T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

TOPLU YEMEK HİZMETLERİNDE MAKİNE ÖĞRENMESİ ALGORİTMALARI İLE TALEP PLANLAMA

Osman Ömer GÜNER

DANIŞMAN
Doç. Dr. Tarık KÜÇÜKDENİZ

Endüstri Mühendisliği Anabilim Dalı

Endüstri Mühendisliği Programı

İSTANBUL-2021

Bu çalışma, 31.08.2021 tarihinde aşağıdaki jüri tarafından Endüstri Mühendisliği Anabilim Dalı, Endüstri Mühendisliği Programında Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Jürisi

Doç. Dr. Tarık KÜÇÜKDENİZ(Danışman)
İstanbul Üniversitesi-Cerrahpaşa
Mühendislik Fakültesi

Doç. Dr. Yusuf Sait TÜRKANI
İstanbul Üniversitesi-Cerrahpaşa
Mühendislik Fakültesi

Dr. Öğr. Üyesi Zeynep GERGİN
İstanbul Kültür Üniversitesi
Mühendislik Fakültesi

Unvan Adı SOYADI
Üniversite
Fakülte

Unvan Adı SOYADI
Üniversite
Fakülte

20.04.2016 tarihli Resmi Gazete' de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, İstanbul Üniversitesi-Cerrahpaşa'nın abonesi olduğu intihal yazılım programı kullanılarak Lisansüstü Eğitim Enstitüsü'nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır..

ÖNSÖZ

Tez çalışmam boyunca her türlü konuda bana yardımcı olan ve desteğini esirgemeyen değerli hocam Doç. Dr. Tarık KÜÇÜKDENİZ' e teşekkürlerimi sunarım.

Yüksek lisans eğitim hayatım boyunca desteklerini esirgemeyen tüm hocalarıma teşekkürlerimi sunarım.

Gerek çalışma hayatım gerekse de eğitim hayatım boyunca beni destekleyen ve yanımda olan, değerli aileme, her zaman borçlu olarak kalacağım anneme ve babama şükranlarımı sunarım.

Ağustos 2021 Osman Ömer GÜNER

İÇİNDEKİLER

Sayfa No

ÖNSÖZ ..... **iv**
İÇİNDEKİLER ..... **v**
ŞEKİL LİSTESİ ..... **vii**
TABLO LİSTESİ ..... **ix**
KISALTMA LİSTESİ ..... **xi**
ÖZET ..... **xiii**
SUMMARY ..... **xv**
1. GİRİŞ **1**
2. GENEL KISIMLAR **6**
3. MALZEME VE YÖNTEM **11**
3.1. YAPAY ZEKA 12
3.1.1. Makine Öğrenmesi 14
3.1.1.1. Karar Ağacı Algoritması *16*
3.1.1.2. Topluluk Öğrenmesi (Ensemble Learning) *22*
3.1.1.3. Varyans ve Yanlılık (Variance and Bias) *25*
3.1.1.4. Yapay Sinir Ağları *27*
3.1.1.5. Tekrarlayan Sinir Ağları (Recurrent Neural Network-RNN) *32*
3.1.1.6. Uzun Kısa Süreli Bellek (Long Short Term Memory-LSTM) ve Kapılı Tekrarlayan Sinir Hücresi (Gated Recurrent Unit-GRU) *34*
3.2. ZAMAN SERİSİ ANALİZİ 38
3.2.1. ARIMA (p, d, q) 41
3.2.1.1. Augmented Dickey Fuller testi (Durağanlık) *43*
3.3. PERFORMANS GÖSTERGELERİ 44
3.4. UYGULAMA 45
3.4.1. Rassal Orman Algoritması Yöntemi ile Tahminleme 47
3.4.1.1. Öznitelik önem dereceleri (Feature importance) *50*
3.4.2. ARIMA ile Tahminleme 60
3.4.3. Yapay Sinir Ağları Modelleri ile Tahminleme 71
4. BULGULAR **75**
5. TARTIŞMA VE SONUÇ **80**

KAYNAKLAR **86**
EKLER **91**
ÖZGEÇMİŞ **106**

ŞEKİL LİSTESİ

Sayfa No

Şekil 1-1: Toplu Yemek Hizmeti Sunan Firmaların Genel Süreç Haritası (T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020).....2

Şekil 3-1: Yapay Zekâ ve alt dalları. 13

Şekil 3-2: Karar ağacı yapısı. 16

Şekil 3-3: Topluluk öğrenmesi yönteminin genel bir yapısı. 23

Şekil 3-4: Biyolojik sinir hücresi (nöron) yapısı. 27

Şekil 3-5: Yapay sinir hücresi (unit) matematiksel modeli. 28

Şekil 3-6: Yapay sinir ağı yapısı. 30

Şekil 3-7: RNN yöntemi sinir hücresi yapısı. 32

Şekil 3-8: RNN zaman göre açılımı (Kızrak, 2020). 33

Şekil 3-9: LSTM yöntemi hücre içi hesaplama süreci (Olah, 2015). 35

Şekil 3-10: GRU yöntemi hücre içi hesaplama süreci (Phi, 2018). 37

Şekil 3-11: Durağan (stationary) bir zaman serisi grafiği. 40

Şekil 3-12: Durağan olmayan (non-stationary) bir zaman serisi grafiği. 40

Şekil 3-13: HavaSicaklik-YemekTuketim ilişki grafiği. 46

Şekil 3-14: BirOncekiGun-YemekTuketim ilişki grafiği. 47

Şekil 3-15: GecenHaftaAyniGun-YemekTuketim ilişki grafiği. 47

Şekil 3-16: Veri seti 1 öznitelik önem dereceleri ilk 20. 51

Şekil 3-17: Veri seti 1 öznitelik önem dereceleri son 20. 52

Şekil 3-18: Veri seti 1 çorbalar için önem dereceleri. 53

Şekil 3-19: Veri seti 1 ana yemek türüne göre önem dereceleri. 53

Şekil 3-20: Veri seti 1 ana yemek ismine göre önem dereceleri. 54

Şekil 3-21: Veri seti 1 Yemek 3 önem dereceleri. 55

Şekil 3-22: Veri seti 1 Yemek 4 önem dereceleri. 55
Şekil 3-23: Veri seti 10 öznitelik önem dereceleri ilk 20. 56
Şekil 3-24: Veri seti 10 çorbalar için önem dereceleri. 57
Şekil 3-25: Veri seti 10 ana yemek türüne göre önem dereceleri. 58
Şekil 3-26: Veri seti 10 ana yemek ismine göre önem dereceleri. 58
Şekil 3-27: Veri seti 10 Yemek 3 önem dereceleri. 59
Şekil 3-28: Veri seti 10 Yemek 4 önem dereceleri. 59
Şekil 3-29: ARIMA ADF testi öncesi veri seti 1 görünümü (d=0). 61
Şekil 3-30: ARIMA ADF testi öncesi veri seti 2 görünümü (d=0). 61
Şekil 3-31: ARIMA ADF testi öncesi veri seti 3 görünümü (d=0). 62
Şekil 3-32: ARIMA ADF testi öncesi veri seti 4 görünümü (d=0). 62
Şekil 3-33: ARIMA ADF testi öncesi veri seti 5 görünümü (d=0). 63
Şekil 3-34: ARIMA ADF testi öncesi veri seti 6 görünümü (d=0). 63
Şekil 3-35: ARIMA ADF testi öncesi veri seti 7 görünümü (d=0). 64
Şekil 3-36: ARIMA ADF testi öncesi veri seti 8 görünümü (d=0). 64
Şekil 3-37: ARIMA ADF testi öncesi veri seti 9 görünümü (d=0). 65
Şekil 3-38: ARIMA veri seti 1 durağanlık görünümü (d=1). 67
Şekil 3-39: ARIMA veri seti 3 durağanlık görünümü (d=1). 67
Şekil 3-40: ARIMA veri seti 8 durağanlık görünümü (d=1). 68
Şekil 3-41: ARIMA veri seti 9 durağanlık görünümü (d=1). 68
Şekil 3-42: ARIMA veri seti 7 durağanlık görünümü (d=2). 69
Şekil 3-43: ARIMA veri seti 4 durağanlık görünümü (d=4). 69
Şekil 3-44: Yapay sinir ağları uygulamasında kullanılan ağ yapısı. 72

TABLO LİSTESİ

Sayfa No

Tablo 2.1: Literatürde kullanılan yöntemler, performans göstergeleri ve MAPE değerleri. 10

Tablo 2.2: Literatürde kullanılan bağımsız değişkenler. 10

Tablo 3.1: Veri seti değişkenleri (features). 12

Tablo 3.2: Biyolojik sinir hücresi ve yapay sinir ağı hücresi terimleri. 28

Tablo 3.3: ROA Veri setleri tablosu. 48

Tablo 3.4: ROA Veri setleri sonuçları. 50

Tablo 3.5: ARIMA veri setleri tablosu. 60

Tablo 3.6: ARIMA ADF test sonuçları (d=0). 65

Tablo 3.7: ARIMA ADF test sonuçları (d=1). 66

Tablo 3.8: ARIMA ADF test sonuçları (d=2). 66

Tablo 3.9: ARIMA ADF test sonuçları (d=4). 66

Tablo 3.10: ARIMA veri setleri parametre optimizasyon aralıkları. 70

Tablo 3.11: ARIMA Veri setleri sonuçları. 71

Tablo 3.12: YSA Veri setleri tablosu. 73

Tablo 3.13: RNN Veri setleri sonuçları. 73

Tablo 3.14: LSTM Veri setleri sonuçları. 74

Tablo 3.15: GRU Veri setleri sonuçları. 74

Tablo 4.1: Veri seti 1 modellerin kıyaslaması. 75

Tablo 4.2: Veri seti 2 modellerin kıyaslaması. 76

Tablo 4.3: Veri seti 3 modellerin kıyaslaması. 76

Tablo 4.4: Veri seti 4 modellerin kıyaslaması. 76

Tablo 4.5: Veri seti 5 modellerin kıyaslaması. 76

Tablo 4.6: Veri seti 6 modellerin kıyaslaması. 77
Tablo 4.7: Veri seti 7 modellerin kıyaslaması. 77
Tablo 4.8: Veri seti 8 modellerin kıyaslaması. 77
Tablo 4.9: Veri seti 9 modellerin kıyaslaması. 77
Tablo 4.10: ROA Veri setleri kıyaslaması.....78
Tablo 4.11: Veri setlerine göre en iyi modeller ve performans göstergeleri. 79
Tablo 5.1: ROA ve YSA yöntemleri MAPE istatistikleri kıyaslaması. 84
Tablo 5.2: Literatür ve Mevcut çalışma MAPE istatistikleri kıyaslaması.....84

KISALTMA LİSTESİ

Kısaltmalar **Açıklama**

ABD : Amerika Birleşik Devletleri
AIC : Akaike Information Criteria
ARIMA : Autoregressive Integrated Moving Average
BS : Benchmark Scenario
BUT : Bütünleme sınavı
BÜD : Basit Üstel Düzleştirme
CART : Classification and Regression Trees
CHAID : Chi-squared Automatic Interaction Detector
ÇR : Çoklu Regresyon
DVM : Destek Vektör Makinesi
DVR : Destek Vektör Regresyon
FPF : Facebook Prophet Forecasting
GBDT : Gradient-Boosted Decision Trees
GRU : Gated Recurrent Unit
GSR : Gauss Süreç Regresyonu
HO : Hareketli Ortalama
HY : Holt Yöntemi
İÜD : İkili Üstel Düzleştirme
LR : Lineer Regresyon
LSTM : Long-Short Term Memory
MA : Moving Average
MAE : Mean Absolute Error
MAPE : Mean Absolute Percentage Error
MDT : Microsoft Decision Trees
MPE : Mean Percentage Error
MSE : Mean Squared Error
NT : Naive Tahmin
PFM : Passenger Flow Model
RAE : Relative Absolute Error
RFR : Random Forest Regression

RMSE : Root Mean Squared Error

RNN : Recurrent Neural Network

ROA : Rassal Orman Algoritması

ROR : Rassal Orman Regresyon

RRSE : Root Relative Squared Error

RT : Regression Trees

SES : Single Exponential Smoothing

WY : Winter Yöntemi

YSA : Yapay Sinir Ağları

ÖZET

YÜKSEK LİSANS TEZİ

TOPLU YEMEK HİZMETLERİNDE MAKİNE ÖĞRENMESİ ALGORİTMALARI İLE TALEP PLANLAMA

Osman Ömer GÜNER

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Endüstri Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Tarık KÜÇÜKDENİZ

Sanayileşme ile çalışanların; üniversitelerin büyük yerleşkelerde faaliyetlerini yürütmesi ile öğrencilerin ve personellerin; devlet kurumlarının genişlemesi ile memurların yemek tüketim ihtiyaçları farklı organizasyonların gelişmesine sebep olmaktadır. Bu organizasyonlar "*catering*" ismi ile adlandırılmaktadır. Catering firmaları müşterilerinin yemek ihtiyaçlarını karşılamak için yemek üretim merkezleri olarak faaliyetlerini sürdürmektedir. Bu faaliyetlerini sürdürürken planlama, satın alma, üretim, dağıtım, lojistik, servis ve kalite kontrol gibi süreçler gerçekleştirilmektedir. Söz konusu süreçler menülerin planlaması ile başlamaktadır. Menü planlama aşamasında yemek çeşitlerinin, porsiyonların ve tüketecek kişi sayısının belirlenmesi oldukça önemli bir aşama olmaktadır. Bu aşamada tüketim sayısının belirlenmesi ile birlikte satın alma gerçekleştirilmektedir. Satın alma gerçekleştirilmeden önce yemek tüketim sayısının belirlenmesi (tahmini) aşaması bu çerçevede oldukça önemli görülmektedir. Uygun ve gerçekçi olmayan yemek tüketim sayısı tahmini; fazla satın alma, fazla stok, işgücü kaybı, enerji kaybı gibi israf kalemlerine sebep olabilmektedir.

Yapılan çalışma ile günlük yemek tüketiminin tahmin edilmesi için makine öğrenme algoritmaları incelenerek, toplu yemek hizmeti sunan tüm catering işletmelerinde kullanılabilecek bir model önerilmiştir. Teknolojik düzeydeki artış ile geliştirilen algoritmalar kullanılarak catering sektörünün kullanımı için bir yaklaşım sunulmuştur. Bu yaklaşımı gerçekleştirmek amacıyla, bir üniversite yemekhanesine ait 233 günlük yemek tüketim sayıları kullanılmış ve geleceğe yönelik yemek tüketim sayısı tahmini gerçekleştirilmiştir. Özel günlere

ve normal günlere ait daha kararlı sonuçlar elde etmek için genel veri seti içerisinden 18 adet farklı veri setleri oluşturulmuştur. Bu veri setleri ile; Rassal Orman Algoritması (ROA), Yapay Sinir Ağları (YSA) içerisinde yer alan Tekrarlayan Sinir Ağları (RNN), Uzun Kısa Süreli Bellek (LSTM) ve Kapılı Tekrarlayan Sinir Hücresi (GRU) yöntemleri, zaman serisi yöntemlerinden olan ARIMA yöntemi kullanılarak eğitim değerleri ile modeller geliştirilmiştir ve test değerleri ile kıyaslamalar yapılmıştır. Modellerin performanslarını ölçmek için MSE, RMSE, MAE, MAPE, R^2 göstergeleri kullanılmıştır. Geliştirilen modeller ile elde edilen performans değerleri, literatürdeki çalışmalara göre daha üstün sonuçlar göstermiştir. Bu çalışma ile catering sektöründe yemek tüketim tahminlerinin yapılması için geniş bir yaklaşım sunulmuştur. Sektörde faaliyet gösteren işletmeler, bu yaklaşımı kolaylıkla faaliyetlerine entegre edebilirler.

Ağustos 2021, 123 sayfa.

Anahtar kelimeler: Rassal orman algoritması, tekrarlayan sinir ağları, uzun kısa süreli bellek, kapılı tekrarlayan sinir hücresi, ARIMA.

SUMMARY

M.Sc. THESIS

DEMAND PLANNING FOR CATERING SERVICES BY USING MACHINE LEARNING ALGORITHMS

Osman Ömer GÜNER

Istanbul University-Cerrahpasa

Institute of Graduate Studies

Department of Industrial Engineering

Supervisor: Assoc. Prof. Dr. Tarık KÜÇÜKDENİZ

The food consumption needs of civil servants with the growth of state institutions, of students with the universities carrying out their activities in large campuses, of workers with industrialization cause the development of different organizations. These organizations are called “catering”. Catering companies continue their activities as food production centers to supply the food needs of their customers. While continuing these activities, processes such as planning, purchasing, production, distribution, logistics, service and quality control are carried out. The processes in question start with the planning of the menus. In the menu planning stage, it is very important to determine the types of food, portions and the number of people who will consume the food. At this stage, the purchase is made after determining the number of food consumption. Before the purchase, the stage of determining (estimating) the number of food consumption is considered very important in this context. The estimation of the inappropriate and unrealistic number of food consumption may cause waste items such as over-purchasing, excess stock, loss of labor, and energy loss.

In the study, machine learning algorithms were examined to predict daily food consumption and a model that could be used in all catering operations offering mass food service was proposed. An approach for use in the catering industry is presented by using algorithms developed with the increase in technological level. In order to realize this approach, the number of food consumption for the future was estimated by using 233 daily food consumption numbers

of a university dining hall. In order to obtain more stable results for special days and normal days, 18 different data sets were created from the general data set. With these data sets; By using Random Forest Algorithm (ROA), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) and ARIMA methods, models with training values were developed and comparisons were made with test values. MSE, RMSE, MAE, MAPE, R^2 indicators were used to measure the performances of the models. The performance values obtained with the developed models showed superior results compared to the studies in the literature. In this study, a broad approach is presented to make food consumption forecasts in the catering industry. Catering businesses operating in the industry can easily integrate this approach into their processes.

August 2021, 123. pages.

Keywords: Random forest algorithm, recurrent neural network, long short-term memory, gated recurrent unit, ARIMA.

1. GİRİŞ

"Catering" sözcüğü Türkçe diline İngilizceden gelmiş olup Türk Dil Kurumu sözlüğünde "yemek hizmeti[1]" anlamında yer almaktadır. "Catering" sözcüğü diğer bir ifade ile "yiyecek tedarik eden" anlamında da kullanılabilmektedir. Günümüzde toplu yemek üretimi ve toplu yemek hizmeti sunan firmalara verilen isim olarak kullanılmaktadır. İşyerlerinde, fabrikalarda, okullarda, üniversitelerde, hastanelerde, sayıca fazla personel bulunduran benzer kurumlarda toplu yemek hizmetleri görülebilmektedir.

Şehirleşme oranındaki ve çalışan kesim oranındaki artış, sanayi alanındaki gelişme, toplumların ekonomik ve kültürel yapılarında değişme insanların ev dışında beslenmelerini zorunlu hale getirmiştir. Bu durum dünyada olduğu gibi Türkiye' de de ev dışında, başkaları tarafından hazırlanan ve sunulan toplu yemek hizmetlerinin öneminin giderek artmasına yol açmıştır. Bu doğrultuda, toplu yemek hizmeti veren firmaların sayısı da artmıştır. Toplu yemek hizmeti veren firmaların temel amacı kaliteli bir beslenme servisi sunmaktır. Temel amaca ulaşabilmek için toplu yemek hizmeti firmalarının tüm süreçlerinin menü planlama çerçevesinde yürütülmesi sağlanmaktadır (T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020).

Toplu yemek hizmeti sunan firmalarda yer alan süreçler birçok işlemi kapsayan bir süreç topluluğudur. Hizmetin her bir aşaması ayrı bir önem taşımaktadır. Süreç, menü planlama ile başlamaktadır. Menü planlama sürecinde hedef kitleye özgü menülerin planlanması toplu yemek hizmet süreçlerinin merkezinde yer alır. Menü planlama süreci sonrasında gerekli besinlerin satın alınması, teslim alınması ve depolanması hizmet kalitesi açısından takip gerektiren en önemli süreçlerdendir. Üretim sürecinde, besinlerin ön hazırlık işlemleri, hazırlık, pişirme, soğutma ve yeniden ısıtma süreçleri yer almaktadır. Üretim sürecini, sevkiyat amaçlı porsiyonlara bölme ve servis süreçleri izlemektedir. Servis sonrasında atıkların kaldırılması, bulaşıkların yıkanması da toplu yemek hizmetlerinde yürütülmesi gereken süreçlerdir (T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020).

[1] https://sozluk.gov.tr/



Şekil 1-1: Toplu Yemek Hizmeti Sunan Firmaların Genel Süreç Haritası (T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020).

Menü planlama, toplu yemek hizmeti merkezinde yer alır ve uzmanlık gerektiren bir süreçtir. İyi planlanmayan menüler, hizmeti satın alan kişilerin kaliteli yemek gereksinimlerini karşılamayacağı gibi, atık ve kayıpların artmasına, tüketici memnuniyetinin azalmasına, çalışanlarda gereksiz iş gücü ve motivasyon kaybına, firmanın ekonomik ve prestij kaybına uğramasına yol açar. Toplu yemek yönetimi ve çalışanlarının başarısı planlanan menülerle yakından ilgilidir. Menü planlamada maliyetleri dengede tutmak, atıkları ve kayıpları mümkün olduğunca en aza indirgemek önem arz etmektedir. Menü planlamada amaç, hitap edilen kitleye yani hizmeti satın alan kişilere; ortama, kültüre, zamana ve mevsime uygun menüyü (çorba, ana yemek, tatlı, içecek, vs.) oluşturmak ve yine aynı faktörlere bağlı olarak tüketim tahmini yapabilmektir (T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020).

Menü planlama aşamasında tüketim tahmini yapılarak fazla veya eksik satın almanın önüne geçilmektedir. Fazla satın alma fazla maliyete sebep olurken, eksik satın alma zaman kaybı ve verimsizliğe sebep olmaktadır. Tüketim tahmini yapılarak fazla stokun önüne geçilmekte ve besinlerin her zaman taze bir şekilde tüketilmesi sağlanmaktadır. Tüketim tahmini ile fazla personel çalıştırılmasının önüne geçilerek sermayenin ve işletme karının doğru yönde kullanılması sağlanmakta olup, eksik personel durumuna engel olarak gecikmelerin ve kalite kaybının önüne geçilmektedir.

Toplu yemek hizmeti sunan firmalarda satın alma, üretim, servis ve servis sonrası aşamalarında oluşabilen israflar da göz önüne alındığında tüketim tahmini oldukça önemli bir noktada bulunmaktadır. Satın alma aşamasında taşıma ve depolama maliyeti artışı; pişirme aşamasında yakılan elektrik, kullanılan su, kullanılan doğalgaz, çalıştırılan personel artışı; servis aşamasında fazladan personel bulundurma, servis sonrası aşamasında; temizlik, bulaşık, atık yönetimi gibi konuların hepsi tek başına israf kalemleri olarak değerlendirilebilmektedir. Bu noktada tüketim sayıları için talep tahmini teknikleri kullanımı öne çıkmaktadır. Tahminin anlamlı olması bakımından zaman serileri için içereceği zaman süreci çok önemlidir.

Talep tahmin tekniklerinde hangi yöntem kullanılırsa kullanılsın hiçbir zaman %100' lük bir doğruluk derecesi veremez. Bu yüzden buna en çok yaklaşan yöntem, ilgili problem için en iyi yöntemdir (Tanyaş & Baskak, 2008). Talep tahmini sezgisel yöntemlerle yapılabileceği gibi belli bir metot içeren niceliksel yöntemlerle de yapılabilir. Niceliksel yöntemler içerisinde kullanılan zaman serisi analizi bu çalışma kapsamında kullanılan yöntemlerden birisi olan ARIMA (p, d, q) yönteminin temelini oluşturmaktadır.

Gelişen teknoloji ve genişleyen veriler ile birlikte bu geçmiş verilerin kullanılması gerektiği fikri ortaya atılmıştır. Böylelikle makine öğrenmesi ve derin öğrenme metotları ortaya çıkmış, günümüzde oldukça ilerleme kaydetmiştir. Bu çalışma kapsamında tüketim tahmini yapılmasında makine öğrenmesi yöntemleri olarak ROA, RNN, LSTM ve GRU yöntemleri kullanılmıştır.

Bu tez çalışmasının amacı makine öğrenmesi yöntemleri ve zaman serisi analizi yöntemi ile bir üniversite yemekhanesine ait olan günlük yemek tüketim sayılarını kullanarak geleceği yönelik tahmin yapmak ve catering sektöründe faaliyet gösteren işletmeler için bir yaklaşım sunmaktır. Geliştirilen modellerin sonuçlarını kıyaslayarak en iyi yönteme karar vermektir. Catering

faaliyeti gösteren yemek üretim firmaları için günlük yemek tahmini yapan bir makine öğrenmesi modeli geliştirerek, düşük tahmin hatası ile planlamada maliyet avantajı sağlanması amaçlanmıştır. Catering sektörünün kullanımına yönelik belli bir çerçeve sunulmuştur. Yapılan çalışmanın, catering sektöründe faaliyet gösteren işletmelere hizmet yönünden kalite seviyelerini artırmaları ve maliyet yönünden avantajlı konuma geçmeleri gibi katkısının olacağı düşünülmektedir. Geliştirilen yöntemlerin sonuçlarını kıyaslamak için performans göstergesi olarak; *Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE),* R^2 *(Korelasyon katsayısı) ve Mean Absolute Percentage Error (MAPE)* kullanılmıştır.

Çalışma kapsamında bir üniversite yemekhanesine ait olan 03.01-20.12.2019 tarihleri arasındaki günlük yemek tüketim sayıları kullanılmıştır. Kullanılan bağımsız değişkenler aşağıda sunulmuştur.

- Haftanın günü
- Mevsim
- Ramazan günü mü?
- Final sınav günü mü?
- BÜT sınav günü mü?
- Sömestir tatili mi?
- Yaz tatili mi?
- Hava sıcaklığı
- Hava durumu
- Çorba (Yemek 1)
- Ana yemek (Yemek 2)
- Ana yemek türü (etliyse türü)
- Yan yemek (Yemek 3)
- Yan yemek (Yemek 4)
- Bir önceki gün yemek tüketim sayısı
- Geçen hafta aynı gün yemek tüketim sayısı

Eklenen bağımsız değişkenler makine öğrenmesi yöntemi olan ROA yöntemi kapsamında kullanılmıştır. Zaman serisi analizi yöntemi ve YSA yöntemlerinin algoritmaları gereğince sadece yemek tüketim sayıları kullanılmıştır.

Çalışmanın ikinci kısmında konu ile ilgili literatürde yapılan çalışmalar incelenmiştir. Üçüncü kısımda kullanılan yöntemler, yöntemlerin temeli olan teorik bilgiler verilmiştir, uygulamaya ait olan detaylı çalışma sunulmuştur. Dördüncü kısımda uygulama sonuçları detaylı olarak sunulmuştur. Beşinci kısımda yöntemlere ve sonuçlara ilişkin değerlendirmeler, geliştirilmesi gereken ve geliştirilebilecek yerler hakkında bilgiler sunulmuştur ve tartışılmıştır.

2. GENEL KISIMLAR

2.1. LİTERATÜR TARAMASI

Bu çalışmada literatür taraması için belirlenen anahtar kelimeler "*catering forecast(ing)*", "*catering predict(ing)*", "*foodservice foreceast(ing)*", "*foodservice predict(ing)*", "*dining hall forecast(ing)*" olmuştur. Taramalar yapılırken kelimeler bütün olarak kullanılmıştır. Anahtar kelimeleri kullanarak taratma işlemi için Google Akademik (scholar.google.com) kullanılmıştır. Ve taratma işlemi tüm yılları kapsayacak biçimde yapılmıştır.

Kullanılan anahtar kelimeler ile, tez konusu hakkında incelenen makale, yüksek lisans tezi ve konferans bildirisi gibi çalışmalar yer almıştır. Literatür taraması yapılacak çalışmalar 1982 ve 2021 yılları arası olmuştur. Bunlarla ilgili bilgiler detaylı olarak sunulmuştur.

Olsen vd. 1982 yılında yaptıkları çalışmada, bir bölgedeki AVM içerisinde yer alan bir restorana ait aylık yemek tüketim sayılarının olduğu 20 aylık veri setini kullanmışlardır. İlk 10 aylık veriyi kullanarak son 10 aylık değerleri tahminde bulunmuşlardır. Basit Üstel Düzleştirme (Simple Exponential Smoothing) ve İkili Üstel Düzleştirme (Double Exponential Smoothing) Yöntemleri kullanılmıştır. Tahmin ve gerçek değerlerini kıyaslamak için herhangi bir performans kriteri kullanılmamış, sadece grafikte görselleştirilmiştir (Olsen & Jose, 1982).

Miller vd. 1993 yılında yapılan çalışmalarında ABD' de bulunan iki restorana ait olan 288 günlük yemek tüketim sayılarını içeren veri setini kullanarak yemek tüketim tahmini yapmışlardır. 90 gün olan kısa dönemli ve 288 gün olan uzun dönemli olarak iki farklı veri seti kullanılarak tahmin yapılmıştır. Bunun için Naive (Forecast) Tahmin, Basit Üstel Düzleştirme ve Basit Hareketli Ortalama yöntemleri ve aynı yöntemlerin mevsimsellik içeren türleri de ayrı olarak kullanılmıştır. Basit Üstel Düzleştirme ve Basit Hareketli Ortalama yöntemleri N değerinin 3, 7 ve 14 değerleri için kullanılmıştır. Toplamda 10 farklı model çalıştırılmıştır. Kısa ve uzun dönem veri setleri ile toplamda 20 farklı tahmin sonucu elde edilmiştir. Sonuçları değerlendirmek için Ortalama Mutlak Hata (MAE), Ortalama Hata Karesi (MSE) performans göstergeleri kullanılmıştır. Restoranlardan biri için en iyi model Basit Üstel Düzleştirme, diğer restoran için Basit Hareketli Ortalama N değeri 3 iken bulunmuştur (Miller, et al., 1993).

Ryu ve Sanchez 2003 yılında yaptıkları çalışmalarında bir üniversite yemekhanesine ait olan 2000 yılı güz dönemi günlük yemek tüketim sayılarını kullanarak 2001 yılı bahar dönemi

yemek tüketim sayılarını tahmin etmişlerdir. 2001 bahar dönemi gerçek yemek tüketim değerleri ile tahmin değerlerini kıyaslamak için performans göstergesi olarak MAE, MSE, MPE, MAPE, RMSE ve Theil U istatistik değerleri kullanılmıştır. Naive Tahmin, Basit Hareketli Ortalama, Basit ve İkili Üstel Düzleştirme, Holt, Winter, Lineer ve Çoklu Regresyon yöntemleri yemek tüketim sayılarını tahmin etmek için kullanılmıştır. En iyi model %7,85 MAPE değeri ile Çoklu Regresyon yöntemi olmuştur (Ryu & Sanchez, 2003).

Ryu vd. 2003 yılında yaptıkları çalışmalarında bir önceki çalışmalarına benzer şekilde bir üniversiteye ait günlük yemek tüketim sayılarını kullanarak tahmin çalışması yapmışlardır. Bahar dönemine ait yemek tüketim sayıları kullanılarak bir sonraki yılın bahar dönemi için tahmin yapılmıştır. Naive Tahmin, MA, SES, Holt, Winter ve Doğrusal Regresyon yöntemleri kullanılarak MSE, MAPE ve MPE performans göstergeleri ile gerçek ve tahmin değerleri kıyaslanmıştır. Mevsimsellik içermeyecek şekilde kullanılan ham verilere göre Winter yöntemi %7,72 MAPE değeri ile en iyi model olarak ortaya çıkmıştır. Mevsimsel düzeltmeler yapıldıktan sonra kullanılan veri seti ile MA (6) yöntemi MAPE %6,48 değeri ile en iyi model olarak ortaya çıkmıştır (Ryu, et al., 2003).

Blecher ve Yeh 2008 yılında yapılan çalışmalarında bir üniversite yemekhanesinin 2004 yılına ait 15 haftalık günlük yemek tüketim sayılarını içeren veri setini kullanarak yemek tüketim sayısını tahmin etmişlerdir. Çalışmalarında Naive Tahmin, Hareketli Ortalama Modeli (3 farklı şekilde) ve Basit Üstel Düzleştirme modelleri kullanmışlardır. Yöntemlerin kıyaslamasını yapmak için kullanılan performans kriterleri MAE, MSE ve MAPE olmuştur. Sabah, öğle, akşam zamanlarında iki farklı salona ait veri setleri ile toplamda 5 yöntem ile yapılan denemeler arasında ortalama %5 MAPE değeri ile Hareketli Ortalama yöntemi üstün çıkmıştır (Blecher & Yeh, 2008). Çalışmalarında kullanılan veri setinde haftanın gününe yönelik tahminler yapıldığı görülmüştür. Örneğin pazartesi günü için yapılan tahmin hesaplamasında veri seti olarak geçmiş pazartesi günleri kullanılmıştır. Diğer günler içinde aynı yöntem izlenmiştir. Tatil öncesi veya tatil sonrası dönemlerin ilk günleri göz ardı edilmiştir.

Bozkır ve Sezer 2010 yılındaki Dünya Bilgi Teknolojileri Kongresindeki bildirileri ile yapılan yemekhane tüketim tahmini çalışmalarını sunmuşlardır. Bu bildiride Hacettepe Üniversitesi yemekhanesine ait olan ve öğrenci, akademisyen ve çalışanların yemek tüketim sayılarını ayrı ayrı içeren veri setini kullanmışlardır. CART, CHAID ve MDT yöntemleri ve performans

göstergesi olarak R^2 değeri kullanılmıştır. Ortalamada en iyi performans göstergesi değerini veren yöntem %69,5 ile MDT olmuştur (Bozkır & Sezer, 2010).

Kılıç (2015) yüksek lisans tezinde yaptığı çalışmasında YSA ve Çoklu regresyon yöntemlerini kullanarak üniversite yemekhanesi tüketim sayılarını tahmin etmiştir. Kullanılan veri setinde normalizasyon işlemi yapılmıştır. Performans göstergesi olarak RMSE kullanılmıştır. YSA yöntemi daha başarılı olmuştur (Kılıç, 2015).

Kılıç vd. (2018) yapılan çalışmalarında DVM, LR ve YSA yöntemlerini kullanarak bir üniversiteye ait olan 3 yerleşkenin ayrı veri setlerine dayanarak yemek tüketim sayılarını tahmin etmişlerdir. MAPE, RMSE, RAE ve RRSE performans göstergeleri kullanılmıştır. Çalışmada kullanılan YSA yönteminde gizli katmandaki nöron sayısının artmasının performansı arttırmadığı sonucuna varılmıştır (Kılıç, et al., 2018).

Dong vd. (2018) yaptıkları çalışmada bir hazır yemek (fast food) zincir mağazalarının sıcak içecek satışlarını haftalık ve aylık olarak tahmin etmişlerdir. Bunun için GBDT, LR, PF ve YSA yöntemlerini kullanmışlardır ve MAPE performans göstergesi ile de kıyaslamışlardır. GBDT yöntemi %11,31 değeri ile en iyi performansı veren yöntem olmuştur (Dong, et al., 2018).

Çetinkaya ve Erdal tarafından 2019 yılında yapılan çalışmada bir üniversite yemekhanesine ait öğrenci ve personel günlük yemek tüketim verileri kullanılarak günlük tahmin yapılmıştır. Bunun için YSA yöntemi farklı nöron sayılarında ve katmanlarda denenmiştir. Performans göstergesi olarak R^2, MAPE, MAE, RMSE ve MSE kullanılmıştır. En iyi MAPE değeri olarak öğrenci veri seti için %19,65, personel veri seti için %16,84 değerleri elde edilmiştir. Çalışmayı yürütenler veri setini hazırlama aşamasında kategorik veriler için sayısal değerler vererek, algoritmanın sayısal veri gibi düşünmesine sebep olmuşlardır. Burada ikili (one-hot encoding yöntemi) olarak temsil edilmesi sağlanabileceği düşünülebilir (Çetinkaya & Erdal, 2019).

Calp (2019) tarafından YSA yöntemi kullanılarak yapılan çalışmasında bir üniversite yemekhanesinin 2016-2018 yılları arasına ait günlük yemek tüketim verileri kullanılarak yemek tüketim tahmini yapılmıştır. YSA yönteminin farklı nöron ve katmanlarda varyasyonları denenmiştir. Önerilen en iyi modelin eğitim veri seti için elde edilen performans göstergelerinde R^2 %98,30, MSE 0,003 ve MAPE %0,7012 değerleriyle öne çıkmıştır. Elde edilen MAPE değeri ve diğer göstergeler modelin eğitimindeki değerlerdir. Calp, çalışmasında kategorik verileri sayısallaştırdıktan sonra veri setine normalizasyon uygulamıştır (Calp, 2019).

Çoban 2019 yılında yaptığı çalışmasında bir gıda işletmesinde 4 farklı ürüne ait olan 52 aylık bir veri setinde üretim miktarı, birim üretim maliyeti gibi 8 adet bağımsız değişken ile aylık satış tahmini yapmıştır. Bunun için YSA ve DVR yöntemlerini kullanmıştır. Yöntemlerin her ikisi için deneysel tasarım kullanarak parametre optimizasyonu yapmıştır. En uygun parametreler ile yöntemleri çalıştırmış ve YSA yöntemi DVR' ye göre daha iyi performans göstermiştir. Kullanılan performans göstergeleri MSE, MAE ve MAPE olmuştur. En iyi performans göstergesini sağlayan ürüne ait MAPE değeri %3,8 olmuştur (Çoban, 2019).

Yergök ve Acı 2019 yılında yapılan çalışmalarında günlük yemek tüketim tahmini çalışması yapmışlardır. Bunun için bir üniversite yemekhanesine ait olan 8 adet bağımsız değişken ve 1 adet bağımlı değişken olmak üzere toplamda 9 adet değişkeni içeren veri setini kullanmışlardır. Veri setinde yer alan kategorik değişkenler sayılarla temsil edilip öylece modele verilmiştir. Herhangi bir normalizasyon işlemi yapılmamıştır. Çalışmalarında kullanılan yöntemler; rasyonel kuadratik, kare üstel, matern 5/2 ve üstel olmak üzere 4 farklı GSR modelleri ile fine tree, medium tree ve coarse tree olmak üzere 3 farklı regresyon ağaçları modelleri olmuştur. Kullanılan performans göstergeleri olarak MSE, RMSE, MAE ve korelasyon katsayısı (R) değerleri olmuştur. Performans göstergelerine göre üstel GSR ve fine tree regresyon ağacı yöntemi karşılaştırmalarda üstün gelmiştir (Yergök & Acı, 2019).

Malefors vd. (2021) halka açık yemek tesislerinde üretilen yemeklerdeki fazla üretim probleminden yola çıkarak israf yapıldığı vurgulanmıştır. Bunun önüne geçmek için 2010 ve 2019 yılları içinde 21 okula ait veri setini kullanarak okulu ziyaret edenlerin yemek tüketim sayılarını tahmin etmişlerdir ve bunun için Benchmark Scenario, Naive Tahmin (Last-Value Forecasting), MA, Prophet Forecasting Model ve YSA yöntemleri kullanılmıştır. Yöntemleri kıyaslamak için MAPE performans kriteri kullanılmıştır. Kullanılan yöntemler arasında YSA yöntemi 11 okulda diğer yöntemlere göre daha düşük bir MAPE değeri sunmuştur. 7 okulda MA yöntemi (2 lag) daha üstün gelmiştir. Toplam 20 okulda yöntemlerin sunduğu MAPE değeri ortalama %5,20 çıkmıştır (Malefors, et al., 2021).

Tablo 2.1: Literatürde kullanılan yöntemler, performans göstergeleri ve MAPE değerleri.

Yıl	Yazarlar	Yöntemler	Performans Göstergeleri	MAPE Değeri	Tahmin Edilen
1982	Michael D. Olsen, Manuel L. Jose	BÜD, İÜD	Grafik gösterim		Aylık yemek tüketim sayısı
1993	James J. Miller, Cynthia S. McCahon, Judy L. Miller	NT, HO, BÜD	MAE, MSE		Günlük yemek tüketim sayısı
2003	Kisang Ryu, Alfonso Sanchez	NT, HO, İkili HO, BÜD, İÜD, HY, WY, LR, ÇR	MAE, MSE, MPE, MAPE, RMSE	7,85%	Günlük yemek tüketim sayısı
2003	Kisang Ryu PhD, SooCheong Shawn Jang PhD, Alfonso Sanchez PhD	NT, HO, BÜD, HY, WY, LR	MSE, MAPE, MPE	6,48%	Günlük yemek tüketim sayısı
2008	Lee Blecher; Ronnie J. Yeh	NT, HO, BÜD	MAE, MSE, MAPE	5%	Günlük yemek tüketim sayısı
2010	Ahmet Selman Bozkir, Ebru Akcapinar Sezer	CART, CHAID, MDT	R kare		Günlük yemek tüketim sayısı
2015	Günay KILIÇ	YSA, ÇR	RMSE		Günlük yemek tüketim sayısı
2018	Fatih Kılıç, Murat Reis Akkaya, Nuran Memili	YSA, DVM, LR	MAPE, RMSE, RAE, RRSE	35,56%	Günlük yemek tüketim sayısı
2018	Jianming Dong, Ziyi Li, Xueci Chen, Fan Yang, Tianyu Gao, Jie Jin	GBDT, LR, PFM, YSA	MAPE	11,31%	KFC mağazaları sıcak içecek günlük satış tahmini
2019	Zeynep Çetinkaya, Erdal Erdal	YSA	R kare, MAPE, MAE, RMSE, MSE	19,65%	Günlük yemek tüketim sayısı
2019	M. Hanefi Calp	YSA	R kare, MSE, MAPE	0,7% (Eğitim veri seti için)	Günlük yemek tüketim sayısı
2019	Feyza Çoban	YSA, DVR	MSE, MAE, MAPE	3,80%	Aylık yemek tüketim sayısı
2019	Derya Yergök, Mehmet Acı	GSR, RT	MSE, RMSE, MAE, R (Korelasyon katsayısı)		Günlük yemek tüketim sayısı
2021	Christopher Malefors, Ingrid Strid, Per-Anders Hansson, Mattias Eriksson	BS, NT, HO, FPF, YSA	MAPE	Ortalama 5,2 %	Günlük yemek tüketim sayısı

Literatür araştırmasında incelenen çalışmalar Tablo 2.1' de görülmektedir. Tabloda görüleceği üzere kullanılan yöntemler, kullanılan performans göstergeleri ve elde edilen MAPE değerleri sunulmuştur. Makine öğrenmesi yöntemlerinde sadece YSA yöntemi kullanılmıştır. Zaman serisi olarak bakıldığında yemek tüketim tahmini çalışmalarında çeşitli yöntemler kullanılmakla beraber, ARIMA yönteminin kullanılmadığı görülmektedir.

Tablo 2.2: Literatürde kullanılan bağımsız değişkenler.

Bağımsız değişkenler	**Kullanım sayısı**
Tatil günleri	6
Yemek 1	6
Yemek 2	6
Yemek 3	6
Yemek 4	6
Yemek tüketim sayıları	6
Haftanın Günü	5
Ramazan ayı	4
Sınav Haftası	4
Personel Sayısı	4
Öğrenci Sayısı	3
Maaş Günü	2
Mevsim	2
Dönem	1
Ay	1

Yemek tüketim tahmini üzerine olan literatürde yapılan çalışmalarda kullanılan bağımsız değişkenler Tablo 2.2' de görülmektedir. İlgili tabloda "*Yemek 1*" çorba, "*Yemek 2*" ana yemek, "*Yemek 3*" yan yemek, "*Yemek 4*" salata- meyve- tatlı yiyeceklerini; yani günlük menü içeriklerini temsil etmektedir. 6 çalışmada *tatil günleri, menü içerikleri, yemek tüketim sayıları*; 5 çalışmada *haftanın günü*; 4 çalışmada *sınav haftası, ramazan ayı, personel sayısı* (günlük çalışan); 3 çalışmada *öğrenci sayısı* (günlük bulunan); 2 çalışmada *maaş günü, mevsim*; 1 çalışmada *dönem, ay* bilgileri bağımsız değişken olarak kullanılmıştır. Tüm çalışmalarda yemek tüketim sayıları tahmin edilmiştir, bağımlı değişken olarak kullanılmıştır. Bu çalışma kapsamında bağımsız değişkenlere karar verilirken literatürdeki bilgilerden yararlanılmıştır.

3. MALZEME VE YÖNTEM

Bu çalışma kapsamında; Machine Learning (Makine Öğrenmesi) yöntemlerinden ROA, zaman serisi tahmin yöntemlerinden olan ARIMA, Yapay Sinir Ağları yöntemlerinden olan RNN, LSTM ve GRU yöntemleri kullanılmıştır. Yöntemler birbirleri ile kıyaslanmıştır. Kıyaslamak için MAPE, MAE, MSE, RMSE ve Korelasyon katsayısı (R^2) performans göstergeleri kullanılmıştır.

Çalışma kapsamında bir üniversitenin 2019 yılına ait günlük yemekhane tüketim sayılarını içeren veri seti kullanılmıştır. Çalışma kapsamında kullanılan yöntemler dahilinde veri setinin uzunluğuna bağlı olarak test verisi oranı değişkenlik göstermiştir. Buna ilişkin bilgiler *Uygulama* başlığı altında sunulmuştur. Toplamda 233 günlük veri yer almaktadır. 03.01-20.12.2019 aralığında yer alan veriler içerisinde 15 değişken (features) bulunmaktadır (EK 1). Bu değişkenler Tablo 3.1' de sunulmuştur. Bunlar içerisinde hedef değişken (y) "*YemekTuketim*" değişkenidir.

Veri seti içerisinde yer alan bağımsız değişkenler metot gereği sadece Rassal Orman Algoritması (ROA) yönteminde kullanılmıştır. Diğerlerinde zaman serisi metotları gereği sadece tarih (Index olarak) ve hedef değişken olan *"YemekTuketim"* değerleri kullanılmıştır.

Tablo 3.1' de görüleceği üzere veri setinde bulunan değişkenler, bulunduğu format ve tanımları ile yer almaktadır. Tarih değişkeni, veri setinde Index olarak yer almıştır. Veri seti içerisinde yer alan değişkenlerden 2 adet değişken sayısal veri türü olup bir tanesi hedef değişkendir. 13 adet değişken kategorik olarak yer almaktadır. Kategorik değişkenler kullanılmadan önce One-Hot-Encoding yöntemi ile ikili (binary) olarak temsil edilmesi için sayısal değerlere dönüştürülmüştür.

Kullanılan metotlar Python 3.8 programı içerisinde çalıştırılmıştır. Bunun için açık kaynak kodlu tümleşik geliştirme ortamı (IDE) olan Spyder 5 programı tercih edilmiştir.

Tablo 3.1: Veri seti değişkenleri (features).

Değişken	**Format**	**Tanım**
Tarih	GG.AA.YYYY	Veri içerisinde Index olarak kullanılmıştır.
Gun	Kategorik	Kategorik değerler olup sayısal değerlere dönüştürülmüştür. (One-Hot Encoding)
Mevsim	Kategorik	Aralık-Şubat **Kış**, Mart-Mayıs **İlkbahar**, Haziran-Ağustos **Yaz**, Eylül-Kasım **Sonbahar**
RamazanGunu	Kategorik	Ramazan ayı ise (Oruç günü) "*Evet*", değilse "*Hayir*"
FinalSinavGunu	Kategorik	Final sınavı günü ise "*Evet*", değilse "*Hayir*"
ButSinavGunu	Kategorik	Bütünleme sınavı günü ise "*Evet*", değilse "*Hayir*"
SomestrTatili	Kategorik	Sömestir tatili günü ise "*Evet*", değilse "*Hayir*"
YazTatili	Kategorik	Yaz tatili günü ise "*Evet*", değilse "*Hayir*"
HavaSicaklik	Ondalık sayı	Günlere ait ortalama sıcaklık değerleridir.
HavaDurumu	Kategorik	Bulutlu, Gunesli, Karli ve Yagmurlu.
Yemek_1	Kategorik	Çorba isimleri yer almaktadır. [2]
Yemek_2	Kategorik	Ana yemek isimleri yer almaktadır. [3]
Yemek_2_Tur	Kategorik	Ana yemek etli mi? Etli ise çeşidinin ne olduğu yer almaktadır.[4]
Yemek_3	Kategorik	Yan yemek çeşitleri yer almaktadır. [5]
Yemek_4	Kategorik	Meşrubat, Tatlı, gibi yiyecekler yer almaktadır. [6]
YemekTuketim (Y)	Sayısal	Günlük yemek tüketim sayısı
BirOncekiGun[7]	Sayısal	Bir önceki güne ait yemek tüketim sayısı (EK 2)
GecenHaftaAyniGun[8]	Sayısal	İlgili günün geçen haftaki yemek tüketim sayısı (EK 2)

3.1. YAPAY ZEKA

Yapay zekâ; bir bilgisayarın veya bilgisayar kontrolündeki bir robotun çeşitli faaliyetlerini, zeki canlılara benzer şekilde yerine getirme kabiliyetidir. Yapay zekâ, insan zekasını taklit etmeye çalışan sistemlerin genel adıdır. Yapay zekâ; makinelerin deneyimden (gerçekleşen verilerden) öğrenmesini, yeni girdilere uyum sağlamasını ve insan benzeri görevleri gerçekleştirmesini mümkün kılar. Bu tanımlardan yola çıkarak; yapay zekâ çalışmaları, insanın düşünme yöntemlerini taklit eden yapay yöntemler geliştirmeye yönelik yapılan çalışmalardır. Yapay zekâ çalışmalarına, satranç oynayan bilgisayarlar, kendi kendine giden arabalar, sesi işleyip yönlendiren bilgisayarlar, kullanıcı verilerini kullanan ve içerik öneren algoritmalar gibi

[2] Sebze, Ezogelin, Prenses, Mercimek, Yayla, Domates, Mantar, Sehriye, Tarhana, Salata, Cacik, Diger

[3] Sote, Kofte, Bezelye, Balik, Firin, Kebap, Mercimek, Kulbasti, Oturtma, Nohut, Tavuk, Turlu, Sinitzel, Ispanak, Doner, Kizartma, Fasulye, Barbunya, Kavurma, Guvec, Dolma, Musakka, Karniyarik, Bamya, Sandvic

[4] Et, Tavuk, Balik, Sandvic, Diger

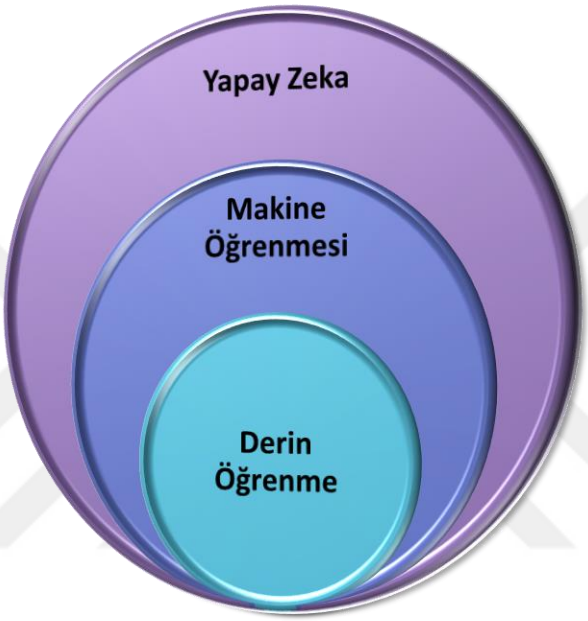
[5] Makarna, Salata, Pilav, Pilaki, Havuc, Manti, Borek, Eriste, Ispanak, Semizotu, Tursu, Pirasa, Brokoli, Piyaz, Zeytinyagli, Kizartma, Kapya, Sarma, DomatesBiber

[6] Meyve, SerbetliTatli, Yogurt, Helva, Komposto, SutluTatli, Ayran, Asure, Cacik, Limonata, Salata, Dondurma, GazliIcecek, Tursu, Kapya

[7] Veri seti 10 ve sonrası için kullanılmıştır.

[8] Veri seti 10 ve sonrası için kullanılmıştır.

çalışmalar örnek olarak gösterilebilir. Bu algoritmaları kullanan bilgisayarlar, devasa büyüklükte verileri işleyerek belirli görevleri yerine getirecek şekilde eğitilmektedir. Bunların yanında yapay zekâ çalışmaları karmaşık olmak zorunda da değildir. Burada önemli olan nokta, zekâ sahibi olmayan varlıkların çözmesini beklemediğimiz problemleri çözebiliyor olmasıdır (Yeşilkaynak, 2020).



Şekil 3-1: Yapay Zekâ ve alt dalları.

Yapay zekâ, makine öğrenmesi ve derin öğrenme yöntemlerinin birbirleri ile ilişkisi Şekil 3-1' de görülebilmektedir. Yapay zekâ 1950-1970 tarihleri arasında ortaya çıkmıştır. Makinelerin zeki olup olmadığı üzerine testler ve deneyler yapılarak yapay zekâ kavramı ortaya atılmıştır. Makine öğrenmesi kavramı 1980 yılı ile literatüre girmiştir. 1980-2010 yılları arasında özellikle üzerinde çalışılmış olan bir disiplin olmuştur. Makine öğrenmesi yapay zekanın bir alt dalıdır. Derin öğrenme sözcüğü 2010 yılı ile ortaya çıkmıştır. Çok katmanlı yapay sinir ağlarının geri yayılım (backpropagation) isimli algoritma ile eğitilmesine verilen bir isimdir. Derin öğrenme, büyük veri ile tek bir katmanda değil, birçok katmanda makine öğreniminde kullanılan matematiksel hesapları tek bir seferde yapan, makine öğreniminde tanımlanan parametreleri kendisi keşfeden, hatta daha iyi parametreler ile değerlendirmelerde bulunabilen bir sistemdir.

3.1.1. Makine Öğrenmesi

Makine öğrenmesi, makinelerin veri analizi yaparak insanlar gibi öğrenmelerini sağlayan bir bilimdir. Bir bilgisayarın doğrudan yönergeler olmadan öğrenmesine yardımcı olmak için matematiksel modelleri kullanma işlemidir. Makine öğrenmesi yapay zekanın bir alt dalı olarak kabul edilir. Makine öğrenmesi, verilerdeki desenleri (pattern) belirlemek için algoritmaları kullanır. Tahmin yapabilen bir veri modeli oluşturmak için de bu desenler kullanılır. Tıpkı insanların daha fazla alıştırma yaptıkça gelişmesi gibi, veri sayısı arttıkça makine öğrenmesinin sonuçları da daha doğru hale gelir. Makine öğrenmesi, insanların karmaşık veri kümelerini keşfetmesine, analiz etmesine ve bunlarda anlam bulmasına yardımcı olan kod parçacıklardır. Makine öğrenmesi modelinin hedefi, insanların tahmin yapmak veya bilgileri kategorilere ayırmak için kullanabileceği desenler oluşturmak veya keşfetmektir (Hurwitz & Kirsch, 2018).

Makine öğrenmesi algoritmaları, büyük kümeyi temsil eden eğitim verilerini temel alan parametreleri kullanır. Eğitim verileri büyüdükçe, algoritma daha doğru sonuçlar ortaya çıkarır. Her algoritma, verileri farklı şekilde analiz eder. Makine öğrenmesi; denetimli öğrenme, denetimsiz öğrenme ve pekiştirmeye dayalı öğrenme olarak gruplandırılır.

- **Denetimli öğrenme:** Denetimli makine öğrenimi algoritmaları en yaygın olarak kullanılanlardır. Bu model sayesinde, veriler algoritmaya bağlı olarak bir kılavuz gibi hareket eder ve algoritmaya hangi sonuçlara varması gerektiğini öğretir. Denetimli öğrenim sırasında algoritma tıpkı resimli bir kitaptan ezberleyerek meyveleri öğrenmeye çalışan bir çocuk gibi, daha önceden etiketlenmiş ve önceden tanımlanmış bir çıktısı olan veri kümeleri ile eğitilir (Güzel, 2018).
- **Denetimsiz öğrenme:** Denetimli öğrenmeden farklı olarak; verileri etiketlemeden, veri içerisinde var olan ilişkilerin ve yapıların öğrenilmesidir (Güzel, 2018). Veri örneklerinin birbirine olan uzaklıklarını, komşuluk ilişkilerini ve yoğunluklarını kullanarak veriyle ilgili çıkarımlar yapılmasını sağlar. Denetimsiz öğrenmenin iki önemli yaklaşımı boyut indirgeme ve kümelemedir. Örneğin, bir sosyal ağda tanınan kişiler arkadaş olarak eklenir. Sosyal ağ sitesi ise üyelerini ekledikleri kişilere göre sınıflandırarak belirli arkadaş grupları oluşturur ve kullanıcılara "*tanıyor olabileceğiniz kişiler*" diyerek önerilerde bulunur. Sistemin kullanıcıya sunduğu "*sadece tanıyor olabileceği kişiler arkadaşlık isteği gönderebilsin*" seçeneği de denetimsiz öğrenmeye örnek gösterilebilir.

- **Pekiştirmeye dayalı öğrenme:** Pekiştirmeli öğrenme, amaca yönelik ne yapılması gerektiğini öğrenen bir makine öğrenmesi yaklaşımıdır. Pekiştirmeli öğrenme, bir konu ya da sorun hakkında çalışma ve çözüm yöntemidir (Soylu, 2017). Satranç oyuncusunun bir hamle kararı aldığında muhtemel hamleleri ve karşıt cevaplarını planlayarak sezgisel yargılarla hareketlerini belirlemesi, bir mobil robotun daha fazla çöp toplamak için yeni bir odaya girip girmeyeceğine karar vermesinde, akünün mevcut şarj seviyesine ve geçmişte şarj cihazını ne kadar çabuk ve kolay bulabildiğine dayanması gibi uygulamalar pekiştirmeye dayalı öğrenmeye örnek gösterilebilir.

Makine öğrenmesi algoritmaları el ile çözümlenmesi oldukça zor olan ve uzun süre alan analizlerdir. Bundan dolayı bilgisayarlar ile çözümleme işlemleri yapılmaktadır. Yaygın olarak kullanılan makine öğrenmesi algoritmaları aşağıda verilmiştir.

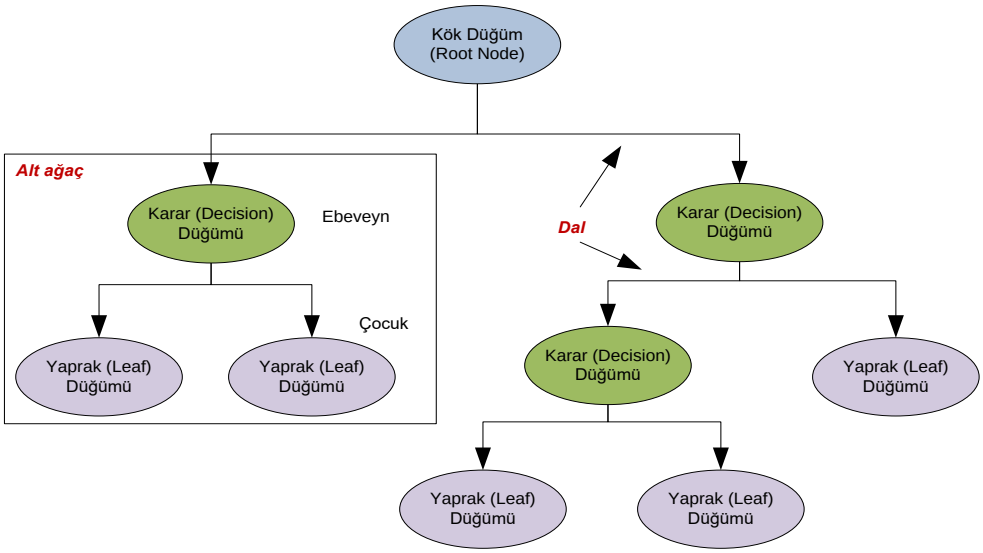
- *İkili sınıflandırma algoritmaları:* Evet/Hayır gibi yalnızca olası iki yanıtı bulunan soruların cevabı için kullanılmaktadır. Naive Bayes sınıflandırıcı algoritması örnek olarak gösterilebilir.
- *Çoklu sınıflı (çok terimli) sınıflandırma algoritmaları:* Veriler üç veya daha fazla kategoriye bölünür. Üç veya daha fazla olası yanıtı bulunan problemler için kullanılır. Karar ağacı algoritmaları örnek olarak verilebilir.
- *Regresyon algoritmaları:* Geçmiş verileri temel alarak yeni bir veri vektörünün değerini tahmin eder. Doğrusal regresyon, Lojistik regresyon, Rassal orman regresyon algoritmaları örnek olarak gösterilebilir.
- *Zaman serisi algoritmaları:* Belirli biz zaman aralığında bir grup değerin zaman içinde nasıl değiştiğini gösterir. Zaman serisi analizi ve zaman serisi tahminiyle, veriler zaman içinde düzenli aralıklarla toplanır ve bu veriler ile tahminler yapılır, eğilimler tespit edilir, mevsimsellik ve durağanlıkları belirlemek için kullanılır. ARIMA, Gri yöntem gibi algoritmalar örnek olarak gösterilebilir.
- *Kümeleme algoritmaları:* Veri noktaları arasındaki benzerlik düzeyini belirleyerek verileri birden fazla gruba böler. K-ortalamalar algoritmaları örnek olarak gösterilebilir.
- *Sınıflandırma algoritmaları:* Verileri önceden ayarlanmış kategorilere atamak için tahmine dayalı hesaplamalar kullanılır. Sınıflandırma algoritmaları giriş (gerçekleşen) verileri ile eğitilir. K-En yakın komşu algoritmaları, Karar ağacı algoritmaları, Destek

vektör makine algoritmaları, Rassal orman sınıflandırma algoritmaları örnek olarak gösterilebilir.

Makine öğrenmesi, bir öğrenme (eğitim) sürecine uygulanabilecek doğru veri kümesini gerektirir. Makine öğrenmesi tekniklerini kullanmak için büyük veriye sahip olunması gerekmez; ancak büyük veri, makine öğrenmesi modellerinin doğruluğunu artırır (Hurwitz & Kirsch, 2018).

3.1.1.1. Karar Ağacı Algoritması

Sınıflandırma işlemi, makine öğrenmesi yöntemlerinde öğrenme (eğitim) ve tahmin (test) olmak üzere iki adımlı bir süreçtir. Kullanılan algoritmaya bağlı olarak, öğrenme adımında, eğitim verilerine göre model geliştirilir. Test aşamasında test verilerine göre tahmin işlemleri gerçekleştirilir. Karar ağacı algoritması, sınıflandırma ve tahmin işlemlerinde kullanılan algoritmalarından biridir. Karar Ağacı algoritması, denetimli öğrenme yöntemleri arasında yer almaktadır. Karar ağacı algoritması hem sınıflandırma hem de regresyon problemleri için kullanılması özelliği ile diğer denetimli öğrenme yöntemleri arasında öne çıkmaktadır (Chauhan, 2020).



Şekil 3-2: Karar ağacı yapısı.

Şekil 3-2' de örnek bir karar ağacı yapısı görülmektedir. Bir karar ağacı, verilerin birbirine bağlanmasıyla oluşturulan, ağaç yapısına benzeyen bir yapıya sahip olan modellerdir.

- ✓ Bir karar ağacının her bir elemanına düğüm (node) denir.
- ✓ Düğümler birbirine dal (edge) yardımıyla bağlanır.
- ✓ Karar ağacının en tepesindeki düğüm kök (root) adını alır. Sonraki bölünmelere ait düğümlere karar düğümü (decision node) denmektedir.
- ✓ Bir düğümün alt ağaçlarına alt ağaç (subtree) denir.
- ✓ Çocuğu olan düğümlere ebeveyn (parent) denir
- ✓ Çocuğu olmayan düğümlere ise yaprak (leaf) denmektedir.
- ✓ Bir düğüme ulaşmak için üzerinden geçilen düğümler yol (path) adını alır.
- ✓ Söz konusu bir düğümün derinliği (depth), o düğümden kök (root) düğüme kadar olan yolun uzaklığıdır.

Karar ağaçlarında her bir değişken (öznitelik) bir düğüm (node) ile temsil edilir. Ara düğümler, bir öznitelik sınamasını, dallar sınama sonuçlarını ve yapraklar (leaf) sınıfları ifade etmektedir.

Karar ağacı algoritmaları yinelemeli (recursive) süreçlerdir. Her adımda, bir veri seti verilir, veri setinde bir değişken (feature) seçilir ve seçilen değişkenin bir değerine göre bir bölme yapılır. Daha sonra bu bölme, veri setini alt gruplara bölmek için kullanılır. Bölünme sonucu oluşan her alt küme, bir sonraki adım için veri seti olarak kabul edilir. Karar ağacı algoritmasında önemli olan nokta, bölmelerin nasıl ve hangi yöntemlerle seçileceğidir (Zhou, 2012).

Karar ağacı algoritması adımları aşağıdaki gibi tanımlanabilir:

i. Veri setinin önem derecesi en yüksek (En iyi bilgi kazanımı veren) bağımsız değişkenini (özniteliği) ağacın köküne (root) yerleştir.

ii. Veri setini alt gruplara ayırın. Alt gruplar, her alt grubun bir öznitelik için aynı değere (veya sayısal değerler için belli bir değerin ötesinde) sahip verileri içerecek şekilde yapılması gerekir.

iii. Karar ağacının tüm dallarında yaprak düğümleri (leaf node) bulana kadar her alt kümede 1. ve 2. adımları tekrarlayın.

Tahmin edilen hedef değişkene bağlı olarak iki tip karar ağacı algoritması olduğu bilinmektedir. Bunlar; kategorik hedef değişkeni olan karar ağacı ve sayısal bir hedef değişkeni olan karar ağacıdır. Karar ağacı algoritması kullanılırken bazı varsayımlar mevcuttur. Başlangıçta tüm eğitim veri seti kök olarak kabul edilmektedir. Değişken (feature) değerlerinin kategorik olması

tercih edilmektedir. Ancak değerler sürekli sayısal değerlerden oluşuyorsa, model oluşturulmadan önce değerler ayrıklaştırılır.

Karar ağacı algoritmasında, bölünme yapılacak değişkenin (feature) kök düğüm veya sonraki seviyelerde seçilmesi işlemi oldukça önemli bir konudur. Bu işlem öznitelik seçimi (attribute selection) olarak bilinmektedir. Öznitelik seçimi için kullanılan farklı yöntemler mevcuttur. Öznitelik seçiminde bölünme işlemi karar ağacı algoritmasının doğruluk derecesini ciddi bir şekilde etkilemektedir. Öznitelik seçimi için kullanılan kriter (yöntem) sınıflandırma ve regresyon modelleri için farklılık göstermektedir (Chauhan, 2020).

Öznitelik seçimi yöntemleri

N sayıda öznitelikten (features) oluşan bir veri setinde, hangi özniteliğin kök olarak veya karar ağacının farklı seviyelerinde yerleştirileceğine karar vermek karmaşık bir süreçtir. Öznitelik seçiminde rastgele olarak bir seçim, sorunun çözümü için uygun bir yöntem değildir. Rastgele bir yaklaşım uygulamak, doğruluk seviyesini düşürücü bir adım olacaktır. Öznitelik seçimi problemini çözmek için aşağıda bazı yöntemler sunulmuştur;

- ✓ Entropi
- ✓ Bilgi kazanımı
- ✓ Gini indeksi (Gini impurity)
- ✓ Kazanım oranı
- ✓ Varyansta azaltma
- ✓ Ki-kare

Bu çalışma kapsamında kullanılan yöntem Rassal Orman Regresyon (ROR) yöntemi olması sebebiyle regresyon temelinde geliştirilen bir karar ağacına göre yapılan öznitelik seçiminde bölme işlemi için MSE değeri kullanılmaktadır. Kullanılan algoritma bilgisayar ortamında Python programlama dili kullanan Spyder IDE programı aracılığıyla çalıştırılmıştır. Yazılım içerisinde "*sklearn.ensemble*" kütüphanesi (framework) kullanılmıştır. Kütüphane içerisindeki "*RandomForestRegressor*" sınıfı (class) çalıştırılmıştır. "*RandomForestRegressor*" sınıfı, kaynak kodun içerisinde temel olarak CART yöntemini kullanmaktadır. CART yöntemi regresyon tipi bir karar ağacı çalıştırırken öznitelik seçiminde bölme kriteri olarak MSE performans ölçütünü kullanmaktadır. Öznitelik seçimi olarak gini indeksi yöntemi

kullanılmaktadır. Gini indeks değeri en küçük olan öznitelik yani belirsizlik azalması en yüksek olan öznitelik, düğümü bölmek için seçilir (Géron, 2017).

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2 \tag{3-1}$$

(3-1' deki denklemde belirtildiği üzere Gini indeksi formüle edilmiştir.

- G_i: Gini indeks değeri
- $p_{i,k}$: i' inci düğümdeki veri setinde k sınıfı örneklerinin oranını ifade etmektedir.
- n: Gini İndeks ikili (binary) ise 2 olarak alınır.

MSE ölçütünü kullanan Gini indeksi aşağıda (3-2)' de yer alan denklemde sunulmuştur (Géron, 2017).

$$J(k, t_k) = \frac{m_{left}}{m} MSE_{left} + \frac{m_{right}}{m} MSE_{right} \tag{3-2}$$

$$MSE_{node} = \sum_{i \in node} \left(\hat{y}_{node} - y^{(i)}\right)^2 \tag{3-3}$$

$$\hat{y}_{node} = \frac{1}{m_{node}} \sum_{i \in node} y^{(i)} \tag{3-4}$$

- k: Veri setinde bölünmek için seçilen öznitelik (feature)
- t_k: Seçilen öznitelik için eşik değeri (threshold)
- m: İlgili düğümdeki veri setinin örnek sayısı
- $m_{left/right}$: Eşik değerinin iki tarafına ait örnek sayıları
- $MSE_{left/right}$: Eşik değerinin iki tarafına ait örneklerin MSE değerleri
- $\hat{y}_{node}$: İlgili düğümde seçilen öznitelik değerlerinin ortalama değeri
- $y^{(i)}$: İlgili düğümde yer alan veri setindeki i' inci örnek

Öznitelik (Değişken) Önemi (Feature Importance)

Geliştirilmiş bir karar ağacına bakıldığında, bağımlı değişken üzerinde önemli olan bağımsız değişkenler (öznitelikler) ağacın köküne (root) daha yakın görünürken, daha önemsiz olanlar genellikle yapraklara (leaf node) daha yakın görünmektedir. Bu nedenle karar ağacında ortalama derinliği hesaplayarak bir özelliğin önemi hakkında bir tahminde bulunmak mümkündür (Géron, 2017). Her bir özniteliğin ne kadar önemli olduğunun hesaplanması için karar noktasının (decision node) kendisine ve o karar noktasından türemiş alt düğümlerine (child node) bakılmaktadır. Bu çalışma kapsamında kullanılan yazılım öznitelik önemi için çalıştırılmıştır. Söz konusu yazılım algoritmayı çalıştırırken Gini indeks (binary) yöntemini kullanmaktadır. Ölçüt ya da bölme kriteri olarak ise MSE kullanılmaktadır. Öznitelik önemi yapılırken aşağıdaki işlem uygulanır (Serengil, 2020).

$$
\begin{aligned}
&\text{Öznitelik önemi} \\
&= \textit{Karar noktasına ait Gini indeksi} \\
&* \textit{Sağlayan örnek sayısı} - \textit{Sol alt düğüme ait Gini indeksi} \\
&* \textit{Sol alt düğümde sağlanan örnek sayısı} - \textit{Sağ alt düğüme ait Gini indeksi} \\
&* \textit{Sağ alt düğümde sağlanan örnek sayısı}
\end{aligned}
$$

Gini indeksi, 0 (homojen) ile 1 (heterojen) aralığında bir homojenlik skorudur. Bölme kriterinin yani Gini indeks değerindeki değişiklikler her özellik için toplanır ve hesaplamanın sonunda normalizasyon yapılır. Üzerinde bölünme gerçekleştirilen öznitelikte, Gini indeksindeki azalmalar yüksek homojenliğe sahip düğümlerle sonuçlanır.

Öznitelik önemi hesaplama adımları aşağıda sunulmuştur (Radchenko, et al., 2020). Verilen adımlar Rassal Orman Algoritması dahilinde sunulmuştur. Yanısıra tek bir karar ağacı içinde kullanılabilmektedir.

a. Topluluktaki her ağaç için $t \in \{1, \dots, N\}$
   i. Her düğüm i için Gini indeks indirgeme değerlerini (MSE ölçütü ile) hesapla

$$RI_i^t = w_i^{(t)}.I_i^{(t)} - w_{LEFT_i}^{(t)}.I_{LEFT_i}^{(t)} - w_{RIGHT_i}^{(t)}.I_{RIGHT_i}^{(t)} \qquad (3\text{-}5)$$

   ii. Karar ağacı t içerisinde her bir öznitelik j için önemi hesapla

$$FI_j^{(t)} = \frac{\sum_{i:j\ özniteliğindeki\ i'inci\ düğüm} RI_i^{(t)}}{\sum_{i \in tüm\ düğümler} RI_i^{(t)}} \quad (3\text{-}6)$$

iii. Topluluktaki tüm ağaçlardaki öznitelik önemlerinin ortalamalarını hesapla

$$FI_j = \frac{\sum_{t=1}^{N} FI_j^{(t)}}{N} \quad (3\text{-}7)$$

- RI_i^t: Gini indeks (indirgeme) değeri
- $w_i^{(t)}$: t ağacının i' inci düğümündeki örneklerin sayısının kendisine oranı,
- $I_i^{(t)}$: t ağacının i' inci düğümündeki MSE değeri
- $w_{LEFT_i}^{(t)}$: t ağacının i' inci düğümündeki sınıflardan birindeki (LEFT) örneklerin sayısının düğümde örnek sayısına oranı,
- $I_{LEFT_i}^{(t)}$: t ağacının i' inci düğümündeki sınıflarından birindeki (LEFT) MSE değeri
- $w_{RIGHT_i}^{(t)}$: t ağacının i' inci düğümündeki sınıflardan diğerindeki (RIGHT) örneklerin sayısının düğümdeki tüm örneklere oranı,
- $I_{RIGHT_i}^{(t)}$: t ağacının i' inci düğümündeki sınıflarından diğerindeki (RIGHT) MSE değeri

Karar ağacı algoritmasında eğitim veri seti üzerindeki öğrenme aşamasında bazen aşırı (uyum) öğrenme (overfitting) durumu ortaya çıkmaktadır. Bu durum eğitim veri setinde %100 doğrulama (%0 MAPE) durumuna sebep olduğundan modelin ezberleme sorununa sebep olmaktadır. Bu da test verilerini yani eğitim veri seti dışında olan verilerin tahmin edilmesinde doğrulama seviyesini etkilemektedir. Aşırı uyumu önlemek için kullanılan iki yöntem mevcuttur (Chauhan, 2020): Budama (Pruning) ve Rassal Orman Algoritması

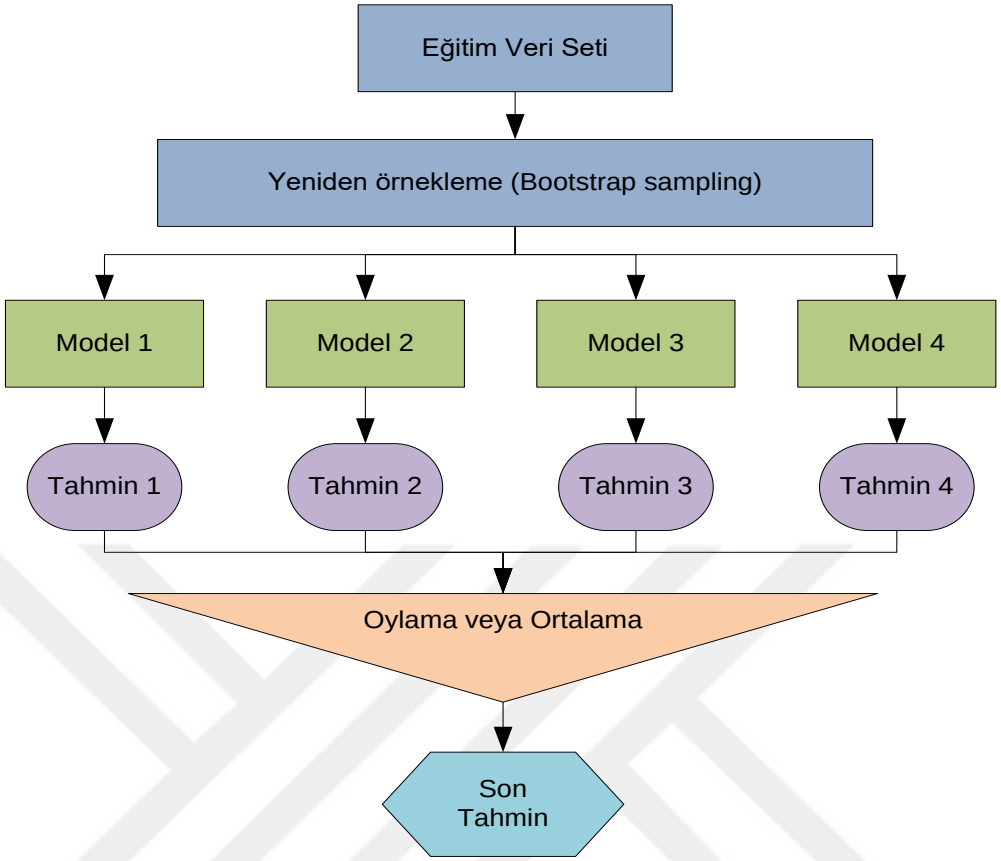
- **Budama (Pruning):** Aşırı uyum gösteren karar ağaçlarının sınıflama işleminde etkisiz kalan dallarında yer alan tahmin edici değişkenlerin karar ağacından çıkarılması işlemidir. İki türlü budama işlemi vardır; öznitelik bölme işleminden önce değişkenleri teker teker ele alarak modelin tahmin gücü açısından hangisinin daha önemli olduğu irdelenerek dallanmalar ilerletilir. Bu yönteme *preprunning (ön budama)* denir. Bitmiş bir karar ağacının modele katkı yapmayan dallarının tespit edilerek modelden çıkarılması işlemine de *postprunning (geç budama)* denir.

- **Rassal Orman Algoritması:** Rassal Orman Algoritması, daha iyi tahmin performansı elde etmek için birden fazla makine öğrenimi algoritmasının birleştirildiği bir topluluk öğrenme yöntemidir. Bu yöntem süreci itibari ile birden fazla karar ağacı üretip, üretilen sonuçların regresyon ise ortalamasını, kategorik ise en fazla olan kategoriyi tahmin sonucu olarak aldığından dolayı aşırı uyumun önüne geçilen bir yöntem olarak düşünülmektedir.

3.1.1.2. Topluluk Öğrenmesi (Ensemble Learning)

Makine öğrenimi ve veri madenciliğinin önemli bir görevi, veri setlerinden iyi modeller oluşturmaktır. Veri setleri özellik (değişken) vektörlerinden oluşur ve her değişken vektörü, bir dizi özelliği olan bir nesnenin açıklamasıdır (Zhou, 2012).

Sınıflandırma algoritmaları ile bir nesnenin hangi sınıfa dahil olduğu tahmin edilmeye çalışıldığında, birçok sınıflandırma yöntemi arasından probleme uygun olanı seçilir. Gerekli optimizasyonlar yapıldıktan sonra, yüksek doğruluk oranları yakalanmaya çalışılır. Bu iş 3, 4 ya da 5 sınıflandırıcı ile yapıldığında veya aynı sınıflandırıcı aynı eğitim setinin farklı alt kümeleri ile eğitildiğinde bulunan sonuçların oylama veya ortalamaların alınmasıyla ortaya çıkan değer topluluk öğrenmesinin bir sonucu olur. Topluluk öğrenmesi yönteminde, aynı sınıflandırma görevinde birden fazla sınıflandırıcı kullanılır. Bu yöntemde farklı doğruluk skorlarına sahip sınıflandırıcıların sonuçları farklı yöntemlerle (oylama, ortalama vb.) birleştirilir. Böylelikle tek bir sınıflandırıcıdan daha iyi sonuçlar elde etme imkânı bulunur (Zhou, 2012).



Şekil 3-3: Topluluk öğrenmesi yönteminin genel bir yapısı.

Yaygın olarak kullanılan topluluk öğrenmesi yöntemleri aşağıdaki gibidir:

- **Bagging:** Bagging olarak da bilinen "**B**ootstrap **Agg**regating" aşırı uyumu (overfitting) önlemek için 1994 yılında Leo Breiman tarafından önerilen güçlü bir topluluk öğrenme yöntemidir. Bagging yönteminin arkasındaki kavram, daha doğru bir tahmin sonucu oluşturmak için veri seti alt gruplarından elde edilen örnekleri kullanan öğrenicilerin tahminlerini birleştirmektir. Değişken (Kararsız) öğrenme algoritmalarında eğitim verilerindeki küçük değişikliklerin tahminlerde büyük değişikliklere yol açtığı görülmüştür. Bagging yöntemi bu değişikliğe çözüm olarak öne sürülmüştür. Sinir ağları ve karar ağaçları gibi algoritmalar değişken (kararsız) öğrenme algoritmalarına örnek olarak gösterilebilir. Bootstrap aggregation (bagging) küçük veri kümelerinde oldukça etkilidir. Bagging yöntemi yüksek değişkenliğe sahip olan algoritmaların

değişkenlik (varyans) düzeyini azaltmaya yardımcı olur. Bagging yöntemi sınıflandırma ve regresyon problemlerini desteklemektedir (Sarkar & Natarajan, 2019).

X bir eğitim veri seti ile yapılan yeniden örneklemeler (bootstraping) sonucunda N adet $X_1, X_2, X_3, \dots, X_N$ gibi örnek grupları elde edilir. Her örnek grupları için veri seti eğitilir. Öğrenici algoritmalar tarafından eğitilen örnek grupları sonucunda sınıflandırmalar elde edilir. Sınıflandırmalar sonucunda birleştirilmiş sınıflandırma (bagging yöntemi) tüm sonuçların ortalamasını alarak bir değer üretir veya oylama sonucundaki değeri verir.

$$b(x) = \frac{1}{N}\sum_{i=1}^{N} b_i(x) \tag{3-8}$$

(3-8)' deki denklemde görüleceği üzere eğitim veri setinden elde edilen örnek grupları $b_i(x)$ sınıflandırıcıları ile eğitilir. Bulunan sonuçların ortalaması alınarak bagging yönteminin sonucu olan $b(x)$ değeri (sonucu) elde edilir.

Bagging yönteminde sınıflandırma yöntemleri için sözde kod (pseudo-code) olan algoritma aşağıda verilmiştir.

$Bagging\ \left(S = \left((x_1 y_1), \dots, (x_m y_m)\right)\right)$

1 $t = 1'den\ T'ye\ kadar\ yap$

2 $S_t \leftarrow Bootstrap(S)$ ⇨ S veri setinden örneklemeleri seç

3 $h_t \leftarrow S_t$ örnek grupları ile sınıflandırma algoritması çalıştır

4 $h_s = x \rightarrow$ Çoğunluk oy $\left(\left(h_1(x), \dots, h_T(x)\right)\right)$ değerini dönder

Regresyon algoritmaları için kullanılan bagging yöntemi aşağıdaki gibidir.

$Bagging\ \left(S = \left((x_1 y_1), \dots, (x_m y_m)\right)\right)$

1 $t = 1'den\ T'ye\ kadar\ yap$

2 $S_t \leftarrow Bootstrap(S)$ ⇨ S veri setinden örneklemeleri seç

3 $h_t \leftarrow S_t$ örnek grupları ile regresyon algoritmasını çalıştır

4 $h_s = x \rightarrow Ortalama\ \left(\left(h_1(x), \dots, h_T(x)\right)\right)$ değerini dönder

- **Random Subsample (Rastgele altuzaylar):** Bu yöntemde örneklerin seçimiyle ilgili bir kısıtlama yoktur. Burada her öğreniciye değişkenler kümesinin değişik alt kümeleriyle öğrenme yaptırılır. Yani eğitim esnasında bazı değişkenler seçilir bazıları seçilmez (Şirin, 2021).

- **Boosting:** Boosting yöntemi bagging yönteminin farklı bir türüdür. Boosting yönteminde öğrenici sonuçlarının ardından sonuçlar, öğrenici için kullanılmaktadır. Diğer yöntemlere göre daha yaygındır. Eğitim için ayrılan veri setinden bir öğrenici için rastgele seçim yapılır. Öğrenme gerçekleşir, model test edilir. Sonuçlardan yanlış sınıflandırılan örnekler belirlenir. Bunlar bir sonraki öğrenici için örnek seçiminde önceliklendirilir. Her seferinde bu bilgi güncellenir (Şirin, 2021).

Bagging işleminde, herhangi bir örnek grubu başka bir örnek grubuna bağlı değildir. Bu nedenle hepsi paralel çalışır. Boosting, sıralı bir şekilde çalışır ve yeniden örnekleme (bootstrap sampling) işlemini içermez. Hem bagging hem de boosting yöntemleri, farklı modellerden (öğrenicilerden) tahminleri tek bir sonuçta birleştirerek varyansı azaltır. Bununla birlikte, tek bir model aşırı uygunsa (overfitting), boosting yöntemi uygun olmayabilir. Model fazla uyum gösteriyorsa (overfitting) bagging daha iyi bir seçenek olabilir. Öte yandan, boosting yöntemi yanlılığı (bias) azaltmaya çalışırken, bagging yanlılığı (bias) nadiren düşürür.

Topluluk öğrenmesi yöntemlerinin iyi performans göstermesi için yeterli miktarda veri olması gerekmektedir. Ayrıca topluluk öğrenmesi yöntemleri, büyük ve doğrusal olmayan (non-linear) veri setlerine sahip olunduğunda daha yararlı olmaktadır (Sarkar & Natarajan, 2019).

3.1.1.3. Varyans ve Yanlılık (Variance and Bias)

Tahmin veya sınıflandırma yapmak için kullanılan modeller ne kadar ayar (tuning) yapıldığına bakılmaksızın, her zaman yüksek varyans (variance) veya yüksek yanlılık (bias) risklerini içinde barındırır. Model eğitilirken yanlılık ve varyans dikkate alınmazsa en iyi model bile başarısız olabilmektedir (Sarkar & Natarajan, 2019).

Hem yanlılık (bias) hem de varyans (variance) bulunan tahmin içerisinde yer almakta ve hatayı temsil etmektedir. Toplam hata; yanlılık (bias), varyans ve kaçınılmaz olan gürültü (noise) hatalarından (azaltılamayan hata) oluşmaktadır. Gürültü, rassal hata veya istatistiksel belirsizlik olarak da adlandırılmaktadır. Gürültüyle ilgili hata, esas olarak eğitim verilerinden

kaynaklanmaktadır ve toplam hatadan kaldırılamaz, her zaman toplam hata içerisinde temsil edilir. Bununla birlikte, yanlılık ve varyans ile ilgili hatalar azaltılabilmektedir (Sarkar & Natarajan, 2019).

Toplam hata aşağıdaki denklem ile temsil edilmektedir (Sarkar & Natarajan, 2019):

$$Toplam\ Hata = Yanlılık^2 + Varyans + Azaltılamayan\ Hata \quad (3\text{-}9)$$

Ortalama kare hatası (MSE) performans ölçütü, tüm bu hataları yakalar ve aşağıdaki gibi temsil edilmektedir (Sarkar & Natarajan, 2019).

$$MSE = E\left[\left(Y - \hat{f}(x)\right)^2\right] \quad (3\text{-}10)$$

Formülde, E beklenen ortalamayı, Y gerçek değeri ve $\hat{f}(x)$ hedef değişken için tahmin edilen değeri temsil etmektedir. MSE ölçütü aşağıda gösterildiği gibi yanlılık (bias), varyans ve gürültü gibi bileşenlerine ayrılabilmektedir (Sarkar & Natarajan, 2019):

$$MSE = \left[E\left[\hat{f}(x)\right] - f(x)\right]^2 + E\left[\left(\hat{f}(x) - E\left[\hat{f}(x)\right]\right)^2\right] + \quad \varepsilon \quad (3\text{-}11)$$

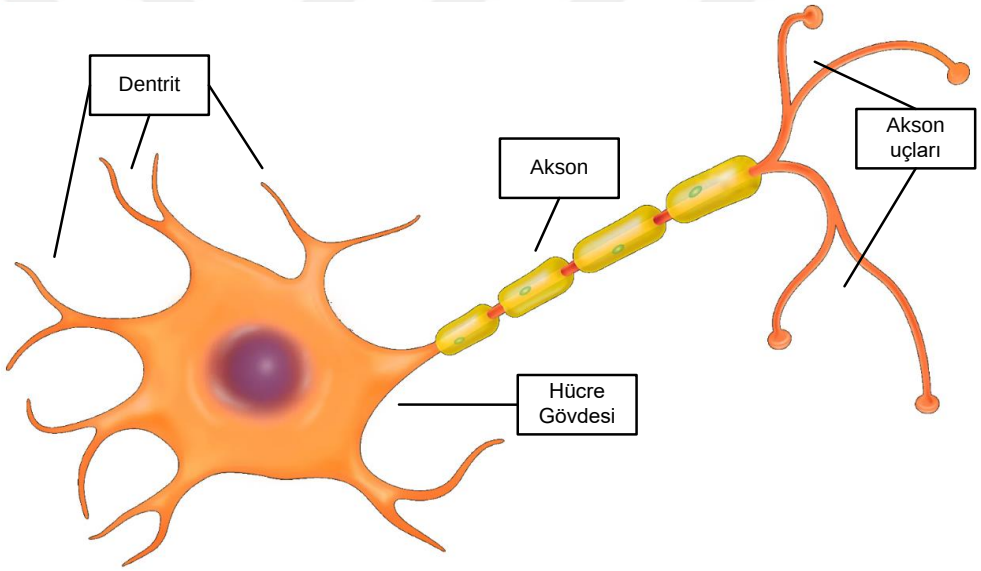
Yanlılık (Bias) Varyans Noise

Yanlılık, gerçek değerin tahminin beklenen değerine ne kadar yakın olduğunu ifade ederken, varyans ise beklenen tahmin değerinden sapmayı ölçmektedir. Her zaman küçük değerli MSE ölçütü tercih edilmektedir. MSE hatasını minimize etmek için, temel olarak gerçek değerlere yakın olması (sıfır veya düşük yanlılık) istenmektedir ve gerçek değerden düşük bir sapmaya (düşük varyans) sahip olması istenmektedir. Yüksek yanlılık, algoritmanın eğitim veri kümesindeki performansını düşürmektedir. Diğer taraftan yüksek varyans, yüksek doğrulama hataları ile nitelendirilir ve eğitim veri setinde olmayan veriler üzerinde öğrenici modellerin performansını düşürmektedir (Sarkar & Natarajan, 2019).

3.1.1.4. Yapay Sinir Ağları

Sinir ağları yapay zekâ alanının en yeni olarak ortaya çıkan bir dalıdır. İlk olarak Warren McCulloch ve Walter Pitts tarafından geliştirilen matematiksel bir yöntem olan sinir ağları insan beyninin problemleri nasıl çözdüğünü tanımlamak için kullanılmıştır. Sinir ağları geliştirilirken biyolojik olarak insan beyni çalışması sürecinden etkilenmiştir (Nagy, 2018).

İnsan beyni temel olarak sinir hücrelerinden diğer adıyla nöronlardan meydana gelmiştir. Bir sinir hücresi biyolojik olarak hücre gövdesi, dentrit ve akson bölmelerinden meydana gelmiştir. Her bir sinir hücresi kendisine iletilen bilgiyi dentrit ile alarak (girdiler), akson ile sonraki nörona taşıyarak (çıktı) bilgiyi işlemekten sorumludur.



Şekil 3-4: Biyolojik sinir hücresi (nöron) yapısı.

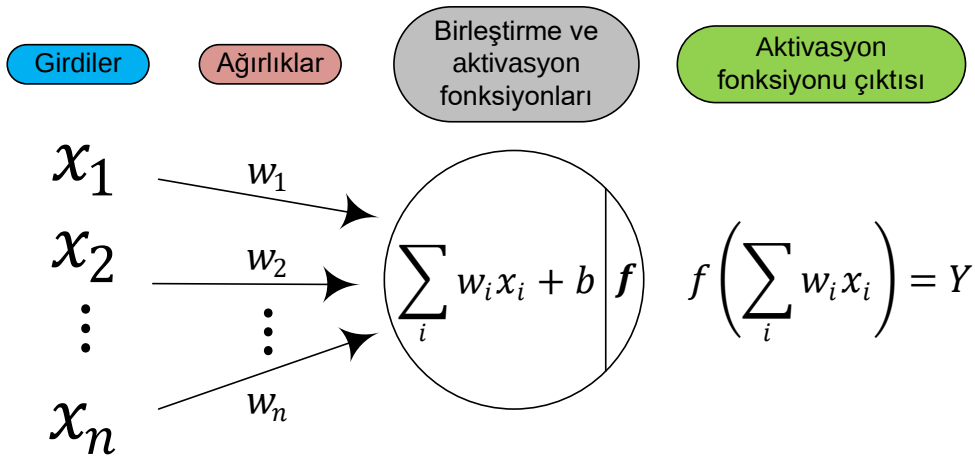
Biyolojik bir sinir hücresi yapısı Şekil 3-4' de görülebilmektedir. Bir sinir hücresine iletilen bilgi öncelikle dentrit tarafından alınır, hücre gövdesi boyunca ilerler, aksonlardan geçerek akson uçlarına ulaşır ve son olarak akson uçları ile bir sonraki nörona ait dentrit arasındaki sinaps sayesinde sonraki sinir hücresine iletim gerçekleştirilir. Benzer şekilde milyonlarca veya milyarlarca sinir hücreleri bir araya gelerek bir sinir ağı oluştururlar. Hücreler arasında sayılamayacak derecede bilgi akışları gerçekleştirilir. Yapay sinir ağları bu düşünceler ile

geliştirilmiştir. Biyolojik sinir hücresine ait terimlerin, yapay sinir ağları yönteminde karşılık gelen anlamları aşağıdaki tabloda (Tablo 3.2) görülebilmektedir.

Yapay sinir ağında yer alan bir sinir hücresi (unit) tıpkı biyolojik sinir hücresinde olduğu gibi belli ağırlık değerleri olan girdi(leri) alarak bir birleştirme fonksiyonundan geçirir, birleştirme fonksiyonu sonucunu bir aktivasyon fonksiyonuna iletir. Aktivasyon fonksiyonu sonucu olarak bir çıkış değeri üretir. Üretilen bu değer bir sonraki yapay sinir ağı birimine iletilir.

Tablo 3.2: Biyolojik sinir hücresi ve yapay sinir ağı hücresi terimleri.

Biyolojik sinir hücresi	**Yapay sinir ağı birimi**
Dentrit	Birleştirme (toplama) fonksiyonu
Hücre Gövdesi	Aktivasyon fonksiyonu
Akson	Aktivasyon (çıkış) değeri
Nöron	İşlemci (Unit)
Sinaps	Ağırlık



Şekil 3-5: Yapay sinir hücresi (unit) matematiksel modeli.

Yapay sinir ağına ait bir birimin (unit) genel bir çalışma biçimi Şekil 3-5' da sunulmuştur (Yıldırım, 2020).

Girdiler: Yapay sinir hücresinin öğrenmesi için gönderilen bilgilerdir. Bu bilgiler bir önceki hücrenin çıktısı olabileceği gibi, ilk giriş değerleri de olabilmektedir.

Ağırlıklar: Yapay sinir hücresine gelen bir bilginin hücre için önemini ve hücreye etkisini belirten değerlerdir. Pozitif, negatif, sıfır veya değişken sayılar gibi değerler alabilmektedirler.

Birleştirme Fonksiyonu: Yapay sinir hücresine gelen girdi değerlerinin ağırlıkları ile çarpılıp toplanması işlemi birleştirme fonksiyonu olarak adlandırılır. Birleştirme fonksiyonunun minimum, maksimum, çarpım gibi farklı tipleri de mevcuttur. Bu çalışma kapsamında toplama tipi kullanılmıştır.

$$s = w_0x_0 + w_1x_1 + \cdots + w_nx_n = \sum_{i=0}^{n} w_ix_i \qquad (3\text{-}12)$$

Aktivasyon Fonksiyonu: Birleştirme fonksiyonu sonucunda elde edilen değeri girdi olarak alan ve bu girdi değerini bir çıktı değerine dönüştüren bir fonksiyondur. ReLU, sigmoid veya tanh gibi tipleri bulunmaktadır.

ReLU (Rectified Linear Unit): $$R(z) = \max(0, z) \qquad \textbf{(3-13)}$$

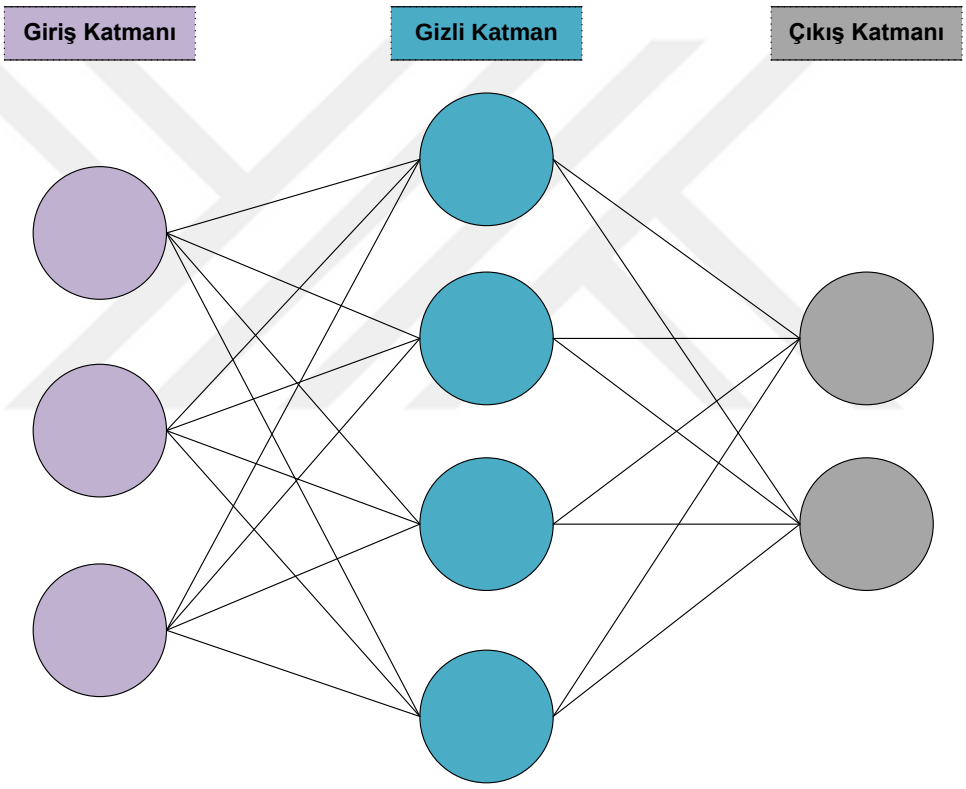
Sigmoid: $$S(z) = \frac{1}{1 + e^{-z}} \qquad \textbf{(3-14)}$$

Hiperbolik Tanjant: $$T(z) = \tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \qquad (3\text{-}15)$$

Çıktılar: Aktivasyon fonksiyonu ile üretilen değer yapay sinir hücresinin çıktısı olmaktadır. Her hücrenin tek bir çıktısı olmaktadır. İstenen çıktı sayısı kadar sinir hücresi gerekecektir. Yapay sinir hücresi tarafından üretilen çıktı ya bir sonraki sinir hücresinin girdisi olarak devam edebilmekte ya da sistemin çıktısı olabilmektedir.

Yapay Sinir Ağı Yapısı

Birden fazla yapay sinir hücreleri bir araya gelerek yapay sinir ağını oluştururlar. Yapay sinir ağı genel olarak; giriş katmanı (input layer), ara/gizli katman (hidden layer), çıkış katmanından (output layer) oluşmaktadır. Yapay sinir hücrelerinin her biri aynı usul ile çalışmaktadır. Sinir hücrelerinin her biri aynı fonksiyonları içermek zorunda değillerdir. Giriş katmanı ilk iletilen bilgileri alarak ara katmana iletir. Buradan da çıkış katmanına doğru iletilir. Çıkış katmanında bulunan nöron (sinir hücresi) sayısı problemin istenen hedef değişken sayısını göstermektedir (Yıldırım, 2020).



Şekil 3-6: Yapay sinir ağı yapısı.

Yapay sinir ağının örnek bir görünümü Şekil 3-6' de sunulmuştur.

Yapay sinir ağları ileri ve geri beslemeli olarak uygulanabilmektedir. İleri beslemeli yapay sinir ağlarında ağ yapısı katmanlar halindedir. Bir katmanda yer alan nöronlar arasında bağlantı bulunmaz. Bilgiler giriş katmanından çıkış katmanına doğru, tek yönlü bir şekilde ilerler. Geri beslemeli yapay sinir ağlarında çıkış ve ara katmanda yer alan nöronların çıktı değerleri bir

önceki katmandaki nöronlara bilgi olarak geri gönderilir. Bu da hem ileri hem de geri doğru bir besleme olduğunu gösterir. Bu uygulama özellikle tahmin çalışmalarında kullanılabilmektedir. Tekrarlayan Sinir Ağları (Recurrent Neural Network), Uzun Kısa Süreli Bellek (Long Short Term Memory), Kapılı Tekrarlayan Sinir Ağları (Gated Recurrent Unit) yöntemleri geri beslemeli sinir ağlarına örnek olarak gösterilebilir. Adı geçen bu üç yöntem bu çalışma kapsamında uygulanmıştır.

Yapay sinir ağları eğitim amacıyla çalıştırılırken ağırlıklar rassal olarak, sabit bir sayı veya sıfır ile başlatılabilmektedir. Bu ağırlık değerleri ağdaki en uygun ağırlık değerleri olmalıdır. Atanan ağırlık değerleri ile ağ bir değer üretir (çıkış değeri). Ancak üretilen bu değer, eğitim veri setindeki gerçek değer ile kıyaslanınca farklılık çıkabilmektedir. Bu kıyaslamada kullanılan fonksiyon, hata (loss) fonksiyonu olarak adlandırılmaktadır. Kıyaslama aşamasında yaygın olarak MAPE, MSE, MAE gibi göstergeler (loss function) kullanılabilmektedir. Kıyaslamada ortaya çıkan sapma değerleri, eğitilen modelin doğruluğu hakkında bilgi vermektedir. Bu aşamada önemli olan en yüksek doğruluk düzeyine ve en küçük sapma değerine (yüzdesine) ulaşabilmektir. En küçük sapma değerine yani en yüksek doğruluk değerine ulaşabilmek için ağırlık değerleri her iterasyon sonucunda güncellenerek model tekrardan çalıştırılmaktadır. Belli bir hata değerine ulaşana kadar ya da belli bir iterasyon sayısına ulaşana kadar model ağ çalıştırılır ve en uygun ağırlık değerleri elde edilir. Hata değeri geriye yayılarak ağırlıkların güncellenmesi sağlanır ve yeni çıktı değerleri elde edilir. Yapılan bu işlem de geri yayılım (backpropagation) algoritması olarak adlandırılır. Geri yayılım algoritması olarak kullanılan farklı algoritmalar mevcuttur. Bu çalışma kapsamında "*adam (adaptive moment estimation) optimizer*" isimli geri yayılım algoritması kullanılmıştır.

Yapay sinir ağlarını uygulamak için genel olarak kullanılan süreç aşağıdaki gibidir:

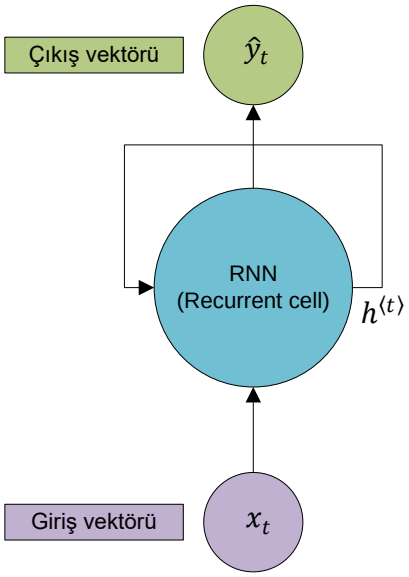
i. Ağırlık matrislerini rassal bir dağılım ile başlat ve bias değerlerini sıfır olarak ata
ii. Tahminleri hesaplamak için ileri beslemeyi gerçekleştir
iii. Kayıp fonksiyon değerini hesapla
iv. Gradyanları (türevleri) hesaplamak için geriye yayılım yap
v. Gradyanlara bağlı olarak ağırlıkları güncelle
vi. 2, 3, 4 ve 5 adımlarını gerçekleştir.

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

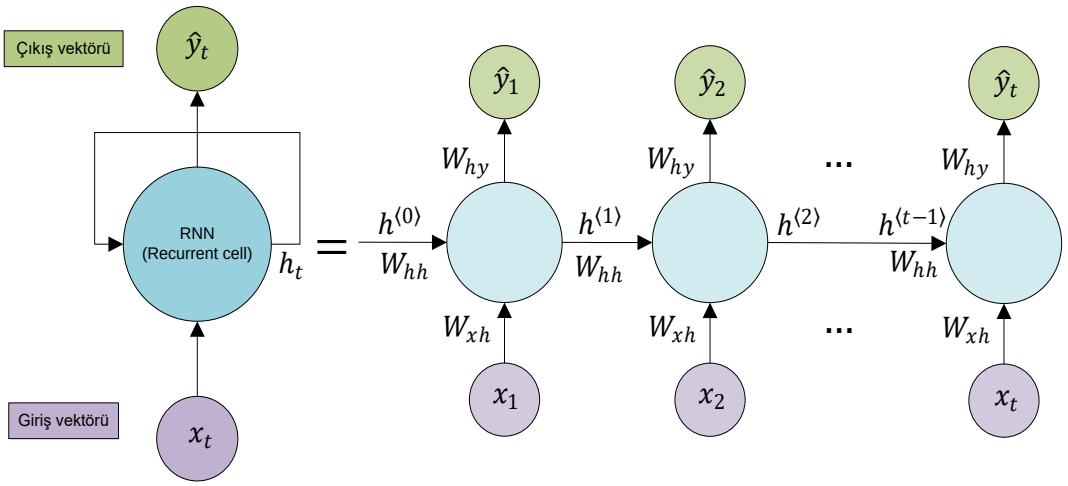
Tekrarlayan sinir ağları yöntemi (RNN) 1990 yılında Elman tarafından geliştirilmiştir. Elman modelini geliştirirken zamanın insan davranışları, sözleri üzerindeki etkisinin önemini vurgulamıştır. Bir andaki bir olay, söz veya olgunun bir sonraki anın girdisi olacağını ve bir sonraki anın çıktısını etkileyeceğini öne sürmüştür (Elman, 1990).

RNN yöntemi yapay sinir ağları yöntemleri içerisinde yer alır. Geri beslemeli bir düzende çalışmaktadır. Sıra bilgisi içeren bir veri dizisinin bir sonraki sıradaki bilgisini tahmin etmek için kullanılan bir yöntemdir. Zaman serisi tahminlerinde de kullanılan bir yöntemdir. Basit olarak çalışma süreci aşağıdaki görselde (Şekil 3-7) sunulmuştur.

Şekil 3-8' da RNN ağ yapısının açılmış bir şekilde görünümü sunulmuştur. Giriş katmanı olarak tanımlanan değerler, sıralı verileri içeren dizilerdir. Örneğin bir numaralı girdi (X_1) 10 adet art arda sıralı verilerden oluşmuş olsun. Bu sıralı veri bilgisi; bir zaman serisi, bir kelimeyi (veya cümleyi) oluşturan harflerin binary kodları, ses dosyası olabilir. Sıralı veri bilgisindeki her bilgi için bir ağırlık (W_{xh}) atanır. Ağırlıklar ve bilgilerin çarpımı sonucu (birleştirme fonksiyonu) değerleri elde edilir. Elde edilen bu değerler aktivasyon fonksiyonundan geçirilerek h^t değerleri elde edilir. Kullanılan aktivasyon fonksiyonu ve ağırlık değerleri sinir hücresi içinde kullanılan zaman serisinin her aşamasında ortak olarak kullanılır.



Şekil 3-7: RNN yöntemi sinir hücresi yapısı.



Şekil 3-8: RNN zaman göre açılımı (Kızrak, 2020).

$$h^{\langle t \rangle} = g\left(W_{hh}h^{\langle t-1 \rangle} + W_{xh}x^{\langle t \rangle} + b_h\right) \quad (3\text{-}16)$$

$$\hat{y}^{\langle t \rangle} = g\left(W_{hy}h^{\langle t \rangle} + b_y\right) \quad (3\text{-}17)$$

- x^t: Giriş vektörü
- $h^{\langle t \rangle}$: t' inci zaman hücre değeri
- $\hat{y}_t$: Çıkış vektörü
- W_{hy}, W_{hh}, W_{xh}: Ağırlık değerleri
- g: Aktivasyon fonksiyonu

Yapay sinir ağları eğitim aşamasında, ağırlık parametrelerinin her biri her geriye yayılım sürecinde (kayıp fonksiyonunun türev alınması ile) güncellenir. Bu güncelleme bir süre sonra ağırlık değişiminin çok küçük olması durumuna yol açar. Bu durumda ağırlık değişimini etkili bir şekilde değişmesini engeller. Ağırlıkların etkili bir şekilde güncellenmemesi kayıp fonksiyonunun bir yerde takılması anlamına gelir ki bu durumda optimizasyon için bir problem olduğu anlamına gelir. Bu problemin literatürdeki adı *kaybolan (vanishing) gradyan problemi* olarak geçmektedir. Aynı şekilde türev (gradyan) değerlerinin çok büyüyerek kullanılamaz hale gelmesi durumunda ise *patlayan (exploding) gradyan problemi* ortaya çıkmaktadır. Bu problemlerin önüne geçmek için LSTM ve GRU yöntemleri geliştirilmiştir.

3.1.1.6. Uzun Kısa Süreli Bellek (Long Short Term Memory-LSTM) ve Kapılı Tekrarlayan Sinir Hücresi (Gated Recurrent Unit-GRU)

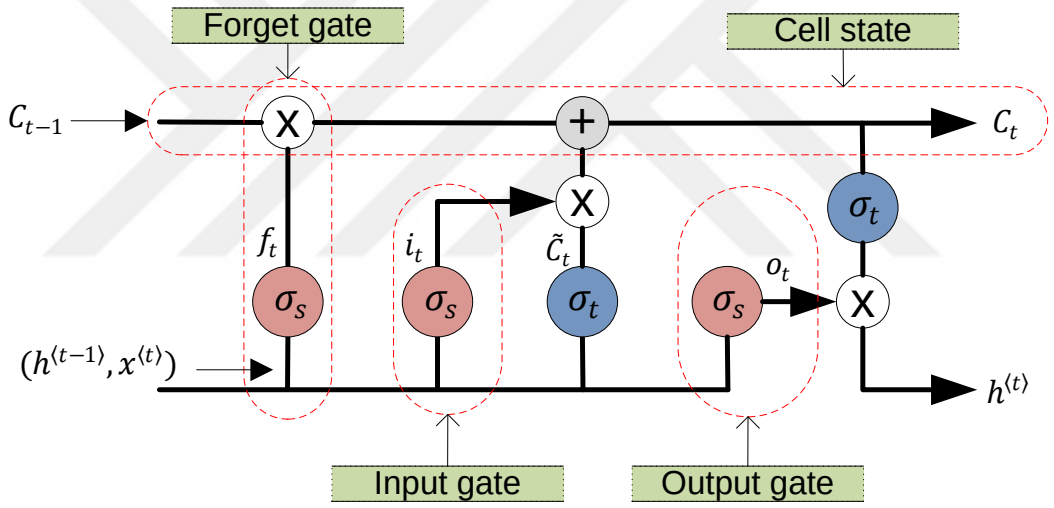
LSTM yöntemi RNN yönteminin özel bir türüdür. RNN yöntemi kısa süreli bir bellek duruma sahip olduğundan dolayı, bu durum bir dezavantaj olarak görülebilmektedir. Örneğin, yeteri uzunlukta bir dizi RNN hücresine girdiğinde en eski zamandaki bilgilerin sona doğru taşınması aşamasında bilgilerin unutulma durumu söz konusudur. Bunun yanında ağın geriye yayılımı aşamasında kaybolan gradyan problemi de ortaya çıkabilmektedir. Bu sebeplerle RNN sinir hücreleri daha uzun dizilerde işledikleri bilgileri unutabilmekte ve daha yakın bilgilerin etkisi daha fazla olduğundan dolayı kısa süreli hafızaya sahip olmaktadır. LSTM ve GRU yöntemleri de kısa süreli hafıza sorununun çözümü için geliştirilmiştir (Phi, 2018).

LSTM yöntemi, 1997 yılında Sepp Hochreiter ve Jürgen Schmidhuber tarafından geliştirilmiştir. LSTM yöntemi RNN gibi bir kontrol çerçevesinde bir hücre akışı gerçekleştirmektedir. Farklı olan ileri besleme yaparken, bilginin ilerletilmesidir. Bir LSTM sinir hücresi; Cell state (Hücre durumu), Input gate (giriş kapısı), Forget gate (Unutma kapısı), Output gate (Çıkış kapısı) gibi mekanizmalardan (kapılardan) oluşmaktadır. Bu kapılar (hesaplamalar) ile, dizideki hangi bilgilerin saklanmasının veya atılmasının (unutulmasının) belirlenmesine karar verilerek öğrenme gerçekleştirilir. Böylelikle tahmin veya sınıflandırma yapmak için önemli bilgilerin daha uzun dizilerde kullanılması sağlanmaktadır.

- Forget gate (Unutma kapısı): Hangi bilgilerin atılması veya saklanması gerektiğine karar verir. Önceki zamandan ($h^{\langle t-1 \rangle}$) gelen bilgi ve mevcut girişten gelen bilgiler sigmoid işlevinden (σ_s) geçirilir. Değerler 0 ile 1 döndürülür. 0'a yakınsa "*unut*", 1'e yakınsa "*sakla*" anlamına gelmektedir. Unutma kapısında f_t değeri elde edilir (Phi, 2018).
- Input gate (Giriş kapısı): Bu kapıda yine önceki zamandan ($h^{\langle t-1 \rangle}$) gelen bilgi ve mevcut girişten gelen bilgiler sigmoid işlevinden (σ_s) geçirilir, i_t değeri elde edilir. Aynı bilgiler (σ_t) fonksiyonundan geçirilerek "*-1, +1*" aralığında bir değer döndürülür ve $\tilde{C}_t$ değeri elde edilir. Böylelikle hangi bilgilerin önemli olduğu belirlenir. σ_s ve σ_t fonksiyonlarından gelen sonuçlar çarpılır ($i_t * \tilde{C}_t$) ve hücre durumuna aktarılır (Phi, 2018).
- Cell state (Hücre durumu): Dizide yer alan bilgilerin dizi boyunca aktarımını sağlar. Hücrenin hafızası olarak düşünülebilir. Hücre durumu boyunca ilerleyen bilgiler,

kapılar aracılığıyla hücre durumuna eklenir veya çıkarılır. Unutma kapısından gelen f_t değeri ile bir önceki zamandan gelen C_{t-1} hücre durumu değeri çarpılır. Burada amaç, unutma kapısından gelen değer sıfıra yakınsa bir önceki zamandan gelen değerin unutulması sağlanır. Aynı şekilde giriş kapısından gelen nihai değer ile de önceki çarpılan değer sonucu toplanır (3-21) (Phi, 2018).

- Output gate (Çıkış kapısı): Son işlem bu kapıda gerçekleştirilir. Bu kapı ile $h^{\langle t \rangle}$ değeri elde edilir. Önceki zamandan ($h^{\langle t-1 \rangle}$) gelen bilgi ve mevcut girişten gelen bilgiler sigmoid işlevinden (σ_s) geçirilir. Elde edilen değer ile hücre durumundan gelen (C_t) ve σ_t fonksiyonundan geçirilerek elde edilen değer çarpılır. Nihai olarak (3-23) numaralı denklemde yer alan $h^{\langle t \rangle}$ değeri elde edilir. Sonraki bir zaman için kullanılmak amacıyla yeni hücre durumu (C_t) ve $h^{\langle t \rangle}$ değerleri taşınır (Phi, 2018).



Şekil 3-9: LSTM yöntemi hücre içi hesaplama süreci (Olah, 2015).

Şekil 3-9' da ayrıntılı şekilde LSTM hücresine ait hesaplama süreci sunulmuştur. Aşağıda görüleceği üzere LSTM yöntemine ait denklemler sunulmuştur (Olah, 2015).

$$f_t = \sigma_s\left(W_f . \left[h^{\langle t-1 \rangle}, x^{\langle t \rangle}\right] + b_f\right) \quad (3\text{-}18)$$

$$i_t = \sigma_s\left(W_i . \left[h^{\langle t-1 \rangle}, x^{\langle t \rangle}\right] + b_i\right) \quad (3\text{-}19)$$

$$\tilde{C}_t = \sigma_t\left(W_C . \left[h^{\langle t-1 \rangle}, x^{\langle t \rangle}\right] + b_C\right) \quad (3\text{-}20)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (3\text{-}21)$$

$$o_t = \sigma_s\left(W_o.\left[h^{\langle t-1\rangle}, x^{\langle t\rangle}\right] + b_o\right) \quad (3\text{-}22)$$

$$h^{\langle t\rangle} = o_t * \sigma_t(C_t) \quad (3\text{-}23)$$

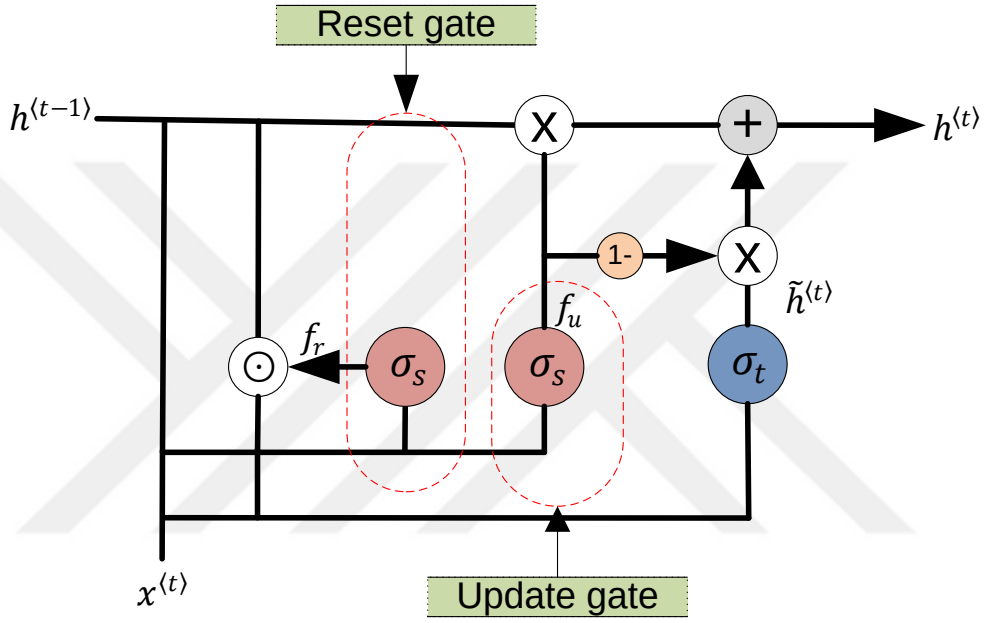
- f_t: t anındaki unutma kapısı (Forget gate) değeri
- i_t: t anındaki giriş kapısı (Input gate) değeri
- C_t: t anındaki hücre durumu (Cell state) değeri
- o_t: t anındaki çıkış kapısı (Output gate) değeri
- σ_t: Tanh aktivasyon fonksiyonu
- σ_s: Sigmoid aktivasyon fonksiyonu
- $h^{\langle t-1\rangle}$: t-1 anındaki LSTM hücresi değeri
- $h^{\langle t\rangle}$: t anındaki LSTM hücresi değeri
- $W_{f,i,C,o}$: Unutma, giriş, hücre durumu ve çıkış kapılarındaki ağırlıklar

GRU yöntemi 2014 yılında Kyunghyun Cho vd. tarafından geliştirilmiştir. LSTM yöntemine göre daha az parametre ile çalışmaktadır. GRU yönteminin hesaplama adımları LSTM yöntemine benzerlik göstermektedir. GRU yönteminde Reset (sıfırlama) ve Update (güncelleme) olmak üzere iki kapı (mekanizma) yer almaktadır (Cho, et al., 2014).

- Reset gate (Sıfırlama kapısı): Sıfırlama kapısı dizideki geçmiş bilgilerin ne kadarının unutulacağına karar vermek için kullanılır. r_t değerinin hesaplaması ile süreç başlar. $h^{\langle t-1\rangle}$ ve $x^{\langle t\rangle}$ değerleri ağırlıkları ile çarpılır ve aktivasyon fonksiyonundan geçerek r_t değeri elde edilir (3-24) (Kostadinov, 2017).
- Update gate (Güncelleme kapısı): Güncelleme kapısı, dizideki geçmiş bilginin (önceki zaman adımlarından) ne kadarının geleceğe aktarılması gerektiğini belirlemesine yardımcı olur. u_t değerinin hesaplaması ile süreç başlar. $h^{\langle t-1\rangle}$ ve $x^{\langle t\rangle}$ değerleri ağırlıkları ile çarpılır ve aktivasyon fonksiyonundan geçerek u_t değeri elde edilir (3-25) (Kostadinov, 2017).

Yukarıdaki kapılar dışında hücre içi gerçekleşen başka hesaplamalar da mevcuttur. Sıfırlama ve güncelleme kapılarında işlemler gerçekleştikten sonra hücre için güncel hafıza içeriği kontrol edilir. Bu işlemde $\tilde{h}^{\langle t\rangle}$ hesaplanması ile devam eder. $\tilde{h}^{\langle t\rangle}$ değerinin hesaplanması için

$x^{\langle t \rangle}$ değeri ağırlık değeri ile çarpılır, $h^{\langle t-1 \rangle}$ değeri ağırlık değeri ile çarpıldıktan sonra r_t değeri ile hadamard metodu uygulanır, toplama işlemi ile $\tilde{h}^{\langle t \rangle}$ değeri ortaya çıkar. u_t değeri 1 (bir) değerinden çıkarılarak $\tilde{h}^{\langle t \rangle}$ değeri ile çarpılır, u_t değeri $h^{\langle t-1 \rangle}$ değeri ile çarpılır, bulunan değerler toplanarak son aşamada $h^{\langle t \rangle}$ değeri (3-27) ortaya çıkar. Bir sonraki zamanda kullanılmak üzere iletilir. Şekil 3-10' da GRU hücresine ait detaylı bir hesaplama süreci sunulmuştur.



Şekil 3-10: GRU yöntemi hücre içi hesaplama süreci (Phi, 2018).

GRU yöntemine ait hücre içi denklemler aşağıda sunulmuştur (Kostadinov, 2017).

$$r_t = \sigma_s\left(W_r.\left[h^{\langle t-1 \rangle}, x^{\langle t \rangle}\right] + b_r\right) \tag{3-24}$$

$$u_t = \sigma_s\left(W_u.\left[h^{\langle t-1 \rangle}, x^{\langle t \rangle}\right] + b_u\right) \tag{3-25}$$

$$\tilde{h}^{\langle t \rangle} = \sigma_t(\widetilde{W}\left[\left(r_t \odot h^{\langle t-1 \rangle}\right), x^{\langle t \rangle}\right] + b_{\tilde{h}}) \tag{3-26}$$

$$h^{\langle t \rangle} = (1 - u_t) * \tilde{h}^{\langle t \rangle} + u_t * h^{\langle t-1 \rangle} + b_h) \tag{3-27}$$

- r_t: t anındaki sıfırlama kapısı (reset gate) değeri
- u_t: t anındaki güncelleme kapısı (update gate) değeri

- $h^{\langle t-1 \rangle}$: t-1 anındaki GRU hücresi değeri
- $h^{\langle t \rangle}$: t anındaki GRU hücresi değeri
- $\odot$: Hadamard çarpımı
- $W_{r,u}$, $\widetilde{W}$: Sıfırlama, güncelleme kapılarındaki ağırlıklar

3.2. ZAMAN SERİSİ ANALİZİ

Bir zaman serisi belli bir zaman aralığında (günlük, haftalık, vs.) belli bir sıra dahilinde gözlemlenen veriler bütünüdür. Zaman serisi bu yönüyle bir veri dizisi olarak anlaşılmaktadır. Belli bir zaman dahilinde gözlenen bu verileri analiz etmek için yöntemler geliştirilmiştir. Bu yöntemler ekonomiden mühendisliğe kadar çeşitli alanlarda kullanılmaktadır. İstatistikte önemli bir alan oluşturmaktadır (Chatfield, 2005).

Bir zaman serisi sürekli gözlenen verilerden oluşmalıdır. Ölçülen gözlemlerin ayrık bir veri kümesi oluşturması durumunda bile, herhangi bir zaman süresince ölçülebilmesi, onun zaman serisi olmasına engel değildir. Ayrık veriler, gözlemlerin eşit aralıklardaki belli zaman dilimlerinde gözlenen verilerdir. Gözlenen veriler sürekli bir değişken olduğunda ayrık terimi kullanılır. Zaman serisi analizleri de ayrık veriler mantığı çerçevesinde geliştirilmiştir. Yani saatlik, günlük, haftalık, aylık gibi zaman aralıklarında ölçümü gerçekleştirilen değerler bir zaman serisi olarak düşünülür. Örnek olarak, günlük ortalama hava sıcaklıkları, saatlik üretim adedi, aylık satış miktarı gibi veriler gösterilebilir.

Zaman serisi analizleri çeşitli amaçlar doğrultusunda kullanılabilmektedir. Bu amaçlar aşağıda sunulmuştur (Chatfield, 2005):

- *Betimleme, Tasvir:* Bu amaç doğrultusunda bir zaman serisinin zaman grafiği çizilir, zamana karşı gözlemlerin hareketi incelenir, uç değerler tespiti yapılabilir ve seriye ait basit tanımlayıcı değişkenler elde edilir. Zaman serisi grafikte incelenince mevsimsel etkiler gözlemlenebilir. Serinin zaman süresince eğilimi tespit edilebilir.
- *Açıklama:* Bir zaman süresince iki veya daha fazla değişken üzerinde gözlemler yapıldığında, bir serinin değişkenliği başka bir serinin değişmesi üzerindeki açıklaması mümkün olabilmektedir. Bu amaç genellikle regresyon analizi üzerinde karşımıza çıkmaktadır. Genellikle de doğrusal (lineer) bir seri olduğunda kullanışlı olabilmektedir.

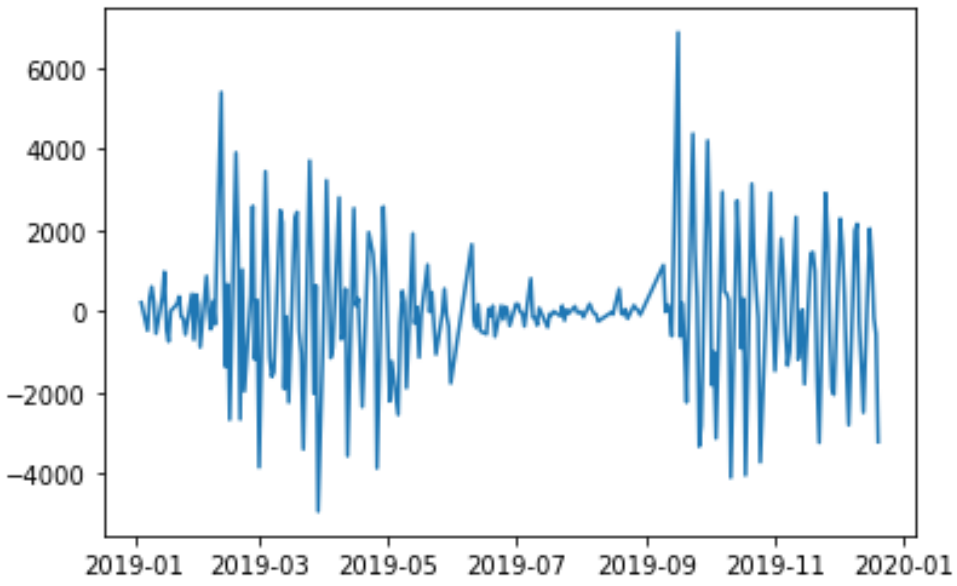
- *Tahmin:* Bir zaman serisinin geleceğe ait tahmin işlemlerinde kullanılması amaçlanabilmektedir. Genellikle de geliştirilen zaman serisi analizleri (yöntemleri) bu amaçla geliştirilmiştir. Zaman serisi kullanılarak bir model geliştirilir. Bu model doğrultusunda geleceğe yönelik tahmin yapılır.
- *Kontrol:* Bir zaman serisi boyunca gözlenen veriler bir üretim hattının doğruluğunu (hata oranı) ölçebilmek için gözlenen veriler olabilmektedir. Bu veriler örneğin, üretim hattı boyunca saatlik ortalama sıcaklık verisi olabileceği gibi, hat boyunca gerçekleşen hata sayısını içerebilmektedir. Kontrol amaçlı kullanılan zaman serisi genellikle kalite kontrol faaliyetleri için kullanılmaktadır.

İstatistiksel teorilerin çoğu, gözlemlerin bağımsızlığı yani rassallığı üzerinde temellendirilerek geliştirilmiştir. Zaman serisi analizleri ise, bir zaman süresi dahilinde sıralı bir şekilde gözlemlenen verilerin birbirinden bağımsız olamayacağı ve zaman kavramının hesaba katılması gerektiğini ifade etmektedir. Ardışık gözlemler bağımlı olduğundan dolayı, gelecekteki değerlerin geçmiş gözlemlerden faydalanılarak tahmin edilebileceği öne sürülmektedir. Çoğu zaman serisi, kısmen geçmiş değerlerle geleceğe ait tahmin yapılması ve belirsizlik içerdiğinden dolayı stokastiktir (rastlantısaldır) (Chatfield, 2005).

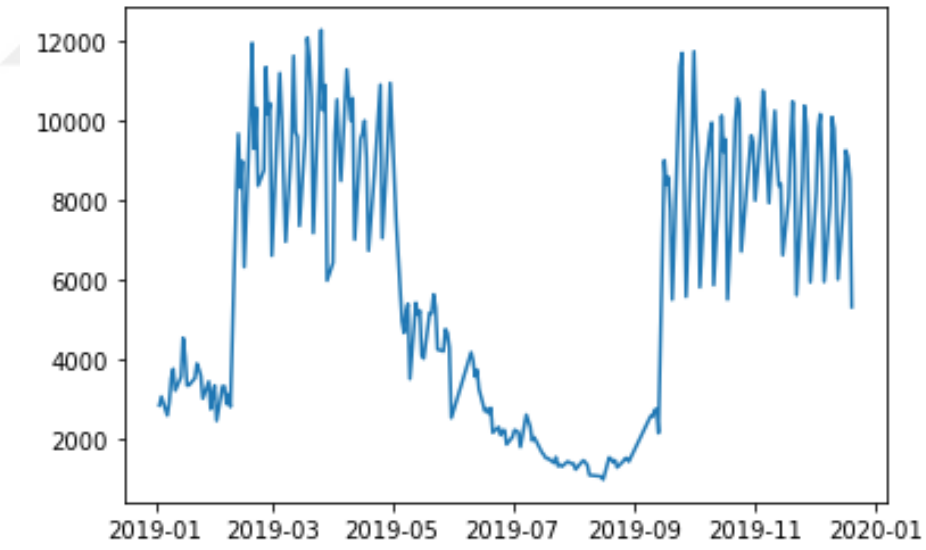
Zaman serisi analizi için geliştirilen belli başlı bazı teknikler mevcuttur. Tek değişkenli zaman serisi için gelecek tahmini yapılabildiği gibi, çok değişkenli bir zaman serisi için de gelecek tahmini yapılabilmektedir. Bu çalışma kapsamında tek değişkenli bir zaman serisi tekniği kullanılmıştır. Tek değişkenli zaman serisi demek, eşit zaman aralıklarıyla sırayla kaydedilen tek bir türdeki gözlemlerden oluşan bir zaman serisini ifade etmektedir.

Zaman serisi için geliştirilen tekniklerde bir model oluşturmak için temel bir varsayım mevcuttur. Zaman serisi durağan (stationary) olmalıdır. Yani zaman serisi içindeki herhangi bir zaman periyodunda ortalama ve varyans sabit olmalıdır. Örnek bir durağan zaman serisi grafiği aşağıdaki Şekil 3-11' de yer alan grafikte sunulmuştur. Durağan olmayan bir grafik ise Şekil 3-12' de sunulmuştur. Şekil 3-11 ve Şekil 3-12' de sunulan grafiklerde kullanılan veriler halihazırda bu çalışma kapsamında uygulama başlığı altında kullanılan verilerdir. Durağan zaman serisi grafiklerinde belli bir zaman periyodunda değişmeyen, sabit bir şekilde görülebilen bir ortalama ve varyans görülmektedir. Durağan olmayan zaman serilerindeki grafiklerde ise mevsimsel olarak ve belli bir zaman periyodunda ortalama ve varyans

değişkenlik göstermektedir. Bir zaman serisinin durağan olup olmadığı Augmented Dickey Fuller (ADF) testi kullanılarak test edilebilmektedir.



Şekil 3-11: Durağan (stationary) bir zaman serisi grafiği.



Şekil 3-12: Durağan olmayan (non-stationary) bir zaman serisi grafiği.

Durağan olmayan zaman serilerini durağan hale getirmek için fark alma (differencing) tekniği uygulanabilmektedir. Seride t anındaki y_t verisinden y_{t-1} verisi çıkarılır, seri boyunca bu işlem devam eder. Ortaya çıkan fark değerleri ile durağan olmayan bir zaman serisi durağan hale

getirilebilmektedir. Bu çalışma kapsamında zaman serisi gelecek tahmini tekniklerinden olan ARIMA yöntemi kullanılmıştır.

3.2.1. ARIMA (p, d, q)

ARIMA (Auto-Regressive Integrated Moving Average) yöntemi zaman serilerini modelleme ve tahmin etmek için kullanılan yaygın yöntemlerden biridir. George Box ve Gwilym Jenkins tarafından geliştirilmiştir. Bu modele Box-Jenkins (BJ) yöntemi de denilmektedir. ARIMA modeli tek değişkenli verileri açıklamaya dayanmaktadır (Montgomery, et al., 2008). ARIMA tekniğinde kullanılan p değeri zaman serisindeki oto regresif değerini yani gecikme değerini (lag), d değeri fark alma değerini, q değeri ise hareketli ortalama modelinin (moving average) derecesini temsil etmektedir. ARIMA tekniği AR (p) (Auto-Regressive), I (d) (Differencing), MA (q) (Moving Average) yöntemlerinin bileşimi ile geliştirilmiştir.

- AR (Auto-Regressive); mevcut bağımlı değişken ile gecikmeli zaman periyotlarında olan bağımlı değişken arasındaki ilişkiyi ifade etmektedir.
- I (Integrated); veriyi durağan hale getirmek için *t-1* anındaki değerin *t* anındaki değerinden çıkarılarak verilerin dönüştürülme işlemini ifade etmektedir.
- MA (Moving Average); Rassal olan bir hata değerinin geçmiş anlardaki değerleri ile arasındaki bağımlılığı ifade etmektedir.

Durağan bir zaman serisi için aşağıda görüleceği üzere hesaplama formülleri sunulmuştur (Ilie, et al., 2020).

AR (p) için;

$$Y_t = \alpha + \Phi_1 Y_{t-1} + \Phi_2 Y_{t-2} + \cdots + \Phi_p Y_{t-p} + \varepsilon_t \tag{3-28}$$

$$Y_t = \mu + \sum_{i=1}^{p} (\Phi_i Y_{t-i}) + \omega_t \tag{3-29}$$

MA (q) için;

$$Y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \tag{3-30}$$

$$Y_t = \mu + \sum_{j=1}^{q} \left(\theta_j \omega_{t-j}\right) + \omega_t \quad (3\text{-}31)$$

Durağan bir zaman serisi için ortaya çıkan nihai hesaplamalar ARMA (p, d=0, q) olarak adlandırılmaktadır. (3-32) ve (3-33) numaralı denklemlerde görüleceği üzere ifade edilebilmektedir.

$$Y_t = \delta + \Phi_1 Y_{t-1} + \cdots + \Phi_p Y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} \quad (3\text{-}32)$$

$$Y_t = \mu + \sum_{i=1}^{p} (\Phi_i Y_{t-i}) + \sum_{j=1}^{q} \left(\theta_j \omega_{t-j}\right) + \omega_t \quad (3\text{-}33)$$

Y_t: t anındaki tahmin değeri

$\delta = \alpha$: Sabit değer

$\theta \; ve \; \Phi$: Çarpan katsayılar

$\varepsilon_t \; ve \; \omega_t$: t anındaki rastgele hata değerleri veya beyaz gürültü (White noise)[9]

μ: Zaman serisinin ortalaması

Durağan olmayan bir zaman serisi için bir üst başlıkta fark alma tekniğinden bahsedilmiştir. Durağan olmayan bir zaman serisi, durağan hale getirildikten sonra kullanılan yöntem ARIMA (p, d, q) olmaktadır. Fark alma tekniği aşağıdaki gibi (3-34) ifade edilmiştir.

d > 0 için;

$$\Delta(Y_t) = Y_t - Y_{t-1} = Y_t' \quad (3\text{-}34)$$

Y_t': Fark alma işleminden sonra kullanılan veriler ile t anındaki tahmin değeri

ARIMA yöntemi Box-Jenkins yöntemi olarak da anılmaktadır. Box-Jenkins yöntemine göre ARIMA modeli kurulması için aşağıdaki adımlar takip edilebilmektedir (Erica, 2021).

i. Verileri durağanlık varsayımını karşılayacak şekilde dönüştür.
ii. p, d ve q değerleri için ilk değerleri belirle.
iii. Önerilen p, d ve q değerleri ile tahmin değerlerini belirle.

[9] Ortalaması sıfır, normal dağılan ve otokorelasyon içermeyen rastsal terime verilen addır (Montgomery, et al., 2008).

iv. Önerilen p, d ve q değerleri ile model performansını değerlendir

v. En iyi model performansını elde etmek için ii-iv arasındaki adımları gerektiği kadar tekrarla.

3.2.1.1. Augmented Dickey Fuller testi (Durağanlık)

Bu çalışmada durağanlık testi için *Augmented Dickey Fuller* (ADF) testi kullanılmıştır. Buna ilişkin durağanlık testini aşağıdaki şekilde tanımlayabiliriz. Bu testte sabitsiz, sabitli ve trendli bir süreç izlenmektedir. Buna göre seri trendli bir süreçte durağan hâle gelmişse, diğer süreçler izlenmeksizin bu değer esas alınmaktadır. Seri durağan hâle gelmemişse, sabit terimli; bunda da durağanlık sağlanamamışsa, sabit terimsiz sınama yapılır ve bu süreç sonucunda seriyi durağan hâle getiren değer baz alınır. ADF testinde aşağıdaki eşitlikler yer almaktadır (Kurt & Karayılmazlar, 2019).

$$\Delta Y_t = \delta Y_{t-1} + \sum_{j=1}^{p} \delta_i \Delta Y_{t-j} + \varepsilon_t \quad (3\text{-}35)$$

$$\Delta Y_t = \mu + \delta Y_{t-1} + \sum_{j=1}^{p} \delta_i \Delta Y_{t-j} + \varepsilon_t \quad (3\text{-}36)$$

$$\Delta Y_t = \mu + \beta t + \delta Y_{t-1} + \sum_{j=1}^{p} \delta_i \Delta Y_{t-j} + \varepsilon_t \quad (3\text{-}37)$$

Δ: Fark alma işlemcisi,

Y_t: t. Dönemdeki zaman serisi değeri,

μ: Sabit terim,

β_t: Zaman trendi,

ε_t: Hata terimi

p: gecikme değeri

ADF testi ile birim kök testi yapılır. Buna ilişkin geliştirilen hipotezler aşağıdaki gibidir.

- $H_0: \delta = 0$ "*Zaman serisinin birim kökü (unit root) vardır ve durağan değildir. Zamana bağlı bir yapıya sahiptir*",
- $H_0: \delta < 0$ "Zaman serisinin birim kökü yoktur, yani durağandır. Zamana bağlı bir yapıya sahip değildir".

Bulunan test istatistiği kritik değerlerden büyükse H_0 hipotezi reddedilemez. Aynı şekilde bulunan p değeri ele alınan güven seviyesinden büyükse H_0 reddedilemez. Tersi durumda H_0 reddedilir.

3.3. PERFORMANS GÖSTERGELERİ

Çalışma kapsamında kullanılan performans göstergelerine ait hesaplama denklemleri aşağıda sunulmuştur.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{3-38}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \tag{3-39}$$

$$MAE = \frac{\sum_{i=1}^{n}\left|\hat{Y}_i - Y_i\right|}{n} \tag{3-40}$$

$$MAPE = \sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| . \frac{100}{n} \tag{3-41}$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \qquad SS_{res} = \sum_{i}\left(Y_i - \hat{Y}_i\right)^2 \qquad SS_{tot} = \sum_{i}(Y_i - \bar{Y})^2 \tag{3-42}$$

n: örnek sayısı

Y_i: i' inci örneğin bağımlı değişken gerçek değeri

$\hat{Y}_i$: i' inci örneğin bağımlı değişken tahmin değeri

$\bar{Y}$: Bağımlı değişkenin gerçek değerlerinin ortalaması

SS_{res}: Hata karelerinin toplamı (The Sum of Squares of Residuals)

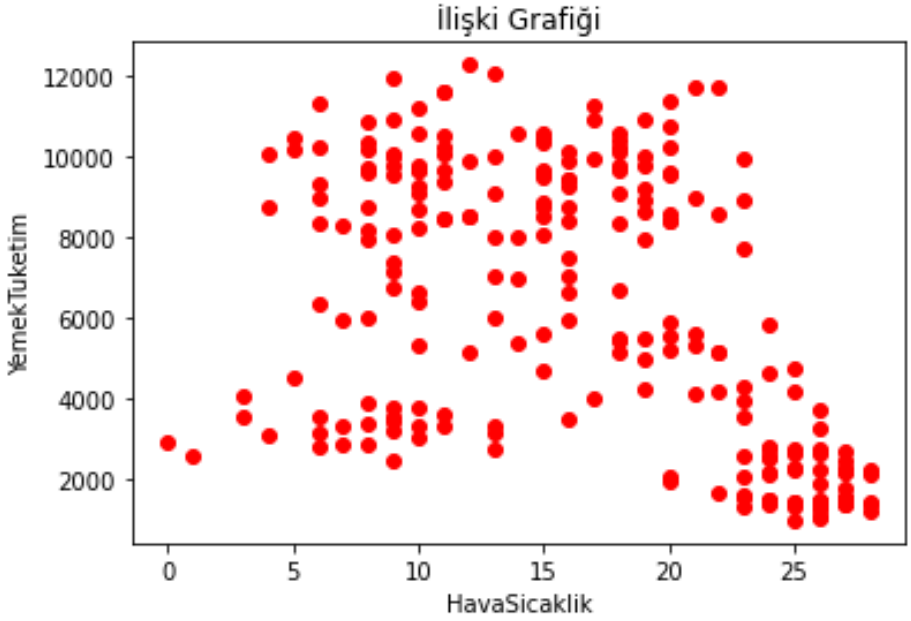
SS_{tot}: Bağımlı değişkenin gerçek değerlerinin ortalamadan farklarının karelerinin toplamı

3.4. UYGULAMA

Bu çalışma kapsamında ROA, RNN, LSTM, GRU ve ARIMA yöntemleri kullanılmıştır. Yöntemlerin uygulama adımları ve sonuçları aşağıda başlıklar halinde sunulmuştur. Yöntemler çalıştırılırken Python 3.8 sürümünü kullanan Spyder 5.0.0 geliştirme ortamı (IDE) kullanılmıştır. Yöntemlerin performansları değerlendirilirken MAPE, MAE, MSE, RMSE ve Korelasyon katsayısı (R^2) performans göstergeleri kullanılmıştır. Çalışma kapsamında kullanılan veri setleri aşağıda sunulmuştur.

- Genel veri seti (Ver Seti 1) (EK 1)
    - Veri Seti 1' in içerisinde; Ramazan ayı günleri, Final sınavı günleri, BUT sınavı günleri, Sömestr tatili günleri, Yaz tatili günlerinin **yer almadığı** (Normal Günler) veri seti (Veri Seti 2)
        - Veri seti 2 içerisinde **sadece** "**Pazartesi**" günleri (Veri Seti 3)
        - Veri seti 2 içerisinde **sadece** "**Salı**" günleri (Veri Seti 4)
        - Veri seti 2 içerisinde **sadece** "**Çarşamba**" günleri (Veri Seti 5)
        - Veri seti 2 içerisinde **sadece** "**Perşembe**" günleri (Veri Seti 6)
        - Veri seti 2 içerisinde **sadece** "**Cuma**" günleri (Veri Seti 7)
    - **Sadece** Ramazan ayı günlerini içeren veri seti (Veri Seti 8)
    - **Sadece** BUT (bütünleme) sınavı günlerini içeren veri seti (Veri Seti 9)
- *BirOncekiGun* ve *GecenHaftaAyniGun* yemek tüketim sayılarını içeren Genel veri seti (Veri Seti 10) (EK 2)
    - Veri Seti 10' un içerisinde; Ramazan günleri, Final sınavı günleri, BUT sınavı günleri, Sömestr tatili günleri ve Yaz tatili günlerinin **yer almadığı** veri seti (Veri Seti 11)
        - Veri seti 11 içerisinde **sadece** "**Pazartesi**" günleri (Veri Seti 12)
        - Veri seti 11 içerisinde **sadece** "**Salı**" günleri (Veri Seti 13)
        - Veri seti 11 içerisinde **sadece** "**Çarşamba**" günleri (Veri Seti 14)
        - Veri seti 11 içerisinde **sadece** "**Perşembe**" günleri (Veri Seti 15)
        - Veri seti 11 içerisinde **sadece** "**Cuma**" günleri (Veri Seti 16)
    - **Sadece** Ramazan ayı günlerini içeren veri seti (Veri Seti 17)
    - **Sadece** BUT (bütünleme) sınavı günlerini içeren veri seti (Veri Seti 18)

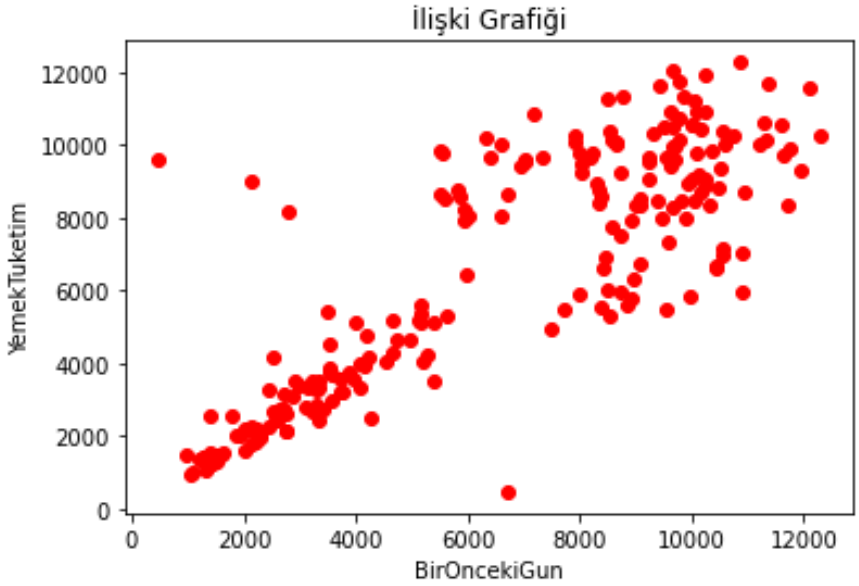
Bu çalışma kapsamında makine öğrenmesi yöntemlerinin kullanılmasına karar verilirken bağımsız değişkene göre bağımlı değişkenin değişimi incelenmiştir. Söz konusu değişim için aşağıdaki grafiklerde ilişkileri sunulmuştur. Grafiklerde sadece sayısal tipteki bağımsız değişkenler görüntülenmiştir.



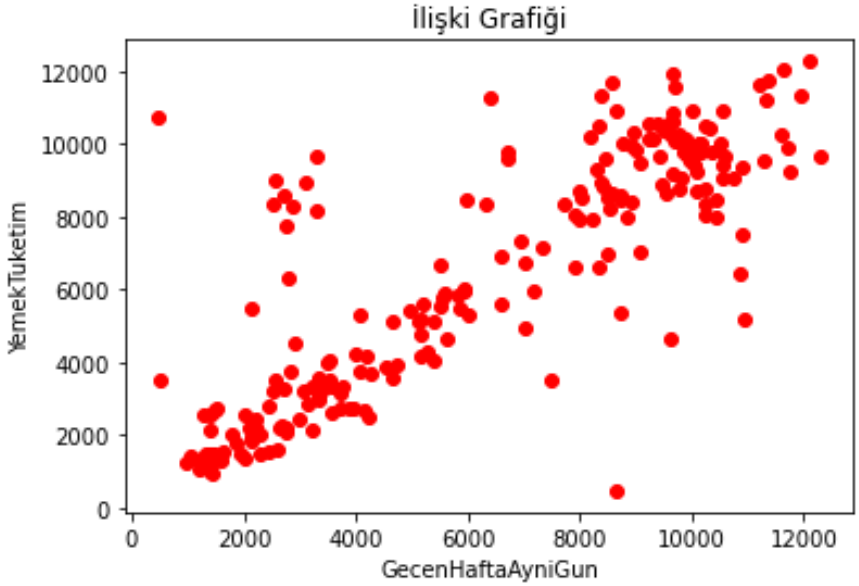
Şekil 3-13: HavaSicaklik-YemekTuketim ilişki grafiği.

Şekil 3-13' de görüleceği üzere hava sıcaklık değişkeni ile yemek tüketim değişkeni arasında doğrusal bir ilişki **bulunmamaktadır**. Dolayısıyla doğrusal modelleme teknikleri kullanılamayacaktır. Doğrusal olmayan ilişkiler için makine öğrenmesi yöntemleri daha doğru sonuçlar verebilmektedir. Bu amaçla makine öğrenmesi yöntemleri kullanılmıştır.

Veri Seti 10 içerisinde yer alan, iki sayısal bağımsız değişken olan *BirOncekiGun* ve *GecenHaftaAyniGun* değişkenlerinin bağımlı değişken ile olan ilişkileri Şekil 3-14 ve Şekil 3-15' de yer alan grafiklerde sunulmuştur. Grafiklerde görüleceği üzere tam olarak bir doğrusal ilişkiden bahsedebilmek mümkün değildir. Dolayısıyla buradan da doğrusal olmayan tekniklerin kullanılmasının daha doğru sonuçlar verebileceği anlaşılmaktadır.



Şekil 3-14: BirOncekiGun-YemekTuketim ilişki grafiği.



Şekil 3-15: GecenHaftaAyniGun-YemekTuketim ilişki grafiği.

3.4.1. Rassal Orman Algoritması Yöntemi ile Tahminleme

Rassal Orman Algoritması (ROA), makine öğrenmesi yöntemlerinden biridir ve topluluk öğrenmesi (ensemble learning) yöntemleri içerisinde yer alır. Topluluk öğrenimi, birden fazla modeli birleştirerek makine öğrenimi sonuçlarının daha iyi sonuç vermesini sağlar. Topluluk öğrenimi, tek bir modele kıyasla daha iyi tahmin performansı üretilmesine olanak tanır. Rassal orman algoritması hem sınıflandırma hem de regresyon tahmin problemlerinde

uygulanabilmektedir. Bu çalışma kapsamında regresyon tahmin türü kullanılmıştır. Kullanılan veri setleri ve veri setlerine ilişkin eğitim ve test oranları ile örnek sayıları aşağıdaki tabloda sunulmuştur (Tablo 3.3).

Tablo 3.3: ROA Veri setleri tablosu.

Veri seti	Örnek Sayısı	Test Oranı	Eğitim Örnek Sayısı	Test Örnek Sayısı
Veri seti 1	233	10%	210	23
Veri seti 2	131	10%	117	14
Veri seti 3	26	20%	20	6
Veri seti 4	25	20%	20	5
Veri seti 5	26	20%	20	6
Veri seti 6	27	20%	21	6
Veri seti 7	27	20%	21	6
Veri seti 8	20	25%	15	5
Veri seti 9	20	25%	15	5
Veri seti 10	230	10%	207	23
Veri seti 11	130	10%	117	13
Veri seti 12	26	20%	20	6
Veri seti 13	24	20%	19	5
Veri seti 14	26	20%	20	6
Veri seti 15	27	20%	21	6
Veri seti 16	27	20%	21	6
Veri seti 17	20	25%	15	5
Veri seti 18	20	25%	15	5

Yöntemi uygulamak için aşağıdaki adımlar uygulanmıştır.

- Veri setleri içerisinde yer alan kategorik değerler One-Hot Encoding yöntemi ile ikili (binary) olarak düzenlenip sayısal bir hale getirilmiştir.
- Veri setlerine 0,1 aralığında normalizasyon işlemi uygulanmıştır.
- Performans kıyaslamalarını ölçmek için model çalıştırıldıktan sonra normalize değerler gerçek değerlere dönüştürülmüştür.
- Eğitim veri setlerine hiperparametre optimizasyonu uygulanmıştır.
- Bulunan en iyi hiperparametreler ile optimizasyon sonuçlandırılmıştır ve bu sonuç en iyi model olarak değerlendirilmiştir.

- Bulunan en iyi model test değerleri ile tahmin edilmiştir ve gerçek değerler ile tahmin değerleri kıyaslanmıştır.

Rassal Orman Algoritması için Python içinde oluşturulan açık kaynak kodlu "*sklearn.ensemble*" kütüphanesinde bulunan "*RandomForestRegressor*" sınıfı (class) kullanılmıştır. Rassal Orman Algoritmasında kullanılan hiperparametrelere (model girdilerine) göre hedef değişken farklı olarak sonuçlanabilmektedir. Bunun için algoritmada en iyi sonucu veren optimizasyonu elde etmek için program içerisinde bir döngü (loop) tasarlanmıştır. Bu döngü her veri setine uygulanmıştır. Döngü aşağıda sunulmuştur.

- Döngü içerisinde ağaç sayısını temsil eden; [1000, 10000] aralığında +500 olarak artan değerler ile bir dizi tanımlanmıştır. Dizi içerisindeki ağaç sayısına göre her seferinde yeni bir model çalıştırılmıştır. Toplam ağaç sayısı 19' dur.
- Döngünün çalışması esnasında veri seti rassal bir şekilde eğitim ve test olarak yeniden ayrılmıştır. Ardından yeni model oluşturulmuş ve sonuçlar, performans değerleri ve model kaydedilmiştir.
- Döngü bittikten sonra genel olarak en iyi model seçiminde MAPE değeri baz alınmıştır.
- Algoritma içindeki diğer hiperparametreler program içerisindeki varsayılan (default) değerleriyle kullanılmıştır.

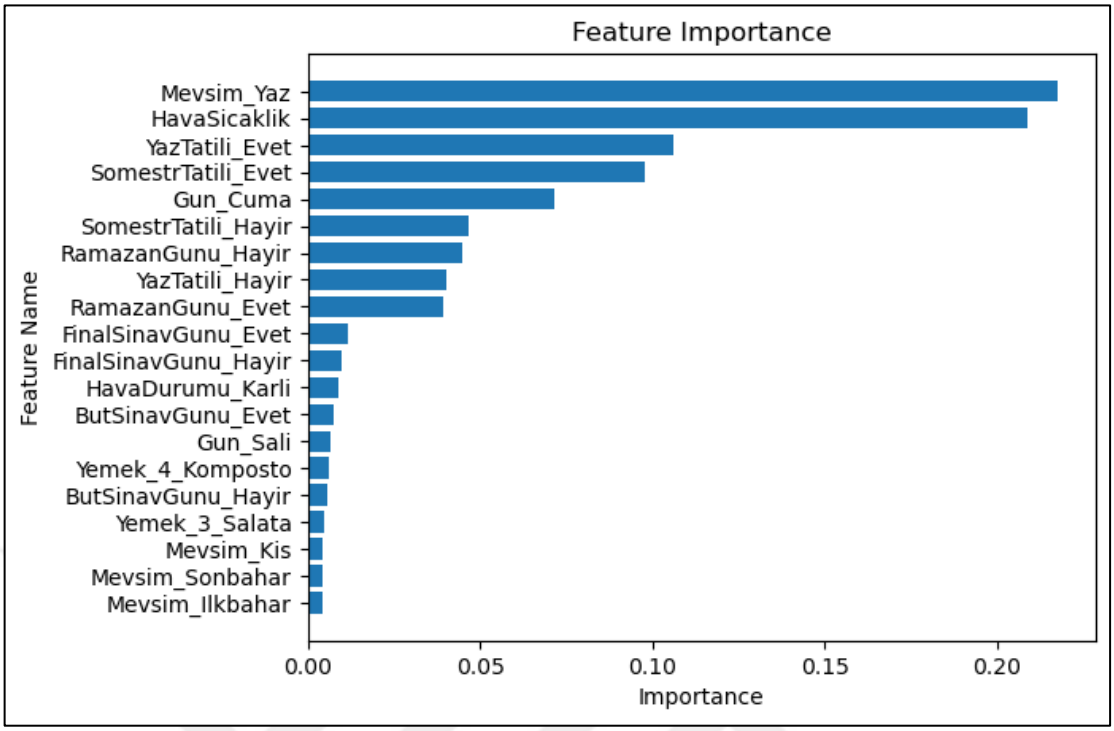
Bu değerler ışığında 18 veri seti ile, her veri seti için 19 farklı model oluşturulmuştur. Her veri setine ait en iyi modeli seçerken MAPE değeri kullanılmıştır. Program içerisinde tasarlanan döngü çalıştırılmış ve optimizasyona ait sonuçlar elde edilmiştir. Her veri setine ait olan sonuçlar aşağıdaki tabloda sunulmuştur.

Tablo 3.4: ROA Veri setleri sonuçları.

Rassal Orman Algoritması Yöntemi						
Veri setleri	**Ağaç sayısı**	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
Veri seti 1	9.500	806	648.973	608	0,902	0,090
Veri seti 2	6.000	742	550.160	637	0,745	0,071
Veri seti 3	10.000	728	529.701	644	-0,475	0,071
Veri seti 4	9.500	599	359.357	536	-0,993	0,054
Veri seti 5	5.500	850	722.992	571	-0,189	0,053
Veri seti 6	6.000	802	642.685	649	-0,071	0,069
Veri seti 7	4.000	793	628.110	511	-0,059	0,068
Veri seti 8	9.500	162	26.083	150	0,735	0,028
Veri seti 9	2.500	195	38.201	183	0,853	0,048
Veri seti 10	4.500	532	283.532	390	0,972	0,068
Veri seti 11	4.000	999	998.030	657	0,680	0,063
Veri seti 12	8.000	807	651.627	584	-0,168	0,062
Veri seti 13	8.500	291	84.814	216	0,607	0,020
Veri seti 14	2.500	271	73.308	209	-1,719	0,021
Veri seti 15	3.000	490	240.466	397	0,616	0,041
Veri seti 16	3.500	532	283.167	313	-0,580	0,042
Veri seti 17	1.000	285	81.305	202	0,277	0,043
Veri seti 18	10.000	196	39.010	188	0,698	0,054

3.4.1.1. Öznitelik önem dereceleri (Feature importance)

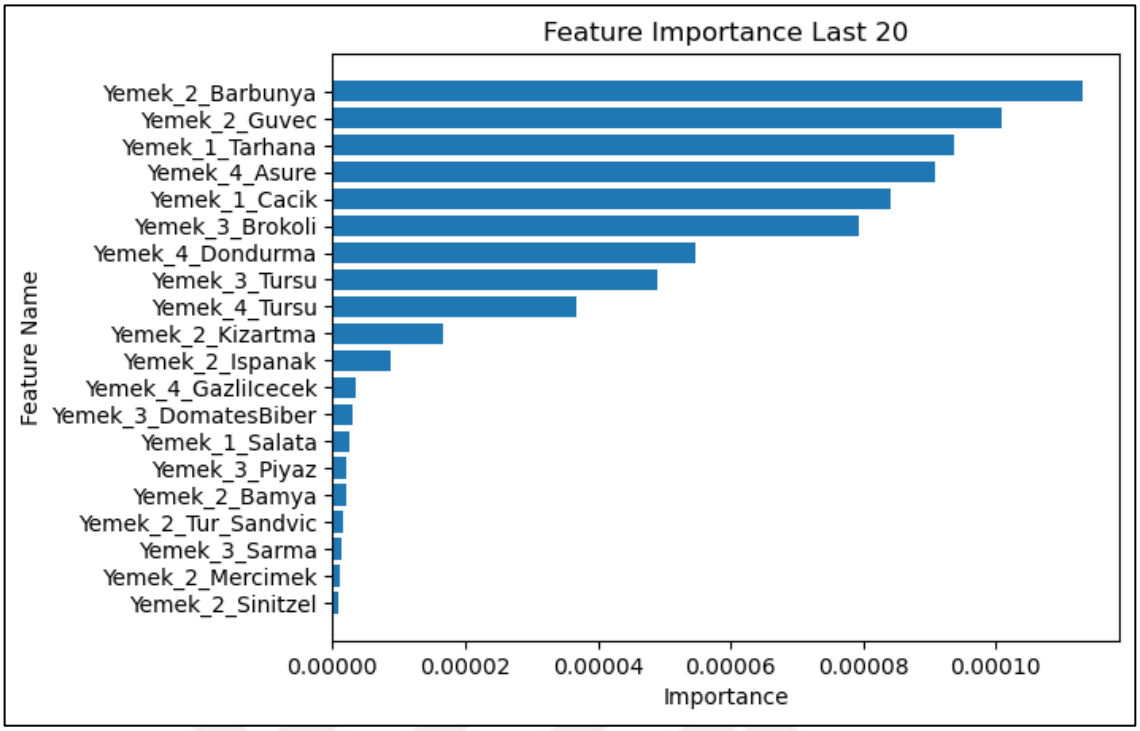
Genel veri setleri olan *Veri seti 1* ve *Veri seti 10*' a göre rassal orman algoritmasından elde edilen en iyi modellere ait öznitelik önem dereceleri hesaplanmıştır. Aşağıda Veri seti 1' e göre olan öznitelik önem dereceleri aşağıda sunulmuştur.



Şekil 3-16: Veri seti 1 öznitelik önem dereceleri ilk 20.

Şekil 3-16' de yer alan öznitelik önem derecelerine bakıldığında yaz mevsimi aylarına ait günlerde mevsim etkisinin en fazla önemde olduğu görülmektedir. Peşinden sırasıyla hava sıcaklığı, yaz tatili dönemine girilmesi, sömestr tatili dönemine girilmesi ve haftanın günleri içerisinde cuma gününün önem derecelerinin sıralamada önde olduğu görülmektedir. Ramazan ayı ve sınav günlerinin etkisi de önemli seviyede bulunmaktadır. Yemek çeşitleri içerisinde sırasıyla komposto ve salatanın önem dereceleri en yüksek çıkmıştır. Diğer yemek çeşitlerinin önem dereceleri nispeten düşük görünmektedir. Yani yemek tüketenlerin sayısına, o günkü yemek çeşidinin etkisinin nispeten az olduğu söylenebilir.

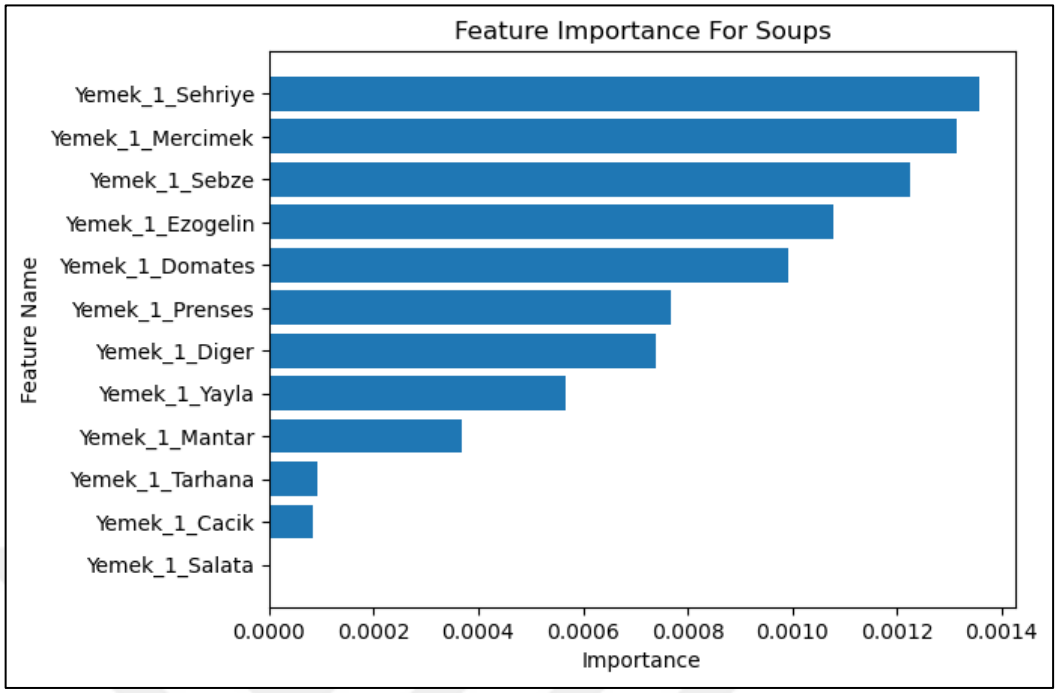
Öznitelik önem dereceleri içerisinde etkisi en az olan 20 öznitelik de (sondan 20) hesaplanmıştır. Aşağıda Şekil 3-17' sunulmuştur.



Şekil 3-17: Veri seti 1 öznitelik önem dereceleri son 20.

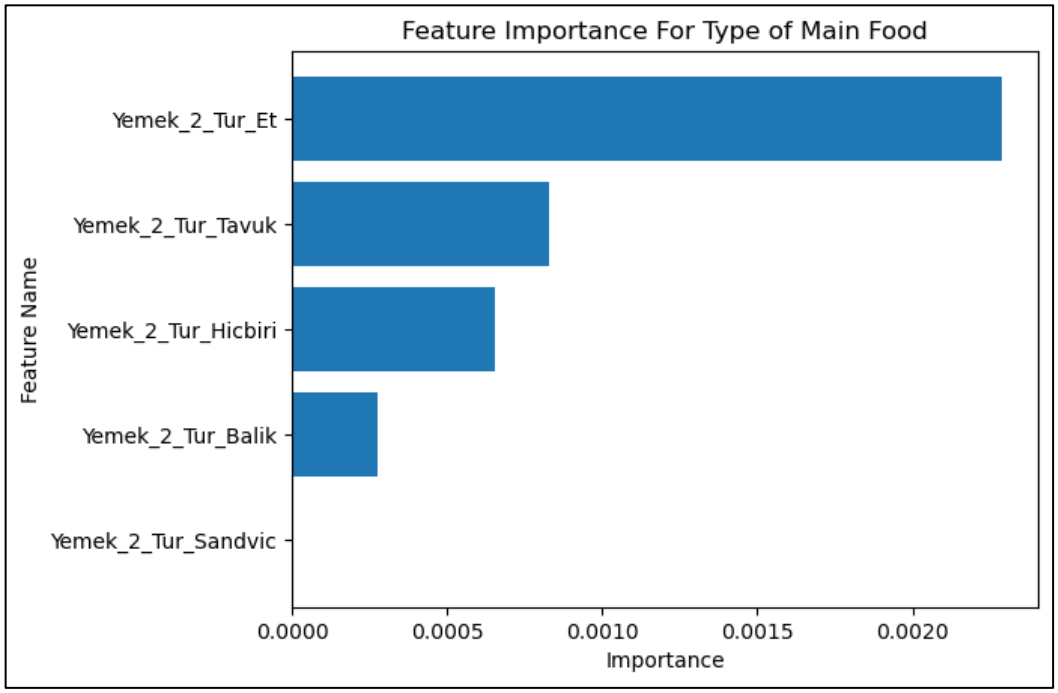
Şekil 3-17 incelendiğinde genelde yemek çeşitlerinin etkisinin yemek tüketim sayılarına etkisinin az derecelerde olduğu görünmektedir.

Çorba çeşitlerinin yemek tüketim sayılarına etkisi ile ilgili önem sıralamaları da aşağıda Şekil 3-18' da sunulmuştur. Buna göre şehriye çorbası ve mercimek çorbası sırası ile önem seviyeleri yüksek olan çorbalar olarak göze çarpmaktadır.



Şekil 3-18: Veri seti 1 çorbalar için önem dereceleri.

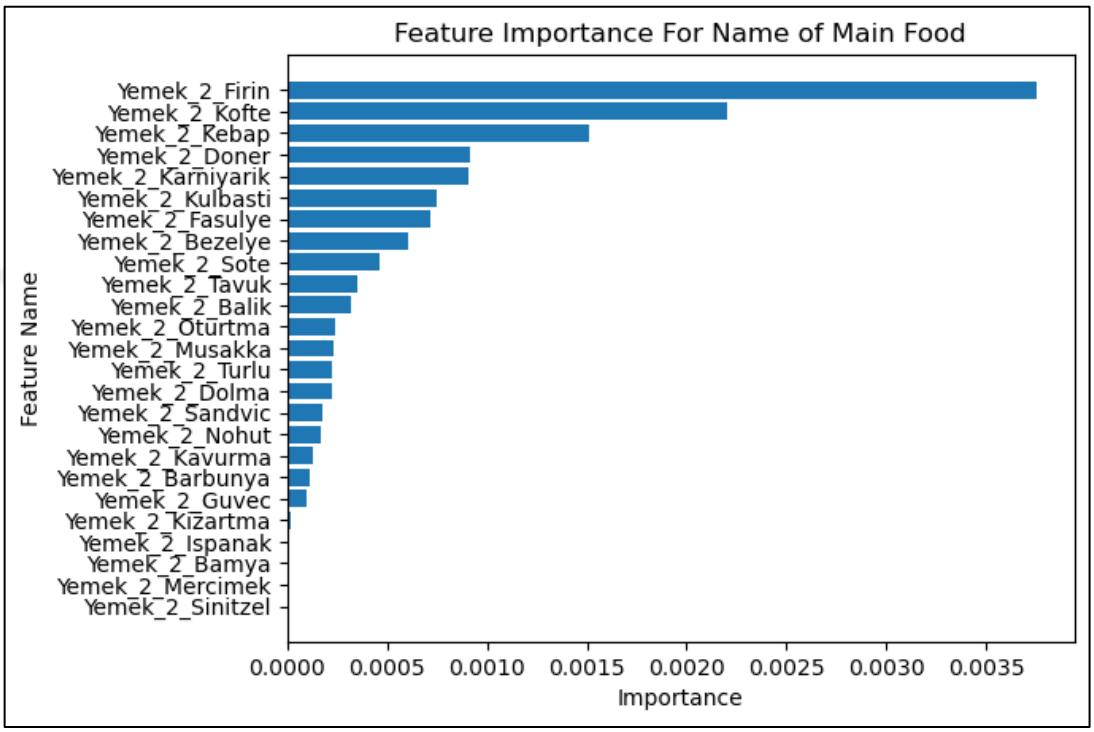
Ana yemek türlerine ait (et, balık, tavuk, vs.) önem dereceleri aşağıdaki şekilde sunulmuştur.



Şekil 3-19: Veri seti 1 ana yemek türüne göre önem dereceleri.

Ana yemek türlerine göre bakıldığında; ana yemeğin et içermesinin yemek tüketim sayısına etkisi, diğer türlerin yemek tüketim sayısı üzerindeki etkisine göre daha önemli olduğu görülmektedir.

Ana yemek çeşidine (genel ismine) göre yemek tüketim sayısı üzerindeki önem dereceleri aşağıdaki şekilde sunulmuştur.

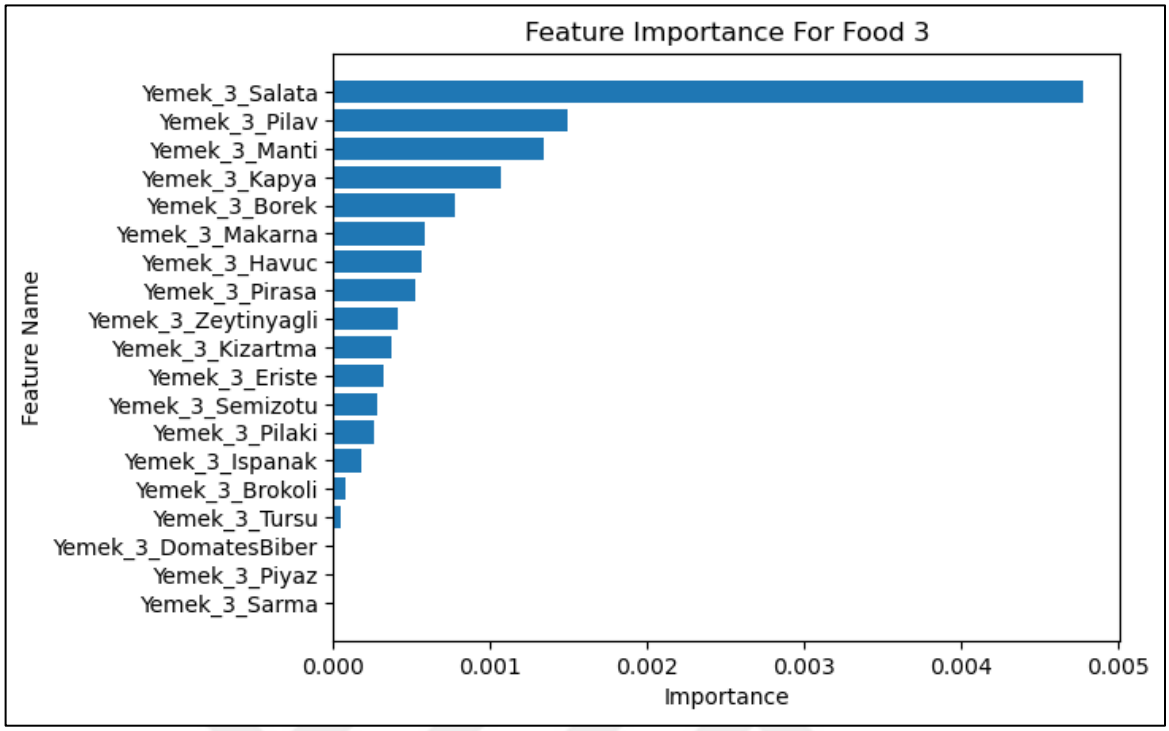


Şekil 3-20: Veri seti 1 ana yemek ismine göre önem dereceleri.

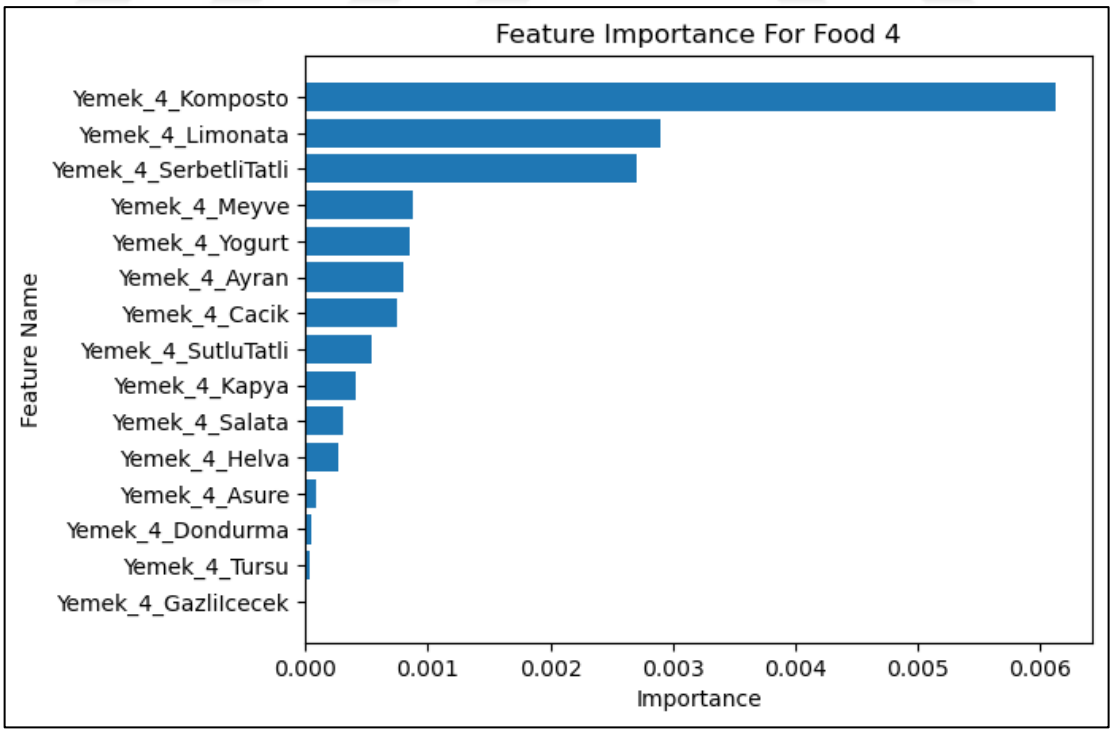
Ana yemek çeşidine göre yani ismine göre bakıldığında yemek tüketim sayısı üzerinde sırasıyla fırında pişirilmiş yemek (genelde tavuk türü), köfte, kebap, döner (et, tavuk) ve karnıyarık gibi yemeklerin etkisinin daha yüksek olduğu Şekil 3-20' de sunulmuştur.

Yan yemek türlerinin (Yemek 3) yemek tüketim sayısına etkisinin önem dereceleri aşağıdaki şekilde (Şekil 3-21) sunulmuştur. Buna göre salata çeşidinin önemi diğerlerine göre oldukça üstün seviyede görülmektedir.

Yan yemek türlerinden tatlı, meyve gibi çeşitlerin olduğu "*Yemek_4*" grubunun yemek tüketim sayısı üzerindeki önem dereceleri aşağıdaki şekilde (Şekil 3-22) sunulmuştur. Buna göre komposto en yüksek önem düzeyinde olup, limonata ve şerbetli tatlının önem seviyeleri diğerlerine göre üstün görünmektedir.

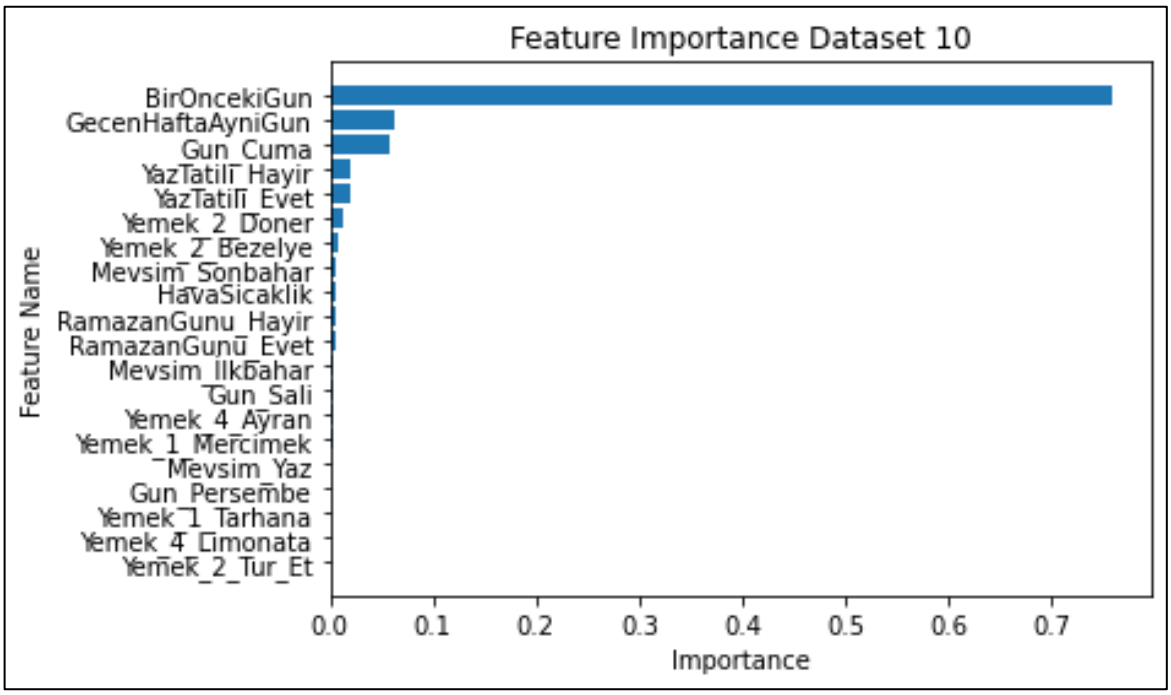


Şekil 3-21: Veri seti 1 Yemek 3 önem dereceleri.



Şekil 3-22: Veri seti 1 Yemek 4 önem dereceleri.

Veri seti 10’ a göre çalıştırılan rassal orman algoritmasında bulunan en iyi model için öznitelik önem dereceleri hesaplanmıştır. Aşağıda görüleceği üzere ilk 20 sıra Şekil 3-23’ de sunulmuştur. Bir önceki günün yemek tüketim sayısının hedef değişken olan yemek tüketim sayısı üzerindeki önemi diğerlerine göre oldukça üstün seviyededir. Bununla beraber aynı günün bir önceki hafta değeri ve cuma gününün yemek tüketim sayısı üzerindeki önem seviyeleri sonraki önemli olan bağımsız değişkenlerdir.



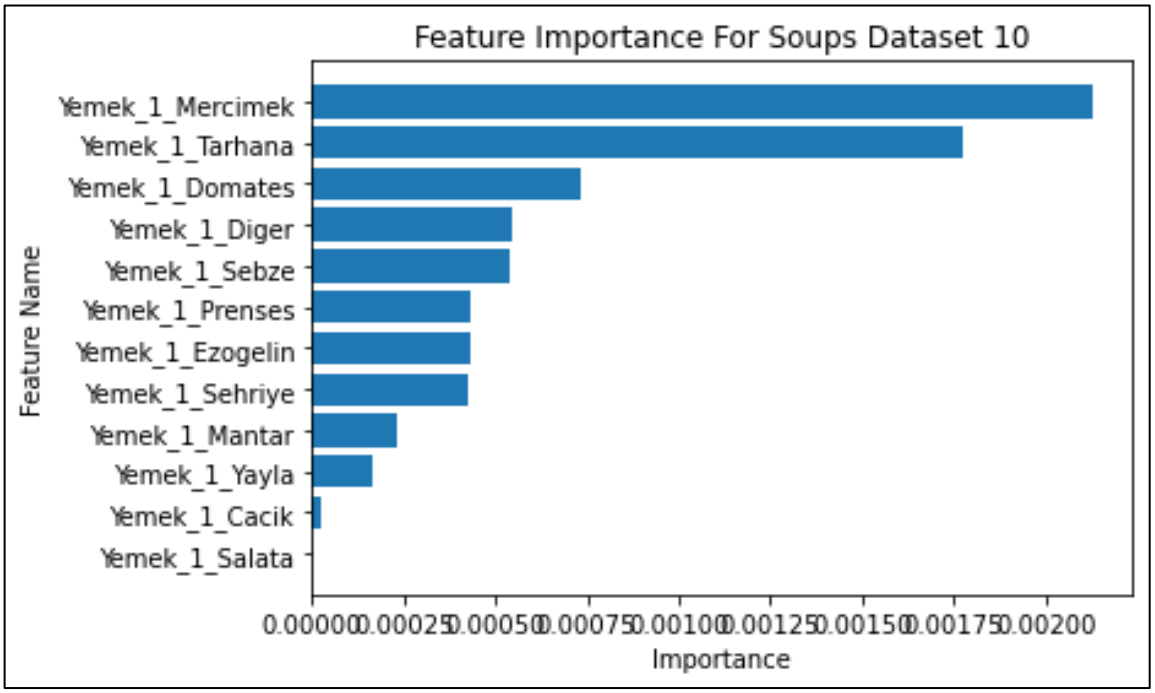
Şekil 3-23: Veri seti 10 öznitelik önem dereceleri ilk 20.

Veri seti 10’ a göre çorbalara ait önem seviyeleri aşağıdaki grafikte (Şekil 3-24) sunulmuştur. Grafiğe göre mercimek, tarhana ve domates çorbaları diğer çorbalara daha önemli düzeyde görülmektedir. Mercimek ve domates çorbaları ilk veri setinde olduğu gibi ikinci veri setinde de önem seviyeleri diğer çorbalara göre yüksek düzeyde çıkmıştır.

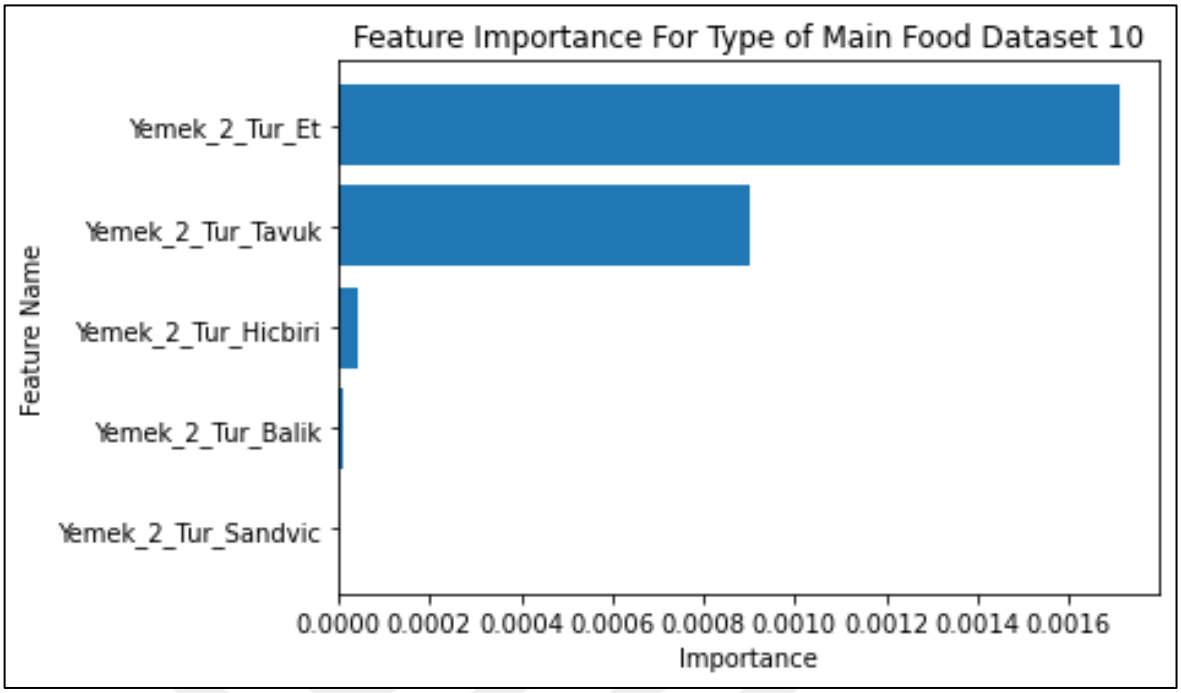
Yemek türüne göre (et, balık, tavuk, vs.) olan öznitelik önem dereceleri Şekil 3-25’ de sunulmuştur. Buna göre ana yemekte et veya tavuk olup olmaması diğer türlere göre yemek tüketenler tarafından daha önemli derecede görülmektedir.

Yemek ismine göre incelendiğinde döner, bezelye ve köfte yemeklerinin (yemek çeşitlerinin) yemek tüketenler tarafından diğer yemek çeşitlerine göre daha yüksek önem derecesinde

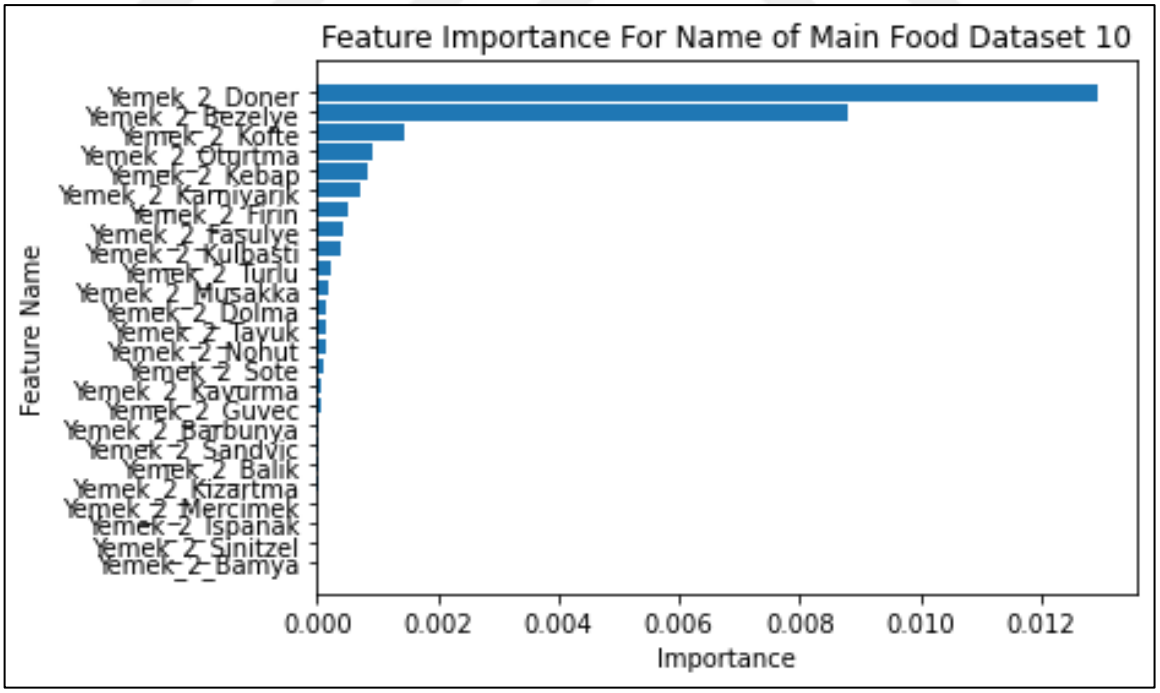
olduğu görülmektedir (Şekil 3-26). Yan yemek (Yemek 3) çeşidine göre incelendiğinde ise pilav, börek ve mantı yemeklerinin diğerlerine göre daha önemli seviyede olduğu görülmektedir (Şekil 3-27). Tatlı, içecek, meyve gibi (Yemek 4) çeşitler bakımından incelendiğinde ayran, limonata ve şerbetli tatlı gibi niteliklerin önem seviyeleri yüksek olarak göze çarpmaktadır (Şekil 3-28).



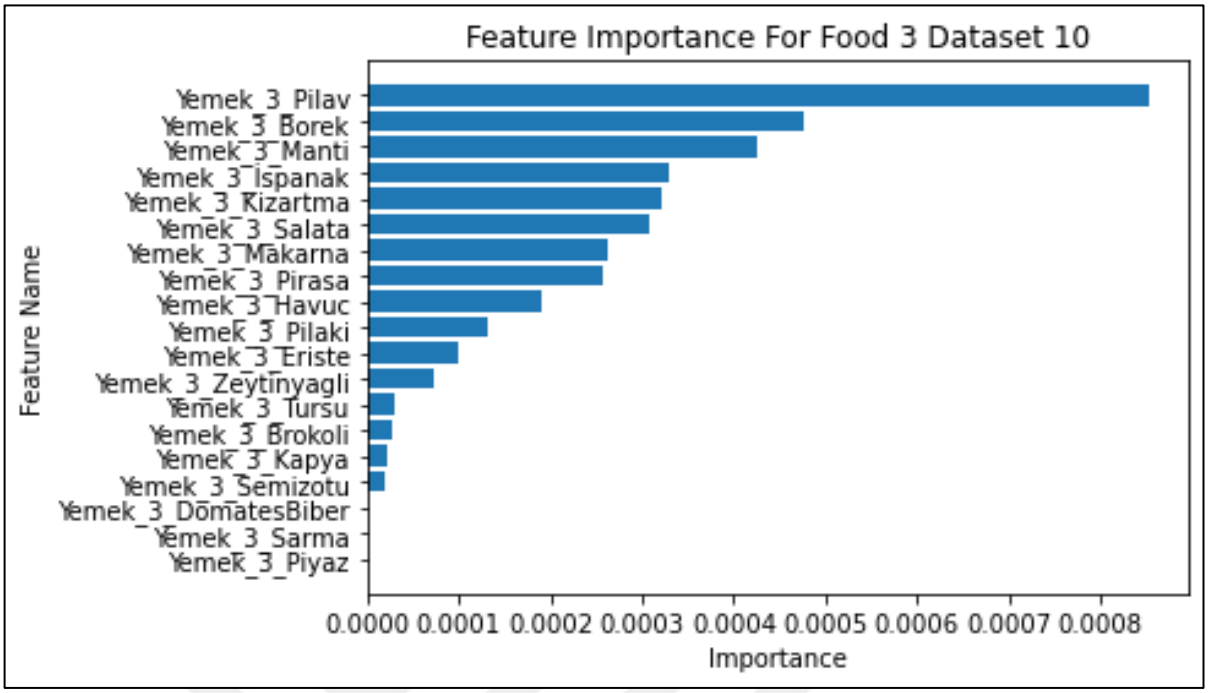
Şekil 3-24: Veri seti 10 çorbalar için önem dereceleri.



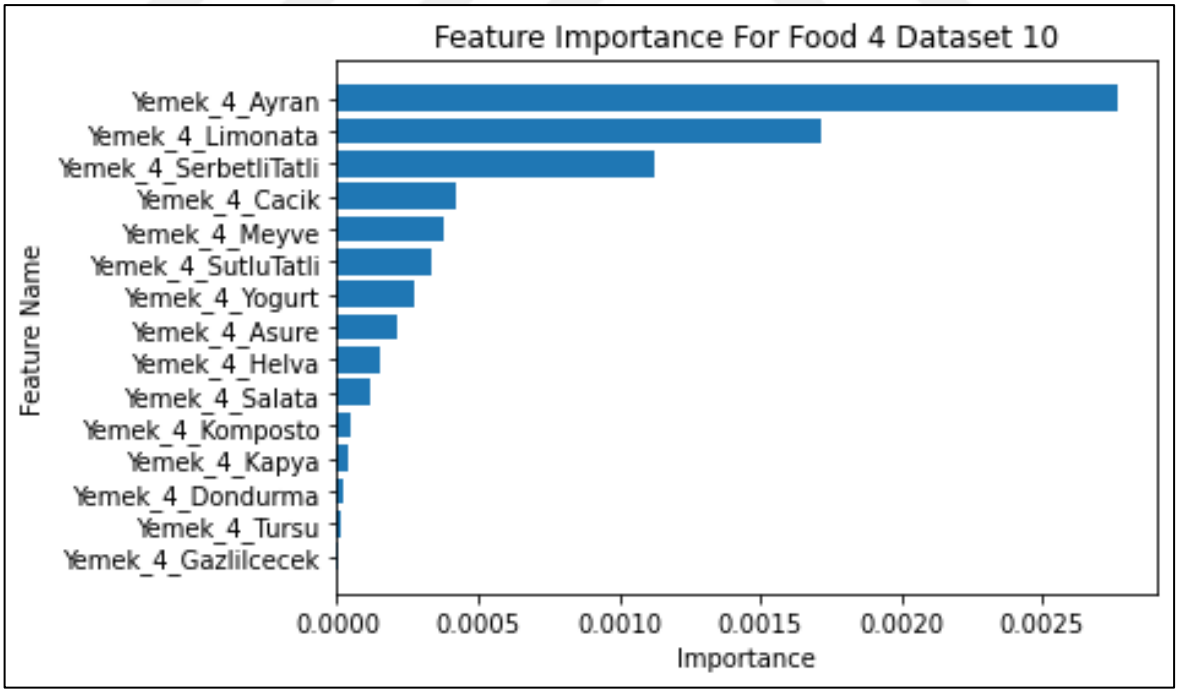
Şekil 3-25: Veri seti 10 ana yemek türüne göre önem dereceleri.



Şekil 3-26: Veri seti 10 ana yemek ismine göre önem dereceleri.



Şekil 3-27: Veri seti 10 Yemek 3 önem dereceleri.



Şekil 3-28: Veri seti 10 Yemek 4 önem dereceleri.

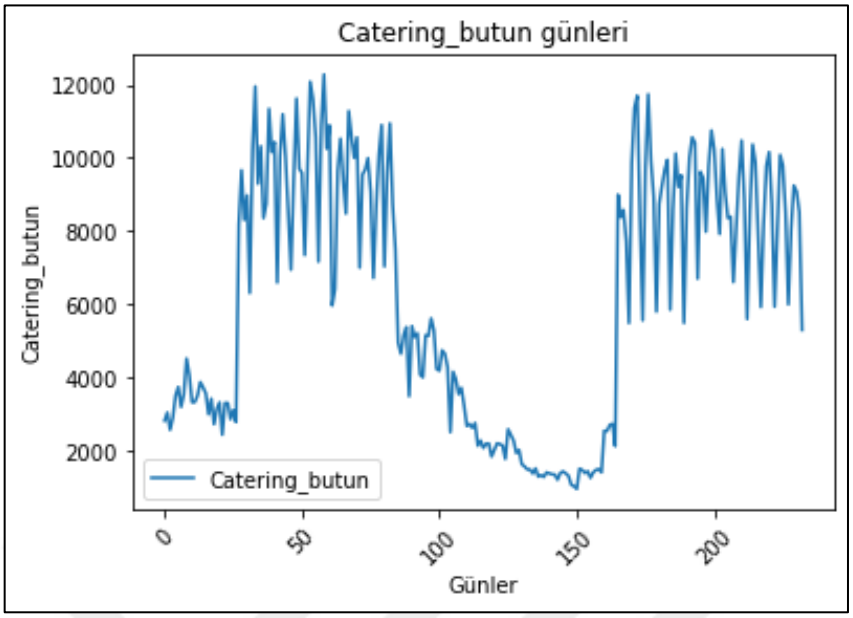
3.4.2. ARIMA ile Tahminleme

ARIMA (Auto-Regressive Integrated Moving Average) modeli hakkında bu bölümde uygulama ile ilgili bilgiler sunulmuştur. ARIMA için kullanılan zaman serisi günlük yemek tüketim sayılarıdır. Bu yöntemde kullanılan veri setleri aşağıdaki tabloda da (Tablo 3.5) sunulmuştur. Öncelikli olarak veri setleri durağanlık testinden geçirilmiştir. Durağan olmayan veri setleri için fark alma işlemi yapılmıştır. Fark alma işleminden sonra tekrar durağanlık testinden geçirilmiştir.

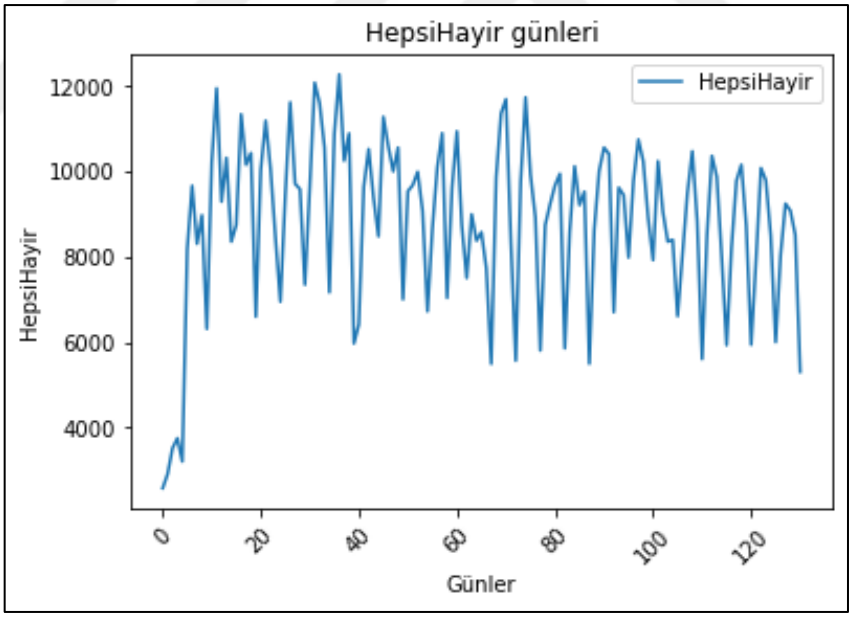
Tablo 3.5: ARIMA veri setleri tablosu.

Veri seti	Örnek Sayısı
Veri seti 1	233
Veri seti 2	131
Veri seti 3	26
Veri seti 4	25
Veri seti 5	26
Veri seti 6	27
Veri seti 7	27
Veri seti 8	20
Veri seti 9	20

Durağanlık testleri yapılmadan önce ($d = 0$) veri setlerinin günlük yemek tüketim grafikleri aşağıdaki şekillerde sunulmuştur (Şekil 3-29, Şekil 3-30, Şekil 3-31, Şekil 3-32, Şekil 3-33, Şekil 3-34, Şekil 3-35, Şekil 3-36, Şekil 3-37).



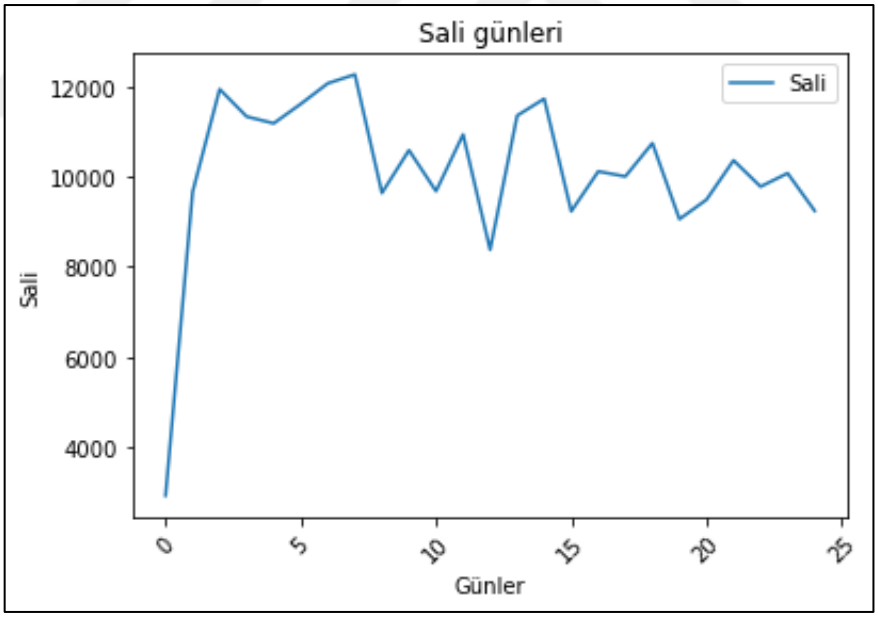
Şekil 3-29: ARIMA ADF testi öncesi veri seti 1 görünümü (d=0).



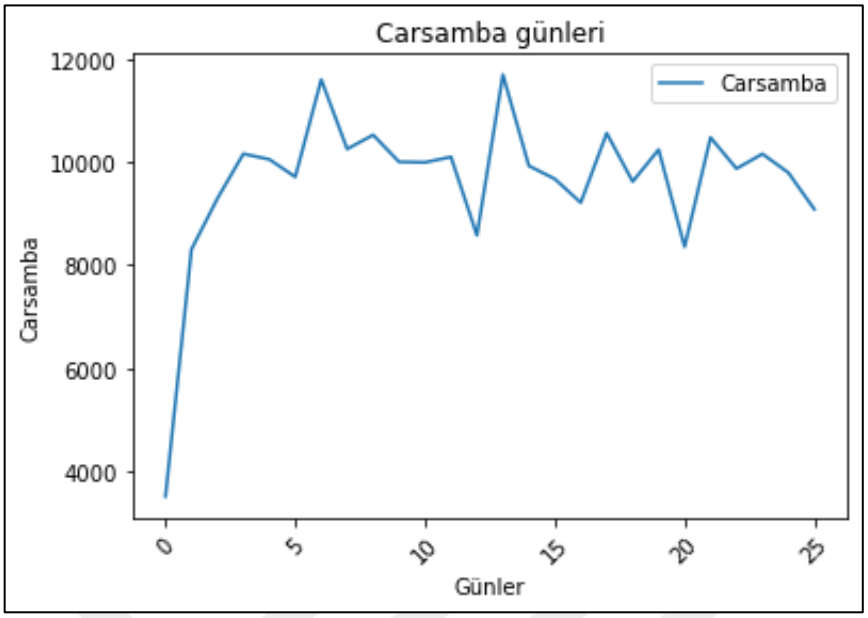
Şekil 3-30: ARIMA ADF testi öncesi veri seti 2 görünümü (d=0).



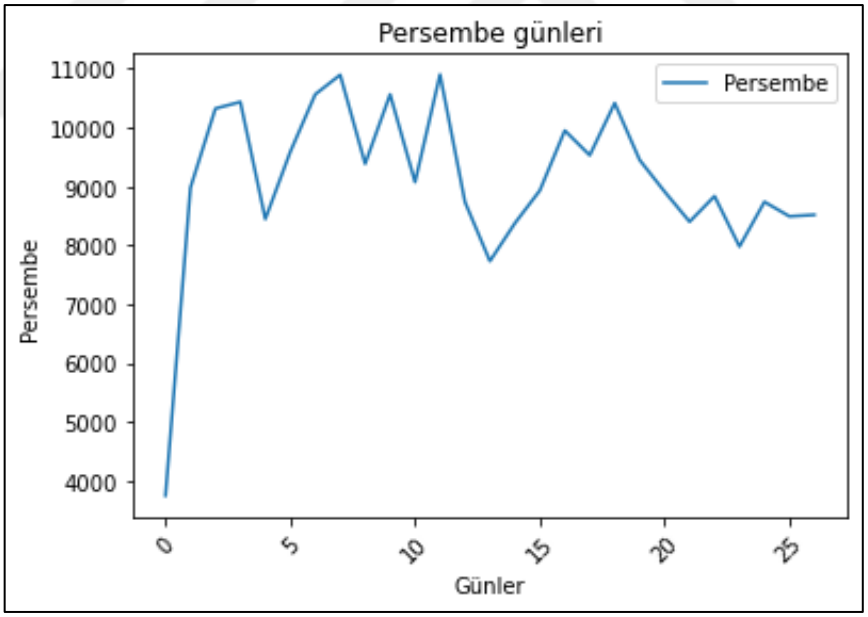
Şekil 3-31: ARIMA ADF testi öncesi veri seti 3 görünümü (d=0).



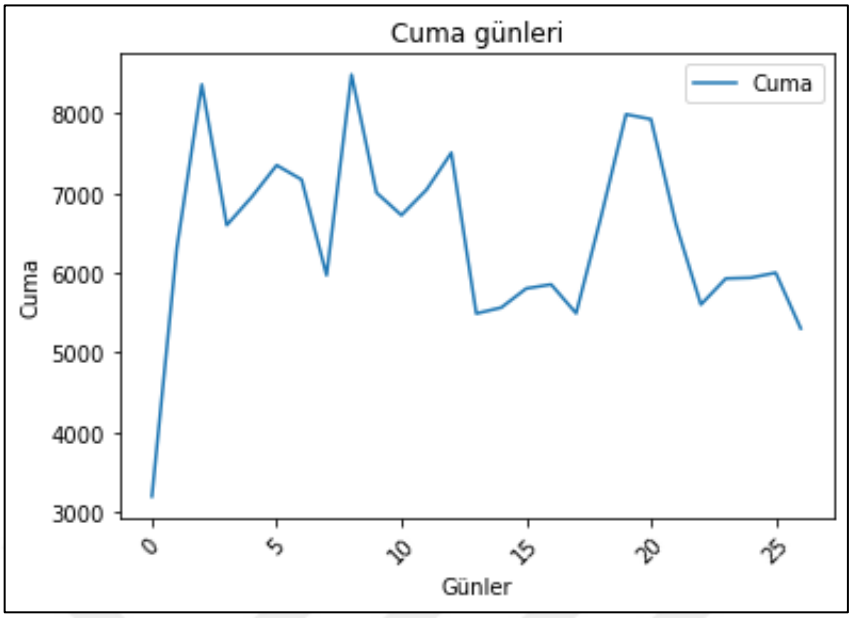
Şekil 3-32: ARIMA ADF testi öncesi veri seti 4 görünümü (d=0).



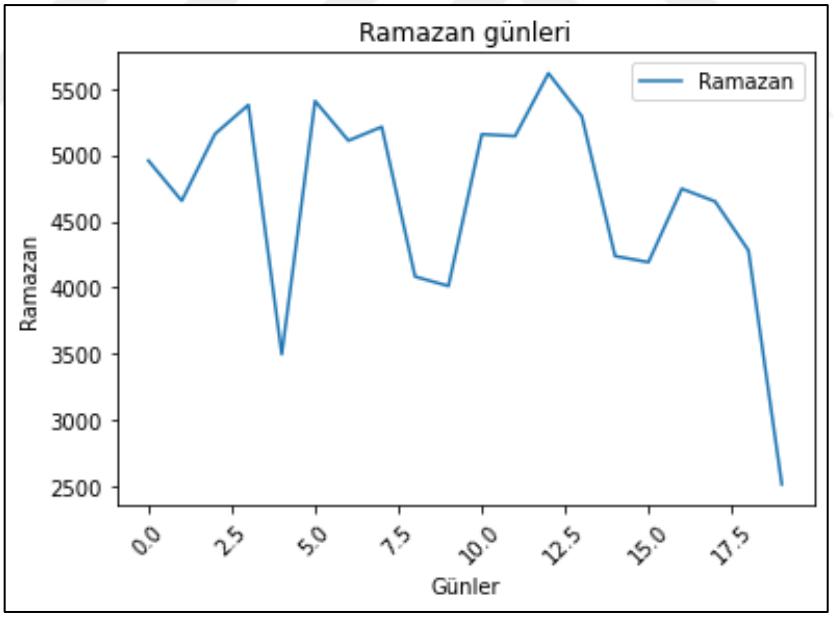
Şekil 3-33: ARIMA ADF testi öncesi veri seti 5 görünümü (d=0).



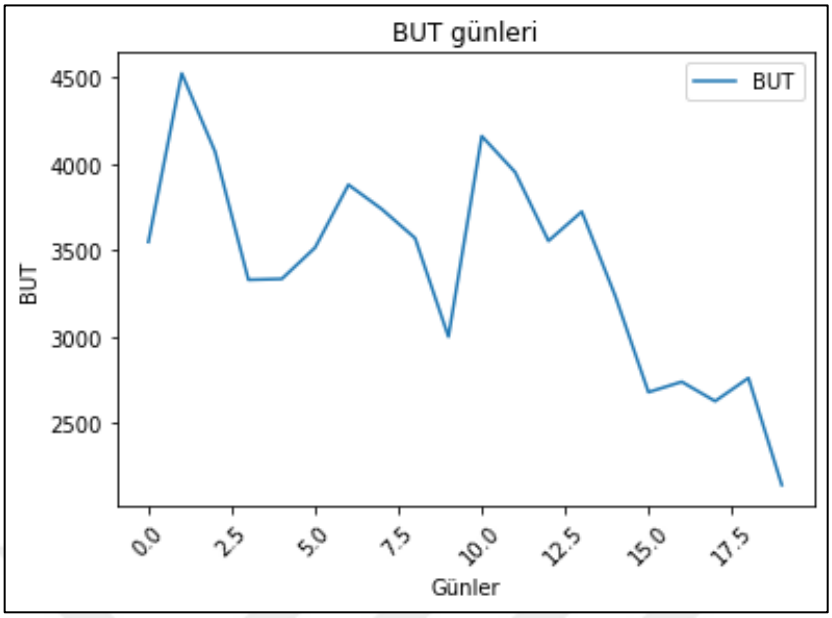
Şekil 3-34: ARIMA ADF testi öncesi veri seti 6 görünümü (d=0).



Şekil 3-35: ARIMA ADF testi öncesi veri seti 7 görünümü (d=0).



Şekil 3-36: ARIMA ADF testi öncesi veri seti 8 görünümü (d=0).



Şekil 3-37: ARIMA ADF testi öncesi veri seti 9 görünümü (d=0).

Durağanlık testinden geçirilen veri setlerine ilişkin test değerleri %5 güven seviyesine göre aşağıdaki tabloda sunulmuştur. Test sonuçları için %5 güven seviyesi değerleri dikkate alınmıştır.

Tablo 3.6: ARIMA ADF test sonuçları (d=0).

ARIMA Veri Setleri Durağanlık Sonuçları (d=0)						
Veri seti içeriği	**Veri setleri**	**ADF Test İstatistiği**	**Kritik değerler**			**P Values**
			1%	**5%**	**10%**	
Catering_butun	Veri seti 1	-1,881	-3,460	-2,875	-2,574	0,341
HepsiHayir	**Veri seti 2**	**-3,582**	**-3,486**	**-2,886**	**-2,580**	**0,006**
Pazartesi	Veri seti 3	2,012	-3,924	-3,068	-2,674	0,999
Sali	Veri seti 4	-1,498	-3,924	-3,068	-2,674	0,534
Carsamba	**Veri seti 5**	**-7,950**	**-3,724**	**-2,986**	**-2,633**	**0,000**
Persembe	**Veri seti 6**	**-6,474**	**-3,711**	**-2,981**	**-2,630**	**0,000**
Cuma	Veri seti 7	-1,576	-3,889	-3,054	-2,667	0,496
Ramazan	Veri seti 8	-1,312	-4,223	-3,189	-2,730	0,624
BUT	Veri seti 9	-1,329	-3,833	-3,031	-2,656	0,616

Tablo 3.6' da görüleceği üzere veri setleri 2, 5 ve 6 durağanlık testinde orijinal değerleri ile $(d = 0)$ durağan oldukları tespit edilmiştir. Durağan olmadıkları tespit edilen diğer veri setleri için fark alınma işlemi $(d = 1)$ uygulanarak tekrar durağanlık testi yapılmıştır.

Tablo 3.7: ARIMA ADF test sonuçları (d=1).

ARIMA Veri Setleri Durağanlık Sonuçları (d=1)						
Veri seti içeriği	**Veri setleri**	**ADF test istatistiği**	**Kritik değerler**			**P değeri**
			1%	**5%**	**10%**	
Catering_butun	**Veri seti 1**	**-4,518**	**-3,460**	**-2,875**	**-2,574**	**0,000**
Pazartesi	**Veri seti 3**	**-4,281**	**-3,788**	**-3,013**	**-2,646**	**0,000**
Sali	Veri seti 4	-3,036	-3,924	-3,068	-2,674	0,032
Cuma	Veri seti 7	-2,001	-3,924	-3,068	-2,674	0,286
Ramazan	**Veri seti 8**	**-5,033**	**-3,859**	**-3,042**	**-2,661**	**0,000**
BUT	**Veri seti 9**	**-5,543**	**-3,859**	**-3,042**	**-2,661**	**0,000**

Tablo 3.7' de görüleceği üzere fark alma işleminden $(d = 1)$ sonra durağan hale gelen veri setleri tespit edilmiştir. Ancak veri seti 4 ve 7 hala durağan hale gelmemiştir. İkinci bir fark alma işlemi gerçekleştirilmiştir. Aşağıdaki tabloda sunulmuştur.

Tablo 3.8: ARIMA ADF test sonuçları (d=2).

ARIMA Veri Setleri Durağanlık Sonuçları (d=2)						
Veri seti içeriği	**Veri setleri**	**ADF test istatistiği**	**Kritik değerler**			**P değeri**
			1%	**5%**	**10%**	
Sali	Veri seti 4	-2,181	-4,069	-3,127	-2,702	0,213
Cuma	**Veri seti 7**	**-4,423**	**-3,964**	**-3,085**	**-2,682**	**0,000**

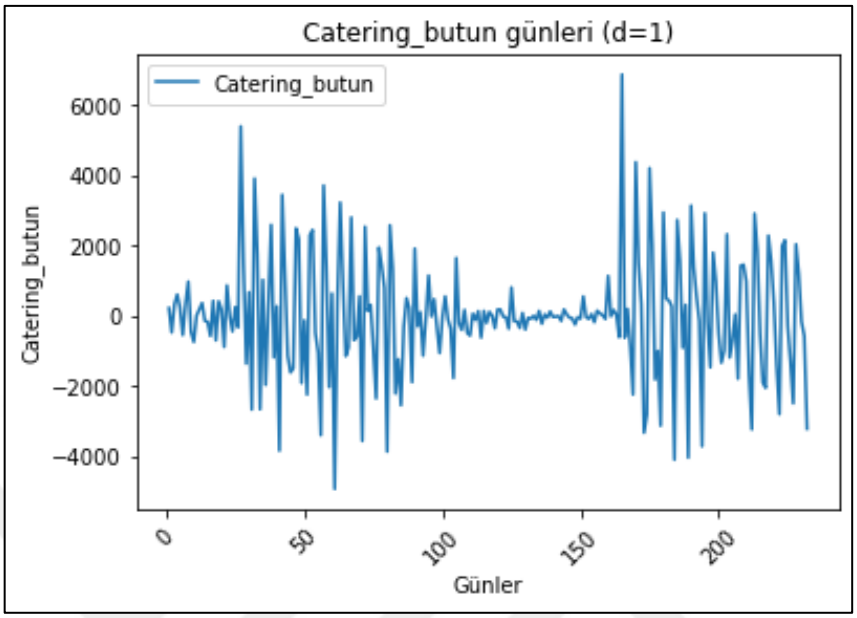
Tablo 3.8' de görüleceği üzere ikinci bir fark alma $(d = 2)$ işleminden sonra veri seti 7' nin durağan hale geldiği tespit edilmiştir. Ancak veri seti 4' ün henüz durağan olmadığı görülmektedir. Veri seti 4 için yapılan fark alma işlemlerinden sonra dördüncü fark alma işleminde durağan hale gelmiştir. Aşağıdaki tabloda test sonucu sunulmuştur.

Tablo 3.9: ARIMA ADF test sonuçları (d=4).

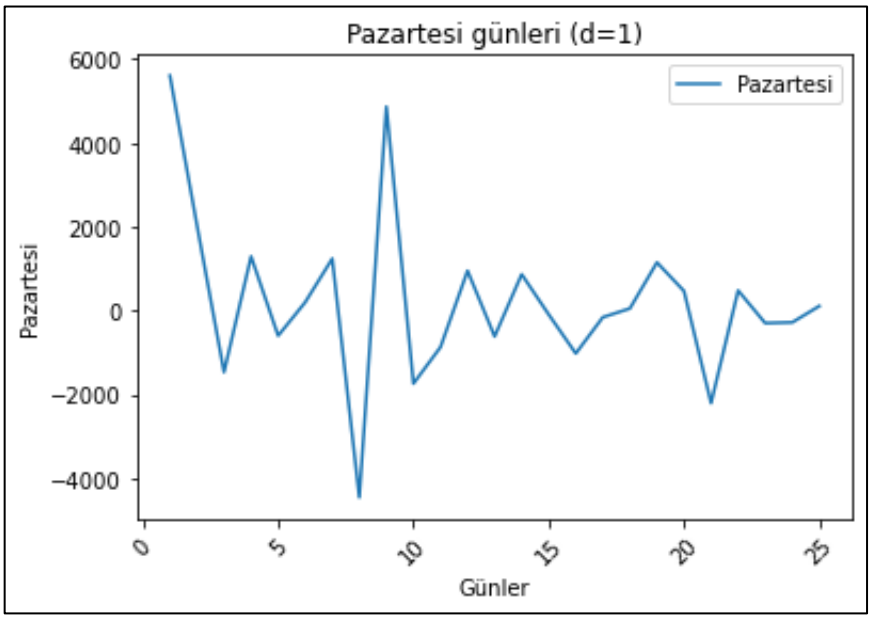
ARIMA Veri Setleri Durağanlık Sonuçları (d=4)						
Veri seti içeriği	**Veri setleri**	**ADF test istatistiği**	**Kritik değerler**			**P değeri**
			1%	**5%**	**10%**	
Sali	**Veri seti 4**	**-3,155**	**-4,138**	**-3,155**	**-2,714**	**0,023**

Tablo 3.9' da görüleceği üzere p değeri güven seviyesi olan 0,05' den küçüktür. Dolayısıyla durağanlık testi sonucunda veri seti 4' ün durağan olduğu sonucu ortaya çıkmaktadır. Fark alma

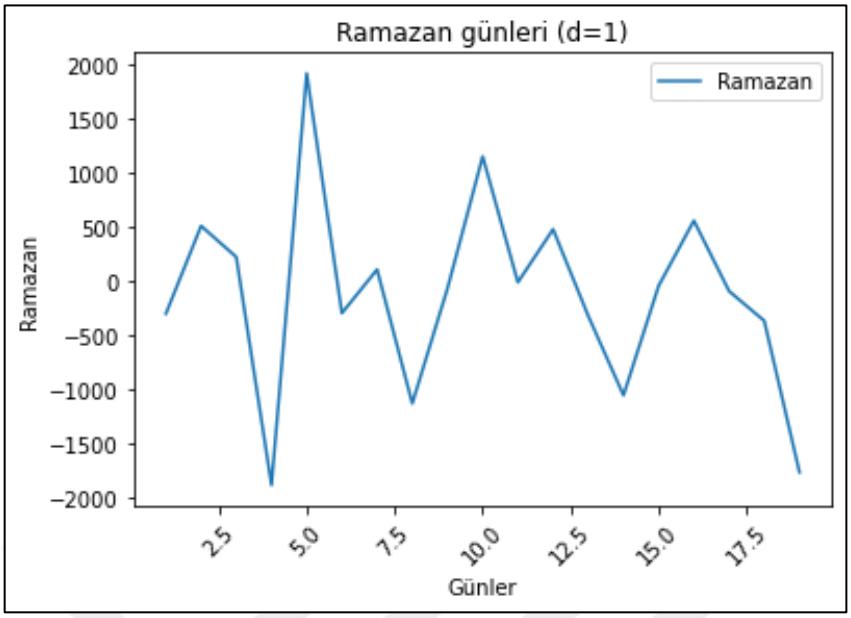
işlemleri ile durağan hale gelen veri setleri aşağıdaki şekillerde sunulmuştur (Şekil 3-38, Şekil 3-39, Şekil 3-40, Şekil 3-41, Şekil 3-42, Şekil 3-43).



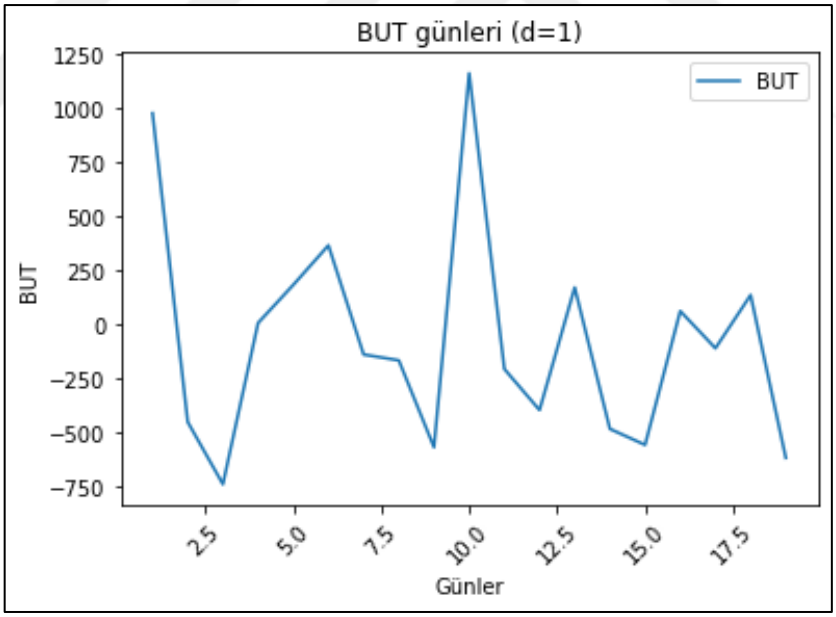
Şekil 3-38: ARIMA veri seti 1 durağanlık görünümü (d=1).



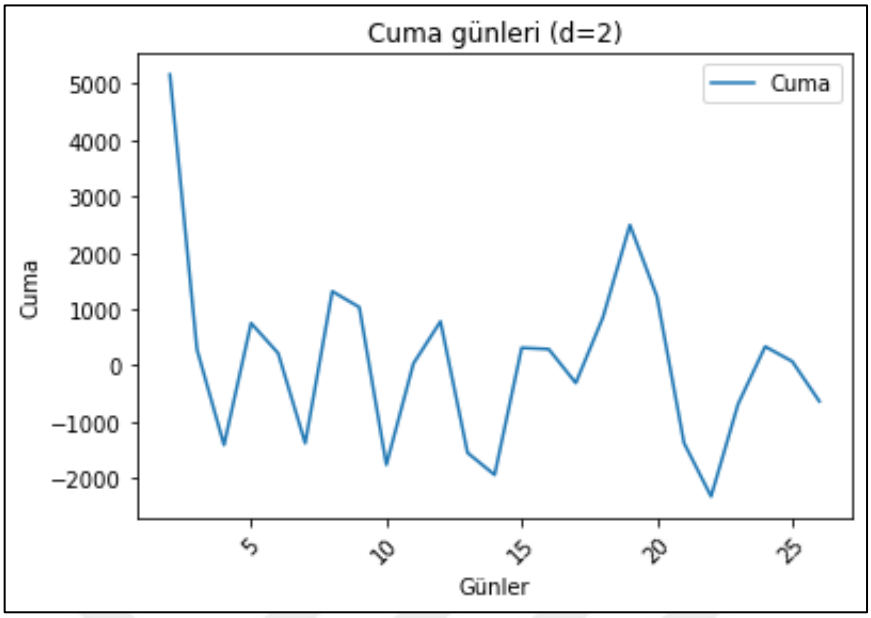
Şekil 3-39: ARIMA veri seti 3 durağanlık görünümü (d=1).



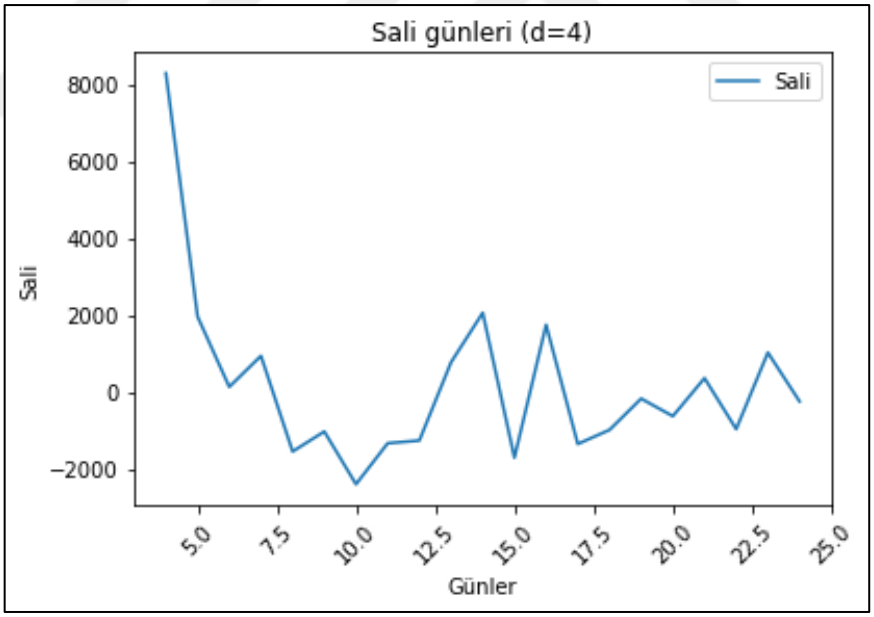
Şekil 3-40: ARIMA veri seti 8 durağanlık görünümü (d=1).



Şekil 3-41: ARIMA veri seti 9 durağanlık görünümü (d=1).



Şekil 3-42: ARIMA veri seti 7 durağanlık görünümü (d=2).



Şekil 3-43: ARIMA veri seti 4 durağanlık görünümü (d=4).

Veri setleri durağan hale getirildikten sonra ARIMA (p, d, q) yöntemi için gerekli parametre optimizasyonları (p ve q) artık yapılabilir hale gelmiştir. Her veri setinde, ARIMA (p, d, q) yöntemi için gerekli *d* değerleri durağanlık sağlanarak elde edilmiştir. Parametre optimizasyonu için *p ve q* değerleri araştırılmıştır. Her veri seti için çalıştırılan ARIMA yönteminde kullanılan parametre değer aralıkları aşağıdaki tabloda (Tablo 3.10) sunulmuştur. Tabloda ayrıca eğitim ve test oranları ile sayıları da sunulmuştur.

Tablo 3.10: ARIMA veri setleri parametre optimizasyon aralıkları.

Veri seti	**Örnek Sayısı**	**Test oranı**	**Eğitim Örnek Sayısı**	**Test Örnek Sayısı**	**p (0' dan)**	**d (sabit)**	**q (0' dan)**
Veri seti 1	233	10%	210	23	15	1	10
Veri seti 2	131	10%	117	14	10	0	5
Veri seti 3	26	20%	20	6	8	1	5
Veri seti 4	25	20%	20	5	8	4	5
Veri seti 5	26	20%	20	6	8	0	5
Veri seti 6	27	20%	21	6	8	0	5
Veri seti 7	27	20%	21	6	8	2	5
Veri seti 8	20	25%	15	5	5	1	3
Veri seti 9	20	25%	15	5	5	1	3

Parametre optimizasyonu yapılırken program içerisinde döngü tanımlanmıştır. Bu döngüde her parametre kombinasyonuna yönelik çalıştırılan ARIMA yönteminde geliştirilen modelin performansı AIC, MAPE, MAE, MSE, RMSE ve Korelasyon katsayısı (R^2) göstergeleri ile ölçülmüştür. Performans göstergeleri en uygun olan model en iyi model olarak elde edilmiştir.

Parametre optimizasyonlarına ilişkin aşağıdaki bilgiler sunulmuştur.

- Veri setleri programa excel' den aktarılmıştır.
- Normalizasyon yapılmamıştır.
- Tanımlanan döngü içerisinde her kombinasyonda ARIMA (p, d, q) modeli, ilgili parametrelere göre çalıştırılmıştır. Her çalıştırma esnasında veri seti, tanımlanan eğitim ve test oranına göre bölünmüştür. Eğitilen model test değerleri ile sınanmıştır. Performans gösterge değerleri elde edilmiştir.
- En iyi modele MAPE göstergesine göre karar verilmiştir.

- Parametre optimizasyonu aşamasında modeller geliştirilirken katsayıların uygunluk testleri de yapılarak en iyi modele karar verilmiştir (EK 3, EK 4, EK 5, EK 6, EK 7, EK 8, EK 9, EK 10, EK 11).

Veri setlerine ilişkin yapılan parametre optimizasyonu çalışmasında geliştirilen en iyi modeller ve performans göstergesi sonuçları aşağıdaki tabloda sunulmuştur. Geliştirilen en iyi modellerin katsayılarının uygunluk testleri gerçekleştirilmiştir ve p değerlerinin %5 güven seviyesinden düşük oldukları tespit edilmiştir.

Tablo 3.11: ARIMA Veri setleri sonuçları.

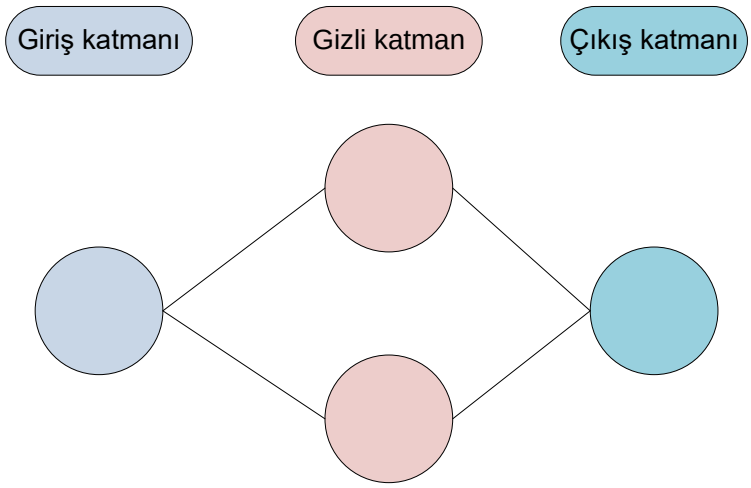
Veri seti	**Adı**	**(p, d, q)**	**AIC**	**R^2**	**MSE**	**RMSE**	**MAE**	**MAPE**
Veri seti 1	Catering_butun	(4, 1, 4)	3.931	0,73	662.868	814	647	0,084
Veri seti 2	HepsiHayir	(4, 0, 5)	2.294	0,59	957.524	979	800	0,114
Veri seti 3	Pazartesi	(1, 1, 1)	439	-0,51	976.593	988	713	0,085
Veri seti 4	Sali	(4, 4, 0)	385	-0,90	305.863	553	495	0,051
Veri seti 5	Carsamba	(1, 0, 2)	443	0,52	241.485	491	443	0,048
Veri seti 6	Persembe	(3, 0, 5)	565	-1,88	216.186	465	420	0,050
Veri seti 7	Cuma	(3, 2, 1)	416	-6,24	1.157.652	1.076	959	0,163
Veri seti 8	Ramazan	(4, 1, 0)	291	-0,48	970.824	985	808	0,246
Veri seti 9	BUT	(0, 1, 0)	276	-1,79	145.948	382	297	0,122

3.4.3. Yapay Sinir Ağları Modelleri ile Tahminleme

Bu çalışma kapsamında kullanılan yöntemlerden diğerleri de yapay sinir ağları modelleri olmuştur. Bu modeller içerisinde zaman serileri için daha uygun sonuçlar veren RNN, LSTM ve GRU modelleri kullanılmıştır. Modeller kullanılırken aynı ağ yapısı kullanılmasına dikkat edilmiştir. Bunun için hem RNN hem LSTM hem de GRU modeli için kullanılan aynı ağ yapısı aşağıdaki şekilde (Şekil 3-44) sunulmuştur. Şekilde görüleceği üzere modellerin ağ yapısı bir giriş katmanı, bir gizli katman ve bir çıkış katmanından oluşmaktadır. Gizli katman iki sinir hücresinden oluşmaktadır. Yöntemler ile ilgili diğer değerler ve çalıştırma biçimleri aşağıda sunulmuştur.

- Modeller kıyaslama aşamasından önce parametre optimizasyonuna tabi tutulmuştur.
- Parametre optimizasyonunda gecikme (lag) değerleri araştırılmıştır.

- Optimizasyonda performans kriterlerine göre en uygun gecikme değeri araştırılmıştır. Bunun için bir döngü tasarlanmıştır. Tablo 3.12' deki gecikme değerlerine kadar olan sayılar aralık olarak kullanılmıştır.
- Her veri seti için aynı döngü kullanılmıştır. MAPE değeri en iyi olan model ilgili veri seti için seçilmiştir. Sonuçları da bu şekilde alınmıştır.
- Modeller çalıştırıldığında başlangıç olarak tüm parametrelere ait **ağırlık değerleri eşit** değerlerde (0.10) atanmıştır.
- Modellerin çalıştırılması için, program içerisinde python kütüphanesi olan "*keras.layer*" içerisinden "*SimpleRNN, LSTM, GRU*" sınıfları kullanılmıştır.
- Veri setleri olarak Tablo 3.12' da yer alan veri setleri kullanılmıştır. Bu veri setlerinde yer alan günlük yemek tüketim sayıları kullanılmıştır. Veri setlerine ilişkin eğitim ve test oranları tabloda belirtildiği gibi alınmıştır.
- Veri setlerine "[0, 1]" aralığında normalizasyon uygulanmıştır. Test değerlerinin geliştirilen modelden elde edilen çıktıları kıyaslama için gerçek değerlerine tekrar dönüştürülmüştür.
- Geri yayılım algoritması olarak "*adam optimizer*" kullanılmıştır. Varsayılan değerleri ile kullanılmıştır.
- Kayıp (loss) fonksiyonu olarak "*MSE*" kullanılmıştır.
- İterasyon (epoch) sayısı "*1.000*" olarak ayarlanmıştır.



Şekil 3-44: Yapay sinir ağları uygulamasında kullanılan ağ yapısı.

Tablo 3.12: YSA Veri setleri tablosu.

Veri seti	Örnek Sayısı	Test oranı	Eğitim Örnek Sayısı	Test Örnek Sayısı	Gecikme değerleri
Veri seti 1	233	10%	210	23	60
Veri seti 2	131	10%	117	14	30
Veri seti 3	26	20%	20	6	8
Veri seti 4	25	20%	20	5	8
Veri seti 5	26	20%	20	6	8
Veri seti 6	27	20%	21	6	8
Veri seti 7	27	20%	21	6	8
Veri seti 8	20	25%	15	5	5
Veri seti 9	20	25%	15	5	5

RNN Yöntemi

RNN yöntemi ile her veri setine ilişkin yapılan optimizasyon neticesinde bulunan en iyi gecikme değerleri ve performans göstergeleri Tablo 3.13' de sunulmuştur.

Tablo 3.13: RNN Veri setleri sonuçları.

Recurrent Neural Network (RNN)						
Veri setleri	**Gecikme değeri (Bulunan)**	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
Veri seti 1	20	614	376.709	491	0,830	0,060
Veri seti 2	21	208	43.260	172	0,970	0,021
Veri seti 3	7	918	842.215	859	-16,250	0,105
Veri seti 4	7	352	124.218	321	-0,163	0,032
Veri seti 5	6	348	121.276	249	-0,682	0,024
Veri seti 6	8	711	505.877	620	-3,600	0,074
Veri seti 7	2	692	478.680	656	-3,456	0,111
Veri seti 8	3	405	164.327	291	-1,954	0,068
Veri seti 9	2	610	372.663	590	-6,723	0,214

LSTM Yöntemi

LSTM yöntemi ile her veri setine ilişkin yapılan optimizasyon neticesinde bulunan en iyi gecikme değerleri ve performans göstergeleri Tablo 3.14' de sunulmuştur.

Tablo 3.14: LSTM Veri setleri sonuçları.

Long Short Term Memory (LSTM)						
Veri setleri	**Gecikme değeri (Bulunan)**	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
Veri seti 1	30	617	381.013	538	0,817	0,067
Veri seti 2	23	253	63.864	211	0,961	0,026
Veri seti 3	1	1.034	1.069.710	963	-0,475	0,116
Veri seti 4	5	375	140.475	306	-0,315	0,031
Veri seti 5	6	352	123.765	251	-0,716	0,024
Veri seti 6	3	700	489.612	565	-4,445	0,068
Veri seti 7	7	722	521.761	647	-20,588	0,112
Veri seti 8	4	369	136.081	267	-1,447	0,063
Veri seti 9	2	617	380.132	597	-6,877	0,217

GRU Yöntemi

GRU yöntemi ile her veri setine ilişkin yapılan optimizasyon neticesinde bulunan en iyi gecikme değerleri ve performans göstergeleri Tablo 3.15' da sunulmuştur.

Tablo 3.15: GRU Veri setleri sonuçları.

Gated Recurrent Unit (GRU)						
Veri setleri	**Gecikme değeri (Bulunan)**	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
Veri seti 1	20	624	390	535	0,820	0,068
Veri seti 2	25	261	68.099	217	0,959	0,028
Veri seti 3	1	1.080	1.166.359	1.039	-0,608	0,124
Veri seti 4	8	328	107.307	297	-0,005	0,030
Veri seti 5	6	306	93.895	245	-0,302	0,024
Veri seti 6	8	780	608.805	702	-4,536	0,084
Veri seti 7	3	832	692.152	735	-5,443	0,125
Veri seti 8	4	381	144.974	319	-1,607	0,074
Veri seti 9	2	768	590.273	747	-11,233	0,272

4. BULGULAR

Çalışma kapsamında ROA, ARIMA, RNN, LSTM ve GRU yöntemleri kullanılmıştır. Toplamda 18 farklı veri seti ile çalışılmıştır. Yapılan çalışmada,

- Genel veri seti ile çalışılmıştır.
- Normal gün olarak değerlendirilmeye alınan; yani herhangi bir resmî tatil, dini gün, sınav günü, sömestr ve yaz tatili günlerini içermeyen veri seti 2 ve 11 ile çalışılmıştır.
- Normal günleri içeren veri setinde sadece *pazartesi* günlerini içeren veri seti 3 ve 12 ile çalışılmıştır.
- Normal günleri içeren veri setinde sadece *salı* günlerini içeren veri seti 4 ve 13 ile çalışılmıştır.
- Normal günleri içeren veri setinde sadece *çarşamba* günlerini içeren veri seti 5 ve 14 ile çalışılmıştır.
- Normal günleri içeren veri setinde sadece *perşembe* günlerini içeren veri seti 6 ve 15 ile çalışılmıştır.
- Normal günleri içeren veri setinde sadece *cuma* günlerini içeren veri seti 7 ve 16 ile çalışılmıştır.
- Sadece ramazan ayına ait günleri içeren veri seti 8 ve 17 ile çalışılmıştır.
- Sadece BUT (bütünleme) sınav günlerini içeren veri seti 9 ve 18 ile çalışılmıştır.
- Çalışılan veri setlerine göre eğitim ve test sayıları da değişik oranlarda alınmıştır.

Yapılan çalışmada, parametre optimizasyonları ile geliştirilen modeller çerçevesinde her veri setine göre en iyi performans göstergelerini sağlayan model ve gösterge değerleri aşağıdaki tablolarda sunulmuştur.

Tablo 4.1: Veri seti 1 modellerin kıyaslaması.

Veri Seti 1 (Genel Veri Seti)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (9.500)	806	648.973	608	0,902	0,090
ARIMA (4, 1, 4)	814	662.868	647	0,729	0,084
RNN (20)	**614**	**376.709**	**491**	**0,830**	**0,060**
LSTM (30)	**617**	**381.013**	**538**	**0,817**	**0,067**
GRU (20)	**624**	**390**	**535**	**0,820**	**0,068**

Tablo 4.2: Veri seti 2 modellerin kıyaslaması.

Veri Seti 2 (Normal Günler)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (6.000)	742	550.160	637	0,745	0,071
ARIMA (4, 0, 5)	979	957.524	800	0,592	0,114
RNN (21)	**208**	**43.260**	**172**	**0,970**	**0,021**
LSTM (23)	**253**	**63.864**	**211**	**0,961**	**0,026**
GRU (25)	**261**	**68.099**	**217**	**0,959**	**0,028**

Tablo 4.3: Veri seti 3 modellerin kıyaslaması.

Veri Seti 3 (Sadece Pazartesi Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (10.000)	**728**	**529.701**	**644**	**-0,475**	**0,071**
ARIMA (1, 1, 1)	988	976.593	713	-0,510	0,085
RNN (7)	918	842.215	859	-16,250	0,105
LSTM (1)	1.034	1.069.710	963	-0,475	0,116
GRU (1)	1.080	1.166.359	1.039	-0,608	0,124

Tablo 4.4: Veri seti 4 modellerin kıyaslaması.

Veri Seti 4 (Sadece Salı Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (9.500)	599	359.357	536	-0,993	0,054
ARIMA (4, 4, 0)	553	305.863	495	-0,903	0,051
RNN (7)	**352**	**124.218**	**321**	**-0,163**	**0,032**
LSTM (5)	**375**	**140.475**	**306**	**-0,315**	**0,031**
GRU (8)	**328**	**107.307**	**297**	**-0,005**	**0,030**

Tablo 4.5: Veri seti 5 modellerin kıyaslaması.

Veri Seti 5 (Sadece Çarşamba Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (5.500)	850	722.992	571	-0,189	0,053
ARIMA (1, 0, 2)	491	241.485	443	0,517	0,048
RNN (6)	**348**	**121.276**	**249**	**-0,682**	**0,024**
LSTM (6)	**352**	**123.765**	**251**	**-0,716**	**0,024**
GRU (6)	**306**	**93.895**	**245**	**-0,302**	**0,024**

Tablo 4.6: Veri seti 6 modellerin kıyaslaması.

Veri Seti 6 (Sadece Perşembe Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (6.000)	802	642.685	649	-0,071	0,069
ARIMA (3, 0, 5)	**465**	**216.186**	**420**	**-1,881**	**0,050**
RNN (8)	711	505.877	620	-3,600	0,074
LSTM (3)	700	489.612	565	-4,445	0,068
GRU (8)	780	608.805	702	-4,536	0,084

Tablo 4.7: Veri seti 7 modellerin kıyaslaması.

Veri Seti 7 (Sadece Cuma Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (4.000)	**793**	**628.110**	**511**	**-0,059**	**0,068**
ARIMA (3, 2, 1)	1.076	1.157.652	959	-6,244	0,163
RNN (2)	692	478.680	656	-3,456	0,111
LSTM (7)	722	521.761	647	-20,588	0,112
GRU (3)	832	692.152	735	-5,443	0,125

Tablo 4.8: Veri seti 8 modellerin kıyaslaması.

Veri Seti 8 (Sadece Ramazan Ayı Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (9.500)	**162**	**26.083**	**150**	**0,735**	**0,028**
ARIMA (4, 1, 0)	985	970.824	808	-0,482	0,246
RNN (3)	405	164.327	291	-1,954	0,068
LSTM (4)	369	136.081	267	-1,447	0,063
GRU (4)	381	144.974	319	-1,607	0,074

Tablo 4.9: Veri seti 9 modellerin kıyaslaması.

Veri Seti 9 (Sadece BUT Sınav Günleri)					
Model	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
ROA (2.500)	**195**	**38.201**	**183**	**0,853**	**0,048**
ARIMA (0, 1, 0)	382	145.948	297	-1,791	0,122
RNN (2)	610	372.663	590	-6,723	0,214
LSTM (2)	617	380.132	597	-6,877	0,217
GRU (2)	768	590.273	747	-11,233	0,272

Tablo 4.10: ROA Veri setleri kıyaslaması.

Rassal Orman Algoritması Yöntemi						
Veri setleri	**Ağaç Sayısı**	**RMSE**	**MSE**	**MAE**	**R^2**	**MAPE**
Veri seti 1 (Genel)	9.500	806	648.973	608	0,902	0,090
Veri seti 10 (Genel)	4.500	532	283.532	390	0,972	0,068
Veri seti 2 (Normal Günler)	6.000	742	550.160	637	0,745	0,071
Veri seti 11 (Normal Günler)	4.000	999	998.030	657	0,680	0,063
Veri seti 3 (Pazartesi Günleri)	10.000	728	529.701	644	-0,475	0,071
Veri seti 12 (Pazartesi Günleri)	8.000	807	651.627	584	-0,168	0,062
Veri seti 4 (Salı Günleri)	9.500	599	359.357	536	-0,993	0,054
Veri seti 13 (Salı Günleri)	8.500	291	84.814	216	0,607	0,020
Veri seti 5 (Çarşamba Günleri)	5.500	850	722.992	571	-0,189	0,053
Veri seti 14 (Çarşamba Günleri)	2.500	271	73.308	209	-1,719	0,021
Veri seti 6 (Perşembe Günleri)	6.000	802	642.685	649	-0,071	0,069
Veri seti 15 (Perşembe Günleri)	3.000	490	240.466	397	0,616	0,041
Veri seti 7 (Cuma Günleri)	4.000	793	628.110	511	-0,059	0,068
Veri seti 16 (Cuma Günleri)	3.500	532	283.167	313	-0,580	0,042
Veri seti 8 (Ramazan)	9.500	162	26.083	150	0,735	0,028
Veri seti 17 (Ramazan)	1.000	285	81.305	202	0,277	0,043
Veri seti 9 (BUT)	2.500	195	38.201	183	0,853	0,048
Veri seti 18 (BUT)	10.000	196	39.010	188	0,698	0,054

Tablo 4.11: Veri setlerine göre en iyi modeller ve performans göstergeleri.

Veri setleri	En İyi Model	Gecikme değeri/ Ağaç sayısı	RMSE	MSE	MAE	R^2	MAPE
Veri seti 1	RNN	20	614	376.709	491	0,830	0,060
Veri seti 2	RNN	21	208	43.260	172	0,970	0,021
Veri seti 3	ROA	10.000	728	529.701	644	-0,475	0,071
Veri seti 4	GRU	8	328	107.307	297	-0,005	0,030
Veri seti 5	GRU	6	306	93.895	245	-0,302	0,024
Veri seti 6	ARIMA	(3, 0, 5)	465	216.186	420	-1,881	0,050
Veri seti 7	ROA	4.000	793	628.110	511	-0,059	0,068
Veri seti 8	ROA	9.500	162	26.083	150	0,735	0,028
Veri seti 9	ROA	2.500	195	38.201	183	0,853	0,048
Veri seti 10	ROA	4.500	532	283.532	390	0,972	0,068
Veri seti 11	ROA	4.000	999	998.030	657	0,680	0,063
Veri seti 12	ROA	8.000	807	651.627	584	-0,168	0,062
Veri seti 13	ROA	8.500	291	84.814	216	0,607	0,020
Veri seti 14	ROA	2.500	271	73.308	209	-1,719	0,021
Veri seti 15	ROA	3.000	490	240.466	397	0,616	0,041
Veri seti 16	ROA	3.500	532	283.167	313	-0,580	0,042
Veri seti 17	ROA	1.000	285	81.305	202	0,277	0,043
Veri seti 18	ROA	10.000	196	39.010	188	0,698	0,054

5. TARTIŞMA VE SONUÇ

Catering sektöründe faaliyet gösteren işletmeler, menü planlama süreci önceliğinde faaliyetlerini sürdürmektedirler. Menü planlama süreci, menü içeriğinin oluşturulması ve tüketim sayısının belirlenmesi aşamasıdır. Buna göre süreç, satın alma ve üretim faaliyetleri ile devam etmektedir. Dolayısıyla menü içeriğinin oluşturulması ve devamında tüketim sayısının belirlenmesi oldukça önemlidir. Yapılan çalışmada bir üniversite yemekhanesine ait belli bir zaman dilimdeki yemek tüketim sayıları kullanılarak geliştirilen modellerin sonuçları kıyaslanmıştır ve catering sektöründe faaliyet gösteren işletmelerde yemek tüketim sayısının belirlenmesi için bir yaklaşım belirlenmiştir. İşletmelerin faaliyetlerini sürdürmesinde başvurabilecekleri bir yaklaşım oluşturulması amacıyla yapılan bu çalışmada elde edilen çıktılar, sektörde uygulanabilirliği göstermektedir. Kullanılan yöntemler çerçevesinde elde edilen MAPE değerleri ile bu uygulanabilirlik kanıtlanmıştır. Veri setleri daha geniş tutularak (bağımsız değişken ve zaman anlamında) sonuçlar daha da iyileştirilebilir. Çalışma içerisinde kullanılan ROA, ARIMA, RNN, LSTM ve GRU yöntemlerinde daha değişik senaryolar izlenerek daha iyi sonuçlar elde edilebilir. Farklı zaman dilimlerine göre elde edilen veri setleri ile daha dinamik modeller geliştirilmesi düşünülmüştür.

MAPE değerlerinde %10 altındaki değerler yüksek doğruluk düzeyi olarak kabul edilmektedir. Yapılan çalışma yemek tüketim sayısı tahmini olduğundan, yapılacak tahminin fazla çıkması durumunda fazla işgücü ve fazla satın alma yaratması, yapılacak tahminin eksik çıkması durumunda memnuniyetsizlik yaratması sebeplerinden dolayı istenen MAPE değerinin olabildiğince düşük çıkması istenmektedir.

Veri setleri içerisinde örnek sayıları nispeten fazla olan veri seti 1 (233) ve veri seti 2' nin (131) kullanıldığı modellerde yapay sinir ağı yöntemi olan RNN diğerlerine göre daha üstün performans sağlamıştır. Ancak diğer YSA yöntemleri olan LSTM ve GRU yöntemleri de RNN' e yakın performans göstermişlerdir. Literatür ile kıyaslandığında (Tablo 2.1), elde edilen MAPE değerleri literatürde incelenen değerlere göre daha üstün çıkmıştır. Literatürde Calp' in 2019 yılındaki yapılan çalışmasında (Calp, 2019) bulunan MAPE değeri modelin eğitim aşamasında bulunan değer olduğundan bu rakam kıyaslamaya alınmamıştır. Bu çalışmada test değerleri üzerinden kıyaslama yapılmıştır. Literatürdeki en yakın MAPE değeri %3,8 ile Çoban' ın (Çoban, 2019) yaptığı çalışma ile bulunmuştur ki bu çalışmada aylık tüketim sayıları

üzerinden elde edilmiştir. Bu çalışmada ise günlük tüketim değerleri üzerinden tahmin çalışması yürütülmüştür.

Haftanın günleri olarak değerlendiğinde;

- ✓ Pazartesi günlerini içeren Veri seti 3' e göre çalıştırılan modellerde Rassal Orman Algoritması ile diğer yöntemlere göre (%7,1) daha üstün MAPE değeri elde edilmiştir.
- ✓ Salı günlerini içeren Veri seti 4' e göre çalıştırılan modellerde RNN, LSTM ve GRU yöntemleri ile diğer yöntemlere göre (ortalama %3) daha üstün MAPE değerleri elde edilmiştir.
- ✓ Çarşamba günlerini içeren Veri seti 5' e göre çalıştırılan modellerde RNN, LSTM ve GRU yöntemleri ile diğer yöntemlere göre (ortalama %2,4) daha üstün MAPE değerleri elde edilmiştir.
- ✓ Perşembe günlerini içeren Veri seti 6' ya göre çalıştırılan modellerde ARIMA yöntemi ile diğer yöntemlere göre (%5) daha üstün MAPE değeri elde edilmiştir.
- ✓ Cuma günlerini içeren Veri seti 7' ye göre çalıştırılan modellerde Rassal Orman Algoritması yöntemi ile diğer yöntemlere göre (%6,8) daha üstün MAPE değeri elde edilmiştir.

Ramazan ayı günlerini içeren Veri seti 8' e göre çalıştırılan modellerde Rassal Orman Algoritması yöntemi ile diğer yöntemlere göre (%2,8) daha üstün MAPE değeri elde edilmiştir.

BUT sınavı günlerini içeren Veri seti 9' a göre çalıştırılan modellerde Rassal Orman Algoritması yöntemi ile diğer yöntemlere göre (%4,8) daha üstün MAPE değeri elde edilmiştir.

Veri setleri 7, 8 ve 9' un kullanıldığı modeller içerisinde ROA yöntemi diğerlerine göre daha üstün performans göstermiştir. Veri setlerinin örnek sayıları sırasıyla 27, 20, 20 gibi az sayıda olması dikkate alınmalıdır. ROA yönteminin üstün performans göstermesi, bağımsız değişkenlerin kullanıldığı belirtilerek açıklanabilir.

Rassal orman algoritması yöntemi ile 9 veri setine ek olarak 9 veri seti ile daha çalışılmıştır. Toplamda 18 veri seti ile çalışılmıştır. İlk 9 veri setine ek olarak *BirOncekiGun* ve *GecenHaftaAyniGun* yemek tüketim sayılarını içeren değişkenler eklenerek veri setleri 18 adet olarak genişletilmiştir. Dolayısıyla eklenen değişkenlerin eklenmemiş veri setleri ile performanslar göstergelerindeki değişimi görmek amacıyla Tablo 4.10 oluşturulmuştur. Tablo

incelendiğinde Ramazan ayı günleri ve BUT sınav günleri dışındaki veri setlerinde eklenen değişkenler performans göstergelerinde pozitif etki yaratmıştır. MAPE değerlerinde;

- ✓ Veri seti 10' da, veri seti 1' e göre %24 düşüş yaşanmıştır.
- ✓ Veri seti 11' de, veri seti 2' ye göre %11 düşüş yaşanmıştır.
- ✓ Veri seti 12' de, veri seti 3' e göre %13 düşüş yaşanmıştır.
- ✓ Veri seti 13' de, veri seti 4' e göre %63 düşüş yaşanmıştır.
- ✓ Veri seti 14' de, veri seti 5' e göre %60 düşüş yaşanmıştır.
- ✓ Veri seti 15' de, veri seti 6' ya göre %41 düşüş yaşanmıştır.
- ✓ Veri seti 16' da, veri seti 7' ye göre %38 düşüş yaşanmıştır.

Farklı veri setleri kullanılarak yılın farklı dönemlerindeki, farklı trendlere sahip zaman aralıklarındaki yanlılığın etkisi azaltılmıştır. Farklı zaman aralıkları kendi içerisinde incelenmiştir. Ramazan ayı günleri, BUT sınav günleri gibi özel günler yemek tüketim sayılarında değişkenliğe yol açmaktadır. Bu amaçla ayrı olarak veri setleri oluşturulmuştur ve modeller geliştirilmiştir. Bu çalışma kapsamında söz konusu günler için oluşturulan veri setlerinin örnek sayıları az olsa da ileri ki çalışmalarda örnek sayılarının geniş tutulduğu çalışmalar yapılarak daha kararlı modeller geliştirilebilir. Aynı zamanda bu alanda tahmin çalışmaları yapacak kişiler ve uzmanlar için bu çalışma kaynak olarak ele alınabilecektir. Ramazan ayı günleri gibi günler işletmeler için de geçerli olduğundan, bu gibi özel günlerin ayrı olarak değerlendirilmesi gerektiği düşünülmektedir.

Özel günler (sınav ve ramazan ayı), yaz ve sömestr tatili günlerini içermeyen veri seti oluşturularak normal günler için ayrı bir veri seti oluşturulmuştur. Bu veri seti içerisinde de haftanın günlerine göre 5 farklı veri seti daha oluşturularak modeller geliştirilmiştir. Öğrencilere ait haftalık ders programlarının çeşitliliği göz önüne alındığında bazı günler yemek tüketim sayısı yüksek olurken, bazı günler düşük sayıda olabilmektedir. Bu amaçla haftanın günlerinin tek tek ele alındığı modeller geliştirilmiştir. İşletmeler açısından bakıldığında haftanın günlerine göre yapılan organizasyonlar, toplantılar, ziyaretler, vs. gibi faaliyetlerin değişkenlik gösterebildiği ele alındığında, haftanın günlerinin tek tek ele alındığı veri setleri oluşturularak, o güne ait modellerin daha kararlı olacağı düşünülmektedir.

Çalışmada ROA yöntemi için kullanılan bağımsız değişkenlerin (Tablo 3.1) catering işletmeleri için haftanın günlerine göre yapılacak yemek tüketim sayısı tahminlerinde oldukça kolaylık sağlayabileceği düşünülmektedir. Buna göre öznitelik önem dereceleri analizi yapılmıştır.

- Yapılan öznitelik önem dereceleri incelemesinde veri seti 1' e göre (Şekil 3-16) analiz edildiğinde *HavaSicaklik* değişkeninin önem derecesinin bağımsız değişkenler içerisinde çok önemli olduğu görülebilmektedir.
- Aynı şekilde *Gun_Cuma*[10] değişkeni de önemli görülmektedir.
- Bunlarla birlikte yemek türü, yemek isimleri gibi değişkenlere göre incelendiğinde, menüde şehriye çorbasının olması modelde önemli görülmüştür (Şekil 3-18).
- Ana yemek türünün etli olması (kırmızı et) modelde önemli olarak analiz edilmiştir (Şekil 3-19).

Öznitelik önem dereceleri analizinin catering sektöründe faaliyet gösteren işletmelere; hava sıcaklığı, hava durumu, menü içeriği, haftanın günü gibi bağımsız değişkenlere göre yemek tüketim sayısı tahminin doğruluk düzeylerinin artırılmasında katkısının olacağı düşünülmektedir.

Çalışma kapsamında kullanılan ARIMA yönteminde katsayıların önemlilik testleri yapılmıştır (EK 3, EK 4, EK 5, EK 6, EK 7, EK 8, EK 9, EK 10, EK 11).

Veri setlerine göre geliştirilen modellerde en iyi performans göstergelerinin elde edildiği modeller Tablo 4.11' de sunulmuştur. Buna göre;

- İlk 9 veri setinin 4' ünde ROA yöntemi, 2' sinde RNN yöntemi, 2' sinde GRU yöntemi ve 1' inde ARIMA yöntemi daha üstün performans değerleri ortaya çıkarmıştır.
- RNN ve GRU yöntemlerinin temel olarak YSA çatısı altında olduğu düşünüldüğünde, YSA yöntemlerinin 4 veri setinde üstün olduğu söylenebilmektedir.
- Sonuç olarak 4 veri setinde ROA yöntemi, 4 veri setinde YSA yöntemleri üstün çıkmıştır.
- ROA yöntemi ve YSA yöntemleri kullanılarak geliştirilen modellerden elde edilen MAPE değerlerinin istatistiki değerlerinin kıyaslaması Tablo 5.1' de sunulmuştur. Buna

[10] One-Hot Encoding yapıldıktan sonra elde edilen bağımsız değişken.

göre; *YSA yöntemleri, ortalama, minimum ve maksimum MAPE değerlerinde daha üstün performans göstermiştir.*

Tablo 5.1: ROA ve YSA yöntemleri MAPE istatistikleri kıyaslaması.

İstatistik	**ROA**	**YSA**
Ortalama	0,054	0,034
Standart sapma	0,020	0,018
Minimum	0,028	0,021
Maksimum	0,071	0,060

Tablo 5.2: Literatür ve Mevcut çalışma MAPE istatistikleri kıyaslaması.

İstatistik	**Literatür**	**Mevcut çalışma**
Ortalama	0,1185	0,045
Standart sapma	0,108	0,018
Minimum	0,038	0,02
Maksimum	0,3556	0,071

Literatür taramasında oluşturulan tablodaki (Tablo 2.1) MAPE değerleri göz önüne alındığında, bu çalışma kapsamında oluşturulan veri setleri ve kullanılan yöntemler ile elde edilen MAPE değerlerinin daha iyi sonuçlar verdiği söylenebilmektedir. Tablo 5.2'de görüleceği üzere;

- Literatürde yer alan çalışmaların ortalama MAPE değeri 0,1185 iken mevcut çalışmada bulunan MAPE değerlerinin ortalaması 0,045 olarak ölçülmüştür.
- Literatürde yer alan çalışmalarda minimum MAPE değeri 0,038 iken mevcut çalışmada 0,02 olarak ölçülmüştür.
- Literatürde yer alan çalışmalarda maksimum MAPE değeri 0,3556 iken mevcut çalışmada 0,071 olarak ölçülmüştür.

Literatürde kullanılan yöntemler de incelenmiştir. Bu çalışmada kullanılan ROA, RNN, LSTM ve GRU yöntemleri yemek tüketim tahmini konusunda ilk defa kullanılmıştır. Dolayısıyla yapılan çalışma, literatürde yapılan benzer çalışmalara göre hem kullanılan daha gelişmiş

algoritmalar hem de elde edilen performans göstergeleri ile daha üstün görünmektedir. Literatürde yer alan çalışmalara göre mevcut çalışmanın performans göstergelerinde elde edilen üstünlüğünde literatüre göre farklı veri setlerinin kullanılması şartları altında gerçekleştiği unutulmamalıdır. Mevcut çalışma için gelecekte yapılacak çalışmalarda daha fazla örnek sayılarının olduğu veri setleri ile daha iyi sonuçlar elde edilebileceği düşünülmektedir. Catering sektöründe faaliyet gösteren işletmeler, mevcut çalışmada çerçevesi çizilen yaklaşımı kullanarak hizmet yönünden ve maliyet avantajı yönünden faaliyetlerine katkıda bulunabileceği düşünülmektedir.

Catering sektöründe faaliyet gösteren işletmeler; ROA ve YSA yöntemlerinden olan RNN, LSTM, GRU yöntemlerinden birini kullanarak günlük yemek tüketim sayılarını belirleyebilecekleri düşünülmektedir. Ancak bunu yaparken veri setlerinin en azından 50 günden fazla olması, varsa özel gün çıkarıldıktan sonra haftanın günlerine göre tahmin çalışmalarının yapılması önerilmektedir. Örneğin tahmin yapılan gün pazartesi ise, kullanılması önerilen veri setinin sadece pazartesi günlerini içermesi önerilmektedir. Diğer günler için yapılacak tahminlerde de veri setlerinin aynı şekilde olması önerilmektedir.

KAYNAKLAR

Blecher, L. & Yeh, R. J., 2008. Forecasting Meal Participation in University Residential Dining Facilities. *Journal of Foodservice Business Research,* 11(4), pp. 352-362.

Bozkır, A. S. & Sezer, E. A., 2010. *Predicting food demand in food courts by decision tree approaches.* Amsterdam, Elsevier Ltd, p. 759–763.

Brownlee, J., 2016. *How to Check if Time Series Data is Stationary with Python.* [Çevrimiçi] Available at: https://machinelearningmastery.com/time-series-data-stationary-python/ [Erişildi: 20 3 2021].

Brownlee, J., 2016. *Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras.* [Çevrimiçi] Available at: https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/ [Erişildi: 21 2 2021].

Calp, M. H., 2019. İşletmeler için personel yemek talep miktarının yapay sinir ağları kullanılarak tahmin edilmesi. *Politeknik Dergisi,* 22(3), pp. 675-686.

Chatfield, C., 2005. *The Analysis of Time Series: An Introduction.* 6. baskı. New York: Chapman & Hall / CRC.

Chauhan, N. S., 2020. *Decision Tree Algorithm, Explained.* [Çevrimiçi] Available at: https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html. [Erişildi: 22 3 2021].

Cho, K. ve diğerleri, 2014. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.* [Çevrimiçi] Available at: https://arxiv.org/abs/1406.1078 [Erişildi: 1 4 2021].

Çetinkaya, Z. & Erdal, E., 2019. *Daily Food Demand Forecast with Artificial Neural Networks: Kırıkkale University Case.* Samsun, IEEE, pp. 233-238.

Çoban, F., 2019. *Belirsiz koşullar altında talep tahmini ve gıda işletmesinde bir uygulama,* Yüksek Lisans Tezi: Pamukkale Üniversitesi.

Dong, J. ve diğerleri, 2018. *Research on Catering Business Demand Forecasting Model Based on the Weather Sensitivity Theory and GBDT Algorithm.* Shanghai, DEStech.

Elman, J. L., 1990. Finding Structure in Time. *Cognitive Science,* 14(2), pp. 179-312.

Erica, 2021. *Introduction to the Fundamentals of Time Series Data and Analysis.* [Çevrimiçi] Available at: https://www.aptech.com/blog/introduction-to-the-fundamentals-of-time-series-data-and-analysis/
[Erişildi: 7. 4. 2021].

Géron, A., 2017. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. 1. baskı. Sebastopol: O'Reilly Media.

Güzel, Y., 2018. *Yapay Sinir Ağlarının Yapısı.* [Çevrimiçi] Available at: https://medium.com/@yasinguzel/yapay-zeka-ders-notlar%C4%B1-04-yapay-sinir-a%C4%9Flar%C4%B1n%C4%B1n-yap%C4%B1s%C4%B1-6b9bfc3004bf
[Erişildi: 25 3 2021].

Hurwitz, J. & Kirsch, D., 2018. *Machine Learning for Dummies.* Sınırlı baskı. New Jersey USA: John Wiley & Sons.

Ilie, O.-D., Ciobica, A. & Doroftei, B., 2020. Testing the Accuracy of the ARIMA Models in Forecasting the Spreading of COVID-19 and the Associated Mortality Rate. *Medicina,* 56(11), p. 566.

Kılıç, F., Akkaya, M. R. & Memili, N., 2018. Yemekhane için Yapay Zeka Teknikleri Kullanımı ile Günlük Talep Tahmini. *Avrupa Bilim ve Teknoloji Dergisi,* Issue 13, pp. 65-71.

Kılıç, G., 2015. *Yapay Sinir Ağları İle Yemekhane Günlük Talep Tahmini,* Yüksek Lisans Tezi: Pamukkale Üniversitesi.

Kızrak, M. A., 2020. *Yinelemeli Sinir Ağlarına Giriş - RNN*. [Çevrimiçi] Available at: https://www.youtube.com/watch?v=AHybjSLXlV8
[Erişildi: 1. 4. 2021].

Koehrsen, W., 2018. *Hyperparameter Tuning the Random Forest in Python.* [Çevrimiçi] Available at: https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
[Erişildi: 26 1 2021].

Kostadinov, S., 2017. *Understanding GRU Networks.* [Çevrimiçi] Available at: https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be [Erişildi: 3. 4. 2021].

Kurt, R. & Karayılmazlar, S., 2019. Türkiye mantar üretimi ve ARIMA (Box-Jenkins) ile projeksiyonu. *Ormancılık Araştırma Dergisi,* 6(1), pp. 72-76.

Malefors, C., Strid, I., Hansson, P.-A. & Eriksson, M., 2021. Potential for using guest attendance forecasting in Swedish public catering to reduce overcatering. *Sustainable Production and Consumption,* 25(January 2021), pp. 162-172.

Miller, J. J., McCahon, C. S. & Miller, J. L., 1993. Foodservice Forecasting: Differences in Selection of Simple Mathematical Models Based on Short- Term and Long-Term Data Sets. *Journal of Hospitality & Tourism Research,* 16(2), pp. 93-102.

Montgomery, D. C., Jennings, C. L. & Kulahci, M., 2008. *Introduction to Time Series Analysis and Forecasting.* New Jersey: John Wiley & Sons. Inc..

Nagy, Z., 2018. *Artificial Intelligence and Machine Learning Fundamentals.* 1. baskı. Birmingham: Packt Publishing.

Olah, C., 2015. *Understanding LSTM Networks.* [Çevrimiçi] Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Erişildi: 3. 4. 2021].

Olsen, M. D. & Jose, M. L., 1982. Time-series forecasting: a testing of applications to the food-service industry. *International Journal of Hospitality Management,* 1, 1(3), pp. 151-156.

Phi, M., 2018. *Illustrated Guide to LSTM's and GRU's: A step by step explanation.* [Çevrimiçi] Available at: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-

step-explanation-44e9eb85bf21
[Erişildi: 2. 4. 2021].

Radchenko, V., Kashnitsky, Y. & Parshutsich, M., 2020. *Ensembles and random forest.* [Çevrimiçi]
Available at: https://mlcourse.ai/articles/topic5-part3-feature-importance/#mlcourse.ai-%E2%80%93-Open-Machine-Learning-Course
[Erişildi: 23. 3. 2021].

Ryu, K. P., Jang, S. S. P. & Sanchez, A. P., 2003. Forecasting Methods and Seasonal Adjustment for a University Foodservice Operation. *Journal of Foodservice Business Research,* 6(2), pp. 17-34.

Ryu, K. & Sanchez, A., 2003. The Evaluation of Forecasting Methods at an Institutional Foodservice Dining Facility. *The Journal of Hospitality Financial Management,* 11(1), pp. 27-45.

Sarkar, D. & Natarajan, V., 2019. *Ensemble Machine Learning Cookbook.* 1. baskı. Birmingham: Packt Publishing.

Serengil, Ş. İ., 2020. *Karar Ağaçlarında Öznitelik Önem Sıralaması.* [Çevrimiçi] Available at: https://bilisim.io/2020/04/27/karar-agaclarinda-oznitelik-onem-siralamasi/ [Erişildi: 10. 4. 2021].

Soylu, V. L., 2017. *Reinforcement Learning (Pekiştirmeli Öğrenme).* [Çevrimiçi] Available at: https://medium.com/@wintermute_/reinforcement-learning-peki%C5%9Ftirmeli-%C3%B6%C4%9Frenme-i%CC%87nsan-beyniyle-aradaki-fark-kapan%C4%B1rken-c0d0a6f7c2e8
[Erişildi: 30. 3. 2021].

Şirin, E., 2021. *Ensemble Yöntemler (Topluluk Öğrenmesi): Basit Teorik Anlatım ve Python Uygulama.* [Çevrimiçi]
Available at: https://www.veribilimiokulu.com/blog/ensemble-yontemler-topluluk-ogrenmesi-basit-teorik-anlatim-ve-python-uygulama/
[Erişildi: 19. 3. 2021].

T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü, 2020. *Toplu Beslenme Sistemleri (Toplu Tüketim Yerleri) İçin Ulusal Menü Planlama ve Uygulama Rehberi.* Yayın No: 1184. Ankara: T.C. Sağlık Bakanlığı Halk Sağlığı Genel Müdürlüğü Sağlıklı Beslenme.

Tanyaş, P. D. M. & Baskak, Y. D. D. M., 2008. *Üretim Planlama ve Kontrol.* 3. baskı. İstanbul: İrfan Yayımcılık.

Yergök, D. & Acı, M., 2019. Toplu Yemek Üretiminde Günlük Talep Tahmini için Alternatif Bir Yaklaşım: Öğrenci Regresyon. *Avrupa Bilim ve Teknoloji Dergisi,* Özel sayı(2019), pp. 64-73.

Yeşilkaynak, V. B., 2020. *Yapay Zeka, Makine Öğrenmesi ve Derin Öğrenme Kavramları Arasındaki Fark Nedir?.* [Çevrimiçi] Available at: https://evrimagaci.org/yapay-zeka-makine-ogrenmesi-ve-derin-ogrenme-kavramlari-arasindaki-fark-nedir-8889
[Erişildi: 18. 3. 2021].

Yıldırım, E., 2020. *Yapay Sinir Ağı (Artificial Neural Network) Nedir?.* [Çevrimiçi] Available at: https://www.veribilimiokulu.com/blog/yapay-sinir-agiartificial-neural-network-nedir/
[Erişildi: 25. 3. 2021].

Zhou, Z.-H., 2012. *Ensemble Methods Foundations and Algorithms.* Florida: Taylor & Francis Group.

EKLER

EK 1: Genel veri seti (233 gün).

Tarih	Gun	Mevsim	RamazanGu	FinalSir	ButSina	Somest	YazTati	HavaSic	HavaDu	Yemek	Yemek	Yemek	Yemek	Yemek	Yemek	Yemek
3.Oca.19	Persembe	Kis	Hayir	Evet	Hayir	Hayir	Hayir	7	Yagmur	Sebze	Sote	Tavuk	Makarna	Meyve	2.827	
4.Oca.19	Cuma	Kis	Hayir	Evet	Hayir	Hayir	Hayir	4	Karli	Ezogelin	Kofte	Et	Salata	SerbetliTa	3.047	
7.Oca.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	1	Karli	Sehriye	Bezelye	Et	Pilav	Yogurt	2.576	
8.Oca.19	Sali	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	0	Karli	Diger	Balik	Balik	Salata	Helva	2.907	
9.Oca.19	Carsamba	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	3	Yagmur	Prenses	Kofte	Et	Pilaki	SerbetliTa	3.513	
10.Oca.19	Persembe	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	9	Yagmur	Mercimek	Firin	Tavuk	Pilav	Komposto	3.746	
11.Oca.19	Cuma	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	9	Yagmur	Yayla	Kebap	Et	Salata	SerbetliTa	3.201	
14.Oca.19	Pazartesi	Kis	Hayir	Hayir	Evet	Evet	Hayir	6	Yagmur	Sebze	Mercimek	Et	Pilav	Meyve	3.548	
15.Oca.19	Sali	Kis	Hayir	Hayir	Evet	Evet	Hayir	5	Yagmur	Mercimek	Kulbasti	Tavuk	Pilav	Yogurt	4.523	
16.Oca.19	Carsamba	Kis	Hayir	Hayir	Evet	Evet	Hayir	3	Gunesli	Diger	Kofte	Et	Salata	SutluTatli	4.070	
17.Oca.19	Persembe	Kis	Hayir	Hayir	Evet	Evet	Hayir	7	Gunesli	Diger	Oturtma	Et	Havuc	SerbetliTa	3.329	
18.Oca.19	Cuma	Kis	Hayir	Hayir	Evet	Evet	Hayir	8	Gunesli	Domates	Kebap	Et	Makarna	Meyve	3.334	
21.Oca.19	Pazartesi	Kis	Hayir	Hayir	Evet	Evet	Hayir	9	Yagmur	Prenses	Nohut	Et	Pilav	Ayran	3.515	
22.Oca.19	Sali	Kis	Hayir	Hayir	Evet	Evet	Hayir	8	Bulutlu	Domates	Tavuk	Tavuk	Manti	Meyve	3.879	
23.Oca.19	Carsamba	Kis	Hayir	Hayir	Evet	Evet	Hayir	10	Yagmur	Mercimek	Turlu	Et	Borek	Yogurt	3.738	
24.Oca.19	Persembe	Kis	Hayir	Hayir	Evet	Evet	Hayir	11	Yagmur	Sebze	Kofte	Et	Makarna	SutluTatli	3.570	
25.Oca.19	Cuma	Kis	Hayir	Hayir	Evet	Evet	Hayir	10	Yagmur	Mantar	Kebap	Et	Pilav	Meyve	3.000	
28.Oca.19	Pazartesi	Kis	Hayir	Hayir	Evet	Evet	Hayir	9	Bulutlu	Ezogelin	Sinitzel	Tavuk	Makarna	Yogurt	3.427	
29.Oca.19	Sali	Kis	Hayir	Hayir	Evet	Evet	Hayir	13	Yagmur	Sehriye	Ispanak	Et	Eriste	SerbetliTa	2.730	
30.Oca.19	Carsamba	Kis	Hayir	Hayir	Evet	Evet	Hayir	13	Yagmur	Mercimek	Sote	Tavuk	Pilav	Meyve	3.146	
31.Oca.19	Persembe	Kis	Hayir	Hayir	Evet	Evet	Hayir	11	Yagmur	Domates	Doner	Et	Salata	SutluTatli	3.325	
1.Şub.19	Cuma	Kis	Hayir	Hayir	Hayir	Evet	Hayir	9	Yagmur	Sebze	Kofte	Et	Makarna	Meyve	2.442	
4.Şub.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Evet	Hayir	13	Gunesli	Prenses	Nohut	Et	Pilav	SerbetliTa	3.309	
5.Şub.19	Sali	Kis	Hayir	Hayir	Hayir	Evet	Hayir	10	Bulutlu	Ezogelin	Tavuk	Tavuk	Manti	Meyve	3.305	
6.Şub.19	Carsamba	Kis	Hayir	Hayir	Hayir	Evet	Hayir	8	Yagmur	Sehriye	Turlu	Et	Pilav	Meyve	2.866	
7.Şub.19	Persembe	Kis	Hayir	Hayir	Hayir	Evet	Hayir	6	Yagmur	Diger	Kizartma	Et	Salata	SerbetliTa	3.114	
8.Şub.19	Cuma	Kis	Hayir	Hayir	Hayir	Evet	Hayir	6	Yagmur	Mantar	Kofte	Et	Pilaki	Meyve	2.790	
11.Şub.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	8	Gunesli	Mercimek	Bezelye	Et	Makarna	SerbetliTa	8.182	
12.Şub.19	Sali	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	11	Yagmur	Domates	Kebap	Et	Pilav	Yogurt	9.665	

EK 2: Genel veri seti (*BirOncekiGun* ve *GecenHaftaAyniGun*) (230 gün).

Tarih	Gun	Mevsim	RamazanGunu	FinalSinavGunu	ButSinavGunu	SomestrTatili	YazTatili	HavaSica	HavaDurumu	Yemek_1	Yemek_2	Yemek_2_Tur	Yemek_3	Yemek_4	YemekTuketim	BirOncekiGu	GecenHaftaAyniG
9.Oca.19	Carsamba	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	3	Yagmur	Prenses	Kofte	Et	Pilaki	SerbetliTatli	3.513	2.907	492
10.Oca.19	Persembe	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	9	Yagmur	Mercimek	Firin	Tavuk	Pilav	Komposto	3.746	3.513	2.827
11.Oca.19	Cuma	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	9	Yagmur	Yayla	Kebap	Et	Salata	SerbetliTatli	3.201	3.746	3.047
14.Oca.19	Pazartesi	Kis	Hayir	Hayir	Evet	Evet	Hayir	6	Yagmur	Sebze	Mercimek	Et	Pilav	Meyve	3.548	3.201	2.576
15.Oca.19	Sali	Kis	Hayir	Hayir	Evet	Evet	Hayir	5	Yagmur	Mercimek	Kulbasti	Tavuk	Pilav	Yogurt	4.523	3.548	2.907
16.Oca.19	Carsamba	Kis	Hayir	Hayir	Evet	Evet	Hayir	3	Gunesli	Diger	Kofte	Et	Salata	SutluTatli	4.070	4.523	3.513
17.Oca.19	Persembe	Kis	Hayir	Hayir	Evet	Evet	Hayir	7	Gunesli	Diger	Oturtma	Et	Havuc	SerbetliTatli	3.329	4.070	3.746
18.Oca.19	Cuma	Kis	Hayir	Hayir	Evet	Evet	Hayir	8	Gunesli	Domates	Kebap	Et	Makarna	Meyve	3.334	3.329	3.201
21.Oca.19	Pazartesi	Kis	Hayir	Hayir	Evet	Evet	Hayir	9	Yagmur	Prenses	Nohut	Et	Pilav	Ayran	3.515	3.334	3.548
22.Oca.19	Sali	Kis	Hayir	Hayir	Evet	Evet	Hayir	8	Bulutlu	Domates	Tavuk	Tavuk	Manti	Meyve	3.879	3.515	4.523
23.Oca.19	Carsamba	Kis	Hayir	Hayir	Evet	Evet	Hayir	10	Yagmur	Mercimek	Turlu	Et	Borek	Yogurt	3.738	3.879	4.070
24.Oca.19	Persembe	Kis	Hayir	Hayir	Evet	Evet	Hayir	11	Yagmur	Sebze	Kofte	Et	Makarna	SutluTatli	3.570	3.738	3.329
25.Oca.19	Cuma	Kis	Hayir	Hayir	Evet	Evet	Hayir	10	Yagmur	Mantar	Kebap	Et	Pilav	Meyve	3.000	3.570	3.334
28.Oca.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Evet	Hayir	9	Bulutlu	Ezogelin	Sinitzel	Tavuk	Makarna	Yogurt	3.427	3.000	3.515
29.Oca.19	Sali	Kis	Hayir	Hayir	Hayir	Evet	Hayir	13	Yagmur	Sehriye	Ispanak	Et	Eriste	SerbetliTatli	2.730	3.427	3.879
30.Oca.19	Carsamba	Kis	Hayir	Hayir	Hayir	Evet	Hayir	13	Yagmur	Mercimek	Sote	Tavuk	Pilav	Meyve	3.146	2.730	3.738
31.Oca.19	Persembe	Kis	Hayir	Hayir	Hayir	Evet	Hayir	11	Yagmur	Domates	Doner	Et	Salata	SutluTatli	3.325	3.146	3.570
1.Sub.19	Cuma	Kis	Hayir	Hayir	Hayir	Evet	Hayir	9	Yagmur	Sebze	Kofte	Et	Makarna	Meyve	2.442	3.325	3.000
4.Sub.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Evet	Hayir	13	Gunesli	Prenses	Nohut	Et	Pilav	SerbetliTatli	3.309	2.442	3.427
5.Sub.19	Sali	Kis	Hayir	Hayir	Hayir	Evet	Hayir	10	Bulutlu	Ezogelin	Tavuk	Tavuk	Manti	Meyve	3.305	3.309	2.730
6.Sub.19	Carsamba	Kis	Hayir	Hayir	Hayir	Evet	Hayir	8	Yagmur	Sehriye	Turlu	Et	Pilav	Meyve	2.866	3.305	3.146
7.Sub.19	Persembe	Kis	Hayir	Hayir	Hayir	Evet	Hayir	6	Yagmur	Diger	Kizartma	Et	Salata	SerbetliTatli	3.114	2.866	3.325
8.Sub.19	Cuma	Kis	Hayir	Hayir	Hayir	Evet	Hayir	6	Yagmur	Mantar	Kofte	Et	Pilaki	Meyve	2.790	3.114	2.442
11.Sub.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	8	Gunesli	Mercimek	Bezelye	Et	Makarna	SerbetliTatli	8.182	2.790	3.309
12.Sub.19	Sali	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	11	Yagmur	Domates	Kebap	Et	Pilav	Yogurt	9.665	8.182	3.305
13.Sub.19	Carsamba	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	7	Yagmur	Sebze	Kofte	Et	Havuc	SerbetliTatli	8.305	9.665	2.866
14.Sub.19	Persembe	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	6	Yagmur	Ezogelin	Kulbasti	Tavuk	Salata	Meyve	8.976	8.305	3.114
15.Sub.19	Cuma	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	6	Yagmur	Diger	Fasulye	Et	Pilav	Ayran	6.311	8.976	2.790
18.Sub.19	Pazartesi	Kis	Hayir	Hayir	Hayir	Hayir	Hayir	6	Gunesli	Ezogelin	Oturtma	Et	ispanak	Asure	10.219	6.311	8.182

EK 3: Veri seti 1 ARIMA (4,1,4) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	209
Model:	ARIMA(4, 1, 4)	Log Likelihood	-1768.061
Date:	Mon, 07 Jun 2021	AIC	3554.123
Time:	21:21:19	BIC	3584.161
Sample:	0	HQIC	3566.269
	- 209		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.9062	0.062	-14.658	0.000	-1.027	-0.785
ar.L2	-0.8207	0.084	-9.723	0.000	-0.986	-0.655
ar.L3	-0.7802	0.070	-11.086	0.000	-0.918	-0.642
ar.L4	-0.8327	0.043	-19.511	0.000	-0.916	-0.749
ma.L1	0.6268	0.087	7.199	0.000	0.456	0.797
ma.L2	0.3118	0.102	3.046	0.002	0.111	0.512
ma.L3	0.2949	0.106	2.772	0.006	0.086	0.503
ma.L4	0.4108	0.092	4.446	0.000	0.230	0.592
sigma2	1.533e+06	8.94e+04	17.140	0.000	1.36e+06	1.71e+06

Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	294.35
Prob(Q):	0.92	Prob(JB):	0.00
Heteroskedasticity (H):	0.92	Skew:	1.22
Prob(H) (two-sided):	0.74	Kurtosis:	8.29

EK 4: Veri seti 2 ARIMA (4,0,5) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	117
Model:	ARIMA(4, 0, 5)	Log Likelihood	-1023.883
Date:	Mon, 07 Jun 2021	AIC	2069.766
Time:	21:31:51	BIC	2100.150
Sample:	0	HQIC	2082.102
	- 117		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	8875.1362	194.380	45.659	0.000	8494.158	9256.115
ar.L1	-0.6756	0.062	-10.814	0.000	-0.798	-0.553
ar.L2	-0.6369	0.081	-7.905	0.000	-0.795	-0.479
ar.L3	-0.6564	0.068	-9.618	0.000	-0.790	-0.523
ar.L4	-0.8681	0.054	-16.113	0.000	-0.974	-0.763
ma.L1	1.1915	0.140	8.540	0.000	0.918	1.465
ma.L2	0.9305	0.113	8.226	0.000	0.709	1.152
ma.L3	0.9588	0.243	3.945	0.000	0.482	1.435
ma.L4	1.2716	0.307	4.138	0.000	0.669	1.874
ma.L5	0.7143	0.141	5.050	0.000	0.437	0.992
sigma2	1.658e+06	3.28e+05	5.056	0.000	1.02e+06	2.3e+06

Ljung-Box (L1) (Q):	1.15	Jarque-Bera (JB):	13.11
Prob(Q):	0.28	Prob(JB):	0.00
Heteroskedasticity (H):	0.20	Skew:	-0.74
Prob(H) (two-sided):	0.00	Kurtosis:	3.72

EK 5: Veri seti 3 ARIMA (1,1,1) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	20
Model:	ARIMA(1, 1, 1)	Log Likelihood	-173.299
Date:	Mon, 07 Jun 2021	AIC	352.597
Time:	21:40:54	BIC	355.430
Sample:	0	HQIC	353.077
	- 20		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.9654	0.182	5.303	0.000	0.609	1.322
ma.L1	-0.9990	0.276	-3.619	0.000	-1.540	-0.458
sigma2	4.848e+06	6.17e-08	7.86e+13	0.000	4.85e+06	4.85e+06

Ljung-Box (L1) (Q):	2.52	Jarque-Bera (JB):	1.70
Prob(Q):	0.11	Prob(JB):	0.43
Heteroskedasticity (H):	0.08	Skew:	0.51
Prob(H) (two-sided):	0.01	Kurtosis:	4.05

EK 6: Veri seti 4 ARIMA (4,4,0) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	20
Model:	ARIMA(4, 4, 0)	Log Likelihood	-151.612
Date:	Mon, 07 Jun 2021	AIC	313.224
Time:	21:43:21	BIC	317.087
Sample:	0	HQIC	313.422
	- 20		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-1.7905	0.209	-8.553	0.000	-2.201	-1.380
ar.L2	-1.8103	0.272	-6.662	0.000	-2.343	-1.278
ar.L3	-1.3879	0.277	-5.006	0.000	-1.931	-0.844
ar.L4	-0.6319	0.237	-2.667	0.008	-1.096	-0.168
sigma2	8.661e+06	8.54e-09	1.01e+15	0.000	8.66e+06	8.66e+06

Ljung-Box (L1) (Q):	0.71	Jarque-Bera (JB):	1.51
Prob(Q):	0.40	Prob(JB):	0.47
Heteroskedasticity (H):	0.37	Skew:	-0.71
Prob(H) (two-sided):	0.30	Kurtosis:	2.49

EK 7: Veri seti 5 ARIMA (1,0,2) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	20
Model:	ARIMA(1, 0, 2)	Log Likelihood	-176.373
Date:	Mon, 07 Jun 2021	AIC	362.746
Time:	21:44:55	BIC	367.724
Sample:	0	HQIC	363.718
	- 20		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	9633.8922	2048.325	4.703	0.000	5619.249	1.36e+04
ar.L1	0.8363	0.227	3.676	0.000	0.390	1.282
ma.L1	-1.1832	0.335	-3.534	0.000	-1.839	-0.527
ma.L2	0.9980	0.314	3.177	0.001	0.382	1.614
sigma2	2.019e+06	0.003	5.91e+08	0.000	2.02e+06	2.02e+06

Ljung-Box (L1) (Q):	1.65	Jarque-Bera (JB):	1.54
Prob(Q):	0.20	Prob(JB):	0.46
Heteroskedasticity (H):	0.17	Skew:	-0.34
Prob(H) (two-sided):	0.03	Kurtosis:	4.18

EK 8: Veri seti 6 ARIMA (3,0,5) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	21
Model:	ARIMA(3, 0, 5)	Log Likelihood	-1377.053
Date:	Mon, 07 Jun 2021	AIC	2774.105
Time:	21:46:31	BIC	2784.550
Sample:	0	HQIC	2776.372
	- 21		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
const	9279.5611	0.002	5.37e+06	0.000	9279.558	9279.564
ar.L1	0.9900	0.004	241.326	0.000	0.982	0.998
ar.L2	0.9880	0.004	224.818	0.000	0.979	0.997
ar.L3	-0.9993	0.001	-870.591	0.000	-1.002	-0.997
ma.L1	-0.9908	0.070	-14.078	0.000	-1.129	-0.853
ma.L2	-1.9867	0.031	-64.832	0.000	-2.047	-1.927
ma.L3	1.9883	0.026	75.838	0.000	1.937	2.040
ma.L4	0.9885	0.077	12.795	0.000	0.837	1.140
ma.L5	-0.9992	0.004	-244.546	0.000	-1.007	-0.991
sigma2	2.25e+04	2.47e-06	9.1e+09	0.000	2.25e+04	2.25e+04

Ljung-Box (L1) (Q):	2.91	Jarque-Bera (JB):	0.07
Prob(Q):	0.09	Prob(JB):	0.96
Heteroskedasticity (H):	0.38	Skew:	-0.11
Prob(H) (two-sided):	0.22	Kurtosis:	3.18

EK 9: Veri seti 7 ARIMA (3,2,1) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	21
Model:	ARIMA(3, 2, 1)	Log Likelihood	-161.137
Date:	Mon, 07 Jun 2021	AIC	332.274
Time:	21:49:58	BIC	336.997
Sample:	0	HQIC	333.074
	- 21		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-1.3098	0.224	-5.845	0.000	-1.749	-0.871
ar.L2	-1.0922	0.343	-3.185	0.001	-1.764	-0.420
ar.L3	-0.7461	0.181	-4.122	0.000	-1.101	-0.391
ma.L1	0.7033	0.350	2.010	0.044	0.018	1.389
sigma2	1.267e+06	5.82e+05	2.175	0.030	1.25e+05	2.41e+06

Ljung-Box (L1) (Q):	0.02	Jarque-Bera (JB):	0.66
Prob(Q):	0.88	Prob(JB):	0.72
Heteroskedasticity (H):	0.35	Skew:	-0.46
Prob(H) (two-sided):	0.23	Kurtosis:	3.00

EK 10: Veri seti 8 ARIMA (4,1,0) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	15
Model:	ARIMA(4, 1, 0)	Log Likelihood	-109.278
Date:	Mon, 07 Jun 2021	AIC	228.557
Time:	21:54:03	BIC	231.752
Sample:	0	HQIC	228.261
	- 15		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.8372	0.181	-4.634	0.000	-1.191	-0.483
ar.L2	-0.7127	0.269	-2.653	0.008	-1.239	-0.186
ar.L3	-0.7124	0.174	-4.103	0.000	-1.053	-0.372
ar.L4	-0.6291	0.167	-3.771	0.000	-0.956	-0.302
sigma2	2.735e+05	1.22e+05	2.245	0.025	3.48e+04	5.12e+05

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2.42
Prob(Q):	0.97	Prob(JB):	0.30
Heteroskedasticity (H):	0.65	Skew:	-1.00
Prob(H) (two-sided):	0.65	Kurtosis:	3.38

EK 11: Veri seti 9 ARIMA (0,1,0) p değerleri.

SARIMAX Results

Dep. Variable:	y	No. Observations:	15
Model:	ARIMA(0, 1, 0)	Log Likelihood	-107.899
Date:	Mon, 07 Jun 2021	AIC	217.797
Time:	21:55:18	BIC	218.437
Sample:	0	HQIC	217.738
	- 15		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
sigma2	2.884e+05	1.15e+05	2.508	0.012	6.3e+04	5.14e+05

Ljung-Box (L1) (Q):	0.97	Jarque-Bera (JB):	1.85
Prob(Q):	0.32	Prob(JB):	0.40
Heteroskedasticity (H):	1.04	Skew:	0.89
Prob(H) (two-sided):	0.96	Kurtosis:	2.93

EK 12: Rassal Orman Algoritması Kaynak Kodu (Koehrsen, 2018).

```
# -*- coding: utf-8 -*-
"""
Created on Fri Jan  1 16:15:30 2021

@author: omer
"""
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score,mean_squared_error,
mean_absolute_error
from six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
import os
from pandas import datetime
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt
from matplotlib import pyplot
import seaborn as sns
import math
import pickle

data = pd.read_excel('catering_data - Kopya -
SpecialDays.xlsx',sheet_name='Catering_butun')
data = data.set_index('Tarih')

data = pd.read_excel('catering_dataa.xlsx',sheet_name='catering_data')
data = data.set_index('Tarih')

data_encoded = pd.get_dummies(data, columns=['Gun',
```

```
                                                'Mevsim',
                                                'RamazanGunu',
                                                'FinalSinavGunu',
                                                'ButSinavGunu',
                                                'SomestrTatili',
                                                'YazTatili',
                                                'HavaDurumu',
                                                'Yemek_1',
                                                'Yemek_2',
                                                'Yemek_2_Tur',
                                                'Yemek_3',
                                                'Yemek_4'
                                                ])

y = data_encoded['YemekTuketim'].copy()
X = data_encoded.drop('YemekTuketim',axis=1).copy()

################ X ve y MinMax Normalizasyonu ########

min_max_scaler_X = MinMaxScaler(feature_range=(0, 1))
min_max_scaler_y = MinMaxScaler(feature_range=(0, 1))
X_values = X.values
y_values = y.values.reshape(len(y),1)
X_Norm = min_max_scaler_X.fit_transform(X_values)
y_Norm = min_max_scaler_y.fit_transform(y_values)
y_Norm = y_Norm.reshape(len(y_Norm),)

#########################################################

trees = [int(x) for x in np.linspace(start = 1000 , stop = 10000, num =
19)]
perfCri_list=[[],[],[],[],[],[]]
perfCri_list[0]= trees
trainTest_list = [[],[],[],[],[],[]]
trainTest_list[0]= trees

for tree in trees:

    def mape(actual, pred):
        actual, pred = np.array(actual), np.array(pred)
        absolute=[]
        for i,j in zip(actual,pred):
            av=abs((i-j)/i)
            absolute.append(av)
        return np.mean(absolute)

    Xtrain, Xtest, ytrain, ytest =
train_test_split(X_Norm,y_Norm,test_size=0.25, shuffle=True)
    ytest_rev = ytest.reshape(len(ytest),1)
    ytest_rev2 = min_max_scaler_y.inverse_transform(ytest_rev)
    ytest_rev2 = ytest_rev2.reshape(len(ytest_rev2),)

    regr = RandomForestRegressor(random_state = 42,
                                 bootstrap = True,
                                 oob_score=True,
```

```
                                max_depth=None,
                                max_features='auto',
                                #min_samples_leaf=1,
                                #min_samples_split=5,
                                n_estimators=tree,
                                n_jobs=-1)
        regr.fit(Xtrain,ytrain)
        yPredict=regr.predict(Xtest)
        yPredict_rev = yPredict.reshape(len(yPredict),1)
        yPredict_rev = min_max_scaler_y.inverse_transform(yPredict_rev)
        yPredict_rev2 = np.around(yPredict_rev) #en yakın sayıya yuvarlar
        yPredict_rev2 = yPredict_rev2.reshape(len(yPredict_rev2),)

        mse = round(mean_squared_error(ytest_rev2, yPredict_rev2), 3)
        rmse = round(math.sqrt(mse), 3)
        mae = round(mean_absolute_error(ytest_rev2, yPredict_rev2), 3)
        r2 = round(r2_score(ytest_rev2, yPredict_rev2), 3)
        mape = round(mape(ytest_rev2, yPredict_rev2), 3)

        perfCri_list[1].append(rmse)
        perfCri_list[2].append(mse)
        perfCri_list[3].append(mae)
        perfCri_list[4].append(r2)
        perfCri_list[5].append(mape)

        trainTest_list[1].append(Xtrain)
        trainTest_list[2].append(ytrain)
        trainTest_list[3].append(Xtest)
        trainTest_list[4].append(ytest)
        trainTest_list[5].append(regr)

        # yTrue=ytest.values
        # plt.plot(yTrue, label='True')
        # plt.plot(yPredict, label='Predict' ,color='red')
        # plt.legend()
        # plt.show()

perfCri_list_df = {'tree': perfCri_list[0],
                   'RMSE': perfCri_list[1],
                   'MSE': perfCri_list[2],
                   'MAE': perfCri_list[3],
                   'R2': perfCri_list[4],
                   'MAPE': perfCri_list[5]}

perfCri_list_pd = pd.DataFrame(perfCri_list_df)

importance=[[],[]]
importance[0] = model.feature_importances_
importance[1] = X.columns.tolist()

importance_df = {'Importance': importance[0],
                 'FeatureName': importance[1]}

importance_pd=pd.DataFrame(importance_df)
```

EK 13: ARIMA Kaynak Kodu (Brownlee, 2016).

```
# -*- coding: utf-8 -*-
"""
Created on Fri Feb 19 00:58:06 2021

@author: omer
"""

import warnings
from math import sqrt
import pandas as pd
import numpy as np
from pandas import read_csv
from pandas import datetime
from statsmodels.tsa.arima.model import ARIMA, ARIMAResults
from sklearn.metrics import mean_squared_error, r2_score,
mean_absolute_error
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller


def test_stationarity(timeseries, window = 12, cutoff = 0.05):

    #Determing rolling statistics
    rolmean = timeseries.rolling(window).mean()
    rolstd = timeseries.rolling(window).std()

    #Plot rolling statistics:
    fig = plt.figure(figsize=(12, 8))
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.xlabel('Time Series')
    plt.ylabel('Values')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show()

    #Perform Dickey-Fuller test:
    print('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-
value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    pvalue = dftest[1]
    if pvalue < cutoff:
        print('p-value = %.4f. The series is likely stationary.' %
pvalue)
    else:
        print('p-value = %.4f. The series is likely non-stationary.' %
pvalue)
```

```
    print(dfoutput)

test_stationarity()

perfCri_list=[[],[],[],[],[],[],[]]
pvalue_control = [[],[]]

def evaluate_arima_model(X, arima_order):
    # prepare training dataset
    train_size = int(len(X) * 0.90)
    train, test = X[0:train_size], X[train_size:]
    history = [x for x in train]
    # make predictions
    predictions = list()

    for t in range(len(test)):
        model = ARIMA(history, order=arima_order)
        model_fit = model.fit()
        yhat = model_fit.forecast()[0]
        predictions.append(yhat)
        history.append(test[t])
    # calculate out of sample error
    rmse = sqrt(mean_squared_error(test, predictions))
    perfCri_list[0].append(arima_order)
    perfCri_list[1].append(model_fit.aic)
    perfCri_list[2].append(r2_score(test, predictions))
    perfCri_list[3].append(mean_squared_error(test, predictions))
    perfCri_list[4].append(rmse)
    perfCri_list[5].append(mean_absolute_error(test, predictions))

    model1 = ARIMA(train, order=arima_order)
    model_fit1 = model1.fit()

    def mape(actual, pred):
        actual, pred = np.array(actual), np.array(pred)
        absolute=[]
        for i,j in zip(actual,pred):
            av=abs((i-j)/i)
            absolute.append(av)
        return np.mean(absolute)

    perfCri_list[6].append(mape(test, predictions))

    model_fit1_p_values = model_fit1.pvalues

    pvalue_control[0].append(arima_order)
    pvalue_control[1].append(model_fit1_p_values)

    return rmse

# evaluate combinations of p, d and q values for an ARIMA model
def evaluate_models(dataset, p_values, d_values, q_values):
    dataset = dataset.astype('float32')
    best_score, best_cfg = float("inf"), None
    for p in p_values:
```

```
        for d in d_values:
            for q in q_values:
                order = (p,d,q)
                try:
                    rmse = evaluate_arima_model(dataset, order)
                    if rmse < best_score:
                        best_score, best_cfg = rmse, order
                    print('ARIMA%s RMSE=%.3f' % (order,rmse))
                except:
                    continue
    print('Best ARIMA%s RMSE=%.3f' % (best_cfg, best_score))

veri_Seti = "Catering_butun"

data = pd.read_excel('catering_data - Kopya -
SpecialDays.xlsx',sheet_name = veri_Seti)
dataYemekTuketim = data['YemekTuketim']

test_stationarity(dataYemekTuketim)

p_values = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
d_values = [1] # range(0, 3)
q_values = [0,1,2,3,4,5,6,7,8,9,10]

warnings.filterwarnings("ignore")
evaluate_models(dataYemekTuketim.values, p_values, d_values, q_values)

perfCri_list_df = {'pdq': perfCri_list[0],
                   'AIC': perfCri_list[1],
                   'r2': perfCri_list[2],
                   'MSE': perfCri_list[3],
                   'RMSE': perfCri_list[4],
                   'MAE': perfCri_list[5],
                   'MAPE': perfCri_list[6]}

pvalue_control_df = {'pdq': pvalue_control[0],
                     'P_values': pvalue_control[1]}
only_pvalue_pd = pd.DataFrame(pvalue_control[1])

pvalue_control_pd = pd.DataFrame(pvalue_control_df)
perfCri_list_pd = pd.DataFrame(perfCri_list_df)

perfCri_list_pd.to_excel("ARIMA_" + veri_Seti + "_perfCri_list_pd.xlsx")
pvalue_control_pd.to_excel("ARIMA_" + veri_Seti +
"_pvalue_control_pd.xlsx")
only_pvalue_pd.to_excel("ARIMA_" + veri_Seti + "_only_pvalues.xlsx")
pvalue_control[1].to_excel(veri_Seti + "only_pvalues.xlsx")
```

EK 14: YSA Kaynak Kodu (Brownlee, 2016).

```
# -*- coding: utf-8 -*-
"""
Created on Fri Jan 29 18:12:56 2021

@author: omer
"""

import numpy
import pandas as pd
import matplotlib.pyplot as plt
from pandas import read_csv, read_excel, get_dummies
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import SimpleRNN, LSTM, GRU
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
import numpy as np
from kerastuner.tuners import RandomSearch, Hyperband,
BayesianOptimization
from kerastuner import HyperModel
import kerastuner as kt
from tensorflow import keras
import tensorflow as tf
from keras.models import load_model

# convert an array of values into a dataset matrix
def create_dataset(dataset, look_back):
        dataX, dataY = [], []
        for i in range(len(dataset)-look_back-1):
                a = dataset[i:(i+look_back), 0]
                dataX.append(a)
                dataY.append(dataset[i + look_back, 0])
        return numpy.array(dataX), numpy.array(dataY)

data = pd.read_excel('catering_data - Kopya -
SpecialDays.xlsx',sheet_name='Catering_butun')
df = data[["Tarih","YemekTuketim"]]
dataframe = df.set_index('Tarih')

dataset = dataframe.values
dataset1 = dataset

scaler = MinMaxScaler(feature_range=(0, 1))
dataset = scaler.fit_transform(dataset)

def mape(actual, pred):
    actual, pred = np.array(actual), np.array(pred)
    absolute=[]
    for i,j in zip(actual,pred):
        av=abs((i-j)/i)
        absolute.append(av)
```

```
    return np.mean(absolute)


################ LOOKBACK SEARCH #####################################

results = [[],[],[],[],[],[],[]]


for a in range(1,20,1):
    print(a)
    look_back = a
    X, y = create_dataset(dataset, look_back)
    # split into train and test sets
    train_size = int(len(X) * 0.90)
    test_size = len(X) - train_size
    trainX, testX = X[0:train_size,:], X[train_size:len(X),:]
    trainY, testY = y[0:train_size,], y[train_size:len(y),]

    trainX = numpy.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
    testX = numpy.reshape(testX, (testX.shape[0], 1, testX.shape[1]))

    k_init = keras.initializers.Constant(value=0.1)
    b_init = keras.initializers.Constant(value=0)
    r_init = keras.initializers.Constant(value=0.1)


    # create and fit the RNN, LSTM or GRU network
    model = Sequential()
    model.add(SimpleRNN(units=2, input_shape=(1, look_back),
kernel_initializer = k_init, bias_initializer = b_init,
recurrent_initializer = r_init))
    # model.add(SimpleRNN(units=look_back, input_shape=(1, look_back),
return_sequences = True))
    model.add(Dense(1)) # activation default olarak linear
    model.compile(loss='mse', optimizer='adam') # The learning rate.
Defaults to 0.001
    model.fit(trainX, trainY, epochs=1000, verbose=1)
    model_predict = model.predict(testX)
    model_predict = scaler.inverse_transform(model_predict)

    testY = scaler.inverse_transform([testY])
    testY = testY.reshape(len(testY[0]),1)

    def mape(actual, pred):
        actual, pred = np.array(actual), np.array(pred)
        absolute=[]
        for i,j in zip(actual,pred):
            av=abs((i-j)/i)
            absolute.append(av)
        return np.mean(absolute)

    mse = mean_squared_error(testY , model_predict)
    rmse = math.sqrt(mse)
    mae = mean_absolute_error(testY , model_predict)
    rkare = r2_score(testY , model_predict)
```

```
    mape = mape(testY, model_predict)

    results[0].append(model)
    results[1].append(look_back)
    results[2].append(rmse)
    results[3].append(mse)
    results[4].append(mae)
    results[5].append(rkare)
    results[6].append(mape)


results_df = {'models': results[0],
              'lookback': results[1],
              'rmse': results[2],
              'mse': results[3],
              'mae': results[4],
              'r2': results[5],
              'mape': results[6]}

results_pd=pd.DataFrame(results_df)

results1 = results_pd.drop('models',axis=1).copy()
```