ML Lab-6 Practice And Assesment

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.preprocessing import LabelEncoder
```

Practice

In [2]:

```
data = pd.read_csv("data.csv")
data.head()
```

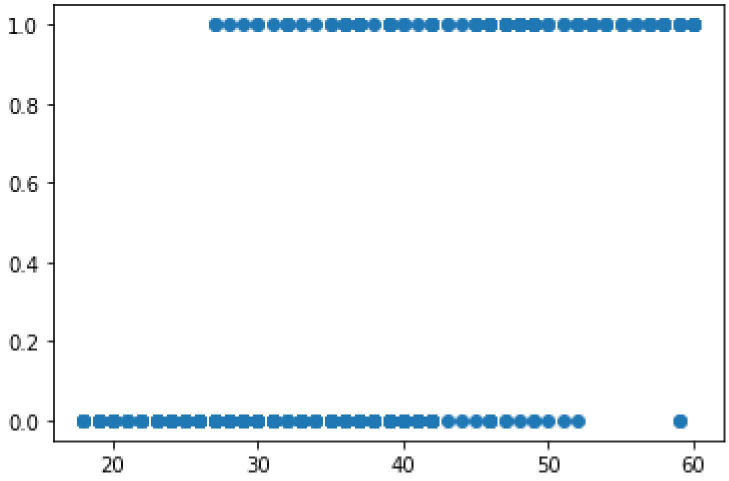
Out[2]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

In [3]:

```
# Visualizing the dataset
plt.scatter(data['Age'], data['Purchased'])
plt.show()

# Divide the data to training set and test set
X_train, X_test, y_train, y_test = train_test_split(data['Age'], data['Purchased'], tes
```



In [4]:

```
lr_model = LogisticRegression(max_iter=10000)
```

In [5]:

```
lr_model.fit(X_train.values.reshape(-1, 1), y_train.values.reshape(-1,))
```

Out[5]:

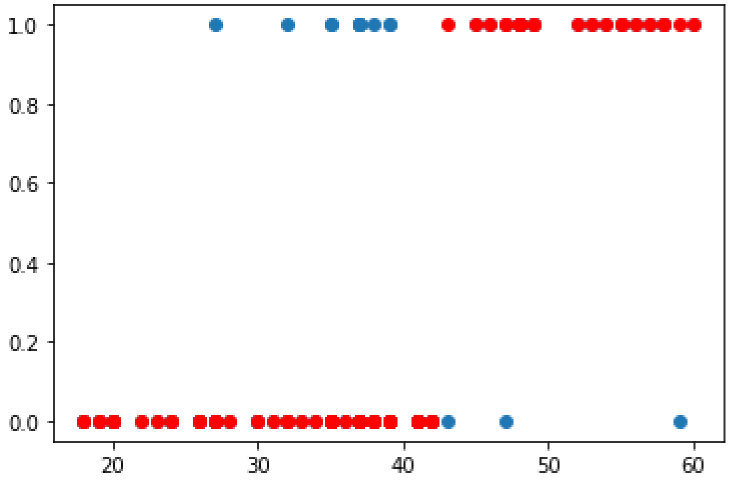
```
LogisticRegression(max_iter=10000)
```

In [6]:

```
y_pred_sk = lr_model.predict(X_test.values.reshape(-1, 1))
```

In [7]:

```
plt.clf()
plt.scatter(X_test, y_test)
plt.scatter(X_test, y_pred_sk, c="red")
plt.show()
```



In [8]:

```
lr_model.score(X_test.values.reshape(-1, 1), y_test.values.reshape(-1, 1))*100
```

Out[8]:

```
82.5
```

In [9]:

```
print(confusion_matrix(y_test,y_pred_sk))
```

```
[[47  3]
 [11 19]]
```

In [10]:

```
print(classification_report(y_test,y_pred_sk))
```

```
              precision    recall  f1-score   support

           0       0.81      0.94      0.87        50
           1       0.86      0.63      0.73        30

    accuracy                           0.82        80
   macro avg       0.84      0.79      0.80        80
weighted avg       0.83      0.82      0.82        80
```

In [11]:

```
Comparison = pd.DataFrame({'Actual':y_test,'Predicted':y_pred_sk})
Comparison.head(15)
```

Out[11]:

	Actual	Predicted
178	0	0
67	0	0
31	1	0
312	0	0
346	1	1
204	1	1
273	1	0
383	1	1
353	0	0
196	0	0
69	0	0
281	0	0
20	1	1
249	1	0
19	1	1

Assesment

In [12]:

```
df = pd.read_csv('framingham.csv')
df.head()
```

Out[12]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes
0	1	39	4.0	0	0.0	0.0	0	0	0
1	0	46	2.0	0	0.0	0.0	0	0	0
2	1	48	1.0	1	20.0	0.0	0	0	0
3	0	61	3.0	1	30.0	0.0	0	1	0
4	0	46	3.0	1	23.0	0.0	0	0	0

In [13]:

```
df.shape
```

Out[13]:

```
(4238, 16)
```

In [14]:

```
df.describe()
```

Out[14]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStrok
count	4238.000000	4238.000000	4133.000000	4238.000000	4209.000000	4185.000000	4238.00000
mean	0.429212	49.584946	1.978950	0.494101	9.003089	0.029630	0.00589
std	0.495022	8.572160	1.019791	0.500024	11.920094	0.169584	0.07658
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000	0.00000
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000	0.00000
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000	0.00000
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000	0.00000
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000	1.00000

In [15]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4238 entries, 0 to 4237
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             4238 non-null   int64  
 1   age              4238 non-null   int64  
 2   education        4133 non-null   float64
 3   currentSmoker    4238 non-null   int64  
 4   cigsPerDay       4209 non-null   float64
 5   BPMeds           4185 non-null   float64
 6   prevalentStroke  4238 non-null   int64  
 7   prevalentHyp     4238 non-null   int64  
 8   diabetes         4238 non-null   int64  
 9   totChol          4188 non-null   float64
 10  sysBP            4238 non-null   float64
 11  diaBP            4238 non-null   float64
 12  BMI              4219 non-null   float64
 13  heartRate        4237 non-null   float64
 14  glucose          3850 non-null   float64
 15  TenYearCHD       4238 non-null   int64  
dtypes: float64(9), int64(7)
memory usage: 529.9 KB
```

In [16]:

```
df.isnull().sum()
```

Out[16]:

```
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
```

```
glucose            388
TenYearCHD           0
dtype: int64
```

In [17]:

```
df['education'].value_counts()
```

Out[17]:

```
1.0    1720
2.0    1253
3.0     687
4.0     473
Name: education, dtype: int64
```

In [18]:

```
df['education'] = df['education'].fillna(1.0)
```

In [19]:

```
df['cigsPerDay'].value_counts()
```

Out[19]:

```
0.0     2144
20.0     734
30.0     217
15.0     210
10.0     143
9.0      130
5.0      121
3.0      100
40.0      80
1.0       67
43.0      56
25.0      55
35.0      22
6.0       18
2.0       18
7.0       12
60.0      11
8.0       11
4.0        9
18.0       8
17.0       7
50.0       6
23.0       6
11.0       5
16.0       3
12.0       3
13.0       3
45.0       3
19.0       2
14.0       2
70.0       1
38.0       1
29.0       1
Name: cigsPerDay, dtype: int64
```

In [20]:

```
df['cigsPerDay'] = df['cigsPerDay'].fillna(1.0)
```

In [21]:

```
df['BPMeds'].value_counts()
```

Out[21]:

```
0.0    4061
1.0     124
Name: BPMeds, dtype: int64
```

In [22]:

```
df['BPMeds'] = df['BPMeds'].fillna(0.0)
```

In [23]:

```
df['totChol'].mean()
```

Out[23]:

```
236.72158548233045
```

In [24]:

```
df['totChol'] = df['totChol'].fillna(236.72)
```

In [25]:

```
df['BMI'].mean()
```

Out[25]:

```
25.80200758473571
```

In [26]:

```
df['BMI'] = df['BMI'].fillna(25.8)
```

In [27]:

```
df['glucose'].mean()
```

Out[27]:

```
81.96675324675324
```

In [28]:

```
df['glucose'] = df['glucose'].fillna(81.96)
```

In [29]:

```
df['heartRate'].mean()
```

Out[29]:

```
75.87892376681614
```

In [30]:

```
df['heartRate'] = df['heartRate'].fillna(75.0)
```

In [31]:

```
df.rename(columns={'male':'gender'},inplace=True)
df.head()
```

Out[31]:

	gender	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabet
0	1	39	4.0	0	0.0	0.0	0	0	
1	0	46	2.0	0	0.0	0.0	0	0	
2	1	48	1.0	1	20.0	0.0	0	0	
3	0	61	3.0	1	30.0	0.0	0	1	
4	0	46	3.0	1	23.0	0.0	0	0	

In [32]:

```
df['gender'] = df['gender'].replace({0:'Female',1:'Male'})
df['currentSmoker'] = df['currentSmoker'].replace({0:'No',1:'Yes'})
df['BPMeds'] = df['BPMeds'].replace({0:'No',1:'Yes'})
```

```
df['prevalentStroke'] = df['prevalentStroke'].replace({0:'No',1:'Yes'})
df['prevalentHyp'] = df['prevalentHyp'].replace({0:'No',1:'Yes'})
df['diabetes'] = df['diabetes'].replace({0:'No',1:'Yes'})
df['TenYearCHD'] = df['TenYearCHD'].replace({0:'No',1:'Yes'})
df.head(10)
```

Out[32]:

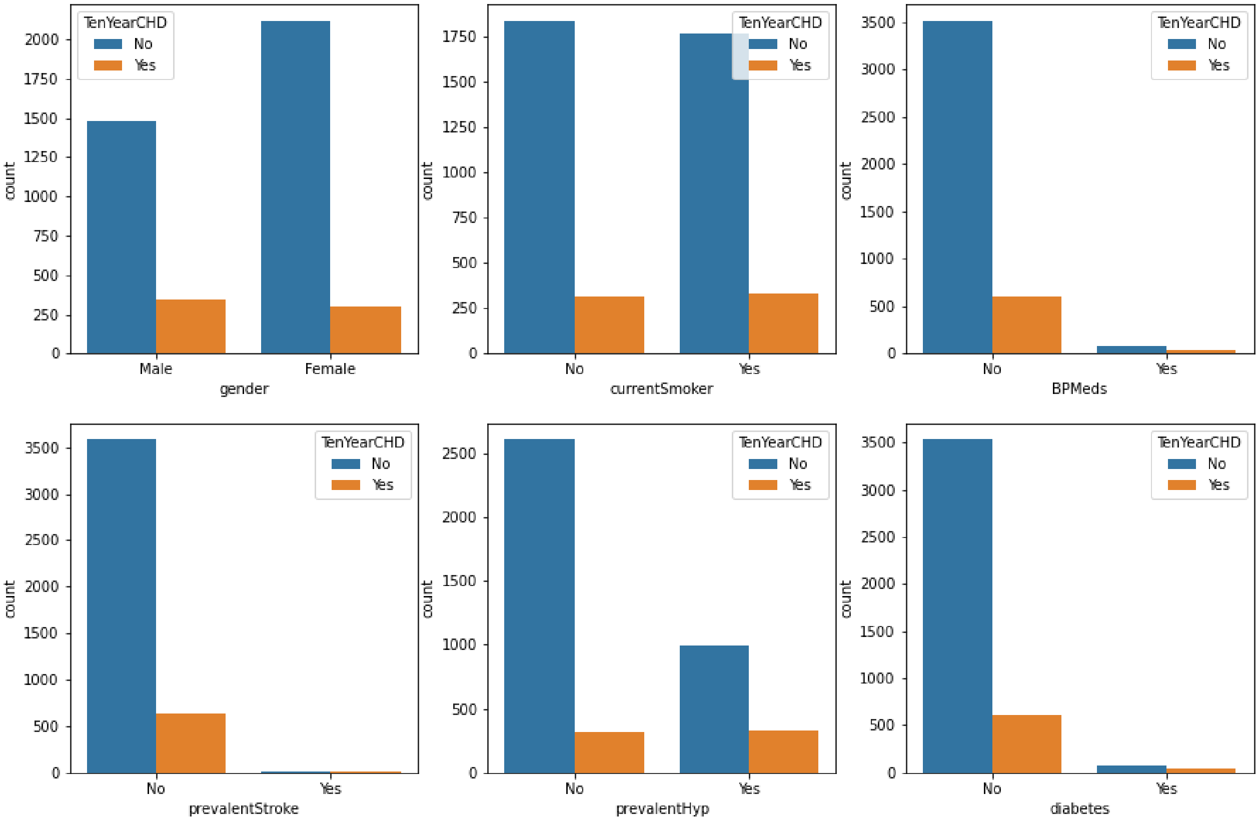
	gender	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabet
0	Male	39	4.0	No	0.0	No	No	No	N
1	Female	46	2.0	No	0.0	No	No	No	N
2	Male	48	1.0	Yes	20.0	No	No	No	N
3	Female	61	3.0	Yes	30.0	No	No	Yes	N
4	Female	46	3.0	Yes	23.0	No	No	No	N
5	Female	43	2.0	No	0.0	No	No	Yes	N
6	Female	63	1.0	No	0.0	No	No	No	N
7	Female	45	2.0	Yes	20.0	No	No	No	N
8	Male	52	1.0	No	0.0	No	No	Yes	N
9	Male	43	1.0	Yes	30.0	No	No	Yes	N

In [33]:

```
plt.figure(figsize=(15, 10))

plt.subplot(2, 3, 1)
sns.countplot(x='gender',hue='TenYearCHD',data=df)
plt.subplot(2, 3, 2)
sns.countplot(x='currentSmoker',hue='TenYearCHD',data=df)
plt.subplot(2, 3, 3)
sns.countplot(x='BPMeds',hue='TenYearCHD',data=df)
plt.subplot(2, 3, 4)
sns.countplot(x='prevalentStroke',hue='TenYearCHD',data=df)
plt.subplot(2, 3, 5)
sns.countplot(x='prevalentHyp',hue='TenYearCHD',data=df)
plt.subplot(2, 3, 6)
sns.countplot(x='diabetes',hue='TenYearCHD',data=df)

plt.show()
```



In [34]:

```
le = LabelEncoder()
df['gender'] = le.fit_transform(df['gender'])
df['currentSmoker'] = le.fit_transform(df['currentSmoker'])
df['BPMeds'] = le.fit_transform(df['BPMeds'])
df['prevalentStroke'] = le.fit_transform(df['prevalentStroke'])
df['prevalentHyp'] = le.fit_transform(df['prevalentHyp'])
df['diabetes'] = le.fit_transform(df['diabetes'])
```

In [35]:

```
X = df.drop('TenYearCHD',axis=1)
y = df['TenYearCHD']
```

In [36]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(3178, 15)
(1060, 15)
(3178,)
(1060,)
```

In [37]:

```
model = LogisticRegression(max_iter=10000)
```

In [38]:

```
model.fit(X_train,y_train)
```

Out[38]:

```
LogisticRegression(max_iter=10000)
```

In [39]:

```
model.score(X_train,y_train)
```

Out[39]:

```
0.8524229074889867
```

In [40]:

```
y_pred = model.predict(X_test)
```

In [41]:

```
print(confusion_matrix(y_test,y_pred))
```

```
[[901   7]
 [135  17]]
```

In [42]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

          No       0.87      0.99      0.93       908
         Yes       0.71      0.11      0.19       152

    accuracy                           0.87      1060
   macro avg       0.79      0.55      0.56      1060
weighted avg       0.85      0.87      0.82      1060
```

In [43]:

```
print(accuracy_score(y_test,y_pred)*100)
```

```
86.60377358490567
```

In [44]:

```
Comparison = pd.DataFrame({'Actual':y_test,'Predicted':y_pred})
Comparison.head(15)
```

Out[44]:

	Actual	Predicted
3188	No	No
764	No	No
3264	No	No
1967	No	No
2185	No	No
393	No	No
2333	Yes	No
1159	No	No
3788	No	No
1674	Yes	Yes
759	No	No
1803	No	No
410	No	No
157	No	No

	Actual	Predicted
3886	No	No