MELİKE ÇEKEN

“MAKİNE ÖĞRENMESİ İLE VERİ ANALİZİ”

Giriş

Bu çalışmada Kaggle üzerinden alınmış olan bir süpermarket satışlarının verileri üzerinden, Jupyter Notebook kullanarak makine öğrenmesi ile veri analizi yapılmıştır.

Veri ön işleme, tahmin ve görselleştirme çalışmaları yapılmıştır.

İÇİNDEKİLER

Giriş 2
Bölüm 1: Veri Ön İşleme 4
Bölüm 2: Tahmin 10
2.1. Gender & Branch 10
2.2. Product_line & City 14
2.3. Product_line & Customer_type 20
Bölüm 3: Görselleştirme 25

Bölüm 1: Veri Ön İşleme

Projeye başlarken öncelikle gerekli kütüphaneler import edilmiştir.

```
In [2]: import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        import numpy as np

        from sklearn.model_selection import train_test_split, GridSearchCV,cross_val_score
        from sklearn.metrics import mean_squared_error , r2_score

        from sklearn import model_selection
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.linear_model import LinearRegression
        from sklearn.feature_selection import f_regression

        from warnings import filterwarnings
        filterwarnings('ignore')

        df=pd.read_csv(r'C:\Users\DELL\Desktop\supermarket_sales.csv')
```

Veriler “supermarket_sales.csv” dosyasından alınmıştır.

```
In [190]: df.head()
```

Out[190]:

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
0	750-67-8428	1	1	1	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26.1415
1	226-31-3081	2	2	2	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3.8200
2	631-41-3108	1	1	2	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16.2155
3	123-19-1176	1	1	1	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23.2880
4	373-73-7910	1	1	2	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30.2085

Verilerin türü

```
In [4]: df.dtypes
```

Out[4]:

```
Invoice ID                 object
Branch                     object
City                       object
Customer type              object
Gender                     object
Product line               object
Unit price                float64
Quantity                    int64
Tax 5%                    float64
Total                     float64
Date                       object
Time                       object
Payment                    object
cogs                      float64
gross margin percentage   float64
gross income              float64
Rating                    float64
dtype: object
```

Verilerin bilgileri

```
In [5]: df.info
```

```
Out[5]: <bound method DataFrame.info of       Invoice ID Branch       City Customer type
        0    750-67-8428      A     Yangon        Member  Female
        1    226-31-3081      C  Naypyitaw        Normal  Female
        2    631-41-3108      A     Yangon        Normal    Male
        3    123-19-1176      A     Yangon        Member    Male
        4    373-73-7910      A     Yangon        Normal    Male
        ..           ...    ...        ...           ...     ...
        995  233-67-5758      C  Naypyitaw        Normal    Male
        996  303-96-2227      B   Mandalay        Normal  Female
        997  727-02-1313      A     Yangon        Member    Male
        998  347-56-2442      A     Yangon        Normal    Male
        999  849-09-3807      A     Yangon        Member  Female

                      Product line  Unit price  Quantity   Tax 5%      Total  \
        0        Health and beauty       74.69         7  26.1415   548.9715
        1    Electronic accessories       15.28         5   3.8200    80.2200
        2        Home and lifestyle       46.33         7  16.2155   340.5255
        3        Health and beauty       58.22         8  23.2880   489.0480
        4        Sports and travel       86.31         7  30.2085   634.3785
        ..                     ...         ...       ...      ...        ...
        995      Health and beauty       40.35         1   2.0175    42.3675
        996      Home and lifestyle      97.38        10  48.6900  1022.4900
        997      Food and beverages      31.84         1   1.5920    33.4320
        998      Home and lifestyle      65.82         1   3.2910    69.1110
        999     Fashion accessories      88.34         7  30.9190   649.2990

                 Date   Time      Payment    cogs  gross margin percentage  \
        0     1/5/2019  13:08      Ewallet  522.83                 4.761905
        1     3/8/2019  10:29         Cash   76.40                 4.761905
        2     3/3/2019  13:23  Credit card  324.31                 4.761905
        3    1/27/2019  20:33      Ewallet  465.76                 4.761905
        4     2/8/2019  10:37      Ewallet  604.17                 4.761905
        ..         ...    ...          ...     ...                      ...
        995  1/29/2019  13:46      Ewallet   40.35                 4.761905
        996   3/2/2019  17:16      Ewallet  973.80                 4.761905
        997   2/9/2019  13:22         Cash   31.84                 4.761905
        998  2/22/2019  15:33         Cash   65.82                 4.761905
        999  2/18/2019  13:28         Cash  618.38                 4.761905

             gross income  Rating
        0         26.1415     9.1
        1          3.8200     9.6
        2         16.2155     7.4
        3         23.2880     8.4
        4         30.2085     5.3
        ..            ...     ...
        995        2.0175     6.2
        996       48.6900     4.4
        997        1.5920     7.7
        998        3.2910     4.1
        999       30.9190     6.6

        [1000 rows x 17 columns]>
```

Data tanımlama

```
In [6]: df.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
Unit price	1000.0	55.672130	2.649463e+01	10.080000	32.875000	55.230000	77.935000	99.960000
Quantity	1000.0	5.510000	2.923431e+00	1.000000	3.000000	5.000000	8.000000	10.000000
Tax 5%	1000.0	15.379369	1.170883e+01	0.508500	5.924875	12.088000	22.445250	49.650000
Total	1000.0	322.966749	2.458853e+02	10.678500	124.422375	253.848000	471.350250	1042.650000
cogs	1000.0	307.587380	2.341765e+02	10.170000	118.497500	241.760000	448.905000	993.000000
gross margin percentage	1000.0	4.761905	6.131498e-14	4.761905	4.761905	4.761905	4.761905	4.761905
gross income	1000.0	15.379369	1.170883e+01	0.508500	5.924875	12.088000	22.445250	49.650000
Rating	1000.0	6.972700	1.718580e+00	4.000000	5.500000	7.000000	8.500000	10.000000

Sütun adlarını değiştirilmesi

Gerekli sütun adları güncellenmiştir.

```
In [11]: df.rename(columns = {"Product line": "Product_line"
                              }, inplace = True)
```

```
In [17]: df.rename(columns = {"Tax 5%": "Tax"
                              }, inplace = True)
```

```
In [21]: df.rename(columns = {"Unit price": "Unit_price"
                              }, inplace = True)
```

```
In [34]: df.rename(columns = {"Customer type": "Customer_type"
                              } , inplace = True)
```

String ifadeleri sayısal ifadelere çevirme

Makine öğrenmesi için veri ön işlemesi aşamasında string ifadeler sayısal ifadelere çevrilmiştir.

Bu ifadeler sırasıyla “Gender”, “Product_line”, “Payment”, “Customer_type”, “City”, “Branch” sütunlarıdır.

Gender

```
In [25]: df["Gender"] = df.Gender.map({"Female" : 1, "Male":2})
```

```
In [26]: df.head()
```

Out[26]:

	Invoice ID	Branch	City	Customer type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
0	750-67-8428	A	Yangon	Member	1	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.1415
1	226-31-3081	C	Naypyitaw	Normal	1	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.8200
2	631-41-3108	A	Yangon	Normal	2	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16.2155
3	123-19-1176	A	Yangon	Member	2	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	4.761905	23.2880
4	373-73-7910	A	Yangon	Normal	2	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	4.761905	30.2085

Product_line

Öncelikle “Product_line” unique ifadeleri bulunmuştur.

```
In [28]: df.Product_line.unique()
Out[28]: array(['Health and beauty', 'Electronic accessories',
                'Home and lifestyle', 'Sports and travel', 'Food and beverages',
                'Fashion accessories'], dtype=object)
```

Sonrasında string ifadelere sayısal değer atanmıştır.

```
In [29]: df["Product_line"] = df.Product_line.map({"Health and beauty": 1, "Electronic accessories" :2, "Home and lifestyle": 3,
                                                   "Sports and travel": 4, "Food and beverages":5, "Fashion accessories" : 6})
```

```
In [30]: df.head()
```

Out[30]:

	Invoice ID	Branch	City	Customer type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
0	750-67-8428	A	Yangon	Member	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.1415
1	226-31-3081	C	Naypyitaw	Normal	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.8200
2	631-41-3108	A	Yangon	Normal	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16.2155
3	123-19-1176	A	Yangon	Member	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	4.761905	23.2880
4	373-73-7910	A	Yangon	Normal	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	4.761905	30.2085

Payment

Öncelikle “Payment” unique ifadeleri bulunmuştur.

```
In [32]: df.Payment.unique()
Out[32]: array(['Ewallet', 'Cash', 'Credit card'], dtype=object)
```

Sonrasında string ifadelere sayısal değer atanmıştır.

```
In [33]: df["Payment"] =df.Payment.map({"Ewallet": 1, "Cash" :2, "Credit card":3})
```

```
In [35]: df.head()
```

Out[35]:

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	ir
0	750-67-8428	A	Yangon	Member	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26
1	226-31-3081	C	Naypyitaw	Normal	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3
2	631-41-3108	A	Yangon	Normal	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16
3	123-19-1176	A	Yangon	Member	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23
4	373-73-7910	A	Yangon	Normal	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30

Customer_type

Öncelikle “Customer_type” unique ifadeleri bulunmuştur.

```
In [36]: df.Customer_type.unique()
Out[36]: array(['Member', 'Normal'], dtype=object)
```

Sonrasında string ifadelere sayısal değer atanmıştır.

```
In [37]: df["Customer_type"] = df.Customer_type.map({"Member": 1, "Normal" :2})
In [38]: df.head()
```

Out[38]:

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	ir
0	750-67-8428	A	Yangon	1	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26
1	226-31-3081	C	Naypyitaw	2	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3
2	631-41-3108	A	Yangon	2	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16
3	123-19-1176	A	Yangon	1	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23
4	373-73-7910	A	Yangon	2	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30

City

Öncelikle “City” unique ifadeleri bulunmuştur.

```
In [39]: df.City.unique()
Out[39]: array(['Yangon', 'Naypyitaw', 'Mandalay'], dtype=object)
```

Sonrasında string ifadelere sayısal değer atanmıştır.

```
In [40]: df["City"] = df.City.map({"Yangon": 1, "Naypyitaw" :2, "Mandalay" :3})
In [41]: df.head()
```

Out[41]:

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
0	750-67-8428	A	1	1	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26.1415
1	226-31-3081	C	2	2	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3.8200
2	631-41-3108	A	1	2	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16.2155
3	123-19-1176	A	1	1	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23.2880
4	373-73-7910	A	1	2	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30.2085

Branch

Öncelikle “Branch” unique ifadeleri bulunmuştur.

```
In [42]: df.Branch.unique()
Out[42]: array(['A', 'C', 'B'], dtype=object)
```

Sonrasında string ifadelere sayısal değer atanmıştır.

```
In [43]: df["Branch"] = df.Branch.map({"A": 1, "C" :2, "B" :3})
In [44]: df.head(1000)
```

Out[44]:

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gr inc
0	750-67-8428	1	1	1	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26.1
1	226-31-3081	2	2	2	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3.8
2	631-41-3108	1	1	2	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16.2
3	123-19-1176	1	1	1	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23.2
4	373-73-7910	1	1	2	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	233-67-5758	2	2	2	2	1	40.35	1	2.0175	42.3675	1/29/2019	13:46	1	40.35	4.761905	2.0
996	303-96-2227	3	3	2	1	3	97.38	10	48.6900	1022.4900	3/2/2019	17:16	1	973.80	4.761905	48.6
997	727-02-1313	1	1	1	2	5	31.84	1	1.5920	33.4320	2/9/2019	13:22	2	31.84	4.761905	1.5
998	347-56-2442	1	1	2	2	3	65.82	1	3.2910	69.1110	2/22/2019	15:33	2	65.82	4.761905	3.2
999	849-09-3807	1	1	1	1	6	88.34	7	30.9190	649.2990	2/18/2019	13:28	2	618.38	4.761905	30.9

1000 rows × 17 columns

Veri ön işleme öncesi veri seti

	Invoice ID	Branch	City	Customer type	Gender	Product_line	Unit price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income	Ra
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.1415	
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.8200	
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16.2155	
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	4.761905	23.2880	
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	4.761905	30.2085	

Veri ön işleme sonrası veri seti

	Invoice ID	Branch	City	Customer_type	Gender	Product_line	Unit_price	Quantity	Tax	Total	Date	Time	Payment	cogs	gross margin percentage	gross income
0	750-67-8428	1	1	1	1	1	74.69	7	26.1415	548.9715	1/5/2019	13:08	1	522.83	4.761905	26.1415
1	226-31-3081	2	2	2	1	2	15.28	5	3.8200	80.2200	3/8/2019	10:29	2	76.40	4.761905	3.8200
2	631-41-3108	1	1	2	2	3	46.33	7	16.2155	340.5255	3/3/2019	13:23	3	324.31	4.761905	16.2155
3	123-19-1176	1	1	1	2	1	58.22	8	23.2880	489.0480	1/27/2019	20:33	1	465.76	4.761905	23.2880
4	373-73-7910	1	1	2	2	4	86.31	7	30.2085	634.3785	2/8/2019	10:37	1	604.17	4.761905	30.2085

Bölüm 2: Tahmin

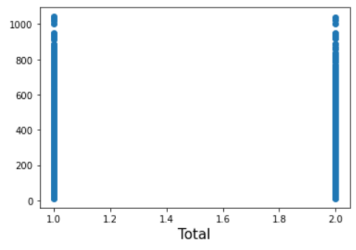
2.1. Gender & Branch

Makine öğrenmesi ile tahmin aşaması için öncelikle bağımlı ve bağımsız değişkenler belirlenir.

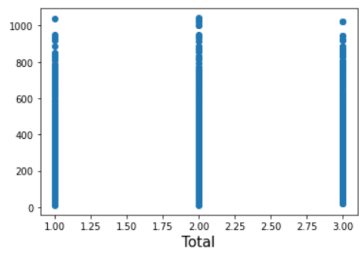
```
In [46]: # bağımlı ve bağımsız değişken

In [47]: #iki bağımsız değişken Gender, Product_line

In [48]: x = df [['Gender','Branch']]

In [49]: #bağımlı değişken total

In [51]: y = df ['Total']
```

Saçılım grafikleri

```
In [53]: plt.scatter(x['Gender'], y)

         plt.xlabel('Gender', fontsize=15)
         plt.xlabel('Total', fontsize=15)

         plt.show()
```




```
In [54]: plt.scatter(x['Branch'], y)

         plt.xlabel('Branch', fontsize=15)
         plt.xlabel('Total', fontsize=15)

         plt.show()
```



Çoklu doğrusal regresyon kullanılacağı için doğrusal regresyon modeli oluşturulur.

```
In [55]: reg = LinearRegression()
         reg.fit(x,y)

Out[55]: LinearRegression()
```

Sonrasında sırasıyla regresyonun kesim noktasını bulma

```
In [56]: #regresyonun kesim noktası

In [57]: reg.intercept_

Out[57]: 352.00029344256734
```

regresyonun kat sayılarını bulma

```
In [58]: #regresynun kat sayıları

In [59]: reg.coef_

Out[59]: array([-24.23310318,   3.66058094])
```

(Kat sayılar “Gender” ve “Branch” arasında zıt yönlü bir ilişki olduğunu gösterir.)

sklearn için de r karesini hesaplama

```
In [ ]: #sklearn içinde r karesini hesaplamak

In [60]: reg.score(x,y) #r kare

Out[60]: 0.002594450424785233
```

ve düzeltilmiş r kare metriğinin oluşturulmasını kolaylaştırmak için x in şeklini bulma

```
In [61]: x.shape

Out[61]: (1000, 2)
```

adımları uygulanır.

Çoklu doğrusal regresyon olduğu için düzeltilmiş r kareyi de buluyoruz. Sebebi modele değişken ekledikçe r kare artmaktadır ve bu artışın sanal olup olmadığını anlamak için gereklidir.

```
In [62]: #düzeltilmiş r kareyi bulmak için, r kareyi, gözlem sayısını ve özellik sayısını bilmeliyiz
         r2= reg.score(x,y)

         #gözlem sayısı  (n), eksen 0 boyunca olan şekildir
         n=x.shape[0]

         #özellik sayısı (öngörücüler , p) eksen 1 boyunca şeklidir
         p= x.shape[1]

         #düzeltilmiş r kareyi aşağıdaki formulle buluruz
         düzeltilmis_r2= 1-(1-r2)*(n-1)/(n-p-1)
         düzeltilmis_r2

Out[62]: 0.0005936368850154583
```

R kare : 0.002594450424785233
Düzeltilmiş R kare : 0.0005936368850154583

R karenin yüksek olması regresyon model uyumunun iyi olduğunu gösterir.

Sonrasında tahmin aşamasına geçebiliriz.

Branch a ve cinsiyeti kadın olan birinin ortalama totali tahmini 331,4277712'dir.

```
In [63]: reg.predict([[1,1]])
Out[63]: array([331.4277712])
```

Branch a ve cinsiyeti erkek olan birinin ortalama totali tahmini 335,08835213'dir.

```
In [64]: reg.predict([[1,2]])
Out[64]: array([335.08835213])
```

Branch c ve cinsiyeti kadın olan birinin ortalama totali tahmini 307,19466802'dir.

```
In [65]: reg.predict([[2,1]])
Out[65]: array([307.19466802])
```

Branch c ve cinsiyeti erkek olan birinin ortalama totali tahmini 310,85524895'dir.

```
In [66]: reg.predict([[2,2]])
Out[66]: array([310.85524895])
```

Branch b ve cinsiyeti kadın olan birinin ortalama totali tahmini 282,96156484'dir.

```
In [67]: reg.predict([[3,1]])
Out[67]: array([282.96156484])
```

Branch b ve cinsiyeti erkek olan birinin ortalama totali tahmini 286,62214578'dir.

```
In [68]: reg.predict([[3,2]])
Out[68]: array([286.62214578])
```

Yaptığımız tahminler sonucunda feature selection yöntemiyle p değerleri hesaplanır. Bu değerleri hesaplamak regresyonumuz için en doğru özellikleri seçmemizi sağlar.

İlk array regresyondaki her regresyon için f istatistiği içerir, ikincisi ise bu f istatistiklerinin p değerleridir.

```
In [72]: f_regression(x,y)
Out[72]: (array([2.44649215, 0.16384422]), array([0.11810485, 0.68572835]))
```

P değerleri ile ilgilendiğimiz için p değerlerini alıyoruz.

```
In [73]: p_values= f_regression(x,y) [1]
         p_values
Out[73]: array([0.11810485, 0.68572835])

In [74]: p_values.round(3)
Out[74]: array([0.118, 0.686])
```

Özet tablo

```
In [75]: reg_summary= pd.DataFrame(data= x.columns.values, columns= ['özellik'])
         reg_summary ['Katsayılar'] = reg.coef_
         reg_summary ['p-değerleri'] = p_values.round(3)
         reg_summary
```

Out[75]:

	özellik	Katsayılar	p-değerleri
0	Gender	-24.233103	0.118
1	Branch	3.660581	0.686

Gender ve Branch değerlerinin p değerlerinin düşük olması istatiksel olarak anlamlı olduğunu gösterir. Datadan çıkarılmamalıdır.

Mevcut dataya yeni data ekleyip tahmin yapmak istersek şu şekilde yapabiliriz.

```
In [69]: yeni_df =pd.DataFrame({'Branch': [4,5,6], 'Gender': [1,2,3] })
         yeni_df
```

Out[69]:

	Branch	Gender
0	4	1
1	5	2
2	6	3

```
In [70]: reg.predict(yeni_df).round(1)
```

```
Out[70]: array([258.7, 238.2, 217.6])
```

```
In [71]: yeni_df['Tahmini Total'] =reg.predict(yeni_df)
         yeni_df
```

Out[71]:

	Branch	Gender	Tahmini Total
0	4	1	258.728462
1	5	2	238.155939
2	6	3	217.583417

2.2. Product_line & City

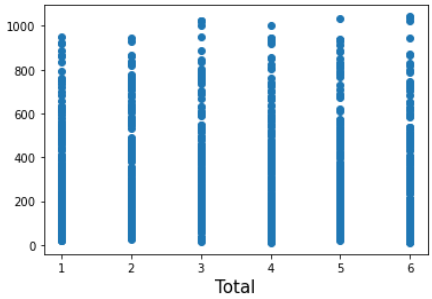
Bağımlı değişkenler “Product_line” ve “City” ve bağımsız değişken “Total” dir.

```
In [107]: x = df [['Product_line','City']]
```

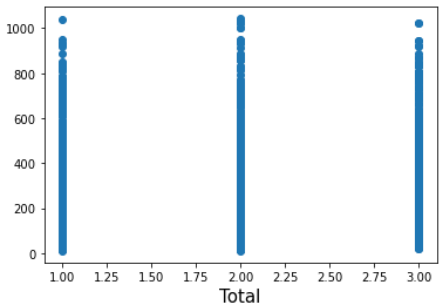
```
In [108]: y = df ['Total']
```

Saçılım grafikleri

```
In [94]: plt.scatter(x['Product_line'], y)

         plt.xlabel('Product_line', fontsize=15)
         plt.xlabel('Total', fontsize=15)

         plt.show()
```




```
In [109]: plt.scatter(x['City'], y)

          plt.xlabel('City', fontsize=15)
          plt.xlabel('Total', fontsize=15)

          plt.show()
```



Çoklu doğrusal regrosyon kullanılacağı için doğrusal regresyon modeli oluşturulur.

```
In [110]: reg = LinearRegression()
          reg.fit(x,y)

Out[110]: LinearRegression()
```

Sonrasında sırasıyla regresyonun kesim noktasını bulma

```
In [111]: reg.intercept_

Out[111]: 324.80319137165895
```

regresyonun kat sayılarını bulma

```
In [112]: reg.coef_

Out[112]: array([-2.69406845,  3.91172603])
```

(Kat sayılar “Product_line” ve “City” arasında zıt yönlü bir ilişki olduğunu gösterir.)

sklearn için de r karesini hesaplama

```
In [113]: reg.score(x,y)
Out[113]: 0.00051408058394442O7
```

ve düzeltilmiş r kare metriğinin oluşturulmasını kolaylaştırmak için x in şekli

```
In [114]: x.shape
Out[114]: (1000, 2)
```

adımları uygulanır.

Çoklu doğrusal regresyon olduğu için düzeltilmiş r kareyi de buluyoruz. Sebebi modele değişken ekledikçe r kare artmaktadır ve bu artışın sanal olup olmadığını anlamak için gereklidir.

```
In [115]: #düzeltilmiş r kareyi bulmak için, r kareyi, gözlem sayısını ve özellik sayısını bilmeliyiz
          r2= reg.score(x,y)

          #gözlem sayısı  (n), eksen 0 boyunca olan şekildir
          n=x.shape[0]

          #özellik sayısı (öngörücüler , p) eksen 1 boyunca şeklidir
          p= x.shape[1]

          #düzeltilmiş r kareyi aşağıdaki formulle buluruz
          düzeltilmis_r2= 1-(1-r2)*(n-1)/(n-p-1)
          düzeltilmis_r2
Out[115]: -0.0014909062152854702
```

R kare : 0.00005140805839444207
Düzeltilmiş R kare : -0.0014909062152854702

R karenin yüksek olması regresyon model uyumunun iyi olduğunu gösterir.

Sonrasında tahmin aşamasına geçebiliriz.

Yangon şehrinden "Health and beauty" kategorisinden alış veriş yapan birinin ortalama totali 326.02084896’dır.

```
In [117]: reg.predict([[1,1]])
Out[117]: array([326.02084896])
```

Yangon şehrinden "Electronic accessories" kategorisinden alış veriş yapan birinin ortalama totali 323.32678051'dir.

```
In [129]: reg.predict([[2,1]])
Out[129]: array([323.32678051])
```

Yangon şehrinden "Home and lifestyle" kategorisinden alış veriş yapan birinin ortalama totali 320.62271206'dır.

```
In [130]: reg.predict([[3,1]])
Out[130]: array([320.63271206])
```

Yangon şehrinden "Sports and travel" kategorisinden alış veriş yapan birinin ortalama totali 317.93864362'dir.

```
In [131]: reg.predict([[4,1]])
Out[131]: array([317.93864362])
```

Yangon şehrinden "Food and beverages" kategorisinden alış veriş yapan birinin ortalama totali 315.24457517'dir.

```
In [132]: reg.predict([[5,1]])
Out[132]: array([315.24457517])
```

Yangon şehrinden "Fashion accessories" kategorisinden alış veriş yapan birinin ortalama totali 312,55050672'dir.

```
In [133]: reg.predict([[6,1]])
Out[133]: array([312.55050672])
```

Naypyitaw şehrinden "Health and beauty" kategorisinden alış veriş yapan birinin ortalama totali 329.93257499'dur.

```
In [134]: reg.predict([[1,2]])
Out[134]: array([329.93257499])
```

Naypyitaw şehrinden "Electronic accessories" kategorisinden alış veriş yapan birinin ortalama totali 327.23850654'dir.

```
In [136]: reg.predict([[2,2]])
Out[136]: array([327.23850654])
```

Naypyitaw şehrinden "Home and lifestyle" kategorisinden alış veriş yapan birinin ortalama totali 324.5444381'dir.

```
In [137]: reg.predict([[3,2]])
Out[137]: array([324.5444381])
```

Naypyitaw şehrinden "Sports and travel" kategorisinden alış veriş yapan birinin ortalama totali 321.85036965'tir.

```
In [138]: reg.predict([[4,2]])
Out[138]: array([321.85036965])
```

Naypyitaw şehrinden "Food and beverages" kategorisinden alış veriş yapan birinin ortalama totali 319.1563012'dir.

```
In [139]: reg.predict([[5,2]])
Out[139]: array([319.1563012])
```

Naypyitaw şehrinden "Fashion accessories" kategorisinden alış veriş yapan birinin ortalama totali 316.46223276'dir.

```
In [140]: reg.predict([[6,2]])
Out[140]: array([316.46223276])
```

Mandalay şehrinden "Health and beauty" kategorisinden alış veriş yapan birinin ortalama totali 333.84430103'tür.

```
In [141]: reg.predict([[1,3]])
Out[141]: array([333.84430103])
```

Mandalay şehrinden "Electronic accessories" kategorisinden alış veriş yapan birinin ortalama totali 331.15023258'dir.

```
In [158]: reg.predict([[2,3]])
Out[158]: array([331.15023258])
```

Mandalay şehrinden "Home and lifestyle" kategorisinden alış veriş yapan birinin ortalama totali 328.45616413'tür.

```
In [142]: reg.predict([[3,3]])
Out[142]: array([328.45616413])
```

Mandalay şehrinden "Sports and travel" kategorisinden alış veriş yapan birinin ortalama totali 325.762009568'dir.

```
In [143]: reg.predict([[4,3]])
Out[143]: array([325.76209568])
```

Mandalay şehrinden "Food and beverages" kategorisinden alış veriş yapan birinin ortalama totali 323.06802724'tür.

```
In [144]: reg.predict([[5,3]])
Out[144]: array([323.06802724])
```

Mandalay şehrinden "Fashion accessories" kategorisinden alış veriş yapan birinin ortalama totali 320.37395879'dur.

```
In [145]: reg.predict([[6,3]])
Out[145]: array([320.37395879])
```

Yaptığımız tahminler sonucunda feature selection yöntemiyle p değerleri hesaplanır. Bu değerleri hesaplamak regresyonumuz için en doğru özellikleri seçmemizi sağlar.

```
In [154]: p_values= f_regression(x,y) [1]
          p_values
Out[154]: array([0.5580581 , 0.68572835])

In [155]: p_values.round(3)
Out[155]: array([0.558, 0.686])
```

Özet Tablo

Product_line ve City değerlerinin p değerlerinin düşük olması istatiksel olarak anlamlı olduğunu gösterir. Datadan çıkarılmamalıdır.

```
In [156]: reg_summary= pd.DataFrame(data= x.columns.values, columns= ['özellik'])
          reg_summary ['Katsayılar'] = reg.coef_
          reg_summary ['p-değerleri'] = p_values.round(3)
          reg_summary
```

Out[156]:

	özellik	Katsayılar	p-değerleri
0	Product_line	-2.694068	0.558
1	City	3.911726	0.686

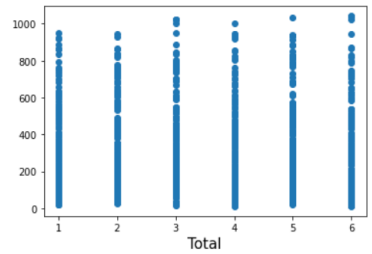
2.3. Product_line & Customer_type

Bağımlı değişkenler “Product_line” ve “Customer_type” ve bağımsız değişken “Total” dir.

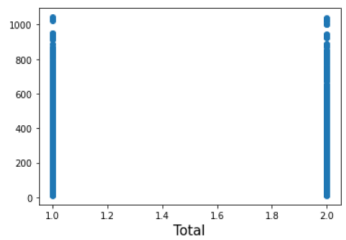
```
In [160]: x = df [['Product_line', 'Customer_type']]

In [161]: y = df ['Total']
```

Saçılım Grafikleri

```
In [162]: plt.scatter(x['Product_line'], y)

          plt.xlabel('Product_line', fontsize=15)
          plt.xlabel('Total', fontsize=15)

          plt.show()
```




```
In [163]: plt.scatter(x['Customer_type'], y)

          plt.xlabel('Customer_type', fontsize=15)
          plt.xlabel('Total', fontsize=15)

          plt.show()
```



Çoklu doğrusal regresyon kullanılacağı için doğrusal regresyon modeli oluşturulur.

```
In [164]: reg = LinearRegression()
          reg.fit(x,y)

Out[164]: LinearRegression()
```

Sonrasında sırasıyla regresyonun kesim noktasını bulma

```
In [165]: reg.intercept_

Out[165]: 347.61673123796913
```

regresyonun kat sayılarını bulma

```
In [166]: reg.coef_

Out[166]: array([-2.74329676, -9.90356212])
```

(Kat sayılar “Product_line” ve “Customer_type”” arasında doğru yönlü bir ilişki olduğunu gösterir.)

sklearn için de r karesini hesaplama

```
In [167]: reg.score(x,y)

Out[167]: 0.0007495890383953929
```

ve düzeltilmiş r kare metriğinin oluşturulmasını kolaylaştırmak için x in şekli

```
In [168]: x.shape
Out[168]: (1000, 2)
```

adımları uygulanır.

Çoklu doğrusal regresyon olduğu için düzeltilmiş r kareyi de buluyoruz. Sebebi modele değişken ekledikçe r kare artmaktadır ve bu artışın sanal olup olmadığını anlamak için gereklidir.

```
In [169]: #düzeltilmiş r kareyi bulmak için, r kareyi, gözlem sayısını ve özellik sayısını bilmeliyiz
          r2= reg.score(x,y)

          #gözlem sayısı  (n), eksen 0 boyunca olan şekildir
          n=x.shape[0]

          #özellik sayısı (öngörücüler , p) eksen 1 boyunca şeklidir
          p= x.shape[1]

          #düzeltilmiş r kareyi aşağıdaki formulle buluruz
          düzeltilmis_r2= 1-(1-r2)*(n-1)/(n-p-1)
          düzeltilmis_r2
Out[169]: -0.0012549253266229687
```

R kare : 0.0007495890383953923
Düzeltilmiş R kare : -0.00125492532662

R karenin yüksek olması regresyon model uyumunun iyi olduğunu gösterir.

Sonrasında tahmin aşamasına geçebiliriz.

“Member” bir müşteri "Health and beauty" kategorisinden alış veriş yaparsa ortalama totali 324.96987236’dır.

```
In [171]: reg.predict([[1,1]])
Out[171]: array([334.96987236])
```

“Member” bir müşteri "Electronic accessories" kategorisinden alış veriş yaparsa ortalama totali 332.2265756’dır.

```
In [172]: reg.predict([[2,1]])
Out[172]: array([332.2265756])
```

“Member” bir müşteri "Home and lifestyle" kategorisinden alış veriş yaparsa ortalama totali 329.48327884’tür.

```
In [173]: reg.predict([[3,1]])
Out[173]: array([329.48327884])
```

“Member” bir müşteri "Sports and travel" kategorisinden alış veriş yaparsa ortalama totali 326.73998208’dir.

```
In [174]: reg.predict([[4,1]])
Out[174]: array([326.73998208])
```

“Member” bir müşteri "Food and beverages" kategorisinden alış veriş yaparsa ortalama totali 323.99668532’dir.

```
In [175]: reg.predict([[5,1]])
Out[175]: array([323.99668532])
```

“Member” bir müşteri "Fashion accessories" kategorisinden alış veriş yaparsa ortalama totali 321.25338856’dır.

```
In [176]: reg.predict([[6,1]])
Out[176]: array([321.25338856])
```

“Normal” bir müşteri "Health and beauty" kategorisinden alış veriş yaparsa ortalama totali 325.06631023’tür..

```
In [177]: reg.predict([[1,2]])
Out[177]: array([325.06631023])
```

“Normal” bir müşteri "Electronic accessories" kategorisinden alış veriş yaparsa ortalama totali 322.32301347’dir.

```
In [178]: reg.predict([[2,2]])
Out[178]: array([322.32301347])
```

“Normal” bir müşteri "Home and lifestyle" kategorisinden alış veriş yaparsa ortalama totali 319.57971672’dir

```
In [179]: reg.predict([[3,2]])
Out[179]: array([319.57971672])
```

“Normal” bir müşteri "Sports and travel" kategorisinden alış veriş yaparsa ortalama totali 316.83641996’dır.

```
In [180]: reg.predict([[4,2]])
Out[180]: array([316.83641996])
```

“Normal” bir müşteri "Food and beverages" kategorisinden alış veriş yaparsa ortalama totali 314.0931232’dir.

```
In [181]: reg.predict([[5,2]])
Out[181]: array([314.0931232])
```

“Normal” bir müşteri "Fashion accessories" kategorisinden alış veriş yaparsa ortalama totali 311.34982644’tür.

```
In [182]: reg.predict([[6,2]])
Out[182]: array([311.34982644])
```

Yaptığımız tahminler sonucunda feature selection yöntemiyle p değerleri hesaplanır. Bu değerleri hesaplamak regresyonumuz için en doğru özellikleri seçmemizi sağlar.

```
In [183]: p_values= f_regression(x,y) [1]
          p_values
Out[183]: array([0.5580581 , 0.53439496])

In [184]: p_values.round(3)
Out[184]: array([0.558, 0.534])
```

Özet Tablo

```
In [185]: reg_summary= pd.DataFrame(data= x.columns.values, columns= ['özelli
          reg_summary ['Katsayılar'] = reg.coef_
          reg_summary ['p-değerleri'] = p_values.round(3)
          reg_summary
```

Out[185]:

	özellik	Katsayılar	p-değerleri
0	Product_line	-2.743297	0.558
1	Customer_type	-9.903562	0.534

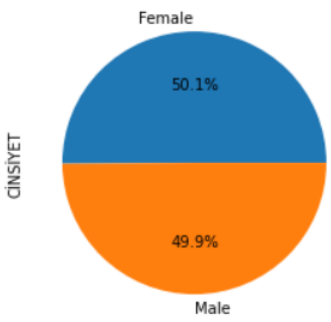
Product_line ve Customer_type değerlerinin p değerlerinin düşük olması istatiksel olarak anlamlı olduğunu gösterir. Datadan çıkarılmamalıdır.

Bölüm 3: Görselleştirme

Verilere görselleştirme işlemi uygulanmıştır.

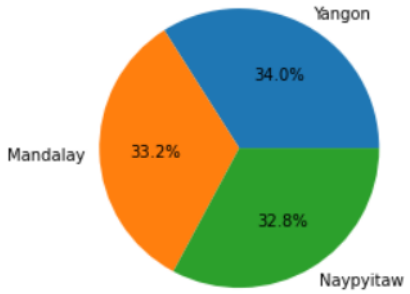
1) Cinsiyet

```
In [7]: df.Gender.value_counts().plot(kind = "pie", autopct = "%.1f%%")
        plt.ylabel("CİNSİYET");
```



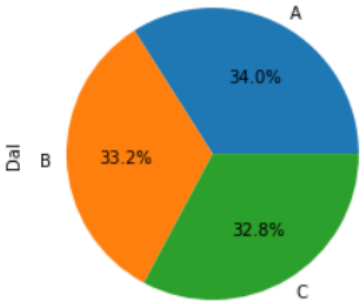
2) City

```
In [8]: df.City.value_counts().plot(kind = "pie", autopct = "%.1f%%")
        plt.ylabel("");
```



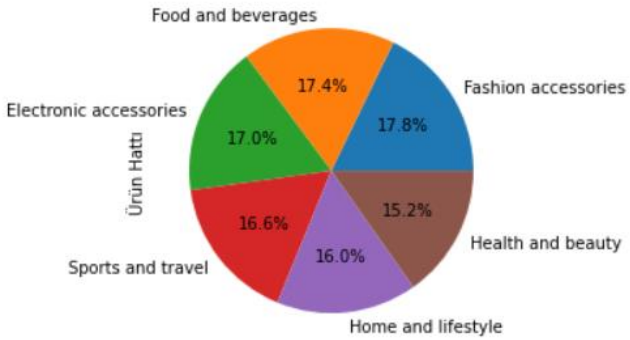
3) Branch

```
In [9]: df.Branch.value_counts().plot(kind = "pie", autopct = "%.1f%%")
        plt.ylabel("Dal");
```



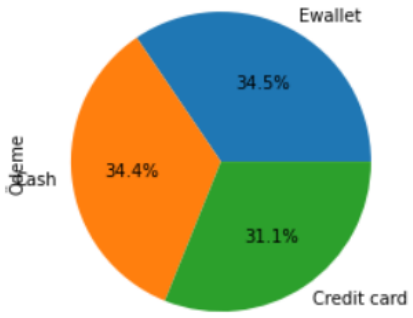
4) Product_line

```
In [12]: df.Product_line.value_counts().plot(kind = "pie", autopct = "%.1f%%")
         plt.ylabel("Ürün Hattı");
```



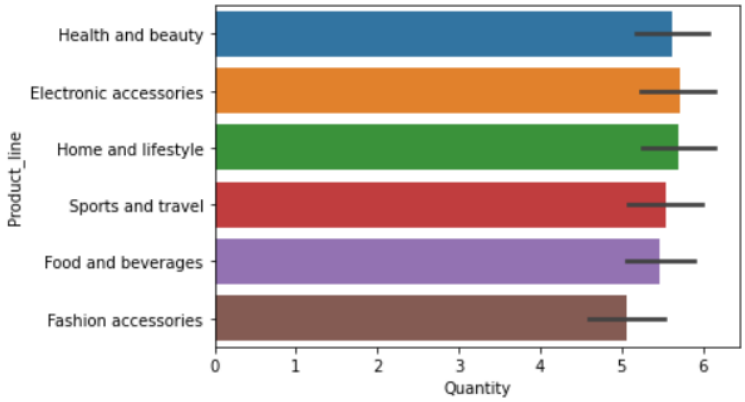
5) Payment

```
In [13]: df.Payment.value_counts().plot(kind = "pie", autopct = "%.1f%%")
         plt.ylabel("Ödeme");
```



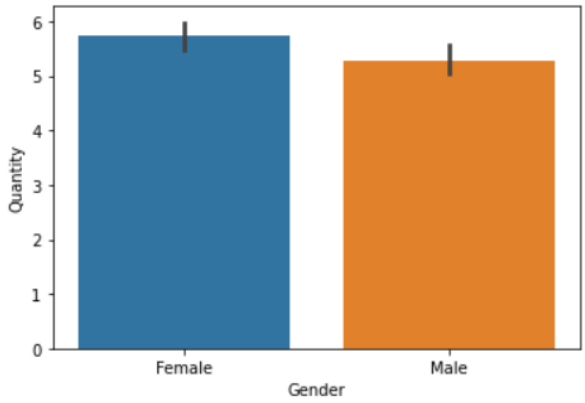
6) Quantity & Product_line

```
In [14]: sns.barplot(x = "Quantity" , y= "Product_line" , data =df);
```



7) Gender & Quantity

```
In [15]: sns.barplot(x = "Gender" , y= "Quantity" , data =df);
```

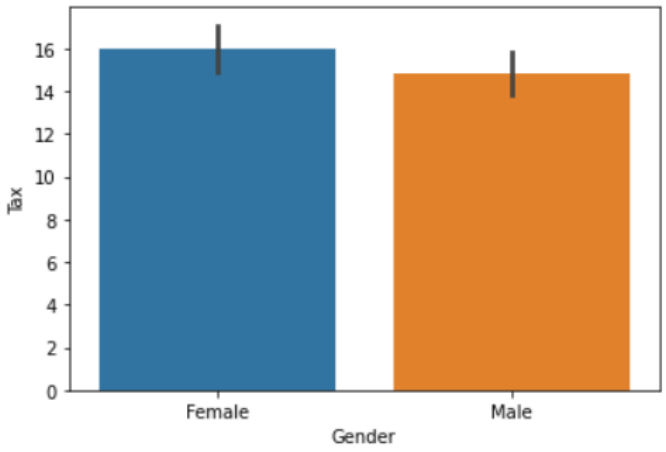


8) Gender & Quantity & Product_line

```
In [16]: sns.barplot(x = "Gender" , y= "Quantity" , hue = "Product_line" , data =df);
```

9) Gender & Tax

```
In [18]: sns.barplot(x = "Gender" , y= "Tax" , data =df);
```



10) Gender & Rating & Product_line

```
In [19]: sns.barplot(x = "Gender" , y= "Rating" , hue = "Product_line" , data =df);
```