MACHINE LEARNING LAB

EXERCISE :: 2

NAME:: Saptarshi Datta

REG NO.:: 19BAI1041

Logistic Regression ::

The dataset is about health report of some people. The goal to determine chances of heart disease.

Link ::

https://www.kaggle.com/ronitf/heart-disease-uci?select=heart.csv

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1		age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
2	0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	7	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
11	9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
12	10	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
13	11	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
14	12	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
15	13	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
16	14	58	0	3	150	283	1	0	162	0	1	2	0	2	1
17	15	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
18	16	58	0	2	120	340	0	1	172	0	0	2	0	2	1
19	17	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1
20	18	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1
21	19	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1
22	20	59	1	0	135	234	0	1	161	0	0.5	1	0	3	1

Code ::

```
import numpy as np
import pandas as pd
import seaborn as sb
from sklearn import linear_model
from sklearn import metrics
```

```
import matplotlib.pyplot as plt

%matplotlib inline


df = pd.read_csv(r"C:\Users\SAPTARSHI\Desktop\ML\Heart attack prediction\data\heart.csv")

print(df.shape)

print(df.info())

print(df)
```

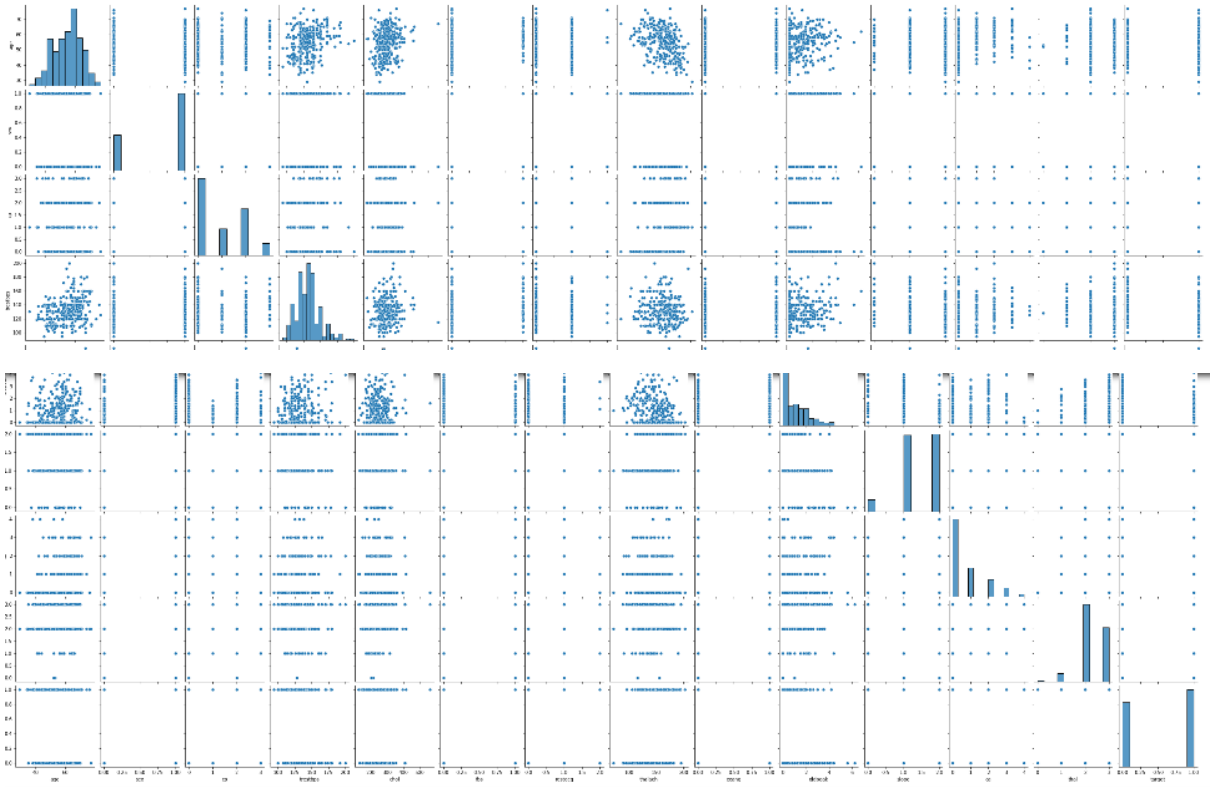
```
(303, 14)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
None
```

```
     age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0     63    1   3       145   233    1        0      150      0      2.3
1     37    1   2       130   250    0        1      187      0      3.5
2     41    0   1       130   204    0        0      172      0      1.4
3     56    1   1       120   236    0        1      178      0      0.8
4     57    0   0       120   354    0        1      163      1      0.6
..   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
298   57    0   0       140   241    0        1      123      1      0.2
299   45    1   3       110   264    0        1      132      0      1.2
300   68    1   0       144   193    1        1      141      0      3.4
301   57    1   0       130   131    0        1      115      1      1.2
302   57    0   1       130   236    0        0      174      0      0.0

     slope  ca  thal  target
0        0   0     1       1
1        0   0     2       1
2        2   0     2       1
3        2   0     2       1
4        2   0     2       1
..     ...  ..   ...     ...
298      1   0     3       0
299      1   0     3       0
300      1   2     3       0
301      1   1     3       0
302      1   1     2       0

[303 rows x 14 columns]
```

print(df[df['target']== 1].count())

```
age         165
sex         165
cp          165
trestbps    165
chol        165
fbs         165
restecg     165
thalach     165
exang       165
oldpeak     165
slope       165
ca          165
thal        165
target      165
dtype: int64
```

sb.pairplot(df)



df1 = df.corr()

print(df1)

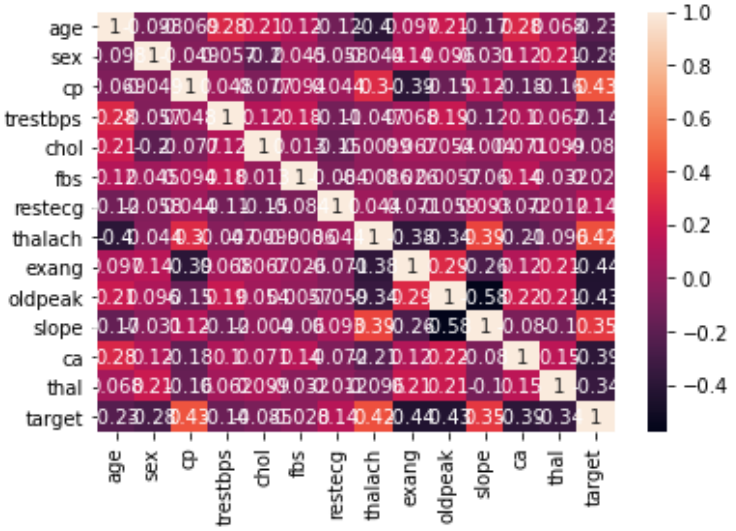
sb.heatmap(df1,annot=True)

	age	sex	cp	trestbps	chol	fbs
age	1.000000	-0.098447	-0.068653	0.279351	0.213678	0.121308
sex	-0.098447	1.000000	-0.049353	-0.056769	-0.197912	0.045032
cp	-0.068653	-0.049353	1.000000	0.047608	-0.076904	0.094444
trestbps	0.279351	-0.056769	0.047608	1.000000	0.123174	0.177531
chol	0.213678	-0.197912	-0.076904	0.123174	1.000000	0.013294
fbs	0.121308	0.045032	0.094444	0.177531	0.013294	1.000000
restecg	-0.116211	-0.058196	0.044421	-0.114103	-0.151040	-0.084189
thalach	-0.398522	-0.044020	0.295762	-0.046698	-0.009940	-0.008567
exang	0.096801	0.141664	-0.394280	0.067616	0.067023	0.025665
oldpeak	0.210013	0.096093	-0.149230	0.193216	0.053952	0.005747
slope	-0.168814	-0.030711	0.119717	-0.121475	-0.004038	-0.059894
ca	0.276326	0.118261	-0.181053	0.101389	0.070511	0.137979
thal	0.068001	0.210041	-0.161736	0.062210	0.098803	-0.032019
target	-0.225439	-0.280937	0.433798	-0.144931	-0.085239	-0.028046

	restecg	thalach	exang	oldpeak	slope	ca
age	-0.116211	-0.398522	0.096801	0.210013	-0.168814	0.276326
sex	-0.058196	-0.044020	0.141664	0.096093	-0.030711	0.118261
cp	0.044421	0.295762	-0.394280	-0.149230	0.119717	-0.181053
trestbps	-0.114103	-0.046698	0.067616	0.193216	-0.121475	0.101389
chol	-0.151040	-0.009940	0.067023	0.053952	-0.004038	0.070511
fbs	-0.084189	-0.008567	0.025665	0.005747	-0.059894	0.137979
restecg	1.000000	0.044123	-0.070733	-0.058770	0.093045	-0.072042
thalach	0.044123	1.000000	-0.378812	-0.344187	0.386784	-0.213177
exang	-0.070733	-0.378812	1.000000	0.288223	-0.257748	0.115739
oldpeak	-0.058770	-0.344187	0.288223	1.000000	-0.577537	0.222682
slope	0.093045	0.386784	-0.257748	-0.577537	1.000000	-0.080155
ca	-0.072042	-0.213177	0.115739	0.222682	-0.080155	1.000000
thal	-0.011981	-0.096439	0.206754	0.210244	-0.104764	0.151832
target	0.137230	0.421741	-0.436757	-0.430696	0.345877	-0.391724

	thal	target
age	0.068001	-0.225439
sex	0.210041	-0.280937
cp	-0.161736	0.433798
trestbps	0.062210	-0.144931
chol	0.098803	-0.085239
fbs	-0.032019	-0.028046
restecg	-0.011981	0.137230
thalach	-0.096439	0.421741
exang	0.206754	-0.436757
oldpeak	0.210244	-0.430696
slope	-0.104764	0.345877
ca	0.151832	-0.391724
thal	1.000000	-0.344029
target	-0.344029	1.000000

Out[53]: <AxesSubplot:>




```
regress = linear_model.LogisticRegression()
df2 = df.values
#print(df2)
train_x = df2[:,0:13]
train_y = df2[:,13]
print(train_x.shape)
print(train_y.shape)
```

```
(303, 13)
(303,)
```

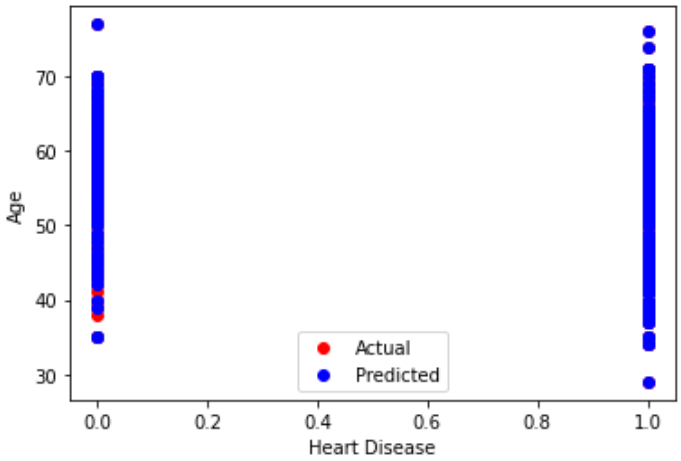
```
regress.fit(train_x, train_y)
print('Coefficients:', regress.coef_)
print('Intercept:', regress.intercept_)
```

```
Coefficients: [[ 0.00958044 -1.32407876  0.83160996 -0.0146217  -0.00273645 -0.08181688
   0.54302194  0.03098253 -0.78672325 -0.52163295  0.51810081 -0.72966656
  -0.91105852]]
Intercept: [0.048591]
```

```
y_predicted = regress.predict(train_x)
for i in range(0,len(train_x)):
    print(train_y[i], y_predicted[i])
df['Pred']=y_predicted
df
df.to_csv(r"C:\Users\SAPTARSHI\Desktop\ML\Heart attack prediction\data\heart_result.csv")
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1		age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Pred
2	0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1	1
3	1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1	1
4	2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1	1
5	3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1	1
6	4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1	1
7	5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1	1
8	6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1	1
9	7	44	1	1	120	263	0	1	173	0	0	2	0	3	1	1
10	8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1	1
11	9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1	1
12	10	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1	1
13	11	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1	1
14	12	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1	1
15	13	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1	1
16	14	58	0	3	150	283	1	0	162	0	1	2	0	2	1	1
17	15	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1	1
18	16	58	0	2	120	340	0	1	172	0	0	2	0	2	1	1
19	17	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1	1
20	18	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1	1
21	19	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1	1
22	20	59	1	0	135	234	0	1	161	0	0.5	1	0	3	1	0
23	21	44	1	2	130	233	0	1	179	1	0.4	2	0	2	1	1
24	22	42	1	0	140	226	0	1	178	0	0	2	0	2	1	1

```
plt.scatter(df['target'], df['age'], color='red', label='Actual')
plt.scatter(df['Pred'], df['age'], color='blue', label="Predicted")
plt.xlabel('Heart Disease')
plt.ylabel('Age')
plt.legend()
plt.show()
```



print('Accuracy:', metrics.accuracy_score(train_y, y_predicted))

print('Confusion Matrix\n:', metrics.confusion_matrix(train_y, y_predicted))

```
Accuracy: 0.8547854785478548
Confusion Matrix
: [[106  32]
 [ 12 153]]
```

from sklearn.metrics import classification_report

print(classification_report(train_y, y_predicted))

	precision	recall	f1-score	support
0.0	0.90	0.77	0.83	138
1.0	0.83	0.93	0.87	165
accuracy			0.85	303
macro avg	0.86	0.85	0.85	303
weighted avg	0.86	0.85	0.85	303