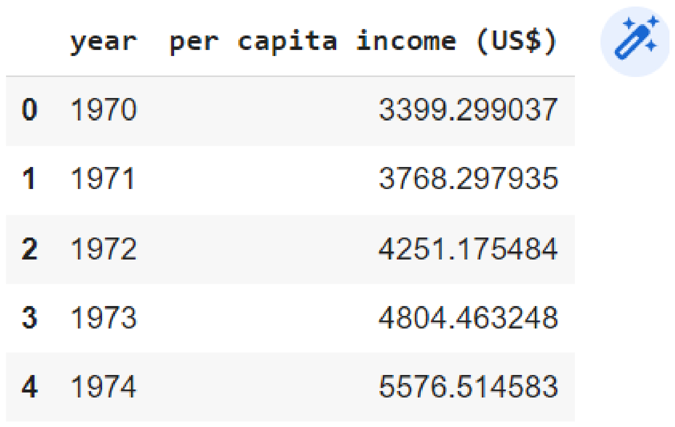
Practical No.3

1. Predict Canada’s per capita income in 2020. There is an exercise folder here on github at the same level as this notebook, download that and you will find the canada_per_capita_income.csv file. Using this build a regression model and predict the per capita income of canadian citizens in year 2020

Implementation:

Program and Outputs:

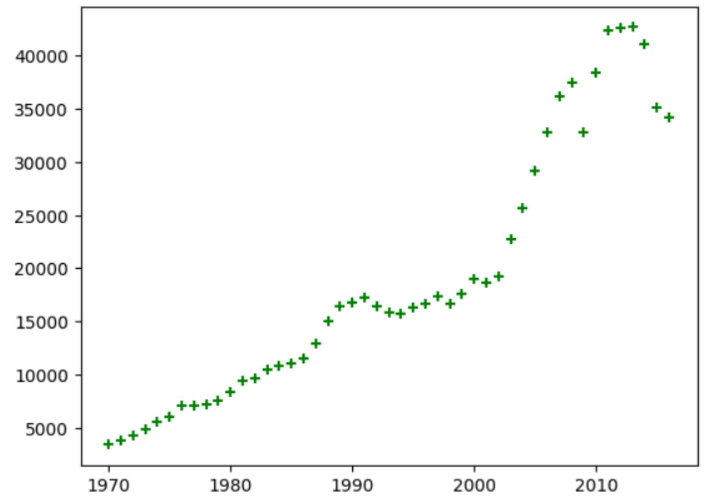
```
#Linear Regression Using Python

import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
from sklearn import linear_model

url='https://raw.githubusercontent.com/codebasics/py/master/ML/1_lin
ear_reg/Exercise/canada_per_capita_income.csv'

lr2=pd.read_csv(url)
lr2.head()
```



	year	per capita income (US$)
0	1970	3399.299037
1	1971	3768.297935
2	1972	4251.175484
3	1973	4804.463248
4	1974	5576.514583

```
lr2 = lr2.rename(columns={'per capita income (US$)': 'Income'})
```

```
plt.scatter(lr2.year,lr2.Income,color='g',marker='+')
plt.show()
```



```
lr3=lr2.drop('Income',axis=1)
lr3.head()
```

	year
0	1970
1	1971
2	1972
3	1973
4	1974

```
rg1=linear_model.LinearRegression()
rg1.fit(lr3,lr2.Income)
```

```
▾ LinearRegression
LinearRegression()
```

```
rg1.predict([[2020]])
```

```
array([41288.69409442])
```

```
rg1.coef_
```

```
array([828.46507522])
```

```
rg1.intercept_
```

```
-1632210.7578554575
```

```
#y=mx+b
828.46507522*2020+(-1632210.7578554575)
```

```
41288.694088942604
```

2. Download employee retention dataset from here:

https://www.kaggle.com/giripujar/hr-analytics.

Now do some exploratory data analysis to figure out which variables have direct and clear impact on employee retention (i.e. whether they leave the company or continue to work)

Plot bar charts showing impact of employee salaries on retention

Plot bar charts showing correlation between department and employee retention

Now build logistic regression model using variables that were narrowed down in

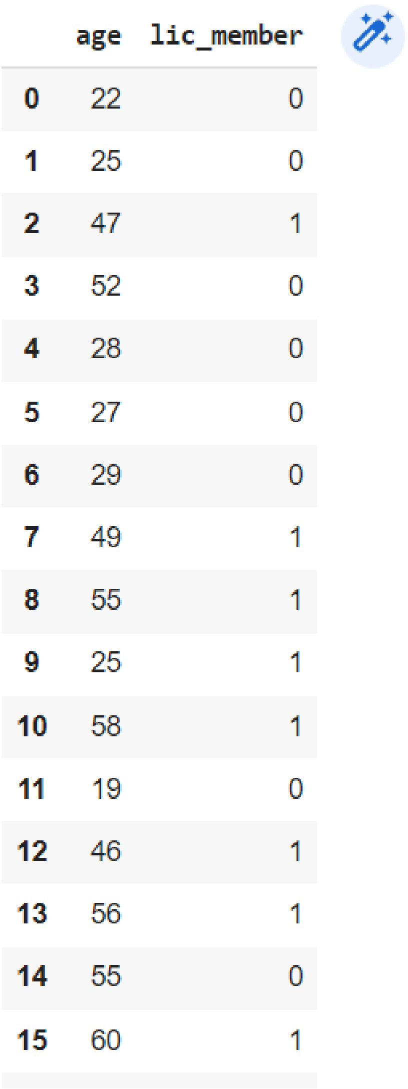
step 1

Measure the accuracy of the model

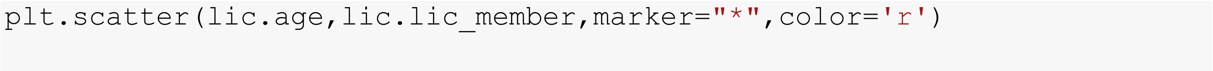
Implementation:

Program And Output:

```
#logitsic regression- Binary classification import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
from sklearn import linear_model
url='http://raw.githubusercontent.com/WamanParulekar/AIML/main/lic.csv'
lic = pd.read_csv(url)
lic
```

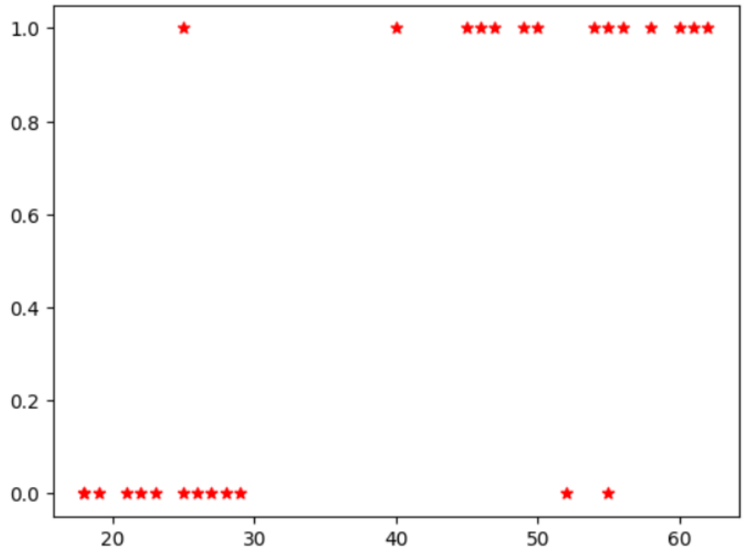


	age	lic_member
0	22	0
1	25	0
2	47	1
3	52	0
4	28	0
5	27	0
6	29	0
7	49	1
8	55	1
9	25	1
10	58	1
11	19	0
12	46	1
13	56	1
14	55	0
15	60	1



```
plt.scatter(lic.age,lic.lic_member,marker="*",color='r')
```

```
<matplotlib.collections.PathCollection at 0x7fa346664fd0>
```



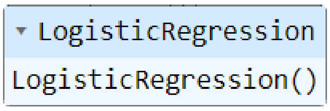
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(lic[['age']],lic.lic_member, train_size=0.8)
len(x_test)
```

```
6
```

```
len(y_test)
```

```
6
```

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()

lr.fit(x_train , y_train)
```

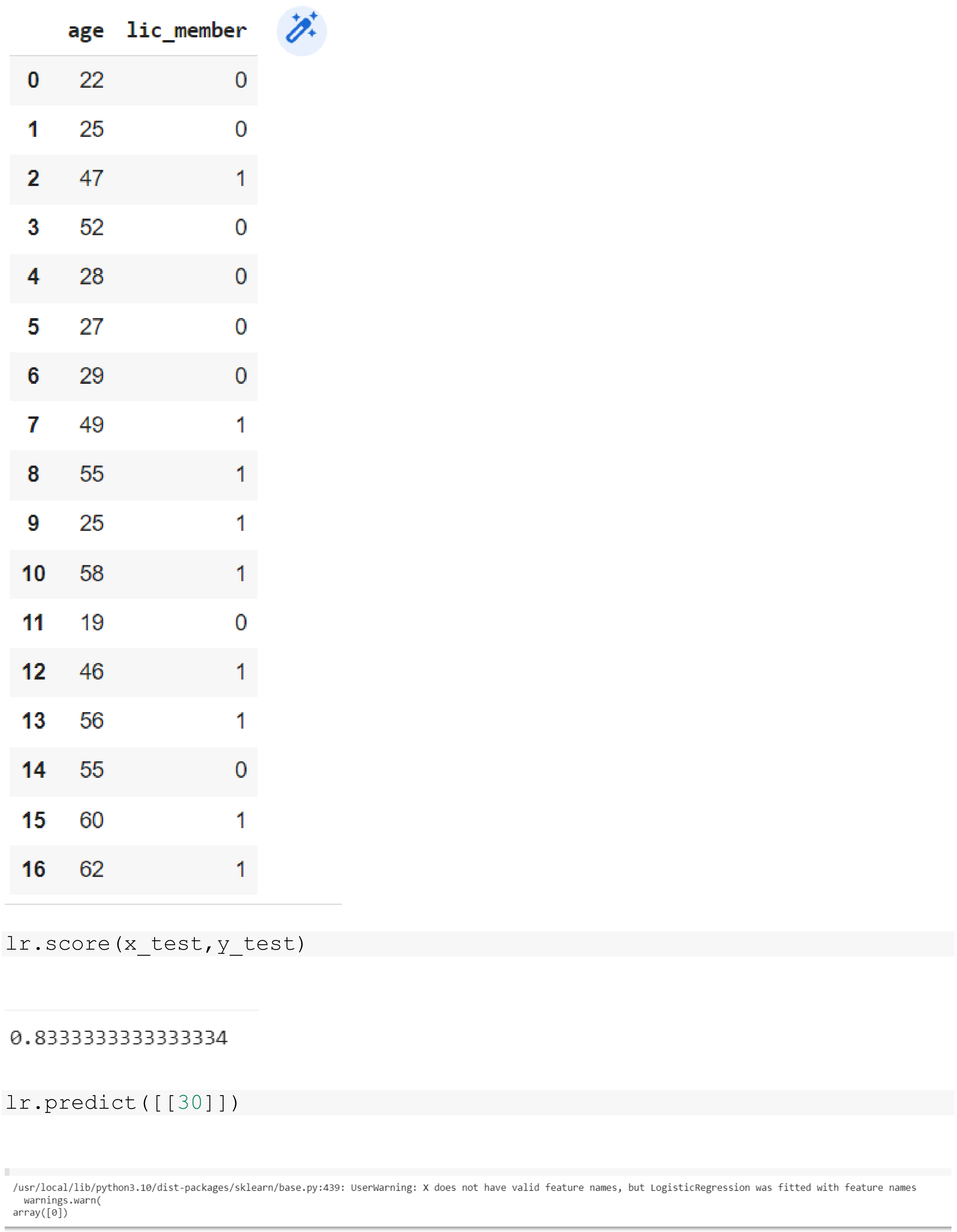


```
▾ LogisticRegression
LogisticRegression()
```

```
lr.predict(x_test)
```

```
array([1, 0, 1, 1, 1, 0])
```

```
lic
```



	age	lic_member
0	22	0
1	25	0
2	47	1
3	52	0
4	28	0
5	27	0
6	29	0
7	49	1
8	55	1
9	25	1
10	58	1
11	19	0
12	46	1
13	56	1
14	55	0
15	60	1
16	62	1

```
lr.score(x_test,y_test)
```

```
0.8333333333333334
```

```
lr.predict([[30]])
```

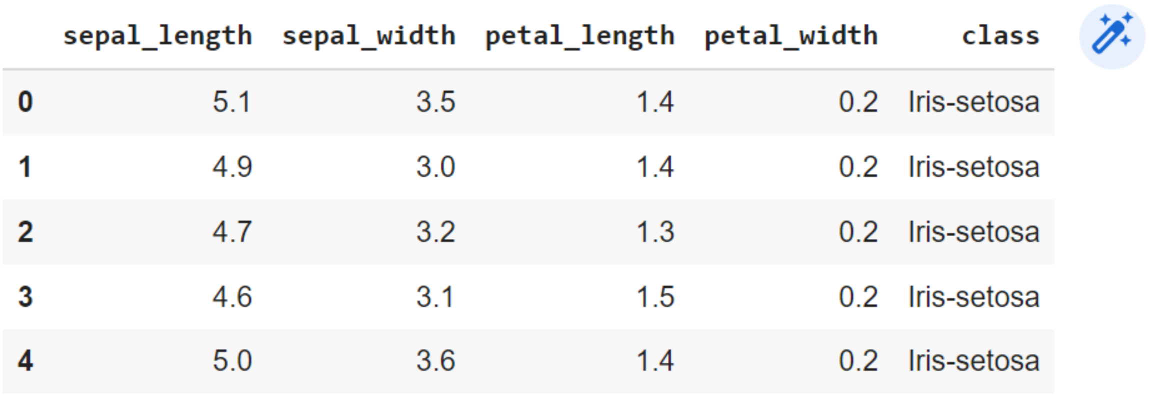
```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
array([0])
```

3. Using K nearest neighbors classification predict type of flower given ‘sepal_length’, ‘sepal_width’,’petal_length’, ‘petal_width’ = 4.8,3.0,1.5,0.3

Implementation:

Program:

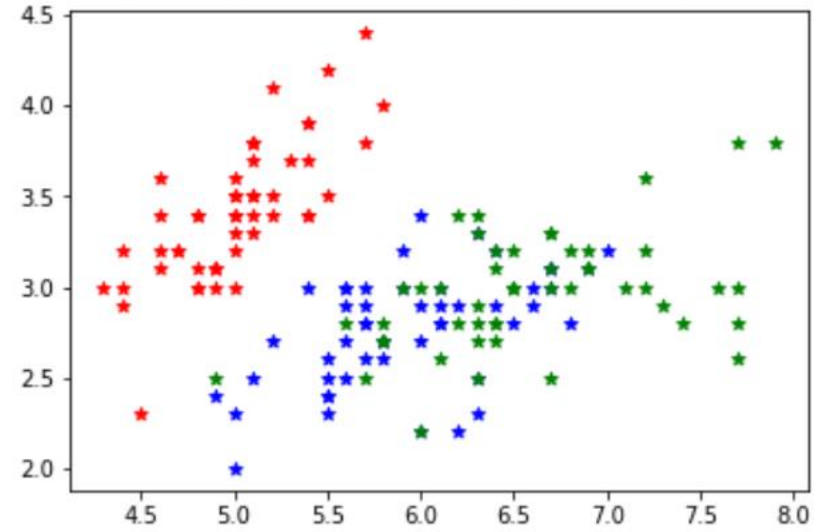
```
#KNN classification
import pandas as pd
import numpy as np
from sklearn import linear_model
import matplotlib.pyplot as plt

url = 'http://archive.ics.uci.edu/ml/machine-learning-
databases/iris/iris.data'
iris = pd.read_csv(url, names=['sepal_length', 'sepal_width', 'petal_le
ngth', 'petal_width', 'class'])

iris.head()
```



	sepal_length	sepal_width	petal_length	petal_width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

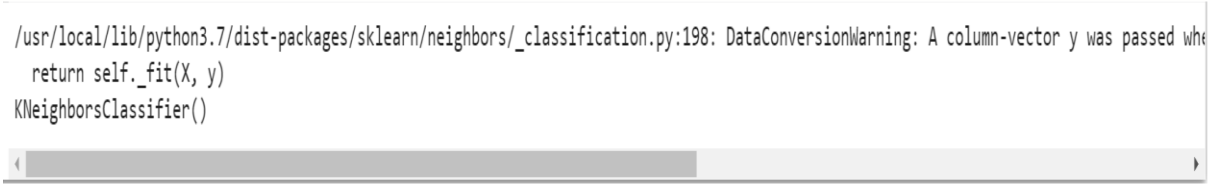
```
iris1 = iris[:50]
iris2 = iris[50:100]
iris3 = iris[100:]


plt.scatter(iris1.sepal_length,iris1.sepal_width,marker="*",color='r')
plt.scatter(iris2.sepal_length,iris2.sepal_width,marker="*",color='b')
plt.scatter(iris3.sepal_length,iris3.sepal_width,marker="*",color='g')
```

```
<matplotlib.collections.PathCollection at 0x7ff2f48f1850>
```



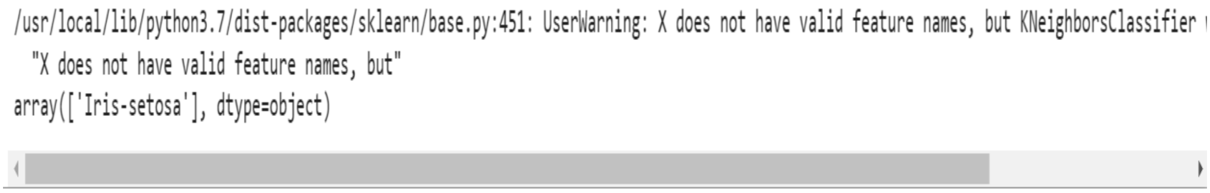
```
from sklearn.model_selection import train_test_split
X = iris.drop('class',axis=1)
y = iris[['class']]
X_train,X_test,y_train,y_test = train_test_split(X,y,train_size=0.8)
```

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
```



```
/usr/local/lib/python3.7/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-vector y was passed whe
  return self._fit(X, y)
KNeighborsClassifier()
```

```
knn.predict([[4.8,3.0,1.5,0.3]])
```



```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but KNeighborsClassifier
  "X does not have valid feature names, but"
array(['Iris-setosa'], dtype=object)
```