BLM3590 İSTATİSTİKSEL VERİ ANALİZİ


# PROJE RAPORU

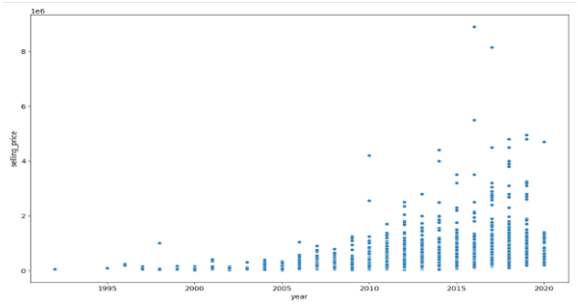
Fatih ALTINCI

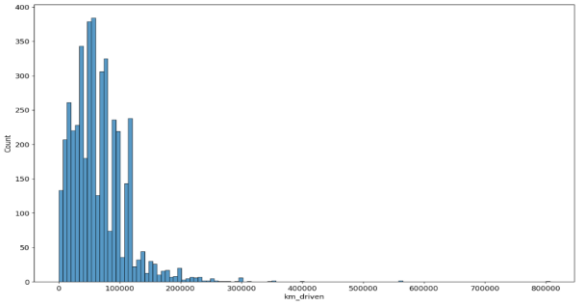
20011610

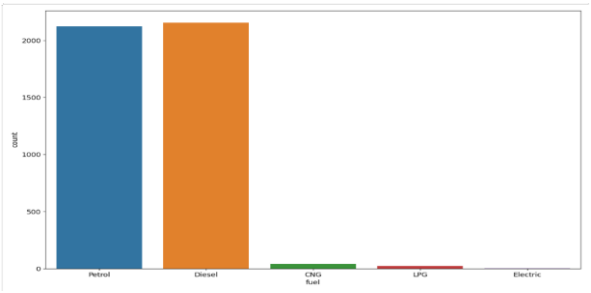
Ders Yürütücüsü: Arş. Gör. Kübra ADALI

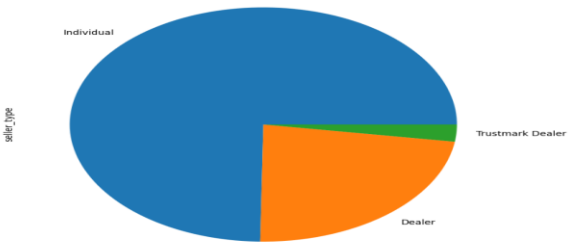
2023

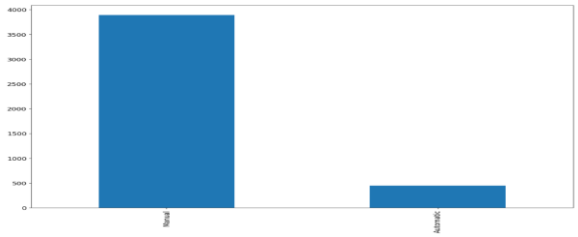
## VERİLERİN UYGUN GRAFİK TİPİNE GÖRE GÖRSELLEŞTİRİLMESİ



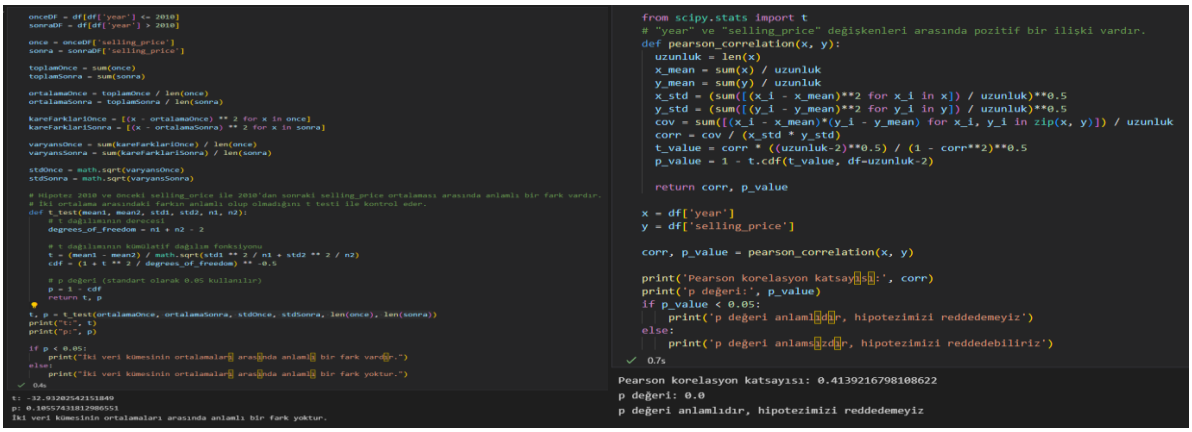
## Verilerin İstatistiksel Özetleri (Ortalama, Mod, Medyan, Açıklıklar, Standart Sapma)

```
# standart sapmalar
print(stdYear)
print(stdSellingPrice)
print(stdKmDriven)
0.4s
4.214858273043597
579482.0750218795
46638.72813954532

# varyanslar
print(varyansYear)
print(varyansSellingPrice)
print(varyansKmDriven)
0.4s
17.765034926184485
3346415150772.28217
2175170962.4744163

# medyan
print(medYear)
print(medSellingPrice)
print(medKmDriven)
0.8s
2017.466)
300000, 162)
70000, 236)

# ortalamalar
print(ortalamaYear)
print(ortalamaSellingPrice)
print(ortalamaKmDriven)
0.4s
2013.0987834101383
504127.3117511521
66215.77741935484

# medyanlar
print(medyanYear)
print(medyanSellingPrice)
print(medyanKmDriven)
0.6s
2014.0
252500.0
80000.0
```

```
year = df['year']
selling_price = df['selling_price']
km_driven = df['km_driven']

toplamYear = sum(year)
toplamSellingPrice = sum(selling_price)
toplamKmDriven = sum(km_driven)

ortalamaYear = toplamYear / len(year)
ortalamaSellingPrice = toplamSellingPrice / len(selling_price)
ortalamaKmDriven = toplamKmDriven / len(km_driven)

kareFarklariYear = [(x - ortalamaYear) ** 2 for x in year]
kareFarklariSellingPrice = [(x - ortalamaSellingPrice) ** 2 for x in selling_price]
kareFarklariKmDriven = [(x - ortalamaKmDriven) ** 2 for x in km_driven]

varyansYear = sum(kareFarklariYear) / len(year)
varyansSellingPrice = sum(kareFarklariSellingPrice) / len(selling_price)
varyansKmDriven = sum(kareFarklariKmDriven) / len(km_driven)

stdYear = math.sqrt(varyansYear)
stdSellingPrice = math.sqrt(varyansSellingPrice)
stdKmDriven = math.sqrt(varyansKmDriven)

from collections import Counter

veriSayisiYear = Counter(year)
veriSayisiSellingPrice = Counter(selling_price)
veriSayisiKmDriven = Counter(km_driven)

modYear = veriSayisiYear.most_common(1)[0]
modSellingPrice = veriSayisiSellingPrice.most_common(1)[0]
modKmDriven = veriSayisiKmDriven.most_common(1)[0]

year.sort_values()
selling_price.sort_values()
km_driven.sort_values()

if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanYear = (year[len(year) // 2] + year[len(year) // 2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanYear = year[len(year) // 2]

if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanSellingPrice = (selling_price[len(selling_price) // 2] + selling_price[len(selling_price) // 2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanSellingPrice = selling_price[len(selling_price) // 2]

if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanKmDriven = (km_driven[len(km_driven) // 2] + km_driven[len(km_driven) // 2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanKmDriven = km_driven[len(km_driven) // 2]
```

## Verinin 2 Alt Kümeye Ayrılarak Hipotez Oluşturulması ve Test Edilmesi, Veriden Seçilen 2 Değişken için Korelasyon Hesaplanması ve Test Edilmesi

```
t = (x̄1 - x̄2) / sqrt[(s1^2 / n1) + (s2^2 / n2)]
x̄1 ve x̄2: ilk ve ikinci grup ortalamaları
s1 ve s2: ilk ve ikinci grup standart sapmaları
n1 ve n2: ilk ve ikinci grup örneklem büyüklükleri
f(t) = (k!) / [((k/2)!)^2 * (n - k - 1)!] * (1 + t^2 / k)^(-(n - k - 1)/2)
t: test istatistiği olan t değeri
k: önkoşul dağılımı t testi için uygun olmayan veri setlerinde, t dağılımının kısmi derece serbestlikleri sayısı
n: örneklem büyüklüğü
p = ∫f(t) dt
f(t): t dağılımının olasılık fonksiyonu
t: test istatistiği olan t değeri
∫: integral sembolü, t dağılımının olasılık fonksiyonunun integralini almayı gösterir.
```



## KODLAR

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
df = pd.read_csv("CAR DETAILS FROM CAR DEKHO.csv")
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
sns.countplot(x='fuel', data=df)
plt.show()
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
df['seller_type'].value_counts().plot(kind='pie')
plt.show()
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
df['transmission'].value_counts().plot(kind='bar', stacked=True)
plt.show()
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
sns.boxplot(x='owner', y='selling_price', data=df)
plt.show()
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
sns.scatterplot(x='year', y='selling_price', data=df)
plt.show()
plt.figure(figsize=(15, 10))
plt.rcParams.update({'font.size': 12})
sns.histplot(df['km_driven'])
plt.show()
year = df['year']
selling_price = df['selling_price']
km_driven = df['km_driven']
toplamYear = sum(year)
toplamSellingPrice = sum(selling_price)
toplamKmDriven = sum(km_driven)
ortalamaYear = toplamYear / len(year)
```

```
ortalamaSellingPrice = toplamSellingPrice / len(selling_price)
ortalamaKmDriven = toplamKmDriven / len(km_driven)
kareFarklariYear = [(x - ortalamaYear) ** 2 for x in year]
kareFarklariSellingPrice = [(x - ortalamaSellingPrice) ** 2 for x in selling_price]
kareFarklariKmDriven = [(x - ortalamaKmDriven) ** 2 for x in km_driven]
varyansYear = sum(kareFarklariYear) / len(year)
varyansSellingPrice = sum(kareFarklariSellingPrice) / len(selling_price)
varyansKmDriven = sum(kareFarklariKmDriven) / len(km_driven)
stdYear = math.sqrt(varyansYear)
stdSellingPrice = math.sqrt(varyansSellingPrice)
stdKmDriven = math.sqrt(varyansKmDriven)
from collections import Counter
veriSayisiYear = Counter(year)
veriSayisiSellingPrice = Counter(selling_price)
veriSayisiKmDriven = Counter(km_driven)
modYear = veriSayisiYear.most_common(1)[0]
modSellingPrice = veriSayisiSellingPrice.most_common(1)[0]
modKmDriven = veriSayisiKmDriven.most_common(1)[0]
year.sort_values()
selling_price.sort_values()
km_driven.sort_values()
if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanYear = (year[len(year) // 2] + year[len(year) // 2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanYear = year[len(year) // 2]
if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanSellingPrice = (selling_price[len(selling_price) // 2] + selling_price[len(selling_price) //
2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanSellingPrice = selling_price[len(selling_price) // 2]
if len(year) % 2 == 0:
    # Veri kümesi çift sayıdaysa ortadaki iki değerin ortalaması
    medyanKmDriven = (km_driven[len(km_driven) // 2] + km_driven[len(km_driven) // 2 - 1]) / 2
else:
    # Veri kümesi tek sayıdaysa ortadaki değer
    medyanKmDriven = km_driven[len(km_driven) // 2]
onceDF = df[df['year'] <= 2010]
sonraDF = df[df['year'] > 2010]
once = onceDF['selling_price']
sonra = sonraDF['selling_price']
# Hipotez 2010 ve önceki selling_orice ile 2010'dan sonraki selling_price ortalaması arasında anlamlı
bir fark vardır.
toplamOnce = sum(once)
toplamSonra = sum(sonra)
ortalamaOnce = toplamOnce / len(once)
```

```
ortalamaSonra = toplamSonra / len(sonra)
kareFarklariOnce = [(x - ortalamaOnce) ** 2 for x in once]
kareFarklariSonra = [(x - ortalamaSonra) ** 2 for x in sonra]
varyansOnce = sum(kareFarklariOnce) / len(once)
varyansSonra = sum(kareFarklariSonra) / len(sonra)
stdOnce = math.sqrt(varyansOnce)
stdSonra = math.sqrt(varyansSonra)
# İki ortalama arasındaki farkın anlamlı olup olmadığını t testi ile kontrol eder.
def t_test(mean1, mean2, std1, std2, n1, n2):
    # t dağılımının derecesi
    degrees_of_freedom = n1 + n2 - 2
    # t dağılımının kümülatif dağılım fonksiyonu
    t = (mean1 - mean2) / math.sqrt(std1 ** 2 / n1 + std2 ** 2 / n2)
    cdf = (1 + t ** 2 / degrees_of_freedom) ** -0.5
    # p değeri (standart olarak 0.05 kullanılır)
    p = 1 - cdf
    return t, p
t, p = t_test(ortalamaOnce, ortalamaSonra, stdOnce, stdSonra, len(once), len(sonra))

print("t:", t)

print("p:", p)

if p < 0.05:
    print("İki veri kümesinin ortalamaları arasında anlamlı bir fark vardır.")
else:
    print("İki veri kümesinin ortalamaları arasında anlamlı bir fark yoktur.")
x = df['year']
y = df['selling_price']
from scipy.stats import t
def pearson_correlation(x, y):
  uzunluk = len(x)
  x_mean = sum(x) / uzunluk
  y_mean = sum(y) / uzunluk
  x_std = (sum([(x_i - x_mean)**2 for x_i in x]) / uzunluk)**0.5
  y_std = (sum([(y_i - y_mean)**2 for y_i in y]) / uzunluk)**0.5
  cov = sum([(x_i - x_mean)*(y_i - y_mean) for x_i, y_i in zip(x, y)]) / uzunluk
  corr = cov / (x_std * y_std)
  t_value = corr * ((uzunluk-2)**0.5) / (1 - corr**2)**0.5
  p_value = 1 - t.cdf(t_value, df=uzunluk-2)
  return corr, p_value
x = df['year']
y = df['selling_price']
corr, p_value = pearson_correlation(x, y)
print('Pearson korelasyon katsayısı:', corr)
print('p değeri:', p_value)
if p_value < 0.05:
    print('p değeri anlamlıdır, hipotezimizi reddedemeyiz')
else:
    print('p değeri anlamsızdır, hipotezimizi reddedebiliriz')
```