YAPAY ZEKA TEKNİKLERİ İLE TEDARİK ZİNCİRİNDE ÖN SİPARİŞ TAHMİNİ

Simge KARABAĞ

T.C.
BURSA ULUDAĞ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY ZEKA TEKNİKLERİ İLE TEDARİK ZİNCİRİNDE ÖN SİPARİŞ TAHMİNİ

Simge KARABAĞ
0000-0002-0197-1445

Prof. Dr. Nursel ÖZTÜRK
(Danışman)

YÜKSEK LİSANS TEZİ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

BURSA – 2023

TEZ ONAYI

Simge KARABAĞ tarafından hazırlanan "YAPAY ZEKA TEKNİKLERİ İLE TEDARİK ZİNCİRİNDE ÖN SİPARİŞ TAHMİNİ" adlı tez çalışması aşağıdaki jüri tarafından oy birliği ile Bursa Uludağ Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Danışman : Prof. Dr. Nursel ÖZTÜRK

Başkan : Aaaaa. Dr. Aaaaaaaa AAAAAAAA İmza
000-000-000-000
Aaaaaaaaa Üniversitesi,
Aaaaaaaaaaaa Fakültesi,
Aaaaaaaa Aaaaaaaaaaa Anabilim Dalı

Üye : Aaaaa. Dr. Aaaaaaaa AAAAAAAA İmza
000-000-000-000
Aaaaaaaaa Üniversitesi,
Aaaaaaaaaaaa Fakültesi,
Aaaaaaaa Aaaaaaaaaaa Anabilim Dalı

Üye : Aaaaa. Dr. Aaaaaaaa AAAAAAAA İmza
000-000-000-000
Aaaaaaaaa Üniversitesi,
Aaaaaaaaaaaa Fakültesi,
Aaaaaaaa Aaaaaaaaaaa Anabilim Dalı

Üye : Aaaaa. Dr. Aaaaaaaa AAAAAAAA İmza
000-000-000-000
Aaaaaaaaa Üniversitesi,
Aaaaaaaaaaaa Fakültesi,
Aaaaaaaa Aaaaaaaaaaa Anabilim Dalı

Üye : Aaaaa. Dr. Aaaaaaaa AAAAAAAA İmza
000-000-000-000
Aaaaaaaaa Üniversitesi,
Aaaaaaaaaaaa Fakültesi,
Aaaaaaaa Aaaaaaaaaaa Anabilim Dalı

Yukarıdaki sonucu onaylarım

Prof. Dr. Hüseyin Aksel EREN
Enstitü Müdürü
../../….

B.U.Ü. Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

17/05/2023

Simge KARABAĞ

TEZ YAYINLANMA
FİKRİ MÜLKİYET HAKLARI BEYANI

Enstitü tarafından onaylanan lisansüstü tezin/raporun tamamını veya herhangi bir kısmını, basılı (kâğıt) ve elektronik formatta arşivleme ve aşağıda verilen koşullarla kullanıma açma izni Bursa Uludağ Üniversitesi'ne aittir. Bu izinle Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet hakları ile tezin tamamının ya da bir bölümünün gelecekteki çalışmalarda (makale, kitap, lisans ve patent vb.) kullanım hakları tarafımıza ait olacaktır. Tezde yer alan telif hakkı bulunan ve sahiplerinden yazılı izin alınarak kullanılması zorunlu metinlerin yazılı izin alınarak kullandığını ve istenildiğinde suretlerini Üniversiteye teslim etmeyi taahhüt ederiz.

Yükseköğretim Kurulu tarafından yayınlanan "**Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge**" kapsamında, yönerge tarafından belirtilen kısıtlamalar olmadığı takdirde tezin YÖK Ulusal Tez Merkezi / B.U.Ü. Kütüphanesi Açık Erişim Sistemi ve üye olunan diğer veri tabanlarının (Proquest veri tabanı gibi) erişimine açılması uygundur.

Danışman Adı-Soyadı Tarih	Öğrencinin Adı-Soyadı Tarih
İmza Bu bölüme kişinin kendi el yazısı ile okudum anladım yazmalı ve imzalanmalıdır.	İmza Bu bölüme kişinin kendi el yazısı ile okudum anladım yazmalı ve imzalanmalıdır.

ÖZET

Yüksek Lisans Tezi

YAPAY ZEKA TEKNİKLERİ İLE TEDARİK ZİNCİRİNDE
ÖN SİPARİŞ TAHMİNİ

Simge KARABAĞ

Bursa Uludağ Üniversitesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Nursel ÖZTÜRK

Ön sipariş, tedarik zincirindeki herhangi bir eksiklik, artan ürün talebi veya ürün envanterinin tükenmesi nedeniyle müşterinin talep ettiği mal veya hizmetlerin gelecekte belirli bir tarihe kadar teslim edilmesini garanti edebilen bir siparıştir. Ön siparişler, ani talep artışları, kötü tedarik zinciri yönetimi ve yanlış envanter yönetimi durumlarında meydana gelir. Tedarik zincirindeki aksamalar, müşteri taleplerinde öngörülemeyen artışlar ve yanlış stok yönetimi durumlarında ön siparişler bir zorunluluk haline gelebilir. Bu çalışmada ön sipariş durumunun analiz edilmesi ve gelecekte karşılaşılabilecek ön siparişlerin tahmin edilmesi amaçlanmıştır. Bu şekilde firmalar ön siparişe girecek ürünleri tahmin ederek tedarik zincirindeki oluşması muhtemel aksamaları önleyebilecektir. Bu çalışmada, ön sipariş durumunun tahmini için yapay zeka makine öğrenimi algoritmalarından Derin Sinir Ağları (DNN), Rastgele Orman (RF) ve Aşırı Gradyan Artırma (XGBoost) teknikleri kullanılarak Python ve TAZI programları üzerinde yapılan çalışma ve uygulamaların sonuçları paylaşılmıştır. Literatürdeki çalışmalarda kullanılan makine öğrenimi sınıflandırıcıları, yöntemleri ve gerçekleşen sonuçlar ile elde edilen algoritmaların başarı oranları açıklanmaktadır. Bu tez çalışmasında elde edilen bulgulara göre önerilen model sonuçları AUC metriği baz alındığında 0.959 değeri ile Rastgele Orman algoritması en iyi performansa sahip algoritmadır. Ön sipariş tahmininin yapay zeka teknikleri kullanılarak yapılması, tedarik zinciri maliyetleri ve envanter yönetimi açısından işletmelere fayda sağlayabilecektir.

Anahtar Kelimeler: Ön sipariş, tahmin, makine öğrenimi teknikleri, yapay zeka, dengesiz veriler
2023, xii + 62 sayfa.

ABSTRACT

MSc Thesis

BACKORDER PREDICTION IN THE SUPPLY CHAIN WITH ARTIFICIAL INTELLIGENCE TECHNIQUES

Simge KARABAĞ

Bursa Uludağ University
Graduate School of Natural and Applied Sciences
Department of Industrial Engineering

Supervisor: Prof. Dr. Nursel ÖZTÜRK

A backorder is an order that can guarantee the delivery of the goods or services requested by the customer by a specified date in the future due to any shortcomings in the supply chain, increased product demand, or depletion of product inventory. Backorders occur in situations of sudden demand increases, weak supply chain management and incorrect inventory management. Backorders can become a necessity in cases of disruptions in the supply chain, unforeseen increases in customer demands and incorrect inventory management. In this study, it is aimed to analyze the backorder status and to predict future backorders. In this way, companies will be able to prevent possible disruptions in the supply chain by predicting the products that will enter the backorder. In this study, the results of the studies and applications made on Python and TAZI programs by using Deep Neural Networks (DNN), Random Forest (RF) and Extreme Gradient Boosting (XGBoost) techniques, which are artificial intelligence machine learning algorithms, are shared for the prediction of the backorder status. The machine learning classifiers used in the studies in the literature, their methods and the actual results and the success rates of the algorithms are explained. According to the findings obtained in this thesis study, the Random Forest algorithm has the best performance with a value of 0.959, based on the proposed model results AUC metric. Backorder estimation will benefit companies using supply chain costs and inventory management issues.

Key words: Backorder, estimation, machine learning techniques, artificial intelligence, imbalanced data
2023, xii + 62 pages.

TEŞEKKÜR

Yüksek lisans süresince ve bu tezi hazırlamamda bilgisi, tecrübesiyle şahsıma ışık tutan, anlayışlı tutumu ve sabrıyla beni motive eden, yardımlarını esirgemeden bana destek olan danışman hocam Sayın Prof. Dr. Nursel ÖZTÜRK'e, mensubu olmaktan gurur duyduğum Ford Otosan ailesine ve yakın çalışma arkadaşlarıma, sürekli yanımda olan ve manevi desteklerini hep hissettiğim annem Semra VATANSEVER'e, babam Sevim VATANSEVER'e ve eşim Emre Can KARABAĞ'a teşekkürlerimi sunarım.

Simge KARABAĞ
17/05/2023

İÇİNDEKİLER

Sayfa

KISALTMALAR DİZİNİ

Kısaltmalar	Açıklama
ADASYN	Adaptive Synthetic Sampling Approach (Uyarlanabilir Sentetik Örnekleme Yaklaşımı)
AI	Artificial Intelligence (Yapay Zeka)
ANN	Artificial Neural Network (Yapay Sinir Ağları)
ART	Adaptive Resonance Theory (Adaptif Rezonans Teorisi)
AUC	Area Under the Curve (Eğrinin Altındaki Alan)
CBUS	Cluster-Based Under Sampling (Kümeleme Tabanlı Yetersiz Örnekleme)
CPD	Conditional Probability Distribution (Koşullu Olasılık Dağılımı)
CPT	Conditional Probability Table (Koşullu Olasılık Tablosu)
DAG	Directed Acyclic Graphs (Yönlendirilmiş Asiklik Grafikler)
DNN	Deep Neural Network (Derin Sinir Ağı)
DT	Decision Tree (Karar Ağacı)
EF	Ensemble Filter (Topluluk Filtresi)
GBDT	Gradient Boosting Decision Trees (Gradyan Artırıcı Karar Ağacı Algoritmaları)
IPF	Iterative Partitioning Filter (Yinelemeli Bölme Filtresi)
KNN	K-Nearest Neighbours Algorithm (K-En Yakın Komşu Algoritması)
LGBM	LightGBM (Hafif Gradyan Artırma Makineleri)
LR	Logistic Regression (Lojistik Regresyon)
LVQ	Linear Vector Quantization (Lineer Vektör Uzayı)
ML	Machine Learning (Makine Öğrenimi)
PGM	Probabilistic Graphical Modelling (Olasılıksal Grafik Modelleme)
RF	Random Forest (Rastgele Orman)
RNN	Recurrent Neural Network (Tekrarlayan Sinir Ağı)
ROC	Receiver Operating Characteristic Curve (Alıcı İşlem Karakteristikleri Eğrisi)
ROS	Random Oversampling (Rastgele Aşırı Örnekleme)
RUS	Random Undersampling Technique (Rastgele Alt Örnekleme Tekniği)
SMOTE	Synthetic Minority Oversampling Technique (Sentetik Azınlık Örnekleme Tekniği)
SVM	Support Vector Machines (Destek Vektör Makineleri)
XGBoost	Extreme Gradient Boosting (Aşırı Gradyan Artırma)

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

Sayfa

1. GİRİŞ

Ön sipariş, yeterli stok olmaması nedeniyle teslim edilemeyen bir sipariş satırıdır. Bu nedenle, ön siparişin teslim edilebilmesi için envanterin yenilenmesi gerekir (Aalst, 2016).

Envanter sisteminin amacı, firma ve kuruluşların müşterilerine yeterli hizmeti sağlamasıdır. Müşteri taleplerinin zamanında karşılanmaması, müşterilerin bekledikleri hizmeti alamamaları nedeniyle olumsuz etkilenmektedir. Bu olumsuz etki, envanter sistemi için kıtlık maliyetlerine çevrilebilir. Eksik maliyet, eksiklik başına sabit bir maliyet olabilir veya müşterinin bir ürün için beklemesi gereken süreye bağlı olabilir (bundan sonra genellikle ön sipariş maliyeti olarak adlandırılır). Eksik maliyet, teslimatın gecikmesi durumunda somut parasal geri ödemeleri veya fiyat indirimlerini yansıtabilir. Gelecekteki satışlar ve kazançlar açısından itibar kaybını da yansıtmalıdır (Charles, CorbettJan, Fransoo ve Tan, 2017).

Ön sipariş, tedarik zincirindeki herhangi bir eksiklik, artan ürün talebi veya ürün envanterinin tükenmesi nedeniyle müşterinin talep ettiği mal veya hizmetlerin gelecekte belirli bir tarihe kadar teslim edilmesini garanti edebilen bir siparıştir. Ön siparişler, ani talep artışları, kötü tedarik zinciri yönetimi ve yanlış envanter yönetimi durumlarında meydana gelir. Tedarik zincirindeki aksamalar, müşteri taleplerinde öngörülemeyen artışlar ve yanlış stok yönetimi durumlarında ön siparişler bir zorunluluk haline gelebilir.

Bu tez çalışmasında, ön sipariş tahmin çalışması için yapay zeka yaklaşımına başvurulmuştur. Yapılan çalışmalarda makine öğrenimi teknikleri kullanılarak ön sipariş durumu tahmin edilmiştir. Kaynak araştırması bölümünde makine öğrenme algoritmaları ve çalışmalardan bahsedilmiştir. Materyal ve yöntem kısmında Python ile veri ön işleme aşamaları ve Derin Sinir Ağı (Deep Neural Network, DNN), Rastgele Orman (Random Forest, RF) ve Aşırı Gradyan Artırma (Extreme Gradient Boosting, XGBoost) tekniklerinin TAZI yapay zeka programı üzerinde uygulamasından bahsedilmiştir.

2. KURAMSAL TEMELLER ve KAYNAK ARAŞTIRMASI

Kuramsal temeller başlığı altında yapay zeka ve makine öğrenimi algoritmaları konularından bahsedilmiş, kaynak araştırması başlığı altında literatürde daha önce bu konuda yapılmış çalışmalar ve kaynak özetleri anlatılmıştır.

2.1. Kuramsal Temeller

Yapay zeka ve makine öğrenimi algoritmaları, farklı algoritmalar kullanarak tekrarlanan denemeler yoluyla tahminler, mevcut durumlarda iyileştirmeler gibi çok yönlü görevleri gerçekleştirme yetkinliğine sahip her bir veri kümesi üzerinde modeller oluşturmak için istisnai tekniklerdir. ML (Machine Learning, Makine öğrenimi) algoritmaları, modeller oluşturur ve bu modeller denetimli öğrenme yoluyla eğitilir. Burada değişkenler ve öznitelikler, veri kümesinin orijinalliği korunurken tekrarlanan denemelerle eşlenmektedir ve süreçte kullanılan her bir sınıflandırıcı veya algoritmaya göre modeller oluşturulmaktadır (Malviya, Chittora, Chakrabarti, Vyas ve Poddar, 2020).

ML algoritmalarının hata yapma olasılığı vardır, ancak sınırlamalarını koruyarak, algoritmalara dayalı çeşitli modeller oluşturulabilir ve AUC (Area Under the Curve, Eğrinin Altındaki Alan), kesinlik, geri çağırma (recall), F puanı, Gini katsayıları vb. gibi çeşitli parametrelerdeki doğruluklar karşılaştırılabilir. Daha sonra bu modeller, sonuçların belirlenmesi için test veri seti üzerinde test edilebilir. Literatürde, mevcut veri setlerine dayalı olarak ön sipariş senaryosunu tahmin etmek için çoklu sınıflandırıcılar kullanılmıştır. Bunlar arasında Yapay Sinir Ağları, Rastgele Ağaç, C5.0, Bayes Ağı, Lojistik Regresyon, Ayrım Analizi ve Destek Vektör Makineleri yer almıştır. Bu algoritmalar aşağıda kategorize edilmiştir.

ANN (Artificial Neural Network, Yapay Sinir Ağı), insan beyni nöron ağlarının işlevselliğini takip eden insan sinir ağı çalışmasının hesaplamalı dijital sürümüdür. Bu nedenle, yapay sinir ağı, insan beyninin simülasyonlarını yapmak ve faaliyetlerin yürütülmesine yönelik geliştirme yapmak için tasarlanmış bir bilgi işlem sistemidir.

Yapay sinir ağı, çok katmanlı bir istatistiksel işleme tekniğidir. Giriş katmanı, veri kümesinden birçok bilgi toplamaktadır. Ayrıca girdi verilerini bir çıktı katmanının anlayabileceği verilere dönüştüren bir veya daha fazla gizli katman da vardır. Bu katmanlar tamamen birbirine bağlantılıdır ve çıktı katmanından giriş katmanına geri yayılım ile ağırlıkların güncellenmesi temel alınarak dizilmiştir. Başlangıçta yapay sinir ağı, farklı alt kümelerin örüntüsünü ayırt etmek için denetimli öğrenme süreciyle eğitilir ve ardından orijinal çıktı ile modellenen çıktı karşılaştırılır. Çıktılar arasındaki fark, minimum hata durumuna ulaşana kadar ağırlıklı bir şekilde uyum sağlamak için çıkış katmanından giriş katmanına gelen dönüş yolu kullanılarak hesaplanmaktadır (Malviya ve diğerleri, 2020).

Rastgele orman veya rastgele ağaç sınıflandırıcı, bir topluluk ve denetimli öğrenme algoritmasıdır. Çok sayıda karar ağacı açısından sonuçları tahmin etmek için bir orman oluşturur ve her karar ağacı sonuçlarla ortaya çıkar. Böylece birçok çözümle sonuçlanır. Bu algoritma, sağlanan veri kümesinden *n*'inci veri parametrelerini seçer ve kesin, doğru ve kararlı bir çözüm elde etmek için bunları birleştirir. Rastgele orman veya rastgele ağaç sınıflandırıcıları çoğunlukla büyük veri kümeleri için de kullanılan sınıflandırıcılardır (Malviya ve diğerleri, 2020).

Lojistik regresyon, ikili yanıtların olasılıklarını hesaplayarak kategorik bağımlı ayrık değişkenler ile bir veya daha fazla bağımsız ayrık değişken arasındaki ilişkiyi kurar. Olasılık tahminlerine dayanarak ve kategorize edilmiş tanımlama sorunsuz bir şekilde yapılabilir (Malviya ve diğerleri, 2020).

C5.0 aynı zamanda hem ayrık hem de sürekli veriler için geçerli olan bir karar ağacı sınıflandırıcısıdır. Veri seti kesikli ve sürekli veri olarak ikiye ayrılır ve buna göre karar ağaçları oluşturulur. C5.0 sınıflandırıcısı verileri birden çok kez böler ve hatayı en aza indirerek tahmin önerir (Malviya ve diğerleri, 2020).

Bayes ağı tekniği, Yönlendirilmiş Asiklik Grafikler (Directed Acyclic Graphs, DAG) ile belirsizlikleri hesaplamak için kullanılan Olasılıksal Grafik Modelleme (Probabilistic

Graphical Modelling, PGM) altında sınıflandırılmıştır. Temel olarak, DAG grafiğinin düğümleri ve bağlantıları vardır. DAG grafikleri, her rastgele değişkenin Koşullu Olasılık Dağılımı'na (Conditional Probability Distribution, CPD) bağlı olarak bir olayın belirsizliğini modeller. Ağdaki her değişkenin CPD'sine karşılık gelmek için bir Koşullu Olasılık Tablosu (Conditional Probability Table, CPT) kullanılır.

Destek vektör makineleri algoritması, eğitim veri setindeki herhangi bir noktadan oldukça uzakta olan iki parametre arasındaki karar sınırlarını bulmak için kullanılır. Verileri, ilişkilerine bağlı olarak istenen grubun alt kümelerine dönüştüren bir çekirdek oluşturur.

Diskriminant analizi veri kümesi arasında ayrı sınıfları veya grupları tanımlar ve en iyi kombinasyonları belirler. Temel amaç, veri boyutunu ortadan kaldırmak ve yeni özellikler, örneklerin aynı gruplar arasında minimum dağılımını oluşturarak, farklı gruplar arasındaki dağılımı maksimuma çıkarmaktır (Malviya ve diğerleri, 2020).

Hız ve performans için oluşturulan gradyan destekli karar ağacı uygulamasına XGBoost adı verilmektedir. Aşırı Gradyan Artırma (Extreme Gradient Boost), XGBoost olarak adlandırılmaktadır. Çok sayıda geliştiricinin katkıda bulunduğu gradyan artırma makinelerinin bir uygulamasıdır. Sınıflandırma ve regresyon modelleme için kullanılan denetimli bir makine öğrenimi tekniğidir. XGBoost adı verilen geliştirilmiş teknik, kayıp fonksiyonu, sütun örneklemesi ve düzenli hale getirmede bir dizi değişikliği gerçekleştirmek için gradyan artırıcı karar ağaçlarını kullanmaktadır (Varghese, Krishnadas ve Antony, 2023).

2.2. Kaynak Araştırması

Tedarik zinciri sisteminde talep tahmini için birçok araç mevcuttur, ancak bazı yöntemler müşterinin taleplerini tahmin etmede daha az doğruluğa sahiptir. Bu durum dolayısıyla araştırmacıları mevcut tedarik zinciri yönetim sisteminde dalgalanan ve sürekli artış gösteren talebe cevap vermek için daha doğru teknikler belirlemeye zorlamıştır. Sonunda, Yapay Zeka (Artificial Intelligence, AI) ve Makine Öğrenimi (Machine Learning, ML)

yöntemlerinin kullanılmaya başlamasıyla, tedarik zinciri yönetim sisteminin üretim, kar, ticaret, talep ve ön sipariş gibi çeşitli yönlerinde nispeten daha doğru sonuçların tahmin edilmesi mümkün olabilmektedir. Bazı araştırmacılar, makine öğrenimi tekniklerinin çeşitli algoritmalarını uygulamış ve farklı algoritmalarla farklı sonuçlar elde etmiştir. Makine öğrenimi yaklaşımları, üreticilerin karmaşık talepleri tahmin etmek için yaygın olarak kullanılmıştır.

De Santis, De Aguiar ve Goliatt (2017), makine öğrenimi sınıflandırıcıları, ön siparişe giren öğelerin göreli sıklığının, ön siparişe girmeyen öğelere kıyasla nadir olduğu dengesiz sınıf sorunu için bir tahmin modeli önermeyi amaçlamışlardır. Sentetik Azınlık Örnekleme Tekniği (Synthetic Minority Oversampling Technique, SMOTE), Rastgele Örnekleme Alt Tekniği (Random Undersampling Technique, RUS) ve Aşırı Örnekleme (Oversampling) teknikleri gibi farklı örnekleme teknikleri ile sınıf dengesizliği sorununun üstesinden gelmeye çalışmışlardır. Ayrıca ham veriler üzerinde Blagging tekniğini kullanmışlardır. Modellerin değerlendirilmesinde hassasiyet (precision), geri çağırma (recall) ve AUC (Eğrinin Altındaki Alan, Area Under the Curve, AUC) metriklerini kullanmışlardır. Lojistik regresyon (Logistic Regression, LR), karar ağacı (Decision Tree), Gradyan Artırıcı Karar Ağacı Algoritmaları (Gradient Boosting Decision Trees, GBDT) ve Blagging gibi farklı denetimli makine öğrenimi algoritmaları kullanılmış ve 5 kat çapraz doğrulama ile hiperparametreler ayarlanmıştır. Gradyan Artırıcı Karar Ağacı Algoritmaları ve Blagging sınıflandırıcı modelleri iyi performans göstermiştir. Değerlendirme metriği, hassas geri çağırma eğrisi altındaki alan olduğunda, Blagging sınıflandırılmış diğer modellerden daha iyi performans göstermiştir.

Malviya, Chittora, Chakrabarti, Vyas ve Poddar (2020), makine öğrenimi algoritmalarını kullanan ön sipariş tahminlerini ele almakta, basitçe anlaşılır ve uygulanabilir ön sipariş karar senaryoları sunmaktadır. Yapay sinir ağı (ANN), Rastgele Orman (Random Forest, RF), lojistik regresyon (LR), c5.0, Bayes ağı, destek vektör makineleri (support vector machine, SVM), diskriminant analizi yöntemlerini kullanarak IBM SPSS Modeler programında model oluşturmuşlardır.

Shajalal, Hajek ve Abedin (2021), ön siparişleri tahmin etmek için derin bir sinir ağı kullanan bir model önermektedir; Karşılıksız siparişler ile karşılanan siparişler arasındaki veri dengesizliği etkin tekniklerle ele alınmıştır. Belirli bir ürün siparişinin ön sipariş edilip edilmeyeceğini tahmin eden derin sinir ağı (deep neural network, DNN) tabanlı bir sınıflandırma modeli önerilmiştir.

Islam ve Amin (2020), karar vermede esneklik, sürecin daha iyi netliği ve daha yüksek doğruluk sağlarken ürünlerin ön sipariş sırasını tahmin ederek iş karar süreci alanında makine öğrenimi modellerini kullanmayı amaçlamıştır. Yazarlar, modeli eğitmek için Rastgele Orman (Random Forest, RF) ve Gradyan Artırıcı Karar Ağacı Algoritmaları (GBDT) yöntemlerini kullanmışlardır, çünkü bu modeller oldukça yorumlanabilir ve model tahminlerine dayalı yönetimsel kararlar almayı kolaylaştırmaktadır.

Hajek ve Abedin (2020), stokastik envanter modellerine büyük veriye dayalı bir ön sipariş tahmini sağlamak için, bekleyen sipariş kararlarının beklenen kârını en üst düzeye çıkarmak için bir eksik örnekleme prosedürüyle donatılmış bir makine öğrenimi modeli önermiştir. ROC eğrisi (Receiver Operating Characteristic Curve), C4.5 ağacı, K-en yakın komşu (K-Nearest Neighbours Algorithm, KNN), lojistik regresyon, sinir ağı, rastgele orman, destek vektör makineleri, SMOTE, RUS, XGBoost, kümeleme tabanlı yetersiz örnekleme (CBUS) teknikleri, K-ortalama kümeleme (K-Means Clustering) ve genetik algoritma (Algorithm Profit-Max CBUS) yaklaşımları Weka 3.8.1. programında uygulanmıştır.

Lawal ve Akintola (2021), büyük ve dengesiz envanter veri setinde tekrarlayan sinir ağı (Recurrent Neural Network, RNN) kullanan bir ön sipariş tahmin modeli önermektedir. Veriler Min-Max Ölçekleyici ile önceden işlendiğinde, dengesiz verilere ve bunların çıktıları RNN'ye beslenen üç veri dengeleme yöntemi (ADASYN, SMOTE ve Rastgele Örnekleme) aynı anda uygulanmıştır. Böylece hangi ürünün ön sipariş verileceğini tahmin etmiştir. Elde edilen sonucun değerlendirilmesi, ADASYN+ RNN'nin 0.901 kesinlik, 0.879 geri çağırma (recall) ve 0.889 F1-Puanı ile daha iyi performans verdiğini

göstermiştir. Önerilen model, diğer makine öğrenimi algoritmalarına kıyasla ürün ön sipariş tahminleri üzerinde önemli bir etkiye sahiptir.

Stok eksikliği ve uzun ürün teslimat gecikmeleri, ek üretim maliyetlerine ve memnun olmayan müşterilere yol açmaktadır. Bu nedenle, Ntakolia, Kokkotis, Karlsson ve Moustakidis (2021), tedarik zincirinin verimliliğini ve dolayısıyla şirket performansını iyileştirmek amacıyla bir envanter sistemindeki ön sipariş oranını etkili bir şekilde tahmin edecek çeşitli makine öğrenimi modellerini uygulayarak karşılaştırmışlardır. Rasgele Orman (RF), LightGBM (LGBM), XGBoost (XGB), Balanced Blagging (BB), Derin Sinir Ağları (DNN), Lojistik Regresyon (LR), Destek Vektör Makineleri (SVM) ve K-En Yakın Komşular (KNN) makine öğrenimi modellerini kullanarak elde ettiği sonuçlarda RF, XGB, LGBM ve BB modellerinin 0,95 AUC puanına ulaştığını gözlemlemişlerdir. Uygulama sonuçları, LGBM modelinin İzotonik Regresyon yöntemi ile kalibrasyondan sonra en iyi performans gösteren model olduğunu göstermiştir. Sınıf etiketine dayalı sınıflandırma problemini çözdükten sonra uyguladıkları post-hoc açıklanabilirlik analizi, bir ürünün envanter stoğunun, teslimat hacminin, acil talebin (satışların) ve gelecekteki talebin doğru tahmin edilmesinin ön siparişlerin doğru bir tahmine önemli ölçüde katkıda bulunabileceğini göstermiştir.

Literatür incelemelerine dayanarak, bu tez çalışmasında ön sipariş tahmini yapabilmek amacıyla doğruluğu ve performansı daha önce kanıtlanmış Derin Sinir Ağı, Rastgele Orman ve XGBoost algoritmaları kullanılmıştır. Bu tez çalışmasının diğer çalışmalardan farkı, veri ön işleme aşamasında daha iyi sonuçlar almaya yardımcı olacak daha detaylı çalışmaların yapılması, diğer alanlarda da performansı kanıtlanmış olan yapay zeka tekniklerinin ön sipariş tahmininde kullanılması ve kullanılan yapay zeka tekniklerinin literatürde ön sipariş tahmininde yapılan çalışmalara göre daha iyi sonuçlar elde edilmesidir.

Ön sipariş tahmini konusunda literatürdeki ilgili çalışmaların bir özeti Çizelge 2.1.'de verilmiştir.

Çizelge 2.1. Ön sipariş tahmini ile ilgili önceki çalışmaların karşılaştırılması

Yazarlar	Tahmin	ML Sınıflandırıcıları	En İyi Yöntem	Değer
De Santis, De Aguiar ve Goliatt (2017)	Malzeme ön sipariş tahmini	Sınıflandırma Ağacı, LR, RUS, SMOTE, RF, GBOOST, Blagging	Blagging	AUC Test: 0.9478±0.0022
Malviya, Chittora, Chakrabarti, Vyas ve Poddar (2020)	Ön sipariş karar senaryoları	ANN, RF, LR, c5.0, Bayes, SVM, DA	C5.0 ve LSVM	F ölçütü : 0,994
İslam ve Amin (2020)	Ürün ön sipariş tahmini	Dağıtılmış Rastgele Orman (DRF) ve Gradient Boosting (GBM)	Dağıtılmış Rastgele Orman (DRF)	F1 skoru: 0.7049
Hajek ve Abedin (2020)	Ön sipariş ve beklenen kar	C4.5, KNN, LR, DNN, RF, SVM, CBUS	Rasgele Orman	Profit-max CBUS: 4.00±0.38
Shajalal, Hajek ve Abedin (2021)	Ön sipariş tahmini	DNN, Weighted_DNN, Ran_Over_DNN, SMOTE_Over_DNN, Com_SMOTE_Under_DNN	Com_SMOTE Under_DNN	AUC Test: 0.9586
Ntakolia, Kokkotis, Karlsson ve Moustakidis (2021)	Envanter sistemi ve ürün ön siparişi	RF, LightGBM, XGBoost, Balanced Blagging (BB), (DNN), LR, SVM and KNN	LGBM - İzotonik Regresyon ile kalibrasyon	AUC : 0.95
Lawal ve Akintola (2021)	Ön sipariş tahmini	RNN, SMOTE, RUS, Adaptive Synthetic (ADASYN)	ADASYN+ RNN	F1 skoru: 0.889

3. MATERYAL ve YÖNTEM

Yapılan çalışmalarda ön sipariş tahmini için son yıllarda çeşitli yapay zeka teknikleri kullanıldığı görülmektedir. Bu tez çalışmasında da önceki çalışmalara konu olan veri seti kullanılarak veri ön işleme aşamalarının tamamlanması ve Derin Sinir Ağı, Rastgele Orman ve XGBoost algoritmaları ile ön sipariş tahmini yapılması amaçlanmıştır. Bu tez çalışmasında veri ön işleme aşamasında Python, makine öğrenimi algoritmalarının uygulanmasında ise TAZI programı kullanılmıştır. Kaggle veri tabanından sağlanmış ürün ön sipariş tahmini veri setinin öznitelikleri ve açıklamaları Çizelge 3.1.'de gösterilmiştir.

Çizelge 3.1. Ön sipariş tahmininde kullanılan öznitelikler

No	Özellikler	Açıklamalar
1	sku	Rastgele ürün kodu
2	national_ inv	Komponentin mevcut envanter düzeyi
3	lead _time	Kaynaktan hedefe geçiş süresi
4	in_ transit_ qty	Kaynaktan taşınan ürün miktarı
5	forecast_3 month	Önümüzdeki 3 ay için satış tahmini
6	forecast_6 month	Önümüzdeki 6 ay için satış tahmini
7	forecast_9 month	Önümüzdeki 9 ay için satış tahmini
8	sales_1 month	Son 1 aylık süre için satış miktarı
9	sales_3 month	Son 3 aylık süre için satış miktarı
10	sales_6 month	Son 6 aylık süre için satış miktarı
11	sales_9 month	Son 9 aylık süre için satış miktarı
12	min_bank	Stokta önerilen minimum miktar
13	potential_issue	Tanımlanan parça için kaynak sorunu (Evet/Hayır)
14	pieces_past_due	Kaynaktan süresi geçmiş parçalar
15	perf_6 moth_avg	Son 6 aydaki kaynak ortalama performansı
16	perf_12 moth_avg	Son 12 aydaki kaynak ortalama performansı
17	local_co_qty	Vadesi geçmiş stok siparişlerinin miktarı
18	deck_risk	Ortak risk işaretleri (Evet/Hayır)
19	oe_constraint	Ortak risk işaretleri (Evet/Hayır)
20	ppap_risk	Ortak risk işaretleri (Evet/Hayır)
21	stop_auto_buy	Ortak risk işaretleri (Evet/Hayır)
22	rev_stop	Ortak risk işaretleri (Evet/Hayır)
23	went_on_backorder	Hedef Değer- Ürün, ön siparişte kaldı

3.1. Veri Ön işleme

Veri ön işleme, veri analizi uygulamalarının en önemli adımlarından biridir. Ham veriler genellikle çeşitli tutarsızlıklar, aralık dışı sapmalar, eksik değerler, gürültü ve/veya aykırı değerler gibi pek çok kusur içerebilmektedir. Veri ön işleme sonrasındaki adımlarda gerçekleştirilecek makine öğrenme algoritmalarının performansı, hatalı ve düşük kaliteli veriler nedeniyle düşmektedir. Bu nedenle ham verinin çeşitli ön işleme aşamalarından geçirilerek kalitesinin arttırılması gerekmektedir (Çetin ve Yıldız, 2021).

Gerçek dünya verileri genellikle eksik veya tutarsızdır. Ayrıca belirli davranış veya önyargılardan yoksundur ve muhtemelen birçok hata içermektedir. Veri ön işleme, bu tür sorunları çözmek için kanıtlanmış bir yöntemdir. Bu aşamanın amacı, ham verileri daha kaliteli sonuçlar üretebilecek ve anlaşılır bir formata dönüştürmektir. Bu tez çalışmasında derin sinir ağı, rastgele orman ve XGBoost algoritmaları kullanıldığı için eğitim veri setindeki tüm özelliklerin sayısal olması gerekmektedir. Dolayısıyla bu bölümde, veri kümesinin ön işlemeye yönelik adımları tanıtılacak ve ayrıca veri içeriği ve ölçeği hakkında bilgiler verilecektir.

Bu çalışmada kullanılan veri seti, Kaggle web sitesinden alınmıştır. Veriler hakkında bilgi Çizelge 3.2.'de bulunmaktadır. Veri seti, tahmin edilecek haftadan önceki 8 hafta için ürünlerin bilgilerini içeren geçmiş verilerdir. Veriler, her haftanın başında haftalık anlık görüntüler olarak alınmıştır. Veri seti toplamda, 1.915.211 girişli ve 7 tanesi kategorik, 1 tanesi kesikli sayısal ve diğer 15 tanesi sayısal olmak üzere 23 öznitelikten oluşmaktadır. Çizelge 3.2., bu 23 özniteliğin kısa açıklamasını ve veri türlerini göstermektedir. Örneğin; SKU, rastgele ürün kodlarını temsil eder. Her kaydın benzersiz bir ürün/hafta kombinasyonunu tanımladığı sıralı bir tanımlayıcı ile rastgele ürün kodları oluşturulmuştur.

Niteliklerden bazıları taşıma durumunu temsil etmektedir. Örneğin: ürünün teslim süresi (teslim süresi, hafta olarak) ve kaynaktan taşınan ürün miktarı (taşıma miktarı, birimler).

Niteliklerden bazıları satışları ve tahmini temsil etmektedir: öznitelik 6-10, bazıları ise kısmen risk ve kaynak sorununu temsil etmektedir: nitelik 11-19.

Çizelge 3.2. Özniteliklerin veri tipi

No	Öznitelikler	Açıklamalar	Veri Tipi
1	sku	Rastgele ürün kodu	Kesikli Sayısal
2	national_ inv	Komponentin mevcut envanter düzeyi	Sayısal
3	lead _time	Kaynaktan hedefe geçiş süresi	Sayısal
4	in_ transit_ qty	Kaynaktan taşınan ürün miktarı	Sayısal
5	forecast_3 month	Önümüzdeki 3 ay için satış tahmini	Sayısal
6	forecast_6 month	Önümüzdeki 6 ay için satış tahmini	Sayısal
7	forecast_9 month	Önümüzdeki 9 ay için satış tahmini	Sayısal
8	sales_1 month	Son 1 aylık süre için satış miktarı	Sayısal
9	sales_3 month	Son 3 aylık süre için satış miktarı	Sayısal
10	sales_6 month	Son 6 aylık süre için satış miktarı	Sayısal
11	sales_9 month	Son 9 aylık süre için satış miktarı	Sayısal
12	min_bank	Stokta önerilen minimum miktar	Sayısal
13	potential_issue	Tanımlanan parça için kaynak sorunu (Evet/Hayır)	Kategorik
14	pieces_past_due	Kaynaktan süresi geçmiş parçalar	Sayısal
15	perf_6 moth_avg	Son 6 aydaki kaynak ortalama performansı	Sayısal
16	perf_12 moth_avg	Son 12 aydaki kaynak ortalama performansı	Sayısal
17	local_co_qty	Vadesi geçmiş stok siparişlerinin miktarı	Sayısal
18	deck_risk	Ortak risk işaretleri (Evet/Hayır)	Kategorik
19	oe_constraint	Ortak risk işaretleri (Evet/Hayır)	Kategorik
20	ppap_risk	Ortak risk işaretleri (Evet/Hayır)	Kategorik
21	stop_auto_buy	Ortak risk işaretleri (Evet/Hayır)	Kategorik
22	rev_stop	Ortak risk işaretleri (Evet/Hayır)	Kategorik
23	went_on_backorder	Hedef Değer- Ürün, ön siparişte kaldı	Kategorik

Sırasıyla veri temizleme, eksik verilerin tamamlanması, verilerin normalize edilmesi, dengesiz veri setinin dengeli hale getirilmesi gibi veri ön işleme aşamaları Python yazılım dili kullanılarak Jupyter Hub aracılığı ile tamamlanmış ve detayları EK 1, EK 2, EK 3 ve EK 4'te verilmiştir. Veri ön işleme aşamasında veri setinin analiz edilmesi, özniteliklerin veri tipleri, istatistiksel bilgilerinin analizi EK 1'de özetlenmiştir.

3.1.1. Veri Temizleme

Gerçek dünyada toplanan veriler genellikle eksik ve gürültülü değerler içermektedir. Bu gürültülü verileri belirlemek ve temizlemek, veri analizinin zorluklarından biridir ve bu adımı atlamak, yanlış analizlere ve güvenilmez kararlara yol açmaktadır. Bu duruma dikkat çekmek için bu bölümde gürültü filtreleme ve kayıp değer atama yöntemleri incelenmiştir (Çetin ve Yıldız, 2021).

Veri setindeki hatalı ölçümler veya insan hatalarından kaynaklanan hatalı verilere gürültü denmektedir. Verilerdeki gürültü kavramı, ölçüm değerinin gürültüsü (özellik gürültüsü) ve sınıf etiketinin gürültüsü olmak üzere ikiye ayrılmaktadır. Sınıf etiketi (hedef değişken) gürültüsünün özellik (öznitelik) gürültüsünden daha zararlı olduğu bilinmektedir. Bunun nedeni genellikle yanlış yönlendirmeye eğilimli olmasıdır. Özellikle sınıflandırma problemlerinin eğitim verilerinde bu iki tip gürültünün bulunması karar vermeyi önemli ölçüde olumsuz etkilemekte ve yüksek hassasiyetli model oluşturulmasında ciddi sorunlara neden olmaktadır. Ayrıca modelin üretim süresini de olumsuz etkileyerek uzatmaktadır. Bu sebeplerden dolayı çeşitli gürültü filtreleme yöntemleri geliştirilmiştir ve veri analizi uygulamalarında sıklıkla kullanılmaktadır. Birden fazla yöntemi birleştirmenin güvenilirliği nedeniyle, yaygın olarak kullanılan gürültü filtreleme yöntemleri aşağıda verilmektedir (Çetin ve Yıldız, 2021):

Topluluk Filtresi (Ensemble Filter, EF): Veri kümesindeki gürültüyü ortadan kaldırmak için öğrenme algoritmalarının kullanılması başarılı sonuçlar vermektedir. Fakat birkaç tane öğrenme algoritmasının birlikte kullanılması, tek öğrenme algoritması kullanılmasına göre gürültü tespiti için çok daha başarılı olmaktadır. Bu yöntem EF olarak isimlendirilmektedir. EF tekniğinin en önemli faydası, tekli yöntemlerin sınırlamalarını aşması ve bazı tekli yöntemlerin yaptığı hataları oylama mekanizması ile düzeltebilmesidir (Çetin ve Yıldız, 2021).

Yinelemeli Bölme Filtresi (Iterative Partitioning Filter, IPF): EF gibi, IPF de gürültüyü filtrelemek için çoklu sınıflandırma algoritmaları kullanır ve sonuçlara oylama

yöntemiyle karar verir. EF yönteminden farkı; bir durma eşiğine ulaşılana kadar filtrelediği verileri tekrarlar. Başka bir deyişle, sonraki adımlarda diğer gürültüleri temizlemek için önceki adımlarda gürültüden arındırılmış verileri kullanır. Sentetik Azınlık Yüksek Örnekleme Tekniği (Synthetic Minority Oversampling Technique, SMOTE), EF gibi verilerde sınıf dağılımını dengelemek için önerilen bir yöntemdir. Ancak, EF'den farklı olarak, azınlık sınıfı örneklerini yüksek hızda örnekleyerek sınıfları dengeler. SMOTE-IPF, IPF ile gürültüyü ortadan kaldırarak sınıflandırma performansını daha da artırmak için sunulmuştur (Çetin ve Yıldız, 2021).

Gürültüyü filtrelemenin tek yolu makine öğrenimi sınıflandırıcıları değildir. Denetimsiz bir makine öğrenmesi yöntemi olan kümeleme de tercih edilen yöntemlerden biridir. Bu yöntemde benzer veriler aynı kümede gruplandırılır ve bu grupların dışındaki veriler gürültü olarak belirlenerek ya silinir ya da değeri en yakın kümeye değiştirilir. K-means, yaygın olarak kullanılan kümeleme algoritmalarındandır (Çetin ve Yıldız, 2021).

Diğer bir gürültü giderme yöntemi ise binning yöntemidir. Öncelikle verilerin sıralandığı Binning yöntemi, diğer yöntemlere göre daha basit bir algoritmaya sahiptir. Daha sonra, bu sıralı veriler eşit parçalara bölünür ve her parçadaki veriler bulundukları bölümün ortalamasına veya minimum/maksimum değerlerine göre değiştirilir. Bu yöntemle doğru ölçülen verilerde de değişiklikler yapılır, ancak gürültülü değerler olması gereken aralığa çekilir. Güncel literatürde diğer yöntemler kadar kullanılmasa da aykırı verilerden çok fazla etkilenmesi ve bozuk verileri iyi tespit edememesi nedeniyle örnekler bulunabilmektedir (Çetin ve Yıldız, 2021).

Bu tez çalışmasında veriler analiz edilirken son 5 satırın incelenmesi aşamasında son kaydın boş olduğu gözlemlenmiştir. Bu nedenle df = df[:-1] komutu ile son satır silinmiştir. Her kaydın benzersiz bir ürün/hafta kombinasyonunu tanımladığı sıralı ve rastgele oluşturulmuş ürün kodları özelliği olan sku özelliği hedef değişken ile ilişkili olmadığı için silinmiştir. Veri temizleme aşamasında geçersiz kayıtların ve ürün kodu sku özniteliğinin silinmesi EK 2'de belirtilmiştir.

3.1.2. Eksik Verilerin Tamamlanması

Bazı örneklerde orijinal veri kümelerinden bir veya daha fazlasında eksik değerler olabilir. Buna kayıp/eksik değer problemi denir. Öğrenme algoritmaları eksik veya kayıp değerler içeren bir veri seti aldığında ya modelin doğruluğu azalır ya da algoritma başarısız olduğu için model oluşturulamaz. Bir veri kümesinden veri çıkarmak için, veri temizlenmeli ve veri madenciliği süreci için hazırlanmalıdır. Veri kümesindeki kayıp değer problemini çözmenin bir yolu, eksik değere sahip örneği kaldırmaktır. Bununla birlikte, bazı dezavantajları vardır. Bu dezavantajlara örnek olarak, çıkarılan örneklerin sınıflandırma algoritmaları için çok önemli özelliklere sahip olması ve örneğin yeniden ölçülememesi gösterilebilir. Ayrıca veri örneklerinin sayısının azalması, veri madenciliği ve veri toplama başarısını zayıflatmaktadır. Bu nedenle veri setindeki her örneğin çok değerli olduğu düşünüldüğünde doğru çözüm eksik değerleri silmek yerine uygun ve mantıksal değerlerle doldurmaktır. Bu çözümü sağlamak için çeşitli eksik değer atama yöntemleri geliştirilmiştir (Çetin ve Yıldız, 2021):

Ortalama, Mod, Medyan: Bu yöntem, bir sütunda (veya öznitelikte) eksik olmayan ve eksik değerle aynı sınıfa ait değerlerin ortalamasını, modunu veya medyanını hesaplayarak sütunlardaki eksik değerleri ayrı ayrı doldurmaktadır. Kategorik verilerin ortalaması alınamadığı için sadece sayısal verilerle kullanılabilmektedir (Çetin ve Yıldız, 2021). Bu nedenle EK 3'te "Yes" ve "No" değerlerini içeren kategorik öznitelikler 'potential_issue', 'deck_risk', 'oe_constraint', 'ppap_risk', 'stop_auto_buy', 'rev_stop', 'went_on_backorder', "Yes":1 ve "No":0 olacak şekilde sayısal (nümerik) veri tipine dönüştürülmüştür.

Bu tez çalışmasında Lead_time özniteliği kayıtlarında eksik değerler tespit edilmiştir. Eksik değerler veri seti için önemli ölçüde kayıt barındırdığı için bu kayıtların silinmemesine karar verilmiştir. Kayıtların silinmesi yerine eksik değerlerin bulunduğu veriler medyan değeri ile değiştirilmiştir. Python kodları ile eksik değerlerin medyan değeri ile tamamlanması aşaması EK 3'te belirtilmiştir.

3.1.3. Farklı Ölçeklerdeki Verilere Normalizasyon İşlemi

Sayısal özelliklerin farklı ölçekleri vardır ve bu bazı makine öğrenimi algoritmaları için sorun olmaktadır. Normalleşme/standartlaşmanın temel fikri hep aynıdır. Farklı ölçeklerde ölçülen değişkenler, model uyumunu ve model öğrenme işlevini tek tip olarak etkilemez ve önyargılara neden olabilir. Bu nedenle, bu potansiyel sorunu ele almak için modeli eğitmeden ve model üzerinden çıkarımda bulunmadan önce, özelliğe özgü normalleştirme (örneğin, Min-Maks Ölçeklendirme) sıklıkla kullanılmaktadır. Girdilerin özniteliklerini/değişkenlerini normalleştirmenin başka bir yolu (işlevleri μ=0 ve σ=1 olacak şekilde ölçekleyen normalleştirme yönteminin yanı sıra) Min-Maks ölçeklendirmedir. Min-Maks normalleştirme yöntemi kullanılarak, tüm nitelikler [0,1] aralığına dönüştürülmektedir. Özniteliğin/değişkenin minimum ve maksimum değerleri sırasıyla 0 ve 1 olacaktır. İlgili matematiksel fonksiyon Denklem 3.1' de verilmiştir. Bu denklemde x, tek bir öznitelik/değişken vektörünü temsil etmektedir ("Towards Data Science", 2023).

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

Bu çalışmada, Python sklearn, preprocessing ve pandas kütüphaneleri indirilerek öznitelikler benzer ölçeğe sahip olacak şekilde Min-Maks normalizasyonu ile [0,1] aralığına ölçeklendirilmiştir. EK 4'te normalizasyon aşaması Python kodları ile verilmiştir.

3.1.4. Dengesiz Veri Setinin Dengeli Hale Getirilmesi

Eşit olmayan bir sınıf dağılımına sahip herhangi bir veri seti teknik olarak dengesizdir. Bununla birlikte, her problem sınıfı için örnek sayısı arasında önemli veya bazı durumlarda aşırı bir dengesizlik varsa, bir veri kümesi dengesizdir. Başka bir deyişle sınıf dengesizliği, bir sınıfı temsil eden örnek sayısı diğer sınıflardan çok daha az olduğunda ortaya çıkmaktadır. Bu nedenle, bir veya daha fazla kategori, veri kümesinde yetersiz temsil edilebilmektedir. Dengesiz sınıflandırma hakkındaki literatürün çoğu, bir sınıfın

diğerine göre önemli ölçüde üstün olduğu (dolayısıyla yeterince temsil edilmediği) ikili sınıflandırma problemlerine ayrılmıştır. Bununla birlikte, çarpık sınıf dağılımlarına sahip çok sınıflı problemler de vardır (Fernández, García, Galar, Prati, Krawczyk ve Herrera, 2018).

Örnekleme yaklaşımları ikiye ayrılmaktadır. Bunlardan ilki, her sınıftaki örnek sayısını eşitlemek amacıyla çoğunluk sınıfına ait örnekleri eleyerek verileri azaltmaktan oluşan eksik örnekleme (undersampling) yaklaşımıdır. İkincisi ise, önem kazanmak için yeni pozitif örnekleri çoğaltmayı veya üretmeyi amaçlayan aşırı örnekleme (oversampling) yaklaşımıdır. Klasik yöntemler olarak toplam dokuz teknik ele alınmaktadır. Bu yöntemlerden ikisi, rastgele alt örnekleme (random undersampling) ve rastgele aşırı örnekleme (random oversampling), başlangıçta bu değerlendirme için temel yöntemler olarak dahil edilen buluşsal olmayan yöntemlerdir. Ancak, genellikle çok kısa yanıt sürelerinde iyi sonuçlar elde etmektedir.

• Rastgele Alt Örnekleme Tekniği (RUS): Bu, dengeli bir örnek seti elde etmek için çoğunluk sınıfından rastgele örnekler çıkararak sınıfların dağılımını dengelemeyi amaçlayan sezgisel olmayan bir tekniktir. Nihai denge oranı ayarlanabilmektedir.

• Rastgele Aşırı Örnekleme (ROS): Azınlık sınıfından örnekleri rasgele çoğaltarak sınıf dağılımını dengelemeyi amaçlayan, buluşsal olmayan başka bir yöntemdir (Fernández ve diğerleri, 2018).

Bu çalışmada dengesiz veri setinin dengeli hale getirilmesi amaçlanmıştır. Büyük veriye sahip olunduğu için örnekleri çoğaltmak yerine Rastgele Alt Örnekleme (RUS) tekniği kullanılarak baskın sınıfa ait örnekler azınlık sınıfa ait örneklerin sayısına eşitlenmiştir. Veri ön işleme aşamaları tamamlandıktan sonra veri seti Excel üzerinden kayıt edilmiştir.

Veri ön işleme adımları tamamlandıktan sonra veri setinin %80'i eğitim %20'si ise test verisi olarak ayrılmıştır. Aşağıda sırasıyla belirtilen üç yapay zeka tekniği ile modeller eğitilerek ön sipariş tahmini için model performansları değerlendirilmiştir.

3.2. Yapay Sinir Ağları

Yapay sinir ağları, insan beyninin en temel işlevi olan öğrenmeyi gerçekleştiren bir bilgisayar sistemidir. Öğrenme sürecini yönlendirmek için örnekler kullanırlar. Bu ağlar, birbirine bağlı süreç öğelerinden (yapay nöronlar) oluşur. Her bağlantının bir ağırlık değeri vardır. Yapay sinir ağı bilgisi bu ağırlık değerlerinde gizlenerek ağa iletilir (Öztemel, 2012).

Ortama adaptasyon sağlayan, uyarlanabilir, eksik bilgiyle çalışabilen, belirsiz şartlar altında karar verebilen ve hataya dayanıklı bu yöntemin başarılı uygulamaları hayatın hemen her alanında görülebilir. Özellikle yapay sinir ağı en etkili sınıflandırma, örüntü tanıma, sinyal filtreleme, veri sıkıştırma ve optimizasyon teknikleri arasında sayılmaktadır (Öztemel, 2012).

Yapay sinir ağları, canlı organizmaların sinir sistemlerini inceleyerek geliştirilmiştir. Canlı organizmalardaki sinir hücreleri birbirleriyle sinapslar aracılığıyla iletişim kurmaktadır. Sinir hücreleri, işlenmiş bilgileri aksonları aracılığıyla diğer hücrelere iletir. Yapay nöronlar da benzer şekilde toplama fonksiyonları ile dışarıdan gelen bilgileri toplar, aktivasyon fonksiyonları aracılığıyla iletir, çıktılar üretir ve ağ bağlantıları vasıtasıyla diğer hücrelere (işleme elemanları) gönderir. Çeşitli toplama ve aktivasyon fonksiyonları vardır.

Yapay sinir ağını birbirine bağlayan bağlantıların değerleri ağırlık değerleri olarak adlandırılmaktadır. İşleme elemanları, bir ağ oluşturmak için birbirine paralel üç tip katman halinde birleştirilir.

- Giriş Katmanı
- Ara Katman / Katmanlar
- Çıkış Katmanı

Akışta giriş katmanından ağa bilgi gönderilir, bilgiler orta katmanda işlenir ve oradan çıkış katmanına gönderilir. Bilgi işleme, ağa bilgi girişinin, ağın ağırlık değerleri kullanılarak çıktıya dönüştürülmesidir. Girdi verildiğinde ağın doğru çıktıyı üretebilmesi için ağırlıkların doğru değerler olması gerekir. Doğru ağırlık değerlerini bulma işlemine ağın eğitimi denir. Bu değerler başlangıçta rastgele atanır (Öztemel, 2012). Ardından, eğitim sırasında ağda her örnek göründüğünde, ağırlıklar ağın öğrenme kurallarına göre değiştirilir. Daha sonra başka bir örnek ağa sunulur ve ağırlıklar yeniden değiştirilerek en doğru değer bulunmaya çalışılır. Bu işlemler, ağ eğitim setindeki tüm örnekler için doğru çıktı üretene kadar tekrarlanır. Daha sonra, test setinin bir örneği ağa sunulur. Bir ağ, test setindeki örneklere doğru yanıt verirse eğitilmiş kabul edilir. Net ağırlıklar belirlendikten sonra, her bir ağırlığın ne anlama geldiği açık değildir. Bu nedenle yapay sinir ağları "kara kutu" olarak adlandırılır. Ancak ağın zekasının bu ağırlıklarda depolandığı söylenebilir. Çünkü ağ, girdileri hakkında kararlar almak için bu ağırlıkları kullanır. Bir ağ, bir olayı o olay için en doğru sinir ağı modelini seçerek öğrenebilir. Yıllar içinde birçok yapay sinir ağı modeli geliştirilmiştir. Aşağıdaki bilgiler yapay sinir ağı modelini karakterize etmektedir (Öztemel, 2012).

- Ağ Topolojisi
- Kullanılan Toplama Fonksiyonu
- Kullanılan Aktivasyon Fonksiyonu
- Öğrenme Stratejisi
- Öğrenme Kuralı

Geliştirilen modeller arasında en yaygın kullanılan ağlar tek katmanlı ve çok katmanlı Perseptron (Basit tek katmanlı algılayıcılar), LVQ (Lineer Vektör Uzayı), ART Network (Adaptif Rezonans Teori Ağları), SOM, Elman Network (Geri Dönüşümlü Ağlar)' tür (Öztemel, 2012).

Yapay sinir ağları yapısal olarak da sınıflandırılabilir. Yapısal olarak, yapay sinir ağları bilgi akışının yönüne ve nöronlar arasındaki bağlantıların yapısına bağlı olarak ileri

beslemeli ve geri beslemeli olmak üzere iki ana gruba ayrılabilir (Yazıcı, Öğüş, Ankaralı, Canan, Ankaralı ve Akkuş, 2007).

1. İleri beslemeli ağlar: Giriş katmanından çıkış katmanına bilgi akışının tek yönlü olduğu ve ara (gizli) katmandan geçtiği ağlardır. Girdi katmanı, gizli katman ve çıktı katmanı olmak üzere üç katmandan oluşur. Ağdaki bilgi akışı girdi katmanından çıktı katmanına doğru hareket eder. Yani nöronlar sırayla beslenir.

2. Geri beslemeli ağlar: Bilginin sinirsel çıkışlardan girişlere aktığı ağlardır. Bu tür ağ yapılarının geri bildirim bağlantıları vardır.

Hem ileri hem de geri olarak tanımlanabilen ağ yapıları da vardır. Tüm yapay sinir ağı modelleri arasında en yaygın olarak kullanılan ağlar, geri yayılımla eğitilmiş çok katmanlı ileri beslemeli ağlar (geri yayılım ağları), radyal tabanlı ağlar, Hopfield ve Kohonen'dir. Geri yayılımla eğitilmiş çok katmanlı ileri beslemeli ağlar, uygun ve güvenilir olması nedeniyle en yaygın kullanılan ağ türüdür. En önemli özelliği tahmin ve sınıflandırma işlemleri için çok uygun olması ve doğrusal olmayan yapıya sahip modeller için çok kullanışlı olmasıdır (Yazıcı ve diğerleri, 2007).

Yapay Sinir Ağlarının avantajları aşağıda verilmektedir (Öztemel, 2012):

- Yapay sinir ağı doğrusal değildir. Bu özellik, herhangi bir alana uygulanmasına izin verir. Herhangi bir sürekli fonksiyon ve onun türevleri üzerinde yakınsama yeteneğine sahiptir ve bu bakımdan evrensel fonksiyonlar yakınsama yöntemleri olarak tanımlanır. Yapay sinir ağları, özellikle doğrusal olmayan zaman serilerindeki başarılarından dolayı tercih edilen tahmin aracıdır.
- Öğrenme yeteneği, bilinen örnekleri kullanarak daha önce deneyimlenmemiş durumlardan genellemeler yapmasını, böylece bozuk veya kayıp verilere çözüm getirmesini sağlar. Eksik bilgileri işleyerek de sonuçlar çıkarılabilir.
- Sınırsız sayıda değişken ve parametre kolayca değiştirilebilir. Yapay sinir ağları, özellikle sorunu çözmek için kesin bir matematiksel model veya algoritma

olmadığında, doğrusal olmayan, çok boyutlu, hatalı veya eksik gözlemlerin, büyük grup içi varyansların analizi için önemli araçlar haline gelmiştir.

- Bir yapay sinir ağı, çeşitli şekillerde birbirine bağlı birçok hücreden oluşur, dolayısıyla paralel dağıtılmış bir yapıya sahiptir ve ağa ait bilgiler ağdaki tüm bağlantılara dağıtılır. Bu nedenle, eğitilmiş bir yapay sinir ağında bazı bağlantıların veya bazı hücrelerin devre dışı bırakılması, ağın doğru bilgi üretme yeteneğini önemli ölçüde etkilemez. Bu nedenle, hatalara tahammül etme yeteneği çok yüksektir.
- Değişken uzayı karmaşık olduğunda ve verilerin dağılımı bilinen istatistiksel dağılımdan saptığında, istatistiksel sınıflandırma yöntemleri daha güvenilir sonuçlar verir. Örneğin, geleneksel parametrik istatistiksel sınıflandırma yöntemleri Öklid, "maksimum olasılık" ve "Mahalanobis" mesafe sınıflandırıcılarını kullanmaktadır. Bu yaklaşımlar, sınıflandırılmış verilerin çok değişkenli normal dağılım varsayımını gerektirir. Değişken uzayındaki her özelliğin çok değişkenli bir normal dağılıma sahip olduğu varsayılır. Ancak veriler bu varsayımı destekleyemez. Bu durumda, dağıtımdan bağımsız bir yapay sinir ağı kullanılması önerilir. Sınıflandırma için kullanılan ve denetimli öğrenme ile eğitilen yapay sinir ağları, sınıflandırma için kullanılan k-en yakın komşu algoritmalarına benzer, ancak yapay sinir ağlarından elde edilen sonuçların daha güvenilir olduğu kanıtlanmıştır.

Yapay Sinir Ağlarının dezavantajları aşağıda verilmektedir (Öztemel, 2012):

- Sinir ağlarını eğitmek ve test etmek, yeterince büyük veri kümeleri gerektirir.
- Uygun bir ağ yapısı genellikle deneme yanılma yoluyla belirlenir.
- Gizli katman nöronlarının sayısı, tanıma işleminin doğruluğunu ve eğitim hızını etkiler. Örneğin, gizli katmandaki az sayıda nöron karmaşık ilişkileri bulamazken, çok sayıda nöron hesaplama açısından çok yoğundur.
- Basit sayılabilecek modelleme yapılarının uygulanması zor ve karmaşık olabilir.
- Diğer birçok tahmin yöntemi gibi, bir model oluşturmaz veya parametre tahminleri sağlamaz.

- Eğitimi iyi yapılamazsa ezberleme (overfitting) eğilimindedir.

Derin öğrenme, beynin yapısal ve işlevsel özelliklerinden ilham alan çok katmanlı ağ yapıları olan "yapay sinir ağları" üzerinde çalışan bir dizi algoritma ve modeldir. Derin sözcüğü birden fazla gizli katmanı (hidden layer) ifade etmektedir. Yapay sinir ağları, 1980'lerin sonunda "geriye yayılım" (back-propagation) yönteminin keşfedilmesi ve sonuçlarının iyi olmasıyla yeniden popüler hale gelmiştir. Derin öğrenme modellerinin klasik yapay sinir ağlarından farkları vardır. Bu farklar içerisinde yeni olan maddeler aşağıda sıralanmıştır (Akbaş, 2018):

- Daha fazla bilgi/veri ve daha fazla işlem gücü
- Yeni doğrusal olmayan (İng. non-lineer) aktivasyon fonksiyonları
- Yeni ilkleme (initialization) yöntemleri
- Yeni düzenlileştirme (regularization) yöntemleri

Yeni düzenlileştirme yöntemi dropout, aşırı uyumu önleyen yeni yöntem olarak 2009 ve sonrasındaki sistemlerin başarısında önemli rol oynamıştır.

Derin öğrenmenin sık kullanılmasının nedeni birçok farklı problem için geleneksel makine öğrenimi yöntemlerinden çok daha yüksek doğruluk sağlamasıdır. Doğruluk seviyesinin ticari uygulamalara olanak sağlaması da tercih nedeni olarak belirtilebilir. Ayrıca derin öğrenme yeni uygulamalara olanak sağlamaktadır (Akbaş, 2018).

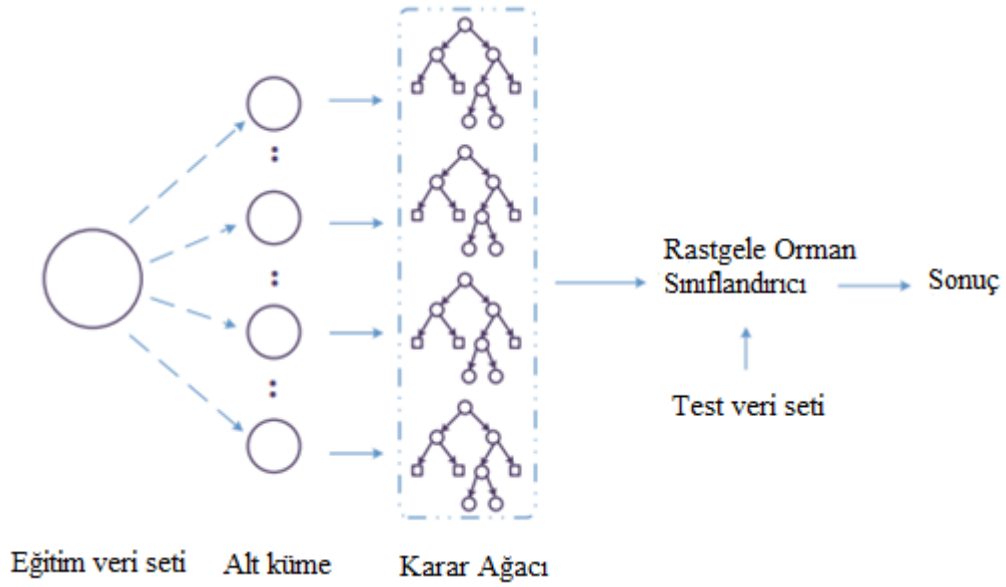
Derin Öğrenmede üç ana derin model türü bulunmaktadır:

- Çok Katmanlı Algılayıcılar (Multilayer Perceptrons)
- Evrişimli Sinir Ağları (Convolutional Neural Networks)
- Tekrarlayan Sinir Ağları (Recurrent Neural Networks)

3.3. Rastgele Orman

Rastgele karar ormanı olarak da bilinen Rastgele Ormanlar (Random Forest, RF), fazla uyumu (modelin ezberlemesini) etkili bir şekilde önleyebilen genel bir öğrenme algoritmasıdır. RF algoritması birden fazla Karar Ağacından (Decision Tree, DT) oluşmaktadır. Sınıflandırma sürecinde, eğitim kümesinin bir alt kümesi rastgele olarak seçilir ve her bir alt küme, bir karar ağacına karşılık gelmektedir. Karar ağaçları daha sonra ayrı eğitim sonuçları elde etmek için bağımsız olarak eğitilmektedir. Her ağaç bağımsızdır ve birbirini etkilemez. Nihai karar, her DT'nin oylama sonuçlarına dayalı olarak bir sınıflandırıcı tarafından belirlenmektedir (Wang, Zhu ve Qian, 2023).

RF, tek bir DT'ye kıyasla daha iyi sınıflandırma performansına sahiptir ve güçlü genelleme yeteneği ile eğitim sürecinde aşırı uydurma sorununun üstesinden gelir. Bir RF'nin temel modeli Şekil 3.1.'de gösterilmiştir (Wang ve diğerleri, 2023).



Şekil 3.1. Rastgele Ormanlar Algoritmasının Şeması (Wang ve diğerleri, 2023)

Rastgele Ormanlar, "sınıflandırma" olarak adlandırılan kategorik bir yanıt değişkeni veya "regresyon" olarak adlandırılan sürekli bir yanıt için kullanılabilir. Benzer şekilde, tahmin değişkenleri kategorik veya sürekli olabilmektedir (Cutler, Cutler ve Stevens, 2012).

Hesaplamalı bir bakış açısından, Rastgele Orman algoritmalarının avantajları aşağıda verilmektedir (Cutler ve diğerleri, 2012):

- Hem regresyon hem de (çok sınıflı) sınıflandırmayı doğal olarak ele almaktadır.
- Eğitilmesi ve tahmin edilmesi nispeten hızlıdır.
- Yalnızca bir veya iki ayar parametresine bağlıdır.
- Yerleşik bir genelleme hatası tahminine sahiptir.
- Yüksek boyutlu problemler için doğrudan kullanılabilir.
- Paralel olarak kolayca uygulanabilir.

İstatistiksel olarak, Rastgele Ormanlar sağladıkları ek özellikler nedeniyle cazip bir yöntemdir. Bu ek özelliklerden bazıları diferansiyel sınıf ağırlıklandırması, eksik değer yükleme, aykırı değer tespiti, görselleştirme ve denetimsiz öğrenmedir (Cutler ve diğerleri, 2012).

Adından da anlaşılacağı gibi, bir Rastgele Orman, her ağacın rastgele değişkenler koleksiyonuna bağlı olduğu ağaç tabanlı bir topluluktur. Daha resmi olarak, gerçek değerli girdiyi veya tahminleyici değişkenleri temsil eden bir p-boyutlu rasgele vektör $X = (X_1 , \ldots , X_p)^T$ ve gerçek değerli yanıtı temsil eden bir rastgele değişken Y için, $P_{xy}(X, Y)$ bilinmeyen bir ortak dağılım varsayılmaktadır. Amaç, Y'yi tahmin etmek için bir tahmin fonksiyonu $f(X)$ bulmaktır. Tahmin fonksiyonu, bir kayıp fonksiyonu $L(Y,f(X))$ tarafından belirlenir ve kaybın beklenen değerini en aza indirmek için tanımlanır (Cutler ve diğerleri, 2012).

$$E_{XY}(L(Y, f(X))) \tag{3.2}$$

Bu denklemde alt simgeler, X ve Y'nin ortak dağılımına ilişkin beklentiyi gösterir. Sezgisel olarak, $L(Y, f(X))$ fonksiyonu *f(X)*'in Y'ye ne kadar yakın olduğunun bir ölçüsüdür; Y'den çok uzakta olan *f(X)* değerlerini cezalandırır. L'nin tipik seçimleri, regresyon için kare hata kaybı $L(Y, f(X)) = (Y - f(X))^2$ ve sınıflandırma için sıfır-bir kayıptır:

$$L(Y, f(X)) = I(Y \neq f(X)) = \begin{cases} 0, & Y = f(X) \\ 1, & diğer\ durumda \end{cases} \quad (3.3)$$

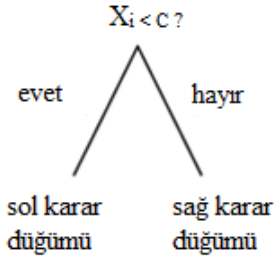
Hata kaybının karesi için $E_{XY}(L(Y, f(X)))$'nin en aza indirilmesinin koşullu beklentiyi verdiği ortaya çıkmıştır.

$$f(X) = E(Y \mid \mathrm{X} = \mathrm{x}) \quad (3.4)$$

Aksi takdirde regresyon işlevi olarak bilinmektedir. Sınıflandırma durumunda, Y'nin olası değerleri kümesi Y ile gösterilirse, *Exy'yi (L(Y, f(x)))* sıfır-bir kayıp için minimizasyonu verir.

$$arg \max_{y \in Y} P(Y = y \mid \mathrm{X} = \mathrm{x}) \quad (3.5)$$

Bayes kuralı olarak da bilinmektedir. Şekil 3.2.'de bölme noktası c kullanılarak sürekli tahmin değişkeni X_i'yi bölme durumu karar ağacı yapısında gösterilmiştir (Cutler ve diğerleri, 2012).



Şekil 3.2. Bölme noktası c kullanılarak sürekli tahmin değişkeni X_i'yi bölme (Cutler ve diğerleri, 2012)

Topluluklar *f*'yi sözde "temel öğrenenler" $h_1(x), \ldots, h_j(x)$, koleksiyonu açısından oluşturur ve bu temel öğreniciler bir araya getirilerek "topluluk öngörücüsü" $f(X)$ elde edilir. Regresyonda, temel öğrenicilerin ortalaması alınmaktadır.

$$f(X) = \frac{1}{J}\sum_{J=1}^{J} h_j(x) \tag{3.6}$$

Sınıflandırmada $f(X)$ en sık tahmin edilen sınıftır.

$$f(X) = \text{argmax}\, y \in y \sum_{J=1}^{J} I(y = h_j(x)) \tag{3.7}$$

Rastgele Ormanlarda *j*. temel öğrenici, $hj(X, \Theta j)$ olarak gösterilen bir ağaçtır; burada Θj, rasgele değişkenlerin bir koleksiyonudur ve Θj'ler, $j = 1,. . . , J$ için bağımsızdır. Bir Rastgele Ormanın tanımı çok genel olsa da, neredeyse her zaman özel şekilde uygulanırlar. Rastgele Orman algoritmasını anlamak için, temel öğreniciler olarak kullanılan ağaç türleri hakkında temel bilgilere sahip olmak önemlidir (Cutler ve diğerleri, 2012).

Rastgele Ormanlar ilk andan itibaren oldukça iyi çalışmakla ünlü olsalar da, belirli durumlar için gelişmiş doğruluk sağlamak üzere ayarlanabilecek üç parametre vardır:

- *m*, her düğümde seçilen rastgele seçilmiş tahminleyici değişkenlerin sayısı

- *J*, ormandaki ağaç sayısı
- *tree size*, bölme için en küçük düğüm boyutu veya maksimum uç düğüm sayısı ile ölçülen ağaç boyutudur.

Rastgele Ormanların biraz hassas olduğu bu parametrelerden yalnızca biri *m* gibi görünmektedir. Sınıflandırmada, standart varsayılan $m = \sqrt{M}$'dir; burada *M*, tahmin edicilerin toplam sayısıdır. Regresyonda varsayılan, *m* = *N*/3'tür; burada *N*, örneklem büyüklüğüdür. Ayarlama gerekli ise, düğümün dışında kalan hata oranı kullanılarak *m* seçilebilir, ancak bu durumda bu artık genelleme hatasının yansız bir tahminini vermeyecektir. Bununla birlikte, tipik olarak Rastgele Ormanlar *m*'ye karşı çok duyarlı değildir, bu nedenle ince ayar gerekli değildir ve *m* seçimine bağlı aşırı uyum (overfitting) etkileri nispeten küçük olmalıdır.

Pek çok topluluk yöntemi için, başlangıçta *J* arttıkça genelleme hatası azalır, ancak bir noktada *J* çok büyük olur ve genelleme hatasında ilişkili bir artışla birlikte fazla uyum devreye girmektedir. Rastgele Ormanlarda durum böyle değildir. *J*'nin küçük değerleri için, düğümden çıkan tahmin kararsız ve hatalı olabilmektedir. Ancak, *J* arttıkça Breiman (2001), Rastgele Ormanlar için genelleme hatasının neredeyse kesinlikle bir sınıra yaklaştığını göstermiştir. Pratikte bu, genelleme hatasını artırma korkusu olmadan *J*'nin istenildiği kadar büyük seçilebileceği anlamına gelmektedir. *J* ile ilgili tek gerçek endişe, çok küçük olmaması ve genellikle *J*'nin tahmin edilen genelleme hatasının dengelenmesini sağlayacak kadar büyük olduğu zaman, düğümden çıkan hata oranının kullanılabilmesidir.

Breiman, (2001) tarafından yapılan bir çalışma, çok büyük ağaçların yetiştirilmesini önermektedir. Çok büyük ağaçlarla ilgili problemler için, kullanıcılar düğüm sayısını veya en küçük düğümün boyutunu ayar için kullanmaktadır. Düğüm dışı hata oranı, ayar parametresini seçmek için kullanılabilir. Bu tür bir kullanım tahmini genelleme hatasını etkiliyor görünmektedir (Cutler ve diğerleri, 2012).

Özetle Rastgele Ormanlar, çok sınıflı sınıflandırma da dahil olmak üzere hem regresyon hem de sınıflandırma problemlerine uygulanabilen çok amaçlı bir makine öğrenme yöntemidir. Rastgele Ormanlar, genellemenin dahili bir tahminini sağlar, dolayısıyla bu yöntem için çapraz doğrulama gerekli değildir. Bunlar ayarlanabilir, ancak genellikle varsayılan ayar parametreleriyle oldukça iyi çalışırlar. Değişken seçimi için kullanılabilecek değişken önem ölçüleri mevcuttur. Rastgele Ormanlar, eksik değerleri atamak için kullanılabilecek yakınlıklar oluşturmaktadır. Yakınlık, verilerin yeni görselleştirmelerini sağlayarak çok bilgilendirici olmaktadır. Rastgele Ormanlar, birçok farklı uygulamada başarıyla kullanılmıştır ve birçok farklı alanda önemli bir popülariteye sahiptir (Cutler ve diğerleri, 2012).

3.4. Aşırı Gradyan Artırma Algoritması (XGBoost)

Büyük verilerin hız kazanmasıyla birlikte, doğruluğu yüksek tahminler yapmak için uygun makine öğrenimi algoritmaları bulma arayışı başlamıştır. Karar ağaçları, çok doğru olan fakat yeni verileri iyi bir şekilde genellemeyen makine öğrenimi modelleri üretmiştir. Grup yöntemlerinin, düğümleri ve artırmayı kullanarak birçok karar ağacını birleştirerek daha verimli olduğu gösterilmiştir. Gradyan artırma algoritması ağaç topluluğu yörüngesinden ortaya çıkan olağanüstü bir algoritmadır (Chen ve Guestrin, 2016).

Yapay zekanın bileşenlerinden biri olan büyük veriler ile uğraşırken daha hızlı algoritmalara duyulan ihtiyaç açıktır. Extreme Gradient Boosting yönteminde algoritmanın adında kullanılan Extreme ifadesi hesaplama sınırlarını en uç noktaya kadar zorlamak anlamına gelmektedir. Hesaplama sınırlarını zorlamak, yalnızca model oluşturma değil, aynı zamanda disk okuma, sıkıştırma, önbellek ve çekirdekler hakkında da bilgi gerektirmektedir (Chen ve Guestrin, 2016).

XGBoost yöntemi eksik değerleri işleme konusunda oldukça yeteneklidir. Herhangi bir değere ayarlanabilecek eksik bir hiperparametre olması durumunda, eksik bir veri noktası verildiğinde, XGBoost farklı bölme seçeneklerini puanlar ve en iyi sonucu vereni seçer.

XGBoost tekniği özellikle hız için tasarlanmıştır. Hızdaki artış, özellikle milyonlarca, milyarlarca veya trilyonlarca satırlık veriyi işlerken önemli olan makine öğrenimi modellerini daha hızlı oluşturmayı mümkün kılar. Bu, endüstri ve bilimin her zamankinden daha fazla veri topladığı büyük veri dünyasında alışıldık bir durumdur. Aşağıdaki yeni tasarım özellikleri, XGBoost'a karşılaştırılabilir topluluk algoritmalarına göre önemli bir hız avantajı sağlamaktadır (Chen ve Guestrin, 2016):

- Yaklaşık bölünmüş bulma algoritması (Approximate split-finding algorithm)
- Seyreklik farkında bölünmüş bulma (Sparsity aware split-finding)
- Paralel bilgi işlem (Parallel computing)
- Önbelleğe duyarlı erişim (Cache-aware access)
- Blok sıkıştırma ve parçalama (Block compression and sharding)

Karar ağaçlarının en iyi sonuçlar için en uygun düğümlerden bölünmeye ihtiyacı vardır. Açgözlü bir algoritma, her adımda en iyi ayrımı seçmekte ve önceki dallara bakmak için geri adım atmamaktadır. Karar ağacını bölmek genellikle açgözlülükle yapılmaktadır. XGBoost, yeni bir yaklaşık bölünmüş arama algoritmasına ek olarak tam bir açgözlü algoritma sunmaktadır. Yaklaşık bölünmüş bulma algoritması, aday bölmeler önermek için bölünmüş sayıları veya yüzdeleri kullanır. Genel tavsiye, eğitim sırasında aynı nicelikler kullanılır ve yerel öneri, her bölme turu için yeni nicelikler sunmaktadır (Chen ve Guestrin, 2016).

Seyrek veriler, girişlerin çoğu 0 veya boş olduğunda oluşmaktadır. Bu, veri kümeleri birincil olarak boş değerlerden oluştuğunda veya bunlar tek geçişte kodlandığında ortaya çıkmaktadır. Seyrek matrisler, yalnızca sıfır olmayan ve boş olmayan değerlere sahip veri

noktalarını depolamak için tasarlanmıştır. Bu değerli alandan tasarruf sağlamaktadır. Seyrekliğe duyarlı bir bölme veya bölmeler aranırken, matrisleri seyrek olduğu için XGBoost'un daha hızlı bir yöntem olduğu gösterilmiştir (Chen ve Guestrin, 2016).

Her ağaç bir önceki ağacın sonuçlarına bağlı olduğundan, hızlanma paralel bilgi işlem için ideal değildir. Ancak, paralelliğin ortaya çıkabileceği fırsatlar da bulunmaktadır. Paralel hesaplama, aynı sorunu aynı anda çözmek için birden fazla bilgi işlem birimi birlikte çalıştığında ortaya çıkmaktadır. XGBoost, verileri bloklar halinde sıralar ve sıkıştırır. Bu blokları birkaç makineye veya harici belleğe (çekirdek dışı) dağıtmak mümkündür. Bloklar kullanılarak veri sıralama daha hızlıdır. Bölünmüş bulma algoritması, bloklardan yararlanır ve bloklar nedeniyle nicel arama daha hızlıdır. Tüm bu durumlarda XGBoost, model oluşturma sürecini hızlandırmak için paralel bilgi işlem sağlamaktadır (Chen ve Guestrin, 2016).

İndirgeme (degrade) istatistikleri söz konusu olduğunda, XGBoost ön belleğe duyarlı erişim sağlamaktadır. XGBoost dahili bir arabellek ayırarak gradyan istatistiklerini getirmekte ve mini yığınlarla toplama gerçekleştirmektedir. XGBoost: A Scalable Tree Boosting System'e göre önceden getirme, okuma/yazma bağımlılığını uzatır ve çok sayıda satıra sahip veri kümeleri için çalışma sürelerini yaklaşık %50 azaltır (Chen ve Guestrin, 2016).

XGBoost, blok sıkıştırma ve blok parçalama yoluyla ek hız kazanımları sağlamaktadır. Blok sıkıştırma, sütunları sıkıştırarak hesaplama açısından pahalı disk okumaya yardımcı olmaktadır. Blok parçalama, verileri okurken değişen birden çok diske parçalayarak okuma sürelerini azaltmaktadır.

XGBoost, gradyan artırmanın ötesinde doğruluk kazanımları (accuracy gains) elde etmek için yerleşik düzenleme eklemektedir. Düzenlileştirme (Regularization), varyansı azaltmak ve aşırı uyumu önlemek için bilgi ekleme işlemidir.

Veriler hiperparametreler kullanılarak ayarlanıp düzenli hale getirilebiliyor olsa da düzenli algoritmalar denenebilir. XGBoost, gradyan artırma ve rastgele ormanların aksine, öğrenme hedefinin bir parçası olarak düzenlileştirmeyi içerir. Düzenlenmiş parametreler, karmaşıklığı cezalandırır ve fazla uyumu önlemek için nihai ağırlıkları yumuşatır. XGBoost, gradyan artırmanın düzenlileştirilmiş bir versiyonudur (Chen ve Guestrin, 2016).

Bir makine öğrenimi modelinin öğrenme hedefi, modelin verilere ne kadar iyi uyduğunu belirlemektir. XGBoost durumunda, öğrenme hedefi iki bölümden oluşmaktadır. Bunlar kayıp fonksiyonu ve düzenlileştirme terimidir. Matematiksel olarak, XGBoost'un öğrenme hedefi Denklem 3.8'deki şekilde tanımlanmaktadır.

$$obj(\theta) = l(\theta) + \Omega(\theta) \tag{3.8}$$

Burada $l(\theta)$, regresyon için Ortalama Karesel Hata (MSE) veya sınıflandırma için günlük kaybı olan kayıp işlevidir ve $\Omega(\theta)$, overfittingi önlemek için bir ceza terimi olan normalleştirme işlevidir. Amaç işlevinin bir parçası olarak bir düzenleme terimi dahil etmek, XGBoost'u çoğu ağaç topluluğundan ayırmaktadır.

Regresyon için MSE olarak tanımlanan kayıp fonksiyonu, Denklem 3.9'daki gibi yazılabilir:

$$l(\theta) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.9}$$

Burada $y(i)$, i. inci satırı için hedef değer ve $\hat{y}i$, i. inci satırı için makine öğrenmesi modeli tarafından tahmin edilen değerdir. Toplama sembolü Σ , i = 1 ile başlayan ve satır sayısı olan $i = n$ ile biten tüm satırların toplandığını göstermektedir.

Belirli bir ağaç için $\hat{y}i$ tahmini, ağaç kökünde başlayan ve bir yaprakta biten bir fonksiyon gerektirir. Matematiksel olarak, bu Denklem 3.10'da ifade edilmiştir.

$$\hat{y}_i = f(x_i), f \in F \tag{3.10}$$

Burada x_i, girişleri i. inci satırının sütunları olan bir vektördür ve $f \in F$, f fonksiyonunun olası tüm CART fonksiyonlarının kümesi olan F'nin bir üyesi olduğu anlamına gelmektedir. CART, Sınıflandırma ve Regresyon Ağaçlarının kısaltmasıdır. CART, sınıflandırma algoritmaları dahil tüm yapraklar için gerçek bir değer sağlamaktadır.

XGBoost'un birçok hiperparametresi mevcuttur. XGBoost temel öğrenici hiperparametreleri, tüm karar ağacı hiperparametrelerini bir başlangıç noktası olarak birleştirmektedir. XGBoost, gradyan artırmanın geliştirilmiş bir versiyonu olduğundan, gradyan artırma hiperparametreleri bulunmaktadır. XGBoost'a özgü hiperparametreler, doğruluğu ve hızı iyileştirmek için tasarlanmıştır. Çizelge 3.3.'te önemli XGBoost hiperparametreleri özetlenmiştir.

Çizelge 3.3. XGBoost hiperparametreleri (Chen ve Guestrin, 2016)

Hiperparametre	Değer	Aralık	Etki	Not
Tahminleyici Sayısı (n_estimators)	100	[1,∞)	Artırmak, büyük verilerde puanları iyileştirebilir.	Topluluktaki ağaç sayısı.
Öğrenme Oranı (learning rate)	0.3	[0,∞)	Azaltmak, aşırı uyumu önler.	Her artırma adımında ağaç ağırlıklarını küçültür.
Maksimum Derinlik (max depth)	6	[0,∞)	Azaltmak, aşırı uyumu önler.	Ağacın derinliği. 0, zarar odaklı büyüme politikasında bir seçenektir.
Gamma	0	[0,∞)	Artırmak, aşırı uyumu önler.	Genellikle 10'dan düşük olan düşük değerler standarttır.
min_child_weight	1	[0,∞)	Artırmak, aşırı uyumu önler.	Bir düğümün bölünmesi için gereken minimum ağırlık toplamı.
subsample	1	(0,1]	Azaltmak, aşırı uyumu önler.	Her artırma turu için eğitim satırlarının yüzdesini sınırlar.
colsample_bytree	1	(0,1]	Azaltmak, aşırı uyumu önler.	Her artırma turu için eğitim sütunlarının yüzdesini sınırlar.
colsample_bylevel	1	(0,1]	Azaltmak, aşırı uyumu önler.	Ağacın her derinlik düzeyi için sütunların yüzdesini sınırlar.
colsample_bynode	1	(0,1]	Azaltmak, aşırı uyumu önler.	Bölmeleri değerlendirmek için sütunların yüzdesini sınırlar.
scale_pos_weight	1	[0,∞)	toplam (negatifler) / toplam (pozitifler) verileri dengeler.	Dengesiz veri kümeleri için kullanılır.
max_delta_step	0	[0,∞)	Artırmak, aşırı uyumu önler.	Yalnızca son derece dengesiz veri kümeleri için önerilir.
lambda	1	[0,∞)	Artırmak, aşırı uyumu önler.	Ağırlıkların L2 düzenlenmesi.
alpha	0	[0,∞)	Artırmak, aşırı uyumu önler.	Ağırlıkların L1 düzenlemesi.
missing	-	(-∞,∞)	En uygun boş değerleri bulur.	Boş değerleri -999.0 gibi sayısal bir değerle değiştirilir, ardından -999.0'a olarak ayarlanır.

3.5. Performans Değerlendirme Metrikleri

Bir ikili sınıflandırma görevindeki herhangi bir sınıflandırıcının performansı genellikle accuracy (doğruluk), precision (kesinlik) ve recall (duyarlılık) ile tahmin edilmektedir. Bu standart değerlendirme ölçütleri, Çizelge 3.4.'te gösterilen karışıklık matrisi kullanılarak hesaplanmaktadır (Shajalal ve diğerleri, 2021).

Çizelge 3.4. Karışıklık matrisi (confusion matrix) (Shajalal ve diğerleri, 2021)

	Tahmin Edilen Negatif	Tahmin Edilen Pozitif
Gerçekleşen Negatif	Gerçek Negatif (TN)	Yanlış Pozitif (FP)
Gerçekleşen Pozitif	Yanlış Negatif (FN)	Doğru Pozitif (TP)

Performans analizi, accuracy, precision, recall, F1-Score vb. gibi çeşitli metriklere dayalı olarak algoritmayı tahmin etmek için kullanılan analizdir. Performans analizi, makine öğrenimi modellerinin doğruluğunu ve performansını karşılaştırmayı amaçlamaktadır. Metrikler dört ölçüt ile değerlendirilmektedir (Sanni ve Guruprasada, 2021).

Gerçek Pozitif (True Pozitif, TP): Sınıflandırıcıya pozitif girdi (veri kümesinden) verildiğinde, pozitif çıktı (öngörülen değer) oranını vermektedir. Doğru tanımlanmış toplam gerçek pozitif vakaları tahmin etmektedir. Pozitif değere sahip olan sınıf değişkeninin pozitif değer olarak doğru şekilde tahmin edilmesi durumudur (Sanni ve Guruprasada, 2021). Tez çalışmasında kullanılan veri seti örnek alındığında ön siparişe giren siparişler için gerçek pozitif değerler 1 (doğru) olacaktır. Gerçek pozitif değer 1 ise tahmin edilen değer 1 olacaktır.

Gerçek Negatif (True Negative, TN): Sınıflandırıcıya negatif girdi (veri setinden) verildiğinde, negatif çıktı (öngörülen değer) vermektedir. Doğru tespit edilen toplam gerçek negatif vaka sayısını tahmin etmektedir. Negatif değere sahip olan sınıf değişkeninin negatif değer olarak doğru şekilde tahmin edilmesi durumudur (Sanni ve Guruprasada, 2021). Tez çalışmasında kullanılan veri seti örnek alındığında ön siparişe giren siparişler için gerçek negatif değerler 0 (yanlış) olacaktır. Gerçek negatif değer 0 (yanlış) ise tahmin edilen değer 1 (doğru) olacaktır.

Yanlış Pozitif (False Positive, FP): Sınıflandırıcıya negatif girdi (veri kümesinden) verildiğinde, pozitif çıktı (öngörülen değer) vermektedir. Yanlış tanımlanan toplam yanlış

pozitif vakaları tahmin etmektedir. Pozitif değere sahip olan sınıf değişkeninin negatif değer olarak yanlış şekilde tahmin edilmesi durumudur. (Sanni ve Guruprasada, 2021). Tez çalışmasında kullanılan veri seti örnek alındığında ön siparişe giren siparişler için yanlış pozitif değerler 0 olacaktır. Yanlış pozitif değer 0 ise tahmin edilen değer 1 olacaktır.

Yanlış Negatif (False Negative, FN): Sınıflandırıcıya pozitif girdi (veri setinden) verildiğinde, negatif çıktı (öngörülen değer) vermektedir. Yanlış tanımlanan toplam yanlış negatif vakaları tahmin etmektedir. Negatif değere sahip olan sınıf değişkeninin pozitif değer olarak yanlış şekilde tahmin edilmesi durumudur (Sanni ve Guruprasada, 2021). Tez çalışmasında kullanılan veri seti örnek alındığında ön siparişe giren siparişler için yanlış negatif değerler 0 olacaktır. Öngörülen değerler yanlışsa tahmin edilen değer 0 olacaktır.

Performans analizi için önemli ölçütler accuracy (doğruluk), recall (duyarlılık), F1-score (F1- puanı) ve precision (kesinlik) metrikleridir. Yukarıdaki ölçütler, metriği tanımlamak için kullanılmaktadır (Sanni ve Guruprasada, 2021).

Doğruluk (Accuracy, A): Doğruluk, test verileri için doğru tahminlere sahip bir performans metriğidir. Verileri test etmek için doğru tahminlerin yüzdesini vermektedir. Makine öğreniminde doğruluk, Denklem 3.11'de gösterilen formül kullanılarak hesaplanmaktadır (Sanni ve Guruprasada, 2021).

$$A = \frac{(T\,P + T\,N)}{(T\,P + T\,N + F\,P + F\,N)} \tag{3.11}$$

Duyarlılık (Recall, R): Duyarlılık metriği, tüm örneklerin pozitif olarak tanımlandığı doğru örneklerin sayısını tahmin etmek için kullanılmaktadır. Toplam pozitifin yüzde

kaçının pozitif olduğunu tahmin etmektedir. Gerçek pozitifler ile gerçek pozitif ve yanlış negatiflerin arasındaki orandır. Tüm gerçek pozitif örneklerden doğru şekilde tanımlanmış pozitif örnekleri ölçmektedir (Sanni ve Guruprasada, 2021). Duyarlılık, Denklem 3.12'de gösterilen formül kullanılarak hesaplanmaktadır.

$$R = \frac{T\,P}{T\,P + F\,N} \tag{3.12}$$

Kesinlik (Precision, P): "Pozitif öngörü değeri" olarak da adlandırılan kesinlik metriğidir. Gerçek pozitiflerin yüzdesini verir. Gerçek pozitifler ile tahmin edilen tüm pozitifler arasındaki orandır. Pozitif olarak tahmin edilen tüm numunelerden doğru şekilde tanımlanmış pozitif numuneleri ölçer. Kesinlik, Denklem 3.13'te gösterilen formül kullanılarak hesaplanmaktadır (Sanni ve Guruprasada, 2021).

$$P = \frac{T\,P}{T\,P + F\,P} \tag{3.13}$$

F1 Puanı (F1−Score): Kesinlik (Precision) ve Duyarlılık (Recall) değerlerinin harmonik ortalamasını göstermektedir. Hesaplama formülü Denklem 3.14'te verilmiştir.

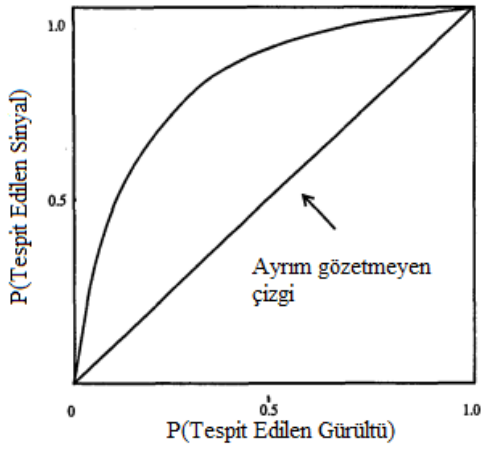
$$F_1 = \frac{2\,x\,R\,x\,P}{R + P} \tag{3.14}$$

Karışıklık matrisi (Confusion Matrix) analizinden elde edilen bir başka ölçü, yanlış pozitif oranıdır (false positive rate) (De Santis ve diğerleri, 2017). Hesaplama formülü Denklem 3.15'te verilmiştir.

$$F = \frac{F\,P}{F\,P + T\,N} \tag{3.15}$$

Alıcı Çalışma Karakteristikleri Eğrisi (Receiver Operating Characteristic, ROC) Analizi tekniği sinyal algılama cihazlarının (alıcıların) performansını değerlendirmek için geliştirildiği 1950'lerin başlarında elektrik mühendisliğinde ortaya çıkmıştır. Daha sonrasında diğer alanlara yayılarak psikolojide ve tıbbi teşhiste faydalı uygulamalar bulmuştur. Başlangıçta tasarlandığında, uçak arayan radar gibi bir alıcı sinyalleri dinlemektedir. Alıcı sürekli olarak az miktarda gürültü görür, bu nedenle gerçek bir sinyal ile arka plan gürültüsünü ayırt etmek için bir eşik ayarlanmalıdır. Eşiğin altındaki her şey "gürültü" olarak sınıflandırılırken, eşiğin üzerindeki her şey "sinyal" olarak sınıflandırılmıştır (Downey, Meyer, Price ve Spitznagel, 1999).

Eşik ne kadar düşük olursa, bir sinyalin algılanma olasılığı o kadar artar, ancak gürültünün sinyallerle karıştırılma olasılığı da o kadar artar. Eşik sürekli olarak değiştirildiğinde, yatay eksende sinyal olmadığında bir sinyali "algılama" şansını ve dikey eksende sinyal varken bir sinyali tespit etme şansı grafik olarak çizildiğinde, bir alıcı çalışma karakteristik eğrisine sahip olunmaktadır. Şekil 3.3.'te bir ROC eğrisi verilmektedir.



Şekil 3.3. ROC Eğrisi (Downey ve diğerleri, 1999)

ROC eğrisinin çok önemli iki özelliği vardır. Birincisi, tespit edilen gürültülerin yüzde kaçının gerçekte sinyal olduğuna bağlı değildir. İkincisi, alıcının herhangi bir ayırt etme yeteneği yoksa, eğri orijin noktasından (1,l) noktasına kadar düz bir çizgi olacaktır; aksi takdirde, bu çizginin üzerinde yer almalıdır. İkinci nokta bir adım daha ileri götürüldüğünde, eğri ideal olarak çok hızlı yükselmeli, (0,l) noktasına çok yaklaşmalı ve ardından (1,l) noktasına doğru hareket etmelidir. Yani, hemen hemen tüm sinyallerin algılandığı, ancak çok az gürültünün sinyalle karıştırıldığı bir eşik olmalıdır (Downey ve diğerleri, 1999).

ROC eğrisine dayalı performans ölçümlerinin en yaygın kullanılanı ve kapsamlısı, eğrinin altındaki alandır (Area Under the Curve, AUC), (Downey ve diğerleri, 1999). Değerlendirme metrikleri, modelin örnekleri farklı sınıflardan ne kadar iyi sıraladığına veya ayırdığına bağlıdır. Değerlendirme metrikleri genellikle grafiksel olarak gösterilir ve kullanıcının modelin performansını farklı eşiklerde görselleştirmesine ve metrikler arasındaki değiş tokuşu daha iyi anlamasına olanak tanımaktadır. Bununla birlikte, modeller arasında birkaç grafiği karşılaştırmak ve en iyisini tespit etmek, özellikle kesiştiklerinde genellikle zordur. Bu nedenle AUC, modeller arasında daha özlü ve kolay karşılaştırmalara izin veren tek değerli bir ölçüm sağladığı için genellikle hesaplanmakta ve kullanılmaktadır (Soares ve Torgo, 2021).

ROC eğrisi, gerçek pozitif oranı (recall) ile yanlış pozitif oranı karşılaştırmaktadır:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{3.16}$$

Denklem 3.16'daki her nokta bir eşik değerini temsil etmektedir (Soares ve Torgo, 2021).

Dengesiz problemlerde sınıflandırma modellerini değerlendirmek için kullanılan standart bir yaklaşım Alıcı Çalışma Karakteristiklerini (ROC) kullanmaktır. Kesinlik-hatırlama (precision-recall) eğrisi gibi, bu grafik de kesinlik ve düşme (precision and fall-out) arasındaki dengenin görselleştirilmesini sağlamaktadır. Bu durum herhangi bir sınıflandırıcının yanlış pozitifleri de artırmadan gerçek pozitiflerin sayısını artıramayacağını kanıtlamaktadır. ROC Eğrisinin Altındaki Alan (AUC), iki uyarandan hangisinin gürültü, hangisinin sinyal artı gürültü olduğunu doğru bir şekilde belirleme olasılığına karşılık gelmektedir. AUC, bir sınıflandırıcının ortalama olarak hangi modelin daha iyi olduğunu değerlendirme yeteneğinin tek bir ölçüsünü sağlamaktadır ve bu alanın nasıl hesaplandığına ilişkin formül aşağıdaki Denklem 3.17'de ele alınmıştır (De Santis ve diğerleri, 2017).

$$AUC = \frac{1+P-F}{2} \tag{3.17}$$

AUC, değerlendirme ve kıyaslama referansı için literatürde en çok uygulananlardan biri olmasına rağmen, diğer metrikler de sınıflandırıcı değerlendirmesinde kullanılmaktadır.

4. BULGULAR ve TARTIŞMA

Bu tezde, ön sipariş tahmin çalışması için yapay zeka yaklaşımına başvurulmuştur. Tez çalışmasının bu bölümünde TAZI programı üzerinde DNN, RF, XGBoost tekniklerinin Kaggle veri setine uygulanmasından ve elde edilen model performans sonuçlarından bahsedilmektedir.

Bu tez çalışmasında, diğer alanlarda performansı kanıtlanmış olan yapay zeka teknikleri ön sipariş tahmininde kullanılmıştır. Diğer çalışmalardan farklı olarak bu tez çalışmasında RUS yöntemi ile veri seti dengeli hale getirilmiştir. Veri ön işleme aşamasında Python ve makine öğrenimi algoritması uygulanması aşamasında TAZI kullanılmıştır. Bu tez çalışmasında, literatürde ön sipariş tahmini yapılan çalışmalara göre, daha iyi sonuçlar elde edilmiştir.

Derin Sinir Ağları ile yapılan deneyler EK 5'te verilmiştir. Yapılan deneylere göre en iyi performansa sahip model Deney 42'dir. DNN ile eğitilen modelin parametreleri; gizli katman sayısı 3, gizli katmanlardaki nöron sayıları sırası ile 130, 100 ve 70, dropout değeri ise 0.9'dur. Modelin performans metrikleri Precision = 0.8568, Recall = 0.8088, F1-Score = 0.8321 ve AUC = 0.902 olarak analiz edilmiştir. Model performansını literatürdeki çalışmalar ile karşılaştırıldığında Çizelge 4.1.'den de görüleceği gibi tez çalışmasında uygulanan DNN modeli;

- Hajek ve Abedin (2020) model performansından (AUC=0.6012) daha iyi sonuç vermiştir.
- Malviya ve diğerleri, (2020) model performansından (AUC=0.58) daha iyi sonuç vermiştir.
- Ntakolia ve diğerleri, (2021) model performans değeri (AUC=0.92) birbirine çok yakın sonuç vermiştir.

Rastgele Orman algoritması ile TAZI yapay zeka programında yapılan deneyler EK 6'da verilmiştir. Model eğitilirken hiperparametreler değiştirilerek deneyler çoğaltılmıştır.

Yapılan deneylere göre en iyi performansa sahip olan model Deney 16'dır. 0,959 AUC ve 0,8948 Recall performans sonuçlarına sahiptir. Model performansı literatürdeki çalışmalar ile karşılaştırıldığında Çizelge 4.1.'den de görüleceği gibi tez çalışmasında uygulanan RF modeli;

- Ntakolia ve diğerleri, (2021) model performansından (AUC= 0.95) daha iyi sonuç vermiştir.
- Hajek ve Abedin (2020) model performansından (AUC= 0.9157) daha iyi sonuç vermiştir.
- Malviya ve diğerleri, (2020) model performansından (AUC= 0.90) daha iyi sonuç vermiştir.
- De Santis ve diğerleri, (2017) model performansından (AUC= 0.9441) daha iyi sonuç vermiştir.

XGBoost algoritması ile yapılan deneyler EK 7'de verilmiştir. XGBoost algoritması ile TAZI yapay zeka programında yapılan deneyler sonucunda en iyi performansa sahip modelin değerlendirme metrikleri sırasıyla AUC = 0.948, Precision = 0.8757 ve Recall = 0.8829'dur. Modelde kullanılan hiperparametreler ise max depth = 6, gamma= 0.7, min_child_weight = 1 olarak belirlenmiştir. XGBoost algoritması model performansı literatürdeki çalışmalar ile karşılaştırıldığında Çizelge 4.1.'den de görüleceği gibi tez çalışmasında uygulanan XGBoost modeli;

- Ntakolia ve diğerleri, (2021) model performansına (AUC= 0.95) çok yakın sonuç vermiştir.
- İslam ve Amin (2020) model performansından (AUC= 0.946) daha iyi sonuç vermiştir.
- De Santis ve diğerleri, (2017) model performansı (AUC= 0.9482) ile benzer sonuç vermiştir.

Tez çalışmasında önerilen model sonuçlarına göre AUC metriği baz alındığında 0.959 değeri ile Rastgele Orman algoritması en iyi performansa sahip algoritmadır. Sonrasında

0.948 AUC değeri ile XGBoost ikinci en iyi performans gösteren algoritma olmuştur. Derin Sinir Ağı Algoritması ise 0.902 AUC performans skoru ile üçüncü sırada yer almıştır.

Bu tez çalışmasında bulunan değerler ile literatürdeki model performans değerlerinin karşılaştırılması Çizelge 4.1.'de yer almaktadır.

Çizelge 4.1. Tez çalışmasında önerilen model ve literatürdeki model performansları

Yazarlar	Hiperparametreler	Model	Precision	Recall	F1-score	AUC
Tez Çalışmasında Önerilen Model	n estimators = 75, min samples leaf = 5, min samples split = 3, Max Depth = 10	**RF**	**0.8921**	**0.8948**	**0.8934**	**0.959**
	layer = 3, hidden layer sizes = (130, 100, 70), Dropout = 0.9	**DNN**	**0.8568**	**0.8088**	**0.8321**	**0.902**
	Max Depth = 6, gamma= 0.7, min_child_weight = 1	**XGBoost**	**0.8757**	**0.8829**	**0.8793**	**0.948**
Liu ve diğerleri, (2022)	RUS: 0.2, learning rate: 0.0001, batch size: 256	RUS_CWGAN-RF	-	0.795	-	0.9644
Shajalal ve diğerleri, (2022)	Layer: Conv1D (20,32) Activation: Relu Dropout: (10,32)	ADASYN + CNN	-	-	-	0.9489
Lawal ve Akintola (2021)	batch size: 1000, epoch: 30, no of layers: 20	ADASYN + RNN	0.901	0.879	0.889	-
		RUS + RNN	0.841	0.885	0.862	-
		SMOTE + RNN	0.894	0.877	0.886	-
Ntakolia ve diğerleri, (2021)	Criterion = entropy, min samples leaf = 1, min samples split = 5, n estimators = 30	RF	0.8810	0.8994	0.8901	0.95
	activation = tanh, alpha = 0.0001, hidden layer sizes = (10, 20, 50), learning rate = constant, solver = adam	DNN	0.8596	0.8554	0.8575	0.92
	gamma = 0.7, max depth = 9, min child weight = 1	XGB	0.8738	0.9026	0.8880	0.95
Shajalal ve diğerleri, (2021)	Activation Function: Relu Hidden Layers: (22,16), (16,32), (32,64), (64,128), (128,128) Dropout: 0.5	Com_SMOTE_Under_DNN	-	-	-	0.9586
Hajek ve Abedin (2020)	maximum depth of 5 100 base classifiers	XGBoost	-	-	-	-
	$P < 0.05$	RF	-	-	-	0.9157
	-	DNN	-	-	-	0.6012
İslam ve Amin (2020)	The number of trees: 50-1000 Learning rate: 0.1 Maximum depth: 10 Sample rate: 0.9	Distributed RF	0.8231	0.8488	0.8357	0.959
	Learning rate: 0.1	Gradient Boosting Machine (GBM)	0.8869	0.7849	0.7845	0.946
Malviya ve diğerleri, (2020)	-	ANN	0.989	0.939	0.963	0.58
	-	Random Tree	0.997	0.877	0.933	0.90
De Santis ve diğerleri, (2017)	Number of estimators 10 Max. depth 5, 6, 7, 8, 9 Min. samples leaf 5	RF	-	-	-	0.9441
	Max. depth 5, 6, 7, 8, 9 Min. samples leaf 5	Gboost	-	-	-	0.9482

5. SONUÇ

Ön sipariş, tedarik zincirindeki herhangi bir eksiklik, artan ürün talebi veya ürün envanterinin tükenmesi nedeniyle müşterinin talep ettiği mal veya hizmetlerin gelecekte belirli bir tarihe kadar teslim edilmesini garanti edebilen bir siparištir. Beklenmeyen siparişler, ani talep artışları, kötü tedarik zinciri yönetimi ve yanlış envanter yönetimi durumlarında meydana gelir. Tedarik zincirindeki aksamalar, müşteri taleplerinde öngörülemeyen artışlar ve yanlış stok yönetimi durumlarında ön siparişler bir zorunluluk haline gelebilir.

Bu çalışmada, ön sipariş durumunun tahmini için yapay zeka makine öğrenmesi algoritmaları kullanılarak yapılan çalışma ve uygulamaların sonuçlarının paylaşılması amaçlanmıştır. Literatürdeki çalışmalarda kullanılan makine öğrenimi sınıflandırıcıları, yöntemleri ve gerçekleşen sonuçlar ile elde edilen algoritmaların başarı oranları açıklanmıştır. Ön sipariş tahmininin yapay zeka teknikleri kullanılarak yapılması ile tedarik zinciri maliyetleri ve envanter yönetimi açısından işletmelere fayda sağlanabilmesi amaçlanmıştır.

Bu tez çalışmasında, literatürde yer alan stok yönetiminin etkin kontrolünü sağlamak için tahmine dayalı bir sistem tasarımı içerisinde yapay zeka ve makine öğrenimi tekniklerinden derin sinir ağları, rastgele ormanlar ve aşırı gradyan artırma algoritmaları kullanılarak ön sipariş tahmini Tazi yapay zeka programı ve Python hibrit kullanımı ile sunulmaktadır.

Bu tez çalışmasında elde edilen bulgulara göre önerilen model sonuçları AUC metriği baz alındığında 0.959 değeri ile Rastgele Orman algoritması en iyi performansa sahip algoritmadır. Sonrasında 0.948 AUC değeri ile XGBoost algoritması ikinci en iyi performans gösteren algoritma olmuştur. Derin Sinir Ağı Algoritması ise 0.902 AUC performans skoru ile üçüncü sırada yer almıştır. En iyi performans için AUC metriğinin 1'e yakın olması istenmektedir. Deneyler sonucunda AUC değeri 0.959 bulunmuştur ve

bu değer 1'e oldukça yakındır. Rastgele Orman algoritması kullanılan diğer makine öğrenimi algoritmalarına göre çok daha iyi bir performansa sahip olduğunu göstermiştir.

Tez çalışmasında önerilen modelin performans sonuçları literatürdeki model performans sonuçları ile karşılaştırıldığında Rastgele Orman Algoritması için literatürdeki en iyi AUC değeri elde edilmiştir.

Diğer çalışmalardan farklı olarak bu tez çalışmasında RUS yöntemi ile veri seti dengeli hale getirilmiştir. Bu tez çalışmasında, diğer alanlarda performansı kanıtlanmış olan yapay zeka teknikleri ön sipariş tahmininde kullanılmıştır. Veri ön işleme aşamasında Python ve makine öğrenimi algoritması uygulanması aşamasında TAZI kullanılmıştır. Bu tez çalışmasında literatürde ön sipariş tahmini yapılan çalışmalara göre, daha iyi sonuçlar elde edilmiştir.

Öngörü sistemleri yaklaşımları kapsamında yapay zeka ve makine öğreniminin, kıtlık olasılığı daha yüksek olan ürünleri kısa sürede gösteren ilginç tahmin mekanizmaları olduğu görülmektedir. Ayrıca bir işletmeye harekete geçmesi için bolca zaman tanıyan uygulanabilir tahmin mekanizmalarıdır. Bu nedenle, yapay zeka ve makine öğrenimi teknikleri kullanılarak karşılanamayan siparişler için ön sipariş tahmini otomotiv, sağlık, gıda vb. endüstrilere etkili değişiklikler yapma konusunda avantaj sağlayabilir.

KAYNAKLAR

Aalst, W. (2016). *Process Mining.* Data Science in Action. S:155.

Akbaş, E. (2018). *Derin Yapay Sinir Ağları ve Derin Öğrenme'ye Kısa Bir Giriş*. Orta Doğu Teknik Üniversitesi BMO Semineri – 24 Kasım 2018

Breiman, L. (2001). *Random Forests*. Machine Learning 45 (1) 5–32

Charles, Y. B., CorbettJan J., Fransoo C. ve Tan T. (2017). *Sustainable Supply Chains.* A Research-Based Textbook on Operations and Strategy

Chen, TQ. ve Guestrin, C, (2016). *XGBoost: A Scalable Tree Boosting System*. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). pp.785-794. DOI: http://dx.doi.org/10.1145/2939672.2939785

Cutler, A., Cutler, D.R. ve Stevens, J.R. (2012). *Random Forests*. In: Zhang, C. and Ma, Y.Q., Eds., Ensemble Machine Learning, Springer, New York, 157-175. http://dx.doi.org/10.1007/978-1-4419-9326-7_5

Çetin V. ve Yıldız O. (2021). *A comprehensive review on data preprocessing techniques in data analysis*. Pamukkale Universitesi Mühendislik Bilimleri Dergisi, 28(2), 299-312, 2022. doi: 10.5505/pajes.2021.62687

De Santis, R. B., De Aguiar, E. P. ve Goliatt, L. (2017). *Predicting material backorders in inventory management using machine learning.* IEEE Latin American Conference on Computational Intelligence (LA-CCI).1-6

Downey, T.J., Jr, Meyer, D.J., Price, R.K. ve Spitznagel, E.L. (1999). *Using the Receiver Operating Characteristic to Asses the Performance of Neural Classifiers*. IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339) Neural networks Neural Networks, 1999. IJCNN '99. International Joint Conference on. 5:3642-3646 vol.5 1999

Fernández A., García S., Galar M., Prati R., Krawczyk B. ve Herrera F. (2018). *Learning from Imbalanced Data Sets*. ISBN 978-3-319-98073-7 ISBN 978-3-319-98074-4 (eBook). https://doi.org/10.1007/978-3-319-98074-4

Hajek, P. ve Abedin, M. Z. (2020). *A Profit Function-Maximizing Inventory Backorder Prediction System using Big Data Analytics.* IEEE Access 8: 58982–58994.

Islam, S. ve Amin, S. H. (2020). *Prediction of probable backorder scenarios in the supply chain using Distributed Random Forest and Gradient Boosting Machine learning techniques.* Journal of Big Data, 7 (1): 1–22.

Kaggle. (2022). *Predict Product Backorders.* Erişim Tarihi: 26 Aralık 2022. Erişim adresi: https://www.kaggle.com/competitions/untadta/data

Lawal, S.O. ve Akintola, K.G. (2021). *A Product Backorder Predictive Model Using Recurrent Neural Network.* IRE Journals V. 4-8. ISSN: 2456-8880.

Malviya, L., Chittora, P., Chakrabarti, P., Vyas, R.S. ve Poddar, S. (2020). *Backorder prediction in the supply chain using machine learning.* Materials Today: Proceedings.

Ntakolia, C., Kokkotis, C., Karlsson, P. ve Moustakidis, S. (2021). An Explainable Machine Learning Model for Material Backorder Prediction in Inventory Management. Sensors 2021, 21, 7926. https://doi.org/10.3390/ s21237926

Öztemel, E. (2012). *Yapay Sinir Ağları*. Papatya Yayıncılık Eğitim. ISBN: 978-975-6797-39-6

Sanni, R. R. ve Guruprasada, H. S. (2021). *Analysis of performance metrics of heart failured patients using Python and machine learning algorithms*. Global Transitions Proceedings 2 (2021) 233–237. https://doi.org/10.1016/j.gltp.2021.08.028

Shajalal, Md., Hajek, P. ve Abedin, M. Z. (2021). *Product backorder prediction using deep neural network on imbalanced data.* Internatıonal Journal of Productıon Research ISSN: 0020-7543.

Soares, C. ve Torgo, L. (2021). *Discovery Science.* 24th International Conference, DS 2021 Halifax, NS, Canada, October 11–13, 2021 Proceedings. ISBN 978-3-030-88942-5 (eBook) https://doi.org/10.1007/978-3-030-88942-5

TAZI (2021), *Tazi Deploy Help*. Erişim Tarihi: 10 Mayıs 2023.

Towards Data Science. (2023). Erişim Tarihi: 30 Nisan 2023. Erişim adresi: https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79

Varghese, A. A., Krishnadas, J. ve Antony, A. (2023). *Robust Air Quality Prediction Based on Regression and XGBoost*. 2023 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)

Wang, M., Zhu, Z. ve Qian, G. (2023). *Modulation Signal Recognition of Underwater Acoustic Communication Based on Archimedes Optimization Algorithm and Random Forest.* Sensors 2023, 23, 2764. https://doi.org/10.3390/ s23052764

Yazıcı, A. C., Öğüş, E., Ankaralı, S., Canan, S., Ankaralı, H. ve Akkuş, Z. (2007). *Yapay Sinir Ağlarına Genel Bakış*. Artificial Neural Networks: Review. Türkiye Klinikleri J Med Sci 2007, 27.

EKLER

EK 1. Ön Sipariş Tahmininde Python ile Veri Önişleme

In [1]:
```
# Yardımcı Python analitik kütüphaneler ve paketleri yüklenmiştir.

import streamlit as st
import numpy as np
import pandas as pd
import seaborn as sns
import os
from joblib import dump, load
import base64
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import os
for dirname, _, filenames in os.walk('C:/User/simge'):
        for filename in filenames:
                print(os.path.join(dirname, filename))
```

In [2]:
```
# Csv dosyayı okuma
df = pd.read_csv('C:/Users/simge/Kaggle_Training_Dataset_v2.csv')
```

In [3]:
```
df.head()
```

Out [3]:

	sku	national_inv	...	stop_auto_buy	rev_stop	went_on_backorder
0	1026827	0.0	...	Yes	No	No
1	1043384	2.0	...	Yes	No	No
2	1043696	2.0	...	Yes	No	No
3	1043852	7.0	...	Yes	No	No
4	1044048	8.0	...	Yes	No	No

5 satır × 23 sütun

In [4]:
```
df.info()
```

Out[4]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1687861 entries, 0 to 1687860
Data columns (total 23 columns):
 #   Column           Non-Null Count    Dtype
---  ------           --------------    -----
 0   sku              1687861 non-null  object
 1   national_inv      1687860 non-null  float64
 2   lead_time        1586967 non-null  float64
```

EK 1. Ön Sipariş Tahmininde Python ile Veri Önişleme (Devamı)

```
 3   in_transit_qty     1687860 non-null  float64
 4   forecast_3_month   1687860 non-null  float64
 5   forecast_6_month   1687860 non-null  float64
 6   forecast_9_month   1687860 non-null  float64
 7   sales_1_month      1687860 non-null  float64
 8   sales_3_month      1687860 non-null  float64
 9   sales_6_month      1687860 non-null  float64
 10  sales_9_month      1687860 non-null  float64
 11  min_bank           1687860 non-null  float64
 12  potential_issue    1687860 non-null  object
 13  pieces_past_due    1687860 non-null  float64
 14  perf_6_month_avg   1687860 non-null  float64
 15  perf_12_month_avg  1687860 non-null  float64
 16  local_bo_qty       1687860 non-null  float64
 17  deck_risk          1687860 non-null  object
 18  oe_constraint      1687860 non-null  object
 19  ppap_risk          1687860 non-null  object
 20  stop_auto_buy      1687860 non-null  object
 21  rev_stop           1687860 non-null  object
 22  went_on_backorder  1687860 non-null  object
dtypes: float64(15), object(8)
memory usage: 296.2+ MB
```

In [5]:
```
df['went_on_backorder'].unique()
```

Out [5]:
```
array(['No', 'Yes', nan], dtype=object)
```

In [6]:
```
df.groupby('went_on_backorder').count()
```

Out [6]:

	sku	national_inv	lead_time	...	stop_auto_buy	rev_stop	went_on_backorder
No	1676567	1676567	1676567	1676567	1676567	1676567	1676567
Yes	11293	11293	11293	11293	11293	11293	11293

2 rows × 22 columns

In [7]:
```
df.went_on_backorder.value_counts(normalize=True)
```

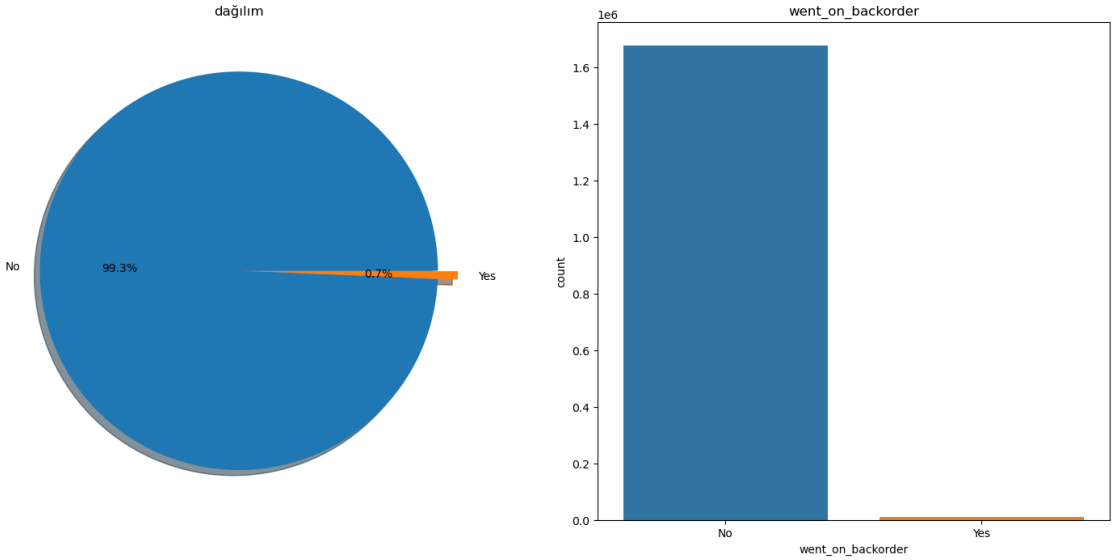
Out [7]:
```
No     0.993309
Yes    0.006691
```

EK 1. Ön Sipariş Tahmininde Python ile Veri Önişleme (Devamı)

In [8]:
```
# Veri setinin çıktı değerine göre dağılımı

f,ax=plt.subplots(1,2,figsize=(18,8))
df['went_on_backorder'].value_counts().plot.pie(explode=[0,0.1],autopct
='%1.1f%%',ax=ax[0],shadow=True)
ax[0].set_title('dağılım')
ax[0].set_ylabel('')
sns.countplot('went_on_backorder',data=df,ax=ax[1])
ax[1].set_title('went_on_backorder')
plt.show()
```

Out [8]:



In [9]:
```
df.count()
```

Out [9]:
```
sku                1687861
national_inv       1687860
lead_time          1586967
in_transit_qty     1687860
forecast_3_month   1687860
forecast_6_month   1687860
forecast_9_month   1687860
sales_1_month      1687860
sales_3_month      1687860
sales_6_month      1687860
sales_9_month      1687860
min_bank           1687860
potential_issue    1687860
pieces_past_due    1687860
perf_6_month_avg   1687860
```

EK 1. Ön Sipariş Tahmininde Python ile Veri Önişleme (Devamı)

```
perf_12_month_avg    1687860
local_bo_qty         1687860
deck_risk            1687860
oe_constraint        1687860
ppap_risk            1687860
stop_auto_buy        1687860
rev_stop             1687860
went_on_backorder    1687860
dtype: int64
```

EK 2. Ön Sipariş Tahmininde Python ile Veri Temizleme

In [10]:
```
# Son 5 satır
df.tail()
```

Out [10]:

	sku	national_inv	lead_time	in_transit_qty	...	stop_auto_buy	rev_stop	went_on_backorder
1687856	1373987	-1.0	NaN	0.0	...	Yes	No	No
1687857	1524346	-1.0	9.0	0.0	...	No	No	Yes
1687858	1439563	62.0	9.0	16.0	...	Yes	No	No
1687859	1502009	19.0	4.0	0.0	...	Yes	No	No
1687860	(1687860 rows)	NaN	NaN	NaN	...	NaN	NaN	NaN

5 rows × 23 columns

In [11]:
```
# Sayısal olmayan verilerin özeti
df.describe(include=['O'])
```

Out [11]:

	sku	potential_issue	…	rev_stop	went_on_backorder
count	1687861	1687860	…	1687860	1687860
unique	1687861	2	…	2	2
top	3275451	No	…	No	No
freq	1	1686953	…	1687129	1676567

EK 2. Ön Sipariş Tahmininde Python ile Veri Temizleme (Devamı)

In [12]:
```
# Sayısal verilerin özeti
df.describe()
```

Out [12]:

	national_inv	lead_time	…	perf_12_month_avg	local_bo_qty
count	1687860000000,00	1586967000000,00	…	1687860000000,00	1687860000000,00
mean	496111800,00	7872267,00	…	-6437947,00	626450,70
std	29615230000,00	7056024,00	…	25843330,00	33722240,00
min	-27256000000,00	0.000000e+00	…	-99000000,00	0.000000e+00
25%	4000000,00	4000000,00	…	660000,00	0.000000e+00
50%	15000000,00	8000000,00	…	810000,00	0.000000e+00
75%	80000000,00	9000000,00	…	950000,00	0.000000e+00
max	12334400000000,00	52000000,00	…	1000000,00	12530000000,00

In [13]:
```
# sku sütununun silinmesi
df = df.drop('sku', axis=1)
```

In [14]:
```
# Son satırın silinmesi
df = df[:-1]
```

In [15]:
```
df.tail()
```

Out [15]:

	national_inv	lead_time	in_transit_qty	...	rev_stop	went_on_backorder
1687855	0.0	2.0	...	No	No
1687856	-1.0	NaN	...	No	No
1687857	-1.0	9.0	...	No	Yes
1687858	62.0	9.0	...	No	No
1687859	19.0	4.0	...	No	No

EK 3. Ön Sipariş Tahmininde Python ile Eksik Verilerin Tamamlanması

```
In [16]:  #Eksik değerlerin kontrol edilmesi. Boş hücrelerin % oranı.
          total = df.isnull().sum().sort_values(ascending=False)
          percent_1 = df.isnull().sum()/df.isnull().count()*100
          percent_2 = (round(percent_1, 1)).sort_values(ascending=False)
          missing_data = pd.concat([total, percent_2], axis=1, keys=['Total', '%'],
          sort=False)
          missing_data.head()
```

Out [16]:

	Total	**%**
lead_time	100894	6.0
min_bank	1	0.0
potential_issue	1	0.0
rev_stop	1	0.0
stop_auto_buy	1	0.0

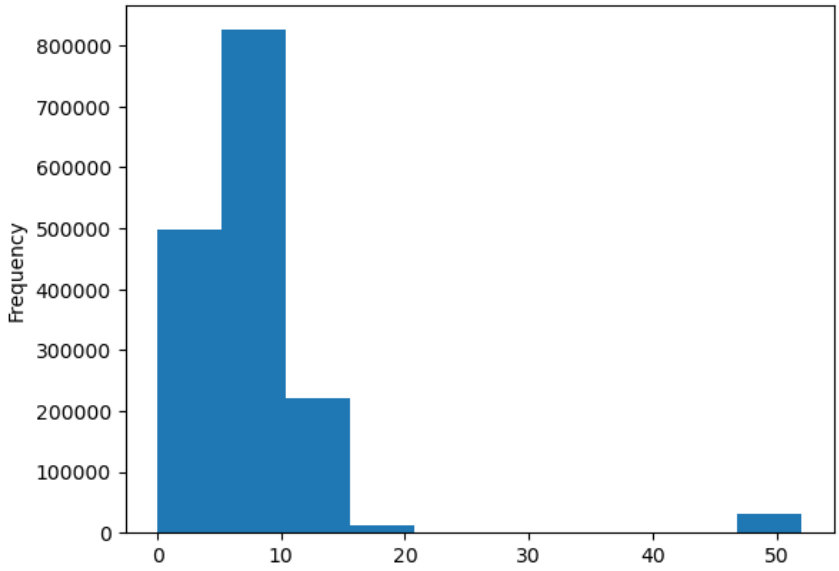
```
In [17]:  missing_data
```

Out [17]:

	Total	**%**
lead_time	100894	6.0
min_bank	1	0.0
potential_issue	1	0.0
rev_stop	1	0.0
stop_auto_buy	1	0.0
ppap_risk	1	0.0
oe_constraint	1	0.0
deck_risk	1	0.0
local_bo_qty	1	0.0
perf_12_month_avg	1	0.0
perf_6_month_avg	1	0.0
pieces_past_due	1	0.0
went_on_backorder	1	0.0
national_inv	1	0.0
sales_9_month	1	0.0
sales_6_month	1	0.0
sales_3_month	1	0.0
sales_1_month	1	0.0
forecast_9_month	1	0.0
forecast_6_month	1	0.0
forecast_3_month	1	0.0
in_transit_qty	1	0.0

EK 3. Ön Sipariş Tahmininde Python ile Eksik Verilerin Tamamlanması (Devamı)

```
In [18]:    # Lead_time histogramı
            df.lead_time.plot.hist()
Out [18]:
<AxesSubplot:ylabel='Frequency'>
```




```
In [19]:    # lead_time
            df.lead_time = df.lead_time.fillna(df.lead_time.median())

In [20]:    # Eksik değerlerin tekrar kontrol edilmesi
            total = df.isnull().sum().sort_values(ascending=False)
            percent_1 = df.isnull().sum()/df.isnull().count()*100
            percent_2 = (round(percent_1, 1)).sort_values(ascending=False)
            missing_data = pd.concat([total, percent_2], axis=1, keys=['Total', '%'],
            sort=False)
            missing_data.head()
            df.lead_time.plot.hist()
Out [20]:
```

	Total	**%**
national_inv	0	0.0
lead_time	0	0.0
rev_stop	0	0.0
stop_auto_buy	0	0.0
ppap_risk	0	0.0

EK 3. Ön Sipariş Tahmininde Python ile Eksik Verilerin Tamamlanması (Devamı)

In [21]:
```
# kategorik sütunların 1 ve 0 kodlanması
categorical_columns = ['potential_issue', 'deck_risk', 'oe_constraint', 'ppap_risk','stop_auto_buy', 'rev_stop', 'went_on_backorder']
for col in categorical_columns:
    df[col] = df[col].map({'No':0, 'Yes':1})
```

In [22]:
```
df.info()
```

Out [22]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1687860 entries, 0 to 1687859
Data columns (total 22 columns):

 #   Column            Non-Null Count    Dtype
---  ------            --------------    -----
 0   national_inv      1687860 non-null  float64
 1   lead_time         1586967 non-null  float64
 2   in_transit_qty    1687860 non-null  float64
 3   forecast_3_month  1687860 non-null  float64
 4   forecast_6_month  1687860 non-null  float64
 5   forecast_9_month  1687860 non-null  float64
 6   sales_1_month     1687860 non-null  float64
 7   sales_3_month     1687860 non-null  float64
 8   sales_6_month     1687860 non-null  float64
 9   sales_9_month     1687860 non-null  float64
 10  min_bank          1687860 non-null  float64
 11  potential_issue   1687860 non-null  int64
 12  pieces_past_due   1687860 non-null  float64
 13  perf_6_month_avg  1687860 non-null  float64
 14  perf_12_month_avg 1687860 non-null  float64
 15  local_bo_qty      1687860 non-null  float64
 16  deck_risk         1687860 non-null  int64
 17  oe_constraint     1687860 non-null  int64
 18  ppap_risk         1687860 non-null  int64
 19  stop_auto_buy     1687860 non-null  int64
 20  rev_stop          1687860 non-null  int64
 21  went_on_backorder 1687860 non-null  int64
dtypes: float64(15), int64(7)
memory usage: 283.3 MB
```

EK 4. Ön Sipariş Tahmininde Python ile Normalizasyon İşlemi

In [23]:
```
# Min-Maks Normalizasyonu
from sklearn import preprocessing
import pandas as pd
scaler = preprocessing.MinMaxScaler()
names = df.columns
d = scaler.fit_transform(df)
scaled_df = pd.DataFrame(d, columns=names)
scaled_df.head()
```

Out [23]:

	national_inv	lead_time	...	stop_auto_buy	rev_stop	went_on_backorder
0	0.002205	0.153846	...	1.0	0.0	0.0
1	0.002205	0.173077	...	1.0	0.0	0.0
2	0.002205	0.153846	...	1.0	0.0	0.0
3	0.002205	0.153846	...	1.0	0.0	0.0
4	0.002206	0.153846	...	1.0	0.0	0.0

In [24]:
```
scaled_df.tail()
```

Out [24]:

	national_inv	lead_time	in_transit_qty	...	rev_stop	went_on_backorder
1687855	0.002205	0.038462	0.000000	...	0.0	0.0
1687856	0.002205	0.153846	0.000000	...	0.0	0.0
1687857	0.002205	0.173077	0.000000	...	0.0	1.0
1687858	0.002210	0.173077	0.000033	...	0.0	0.0
1687859	0.002206	0.076923	0.000000	...	0.0	0.0

In [25]:
```
#Veri ön işleme temizleme işlemleri yapılmış veri setinin export edilmesi
scaled_df.to_csv("kaggletraindata.csv",index=False)
```

EK 5. DNN ile Yapılan Deneylerdeki Model Performans Sonuçları

	Gizli katman Sayısı	Gizli katmanlardaki nöron sayıları	Dropout	AUC	Precision	F1-Score	Recall
Deney 1	1	11	0,9	0,775	0,7157	0,7298	0,7444
Deney 2	1	130	0,9	0,882	0,8535	0,7974	0,7482
Deney 3	1	5	0,9	0,771	0,8124	0,537	0,4011
Deney 4	1	20	0,9	0,896	0,8434	0,8216	0,8009
Deney 5	1	30	0,9	0,887	0,843	0,8139	0,7868
Deney 6	1	30	0,5	0,856	0,7725	0,7525	0,7336
Deney 7	1	30	0,1	0,783	0,5258	0,6784	0,9559
Deney 8	1	70	0,9	0,899	0,8675	0,8119	0,763
Deney 9	1	80	0,9	0,903	0,8491	0,8317	0,815
Deney 10	1	90	0,9	0,9	0,8453	0,8255	0,8066
Deney 11	1	40	0,9	0,896	0,8564	0,8286	0,8026
Deney 12	1	50	0,9	0,887	0,8337	0,8075	0,7828
Deney 13	1	60	0,9	0,896	0,8252	0,8117	0,7986
Deney 14	1	100	0,9	0,904	0,846	0,8346	0,8235
Deney 15	2	10, 20	0,9	0,848	0,7428	0,7151	0,6894
Deney 16	2	30 , 20	0,9	0,896	0,8741	0,7762	0,6981
Deney 17	2	10, 30	0,9	0,89	0,8226	0,7675	0,7193
Deney 18	2	11 , 11	0,9	0,868	0,7682	0,7574	0,7469
Deney 19	2	11 , 15	0,9	0,764	0,7354	0,7406	0,7458
Deney 20	2	20, 40	0,9	0,839	0,7875	0,7526	0,7206
Deney 21	2	30 , 50	0,9	0,844	0,8168	0,717	0,639
Deney 22	2	60 , 80	0,9	0,845	0,8208	0,7242	0,648
Deney 23	2	100, 120	0,9	0,866	0,8432	0,7574	0,6875
Deney 24	2	20, 20	0,9	0,879	0,789	0,7867	0,7845
Deney 25	2	25, 25	0,9	0,887	0,8137	0,7968	0,7805
Deney 26	2	30, 30	0,9	0,896	0,8394	0,8138	0,7896
Deney 27	2	30, 30	0,2	0,182	0,3822	0,4558	0,5645
Deney 28	2	40, 40	0,9	0,834	0,7958	0,77	0,7458
Deney 29	2	50, 30	0,75	0,775	0,7074	0,734	0,7627
Deney 30	2	60, 60	0,8	0,874	0,8499	0,7644	0,6945
Deney 31	2	5 , 10	0,9	0,869	0,6574	0,6735	0,6904
Deney 32	3	5, 10, 20	0,9	0,896	0,8744	0,7276	0,6229
Deney 33	3	10, 20, 20	0,9	0,855	0,7986	0,6846	0,5991

EK 5. DNN Yapılan Deneylerdeki Model Performans Sonuçları (Devamı)

	Gizli katman Sayısı	Gizli katmanlardaki nöron sayıları	Dropout	AUC	Precision	F1-Score	Recall
Deney 34	3	10, 20, 30	0,9	0,882	0,8425	0,7978	0,7575
Deney 35	3	10, 30, 50	0,9	0,886	0,8416	0,7749	0,7181
Deney 36	3	11, 11, 11	0,9	0,824	0,691	0,7041	0,7176
Deney 37	3	5, 10, 20	0,9	0,895	0,8722	0,7268	0,6229
Deney 38	3	5, 10, 11	0,9	0,818	0,8223	0,6424	0,5271
Deney 39	3	2, 3, 5	0,9	0,488	0,4909	0,6146	0,8218
Deney 40	3	5, 8, 11	0,9	0,663	0,4945	0,6243	0,8464
Deney 41	3	40, 50, 60	0,9	0,884	0,8048	0,7687	0,7357
Deney 42	3	130, 100, 70	0,9	0,902	0,8568	0,8321	0,8088
Deney 43	3	50, 100, 130	0,9	0,881	0,8671	0,773	0,6974
Deney 44	3	10, 15, 20	0,9	0,872	0,8183	0,7793	0,7438
Deney 45	3	15, 15, 15	0,9	0,885	0,8635	0,7909	0,7296
Deney 46	3	10, 15, 20	0,9	0,872	0,8257	0,7756	0,7313
Deney 47	3	20, 20, 20	0,75	0,84	0,7944	0,742	0,6961
Deney 48	3	20, 30, 40	0,8	0,675	0,6324	0,6568	0,6831
Deney 49	3	30, 30, 30	0,5	0,617	0,7926	0,0896	0,0475
Deney 50	3	30, 40, 50	0,75	0,823	0,7533	0,7587	0,7643
Deney 51	3	30, 50, 50	0,8	0,843	0,7511	0,7815	0,8146
Deney 52	3	50, 60, 60	0,75	0,774	0,7551	0,6426	0,5592
Deney 53	3	60, 70, 80	0,8	0,798	0,7487	0,7106	0,6761
Deney 54	3	30, 50, 50	0,9	0,894	0,8275	0,8192	0,8111
Deney 55	3	30, 50, 50	0,75	0,851	0,7316	0,7835	0,8433
Deney 56	3	30, 50, 50	0,8	0,895	0,8479	0,8119	0,7788
Deney 57	3	30, 50, 50	0,85	0,891	0,8966	0,781	0,6917
Deney 58	4	5, 10, 15, 20	0,9	0,462	0,7738	0,5871	0,7738
Deney 59	4	10, 15, 20, 30	0,9	0,421	0,4412	0,4614	0,4836
Deney 60	4	11, 15, 15, 11	0,9	0,85	0,8668	0,5722	0,427
Deney 61	5	20, 20, 20, 20, 20	0,9	0,852	0,8388	0,7112	0,6173
Deney 62	5	5, 10, 15, 20, 25	0,9	0,784	0,8571	0,6063	0,469
Deney 63	5	10, 11, 12, 13, 14	0,9	0,784	0,6264	0,7503	0,9353
Deney 64	5	60, 80, 40, 50, 100	0,9	0,881	0,8533	0,763	0,69
Deney 65	5	20, 50, 80, 100, 130	0,9	0,875	0,8874	0,6644	0,531

EK 6. RF ile Yapılan Deneylerdeki Model Performans Sonuçları

	n_ estimators	Max Depth	Min Sample Split	Min Sample Leaf	max_ features	AUC	Precision	F1-Score	Recall
Deney 1	10	5	3	5	log2	0,932	0,8811	0,868	0,8552
Deney 2	10	5	3	3	log2	0,932	0,8802	0,8678	0,8558
Deney 3	10	5	3	3	log2	0,934	0,8764	0,8714	0,8665
Deney 4	100	5	2	1	log2	0,933	0,8796	0,8716	0,8637
Deney 5	50	10	3	5	log2	0,943	0,8779	0,884	0,8903
Deney 6	10	5	3	3	log2	0,932	0,8802	0,8678	0,8558
Deney 7	50	10	3	5	log2	0,946	0,882	0,885	0,888
Deney 8	100	5	2	1	log2	0,933	0,8796	0,8716	0,8637
Deney 9	50	6	3	5	log2	0,935	0,8792	0,8779	0,8767
Deney 10	10	6	3	3	log2	0,935	0,8747	0,8717	0,8688
Deney 11	20	6	3	5	log2	0,936	0,8813	0,8753	0,8693
Deney 12	20	6	2	1	log2	0,934	0,8841	0,8758	0,8676
Deney 13	75	6	3	5	log2	0,942	0,8751	0,8698	0,8647
Deney 14	80	8	3	5	log2	0,957	0,8939	0,8909	0,888
Deney 15	80	6	3	5	log2	0,947	0,8822	0,8735	0,8649
Deney 16	75	10	3	5	log2	0,959	0,8921	0,8934	0,8948
Deney 17	75	7	3	5	log2	0,937	0,8767	0,8767	0,8767
Deney 18	75	5	3	5	log2	0,933	0,8824	0,8738	0,8654
Deney 19	75	8	3	5	log2	0,943	0,8777	0,8792	0,8807
Deney 20	75	9	3	5	log2	0,944	0,8793	0,8825	0,8857
Deney 21	80	10	3	5	log2	0,944	0,8782	0,8817	0,8852

EK 7. XGBoost ile Yapılan Deneylerdeki Model Performans Sonuçları

	Max Depth	gamma	min child weight	AUC	Precision	F1-Score	Recall
Deney 1	5	0.7	1	0,947	0,8757	0,8793	0,8829
Deney 2	6	0.7	1	0,948	0,8757	0,8793	0,8829
Deney 3	7	0.7	1	0,947	0,8757	0,8793	0,8829
Deney 4	8	0.7	1	0,948	0,8757	0,8793	0,8829
Deney 5	9	0.7	1	0,948	0,8757	0,8793	0,8829
Deney 6	5	1	2	0,941	0,8768	0,8829	0,889
Deney 7	6	5	3	0,948	0,8751	0,8794	0,8837
Deney 8	7	7	5	0,948	0,8751	0,8794	0,8837
Deney 9	8	8	2	0,948	0,8751	0,8794	0,8837
Deney 10	8	1	5	0,948	0,8751	0,8794	0,8837

ÖZGEÇMİŞ

Adı Soyadı : Simge KARABAĞ
Doğum Yeri ve Tarihi :
Yabancı Dil : İngilizce

Eğitim Durumu
Lise : Bursa Emir Sultan Lisesi, 2013
Lisans : Kocaeli Üniversitesi – Endüstri Mühendisliği, 2017

Çalıştığı Kurum/Kurumlar : Akia Hess Otomotiv Ltd.Şti.,
Üçge Mağaza Ekipmanları A.Ş.,
Ford Otomotiv Sanayi A.Ş.

İletişim (e-posta) :

Yayınları :

Karabağ S., Öztürk N. 2022. Artificial Intelligence Techniques For Backorder Prediction, 3rd International Anatolian Congress, December 27-29, 2022, Kayseri, Türkiye.