilmekte ve yapay sinir ağı modelinden alınabilecek maksimum performansı alınamamaktadır. Bu nedenle de bu soruna alternatif olabilecek aktivasyon fonksiyonlarını da incelemek gerekmektedir [83].



Şekil 3.8: Sigmoid fonksiyonu ve türevi [83]

3.2.4.4 Hiperbolik tanjant fonksiyonu

Sigmoid fonksiyonuna çok benzeyen bir fonksiyondur. Fakat fonksiyon değerleri bu defa (-1, +1) aralığında tanımlanmaktadır. Sigmoid fonksiyonuna göre türevinin daha dik olması avantajdır. Yani daha çok değer alabilmektedir. Böylece sınıflandırma ve hızlı öğrenme işlemleri için daha geniş aralığa sahip olacağından daha verimli olacaktır. Fakat fonksiyon uçlarına doğru gradyanların ölmesi sorununun devam ettiği görülmektedir. Sıklıkla karşılaşılan bir aktivasyon fonksiyonu olmasına rağmen daha iyisi için arayışların devam etmesi gerekmektedir [83].

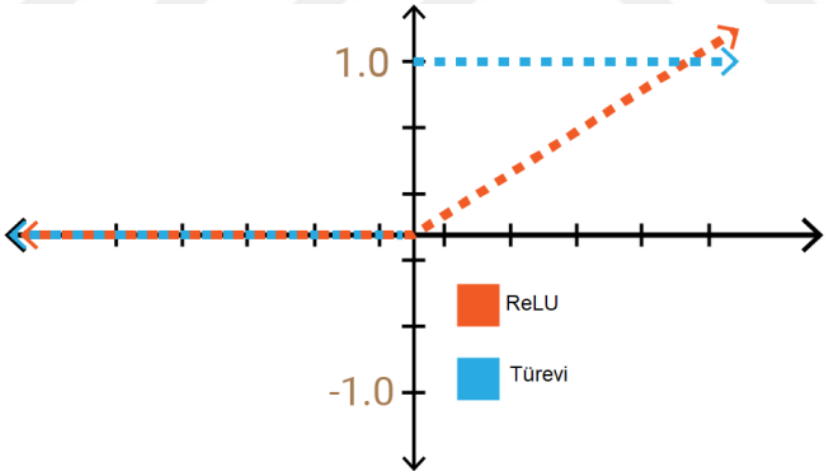


Şekil 3.9: Hiperbolik tanjant fonksiyonu ve türevi [83]

3.2.4.5 Relu (Rectified linear unit) fonksiyonu

Bu fonksiyon her ne kadar pozitif eksende doğrusal fonksiyona benzer gibi görünse de aslında ReLU fonksiyonu doğası gereği doğrusal değildir. İyi bir tahmin edicidir. ReLU kombinasyonları ile farklı bir fonksiyona yakınsamak mümkün olduğundan yapay sinir ağının katmanlarını sıralamak mümkün olmaktadır [83].

ReLU [0, +∞) arası değerler almaktadır. Hiperbolik tanjant ve Sigmoid çok fazla nörona sahip büyük sinir ağlarında bütün nöronların aynı şekilde aktifleşmesine neden olmaktadır. Bu durum da çok fazla işlem gerektirmektedir. Oysa ağdaki bazı nöronların aktif olması, aktivasyonun seyrek olmasına yani verimli bir hesaplama yükü meydana getirmesine neden olacaktır. İşte ReLU fonksiyonu [0, +∞) aralığında değerler alarak bunu sağlamaktadır. Negatif eksende 0 değerini alması ağın daha hızlı çalışacağı sonucunu da vermektedir. Böylece hesaplama yükü sigmoid ve hiperbolik tanjant fonksiyonlarından daha az olmakta ve çok katmanlı ağlar için daha fazla kullanılmasına neden olmaktadır. Ancak ReLU fonksiyonunun da eksik kısmı işlemleri hızlandıran sıfır değer bölgesinin türevinin de sıfır olmasıdır. Dolayısıyla o bölgede öğrenme gerçekleşmemektedir. Bu yüzden daha iyi bir aktivasyon fonksiyonu bulunmalıdır [83].

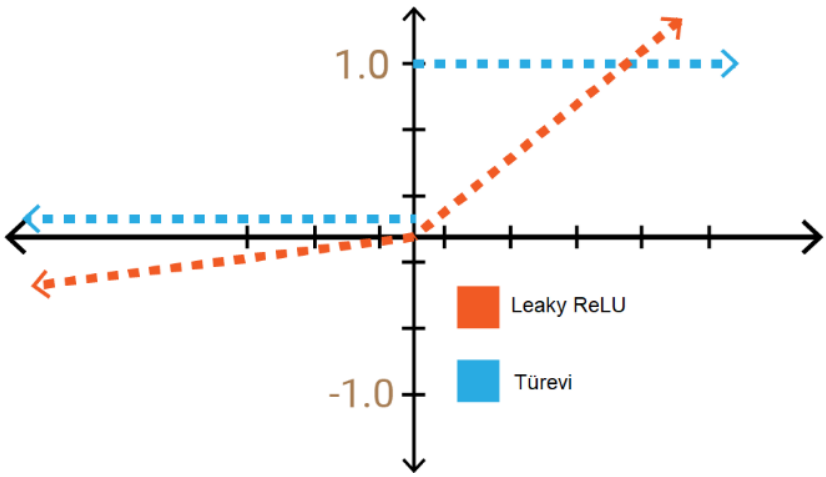


Şekil 3.10: Relu fonksiyonu ve türevi [83]

3.2.4.6 Sızıntı (Leaky) relu fonksiyonu

Şekilde negatif düzlemde fonksiyonun sızıntı kısmını görülmektedir. Burada sızıntı değeri olarak 0,01 verilmektedir. Sıfıra yakın farklı bir değer verilmesi durumunda fonksiyonun adı rastgele Leaky ReLU olarak değişir. Sızdırılan ReLU'nun tanım aralığı eksi sonsuza kadar uzanır. 0'a yakın ama 0 olmayan bu değer sayesinde ReLU'daki ölen

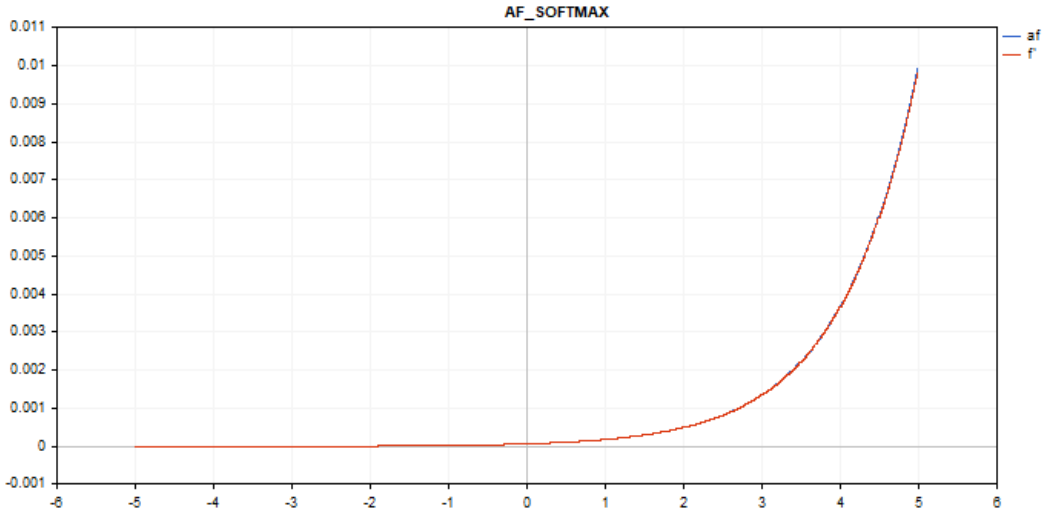
gradyanlar sorunu çözülmüş, yani negatif bölge için de öğrenme sağlanmış olmaktadır [83].



Şekil 3.11: Leaky relu fonksiyonu ve türevi [83]

3.2.4.7 Softmax fonksiyonu

Sigmoid fonksiyonuna çok benzer bir fonksiyondur. Sınıflayıcı olarak kullanıldığı zaman Sigmoid gibi başarılı sonuçlar verir. Sigmoid gibi ikiden fazla sınıflandırma gereken durumlarda kullanılmaktadır. Derin öğrenme modellerinde çıkış katmanında sıklıkla kullanılmaktadır. Girdinin hangi sınıfa ait olduğunu 0-1 aralığında değerler üreterek belirler ve böylece olasılıksal bir yorumlama gerçekleştirir [83].

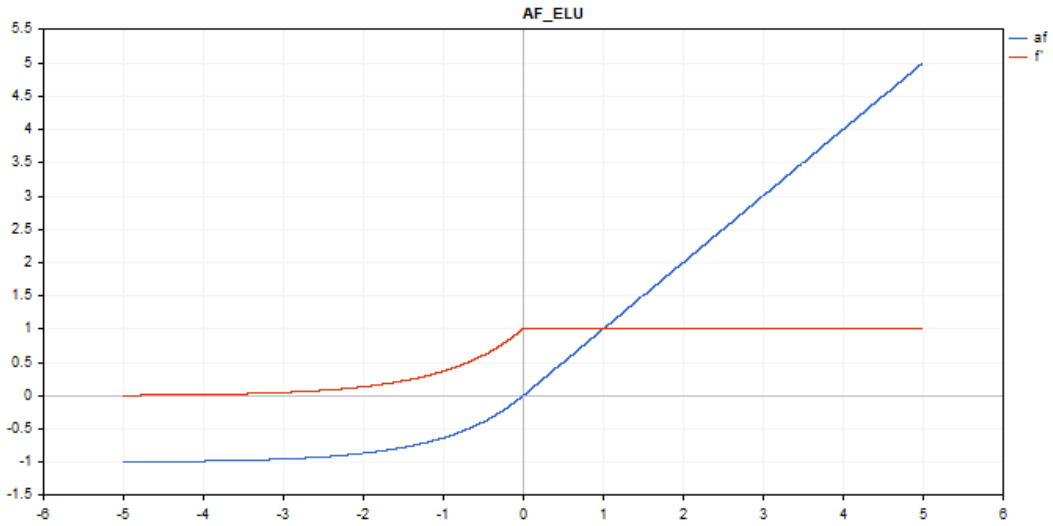


Şekil 3.12: Softmax fonksiyonu [84]

3.2.4.8 Üssel lineer birim (Exponential Linear Unit, ELU) aktivasyon fonksiyonu

Exponential Linear Unit (ELU) aktivasyon fonksiyonu, 2016 yılında Djork-Arne Clevert tarafından üretilmiştir. ELU, çoğunlukla derin öğrenme çalışmalarında tercih

edilen bir fonksiyondur. ELU diğer aktivasyon fonksiyonlarına nazaran daha geniş bir öğrenme aralığına sahiptir. Bununla birlikte ölü nöron gibi bazı sorunlara karşı daha dirençlidir [85].



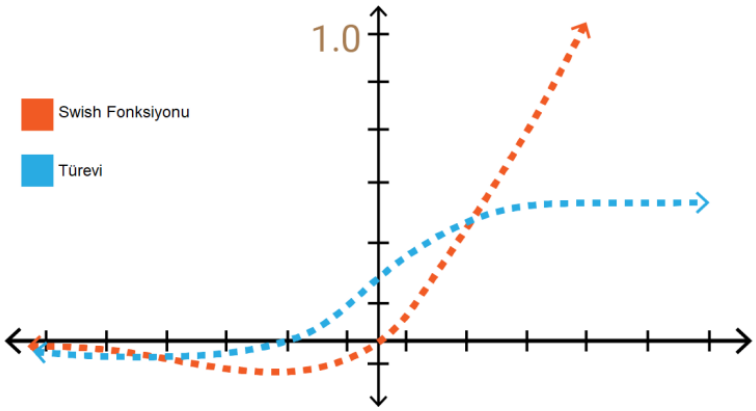
Şekil 3.13: Elu fonksiyonu ve türevi [84]

3.2.4.9 Swish (A self-gated/kendinden geçitli) fonksiyonu

ReLU'ya göre farkı negatif bölgede değer almasıdır. Leaky ReLU'ya göre farkı ise negatif bölgede aldığı değerlerin doğrusal olmamasıdır. Diğer tüm aktivasyon fonksiyonları monotondur. Swish fonksiyonu için girdi değeri arttığında bile çıktı değerinin düşebileceği görülmektedir. Bu özellik swish'e has bir yetenektir [83].

$$f(x) = 2x * \text{sigmoid}(\text{be ta} * x) \tag{3.2}$$

Üstte verilen 3.2'deki tanımda bulunan beta'nın öğrenilebilir bir değişken olduğu değerlendirildiğinde, eğer beta=0 ise sigmoid kısmı her zaman 1/2 olur ve f(x) → doğrusal olur. Eğer beta için çok büyük bir değer alınırsa sigmoid neredeyse ikili bir basamak fonksiyonuna dönüşür (0 için x<0,1 için x>0). Böylece f(x) ReLU fonksiyonuna yakınsar. Bu yüzden standart Swish fonksiyonunda beta=1 alınır. Bu şekilde yumuşak bir enterpolasyon (değişken değer setlerini verilen aralıkta ve istenilen hassasiyette bir fonksiyon ile ilintilendirme) sağlanmış olur. Böylece gradyanların ölmesi problemi de çözülmüş olur [83].



Şekil 3.14: Swish fonksiyonu ve türevi [83]

3.2.5 Geri Yayılım Algoritması

Geriye yayılma algoritması, çok katmanlı, ileri beslemeli yapay sinir ağlarını eğitmek için kullanılan bir denetimli öğrenme yöntemidir. Girdilerin ve çıktıların işlenmesini ve ağırlıkların buna göre güncellenmesini içerir. Ağ eğitim sürecinin bir parçasıdır. Ana işlevi, hata fonksiyonundan gelen geri bildirimi kullanarak tüm ağ ağırlıklarını paralel olarak güncellemektir. Bu, toplam hata değerinin en aza indirilmesini sağlar [86].

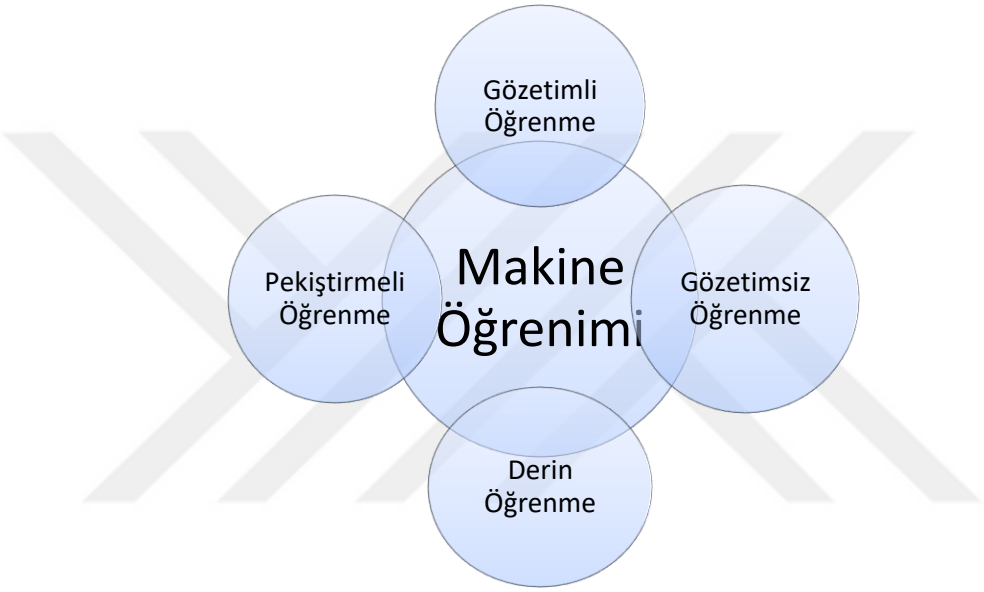
3.3 Makine Öğrenmesi

Makine öğrenimi, nispeten zeki sonuçlar üretebilen bir yapay zeka dalıdır. Bilgisayara verilen verilere bağlı olarak bilgisayara bir karar verme kabiliyeti kazandırmaya makine öğrenmesi denilebilir. Kavram, 1959 yılında Arthur Samuel tarafından ortaya atılmış ve Samuel tarafından mevcut sınırları aşarak belirli bir görevi yerine getirme yeteneği olarak ifade edilmiştir. Samuel'in dama algoritması, hatalarından ders çıkarıp, öğrenen bir makine öğrenimi algoritmasıdır. Makine öğrenimi, kendisine tarif edilen veya tanımlanan çok sayıdaki kedi köpek resimleri üzerinden eğitilen cihazın sonradan verilen hiç görmediği bir resimdeki kedi veya köpeği tahminleme yetisidir. Makine öğrenimi algoritmaları belirli bir eğitime tabi tutulmaktadır. Sonrasında ise resimlerdeki desen ve pikseller üzerinden sonuçlar üretilebilmektedir.

Makine öğrenimini bilgi edinme, fonksiyonel regresyon, kümeleme, nesne algılama ve örüntü tanıma gibi konulardan oluşmaktadır. Nesne algılama (Object detection) bilgisayarlı görme ve ayırt etme işlemlerinden oluşur. Bilgi edinme (Knowledge acquisition) bilgi merkezli bir küme için gerekli kuralları ve varlıkları ifade eder.

Fonksiyonel regresyon (Functional regression) regresyon analizinin cevap ya da değişkenlerin fonksiyonel veri içeren çeşitleridir. Kümeleme (Clustering) verileri anlamlı parçalara ayırır. Örüntü tanıma (Pattern recognition) devamlılık arz eden durumlarda tekrarlayan nesneleri ayıklamadır [87].

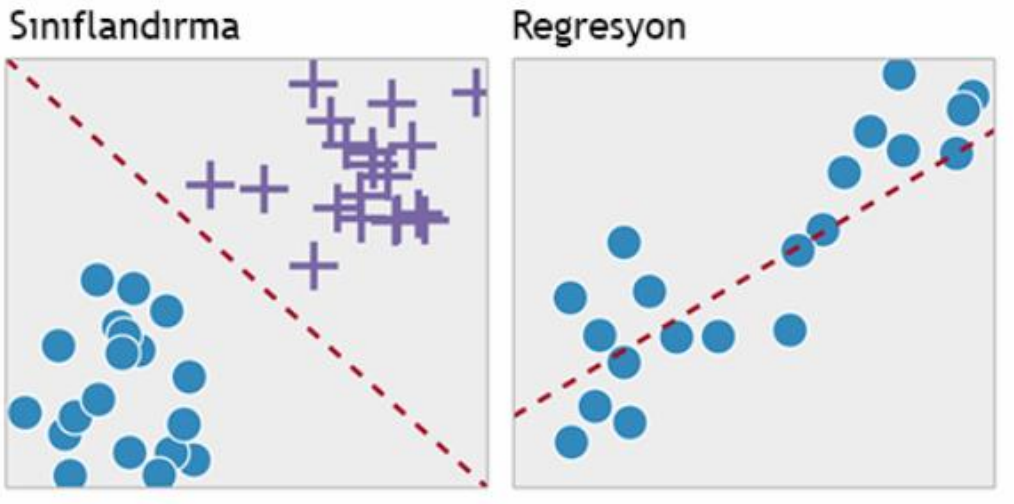
Makine öğreniminde etiketlenmiş veri veya etiketlenmemiş veri gibi veri tiplerine, çalışılan alana, verinin boyutuna ve veri büyüklüklerine bağlı olarak farklı yaklaşımlar kullanılmaktadır. Bu yaklaşımları Şekil 3.15'deki gibi gözetimli öğrenme, gözetimsiz öğrenme, pekiştirmeli öğrenme ve derin öğrenme olarak sıralamak mümkündür.



Şekil 3.15: Makina öğreniminin alt konuları.

3.3.1 Gözetimli Öğrenme

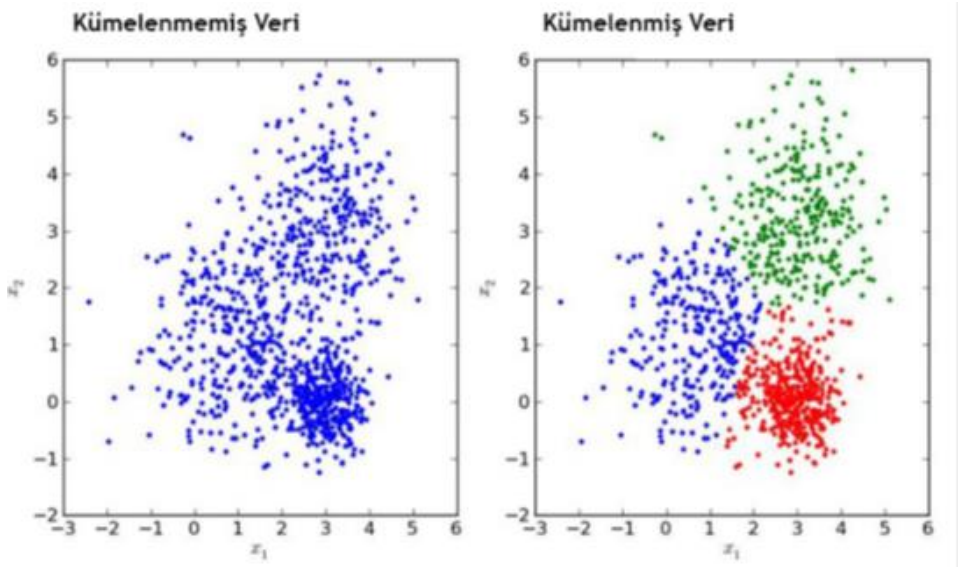
Gözetimli öğrenmede (Supervised Learning) eldeki verilerde hem giriş hem de çıkış değerleri mevcuttur. Başka bir deyişle veriler etiketlenmiştir. Yani hangi veriden (x değerinden) hangi sonuca (y değerine) ulaşılacağı bellidir. Amaç giriş verileri üzerinden çıkış değerlerine ulaşabilecek bir fonksiyon üretmektir. Bu çalışma şeklinde üretilecek modeller ile sınıflandırma ve regresyon algoritmaları üretmek ve bu tarz problemlere çözüm üretmek mümkündür.



Şekil 3.16: Gözetimli öğrenme (sınıflandırma ve regresyon) [87]

3.3.2 Gözetimsiz Öğrenme

Gözetimsiz öğrenmede (Unsupervised Learning) eldeki verilerde sadece giriş değerleri mevcuttur. Başka bir deyişle veriler etiketlenmemiştir. Yani hangi veriden (x değerinden) hangi sonuca (y değerine) ulaşılacağı belirtilmemiştir. Bu yüzden bu tür verilerde giriş verileri üzerinden çıkış değerlerine ulaşılması mümkün görülmemektedir. Ancak bu tür verileri kendi aralarında kümelemek ve yeni gelecek bir değerin hangi kümeye ait olacağını tahminlemek ve bu tarz problemlere çözüm üretmek mümkündür.



Şekil 3.17: Gözetimsiz öğrenme (kümelenmemiş ve kümelenmiş veri) [87]

3.3.3 Pekiştirmeli Öğrenme

Takviyeli öğrenme de denilen pekiştirmeli öğrenmede (Reinforcement Learning) model veri çözümlemeleri sonucunda geri beslemeler alır ve bu şekilde en iyi sonuca

ulaşmayı amaçlar. Modelin geri beslenmesi ile aldığı tepkiler sayısal ödül sinyalleridir. Burada modelin bu ödülü maksimum seviyeye çıkarmaya çalışır. Bu yaklaşım birçok kaynakta deneme-yanılma şeklinde de ifade edilir [87].

3.3.4 Derin Öğrenme

Derin öğrenme çok daha karışık sorunlara çözüm üretmeye çalışan ve insan beyin ve sinir yapısını taklide çalışan bir öğrenme tarzıdır. Bu yöntemde daha çok veriyle çalışılır ve etiketleme yapılmadan öznitelikleri cihazın kendisinin çıkarması beklenir. Son yıllarda kullanım miktarı ve problemlere çözüm üretme başarısı dikkat çekmektedir. Bu tez çalışmasının ana teması derin öğrenme olması nedeniyle derin öğrenme konusu bir sonraki bölümde detaylıca incelenecektir.

3.4 Makine Öğrenmesi İle Küçük Bir Veri Seti Üzerinden Yapılan Regresyon Çalışması

Makine öğrenmede farklı problemlerin çözümü amacıyla tasarlanmış farklı mimariler mevcuttur. Bu mimarilerin en bilinenlerini doğrusal regresyon, polinomal regresyon, destek vektör regresyonu, karar ağaçları ve rasgele orman halinde sıralanabilir. Bu çalışmada, Diyarbakır için geçmiş nüfus ve elektrik tüketim bilgileri kullanılmıştır. Diyarbakır nüfus verileri üzerinden veriler üç kısım olacak şekilde birinci kısmı eğitim sürecine tabi tutulmakta, ikinci kısmı eğitilen bu modellerin doğrulanması için kullanılmakta, üçüncü kısım ise test için kullanılmaktadır. Bu şekilde, eğitim seti ile modeller eğitilecek, eğitilen modellerin doğrulama veri seti üzerinde performansı gözlemlenecek, sonrasında ise bu iki grup dışında kalan test verileri için tahmin yapılacaktır. Diyarbakır elektrik tüketim tahmini için verilen verilerin tümü kullanılarak güncel değerlere yönelik tahminler yapılacak ve değerlendirilecektir. Ayrıca yine gelecek yıllar için de elektrik tüketim tahmini yapılacaktır. Yöntem olarak ise makine öğrenmesi yöntemlerinden doğrusal regresyon, polinom regresyonu, destek vektör regresyonunu (SVR), karar ağaçları ve rasgele orman kullanılacaktır.

3.4.1 Veri Setininin Hazırlanması

Veriler farklı kaynaklardan toplanmıştır. Diyarbakır nüfus verileri Türkiye İstatistik Kurumu (TUİK) verilerinden ve internetten elde edilmiştir. Diyarbakır elektrik tüketim verileri ise Enerji Piyasası Denetleme Kurulu (EPDK) ve Türkiye Elektrik Dağıtım

Anonim Şirketi (TEDAŞ) raporları ile Karacadağ Kalkınma Ajansı dökümanlarından toparlanmıştır. Diyarbakır elektrik tüketim verileri toplam tüketilen değil toplam faturalandırılan elektrik tüketim verileridir. Çalışma için kaynak oluşturan veriler Çizelge 3.1'de verilmiştir.

Çizelge 3.1 Diyarbakır nüfus ve elektrik tüketim değerleri. [88-92]

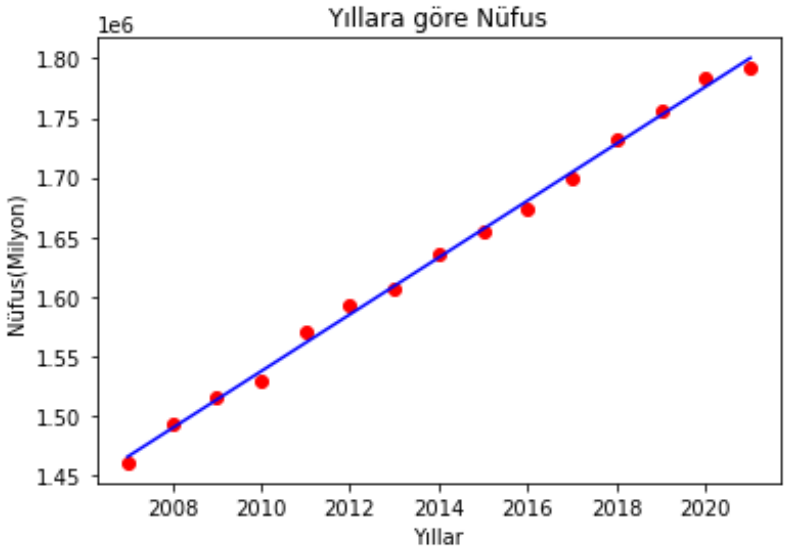
Yıl	Diyarbakır Nüfusu (Kişi)	Diyarbakır Elektrik Tüketimi (MWh)
2007	1.460.714	1.167.058
2008	1.492.828	1.248.844
2009	1.515.011	1.115.711
2010	1.528.958	1.195.375
2011	1.570.943	1.282.682
2012	1.592.167	1.239.729
2013	1.607.437	1.285.746
2014	1.635.048	1.482.825
2015	1.654.196	1.734.727
2016	1.673.119	1.985.576
2017	1.699.901	2.266.210
2018	1.732.396	2.752.770
2019	1.756.353	2.596.431
2020	1.783.431	2.858.333
2021	1.791.373	3.106.284

3.4.2 Program Arayüzünün Oluşturulması

Öncelikle bu bölümdeki çalışmalarda Anaconda bünyesindeki Spyder idesi üzerinden Python programı kullanılmıştır. Çalışma kısmındaki tüm grafikler Python programı ile çalışma sonucu kısmındaki grafikler ise Microsoft Excel programı ile çizdirilmiştir. Ayrıca bazı çalışmalar ve tablolarda da Microsoft Excel programından faydalanılmıştır. R^2 hesaplamaları Microsoft Excel programında hazırlanan tablolar yardımıyla hesaplanmıştır.

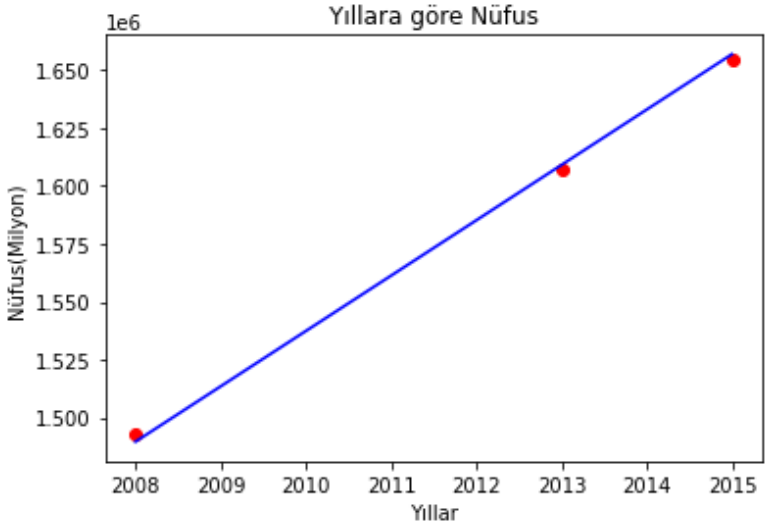
3.4.3 Doğrusal Regresyon Yöntemiyle Diyarbakır Nüfus Tahmin Çalışması

Şekil 3.18'da Çizelge 3.1'deki mevcut verilere göre yıllara göre nüfus verileri kırmızı noktalarla belirtilmekte ve doğrusal regresyon yöntemi ile hesaplanan nüfus tahmini mavi çizgi ile belirtilmektedir.



Şekil 3.18: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon çizgisi (mavi çizgi)

Doğrusal regresyon yönteminin verilerimiz üzerinde başarılı olduğu görüldüğünden mevcut verilerimiz %80 eğitim (train) ve %20 test olmak üzere iki kısma ayrılmıştır. Şekil 3.19'da uygulanan doğrusal regresyon yöntemine göre x_test değerleri için verilen y_test gerçek değerleri (kırmızı noktalar) ve y_test tahmin çizgisi (mavi çizgi) görülmektedir.

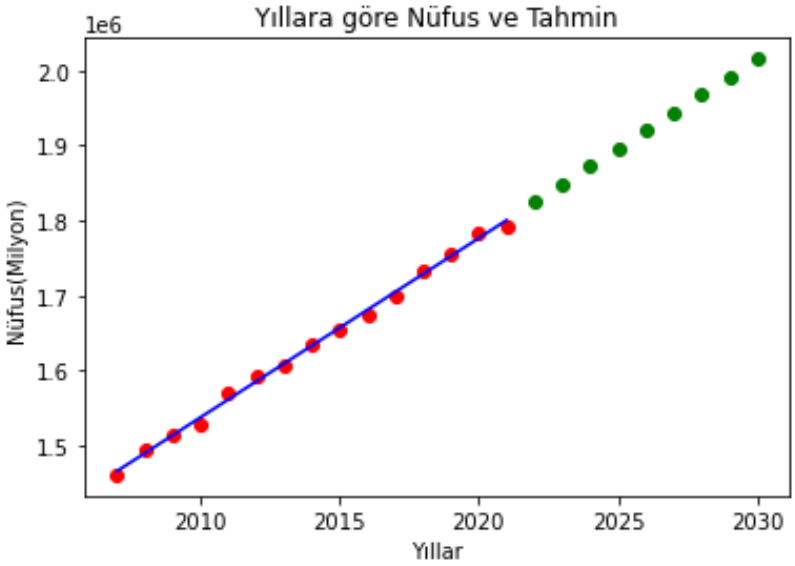


Şekil 3.19: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon test çizgisi (mavi çizgi)

Doğrusal regresyon yönteminin başarılı olduğu görüldüğünden (R^2 = 0,9968) 2022-2030 yılları için aynı yöntemle tahmin yapılmış ve Çizelge 3.2'deki değerler ve Şekil 3.20 elde edilmiştir.

Çizelge 3.2 Gelecek yıllar Diyarbakır tahmini nüfus değerleri

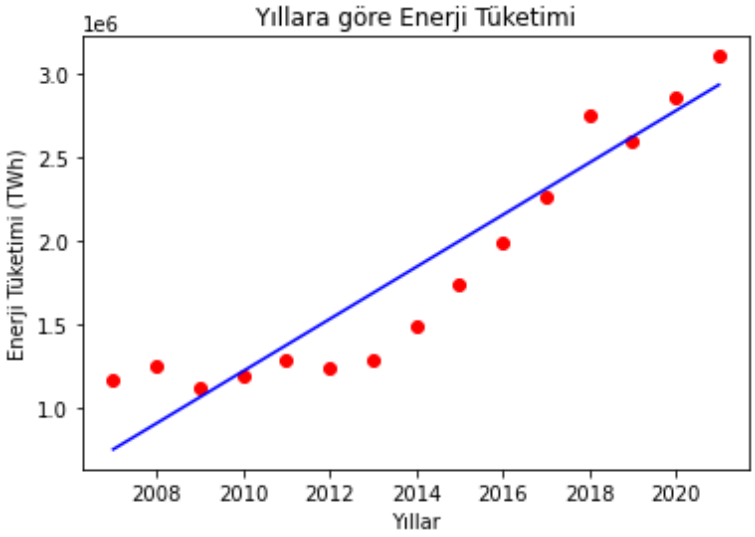
Yıl	Doğrusal Regresyon ile Elde Edilen Tahmini Nüfus (Kişi)
2022	1824283
2023	1848194
2024	1872105
2025	1896016
2026	1919927
2027	1943838
2028	1967749
2029	1991660
2030	2015571



Şekil 3.20: Gerçek değerler (kırmızı noktalar), doğrusal regresyon test çizgisi (mavi çizgi) ve tahmin değerleri (yeşil noktalar)

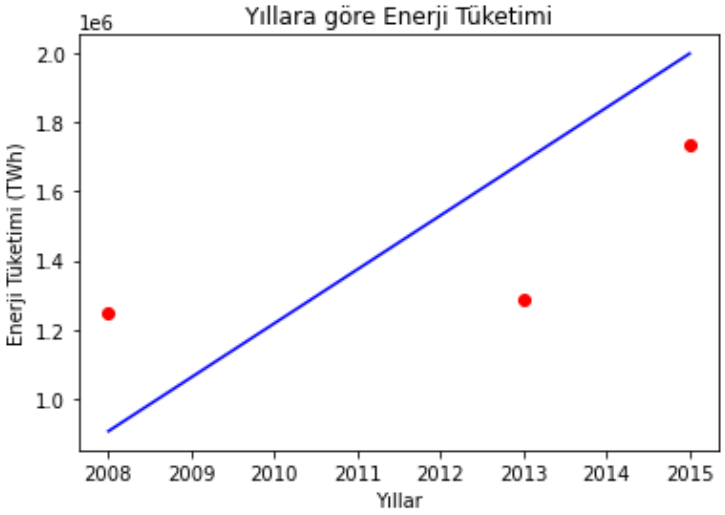
3.4.4 Doğrusal Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

Şekil 3.21'da Çizelge 3.1'deki verilere göre yıllara denk gelen elektrik tüketimi kırmızı noktalarla belirtilmekte ve doğrusal regresyon yöntemi ile üretilen elektrik tüketim tahmini mavi çizgi ile belirtilmektedir. Elektrik tüketim değerleri Bölüm 3.4 boyunca tüm grafiklerde TWh olarak değerlendirilmektedir.



Şekil 3.21: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon çizgisi (mavi çizgi)

Doğrusal regresyon yönteminin verilerimiz üzerinde başarılı olmadığı görülmekle beraber mevcut verilerimiz %80 eğitim (train) ve %20 test olmak üzere iki kısma ayrılmıştır. Şekil 3.22'de doğrusal regresyon test sonuçlarına göre x_test değerleri için verilen y_test gerçek değerleri (kırmızı noktalar) ve y_test tahmin çizgisi (mavi çizgi) görülmektedir.

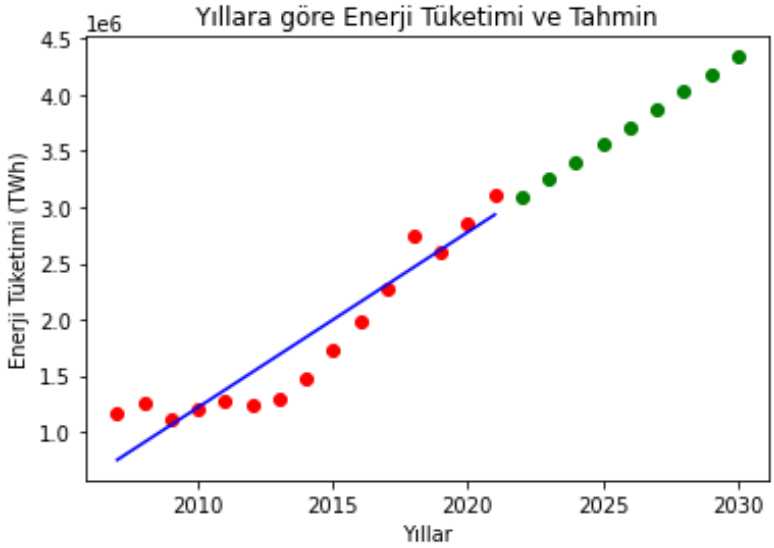


Şekil 3.22: Gerçek değerler (kırmızı noktalar) ve doğrusal regresyon test çizgisi (mavi çizgi)

Doğrusal regresyon yönteminin başarılı olmadığı görülmekle beraber ($R^2 = 0,8767$) 2022-2030 yılları için aynı model üzerinden tahmin yapılmış ve Çizelge 3.3'teki değerler ve Şekil 3.23 elde edilmiştir.

Çizelge 3.3 Gelecek yıllar Diyarbakır tahmini elektrik tüketim değerleri

Yıl	Doğrusal Regresyon ile Elde Edilen Tahmini Enerji Tüketimi (MWh)
2022	3090483,37
2023	3246448,00
2024	3402412,63
2025	3558377,26
2026	3714341,88
2027	3870306,51
2028	4026271,14
2029	4182235,77
2030	4338200,39



Şekil 3.23: Gerçek değerler (kırmızı noktalar), doğrusal regresyon test çizgisi (mavi çizgi) ve tahmin değerleri (yeşil noktalar)

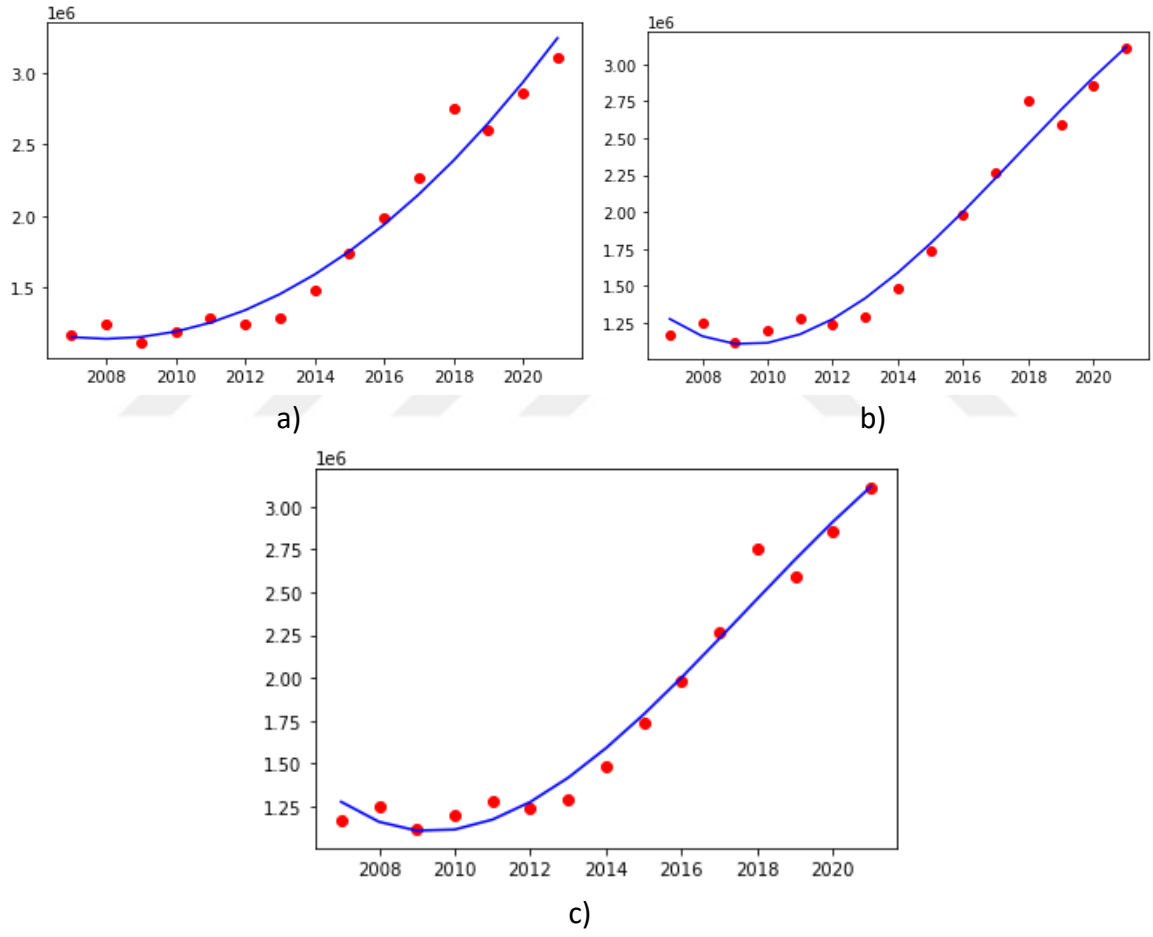
3.4.5 Diğer Regresyon Yöntemleriyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

Çizelge 3.1'deki mevcut değerler baz alınarak doğrusal regresyon yöntemiyle oluşturulan Şekil 3.21'da kırmızı renkte gösterilen mevcut değerler ile mavi çizgi ile gösterilen tahmin çizgisinin pek uymadığı görülmektedir. Ayrıca Şekil 3.22'de doğrusal regresyon ile test edilen çalışmanın yeterince başarılı olmadığı (R^2 = 0,8767) görüldüğünden diğer regresyon yöntemleri denenmektedir. Mevcut verilerin hepsi eğitim

(train) amacıyla kullanılmış sonrasında ise poligonal regresyon, destek vektör regresyonu (SVR), karar ağacı ve rasgele orman regresyon yöntemleriyle tahmin çalışmaları yapılmaktadır.

3.4.6 Poligonal Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

Tüm değerler üzerinden 2. 3. ve 4. dereceden poligonal regresyon grafikleri çizilmiştir. Şekil 3.24'de kırmızı renkteki mevcut değerler ile mavi çizgi ile gösterilen tahmin çizgisinin Şekil 3.24 a)'ya göre Şekil 3.24 b) ve Şekil 3.24 c)'de daha çok örtüştüğü görülmektedir.



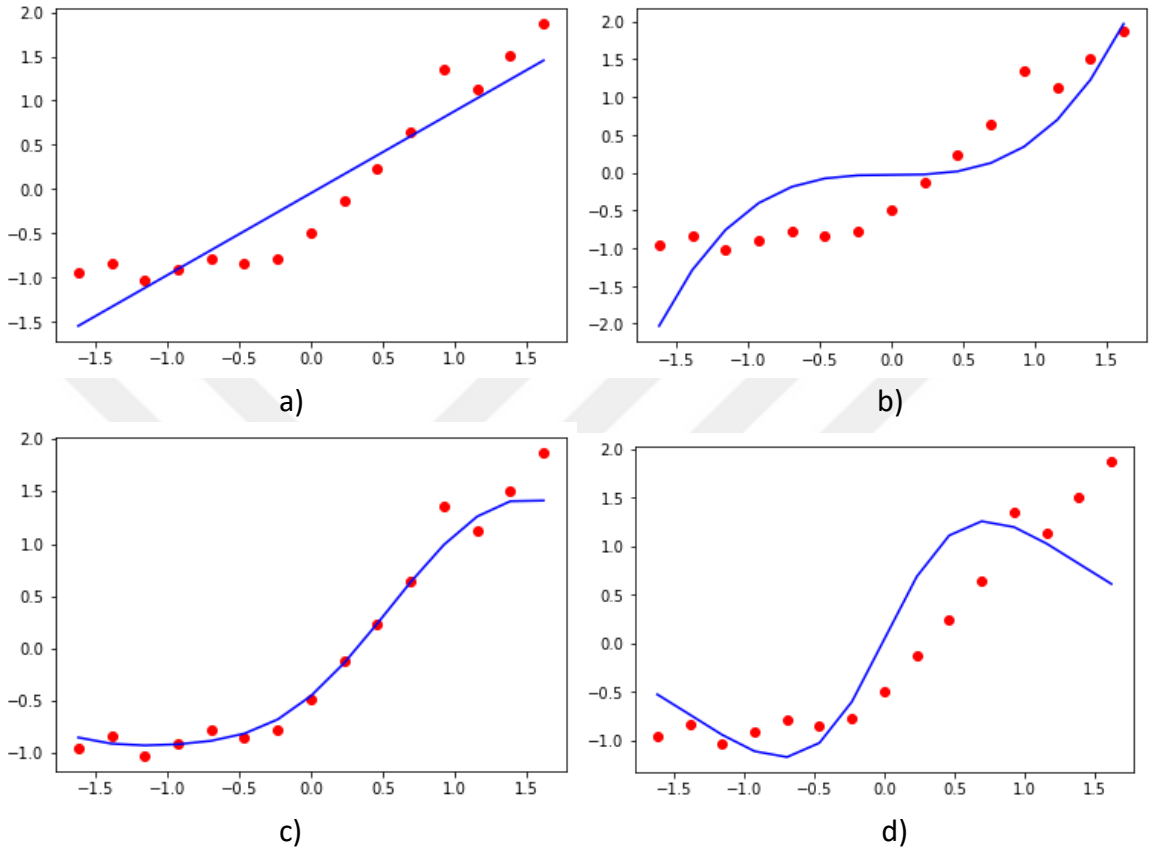
Şekil 3.24: Poligonal regresyon grafikleri **a)** 2. dereceden, **b)** 3. dereceden, **c)** 4. dereceden

3.4.7 Destek Vektör Regresyonu (SVR) Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

Yine tüm değerler üzerinden destek vektör regresyon grafikleri çizilmiştir. Şekil 3.25'te kırmızı renkte gösterilen mevcut değerler ile mavi çizgi ile gösterilen tahmin

çizgisinin Şekil 3.25 a), Şekil 3.25 b) ve Şekil 3.25 d) grafiklerinde pek örtüşmediği Şekil 3.25 c) grafiğinde ise daha iyi örtüştüğü görülmektedir.

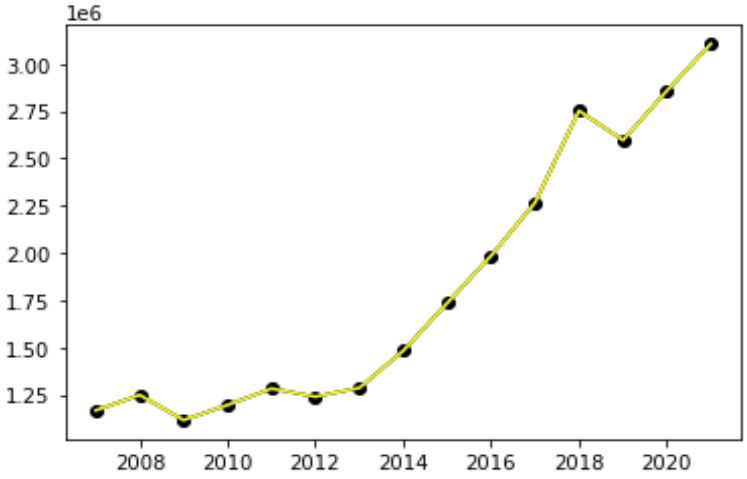
Bu yöntemde çekirdek fonksiyonu için sırasıyla Şekil 3.25 a)'da linear, Şekil 3.25 b)'de poly, Şekil 3.25 c)'de rbf ve Şekil 3.25 d)'de sigmoid tercih edilmiştir.



Şekil 3.25: Çekirdek fonksiyonu için farklı tercihler sonucu oluşturulan destek vektör grafikleri. **a)** linear, **b)** poly, **c)** rbf. **d)** sigmoid

3.4.8 Karar Ağacı Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

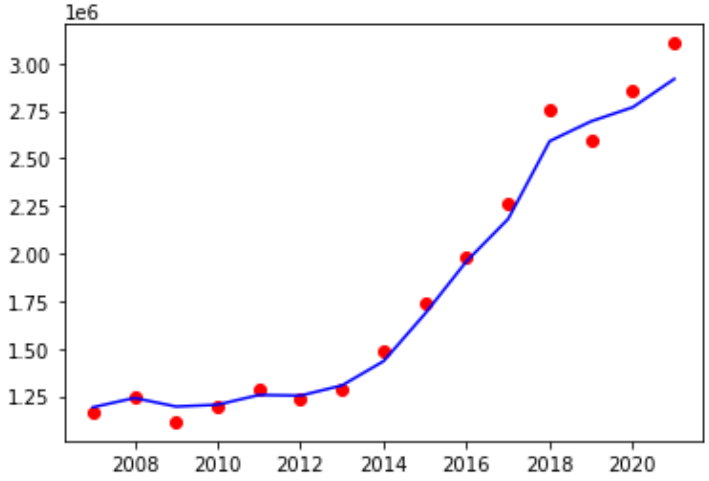
Tüm değerler üzerinden karar ağacı regresyon grafiği çizilmiştir. Şekil 3.26'da yeşil renkte gösterilen mevcut değerler ile sarı çizgi ile gösterilen tahmin çizgisinin çok uyuştuğu görülmektedir. Fakat bu yöntem mevcut veriler ile uyuştuğu halde mevcut dışı aralıkta (farklı X değerlerine karşılık gelen Y değerlerinde) başarı gösterememektedir. Yani yeni tahminlerde de eldeki mevcut sonuçlar üzerinden tahmin yapmaktadır.



Şekil 3.26: Karar ağacı regresyon grafiği.

3.4.9 Rasgele Orman Regresyon Yöntemiyle Diyarbakır Elektrik Tüketim Tahmin Çalışması

Şekil 3.27'de tüm veriler üzerinden rasgele orman regresyon grafiği çizilmiştir. Grafikte kırmızı renkte gösterilen mevcut değerler ile mavi renkte gösterilen tahmin çizgisinin uyuştuğu görülmektedir. Fakat bu yöntem de yine karar ağacı gibi mevcut verileri gözeterek çalışmaktadır. Karar ağacı gibi tahmin yaparken eski sonuçlara tamamen bağımlı olmamasına karşın bu örneğimizde y değerlerinde başarı gösterememektedir. Yani yeni tahminlerde de eldeki mevcut sonuçlara bağlı kalmaktadır.

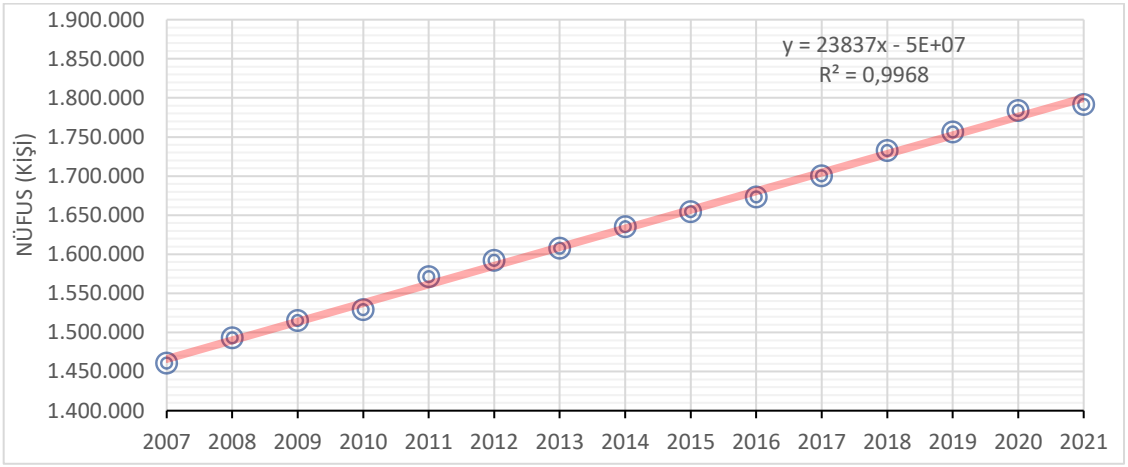


Şekil 3.27: Rasgele orman regresyon grafiği.

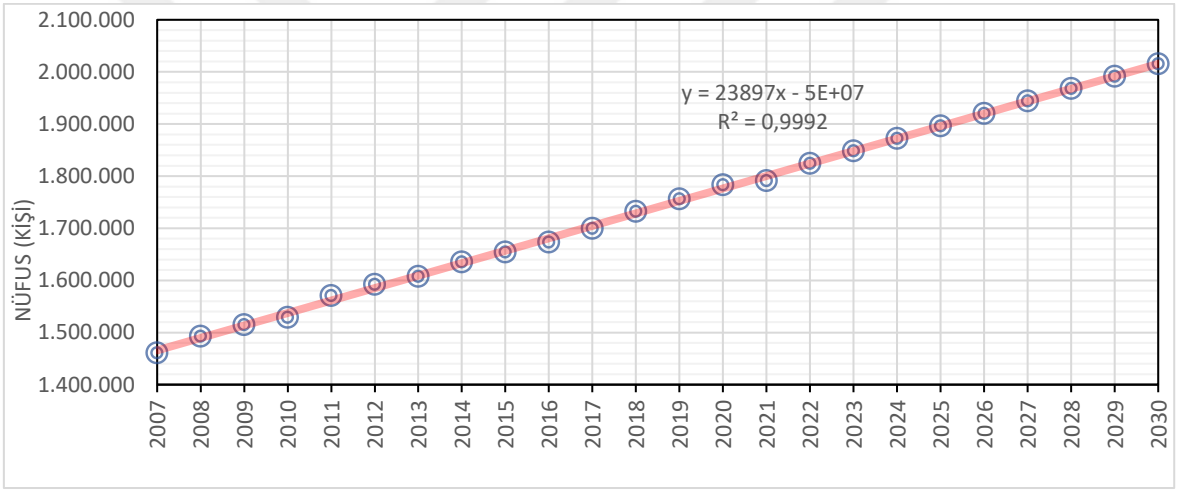
3.4.10 Diyarbakır Nüfus Tahmini Sonuçları

Diyarbakır ili için eldeki veriler baz alınarak nüfus değerleri için doğrusal regresyon ile tahminleme yönteminin Şekil 3.28 a)'da görüleceği üzere yüksek başarım

verdiği görülmektedir. Bu nedenle aynı yöntem gelecek yıllar nüfus tahmini için kullanılmış ve Şekil 3.28 b) 'deki grafik elde edilmiştir.



a)

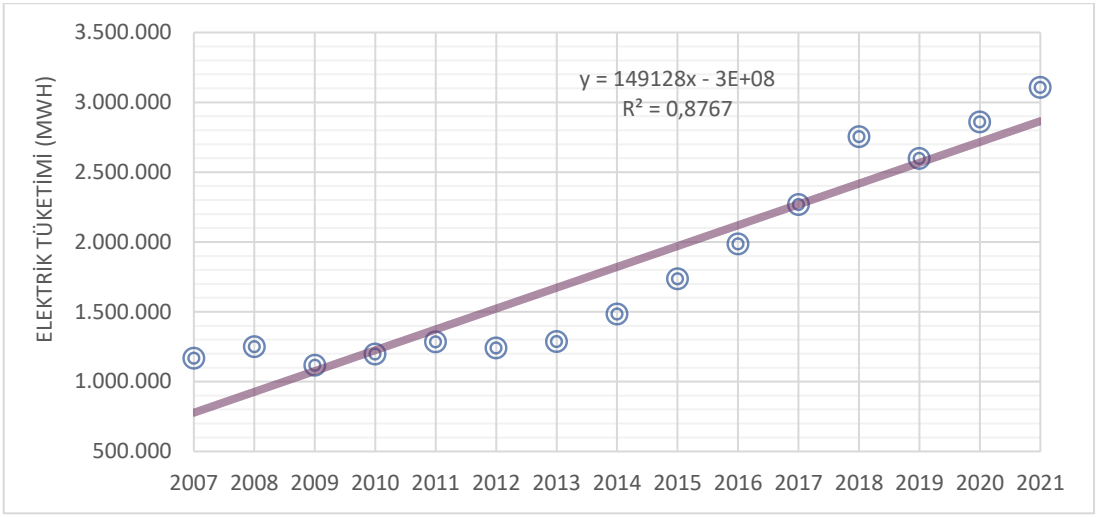


b)

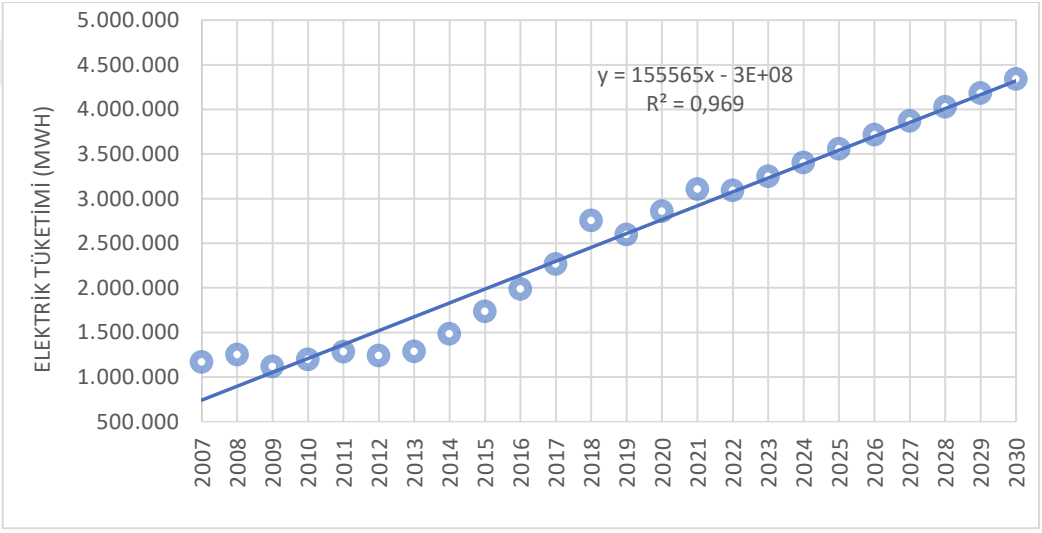
Şekil 3.28: Doğrusal regresyon ile Diyarbakır nüfus tahmin grafikleri **a)** mevcut durum, **b)** tahmin

3.4.11 Diyarbakır Elektrik Tüketim Tahmini Sonuçları

Diyarbakır ili için eldeki veriler baz alınarak elektrik tüketim değerleri için doğrusal regresyon ile tahminleme yönteminin Şekil 3.29 a) 'da görüleceği üzere doğru sonuçlar ve yüksek başarım vermediği görülmektedir. Fakat yine de aynı yöntem gelecek yıllar elektrik tüketim tahmini için kullanılmış ve Şekil 3.29 b) 'deki grafik elde edilmiştir.



a)



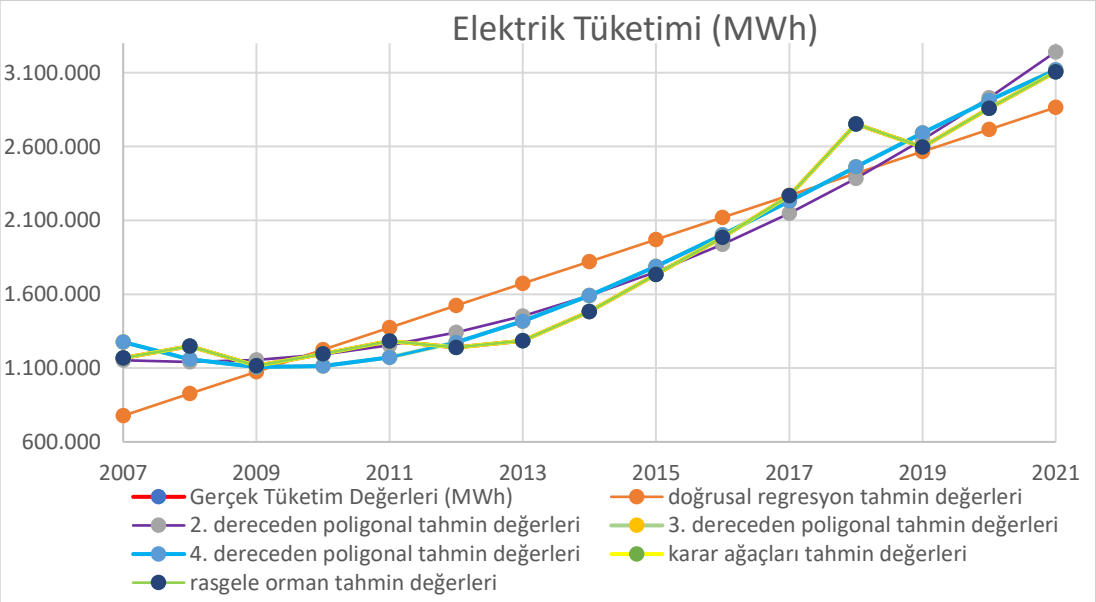
b)

Şekil 3.29: Doğrusal regresyon ile Diyarbakır elektrik tüketim tahmin grafikleri **a)** mevcut durum, **b)** tahmin

Diyarbakır ili için eldeki veriler baz alınarak elektrik tüketim değerleri için doğrusal regresyon ile tahminleme yönteminin başarımının düşük olduğu görülerek daha yüksek başarım için farklı yöntemlere başvurulmuştur. Kullanıla bu yöntemler ve sonuçta elde edilen değerler Çizelge 3.4'te görülmektedir. Şekil 3.30'da ise bu değerlere göre çizilen grafikler görülmektedir. Gerçek tüketim değerleri ile karar ağaçları ve rasgele orman tahmin değerleri ve 3. dereceden poligonal tahmin değerleri ile 4. dereceden poligonal tahmin değerleri çakıştığından dört grafik çizilmiş gibi görünmektedir.

Çizelge 3.4 Diyarbakır'ın mevcut verileri için farklı regresyon yöntemleri ile elektrik tüketim tahmin değerleri.

Yıl	Gerçek Tüketim Değerleri (MWh)	Doğrusal regresyon tahmin değerleri	2. dereceden poligonal tahmin değerleri	3. dereceden poligonal tahmin değerleri	4. dereceden poligonal tahmin değerleri	Karar ağaçları tahmin değerleri	Rasgele orman tahmin değerleri
2007	1.167.058	777.321	1.153.395	1.276.089	1.275.605	1.167.058	1.167.058
2008	1.248.844	926.450	1.141.349	1.159.025	1.158.890	1.248.844	1.248.844
2009	1.115.711	1.075.578	1.154.099	1.107.129	1.107.165	1.115.711	1.115.711
2010	1.195.375	1.224.706	1.191.645	1.113.674	1.113.763	1.195.375	1.195.375
2011	1.282.682	1.373.835	1.253.987	1.171.932	1.172.006	1.282.682	1.282.682
2012	1.239.729	1.522.963	1.341.125	1.275.176	1.275.207	1.239.729	1.239.729
2013	1.285.746	1.672.091	1.453.059	1.416.678	1.416.668	1.285.746	1.285.746
2014	1.482.825	1.821.220	1.589.790	1.589.713	1.589.681	1.482.825	1.482.825
2015	1.734.727	1.970.349	1.751.316	1.787.551	1.787.529	1.734.727	1.734.727
2016	1.985.576	2.119.477	1.937.639	2.003.467	2.003.485	1.985.576	1.985.576
2017	2.266.210	2.268.605	2.148.757	2.230.733	2.230.810	2.266.210	2.266.210
2018	2.752.770	2.417.734	2.384.672	2.462.621	2.462.756	2.752.770	2.752.770
2019	2.596.431	2.566.862	2.645.383	2.692.405	2.692.567	2.596.431	2.596.431
2020	2.858.333	2.715.991	2.930.890	2.913.357	2.913.473	2.858.333	2.858.333
2021	3.106.284	2.865.119	3.241.193	3.118.751	3.118.697	3.106.284	3.106.284

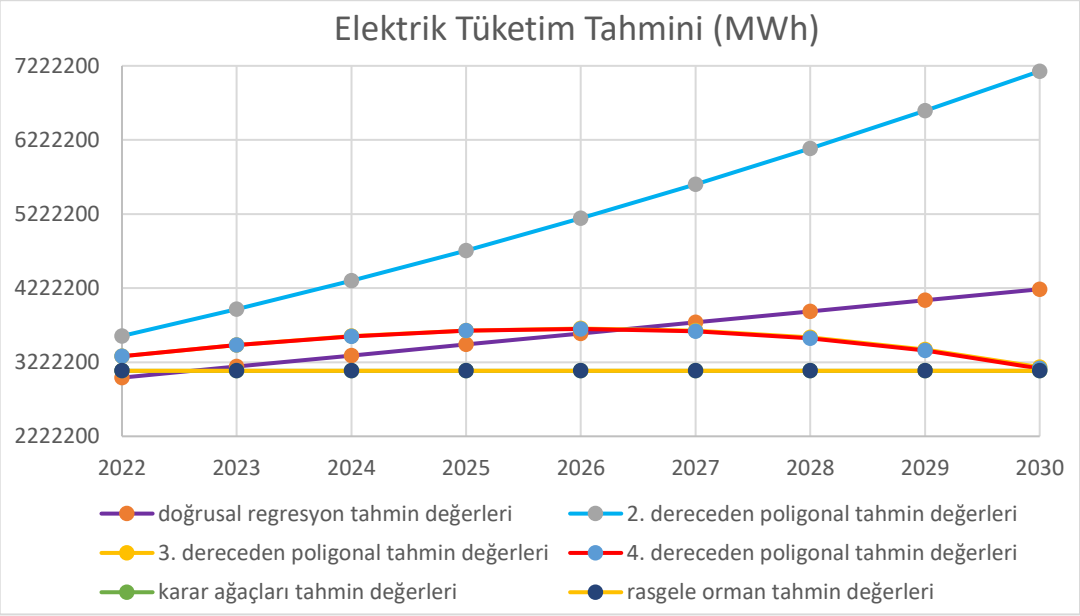


Şekil 3.30: Farklı regresyon yöntemleri ile Diyarbakır'ın mevcut verileri için elektrik tüketim tahmin grafiği

Sonuç olarak 3. ve 4. dereceden poligonal regresyon yönteminin değerlerimize uyduğu ve tahmin için en iyi yöntemin 4. dereceden poligonal regresyon yönteminin olduğu ve bu yöntemin gelecek yıllar için elektrik tüketim tahminlerine yönelik daha doğru sonuçlar vereceği görülmektedir. Fakat yine de tüm durumlar için gelecek yıllar için tahminleme yapılmış ve Çizelge 3.5'e işlenmiştir. Ayrıca Şekil 3.31'de elde edilen bu tahmin değerleri üzerinden grafikler çizilmiştir. Karar ağaçları tahmin değerleri ile rasgele orman tahmin değerleri ve 3. dereceden poligonal tahmin değerleri ile 4. dereceden poligonal tahmin değerleri çakıştığından 4 grafik çizilmiş gibi görünmektedir.

Çizelge 3.5 Gelecek yıllar Diyarbakır tahmini elektrik tüketim değerleri

Yıl	Doğrusal regresyon tahmin değerleri	2. dereceden poligonal tahmin değerleri	3. dereceden poligonal tahmin değerleri	4. dereceden poligonal tahmin değerleri	Karar ağaçları tahmin değerleri	Rasgele orman tahmin değerleri
2022	3014247	3576292	3301858	3301451	3106284	3106284
2023	3163376	3936188	3455951	3454937	3106284	3106284
2024	3312504	4320879	3574304	3572346	3106284	3106284
2025	3461633	4730367	3650189	3646862	3106284	3106284
2026	3610761	5164650	3676879	3671654	3106284	3106284
2027	3759889	5623730	3647647	3639886	3106284	3106284
2028	3909018	6107606	3555765	3544709	3106284	3106284
2029	4058146	6616278	3394506	3379264	3106284	3106284
2030	4207275	7149746	3157143	3136683	3106284	3106284



Şekil 3.31: Farklı regresyon yöntemleri ile Diyarbakır'ın gelecek yıllar için elektrik tüketim tahmin grafiği

3.4.12 R^2 Sonuçları

Yapılan ilk iki çalışmada Diyarbakır nüfus değerlerinin doğrusal regresyon çizgisi ile örtüştüğü ve R^2 değerinin Diyarbakır için 0,9968 değerini verdiği görülmektedir.

Diyarbakır enerji tüketim tahmin çalışmasında ise Çizelge 3.6'da görüleceği üzere en yüksek değeri 3. ve 4. derece polinomal regresyon yöntemi vermektedir. Karar ağacı ve rasgele orman değerinin yüksek olması bizi aldatmamalıdır. Tahmin değerleri kontrol edildiğinde bu değerin bizi yanılttığı görülmektedir.

Çizelge 3.6 Elektrik tüketim tahmini için R^2 değerleri

Regresyon Tipi	Doğrusal	Poligonal 2. Derece	Poligonal 3. Derece	Poligonal 4. Derece	Karar Destek Regresyonu (SVR)	Karar Ağacı	Rasgele Orman
R^2 Değeri	0,8767203	0,965999136	0,97610121	0,976124158	0,692306151	1	0,98586

4. DERİN ÖĞRENME

Derin öğrenme makine öğrenmesinin bir alt dalı veya başlığıdır. Temelde yapay sinir ağları kullanılarak inşa edilmiştir. Bu sinir ağlarının hem katman-gizli katman yapıları çok arttırılarak ağ yapısına derinlik kazandırılmış hem de hiperparametre optimizasyonu gibi bazı metotlar kullanılarak çok geliştirilmiştir.

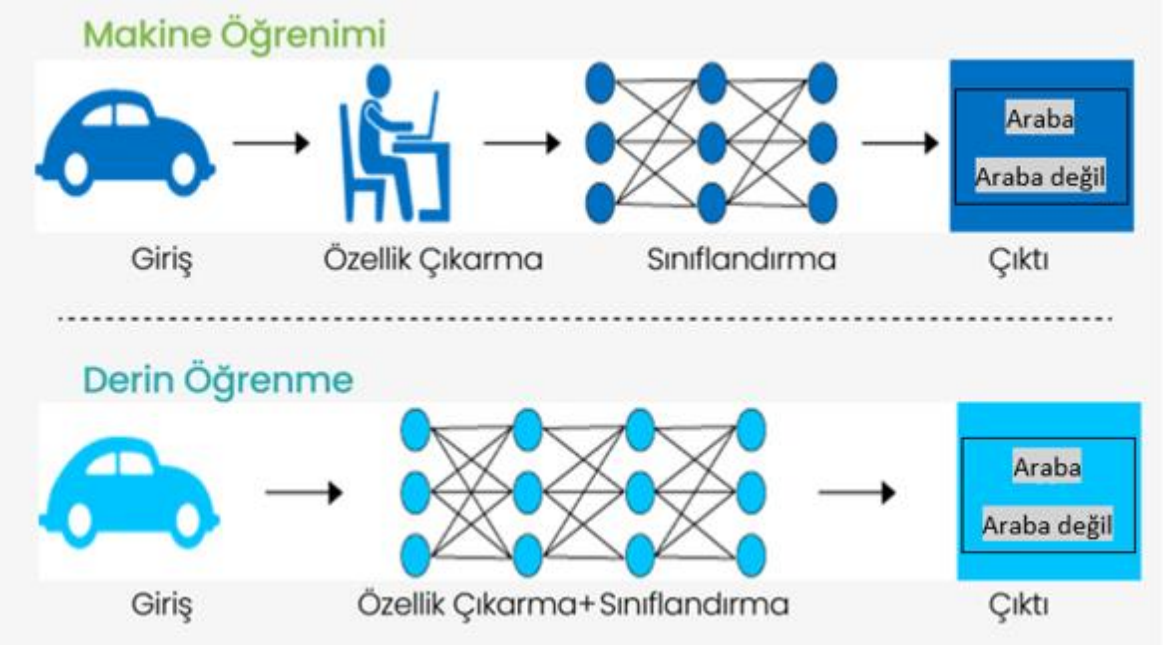


Şekil 4.1: Yapay zekâ – makine öğrenmesi – derin öğrenme ilişkisi.

Her ne kadar derin öğrenme yapı olarak temelde yapay sinir ağlarından üretilmiş olsa da yapısında farklılıklar mevcuttur. Derin öğrenmedeki hesaplama sistemi YSA daki hesaplama yöntemlerini baz almaktadır. Derin öğrenme bilinen YSA yöntemlerine oranla daha başarılı sonuçlar üreten bir yöntemdir. Makine öğrenme yöntemlerinin çoğu yakın zamana kadar, sığ yapılı mimarilerden meydana gelmekteydi. Bu yapılarda en fazla bir veya iki kat doğrusal olmayan özellik dönüşümü içermekteydi. Bu yapılar Gaussian Mixture Models (GMMs), doğrusal veya doğrusal olmayan dinamik sistemler, maksimum entropi (MaxEnt) modelleri, SVM'ler, Conditional Random Fields (CRFs), Logistic Regression, Kernel Regression, Multi Layer Perceptrons (MLP) ve tek bir gizli katman içeren Extreme Learning Machines (ELMs) olarak sıralanabilir [93-96]. Bu yapılar, sınırları belirlenmiş veya basit olan problemleri çözme başarı sağlasa da modelleme ve temsil gücünün sınırlı olduğu alanlarda, dil işleme ve doğal ses çalışmalarında, insan konuşması, görüntü ve görsel sahneleri içeren daha kompleks uygulamalarda zorlanmıştır [97].

Bununla birlikte, veri analizinde, örneğin; görüntü veya ses gibi kompleks yapıları çözmede sığ mimarilerin yeterli olmadığı ve derin mimarilerin gerektiği açıktır. Mesela, insandaki gözün görme sistemi katmanlı hiyerarşik yapılardan meydana gelmektedir [98]. Bu yapı benzerliği model oluşturmada ve çözümlemede derin mimarilerin kullanımını kolaylaştırmaktadır. İleri beslemeli ağlar Deep Neural Networks (DNN) olarak nitelendirilen ve yapısında çok sayıda gizli katman barındıran ileri besleme ağları (Feed-Forward Neural Netwoks-FFNN) veya MLP'ler, bu derin mimarinin ilk örnekleridir.

Makine öğrenmesi derin öğrenmenin ilerleme sağlamasıyla beraber öznitelik mühendisliğinden mimari mühendisliğe doğru bir dönüşüm yaşamaktadır. Araştırmacılar önceki çalışmalarında ilgili problemlerin çözümünde; en iyi öznitelikleri çıkarmak ve bu öznitelikler arasında temsil kabiliyeti en fazla olanları belirlemek üzerine yoğunlaşıyorlardı. Derin öğrenme yöntemlerinin kullanılmasıyla beraber çok katmanlı yapay sinir ağının yapısı; içerdiği nöron sayısı, kaç katmandan oluştuğu, hangi aktivasyon fonksiyonu ya da optimizasyon algoritmasının uygulanacağı problem çözümlerinde en önemli konular haline geldi [99].



Şekil 4.2: Makine öğrenmesi – derin öğrenme [100]

Şekilde görüldüğü gibi makine öğrenme yönteminde nesnenin özellikleri özellik çıkarma ile sınıflandırılır ve bu özellikler modelin oluşturulması amacıyla kullanılır. Derin öğrenmede ise özellik çıkarma aşaması otomatik bir şekilde görüntülerden üretilir ve bilgisayar modelleme işini kendi öğrenir ve yapar. Derin öğrenmedeki en büyük avantajlardan biri belli bir seviyeye kadar verilerin boyutu arttırıldıkça iyileşmenin de

artmasıdır. Böylece çok daha karmaşık problemler, üstesinden gelinebilir bir hale gelmektedir. Derin öğrenme örneklerindeki çeşitliliğin gittikçe fazlalaşması da bu konu ile ilgilidir [100].

Derin öğrenmenin gelişmesinin ve kullanımının artmasının bazı önemli nedenleri olmuştur. Yoksa daha önceleri de derin öğrenme ve benzer çalışmalar yapılmış ama aşağıda belirtilen kıstaslar nedeniyle çalışmalar ilerletilememiş veya gerek görülmemiştir. Bu etmenler: 1. Verinin artması, 2. Bu verilere ulaşmanın kolaylığı, 3. Donanımsal yapıların gelişmesi, küçülmesi, ucuzlaması ayrıca internet üzerinden donanım desteğinin sağlanabilmesidir.

Yapay zekâ uygulamalarının ana malzemesi veridir. Yapılan tüm çalışmalar veri setleri üzerinden gerçekleşmektedir. Bilgi çağına geçişle birlikte internetin de aracılığı ile veri miktarı çok artmaya başlamış ve veriye ulaşım ise kolaylaşmıştır. Günümüzde internet sayesinde, metin, resim, ses ve video verileri devasa boyutlara ulaşmış ve veri erişimi çok kolay hale gelmiştir. Veri ve veriye ulaşım arttıkça da yapay zekâ çalışmaları artmıştır. Ayrıca öğrenmenin derinleştirilmesi için gereken çok fazla veri elde edilmesi sayesinde derin öğrenme olayının gerçekleştirilebilmesi için imkân doğmuş ve büyük verilere erişim arttıkça derin öğrenme daha başarılı ve etkili hale gelmiştir.

Teknolojik gelişmeler sayesinde zekâ çalışmalarının önemli bir ayağı olan bilgisayarlar da gelişmiştir. Eskinin büyük bilgisayarları ortadan kalkmış, yerine taşınabilir, daha küçük, güçlü, işlevsel ve ucuz bilgisayarlar üretilmiştir. Böylece kullanım artmış, yapılamayan hesaplar yapılmaya başlanmıştır. Ucuzlayarak her eve giren bilgisayarlar sayesinde her kullanıcı bu alanda çalışma ve katkı sunabilir olmuştur. Derin öğrenme gibi karmaşık modeller kullanılabilmiş, yeni yöntem ve modeller geliştirilebilmiştir. Ayrıca bazı şirketlerin sağladığı bulut desteği ile uzaktan hem donanım hem yazılım desteği sağlanarak çalışmaların çok daha güçlenmesi ve hız kazanması sağlanmıştır.

Derin öğrenmenin zamanla doğruluk oranını arttırması ve daha karmaşık olayları çözmesi sayesinde derin öğrenmeye olan dikkat ve ilgi de artmıştır. Bu alanda şirketler kurulmaya başlanmış, yeni bir sektör ortaya çıkmıştır.

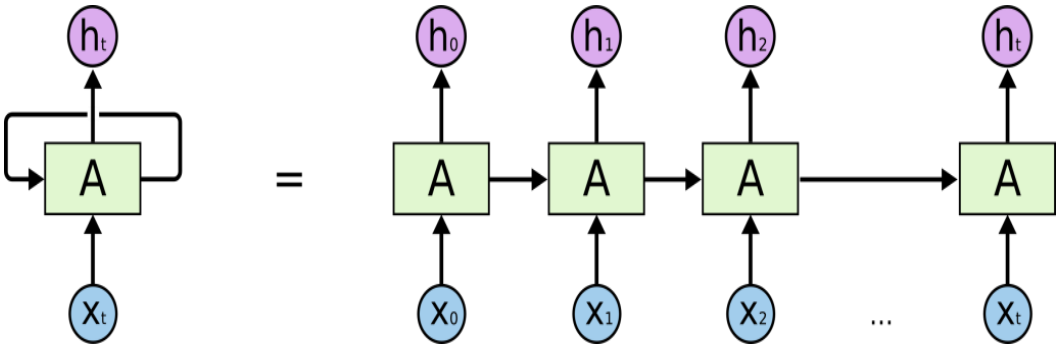
4.1 Derin Öğrenme Mimarileri

Derin öğrenmede farklı problemlerin çözümü amacıyla tasarlanmış farklı mimariler mevcuttur. Bu mimarileri Auto-Encoder (Otomatik Kodlayıcı), CNN (Evrişimli Sinir Ağları), RNN (Özyinelemeli Sinir Ağları), RBM (Kısıtlı Boltzmann Makinesi), DBN (Derin İnanç Ağları), GAN (Çekişmeli Üretici Ağlar), Transfer Learning (Transfer Öğrenimi), DRL (Derin Pekiştirmeli Öğrenme) halinde sıralayabiliriz. Biz burada zaman serileri için sıkça kullanılan RNN ve RNN mimarisinden dönüşümle üretilen LSTM mimarisine bakacağız.

4.1.1 RNN (Özyinelemeli sinir ağları)

Basit Tekrarlayan Ağ (Simple Recurrent Network-SimpleRNN), ilk olarak Jeff Elman tarafından üretilmiştir. Elman'ın cümle yapısını simüle etmek amacıyla ürettiği bu modelde her sözcük için gizli kalıpların üzerinde ortalama örüntü kümelemesi sonucunda sözcükler isim ve fiil olarak sınıflandırılmışlardır. Bunun yanında isimlere göre yapılan canlı- cansız, insan-hayvan, hatta hayvanlar arası avcı-yırtıcı gibi kümeleme ve sınıflandırmalar başarıyla yerine getirilmiştir [101]. RNN, kısımlar arasındaki bağlantıların yönlendirilmiş bir döngü meydana getirdiği yapay sinir ağı mimarisidir. Bu döngü sayesinde, dinamik zamansal davranış sergilenmesine imkan sağlayan bir ağ iç durumu meydana getirilmiştir. FFNN'lerin aksine, RNN'ler kendi giriş belleğini girdilerin rastgele dizilerini işlemek için kullanabilmektedirler [102]. RNN'in temel amacı sıralı bilgi veya zaman serilerini kullanabilmesidir. Görüntü tabanlı verilerde tüm girdilerin bağımsız veriler olduğu değerlendirilmektedir. Ancak NLP gibi zaman değişkeni olan verilerde bu durum mümkün olmamaktadır. Bir cümle içinde sonradan gelen bir kelimenin tahmin edilmesi için, o anki kelimeden önceki kelimelerin de bilinmesi gerekmektedir. RNN mimarisinin tekrarlanan (recurrent) olarak nitelendirilmesinin nedeni, bir dizinin her öğesi için (cümledeki sözcükler gibi) aynı görevi önceki çıktılara bağlı olarak ifa etmesidir [103].

Alex Graves, ses verilerinin fonetik sunumuna ihtiyaç olmadan direk metne dönüştüren RNN tabanlı bir konuşma tanıma sistemi oluşturmuştur [104]. Ayrı bir çalışmada, RNN'ler CNN'le beraber kullanılmış, böylece etiketlenmemiş görüntüler için tanımlayıcı üreten bir modelin parçası olarak kullanılmış ve hem görüntüdeki nesneler tanımlamış hem de tanımlayıcıların görüntülerdeki konumlarını bulunmuştur [105].

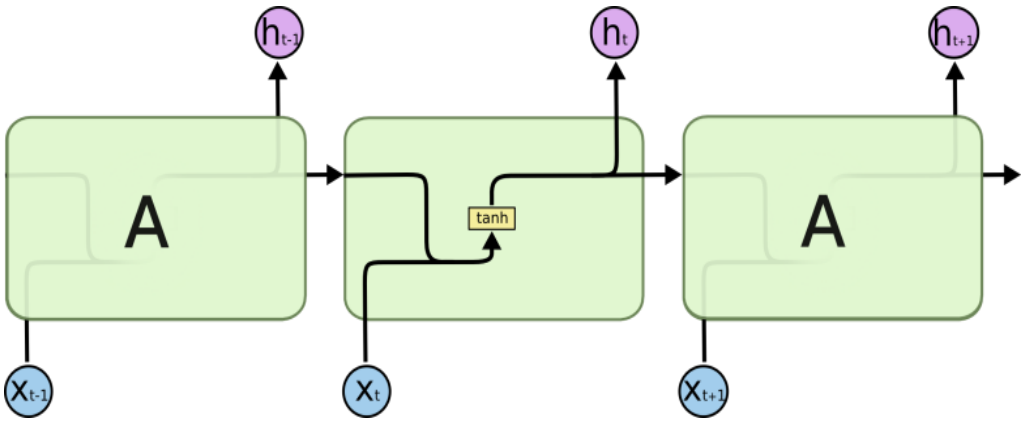


Şekil 4.3: Kaydedilmemiş bir tekrarlayan sinir ağı. [106]

4.1.2 LSTM (Uzun kısa vadeli hafıza ağları)

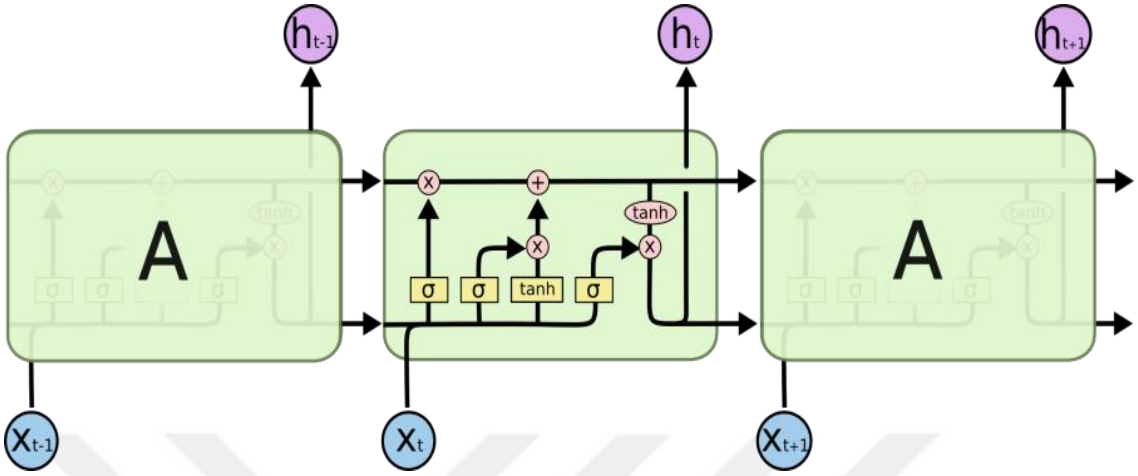
RNN mimarileri önceki bilgiyi de kullanıma dayanan bir yaklaşımdır. Bir örnek verecek olursak; "Ağaç toprakta yetişir" cümlesinde "toprak" kelimesinin tahmini basittir. Ancak bağlamlar arasındaki boşluklar arttığında RNN'in geçmişten gelen bilgileri değerlendirmesi zorlaşmaktadır. Örneğin, "İran'da büyüdüm … Akıcı bir şekilde Farsça konuşurum." gibi bir metinde "Farsça" kelimesinin tahmininde, geçtiği cümleden yola çıkılarak bir dil adı olacağını tahmin edilebilir. Fakat doğru sözcüğün "Farsça" olduğunu tahmin edebilmek için, başta geçen cümledeki yer bilgisinin hafızada tutulması gerekmektedir [103]. Teoride mümkün görünen "uzun-vadeli bağımlılıklar", pratikte büyük sorunlara neden olmaktadır [107]. Bu sorunun çözümü amacıyla Hochreiter ve Schmidhuber 1997 yılında, uzun vadeli bağımlılıkları öğrenebilen özel bir RNN türü olan Uzun Kısa Vadeli Bellek (Long Short Term Memory - LSTM) ağlarını üretmiştir [108].

Tüm tekrarlayan sinir ağlarının yapısı, sinir ağının tekrarlayan modülleri zinciri şeklindedir. Normal RNN'lerde, bu tekrarlanan modül, tek tanh tabakası gibi çok basit bir yapıya sahip olacaktır [106].



Şekil 4.4: Standart bir RNN'de tek bir katman içeren yinelenen modül [106]

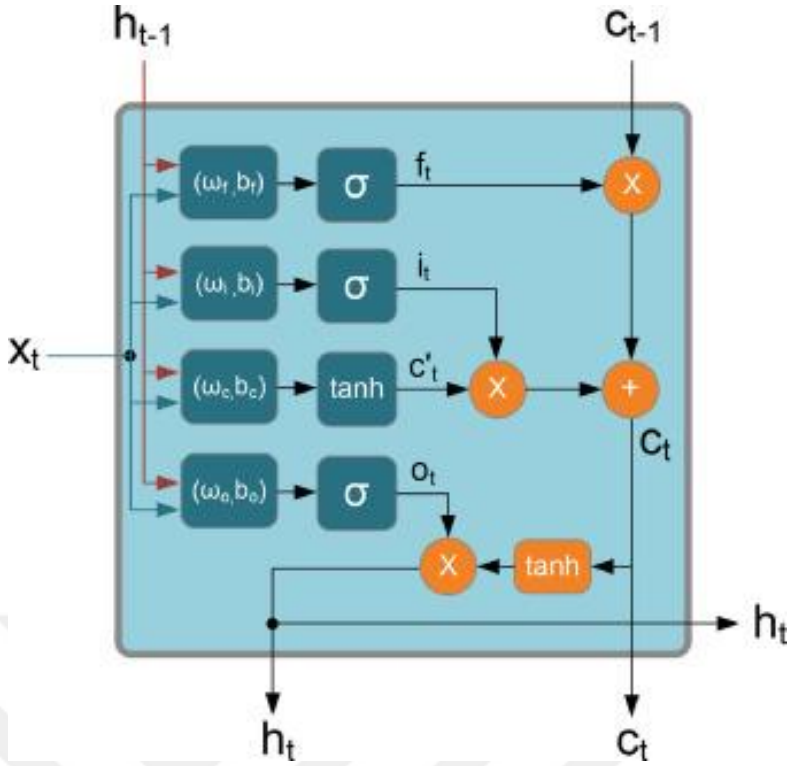
LSTM'ler de bu zincir benzeri yapıya sahiptir, fakat tekrarlayan modül farklı bir yapıdadır. Tek bir sinir ağı katmanına değil, çok özel bir şekilde etkileşime giren dört katman içeren farklı bir yapıya sahiptir [106].



Şekil 4.5: LSTM'deki birbiriyle etkileşen dört katman içeren yinelenen modül [106]

LSTM yapısında giriş, unutma ve çıkış olmak üzere 3 farklı kapıya ve blok girişi, sabit hata döngüsü, çıkış aktivasyon fonksiyonu ve gözetleme (peephole) olmak üzere de 4 farklı bağlantıya sahiptir [109]. Bloğun çıktısı tekrar tekrar bloğun girişine ve tüm kapılarına bağlanır. İlk geliştirilen mimaride unutma kapısı ve gözetleme bağlantıları mevcut değildi. Sonradan LSTM'ye durumunu sıfırlaması amacıyla unutma kapısı [110] ve kesin zamanlamaları kolayca öğrenmesi için ise gözetleme bağlantıları ilave edilmiştir [111].

Şekil 4.6'dan görüleceği üzere giriş kapısı, geçerli durum x(t) ve gizli durum h(t-1)' i ikinci σ-fonksiyonuna aktararak mevcut halin aktifleştirilip aktifleştirilmeyeceğini belirler. Sonrasında, gizli durum ve geçerli durum verileri tanh fonksiyonundan geçirilir ve aktivasyon fonksiyonları tarafından üretilen çıktı değerleri çarpmaya hazır hale getirilir. Sonunda, unut kapısı hangi dikkate alınacak veya unutulacak bilgiye karar verir. Giriş x(t) ve gizli durum h(t-1) üzerinden aktarılan bilgi, f(t)'nin 0-1 arası değerler alması amacıyla sigmoid fonksiyonundan geçirilir. Böylece, çıkış kapısı sonra gelen gizli katmanın durumunu ayarlamak için önceki girdilerden aktarılan bilgiyi kullanır. Bu yüzden, bu birim bir sonraki gizli durumu sona erdirir [112].



Şekil 4.6: LSTM mimarisinin genel yapısı. [112]

LSTM konuşma/metin işleme alanında oldukça başarılı sonuçlar elde etmektedir. Çerçeve tabanlı ses sınıflandırma çalışmasında farklı lehçelerdeki zengin içerikler barındıran TIMIT veri kümesi kullanılarak %70 başarı sağlanmıştır [113]. Uçtan uca eğitimin uygulandığı başka bir LSTM modeli ile aynı veri kümesi üzerinde bazı düzenlemeler yapıldıktan sonra %83 başarı elde edilmiştir [114]. Başka bir çalışmada Verbmobil veri kümesi kullanılarak anahtar kelime tespitinde %84,5'lik bir doğruluk oranına LSTM ile ulaşılmıştır [115].

Ayrıca LSTM mimarisi blues türünde müzikleri üretebilen bir model üretilmesinde [116], hizalamadan bağımsız olarak protein homolojisinin algılanmasında [117], robotik kalp cerrahisi alanında düğümleri bağlamaya çalışan bir tasarımda [118], düzensiz dillerde öğrenme [119], çevrimdışı el yazı tanınmasında [120] ve daha birçok farklı alanda kullanılmaktadır.

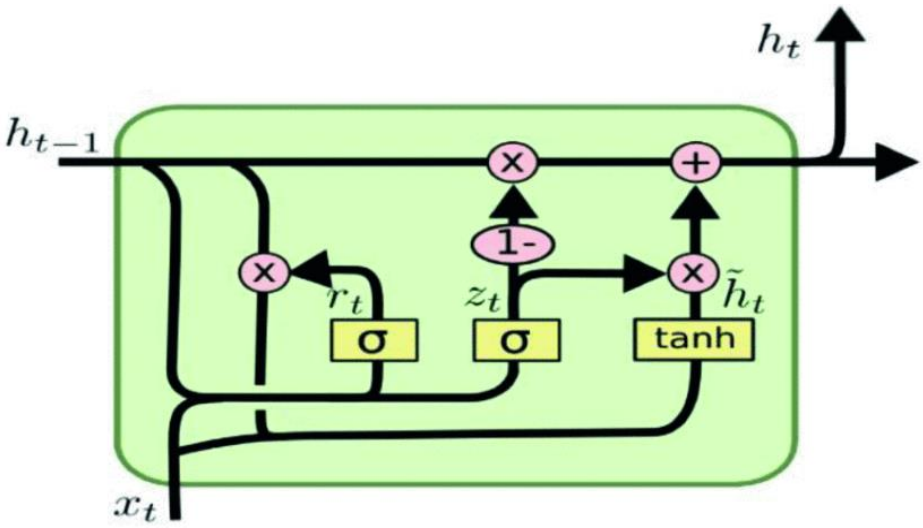
4.1.3 GRU (Geçitli tekrarlayan birim)

GRU ünitesi, geçerli giriş ve önceki zaman adımının gizli durumundan oluşacak şekilde iki giriş değişkeni ile karakterize edilir. GRU modeli, uzun vadeli zaman serisi tahminlerinin doğruluğunu artırmak için geçit mekanizmaları sunan başka bir ifade ile gating mekanizmalarını modele dahil ederek ele alan ve böylece RNN'nin eğim sorununu

çözmeye çalışan yeni bir tekrarlayan sinir ağı türüdür. Geçit kontrol mekanizması, modelin birimlerinin bilgileri tutmasını ve unutmasını, yani tekrarlayan ünitelerin genel bilgilerini içermek yerine, bilgilerin artırılıp azaltılmasını sağlar. Bu şekilde uzun dizilerin analizi ve öğrenilmesi sağlanır [121].

Şekil 4.7'de GRU modelinin yapısı görülmektedir. GRU'nun her yinelenen biriminde bir güncelleme ve bir sıfırlama kapısı mevcuttur. Burada güncelleme kapıları uzun vadeli hafızayı yönetmeye yarar. Modelin, geleceğe aktarılacak önceki zaman adımlarından bilgi seçmesini sağlar. Bu oldukça dinamik bir özelliktir. Çünkü model önceki bilgilerin hepsini kopyalamaya ve kaybolan bir gradyan ihtimalini tümüyle ortadan kaldırmaya karar verebilir. Güncelleme kapısı, önceki gizli durumun hangi miktarda geçerli girdisinin kullanılarak güncellenebileceğini gösterir. Sıfırlama kapısı, kısa vadeli hafızayı veya ağın gizli durumunu yönetir. Uzun vadeli zaman serisindeki bağımlılıkları simüle etmek amacıyla sıfırlama kapısı, GRU'nun önceki gizli durumdan aktarılan bilgilerin unutulacağına veya saklanacağına karar verdiği için değerlidir [122].

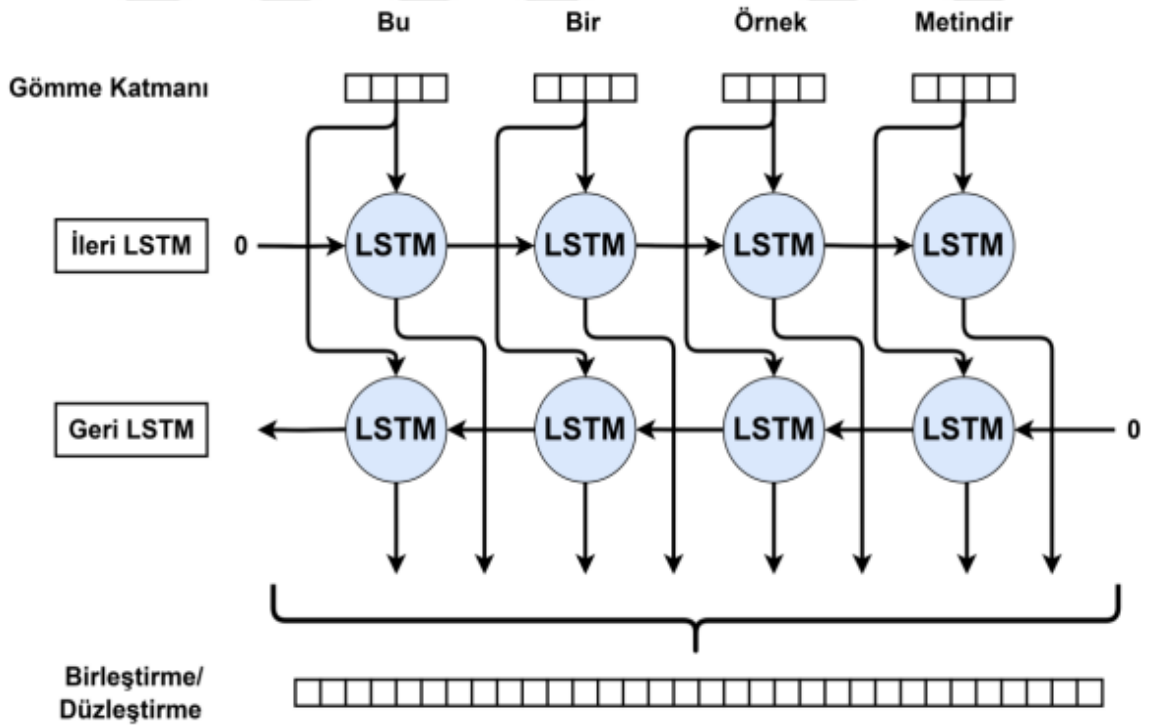
GRU, bilinen RNN'den daha gelişmiş bir modeldir. Uzun vadeli zaman serisi verilerinin amacı ardışık verileri işlemektir LSTM gibi tüm ağ boyunca bilgi akışını düzenlemek için geçitleme tekniklerinden yararlanılmaktadır. Bununla birlikte, bir GRU daha az parametreye sahip olduğundan bir LSTM'den daha hızlı eğitilir ve daha fazla hesaplama açısından etkilidir [122].



Şekil 4.7: GRU modeli [122]

4.1.4 Bi-LSTM (Çift yönlü uzun kısa vadeli hafıza ağları)

Bidirectional LSTM (Bi-LSTM) modeli, iki yönlü bilgi akışı ile geleneksel LSTM mimarisine genişlik kazandırıp, daha iyi performans gösteriyor. BiLSTM giriş verilerini iki kez (hem soldan sağa hem de sağdan sola) işleyerek temel bağlamı daha iyi yakalamaktadır [69]. Şekil 4.8'de gösterilen Bi-LSTM hem geçmiş hem gelecek bilgileri değerlendirmek üzere üretilmiştir. Bu, iki ayrı gizli katmandaki nöronların tekil bir çıktı katmanına bağlanmasıyla sağlanır. Tek yönlü LSTM'lerin aksine, Bi-LSTM giriş dizisini hem ileri hem de geri yönde çalışır. Bu çift yönlülük, modelin yalnızca geçmiş zaman adımlarından değil, aynı zamanda gelecekteki zaman adımlarından da bağımlılıkları yakalamasını sağlar. Böylece zamansal ilişkiler ile ilgili kapsamlı bir anlayış sağlar. Bi-LSTM, yapısında LSTM gibi bilgi akışını düzenlemek için kapılar ve bellek hücreleri bulundurur. Çift yönlü yapı, diziyi ileri doğru ve geri doğru işleyen iki bellek hücre kümesi sağlar. Bu çift bellekli hücre yaklaşımı, modelin her iki yöndeki karmaşık bağımlılıkları yakalama kapasitesini artırır [123]. Özetle, Bi-LSTM modeli, her iki yöndeki zamansal bağımlılıkları yakalamak için gelişmiş bir kapasite sunar. Bu çift yönlü yaklaşım, modelin bağlamsal anlayışını ve uyarlanabilirliğini artırarak, onu doğru zaman serisi tahmini için etkin kılar [124].



Şekil 4.8: Bi-LSTM modeli [125]

4.2 Derin Öğrenmede Kullanılan Donanımlar

Derin öğrenme modeli tasarlamak için gelişmiş ve internete bağlanabilen bir bilgisayar yeterlidir. Herkes bilgisayar başında bulunduğu yerden derin öğrenme modelleri üretebilir. Ancak bilimsel disipline sahip olmadan model tasarlamak mümkün olmamaktadır. Bu nedenle daha önceden bir model tasarımının temelden incelemiş olması gerekir. Ayrıca maddi imkanların kısıtlı olduğu durumlarda ücretsiz bulut servislerinin çalışmalarda kullanması önerilmektedir (Microsoft Azure Notebook, Google Colab gibi.) [126].

Kişisel bilgisayarda ise kullanılan programa uygun IDE (Integrated Development Environment-Tümleşik Geliştirme Ortamı) tercihi yapılmalıdır. Python programı ile çalışılıyorsa Anaconda (Package Management Tool-Paket Yönetim Servisi) veya Visual Studio Code, Java için Eclipse kullanılabilir. Ücretsiz bulut ortamında: Microsoft Azure Notebook (sadece CPU) ve Google Colab (GPU desteği var) bilgisayara kuruluma ihtiyaç duymaksızın uygulamaların geliştirmesine olanak tanımaktadır [126].

4.3 Derin Öğrenmede Kullanılan CPU, GPU ve TPU

Derin öğrenme çalışmalarında büyük veriler işlenmekte, çok fazla matematiksel işlemler yapılmaktadır. Bu yüzden de yapılan uygulamalar çok zaman almaktadır. Zaten derin öğrenmenin şimdiye kadar gelişememesinin önemli nedenlerinden biri de bilgisayar donanımı ya da CPU (Central Processing Unit – Merkezi İşlem Birimi)'nin yeterince gelişmemiş olmasıdır. CPU'lar bilgisayarların beyni olarak nitelendirilen merkezi işlem birimidir. Türkçede işlemci de denmektedir. Bilgisayardaki aritmetik ve mantıksal işlemleri yapan, aynı zamanda kontrol ve adresleme gibi önemli görevleri de yerine getiren en önemli birimdir. Bilgisayarın hızını belirleyen en önemli parçadır.

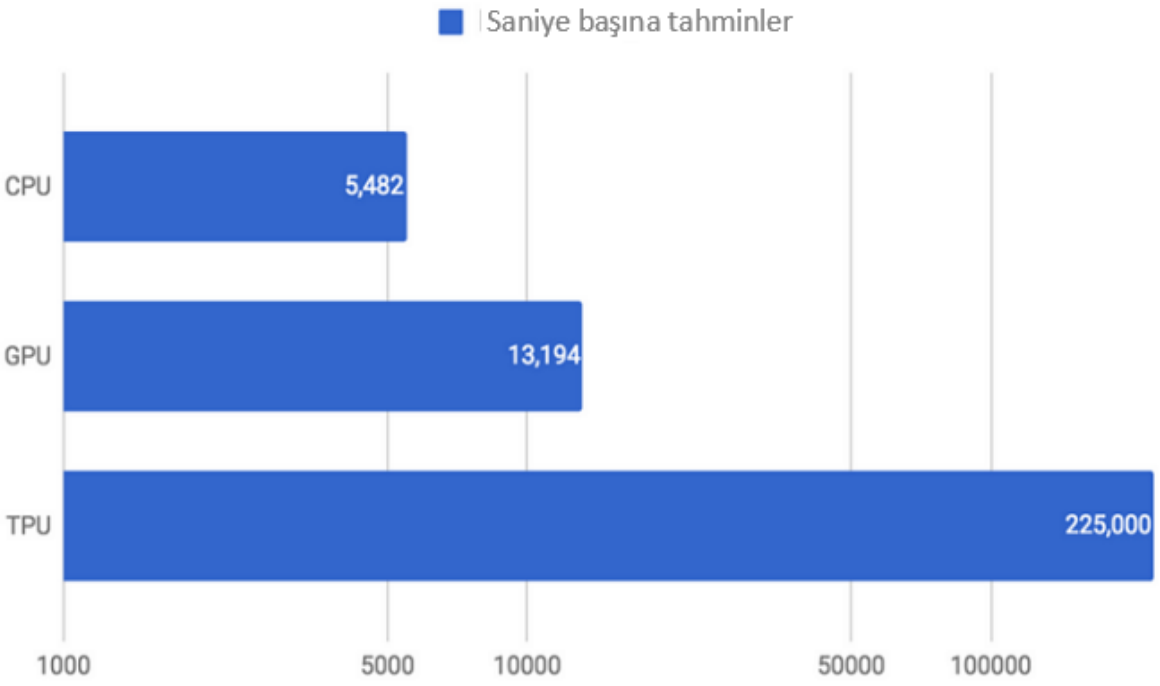
CPU'nun kabiliyetlerine rağmen gelişen teknoloji sayesinde oyun sektörü de grafiksel anlamda çok mesafe katettiğinden CPU artık grafiksel işlemler alanında bile eksik kalmaktadır. Bu yüzden ekran kartları üzerine yerleştirilmiş ve grafik işlemleri için tercih edilen işlemciler geliştirilmektedir. Hız ve performans isteyen işlemlerde ekran kartı üzerinde bulunan bu işlemci kullanılarak işlemler daha seri olarak gerçekleştirilmektedir. GPU (Graphics Processing Unit – Grafik İşleme Birimi) denen bu birim ile aynı anda binlerce işlem gerçekleştirilmektedir. GPU'nun amacı hızlı bir şekilde grafiklerin oluşturulmasını sağlamaktır. Ayrıca makine öğrenme ve derin öğrenme alanında

modellerin eğitilmesi, denenmesi, test edilmesi, analizi gibi birçok alanda kullanılmaktadır [127].

Günümüz işlemcisi; "büyük ölçekte" ama 8 veya 16 çekirdek/işlem-parçacığı sunarken, bir grafik kartı "küçük ölçekte" ama 2000 veya 3000 adet işlem parçacığı sunabilmektedir. Bu da ciddi hız farkı oluşturmaktadır [128].

30 yıldır mikroişlemci performansları her yıl %50 artmaya devam etmektedir. Ancak yarı iletken fiziğinin limitleri dikkate alındığında, artık CPU performansı yılda yalnızca %10 artmaktadır. NVIDIA GPU hesaplama, sektörü daha ileri taşıyarak 2025'te 1000 kat hızlanma sağlayacağını söylemektedir [129].

Derin öğrenme çalışmaları için GPU'lar büyük katkı sağlamaktadır. Fakat Google bu konuda GPU teknolojisini daha ilerilere taşıyıp TPU (Tensor Processing Unit – Tensör İşleme Birimi) teknolojisini üretti. TPU derin öğrenme konusuyla ilgili çalışanlara ücretsiz sunulan bir teknolojidir [127]. TPU'nun diğer işlemcilerden farkı, derin öğrenmenin temeli sayılacak lineer cebir (matris) işlemlerinin paralel ve çok boyutlu olarak gerçekleştirilmesi için oluşturulmuş bir donanımdır. Google'ın sunduğu bu TPU (Tensor İşlem Birimi) teknolojisini üreten ekibe göre, "TPU'lar, CPU ve GPU'lara göre 15 ila 30 kat daha hızlı" olarak değerlendirilmektedir. Şekil 4.9'da CPU-GPU-TPU arasındaki saniyedeki kestirim sayısı karşılaştırması görülmektedir [130].



Şekil 4.9: CPU-GPU-TPU saniyedeki kestirim sayısı karşılaştırması [130]

Google Colab, Google e-posta hesabı üzerinden ücretsiz olarak derin öğrenme çalışmaları yapmaya olanak sağlamaktadır. Colab ile derin öğrenme kodları yüklenmekte ve hangi işlemci üzerinde (CPU, GPU ve TPU) çalıştırılacağı seçilip çalışmalar yapılabilmektedir [127].

Bununla birlikte derin öğrenme çalışmaları için GPU veya TPU her zaman bir zorunluluk değildir. GPU ihtiyaç olup olmadığı çalışılacak veri setine, modelin karmaşıklığına ve zaman kısıtlamasına bağlıdır [130].

4.4 Derin Öğrenmede Kullanılan Aktivasyon Fonksiyonları

Önceki bölümlerde aktivasyon fonksiyonlarından detaylıca bahsetmiştik. Aktivasyon fonksiyonlarının kullanıldıkları yere göre birbirlerine üstünlükleri, belirli avantaj ve dezavantajları mevcuttur. Bu fonksiyonların üstünlüklerini sıralayacak olursak: Geniş aralıkta aktive olması dolayısıyla sınıflayıcı bir fonksiyon olan hiperbolik tanjant dikkat çekmektedir. Modelin daha yavaş öğrenmesi sorun olmayacak ise sigmoid fonksiyonu dikkat çekmektedir. Ağ yapısı çok derinse ve işlem yükü çok ise ReLU tercih edilmektedir. ReLU'da da gradyanların ölmesi sorunuyla karşılaşılmaktadır. Gradyanların ölmesi sorununa çözüm olarak ise Leaky ReLU önerilmekte, fakat ReLU'ya oranla daha çok işlem yapılmaktadır [83].

İlk kriter olarak ağın kolay ve hızlı yakınsaması tercih edilmesi durumunda; hız bakımından ReLU avantajlı sayılmakla birlikte gradyanların ölmesini göze almak gerekmektedir. Bu yüzden ReLU genellikle çıkış değil de ara katmanlarda kullanılmaktadır. Leaky ReLU gradyanların ölmesi problemine ilk çözüm olabilmektedir [83].

Derin öğrenme modelleri için ReLU ile denemelere başlanması uygun görülmektedir. Çıkış katmanlarında ise genellikle Softmax kullanılmaktadır [83].

Genel olarak bu bilgiler dikkat çekmesine rağmen net bir karar verilebilmesi için, bu bilgiler irdelendikten sonra derin öğrenme modelinin durumuna göre kullanılacak aktivasyon fonksiyonunun belirlenmesi kritik bir optimizasyon problemidir Bu nedenle daha net analiz yapılabilmesi için çalışmada 6 farklı aktivasyon fonksiyonu kullanılmaktadır.

4.5 Gradyan İniş Optimizasyonu Algoritmaları

Optimizasyon algoritmalarının amacı modelin kayıp fonksiyonunu yani tahmin edilen değerin gerçek değere olan farklı ölçüm yöntemlerini minimuma indirmektir. Modelin ağırlıklarını güncellemek için gradyan iniş yöntemi ve türev tabanlı yöntemler tercih edilmektedir. Yakınsama amacıyla üç gradyan iniş yöntemi olan toplu gradyan inişi, stokastik eğim inişi veya mini parti degrade inişi yöntemleri sıklıkla kullanılmaktadır. Fakat bunların iyi yakınsama yapacağı kesin değildir. Bu algoritmaları uygularken bazı zorluklar ile karşılaşılmaktadır. Bu durumun üstesinden gelmek için yaygın olarak kullanılan bazı gradyan iniş optimizasyon algoritmaları aşağıda sıralanmaktadır.

4.5.1 SGD

SGD, özellikle ağırlıklar ve sapma değişkeni gibi model parametrelerini ayarlayarak kayıp fonksiyonunu minimuma indiren bir optimizasyon algoritmasıdır. Kayıp fonksiyonu, tahmin edilen değerler ile gerçek değerler arasındaki farkı hesaplar. SGD, kayıp fonksiyonunu en aza indirecek şekilde model ağırlıklarını günceller. Fakat SGD, özellikle gürültülü güncellemeler ve karmaşık kayıp görüntüleri söz konusu olduğunda sıklıkla zorluklarla karşılaşmaktadır [131].

4.5.2 Adagrad

Adagrad, çok denk gelen özelliklere ait parametrelere düşük öğrenme oranı ile güncellemeler yapan, az denk gelen özelliklere ait parametrelere yüksek öğrenme oranı ile güncellemeler yapan bir yöntemdir. Bu yüzden verinin az olduğu durumlarda daha kullanışlıdır [132]. Bu yöntemde her parametrenin kendisine ait öğrenme hızı mevcuttur ve yöntemin özelliklerine göre gittikçe azalan bir öğrenme hızına sahiptir. Gittikçe azalan bu öğrenme hızı nedeniyle sistem bir süre sonra öğrenmeyi bırakır [86].

4.5.3 Adadelta

Bu yöntem Adagrad'ın bir uzantısıdır. Bu yöntem geçmiş gradyan kareleri değerlerini depolamaz. Bunun aksine geçmiş gradyan kareleri değerlerini sabit bir w değerine eşitler. Böylece yukarıda bahsedilen adagrad'ın azalan ve sonra yok olan öğrenme problemi engellenmiş olur [133].

4.5.4 RMSProp

Yakın geçmişte farklı ekiplerce adadelta yönteminin geliştirilmesi ile oluşturulan bir yöntemdir. Adagrad'ın azalan ve sonra yok olan öğrenme problemine çözüm olarak geliştirilmiştir [134].

4.5.5 Adam

Adam yöntemi her basamakta; Adadelta ve RMSProp yöntemleri gibi önceki gradyanların karelerinin ortalamalarıyla beraber momentum yöntemindeki gibi önceki gradyanların ortalamasını da depolar [135]. Bu yöntem momentum ve RMSProp yöntemlerinin birleşimi niteliğindedir [86].

4.5.6 Nadam

Nadam (Nesterov-hızlandırılmış uyarlanabilir moment tahmini), adam optimizasyon yöntemini ve nesterov hızlandırılmış gradyan (NAG) kavramını birleştiren bir optimizasyon algoritmasıdır. Nadam, Timothy Dozat'ın üretmiş olduğu bir yöntemdir. Adam, her parametre için uyarlanabilir öğrenme oranlarını hesaplayan bir optimizasyon yöntemi olmasına karşın NAG yöntemi gradyan vektörlerini doğru yönlerde hızlandırmaya çalışan ve bunun sonucunda da daha hızlı yakınsamaya yol açan bir yöntemdir [136].

4.5.7 Adamax

Adamax, Adam ve RMSProp yöntemlerini bir araya getiren bir optimizasyon yöntemidir. Adam gibi, birinci ve ikinci moment tahminlerini baz alarak model parametreleri için aktif bir güncelleme hesaplar. RMSProp gibi, geçmiş gradyanların karelerinin toplamına bir bozunma oranı ekler. Adamax, iyi yakınsama özellikleri ve uygulama kolaylığı nedeniyle çoğu derin öğrenme çalışmasında tercih edilmektedir. Bununla birlikte, hiperparametrelerin seçimine, özellikle de öğrenme oranına karşı hassasiyet göstermektedir [137].

Adagrad, RMSProp ve Adamax derin öğrenme çalışmalarında sıklıkla kullanılan optimizasyon yöntemleridir. Her birinin kendine özgü özellikleri olduğundan farklı veri ve farklı görevlerde kullanımları oldukça uygundur. Veri bilimciler, mevcut çeşitli optimize edicileri ve özelliklerini anlayarak hangi görevde hangi optimize edicinin kullanılacağı konusunda daha doğru karar verebilirler [137].

4.5.8 Ftrl

Ftrl, Google tarafından seyrek ve sığ sıralı veri kümeleri için tıklama oranını tahmin etmek amacıyla üretilmiş bir optimizasyon yöntemidir. Doğal olarak hızlı olmakla birlikte fazla sayıda hiper-parametre ayarlama seçeneği sunmaktadır. Bu nedenle çalışmalarda kayıp ve doğruluğu hesaplamak için birçok seçenek arasından doğru öğrenme oranının araştırılması ve ona göre ayarlanması gerekmektedir [138].

Fakely-Tibshirani-Ridge Learning (FTRL), öncelikle verilerin sıralı olarak geldiği çevrimiçi öğrenme durumları için öncelikli olarak üretilmiş bir optimizasyon yöntemidir. Özellikle doğrusal modellerde ve belirli genelleştirilmiş doğrusal model türlerinde daha yaygındır. CNN'ler genellikle SGD, Adam ve RMSprop ve diğer gradyan tabanlı optimizasyon algoritmaları kullanılarak eğitilmektedir. Bu optimizasyon yöntemlerinin çeşitli bilgisayarla görme çalışmalarında CNN'ler de dahil olmak üzere derin öğrenme modellerinin eğitilmesinde etkili oldukları kanıtlanmış olup yaygın olarak kullanılmaktadırlar. FTRL'nin CNN'lerle kullanılmamasının asıl sebebi, CNN'lerin çok sayıda parametreye sahip olmaları ve oldukça doğrusal olmayan, karmaşık bir optimizasyon ortamına sahip olmalarıdır. FTRL öncelikle çevrimiçi öğrenme ve doğrusal modeller için tasarlandığından, bu tür karmaşıklıklarla başa çıkmakta veya CNN'lerin yüksek boyutlu parametre uzayında verimli bir şekilde gezinmekte zorlanabilir [138].

4.6 Derin Öğrenmede Kullanılan Araçlar ve Programlama Dilleri

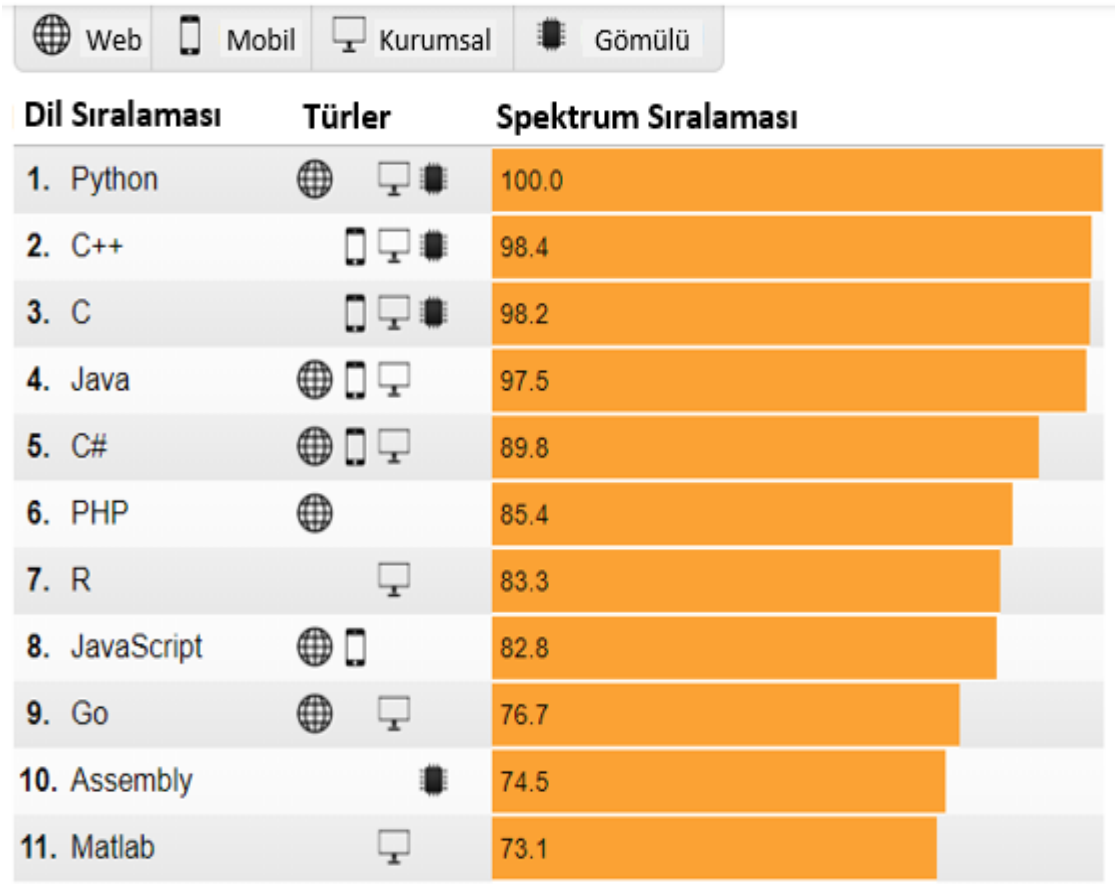
Yapay zekâ çalışmalarında kullanılmak üzere fazla sayıda açık kaynak kodlu, ücretsiz veya ücretli platform bulunmaktadır. Verilerin işlenmesinden, değerlendirme ve sonuç çıkarma kısmına kadar bu alanda kullanılabilecek platformun doğru seçimi çok önem arz etmektedir. Derin öğrenme uygulamalarında; veri setindeki hangi verilerin değerlendirileceği, veriler arasındaki bağlantıların tespit edilmesi ve hangi verilerin kullanılması durumunda en yüksek başarıya ulaşılacağı önemlidir. Bu konuda çok fazla deneme yapma ihtiyacı olabilir. Bu nedenle bu platformlar; tercih edilen yöntemlerden en çabuk bir şekilde sonuç alma, sonucu en verimli halde görsele aktarma, veri setine ait dosya ile ilgili değişiklik yapabilme gibi birçok yönden birbirlerine üstünlük sağlamaktadırlar. Bu üstünlüklere bakılırken fazla varyasyon olduğundan en faydalıyı belirlemek zorlaşmaktadır. Bu nedenle çalışmada en iyi sonucu verecek platformu tercih etmek; programcılara rahat erişim, zaman, doğru sonuç elde etme ve mali yönden fayda kazandıracaktır [139].

Veri bilimi çalışmalarında sıklıkla tercih edilen programlama dillerinden en önemli 5 tanesi Python (%57), C/C++ (%44), Java (%41), R (%37) ve JavaScript (%28) şeklinde sıralanabilir [126].

Python'un en çok tercih edilen dil olmasının nedeni yapay zekâ uygulamalarında, kolay programlanabilmesi ve geniş çevrimiçi topluluklarına sahip olmasıdır. 1990 yılından itibaren geliştirilen Python, kullanımı oldukça kolay, nesne yönelimli, başka bir derleyiciye ihtiyaç duymayan, yapısal ya da fonksiyonel programlama gibi çok sayıda programlama paradigması mevcut bir dildir. Yapay zekâ uygulamalarında kullanılan birçok kütüphanenin arka planında tercih edilen programlama dilidir [86]. Veri görselleştirme için ise en çok kullanılan dil R'dır. Ayrıca Python ticari ve akademik uygulamaların her ikisi için de kullanılabilen bir programdır [126].

IEEE Spectrum'un 2018 yılına ait bir araştırmasına göre en çok tercih edilen programlar ve kullanım oranları Çizelge 4.1'de görülmektedir.

Çizelge 4.1 En Çok Kullanım Oranına Göre Derin Öğrenme Programlama Dilleri [126]



Şüphesiz bu programlama dilleri dışında başka programlama dilleri de kullanılmamaktadır. En sık kullanılan 5 dilden sonra kullanılan programlar sırasıyla: Scala, Julia, Ruby, Octave, MATLAB ve SAS olarak sıralanabilir [126].

Yapay zekâ uygulaması geliştirmek için kullanılacak araçlar için de çok sayıda seçenek mevcuttur. Kişisel bilgisayarda kullanılmak üzere yukarıdaki dillerden yapılan seçime göre uygun IDE (Integrated Development Environment-Tümleşik Geliştirme Ortamı) tercih edilmelidir. Mesela Python programı tercih edilmişse Anaconda (Package Management Tool-Paket Yönetim Servisi) ve/veya Visual Studio Code, Java tercih edilmişse Eclipse sıklıkla kullanılmaktadır. Ücretsiz bulut ortamında ise, Microsoft Azure Notebook (sadece CPU) ve Google Colab (GPU ve TPU destekleri mevcut) herhangi bir kurulum gerektirmeksizin uygulamaların geliştirilmesini mümkün kılmaktadır [126].

4.7 Derin Öğrenmede Kullanılan Kütüphaneler ve Veri Kümeleri

Yapay zekâ uygulamalarında kullanılmak üzere farklı şirketler ve üniversiteler tarafından geliştirilen farklı özelliklerde çok sayıda API (Application Programming Interface-Uygulama Programlama Arayüzü) ve kütüphaneler mevcuttur [126]. Bunların en meşhur olanları başlıklar halinde sıralanmıştır.

4.7.1 Tensorflow

Tensorflow, derin öğrenme çalışmalarında sıkça kullanılan bir açık kaynaklı kütüphanedir. Google Brain Team, başlangıçta büyük hesaplamalar yapmak için Tensorflow'u oluşturmuş, yani özellikle derin öğrenme için oluşturulmamıştır. Fakat az bir zaman sonra Tensorflow'un derin öğrenme çalışmaları için faydalı olduğunu görüldüğünden o vakitten sonra Tensorflow açık kaynaklı bir çözüm haline gelmiştir [140].

Tensorflow bir dizi görev arasında veri akışı ve türevlenebilir programlama için kullanılan ücretsiz ve açık kaynaklı bir kütüphanedir. Daha çok sembolik matematik işlemleri ve sinir ağları gibi makine öğrenimi uygulamalarında tercih edilmektedir. Hem araştırma hem de üretim amacıyla Google'da sıkça tercih edilmektedir. Kurucu ve geliştiricisi Google Brain Team hem kurucusu hem de geliştiricisi olarak anılmaktadır [141].

Tensorflow, çoklu makine öğrenimi, derin öğrenme algoritmaları ve modellerini bir araya getirir. Python'u makine öğrenimi için kullanmanıza olanak tanır ve uygulamalar oluşturmak için bir front end API sunar [140].

Bu uygulamaları yürütmek ve yüksek performansla ve verimli bir şekilde kullanmak için çoğunlukla Python programlama dili tercih edilmektedir.

Tensorflow karmaşık matematik işlemleri yapmak için düşük seviyeli bir araç takımıdır ve deneysel öğrenme mimarileri oluşturmak, onlarla oynamak ve bunları çalışan bir yazılıma dönüştürmek için ne yaptığını bilen araştırmacıları hedefler. Tensorflow ile çeşitli makine ve derin öğrenme uygulamaları kolayca eğitebilir ve çalıştırabilir. Kelime yerleştirmeleri, el yazısıyla yazılmış rakamların sınıflandırılması, RNN'ler, görüntü tanıma, doğal dil işleme ve kısmi diferansiyel denklem simülasyonları Tensorflow ile yapılan çalışmalara örnek olarak verilebilir. Bu örnekler yanında Tensorflow, aynı modelleri eğitmek için kullanılabildiği için, üretim veya tüketim tahminini büyük ölçekte gerçekleştirmeye imkan tanır. Tensorflow, ücretsiz ve kullanımı kolay olmasının yanı sıra amatör düzeydeki makine öğrenimi kullanıcı ve geliştiricilerinin tüm AI modellerini sıfırdan oluşturmaları için güçlü bir kütüphaneye erişimi sağlaması özelliğiyle de oldukça popülerdir [140].

Data Tensorflow içinde n-boyutunda bir dizi olarak değerlendirilir. Bu n- boyutlu dizilere de tensör denir. Grafikler, bu tensörler ve matematiksel işlemlerden meydana gelmektedir [86].

4.7.2 Caffe

Caffe, en yeni derin öğrenme algoritmaları ve referans modelleri koleksiyonu için temiz ve dönüştürülebilir bir çerçeve sunmaktadır. Bu çerçeve, genel amaçlı evrişimli sinir ağlarını ve diğer derin öğrenme modelleri mimarilerini verimli bir şekilde eğiten ve dağıtan Python ve MATLAB bağlarına sahip BSD lisanslı bir C ++ kütüphanesidir. Caffe, tek bir K40 veya Titan GPU'da (görüntü başına yaklaşık 2 ms) günde 40 milyondan fazla görüntüyü işleyerek CUDA GPU hesaplamasıyla endüstri ve internet ölçeğinde medya ihtiyaçlarını karşılar. Model temsilini gerçek uygulamadan ayıran Caffe, prototipleme makinelerinden bulut ortamlarına geliştirme ve dağıtım kolaylığı için platformlar arasında deneme ve kesintisiz geçişe uygundur. Caffe, Berkeley Vision and Learning Center (BVLC) tarafından GitHub'da aktif bir katılımcı topluluğunun desteğiyle korunmakta ve geliştirilmektedir. Görme, konuşma ve multimedyada devam eden araştırma projelerine,

büyük ölçekli endüstriyel uygulamalara ve başlangıç prototiplerine destek vermektedir [142].

4.7.3 Torch

Torch Ronan Collobert, Koray Kavukçuoğlu ve Clement Farabet tarafından derin öğrenme algoritmalarında kullanılmak için tasarlanan bir Lua derin öğrenme çerçevesidir. NYU'daki CILVR Laboratuvarı tarafından kullanılmakta ve tanıtılmaktadır (Yann LeCun'un evi). Torch Facebook AI laboratuvarı, Google DeepMind, Twitter ve diğer pek çok kişi tarafından kullanılıyor ve geliştirilmektedir. Torch, CU / C ++ kütüphanelerinin yanı sıra GPU için CUDA'yı da kapsamaktadır. Çok fazla belge bulunmakta, ama bu bir karmaşa oluşturmaktadır. Torch popüler uygulamaları olan konvolüsyonel sinir ağları ve derin güçlendirme öğrenimi olan daha karmaşık alanlarda ajanlar ile denetimli görüntü sorunlarını çözmeye çalışır. Öncelikle takviye öğrenimi için Torch çok uygun bir platformdur [143].

4.7.4 Keras

Keras, derin öğrenme için yazılmış bir python kütüphanesidir. Keras derin öğrenmede kullanılan Tensorflow veya theano kütüphaneleri üzerinden çalışmaktadır. GPU ya da CPU üzerinde çalışması için bu kütüphanelere ihtiyaç duyar. Keras, Tensorflow ya da theano'a nispeten daha üst düzey bir kütüphane olduğundan bu kütüphanede uygulama yapmak daha kolaydır [144].

4.7.5 DeepLearning4j

Eclipse deeplearning4j java ve scala için yazılmış ilk ticari sınıf, açık kaynaklı, dağıtılmış derin öğrenme kütüphanesidir. Apache spark ve hadoop ile entegre olan DL4J, AI'yı dağıtılmış GPU'larda ve CPU'larda işlemek amacıyla iş ortamlarına ulaştırır [145]. DeepLearning4J kaygan bir platformdur ve derin inanç ağları, yığılmış gürültü giderici otomatik kodlayıcılar, evrişimli sinir ağları, uzun kısa süreli bellek birimleri ve tekrarlayan sinir ağları ile sınırlı olmamak üzere son teknoloji derin öğrenme algoritmaları sunar [143].

4.7.6 Numpy

Numpy, Python'daki en popüler makine öğrenimi kütüphanelerinden biridir. Numpy birden çok işlem gerçekleştirmek amacıyla Tensorflow ve diğer kütüphaneleri dahili olarak kullanır. Numpy'nin dizi arayüzü en iyi ve en önemli özelliğidir. Bu arayüz,

görüntüleri, ses dalgalarını ve diğer ikili ham akışları N-boyutlu gerçek sayılar dizisi olarak ifade etmek için kullanılmaktadır. Bu kütüphanenin makine öğreniminde kullanılması için Numpy bilgisine sahip olmak, tam yığın geliştiriciler için önemlidir [146].

4.7.7 Scikit-learn

Bir Python kütüphanesi Numpy ve Scipy ile irtibatlıdır. Karmaşık verilerle çalışmak için en iyi kütüphanelerden biri olarak kabul edilmektedir. Bu kütüphanede birçok değişiklik yapılmaktadır. Değişikliklerden biri, birden fazla metrik kullanma yeteneği sağlayan çapraz doğrulama yeteneğidir. Lojistik regresyon ve en yakın komşular gibi birçok eğitim yöntemi için küçük bazı iyileştirmeler yapılmıştır. Boyutsallığı sınıflandırmayı, regresyonu, kümelemeyi ve model seçimini azaltmak gibi standart makine öğrenimi ve veri madenciliği için çok sayıda algoritma içerir [146].

4.7.8 Pandas

Pandas, python'da yüksek seviyeli veri yapıları ve analiz için çok farklı araçlar sağlayan bir makine öğrenme kütüphanesidir. Bu kütüphanenin en önemli özelliklerinden biri, bir veya iki komut kullanarak karmaşık işlemleri verilerle çevirebilmesidir. Pandalar, zaman serileri işlevselliğinin yanı sıra gruplama, veri birleştirme ve filtreleme için birçok dahili yönteme sahiptir. Şu anda, yüzlerce yeni özellik, hata düzeltmeleri, geliştirmeler ve API'daki değişiklikler içeren panda kütüphanesinin fazla sürümü yoktur. Pandalardaki gelişmeler, verileri gruplama ve sıralama, uygulama yöntemi için en uygun çıktıyı seçme ve özel tür işlemlerini gerçekleştirme desteği sağlar. Veri analizi, pandaların kullanımı söz konusu olduğunda dikkat çeker. Ancak, pandalar diğer kütüphaneler ve araçlarla birlikte kullanıldığında yüksek işlevsellik ve iyi esneklik sağlar [146].

4.7.9 Statsmodels

Farklı veri türleri işlemesi, geniş tanımlayıcı istatistik yapısı, istatistiksel test çözümleri, istatistiksel veri incelemesi, çok sayıda farklı istatistiksel modelin tahmini çizim fonksiyonu ve sonuç istatistikleri listesi için bu Python kütüphanesinden sıkça yararlanılmaktadır. Her tahmin edici için kapsamlı bir sonuç istatistikleri listesi mevcuttur. Sonuçlar doğrulukları açısından mevcut istatistiksel paketlerle test edilmektedir. Paket, açık kaynaklı Değiştirilmiş BSD (3 maddeli) lisansı altında yayınlanmıştır. Kurucusu Wes McKinney ve geliştiricisi Ursa Lab'dır [141,147].

4.7.10 Matplotlib

Matplotlib, veri görselleştirme için en popüler ve en çok tercih edilen python kütüphanelerinden biridir. Çizgi grafikler, histogramlar, pasta grafikler, dağılım grafikleri, çubuk grafikler ve pek çok görselleştirme türlerini hazırlamayı mümkün kılar. Statik, etkileşimli ve animasyonlu grafikler üretme yeteneğine sahiptir. Python'un sayısal hesaplama kütüphanesi olan numpy ile güçlü bir entegrasyonu vardır [148].

Matplotlib, özelleştirilebilir yapısıyla eksen ayarları, renk düzenlemeleri, etiketlendirme ve farklı stil seçenekleri sunarak kullanıcıların görselleştirmelerini kendi istekleri doğrultusunda hazırlamalarını sağlar. Akademik araştırmalardan iş dünyasına, veri biliminden mühendislik uygulamalarına kadar geniş bir kullanım alanına sahiptir. Seaborn gibi diğer görselleştirme kütüphaneleri de matplotlib üzerine inşa edilmiştir, bu da onun veri görselleştirme dünyasındaki temel taşlardan biri olduğunu göstermektedir [148].

4.7.11 Seaborn

Seaborn oldukça etkin bir veri kütüphanedir. İstatistiksel analizlerin görselleştirilmesinde aktiftir. Matplotlib üzerine bina edilmesi sayesinde daha karmaşık ve estetik grafiklerin üretilmesini sağlayan üst düzey bir API sunar. Veri setlerinin iç yapısını ve ilişkilerini derinlemesine kavramaya yönelik araçlar sunar. Fonksiyonlar arasında pairplot, heatmap ve violin plot yer alır [149].

Çok boyutlu veri analizi ve örüntülerin keşfi amacıyla gelişmiş özellikler sağlar. PairPlot değişkenler arasındaki ilişkilerin ve dağılımların çok boyutlu görselleştirilmesine imkan sağladığından modelleme ve anomali tespit süreçlerinde ekstra fayda sunar. Isı haritası işlevi korelasyon matrisleri gibi yoğun veri yapılarını görselleştirerek veriler arasındaki ilişkilerin görsel olarak analiz edilmesini sağlar [149].

Bunlar dışında kullanılabilecek diğer kütüphaneler: SciKit-Learn, Accord.NET, Spark MLIib, Azure ML Studio, Amazon Machine Learning, H2O, DL4J ve NVIDIA-DIGITS olarak sıralanabilir [126].

Bunun dışında ücretsiz olarak ulaşılabilecek birçok veri seti sağlayıcısı da mevcuttur. Bu veri setlerinin meşhurları: UCI machine learning repository: data sets, kaggle datasets, google datasets tool olarak sıralanabilir [126].

Bu kütüphaneler kullanım yoğunluğuna göre sıralandığında aşağıdaki Çizelge 4.2 elde edilmektedir.

Çizelge 4.2 Kullanım Yoğunluğuna Göre Derin Öğrenme Kütüphaneleri [150]

Kütüphane	*Geliştirici*	*Programlandığı Dil*	*Özellikleri*
TensorFlow	Google	Python	-Hızlı derleme yapabilmektedir. -TensorBoard ile görselleştirme yapabilmektedir. -Veri ve model paralelliği sağlar. -GPU veya CPU'da paralel çalışabilmektedir.
Caffe	Berkeley Vision and Learning Center (BVLC)	Python	-İleri beslemeli ağlar ve görüntü işleme konularında hızlıdır. -Hassas ayarlama (finetuning) için önceden eğitilmiş modelleri vardır. -Hiçbir kod yazmadan model eğitilebilir. -Python arayüzü oldukça kullanışlıdır. GPU desteği vardır.
Caffe2	Facebook	Python	-Python API ile C++ desteği sağlar. Berkeley yazılım dağıtım lisansı vardır. -GPU desteği vardır.
Torch/ PyTorch	Google/ Facebook	Lua/ Python	-Birçok modüler parçayı birleştirmek kolaydır. -Yeni katmanları yazıp GPU üzerinde çalıştırması kolaydır. -Çokça önceden eğitilmiş model vardır.
Keras	Francois Chollet-Google	Python	-Torch kütüphanesinden esinlenilmiş sezgisel bir API'dir. -Theano, TensorFlow, Deeplearning4j ve CNTK arkaplanda kullanmaktadır. -Hızlı büyüyen bir yapısı vardır. -GPU veya CPU'da paralel çalışabilmektedir.
MxNet	Pedro Domingos, Amazon AWS	R, Python ve Julia	-Çoklu GPU desteği vardır. -Eğitim hızı ve verimliliği yüksektir. -Yeni katmanlar eklemek kolaydır.
CNTK	Microsoft	C++	-Geri planda Python API kullanımına izin verir.
DeepLearning4j	Adam Gibson	Java, Scala	-Java sanal makinesi (Java Virtual Envoirement-JVM) tabanlı olmasıdır.
KNet	Deniz Yüret, Koç Üniversitesi	Julia	-Kolay anlaşılır olması ve ifade gücü yüksektir. GPU Desteği vardır.
Theano	Montreal Institute for Learning Algorithms (MILA) Lab.	Python	-Eylül 2017'de resmi olarak Theano kütüphanesnin geliştirilmeye devam edilmeyeceği duyuruldu.

4.8 Hata Fonksiyonları

Hata fonksiyonları tahmin edilmeye çalışılan değer ile tahmin edilen değer arasındaki farklılığı niceliksel olarak belirlemeye çalışan yöntemlerdir. Kullanılan modellerin performansını ölçmek ve ağırlık değerini güncellemek için kullanılmaktadırlar. Kullanılan bazı hata fonksiyonları aşağıda açıklanmaktadır. Derin öğrenme çalışmalarında bu yöntemlerden ortalama karesel hata (MSE) kaybı, kök ortalama karesel hata (RMSE) kaybı ve ortalama mutlak hata (MAE) kaybı sıklıkla kullanılmaktadır.

4.8.1 Kategorik Çapraz Entropi Kaybı

Kategorik çapraz entropi, sınıflandırma problemlerinde yaygın olarak kullanılan ve sınıflar arasındaki farkı ölçmek için kullanılan bir yöntemdir. Bu yöntemle gerçek etiketler ile yapılan tahminler arasındaki çapraz entropi hesaplanmaktadır [151].

4.8.2 İkili Çapraz Entropi Kaybı

Binary crossentropy, ikili sınıflandırma problemlerinde kullanılmaktadır. Her sınıf için gerçek etiketler ile yapılan tahminler arasındaki çapraz entropiyi bağımsız bir şekilde ölçmektedir [151].

4.8.3 Seyrek Kategorik Çapraz Entropi Kaybı

Seyrek kategorik çapraz entropi, kategorik çapraz entropiye benzerdir fakat gerçek etiketler tek sıcak kodlama yerine tam sayılar olarak gösterildiğinde kullanılmaktadır [151].

4.8.4 Ortalama Karesel Hata (MSE) Kaybı

Ortalama karesel hata (MSE), sık kullanılan kayıp fonksiyonlarından biridir. Çalışmadaki her bir tahmin için gerçek değer ile tahmin arasındaki farkın karesi alınıp bu farklar toplandıktan sonra aritmetik ortalaması alınarak bulunan değerdir.

4.8.5 Kök Ortalama Karesel Hata (RMSE) Kaybı

Kök ortalama karesel hata (RMSE), MSE'den gelmektedir. RMSE, hata metriği hedef değişkenimizle aynı ölçekte ifade edildiğinden özellikle kullanışlıdır ve bu da burada gerçekten neler olup bittiğine dair daha sezgisel bir yorumlama sağlamaktadır [152].

Öncelikle çalışmadaki her bir tahmin için gerçek değer ile tahmin arasındaki farkın karesi alınıp bu farklar toplandıktan sonra aritmetik ortalaması alınıp, sonrasında bu sonucun karekökü alınarak elde edilen değerdir.

Avantajı MSE değerinin çıkış birimi ile aynı olması durumunda kaybın yorumlanmasının kolaylaşmasıdır. Dezavantajı ise aykırı değerlere karşı dayanıklı olmamasıdır [152].

4.8.6 Ortalama Mutlak Hata (MAE) Kaybı

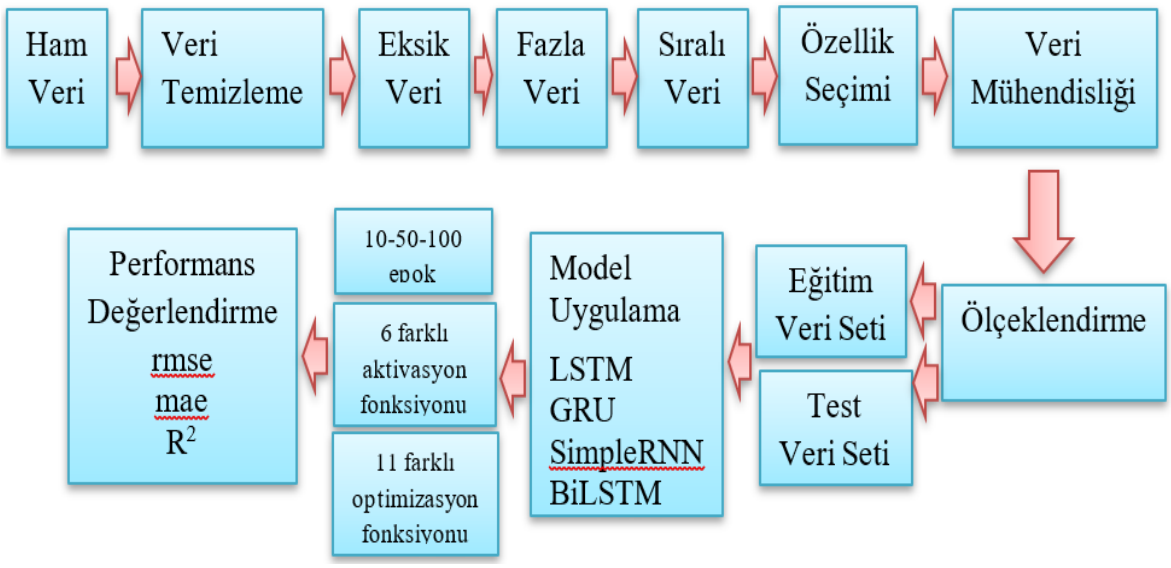
Ortalama Mutlak Hata (MAE), regresyon modeli hassasiyetinin yaygın olarak kullanılan bir diğer ölçüsüdür. MSE'den farklı olarak, MAE tahmin edilen ve gerçek değerler arasındaki mutlak farkların ortalamasını hesaplamaktadır. Yönlerini göz ardı ederek, göreceli büyüklük hatalarının ortalama bir ölçüsünü sunmaktadır [152].

Avantajları aykırı değerlere karşı dayanıklı olması ve aynı birimden olması gerektiğidir. Dezavantajları ise yakınsamanın genellikle daha fazla zaman alması ve optimizasyonda karmaşık süreçleri ele almasıdır [152].

5. MATERYAL VE METOD

Bu çalışmada, literatürde elektrik tüketiminin tahmin edilmesinde performansı test edilmiş farklı yapay zekâ yöntemlerinden makine öğrenmesinin bir alt dalı olan derin öğrenme algoritmaları kullanılarak tahmin çalışmaları yapılmaktadır. Yapılan çoğu çalışmada elektrik enerjisi talep, üretim veya tüketim tahmini gibi ortak çalışmalar yapıldığı görülmektedir. Fakat bu konuda henüz genelleştirilebilir yöntemlere ulaşılamamış, hatta bazı yöntemler benzer veri setlerine uygulandığı halde farklı sonuçlara ulaşılmıştır. Yani yapılacak yeni ve farklı uygulama ve denemelere sıklıkla ihtiyaç duyulmaktadır. Ayrıca henüz net çalışılmamış birçok alan bulunmakta ve geliştirilmeyi beklemektedir. Bu yüzden veri setleri için model önerisi yapmak amacıyla farklı tekniklerin karşılaştırılmasına ve farklı modellerin üretilmesine ihtiyaç vardır. Bazı çalışmalarda veri seti düzenlenmeden çalışmalar yapılmış ve hiper parametreler için daha önce başarılı görülmüş klasik yöntemler kullanılarak kısıtlara gidilmiştir. Bu kısıtlar ele alınarak modellerin geliştirilmesi yapılacak çalışmanın en önemli ve farklı kısmını oluşturacaktır.

Yapılan bu tez çalışmasının adımları ve akış diyagramı genel hatlarıyla Şekil 5.1'de gösterilmektedir.



Şekil 5.1 : Yapılan çalışmanın akış diyagramı.

Tezin amacı doğrultusunda yapay zeka alanında makine öğrenmesi ve alt dalı olan derin öğrenme modelleri yardımıyla saatlik tahminler yapılacaktır. Bu çalışmada,

ABD'deki bölgesel bir iletim organizasyonu olan PJM'ye ait 17 yıllık (01.01.2002-03.08.2018) megavat cinsinden saatlik enerji tüketim verileri kullanılacaktır.

Öncelikle veriler incelenmekte ve düzenlenmektedir. Fazla olan veriler çıkarılmakta eksik olan veriler tamamlanmaktadır.

Tüketim değerleri dışında yıl, ay, hafta, gün ve saat değerlerine ait yeni sütunlar oluşturulmaktadır. Bu değerler giriş değerlerimizin oluşturulmasında kullanılmaktadır. Bu şekilde 5 sütun ve tüketim değeriyle birlikte 6 sütuna ait önceki 24 saate ait değerler giriş 25. saate ait tüketim değeri de çıkış değeri olarak kullanılmaktadır.

5.2 bölümde detaylandırıldığı üzere gün öncesi piyasasındaki her bir güne ait 24 saatlik veriler alınmakta ve sonraki saat tüketim tahmininde kullanılmaktadır. Yani 24 saatlik geçmiş veri ile 25. saati tahmin edilmesi amaçlanmaktadır. (n_hours = 24 ile) Bunun için hazır bir fonksiyondan faydalanarak gözetimli öğrenme metodu uygulanmaktadır. Böylece tek sütun olan tüketim verilerinin 6 sütuna çıkarılması ve 24 saatlik örnek değerler alınınca 144 sütuna erişmesi amaçlanmaktadır. Böylece veri setinin datatime sütunu ile birlikte 145 sütun ve 145368 satırdan oluşması amaçlanmaktadır. Bu teknikle basit tek bir sütundan oluşan veri çok katmanlı algılayıcı (MLP) modeline uyarlanmaktadır.

Normal çalışmalarda veriler iki kısım olacak şekilde birinci kısmı eğitim sürecine tabi tutulacak (%80) ikinci kısmı (%20) eğitilen bu modellerin test başarımını ölçmek için kullanılacaktır. Bu şekilde, eğitim seti ile modeller eğitilecek, eğitilen modeller test verileri üzerinden tahmin yapılarak performansları gözlemlenecektir.

Yöntem olarak ise derin öğrenme alanında zaman serileri ile çalışmak için tasarlanan bir model olan RNN, RNN'nin bir alt dalı olan LSTM, GRU ve BiLSTM mimarileri kullanılmakta ve sonuçlar karşılaştırılmaktadır. Yapılan çoğu çalışmada veri seti ve hiper parametreler için kısıtlara gidildiği görülmektedir. Bu kısıtlar ele alınarak modellerin geliştirilmesi ve bu hiper parametrelerin değiştirilerek sonuçların karşılaştırılması da yapılacak bu çalışmanın ana konularından birini oluşturmaktadır. Bu amaçla her model için farklı aktivasyon fonksiyonları ile denemeler yapılmaktadır. Her model için 10, 50 ve 100 epok denemeleri yapılmakta ve sonuçlar yorumlanmaktadır. Ayrıca her model için farklı aktivasyon fonksiyonları ve optimizasyon yöntemleri kullanılarak modeller arasında karşılaştırmalı bir analiz yapılmaktadır. Son olarak da benzer veri setleri kullanılarak yapılan 3 çalışma ile yaptığımız çalışma meta-analiz

metoduyla karşılaştırılmakta ve 50 epok ile yapılan tüm denemelerde en yüksek başarımı sağlayan 10 model sunulmaktadır.

Literatür çalışmalarında görüldüğü üzere tüm derin öğrenme modelleri ile çalışmalar mevcuttur. Yani akıllı şebeke ile ilgili yapılacak olan çalışmalarda tüm modellerin kullanılması olasıdır. Nesne tanıma alanlarında CNN ve sıralı veya dizi halindeki verilerin değerlendirilmesinde RNN modelleri daha öne çıkmaktadır. Fakat RNN modellerinin eski bilgileri hatırlayıp değerlendirmede sorun yaşaması yüzünden yine bir RNN modeli olan LSTM (hafızalı RNN) tercih edilmektedir. LSTM yöntemi özellikle tahmin etme konusunda çok başarılı çalışmalara sahne olduğu için tahmin problemlerinde çokça kullanılmaktadır. Özellikle yük tahmini veya üretilecek enerji tahmininde son yıllarda çokça çalışma yapılmaktadır. Ayrıca yine son yıllarda verinin az olduğu durumlarda derin öğrenme mimarilerinin başarımı azalacağından bu konuda özellikle DRL mimarisi önerilmekte ve başarılı sonuçlar alınmaktadır.

5.1 Veri Analizi

5.1.1 Veri

Bu tez çalışmasında kullanmak amacıyla, Kaggle'ın ücretsiz olarak paylaşmış olduğu PJM Interconnection LLC (PJM)'ye ait 2002-2018 yıllarına dair Saatlik Enerji Tüketim verileri kullanılmaktadır. Bu veri seti, Delaware, Illinois, Indiana, Kentucky, Maryland, Michigan, New Jersey, Kuzey Karolina, Ohio, Pensilvanya, Tennessee, Virginia, Batı Virginia ve Columbia Bölgesi'ne elektrik sağlayan ABD'deki bölgesel bir iletim organizasyonu olan PJM'ye ait Saatlik Enerji Tüketimi verileridir. PJM Interconnection LLC'ye ait veri kümesi (PJME), 145.366 satırdan ve 2 özellikten oluşan bir zaman serisi veri seti sağlamaktadır. Tüketim verileri megawatt (MW) cinsindendir [153].

5.1.2 Keşifsel veri analizi

Keşifsel Veri Analizi (EDA), olasılık ve istatistiksel yöntemlerden faydalanarak mevcut verilerden ilgili bilgilerin elde edilmesi için bu analizlerin anlaşılır ve kolay bir şekilde sunulması, özetlenmesi ve grafiklerle sunulmasıdır. Bu tarz çalışmalarla özellik mühendisliğinin temellerinin atılmasını sağlanır. Yani, modelin mümkün olan en verimli şekilde çalışabilmesi için veri setinden özellikler oluşturma, dönüştürme ve çıkarımlar yapmayı sağlayan bir çalışmadır.

Öncelikle veri setinin en önemli özelliklerini çıkaran ve özetleyen ve zaman serilerine odaklanan net bir keşifsel veri analizi şablonu oluşturulmaktadır. Bunun için Pandas 1.4.4, Matplotlib 3.6.2, Numpy 1.19.2, Seaborn 0.12.2 ve Statsmodels 0.13.5 gibi bazı yaygın Python kütüphanelerinden 3.8.20 faydalanılmaktadır.

Zaman serileri ile çalışırken en başta yapılması gereken çalışma verilerin tipini, değerlerini, hatalı veya eksik değer olup olmadığını kabaca gözlemlemek ve kontrol etmektir. Sonra yapılması gereken önemli şeylerden biri verileri çizmektir. Böylece grafikler, desenler, alışılmadık gözlemler, zaman içindeki değişimler ve değişkenler arasındaki ilişkiler gibi birçok özellik buradan gözlemlenebilmektedir. Bu gözlemlerden çıkan sonuçlar mümkün olduğunca tahmin modeline dahil edilmektedir. Ayrıca, tanımlayıcı istatistikler ve zaman serisi ayrıştırma gibi bazı matematiksel araçlar da bize çok yarar sağlamaktadır [154,155].

EDA altı adımdan oluşmaktadır:

- Tanımlayıcı İstatistikler,
- Zaman Grafiği,
- Mevsimsel Grafikler,
- Kutu Grafikleri,
- Zaman Serisi Ayrıştırma,
- Gecikme Analizi.

Kullanılan python programımızda EDA için uygulanabilen tüm 6 adım uygulamakla birlikte burada EDA için uygulanan ilk 3 adıma ait tanımlayıcı istatistikler, yıllık, aylık, haftalık, günlük ve saatlik bazda zaman grafiklerinin çizilmesi ve mevsimsel grafiklerin çizilmesi ile ilgili çalışmalar ve çıkarımlar yapılacaktır.

5.1.2.1 Tanımlayıcı istatistikler, eksik verilerin tamamlanması ve fazla verilerin çıkarılması

Verilerimize ait ilk 5 satırlık değerler Çizelge 5.1'de görülmektedir. Çizelge 5.1 (a)'dan görüleceği üzere zaman serisine ait değerler yıllara ve saatlere göre sıralanmış fakat ay ve günlere göre sıralanmamıştır. Yani yıl ve saat değerleri küçükten büyüğe doğru sıralanmış olmasına rağmen ay ve gün değerleri büyükten küçüğe doğru sıralanmaktadır. Dolayısıyla tarihler veya datatime denilen zaman serisi verileri karışık bir hal almaktadır.

Yaptığımız düzeltmelerle artık tarihler veya zaman serisi sütunu geçmişten geleceğe doğru düzgün sıralanmaktadır. Ayrıca veri başlık satırı hariç 145366 satır ve 2 sütundan oluşmaktadır. Zamana göre yapılan incelemeler sonucunda 30 adet satırın eksik olduğu, 4 adet satırın ise tekrarlandığı gözlemlenmektedir. Eksik ve sırasız veriler hem puant saatleri hem de tatil saatlerini kaydıracaktır. Bu nedenle eksik değerler ortalama değer hesaplanıp eklenmiştir. Fazla satırlardan ise daha uygun görülen seçilip diğer satır silinmiştir. Verilerin düzenlenmiş hali Çizelge 5.1 (b)'de görülmektedir.

Çizelge 5.1 (a) Düzenlenmemiş ve (b) Düzenlenmiş veriye ait ilk ve son 5 satır.

(a)

Satır No	Zaman Serisi	PJME_MW
1	2002.12.31 01:00	26498.0
2	2002.12.31 02:00	25147.0
3	2002.12.31 03:00	24574.0
4	2002.12.31 04:00	24393.0
5	2002.12.31 05:00	24860.0
…..	……	…..
145362	2018.01.1 20:00	44284.0
145363	2018.01.1 21:00	43751.0
145364	2018.01.1 22:00	42402.0
145365	2018.01.1 23:00	40164.0
145366	2018.01.2 00:00	38608.0

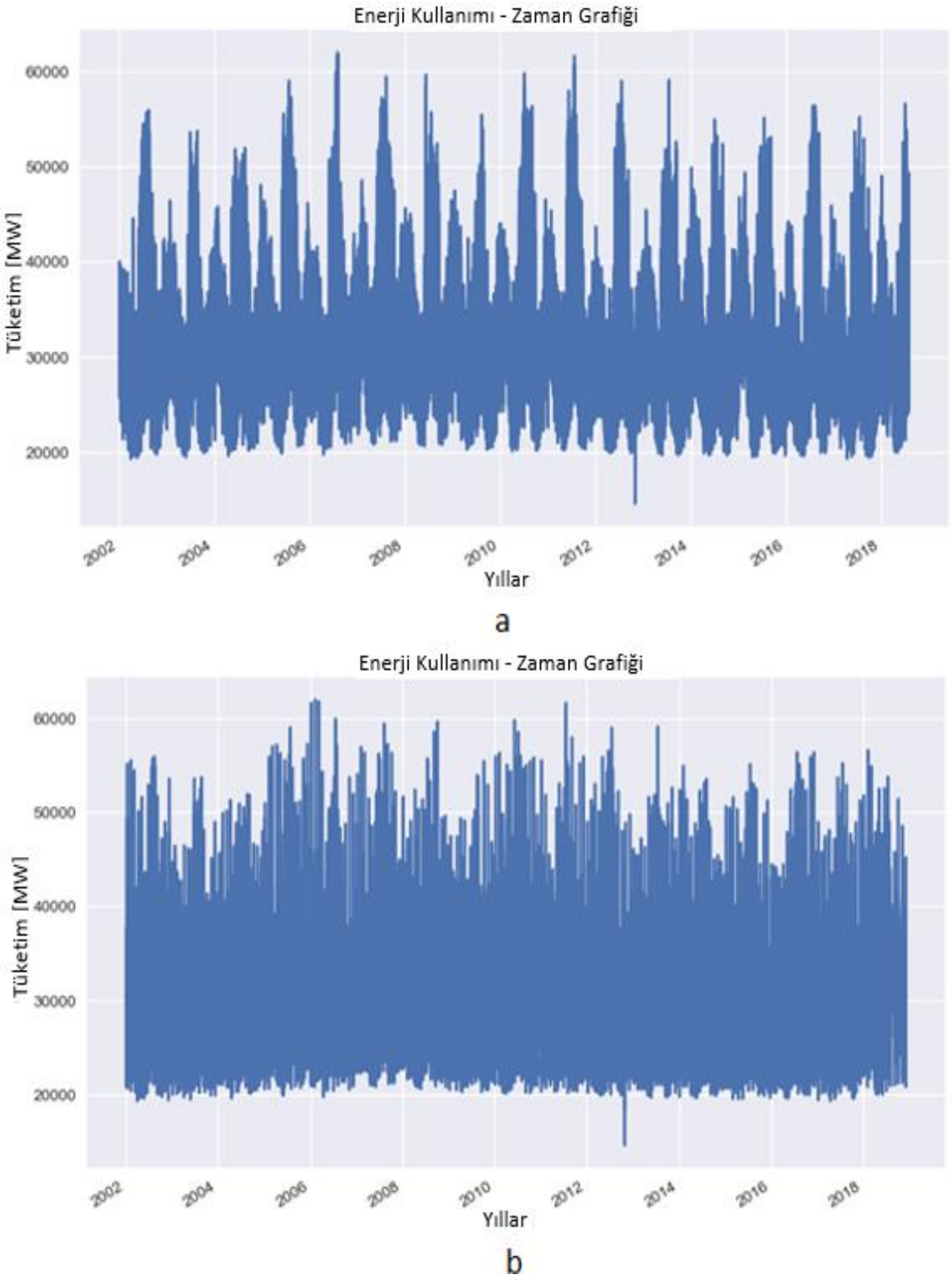
(b)

Satır No	Zaman Serisi	PJME_MW
1	1.01.2002 01:00	30393.0
2	1.01.2002 02:00	29265.0
3	1.01.2002 03:00	28357.0
4	1.01.2002 04:00	27899.0
5	1.01.2002 05:00	28057.0
…..	……	…..
145388	2.08.2018 20:00	44057.0
145389	2.08.2018 21:00	43256.0
145390	2.08.2018 22:00	41552.0
145391	2.08.2018 23:00	38500.0
145392	3.08.2018 00:00	35486.0

5.1.2.2 Zaman grafiği

Şekil 5.2.a'da ham verilere göre, Şekil 5.2.b'de düzenlenmiş verilere göre tüketimin zamana bağlı dağılımı görülmektedir. Burada eski ham verilerin bizi yanıltacağı görülmektedir. Ayrıca eksik ve sıralanmamış verilerin hem puant saatleri hem de tatil saatlerini değiştireceği değerlendirilmektedir. Şekil 5.2.b'de yıllar itibarıyla belirgin bir

artış/azalış eğilimi görülmemekte, ortalama tüketimler Şekil 5.2.a'ya kıyasla daha durağan kalmaktadır.



Şekil 5.2 : Düzenlenmemiş ham veri (a) ve düzenlenmiş veri (b).

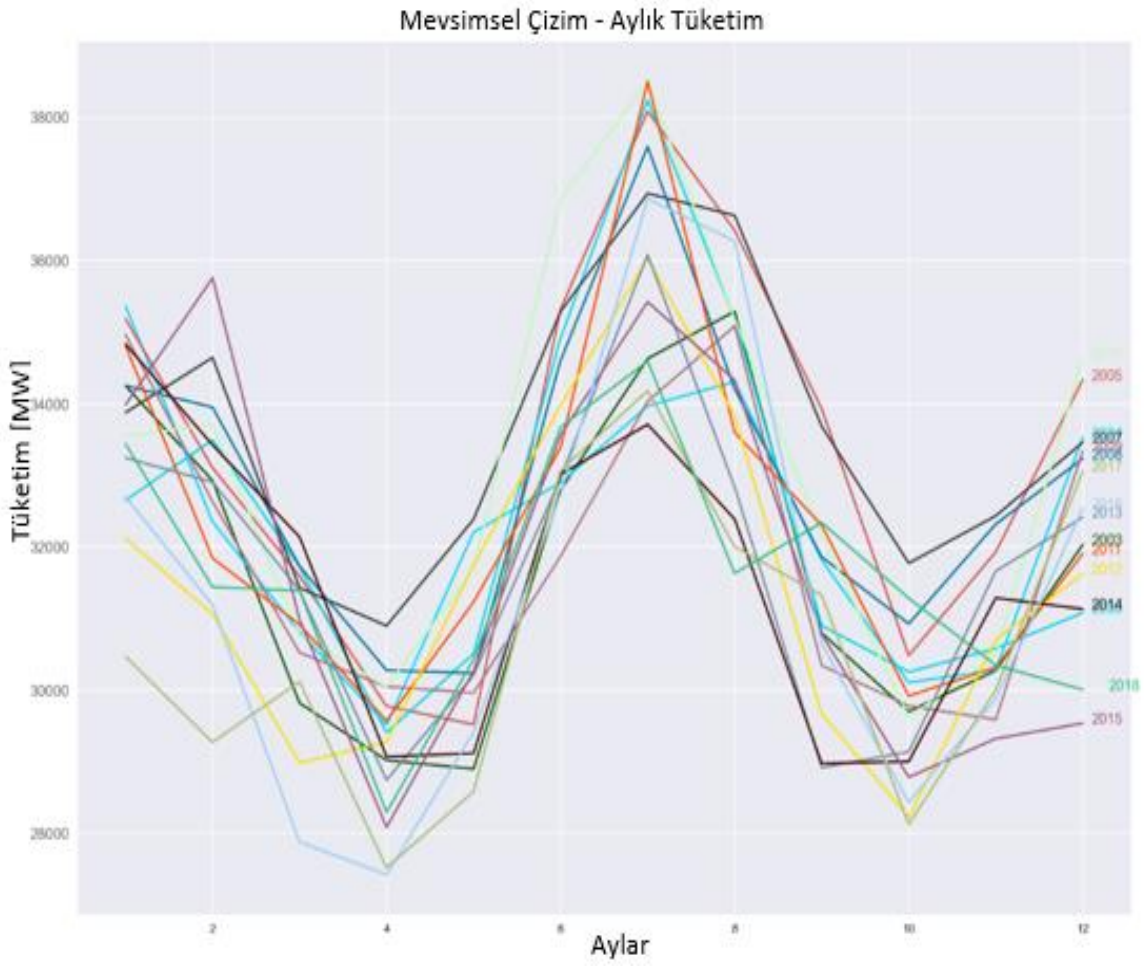
5.1.2.3 Mevsimsel grafikler

Grafikler yıllık, aylık, haftalık ve günlük olarak çizdirildiğinde mevsimsellik denilen tekrarlanan olaylar veya farklılıklar daha net ortaya çıkmaktadır.

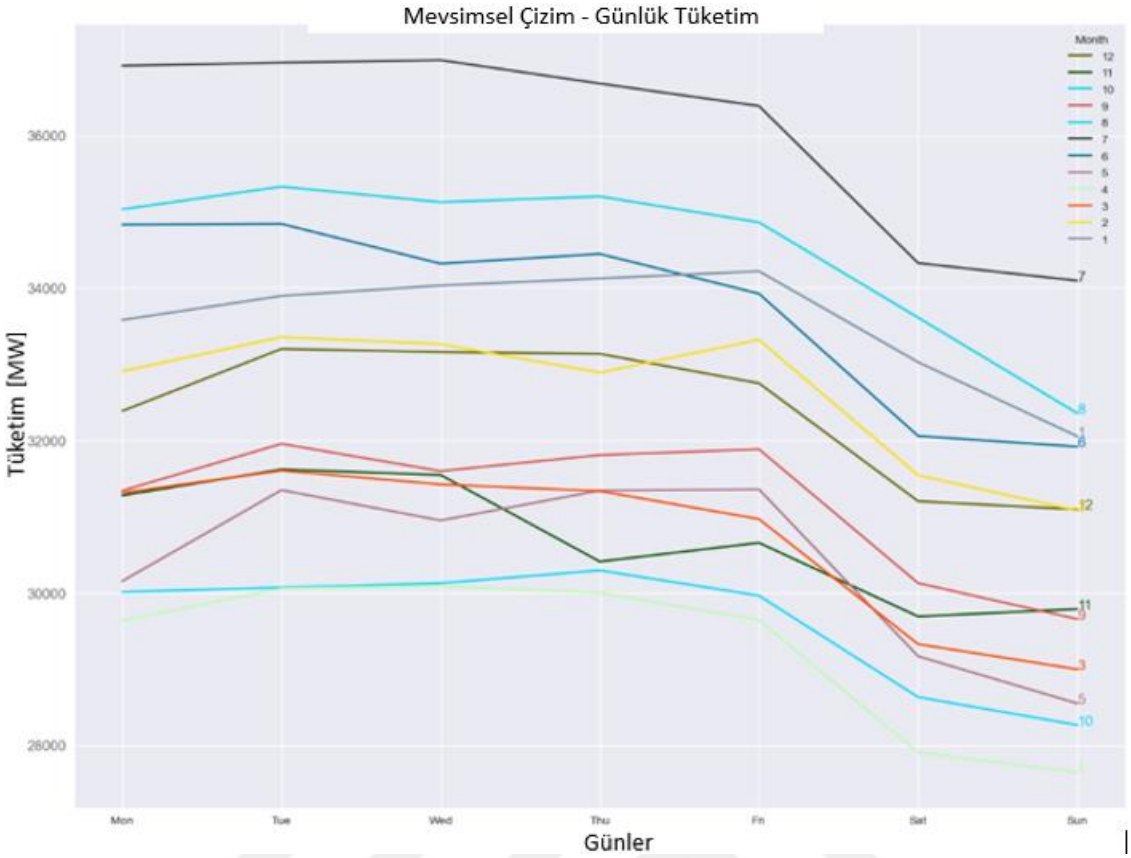
Şekil 5.3‘e göre tüm yılların benzer desenlere sahip oldukları görülmektedir. Tüketim değerlerinin kışın ve yazın (ısıtma/soğutma amaçlı) önemli ölçüde arttığı ve yazın en yüksek seviyelere çıktığı görülmektedir. Fakat ılıman olan ilkbahar ve sonbaharda ise tüketimin minimuma indiği görülmektedir.

Şekil 5.3 ve Şekil 5.4’ten görüldüğü üzere tüketim değerleri 7. ve 8. aylarda maksimum 4. ve 10. aylarda ise minimum değerleri almaktadır.

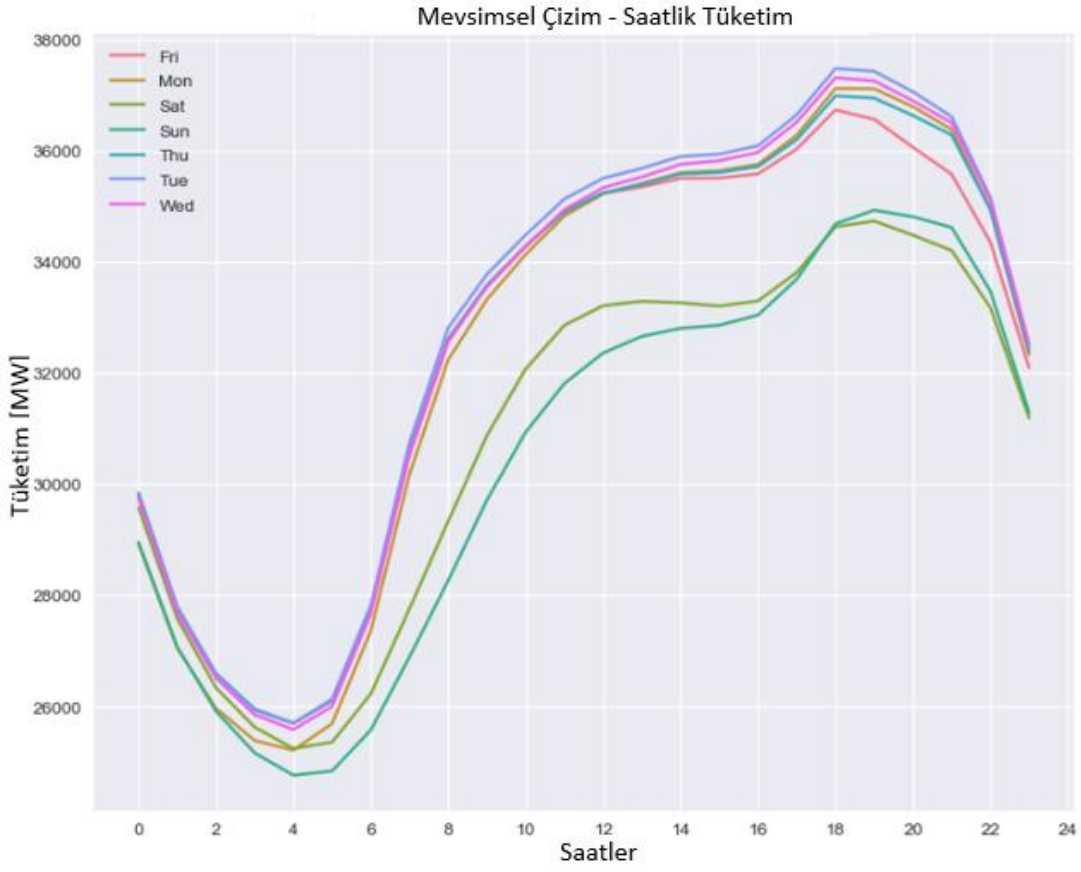
Şekil 5.5’ten hafta içi tüketimin hafta sonu tüketiminden daha fazla olduğu görülmektedir. Ayrıca en yüksek fark gün içi saatler arasında meydan gelmektedir. En fazla tüketim 18:00, 19:00’ da (puant saatleri) görülürken en az tüketim saat 3:00, 4:00 ve 5:00 te (gece yarısı) görülmektedir.



Şekil 5.3 : Yıllara göre aylık tüketim grafiği.



Şekil 5.4 : Aylara göre günlük tüketim grafiği.



Şekil 5.5 : Günlere göre saatlik tüketim grafiği.

Derin öğrenme çalışmasında kullanılacak zaman serileri için dikkate alınması gereken değerler mevsimsel davranış gösteren ve en fazla değişkenlik gösteren değerlerdir. Bu nedenle çalışmamızda örnek alınması gereken değerler veya zaman aralıkları son 24 saatlik değerler olarak görülmektedir. Çünkü Şekil 5.2, 5.3, 5.4 ve 5.5'te yıllık aylık haftalık ve günlük değerlere bakıldığında verilerin benzer davranışlar gösterdiği ve değerler arasında aşırı fark olmadığı görülmektedir. Ancak gün içinde saatlik değerler arasında maksimum fark görülmektedir. Ayrıca örneğin önceki gün 11:00'a ait tüketim değeri ile sonraki gün 11:00'a ait tüketim değeri benzerlik göstermektedir. Dolayısıyla o güne ait tüketim tahmini yapılırken en benzer değerler olarak önceki güne bakılmalıdır.

Şekil 5.2, 5.3, 5.4 ve 5.5'e göre yıl, ay, hafta, gün ve saatin tüketim değerlerini etkilediği görülmektedir. Dolayısıyla tüketim tahmini yapılırken bu bilgiler de sütunlara dönüştürülmeli ve tahminleme yapılırken kullanılmalıdır. Burada giriş değerimiz zaman serisi sütunu, çıkış değerimiz ise PJME_MW sütunudur. Ancak çalışmadaki zaman serisi verisi yıl, ay, gün ve saat olarak sütunlara bölünmekte ve hafta sütunu da program vasıtasıyla oluşturulmaktadır. Böylece analiz edilecek sütunlar; yıl, ay, hafta, saat, gün ve PJME_MW sütunları olarak 6 sütundan oluşmaktadır. Bu duruma göre verinin son halinin ilk 5 satırı Çizelge 5.2'de görülmektedir.

Çizelge 5.2 Düzenlenmiş ham veriye ait ilk 5 satır.

Zaman serisi	PJME_MW	Yıl	Ay	Hafta	Saat	Gün
1.01.2002 01:00	30393.0	2002	1	1	1	1
1.01.2002 02:00	29265.0	2002	1	1	2	1
1.01.2002 03:00	28357.0	2002	1	1	3	1
1.01.2002 04:00	27899.0	2002	1	1	4	1
1.01.2002 05:00	28057.0	2002	1	1	5	1

5.2 Derin Öğrenme Çalışması

Derin öğrenme modelinde kullanılacak bazı parametreler aşağıda görülmektedir.

```
SEED = 123
batch_size = 128
return_sequence = True
target = 'PJME_MW'          #Çıkış. Tek bir boyut alındı, birden fazla alınabilirdi.
n_hours = 24                #Kaç saat önceki değerlerin alınıp değerlendirileceği.
train_size = 0.8            #Verinin %80'i eğitim %20'si test için kullanılacak
cols_to_analyze = ["PJME_MW", "year", "month", "week", "day", "hour"]
```

Her seferinde aynı değerleri üretebilmek için sabitleme değeri olan seed i daha önce belirlenen seed değerine yani 123'e eşitliyoruz.

Sonrasında derin öğrenmede önemli olan bir adım verilerin scale edilmesi yani ölçeklendirilmesidir. Bu amaçla verileri minmaxscaler fonksiyonu ile 0-1 arasında ölçeklendirdik, normalize ettik.

Kısa dönemlik yani gelecek saate veya birkaç ssat sonrasına ait kısa dönemlik enerji tüketim değerini tahmin etmeye çalışıyoruz. Elbette ki bir önceki günün enerji tüketim değerleri bizim için önemli olacaktır. Grafiklerden de aynı gün veya 24 saat içinde değerlerin en yüksek ve en düşük değerlere ulaştığı, yani en büyük farkın aslında 24 saatte gerçekleştiği görülmektedir. Geçmiş değerlere gecikme adı verilir. t anındaki değer, t-1 anındaki değerden büyük ölçüde etkilenir. Bu uygulamaya özellik mühendisliği alanında çerçeveleme denir. Yine gün öncesi piyasasındaki her bir güne ait 24 saatlik veriler alınmakta ve sonraki saat tüketim tahmininde kullanılmaktadır. Başka bir deyişle, amaç 24 saatlik geçmiş veriyle (n_hours = 24) 25. saati tahmin etmektir. Daha sonra, geçmiş 2-25 saatten 26. saati tahmin eden bir çerçeveye ihtiyacımız vardır. Daha sonra, geçmiş 3-26 saatten 27. saati tahmin eden bir çerçeveye ihtiyacımız vardır. Bu şekilde devam eden bir çalışma yapılmaktadır. Böylece bir gözetimli öğrenme adımı uygulanması ve başarımın artırılması amaçlanmaktadır. Çizelge 5.2'den görüldüğü üzere bu yöntem ile veri tek sütundan, sadece tüketim verilerinden oluşurken yeni haliyle yıl, ay, hafta, saat, gün ve tüketim değerleri olarak 6 sütuna çıkarıldığından ve her bir sütuna ait son 24 saatlik değerler örnek alınacağından verinin yeni haliyle 144 sütuna erişmesi amaçlanmaktadır. Böylece veri setinin zaman serisi denilen datatime sütunu ile birlikte 145 sütun ve 145368 satırdan oluşması amaçlanmaktadır. Bu teknikle basit tek bir sütundan oluşan verinin birçok katmanlı algılayıcı (MLP) modeline uyarlanması düşünülmektedir. Başka bir ifadeyle, her 24 saatlik değere göre bir sonraki saat tahmini yapılmakta ve öğrenme katsayıları bu sonuca göre tekrar tekrar güncellenerek en doğru tahmin yapılmaya

çalışılmaktadır. Bu çalışmaya ait 6 sütundaki yıl (var1), ay (var2), hafta (var3), gün (var4), saat (var5) ve PJM tüketim değerine (var6) ait son 24 saatlik değerler (t-1'den t-24'e kadar) Çizelge %.3'te görülmektedir. Çizelge 5.3 yatay olarak sayfaya yerleşemediğinden devam kısmı alta eklenmiştir.

Çizelge 5.3 Analiz edilecek 6 sütuna ait son 24 saatlik veriler.

Zaman Serisi	var1(t-24)	var2(t-24)	var3(t-24)	var4(t-24)	var5(t-24)	var6(t-24)	var1(t-23)	var2(t-23)	var3(t-23)	var4(t-23)	...
1.02.2002 01:00	0.333909	0.0	0.000000	0.000000	0.166667	0.043478	0.310144	0.0	0.000000	0.000000	...
1.02.2002 02:00	0.310144	0.0	0.000000	0.000000	0.166667	0.086957	0.291014	0.0	0.000000	0.000000	...
1.02.2002 03:00	0.291014	0.0	0.000000	0.000000	0.166667	0.130435	0.281365	0.0	0.000000	0.000000	...
1.02.2002 04:00	0.281365	0.0	0.000000	0.000000	0.166667	0.173913	0.284694	0.0	0.000000	0.000000	...
1.02.2002 05:00	0.284694	0.0	0.000000	0.000000	0.166667	0.217391	0.297272	0.0	0.000000	0.000000	...
...	...	...	...	...	...	...	...	...	...	...	...
8.02.2018 20:00	0.681934	1.0	0.000000	0.019231	0.000000	0.869565	0.662404	1.0	0.000000	0.019231	...
8.02.2018 21:00	0.662404	1.0	0.000000	0.019231	0.000000	0.913043	0.622564	1.0	0.000000	0.019231	...
8.02.2018 22:00	0.622564	1.0	0.000000	0.019231	0.000000	0.956522	0.550342	1.0	0.000000	0.019231	...
8.02.2018 23:00	0.550342	1.0	0.000000	0.019231	0.000000	1.000.000	0.476435	1.0	0.090909	0.096154	...
8.03.2018 00:00	0.476435	1.0	0.090909	0.096154	0.500000	0.000000	0.415864	1.0	0.090909	0.096154	...

...	var4(t-2)	var5(t-2)	var6(t-2)	var1(t-1)	var2(t-1)	var3(t-1)	var4(t-1)	var5(t-1)	var6(t-1)	var1(t)
...	0.000000	0.166667	1.000.000	0.316423	0.0	0.090909	0.076923	0.666667	0.000000	0.286042
...	0.076923	0.666667	0.000000	0.286042	0.0	0.090909	0.076923	0.666667	0.043478	0.271632
...	0.076923	0.666667	0.043478	0.271632	0.0	0.090909	0.076923	0.666667	0.086957	0.268766
...	0.076923	0.666667	0.086957	0.268766	0.0	0.090909	0.076923	0.666667	0.130435	0.273654
...	0.076923	0.666667	0.130435	0.273654	0.0	0.090909	0.076923	0.666667	0.173913	0.292026
...	...	...	...	...	...	...	...	...	...	...
...	0.096154	0.500000	0.782609	0.655156	1.0	0.090909	0.096154	0.500000	0.826087	0.621784
...	0.096154	0.500000	0.826087	0.621784	1.0	0.090909	0.096154	0.500000	0.869565	0.604909
...	0.096154	0.500000	0.869565	0.604909	1.0	0.090909	0.096154	0.500000	0.913043	0.569009
...	0.096154	0.500000	0.913043	0.569009	1.0	0.090909	0.096154	0.500000	0.956522	0.504709
...	0.096154	0.500000	0.956522	0.504709	1.0	0.090909	0.096154	0.500000	1.000.000	0.441209

Çalışmada modellerin ezberleyip ezberlemediğini anlama konusunda da bazı yöntemler kullanılmaktadır. Çapraz doğrulama ve kayan pencere ayrımı bu yöntemlerden ikisidir. Çapraz doğrulamada, veriler birkaç alt kümeye ayrılmakta ve makine öğrenimi modelinin performansı değerlendirilmektedir. Verileri eğitim ve test verileri olarak ayırmak bu amaca hizmet etmektedir. Ayrıca, 6 sütundan son 24 saatlik verileri alıp bir sonraki saati tahmin etmeye çalışılmaktadır. Aslında, her tahmin için veriler değiştirilmekte ve çapraz doğrulama gerçekleştirilmektedir. Ayrıca, her tahminde kullanılan verileri birer saat kaydırılarak bir saatlik kayma ile sonraki 24 saatin değerleri kullanılarak kayan pencere yöntemi kullanılmaktadır.

Çalışmanın en önemli kısmını oluşturan giriş ve çıkış değerleri için zaman serisi sütunu giriş, tüketim değerleri çıkış olarak görülmekte iken sütunların arttırılması ve çerçeveleme sonucunda durumun karışık bir hal aldığı görülmektedir. Bu yüzden giriş ve çıkışın ne olduğunun daha net ifade edilmesi gerekmektedir. Çizelge 5.2 de 6 sütunun son 24 saatteki değerleri görülmektedir. Daha açık bir ifadeyle, Çizelge 5.2'deki mevcut son sütun hariç tüm sütunlar giriş olarak kullanılmaktadır. Yani 6 sütuna ait 1 saat önceki değerden 24 saat önceki değere kadar 6*24=144 değer giriş değeri olarak alınmaktadır. Çizelge 5.2'deki mevcut son sütun yani t zamanındaki tüketim değeri ise çıkış değeri olarak kullanılmaktadır. Programsal olarak aşağıdaki komutlar kullanılmaktadır. Burada x giriş değerlerini, y ise çıkış değerini ifade etmektedir.

X = reframed_df.iloc[:, :-1] **Girdi değişkeni, sadece son sütunu almayacak.*

y = reframed_df.iloc[:, -1] **Çıktı değişkeni, sadece son sütunu alacak.*

print(X.shape, "x", y.shape, "y") **x ve y şeklinde yazdırıldı. Sonuçta x değişkeni 120 sütun, 145368 satırdan oluşmakta. y de tek boyutlu bir değer vermekte.*

Program bünyesinde yukarıdaki komutlar sonrasında ekrana "(145368, 144) x (145368,) y" ibaresi yazıldığı görülmektedir. Bu ifadeden girişin 145368 satır, 144 katmandan oluştuğu; çıkışın ise 145368 tane y değerinden (tahmin) oluştuğu görülmektedir.

Derin öğrenme için yukarıdaki kısımda kullanılan kütüphanelere ek olarak sklearn, Tensorflow ve keras gibi bazı yaygın python kütüphanelerine de ihtiyaç duyulmaktadır.

Model olarak aşağıdaki model kullanılmaktadır.

```
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
```

Sonraki kısma program üzerinden devam edilmekte ve bazı parametrelerin değiştirilmesi sonucunda modellerin performansı mae, rmse ve r^2_score indeksleri ile ölçülmektedir.

Program ile ilgili kullanılan komutlar ve gerekli açıklamalar github sitesine yüklenmiştir [156].

5.3 Program Arayüzünün Oluşturulması

Öncelikle tüm çalışmalarda Anaconda bünyesindeki jupyter idesi üzerinden python programı kullanılmıştır. Çalışma kısmındaki grafikler Python programı ile çalışmanın sonucu kısmındaki grafikler ise Microsoft Excel programı ile çizdirilmiştir. Ayrıca bazı çalışmalar ve çizelgelerin oluşturulmasında Microsoft Excel programından faydalanılmıştır. RMSE MAE ve R^2 hesaplamaları Python aracılığıyla elde edilmiştir.

Bilgisayar olarak İ3 1. nesil bir işlemciye sahip, 6GB ram ve 1GB harici ekran kartına sahip bir masaüstü bilgisayar kullanılmıştır. Çalışmaları hızlandırmak adına ikinci bir bilgisayar olarak da İ5 1. nesil bir işlemciye sahip, 4GB ram ve onboard ekran kartına sahip bir dizüstü bilgisayar kullanılmıştır.

6. ÇALIŞMA

Bu çalışma kapsamında 44 defa-10 epok, 11 defa-70 epok, 33 defa-100 epok ve 264 defa-50 epok olmak üzere 352 denemeden oluşan 32 farklı grup çalışması yapılmıştır. Ayrıca önceki çalışmadan elde edilen öğrenmelerin sonraki çalışmayı etkilememesi ve ram'lerin enerjisinin tamamen boşalması için her çalışmada bilgisayar kapatılıp yeniden açılmıştır.

Tüm çalışmalarda sırasıyla LSTM, GRU, SimpleRNN, BiLSTM modelleri kullanılmaktadır. Yapılan çalışmalarda bu 4 farklı derin öğrenme modeli ile birlikte optimizasyon için keras kütüphanesine ait 11 farklı optimizasyon yöntemi denenmiştir. Bunlar adam, nadam, adamax, rmsprop, rmsprop*, sgd, sgdtrue, sgdfalse adagrad, ftrl ve adadelta yöntemleridir. Birkaç optimizasyon modeli daha denenmiş fakat sonuç elde edilemediği için listelere eklenmemiştir. En başarılı görülen 7 model rahat karşılaştırma yapılabilmesi amacıyla sonraki kısımlarda görülen çizelgelerde aynı renklere boyanmıştır. Çalışmada standart yöntemler için varsayılan değerler kullanılmıştır. Fakat rmsprop*, sgdtrue ve sgdfalse yöntemleri için varsayılan bazı değerlerde değişime gidilmiştir. Bu üç yöntemin detayları Çizelge 6.1'de listelenmiştir.

Çizelge 6.1 Kullanılan yöntemlerin detayları

Yöntem	Programda Kullanılış Şekli
RMSprop*	optimizer=tf.keras.optimizers.RMSprop(learning_rate=0.01, rho=0.9)
SGDTrue	optimizer=tf.keras.optimizers.SGD(lr=0.01, decay=1e-5, momentum=0.9, nesterov=True)
SGDFalse	optimizer=tf.keras.optimizers.SGD(lr=0.001, decay=1e-5, momentum=1.0, nesterov=False)

6.1 4 Farklı Model Üzerinden 11 Farklı Optimize Edici Yöntem Kullanılarak Tüketim Tahmininin Analiz Edilmesi

4 farklı derin öğrenme modeli LSTM, GRU, SimpleRNN ve BiLSTM kullanılarak her bir model için 11 farklı optimizasyon yöntemi denenmiştir. Yapılan çalışmanın sonuçları RMSE, MAE ve R^2 metrikleriyle ölçülmüş ve sonuçlar Çizelge 6.2.'ye aktarılmıştır. Sonraki kısımlarda gelecek tüm çizelgelerde modeller yukarıdan aşağıya doğru başarı durumlarına göre sıralanmıştır. Bu denemelerde epok değeri 50 olarak alınmıştır.

Çizelge 6.2 4 model -11 optimizer ile çalışma sonuçları

MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2
model = Sequential() model.add(LSTM(50, activation='relu', input_shape = (X_train.shape[1], X_train.shape[2]))) model.add(Dense(1))	Adam	339,33	242,54	0,9972	model = Sequential() model.add(SimpleRNN(50, activation='relu', input_shape = (X_train.shape[1], X_train.shape[2]))) model.add(Dense(1))	Nadam	368,15	265,96	0,9968
	Nadam	428,91	321,56	0,9954		Adam	425,05	307,47	0,9956
	RMSprop	420,57	288,89	0,9957		RMSprop	486,2	357,39	0,9942
	RMSprop*	500,28	376,88	0,9938		adamax	567,34	440,71	0,9921
	adamax	517,6	406,57	0,9935		SGDTrue	667,51	503,7	0,9889
	SGDTrue	768,55	586,43	0,9853		RMSprop*	812,6	664,71	0,9832
	SGD	1602,39	1284,1	0,9294		SGD	904,34	684,67	0,9792
	adagrad	2278,76	1838,13	0,8518		adagrad	1857,1	1488,04	0,9031
	Ftrl	5526,37	4603,25	-5,2028		Ftrl	3173,83	2492,14	0,4742
	adadelta	6749,2	5649,35	-5,44		adadelta	5071,94	4243,3	-0,5113
	SGDFalse	13125,06	11938,01	-1,1E+13		SGDFalse	17577,15	16466,97	-2E+12
model = Sequential() model.add(GRU(50, activation='relu', input_shape = (X_train.shape[1], X_train.shape[2]))) model.add(Dense(1))	Adam	412,18	296,07	0,996	model = Sequential() model.add(Bidirectional(LSTM(50 , activation='relu', input_shape = (X_train.shape[1], X_train.shape[2]))) model.add(Dense(1)))	Adam	313,04	223,66	0,9976
	Nadam	404,5	296,4	0,996		Nadam	345,03	255,65	0,9971
	adamax	481,16	366,41	0,9945		adamax	458,3	342,28	0,9948
	RMSprop	474,57	347,39	0,9944		RMSprop	564,17	458,31	0,9923
	RMSprop*	659,23	537,33	0,9887		RMSprop*	655,86	518,52	0,989
	SGDTrue	961,56	721,47	0,9754		SGDTrue	831,81	635,59	0,9829
	SGD	1336,02	1082,17	0,9546		SGD	1480,73	1165,06	0,9419
	adagrad	1935,03	1535,2	0,8824		adagrad	2212,38	1727,57	0,865
	Ftrl	4800,64	3921,81	-1,7584		Ftrl	4940,55	4151,66	-2,6601
	adadelta	6497,18	5390,8	-5,6841		adadelta	6296,76	5378,12	-8,6355
	SGDFalse	12246,21	10396,46	-3,9E+13		SGDFalse	11978,96	10797,69	0

6.2 4 Farklı Model Üzerinden 11 Farklı Optimize Edici Yöntem Kullanılarak 10 epok - 50 epok ve 100 epok Kullanımıyla Tüketim Tahmininin Analiz Edilmesi

4 farklı derin öğrenme modeli LSTM, GRU, SimpleRNN, BiLSTM kullanılarak 10 epok, 50 epok ve 100 epok ile 11 farklı optimizasyon yöntemi denenmiştir. Fakat BiLSTM modeliyle yapılan çalışmada 100 epok değerine ulaşılmadan bilgisayar hata vermekte ve çalışma tamamlanmamaktadır. Bu yüzden BiLSTM modeli için 100 epok yerine 70 epok ile deneme yapılmıştır. Yapılan çalışmanın sonuçları RMSE MAE ve R^2 metrikleriyle ölçülmüş ve sonuçlar Çizelge 6.3'e aktarılmıştır. Çizelgedeki değerler her model için yukarıdan aşağıya doğru başarı durumlarına göre sıralanmıştır. Bu kısımdaki tüm denemelerde aktivasyon fonksiyonu olarak relu kullanılmıştır.

Çizelge 6.3 4 model-11 optimizer ile 10-50-100 epok deneme sonuçları

MODEL	Optimizer	Epoch	test_rmse	test-mae	test_R^2	Epoch	test_rmse	test-mae	test_R^2	Epoch	test_rmse	test-mae	test_R^2
LSTM(50, activation='relu'	Adam	10 epoch	570.94	448.44	0.9918	50 epoch	339.33	242.54	0.9972	100 epoch	288.81	208.39	0.9979
	Nadam		660.19	514.92	0.9885		428.91	321.56	0.9954		311.34	229.17	0.9976
	RMSprop		810.04	652.61	0.9838		420.57	288.89	0.9957		536.31	429.66	0.9931
	RMSprop*		827.86	691.43	0.9819		500.28	376.88	0.9938		2601.76	2198.17	0.7719
	adamax		1281.91	985	0.9542		517.6	406.57	0.9935		344.83	256.33	0.9971
	SGDTrue		1726.52	1347.86	0.9095		768.55	586.43	0.9853		606.97	448.22	0.9909
	SGD		2219.27	1791.71	0.8494		1602.39	1284.1	0.9294		965.07	752.74	0.9761
	adagrad		4293.3	3464.12	-0.6808		2278.76	1838.13	0.8518		2075.68	1678.33	0.8796
	Ftrl		6400.95	5272.15	-8.7777		5526.37	4603.25	-5.2028		4754.48	3896.32	-1.4893
	adadelta		9461.96	7775.31	-2.6307		6749.2	5649.35	-5.44		4213.74	3414.96	-0.5044
	SGDFalse		6851.15	6364.68	-0.2154		13125.06	11938	-1E+13		33342.09	32704.7	-7.4605
GRU(50, activation='relu'	Adam	10 epoch	600.85	471.83	0.991	50 epoch	412.18	296.07	0.996	100 epoch	339.33	236.88	0.9972
	Nadam		642.95	488.15	0.9896		404.5	296.4	0.996		358.36	266.16	0.9969
	adamax		907.96	706.62	0.9787		481.16	366.41	0.9945		418.22	308.26	0.9959
	RMSprop		729.12	586.74	0.9868		474.57	347.39	0.9944		538.22	430.82	0.993
	RMSprop*		825.57	624.58	0.9809		659.23	537.33	0.9887		690.01	563.86	0.9873
	SGDTrue		1328.25	1032.72	0.9534		961.56	721.47	0.9754		732.58	545.86	0.9862
	SGD		1725.28	1377.51	0.9207		1336.02	1082.17	0.9546		1172.66	941.76	0.965
	adagrad		3867.61	3062.75	-0.3151		1935.03	1535.2	0.8824		1713.39	1385.5	0.9248
	Ftrl		6173.72	5109.03	-9.9445		4800.64	3921.81	-1.7584		3566.78	2837	0.1859
	adadelta		7175.8	5724.15	-4.2308		6497.18	5390.8	-5.6841		4604.73	3768.18	-1.6678
	SGDFalse		12883.23	11131.3	-4E+13		12246.21	10396.5	-4E+13		47909.94	47468.6	-3E+14
SimpleRNN(50, activation='relu'	Nadam	10 epoch	583.1	443.86	0.9912	50 epoch	368.15	265.96	0.9968	100 epoch	397.43	299.79	0.9962
	Adam		764.85	591.93	0.985		425.05	307.47	0.9956		395.97	274.83	0.9962
	RMSprop		925.99	786.24	0.9783		486.2	357.39	0.9942		440.47	305.99	0.9953
	adamax		788.64	602.13	0.9844		567.34	440.71	0.9921		416.41	310.32	0.9958
	SGDTrue		929.72	715.81	0.9779		667.51	503.7	0.9889		614.56	457.05	0.9906
	RMSprop*		831.6	615.19	0.9814		812.6	664.71	0.9832		1150.18	941.03	0.963
	SGD		1640.43	1300.16	0.9215		904.34	684.67	0.9792		744.27	554.69	0.9861
	adagrad		2566.29	2034.92	0.7921		1857.1	1488.04	0.9031		1550.06	1242.25	0.9355
	Ftrl		5402.83	4469.17	-3.4172		3173.83	2492.14	0.4742		1673.12	1350.15	0.9184
	adadelta		9940.89	8008.79	-0.7133		5071.94	4243.3	-0.5113		2685.92	2132.31	0.7625
	SGDFalse		14965.22	13796.1	-4E+15		17577.15	16467	-2E+12		19404.24	18287.4	0
Bidirectional (LSTM(50, activation='relu'	Adam	10 epoch	532.32	404.01	0.9929	50 epoch	313.04	223.66	0.9976	70 epoch	320.99	228.34	0.9975
	Nadam		563.31	437.94	0.992		345.03	255.65	0.9971		336.53	252.75	0.9972
	adamax		749.38	613.66	0.9855		458.3	342.28	0.9948		431.9	319.89	0.9954
	RMSprop		747.82	596.63	0.9857		564.17	458.31	0.9923		486.09	344.27	0.9943
	RMSprop*		893.25	754.15	0.9796		655.86	518.52	0.989		error	error	error
	SGDTrue		1380.79	1070.89	0.9479		831.81	635.59	0.9829		718.31	543.88	0.9873
	SGD		2035.9	1567.92	0.8873		1480.73	1165.06	0.9419		1356.59	1071.95	0.9515
	adagrad		3598.94	2860.68	0.0957		2212.38	1727.57	0.865		2117.37	1648.89	0.8806
	Ftrl		6267.43	5193.4	-9.7918		4940.55	4151.66	-2.6601		4371.47	3638.93	-0.8451
	adadelta		10693.06	8917.82	-3.7993		6296.76	5378.12	-8.6355		4992.94	4202.21	-2.7762
	SGDFalse		10525.63	9369.66	-7E+15		11978.96	10797.7	0		18980.5	17837.2	-4E+14

6.3 4 Farklı Model Üzerinden 6 Farklı Aktivasyon Fonksiyonu Kullanılarak 11 Farklı Optimize Edici Yöntem Kullanımıyla Tüketim Tahmininin Analiz Edilmesi

Bu çalışmada 4 farklı derin öğrenme modeli LSTM, GRU, SimpleRNN, BiLSTM kullanılarak 6 farklı aktivasyon fonksiyonu relu, tanh, elu, LeakyReLU, sigmoid, softmax ile 11 farklı optimizasyon yöntemi denenmiştir. Bu denemelerde epok değeri 50 olarak alınmıştır. Bu yöntemle toplamda 264 deneme yapılmıştır. Yapılan çalışmanın sonuçları RMSE MAE ve R^2 metrikleriyle ölçülmüş ve sonuçlar Çizelge 6.4'e aktarılmıştır. Çizelgedeki değerler her model için yukarıdan aşağıya doğru başarı durumlarına göre sıralanmıştır.

Çizelge 6.4 4 model-6 aktivasyon fonksiyonu-11 optimizer ile çalışma sonuçları

MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2
LSTM(50, activation='relu'	Adam	339,33	242,54	0,9972	LSTM(50, activation='tanh'	Adam	348,22	248,38	0,997	LSTM(50, activation='elu'	Adam	345,19	249	0,997
	Nadam	428,91	321,56	0,9954		Nadam	438,27	327,41	0,9953		adamax	444,34	341,36	0,9951
	RMSprop	420,57	288,89	0,9957		adamax	487	377,74	0,9942		Nadam	503,05	394,25	0,9937
	RMSprop*	500,28	376,88	0,9938		RMSprop	512,41	380,82	0,9937		RMSprop	515,18	387,34	0,9935
	adamax	517,6	406,57	0,9935		RMSprop*	854,81	723,46	0,9814		SGDTrue	640,39	472,64	0,9898
	SGDTrue	768,55	586,43	0,9853		SGDTrue	1039,09	800,82	0,9714		SGD	1721,9	1386,44	0,9179
	SGD	1602,39	1284,1	0,9294		SGD	1781,02	1437,1	0,9119		adagrad	2337,08	1879,03	0,8366
	adagrad	2278,76	1838,13	0,8518		adagrad	2341,44	1879,26	0,8342		RMSprop*	1963629	166538,9	-0,0027
	Ftrl	5526,37	4603,25	-5,2028		adadelta	4415,26	3534,73	-1,10264		adadelta	4539,57	3641,3	-1,4515
	adadelta	6749,2	5649,35	-5,44		Ftrl	5195,91	4282,71	-3,47045		Ftrl	5233,05	4318,77	-3,6805
	SGDFalse	13125,06	11938,01	-1,1E+13		SGDFalse	9368,4	8245,17	-5,8E+12		SGDFalse	27722,16	26952,24	-5E+13
LSTM(50, model.add(LeakyReLU (alpha=0.05)) model.add(Dense(1)) model.add(LeakyReLU (alpha=0.05))	Adam	356,31	257,81	0,9969	LSTM(50, activation='sigmoid'	Adam	571,61	424,7	0,9919	LSTM(50, activation='softmax'	RMSprop	674,48	513,68	0,9883
	RMSprop	476,36	360,91	0,9943		adamax	603,72	478,44	0,9909		RMSprop*	707,29	582,36	0,9869
	Nadam	480,3	373,44	0,9943		Nadam	651,32	527,85	0,9893		Nadam	738,32	550,96	0,9861
	adamax	527,03	414,8	0,9932		RMSprop	676,89	554,74	0,9883		Adam	749,56	576,29	0,9856
	RMSprop*	735,21	611,63	0,9871		RMSprop*	734,95	602,66	0,9857		adamax	1009,98	803,72	0,9736
	SGDTrue	931,44	706,21	0,9777		SGDTrue	953,98	742,45	0,9772		SGDFalse	7976,18	7768,94	-1,01628
	SGD	1853,96	1502,1	0,8977		SGD	1310,82	1039,92	0,9514		SGDTrue	7284,76	6206,36	-1506,18
	adagrad	2496,75	1981,63	0,7888		adagrad	4911,14	3931,39	-4,5018		adagrad	6477,86	5000	-61388,4
	adadelta	5582,12	4529,61	-10,04		adadelta	6291,91	5116,46	-87,2366		SGD	6546,61	5236,91	-160784
	Ftrl	5595,3	4649,56	-6,63		Ftrl	6314,42	5103,01	-219,581		adadelta	11096,23	9103,18	-239028
	SGDFalse	19643,64	18541,27	-2E+09		SGDFalse	7131,07	5998,08	-2,7E+08		Ftrl	6483,96	5016,66	-239995
MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2
GRU(50, activation='relu'	Adam	412,18	296,07	0,996	GRU(50, activation='tanh'	Adam	385,32	290,82	0,9964	GRU(50, activation='elu'	Adam	373,14	275,81	0,9965
	Nadam	404,5	296,4	0,996		RMSprop	457,3	339,17	0,995		Nadam	491,97	378,52	0,9938
	adamax	481,16	366,41	0,9945		Nadam	453,69	355	0,9948		adamax	535,19	416,84	0,993
	RMSprop	474,57	347,39	0,9944		adamax	504,27	392,93	0,9939		RMSprop	593,09	483,17	0,9914
	RMSprop*	659,23	537,33	0,9887		SGDTrue	988,08	746,91	0,9743		SGDTrue	1063,88	815,21	0,9701
	SGDTrue	961,56	721,47	0,9754		SGD	1304,12	1059,14	0,9563		SGD	1316,11	1070,04	0,9559
	SGD	1336,02	1082,17	0,9546		SGDFalse	1354,18	1105,34	0,953		adagrad	1808,68	1471,06	0,9117
	adagrad	1935,03	1535,2	0,8824		adagrad	1772,16	1444,78	0,9156		Ftrl	4373,06	3502,84	-0,6797
	Ftrl	4800,64	3921,81	-1,7584		adadelta	4079,86	3355,02	-0,40214		adadelta	4295,75	3512,75	-0,8821
	adadelta	6497,18	5390,8	-5,6841		RMSprop*	5443,75	4511,16	-0,51661		RMSprop*	38568,77	38092,7	-19,4889
	SGDFalse	12246,21	10396,46	-3,9E+13		Ftrl	4341,58	3476,94	-0,63074		SGDFalse	gradyanla	5 .epokta	loss nan
GRU(50, model.add(LeakyReLU (alpha=0.05)) model.add(Dense(1)) model.add(LeakyReLU (alpha=0.05))	Adam	407,3	313,25	0,996	GRU(50, activation='sigmoid'	Adam	486,33	365,09	0,9942	GRU(50, activation='softmax'	Nadam	580,26	426,54	0,9914
	adamax	434,32	329,73	0,9954		adamax	556,56	403,58	0,9923		Adam	582,03	437,17	0,9913
	Nadam	434,56	336,82	0,9954		RMSprop*	585,6	473,18	0,9909		RMSprop	605,46	480,02	0,9907
	RMSprop	484,9	364,8	0,9944		Nadam	684,04	562,61	0,9881		adamax	759,5	587,8	0,9851
	SGDTrue	888,5	664,38	0,9794		SGDTrue	1099,16	875,42	0,9702		RMSprop*	771,55	589,59	0,984
	RMSprop*	1181,28	1044,98	0,9601		RMSprop	1566,11	1439,63	0,9373		SGDTrue	1467,28	1163,17	0,9472
	SGD	1349,08	1097,58	0,9531		SGD	1729,4	1373,13	0,9118		adagrad	6507,02	5231,44	-589,92
	adagrad	1995,1	1606,02	0,8833		adagrad	5008,91	3992,44	-4,7411		SGD	6417,57	5136,38	-1050,57
	Ftrl	4923,3	4031,88	-2,1815		adadelta	6400,78	5206,39	-25,4117		adadelta	9055,85	6966,22	-1384,89
	adadelta	5487,66	4555,68	-3,2035		Ftrl	5875,25	4733,97	-28,5895		Ftrl	6550,42	5234,4	-2,9E+12
	SGDFalse	34729,52	34118,11	0		SGDFalse	17975,84	16764,13	-3,4E+14		SGDFalse	16741,91	15609,55	-1,8E+13

Çizelge 6.4 (devam) 4 model-6 aktivasyon fonksiyonu-11 optimizer ile çalışma sonuçları

MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2
SimpleRNN(50, activation='relu'	Nadam	368,15	265,96	0,9968	SimpleRNN(50, activation='tanh'	Nadam	444,34	331,85	0,9952	SimpleRNN(50, activation='elu'	adamax	430,17	328,57	0,9956
	Adam	425,05	307,47	0,9956		adamax	499,42	391,15	0,9939		Nadam	453,81	348,36	0,9949
	RMSprop	486,2	357,39	0,9942		Adam	503,77	380,75	0,9938		Adam	504,19	391,36	0,9937
	adamax	567,34	440,71	0,9921		SGDTrue	578,51	429,66	0,9917		RMSprop	626,87	487,26	0,9902
	SGDTrue	667,51	503,7	0,9889		RMSprop	637,36	511,22	0,9899		SGDTrue	637,47	473,76	0,9896
	RMSprop*	812,6	664,71	0,9832		SGD	987,79	742,87	0,9748		SGD	902,93	684,2	0,9789
	SGD	904,34	684,67	0,9792		adagrad	1817,51	1385,47	0,911		adagrad	1770,56	1351,29	0,9182
	adagrad	1857,1	1488,04	0,9031		Ftrl	2619,27	2056,14	0,6899		Ftrl	2351,54	1839,02	0,7707
	Ftrl	3173,83	2492,14	0,4742		adadelta	3751,66	2969,22	0,295		adadelta	3982,15	3054,89	0,5751
	adadelta	5071,94	4243,3	-0,5113		RMSprop*	8552,78	7295,31	-9,03022		RMSprop*	gradyanlar	20 .epokta	loss nan
	SGDFalse	17577,15	16466,97	-2,2E+12		SGDFalse	16186,5	15041,31	0		SGDFalse	gradyanlar	1, .epokta	loss nan
SimpleRNN(50, model.add(LeakyReLU (alpha=0.05)) model.add(Dense(1)) model.add(LeakyReLU (alpha=0.05))	adamax	434,25	329,72	0,9955	SimpleRNN(50, activation='sigmoid'	Adam	574,77	429,73	0,9918	SimpleRNN(50, activation='softmax'	Adam	743,72	568,27	0,9859
	Adam	471,85	369,93	0,9946		adamax	587,03	436,27	0,9913		adamax	758,64	594,51	0,9856
	Nadam	481,3	365,27	0,9944		Nadam	603,4	461,66	0,991		RMSprop*	802,43	625,98	0,9832
	RMSprop	496	354,68	0,9941		RMSprop	767,14	621,73	0,9851		Nadam	822,38	636,83	0,9828
	SGDTrue	702,21	549,39	0,988		SGDTrue	907,31	717,74	0,9798		RMSprop	836,27	649,21	0,9827
	SGD	1147,45	880,64	0,966		RMSprop*	981,02	849,32	0,9759		SGDTrue	1469,35	1174,24	0,9484
	adagrad	2053,06	1611,04	0,8802		SGD	1406,61	1123,17	0,9477		SGD	6318,53	5067,66	-280,158
	RMSprop*	3541,3	3188,97	0,4623		adagrad	4847,98	3832,69	-3,4409		adagrad	6491,58	5214,82	-363,936
	adadelta	4410,39	3462,33	0,0894		Ftrl	5448,03	4472,32	-11,3086		adadelta	9206,51	7113,67	-957,729
	Ftrl	4199,4	3377,07	-0,3491		adadelta	6622,92	5302,6	-16,2007		SGDFalse	6535,04	5202,87	-1,2E+12
	SGDFalse	47858,36	47416,54	-2,4E+15		SGDFalse	8772,23	6676,59	-2E+13		Ftrl	6550,41	5234,33	-2,4E+13
MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2	MODEL	OPTİMİZER	test_rmse	test-mae	test_R2
Bidirectional(LSTM(50, activation='relu'	Adam	313,04	223,66	0,9976	Bidirectional(LSTM(50, activation='tanh'	Adam	449,23	326,6	0,9953	Bidirectional(LSTM(50 , activation='elu'	adamax	407,06	309,82	0,996
	Nadam	345,03	255,65	0,9971		adamax	469,17	356,01	0,9947		Adam	409,33	295,1	0,996
	adamax	458,3	342,28	0,9948		Nadam	486,21	385,27	0,9942		Nadam	456,99	360,97	0,9949
	RMSprop	564,17	458,31	0,9923		RMSprop	546,81	420,88	0,9927		RMSprop	489,95	367,67	0,9941
	RMSprop*	655,86	518,52	0,989		RMSprop*	759,2	650,4	0,9854		SGDTrue	787,53	604,32	0,9844
	SGDTrue	831,81	635,59	0,9829		SGDTrue	1113,75	847,13	0,9678		adagrad	2146,96	1656,58	0,8749
	SGD	1480,73	1165,06	0,9419		SGD	1520,68	1181,71	0,9395		SGD	1478,88	1147,76	0,9423
	adagrad	2212,38	1727,57	0,865		adagrad	2134,76	1652,72	0,8748		adadelta	3501,27	2709,49	0,1778
	Ftrl	4940,55	4151,66	-2,66009		adadelta	3372,01	2585,07	0,3352		Ftrl	4474,77	3705,9	-1,0879
	adadelta	6296,76	5378,12	-8,63554		Ftrl	4397,66	3641,77	-0,8918		SGDFalse	25000,67	24144,33	-2,9E+14
	SGDFalse	11978,96	10797,69	0		SGDFalse	26463,48	25655,82	-2,9E+15		RMSprop*	gradyanlar	14 .epokta	loss nan
Bidirectional(LSTM(50, model.add(LeakyReLU (alpha=0.05)) model.add(Dense(1)) model.add(LeakyReLU (alpha=0.05))	Adam	345,7	255,37	0,9971	Bidirectional(LSTM(50, activation='sigmoid'	Adam	532,85	385,52	0,9929	Bidirectional(LSTM(50 , activation='softmax'	Nadam	583,81	422,28	0,9914
	Nadam	375,89	275,41	0,9966		adamax	551,37	395,19	0,9924		RMSprop	625,43	495,37	0,9901
	adamax	422,55	319,73	0,9956		Nadam	583,35	474,81	0,9914		Adam	660,37	479,33	0,9889
	RMSprop	569	443,41	0,9923		RMSprop	647,53	483,6	0,9893		RMSprop*	669,16	523,6	0,9882
	SGDTrue	1044,11	781,07	0,9706		RMSprop*	717,72	601,09	0,9868		adamax	1081,53	838,94	0,97
	SGD	1617,35	1271,56	0,9252		SGDTrue	942,03	741,7	0,9783		SGDTrue	7147,07	6100,91	-323,78
	adagrad	2356,08	1826,02	0,8345		SGD	1795,67	1358,93	0,8969		adagrad	6489,96	5150,1	-19493,5
	RMSprop*	4742,55	3692,83	-1,185		adagrad	4410,44	3545,21	-0,9066		SGD	6525,39	5221,89	-20221,4
	adadelta	4761,75	3744,96	-2,6703		adadelta	5906,26	4933,44	-14,4493		Ftrl	6524,51	5188,54	-272439
	Ftrl	5079,6	4236,37	-3,9808		Ftrl	5808,72	4697,45	-25,3095		adadelta	9915,51	7836,71	-77048,6
	SGDFalse	36371,31	35824,81	-14357,3		SGDFalse	32291,68	31633,17	-6,8E+13		SGDFalse	18929,85	18877,85	-7,3997

7. ÇALIŞMANIN SONUÇLARI

Yukarıdaki çizelgelerdeki değerler her model için başarım oranlarına göre yukarıdan aşağıya doğru sıralanmıştır. Başarılı olan modellerin RMSE MAE ve R^2 metriklerinin üçünde de başarılı oldukları görülmekle birlikte, istisna olarak bazen üstteki deneme ile alttaki denemede birinin bir metrikte diğerinin diğer metrikte başarılı olduğu görülmektedir. Bu yüzden kolaylık olması için hem listelemede hem de grafiklerde R^2 metriği ölçüt alınmıştır.

R^2 değeri 0,99 ve üzeri olan değerlerin arka planı yeşile, 0,98-0,99 arası olan değerlerin arka planı pembeye boyanmıştır. R^2 değeri 0 veya negatif olan değerlerin arka planı ise kahverengiye boyanmıştır.

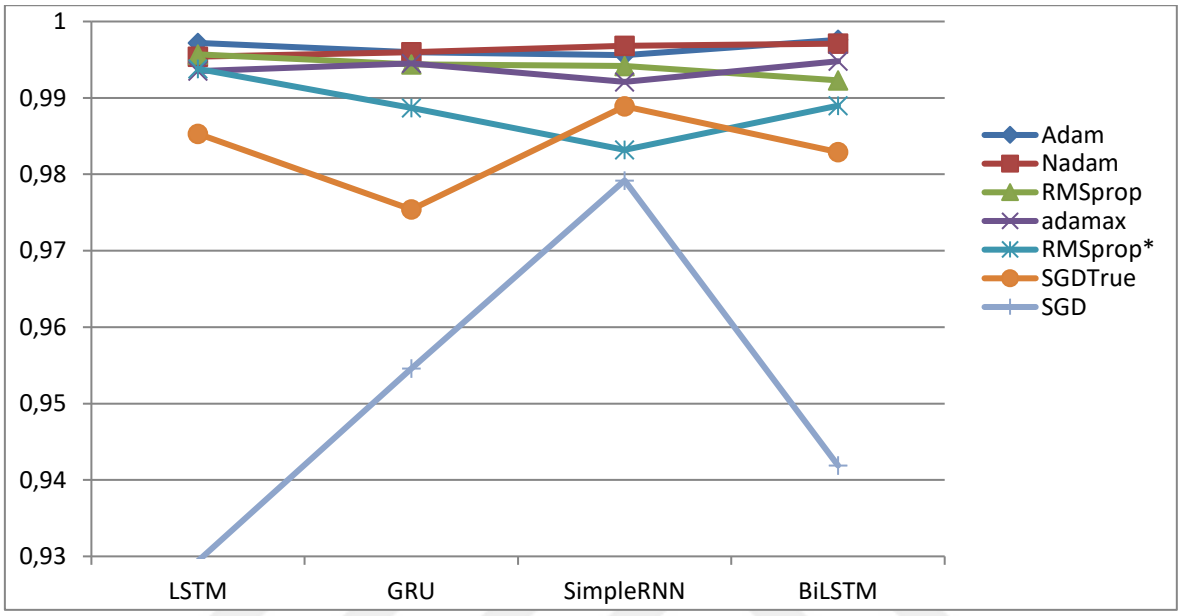
Önceden yapılmış çalışmaların çok büyük bir kısmında optimizasyon fonksiyonu olarak adam, aktivasyon fonksiyonu olarak relu kullanılmaktadır. Bu çalışmada başarım olarak adam ile yarışabilecek ve bazı denemelerde daha yüksek başarımlar elde eden yöntemler görülmüştür. Yapılan çalışma sonuçlarında 6 grup çalışmada nadam, 3 grup çalışmada adamax ve 1 grup çalışmada rmsprop en yüksek değerleri elde etmiş, diğer gruplarda ise hep adam yöntemi en yüksek değerlere ulaşmıştır. Bu yönüyle literatürde pek yer bulmayan nadam ve adamax dikkat çekmektedir.

Ftrl, adadelta ve sgdfalse yöntemleriyle denenen çalışma sonuçlarının çok büyük kısmının kahverengi arka planla boyandığı ve başarısız olduğu gözlemlenmektedir. Ftrl yönteminin sadece 5 denemede pozitif değerler aldığı ve sadece 1 çalışmada 0,91 R^2 değerine ulaştığı görülmektedir. Adadelta yönteminin sadece 6 denemede pozitif değerler aldığı ve ancak 1 çalışmada 0,76 R^2 değerine ulaştığı görülmektedir. Sgdfalse yönteminin sadece 1 denemede pozitif değer aldığı ve 0,95 R^2 değerine ulaştığı görülmektedir. Bu yüzden grafik çizimlerinde bu üç yöntem değerlendirmeye alınmayacaktır. Adagrad yöntemiyle yapılan denemelerin diğer yöntemlere düşük değerler alması yüzünden bu yöntemin sonucu da bazı grafiklerden çıkarılmıştır.

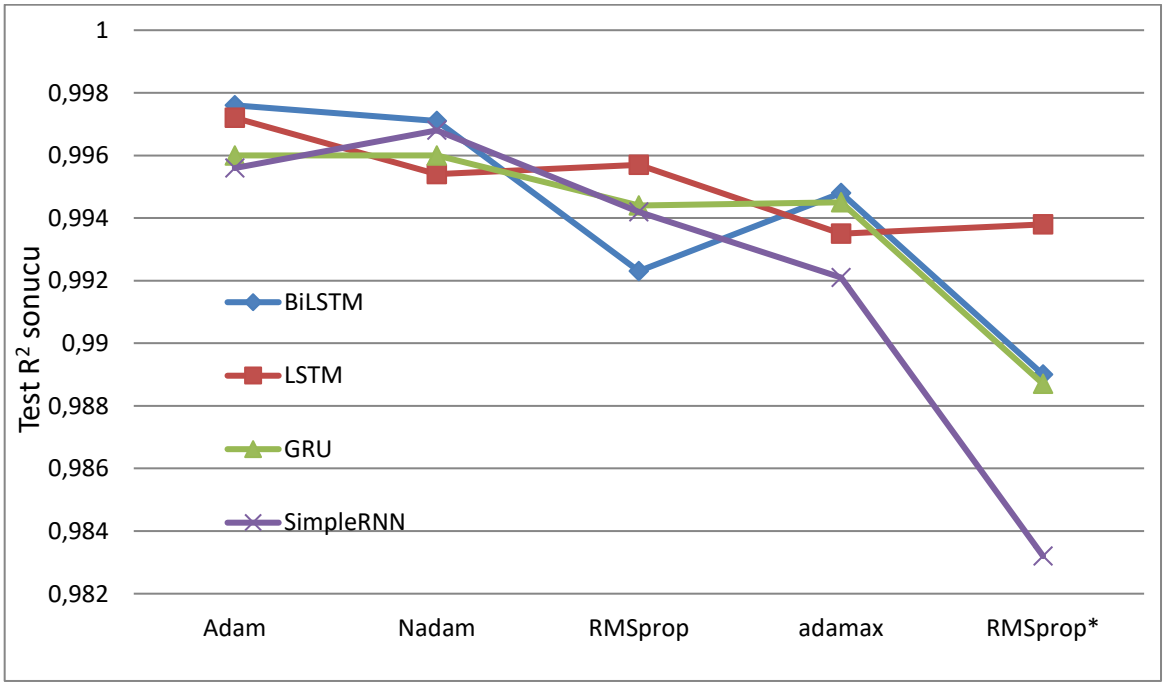
7.1 4 Farkli Model 11 Farklı Optimize Edici Kullanılarak Yapılan Çalışmanın Sonuçları

4 modele göre en başarılı 7 optimizasyon yönteminin sonuçlarına göre Şekil 7.1. ve Şekil 7.2 grafikleri elde edilmektedir. Bu çalışmada aktivasyon fonksiyonu olarak relu

kullanılmakta ve 50 epok ile çalışma yapılmaktadır. Şekil 7.1.'e 4 modele göre de en başarılı sonuçlara bakılınca ilk sırayı adam ve nadam sonrasını ise adamax ve rmsprop almaktadır. Adagrad yöntemi daha düşük sonuçlar aldığı için gösterilmemiştir. Şekil 7.2.'ye göre en başarılı modeller olarak çok net bir ayrım olmamakla beraber BiLSTM ve LSTM dikkat çekmektedir.



Şekil 7.1 : 4 model - 7 optimizer - R2 grafiği

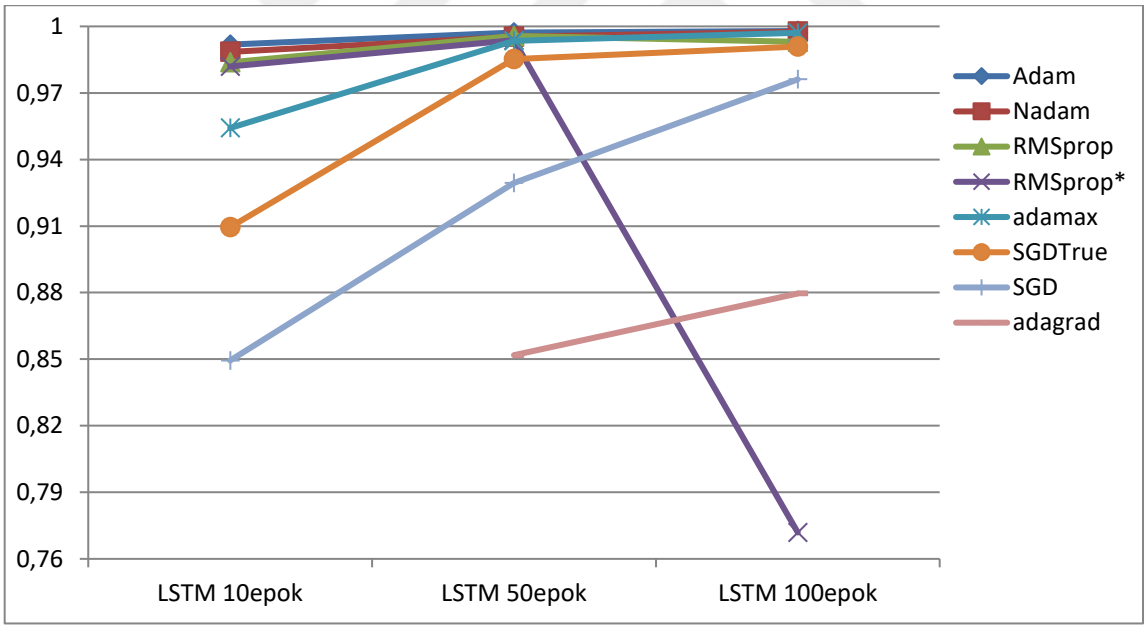


Şekil 7.2 : 4 model - 5 optimizer - R2 grafiği

7.2 4 Farklı Model 11 Farklı Optimize Edici ve 10 epok - 50 epok ve 100 epok Kullanılarak Yapılan Çalışmanın Sonuçları

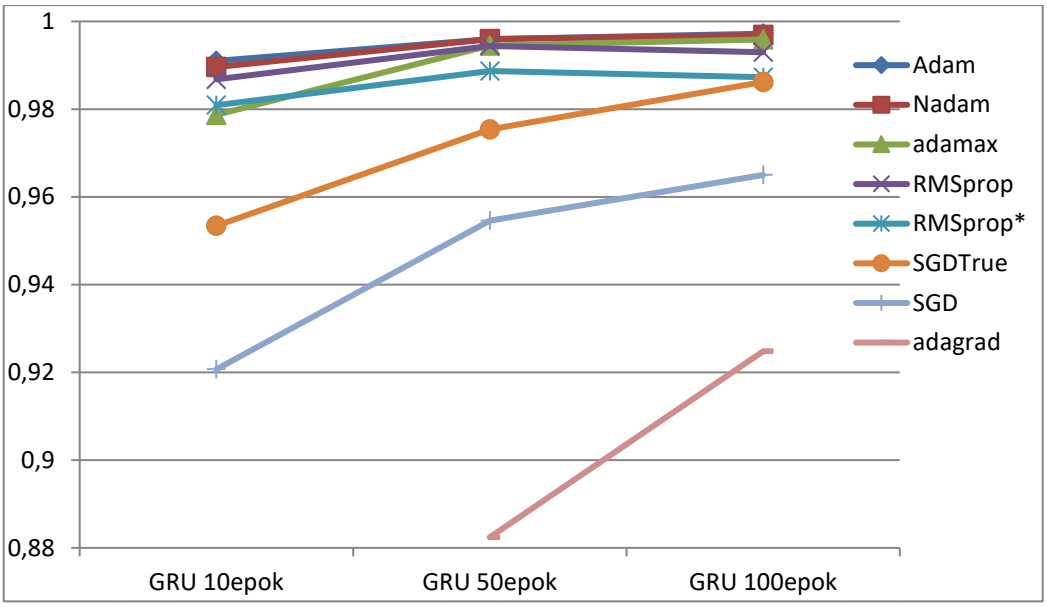
Aşağıdaki Şekil 7.3., 7.4., 7.5. ve 7.6.'da sırasıyla LSTM, GRU, SimpleRNN ve BiLSTM modelleri ve relu aktivasyon fonksiyonu kullanılarak 8 farklı optimizasyon fonksiyonu kullanılarak çalışma değerlendirilmiştir. Epok değeri için 10, 50 ve 100 denenmiştir. Çalışmada çoğu denemede epok sayısının artırılmasının sonuçları olumlu etkilediği görülmektedir. Aslında her model için değerin düşeceği bir sınır epok değeri mevcuttur. Ama 100 epok değeri şimdilik yeterli görüldüğünden epok değerleri daha fazla arttırılmamıştır. Hatta hem zamandan tasarruf etmek hem de kötü sonuçlara denk gelmemek için diğer iki bölümdeki tüm çalışmalarda epok değeri 50 alınmıştır.

Şekil 7.3'te adagrad yöntemi için 10 epok değeri negatif sonuç verdiğinden şekilde gösterilmemiştir. Burada epok sayısı arttıkça R^2 değeri artarken hem rmsprop hem de rmsprop* optimizasyon fonksiyonu için 50 epokta değer artmış fakat 100 epokta azalmıştır.



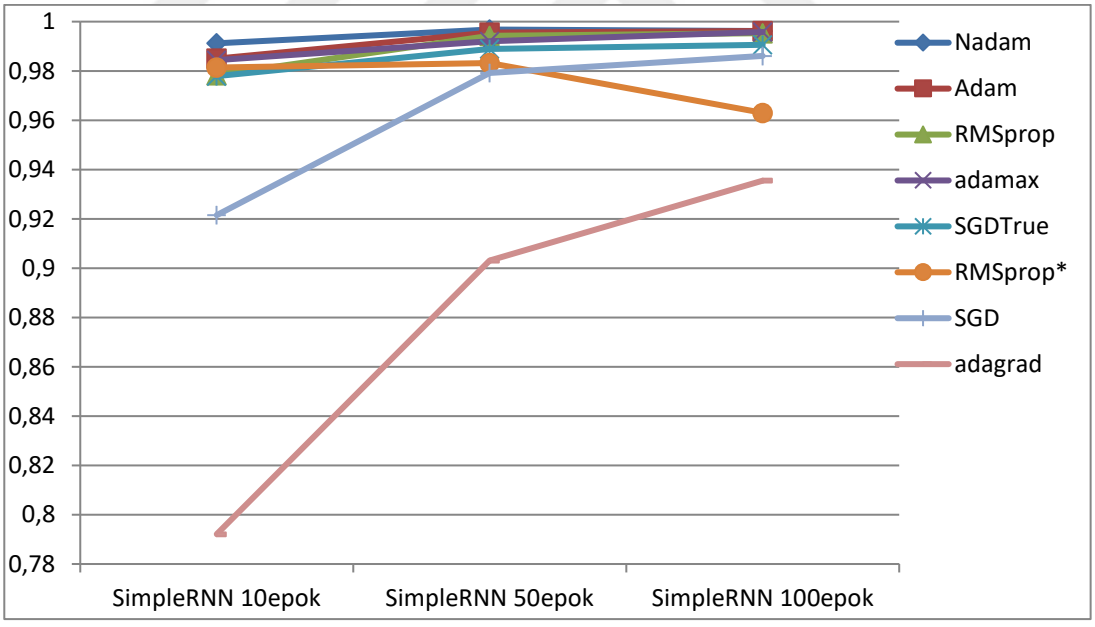
Şekil 7.3 : LSTM-8 optimizer 1-50-100 epok - R2 grafiği

Şekil 7.4'te adagrad yöntemi için 10 epok değeri negatif sonuç verdiğinden şekilde gösterilmemiştir. Burada epok sayısı arttıkça R^2 değeri artarken rmsprop optimizasyon fonksiyonu için 50 epokta değer artmış fakat 100 epokta azalmıştır.



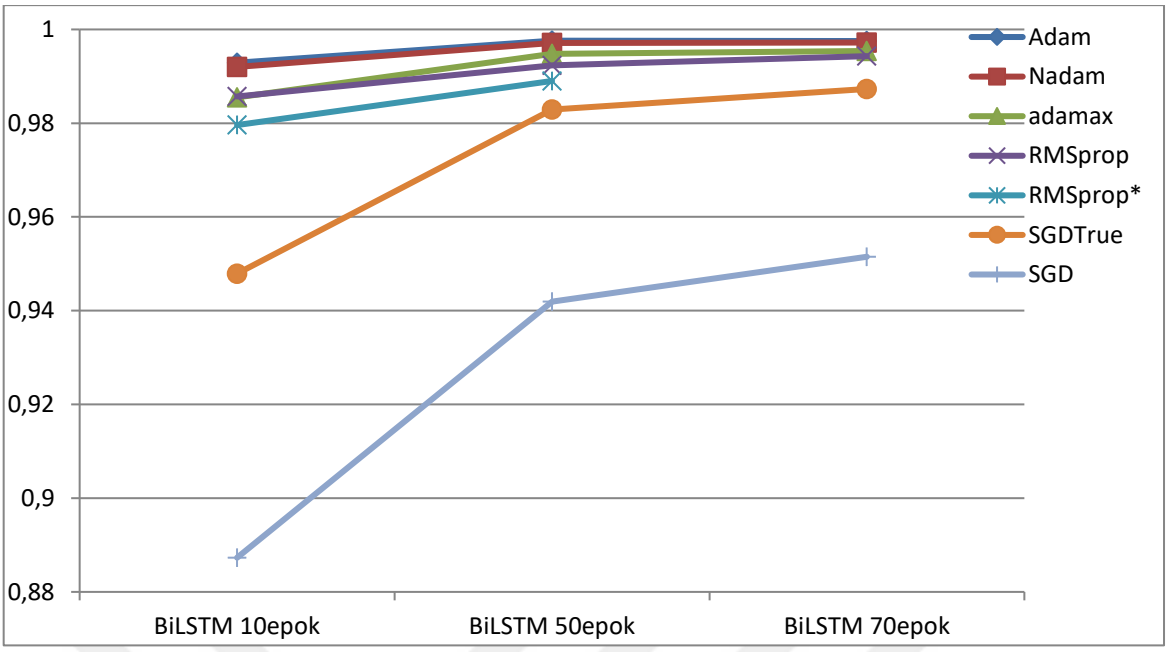
Şekil 7.4 : GRU - 8 optimizer 1-50-100 epok - R2 grafiği

Şekil 7.5'te epok sayısı arttıkça R2 değeri artarken hem rmsprop* hemde nadam optimizasyon fonksiyonu için 50 epokta değer artmış fakat 100 epokta azalmıştır. Buna rağmen nadam yöntemi en başarılı model olarak dikkat çekmektedir.



Şekil 7.5 : SimpleRNN - 8 optimizer 1-50-100 epok - R2 grafiği

Şekil 7.6'da epok sayısı arttıkça R2 değeri artarken adam optimizerı için 50 epokta değer artmış fakat 100 epokta azalmıştır. Ayrıca rmsprop* yönteminde 70 epok değeri gradyanların patlaması sorunu nedeniyle nan hatası verdiğinden ölçülememiştir.



Şekil 7.6 : BiLSTM - 8 optimizer 1-50-100 epok - R2 grafiği

7.3 4 Farklı Model 6 Farklı Aktivasyon Fonksiyonu 11 Farklı Optimize Edici Kullanılarak Yapılan Çalışmanın Sonuçları

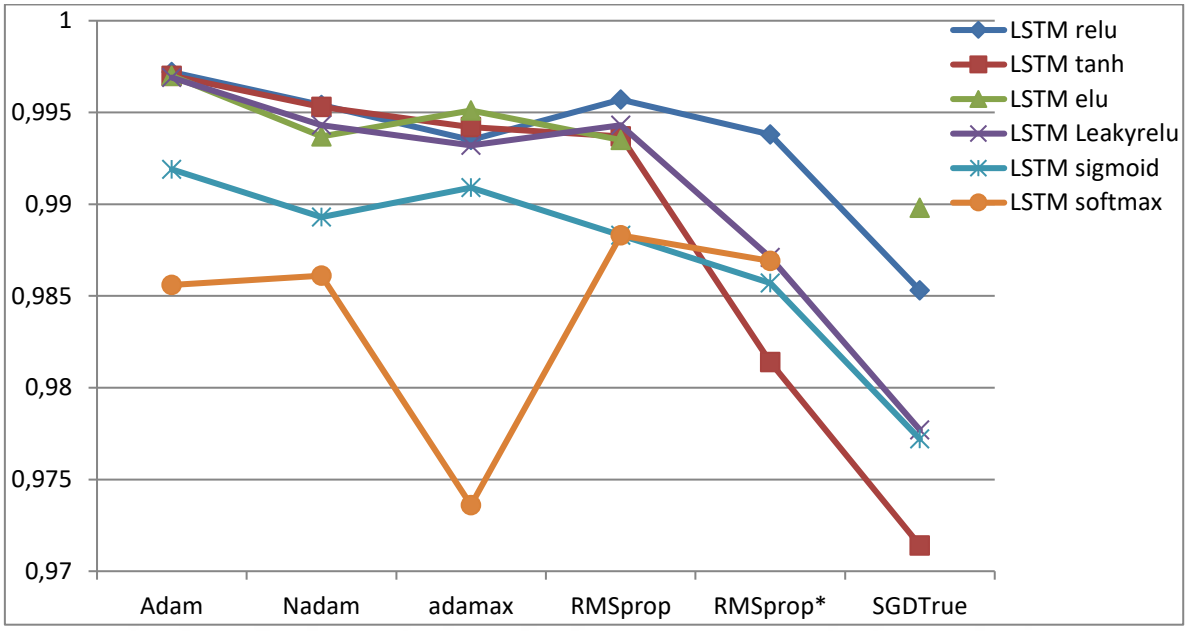
Aşağıdaki Şekil 7.7., Şekil 7.8., Şekil 7.9. ve Şekil 7.10'da sırasıyla LSTM, GRU, SimpleRNN ve BiLSTM modelleri ve 6 farklı aktivasyon fonksiyonu relu, tanh, elu, LeakyReLU, sigmoid, softmax ile 6 farklı optimizasyon fonksiyonu için sonuçlar çizdirilmiştir. Epok değeri 50 alınmıştır. Adagrad yöntemi 24 denemenin 8'inde negatif değerler aldığı için grafikte belirtilmemiştir. Sgd yöntemi ise daha düşük başarım elde ettiği için grafikte belirtilmemiştir. Ayrıca diğer yöntemlerde de negatif olan değerler gösterilmemiştir.

Şekil 7.7 a, Şekil 7.8 a, Şekil 7.9 a ve Şekil 7.10 a'dan görüleceği üzere LSTM, GRU, SimpleRNN ve BiLSTM modelleri için en başarılı aktivasyon fonksiyonları relu, tanh, elu, Leakyrelu olarak görülmektedir. Sigmoid ve softmax ın ise daha başarısız olduğu görülmektedir. Şekil 7.9'da çalışmanın daha net görülebilmesi için SimpleRNN, Leakyrelu ve rmsprop* yöntemlerinin kullanıldığı çalışmanın sonuç değeri olan 0,45 silinmiştir. Ayrıca Şekil 7.9'da SimpleRNN, elu ve rmsprop* yöntemleri ile Şekil 7.10'da BiLSTM, elu ve rmsprop* yöntemlerinin kullanıldığı çalışmalarda gradyanların patlaması nedeniyle nan sorunu yaşanmıştır.

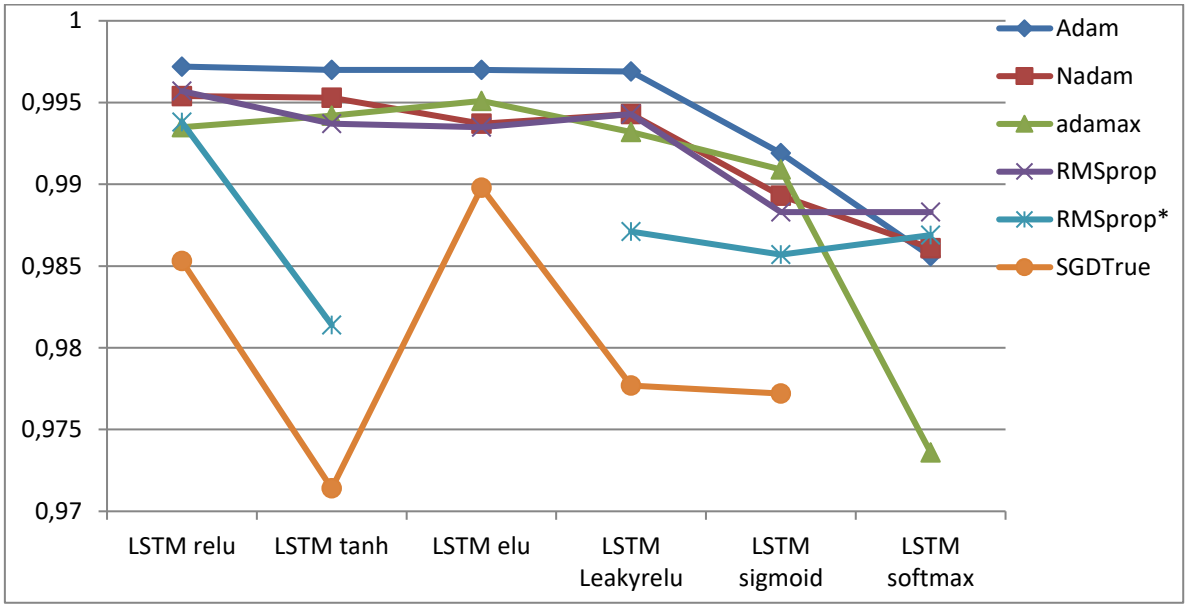
Şekil 7.7 b, Şekil 7.8 b, Şekil 7.9 b ve Şekil 7.10 b'den görüleceği üzere LSTM, GRU, SimpleRNN ve BiLSTM modelleri için en başarılı optimizasyon fonksiyonları

adam, nadam, adamax ve rmsprop olarak görülmektedir. Rmsprop* ve sgdtrue nin ise daha başarısız olduğu görülmektedir.

Çalışmanın tümünde 1 deneme hariç sgdtrue yönteminin sgd yönteminden başarılı olduğu açıkça görülmektedir. Sgdfalse yönteminin ise bu iki yönteme göre çok başarısız olduğu görülmektedir. Ayrıca tüm çalışma boyunca 2 deneme hariç rmsprop yönteminin rmsprop* yönteminden daha başarılı olduğu görülmektedir.

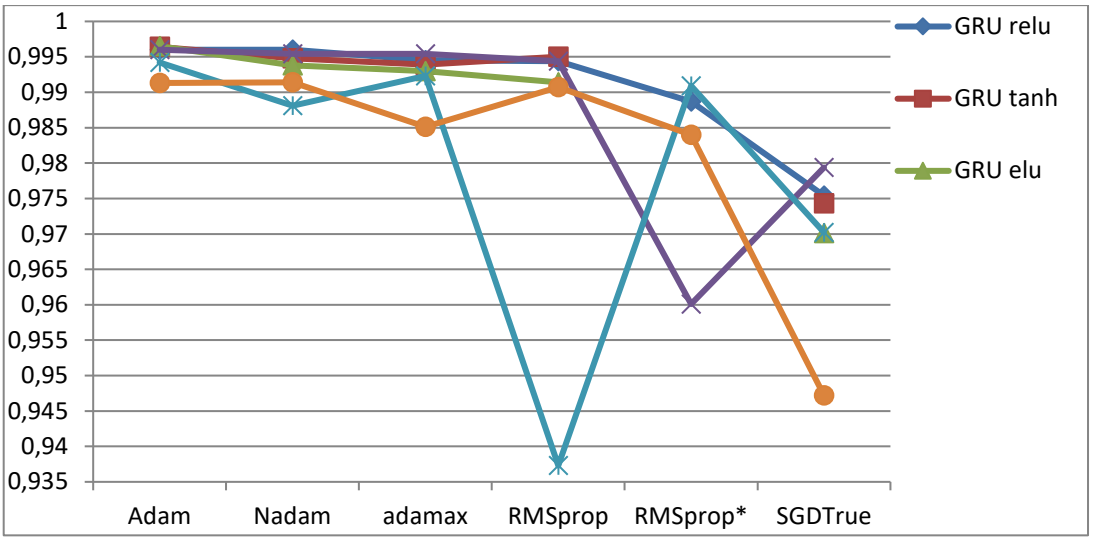


a)

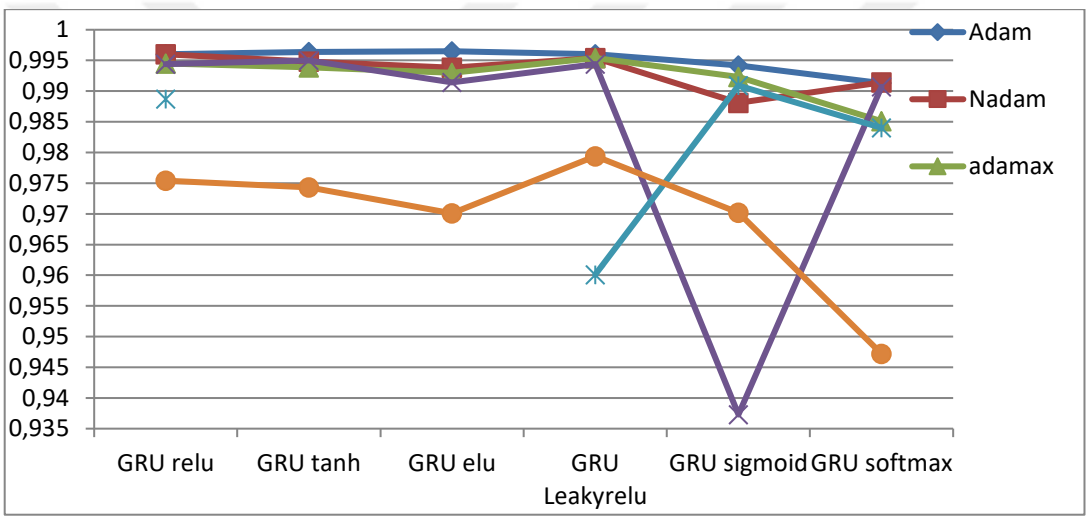


b)

Şekil 7.7 : a) LSTM - 6 aktivasyon fonksiyonu ile 6 optimizer - R2 grafiği. b) LSTM - 6 optimizer ile 6 aktivasyon fonksiyonu - R2 grafiği

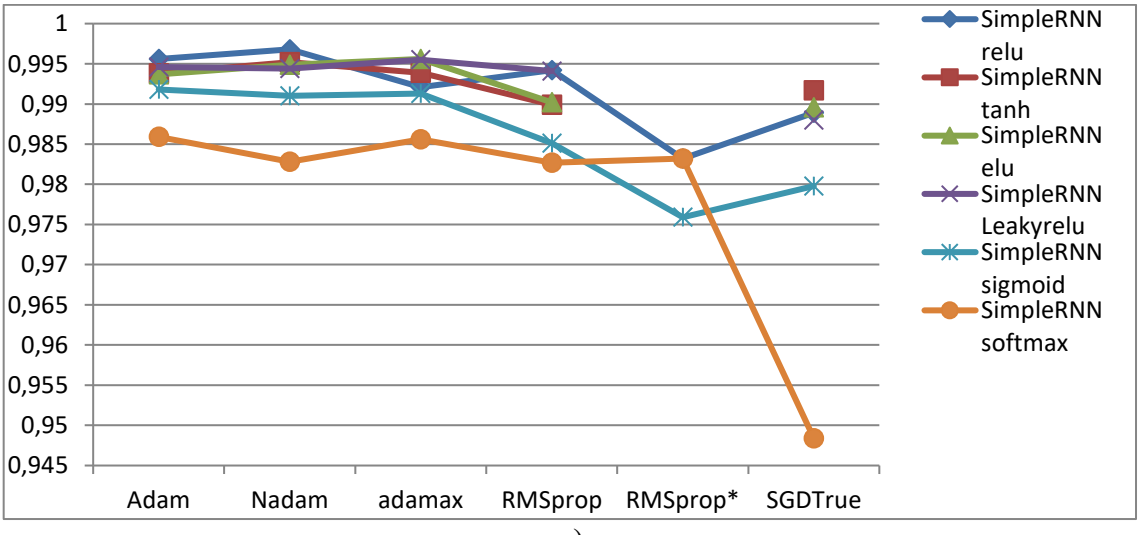


a)

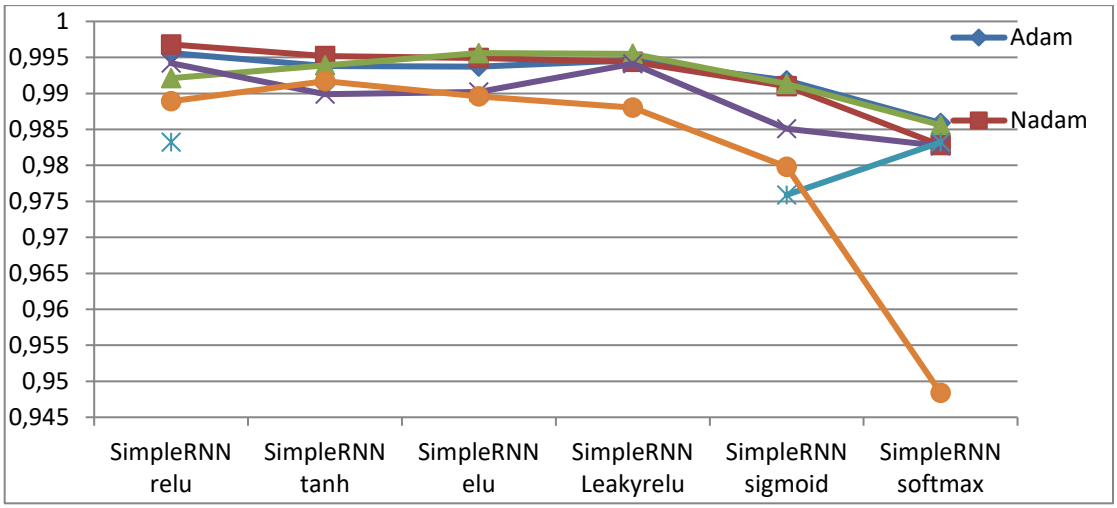


b)

Şekil 7.8 : a) GRU - 6 aktivasyon fonksiyonu ile 6 optimizer - R2 grafiği. b) GRU - 6 optimizer ile 6 aktivasyon fonksiyonu - R2 grafiği

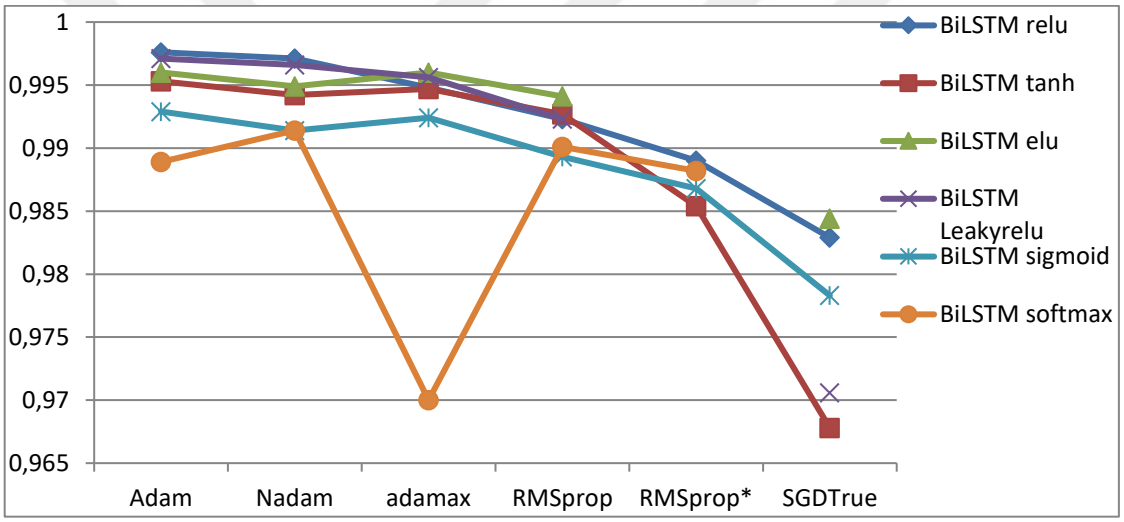


a)

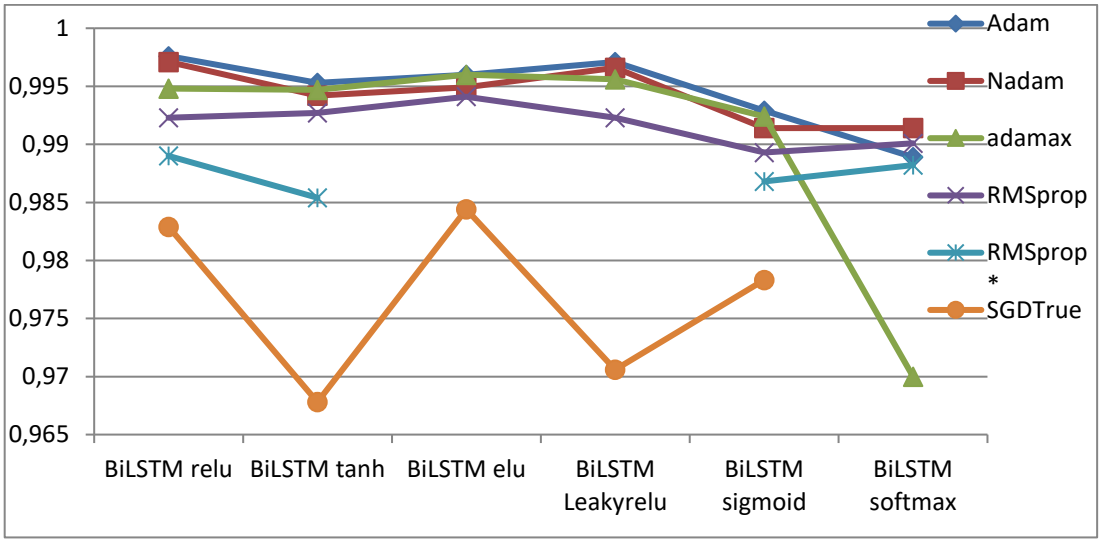


b)

Şekil 7.9 : a) SimpleRNN - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği b) SimpleRNN - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği



a)



Şekil 7.10 : a) BiLSTM - 6 aktivasyon fonksiyonu ile 6 optimizer - R^2 grafiği b) BiLSTM - 6 optimizer ile 6 aktivasyon fonksiyonu - R^2 grafiği

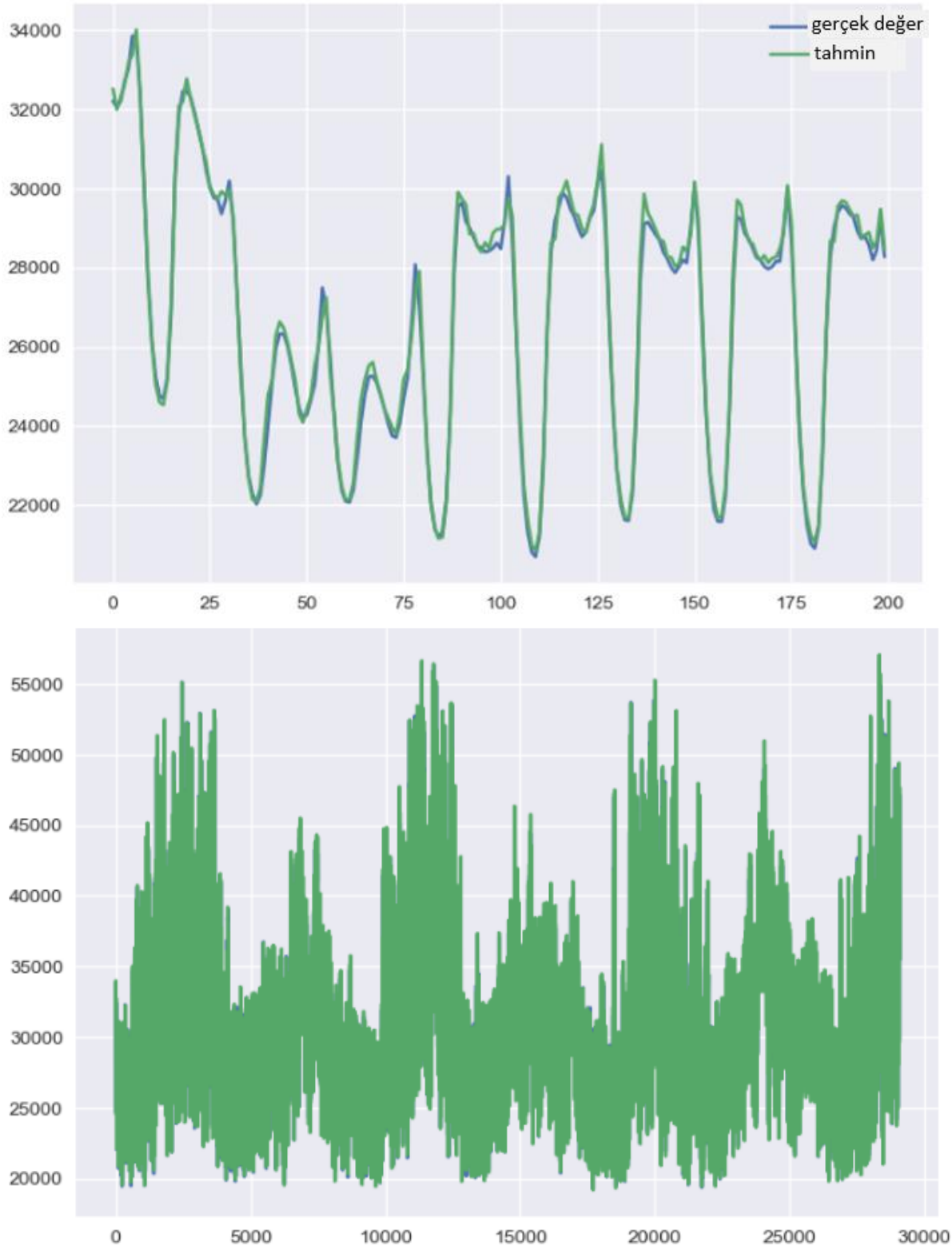
7.4 Elde Edilen En İyi Sonuçlar

Literatür taraması kısmında aktarıldığı üzere yapılan bu çalışma Çizelge 7.1'de görüldüğü gibi aynı veri setini kullanarak yapılan üç farklı çalışma ile kıyaslanmaktadır. Yapılan bu karşılaştırmada meta-analiz yöntemi kullanılmaktadır. Yapılan karşılaştırmalara göre bu çalışmalarda veri setindeki eksik-fazla verilerle ve verilerin sıralanması ile ilgili çalışma yapılıp yapılmadığı ve hangi modellerin kullanıldığı ilgili çalışmalara göre belirtilmiştir. Bu üç çalışmaya göre hem test mse hem de test R^2 metrikleri açısından çalışmamızın başarılı olduğu görülmektedir.

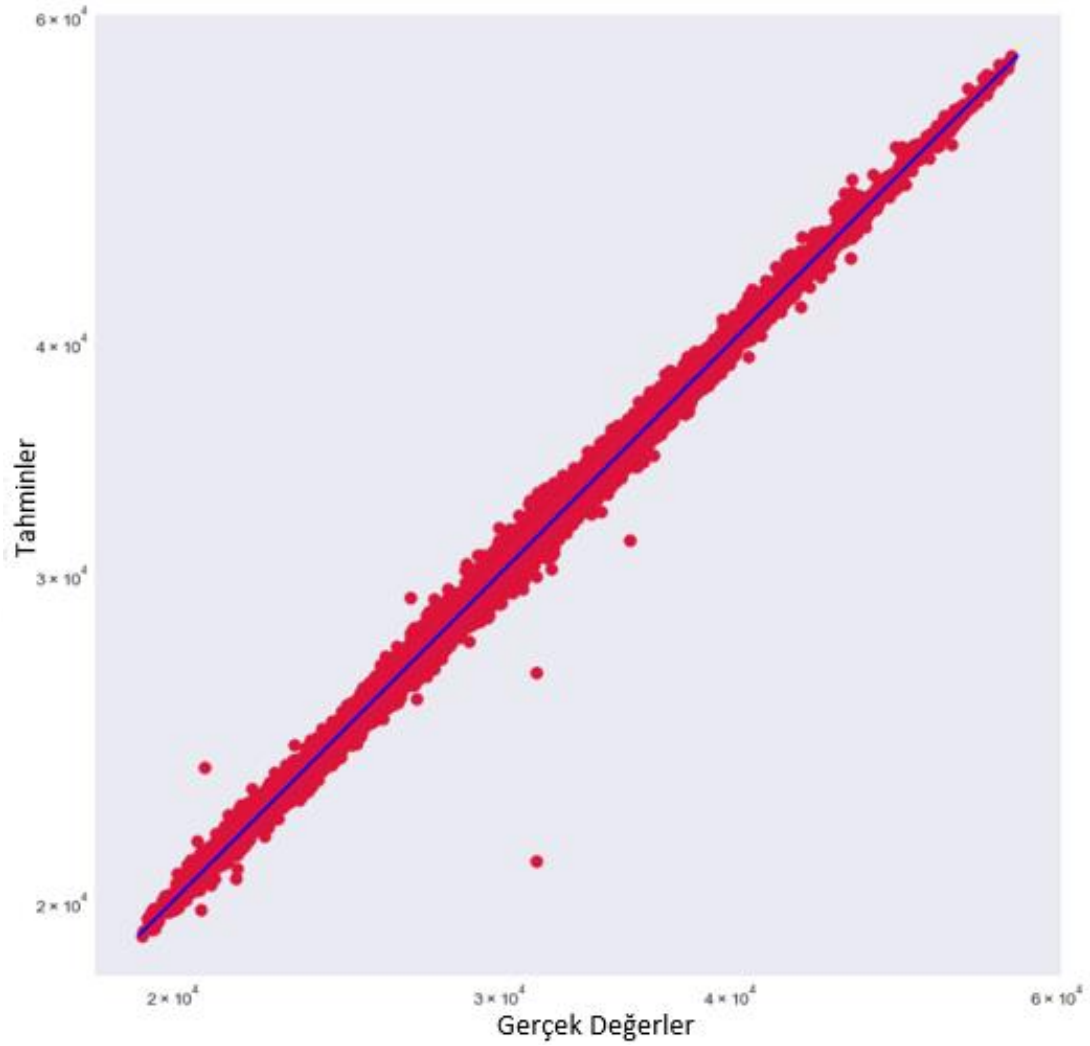
Çizelge 7.1 Yapılan çalışmanın meta-analiz yöntrmi ile aynı veri setini kullanan üç benzer çalışmaya göre kıyaslanması

Çalışma	Veri setindeki eksik ve fazla verilerin analiz edilmesi	Verilerin zamana göre sıralanması	Kullanılan mimari	test MSE	test R^2
[44]	Gerçekleştirilmedi	Gerçekleştirilmedi	Çok katmanlı ve karmaşık bir SimpleRNN ve LSTM mimarisi		0,9814
[45]	Gerçekleştirilmedi		RNN, LSTM ve GRU mimarisi	RNN için 0,4, LSTM için 0,31 GRU için 0,24	
[46]	Gerçekleştirildi			LSTM için 0,01	
Bu çalışma	Gerçekleştirildi	Gerçekleştirildi	SimpleRNN, LSTM, GRU ve BiLSTM mimarisi	Bu metriğin 0,002 değerine düştüğü gözlemlenmiştir	0,9972

Tüm bu çalışmalar sonucunda MAE, RMSE ve R^2_score indeksleri göz önüne alınarak 50 epok ile en başarılı 10 çalışma aşağıda Çizelge 7.1'de listelenmektedir. En yüksek başarı oranı BiLSTM modeli relu aktivasyon fonksiyonu ve adam optimizasyon fonksiyonu ile 0,9976 olarak elde edilmiştir. Bu modele ait ilk 200 değer ve 30000 değer için çizilen gerçek ve tahmin değerler grafiği Şekil 7.11'de görülmektedir. Şekil 7.12'de ise tahminlerin gerçek değerler çizgisine ne kadar yaklaştığı görülmektedir. Diğer modellerde başarı sıralaması en yüksekten aşağı doğru azalmaktadır.



Şekil 7.11 : En başarılı modele göre ilk 200 ve 30000 değer için gerçek değer-tahmin grafikleri

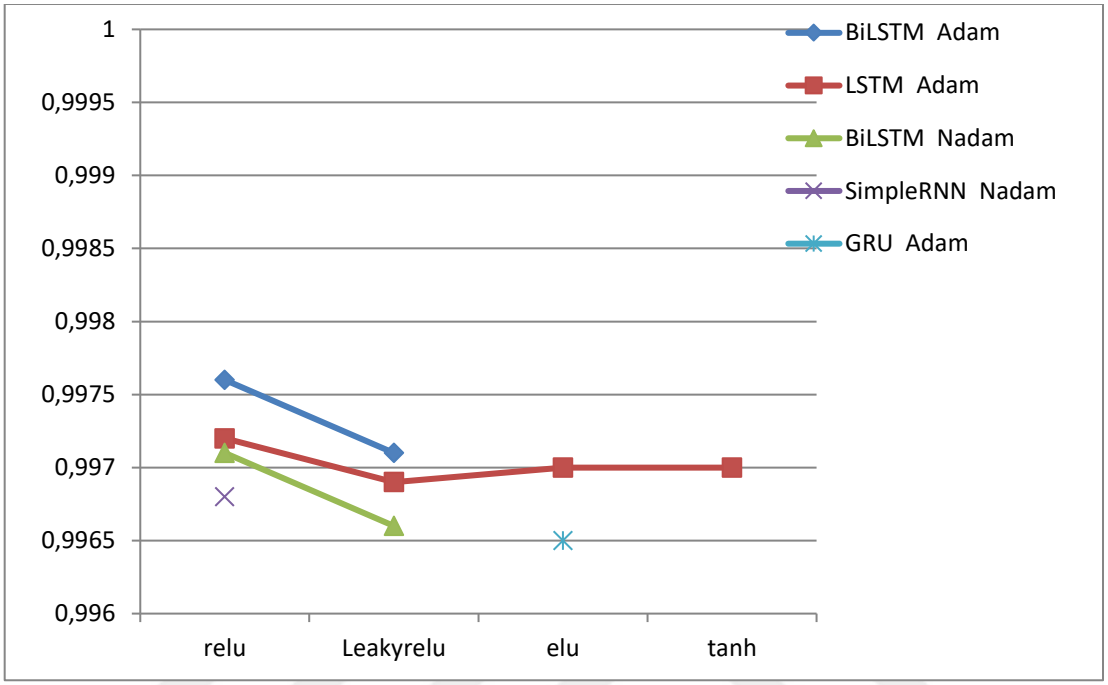


Şekil 7.12 : En başarılı modele göre tüm değerler için gerçek değer çizgisi ve tahmin popülasyonu

Çizelge 7.2 50 Epok İle Yapılan Çalışmalara Ait En İyi 10 Sonuç

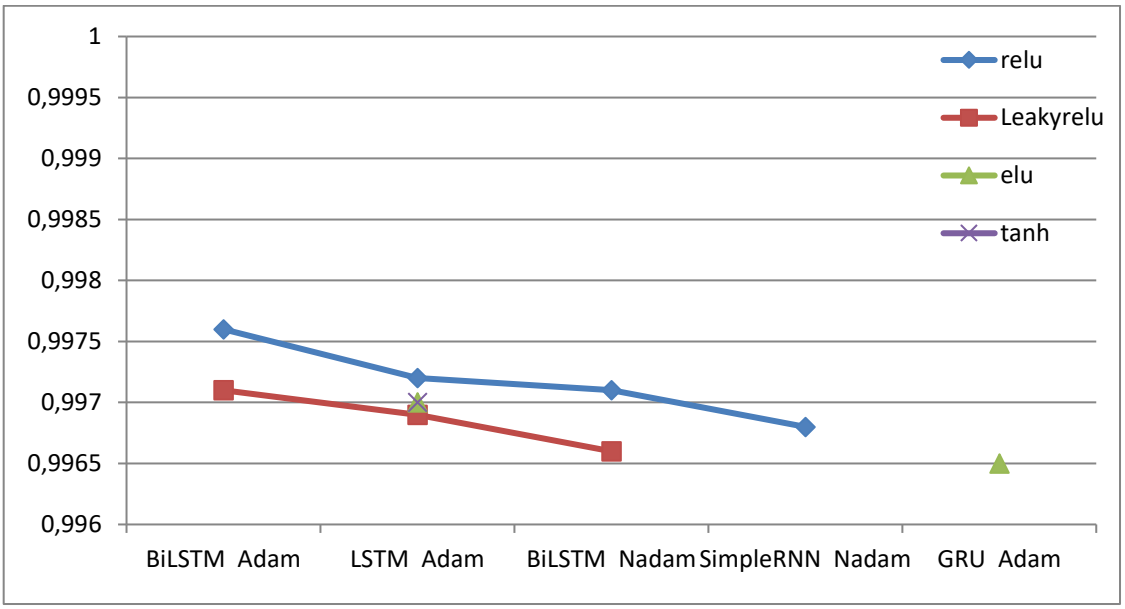
Model	Katman Sayısı	Aktivatör	Optimizer	Epok	test_mse	test-mae	test_r^2
BiLSTM	50	relu	Adam	50	313,04	223,66	0,9976
LSTM	50	relu	Adam	50	339,33	242,54	0,9972
BiLSTM	50	relu	Nadam	50	345,03	255,65	0,9971
BiLSTM	50	Leakyrelu	Adam	50	345,7	255,37	0,9971
LSTM	50	elu	Adam	50	345,19	249	0,997
LSTM	50	tanh	Adam	50	348,22	248,38	0,997
LSTM	50	Leakyrelu	Adam	50	356,31	257,81	0,9969
SimpleRNN	50	relu	Nadam	50	368,15	265,96	0,9968
BiLSTM	50	Leakyrelu	Nadam	50	375,89	275,41	0,9966
GRU	50	elu	Adam	50	373,14	275,81	0,9965

Çizelge 7.1'e göre en başarılı 10 modelin model ve optimizasyon fonksiyonuna göre grafiği Şekil 7.13'te görülmektedir. Buna göre en başarılı modeller sırasıyla BiLSTM-adam, LSTM-adam, BiLSTM-nadam, SimpleRNN-nadam ve GRU-adam olarak sıralanabilir.



Şekil 7.13 : En başarılı 10 sonuca göre aktivasyon fonksiyonu - R^2 grafiği

Çizelge 7.1'e göre en başarılı 10 modelin model ve aktivasyon fonksiyonuna göre grafiği Şekil 7.14'te görülmektedir. Buna göre en başarılı aktivasyon fonksiyonları sırasıyla relu, Leakyrelu, elu ve tanh olarak sıralanabilir.



Şekil 7.14 : En başarılı 10 sonuca göre model ve optimizer - R^2 grafiği

8. SONUÇ

Yapılacak tahmin çalışmasında hangi yöntemin başarılı olacağı hakkında kesin çalışmalar mevcut olmamakla birlikte bazı modellerin daha başarılı olduğu görülmekte ve bu konuda genelleştirme yapılabilecek uygulamalar üzerinde çalışmalar devam etmektedir.

Hangi problemde hangi modelin tercih edileceğine karar verme konusunda daha önce benzer problemlere nasıl yöntemlerle çözüm bulunduğuna bakmak faydalı olacaktır Literatür taraması yapılması bu konuda büyük yarar sağlayacaktır.

Yapılan ilk makine öğrenmesi çalışmasında Diyarbakır nüfus değerlerinin doğrusal regresyon çizgisi ile örtüştüğü görüldüğünden nüfus tahminleri doğrusal tahmin yöntemiyle tahmin edilmeye çalışılmıştır. Diyarbakır faturalandırılmış elektrik tüketim verileri üzerinden yapılan çalışmada ise en başarılı model olarak 4. dereceden poligonal regresyon yöntemi başarılı görülmüş fakat tahmin çalışması tüm modellere göre yapılmıştır.

Yapılan çoğu çalışmada elektrik enerjisi talep, üretim veya tüketim tahmini gibi ortak çalışmalar yapılıyor görünse dahi model geliştirme amacıyla ortak bir metot veya kural üretilememiş, hatta bazı yöntemler benzer veri setlerine uygulandığı halde farklı sonuçlar elde edilmiştir. Bu yüzden veri setleri için model önerisi yapmak amacıyla farklı tekniklerin karşılaştırılmasına ve farklı modellerin üretilmesine ihtiyaç vardır. Hatta bu konuda verilerin tek model üzerinden değil de hibrit model denilen birden fazla modelin kullanıldığı modeller geliştirilerek bu modeller üzerinden çalışmanın sürdürülmesi ileriki çalışmalarda kullanılacak diğer bir yöntem olarak dikkat çekmektedir.

Literatür çalışmalarında görüldüğü üzere tüm derin öğrenme modelleri ile çalışmalar mevcuttur. Yani akıllı şebeke ile ilgili yapılacak olan çalışmalarda tüm modellerin kullanılması olasıdır. Nesne tanıma alanlarında CNN ve sıralı veya dizi halindeki verilerin değerlendirilmesinde RNN modelleri daha öne çıkmaktadır. Fakat RNN modellerinin eski bilgileri hatırlayıp değerlendirmede sorun yaşaması yüzünden yine bir RNN modeli olan LSTM (hafızalı RNN) tercih edilmektedir. LSTM yöntemi özellikle tahmin etme konusunda çok başarılı çalışmalara sahne olduğu için tahmin problemlerinde çokça kullanılmaktadır. Özellikle yük tahmini veya üretilecek enerji tahmininde son yıllarda çokça çalışma yapılmaktadır.

Derin öğrenme yöntemleriyle yapılan 3 ayrı grup çalışması sonucuna göre; yapılan 352 denemenin 89 unda yani %25'inde R^2 değerinin 0,95 ve üzerinde sonuçlar verdiği görülmektedir. 3'ü rmsprop* yöntemiyle 2'si sgdfalse yöntemiyle olmak üzere 5 denemede gradyanların patlaması sorunuyla karşılaşılmıştır.

Öncelikle BiLSTM sonrasında ise LSTM modelinin en başarılı modeller olduğu görülmektedir.

En başarılı optimizasyon yöntemleri sırasıyla adam, nadam, adamax ve rmsprop olarak görülmektedir. Rmsprop* ve sgdtrue'nin ise daha başarısız olduğu görülmektedir. Ftrl, adadelta ve sgdfalse yöntemleriyle yapılan çalışmalarının %96,5 inin çok kötü sonuçlar verdiği görülmektedir. Literatürde genellikle adam optimizasyon yöntemi kullanılmaktadır. 32 grup çalışmanın 6'sında nadam, 3'ünde adamax ve 1'inde rmsprop yöntemi ile en yüksek değerler elde edilmektedir. Bu yönüyle literatürde pek yer bulmayan nadam ve adamax dikkat çekmektedir.

Epok sayısının artırılmasının sonuçları olumlu etkilediği görülmektedir.

En başarılı aktivasyon fonksiyonları sırasıyla relu, Leakyrelu, elu ve tanh olarak sıralanabilir. Sigmoid ve softmax'ın ise daha başarısız olduğu görülmektedir. Literatürdeki çalışmaların çoğu relu yöntemi ile yapılmaktadır. Çalışmadaki en başarılı modellere bakıldığında ise %40 relu, %30 Leakyrelu, %20 elu ve %10 tanh kullanımı dikkat çekmektedir.

Sgdtrue yönteminin sgd yönteminden başarılı olduğu, sgdfalse yönteminin ise çok başarısız olduğu ve rmsprop yönteminin rmsprop* yönteminden daha başarılı olduğu görülmektedir.

50 epok ile yapılmış en başarılı 10 çalışmaya bakıldığında BiLSTM modelinin relu ve adam kullanımıyla R^2 metriğine göre 0,9976 gibi çok yüksek bir değere ulaştığı görülmektedir

Yapılan çoğu çalışmada elektrik enerjisi talep, üretim veya tüketim tahmini gibi ortak çalışmalar yapılıyor görünse bile model geliştirme için ortak bir yöntem veya kural üretilememiştir. Bu yüzden bu çalışmada model önerisi yapmak amacıyla farklı tekniklerin karşılaştırılmakta, farklı modellerin üretilmekte ve farklı hiperparametreler denenmektedir.

Bu çalışmada verilerimizde mevsimsellik olduğu ve tüketim verileri daha düzenli olduğu için başarım yüksek çıkmaktadır. Buradaki yöntemlerin benzer veri setlerine

uygulandığında başarılı olacağı fakat farklı veri setlerinde başarımının azalacağı değerlendirilmektedir.

KAYNAKLAR

[1]. **Anil, K.J., Jianchang, M. ve Mohuiddin, K.M.** 1996, Artificial Neural Networks: A Tutorial, Computer - Special issue: neural computing, 29 (3), 31-44.

[2]. **Tiken, Cihan.** 2015, Derin Öğrenme Uygulamaları, İstanbul Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans Tezi, Haziran 2015.

[3]. **Salakhutdinov, R. ve Hinton, G.** 2012, An efficient learning procedure for deep Boltzmann machines, Neural Computation, 24 (8), 1967-2006.

[4]. **Deng, L., Platt, J. and Yu, D.** 2012, Scalable stacking and learning for building deep architectures, Int. Conf. on Acoustics, Speech and Signal Processing, 25-30 Mar. 2012 Kyoto, Japan, IEEE, 2133-2136.

[5]. **Collobert, R., Mobahi, H., Rathe, F. ve Weston, J.** 2012, Deep Learning via semi-supervised embedding, Neural Networks: Tricks of the Trade, In: Grégoire, M. (ed.), Chapter 26, Springer Berlin Heidelberg, USA, ISBN: 978-3-642-35288-1, 639-655.

[6]. **Bengio, Y., Courville, A., Manzagol, P. A. ve Vincent, P.** 2010, Why does unsupervised pre-training help deep learning? The Journal of Machine Learning Research, 11 (2), 625-660.

[7] **Url-3** <*http://blog.drhongtao.com/2014/10/very-short-short-medium-long-term-load-forecasting.html*>, date retrieved 20.07.2025.

[8] **Uçar MT., Kaygusuz A.** (2023). Cumhuriyetin 100. Yılında Cumhuriyet'ten Günümüze Ergani, Prof. Dr. Oktay BOZAN, Editör, Ekin Yayınevi, Bursa, s.385-413, 2023.

[9] **Uçar MT., Kaygusuz A.** (2025). Short-Term Energy Consumption Forecasting Analysis Using Different Optimization and Activation Functions with Deep Learning Models. Applied Sciences, 15(12), 6839. https://doi.org/10.3390/app15126839

[10] **G. Şişmanoğlu, F. Koçer, M. A. Önde ve Ö. K. Şahingöz**. 2020. Derin Öğrenme Yöntemleri ile Borsada Fiyat Tahmini. Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, cilt 9, no. 1, pp. 434 - 445, 2020.

[11] **Boa, W. ve Yue, J. ve Rao, Y.** (2017, Jul 14) ―A deep learning framework for financial time series using stacked autoencoders and long-short term memory‖, PLoS One. Published. doi:10.1371/journal.pone.0180944, URL: https://www.ncbi.nlm.nih.gov/pmc/ articles/ PMC5510866/

[12] **Bodyanskiy, Y. ve Popov, S.** (2006), Neural network approach to forecasting of quasiperiodic financial time series‖, Eur J Oper Res. 2006, 175(3):1357–66.

[13] **Hussain AJ., Knowles A., Lisboa PJG., El-Deredy W.** (2008), ―Financial time series prediction using polynomial pipelined neural Networks‖, Expert Systems with Applications an International Journal. 2008;35(3):1186–99

[14] **Jerzy Korczak, Marcin Hernes.** (2017). Deep Learning for Financial Time Series Forecasting in A-Trader System. Federated Conference on Computer Science and Information Systems (FedCSIS), 2017.

[15] **Manuel R. Vargas, Beatriz S. L. P. de Lima, Alexander G. Evsukoff**. (2017). Deep Learning for Stock Market Prediction from Financial News Articles

[16] **Moinul Hossain, Banafsheh Rekabdar, Banafsheh Rekabdar, Sergiu Dascalu**. (2015). Forecasting the Weather of Nevada: A Deep Learning Approach

[17] **A. Ö. Akyüz, K. Kumaş, M. Ayan ve A. Güngör**. 2020. Antalya İli Meteorolojik Verileri Yardımıyla Hava Sıcaklığının Yapay Sinir Ağları Metodu ile Tahmini, Gümüşhane Üniversitesi Fen Bilimleri Dergisi, cilt 10, no. 1, pp. 146-154, 2020.

[18] **Hakan Acikgoz, Deniz Korkmaz.** 2025. Short-term offshore wind speed forecasting approach based on multi-stage decomposition and deep residual network with self-attention, Engineering Applications of Artificial Intelligence, Volume 146, 2025, 110313, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2025.110313.

[19] **H. Kılıç, B. Gümüş ve M. Yılmaz.** 2016. Güneydoğu Anadolu bölgesi için global güneş ışımasının ve güneşlenme süresinin istatiksel metodlar ile tahmin edilmesi ve karşılaştırılması, Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, cilt 7, no. 1, pp. 73-83, 2016.

[20] **Y. Gültepe**. 2019. Makine Öğrenmesi Algoritmaları ile Hava Kirliliği Tahmini Üzerine Karşılaştırmalı Bir Değerlendirme, Avrupa Bilim ve Teknoloji Dergisi, no. 16, pp. 8-15, 2019.

[21] **J. Hong, Z. Wang, and Y. Yao**. 2019 "Fault prognosis of battery system based on accurate voltage abnormity prognosis using long short-term memory neural networks," Applied Energy, vol. 251, p. 113381, 2019.

[22] **H. Wang, H. Wang, G. Jiang, J. Li ve Y. Wang**. 2019 "Early fault detection of wind turbines based on operational condition clustering and optimized deep belief network modeling," Energies, vol. 12, no. 6, p. 984, 2019.

[23] **S. Dong, J. Xiao, X. Hu, N. Fang, L. Liu ve J. Yao**. 2022. Deep transfer learning based on Bi-LSTM and attention for remaining useful life prediction of rolling bearing, Reliability Engineering & System Safety, cilt 230, 2022.

[24] **Y. Ran, X. Zhou, P. Lin, Y. Wen ve R. Deng.** 2019. A Survey of Predictive Maintenance: Systems, Purposes and Approaches, IEEE Communications Surveys & Tutorials, cilt 20, no. 20, 2019.

[25] **Fentis, A. ve diğ.** (2016). Short-term PV power forecasting using Support Vector Regression and local monitoring data. In: 2016 International Renewable and Sustainable Energy Conference (IRSEC). IEEE, pp. 1092–1097.

[26] **Alfadda, A. ve diğ.** (2017). Hour-ahead solar PV power forecasting using SVR based approach. In: 2017 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT). IEEE, pp. 1–5.

[27] **F. Alpsalaz, Z. Yalçinöz, A. Kaygusuz ve MS Mamiş,** "Yapay Sinir Ağı Modeli Kullanılarak Elektrik İletim Hatlarında Arıza Yeri Tahmini", 2024 8. Uluslararası Yapay Zeka ve Veri İşleme Sempozyumu (IDAP) , Malatya, Türkiye, 2024, ss. 1-6, doi: 10.1109/IDAP64064.2024.10710637.

[28] **C. Kuster, Y. Rezgui, M. Mourshed.** 2017. Elektrik yükü tahmin modelleri: kritik bir sistematik inceleme. Sustain Cities Soc, 35, (2017), s. 257 - 270

[29] **Z. Tan, J. Zhang, J. Wang, J. Xu.** 2010 ARIMA ve GARCH modelleri ile birlikte dalgacık dönüşümü kullanılarak gün öncesi elektrik fiyatı tahmini. Appl Energy, 87 (2010), s. 3606 – 3610

[30] **İdil IŞIKLI ESENER, Tolga YÜKSEL, Mehmet KURBAN.** 2012 ELECO '2012 Elektrik - Elektronik ve Bilgisayar Mühendisliği Sempozyumu, 29 Kasım - 01 Aralık 2012, Bursa

[31] **T. Akman, C. Yılmaz ve Y. Sönmez**. 2018. Elektrik Yükü Tahmin Yöntemlerinin Analizi, Gazi Mühendislik Bilimleri Dergisi, cilt 4, no. 3, pp. 168-175, 2018.

[32] **A. Arı ve H. Önder**. 2013. Farklı Veri Yapılarında Kullanılabilecek Regresyon Yöntemleri, Anadolu Tarım Bilimleri Dergisi, cilt 28, no. 3, pp. 168-174, 2013.

[33] **H. A. Es, F. Y. Kalender ve C. Hamzaçebi**. 2014. Yapay sinir ağları ile Türkiye net enerji talep tahmini, Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, cilt 29, no. 3, pp. 495-504, 2014.

[34] **C. Karaca ve H. Karacan.** 2016. Çoklu Regresyon Metoduyla Elektrik Tüketim Talebini Etkileyen Faktörlerin İncelenmesi, Selçuk Üniversitesi Mühendislik, Bilim Ve Teknoloji Dergisi, cilt 4, no. 3, pp. 182-195, 2016.

[35] **Y. Alcan, M. Demir, S. Ersöz ve Ö. Alcan**. 2017. Sinop İli Elektrik Enerji Tüketiminin Farklı Yöntemler İle Tahmini Ve Karşılaştırılması, UMTEB I-Internatıonal Congress On Vocatıonal and Technıcal Scıences, Batumi-Georgia, April 8-12-2017.

[36] **A. O. Özkan, A. Özkan ve H. Soy**. 2019. Türkiye'nin Enerji Talebinin Regresyon Analiz Teknikleriyle Uzun Dönem Tahmini. CİLT 3, 2019, 98, 8. Uluslararası Meslek Yüksekokulları Sempozyumu - UMYOS'19, Sinop, 2019.

[37] **G. Turan**. 2020. Hane Halkı Elektrik Kullanımının Makine Öğrenme Algoritmalarıyla Analizi, YBS Ansiklopedi, cilt 8, no. 2, pp. 1-12, Haziran 2020.

[38] **E. E. Nebati, M. Taş ve G. Ertaş**. 2021. Türkiye'de Elektrik Tüketiminde Talep Tahmini: Zaman Serisi Ve Regresyon Analizi İle Karşılaştırma, Avrupa Bilim ve Teknoloji Dergisi, cilt Ek Sayı 1, no. 31, pp. 348-357, Aralık 2021.

[39] **H. Ülkü ve Ş. Yalpır**. 2021. Enerji talep tahmini için metodoloji geliştirme: 2030 yılı Türkiye örneği, Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi., cilt 10, no. 1, pp. 188-201, 2021.

[40] **Y. Er ve E. Karaca**. 2021. Farklı Yöntemlerle Karadeniz Bölgesi'nin Aylık Elektrik Tüketim Tahmini,» %1 içinde Sürdürülebilirlik İçin Akademik Araştırmalar - I, C. Kahraman, Dü., Artikel Akademi, 2021, pp. 137-147.

[41] **H. Son ve C. Kim**. 2017. Short-term forecasting of electricity demand for the residential sector using weather and socialvariables,» Resources, conservation and recycling, cilt 123, pp. 200-207, 2017.

[42] **O. Kaynar, A. G. Yüksek ve F. Demirkoparan**. 2016. Genetik Algoritma İle Eğitilmiş Destek Vektör Regresyon Kullanılarak Türkiye'nin Elektrik Tüketim Tahmini,» İstanbul Üniversitesi İktisat Fakültesi Mecmuası, cilt 66, no. 2, pp. 45-60, 2016.

[43] **Zhang X. M., Grolinger K., Capretz M. A. M**. 2018. Forecasting Residental Energy Consumption Using Support Vektor Regressions, Proc. Of the IEEE International Conference on Machine Learning and Applications, 17-20 December 2018, Orlando, FL, 110-117

[44] **Yang, M. et al.** (2016). Parameters Optimization Improvement of SVM on Load Forecasting. In: 2016 8t h International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC). IEEE, pp. 257–260.

[45] **M. E. Baltaş ve C. Akbay**. 2021. Akdeniz Elektrik Dağıtım Bölgesi (Antalya-Isparta-Burdur) Elektrik Tüketim Talep Tahmini. Yönetim ve Ekonomi Araştırmaları Dergisi, cilt 19, no. 2, pp. 222-238, Haziran 2021.

[46] **Zhikun Ding, Weilin Chen, Ting Hu, Xiaoxiao Xu.** 2021 Evolutionary double attention-based long short-term memory model for building energy prediction: Case study of a green building, Applied Energy, Volume 288, 2021, 116660, ISSN 0306-2619, https://doi.org/10.1016/j.apenergy.2021.116660.
URL: https://www.sciencedirect.com/science/article/pii/S0306261921001902

[47] **C. Fan, Y. Sun, Y. Zhao, M. Song, J. Wang.** 2019 Gelişmiş bina enerjisi tahmini için derin öğrenmeye dayalı özellik mühendisliği yöntemleri. Appl Energy, 240 (2019), s. 35 – 45

[48] **M. Mutlu YAPICI, Adem TEKEREK, Nurettin TOPALOĞLU**. 2019 ” Derin Öğrenme Araştırma Alanlarının Literatür Taraması,” Gazi Mühendislik Bilimleri Dergisi 2019, 5(3): 188-215 Derleme Makalesi/Review Article https://dergipark.org.tr/gmbd
URL: https://www.researchgate.net/profile/Adem_Tekerek/publication/338238434_Literatüre_Review_of_Deep_Learning_Research_Areas/links/5e11ea2392851c8364b13196/Literatüre-Review-of-Deep-Learning-Research-Areas.pdf

[49] **Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen ve Ruilong Deng**. 2019 “A Survey of Predictive Maintenance: Systems, Purposes and Approaches,” Ieee Communıcatıons Surveys & Tutorıals, Vol. XX, No. XX, Nov. 2019

[50] **C. Fan, J. Wang, W. Gang, S. Li.** 2019 Kısa vadeli bina enerji tahminleri için derin tekrarlayan sinir ağı tabanlı stratejilerin değerlendirilmesi. Appl Energy, 236 (2019), s. 700 - 710

[51] **W. Kong, ZY Dong, Y. Jia, DJ Hill, Y. Xu ve Y. Zhang**. 2019 "LSTM Recurrent Neural Network'e Dayalı Kısa Vadeli Konut Yükü Tahmini", IEEE Transactions on Smart Grid, cilt. 10, hayır. 1, s. 841-851, Ocak 2019, doi: 10.1109/TSG.2017.2753802.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8039509&isnumber=8595470

[52] **Shahzad Muzaffar, Afshin Afshari.** 2019. Short-Term Load Forecasts Using LSTM Networks, Energy Procedia, Volume 158, 2019, Pages 2922-2927, ISSN 1876-6102, https://doi.org/10.1016/j.egypro.2019.01.952.
URL: https://www.sciencedirect.com/science/article/pii/S1876610219310008

[53] **A. Wong ve diğ.** 2022. Saatlik Elektrik Kullanımının Günlük Tahmininde Makine Öğrenimi Modelleri Uygulaması, 2022 IEEE Uluslararası Sistemler Konferansı (SysCon), 2022, pp. 1-5, doi: 10.1109/SysCon53536.2022.9773835.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9773835&isnumber=9773791

[54] **S. Siami-Namini, N. Tavakoli ve A. Siami Namin**. 2018 "A Comparison of ARIMA and LSTM in Forecasting Time Series", 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, s. 1394-1401, doi: 10.1109/ICMLA.2018.00227.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8614252&isnumber=8614025

[55] **IA Dahlan, D. Ariateja, F. Hamami ve Heryanto**. 2021 "LSTM Sinir Ağı kullanarak Akıllı Akıllı Enerji Oluşturmanın Uygulaması", 2021 Uluslararası Yapay Zeka ve Mekatronik Sistemler Konferansı (AIMS), 2021, s. 1-5, doi: 10.1109/AIMS52415.2021.9466046.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9466046&isnumber=9466004

[56] **F. Li, X. Yu, X. Tian ve Z. Zhao**. 2021 "Enerji Depolamayı Düşünerek LSTM-RNN Kullanarak Bir Endüstri Parkı İçin Kısa Vadeli Yük Tahmini," 2021 3. Asya Enerji ve Elektrik Mühendisliği Sempozyumu (AEEES), 2021, s 684-689, doi: 10.1109/AEEES51875.2021.9403118.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9403118&isnumber=9402948

[57] **Jun Lin, Jin Ma, Jianguo Zhu, Yu Cui**. 2021 Short-term load forecasting based on LSTM networks considering attention mechanism, International Journal of Electrical Power & Energy Systems, Volume 137, 2022, 107818, ISSN 0142-0615, https://doi.org/10.1016/j.ijepes.2021.107818.
URL: https://www.sciencedirect.com/science/article/pii/S0142061521010346

[58] **Zhikun Ding, Weilin Chen, Ting Hu, Xiaoxiao Xu**. 2021 Evolutionary double attention-based long short-term memory model for building energy prediction: Case study of a green building. Applied Energy, Volume 288, 2021, 116660, ISSN 0306-2619, https://doi.org/10.1016/j.apenergy.2021.116660.
URL: https://www.sciencedirect.com/science/article/pii/S0306261921001902

[59] **Jian Qi Wang, Yu Du, Jing Wang.** 2020 LSTM based long-term energy consumption prediction with periodicity, Energy, Volume 197, 2020, 117197, ISSN 0360-5442, https://doi.org/10.1016/j.energy.2020.117197.
URL: https://www.sciencedirect.com/science/article/pii/S0360544220303042

[60] **Zilong Zhao, Jinrui Tang, Jianchao Liu, Ganheng Ge, Binyu Xiong, Yang Li**. 2022 Short-term microgrid load probability density forecasting method based on k-means-deep learning quantile regression, Energy Reports, Volume 8, Supplement 5, 2022, Pages 1386-1397, ISSN 2352-4847, https://doi.org/10.1016/j.egyr.2022.03.117.

URL: https://www.sciencedirect.com/science/article/pii/S2352484722006758

[61] **Le, T., Vo, M. T., Kieu, T., Hwang, E., Rho, S., & Baik, S. W.** (2020). Multiple electric energy consumption forecasting using a cluster-based strategy for transfer learning in smart building. Sensors, 20(9), 2668.

[62] **Yi Wang, Dahua Gan, Mingyang Sun, Ning Zhang, Zongxiang Lu, Chongqing Kang**. 2019 Probabilistic individual load forecasting using pinball loss guided LSTM, Applied Energy, Volume 235, 2019, Pages 10-20, ISSN 0306-2619, https://doi.org/10.1016/j.apenergy.2018.10.078. URL: https://www.sciencedirect.com/science/article/pii/S0306261918316465

[63] **X. Shao ve CS Kim**. 2020. Müşteri Davranışını Göz önünde bulundurarak Zaman Konumlu Çok Kanallı LSTM Kullanarak Çok Adımlı Kısa Vadeli Güç Tüketimi Tahmini, IEEE Access, cilt. 8, s. 125263-125273, 2020, doi: 10.1109/ACCESS.2020.3007163.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9133384&isnumber=8948470

[64] **S. Nazir, Azlan Ab Aziz, J. Hosen, Nor Azlina Aziz ve G. Ramana Murthy.** 2020 Journal of Physics: Conference Series, Volume 1933, Virtual Conference on Engineering, Science and Technology (ViCEST) 2020, 12-13 Ağustos 2020, Kuala Lumpur, Malezya https://doi.org/10.1088/1742-6596/1933/1/012054. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1933/1/012054/meta

[65] **Ungureanu, S.; Topa, V.; Cziker.** 2021. AC Kısa Vadeli Yük Tahmini için Derin Öğrenme Endüstriyel Tüketici Vaka Çalışması. Uygulama bilim 2021, 11, 10126. https://doi.org/10.3390/app112110126. URL: https://www.mdpi.com/2076-3417/11/21/10126/htm

[66] **AJ Almalki ve P. Wocjan**. 2020 "GRU-tabanlı Derin Öğrenme Modeline dayalı Tahmin Yöntemi," 2020 Uluslararası Hesaplamalı Bilim ve Hesaplamalı Zeka (CSCI) Konferansı , 2020, s. 534-538, doi: 10.1109/CSCI51800.2020.00096.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9458184&isnumber=9457560

[67] **F. Karim, S. Majumdar ve H. Darabi**. 2019 "Insights Into LSTM Fully Convolutional Networks for Time Series Classification", IEEE Access, cilt. 7, s. 67718-67725, 2019, doi: 10.1109/ACCESS.2019.2916828. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8713870&isnumber=8600701

[68] **S. Yang, X. Yu ve Y. Zhou**. 2020 "LSTM ve GRU Sinir Ağı Performans Karşılaştırma Çalışması: Yelp İnceleme Veri Kümesini Örnek Olarak Almak", 2020 Uluslararası Elektronik İletişim ve Yapay Zeka Çalıştayı (IWECAI), 2020, s. 98- 101, doi: 10.1109/IWECAI50956.2020.00027. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9221727&isnumber=9221654

[69] **S. Siami-Namini, N. Tavakoli ve AS Namin**. 2019 "The Performance of LSTM and BiLSTM in Forecasting Time Series", 2019 IEEE International Conference

on Big Data (Big Data) , 2019, s. 3285-3292, doi: 10.1109/ BigData47090.2019.9005997.
URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9005997 &isnumber=9005444

[70] **Farah Shahid, Aneela Zameer, Muhammad Muneeb**. 2020. Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM, Chaos, Solitons & Fractals, Volume 140, 2020, 110212, ISSN 0960-0779, https://doi.org/10.1016/j.chaos.2020.110212.
URL: https://www.sciencedirect.com/science/article/pii/S0960077920306081

[71] **Gözde ŞİŞMANOĞLU, Furkan KOÇER, Mehmet Ali ÖNDE, Özgür Koray ŞAHİNGÖZ.** 2020 BEÜ Fen Bilimleri Dergisi BEU Journal of Science 9 (1), 434-445, 2020 9 (1), 434-445, 2020.
URL: https://app.trdizin.gov.tr/makale/TXpjNE5qYzVPUT09/derin-ogrenme-yontemleri-ile-borsada-fiyat-tahmini

[72] **Irfan M, Shaf A, Ali T, Zafar M, Rahman S, Mursal SNF ve diğ.** (2023). Multi-region electricity demand prediction with ensemble deep neural networks. PLoS ONE 18(5): e0285456. https://doi.org/10.1371/journal.pone.0285456

[73] **Alioghli, A. A. ve Yildrim Okay, F.** (2024). IoT-Based Energy Consumption Prediction Using Transformers. GU J Sci, Part A, 11(2), 304-323. doi:10.54287/gujsa.1438011

[74] **Zulfiqar Ahmad Khan, Amin Ullah, Ijaz Ul Haq, Mohamed Hamdy, Gerardo Maria Mauro, Khan Muhammad, Mohammad Hijji, Sung Wook Baik**. 2022 Efficient Short-Term Electricity Load Forecasting for Effective Energy Management, Sustainable Energy Technologies and Assessments, Volume 53, Part A, 2022, 102337, ISSN 2213-1388, https://doi.org/10.1016/j.seta.2022.102337.
(https://www.sciencedirect.com/science/article/pii/S2213138822003897)

[75] **Ergezer, H.& Dikmen, M.& Özdemir, E.** (2003). Yapay Sinir Ağları Ve Tanıma Sistemleri Pivolka. Uygulamalı Yerbilimleri Dergisi, 2(6), s.71-79, Kocaeli.

[76] **Der: Becerikli, Y.** Yapay Sinir Ağları ile Duygu Analizi. Kocaeli Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği.

[77] **Ataseven, B.** (2013) Yapay Sinir Ağları ile Öngörü Modellemesi. Dergipark, s.101-115, İstanbul.

[78] **Çayıroğlu, İ.** İleri Algoritma Analizi - 5 Yapay Sinir Ağları. Karabük Üniversitesi Mühendislik Fakültesi

[79] **Url-3** <*https://miuul.com/blog/derin-ogrenme-ile-derinlere-yolculuk*>, date retrieved 20.07.2025.

[80] **Url-3** < *https://ayyucekizrak.medium.com/%C5%9Fu-kara-kutuyu-a%C3%A7alim-yapay-sinir-a%C4%9Flar%C4%B1-7b65c6a5264a*>, date retrieved 20.07.2025.

[81] **Url-3** <*http://www.derinogrenme.com/2017/03/04/yapay-sinir-aglari*>, date retrieved 20.07.2025.

[82] **Aşkın, D. & İskender, İ. ve Mamızadeh, A.** (2011) Farklı Yapay Sinir Ağları Yöntemlerini Kullanarak Kuru Tip Transformatör Sargısının Termal Analizi. Gazi Üniv. Müh. Mim. Fak. Der. Cilt 26, No 4, 905-913, Ankara.

[83] **Url-3** *<https://medium.com/@ayyucekizrak/derin-%C3%B6%C4%9Frenme-i%C3%A7in-aktivasyon-fonksiyonlar%C4%B1n%C4%B1n-kar%C5%9F%C4%B1la%C5%9Ft%C4%B1r%C4%B1lmas%C4%B1-cee17fd1d9cd>*, date retrieved 20.07.2025.

[84] **Url-3** <*https://www.mql5.com/tr/articles/12627*>, date retrieved 20.07.2025.

[85] **Url-3** *<https://burakdilber91.medium.com/yapay-sinir-a%C4%9Flar%C4%B1nda-aktivasyon-fonksiyonlar%C4%B1-cec328e2c55c>*, date retrieved 20.07.2025.

[86] **Murat BAĞ**. 2019 "Derin Öğrenme Kullanarak Ip Üzerinden Ses Hizmeti Veren Şebekelerde Sahtekarlığa Yönelik Çağrıların Tespiti", Yüksek Lisans Tezi, Ankara Üniversitesi Fen Bilimleri Enstitüsü Yüksek Lisans Tezi. Elektrik-Elektronik Mühendisliği Anabilim Dalı Ankara 2019

[87] **Caner Boyraz.** 2019 Derin Öğrenme İle Debi Tahmini. Yüksek Lisans Tezi. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, (2019). s.14,15.

[88] **Diyarbakır Valiliği ve Karacadağ Kalkınma Ajansı.** (2016). İstatistiklerle Diyarbakır. https://www.karacadag.gov.tr/Dokuman/Dosya/www.karacadag.org.tr_174_XN3L37GB_istatistiklerle_diyarbakir_2016.pdf. 12 07, 2022

[89] **EPDK.** (2022). EPDK I Enerji Piyasası Düzenleme Kurulu: 2015-2021 Elektrik Piyasası Gelişim Raporu. EPDK. 2022 tarihinde https://www.epdk.gov.tr/Detay/Icerik/3-0-24/elektrikyillik-sektor-raporu

[90] **Nufusu.com.** (2022). https://www.nufusu.com/ilce/ergani_diyarbakir-nufusu. 12.06.2022

[91] **TEDAS.** (2022). TEDAŞ: 2007-2012 Faaliyet Raporları. TEDAŞ. https://www.tedas.gov.tr/#!tedas_faaliyet_raporlari

[92] **TUİK.** (2022). Yıllara Göre İl Nüfusları 2000-2021. Türkiye İstatistik Kurumu. TUIK. 2022 tarihinde https://data.tuik.gov.tr/

[93] **Aslan O., Cheng H., Schuurmans D. ve Zhang X.** 2013. Convex two-layer modeling, In Proceedings of Neural Information Processing Systems (NIPS).

[94] **Cho Y. ve Saul L.** 2009. "Kernel Methods for Deep Learning", In Proceedings of Neural Information Processing Systems (NIPs), pages 342-350.

[95] **Deng L., Tur G. ve Hakkani-Tur D.** 2012. "Use of Kernel Deep Convex Networks and End-To-End Learning for Spoken Language Understanding", In Proceedings of IEEE Workshop on Spoken Language Technologies.

[96] **Viyals O., Jia Y., Deng L. ve Darrell T.** 2012. Learning with Recursive Perceptual Representations, In Proceedings of Neural Information Processing Systems (NIPs).

[97] **Li Deng, Dong Yu**. 2014. Deep Learning Methods and Applications, Foundations and Trends Signal Processing, vol.7 ,198-250.

[98] **George D.** 2008. “How to Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition”, Ph.D. Thesis, Standford University.

[99] **Url-3** <*https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4*>, date retrieved 20.07.2025.

[100] **Url-3** <*https://tr.newworldai.com/deep-learning-derin-ogrenme-nedir/*>, date retrieved 20.07.2025.

[101] **J. L. Elman**. 1990. Finding Structure in Time, Cogn. Sci., vol. 14, no. 2, pp. 179–211, Mar. 1990.

[102] **T.** Mikolov ve diğ. 2010. Recurrent neural network based language model, in Interspeech. 26-30 September 2010, Makuhari, Chiba, Japan.

[103] **Abdulkadir ŞEKER, Banu DİRİ, Hasan Hüseyin BALIK**. 2017 “Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme”, Şeker, Diri, Balık / Gazi Mühendislik Bilimleri Dergisi 3 (3). (2017) 47-64

[104] **A. Graves ve N. Jaitly**. 2014 “Towards End-To-End Speech Recognition with Recurrent Neural Networks.,” in ICML, 2014, pp. 1764–1772.

[105] **A. Karpathy ve L. Fei-Fei**. 2015 “Deep VisualSemantic Alignments for Generating Image Descriptions,” in CVPR, 2015, pp. 3128–3137.

[106] **Url-3** <*http://colah.github.io/posts/2015-08-Understanding-LSTMs/*>, date retrieved 20.07.2025.

[107] **Y. Bengio, P. Simard ve P. Frasconi**. 1994 “Learning long-term dependencies with gradient descent is difficult,” IEEE Trans. Neural Networks, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[108] **S. Hochreiter ve J. Schmidhuber**. 1997 “Long ShortTerm Memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[109] **K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink ve J. Schmidhuber**. 2015 “LSTM: A Search Space Odyssey,” Mar. 2015.

[110] **F. A. Gers, J. Schmidhuber ve F. Cummins**. 2000 “Learning to forget: continual prediction with LSTM.,” Neural Comput., vol. 12, no. 10, pp. 2451– 71, Oct. 2000.

[111] **F. A. Gers ve J. Schmidhuber**. 2000 “Recurrent nets that time and count,” in Proceedings of the IEEEINNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium, 2000, pp. 189–194 vol.3.

[112] **Gürkan Kavuran.** 2022. When machine learning meets fractional-order chaotic signals: detecting dynamical variations, Chaos, Solitons & Fractals, Volume 157, 2022, 111908, ISSN 0960-0779, https://doi.org/10.1016/j.chaos.2022.111908.

[113] **A. Graves ve J. Schmidhuber** 2005 “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” Neural Networks, vol. 18, no. 5–6, pp. 602–610, Jul. 2005.

[114] **A. Graves, A. Mohamed ve G. Hinton.** 2013 “Speech recognition with deep recurrent neural networks,” in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 6645–6649.

[115] **S. Fernández, A. Graves ve J. Schmidhuber**. 2007 “An Application of Recurrent Neural Networks to Discriminative Keyword Spotting,” in International Conference on Artificial Neural Networks, 2007, pp. 220–229.

[116] **D. Eck ve J. Schmidhuber**. 2002. Learning the Long-Term Structure of the Blues, Springer, Berlin, Heidelberg, 2002, pp. 284–289.

[117] **S. Hochreiter, M. Heusel ve K. Obermayer**. 2007. Fast model-based protein homology detection without alignment, Bioinformatics, vol. 23, no. 14, pp. 1728–1736, Jul. 2007.

[118] **H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll ve J. Schmidhuber**. 2006. A System for Robotic Heart Surgery that Learns to Tie Knots Using Recurrent Neural Networks, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 543–548.

[119] **J. Schmidhuber, F. Gers ve D. Eck**. 2002. Learning Nonregular Languages: A Comparison of Simple Recurrent Networks and LSTM, Neural Comput., vol. 14, no. 9, pp. 2039–2041, Sep. 2002.

[120] **Alex Graves ve Jürgen Schmidhuber.** 2008. Offline handwriting recognition with multidimensional recurrent neural networks. In Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'08). Curran Associates Inc., Red Hook, NY, USA, 545–552.

[121] **L. Wensheng, Z. Hongshi, W. Kun, W. Zhen ve C. Weilong.** 2024. "Geliştirilmiş Aritmetik Optimizasyon Algoritması ile Optimize Edilen GRU Sinir Ağına Dayalı Kısa Vadeli Net Yük Tahmin Modeli", *2024 IEEE 7. Uluslararası Bilgi Sistemleri ve Bilgisayar Destekli Eğitim Konferansı (ICISCAE)*, Dalian, Çin, 2024, s. 52-56, doi: 10.1109/ICISCAE62304.2024.10761548.

[122] **M. Subramanian, LS S ve R. VR.** 2023. "Melodi Üretimi için Derin Öğrenme Yaklaşımları: LSTM, BILSTM ve GRU Modellerini Kullanarak Bir Değerlendirme", 2023 14. Uluslararası Bilgisayar İletişim ve Ağ Teknolojileri Konferansı (ICCCNT) , Delhi, Hindistan, 2023, s. 1-6, doi: 10.1109/ICCCNT56998.2023.10308344.

[123] **J. Kim, N. Moon**. 2019. Tahmin için birden fazla alanda çok değişkenli zaman serisi verilerine dayalı BiLSTM modeli ticaret alanı, J Ambient Intell Humaniz Comput (2019) 1–10. https://doi.org/10.1007/S12652-019-01398-9/METRİKLE

[124] **Url-3** <*https://www.researchgate.net/publication/380672880_Solar_Energy_Production_Forecasting_A_Comparative_Study_of_Bi-LSTM_LSTM_XGBoost_Models_with_Activation_Function_Analysis* >, Güneş Enerjisi Üretim Tahmini: Aktivasyon Fonksiyonu Analizi ile Bi-LSTM, LSTM, XGBoost Modellerinin Karşılaştırmalı Çalışması. date retrieved 20.07.2025.

[125] **Şahin, O., Yayla, R.** (2024). BiLSTM Derin Öğrenme Yöntemi ile Uzun Metinlerden Yeni Özet Metinlerin Türetilmesi. Karadeniz Fen Bilimleri Dergisi, 14(3), 1096-1119

[126] **Url-3** <*https://medium.com/@ayyucekizrak/yapay-zekâya-ba%C5%9Flama-rehberi-91e79d3de8e1*>, date retrieved 20.07.2025.

[127] **Okan Alkan**. 2019. Parkinson Hastalığının Teşhisinde Derin Öğrenme Yöntemi İle Spect Görüntü Analizi, Ağrı İbrahim Çeçen Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı Yüksek Lisans Tezi, Ağrı-2019, s:20.

[128] **Url-3** <*https://www.themt.co/blog/109-bilgisayar-grafigi/317-gpu-ve-render-hakkinda-bilinmesi-gerekenler*>, date retrieved 20.07.2025.

[129] **Url-3** < *https://www.nvidia.com/tr-tr/about-nvidia/ai-computing/*>, date retrieved 20.07.2025.

[130] **Url-3** < *https://medium.com/@ayyucekizrak/ad%C4%B1m-ad%C4%B1m-google-colab-%C3%BCcretsiz-tpu-kullan%C4%B1m%C4%B1-621dc6e5487d* >, date retrieved 20.07.2025.

[131] **Url-3** <*https://medium.com/@piyushkashyap045/understanding-sgd-with-momentum-in-deep-learning-a-beginner-friendly-guide-0252ede605b4*>, date retrieved 20.07.2025.

[132] **Duchi, John, Elad Hazan, ve Yoram Singer.** 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. The Journal of Machine Learning Research, 2: 2121-2159.

[133] **Zeiler, Matthew D.** 2012. Adadelta: An Adaptıve Learnıng Rate Method. arXiv:1212.5701.

[134] **Ruder, Sebastian.** 2016. An overview of gradient descent optimization algorithms. arXiv:1609.04747.

[135] **Kingma, Diederik P. ve Jimmy Lei Ba.** 2015. Adam: A Method for Stochastic Optimization. Interna-tional Conference on Learning Representations. San Diego. 1-13.

[136] **Url-3** <*https://medium.com/@nerdjock/deep-learning-course-lesson-7-5-nadam-nesterov-accelerated-adaptive-moment-estimation-efe9050d5b9b*>, date retrieved 20.07.2025.

[137] **Url-3** <*https://itsudit.medium.com/mastering-the-art-of-optimization-a-comprehensive-guide-to-adagrad-rmsprop-and-adamax-d6e49cd3b0c2*>, date retrieved 20.07.2025.

[138] **S. Sharma, N. Sharma, A. Sindgi, A. Malhan, P. T ve S. E.** 2024. "Müzik Ruh Hali Sınıflandırmasının Yoğun ve Derin Modeli için FTRL algoritmasının Performans Analizi", 2024 Akıllı ve Sürdürülebilir Teknolojideki Son Yenilikler Uluslararası Konferansı (ICRISST) , Bengaluru, Hindistan, 2024, s. 1-4, doi: 10.1109/ICRISST59181.2024.10921949.

[139] **Doğan, O**. 2017. Ücretsiz Veri Madenciliği Araçları ve Türkiye'de Bilinirlikleri Üzerine Bir Araştırma, Ege Stratejik Araştırmalar Dergisi, (2017), 8(1): 77-93.

[140] **Url-3** <*https://bulutistan.com/blog/Tensorflow-nedir/*>, date retrieved 20.07.2025.

[141] **Url-3** <*https://miracozturk.com/python-kutuphaneleri-ve-ozellikleri/*>, date retrieved 20.07.2025.

[142] **Y. Jia ve diğ**. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding, Proceedings of the 22nd ACM International Conference on Multimedia. November 2014. Pages 675–678 .https://doi.org/10.1145/2647868.2654889

[143] **Url-3** <*https://machinelearningmastery.com/popular-deep-learning-libraries/*>, Keras Documentation. Jason Brownlee, "Popular Deep Learning Libraries", date retrieved 20.07.2025.

[144] **Url-3** <*https://keras.io/*>, Keras Documentation. date retrieved 20.07.2025.

[145] **Url-3** <*https://deeplearning4j.org* >, date retrieved 20.07.2025.

[146] **Url-3** <*https://www.edureka.co/blog/python-libraries/*>, Rao, Anirudh. date retrieved 10.07.2020.

[147] **Url-3** <*https://www.statsmodels.org/stable/index.html* >, date retrieved 20.07.2025.

[148] **Url-3** <*https://www.patika.dev/blog/populer-python-kutuphaneleri-mutlaka-bilmeniz-gerekenler* >, date retrieved 20.07.2025.

[149] **Aksoy, F., Özdem, M., & Daş, R.** (2025). A Perspective View on Data Visualization Libraries Used in Data Analytics. European Journal of Technique (EJT), 15(1), 81-96. https://doi.org/10.36222/ejt.1616824

[150] **Merve Ayyüce KIZRAK, Bülent BOLAT**. 2020. Derin Öğrenme ile Kalabalık Analizi Üzerine Detaylı Bir Araştırma, Bilişim Teknolojileri Dergisi, Cilt: 11, Sayı: 3, Temmuz 2018 https://dergipark.org.tr/tr/download/article-file/520262 Erişim Tarihi: 10.06.2020.

[151] **Url-3** <*https://aoyilmaz.medium.com/loss-functions-and-optimization-algorithms-in-deep-learning-fc9602a55b79*>, date retrieved 20.07.2025.

[152] **Url-3** <*https://alok05.medium.com/performance-metrics-or-loss-function-in-machine-learning-for-regression-7fae992577f0*>, date retrieved 20.07.2025.

[153] **Kaggle.** "Energy Consumption Dataset". Accessed: Apr. 7, 2025. [Online]. Available: https://www.kaggle.com/datasets/raminhuseyn/energy-consumption-dataset/data Erişim tarihi: 20.05.2025

[154] **Kaggle.** "Time Series Forecasting: Exploratory Data Analysis". Accessed: Apr. 7, 2025. [Online]. Available: https://www.kaggle.com/code/raminhuseyn/time-series-forecasting-exploratory-data-analysis Erişim tarihi: 20.05.2025

[155] **towars data science.** "Time Series Forecasting: A Practical Guide to Exploratory Data Analysis". Accessed: Apr. 7, 2025. [Online]. Available: https://towardsdatascience.com/time-series-forecasting-a-practical-guide-to-exploratory-data-analysis-a101dc5f85b1/ Erişim tarihi: 20.05.2025

[153] **Url-3** <*https://github.com/yusuficolab/MDPI-Makalesi/blob/main/50%20epok%20LSTM%20relu%20adammakale.ipyn*>, date retrieved 20.07.2025.

ÖZGEÇMİŞ

Ad-Soyad : Mehmet Tahir UÇAR

ÖĞRENİM DURUMU:

- **Lisans :** 2002, Dicle Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü.

  :2011, Dicle Üniversitesi, Mühendislik Mimarlık Fakültesi, İnşaat Mühendisliği
- **Yüksek Lisans:** 2018, Erciyes Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik Elektronik Mühendisliği Anabilim Dalı, Kayseri.
- **Doktora :** 2025, İnönü Üniversitesi, Elektrik Elektronik Mühendisliği Anabilim Dalı, Doktora Programı.

MESLEKİ DENEYİM:

- 2002-2003 Enba İnşaat Ltd Şti. Elektrik Elektronik mühendisi. Orta Gerilim ve Alçak Gerilim Enerji Nakil Hatları Tesislerinde Proje üzerinde çalışma, Şebeke tesisi, Kök binasına ek hücre tesisi, ENH (34,5 kV) Şantiyesinde Şantiye Şefliği,
- 2003-2004 Hava Kuvvetleri Komutanlığı (Meslek kurası ile gidildi). Mühendis Asteğmen
- 2004-2005 Karel İnşaat Ltd Şti. Elektrik Elektronik mühendisi. Yenibosna Bedaş'ın sorumluluğu altında olan ve müteahhit firmanın üstlendiği Orta Gerilim ve Alçak Gerilim Bakım Onarım işinde Şantiye Şefliği,
- 2005-2010 Eltemtek Ltd. Şti. Elektrik Elektronik mühendisi. Kralkızı Hes'te vardiya sorumlu mühendisliği, her türlü arızanın acilen giderilmesi ve bakım onarım çalışmalarının yürütülmesi,
- 2010-2025 Dicle Üniversitesi Ergani MYO. Öğretim Görevlisi.

YÜKSEK LİSANS VEYA DOKTORA TEZİNDEN TÜRETİLEN ÇALIŞMALAR (Makaleler, Bildiriler, Patentler v.b.)

- **Uçar MT., Kaygusuz A. (2025).** Short-Term Energy Consumption Forecasting Analysis Using Different Optimization and Activation Functions with Deep Learning Models. Applied Sciences, 15(12), 6839. https://doi.org/10.3390/app15126839 (Makale Örneği)
- **Uçar MT., Kaygusuz A. (2025).** 4th Internatıonal Thales Congress On Lıfe, Engıneerıng, Archıtecture And Mathematıcs. "Energy Consumptıon Estımatıon Usıng Multıple Regressıon Method", July 20-22, 2025 in Cairo, Egypt.(Bildiri Özeti)

- **Uçar MT., Kaygusuz A. (2023).** Cumhuriyetin 100. Yılında Cumhuriyet'ten Günümüze Ergani, Prof. Dr. Oktay BOZAN, Editör, Ekin Yayınevi, Bursa, s.385-413, 2023. (Kitap Bölümü)
- **Uçar MT., Kaygusuz A. (2023).** Makine Öğrenmesi İle Diyarbakır Ve Ergani İçin Gelecek Yıllar Nüfus Ve Enerji Tüketim Tahmini. Geçmişten Günümüze Uluslararası Ergani Sempozyumu.24-26 Kasım 2022 . (Bildiri Özeti)
- **Uçar, M. T., Üstkoyuncu, N. (2018).** "Rüzgar Enerjisi ve Türkiye'de Mardin İlinde Uygulanabilirliği", 3. Uluslararası Kültür ve Medeniyet Kongresi, Mardin, Artuklu Üniversitesi, (Nisan 2018). (Bildiri Özeti)
- **Uçar, M. T., Üstkoyuncu, N. (2018).** "Türkiye'de Mardin İlinde Güneş Enerjisi Potansiyeli", 3. Uluslararası Kültür ve Medeniyet Kongresi, Mardin, Artuklu Üniversitesi, (Nisan 2018). (Bildiri Özeti)
- **Uçar, M. T., Üstkoyuncu, N. (2016).** "Rüzgar enerjisi ve Türkiye'de Diyarbakır ilinde uygulanabilirliği". Uluslararası Diyarbakır Sempozyumu, Diyarbakır, (Kasım 2016) (Bildiri)
- **Uçar, M. T., Üstkoyuncu, N. (2016).** "Solar energy and evaluation of its applicabilty in the southeastern anatolia region", pp. 129-134. 6th International 100% Renewable Energy Conference, İstanbul, IRENEC, (Eylül 2016). (Bildiri)
- **Uçar, M. T., Üstkoyuncu, N. (2016).** "Türkiye'de Doğu Anadolu Bölgesinde güneş enerjisi potansiyeli", cilt 2 s. 282-291. UNİDAP Uluslararası Bölgesel Kalkınma Konferansı "Sosyal Kalkınma", Muş, Muş Alparslan Üniversitesi, (Mayıs 2016). (Bildiri)