Ankara’daki Emlak Verilerinin Çekilmesi ve Makine Öğrenme Yöntemleriyle Fiyat Tahmini

Sercan YILDIRIM

Doç. Dr. Aydın Kaya

Proje Danışmanı

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav

Yönetmeliğinin Bilişim Sistemleri Enstitüsünün

Yazılım Mühendisliği için Öngördüğü

DÖNEM PROJESİ olarak

hazırlanmıştır.

Haziran, 2024

Ankara’daki Emlak Verilerinin Çekilmesi ve Makine Öğrenme Yöntemleriyle Fiyat Tahmini

Sercan YILDIRIM

Doç. Dr. Aydın Kaya

Proje Danışmanı

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

Bilişim Sistemleri Enstitüsünün

Yazılım Mühendisliği için Öngördüğü

DÖNEM PROJESİ olarak hazırlanmıştır.

Haziran, 2024

Sercan YILDIRIM'ın hazırladığı **"Ankara'daki Emlak Fiyatlarının Verilerinin Çekilmesi ve Analizi"** adlı bu çalışma **Doç.Dr. Aydın Kaya** tarafından **Bilişim Enstitüsü Yazılım Mühendisliği Programı**'nda Dönem Projesi olarak kabul edilmiştir.

Doç.Dr. Aydın Kaya

Proje Danışmanı

BİLDİRİM

Hacettepe Üniversitesi Bilişim Enstitüsü, dönem projesi yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında,

- dönem projesi içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğimi,
- kullanılan verilerde herhangi bir tahrifat yapmadığımı,

ve bu dönem projesinin herhangi bir bölümünü bu üniversitede veya başka bir üniversitede başka bir tez/dönem projesi çalışması olarak sunmadığımı beyan ederim.

10.06.2024

Sercan YILDIRIM

ÖZET

Ankara’daki Emlak Verilerinin Çekilmesi ve Makine Öğrenme Yöntemleriyle Fiyat Tahmini

Sercan Yıldırım

Haziran, 2024

Bu çalışmada önce emlakjet isimli emlak satış sitesinden web scraping(veri kazıma) yöntemleri kullanılarak 1360 ev ilanın fiyat ve çelitli özellikleri çekilerek csv olarak kaydedilmiştir. Akabinde bu veriler üzerinde veri düzenleme ve görselleştirme çalışmaları uygulanarak veri hakkında detaylı bilgiler elde edilmiştir. Sonrasında makine öğrenmesi yöntemleriyle ev fiyatlarının tahmin edilmesi gerçekleştirilmiştir. Makine öğrenmesi yöntemlerinden Doğrusal Regresyon, Karar Ağaçları, Destek Vektör Regresyonu, En Yakın Komşu Algoritması, Ridge Regresyonu kullanılmıştır. Tahmin edilen fiyatlar ile test verilerinin fiyatları kıyaslanmış, aralarında tahmin başarı yüzdeleri grafik ile görselleştirilmiştir. Test verilerine en yakın tahmin sonucunu % 48 ile , En Yakın Komşu Algoritması vermiştir. Veri çekilmesi için selenium, görselleştirilmesi için matplotlib ve seaborn kütüphanelerinden faydalanılmıştır.

Anahtar Kelimeler

Makine Öğrenmesi, Doğrusal Regresyon, Karar Agaçları, SVR, Destek Vektör Regeresyonu, KNN, En Yakın Komşu Algoritması, Piton, Veri Kazıma, Veri Düzenleme, Veri Görselleştirme, Selenium, Matplotlib, Seaborn, Pandas

İÇİNDEKİLER

ŞEKİLLER

1. GİRİŞ

Yerleşim imkanlarının artışı ülkelerin ekonomik gelişmişliklerini gösteren unsurlardan birisidir. Gelişen teknoloji ve refah seviyesi ile birlikte insanlar kırsal alanlardan kentlere yerleşme eğilimdedir. Bu durum, ev ihtiyacının artmasına sebep olmuştur. İnsanların hayatlarının büyük çoğunluğunu geçirecekleri evleri seçerken ortak beklentileri vardır. Ulaşım kolaylığı, evin genişliği, yapı kalitesi, kullanışlı olması bunlara örnek olarak gösterilebilir. [1]

İnsanların en temel ihtiyaçlarından biri olan barınma ihtiyacını karşılayan evlere rağbet bazen beklenin üzerinde olabilmektedir. Kendini güvende hissetmek isteyen insanlar ekonomik sıkıntılı dönemlerde özellikle ev sahibi olmak için yüksek fiyatlara rağmen borçlar altına girerek bu taleplerini karşılamaya çalışmaktadır. Ülkemizde özellikle pandemi süreci sonrası ev fiyatlarında ciddi artışlar gözlemlenmiştir. Bu artışın öngörülebilir olabilmesi yatırımcılar ve ev almak/satmak isten insanlar için son derece önemlidir. Ev fiyatlarının doğru şekilde tahmin edilebilmesi için öncelikle geçmiş verilere ihtiyaç bulunmaktadır. Bu kapsamda evlerin fiyatları ve evlerde yaşamak isteyen kullanıcıların tercih sebeplerinin iyi analiz edilmesi ve toplanması gerekmektedir. [2]

Bu dönem projesi kapsamında ilk olarak, emlakjet.com üzerinde yer alan Ankara'daki tüm dairelerin fiyat ve çeşitli bilgilerinin yer aldığı bir veri tabanı oluşturulmuştur. 1360 ev ilanının bilgilerini içeren bu verinin toplanması sırasında python kodlama dilinin selenium kütüphanesi kullanılmıştır. Veri bilimi ile uğraşanlar tarafından selenium kütüphanesiyle veri kazıma(web scraping) işlemi sıklıkla gerçekleştirilmektedir. Dönem projesi kapsamında önce verilerin toplanması gerçekleştirilmiştir. Veri toplama işlemleri sonrası, keşifsel veri analiz çalışmaları gerçekleştirilerek veri temizleme, düzenleme ve görselleştirme işlemleri uygulanmıştır. Son olarak geçmişteki verilerin algoritmalarla makinelere öğretilip istatistiksel metotlar kullanılarak fiyat tahminlemesinin yapılabilmesi için makine öğrenmesi uygulamaları gerçekleştirilmiştir.

Veri görselleştirmeleri ev fiyatlarını etkileyen farktörlerin daha açık bir şekilde görüntülenebilmesine imkan tanımaktadır. Veri görselleştirmesi yapılırken numpy, pandas, scikit-learn, seaborn kütüphaneleri kullanılmıştır. Animasyon içeren

görselleştirmelerden yararlanılmaya özen gösterilmiştir.

Makine öğrenmesi metotlarından doğrusal ve doğrusal olmayan iki kategorideki yöntemler aynı veri üzerine uygulanmış, sonuçları ve tahmin başarı yüzdeleri sonuç kısmında kıyaslanmıştır.

2. VERİ TOPLAMA

Dönem projesinin fikri oluştuktan sonra ihtiyaç duyulan veriye nasıl ve nereden ulaşılabileceği düşünülmüştür. Buna yönelik literatür tarama işlemleri gerçekleştirilmiştir. Bir ev fiyatının belirlenmesinde kullanılan faktörler ile alakalı emlak siteleri incelenmiş. Ev fiyatlarının değerlemesinde çoğu emlak sitesinin belirli bir standartta oluşturulduğu fark edilmiştir. Akabinde bu sitelerin veri kaynağı olarak kullanılabileceği değerlendirilmiştir. Bu alanda yapılan çalışmalara bakıldığında Veri Kazıma(Web Scraping) yöntemlerine başvurulabileceği anlaşılmıştır.

2.1. Veri Kazıma(Web Scraping) Nedir

Günümüzde hayatımızın her noktasında sıklıkla faydalandığımı internet aynı zamanda bünyesinde ciddi miktarda veriyi, bilgiyi ihtiva etmektedir. Bu ciddi miktarda veri metin, ses yada görüntü formatlarında olabilmektedir. Verileri barındıran web siteleri aynı zamanda bir veri ambarı konumundadır. Günümüzde bu veri ambarlarından çeşitli veri kazıma yöntemleri yardımıyla yararlanılabilmektedir. Web sitelerinin verileri belirli formatta, otomatik şekilde tutması veri analizi ile uğraşanların işlerini de kolaylaştırmaktadır. Veri kazıma işlemi bu yönden veri temini işlemlerinde kullanılabilmektedir.

Web sitelerinde hazır olarak bulunan verilerini çeşitli kütüphaneler kullanılarak bir kural eşliğinde otomatik olarak çekilmesi işlemine veri kazıma denilmektedir. [3]

2.2. Emlakjet Web Sitesinin Önyüz Tanıtımı

Veri kazıma çalışması için Ankara ilinde yer alan tüm daire statüsündeki ilanların verilerinin .csv formatında emlakjet sitesinden alınması kararlaştırılmıştır. Bu sitenin seçilmesinde, önyüzündeki standart yapı, veri özelliklerinin çeşitli olması gibi faktörler etkili olmuştur. Yaklaşık bin beşyüz ilan taranması hedeflenmiştir. Bu bin beşyüz ilanın yaklaşık iki yüz adedinin reklam olduğu tahmin edilmiştir.

2.3. Ulaşılmak İstenen Veri Etiketlerinin Belirlenmesi

Emlakjet önyüzü incelendiğin de her bir ilan objesinin bir class'a ait olduğu ve xpath'lerinin sonundaki numaranın sıralı olarak ilerlediği fark edilmiştir. Bu ilan

numaralarındaki sıralı artışın bir while döngüsü ile ilerletilebileceği üzerine düşünülmüştür. Ankara ilinde yer alan tüm daire tipindeki ilanlar ise url erişimi ile ilk girişte süzülebilmektedir. İlan içerisine girdikten sonra fiyat ve özellik bilgilerinin bağımsız bir sınıf olarak belirlendiği görülmüştür.

2.4. Veri Toplama Adımları ve Algoritması

Veri toplama/kazıma işlemine başlanmadan önce anaconda yazılımı bilgisayara kurulmuştur. İşlemler, jupyter lab isimli geliştirme ortamında gerçekleştirilmiştir. İlk olarak web tarayıcının çalışması için gerekli olan uyumlu “crome driver” bilgisayara indirilmiştir. Sonrasında ilgili kütüphanelerin yükleme işlemleri komut satırı üzerinden gerçekleştirilmiştir. Sonrasında aşağıdaki işlem adımlarını takip edecek bir kod parçacığı geliştirilmiştir.

#	İşlem
1	Driver Path ve Açılmak İstenen URL Tanımlanır
2	Class Erişimi ile Çekilecek Olan Veriyi String'e Çevirecek Bir Metot Yazılır
3	Sayfaları Dolaşması için Bir While Döngüsü Kurulur
4	İlanları Dolaşması için İkinci While Döngüsü İlk Döngünün İçersine Kurulur
5	Reklam Olan İndisler Bir Listeye Kaydedilir
6	Reklam Olan İndislerin Tıklanmadan Atlanması için Bir While-If Cümleciği Kurulur
7	Sayfaların Doğru Yüklendiğinden Emin Olmak için "implicitly_wait" metodu eklenir
8	Tıklanacak İlanın Önüne Her Hangi bir Pop-up Penceresi Gelmemesi için Try-Except Yapısın İçersininde Çıkan Ekranlar Gizlenir
9	Açık Bir Şekilde Görünür Olduğundan Emin Olunan İlan Kutusuna XPath'I Kullanılarak Tıklama İşlemi Gerçekleştirilir
10	Sayfaların Doğru Yüklendiğinden Emin Olmak için "implicitly_wait" metodu eklenir
11	Açılan İlan Sayfasından İstenen Bilgiler "find_elements" Metodu ile Sınıf İsimlerine Göre Çekilir.
12	Çekilen Bilgiler Bir String'e Çevrilir

#	İşlem
13	String'e Çekilen Değişken Boşluklarına Göre Ayırılıp Bir List'e Dahil Edilir
14	İstenen Veri Bir Data Frame'e Çevrilir
15	İlgili Data Frame .csv Uzantılı Dosyaya Kayıt Edilir
16	Bir Önceki Sayfaya Dönülür
17	Sayfadaki Tüm İlanlar Dolaşılıncaya Kadar İşlemler Tekrarlanır
18	Tüm İlanlar Dolaşıldıktan Sonra Bir Sonraki Sayfaya Geçilir

Tablo 1 Veri Kazıma Adımları

2.5. Kullanan Kütüphaneler ve Metotlar

Proje kapsamında kod geliştirilirken mümkün olduğunca temiz, anlaşılır ve bakımı kolay bir yazıma sahip bir blok oluşturulmaya çalışılmıştır. Kullanılan kütüphaneler Şekil-1'de verildiği gibidir.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.options import Options
import time
import pandas as pd
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import ElementClickInterceptedException
from selenium.webdriver.common.action_chains import ActionChains
```

Şekil 1 Veri Kazıma Kısmında Kullanılan Kütüphaneler

Veri kazıma aşamasında kullanılan kütüphanelerin hangi amaç ile kullanıldıkları aşağıda kısaca ile açıklanmıştır.

- **Webdriver** – Web tarayışını başlatmak otomatik olarak kontrol edebilmek için kullanılan kütüphanedir.
- **Service** – Hizmet sağlayıcısının belirlenmesinde görev yapar.
- **By** – Önyüzde belirlediğimiz xpath, sınıf gibi HTML tag'lerinin hangilerini seçeceğimizi gösterir.

- **Options** – Tarayıcı ayarlarını yapmamızı sağlar. Pencerenin büyütülmesi için kullanılmıştır.
- **Time** – Gerekli durumlarda bekleme işlemini gerçekleştirmek için dahil edilmiştir.
- **Pandas** – Tabular veri ile dataframe'lerle kolaylıkla çalışılmasını sağlar, düzenleme-sıralama gibi işlemlerin çoğu bu kütüphane sayesinde gerçekleştirilmiştir. [4]
- **WebDriverWait** – Sayfa yüklenmesinin belirli bir koşula bağlı oluncaya kadar işlemlerin bekletilmesini sağlamaktadır.
- **Expected_Conditions** – WebDriverWait ile birlikte kullanılır. Sayfa yüklemesinin tamamlandığı koşulun tanımlandığı yerdir.
- **ElementClickInterceptedException** – Web sayfalarında sıkça rastlanan açılır pencerelerin kodun işleyişini bozmaması için eklenmiştir. Doğru butona tıklanabilmesi için gereklidir. Aksi takdirde tıklama ve giriş işlemlerinde sorunlar yaşanabilmektedir.
- **ActionChain** – Bu kütüphane bir butona tıklama yapılacağı zaman buton ile üst üste gelen bildirimlerin tıklanmaması için ekranın belirli bir lokasyonuna tıklanmasını sağlayan yeteneği kazandırmaktadır. Sol üst köşenin 0,0 olarak baz alındıktan sonra istenen marjlarda x ye y eksenlerinde kaydırılarak uygun noktaya tıklanmasına imkan tanımaktadır.

```
# İlan Sayfalarını dolaşabilmek için döngü yapıldı
sf = 3
while sf <= 50:
    # Sayfa içindeki ilanları dolaşabilmek için döngü yapıldı.
    # ilk 9 sf 39 / 10 dan itibaren 35
    il = 1
    while il <= 35:
        # Tıklanması istenmeyen reklamların indisleri liste yapıldı.
        atlama_listesi = [3 , 8 , 9 , 11 , 14 , 18 , 25 , 26 , 33 , 40 ]
        u = 0
        while u < len(atlama_listesi):
            if atlama_listesi[u] == il:
                il = il+1
                u=0
            else:
                u = u + 1
```

Şekil 2 Sayfa ve İlan Açma Kod Bloğu

Kütüphanelerin yüklenmesi sonrası, web tarayıcı get metodu ile başlatılmıştır. Akabinde Şekil 2'de görterildiği gibi iç içe iki while döngüsü kurulmuş, sırasıyla sayfa ve ilanlarında gezilmiştir.

"atlama_listesi" adında bir listeye reklam içeren xpath numaraları eklenmiştir. Döngü sayaçları bu değerlere geldiğinde atlayabilmesi için bir if yapısı ile bir kontrol aşaması eklenmiştir.

```
try:
    driver.implicitly_wait(20)
    tikla = driver.find_element(By.XPATH, "//*[@id='listing-search-wrapper']/div["+str(il)+"]")
    driver.execute_script("arguments[0].scrollIntoView(true);", tikla)
    driver.execute_script("window.scrollBy(0, -100);")
    tikla.click()
    driver.implicitly_wait(10)
except ElementClickInterceptedException:
    driver.implicitly_wait(20)
    overlapping_element = driver.find_element(By.XPATH, "//*[@id='listing-search-wrapper']/div["+ str(il)+"]/div")
    driver.execute_script("arguments[0].style.visibility='hidden'", overlapping_element)
    driver.execute_script("arguments[0].scrollIntoView(true);", tikla)
    driver.execute_script("window.scrollBy(0, 150);")
    # offsetli bir şekilde tıklama yaparak muhtemel pop up kesişiminden kaçınıyoruz.
    x_offset = 200  # Tıklanacak x koordinatı
    y_offset = - 10  # Tıklanacak y koordinatı
    actions = ActionChains(driver)
    actions.move_to_element_with_offset(tikla, x_offset, y_offset).click().perform()
    print( str(sf) + " - " + str(il) + " excepte düştü.")
    # tikla.click()
    driver.implicitly_wait(10)
    driver.execute_script("arguments[0].style.visibility='visible'", overlapping_element)
```

Şekil 3 İlan Başlıklarına Tıklayan Kod Bloğu

Sonrasındaki try-except yapısı ise ilana tıklama işlemleri sırasında ilan butonunun üzerine denk gelen açılır pencereleri yönetmek için kullanılmıştır. Açılır pencereler aktif olduğu durumlarda except içerisinde tanımlanmış olan, açılır pencerelerin gizlenmesi işlevi devreye girmektedir. Böylece kodun çakılmasının önüne geçilmiştir.

"find_elements" istenen ilanın xpath'ini bularak, tıklanması için kaydetmektedir. Akabinde "implicitly_wait" metodu max 10 sn bekledikten sonra yükleme işlemi hazır olduğunda ilana giriş yapılmaktadır.

```
driver.implicitly_wait(10)
elements = driver.find_elements(By.CSS_SELECTOR, "._3tH_Nw")
driver.implicitly_wait(10)
fiyatlar = driver.find_elements(By.CSS_SELECTOR,"._2TxNQv")
detaylar = []
fiyat = []
for i in fiyatlar:
    fiyat.append(i.text)
for k in elements:
    detaylar.append(k.text)
det_str = listToString(detaylar)
ayrı = det_str.split("\n")
df = pd.DataFrame(ayrı)
df_yeni = df[:]
df_yeni = df_yeni.reset_index()
df_yeni.drop("index", axis = 1, inplace = True)
df_liste = df_yeni.values.tolist()
içerikler =[]
l = 1
while l <= 27:
    içerikler.append(df_liste[l])
    l = l+2
fiyat_sade = fiyat[1].strip()
fiyat_sade = fiyat_sade.replace("TL","")
içerikler.append([fiyat_sade])
df_içerikler = pd.DataFrame(içerikler).T
df_içerikler.to_csv(r"emlakjetveri.csv",encoding="utf-8",index=False, mode="a",header=False)
il = il+1
driver.execute_script("window.history.go(-1)")
driver.implicitly_wait(10)
```

Şekil 4 Bilgilerin .csv Uzantılı Dosyaya Eklendiği Kod Bloğu

Şekil 3'te gösterilen kısımda ise açılan ilandan önce elements ve fiyatlar adında gerekli bilgileri içeren listeler çekilmiştir. Bu listedeki değerler boşlıklarına göre ayrılıp "df_içerikler" adında bir dataframe'de birleştirilmiştir. Her ilanda çekilen bilgiler bu dataframe değişleni ile .csv uzantılı dosyaya eklenerek süreç devam ettirilmiştir.

```
driver.implicitly_wait(10)
sf = sf + 1
print(str(sf-1) +  "  . sayfa tamamlandı")
driver.get("https://www.emlakjet.com/satilik-daire/ankara/"+str(sf)+"/")
driver.implicitly_wait(10)
```

Şekil 5 Sayfa Değişiminin Yapıldığı Kod Bloğu

Şekil 4'te ise sayfa sonlarında url değiştirilerek sonraki sayfaya geçilmesinin sağlandığı bir kısım gösterilmiştir. Sayfaların yüklenmesinde network'te yaşanabilecek gecikmeleri tolere edebilmesi için "implicitly_wait" mototları

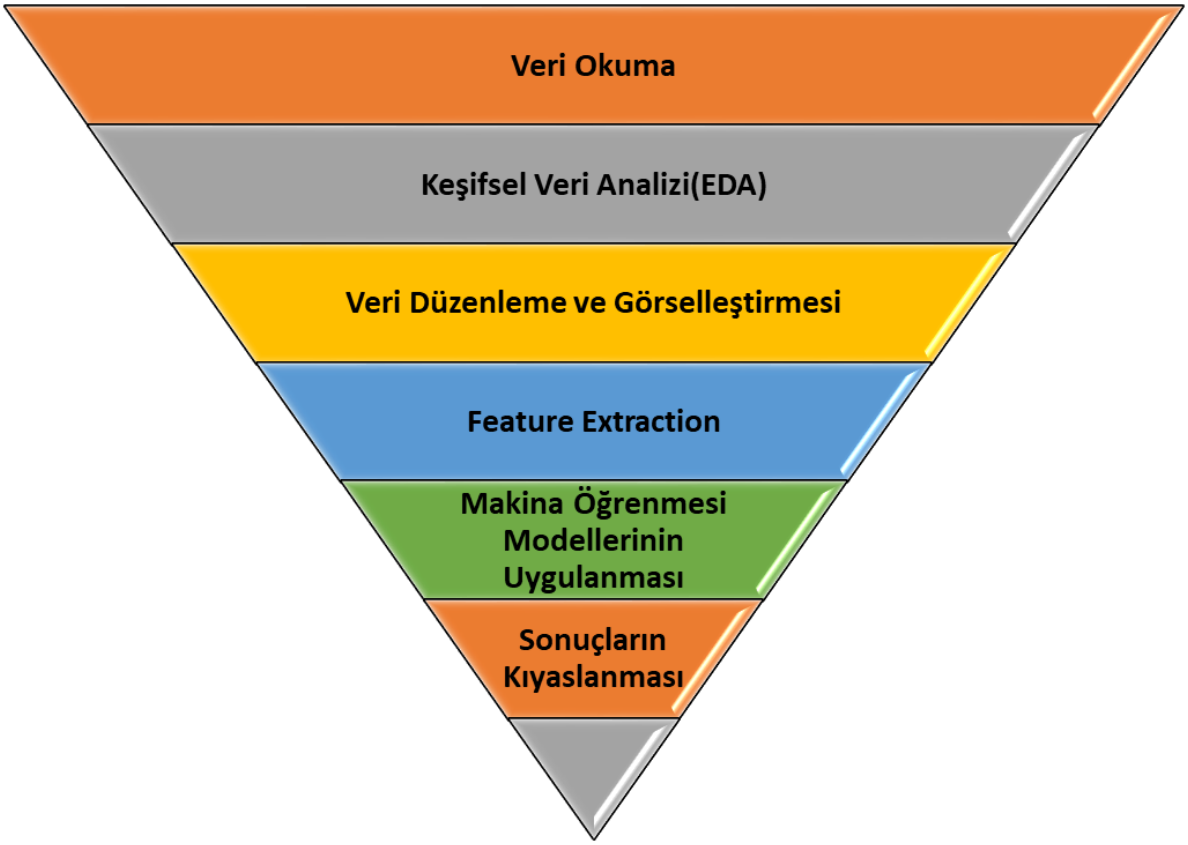
kullanılmıştır.

Veri toplama işlemi tamamlandıktan sonra .csv uzantılı dosya her ortamdan erişilebilmesi için github'a yüklenmiştir. Github linki referans olarak kaynakçaya eklenmiştir. [5]

3. VERİ DÜZENLEME VE MAKİNE ÖĞRENMESİ UYGULAMASI

3.1. Veri Düzenleme İşlemleri

Veri GitHub üzerine eklendikten sonra okuma işlemi gerçekleştirilmiştir. Sırasıyla Şekil 6'da belirtilen adımlar uygulanmıştır.



Şekil 6 İşlem Adımları

Veri okuma işlemleri sonrası öncelikle sütun başlıkları eklenmiştir. Reklamlar hariç 1360 veri üzerine bir analiz gerçekleştirilmiştir. Gereksiz olduğu değerlendirilen, "Kategorisi", "Türü" ve "Tipi" sütunları silinmiştir. İlan numarası biricik olduğu için index sütun olarak ayarlanmıştır.

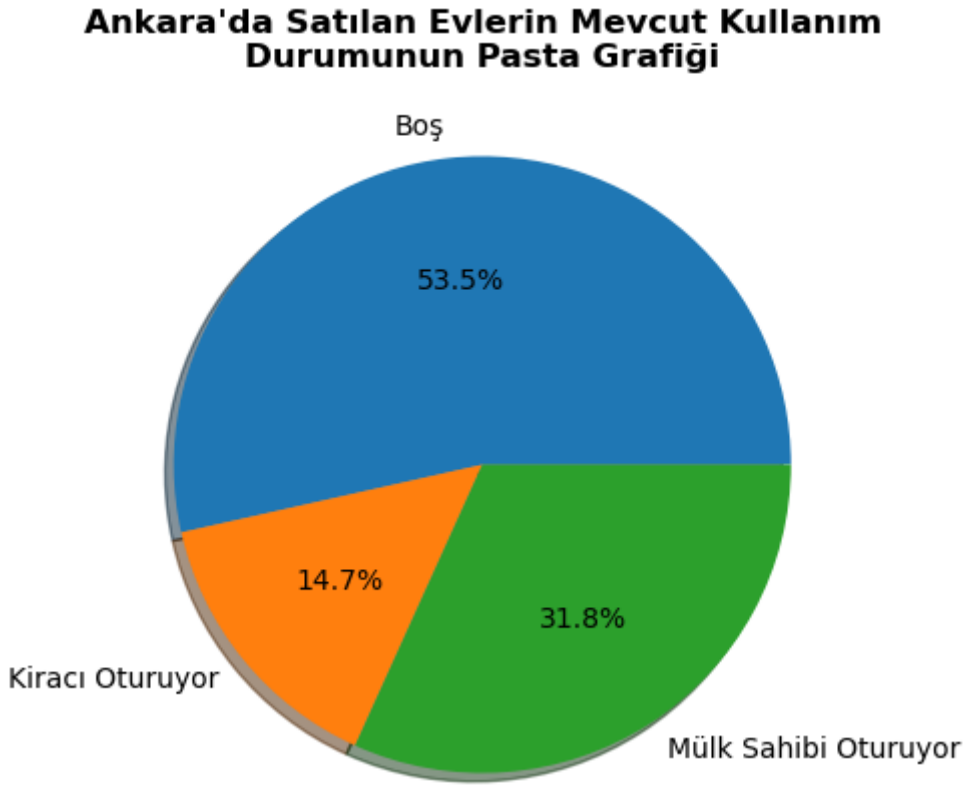
Öncelikle veriyi anlamak adına her sütun elemanının tekrarlayan değerlerine bakılmıştır.

Fiyat, Brüt Metrekare ve Net Metrekare sütunlarındaki birimler temizlenmiştir. Aralık olarak belirtilen “Bina Yaşı” sütunundaki değerlerin ortalamaları atanmıştır.

Bulunduğu kat ve oda sayısı sütunlarında gereksiz olarak verilmiş değerler “diğer” şeklinde gruplandırılmıştır.

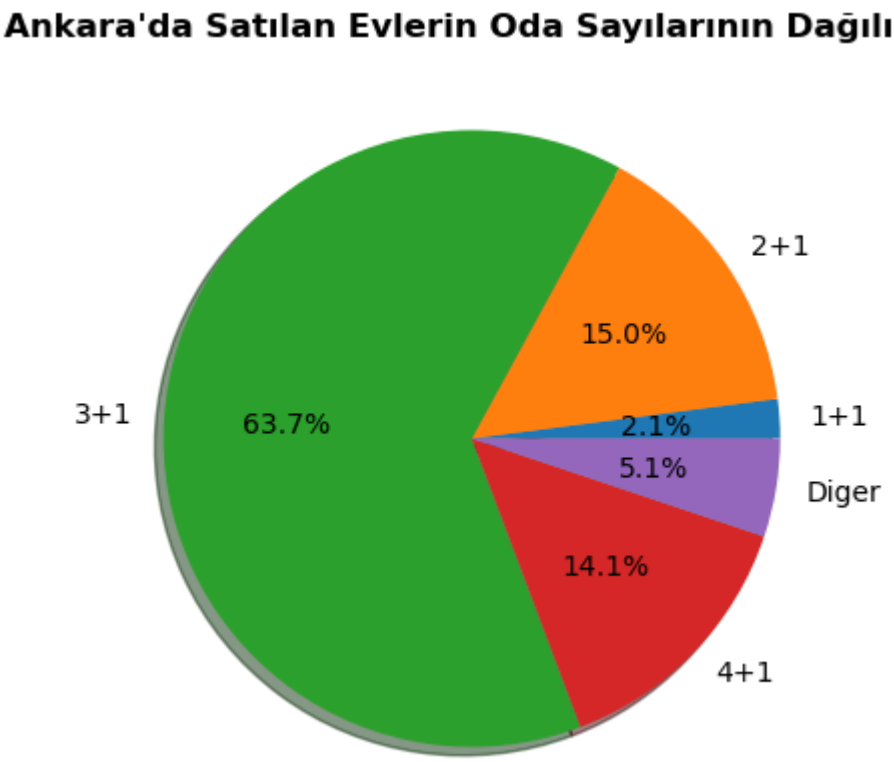
3.2. Veri Görselleştirmesi

Veri görselleştirmesi çalışmalarında matplotlib ve seaborn kütüphaneleri kullanılmıştır. Öncelikle Şekil 7’de gösterilen ilandaki evlerin mevcut oturum durumlarına bakılmıştır. İlandaki evlerin büyük bir kısmının(%53) boş olduğu, ikinci olarak da % 31 ile mülk sahiplerinin aktif olarak oturduğu belirlenmiştir.



Şekil 7 İlanların Mevcut Kullanım Durumları

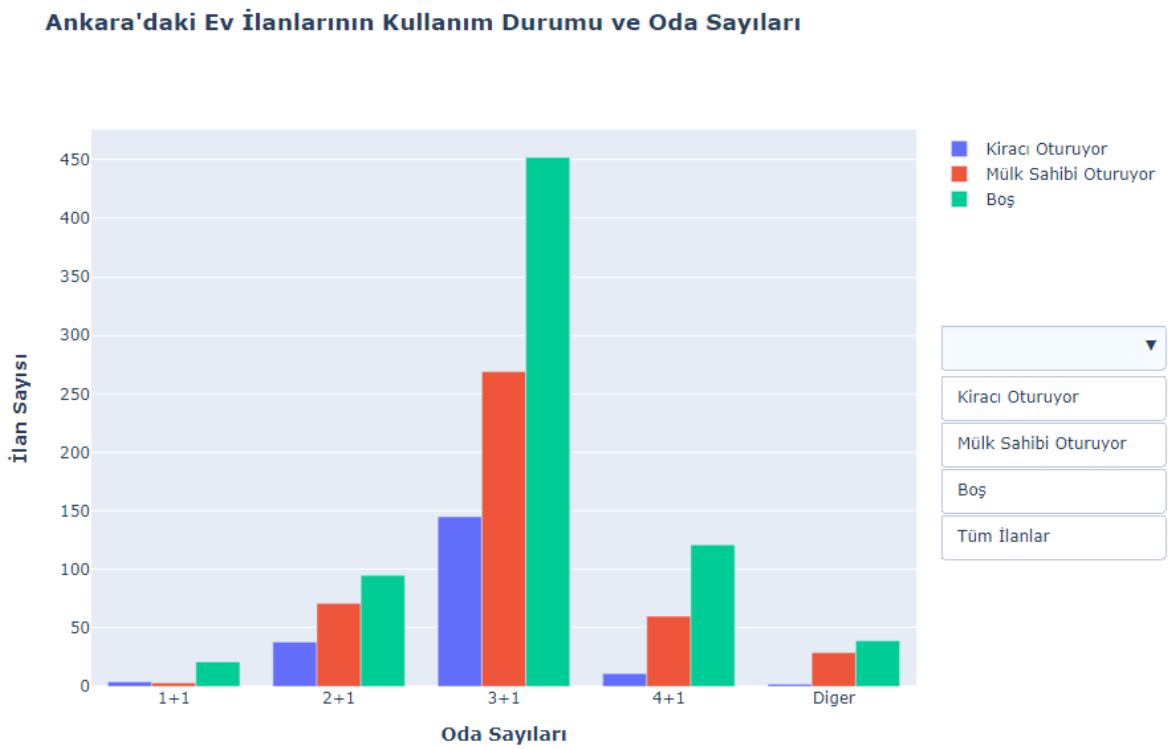
“İlandaki evler oda sayıları bakımından bir yığılma gösteriyor mu?” sorusu akıllara geldiği için bir pasta grafiği de oda sayıları için çizdirilmiştir. Sonuç Şekil 8’de paylaşılmıştır.



Şekil 8 İlandaki Evlerin Oda Sayılarının Dağılımı

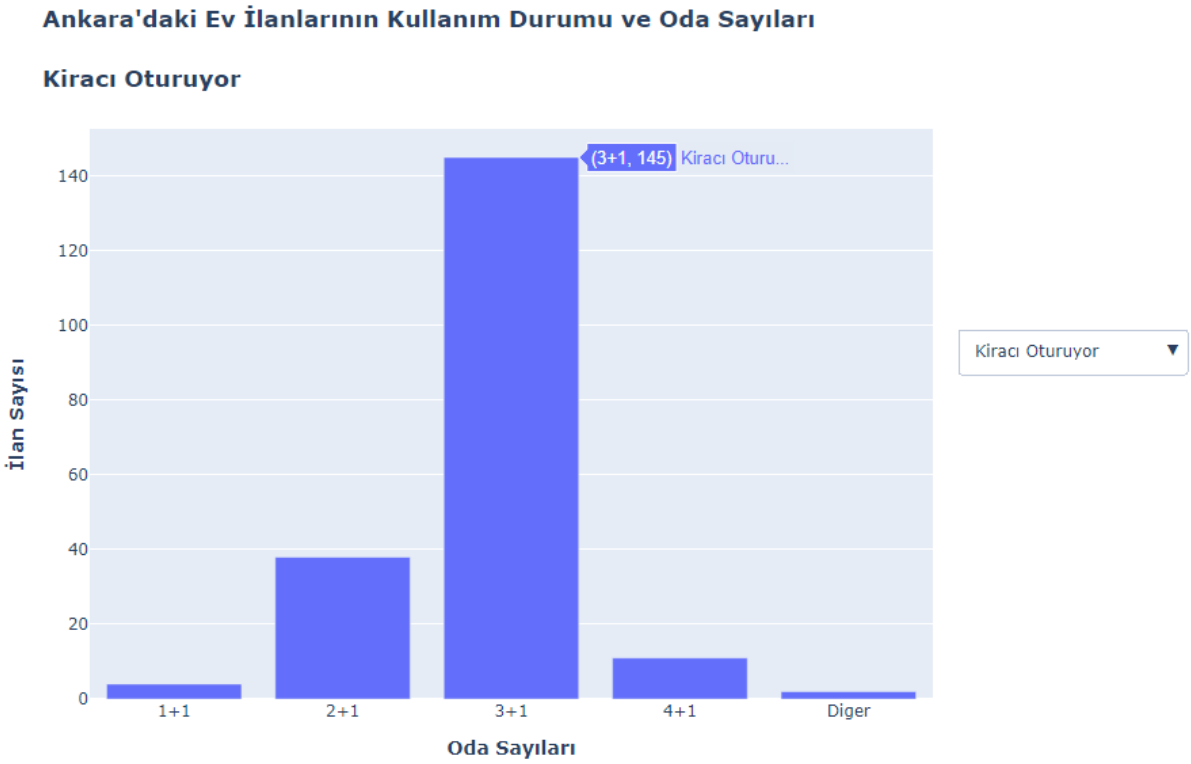
Pasta grafiğinde net bir şekilde görüldüğü üzere ilanda satılık evlerin çoğunlukla 3+1 sonrasında 4+1 olduğu tespit edilmiştir.

Bu iki kritik durumun birlikte analiz edilebilmesi bir etkileşimli bir görselleştirme üzerine çalışılmıştır. Yapılan etkileşimli gösterimde bir dropdown menü ile istenen kullanım durumuna ait evlerin oda sayıları görüntülenebilir duruma gelmiştir. Ayrıca sağ üstteki cetvel ile de bir yada birden fazla oturum durumu eklenip çıkarılabilmektedir. İki değişkenin birbirleri ile birlikte süzülmesine imkan tanınmıştır. Etkileşimli grafik iki kez tıklanması halinde ilk baştaki konuma geri dönebilmektedir.

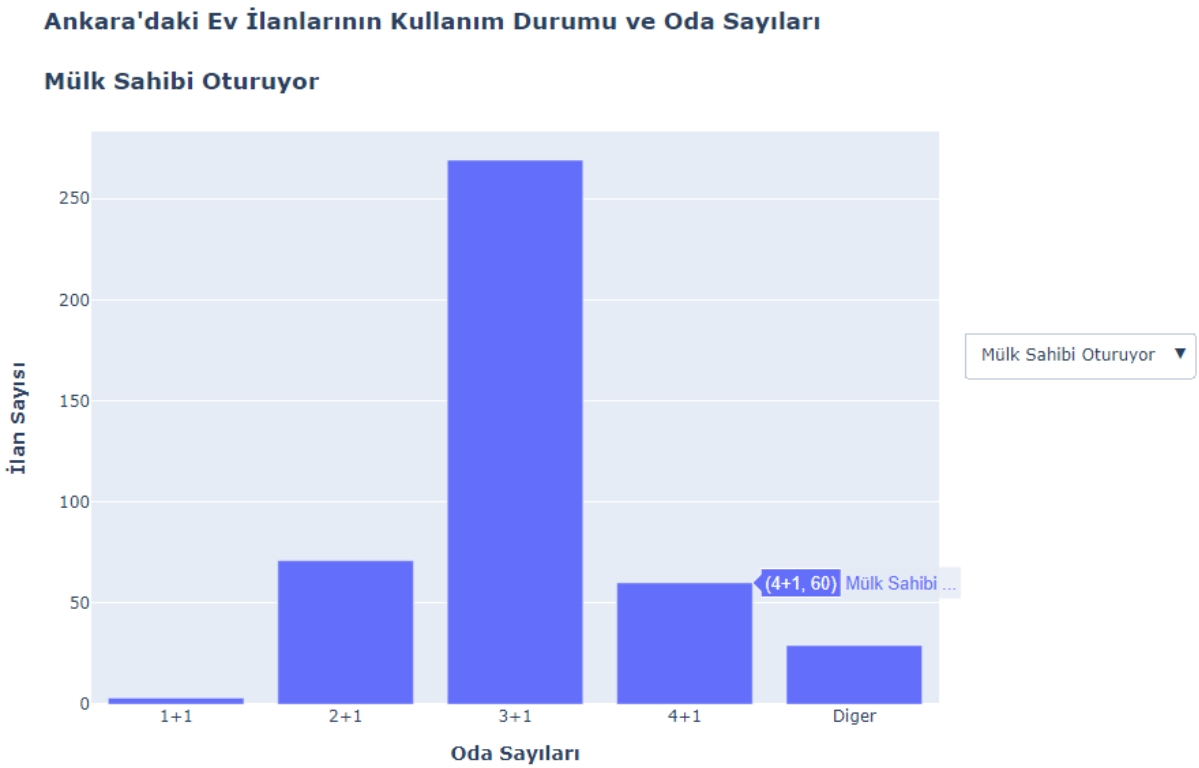


Şekil 9 Etkileşimli Grafik - Tüm İlanlar

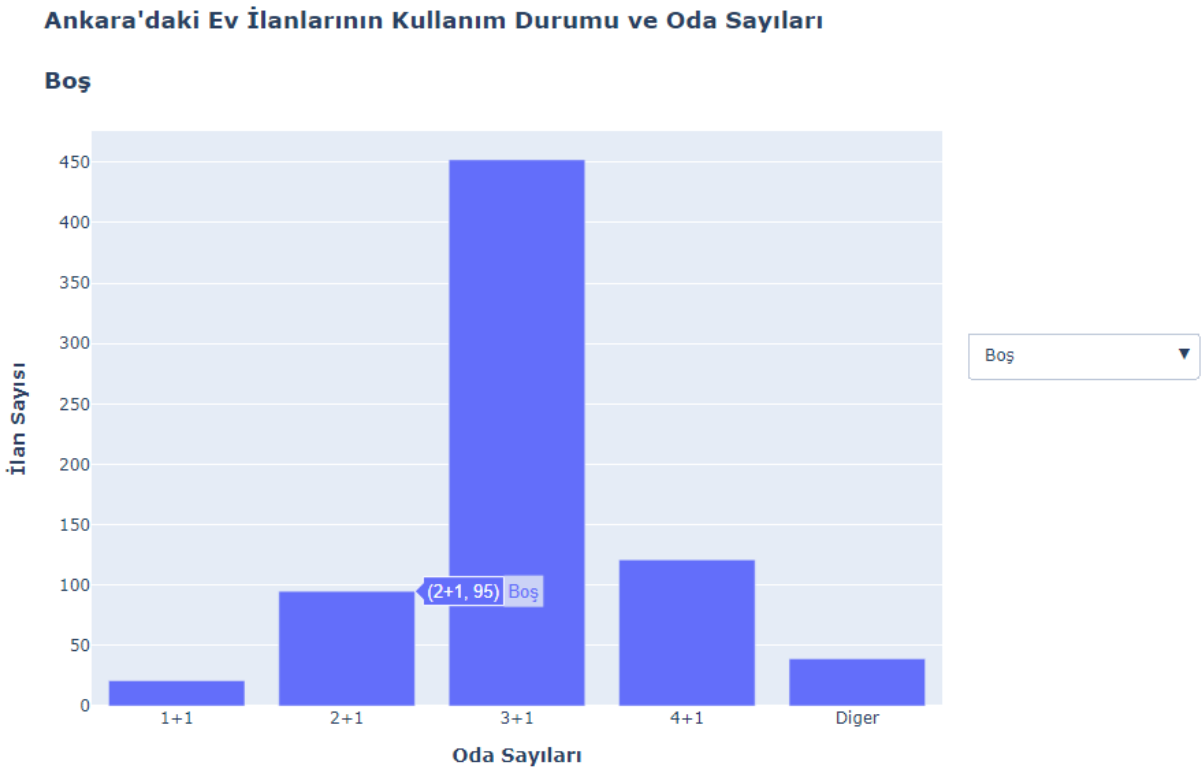
İstenmesi durumunda barların üzerinde durularak hangi kullanıma ait evden kaç adet olduğu pop-up şeklinde kullanıcı ile paylaşılmaktadır.



Şekil 10 Etkileşimli Grafik - Kiracılı İlanlar



Şekil 11 Etkileşimli Grafik - Mülk Sahibinin Oturduğu İlanlar

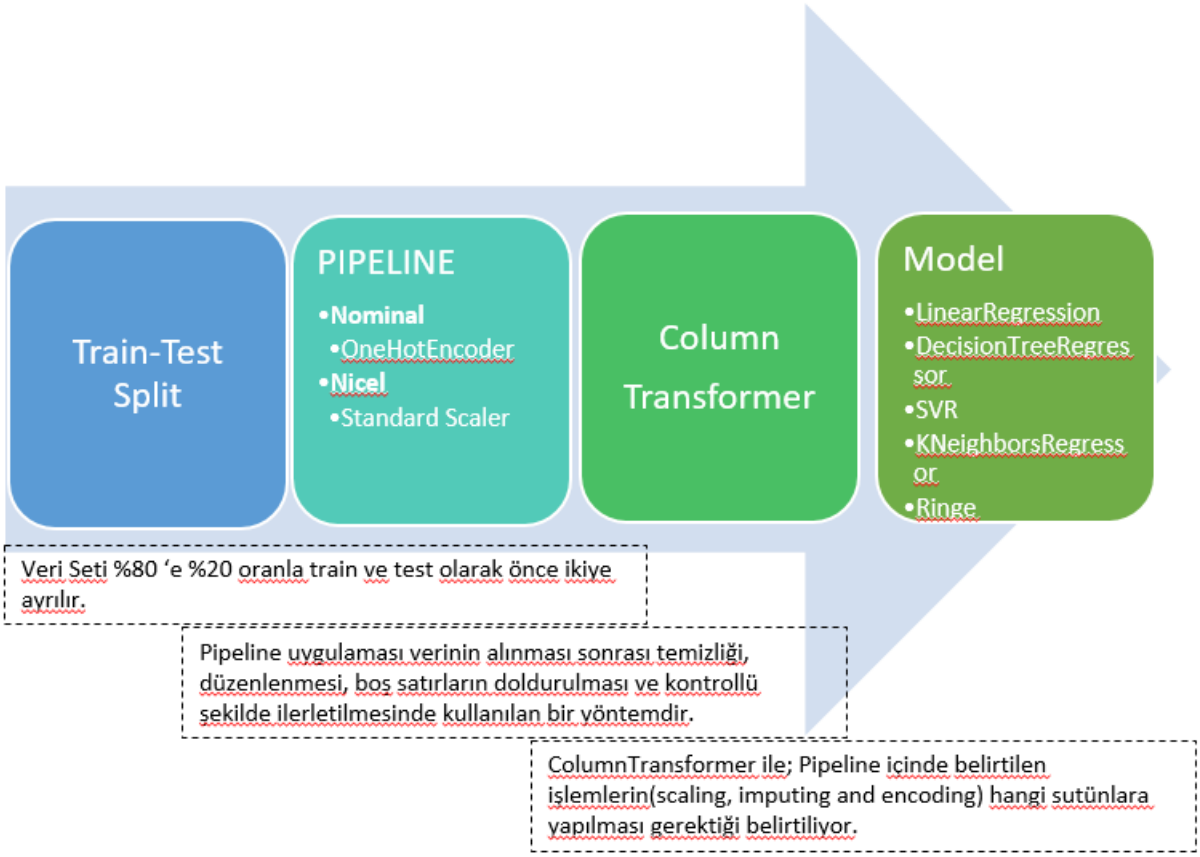


Şekil 12 Etkileşimli Grafik - Boş İlanlar

Görselleştirme çalışmalarından sonra makine öğrenim için verinin hazırlanması işlemlerine geçilmiştir.

3.3. Makine Öğrenmesi Yöntemleri

Makine öğrenmesi modelleri kullanılarak elde edilen ve görselleştirilen veriler üzerinde bir fiyat tahmininin yapılabilirliği değerlendirilmek istenmiştir. Bu amaçla sırasıyla aşağıdaki adımlar uygulanmıştır.



Şekil 13 Makine Öğrenimi İşlem Adımları

Öncelikler veri seti eğitim ve test olarak ikiye ayrılmıştır. Sonrasında kategorik veriler için OnerHotEncoder; nicel veriler için Standard Scaler metotları pipline'a ekleniştir. Kurulan pipeline modeli sütun dönüştürücüleri yardımıyla çalıştırılmıştır.

Sonrasında aşağıdaki kısaca açıklanan metotlar veri setine öğretilmiştir.

Çalışmaya başlanmadan önce bu tahmin problemi için kullanılabilecek makine öğrenmesi yöntemleri hakkında bir literatür taraması gerçekleştirilmiştir. Yapılan araştırmalar neticesi; Doğrusal Regresyon, Karar Ağaçları Regresyonu(KAR), Destek Vektör Regresyonu, En Yakın Komşu Algoritması, Ridge Regresyonunun kullanılması kararlaştırılmıştır. [6]

Doğrusal Regresyon(Lineer Regression)

Doğrusal Regresyon, makine öğrenmesinin denetimli öğrenme modellerinde, bağımsız değişkenler ile bağlı değişken arasındaki en uyumlu doğruyu çizen bir algoritmadır. Bir veya daha fazla girdi ile çıktı arasındaki istatistiksel ilişkiyi tanımlamada yardımcı olur. Girdi sayısına göre tekli veya çoklu olarak adlandırılmaktadır. Modelin amacı gerçek çıktı ile tahmin arasındaki farkı minimum etmektir. Bu hata minimizasyonu işleminde gradient desent yöntemi kullanılmaktadır.

Karar Ağaçları Regresyonu(Decision Tree Regression)

KAR yöntemi kısaca, bağımsız değişkenleri bilgi kazançlarına göre aralıklara bölmektedir.Tahminleme kısmında ilgili aralığa denk gelen bir değer sorulduğunda, o aralığın ortalamasını döndürmektedir. Diğer regresyon modellerinden farklı olarak kesikli bir yapıya sahiptir. İlgili aralığa düşen tüm değerler için aynı ortalama değerini verecektir. [7]

Destek Vektör Regresyonu (Support Vector Regression)

Destek Vektör Makineleri öncelikle sınıflandırma problemleri için kullanılmıştır. Sonrasında regresyon modellerinde de bu yöntemden faydalanılmıştır. Makine öğreniminin denetimli öğrenme modelleri başlığı altında değerlendirilir.[3] Amaç çizilecek doğrunun maksimum sayıda noktayı içermesini sağlamaktır. Bu noktalar destek noktası şeklinde isimlendirilmektedir. SVR modeli uygulanırken Raidal Basis Function(RBF) metodu ile birlikte kullanılırsa, doğrusal olmayan problemlerin çözümünde de kullanılabilmektedir. [8]

En Yakın Komşu Regresyonu(KNeighbors Regression)

KNN temel olarak problemimizdeki noktaların k adet kümeye ayrılırken küme içindeki noktalar arasındaki mesafeyi minimum, kümeler arasındaki mesafeyide maksimum yapmayı amaçlamaktadır. Regresyonda uygulanmasında ise tahmin edilecek noktanın en yakınındaki küme belirlenir ve küme elemanlarının ortalaması tahmin olarak belirlenmektedir. [6]

Ridge Regresyon

Ridge regresyon temelde en küçük kareler yöntemine dayanan bir yöntemdir. [5] Genellikle aşırı derecede uyum(overfitting) probleminin olduğu durumlarda

kullanılmaktadır. Çoklu bağımsız değişkenlerin olduğu problemlerin regresyon denklemlerindeki katsayıları küçültmek için bir cezalandırma terimi kullanır. Bu ceza terimi katsayıların değerini sıfıra yaklaştırarak modeli daha basit bir hale getirmeyi sağlar. Böylece modelin genelleme yeteneği artırılmış olmaktadır. Bu ceza terimi lambda olarak isimlendirir ve bir hiperparametre ile kontrol edilir.

```
models=[LinearRegression(),DecisionTreeRegressor(),SVR(),KNeighborsRegressor(),Ridge()]
results=[]
names=[str(i).strip("()") for i in models ]
model_name=[]
```

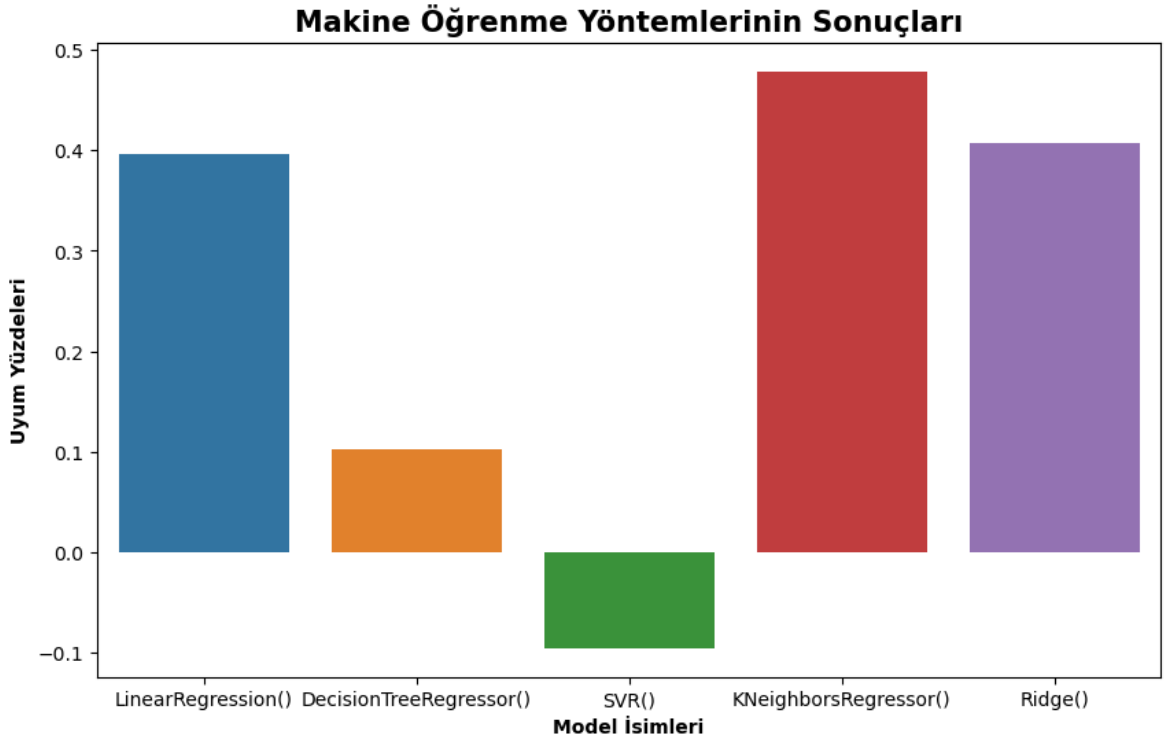
```
results=[]
name_models=[]
for i in models:
  X_transformed_columns=ct.fit_transform(X_train)
  X_validation_Transformed=ct.transform(X_validation)
  kf=KFold(n_splits=10,shuffle=True,random_state=7)
  final_model=i
  final_model.fit(X_transformed_columns,Y_train)
  mean_score=cross_val_score(final_model,X_validation_Transformed,Y_validation, cv=kf).mean()
  results.append(mean_score)
  name_models.append(str(i))
  print("Obtained mean score is {} for the model {}".format(mean_score,i))
```

Şekil 14 Makine Öğrenmesi Metotlarını Uygulandığı Kod Bloğu

Kısaca açıklanmış olan bu makine öğrenme yöntemleri bir liste içersine tanımlanmış ve her bir liste elemanı bir döngü içersinde çağrılarak koşturulmuştur. Sonuçlar bir listeye aktarılmış ve Şekil 15'teki gibi görselleştirilmiştir. Burdaki değerler test verileri ile tahmin değerleri arasındaki uyumu göstermektedir.

3. SONUÇ

Şekil 15'te gösterildiği üzere yapılan çalışmaların neticesi olarak, ev fiyatlarını tahminleme problemini en iyi şekilde çözebilecek metot olarak en yakın komşular yöntemi belirlenmiştir.



Şekil 15 Makina Öğrenmesi Yöntemlerinin Sonuçları

Yüzde 48 değerinin daha da iyileştirilebilmesi için daha nicel verilerin yer aldığı veri seti temin edilebilir. Boosting algoritmaları ile sonuçlar iyileştirilmeye çalışılabilir. Tam sonuçlar şekil 16'da paylaşıldığı şekildedir.

```
Obtained mean score is 0.3968383919123771 for the model LinearRegression()
Obtained mean score is 0.10194888262942317 for the model DecisionTreeRegressor()
Obtained mean score is -0.09527791734040328 for the model SVR()
Obtained mean score is 0.47864679704028124 for the model KNeighborsRegressor()
Obtained mean score is 0.4074341663730097 for the model Ridge()
```

Şekil 16 Makine Öğrenme Yöntemleri Tam Sonuçlar

Bu proje ile verilerin otomatik şekilde çekilmesi, verilerin anlaşılması/görselleştirilmesi, düzenlenmesi ve sonrası makine öğreniminde kullanılması gerçekleştirilmiştir.

KAYNAKÇA

[1] Y. L. Z. W. ,. M. Z. ,. T. W. C. Choujun Zhan, «A hybrid machine learning framework for forecasting house price,» *Expert Systems With Applications,* cilt 233, p. 120981, 2023.

[2] B. M. S. P. Jean-Charles Bricongne, «Web-scraping housing prices in real-time: The Covid-19 crisis in the UK,» *Journal of Housing Economics,* cilt 59, p. 101906, 2023.

[3] U. B. R. Sumit Kumar, «A technique of data collection: web scraping with python,» Statistical Modeling in Machine Learning, KARNATAKA, 2023.

[4] O. N. A. F. A. A. O. O. Y. F. A. G. O. Abigail Bola Adetunji, «House Price Prediction using Random Forest Machine Learning Technique,» %1 içinde *The 8th International Conference on Information Technology and Quantitative Management(ITQM 2020 & 2021)*, 2022.

[5] S. Yıldırım, «GitHub,» 07 06 2024. [Çevrimiçi]. Available: https://raw.githubusercontent.com/ysercan/emlakdata/main/emlakjetveriseti.csv. [Erişildi: 07 06 2024].

[6] S. B. ,. C. A. Salim Lahmiri, «A comparative assessment of machine learning methods for predicting housing prices using Bayesian optimization,» *Decision Analytics Journal,* cilt 6, p. 100166, 2023.

[7] S. Elnagar, «Using deep learning to enhance electronic service quality: Application to real estate websites,» *Intelligent Systems with Applications ,* cilt 21, p. 200330, 2024.

[8] M. R. B. L. Katharina Baur, «Automated real estate valuation with machine learning models using property descriptions,» *Expert Systems With Applications,* cilt 213, p. 119147, 2023.