

```
In [1]: import pandas as pd
```

```
In [2]: df=pd.read_csv('Social_Network_Ads.csv');
```

```
In [3]: df
```

```
Out[3]:
```

	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
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

```
In [4]: df=df.drop(columns='Gender')
```

```
In [ ]:
```

```
In [5]: y=df['Purchased']
x=df.drop(columns='Purchased')
```

```
In [6]: from sklearn.model_selection import train_test_split
```

```
In [7]: xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.3,random_state=0)
```

```
In [8]: xtrain
```

Out[8]:

	User ID	Age	EstimatedSalary
<b>92</b>	15809823	26	15000
<b>223</b>	15593715	60	102000
<b>234</b>	15619407	38	112000
<b>232</b>	15813113	40	107000
<b>377</b>	15800215	42	53000
...	...	...	...
<b>323</b>	15619465	48	30000
<b>192</b>	15779581	29	43000
<b>117</b>	15591433	36	52000
<b>47</b>	15776348	27	54000
<b>172</b>	15794661	26	118000

280 rows × 3 columns

In [9]: xtest

Out[9]:

	User ID	Age	EstimatedSalary
<b>132</b>	15725660	30	87000
<b>309</b>	15652400	38	50000
<b>341</b>	15776844	35	75000
<b>196</b>	15738448	30	79000
<b>246</b>	15638003	35	50000
...	...	...	...
<b>216</b>	15636023	49	65000
<b>259</b>	15815236	45	131000
<b>49</b>	15793813	31	89000
<b>238</b>	15617877	46	82000
<b>343</b>	15629739	47	51000

120 rows × 3 columns

In [10]: `from sklearn.linear_model import LogisticRegression`In [11]: `model1=LogisticRegression()  
model1.fit(xtrain,ytrain)`Out[11]: 

▼ LogisticRegression

LogisticRegression()

In [12]: `Y_predict=model1.predict(xtest) #predict without scaling`

In [13]: Y\_predict

Out[13]: array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1,  
0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  
1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,  
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,  
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0,  
0, 0, 0, 0, 0, 0, 1, 0, 0, 0]) dtype=int64)

In [14]: `from sklearn.metrics import accuracy_score`  
`accuracy=accuracy_score(Y_predict,ytext)`

In [15]: `accuracy # accuracy without scaling`

Out[15]: 0.7916666666666666

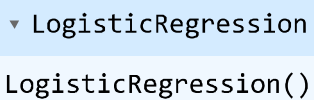
In [16]: `from sklearn.preprocessing import StandardScaler`

In [17]: `std=StandardScaler()`

In [18]: `xtest=std.fit_transform(xtest)`

In [19]: `xtrain=std.fit_transform(xtrain)`

In [20]: `model2=LogisticRegression()`  
`model2.fit(xtrain,ytrain)`

Out[20]: 

In [21]: `Y1_predict=model2.predict(xtest) #predict with scaling`

In [22]: `accuracy=accuracy_score(Y1_predict,ytext)`  
`accuracy`

Out[22]: 0.875

In [23]: `x_new=[[0,0,0],[15794698,30,6000],[26794698,35,600],[17994698,40,7000]]`  
`x_new`

Out[23]: [[0, 0, 0], [15794698, 30, 6000], [26794698, 35, 600], [17994698, 40, 7000]]

In [24]: `x_new=std.fit_transform(x_new)`

In [25]: `new_predict=model2.predict(x_new)`

In [26]: `new_predict`

Out[26]: array([0, 1, 0, 1], dtype=int64)

In [27]: `#to compute confusion matrix`  
`from sklearn.metrics import confusion_matrix,recall_score,precision_score,f1_scc`

```
In [28]: con=confusion_matrix(ytext,Y1_predict)
```

```
In [29]: print(con)
```

```
[[73  6]
 [ 9 32]]
```

```
In [30]: recall=recall_score(ytext,Y1_predict)
```

```
In [31]: recall
```

```
Out[31]: 0.7804878048780488
```

```
In [32]: f1score=f1_score(ytext,Y1_predict)
```

```
In [33]: f1score #inbalance
```

```
Out[33]: 0.810126582278481
```

```
In [34]: ps=precision_score(ytext,Y1_predict)
```

```
In [35]: ps
```

```
Out[35]: 0.8421052631578947
```