Question 1:

Veri Ön işleme :

Çift değerlerin silinmesi:

I will create a matrix that contains duplicated numbers and use the Pandas function to delete the repeated number.

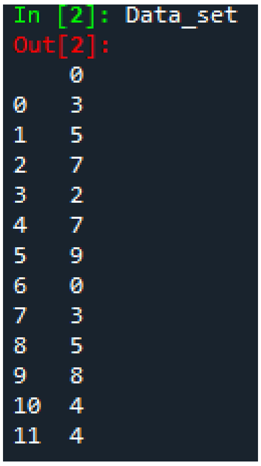
I used this command to create an array and conver it to DataFrame type:

Data_set=pandas.DataFrame(([3,5,7,2,7,9,0,3,5,8,4,4]))

To delete the duplicated number I used this command:

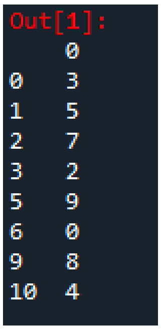
Data_set.drop_duplicates()

As shown in the pictures below:



```
In [2]: Data_set
Out[2]:
    0
0   3
1   5
2   7
3   2
4   7
5   9
6   0
7   3
8   5
9   8
10  4
11  4
```

Before Delete duplicated number



```
Out[1]:
    0
0   3
1   5
2   7
3   2
5   9
6   0
9   8
10  4
```

After deleting the duplicated number

Alakasız değerlerin silinmesi

Tutarsız değerlerin kaldırılması

İstenmeyen sütunun veya satırın kaldırılması

Here I used a dataset named *company.csv*. I loaded the data set and show the first five elements by using this command line:

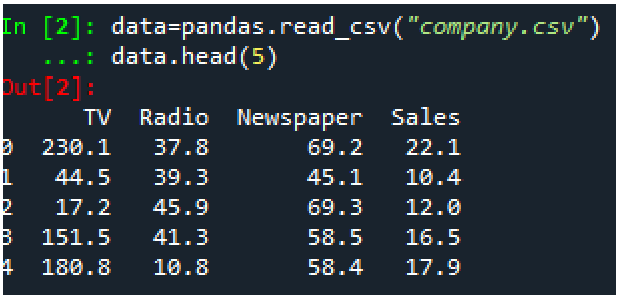
data=pandas.read_csv("company.csv")

data.head(5)

if we want to delete the TV and the redio columns, so we will used this command line:

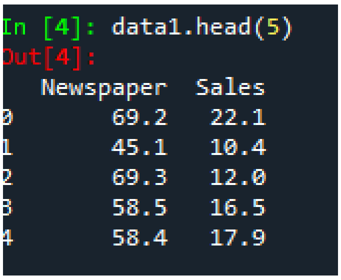
data1=data.drop(['TV','Radio'],axis=1)

The dataset before and after columns deleting can show in the figures below:



```
In [2]: data=pandas.read_csv("company.csv")
   ...: data.head(5)
Out[2]: 
      TV  Radio  Newspaper  Sales
0  230.1   37.8       69.2   22.1
1   44.5   39.3       45.1   10.4
2   17.2   45.9       69.3   12.0
3  151.5   41.3       58.5   16.5
4  180.8   10.8       58.4   17.9
```

Before



```
In [4]: data1.head(5)
Out[4]: 
   Newspaper  Sales
0       69.2   22.1
1       45.1   10.4
2       69.3   12.0
3       58.5   16.5
4       58.4   17.9
```

After

İstenmeyen sütunun veya satırın kaldırılması

Repeated Question

Eksik değerlerin silinmesi

At this question i used Stock market *ADANIPORTS.csv* dataset. To check is there is missing vlaue I used this command lines:

data=pandas.read_csv("ADANIPORTS.csv")

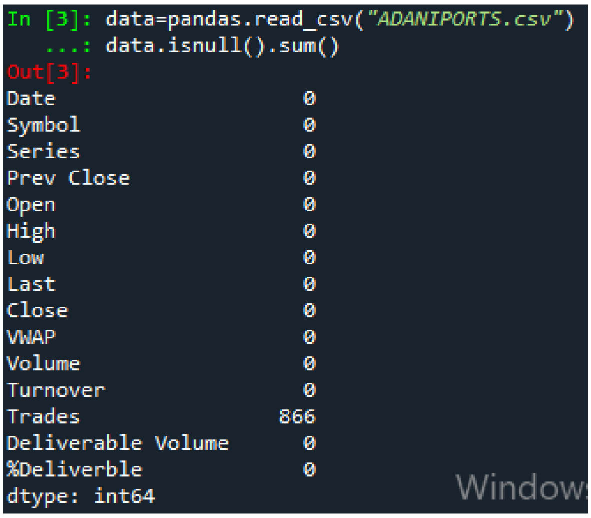
data.isnull().sum()

There were 866 missing value in Trades Column, i deleted missing values by this command line:

data2=data.dropna(axis='columns')

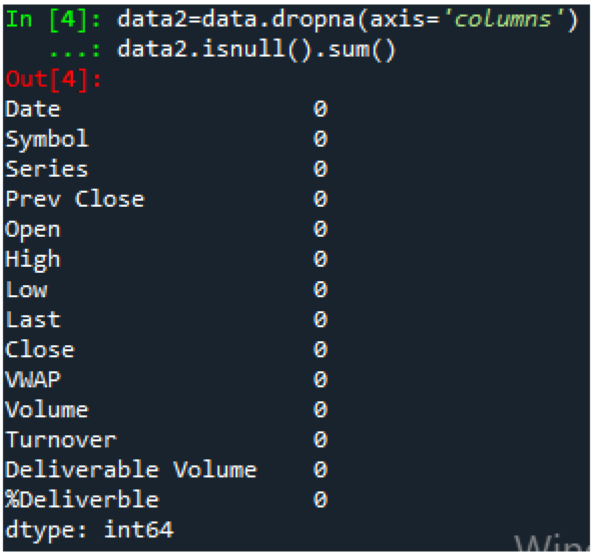
data2.isnull().sum()

The pictures below show the dataset before and after deleting missing values.



```
In [3]: data=pandas.read_csv("ADANIPORTS.csv")
   ...: data.isnull().sum()
Out[3]: 
Date                    0
Symbol                  0
Series                  0
Prev Close              0
Open                    0
High                    0
Low                     0
Last                    0
Close                   0
VWAP                    0
Volume                  0
Turnover                0
Trades                866
Deliverable Volume      0
%Deliverble             0
dtype: int64
```

Before



```
In [4]: data2=data.dropna(axis='columns')
   ...: data2.isnull().sum()
Out[4]: 
Date                  0
Symbol                0
Series                0
Prev Close            0
Open                  0
High                  0
Low                   0
Last                  0
Close                 0
VWAP                  0
Volume                0
Turnover              0
Deliverable Volume    0
%Deliverble           0
dtype: int64
```

After

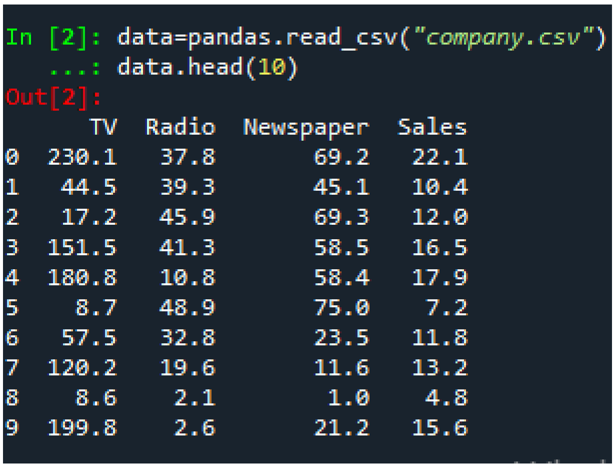
Aykırı değerlerin kaldırılması:

If we want the values in the column between numbers, for example, the values that are more than 50, we can achieve this by following command lines:

data=pandas.read_csv("company.csv")

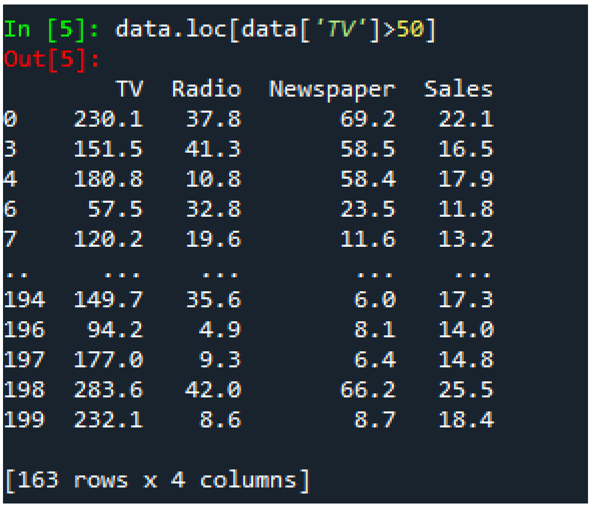
data.loc[data['TV']>50]

we deleted and number that more the 50, the output is shown in the figures below:



```
In [2]: data=pandas.read_csv("company.csv")
   ...: data.head(10)
Out[2]:
      TV  Radio  Newspaper  Sales
0  230.1   37.8       69.2   22.1
1   44.5   39.3       45.1   10.4
2   17.2   45.9       69.3   12.0
3  151.5   41.3       58.5   16.5
4  180.8   10.8       58.4   17.9
5    8.7   48.9       75.0    7.2
6   57.5   32.8       23.5   11.8
7  120.2   19.6       11.6   13.2
8    8.6    2.1        1.0    4.8
9  199.8    2.6       21.2   15.6
```

Before



```
In [5]: data.loc[data['TV']>50]
Out[5]:
        TV  Radio  Newspaper  Sales
0    230.1   37.8       69.2   22.1
3    151.5   41.3       58.5   16.5
4    180.8   10.8       58.4   17.9
6     57.5   32.8       23.5   11.8
7    120.2   19.6       11.6   13.2
..     ...    ...        ...    ...
194  149.7   35.6        6.0   17.3
196   94.2    4.9        8.1   14.0
197  177.0    9.3        6.4   14.8
198  283.6   42.0       66.2   25.5
199  232.1    8.6        8.7   18.4

[163 rows x 4 columns]
```

After

Opposite the first picture, the second picture doesn't have values less than 50.

ExtraTreesClassifier ile Öznitelik önemi (Feature importance) çıkarınız

ExtraTreesClassifier This class implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

At this question I used *company.csv*, In the command lines below, ceiling the dataset values to higher integral number and finally preparing the data to classify, by specifying the input and output of classifier.

data = pandas.read_csv("company.csv")

data=data.apply(numpy.ceil)

X = data.iloc[:,0:6]

y = data.iloc[:,-1]

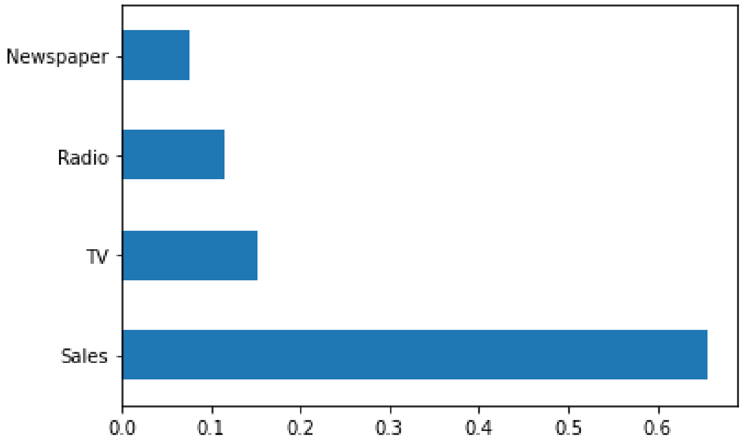
After that i applied *ExtraTreesClassifier* by command lines:

```
model = ExtraTreesClassifier()
model.fit(X,y)
```

To show the output I used this commands:

```
feat_importances = pandas.Series(model.feature_importances_, index=X.columns)
feat_importances.nlargest(20).plot(kind='barh')
plt.show()
```

the output is shown in the figure below:



ExtraTreesClassifier output.

Veri Setindeki en iyi öznitelikleri skorlandırınız, çıkarınız (SelectKBest)

The SelectKBest method **selects the features according to the k highest score**. By changing the 'score_func' parameter we can apply the method for both classification and regression data.

I used the *company.csv* dataset in this question

After loading and putting the parameter that the classifier will use to train, I speicfyed the class to extract top 5 best features and used the commands below.

```
bestfeatures = SelectKBest(score_func=chi2, k=4)
fit = bestfeatures.fit(X,y)
dfscores = pandas.DataFrame(fit.scores_)
dfcolumns = pandas.DataFrame(X.columns)
```

```
scores = pandas.concat([dfcolumns,dfscores],axis=1)

scores.columns = ['specs','score']

print(scores.nlargest(4,'score'))
```

Korelasyon ısı haritası çıkarınız (Correlation heat map)

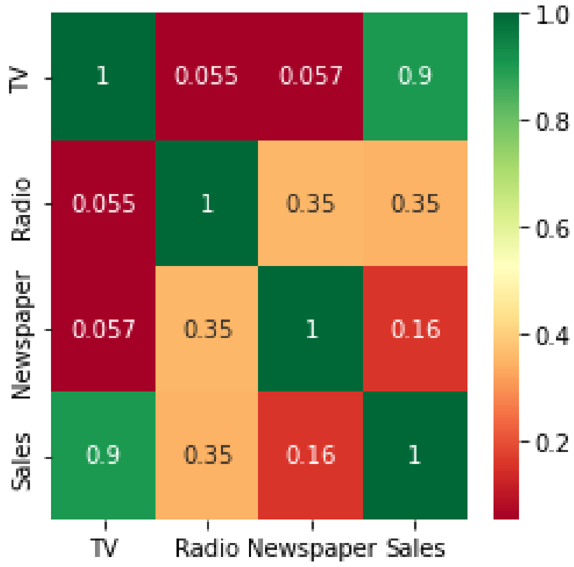
A correlation heatmap is a heatmap that shows a 2D correlation matrix between two discrete dimensions, using colored cells to represent data from usually a monochromatic scale.

At this question I used *company.csv* , for creating a correlation heatmap first we have to correlating the dataset values after that create the figure – according to the colunms number here will be 4- that the picture will lay in as shown in command lines below:

```
data = pandas.read_csv("company.csv")

correlation_matrix = data.corr()

top_corr_features = correlation_matrix.index

plt.figure(figsize=(4,4))
```

To plot the heatmap I used this command line:

```
g=seaborn.heatmap(data[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```



Heatmap Output.

Normal dağılıma sahip olmayan verileri standartlaştırınız.

For this question, I used the *company.csv* dataset.

For standarlizing I used this formula shown in command line:

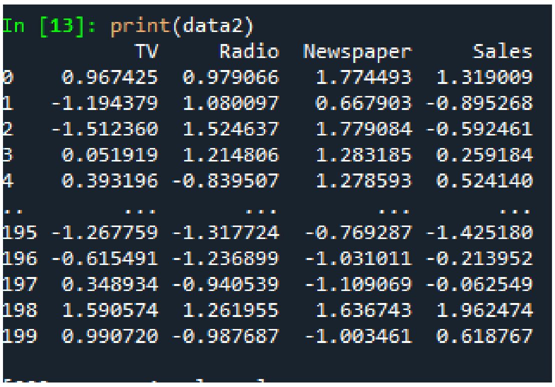
data = pandas.read_csv("company.csv")

#Standartlaştırma

data2 = (data- data.mean()) / data.std()

print(data2)

the output is shown in the figure below:



```
In [13]: print(data2)
             TV     Radio  Newspaper     Sales
0      0.967425  0.979066   1.774493  1.319009
1     -1.194379  1.080097   0.667903 -0.895268
2     -1.512360  1.524637   1.779084 -0.592461
3      0.051919  1.214806   1.283185  0.259184
4      0.393196 -0.839507   1.278593  0.524140
..          ...       ...        ...       ...
195   -1.267759 -1.317724  -0.769287 -1.425180
196   -0.615491 -1.236899  -1.031011 -0.213952
197    0.348934 -0.940539  -1.109069 -0.062549
198    1.590574  1.261955   1.636743  1.962474
199    0.990720 -0.987687  -1.003461  0.618767
```

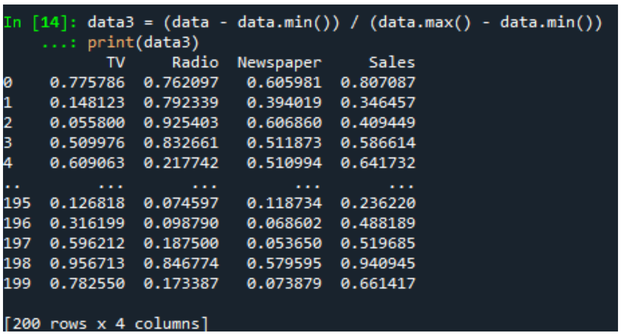
Standardizing output.

For normalizing I used this formula:

data3 = (data - data.min()) / (data.max() - data.min())

print(data3)

The output is shown in the figure below:



```
In [14]: data3 = (data - data.min()) / (data.max() - data.min())
    ...: print(data3)
           TV     Radio  Newspaper     Sales
0    0.775786  0.762097   0.605981  0.807087
1    0.148123  0.792339   0.394019  0.346457
2    0.055800  0.925403   0.606860  0.409449
3    0.509976  0.832661   0.511873  0.586614
4    0.609063  0.217742   0.510994  0.641732
..        ...       ...        ...       ...
195  0.126818  0.074597   0.118734  0.236220
196  0.316199  0.098790   0.068602  0.488189
197  0.596212  0.187500   0.053650  0.519685
198  0.956713  0.846774   0.579595  0.940945
199  0.782550  0.173387   0.073879  0.661417

[200 rows x 4 columns]
```

Normalizing output

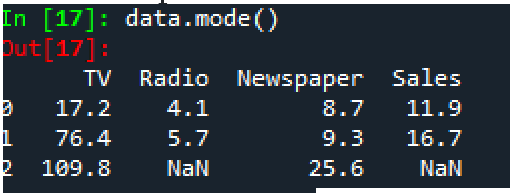
Veri üzerinde temel istatistik bilgileri çıkarınız

For this question, I used the *Company.csv* dataset.

Code

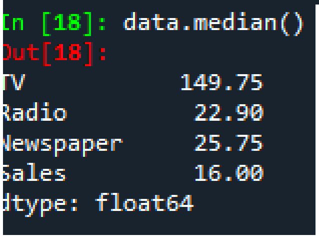
Code Output

```
#mod
data.mode()
```



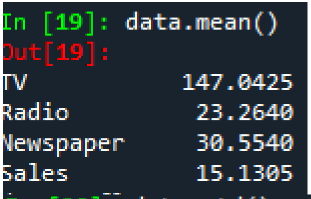
```
In [17]: data.mode()
Out[17]:
      TV  Radio  Newspaper  Sales
0   17.2    4.1        8.7   11.9
1   76.4    5.7        9.3   16.7
2  109.8    NaN       25.6    NaN
```

```
#medyan
data.median()
```



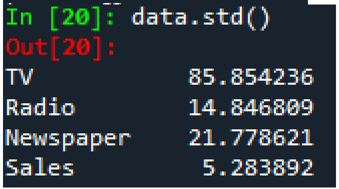
```
In [18]: data.median()
Out[18]:
TV           149.75
Radio         22.90
Newspaper     25.75
Sales         16.00
dtype: float64
```

```
#aritmetik ortalama
data.mean()
```



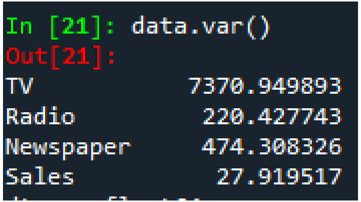
```
In [19]: data.mean()
Out[19]:
TV           147.0425
Radio         23.2640
Newspaper     30.5540
Sales         15.1305
```

```
#standart sapma
data.std()
```



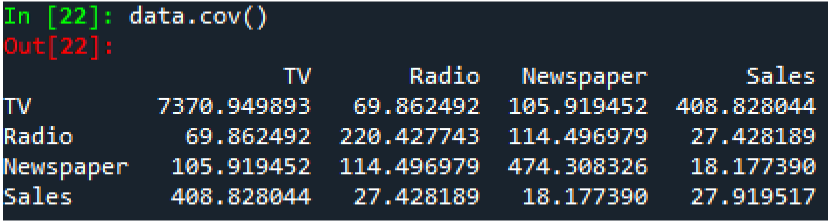
```
In [20]: data.std()
Out[20]:
TV           85.854236
Radio        14.846809
Newspaper    21.778621
Sales         5.283892
```

```
#varyans
data.var()
```



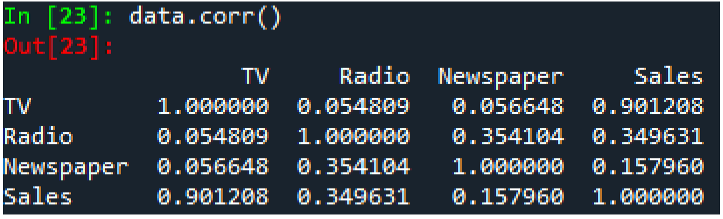
```
In [21]: data.var()
Out[21]:
TV           7370.949893
Radio         220.427743
Newspaper     474.308326
Sales          27.919517
```

```
#kovaryans
data.cov()
```



```
In [22]: data.cov()
Out[22]:
                    TV       Radio   Newspaper       Sales
TV         7370.949893   69.862492  105.919452  408.828044
Radio        69.862492  220.427743  114.496979   27.428189
Newspaper   105.919452  114.496979  474.308326   18.177390
Sales       408.828044   27.428189   18.177390   27.919517
```

```
#korelasyon
data.corr()
```



```
In [23]: data.corr()
Out[23]:
                 TV     Radio  Newspaper     Sales
TV         1.000000  0.054809   0.056648  0.901208
Radio      0.054809  1.000000   0.354104  0.349631
Newspaper  0.056648  0.354104   1.000000  0.157960
Sales      0.901208  0.349631   0.157960  1.000000
```