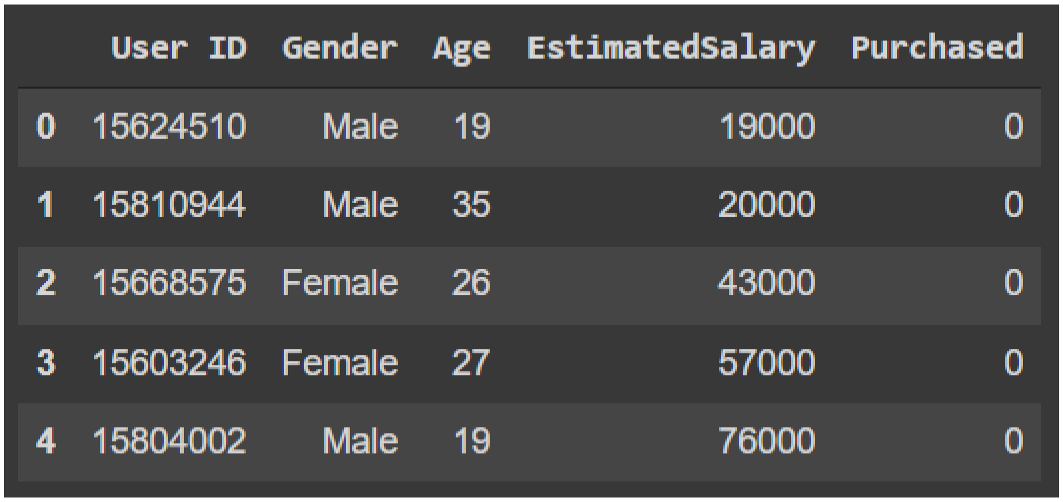
- The parameter “solver” of the logistic regression is used for selecting different solving strategies for classification for better MLE formulation.

Import library:

```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import
train_test_split
from sklearn import preprocessing
```

Read the data:

```
df=pd.read_csv("Social_Network_Ads.csv")
df.head()
```

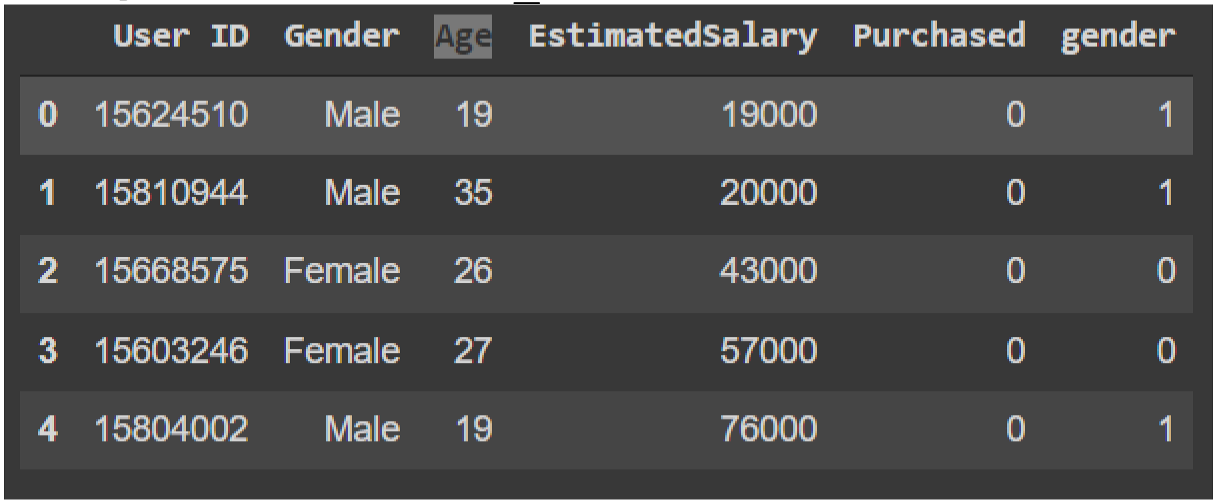


	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

- The data is related to the social networking ads which have the gender, age and estimated salary of the users of that social network.
- The gender is a categorical column that needs to be labelled encoded before feeding the data to the learner.

Encoding the data:

```
le = preprocessing.LabelEncoder()
df['gender']=le.fit_transform(df['Gender'])
```



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

- The encoded outcomes are stored in a new feature called ‘gender’ so that the original is kept unchanged.
- Now, split the data into training and test for training and validating the learner.

Splitting the data:

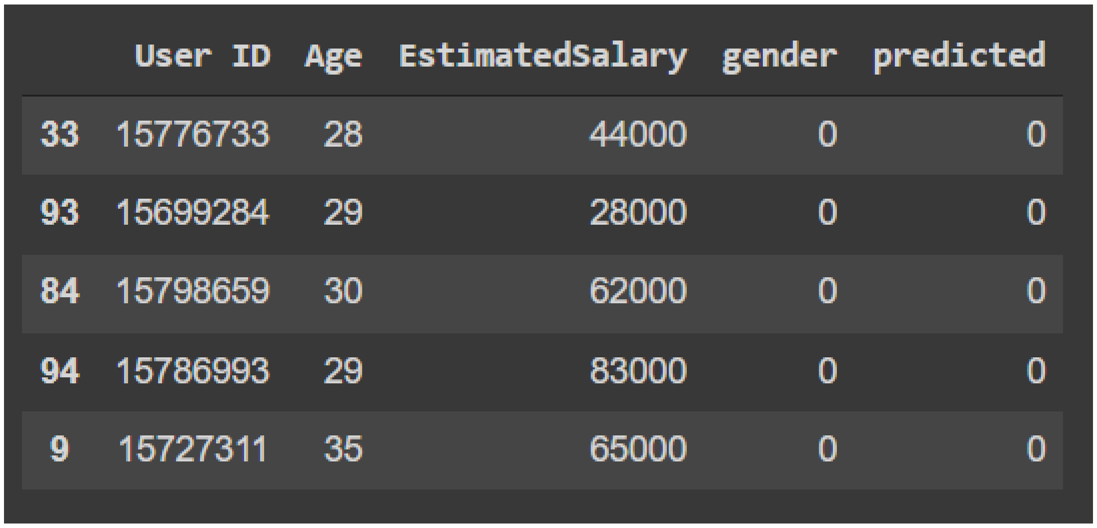
```
X=df.drop(['Purchased','Gender'],axis=1)
y=df['Purchased']
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.30,
random_state=42)
```

This is split into a 70:30 ratio as per standard rules.

Fitting the data in learner:

```
lr=LogisticRegression(max_iter=100,solver='lbfgs')
lr.fit(X_train,y_train)
lr_pred=lr.predict(X_test)
```

```
df_pred=pd.merge(X_test,pd.DataFrame(lr_pred,column
s=['predicted']),left_index=True,right_index=True)
```

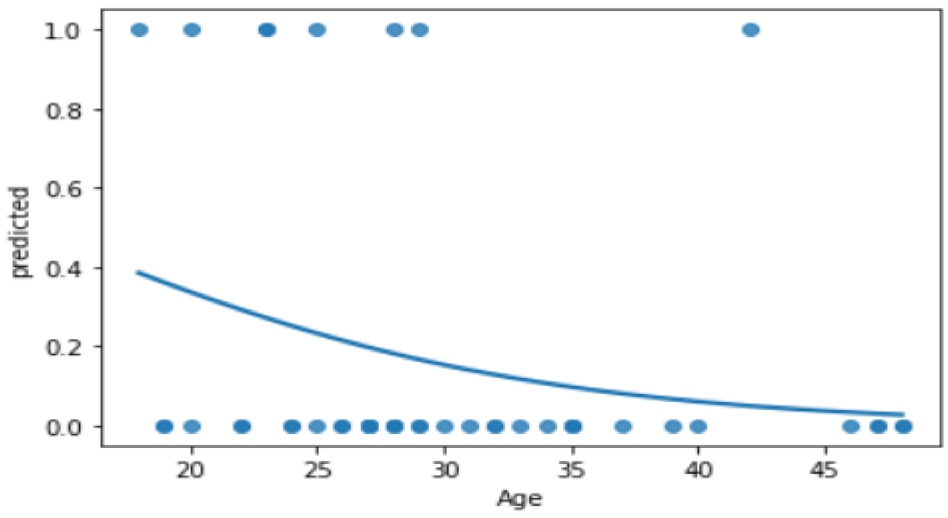


	User ID	Age	EstimatedSalary	gender	predicted
33	15776733	28	44000	0	0
93	15699284	29	28000	0	0
84	15798659	30	62000	0	0
94	15786993	29	83000	0	0
9	15727311	35	65000	0	0

The predicted outcomes are added to the test dataset under the feature ‘predicted’.

Plotting the learner line:

```
sns.regplot(x="Age", y='predicted',data=df_pred
,logistic=True, ci=None)
```



- In the above plot which is between the feature age and prediction, the learner line is formed using the principle of maximum likelihood estimation which helped the Logistic regression model to classify the outcomes.

- So, in the background algorithm picks a probability scaled by age of observing “1” and uses this to calculate the likelihood of observing “0”.

- This will do for all the data points and at last, it will multiply all those likelihoods of data given in the line.

- This process of multiplication will be continued until the maximum likelihood is not found or the best fit line is not found.

Final Words

- The maximum likelihood approach provides a persistent approach to parameter estimation as well as provides mathematical and optimizable properties.

- With a hands-on implementation of this concept in this article, we could understand how Maximum Likelihood Estimation works and how it is used as a backbone of logistic regression for classification.