

**T. C.**

**İSTANBUL GELİŞİM ÜNİVERSİTESİ**

**MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ**

**Bilgisayar Mühendisliği Bölümü**

**Makine Öğrenimi İle Konut-Fiyat Analizi**

**İbrahim SEZGİN**

BIL498 BİLGİSAYAR PROJESİ

**BİLGİSAYAR MÜHENDİSLİĞİ**

DANIŞMAN

Dr.Öğr.Üyesi ELHAM PASHAEİ

İSTANBUL, 2023

|  |  |  |
| --- | --- | --- |
| **Yazar Adı Soyadı** | **:** | İbrahim SEZGİN |
| **Tezin Dili** | **:** | Türkçe |
| **Tezin Adı** | **:** | Makine Öğrenimi İle Konut-Fiyat Analizi |
| **Fakülte** | **:** | İstanbul Gelişim Üniversitesi Mühendislik ve Mimarlık Fakültesi |
| **Bölüm** | **:** | Bilgisayar Mühendisliği |
| **Tezin Türü** | **:** | Bitirme Çalışması |
| **Tezin Tarihi** | **:** | 20.06.2023 |
| **Sayfa Sayısı** | **:** | 69 |
| **Tez Danışmanı** | **:** | Dr.Öğr.Üyesi Elham PASHAEİ |
| **Dağıtım Listesi** | **:** | Mühendislik ve Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü |

İbrahim SEZGİN

**BEYAN**

Bu tezin hazırlanmasında bilimsel ahlak kurallarına uyulduğu, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğu, kullanılan verilerde herhangi tahrifat yapılmadığını, tezin herhangi bir kısmının bu üniversite veya başka bir üniversitedeki başka bir tez olarak sunulmadığını beyan ederim.

İbrahim SEZGİN

20.06.2023

|  |  |  |  |
| --- | --- | --- | --- |
| **İSTANBUL GELİŞİM ÜNİVERSİTESİ**  **MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ** | | | |
| ...............................................’ın ................................................................................. ........................................................................... adlı tez çalışması, jürimiz tarafından Bilgisayar Mühendisliği Bölümü Bitirme tezi olarak kabul edilmiştir. | | | |
|  | Başkan | *İmza* |
| *Prof. Dr. Adı SOYADI* |
|  | Üye | *İmza* |
| *Prof. Dr. Adı SOYADI*  (Danışman) |
|  | Üye | *İmza* |
| *Prof. Dr. Adı SOYADI* |
|  | Üye | *İmza* |
| *Doç. Dr. Adı SOYADI* |
|  | Üye | *İmza* |
| *Dr. Adı SOYADI*  (Var ise İkinci Danışman) |

**Makine Öğrenimi ile Konut-Fiyat Analizi**

**İbrahim Sezgin**

**İSTANBUL GELİŞİM ÜNİVERSİTESİ**

**MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ**

**OCAK 2023**

# ÖZET

Veri Bilimi, verilerden öğrenebilen ve karar verme süreçlerinde kullanılabilen bir Yapay Zeka alt alanıdır.

Günümüzde ise sağlık, finans ve ulaşım dahil olmak üzere birçok sektörde ve alanda faaliyet göstermektedir.

Bu tezde emlak sitesinden toplanan verilerle konut-fiyat makine öğrenimi modeli oluşturulmuştur.

**Anahtar Kelimeler:** Veri Bilimi, Makine Öğrenimi, Konut-Fiyat Analizi

**Makine Öğrenimi ile Konut-Fiyat Analizi**

**İbrahim Sezgin**

**ISTANBUL GELİŞİM UNIVERSITY**

**FACULTY OF ENGINEERING & ARCHITECTURE**

**January 202****3**

# ABSTRACT

Data Science is a subfield of Artificial Intelligence that can learn from data and be used in decision making processes.

Today, it operates in many sectors and fields, including health, finance and transportation.

In this thesis, a housing-price machine learning model was created with the data collected from the real estate site.

**Keywords:** Artificial Intelligence, Machine Learning, Deep Learning

# İÇİNDEKİLER

[ÖZET 7](#_Toc137933223)

[ABSTRACT 8](#_Toc137933224)

[İÇİNDEKİLER 9](#_Toc137933225)

[KISALTMALAR 11](#_Toc137933226)

[ŞEKİLLERİN LİSTESİ 12](#_Toc137933227)

[GİRİŞ 14](#_Toc137933228)

[BİRİNCİ BÖLÜM 15](#_Toc137933229)

[LİTERATÜR TARAMASI 15](#_Toc137933230)

[1.1 Veri 15](#_Toc137933231)

[1.1.1 Veri Nedir? 15](#_Toc137933232)

[1.1.2 Büyük Veri Nedir? 16](#_Toc137933233)

[1.1.2.1 Büyük veri bileşenleri 16](#_Toc137933234)

[1.1.3 Veriden Nasıl Anlamlı Değer Üretilir? 17](#_Toc137933235)

[1.1.3.1 Veri analitiği türleri 17](#_Toc137933236)

[1.2 Yapay Zeka 18](#_Toc137933237)

[1.2.1 Yapay Zeka Bileşenleri 19](#_Toc137933238)

[1.2.1.1 Doğal dil işleme 19](#_Toc137933239)

[1.2.1.2 Görüntü işleme 20](#_Toc137933240)

[1.2.1.3 Uzman sistemler 21](#_Toc137933241)

[1.2.2.4 Planlama ve optimizasyon 21](#_Toc137933242)

[1.2.2.5 Robotik 22](#_Toc137933243)

[1.2.2.6 Konuşma tanıma 23](#_Toc137933244)

[1.2.2.7 Makine öğrenmesi 24](#_Toc137933245)

[1.3 Veri Analizinde Makine Öğrenmesi 25](#_Toc137933246)

[1.3.1 Makine öğrenmesi aşamaları 25](#_Toc137933247)

[1.3.1.1 Veri toplama 25](#_Toc137933248)

[1.3.1.2 Veri hazırlama 26](#_Toc137933249)

[1.3.1.3 Algoritmanın eğitilmesi 28](#_Toc137933250)

[1.3.1.4 Model hiperparametre ayarlaması 28](#_Toc137933251)

[1.3.1.5 Model doğrulama 29](#_Toc137933252)

[1.3.2 Makine öğrenmesi algoritmaları 36](#_Toc137933253)

[1.3.2.1 Denetimli öğrenme 36](#_Toc137933254)

[1.3.2.2 Denetimsiz öğrenme 54](#_Toc137933255)

[1.3.2.3 Pekiştirmeli öğrenme 56](#_Toc137933256)

[1.4 Veri Bilimi Ve Python 57](#_Toc137933257)

[İKİNCİ BÖLÜM 58](#_Toc137933258)

[UYGULAMA 58](#_Toc137933259)

[2.1 Veri Toplama 58](#_Toc137933260)

[2.2 Veri Ön İşleme 59](#_Toc137933261)

[2.3 Algoritmanın Eğitilmesi 71](#_Toc137933262)

[2.4 Model Hiperparametre Ayarlaması 78](#_Toc137933263)

[78](#_Toc137933264)

[2.5 Model Doğrulama 80](#_Toc137933265)

[SONUÇ 81](#_Toc137933266)

[KAYNAKÇA 82](#_Toc137933267)

# KISALTMALAR

Bu çalışmada kullanılmış kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

**Kısaltmalar Açıklamalar**

**AUC (Area Under The Curve -** Eğri Altında Kalan Alan**)**

**ARI (Adjusted Rand Index -** Düzeltilmiş Rand İndeksi**)**

**CART (Classifation And Regression Trees –** Sınıflandırma ve Regresyon Ağaçları**)**

**GBM (Gradient Boosting Machine -** Gradyan Artırma Makinesi**)**

**KNN**  **(K-Nearest Neighbor -** K-En Yakın Komşuluk**)**

**MAE (Mean Absolute Error -** Ortalama Mutlak Hata**)**

**MIS (Mutual Information Score -** Karşılıklı Bilgi Skoru)

**MSE (Mean Squared Error -** Hata Kareler Ortalaması**)**

**PCA (Principal Component Analysis -** Temel Bileşen Analizi**)**

**PCR (Principal Component Regression -** Temel Bileşen Regresyonu**)**

**PLS (Partial Least Squares -** Kısmi En Küçük Kareler**)**

**RMSE (Root Mean Squared Error –** Kök Ortalama Kare Hatası**)**

**ROC (Receiver Operating Characteristic**

**SVM (Support Vector Machines –** Destek Vektör Makinesi**)**

**SVR (Support Vector Regression –** Destek Vektör Regresyonu**)**

**XGBoost (eXtreme Gradient Boosting-**Ekstrem Gradyan Artırma**)**

# ŞEKİLLERİN LİSTESİ

[**Şekil 1**. Dataframe 58](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093686)

[**Şekil 2.** WHO\_IS\_THE\_SELLER Sütun Değeri Hayır Olan PRICE Değişkenindeki NaN Değerleri Doldurma 59](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093687)

[**Şekil 3**. WHO\_IS\_THE\_SELLER Sütun Değeri Evet Olan PRICE Değişkenindeki NaN Değerleri Doldurma 60](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093688)

[**Şekil 4.** PRICE Değişkeni Değerlerinin Düzeltilmesi 61](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093689)

[**Şekil 5.** PRICE Değişkeni Değerlerinin Düzeltilmeden Önce Değerleri 61](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093690)

[**Şekil 6.** IS\_THERE\_BALCONY sütunu değeri null olan satırların “Var” olarak güncellenmesi 62](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093691)

[**Şekil 7.** FLOOR\_LOCATION sütunu değeri null olan satırların koşula göre güncellenmesi 63](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093692)

[**Şekil 8.** NUMBER\_OF\_ROOMS sütunu değerinin koşula göre güncellenmesi 63](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093693)

[**Şekil 9.** Koşula göre NEW\_IS\_IT\_TWOPLY sütunu oluşturulması 63](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093694)

[**Şekil 10.** FLOOR\_LOCATION sütunu değerlerinin koşula göre güncellenmesi 64](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093695)

[**Şekil 11.** TOTAL\_NUMBER\_OF\_FLOORS sütunu değerinin koşuluna göre HOUSE\_TYPE değerinin güncellenmesi 65](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093696)

[**Şekil 12.** TOTAL\_NUMBER\_OF\_FLOORS sütunu değerinin koşuluna göre HOUSE\_TYPE değerinin güncellenmesi 65](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093697)

[***Şekil 13.*** TYPE\_OF\_HEATING sütunu değerinin koşula göre güncellenmesi 65](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093698)

[**Şekil 14.**TOTALNUMBER\_OF\_FLOORS sütunu değerinin koşula göre güncellenmesi 66](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093699)

[**Şekil 15.** IS\_IT\_FURNISHED sütunu değerinin koşula göre güncellenmesi 67](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093700)

[**Şekil 16.** WHO\_IS\_AT\_HOME sütunu değeri “Kiracılı” olan sütunların değerlerinin güncellenmesi 67](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093701)

[**Şekil 17.** TITLE\_DEED sütunu değeri “Bilinmiyor” olan sütunların değerlerinin güncellenmesi 67](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093702)

[**Şekil 18.** WHO\_IS\_THE\_SELLER sütunu değerlerinin koşula göre güncellenmesi 68](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093703)

[**Şekil 19.** NUMBER\_OF\_BATHROOMS sütununun değerlerinin belirli koşula göre güncellenmesi 68](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093704)

[**Şekil 20.** DUES sütunundaki değerlerin belirli koşullara göre güncellenmesi 69](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093705)

[**Şekil 21.** DUES sütununda değerlerin koşula göre eksik ve hatalı değerlerin güncellenmesi 70](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093706)

[**Şekil 22.** HOUSE\_TYPE sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi 71](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093707)

[**Şekil 23.** NUMBER\_OF\_ROOMS sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi 71](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093708)

[**Şekil 24.** AGE\_OF\_BUILDING sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi 72](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093709)

[**Şekil 25.** Binary sütunların sayısal değerlere dönüştürülmesi 73](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093710)

[**Şekil 26.** Dataframedeki sütunların koşula göre One-Hot Encoding yöntemiyle dönüştürülmesi 74](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093711)

[**Şekil 27.** Dataframe bulunan sayısal sütunları “RobustScaler” ile Normalizasyon işlemi 75](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093712)

[**Şekil 28.** Regresyon modellerini uygulama 76](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093713)

[**Şekil 29.** En iyi sonuç veren regresyon modellerinin hiperparametre optimizasyonu ile uygulanması 77](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093714)

[**Şekil 30.** Catboost regresyon modelinin hiperparametre optimizasyonu ile uygulanması 78](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093715)

[**Şekil 31.** Catboost regresyon modelinin hiperparametre ile uygulanmasının sonuçları 80](file:///C:\Users\ibrahimsezgin\Desktop\OKUL\bitirmetezi.docx#_Toc138093716)

# GİRİŞ

Konut piyasası, son yıllarda artan talep ve arz dengesizliği nedeniyle oldukça önemli hale gelmiştir. Konut fiyatları, pek çok faktöre bağlı olarak değişiklik göstermektedir. Bu nedenle, konut piyasasında tahminler yapmak için çok sayıda veri incelenmelidir. Bu makalede, konut fiyatlarını tahmin etmek için makine öğrenimi yöntemlerinin kullanımını inceleyeceğiz.

Herhangi bir daire sahibi birey, evinin fiyatını profesyonel bir iş bilen ile birlikte belirleyebilirdi. Ama şimdi bu düzeni değiştirecek büyük bir değişiklik olan Makine Öğrenimi var. Petrol nasıl geride bıraktığımız yüzyılın şekillenmesinde belirleyici olduysa veri de bulunduğumuz yüzyılı şekillendiriyor. Veriler, Makine Öğreniminin kalbidir. Temel olarak Makine Öğreniminde, bilgisayarların veri setleri üzerinden öğrenimini mümkün kılan algoritmaların tasarım, geliştirme ve uygulama süreçlerini konu edinen yapay zekanın bir alt dalıdır. Bir konut fiyat tahmininin modeli oluşturulması, konut fiyatlarının formüle edilmesine büyük ölçüde yardımcı olabilir ve gelecekteki gayrimenkul tahmininin doğruluğunu artırabilir. Bu projede konut fiyat tahmini, farklı Makine Öğrenimi kullanılarak yapılır. Bu amaçla, evlerin özelliklerini (yaş, bölge, büyüklük, oda sayısı vb.) içeren bir veri seti toplandı. Veri seti, cat boost regresyon modeli kullanılarak işlendi ve model eğitimi gerçekleştirildi. Ardından, modelin doğruluğunu artırmak için farklı makine öğrenimi teknikleri uygulandı ve sonuçlar karşılaştırıldı.

Yapılan analizler sonucunda, cat boost regresyon modeli ile ev fiyatları tahmin edilirken %65 doğruluk oranına ulaşıldı. Makine öğrenimi, konut fiyatlarının tahmin edilmesi gibi pek çok alanda kullanılabilir. Ancak, doğru sonuçlar elde etmek için doğru verilerin toplanması ve uygun algoritmaların kullanılması gerekmektedir.

# BİRİNCİ BÖLÜM

# LİTERATÜR TARAMASI

## Veri

### Veri Nedir?

Veri, ölçülebilir veya gözlemlenebilir gerçekler, sayılar, semboller veya karakterler gibi farklı şekillerde ifade edilebilen her türlü bilgidir. Veriler, insanlar tarafından manuel olarak toplanabileceği gibi, otomatik olarak da toplanabilir. Örneğin, bir anket sonucu, bir satış işlemi, bir sensörden alınan ölçüm, bir web sitesinden indirilen veri dosyası gibi farklı kaynaklardan toplanabilirler.

Veriler genellikle yapılandırılmış veya yapılandırılmamış olarak sınıflandırılır. Yapılandırılmış veriler, belirli bir düzen veya yapıya sahip olan verilerdir. Örneğin, bir Excel tablosundaki veriler, yapılandırılmış veri örneğidir. Yapılandırılmamış veriler ise belirli bir düzene sahip olmayan veya yapılandırılmamış olan verilerdir. Örneğin, bir e-posta mesajı, bir tweet veya bir video dosyasındaki ses gibi veriler, yapılandırılmamış veri örnekleridir.

Veriler, çeşitli sektörlerde ve işlevlerde kullanılır. Örneğin, finansal analiz, sağlık bilimleri, ürün geliştirme, müşteri ilişkileri yönetimi ve pazarlama gibi alanlarda veriler kullanılır. Veriler, işletmelerin karar verme sürecine yardımcı olan önemli bir araçtır ve doğru kullanıldığında değerli bir rekabet avantajı sağlayabilir.

Elektronik ortam gelişip yaygınlaştıkça, veri boyutu sürekli artmakta, yeni veri boyutu ölçüleri Petabayt, Eksabayt, Zetabayt artık günlük hayatımızda kullanılmaya başlanmıştır. Ancak bol miktarda bulunan şeyin değerinin az olması ile ilgili bir kural yok muydu? Veriyi diğer madenlerden, ürünlerden, piyasa çeşitlerinden farklı kılan da nedir? Cevap büyük-veri kavramında yatıyor.

### Büyük Veri Nedir?

Büyük veri, geleneksel veri işleme yöntemleriyle işlenemeyecek kadar büyük, hızlı ve karmaşık veri kümeleridir. Bu veriler, farklı kaynaklardan toplanabilirler ve genellikle yapısal veya yapısal olmayan veriler içerirler. Gökalp, M. O., Kayabay, K., Çoban, S., Yandık, Y. B., & Eren, P. E. (2018, November) Büyük veri kavramı Nesnelerin İnterneti, web kaynakları ve benzer bilgi teknoloji servisleri ile üretilen yapılandırılmış, yarı yapılandırılmış veya yapılandırılmamış yapıda, hızlı ve hacimli verileri tanımlamak için kullanılır ve işletmelerin veri analizi, iş kararları alması, müşteri ilişkileri yönetimi, pazarlama ve daha birçok alanda avantaj sağlaması için önemli bir araçtır.

Büyük veri altı ana bileşenden oluşmaktadır.

#### 1.1.2.1 Büyük veri bileşenleri

##### *1.1.2.1.1* Volume (Hacim):

Verinin miktarı için kullanılan terimdir. Bu veri miktarı işlenen veriye göre artıp azalmaktadır.

1.1.2.1.2 Variety (Çeşitlilik):

Veri türlerinin ve veri kaynaklarının çeşitliliğini ifade eder. Veriler her zaman yalnızca yapılandırılmış, yani sıra ve sütun düzeni içinde çoğunlukla hücrelerdeki sayılarla dosyalanmış şekilde bulunmaz. Veriler farklı formatlarda ve türde bulunur. Örneğin bir metin ve ses iki veri türüdür.

##### *1.1.2.1.3* Velocity (Hız):

Verilerin üretildiği, analiz edildiği ve yeniden işlendiği hızı ifade eder. Hız oranı ne kadar yüksekse o kadar değerlidir. Günümüzde gerçek zamanlı hızla çalışan birçok cihaz bulunmasının sebebi hızın önemli bir bileşen olmasıdır.

##### *1.1.2.1.4* Veracity (Değişkenlik):

Veri setlerinin sürekli bir akış, yeni verilerin eklenmesiyle güncellendiğini ifade eder.

##### *1.1.2.1.5* Validity(Geçerlilik):

Verilerin yüksek hızlarda güncellenip sürekli daha yüksek hacimlere ulaşması sonucu ortaya çıkan kalite eksikliğini ifade eder. Bu iki unsur belirsizliğin iki yüzüdür. Güvenilir olmayan verilerle herhangi bir karar ya alınamaz ya da alınan her karar güvenilir olmaz. Yüksek hızda sağlanan verilerin geçerliliği/doğruluğu veri güvenliğinin de bir parçasıdır.

##### *1.1.2.1.6* Value (Değer):

Veri anlamlı olduğu kadar değerlidir. Karar alma süreçlerine katkısı, karar alıcıların etkin sonuç almaları veriden elde edilen katma değere bağlıdır. Şirketlerin ve kurumların kendi veri platformlarını, bulut bilişim sistemlerine yatırım yapmalarının arkasında bu motivasyon yatmaktadır.

### 1.1.3 Veriden Nasıl Anlamlı Değer Üretilir?

Veriler, işlenerek değerli bilgiye ve anlamlı sonuçlara dönüştürülebilirler. Bu süreç, veri analizi olarak adlandırılır ve verilerin keşfedilmesi, temizlenmesi, dönüştürülmesi, özetlenmesi ve yorumlanması işlemlerini içerir. Bu işlemler, farklı veri analizi teknikleri kullanılarak gerçekleştirilir. İşlenmiş veriler sonuçta raporlar, grafikler veya tahmin modelleri gibi farklı çıktılara dönüştürülebilirler. Bu tekniklerin kullanımı, verilerin işlevsel değerinin artırılmasına ve işletmelerin karar verme sürecinde daha iyi bilgi sahibi olmalarına yardımcı olabilir. Verilerin değeri, doğru bir şekilde analiz edildiğinde, işletmelerin müşterilerinin davranışlarından, ürünlerinin performansından veya pazar koşullarından kaynaklanan sorunları belirlemelerine, iş süreçlerinde verimlilik artışına veya müşteri memnuniyeti artışına kadar birçok alanda kullanılabilir.

#### 1.1.3.1 Veri analitiği türleri

1.1.3.1.1 Tanımlayıcı analitik(Descriptive Analytics):

Geçmiş verileri, insanların kolayca okuyabileceği bir formda özetler. Veri toplama, veri madenciliği, kümeleme ve özet istatistikler (ortalama, standart sapma vb.) gibi tekniklerin tümü, geçmiş bir durumla ilgili analitik sağlamaya hizmet eder. Bu, bir şirketin geliri, kârı, satışları, sosyal medya ölçümleri vb. gibi raporların oluşturulmasına yardımcı olur.

1.1.3.1.2 Teşhis analitiği(Diagnostic Analytics):

Tanımlayıcı analitik gibi teşhis analitiği de geçmişe odaklanır. Bununla birlikte, bu tür analizler bir şeyin neden gerçekleştiğini göstermek için neden ve sonuç arar. Amaç, nedenleri belirlemek için geçmiş olayları karşılaştırmaktır. Aykırı değerlerin ortaya çıkarılması, regresyon analizi, keşfi veri analizi, veri madenciliği, boyut azaltma, vb. teknikler teşhis analizinde sık kullanılan tekniklerdendir. Kurumlar, belirli bir soruna derinlemesine bir bakış açısı sağladıkları için teşhis analitiğini kullanırlar.

##### 1.1.3.1.3 Tahmine dayalı analitik(Predictive Analytics):

Bu tür analitik, geleceğe ilişkin potansiyel sonuçlar hakkında tahminlerde bulunmak için geçmiş ve mevcut verilere bakar. Tahmine dayalı analitik, mevcut verileri analiz etmek ve gelecek hakkında tahminlerde bulunmak için veri madenciliği, yapay zekâ ve makine öğrenimini kullanır. Müşteri eğilimlerini, pazar eğilimlerini vb. tahmin etmeye çalışır.

1.1.3.1.4 Kural oluşturucu analitik(Prescriptive Analytics): İstenen tahmin edilen sonuçlara göre öngörülen eylemlerin belirlenmesine ve belirli bir sorunun çözümüne yönelik senaryo ve kural setleri, yol haritaları oluşturur. Hem tanımlayıcı hem de tahmine dayalı analitik ile çalışır. Çoğu zaman yapay zekâ ve makine öğrenimine dayanır. Eylem ve olay ilişkilerini sürekli olarak analiz etmek ve en uygun çözümü önermek için veriler üzerinden sürekli kendini eğitir. Geri bildirim mekanizmaları aracılığıyla yol haritasında güncellemeler yapar. Yapay zekâ, makine öğrenimi, derin öğrenme ve yapay sinir ağları algoritmaları belirli önerilerde bulunmaya yardımcı olarak kuralcı analitiği desteklemek için sıklıkla kullanılır.

Tanımlayıcı ve teşhise yönelik analitik türleri görece daha az karmaşıktır ve karar alıcılar açısından görece daha az katma değer sunmaktadır. Buna karşın tahmine dayalı analitik ve kural oluşturucu analitik geleceğe yönelik belirsizlikleri kaldırması bakımından daha az bilgiye ve işleme ihtiyaç duyduğundan görece daha karmaşıktır. Ancak bilginin değere dönüşmesinde en yüksek katma değeri bu sonuncular sağlamaktadır.

## Yapay Zeka

Son yetmiş yıldaki çok boyutlu ve hızlı gelişimi göz önünde bulundurulduğunda net, genel kabul görmüş bir yapay zekâ tanımlaması yapmanın oldukça zor olduğu bir gerçek. İlgili literatürde sıkça rastlanan bir tanımlama olan ERKUL, R. E. (2021) bir bilgisayarın veya bilgisayar kontrolündeki bir robotun çeşitli faaliyetleri zeki canlılara benzer şekilde yerine getirme kâbiliyeti olarak tanımlanmaktadır.

Matematikçi Alan Turing İngiliz ordusu için geliştirdiği Enigma isimli makine ile Nazilerin şifreli haberleşmelerini kırıp savaşın seyrini değiştirmiş biri olarak tanınmış olsa da günümüzde yapay zekâ çalışmalarının öncüsü kabul edilmektedir. 1950'de bir felsefe dergisinde kaleme aldığı, bir bilgisayarın düşünebilmesinin mümkün olup olmayacağına dair makalesinde öne sürdüğü mantıkî test günümüzde de geçerliliğini sürdürmektedir. Buna göre "sorgulayıcı" diye isimlendirilen bir insan hiç görmediği bir başka insan ve makine ile bilgisayara yazı yazarak sohbet eder. Başlangıçta sorgulayıcı bilgisayar ile yazışarak iletişim kurduğu insan ve makineyi görmemekte ve hangisinin insan hangisinin makine olduğunu bilmemektedir. Sorgulayıcı her ikisine de sorular sorarak hangisinin makine olduğunu anlamaya çalışır. Bu soru-cevap oturumları defalarca tekrarlanır. Nihayetinde sorgulayıcı tekrarlanan tüm oturumlarda tutarlı bir şekilde insanı makineden ayırt edemediğinde makine Turing testini geçmiş olur.

Turing testinde ilginç olan bilgisayarların bir şey bilmeye ihtiyaçlarının olmamasıdır. Buna karşın makineler büyük miktarda veriyi, yüksek hızlarla işleyip, insan ile iletişim kurabilen, "insanmış gibi" davranabilen, insanı taklit eden simülasyon araçlarıdır. Turing testini geçmek için bir makinenin bir dizi kapasiteye sahip olması yeterlidir: bilgiden öğrenip ilişki kurmak ve yeni durumları algılayıp muhtemel çözümler üretmek için makine öğrenmesine, metinleri çözümlemek için doğal dil işleme sistemine, görüntüleri çözümlemek için görüntü işleme sistemine, ses ve konuşulanları çözümlemek için konuşma tanıma sistemine, meslekî bilgi ve tecrübeyi otomatikleştirip geliştirmek için uzman sisteme, öğrenimini geliştirmek ve şartlara göre adapte edebilmek için planlama ve optimizasyon sistemine, nesne, çevre ve insanlarla fizikî temasa geçebilmek için robotiğe... Zaten tüm bunlar yapay zekâ çalışma alanının alt dallarını oluşturmaktadır.

### 1.2.1 Yapay Zeka Bileşenleri

#### Doğal dil işleme

Doğal Dil İşleme, insanların doğal dilini (örneğin İngilizce, Türkçe, vb.) bilgisayarlar tarafından anlaşılabilen bir formata dönüştürmek, işlemek ve kullanmak için kullanılan bir yapay zeka alt dalıdır. Yani, kısaca bilgisayar dili ve doğal dil arasında köprü oluşturmak şeklinde tanımlanabilir.

Doğal Dil işleme farkında olmasak bile günlük hayatımızın birçok alanına dahil olmuştur. Örneğin, internette ve sosyal medyada bir çok yere entegre edilmiş olan otomatik makine çevirisi, e-posta kutusunu dolduran gereksiz veya zararlı e-postaları filtreleyip "Gereksiz e-posta" kutusuna taşıyan metin sınıflandırma, dijital asistanlar tarafından kullanıcıların komutlarını anlamak ve cevap vermek, otomatik dil çevirisi, metin sınıflandırma, duygu analizi, konuşma tanıma, arama motorlarında "Bunu mu demek istediniz?" gibi sorular günlük hayatımızda kullandığımız doğal dil işleme uygulamalarından sadece birkaçıdır.

Doğal Dil İşleme'nin temel amaçlarından biri, bilgisayarların insanlarla etkileşim kurmasını ve insanların doğal dili kullanarak bilgisayarlarla etkileşim kurmasını kolaylaştırmaktır. Bu, insanların bilgisayarlarla daha doğal ve verimli bir şekilde etkileşim kurabilmesine olanak tanır.

#### Görüntü işleme

Görüntü işleme, dijital görüntülerin alınması, analizi, iyileştirilmesi ve çıkarılmasıyla ilgilenen bir bilgi işlem disiplinidir. Görüntü işleme, bir veya daha fazla dijital görüntüyü alır, bu görüntüleri analiz eder, değiştirir veya geliştirir ve ardından istenen sonuçları elde etmek için bu işlenmiş görüntüleri kullanır.

Görüntü işleme, birçok farklı uygulama alanında kullanılır. Bunlar arasında tıp, robotik, otomotiv, güvenlik, tarım, sinema ve daha birçok sektör yer alır. Görüntü işleme teknikleri, görüntülerdeki desenleri, nesneleri, renkleri, parlaklıkları ve diğer özellikleri algılamak, tanımak ve analiz etmek için kullanılır.

Görüntü işleme yöntemleri arasında filtreleme, görüntü düzeltme, kenar tespiti, nesne tanıma, görüntü sıkıştırma ve segmentasyon gibi teknikler bulunur. Bu teknikler, gürültüyü azaltmak, görüntü kalitesini artırmak, nesneleri ayırt etmek, görüntüleri sıkıştırmak veya görüntüdeki farklı bölümleri tanımak gibi farklı amaçlar için kullanılabilir.

Görüntü işleme algoritmaları genellikle dijital görüntüler üzerinde matematiksel ve istatistiksel işlemler uygular. Bilgisayarlı görü veya yapay zeka teknikleri de görüntü işlemede kullanılabilir. Görüntü işleme, bilgisayarlı görü ve yapay zeka ile birleştirildiğinde, otomatik görüntü analizi, nesne tanıma, yüz tanıma, otonom araçlar ve daha birçok karmaşık uygulama geliştirmek için kullanılabilir.

#### Uzman sistemler

Uzman sistemler, belirli bir uzmanlık alanında insan uzmanların bilgisini taklit eden ve çeşitli sorunları çözmek veya karar vermek için kullanılan bir yapay zeka alt dalıdır. Uzman sistemleri, bilgisayar programlarının uzmanlık gerektiren bir konuda uzman gibi davranabilmesini sağlayan bilgi tabanlarına dayanır.

Uzman sistemleri, genellikle belirli bir uzmanlık alanındaki bilgileri içeren bir bilgi tabanına sahip olurlar. Bu bilgi tabanı, uzmanların bilgi ve deneyimlerini kapsayan bir dizi kural ve ilişkilerden oluşur. Ayrıca, genellikle kullanıcıların sorularını anlayabilen ve uygun çözümleri sunabilen bir çıkarım motoru veya akıl yürütme mekanizması da içerir.

Uzman sistemleri, belirli bir uzmanlık alanında teşhis yapma, sorun giderme, planlama, tahmin yapma, karar verme gibi görevleri gerçekleştirmek için kullanılabilir. Örneğin, tıp alanında uzman sistemler, belirli semptomlara dayanarak hastalıkları teşhis etmek veya tedavi önerileri sunmak için kullanılabilir. Otomotiv endüstrisinde ise uzman sistemleri, arıza teşhisi yapmak veya bakım talimatları sağlamak gibi görevlerde kullanılabilir.

Uzman sistemleri, kullanıcılara hızlı ve doğru bir şekilde uzmanlık bilgisi sunarak karar alma süreçlerini iyileştirebilir. Ayrıca, uzman sistemleri, insan uzmanların bilgisini geniş bir kullanıcı kitlesine aktararak uzmanlığın daha geniş bir alanda erişilebilir hale gelmesini sağlar.

Uzman sistemleri, genellikle bilgi tabanı oluşturma, kuralların tanımlanması ve akıl yürütme mekanizmasının tasarlanması gibi aşamaları içeren karmaşık bir geliştirme süreci gerektirir. Ayrıca, uzman sistemleri, güncel ve doğru bilgiye erişimi sağlamak için periyodik olarak güncellenmeli ve sürdürülmelidir.

#### 1.2.2.4 Planlama ve optimizasyon

Planlama ve optimizasyon, yapay zekanın bir alt dalıdır ve çeşitli görevleri veya problemleri optimize etmek veya en iyi çözümü bulmak için matematiksel modeller ve algoritmalar kullanır. Bu alan, kaynakların etkin bir şekilde kullanılması, zaman ve maliyet tasarrufu, en iyi rotaların veya programların belirlenmesi gibi çeşitli uygulamalarda önemli bir rol oynar.

Planlama, belirli bir hedefe ulaşmak için bir dizi adımın veya eylemin sırasını veya düzenini belirlemekle ilgilidir. Örneğin, robotlar için bir planlama problemi, bir robota belirli bir görevi yerine getirmek için hangi adımları atması gerektiğini belirlemeyi içerebilir. Planlama algoritmaları, probleme özgü kısıtlar ve hedefler dikkate alınarak en etkili veya en optimal çözümü bulmak için kullanılır.

Optimizasyon ise belirli bir problemdeki en iyi çözümü veya sonucu bulma sürecidir. Bu süreç, bir fonksiyonun en küçük veya en büyük değerini bulmayı, bir sistemin en verimli şekilde çalışmasını sağlamayı veya belirli kısıtlar altında en iyi sonucu elde etmeyi hedefler. Optimizasyon algoritmaları, matematiksel modeller ve kısıtlarla tanımlanan bir probleme uygulanarak en iyi çözümü bulmak için kullanılır.

Planlama ve optimizasyon, birçok alanda uygulanabilir. Örneğin, lojistik sektöründe, optimal taşıma rotalarını veya teslimat programlarını belirlemek için kullanılabilir. Üretim endüstrisinde, kaynakların en verimli şekilde kullanılmasını sağlamak için kullanılır. Telekomünikasyon alanında, ağlar arası veri yönlendirmesinin en iyi yollarını belirlemek için optimizasyon teknikleri kullanılabilir.

Yapay zeka, planlama ve optimizasyon yöntemlerini kullanarak karmaşık problemlerin çözülmesine yardımcı olur. Bu alanlarda kullanılan algoritmalar genellikle matematiksel programlama, graf teorisi, genetik algoritmalar, simülasyon ve meta-sezgisel yaklaşımlar gibi tekniklere dayanır. Bu sayede, verimlilik, zaman tasarrufu, maliyet azaltma gibi hedeflere ulaşmak için optimize edilmiş çözümler sağlanır.

#### 1.2.2.5 Robotik

Robotik, yapay zeka ve mühendislik prensiplerinin birleştiği bir alan olup, robotların tasarımı, geliştirilmesi, kontrolü ve işletilmesi ile ilgilenir. Robotik, bilgisayar bilimi, makine mühendisliği, elektrik mühendisliği ve diğer ilgili disiplinlerin kesişim noktasında yer alır.

Robotlar, programlanabilir makinelerdir ve fiziksel dünyada çeşitli görevleri yerine getirmek üzere tasarlanırlar. Robotlar, genellikle sensörler, hareketli parçalar ve kontrol sistemlerinden oluşur. Yapay zeka teknikleri, robotların çevrelerini algılamasını, karar vermesini ve hareket etmesini sağlayarak onları daha akıllı ve özerk hale getirebilir.

Robotik, endüstriyel uygulamalardan evde kullanılan robotlara, sağlık sektöründen tarım alanına kadar geniş bir yelpazede kullanılır. Endüstriyel robotlar, fabrikalarda otomasyon süreçlerinde kullanılır ve tekrarlayan, tehlikeli veya karmaşık görevleri yerine getirebilirler. Hizmet robotları, evlerde veya kamu alanlarında insanlara yardımcı olabilir, temizlik yapabilir, güvenlik görevlerini üstlenebilir veya bakım hizmetleri sunabilir. Tıp alanında, cerrahi robotlar karmaşık operasyonlarda cerrahları destekler ve hassas manipülasyonları gerçekleştirebilir.

Yapay zeka, robotik alanında birçok uygulama sunar. Örneğin, görüntü işleme ve makine öğrenimi teknikleri, robotların çevrelerini görme, nesneleri tanıma veya yüzleri algılama gibi görevleri gerçekleştirmelerine yardımcı olur. Planlama ve optimizasyon algoritmaları, robotların hareketlerini planlamak, en uygun yolları veya işlem sırasını bulmak için kullanılır. Doğal dil işleme ve konuşma tanıma, insanlarla etkileşimde bulunan robotlar için önemli bir rol oynar.

Robotik, insanların günlük yaşamını kolaylaştıran, üretkenliği artıran ve tehlikeli görevleri üstlenen teknolojiler sunar. Yapay zeka ile birleştirildiğinde, daha akıllı, öğrenen ve adaptif robotlar geliştirilebilir. Ancak, robotik alanında hala birçok zorluk ve etik sorunlar bulunmaktadır ve bu alan sürekli olarak araştırma ve geliştirmeye açıktır.

#### 1.2.2.6 Konuşma tanıma

İnsan konuşmasını anlamak, işlemek ve metne dönüştürmekle ilgilenir. Konuşma tanıma, ses dalgaları veya ses kayıtları üzerinde çalışarak konuşmayı metne dönüştürme sürecini içerir.

Konuşma tanıma, bir dizi adımı içeren karmaşık bir süreçtir. İlk adım, ses kaynağından gelen ses dalgalarını almak ve sayısal bir temsil haline getirmektir. Bu işlem, analog sesin dijital bir formata dönüştürülmesini içerir. Daha sonra, konuşma tanıma sistemleri, dönüştürülen sesi analiz ederek konuşmanın farklı bileşenlerini ayırt etmeye çalışır. Bunlar arasında sesli harfler, kelimeler, cümleler ve dilbilgisi yapıları yer alır. Bu aşamada, işitme modelleri veya yapay sinir ağları gibi yöntemler kullanılarak sesin farklı bileşenlere ayrılması ve anlamlandırılması sağlanır. Son olarak, tanınan konuşma metne dönüştürülür ve kullanılabilir bir formatta sunulur. Bu, metnin yazılı olarak temsil edilmesini içerir. Konuşma tanıma sistemleri, çıktıyı düzeltmek veya daha iyi sonuçlar elde etmek için doğal dil işleme tekniklerini kullanabilir.

Konuşma tanıma, birçok uygulama alanında kullanılır. Örneğin, sesli komutlarla çalışan sanal asistanlar veya akıllı hoparlörler, konuşma tanıma teknolojisini kullanarak kullanıcıların komutlarını anlar ve yanıtlar. Otomotiv sektöründe, araç içi sesli kontrol sistemleri, sürücülerin ellerini direksiyondan ayırmadan telefon görüşmelerini yapmalarını sağlar. Müşteri hizmetleri alanında, otomatik yanıt sistemleri, kullanıcıların telefonla yapılan soruları anlayarak doğru yönlendirmeler yapabilir.

Konuşma tanıma, dil engellerini aşma, metin tabanlı verileri otomatik olarak dönüştürme, konuşma analizi ve anketlerde veri toplama gibi birçok avantaj sunar. Ancak, karmaşık aksanlar, arka plan gürültüsü veya konuşmanın doğal olmayan bir şekilde olması gibi zorluklarla da karşılaşabilir. Konuşma tanıma teknolojisi, dil çeşitliliği ve akustik koşullar gibi faktörleri dikkate alarak daha da geliştirilmeye devam etmektedir.

#### 1.2.2.7 Makine öğrenmesi

Makine öğrenmesi, bilgisayarların veri setleri üzerinden öğrenimini mümkün kılan algoritmaların tasarım, geliştirme ve uygulama süreçlerini konu edinen yapay zekanın bir alt dalıdır. Makine öğrenmesi veriden en iyi sonuçları elde etmek için güçlü bir teknik çerçeve sunmaktadır. Makine öğrenmesi, bilgisayar sistemlerinin verilerden öğrenme yapmasını ve deneyimlerden bilgi çıkarmasını sağlayan bir dizi teknik ve algoritmayı içerir. Makine öğrenmesi, bilgisayarların belirli bir programlama veya talimat kümesine bağlı kalmadan verileri analiz ederek desenleri tanımlamasına, kararlar almasına ve tahminler yapmasına olanak tanır.

Makine öğrenmesi, geleneksel programlamadan farklı bir yaklaşım benimser. Geleneksel programlamada, bir programcı belirli bir problem için adımları ve talimatları açıkça belirtir. Ancak makine öğrenmesinde, bir model veya algoritma, öğrenme süreci yoluyla verilere dayalı olarak davranışlarını ve çıktılarını adapte eder.

Makine öğrenmesi algoritmaları, genellikle büyük miktarda veriye dayanarak örüntüleri ve ilişkileri tanımlar. Bu veri, eğitim veri seti olarak bilinir ve genellikle özellikler veya etiketlerle birlikte gelir. Özellikler, veri noktalarının özniteliklerini temsil ederken, etiketler, bu verilerin hedef çıktılarını veya sınıflarını belirtir. Makine öğrenmesi algoritmaları, eğitim veri setini kullanarak modelin içindeki parametreleri ayarlar ve bu parametreler aracılığıyla yeni verilere uygulama yapar. Bu sayede, model, daha önce görmediği verilere dayalı olarak çıktılar üretebilir veya tahminler yapabilir. Makine öğrenmesi algoritmaları, büyük miktardaki veri setlerinden örüntüler çıkarırken aynı zamanda gelecekteki tahminlerde de kullanılabilir. Örneğin, finansal verileri analiz ederek hisse senedi fiyatlarının gelecekteki hareketlerini tahmin etmek veya müşteri davranışını anlamak ve kişiselleştirilmiş öneriler sunmak için kullanılabilir. Bununla birlikte, makine öğrenmesi yöntemleri bazen veriye olan bağımlılıkları ve yanlılıkları nedeniyle sınırlamalarla karşılaşabilir. Eğitim veri setinin kalitesi, veri toplama sürecindeki önyargılar veya veri eksikliği gibi faktörler, modelin performansını etkileyebilir. Ayrıca, bazı durumlarda makine öğrenmesi modelleri, insan açıklaması ve yargıya ihtiyaç duyduğu için etik ve hukuki sorunlar ortaya çıkabilir.

Makine öğrenmesi sürekli olarak gelişen bir alandır ve araştırmalar devam etmektedir. Derin öğrenme gibi yeni teknikler ve yaklaşımlar, daha karmaşık ve ölçeklenebilir modellerin geliştirilmesini sağlamıştır. Aynı zamanda, veri gizliliği, etik ve şeffaflık gibi konular üzerinde de çalışmalar yapılmaktadır, çünkü makine öğrenmesi uygulamalarının toplumda olumlu bir etki yaratması ve güvenilir olması önemlidir.

## 1.3 Veri Analizinde Makine Öğrenmesi

### 1.3.1 Makine öğrenmesi aşamaları

#### 1.3.1.1 Veri toplama

Makine öğrenmesi sürecinin ilk adımı, ilgili probleme yönelik veri toplamaktır. Bu veriler, örneklemlemeye dayalı olarak veya mevcut veri kaynaklarından elde edilebilir. Veriler, girdi özelliklerini ve hedef çıktıları içerebilir. Makine öğrenmesi algoritmalarına ne kadar çok veri sağlarsak o kadar yüksek performanslı bir model oluşturur ve bu model incelenen olayı daha iyi tahmin eder. Eğer veri hem hacim hem de kalite açısından yeterli değilse tahmin performansı iyi olsa bile yine de model hakkında şüpheler oluşacaktır. Makine öğrenmesi sistemleri tarafından yapılan tahminler, ancak eğitildikleri veriler kadar iyi olabilir.

##### 1.3.1.1.1 Veri toplama aşamasında karşılaşılan sorunlar

##### 1.3.1.1.1.1 Hatalı veriler

Toplanan veriler incelenen olay ya da problemle ile ilgisiz olabilir

Veri yapısı: Toplanan veriler her zaman yapılandırılmış veriler olmayabilir. Çoğu veri yapılandırılmıştır.

##### 1.3.1.1.1.2 Eksik veriler

değişkenlerin içerdiği değerler de eksiklikler olabilir. Örneğin 2000-2021 yıllarına ait günlük satış verileri serisini düşünelim. Bazı haftalar tadilat sebebi ile mağazalar kapalı tutulmuş ve direkt satış olmamıştır. Ancak aynı haftalarda online satış verileri mevcuttur. Bu durum veri setine ilgili sütunun bazı hücrelerinin boş olması şeklinde yansır.

##### 1.3.1.1.1.3 Veri Dengesizliği

Verilerdeki bazı sınıflar veya kategoriler orantısız şekilde yüksek veya düşük sayıda örneğe sahip olabilir. Sonuç olarak, modelde yetersiz temsil edilme riski taşırlar. Bir iş yerinde kadın ve erkek çalışanlar arasındaki gelir ve iş yükü ile ilgili 100 gözlemlik bir veri setimizin olduğunu düşünelim. 10 erkek ve 90 kadın çalışan bulunuyorsa, bu dengesiz dağılım modelin sonucunu etkileyecektir. Algoritma kadın çalışanları daha az tanıyacağından yanlı bir sonuç çıkarabilecektir.

##### 1.3.1.1.1.4 Yanlı veri

Verilerin, öznelerin ve etiketlerin kendilerinin nasıl seçildiğine bağlı olarak model, örneğin cinsiyet, politika, yaş veya bölge gibi önyargılar ihtiva edebilir. Bu tür durumların ilk bakışta tespit edilmesi ve kaldırılması zordur.

#### 1.3.1.2 Veri hazırlama

Toplanan ham veriler genellikle eksik gözlemlere sahip, farklı formatlarda kaydedilmiş veya istatistiki olarak belirli davranış veya eğilimlerden yoksun olabilir. Ayrıca birçok hata içermesi muhtemeldir. Bu nedenle veriler toplandıktan sonra makine öğrenmesi algoritmalarının model oluşturabilmesi için düzenlenip belli bir formatta sunulması gerekir. Veri hazırlama aşaması, sırasıyla keşfi veri analizi(EDA ya da Data Analysis), Veri Ön İşleme(Data Preprocessing) ve Veri Seti Bölümlenmesi(Data Split) adımlarından oluşur.

##### 1.3.1.2.1 Keşfi veri analizi

Keşfi veri analizi, veri setlerinin temel özelliklerini, genellikle görsel ve istatistiki yöntemlerle özetlemek için analiz etmeye yönelik bir yaklaşımdır. Öncelikle amacı incelenen soruna yönelik ortaya atılan hipotezleri test etmeden önce veri setindeki değişkenlerin dağılımları, aykırı değerleri ve anormallikleri tespit etmektir. EDA, verilerin model oluşturmaya uygun olup olmadığını, ne tür modeller için uygun olduğunu anlamamıza yardımcı olur. Bu adımda amaç kelimenin tam anlamıyla veriyi keşfetmek ve anlamaktır.

Keşfi veri analizinde; veri setinin tanımlayıcı istatistiği kullanılarak ortalama, standart hata, minimum ve maksimum değerler elde edilir, uygun grafikler kullanılarak normal dağılıma uygunluğuna bakılır, serpilme diyagramı, zaman serisi grafiği vb. grafiklerle değişkenler arasındaki ilişki görselleştirilir, aykırı değerler görselleştirilerek veya IQR hesabıyla tespit edilir, hedef değişken ve öznitelik değişkenleri belirlenir ve son olarak hedef değişkenin her bir öznitelik değişkeniyle ilişkisi tespit edilir.

##### 1.3.1.2.2 Veri ön işleme

Veri Ön İşleme adımı makine öğrenmesi sürecinde en fazla zaman alan kısımdır. Bir çalışmaya göre, veri bilimciler, zamanlarının %60 ını verileri temizlemek ve düzenlemek için harcamaktadır. Verilerin iyi modellenmesi bu adımın sağlıklı bir şekilde yürütülmesine bağlıdır.

Veri ön-işlemede, veri temizleme, veri yapılandırma, eksik veri tamamlama, aşırı örnekleme, veri entegrasyonu, veri normalleştirme, öznitelik seçimi, öznitelik çıkarımı adımları bulunur.

Veri temizleme; yanlış eklenen veya sınıflandırılan verilerin el yordamıyla ya da otomatik olarak kaldırılır.

Veri yapılandırma; yapılandırılmamış verileri yapılandırılmış verilere dönüştürülür. Örneğin metin, fotoğraf, ses yad da video kaydı gibi yapılandırılmamış veriler algoritmalar yardımıyla kantitatif verilere dönüştürüldükten sonra sütun ve satır düzeninde formatlanır.

Eksik veri tamamlama; sınıf ve kategoriler orantısız olan verileri dengelemek ve eksik verileri doldurmak için geliştirilmiş yöntemler kullanılarak veri setindeki bilgi kaybı en aza indirilir.

Aşırı örnekleme; veri kümesindeki yanlılık veya dengesizlik, daha fazla gözlem/örneklem oluşturarak düzeltilebilir.

Veri entegrasyonu; daha geniş bir veri seti elde etmek için birden fazla veri setini birleştirmek, tek bir veri kümesindeki eksikliğin üstesinden gelebilir.

Veri normalleştirme; veri setindeki öznitelik değişkenleri ve hedef değişken farklı ölçeklerde olduğunda çoğu makine öğrenmesi algoritması iyi performans göstermez. Bu nedenle, verileri normalleştirmek veya standartlaştırmak bu soruna yardımcı olabilir.

Öznitelik seçimi; öznitelik sayısı ne kadar fazlaysa veri seti de o kadar büyük olacaktır. Ancak yüksek boyutlu bir veri setinde alakasız veya önemsiz bazı öznitelikler olabilir. Bu tür özniteliklerin algoritmayı eğitmeye ve model oluşturmaya katkısının, kritik özelliklere kıyasla genellikle göz ardı edilebilecek derecede düşüktür. Sıfır katkıları da olabilir. Bu öznitelikler verimli tahmine dayalı modelleme sürecini engelleyen bir dizi soruna neden olur. Öznitelik seçim, bir veri setinden en önemli öznitelikleri seçme işlemidir. Öznitelik seçimi algoritmanın öğrenmesini hızlandırırken oluşturduğu modelin tahmin performansını çoğu zaman artırmaktadır.

Öznitelik çıkarımı; öznitelik seçimi veri setindeki alakasız veya gereksiz öznitelikleri filtreleyerek azaltmak ya da ayıklamak için kullanılırken öznitelik çıkarımı çok sayıda özniteliğin içerdiği değerleri dönüştürerek az sayıda özniteliğe indirgemek için kullanılır. Öznitelik çıkarımında, özniteliklerin değerleri matematiksel olarak dönüştürülmüş ve daha az sayıda yeni özniteliğe dönüştürülür ancak, bu az sayıdaki yeni öznitelik önceki çok sayıdaki özniteliğin içerdiği bilgiyi içermektedir. Öznitelik çıkarımı sırasında yapılan matematiksel dönüştürme sırasında az miktarda da olsa bilgi kaybı söz konusu olabilmektedir.

##### 1.3.1.2.3 Veri setinin bölümlenmesi

Veri Bölümleme ya da Eğitim-Test Seti Bölümlemesi adımında veri seti eğitim seti ve test seti olarak iki alt gruba ayrılacak şekilde bölünür. Eğitim verileri, algoritmanın veriler üzerinden eğitilmesi ve tahmin modeli oluşturması için kullanılır. Üretilen algoritmik modelin performasını ve yeni durumları öngörüp göremediği test veri seti kullanılarak sınanır. Eğitim-test seti bölümlenmesi genellikle rassal olarak yapılır. Örneğin, 10 gözlemlik bir veri setinde 1,4,6,7,8,10. gözlemler eğitim seti için rassal seçilirken arta kalan 2,3,5,9. gözlemler ise test seti olarak atanır. Ayrıca, genel geçer bir kural olmamakla birlikte uygulamada veri setindeki gözlemlerin %80 inin eğitim seti %20 sinin de test seti olarak atanması oldukça sık rastlanılan bir durumdur.

#### 1.3.1.3 Algoritmanın eğitilmesi

Probleme en uygun olan makine öğrenimi modeli seçildikten sonra önceden ayrılmış eğitim veri seti üzerinde eğitilir. Model, giriş verilerini (özellikler) ve hedef çıktılarını (etiketler veya hedef değerler) kullanarak ilişkileri öğrenir. Eğitim süreci, genellikle gerçekleştirilen bir optimizasyon algoritması kullanılarak modelin parametrelerini ayarlayarak gerçekleştirilir.

#### 1.3.1.4 Model hiperparametre ayarlaması

Makine öğrenmesi modelleri, öğrenilen parametrelere ya da veriyi bölümlendirme şekillerine ek olarak öğrenilemeyen, kullanıcı tarafından baştan belirlenmesi gereken parametreler bulundurmakta ve bunlara hiperparametre denilmektedir.

Genellikle hangi hiperparametrelerin model için daha uygun olduğunu bilemeyiz. Bunu model içinde deneyerek veya modele otomatik ayarlama yaparak, optimal değerlerin bulunmasını sağlamaya çalışırız. İşte bu optimal parametreleri bulma işlemine hiperparametre ayarlaması (Hyperparameter Tuning) diyoruz.

Hiperparametre ayarlaması önemlidir çünkü  veri bilimcilerin en uygun sonuçlara ulaşması için model performansında hassas ayarlama yapmalarını sağlar. Ayrıca model performansı üzerinde doğrudan etkilidirler.

#### 1.3.1.5 Model doğrulama

Modellerin ürettiği sonuçların doğru değerlendirilmesi çalışmasıdır. Böylelikle model, başarı sonuçlarını daha doğru değerlendirebilecektir. Eğitim seti üzerine kurulan modelin ürettiği sonuçların doğruluğu çeşitli yöntemlerle değerlendirilmelidir.

**Tablo 1.** Makine öğrenmesinde model doğrulama yöntemleri

|  |  |  |
| --- | --- | --- |
| Regresyon | Sınıflandırma | Kümeleme |
| MSE | Accuracy | ARI |
| RMSE | Precision | MIS |
| MAE | Recall | V-Measure |
| R2 | F1-Score | Elbow |
| Adjusted R2 | ROC-AUC Eğrisi |  |

1.3.1.5.1 Regresyon modellerinde model doğrulama yöntemleri

Regresyon modellerinde kullanılan başarı ölçümlendirme metrikleri, gerçek değerler ile tahmin edilen değerler arasındaki farkları değerlendirme imkanı sağlar.

##### 1.3.1.5.1.1 MSE

MSE, tahmin edilen değerlerle gerçek değerler arasındaki farkları ölçer ve bu farkların karelerinin ortalamasını alır.

MSE'nin değeri, daha küçük olması arzulanır. Yani, düşük MSE değeri, modelin tahminlerinin gerçek değerlere daha yakın olduğunu gösterir. MSE, hataların karelerini aldığı için, büyük hataların daha fazla ağırlığı vardır ve bu nedenle aykırı değerlere karşı daha hassas bir ölçüdür.

##### 1.3.1.5.1.2 RMSE

RMSE, tahmin edilen değerlerle gerçek değerler arasındaki farkları ölçer ve bu farkların karelerinin ortalamasını alındıktan sonra kare kökü alınır.

RMSE, MSE ile aynı özelliklere sahiptir ancak hataların orijinal birimde ifade edilmesini sağlar. MSE, hataların karelerini aldığı için, büyük hataların daha fazla ağırlığı vardır. RMSE, bu kare kök alınarak elde edildiğinde, hataların orijinal birimde ifade edilmesini sağlar ve yorumlaması daha kolay olur.

RMSE değeri, daha küçük olması arzulanır. Düşük RMSE değeri, modelin tahminlerinin gerçek değerlere daha yakın olduğunu ve daha iyi bir tahmin performansına sahip olduğunu gösterir.

##### 1.3.1.5.1.3 MAE

MAE, tahmin edilen değerlerle gerçek değerler arasındaki mutlak farkların ortalamasını temsil eder.

MAE, MSE'ye kıyasla daha dirençlidir, çünkü hataları kare almaz ve büyük hataların daha fazla ağırlığı olmaz. Bu nedenle, MAE, aykırı değerlerin olduğu durumlarda daha istikrarlı bir hata ölçüsü olarak tercih edilebilir.

MAE'nin değeri, daha küçük olması arzulanır. Düşük MAE değeri, modelin tahminlerinin gerçek değerlere daha yakın olduğunu ve daha iyi bir tahmin performansına sahip olduğunu gösterir.

##### 1.3.1.5.1.4 R2

Bağımsız değişkenin bağımlı değişkende açıkladığı varyasyon miktarını gösterir. R2 değeri genellikle yüzde cinsinden ifade edilir ve 0 ile 1 arasında değer alır. R² arttıkça, verilerdeki daha fazla varyasyon model tarafından açıklanır ve model tahminde daha iyi olur. Düşük bir R², modelin verilere tam olarak uymadığını ve bağımsız bir değişkenin bağımlı değişkendeki değişimi iyi açıklamadığını gösterir.

##### 1.3.1.5.1.5 Adjusted R2

Adjusted R2, regresyon analizinde kullanılan bir istatistiksel ölçüdür ve R2'nin bir düzeltilmiş versiyonudur. Adjusted R2, regresyon modelinin uyum düzeyini ve bağımsız değişkenlerin bağımlı değişkendeki değişkenliği açıklama yeteneğini değerlendirirken, modelin karmaşıklığını da dikkate alır.

Adjusted R2, bağımsız değişkenlerin sayısını ve modelin serbestlik derecesini göz önünde bulundurur. Bu sayede, modelin açıklama gücünü düzeltir ve fazla değişken içeren veya aşırı uyarlanmış modellerin değerlendirmesinde daha güvenilir bir ölçüdür.

1-

##### 1.3.1.5.2 Sınıflandırma modellerinde model doğrulama yöntemleri

##### 1.3.1.5.2.1 Accuracy

Doğru olarak sınıflandırılan gözlem sayısının toplam gözlem sayısına oranını temsil eder.

Accuracy, tahminlerin doğruluğunu yüzde olarak ifade eder. Yüksek accuracy değeri, modelin doğru sınıflandırma yapma yeteneğinin daha iyi olduğunu gösterir. Ancak, accuracy tek başına modelin performansını tam olarak yansıtmayabilir ve sınıf

dengesizliği, hata türleri ve diğer faktörler dikkate alınmalıdır.

##### 1.3.1.5.2.2 Precision

Kurt, A. (2021) Sınıflandırma algoritmaları tarafından doğru sınıflandırılan pozitif örnek sayısının yine sınıflandırma algoritmaları tarafından doğru ve yanlış sınıflandırılan pozitif örnek sayısına bölümüdür. Bir başka deyişle, modelin saldırı olarak sınıflandırdığı örneklerin kaçının gerçekte saldırı olduğunun oranını vermektedir

1.3.1.5.2.3 Recall

Kurt, A. (2021) Sınıflandırma algoritmaları tarafından doğru sınıflandırılan pozitif örnek sayısının, gerçekte saldırı olan, model tarafından saldırı olarak sınıflandırdığı pozitif örneklerin ve gerçekte saldırı olan ama modelin saldırı olarak tanımlamadığı negatif örneklerin toplam sayısına bölümüdür. Bir başka deyişle, gerçekte saldırı olan örneklerin ne kadarının doğru sınıflandırıldığının oranı verilmektedir.

##### 1.3.1.5.2.4 F1-Score

Modelin precision ve recall performansını birleştiren bir ölçüdür.

F1-Score, Precision ve Recall arasında bir denge sağlar. Yüksek bir F1-Score, hem Precision’ın hem de Recall'ın yüksek olduğunu gösterir, yani modelin hem pozitif tahminleri doğru yapma becerisi hem de gerçek pozitifleri kaçırmama becerisi iyi seviyededir.

F1-Score, özellikle dengesiz sınıf dağılımlarına sahip veri setlerinde kullanışlıdır. Eğer sınıflar arasında dengesizlik varsa sadece Accuracy metriği kullanmak yetersiz olabilir. Bu durumda F1-Score, modelin performansını daha doğru bir şekilde değerlendirmeye yardımcı olur.

##### 1.3.1.5.2.5 ROC-AUC eğrisi

ROC-AUC eğrisi, ROC eğrisinin altında kalan alanın hesaplanmasıyla elde edilir. Bu alanın değeri 0 ile 1 arasında değişir. ROC-AUC değeri ne kadar yüksekse, modelin sınıflandırma performansı o kadar iyidir. Eğer ROC-AUC değeri 0.5'e yakınsa, modelin performansı rastgele tahmin etmekle aynı seviyededir. ROC-AUC değeri 1'e yaklaştıkça, modelin performansı o kadar mükemmel hale gelir.

ROC-AUC eğrisi, bir modelin sınıflandırma performansını genel olarak değerlendirmek için kullanışlı bir ölçüdür, ancak tek başına bir modelin ne kadar iyi olduğunu belirlemek için yeterli değildir. Dolayısıyla, diğer metrikler ve probleme özgü faktörler de göz önünde bulundurulmalıdır.

##### 1.3.1.5.3 Kümeleme modellerinde model doğrulama yöntemleri

##### 1.3.1.5.3.1 ARI

Veri noktalarını benzerliklerine göre gruplara ayırmayı amaçlayan bir makine öğrenmesi yöntemidir. ARI, bu kümeleme sonuçlarının bir referans etiketleme (ground truth labeling) ile ne kadar uyumlu olduğunu ölçmek için kullanılır.

ARI, 0 ile 1 arasında değer alır. 1, kümeleme sonuçlarının referans etiketlemeyle tam uyumlu olduğunu, 0 ise rastgele bir kümeleme sonucu elde edildiğini ifade eder.

ARI hesaplanırken, her bir veri noktası için gerçek etiketleme (ground truth labeling) ve kümeleme sonucu etiketleme arasındaki uyumu karşılaştırır. Bu karşılaştırma, çiftlerin aynı kümeleme grubunda veya farklı kümeleme gruplarında olmasına dayanır.

ARI, gerçek etiketleme ve kümeleme sonucu etiketlemeleri arasındaki uyumu rastgele bir etiketleme ile karşılaştırarak, rastgelelik faktörünü düzeltir. Bu nedenle "Adjusted" (düzeltme) terimi kullanılır.

ARI, kümeleme modellerinin performansını değerlendirmek için kullanışlıdır. Yüksek bir ARI değeri, kümeleme sonuçlarının referans etiketlemeye daha yakın olduğunu gösterir. Düşük bir ARI değeri, kümeleme sonuçlarının rastgele bir etiketleme ile benzer düzeyde olduğunu veya uyumsuz olduğunu gösterir.

ARI, diğer kümeleme doğrulama yöntemleriyle birlikte kullanılarak kümeleme modellerinin performansını daha kapsamlı bir şekilde değerlendirmek için kullanılabilir. Ancak, ARI'nın da bazı sınırlamaları vardır ve etiket sayısı veya küme sayısı gibi faktörlerden etkilenebilir. Bu nedenle, ARI sonuçlarının yorumlanması ve diğer performans ölçüleriyle birlikte değerlendirilmesi önemlidir.

##### 1.3.1.5.3.2 MIS

Kümeleme sonuçlarının birbirine olan benzerliğini ölçmek için kullanılan bir metriktir. MIS, her bir küme içindeki örnekler arasındaki benzerliği hesaplar ve küme arasındaki benzerlikleri karşılaştırır.

MIS değeri, 0 ile 1 arasında bir ölçüm alır. 1, küme içindeki örneklerin birbirine çok benzer olduğunu ve küme arasında herhangi bir benzerlik olmadığını gösterirken, 0, küme içindeki örneklerin birbirine benzemediğini veya küme arasında yüksek benzerlik olduğunu gösterir.

MIS değeri, her bir küme içindeki örnekler arasındaki benzerliği hesaplamak için bir benzerlik ölçüsü kullanır. Örnek olarak, Euclidean mesafesi kullanılabilir. Daha sonra, küme içi benzerliklerin ortalaması ve küme arası benzerliklerin ortalaması hesaplanır ve bu değerler karşılaştırılarak MIS hesaplanır.

MIS, kümeleme modellerinin performansını değerlendirmek ve farklı kümeleme algoritmalarını karşılaştırmak için kullanılır. Yüksek bir MIS değeri, küme içindeki örneklerin birbirine benzemesini ve küme arasında düşük benzerlik olduğunu gösterir, bu da kümeleme modelinin başarılı olduğunu gösterir. Düşük bir MIS değeri, küme içindeki örneklerin birbirinden farklı olduğunu veya küme arasında yüksek benzerlik olduğunu gösterir, bu da kümeleme modelinin iyileştirilmesi gerektiğini gösterebilir.

MIS, diğer kümeleme doğrulama yöntemleriyle birlikte kullanılarak daha kapsamlı bir performans değerlendirmesi sağlar. Ancak, her bir küme içindeki örnek sayısı ve veri setinin özellikleri gibi faktörler MIS sonuçlarını etkileyebilir. Dolayısıyla, MIS sonuçlarının yorumlanması ve diğer metriklerle birlikte değerlendirilmesi önemlidir.

##### 1.3.1.5.3.3 V- Measure

Kümeleme sonuçlarının bir referans etiketleme ile ne kadar uyumlu olduğunu ölçen bir metriktir. V-Measure, hem kümeleme sonuçlarındaki homojenliği (h-score) hem de kümeleme sonuçlarının tamamlanmışlık skoru(c-score) birleştirir.

V-Measure, 0 ile 1 arasında bir değer alır. 1, kümeleme sonuçlarının referans etiketlemeyle mükemmel bir uyum içinde olduğunu, 0 ise hiçbir uyum olmadığını gösterir.

V-Measure, h-score ve tamamlanmışlık skoru c-score olmak üzere iki bileşenden oluşur.

##### V-Measure, hem homojenlik hem de tamamlanmışlık skorlarını dikkate alarak kümeleme sonuçlarının performansını değerlendirir. Hem homojenlik hem de tamamlanmışlık yüksek olduğunda, V-Measure değeri de yüksek olur. Bu durum, kümeleme sonuçlarının hem içerideki küme örneklerinin birbirine benzediğini hem de aynı sınıfa ait tüm örnekleri bir araya topladığını gösterir.

V-Measure, kümeleme modellerinin performansını değerlendirmek için kullanışlı bir ölçüdür. Ancak, homojenlik ve tamamlanmışlık skorları da tek başına yeterli değildir. Bu nedenle, diğer kümeleme doğrulama yöntemleriyle birlikte kullanılması ve veri setinin özellikleri ve probleme göre değerlendirilmesi önemlidir.

##### 1.3.1.5.3.4 Elbow

Kümeleme modellerinde optimal küme sayısını belirlemek için kullanılan bir doğrulama yöntemidir. Elbow yöntemi, küme sayısı ile kümeleme hatasının ilişkisini görselleştirerek dirsek benzeri bir eğri analizi yapar.

Elbow yöntemi, kümeleme algoritması (örneğin, K-Means) farklı küme sayıları için uygulanır. Her bir küme sayısı için, kümeleme hatası (inertia veya sum of squared distances) hesaplanır. Kümeleme hatası, her bir veri noktasının kümesinin merkezine olan uzaklıklarının karelerinin toplamını ifade eder. Küme sayısı ve kümeleme hatası arasındaki ilişkiyi görselleştirmek için bir grafik çizilir. Grafikte, küme sayısı x-ekseninde ve kümeleme hatası y-ekseninde yer alır. Grafik incelendiğinde, küme sayısı arttıkça kümeleme hatasının genellikle azaldığı görülür. Ancak, bir noktadan sonra kümeleme hatasındaki düşüş miktarı azalır. Elbow yöntemi, bu grafikteki "dirsek" noktasını bulmayı amaçlar. Dirsek noktası, küme sayısını belirlemek için bir gösterge olarak kullanılır. Dirsek noktası, küme sayısının artmasıyla düşüş miktarının belirgin bir şekilde yavaşladığı noktadır. Elbow yöntemi, küme sayısının seçiminde bir kriter olarak kullanılabilir. Dirsek noktası genellikle optimal küme sayısına işaret eder. Ancak, bu yöntem tek başına yeterli değildir ve diğer doğrulama yöntemleriyle birlikte değerlendirilmelidir. Veri setinin özellikleri, problemin gereksinimleri ve hedefler de dikkate alınmalıdır.

### 1.3.2 Makine öğrenmesi algoritmaları

#### 1.3.2.1 Denetimli öğrenme

##### 1.3.2.1.1 Sınıflandırma

ABBASOĞLU, B. (2020) Sınıflandırma işlemi en basit şekliyle bağımsız değişken değerleri belli iken, bağımlı değişkenin değerini tahmin etme işlemidir.

##### 1.3.2.1.1.1 KNN

K-En Yakın Komşu (KNN) algoritmasının sınıflandırma görevlerinde kullanılan bir uygulamasıdır. KNN, verilen bir giriş örneğini sınıflandırmak için KNN algoritmasını kullanır.

KNN algoritmasının çalışma prensibi, sınıflandırma yapılacak yeni bir giriş örneği olduğunda, KNN algoritması önce bu giriş örneğini sınıflandırmak için en yakın komşularını bulur. Bunun için, giriş örneğinin diğer eğitim örneklerine olan uzaklığı hesaplanır. Uzaklık genellikle Öklid veya Manhattan gibi bir uzaklık metriği kullanılarak hesaplanır. Uzaklıklar hesaplandıktan sonra, KNN algoritması K en yakın komşuyu (K parametresi tarafından belirlenen sayı) seçer. Bu, giriş örneğine en yakın olan K eğitim örneğini ifade eder. Seçilen K komşunun sınıf etiketleri gözlemlenir. Sınıflandırma durumunda, en yaygın olarak karşılaşılan sınıf etiketi, giriş örneğinin tahmin edilen sınıf etiketi olarak kullanılır. Örneğin, eğer çoğunlukla K komşu çiçekler "mavi" olarak etiketlenmişse, giriş örneği de "mavi" olarak sınıflandırılır.

KNN algoritmasının, uygulamasının kolay olması, fazla hiperparametresinin olmaması, önceden veri üzerinden eğitilip öğrenmesini gerçekleştirmek için model tahmin modeli oluşturmaya ihtiyaç duymaması gibi avantajları bulunmaktadır.

KNN algoritmasının, öznitelik sayısı 2-3'ü geçtiğinde yeni veri noktalarının setteki veri noktalarına mesafesinin ölçülmesi zor olması ve zaman alması, veri setindeki gözlem sayısı büyük olduğunda yeni veri noktalarının her birine mesafesinin ölçülmesi de ciddi zaman maaliyetine sebep olması gibi dezavantajları bulunmaktadır.

KNN, basit ve anlaşılır bir sınıflandırma yöntemidir. Ancak, K parametresinin iyi bir şekilde seçilmesi ve uzaklık ölçümünün uygun bir şekilde belirlenmesi önemlidir. Ayrıca, veri setinin ölçeklendirilmesi ve öznitelik seçimi gibi ön işleme adımları, KNN sınıflandırıcısının performansını etkileyebilir.

##### 1.3.2.1.1.2 Naive bayes

Naive Bayes algoritması, makine öğrenmesi alanında sınıflandırma problemlerini çözmek için kullanılan bir olasılık tabanlı sınıflandırma algoritmasıdır. Bu algoritma, Bayes Teoremi'ne dayanır ve "naive" (saf) olarak adlandırılır, çünkü özellikler arasındaki bağımsızlık varsayımını yapar.

Naive Bayes algoritmasının çalışma prensibi, özellikler arasındaki bağımsızlığı varsayar yani, her bir özellik sınıfın bilgisine dayanarak bağımsız olarak modellenebilir. Bu varsayım, algoritmanın basit ve hızlı çalışmasını sağlar, ancak bazı durumlarda gerçek dünyadaki ilişkileri göz ardı edebilir. Her sınıf için verilen özelliklerin olasılıklarını hesaplar. Bu, her bir özelliğin sınıfın olasılığını nasıl etkilediğini belirlemek için sınıf koşullu olasılıklarını kullanır. Bayes Teoremi'ni kullanarak özelliklerin olasılıklarını ve sınıfların olasılıklarını birleştirir. Bayes Teoremi, verilen özellikler altında sınıfın olasılığını hesaplamak için kullanılır. Son olarak, Naive Bayes algoritması, verilen özellikler için en yüksek olasılığa sahip olan sınıfı tahmin eder. Yani, en yüksek olasılığa sahip sınıfı seçer ve giriş örneğini bu sınıf ile sınıflandırır.

Naive Bayes algoritmasının avantajları;

Naive Bayes algoritması, hızlı ve basit bir sınıflandırma yöntemidir.

Küçük bir eğitim veri seti ile iyi sonuçlar verebilir.

Bağımsızlık varsayımı, büyük boyutlu özellik uzaylarında iyi çalışabilir.

Naive Bayes algoritması, eğitim veri setindeki sınıflar arasındaki dengesizliklere duyarlı olabilir. Nadir sınıfların doğru bir şekilde sınıflandırılmasını zorlaştırabilir.

Bu avantajlar ve dezavantajlar göz önüne alındığında, Naive Bayes algoritması, basitlik, hız ve küçük boyutlu veri setleriyle iyi performans gösterebilir. Ancak, özellikler arasındaki bağımlılıkları yakalamak veya karmaşık sınıflandırma problemleriyle başa çıkmak için daha karmaşık yöntemler tercih edilebilir. Naive Bayes, özellikle metin sınıflandırması, spam filtreleme gibi problemlerde ve başlangıç seviyesi sınıflandırma görevlerinde sıkça kullanılır.

##### 1.3.2.1.1.3 SVM

Karakoyun, M., & Hacibeyoğlu, M. (2014) SVM, Vapnik tarafından geliştirilmiş ve istatistiksel öğrenme teorisi alanında ortaya çıkmış bir öğrenme metodudur.

SVM'nin çalışma prensibi, sınıflar arasında bir ayrım yapabilen bir hiper düzlem (decision boundary) bulmaya çalışır. Bu hiper düzlem, örneklerin sınıflara mümkün olduğunca iyi bir şekilde ayrılmasını sağlamalıdır. SVM, bu optimizasyon problemine dönüştürülebilir bir denklem kullanarak en uygun hiper düzlemi bulur. Bu hiper düzlemi destekleyen ve sınıfları ayıran en yakın örnekler olan destek vektörleri belirler.Son olarak, yeni bir giriş örneği verildiğinde, SVM, bu örneğin hangi tarafında olduğunu belirlemek için hiper düzlemi kullanır.

SVM, lineer olmayan veri kümelerini çekirdek (kernel) fonksiyonları kullanarak sınıflandırabilir. Bu sayede veriler daha yüksek boyutlu bir uzaya taşınarak lineer olarak ayrılabilir hale getirebilmesi, yüksek boyutlu veri setlerinde iyi performans gösterebilmesi ve aşırı öğrenme eğilimli olmaması, destek vektörlerin dışındaki diğer eğitim örneklerini görmezden gelerek sadece destek vektörlerle çalışır, bu da algoritmanın hafızayı ve hesaplama maliyetini azaltması, sınıflandırma sırasında doğru tahminleri maksimize etmeye odaklanır ve sınıf sınırlarının genelleme kabiliyetini artırması gibi avantajları vardır.

Ancak SVM'nin dezavantajları arasında tüm eğitim örneklerini kullanarak bir karar sınıfı oluşturması nedeniyle, büyük veri setleriyle uygulandığında hesaplama süresi artabililir. SVM'nin performansı, doğru hiper parametre ayarına duyarlı olabilir. Örneğin, çekirdek fonksiyonu seçimi veya düzenlilik parametresi gibi parametrelerin doğru bir şekilde belirlenmesi önemlidir. SVM, etiket dengesizliği durumunda yanlı sonuçlara yol açabilir. Nadir sınıfların doğru bir şekilde sınıflandırılmasını zorlaştırabilir. SVM, her seferinde tam olarak doğru bir model bulma garantisi vermez. Veri setinin yapısı ve özelliklerine bağlı olarak farklı sınıflandırma sonuçları elde edilebilir. SVM, genellikle küçük ve orta ölçekli veri setleriyle iyi çalışırken, büyük veri setleri veya yüksek boyutlu veri kümeleriyle uygulandığında bazı zorluklarla karşılaşabilir. Ancak, doğru hiper parametre ayarlarıyla SVM, yüksek performanslı sınıflandırma sonuçları elde etmek için etkili bir seçenek olabilir.

##### 1.3.2.1.1.3 CART

CART, sınıflandırma problemlerini çözmek için kullanılan bir karar ağacı algoritmasıdır. CART algoritması, veri setindeki özelliklerin değerlerine dayanarak sınıflandırma veya regresyon kararlarını yapar.

CART algoritması, veri setini bölerek bir karar ağacı oluşturur. Başlangıçta, tüm veri seti bir kök düğümünde bulunur. Sonra, veri seti özelliklerin değerlerine göre bölünür ve alt düğümlere ayrılır. Bölünme, bir özellik ve onun değeri arasında en iyi bölünme kriterine göre yapılır. Bu kriter, veri setinin homojenlik veya saflık derecesini maksimize etmeye veya hatayı minimize etmeye çalışır. CART algoritması, sınıflandırma problemleri için genellikle Gini impurity veya entropy kullanır. Gini impurity, veri setindeki örneklerin sınıflara göre homojenlik derecesini ölçerken, entropy ise bilgi kazancını hesaplayarak veri setinin düzenliğini ölçer. Ağaç oluşturma işlemi, bölünme kriterine göre tekrarlanarak alt düğümler oluşturulur. Bu işlem, belirli bir durma kriteri sağlanana veya belirli bir ağaç derinliğine ulaşılıncaya kadar devam eder. Son olarak, yeni bir giriş örneği verildiğinde, bu örnek karar ağacının yapısını takip ederek sınıflandırma sonucunu verir. Sınıflandırma problemlerinde sınıf etiketi belirlenir.

CART algoritmasının; ağaç yapısının sonuçları açık ve anlaşılır bir şekilde sunduğu için karar verme sürecini görselleştirme ve yorumlaması kolay olması, özelliklerin önem sıralamasını sağlaması, hangi özelliklerin sınıflandırma veya regresyon için daha etkili olduğunu belirleyebildiği gibi avantajları vardır.

CART algoritmasının; ağaç yapısı karmaşık hale gelebilir ve aşırı öğrenmeye (overfitting) eğilimli olabilir. Bu durum, ağacın veriye fazla uyarlanmasına ve genelleme kabiliyetinin azalmasına neden olabilir. Bu nedenle, ağaç oluştururken durma kriterlerinin doğru bir şekilde ayarlanması önemlidir. Ağaç yapısı, özelliklerin ve değerlerin kesin bölünmelerine dayandığı için, veri setindeki küçük dalgalanmalara ve gürültülere duyarlı olabilir. Bu da ağacın dalgalanmalara ve veri setindeki değişikliklere hassas olmasına yol açabilir. CART algoritması, veri setindeki sınıflar veya değerler arasındaki dengesizlikleri ele almakta zayıf olabilir. Nadir sınıflar veya nadir değerler, ağaç yapısının oluşturulması sırasında düşük bir öncelikle ele alınabilir ve doğru bir şekilde modellenmeyebilir.

CART algoritması, sınıflandırma problemlerinde genellikle iyi performans gösterir. Özellikle veri setleri arasında doğrusal olmayan ilişkiler veya karmaşık karar sınırları olduğunda etkili olabilir. Ayrıca, ağaç yapısının açık ve anlaşılır olması, karar vericilerin ve uzman olmayan kullanıcıların sonuçları kolayca yorumlamasına olanak sağlar.

##### 1.3.2.1.1.4 Random forest

Random Forest, makine öğrenmesinde sınıflandırma ve regresyon problemlerini çözmek için kullanılan bir ensemble öğrenme algoritmasıdır. Random Forest, birden çok karar ağacını bir araya getirerek daha güçlü bir sınıflandırma veya regresyon modeli oluşturur.

Random Forest algoritmasının çalışma prensibi, veri setinden rastgele örnekleme yaparak birden fazla alt veri kümesi oluşturur. Bu alt veri kümeleri, orijinal veri setinin boyutuna yakın olacak şekilde rastgele örnekler içerir. Her alt veri kümesi için ayrı bir karar ağacı oluşturulur. Karar ağacı oluşturma işlemi, CART algoritmasının çalışma prensibine benzer. Ancak, her ağaç için özellik seçiminde rastgelelik uygulanır. Bu, her ağacın farklı özellikler üzerinde eğitilmesini sağlar. Oluşturulan karar ağaçları birleştirilerek bir ensemble modeli oluşturulur. Sınıflandırma problemleri için, ensemble modeli, sınıflandırma sonuçlarının oylama veya çoğunluk kararı ile belirlendiği bir yöntem kullanır. Regresyon problemleri için ise ensemble modeli, ağaçların tahmin değerlerinin ortalaması veya ağırlıklı ortalaması kullanılarak bir tahmin yapar.

Rastgele örnekleme ve rastgele özellik seçimi sayesinde, Random Forest aşırı öğrenmeye karşı dirençlidir. Bu, veri setindeki gürültüye ve gereksiz özelliklere karşı daha iyi bir genelleme kabiliyeti sunması, büyük veri setleri üzerinde etkili bir şekilde çalışabilir ve hesaplama süresini azaltmak için paralel işlem yeteneklerinden faydalanılabilmesi, veri setindeki özelliklerin önem sıralamasını da sağlayabilmesi ve hangi özelliklerin sınıflandırma veya regresyon için daha etkili olduğunu belirleyebilmesi gibi avantajları vardır.

Random Forest modelleri genellikle daha karmaşık olabilir ve ağaç sayısı, derinlik gibi hiperparametrelerin doğru bir şekilde ayarlanması gerekebilir. Bu da modelin eğitim süresini artırabilir. Modelin sonucunu yorumlamak bazen zor olabilir, çünkü birden fazla karar ağacının birleşimiyle oluşan bir modelden dolayı, hangi özelliklerin ve bölünmelerin sonucu nasıl etkilediğini belirlemek karmaşıklaşabilir. Veri setinde dengesizlik varsa, yani sınıflar arasında eşitsiz bir dağılım söz konusu ise, Random Forest modeli nadir sınıflara daha az dikkat edebilir ve doğru sınıflandırmaları zorlaştırabilir. Bu durumda, dengesizlikle başa çıkmak için özel ön işleme yöntemleri veya hiperparametre ayarları gerekebilir.

Random Forest algoritması, genel olarak yüksek doğruluk, iyi genelleme kabiliyeti ve dirençli bir model olma özellikleri nedeniyle yaygın olarak kullanılan bir sınıflandırma ve regresyon algoritmasıdır. Ancak, veri setinin özelliklerine ve problem alanına bağlı olarak, diğer algoritmalara kıyasla bazı durumlarda daha yüksek hesaplama maliyetine sahip olabilir.

##### 1.3.2.1.1.5 GBM

GBM, bir sınıflandırma ve regresyon algoritmasıdır. Gradient boosting yöntemini kullanarak, genellikle karar ağaçları bir araya getirerek güçlü bir sınıflandırma modeli oluşturur.

GBM, başlangıçta basit bir modelle başlar. Bu başlangıç modeli, veri setinin ortalama değeri veya bazı varsayımlara dayalı bir değer olabilir.

GBM algoritmasının çalışma prensibi, başlangıç modelinin tahminlerinden kaynaklanan hatalar hesaplanır. Bu hatalar, orijinal hedef değerlerle başlangıç modelinin tahminleri arasındaki farklardır. Hataları azaltmak için karar ağaçları oluşturulur. Her bir ağaç, hatalar üzerine optimize edilir. Yeni bir ağaç, önceki ağaçların hatalarını azaltmaya odaklanır. Oluşturulan zayıf ağaçlar bir araya getirilerek güçlü bir model oluşturulur. Bu birleştirme genellikle ağırlıklı oylama veya tahminlerin ortalamasıyla yapılır.

GBM algoritması, heterojen veri setleri üzerinde iyi performans gösterir ve birçok farklı türden özellikleri ve etkileşimleri ele alabilir. GBM, ağaçların birbirini takip eden şekilde oluşturulması sayesinde, her bir ağacın önceki ağaçların hatalarını düzeltmeye odaklanmasıyla önceki ağaçlara göre daha etkili hale gelebilir.

GBM, parametre ayarlarına hassas olabilir. İyi bir performans elde etmek için çeşitli hiperparametrelerin (ağaç sayısı, derinlik, öğrenme oranı vb.) doğru şekilde ayarlanması gerekebilir. Bu, deneyim ve hiperparametre optimizasyonu gerektirebilir.

GBM, diğer ensemble yöntemlerine kıyasla daha uzun sürebilir. Çünkü ağaçlar birbirini takip eden şekilde oluşturulur ve her bir ağaç, önceki ağaçların hatalarını düzeltmek için optimize edilir. Bu, büyük veri setleri veya karmaşık problemler için hesaplama süresini uzatabilir. Overfitting'e (aşırı uydurma) eğilimli olabilir. GBM, veri setine fazla uyum sağlayarak eğitim verilerine aşırı öğrenme yapabilir. Bu durumda, düzgün bir genelleme yapma yeteneği düşebilir. Bu nedenle, hiperparametre ayarları ve durma kriterlerinin doğru bir şekilde ayarlanması önemlidir. GBM, veri setindeki gürültüye ve anormalliklere hassas olabilir. Anormal değerler veya aykırı noktalar, ağaçların hatalarını etkileyebilir ve modelin performansını düşürebilir. Bu nedenle, veri ön işleme yöntemlerinin kullanılması önemlidir.

GBM algoritması, genellikle büyük ve karmaşık veri setleri üzerinde iyi performans gösterir. Ayrıca, değişkenler arasındaki etkileşimleri ve karmaşıklıkları ele alabilme yeteneği sayesinde diğer sınıflandırma ve regresyon algoritmalarına göre daha iyi sonuçlar verebilir. Ancak, doğru hiperparametre ayarlarına ve veri ön işleme adımlarına dikkat edilmesi önemlidir.

##### 1.3.2.1.1.6 XGBoost

XGBoost, Gradient Boosting yöntemini temel alan ve özellikle sınıflandırma ve regresyon problemlerinde yüksek performans sağlayan bir makine öğrenmesi algoritmasıdır.

XGBoost algoritmasının çalışma prensibi, başlangıçta bir sabit değerle başlar. Bu değer, veri setinin ortalama değeri veya bazı varsayımlara dayalı bir tahmin olabilir. Gradient Boosting yöntemini kullanarak genellikle karar ağaçları bir araya getirir. Her aşamada, önceki modelin tahminlerinden kaynaklanan hatalar hesaplanır ve sonraki model bu hataları azaltmaya odaklanır. Özel kayıp fonksiyonlarını optimize ederek farklı problemlere uyum sağlayabilir. Örneğin, sınıflandırma problemleri için logaritmik kayıp (log loss) fonksiyonu kullanılabilir. Bu, sınıflandırma hatasını minimize etmeye çalışır. Aşırı öğrenmeye karşı dirençli olmak için çeşitli normalleştirme teknikleri kullanır. Bunlar arasında ağaç derinliğini sınırlama, ağaç yapısını basitleştirme ve özellik ölçeklendirme yer alır.

XGBoost, hızlı hesaplama yetenekleri ve paralel işleme desteği sayesinde büyük veri setleri üzerinde etkili bir şekilde çalışabilir. Aynı zamanda GBM'e kıyasla daha hızlı bir eğitim süresine sahiptir. Aşırı öğrenmeye karşı dirençli olmak için çeşitli normalleştirme tekniklerini kullanır. Bu nedenle, veri setindeki gürültüye veya gereksiz özelliklere daha iyi tolerans gösterebilir ve daha iyi genelleme yapabilir.

Çeşitli kayıp fonksiyonlarını optimize edebilir, böylece farklı problemlere uyum sağlayabilir. Özel kayıp fonksiyonları, belirli bir probleme odaklanarak daha iyi sonuçlar elde etmeyi sağlar. Her bir özelliğin sınıflandırmadaki veya regresyondaki önemini değerlendirmek için bir özellik önemi skoru sağlar. Bu, hangi özelliklerin daha etkili olduğunu belirlemek ve veri setini daha iyi anlamak için faydalı bir bilgidir.

XGBoost'un birçok hiperparametresi vardır ve doğru şekilde ayarlanmaları gerekmektedir. Bu, deneyim ve hiperparametre optimizasyonu gerektirebilir. Yanlış ayarlanmış parametreler, modelin performansını etkileyebilir. XGBoost, veri setindeki eksik değerleri veya aykırı noktaları doğrudan ele alamaz. Bu nedenle, veri ön işleme adımları (eksik veri doldurma, aykırı değer işleme vb.) genellikle XGBoost ile birlikte kullanılmalıdır. XGBoost, diğer basit sınıflandırma veya regresyon algoritmalarına kıyasla daha yüksek bir hesaplama maliyetine sahip olabilir. Büyük veri setleri veya karmaşık problemler için daha uzun eğitim süreleri gerektirebilir.

XGBoost, genel olarak yüksek performans, iyi genelleme kabiliyeti ve özellik önemi değerlendirmesi gibi avantajları nedeniyle tercih edilen bir sınıflandırma ve regresyon algoritmasıdır. Ancak, doğru parametre ayarlarına ve veri ön işleme adımlarına dikkat edilmesi gerekmektedir.

##### 1.3.2.1.1.7 LightGBM

LightGBM, büyük ve karmaşık veri setleri üzerinde hızlı ve yüksek performanslı sınıflandırma ve regresyon problemleri için kullanılan bir makine öğrenmesi algoritmasıdır. LightGBM, Gradient Boosting yöntemini temel alır, ancak ağaç büyütme işleminde bazı optimizasyonlar yaparak daha verimli bir çalışma sağlar.

LightGBM algoritmasının çalışma prensibi, LightGBM, veri setini hafızada optimize edilmiş bir veri yapısı olan "Histogram Tabanlı Hesaplama" kullanarak temsil eder. Bu, hızlı hesaplamalar ve bellek kullanımının azaltılması için geliştirilmiş bir yaklaşımdır. Gradient Boosting yöntemini kullanarak genellikle karar ağaçları bir araya getirir. Ancak, LightGBM, diğer GBM uygulamalarına kıyasla ağaçların büyütülmesi sırasında bazı optimizasyonlar yapar. Farklı problemlere uyum sağlamak için özel kayıp fonksiyonlarına izin verir. Kayıp fonksiyonları, belirli bir probleme odaklanarak optimum tahminleri elde etmeyi sağlar.

LightGBM, ağaç büyütme sırasında daha verimli bir yaklaşım kullanır. LightGBM ağaç büyütme işlemini yaprak düzeyinde gerçekleştirir ve ağaçları paralel olarak büyütür. Bu, hesaplama süresini önemli ölçüde azaltır.

LightGBM, büyük veri setleri üzerinde hızlı bir şekilde çalışabilir. Ağaç büyütme işlemi, yaprak düzeyinde gerçekleştirildiği ve paralel olarak çalıştığı için hesaplama süresi önemli ölçüde azalır. LightGBM, veri setini hafızada optimize edilmiş bir histogram tabanlı veri yapısı kullanarak temsil eder. Bu, bellek kullanımını azaltır ve daha büyük veri setlerini işleme kabiliyetini artırır. LightGBM, aşırı öğrenmeye karşı dirençli olmak için çeşitli normalizasyon tekniklerini kullanır. Bu, daha iyi genelleme yapabilmesini sağlar ve veri setindeki gürültüye veya gereksiz özelliklere daha iyi tolerans gösterir. LightGBM, her bir özelliğin sınıflandırmadaki veya regresyondaki önemini değerlendirmek için bir özellik önemi skoru sağlar. Bu, hangi özelliklerin daha etkili olduğunu belirlemek ve veri setini daha iyi anlamak için faydalı bir bilgidir.

LightGBM, birçok hiperparametreye sahiptir ve doğru şekilde ayarlanmaları gerekmektedir. Yanlış ayarlanmış parametreler, modelin performansını etkileyebilir. Bu nedenle, hiperparametre optimizasyonu gerektirebilir. LightGBM, özellikle küçük veri setleri için tasarlanmamıştır. Küçük veri setlerinde diğer algoritmalara kıyasla daha düşük performans gösterebilir. LightGBM, kategorik değişkenlere otomatik olarak işlem yapabilme yeteneğine sahip değildir. Bu nedenle, kategorik değişkenleri önceden dönüştürmek veya kodlamak gerekebilir.

LightGBM, hızlı hesaplama yetenekleri, düşük bellek kullanımı, yüksek genelleme kabiliyeti ve özellik önemi değerlendirmesi gibi avantajları nedeniyle popüler bir sınıflandırma ve regresyon algoritmasıdır. Ancak, parametre ayarı ve küçük veri setlerindeki performansı gibi bazı dezavantajlar dikkate alınmalıdır.

##### 1.3.2.1.1.8 CatBoost

CatBoost, kategorik değişkenlere sahip veri setlerinde kullanılan bir sınıflandırma ve regresyon algoritmasıdır. CatBoost, Gradient Boosting yöntemini temel alır, ancak kategorik değişkenlere özel olarak tasarlanmış bir dizi teknik kullanarak daha iyi bir performans sağlar.

CatBoost algoritmasının çalışma prensibi, kategorik değişkenleri özel olarak ele alır ve bu değişkenleri doğrudan kullanabilme yeteneğine sahiptir. Otomatik olarak kategorik değişkenlerin içerisindeki düzgün bir şekilde kodlamasını yapar. Gradient Boosting yöntemini kullanarak genellikle karar ağaçları bir araya getirir. Ancak, CatBoost, diğer GBM uygulamalarına kıyasla kategorik değişkenlere özel olarak tasarlanmış bir dizi teknik kullanır. Ağaç büyütme işlemi sırasında asenkronluk kullanır. Bu, birden çok işlemci çekirdeğini veya makineyi kullanarak hesaplamaları hızlandırır.

CatBoost, kategorik değişkenleri özel olarak ele alır ve bu değişkenleri doğrudan kullanabilir. Değişkenlerin otomatik olarak kodlanması, kullanıcının ek ön işleme adımlarını yapmasını engeller. CatBoost, aşırı öğrenmeye karşı dirençli olmak için çeşitli normalizasyon tekniklerini kullanır. Bu, daha iyi genelleme yapabilmesini sağlar ve veri setindeki gürültüye veya gereksiz özelliklere daha iyi tolerans gösterir.

CatBoost, birçok hiperparametreye sahiptir ve doğru şekilde ayarlanmaları gerekmektedir. CatBoost, büyük veri setlerinde diğer algoritmalara kıyasla daha fazla bellek kullanabilir. Bu, özellikle bellek kısıtlamaları olan sistemlerde performans sorunlarına neden olabilir. CatBoost, uzun eğitim süreleri gerektirebilir. Bu, büyük veri setlerinde veya karmaşık modellerde eğitim süresinin önemli ölçüde artmasına neden olabilir. Özellikle yerel cihazlarda veya hafıza kısıtlamaları olan sistemlerde.

CatBoost, modelin büyüklüğünü artırabilir. Bu, modelin depolanması ve dağıtılması açısından bazı zorluklar yaratabilir. CatBoost'un birçok hiperparametresi olduğundan, bu parametrelerin doğru şekilde ayarlanması önemlidir.

CatBoost, kategorik değişkenlere özel tasarlanmış teknikler kullanarak yüksek performanslı sınıflandırma ve regresyon modelleri oluşturabilen güçlü bir algoritmadır. Ancak, bellek kullanımı, eğitim süresi ve model boyutu gibi bazı dezavantajları dikkate alınmalıdır.

##### 1.3.2.1.2 Regresyon

##### 1.3.2.1.2.1 Doğrusal regresyon algoritmaları

##### 1.3.2.1.2.1.1 Basit doğrusal regresyon

Bağımlı değişken ile yalnızca bir adet bağımsız değişken arasındaki doğrusal ilişkiyi modellemek için kullanılan bir doğrusal regresyon algoritmasıdır. Basit Doğrusal Regresyon, veri noktaları arasındaki doğrusal ilişkiyi en iyi şekilde ifade eden bir doğruyu bulmaya çalışır.

Basit Doğrusal Regresyon, bağımlı değişken (Y) ile bağımsız değişken (X) arasındaki ilişkiyi, denklemiyle ifade eder. Burada, a regresyon sabitini (kesim noktasını) ve b eğim (katsayıyı) temsil eder. Algoritma, veri noktalarının dağılımını dikkate alarak a ve b parametrelerini tahmin etmeye çalışır. Bu tahminler, en küçük kareler yöntemi gibi istatistiksel tekniklerle yapılır.

Basit ve anlaşılır bir modelleme yöntemi olması, modelin parametreleri kolayca yorumlanabilmesi, veri seti üzerinde hızlı bir şekilde eğitilip ve tahmin yapabilmesi ve veri noktaları arasındaki doğrusal ilişkiyi iyi yakalaması gibi avantajları vardır.

Yalnızca bir bağımsız değişkeni kullanılması dolayısıyla çoklu faktörlerin etkisi göz ardı edilmesi, doğrusal olmayan ilişkileri yakalayamaması yani veri noktaları arasında doğrusal olmayan bir ilişki varsa modelin performansı düşebilmesi, aykırı değerlere duyarlı olabilmesi yani modelin tahminlerini yanıltabilmesi ve varsayımlara dayalı çalıştığından, özellikle bağımsız değişkenlerin hatalarının normal olarak dağıldığı ve sabit varyans olduğu varsayımına dayanması gibi dezavantajları vardır.

Basit Doğrusal Regresyon, tek bir bağımsız değişkenin bağımlı değişken üzerindeki etkisini anlamak ve tahminler yapmak için kullanılan temel bir doğrusal regresyon yöntemidir. Ancak, karmaşık ilişkileri yakalayamaz ve birden fazla faktörü dikkate almaz.

##### 1.3.2.1.2.1.2 Çoklu doğrusal regresyon

Çoklu Doğrusal Regresyon, birden fazla bağımsız değişkenin bağımlı değişken üzerindeki etkisini modellemek için kullanılan bir doğrusal regresyon algoritmasıdır.

Çoklu Doğrusal Regresyon, bağımlı değişkeni, birden fazla bağımsız değişkenin ağırlıklı toplamı olarak ifade eder. Genel olarak, formülü kullanılır, burada Y bağımlı değişkeni, X1, X2, ..., Xn ise bağımsız değişkenleri temsil eder. a regresyon sabitini, b1, b2, ... bn ise bağımsız değişkenlerin katsayılarını ifade eder. Algoritma, veri setindeki bağımlı ve bağımsız değişkenler arasındaki en iyi uyumu sağlayacak a ve b katsayılarını tahmin etmeye çalışır.

Çoklu Doğrusal Regresyon, veri setindeki bağımlı değişkenin doğrusal olarak birden fazla bağımsız değişkenle ilişkili olduğu durumları modellemek için kullanılır. Algoritma, veri noktalarının dağılımını analiz ederek ve en küçük kareler yöntemi gibi istatistiksel teknikler kullanarak a ve b parametrelerini tahmin eder.

Birden fazla bağımsız değişkeni dikkate alabilmesi sayesinde karmaşık ilişkileri yakalayabilmesi, modelin parametreleri kolayca yorumlanabilmesi gibi avantajları vardır.

Doğrusal ilişkileri modelleyebildiği için doğrusal olmayan ilişkileri yakalayamaması, aşırı öğrenmeye eğilimli olabilmesi, bağımsız değişkenler arasında çoklu doğrusal ilişkiler olduğunda modelin performansı düşebilmesi gibi dezavantajlara sahiptir.

Çoklu Doğrusal Regresyon, bağımsız değişkenlerin etkisini doğru bir şekilde yakalamak için doğru varsayımlara dayanır. Bu varsayımlardan biri, hata terimlerinin bağımsız değişkenlere göre normal dağılım göstermesidir. Eğer hata terimleri normal dağılım göstermezse, modelin istatistiksel sonuçları güvenilir olmayabilir. Ayrıca, Çoklu Doğrusal Regresyon, bağımsız değişkenler arasındaki çoklu doğrusal ilişkileri ele alamaz. Eğer bağımsız değişkenler arasında yüksek korelasyon varsa, modelin performansı düşebilir ve parametre tahminleri istikrarsız olabilir. Veri setindeki aykırı değerler Çoklu Doğrusal Regresyon modelini etkileyebilir. Aykırı değerler, modelin parametre tahminlerini yanıltabilir ve sonuçların güvenilirliğini azaltabilir. Bu nedenle, aykırı değerlerin tespit edilmesi ve yönetilmesi önemlidir.

##### 1.3.2.1.2.1.3 PCR

PCR, doğrusal regresyon yöntemiyle birlikte PCA kullanarak modelleme yapmayı amaçlayan bir regresyon algoritmasıdır. PCA, bağımsız değişkenler arasındaki çoklu doğrusal ilişkileri ele almak ve değişkenlerin boyutunu azaltmak için kullanılır.

PCR çalışma prensibi, veri setindeki bağımsız değişkenler üzerinde PCA uygulanır. Bu adımda, bağımsız değişkenlerin varyansını açıklamak için temel bileşenler elde edilir. Temel bileşenler, veri setindeki değişkenlik açısından en fazla bilgiyi içeren doğrusal kombinasyonlardır. PCA sonucunda elde edilen temel bileşenler arasından, bağımlı değişkenle en iyi ilişkiye sahip olanları seçmek için istatistiksel yöntemler kullanılır. Bu seçim işlemi, bağımlı değişkenle ilişkisi en yüksek olan temel bileşenlerin belirlenmesini sağlar. Seçilen temel bileşenler, doğrusal regresyon modelinde bağımlı değişkenin tahmin edilmesi için kullanılır. Bu adımda, temel bileşenlerin ve bağımlı değişkenin ilişkisini ifade eden regresyon katsayıları tahmin edilir.

PCR, doğrusal regresyon yöntemiyle PCA’nin birleştirilmesini sağlayan bir regresyon algoritmasıdır. Avantajları arasında çoklu doğrusal ilişkileri ele alabilme, değişken boyutunu azaltma ve yorumlanabilir katsayılar elde etme bulunurken, doğrusal olmayan ilişkileri modelleyememe, bazı bilgilerin kaybedilmesi ve aykırı değerlerin etkisi dezavantajları arasındadır.

##### 1.3.2.1.2.1.4 PLS

PLS, hem bağımsız değişkenler hem de bağımlı değişkenler arasındaki çoklu doğrusal ilişkileri ele alan bir doğrusal regresyon algoritmasıdır. PLS, bağımlı değişkenin en fazla varyansını açıklayan bileşenleri belirler ve bu bileşenler üzerinden regresyon modelini oluşturur.

PLS algoritmasının çalışma prensibi, bağımlı değişkenin en fazla varyansını açıklayan bileşenleri belirler. Bu adımda, bağımlı değişkenle bağımsız değişkenler arasındaki ilişkiyi en iyi yakalayan bileşenler seçilir. Seçilen bileşenler, bağımlı değişken ve bağımsız değişkenler arasındaki ilişkiyi ifade eden yeni bileşenler oluşturulur. Oluşturulan bileşenler üzerinden doğrusal regresyon modeli oluşturulur. Bu adımda, bağımsız değişkenlerin regresyon katsayıları tahmin edilir.

PLS, çoklu doğrusal ilişkileri ele alabilme yeteneğine sahiptir. Bağımsız değişkenler arasındaki çoklu doğrusal ilişkileri modelleyebilir ve bu şekilde veri setindeki değişkenlik açıklanabilmesi, bileşenlerin seçimi ve oluşturulması sayesinde, PLS veri setindeki boyutu azaltılması nedeniyle modelin hesaplama süresini ve bellek gereksinimlerini azaltabilmesi ve aşırı öğrenmeyi önleyebilme yeteneğine sahip olması gibi avantajlara sahiptir.

PLS, bağımlı değişkeni açıklayan bileşenleri seçerken subjektif bir yaklaşım kullanır. Bu nedenle, bileşenlerin seçimi sübjektif olabilir ve modelin sonuçları üzerinde etkisi olabilir. PLS, veri setindeki aykırı değerlere hassastır. PLS, bazı durumlarda aşırı öğrenme eğilimine sahip olabilir. Eğer veri setindeki gürültü miktarı yüksekse veya bağımsız değişkenler arasında güçlü doğrusal ilişkiler bulunmuyorsa, PLS modeli aşırı uyuma meyilli olabilir. PLS, çoklu doğrusal ilişkilerin yanı sıra doğrusal olmayan ilişkileri modelleyemez. Eğer veri setinde doğrusal olmayan ilişkiler bulunuyorsa, PLS modeli bu ilişkileri yakalayamaz. PLS, büyük ölçekli veri setlerinde hesaplama süresi ve bellek gereksinimleri açısından daha fazla kaynak tüketebilir. Özellikle çok sayıda bağımsız değişken olduğunda, PLS'nin hesaplama maliyeti artabilir.

##### 1.3.2.1.2.1.5 Ridge

Ridge Regresyonu, doğrusal regresyon yöntemlerinden biridir ve aşırı öğrenme problemini çözmek amacıyla kullanılır. Bu algoritma, regresyon modeline bir düzenlilik terimi ekleyerek, modelin katsayılarını sınırlar.

Ridge Regresyonu'nun çalışma prensibi; Ridge Regresyonu, hata kareler toplamını minimize etmek için kullanılır, ancak aynı zamanda katsayıların büyüklüğünü de sınırlamak için bir düzenlilik terimi ekler. Bu düzenlilik terimi, katsayıların mutlak değerlerinin toplamını cezalandırır. Ridge Regresyonu, katsayıların büyük değerler almasını engelleyerek modelin aşırı uyuma eğilimini azaltır.

Ridge Regresyonu, aşırı uyumu azaltarak modelin genelleme yeteneğini artırarak veri setindeki gürültüyü ve aşırı değişkenlikleri kontrol etmesi, bağımsız değişkenler arasında yüksek korelasyonlar bulunuyorsa Ridge Regresyonu bu ilişkileri modelleyebilmesi gibi avantajları vardır.

Ridge Regresyonu, düzenlilik terimi için bir hiperparametre olan lambda değerini belirlemeyi gerektirir. Bu değeri optimize etmek, modelin performansını etkileyebilir ve deneyimsel olarak belirlenmelidir. Katsayıların küçültülmesi nedeniyle modelin yorumlanabilirliğini azaltabilir. Katsayılar, orijinal özelliklerin doğrudan yorumlanması zor olduğunda, genellikle etkilerini anlamak için ek analizler gerektirir.

Ridge Regresyonu, doğrusal olmayan ilişkileri modelleyemez. Eğer veri setinde doğrusal olmayan ilişkiler bulunuyorsa, Ridge Regresyonu bu ilişkileri yakalayamaz.

1.3.2.1.2.1.6 Lasso

Lasso Regresyonu, doğrusal regresyon yöntemlerinden biridir ve düzenlilik yöntemini kullanarak aşırı öğrenmeyi kontrol etmeyi amaçlar.

Lasso Regresyonu çalışma prensibi, hata kareler toplamını minimize etmek için kullanılır, ancak Ridge Regresyonu'ndan farklı olarak, katsayıların mutlak değerlerinin toplamını cezalandırır. Bu sayede, Lasso Regresyonu, bazı katsayıları tamamen sıfır yaparak değişken seçimi yapar. Böylece, modelde etkisiz veya gereksiz olan bağımsız değişkenlerin etkisini ortadan kaldırır.

Lasso Regresyonu, aşırı uyumu azaltarak modelin genelleme yeteneğini artırması, gürültülü veri setlerinde daha iyi performans gösterebilmesi, değişken seçimi yaparak modelin daha anlaşılır ve yorumlanabilir olmasını sağlaması ve etkisiz veya gereksiz bağımsız değişkenleri çıkararak, modelin sadeleştirilmesini sağlaması gibi avantajları vardır.

Lasso Regresyonu, düzenlilik terimi için bir hiperparametre olan lambda değerini belirlemeyi gerektirir. Bu değeri optimize etmek, modelin performansını etkileyebilir ve deneyimsel olarak belirlenmelidir. Eğer veri setinde bağımsız değişkenler arasında yüksek korelasyonlar bulunuyorsa, Lasso Regresyonu bazı değişkenleri sıfır yapabilir. Bu durumda, modelin yorumlanabilirliği azalabilir ve bazı önemli değişkenlerin etkisi göz ardı edilebilir. Özellikle büyük ölçekli veri setlerinde veya çok sayıda bağımsız değişken olduğunda. Lasso Regresyonu, Ridge Regresyonu'na kıyasla daha hesaplama açısından yoğun olabilir.

##### 1.3.2.1.2.1.7 ElasticNet

ElasticNet Regresyonu, doğrusal regresyon yöntemlerinden biridir ve Ridge Regresyonu ile Lasso Regresyonu'nun birleşimini sağlayarak hem düzenlilik yapmayı hem de değişken seçimini gerçekleştirir.

ElasticNet Regresyonu, hem L1 hem de L2 düzenlilik terimlerini kullanarak, Ridge ve Lasso Regresyonu'nun avantajlarını birleştirmesi sayesinde, modelin aşırı uyumu kontrol ederken değişken seçimi yapabililmesi, değişkenler arasındaki korelasyonları ele alırken, aynı zamanda gürültülü veri setlerinde iyi bir performans gösterebilmesi sayesinde büyük veri setlerinde etkili olabilmesi, ElasticNet Regresyonu, Ridge ve Lasso Regresyonu'na göre daha esnek bir yöntem olduğu için iki hiperparametre olan alpha ve lambda değerlerini ayarlayarak düzenlilik düzeyini ve değişken seçimi esnekliğini kontrol edebilmesi gibi avantajları vardır.

ElasticNet Regresyonu, hiperparametre seçimi gerektirir. Alpha (α) değeri, Ridge ve Lasso düzenlilik terimleri arasındaki oranı belirlerken, lambda (λ) değeri düzenlilik düzeyini kontrol eder. Bu değerlerin optimize edilmesi, modelin performansını etkileyebilir. Veri setindeki bağımsız değişkenler arasında yüksek korelasyonlar varsa, ElasticNet Regresyonu bazı değişkenleri sıfır yapabilir. Bu durumda, modelin yorumlanabilirliği azalabilir ve önemli değişkenlerin etkisi göz ardı edilebilir. Özellikle büyük veri setlerinde veya çok sayıda bağımsız değişken olduğunda ElasticNet Regresyonu, Ridge ve Lasso Regresyonu'na kıyasla daha hesaplama açısından yoğun olabilir.

##### 1.3.2.1.2.2 Doğrusal olmayan regresyon algoritmaları

##### 1.3.2.1.2.2.1 CART

CART, doğrusal olmayan regresyon problemlerini çözmek için kullanılan bir karar ağacı algoritmasıdır. CART algoritması, veri setindeki desenleri ve ilişkileri kullanarak bir ağaç yapısı oluşturur.

CART algoritmasının çalışma prensibi, veri setinin en iyi bölünme noktasını bulmak için bir özellik ve eşik değeri seçilir. Veri seti bu bölünme noktasına göre iki alt kümeye ayrılır. Ağaç oluşturma adımı, veri setini en iyi şekilde bölen bir kriter kullanılarak gerçekleştirilir. Bu işlem, veri seti alt kümeleri için tekrarlanır ve ağaç yapısı büyüyerek devam eder. Her bir adımda, en iyi bölünme noktası seçilerek veri seti alt kümeleri oluşturulur. Büyüme durdurma kriterleri genellikle ağacın maksimum derinliği, minimum örnek sayısı veya düğüm homojenliği gibi faktörlere dayanır. Oluşturulan ağaç genellikle aşırı öğrenmeye yatkın olabilir. Bu nedenle, ağaç üzerinde budama işlemi gerçekleştirilir. Budama işlemi, ağaçtaki düğümleri ve dalları kaldırarak modelin genelleme yeteneğini artırır. Budama işlemi, çeşitli kriterlere göre (örneğin, azaltılan hata, çapraz doğrulama hatası) yapılır.

Doğrusal olmayan ilişkileri modelleyebilmesi ve veri setindeki karmaşık yapıları keşfedebilmesi, ağaç yapısı sayesinde her bir düğümdeki özellikleri ve bölünme kriterlerini açık bir şekilde göstermesi sayesinde yorumlanabilirlik sağlaması, değişkenlerin ölçeği veya dağılımı üzerinde ön işleme gerektirmemesi sayesinde doğrudan veri seti üzerinde çalışabilmesi ve karar ağaçlarının kategorik ve sayısal verileri bir arada kullanabilmesi gibi avantajlar vardır.

Çok büyük veya karmaşık veri setlerinde aşırı uyum riski vardır. Ağaç yapısı, veri setine tam olarak uyan bir model oluşturma eğrilebilir, bu da aşırı öğrenmeye yol açabilir. Karar ağaçları, veri setindeki küçük değişikliklere duyarlı olabilir. Veri setindeki küçük bir değişiklik bile ağaç yapısının tamamen farklı olmasına neden olabilir. Ağaç yapısı, veri setindeki hatalı veya eksik değerlere hassas olabilir. Bu tür değerlerin doğru bir şekilde işlenmemesi, ağaç yapısının performansını etkileyebilir.

##### 1.3.2.1.2.2.2 SVR

SVR, doğrusal olmayan regresyon problemlerini çözmek için kullanılan bir makine öğrenmesi algoritmasıdır. SVR, SVM temel alınarak geliştirilmiştir. SVM'deki sınıflandırma problemini regresyon problemlerine uyarlamak için kullanılır.

SVR'nin çalışma prensibi, veri setindeki bağımsız değişkenler belirli bir özellik uzayına dönüştürülür. Bu dönüşüm, veri setinin doğrusal veya doğrusal olmayan bir şekilde ayrılmasını kolaylaştırır. SVR modeli, dönüştürülen veri seti üzerinde eğitilir. Eğitim sürecinde, bir hiperdüzlem (veya hiper-düzlemler) oluşturulur. Bu hiperdüzlem, veri noktalarının en yakınındaki destek vektörlerin mümkün olduğunca yakınından geçer ve hata toleransı (epsilon) içinde maksimum marjinal ayrım elde edilmeye çalışılır. Eğitim süreci tamamlandıktan sonra, SVR modeli, yeni gelen veri noktalarının hedef değerlerini tahmin etmek için kullanılabilir. Tahmin işlemi, test veri seti üzerinde gerçekleştirilir ve modelin öğrendiği hiperdüzlem tarafından belirlenen bir regresyon fonksiyonu kullanılır.

Özetlemek gerekirse, avantajları arasında doğrusal olmayan problemleri çözebilme, gürültüye dayanıklılık, genelleme yeteneği ve yüksek boyutlu verilerde etkinlik yer alırken, dezavantajları arasında hiperparametre ayarının önemi, hesaplama yoğunluğu ve yorumlanabilirlik zorlukları yer almaktadır. Bu nedenle, veri setinin özelliklerine ve problemin gereksinimlerine bağlı olarak kullanılması uygun olabilir.

##### 1.3.2.1.2.2.3 Bagged Trees

Bagged Trees Regresyonu, doğrusal olmayan regresyon problemlerini çözmek için kullanılan bir makine öğrenmesi algoritmasıdır. Bagging olarak adlandırılan bir teknik kullanır ve temel olarak birçok karar ağacının bir araya getirilerek bir tahmin yapılmasını sağlar.

Bagged Trees Regresyon çalışma prensibi, veri setinin bootstrap yöntemiyle tekrar tekrar örnekleme yapılır. Her bir örnekleme, aynı boyutta ancak orijinal veri setinden rastgele örnekler içerir. Her örnekleme üzerinde ayrı ayrı karar ağaçları oluşturulur. Her bir ağaç, örneklemeyle oluşturulan veri setine göre eğitilir. Karar ağaçları genellikle tam büyüklükte oluşturulur ve her bir ağaç, veri setindeki farklı desenleri ve ilişkileri yakalamaya çalışır. Test veri seti üzerinde her bir karar ağacı için tahmin yapılır. Ardından, tüm ağaçların tahminleri birleştirilerek bir ortalaması veya medyanı alınır. Bu, final tahmini oluşturur.

Bagged Trees Regresyonu, doğrusal olmayan ilişkileri modellemesi, çoklu ağaçların bir araya getirilmesiyle birlikte modelin daha genellemesi, aşırı uyum riskini azaltması açısından faydalı olması ve gürültülü verilere karşı dirençli olması ve veri setindeki anormal değerlerin etkisini azaltabilmesi gibi avantajları vardır.

Her bir ağaç bağımsız olarak eğitildiği için yüksek hesaplama gücü ve zaman gerektirebilmesi, birden fazla karar ağacının birleştirilmesi nedeniyle sonuçların yorumlanabilirliği zorlaşabilmesi, eğitim veri seti boyutuna bağımlı olarak hafıza gereksinimleri artabilmesi gibi dezavantajları vardır.

##### 1.3.2.1.2.2.4 GBM

GBM, doğrusal olmayan regresyon problemlerini çözmek için kullanılan bir makine öğrenmesi algoritmasıdır. GBM, genellikle karar ağaçları bir araya getirilmesiyle güçlü bir tahmin modeli oluşturur.

GBM Regresyonunun çalışma prensibi, veri setinin ortalaması veya medyanı gibi bir basit tahmin yapılır.İlk tahmin ile gerçek değerler arasındaki hata hesaplanır. Bir karar ağacı oluşturulur ve hata üzerine odaklanır. Karar ağacı, hata ile ilişkili özelliklerin bölme noktalarını belirler. Oluşturulan karar ağacı, hata ile çarparak ağırlıklandırılır.

İlk tahminin üzerine ağırlıklandırılmış karar ağacının tahmini eklenir. Bu, yeni bir tahmin değeri oluşturur. Tüm tahminler bir araya getirilerek final tahmin değeri oluşturulur.

GBM, doğrusal olmayan ilişkileri modellemesi, veri setindeki gürültüye ve anormal değerlere karşı dirençli olması, özelliklerin önem sıralamasını sağlaması ve bu sayede değişken seçimi kolaylaştırması gibi avantajları vardır. Dezavantajları arasında GBM, diğer basit modellere göre daha karmaşık bir yapıya sahiptir, bu nedenle eğitim ve ayarlama süreci daha uzun sürebilir. Modelin aşırı öğrenme riski vardır, bu nedenle uygun düzenleme ve parametre ayarları gerektirebilir. Özellikle büyük veri setlerinde ve derin ağaç yapılarında yüksek hesaplama gücü ve bellek gerektirebilir.

##### 1.3.2.1.2.2.5 XGBoost

XGBoost, doğrusal olmayan regresyon problemlerini çözmek için kullanılan bir makine öğrenmesi algoritmasıdır. XGBoost, Gradient Boosting yöntemini temel alır, ancak daha fazla optimize edilmiş ve performansı iyileştirilmiştir.

XGBoost'un çalışma prensibi, veri setinin ortalaması veya medyanı gibi bir basit tahmin yapılır.İlk tahmin ile gerçek değerler arasındaki hata hesaplanır. Bir karar ağacı oluşturulur ve hata üzerine odaklanır. Karar ağacı, hata ile ilişkili özelliklerin bölme noktalarını belirler. Oluşturulan karar ağacının tahmini kullanılarak hesaplanan hata değeri belirlenir. Hata değerine göre bir gradiyan hesaplanır. Bu gradiyan, bir sonraki karar ağacının oluşturulmasında kullanılır. Her bir ağacın tahmin değeri öğrenme oranı ve ağırlıklandırma kullanılarak belirlenir. Bu döngü ile her bir iterasyonda yeni bir karar ağacı oluşturulur ve hata düzeltilir. Tüm tahminler bir araya getirilerek final tahmin değeri oluşturulur.

XGBoost, yüksek tahmin performansı sağlayan güçlü bir algoritma olması, özelliklerin önem sıralamasını sağlaması ve değişken seçimine yardımcı olması, aşırı öğrenmeye dirençlidir ve iyi bir genelleme yapabilmesi, paralel hesaplama ve hızlı öğrenme yetenekleri sayesinde büyük veri setlerinde etkili çalışabilmesi gibi avantajları vardır. Dezavantajları arasında ise, yüksek hesaplama gücü ve bellek gerektirebilir, özellikle büyük veri setlerinde ve derin ağaç yapılarında. Parametre ayarları ve model ayarlama süreci daha karmaşık olabilir.Veri setindeki gürültüye karşı hassas olabilir.

##### 1.3.2.1.2.2.6 LightGBM

Light GBM, doğrusal olmayan regresyon problemlerinde kullanılan bir makine öğrenmesi algoritmasıdır. XGBoost ve Gradient Boosting gibi diğer gradient boosting yöntemleriyle benzer bir prensibe sahiptir, ancak bazı farklılıkları vardır.

Light GBM, karar ağaçlarının ensemble olarak oluşturulduğu bir yöntemdir. Temel olarak, karar ağaçları ardışık olarak birleştirilerek güçlü bir tahmin modeli oluşturulur. Light GBM, bu ağaçları oluştururken önceki ağaçlardan gelen hataları düzeltmeye odaklanır. Algoritma, veri setini küçük alt gruplara böler ve her alt grup üzerinde bir karar ağacı oluşturur. Daha sonra, oluşturulan ağaçların hatalarını hesaplayarak, bir sonraki ağaç için öncelikli olarak bu hatalara odaklanır. Bu şekilde, önceki ağaçların hatalarını düzelten yeni ağaçlar oluşturulur. Sonunda, tüm ağaçlar birleştirilerek final tahmin modeli elde edilir.

Light GBM, paralel hesaplama yetenekleri sayesinde büyük veri setleriyle bile başa çıkabilmesi, hızlı ve verimli bir şekilde çalışabilmesi, diğer gradient boosting yöntemlerine kıyasla daha az bellek kullanması ve özellikle karmaşık ve yüksek boyutlu veri setlerinde yüksek tahmin doğruluğu sağlaması gibi avantajlara sahiptir. Dezavantajları arasında ise Light GBM, veri setindeki gürültüye karşı hassas olabilir. Gürültülü veya anormal değerler içeren veri setleriyle çalışırken dikkatli olunması gerekir. Eğer veri seti dengesiz bir sınıf dağılımına sahipse (imbalance), Light GBM'nin yanlılık yapma eğilimi olabilir. Bu durumda, sınıf ağırlıklarının ayarlanması veya örnekleme teknikleri kullanılması gerekebilir. Light GBM'nin birkaç önemli parametresi vardır ve bu parametrelerin doğru ayarlanması önemlidir. Yanlış parametre ayarları, modelin performansını olumsuz etkileyebilir.

##### 1.3.2.1.2.2.7 CatBoost

CatBoost, kategorik değişkenleri etkili bir şekilde işleyen bir doğrusal olmayan regresyon algoritmasıdır. Bu algoritma, diğer yöntemlere kıyasla daha iyi bir performans ve doğruluk sağlamak amacıyla özellikle kategorik değişkenlerin işlenmesinde geliştirmeler içermektedir.

CatBoost algoritması, içerisinde kategorik değişkenler bulunan veri setlerini otomatik olarak işleyebilme özelliğine sahiptir. Bu, veri setindeki kategorik değişkenlerin sayısal değerlere dönüştürülmesi ve kategorik değişkenlerin önemli olduğu durumlarda ayrı bir işlem adımı eklenmesi gerekmeksizin doğrudan kullanılabilmesi anlamına gelir.

CatBoost algoritmanın çalışma prensibi, gradient boosting yöntemine dayanmaktadır. Birçok karar ağaçlarından oluşan bir ansambl oluşturur ve bu öğrenicileri adım adım geliştirerek tahminleri iyileştirir. Ayrıca, öğrenme sürecinde değişken önemini otomatik olarak belirler ve bu önemi kullanarak modelin doğruluğunu artırır.

CatBoost'un avantajları olarak, kategorik değişkenlerin doğrudan kullanılabilmesidir. Diğer yöntemlerde kategorik değişkenlerin dönüştürülmesi veya etiketlenmesi gerekebilirken, CatBoost bu işlemi otomatik olarak yapar ve bu da zaman tasarrufu sağlar. Ayrıca, yüksek performanslı ve ölçeklenebilir bir algoritmadır. Büyük veri setleri üzerinde etkili bir şekilde çalışabilir ve paralel işleme yetenekleri sayesinde hızlı sonuçlar elde edebilir. Ancak, CatBoost'un bazı dezavantajları da vardır. Öncelikle, eğitim süreci diğer algoritmalara göre daha uzun sürebilir, özellikle büyük ve karmaşık veri setleriyle çalışırken. Ayrıca, modelin daha karmaşık olması ve daha fazla bellek kullanması nedeniyle, bazı durumlarda yüksek bellek gereksinimleri olabilir. Bununla birlikte, bu dezavantajlar genellikle büyük ölçekli ve karmaşık problemlerle uğraşan kullanıcılar için önemli bir sorun teşkil etmez.

#### 1.3.2.2 Denetimsiz öğrenme

Bu tür algoritmalar, etiketlenmemiş veri setleriyle çalışır. Algoritma, verilerdeki gizli yapıları, desenleri veya gruplamaları tespit etmek için veriye dayalı olarak bir model oluşturur.

1.3.2.2.1 K-Means

K-Means, denetimsiz öğrenme algoritmalarından biridir ve kümeleme problemlerini çözmek için kullanılır. Bu algoritma, veri setindeki örnekleri belirli bir sayıda kümeye gruplandırır. Her bir küme, benzer özelliklere sahip örnekleri içerir.

K-Means algoritmasının çalışma prensibi, kullanıcı tarafından belirtilen k sayısı kadar küme merkezi rastgele seçilir. Her bir örnek, en yakın küme merkezine atanır. Öklid mesafesi, Manhattan mesafesi veya benzer bir mesafe ölçütü kullanılarak uzaklık hesaplanır. Her örnek atandıktan sonra, küme merkezleri yeniden hesaplanır. Her kümenin yeni merkezi, o kümeye ait örneklerin ortalaması olarak belirlenir. Küme merkezleri değişmeyene kadar tekrarlanır. Küme merkezleri artık değişmiyorsa, algoritma sonlanır.

Basit ve hızlı bir algoritma olması, büyük veri setleriyle bile etkili bir şekilde çalışabilir olması, veri setindeki benzer örnekleri bir araya getirerek anlamlı gruplamalar yapabilmesi, kullanıcı tarafından belirlenmesi gereken tek parametre k sayısı olması gibi avantajları vardır. Dezavantajları arasında ise, başlangıçta rastgele seçilen küme merkezleri, sonuçları etkileyebilir ve farklı başlangıç noktaları farklı sonuçlar üretebilir.Küme sayısının önceden bilinmesi gereklidir. Yanlış k değeri seçimi, yanlış gruplamalara yol açabilir. Küme merkezlerine olan başlangıç noktalarının hassas seçilmediği durumlarda, lokal minimumlara sıkışabilir ve global optimumu bulamayabilir.

1.3.2.2.2 PCA

PCA, denetimsiz öğrenme algoritmalarından biridir ve veri setinin boyutunu azaltmak ve değişkenler arasındaki ilişkileri anlamak için kullanılır.

PCA'nın çalışma prensibi, veri setindeki her bir değişkenin ortalamasını sıfırlayarak veriyi standartlaştırma işlemi yapılır. Standartlaştırılmış veri üzerinde kovaryans matrisi hesaplanır. Kovaryans matrisi, değişkenler arasındaki ilişkileri ölçer. Kovaryans matrisinin özvektörleri ve özdeğerleri bulunur. Özvektörler, yeni eksenleri temsil ederken, özdeğerler ise her bir özvektörün varyansı temsil ettiği önem düzeyini ifade eder. Özvektörler, özdeğerlerine göre sıralanır ve en yüksek özdeğere sahip olanlar en önemli bileşenler olarak seçilir. İstenilen boyutta (genellikle daha düşük boyutta) bir bileşen uzayı seçilir ve veri seti bu yeni bileşen uzayına projeksiyon yapılır.

Veri setindeki boyutu azaltarak veri işleme süresini ve bellek gereksinimlerini azaltabilmesi, değişkenler arasındaki ilişkileri anlamayı sağlar ve veri setinin temel yapılarını ortaya çıkarması, gürültülü veriye karşı dirençli olması,veri setindeki anlamsız değişkenlerin etkisini azaltması, görselleştirme ve veri keşfi için kullanışlı olması gibi avantajları vardır. Dezavantajları arasında ise, PCA, veri setindeki açıklanabilirlik oranını göz ardı eder. Bazen, veri setinin önemli bir kısmını açıklayamayan bileşenlere sahip olabilir. PCA, değişkenler arasındaki ilişkileri lineer olarak ele alır. Eğer veri setinde non-lineer ilişkiler varsa, PCA bunları yakalayamayabilir. PCA, yalnızca veri setindeki değişkenlerin yapısını açıklar, ancak veri setindeki herhangi bir etiket veya sınıflandırma bilgisini dikkate almaz. PCA, boyut azaltma, veri görselleştirme ve veri setinin temel yapılarını anlama gibi birçok uygulama alanında kullanılan yayggın bir denetimsiz öğrenme algoritmasıdır. Özellikle büyük ve karmaşık veri setleriyle çalışırken faydalı olabilir.

PCA'nın kullanım alanları olarak, boyut azaltma yani PCA, yüksek boyutlu veri setlerini daha düşük boyutlu bir bileşen uzayına dönüştürerek boyut azaltma sağlar. Bu, veri setinin karmaşıklığını azaltırken önemli bilgi kaybını en aza indirir. Örneğin, görüntü işleme veya biyomedikal veriler gibi yüksek boyutlu veri setleri üzerinde kullanılabilir. Veri görselleştirme yani PCA, veri setindeki değişkenlerin arasındaki ilişkileri anlamak ve veriyi daha kolay görselleştirmek için kullanılabilir. Veri setini iki veya üç boyutlu bir bileşen uzayına dönüştürerek veri noktalarını daha iyi anlayabileceğimiz bir şekilde görselleştirebiliriz. Veri sıkıştırma yani PCA, veri setini daha az bellek alanına sıkıştırarak veri saklama veya veri iletimi gibi alanlarda kullanılabilir. Önemli bilgi kaybını minimize ederek veriyi daha az miktarda bellek veya depolama alanında tutabiliriz. Veri Ön İşleme yani PCA, veri setindeki gürültüyü ve anlamsız değişkenleri ortadan kaldırarak veri ön işleme aşamasında kullanılabilir. Bu sayede daha temiz ve anlamlı veri setleri elde edebiliriz.

#### 1.3.2.3 Pekiştirmeli öğrenme

Bir ajanın etkileşim halinde olduğu bir çevre içinde, belirli bir hedefi gerçekleştirmek için en uygun davranışları öğrenme sürecini ifade eder. Bu öğrenme süreci, ajanın çevre ile etkileşim halinde olduğu denemeler (trial) ve hatalar (error) üzerinden gerçekleşir.

Pekiştirmeli öğrenme, temelde bir ödül-ceza (reward-punishment) mekanizmasıyla çalışır. Ajan, çevreyle etkileşimde bulunur ve her etkileşimde bir durum görür. Duruma bağlı olarak bir eylem seçer ve bu eylemi gerçekleştirir. Ardından çevreden bir geri bildirim alır, bu geri bildirim ödül veya ceza şeklinde olabilir. Ajanın amacı, toplamda en yüksek ödülü elde etmek için doğru eylemleri seçmeyi öğrenmektir.

Reinforcement Learning'de, ajanın öğrenme süreci genellikle bir dizi adımdan oluşur. Başlangıçta ajan, belirli bir durumda bulunur ve duruma bağlı olarak bir eylem seçer. Seçilen eylem çevreye uygulanır ve ajan yeni bir duruma geçer. Bu süreç tekrarlanarak ajanın davranışları ve stratejileri gelişir.

Reinforcement Learning, farklı uygulama alanlarında etkili bir şekilde kullanılabilmesi, belirli bir hedefi gerçekleştirmek için en uygun davranışları öğrenme üzerine odaklanması sayesinde belirli hedefler için optimize edilmiş çözümler bulunabilmesi, ajanların çevreyle etkileşim halindeyken en uygun kararları verme yeteneğini geliştirmesi sayesinde karmaşık karar verme problemlerine yönelik çözümler üretilebilmesi gibi avantajları vardır. Dezavantajları arasında ise, Reinforcement Learning algoritmaları genellikle veriye dayalı öğrenme yöntemleri olduğu için, veri eksikliği durumunda performansları etkilenebilir. Reinforcement Learning, bazen uzun süreler boyunca etkileşim halinde olduğu çevreyle denemeler yaparak öğrenme gerçekleştirir. Bu süreç, zaman ve hesaplama gücü gerektirebilir. Özellikle karmaşık çevrelerde ve büyük veri setleriyle çalışırken, öğrenme sürecinin tamamlanması uzun sürebilir. Reinforcement Learning, ajanın çevreyle etkileşimde olduğu gerçek dünyaya dayanır. Bu durumda, yanlış kararlar alınabilir veya yanlış ödüller verilebilir. Bu da öğrenme sürecinin etkisini olumsuz yönde etkileyebilir. Reinforcement Learning algoritmaları, ajanın davranışını şekillendiren bir ödül fonksiyonuna dayanır. İyi bir ödül fonksiyonunun belirlenmesi önemlidir çünkü yanlış veya yanıltıcı bir ödül fonksiyonu, ajanın istenmeyen davranışlar sergilemesine veya hedefe ulaşamamasına neden olabilir.

## 1.4 Veri Bilimi Ve Python

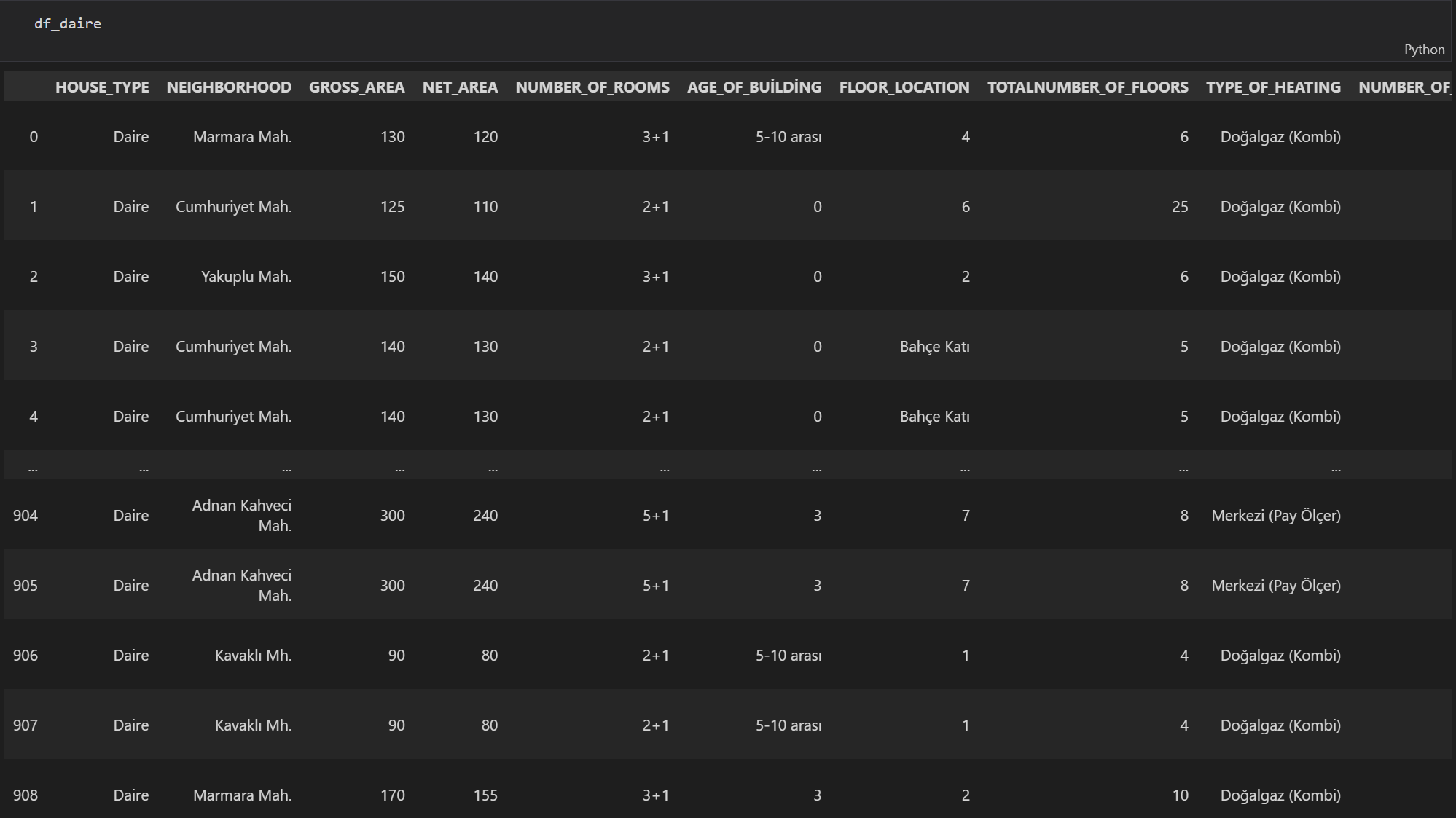
Dünyada veri bilimi çalışma alanlarına olan ilgi özellikle son yıllarda belirgin bir artış göstermiş bulunuyor. Bunda şüphesiz yapay zekâ ve büyük veri çalışmalarının payı büyüktür. Makine Öğrenmesi, Veri Bilimi, Derin Öğrenme, Veri Analitiği, Veri Görselleştirme ile en bağlantılı aramalardaki konu başlıkları detaylı incelendiğinde Python programlama dili ön plana çıkıyor. Python, yapay zeka, makine öğrenmesi ve veri analitiği gibi alanlarda popüler bir dil olmasının nedenleri arasında kolay okunabilirlik ve basitlik, geniş kütüphane desteği, farklı işletim sistemlerinde çalışabilir olması büyük topluluk ve iyi dokümantasyon, geniş uygulama alanları bulunuyor olması Python'un yapay zeka ve makine öğrenmesi alanında tercih edilen bir dil olmasını sağlamıştır. Ancak, her projenin ihtiyaçları farklı olabilir ve başka dillerin tercih edilmesi gerekebilir.

# İKİNCİ BÖLÜM

# UYGULAMA

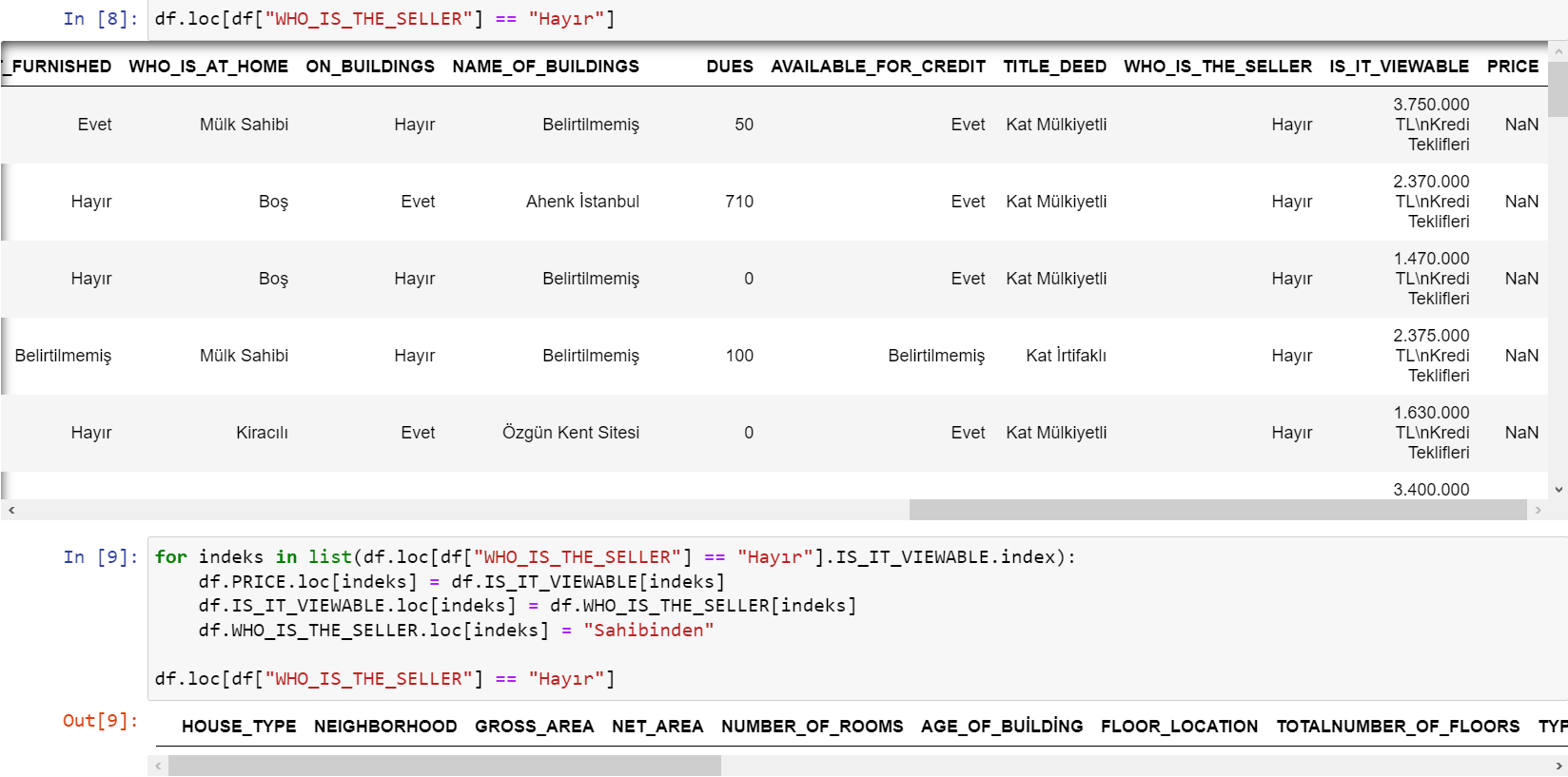
## 2.1 Veri Toplama

İstanbul’un Beylikdüzü bölgesindeki emlak sitesinde olan ilanları toplamak için kullanılan bir fonksiyon tanımlandı. Bu fonksiyon sonunda emlak sitesinde konut tipine göre oluşturulan df\_daire, df\_residence, df\_müstakilEv ve df\_villa gibi dört farklı DataFrame'i birleştirmek için Python’da Pandas kütüphanesinin Concat() fonksiyonundan yararlanıldı.



**Şekil 1**. Dataframe

## 2.2 Veri Ön İşleme



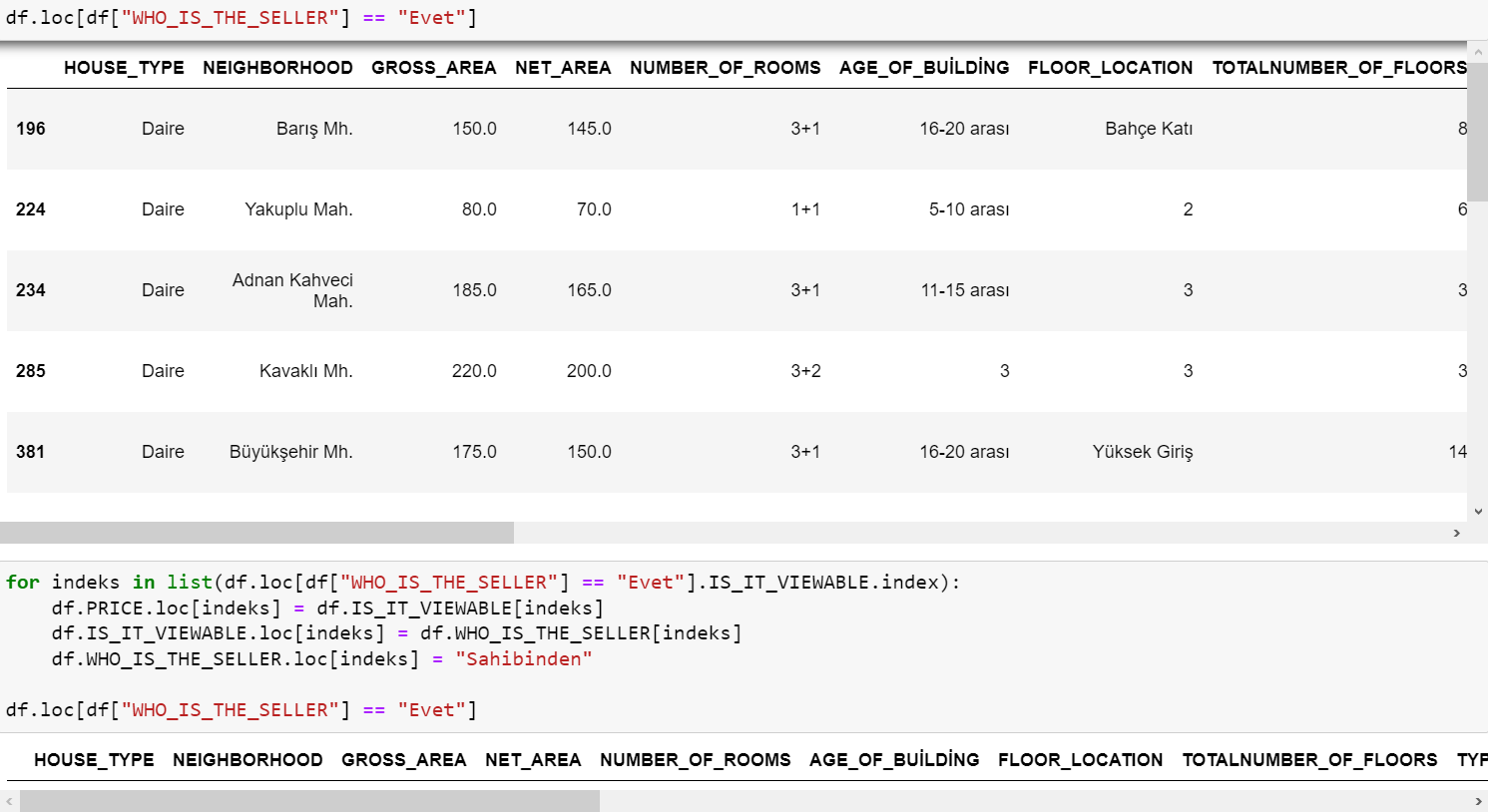
**Şekil 2.** WHO\_IS\_THE\_SELLER Sütun Değeri Hayır Olan PRICE Değişkenindeki NaN Değerleri Doldurma

Bu kod parçacığı, "WHO\_IS\_THE\_SELLER" sütunu değeri "Hayır" olan satırları döngüyle işlemektedir. Her döngü adımında şu işlemler gerçekleştirilir:

df.PRICE.loc[indeks] = df.IS\_IT\_VIEWABLE[indeks]: "IS\_IT\_VIEWABLE" sütunundaki değeri alır ve bunu "PRICE" sütunundaki ilgili satıra atar. Yani, "IS\_IT\_VIEWABLE" sütunundaki değerleri "PRICE" sütununa kopyalar.

df.IS\_IT\_VIEWABLE.loc[indeks]=df.WHO\_IS\_THE\_SELLER[indeks]: "WHO\_IS\_THE\_SELLER" sütunundaki değeri alır ve bunu "IS\_IT\_VIEWABLE" sütunundaki ilgili satıra atar. Yani, "WHO\_IS\_THE\_SELLER" sütunundaki değerleri "IS\_IT\_VIEWABLE" sütununa kopyalar.

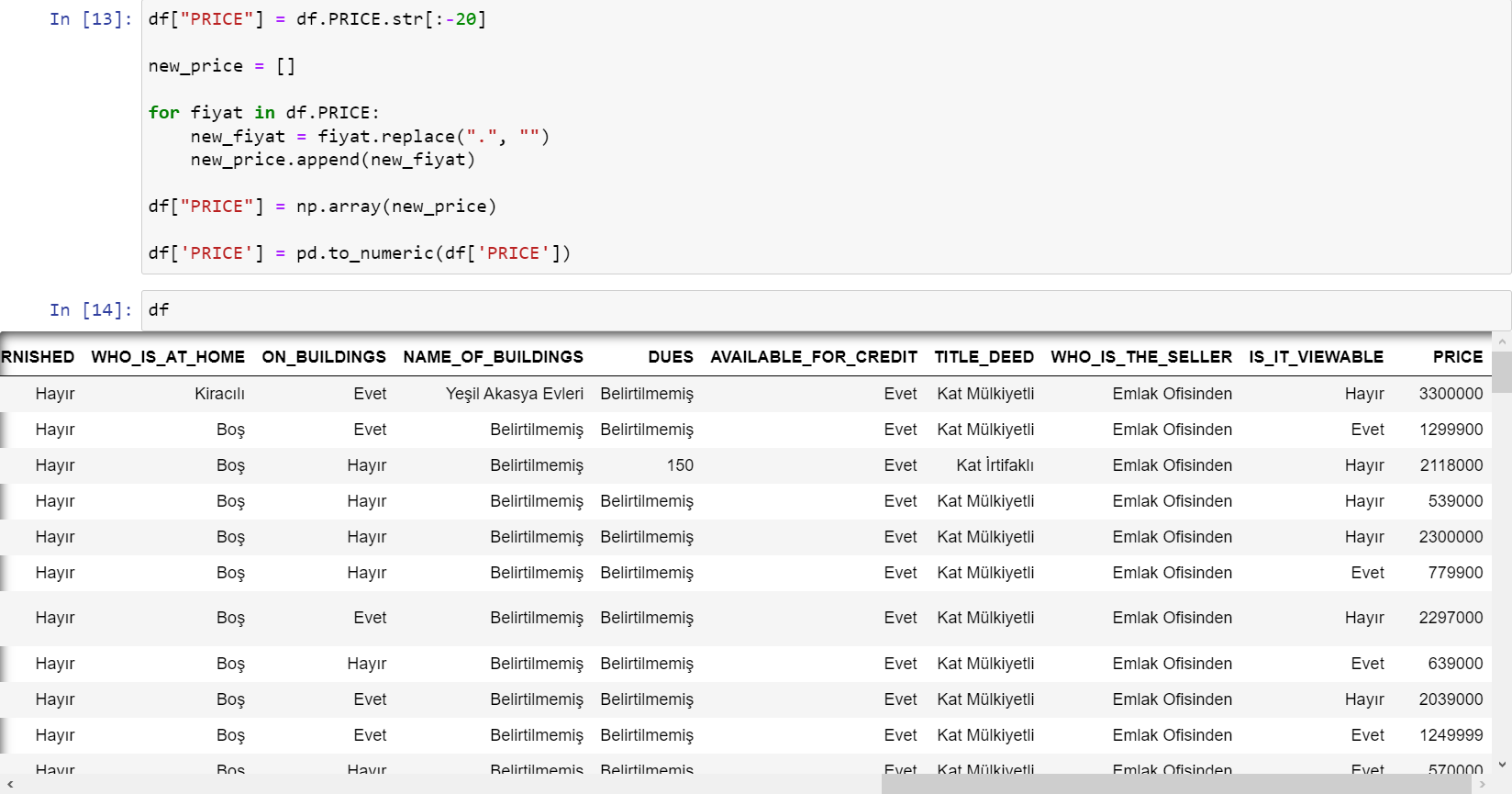
df.WHO\_IS\_THE\_SELLER.loc[indeks] = "Sahibinden": "WHO\_IS\_THE\_SELLER" sütunundaki değeri "Sahibinden" olarak günceller.



**Şekil 3**. WHO\_IS\_THE\_SELLER Sütun Değeri Evet Olan PRICE Değişkenindeki NaN Değerleri Doldurma

Bu kod parçacığı, "WHO\_IS\_THE\_SELLER" sütunu değeri "Evet" olan satırları döngüyle işlemektedir. Her döngü adımında şu işlemler gerçekleştirilir:  
df.PRICE.loc[indeks] = df.IS\_IT\_VIEWABLE[indeks]: "IS\_IT\_VIEWABLE" sütunundaki değeri alır ve bunu "PRICE" sütunundaki ilgili satıra atar. Yani, "IS\_IT\_VIEWABLE" sütunundaki değerleri "PRICE" sütununa kopyalar.  
df.IS\_IT\_VIEWABLE.loc[indeks]=df.WHO\_IS\_THE\_SELLER[indeks]: "WHO\_IS\_THE\_SELLER" sütunundaki değeri alır ve bunu "IS\_IT\_VIEWABLE" sütunundaki ilgili satıra atar.

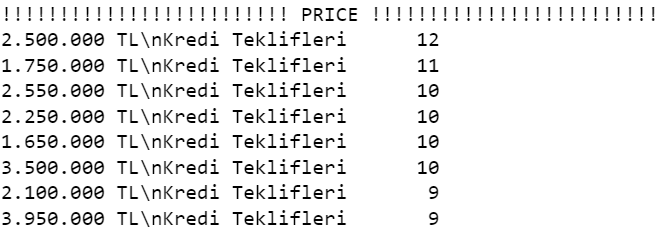
df.WHO\_IS\_THE\_SELLER.loc[indeks] = "Sahibinden": "WHO\_IS\_THE\_SELLER" sütunundaki ilgili satıra "Sahibinden" değerini atar. Yani, "WHO\_IS\_THE\_SELLER" sütunundaki değeri "Sahibinden" olarak günceller.



**Şekil 4.** PRICE Değişkeni Değerlerinin Düzeltilmesi

Bu kod parçacığı bir dizi işlem gerçekleştirerek "PRICE" sütunundaki değerleri düzenlemektedir. İşlemlerin amacı ise şu şekildedir:

df["PRICE"] = df.PRICE.str[:-20]: "PRICE" sütunundaki değerlerin son 20 karakterini keser. Bu, PRICE değişkeni bir integer değer olduğu için TL\nKredi Teklifleri gibi string ifadeyi kaldırmaktır.



**Şekil 5.** PRICE Değişkeni Değerlerinin Düzeltilmeden Önce Değerleri

new\_price = []:

Boş bir liste oluşturulur. Bu liste, düzenlenmiş fiyat değerlerini tutmak için kullanılacak.  
for fiyat in df.PRICE:

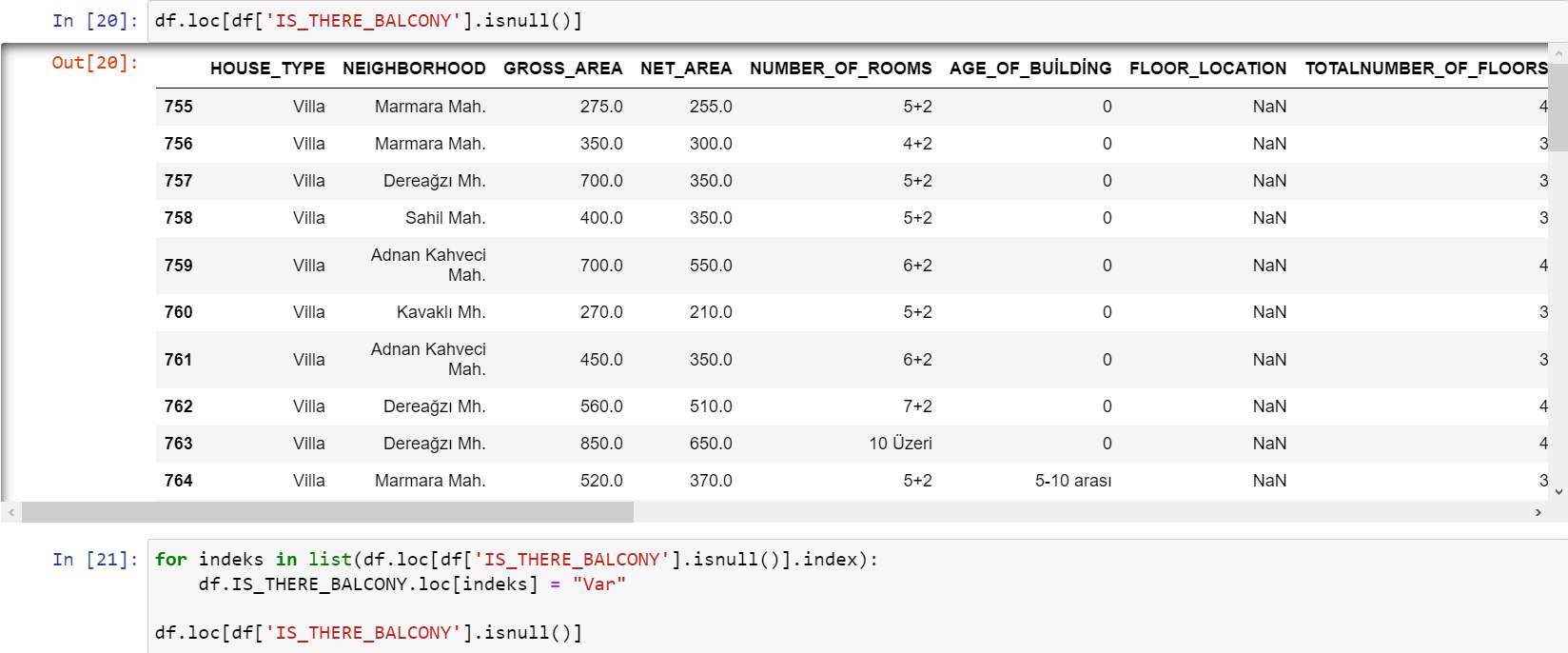
"PRICE" sütunundaki her bir fiyat değeri için bir döngü başlatılır.

new\_fiyat = fiyat.replace(".", ""):

Her fiyat değerindeki nokta karakterlerini (varsayılan olarak) boş bir dizeyle değiştirir.   
new\_price.append(new\_fiyat): Düzenlenmiş fiyat değerini new\_price listesine ekler.  
df["PRICE"] = np.array(new\_price):

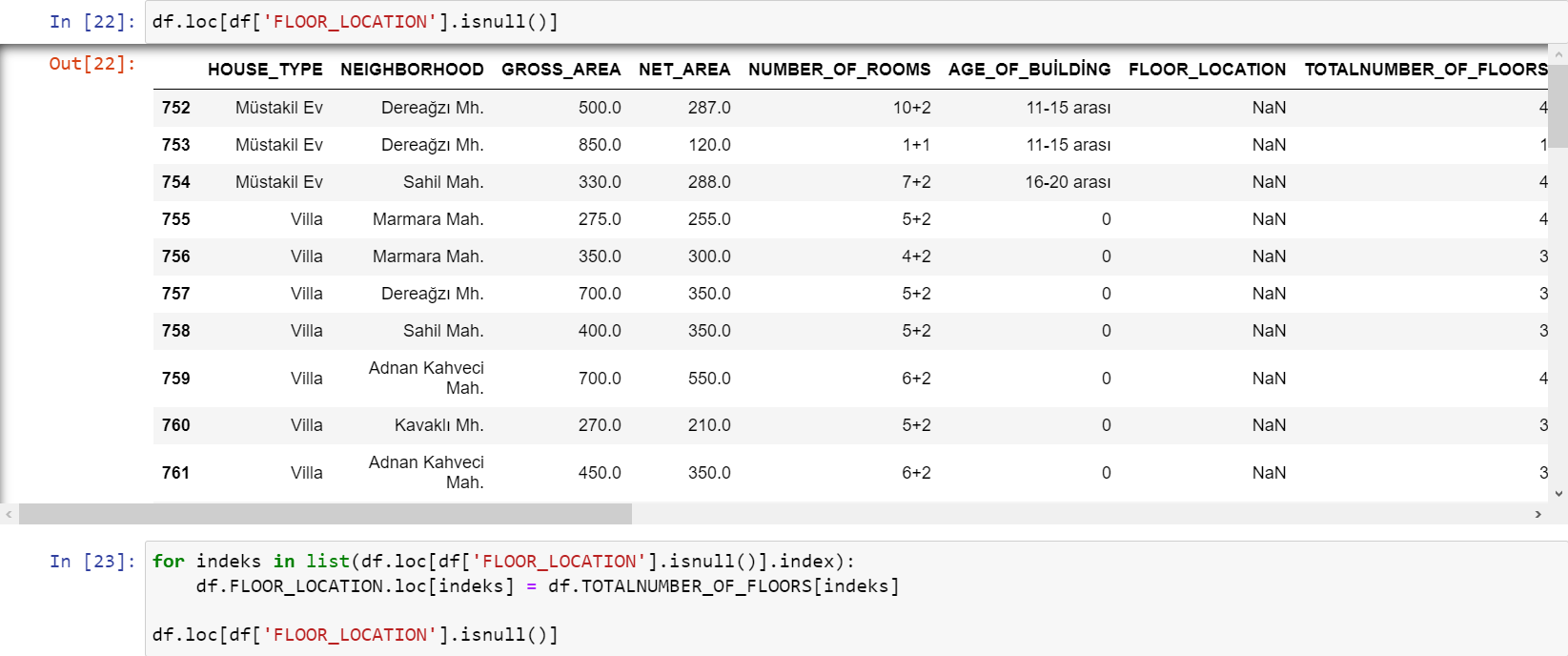
new\_price listesini numpy dizisine dönüştürerek "PRICE" sütununu günceller.  
df['PRICE'] = pd.to\_numeric(df['PRICE']):

"PRICE" sütununu sayısal veri tipine dönüştürür. Bu, fiyat değerlerini sayısal olarak işlemek ve matematiksel operasyonlar yapabilmek için gereklidir



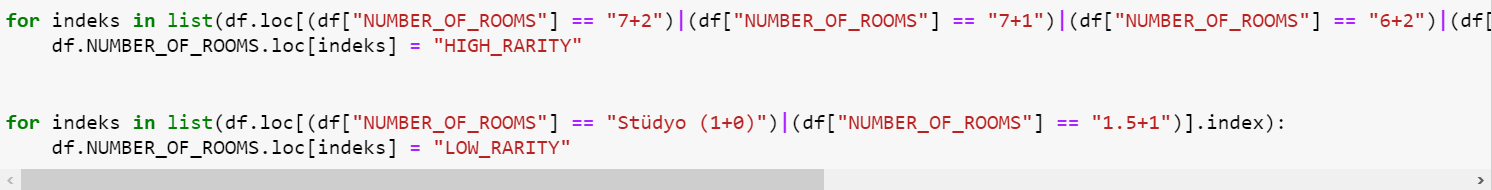
**Şekil 6.** IS\_THERE\_BALCONY sütunu değeri null olan satırların “Var” olarak güncellenmesi

Bu kod parçacığı, "IS\_THERE\_BALCONY" sütunu değeri null (NaN) olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise HOUSE\_TYPE değeri Villa olan evlerin IS\_THERE\_BALCONY değerinin Var olarak güncellenmesi istenilmiştir.



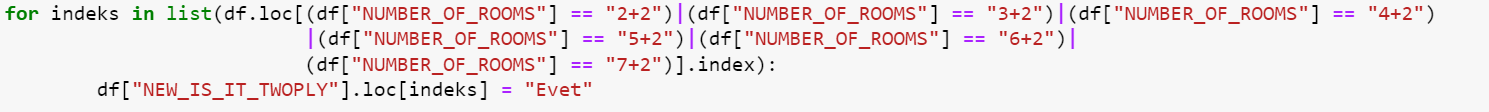
**Şekil 7.** FLOOR\_LOCATION sütunu değeri null olan satırların koşula göre güncellenmesi

Bu kod parçacığı, "FLOOR\_LOCATION" sütunu değeri null (NaN) olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise HOUSE\_TYPE değeri Villa olan evlerin TOTAL\_NUMBER\_OF\_FLOORS değerinin FLOOR\_LOCATION ile değerine kopyalanmasıdır.



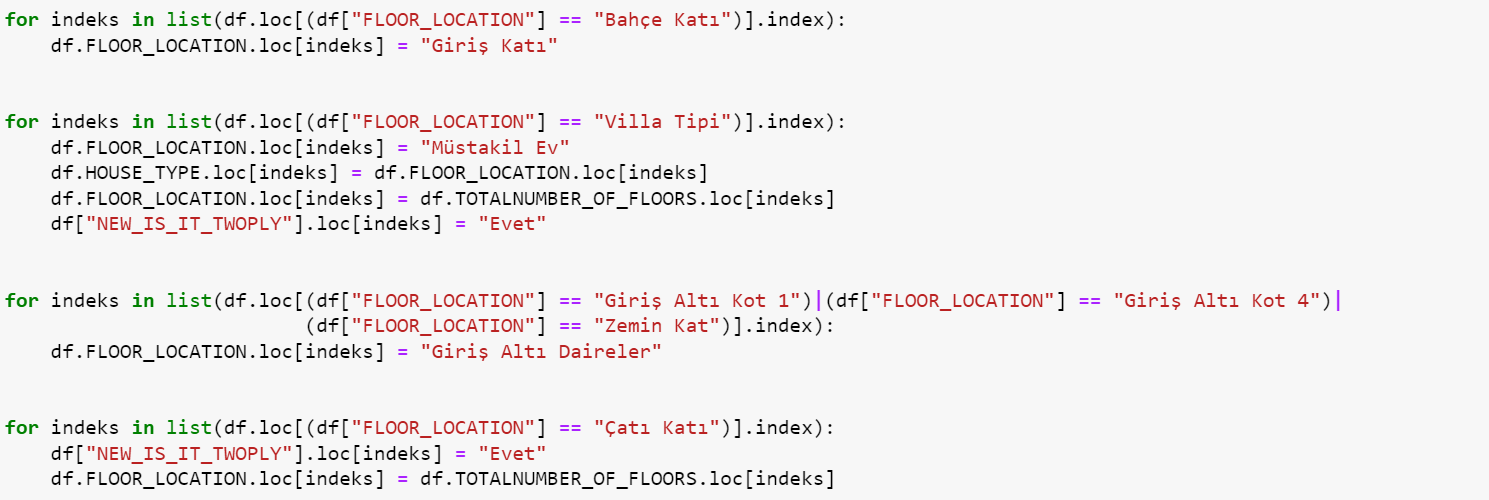
**Şekil 8.** NUMBER\_OF\_ROOMS sütunu değerinin koşula göre güncellenmesi

Bu iki farklı kod parçacığı, "NUMBER\_OF\_ROOMS" sütunu değerleri “7+1”, “6+2”, “6+1”, “5+2”, “5+1” ve “Stüdyo (1+0)”, “1.5+1” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise NUMBER\_OF\_ROOMS değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “HIGH\_RARITY” ve “LOW\_RARITY” değerleriyle güncellendi.



**Şekil 9.** Koşula göre NEW\_IS\_IT\_TWOPLY sütunu oluşturulması

Bu kod parçacığı, "NUMBER\_OF\_ROOMS" sütunu değeri “2+2”, “3+2”, “4+2”, “5+2”, “6+2”, “7+2” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise NUMBER\_OF\_ROOMS değeri “2+2”, “3+2”, “4+2”, “5+2”, “6+2”, “7+2” olan satırların yeni bir sütun olarak “NEW\_IS\_IT\_TWOPLY” değerinin “Evet” olarak güncellenmesi istenilmiştir.



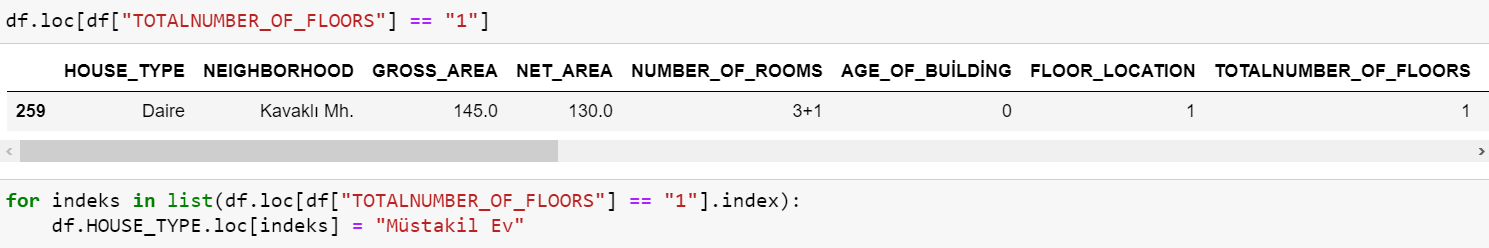
**Şekil 10.** FLOOR\_LOCATION sütunu değerlerinin koşula göre güncellenmesi

Bu dört farklı kod parçacığından ilk sıradaki for döngüsü, "FLOOR\_LOCATION" sütunu değeri “Bahçe Katı” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "FLOOR\_LOCATION" değeri “Bahçe Katı” olan değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine FLOOR\_LOCATION değerinin “Giriş Katı” olarak güncellenmesi istenilmiştir.

Bu dört farklı kod parçacığından ikinci sıradaki for döngüsü, "FLOOR\_LOCATION" sütunu değeri “Villa Tipi” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "FLOOR\_LOCATION" değeri “Villa Tipi” olan değerin, HOUSE\_TYPE değeri “Müstakil Ev” TOTALNUMBER\_OF\_FLOORS değeri “FLOOR\_LOCATION” değeriyle güncellenmiş ve “NEW\_IS\_IT\_TWOPLY” değeri “Evet” olarak atanmıştır.

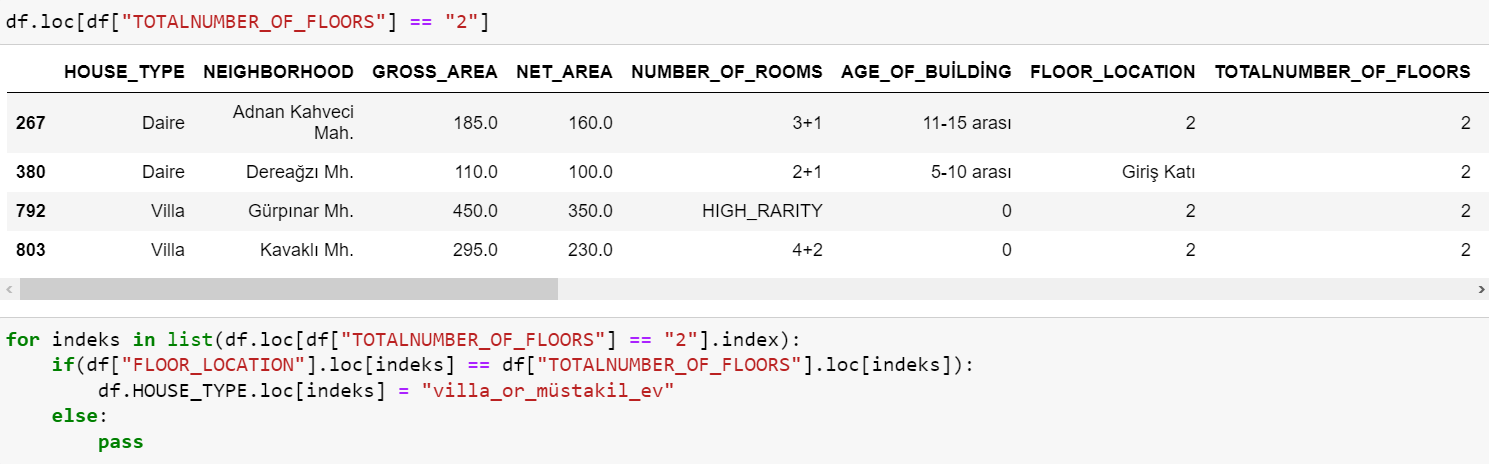
Bu üç farklı kod parçacığından üçüncü sıradaki for döngüsü, "FLOOR\_LOCATION" sütunu değeri “Giriş Altı Kot 1”, “Giriş Altı Kot 4”, “Zemin Kat” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "FLOOR\_LOCATION" değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “Giriş Altı Daireler” değerleriyle güncellenmesi istenilmiştir.

Bu dört farklı kod parçacığından dördüncü sıradaki for döngüsü, "FLOOR\_LOCATION" sütunu değeri “Çatı Katı” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "FLOOR\_LOCATION" değeri “Çatı Katı” olan satırların “NEW\_IS\_IT\_TWOPLY” değerini “Evet” ve “TOTALNUMBER\_OF\_FLOORS” değerini “FLOOR\_LOCATION” değeri olarak güncellemesini sağlar.



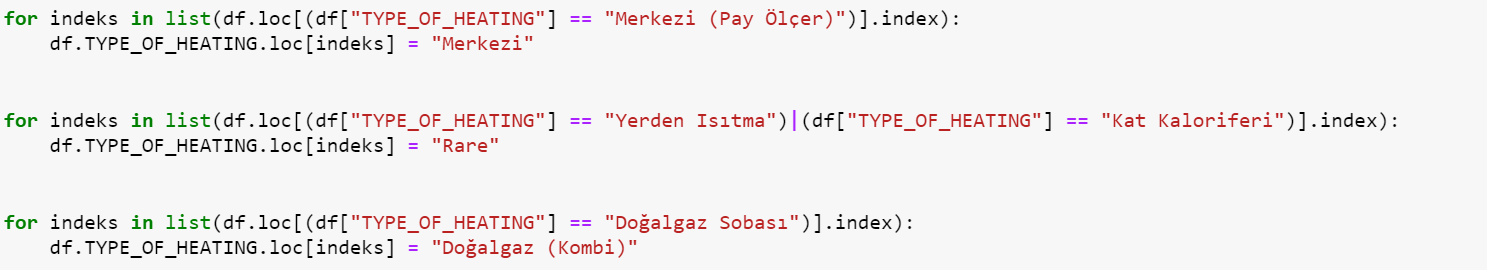
**Şekil 11.** TOTAL\_NUMBER\_OF\_FLOORS sütunu değerinin koşuluna göre HOUSE\_TYPE değerinin güncellenmesi

Bu kod parçacığı, "TOTALNUMBER\_OF\_FLOORS" sütunu değeri “1” olan satırları tespit ederek, "TOTALNUMBER\_OF\_FLOORS" değeri “1” olan satırların HOUSE\_TYPE değeri “Müstakil Ev” olarak güncellenmesi istenilmiştir.



**Şekil 12.** TOTAL\_NUMBER\_OF\_FLOORS sütunu değerinin koşuluna göre HOUSE\_TYPE değerinin güncellenmesi

Bu kod parçacığı, "TOTALNUMBER\_OF\_FLOORS" sütunu değeri “2” olan satırları tespit ederek, "TOTALNUMBER\_OF\_FLOORS" değeri “2” olan satırların HOUSE\_TYPE değeri “villa\_or\_müstakil\_ev” olarak güncellenmesi istenilmiştir.

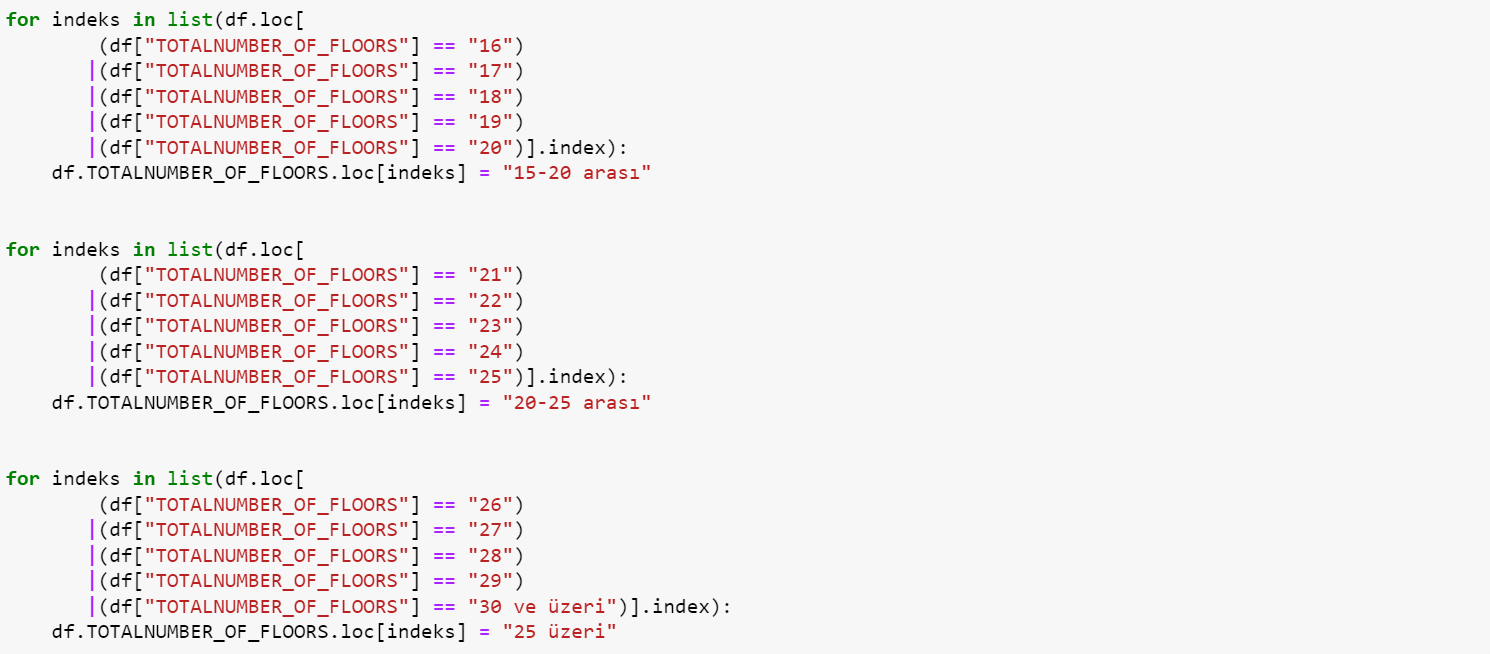


***Şekil 13.*** TYPE\_OF\_HEATING sütunu değerinin koşula göre güncellenmesi

Bu üç farklı kod parçacığından ilk sıradaki for döngüsü, "TYPE\_OF\_HEATING" sütunu değeri “Merkezi (Pay Ölçer)” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TYPE\_OF\_HEATING" değeri “Merkezi (Pay Ölçer)” olan satırların TYPE\_OF\_HEATING değeri “Merkezi” olarak güncellenmesi istenilmiştir.

Bu üç farklı kod parçacığından ilk sıradaki for döngüsü, "TYPE\_OF\_HEATING" sütunu değeri “Yerden Isıtma”, “Kat Kaloriferi” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TYPE\_OF\_HEATING" değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “Rare” değerleriyle güncellenmesi istenilmiştir.

Bu üç farklı kod parçacığından ilk sıradaki for döngüsü, "TYPE\_OF\_HEATING" sütunu değeri “Doğalgaz Sobası” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TYPE\_OF\_HEATING" değeri “Doğalgaz Sobası” olan satırların TYPE\_OF\_HEATING değeri “Doğalgaz (Kombi)” olarak güncellenmesi istenilmiştir.



**Şekil 14.**TOTALNUMBER\_OF\_FLOORS sütunu değerinin koşula göre güncellenmesi

Bu üç farklı kod parçacığından ilk sıradaki for döngüsü, "TOTALNUMBER\_OF\_FLOORS" sütunu değeri “16”, “17”, “18”, “19”, “20” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TOTALNUMBER\_OF\_FLOORS" değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “15-20 arası değerleriyle güncellenmesi istenilmiştir.

Bu üç farklı kod parçacığından ikinci sıradaki for döngüsü, "TOTALNUMBER\_OF\_FLOORS" sütunu değeri “21”, “22”, “23”, “24”, “25” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TOTALNUMBER\_OF\_FLOORS" değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “20-25 arası” değerleriyle güncellenmesi istenilmiştir.

Bu üç farklı kod parçacığından üçüncü sıradaki for döngüsü, "TOTALNUMBER\_OF\_FLOORS" sütunu değeri “26”, “27”, “28”, “29”, “30 ve üzeri” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TOTALNUMBER\_OF\_FLOORS" değeri bu değerlere sahip olanların toplam veriye bakıldığında az olmasından dolayı, bu verilere sahip satırları kaldırmak yerine “25 üzeri” arası değerleriyle güncellenmesi istenilmiştir.



**Şekil 15.** IS\_IT\_FURNISHED sütunu değerinin koşula göre güncellenmesi

Bu kod parçacığı, "IS\_IT\_FURNISHED" sütunu değeri “Belirtilmemiş” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "IS\_IT\_FURNISHED" değerini modelin performansı için daha yüksek frekansa sahip olan “Hayır” değeri olarak güncellenmesi istenilmiştir.



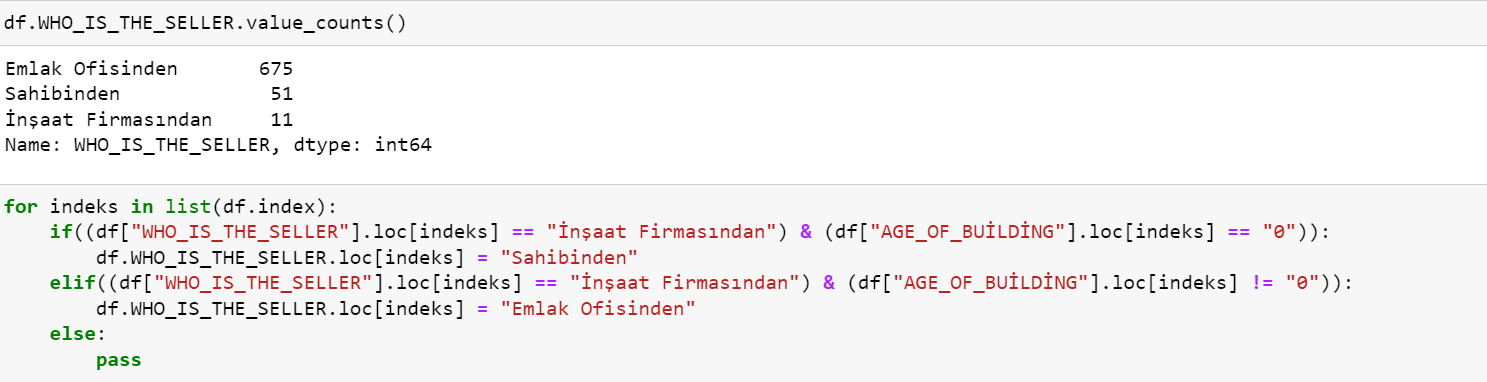
**Şekil 16.** WHO\_IS\_AT\_HOME sütunu değeri “Kiracılı” olan sütunların değerlerinin güncellenmesi

Bu kod parçacığı, "WHO\_IS\_AT\_HOME" sütunu değeri “Kiracılı” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "WHO\_IS\_AT\_HOME" değeri “Kiracılı” olan satırların değeri “Mülk Sahibi” olarak güncellenmesi istenilmiştir.



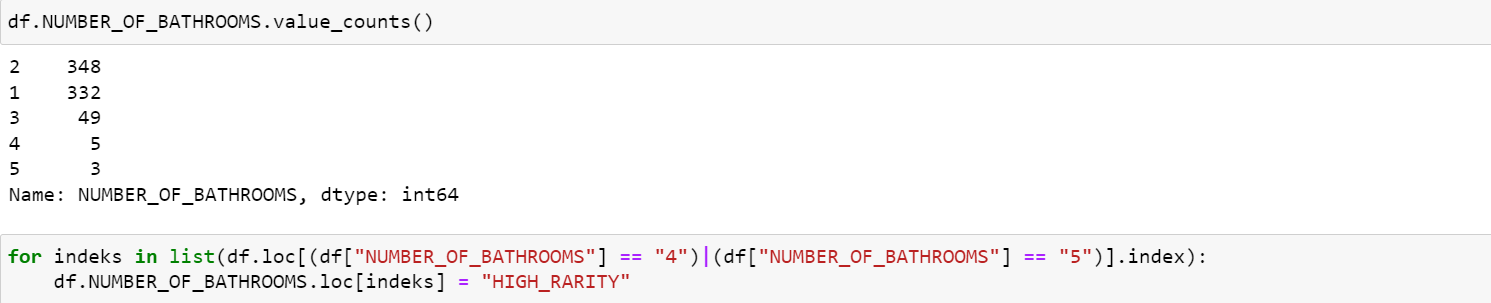
**Şekil 17.** TITLE\_DEED sütunu değeri “Bilinmiyor” olan sütunların değerlerinin güncellenmesi

Bu kod parçacığı, "TITLE\_DEED" sütunu değeri “Bilinmiyor” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "TITLE\_DEED" değeri modelin performansı için daha yüksek frekansa sahip olan “Kat Mülkiyetli” değeri olarak güncellenmesi istenilmiştir.



**Şekil 18.** WHO\_IS\_THE\_SELLER sütunu değerlerinin koşula göre güncellenmesi

Bu kod parçacığı, "WHO\_IS\_THE\_SELLER" sütunu değeri “İnşaat Firmasından” olan ve “AGE\_OF\_BUILDING” sütunu değeri “0” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "WHO\_IS\_THE\_SELLER" sütunu değeri “İnşaat Firmasından” olan ve “AGE\_OF\_BUILDING” sütunu değeri “0” olan satırların "WHO\_IS\_THE\_SELLER" değeri “Sahibinden” olarak, "WHO\_IS\_THE\_SELLER" sütunu değeri “İnşaat Firmasından” olan ve “AGE\_OF\_BUILDING” sütunu değeri “0” olmayan değeri “Emlak Ofisinden” yapılan araştırmalar sonucunda güncellenmiştir.



**Şekil 19.** NUMBER\_OF\_BATHROOMS sütununun değerlerinin belirli koşula göre güncellenmesi

Bu kod parçacığı, "NUMBER\_OF\_BATHROOMS" sütunu değeri “4” veya “5” olan satırları tespit ederek işlemler gerçekleştirir. İşlemin amacı ise "NUMBER\_OF\_BATHROOMS" değeri modelin performansını yükseltmek için frekans sıklığı az olduğundan “HIGH\_RARITY” şeklinde güncellenmiştir.



**Şekil 20.** DUES sütunundaki değerlerin belirli koşullara göre güncellenmesi

Bu kod parçası, belirli koşulları sağlayan satırlarda "DUES" sütunundaki "Belirtilmemiş" değerlerini başka değerlerle değiştirmek için kullanılır. Ayrıca, "DUES" sütunundaki değerlerin sayısal tipe dönüştürülmesini sağlar.  
İlk olarak, without\_belirtilmemis\_indeks() adlı bir fonksiyon tanımlanır. Bu fonksiyon, "DUES" sütununda "Belirtilmemiş" değerine sahip satırların indekslerini black\_list adlı bir liste olarak alır. Ardından, without\_belirtilmemis\_indeks\_list adlı boş bir liste oluşturulur.Daha sonra, bir döngü başlatılır ve df DataFrame'inin indekslerini tek tek gezerek işlem yapılır. Her döngü adımında, if koşuluyla kontrol edilir. Eğer satırın "DUES" sütunundaki değeri "Belirtilmemiş" ise ve "ON\_BUILDINGS" sütunundaki değeri "Evet" ise, df.loc[without\_belirtilmemis].groupby("ON\_BUILDINGS")["DUES"].median()["Evet"] ifadesi kullanılarak "Evet" değerine karşılık gelen "DUES" değeri median (ortanca) değeriyle değiştirilir. Aynı şekilde, "ON\_BUILDINGS" sütunundaki değeri "Hayır" ise, "DUES" değeri median değeriyle değiştirilir.Son olarak, pd.to\_numeric() fonksiyonuyla "DUES" sütunundaki değerlerin sayısal tipe dönüştürülmesi sağlanır.

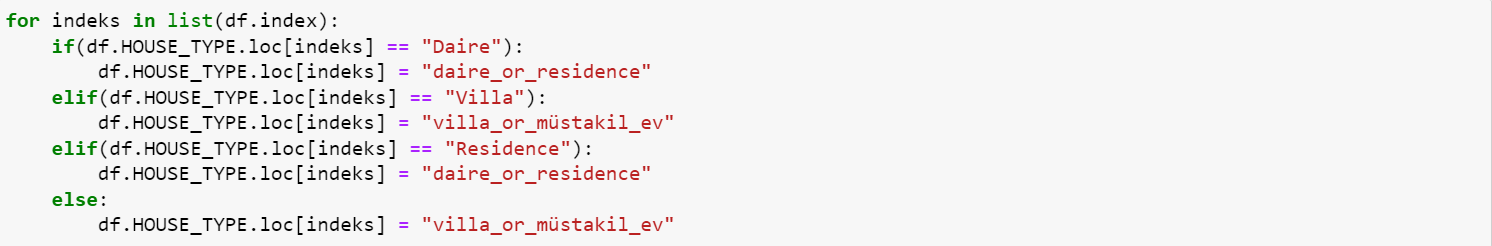


**Şekil 21.** DUES sütununda değerlerin koşula göre eksik ve hatalı değerlerin güncellenmesi

Bu kod parçası, "DUES" sütununda belirli koşulları sağlayan değerleri değiştirerek veri setindeki eksik veya hatalı değerleri düzeltmeyi amaçlar. Koşulların tam olarak neye göre belirlendiği ve hangi değerlerin atandığına bağlı olarak, veri setinin spesifik gereksinimlerine göre özelleştirilmiş bir veri düzeltme işlemi gerçekleştirir.  
İlk olarak, bir döngü başlatılır ve df DataFrame'inin indeksleri tek tek gezilir.

Eğer satırın "NAME\_OF\_BUILDINGS" sütunundaki değeri "Belirtilmemiş" ve "DUES" sütunundaki değeri 0 ise, "DUES" değeri ilgili "NEIGHBORHOOD" ve "ON\_BUILDINGS" değerlerine göre gruplanarak median (ortanca) değeri ile değiştirilir.  
Eğer satırın "NAME\_OF\_BUILDINGS" sütunundaki değeri "Belirtilmemiş" değil ve "DUES" sütunundaki değeri 0 ise, "DUES" değeri daha önce aynı "NAME\_OF\_BUILDINGS" değerine sahip diğer satırların "DUES" değerleriyle karşılaştırılır. Eğer farklı bir değer bulunursa, "DUES" değeri bu değerle değiştirilir. Aksi takdirde, "DUES" değeri ilgili "NEIGHBORHOOD" ve "ON\_BUILDINGS" değerlerine göre gruplanarak median değeri ile değiştirilir.  
Eğer satırın "NAME\_OF\_BUILDINGS" sütunundaki değeri "Belirtilmemiş" ve "DUES" sütunundaki değeri 0 değilse, "DUES" değeri ilgili "NEIGHBORHOOD" ve "ON\_BUILDINGS" değerlerine göre gruplanarak median değeri ile değiştirilir.  
Yukarıdaki koşulların hiçbiri sağlanmazsa, belirli indeksler için özel durumlar kontrol edilir. İndeks 336 için "DUES" değeri 1250 olarak değiştirilir, indeks 512 için 255 olarak değiştirilir ve indeks 556 için "NEIGHBORHOOD", "NUMBER\_OF\_ROOMS" ve "ON\_BUILDINGS" değerlerine göre gruplanarak median değeri ile değiştirilir.

## 2.3 Algoritmanın Eğitilmesi



**Şekil 22.** HOUSE\_TYPE sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi

Bu kod parçası, DataFrame'deki "HOUSE\_TYPE" sütunundaki değerleri kontrol ederek yeni değerler atamak için kullanılır.

Her bir indeks için "HOUSE\_TYPE" sütunundaki değer kontrol edilir ve belirli koşullara göre yeni bir değer atanır:

Eğer "HOUSE\_TYPE" sütunundaki değer "Daire" ise, bu değer "daire\_or\_residence" olarak değiştirilir.

Eğer "HOUSE\_TYPE" sütunundaki değer "Villa" ise, bu değer "villa\_or\_müstakil\_ev" olarak değiştirilir.

Eğer "HOUSE\_TYPE" sütunundaki değer "Residence" ise, bu değer "daire\_or\_residence" olarak değiştirilir.

Yukarıdaki koşulların hiçbiri sağlanmazsa, "HOUSE\_TYPE" değeri varsayılan olarak "villa\_or\_müstakil\_ev" olarak atanır.

Bu kod parçası, "HOUSE\_TYPE" sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesini sağlar. Böylece, farklı ev türlerini belirli kategorilere dahil etmek ve veri setini daha homojen hale getirmek amaçlanır.



**Şekil 23.** NUMBER\_OF\_ROOMS sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi

Bu kod parçası, DataFrame'deki "NUMBER\_OF\_ROOMS" sütunundaki değerleri kontrol ederek yeni değerler atamak için kullanılır.

Her bir indeks için "NUMBER\_OF\_ROOMS" sütunundaki değer kontrol edilir ve belirli koşullara göre yeni bir değer atanır:

Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "LOW\_RARITY" ise, bu değer 1 olarak değiştirilir.

Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "1+1" ise, bu değer 2 olarak değiştirilir.

Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "2+1" ise, bu değer 3 olarak değiştirilir.

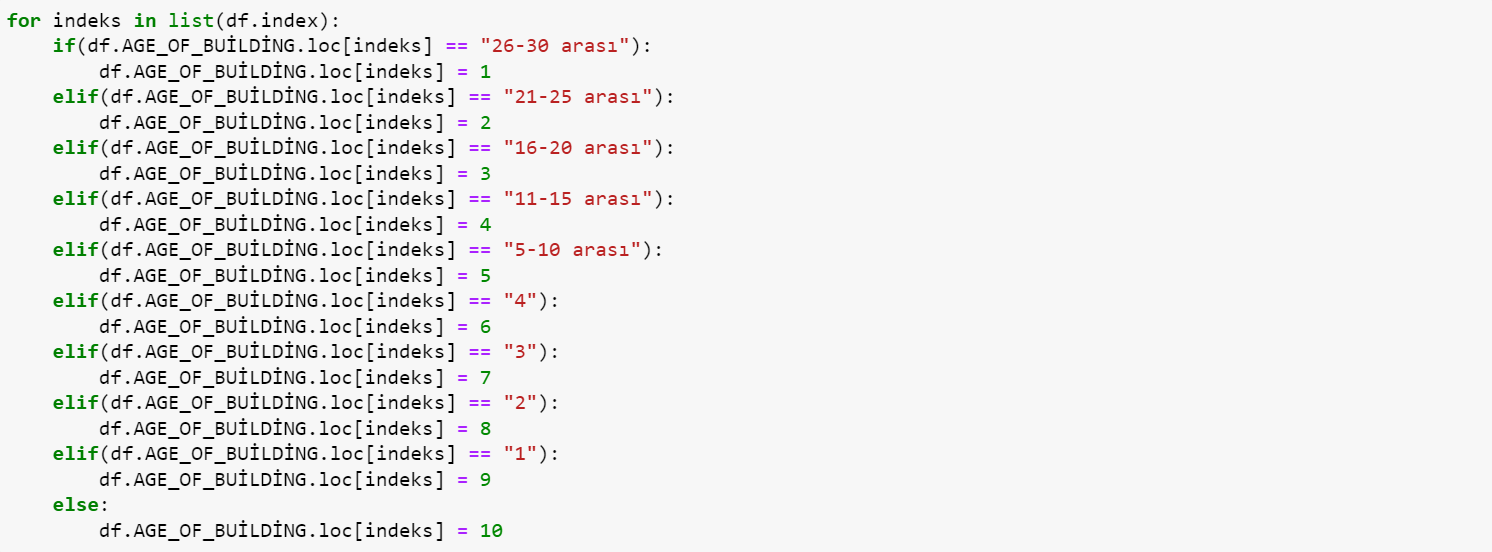
Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "2+2", "3+1" veya "3.5+1" ise, bu değer 4 olarak değiştirilir.

Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "3+2", "4+1" veya "4.5+1" ise, bu değer 5 olarak değiştirilir.

Eğer "NUMBER\_OF\_ROOMS" sütunundaki değer "4+2" ise, bu değer 6 olarak değiştirilir.

Yukarıdaki koşulların hiçbiri sağlanmazsa, "NUMBER\_OF\_ROOMS" değeri varsayılan olarak 7 olarak atanır.

Bu kod parçası, "NUMBER\_OF\_ROOMS" sütunundaki farklı değerleri belirli kategorilere atayarak veri setinin düzenlenmesini sağlar. Böylece, farklı oda sayılarını belirli kategorilere dahil etmek ve veri setini daha anlaşılır hale getirmek amaçlanır.



**Şekil 24.** AGE\_OF\_BUILDING sütunundaki farklı değerleri daha genel kategorilere atayarak veri setinin düzenlenmesi

Bu kod parçası, DataFrame'deki "AGE\_OF\_BUILDING" sütunundaki değerleri kontrol ederek yeni değerler atamak için kullanılır.

Her bir indeks için "AGE\_OF\_BUILDING" sütunundaki değer kontrol edilir ve belirli koşullara göre yeni bir değer atanır:

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "26-30 arası" ise, bu değer 1 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "21-25 arası" ise, bu değer 2 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "16-20 arası" ise, bu değer 3 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "11-15 arası" ise, bu değer 4 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "5-10 arası" ise, bu değer 5 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "4" ise, bu değer 6 olarak değiştirilir.

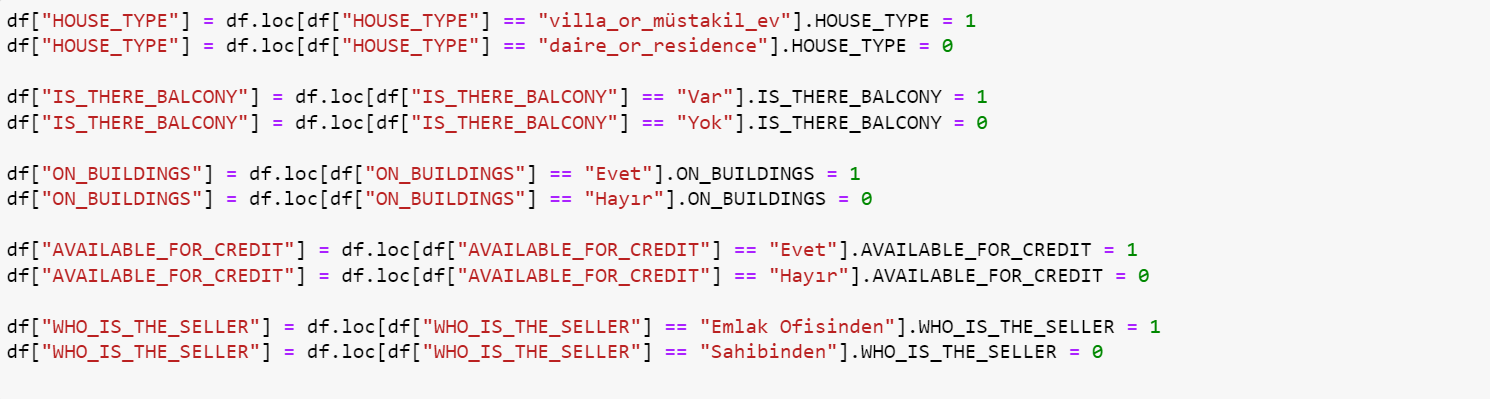
Eğer "AGE\_OF\_BUILDING" sütunundaki değer "3" ise, bu değer 7 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "2" ise, bu değer 8 olarak değiştirilir.

Eğer "AGE\_OF\_BUILDING" sütunundaki değer "1" ise, bu değer 9 olarak değiştirilir.

Yukarıdaki koşulların hiçbiri sağlanmazsa, "AGE\_OF\_BUILDING" değeri varsayılan olarak 10 olarak atanır.

Bu kod parçası, "AGE\_OF\_BUILDING" sütunundaki farklı değerleri belirli kategorilere atayarak veri setinin düzenlenmesini sağlar. Böylece, binaların yaşlarını belirli kategorilere dahil etmek ve veri setini daha anlaşılır hale getirmek amaçlanır.



**Şekil 25.** Binary sütunların sayısal değerlere dönüştürülmesi

Bu işlemler, kategorik verilerin sayısal değerlere dönüştürülmesini sağlar. Böylece, veri setindeki kategorik özelliklerin sayısal değerlerle temsil edilmesi ve bazı analiz ve modelleme işlemlerinin gerçekleştirilmesi kolaylaşır.

Her bir sütun için aşağıdaki işlemler gerçekleştirilir:

"HOUSE\_TYPE" sütunu için:

Eğer "HOUSE\_TYPE" sütunundaki değer "villa\_or\_müstakil\_ev" ise, bu değer 1 olarak değiştirilir.

Eğer "HOUSE\_TYPE" sütunundaki değer "daire\_or\_residence" ise, bu değer 0 olarak değiştirilir.

"IS\_THERE\_BALCONY" sütunu için:

Eğer "IS\_THERE\_BALCONY" sütunundaki değer "Var" ise, bu değer 1 olarak değiştirilir.

Eğer "IS\_THERE\_BALCONY" sütunundaki değer "Yok" ise, bu değer 0 olarak değiştirilir.

"ON\_BUILDINGS" sütunu için:

Eğer "ON\_BUILDINGS" sütunundaki değer "Evet" ise, bu değer 1 olarak değiştirilir.

Eğer "ON\_BUILDINGS" sütunundaki değer "Hayır" ise, bu değer 0 olarak değiştirilir.

"AVAILABLE\_FOR\_CREDIT" sütunu için:

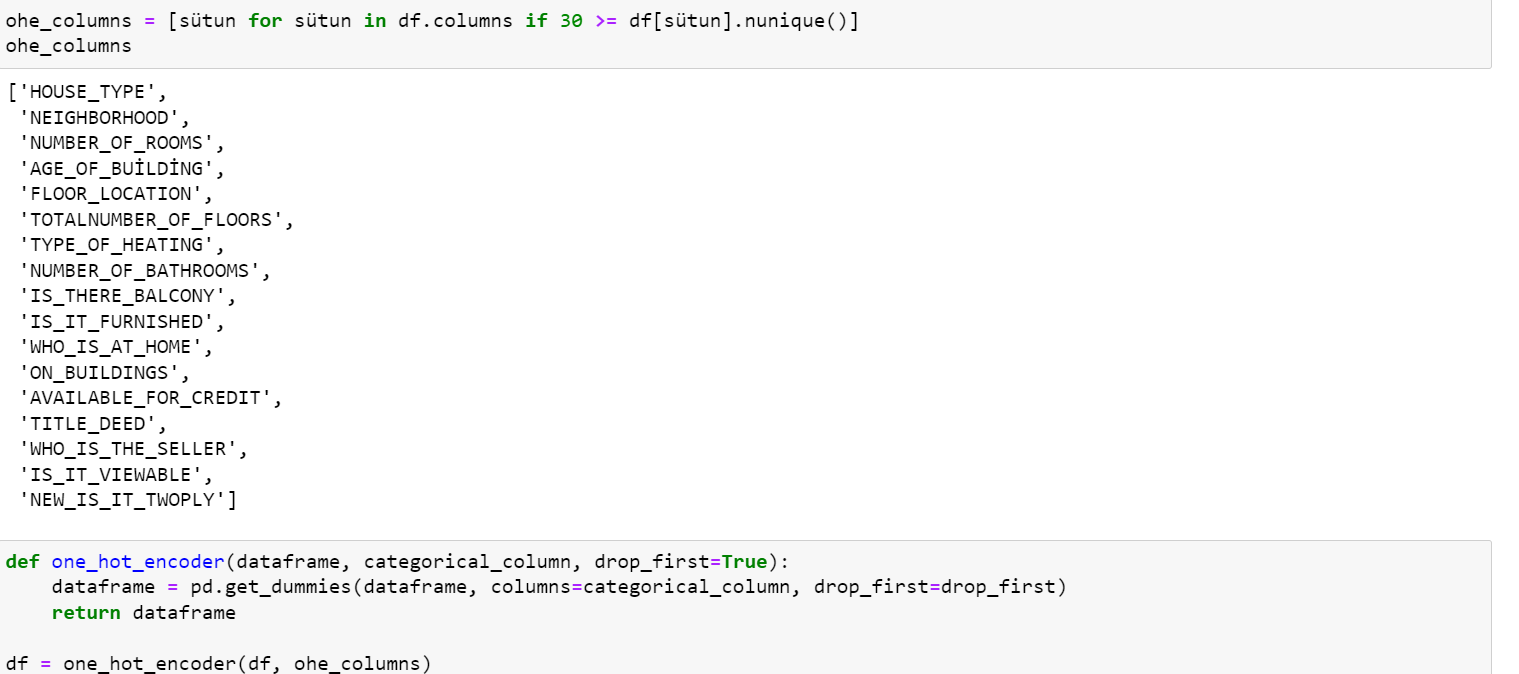
Eğer "AVAILABLE\_FOR\_CREDIT" sütunundaki değer "Evet" ise, bu değer 1 olarak değiştirilir.

Eğer "AVAILABLE\_FOR\_CREDIT" sütunundaki değer "Hayır" ise, bu değer 0 olarak değiştirilir.

"WHO\_IS\_THE\_SELLER" sütunu için:

Eğer "WHO\_IS\_THE\_SELLER" sütunundaki değer "Emlak Ofisinden" ise, bu değer 1 olarak değiştirilir.

Eğer "WHO\_IS\_THE\_SELLER" sütunundaki değer "Sahibinden" ise, bu değer 0 olarak değiştirilir.

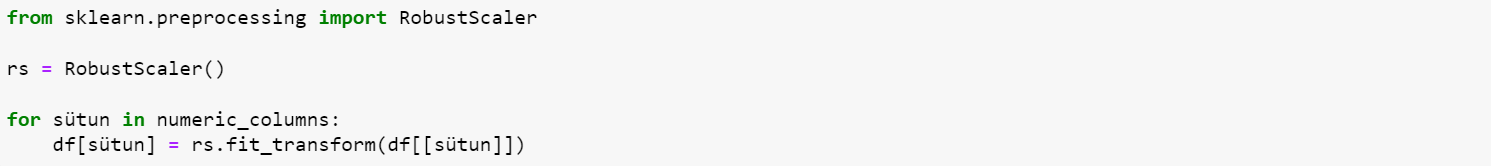


**Şekil 26.** Dataframedeki sütunların koşula göre One-Hot Encoding yöntemiyle dönüştürülmesi

Bu kod parçası, DataFrame'deki belirli sütunları One-Hot Encoding yöntemiyle dönüştürmek için kullanılır.

İlk olarak, ohe\_columns adlı bir liste oluşturulur. Bu liste, DataFrame'deki sütunları içerir, ancak sadece bu sütunlardaki benzersiz değer sayısı 30 veya daha az olan sütunlar seçilir.Daha sonra, one\_hot\_encoder adlı bir fonksiyon tanımlanır. Bu fonksiyon, verilen bir DataFrame, kategorik bir sütun ve bir "drop\_first" parametresi alır. Fonksiyon, pandas'ın get\_dummies fonksiyonunu kullanarak One-Hot Encoding işlemini gerçekleştirir. get\_dummies fonksiyonu, belirtilen kategorik sütunu dummy değişkenlere dönüştürür. "drop\_first" parametresi varsayılan olarak True olarak ayarlanmıştır, bu da ilk dummy değişkenin düşürülmesini sağlar. Son olarak, df DataFrame'i one\_hot\_encoder fonksiyonu kullanılarak One-Hot Encoding'e tabi tutulur ve dönüştürülmüş DataFrame df değişkenine atanır.

Bu işlem, kategorik sütunlardaki kategorik değerleri dummy değişkenlere dönüştürerek, her bir kategorinin ayrı bir sütunda temsil edilmesini sağlar. Bu, makine öğrenimi algoritmaları için gerekli olan sayısal veri formatına dönüşümü gerçekleştirir.



**Şekil 27.** Dataframe bulunan sayısal sütunları “RobustScaler” ile Normalizasyon işlemi

Bu kod parçası, DataFrame'deki sayısal sütunları RobustScaler kullanarak ölçeklendirir.

Öncelikle, sklearn.preprocessing modülünden RobustScaler sınıfı import edilir.Daha sonra, bir RobustScaler nesnesi rs oluşturulur.Sonraki adımda, bir döngü kullanılarak numeric\_columns listesindeki her bir sütun üzerinde dönülür. Bu sütunlar, DataFrame'in sayısal değerler içeren sütunlarını temsil eder.Her bir sütunun değerleri, rs.fit\_transform() yöntemi kullanılarak RobustScaler ile ölçeklendirilir. Bu yöntem, sütunun değerlerini RobustScaler'a uygun şekilde dönüştürür. fit\_transform() yöntemi, önce sütunun istatistiksel özelliklerini (ör. medyan ve çeyreklikler) hesaplar ve ardından bu özelliklere dayanarak sütunun değerlerini ölçeklendirir.Sonuç olarak, DataFrame'deki sayısal sütunların değerleri RobustScaler ile ölçeklendirilir ve güncellenmiş değerler DataFrame'e atanır. RobustScaler, verilerdeki aykırı değerlere karşı dayanıklıdır ve medyan ve çeyreklikleri kullanarak ölçeklendirme yapar, bu nedenle veri kümesindeki değerlerin dağılımını koruyarak ölçeklendirme yapar. Bu, makine öğrenimi modellerinin daha iyi performans göstermesini sağlayabilir.



**Şekil 28.** Regresyon modellerini uygulama

Bu kod parçası, veri kümesindeki özelliklere dayalı olarak çeşitli regresyon modellerini uygulayan bir fonksiyon içerir.

İlk olarak, train\_test\_split fonksiyonu kullanılarak veri kümesi X ve hedef değişken y için eğitim ve test alt kümelerine ayrılır. Test alt kümesi, genel performansı değerlendirmek için kullanılır.

Döngü kullanılarak her bir model sırasıyla eğitilir ve test verileri üzerinde tahmin yapılır. Ardından, tahminler ile gerçek değerler arasındaki hata ölçüsü olan ortalama karesel hata (MSE) hesaplanır ve yazdırılır.

Ayrıca, bazı regresyon modelleri için özellik önem düzeyini görselleştirmek amacıyla özellik önem sıralamaları da çizdirilir.

Son olarak, eğitim ve test verileri üzerindeki gerçek ve tahmin edilen değerler çizdirilir.

Eğer regresyon parametresi False olarak belirlenirse, henüz sınıflandırma modelleriyle ilgili bir uyarı mesajı verilir.



**Şekil 29.** En iyi sonuç veren regresyon modellerinin hiperparametre optimizasyonu ile uygulanması

Bu kod parçası, veri kümesindeki özelliklere dayalı olarak çeşitli regresyon modellerini uygulayan ve hiperparametre optimizasyonu gerçekleştiren bir fonksiyon içerir. Ayrıca, model performansını değerlendirmek ve özellik önem düzeylerini görselleştirmek için çeşitli grafikler çizdirme seçeneği sunar.

İlk olarak, train\_test\_split fonksiyonu kullanılarak veri kümesi X ve hedef değişken y için eğitim ve test alt kümelerine ayrılır. Test alt kümesi, genel performansı değerlendirmek için kullanılır.

Daha sonra, çeşitli regresyon modelleri (GradientBoostingRegressor, XGBRegressor, CatBoostRegressor) ve bu modellerin hiperparametrelerinin olası değerleri tanımlanır.

Döngü kullanılarak her bir model için GridSearchCV kullanılarak hiperparametre optimizasyonu gerçekleştirilir. En iyi hiperparametre kombinasyonları seçilir ve final\_model oluşturulur.

Sonrasında, final\_model kullanılarak test verileri üzerinde tahmin yapılır ve ortalama karesel hata (MSE) hesaplanır.

Eğer show parametresi True olarak belirlenirse, model performansını değerlendirmek için gerçek ve tahmin edilen değerlerin çizdirilmesi ve özellik önem düzeylerinin görselleştirilmesi gerçekleştirilir. Gerçek ve tahmin edilen değerlerin çizdirildiği grafikler ve özellik önem düzeylerini gösteren grafikler çizdirilir.

Eğer regresyon parametresi False olarak belirlenirse, henüz sınıflandırma modelleriyle ilgili bir uyarı mesajı verilir.

## 2.4 Model Hiperparametre Ayarlaması

## 

**Şekil 30.** Catboost regresyon modelinin hiperparametre optimizasyonu ile uygulanması

Bu kod parçası, CatBoostRegressor modelini kullanarak GridSearchCV yöntemiyle hiperparametre ayarlama gerçekleştirir.

GridSearchCV, verilen parametre ızgarası üzerinde çapraz doğrulama (cross-validation) yaparak en iyi parametre kombinasyonunu bulmaya çalışır. Bu sayede modelin performansını artırmak için en uygun hiperparametrelerin seçilmesine yardımcı olur.

Önce CatBoostRegressor modeli tanımlanır ve ardından parametre ızgarası belirlenir. Parametre ızgarası, farklı değer kombinasyonlarını denemek için kullanılır. Bu örnekte, 'iterations', 'learning\_rate' ve 'depth' gibi parametrelerin farklı değerleri denenecektir.

GridSearchCV, belirtilen parametre ızgarası ve çapraz doğrulama sayısıyla birlikte modeli eğitim veri seti üzerinde otomatik olarak eğitir ve en iyi parametre kombinasyonunu seçer.

Sonrasında, en iyi parametrelerle tekrar model eğitimi yapılır ve bu model kullanılarak test veri setinde tahminler yapılır. Tahminlerle gerçek değerler arasındaki ortalama karesel hata (MSE) ve R-kare (R2) değerleri hesaplanır ve ekrana yazdırılır.

Ayrıca, gerçek ve tahmin edilen değerlerin çizdirildiği grafikler ve özellik önem düzeylerinin görselleştirildiği bir çubuk grafik de oluşturulur.

## 2.5 Model Doğrulama



**Şekil 31.** Catboost regresyon modelinin hiperparametre ile uygulanmasının sonuçları

0.368 MSE ve 0.64 R2 değerlerine sahip CatBoost regresyon modeli, genel olarak iyi bir performans sergilemektedir. MSE değerinin düşük olması, tahminlerin gerçek değerlere yakın olduğunu gösterirken, R2 değerinin 0.64 olması, modelin veriye uyum sağladığını ve bağımlı değişkenin büyük bir kısmını açıklayabildiğini göstermektedir.

# SONUÇ

CatBoost regresyon algoritması kullanılarak İstanbul ilinin Beylikdüzü ilçesinin konut fiyatlarının analizi yapılmıştır. Modelin performansını değerlendirmek için MSE ve R2 ölçütleri kullanılmıştır.

CatBoost regresyon modeli, veri setindeki konut özniteliklerini kullanarak genel olarak iyi bir performans sergilemiştir. Elde edilen sonuçlara göre, MSE değeri 0.368, R2 değeri ise 0.640 olarak hesaplanmıştır. Bu sonuçlar, modelin konut fiyat tahminlerinde genel olarak iyi bir performans sergilediğini göstermektedir.

Ayrıca, öznitelik önem skorları analizi yapılarak hangi özelliklerin fiyat tahmininde daha etkili olduğu belirlenmiştir. Sonuçlara göre, konutun net ve brüt alanı, mahallenin önemi, banyo sayısı ve bina yaşı gibi özelliklerin tahmin üzerinde önemli bir etkiye sahip olduğu tespit edilmiştir.

CatBoost regresyon algoritması, diğer regresyon yöntemleriyle karşılaştırılmıştır. Karşılaştırmalara göre, CatBoost modeli, karar ağaçlarına ve SVR algoritmalarına kıyasla daha iyi bir performans göstermiştir. Ayrıca, gradient boosting algoritmalarıyla karşılaştırıldığında da rekabetçi sonuçlar elde edilmiştir.

Sonuç olarak, bu çalışma CatBoost regresyon algoritmasının konut fiyat tahmininde etkili bir yöntem olduğunu göstermektedir. Modelin yüksek tahmin doğruluğu, açıklanabilirlik sağlayan öznitelik önem skorları ve genelleme yeteneği, konut piyasası analizi ve benzeri uygulamalarda kullanılabilirliğini desteklemektedir.

# KAYNAKÇA

ERKUL, R. E. (2021). Yapay Zekâ ve Büyük Veri Nasıl Bir Gelecek Vadediyor?. *TRT Akademi*, *6*(11), 192-201.

Kurt, A. (2021). *Ağ Tabanlı Saldırı Tespit Sistemlerinde Topluluk Öğrenme Yöntemlerinin Karşılaştırmalı Performans Analizi* (Master's thesis, Sakarya Üniversitesi).

Karakoyun, M., & Hacibeyoğlu, M. (2014). Biyomedikal Veri Kümeleri İle Makine Öğrenmesi Siniflandirma Algoritmalarinin İstatistiksel Olarak Karşilaştirilmasi. *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi*, *16*(48), 30-42.

Gökalp, M. O., Kayabay, K., Çoban, S., Yandık, Y. B., & Eren, P. E. (2018, November). Büyük Veri Çağında İşletmelerde Veri Bilimi. In *5th International Management Information Systems Conference* (pp. 94-97).

ABBASOĞLU, B. (2020). Ortaokul öğrencilerinin akademik başarılarının eğitsel veri madenciliği yöntemleri ile tahmini. *Veri Bilimi*, *3*(1), 1-10.

Bayazıt, Ö. Ü. N. G. BUZULLAR ÜZERİNE VERİ BİLİMİ ve İLİŞKİLENDİRME ÇALIŞMASI.