

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
In [2]: from sklearn.linear_model import LogisticRegression
```

```
In [3]: df_train=pd.read_csv("C8_loan-train.csv")
df_test=pd.read_csv("C8_loan-test.csv")
df_train
```

```
Out[3]:
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIn
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
...
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 13 columns



```
In [4]: df_test
```

```
Out[4]:
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIn
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

367 rows × 12 columns

```
In [5]: df1=df_train.dropna()
```

```
In [6]: df2=df_test.dropna()
```

```
In [7]: df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 480 entries, 1 to 613
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Loan_ID               480 non-null    object
1   Gender                480 non-null    object
2   Married               480 non-null    object
3   Dependents            480 non-null    object
4   Education             480 non-null    object
5   Self_Employed         480 non-null    object
6   ApplicantIncome       480 non-null    int64
7   CoapplicantIncome     480 non-null    float64
8   LoanAmount            480 non-null    float64
9   Loan_Amount_Term      480 non-null    float64
10  Credit_History        480 non-null    float64
11  Property_Area         480 non-null    object
12  Loan_Status           480 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 52.5+ KB
```

```
In [8]: df1.columns
```

```
Out[8]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
              'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
              'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
              dtype='object')
```

```
In [9]: feature_matrix=df1[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
                           'Loan_Amount_Term', 'Credit_History']]
        target_vector=df1[['Self_Employed']]
```

```
In [10]: feature_matrix.shape
```

Out[10]: (480, 5)

In [11]: `target_vector.shape`

Out[11]: (480, 1)

In [12]: `from sklearn.preprocessing import StandardScaler`

In [13]: `fs=StandardScaler().fit_transform(feature_matrix)`

In [14]: `logr=LogisticRegression()
logr.fit(fs,target_vector)`

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
return f(*args, **kwargs)

Out[14]: LogisticRegression()

In [15]: `df2.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 289 entries, 0 to 366
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   Loan_ID               289 non-null   object
 1   Gender                289 non-null   object
 2   Married               289 non-null   object
 3   Dependents            289 non-null   object
 4   Education             289 non-null   object
 5   Self_Employed         289 non-null   object
 6   ApplicantIncome       289 non-null   int64
 7   CoapplicantIncome     289 non-null   int64
 8   LoanAmount            289 non-null   float64
 9   Loan_Amount_Term      289 non-null   float64
10   Credit_History        289 non-null   float64
11   Property_Area         289 non-null   object
dtypes: float64(3), int64(2), object(7)
memory usage: 29.4+ KB
```

In [16]: `df2.columns`

Out[16]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area'], dtype='object')

In [17]: `observation=df2[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History']]`

```
In [18]: prediction=logr.predict(observation)
          print(prediction)
```

[illegible]

```
In [19]: logr.classes_
```

```
Out[19]: array(['No', 'Yes'], dtype=object)
```

```
In [20]: logr.predict_proba(observation)
```

[illegible]

127.0.0.1:8888/nbconvert/html/loan.ipynb?download=false

127.0.0.1:8888/nbconvert/html/loan.ipynb?download=false

127.0.0.1:8888/nbconvert/html/loan.ipynb?download=false

127.0.0.1:8888/nbconvert/html/loan.ipynb?download=false


```
[0.      , 1.      ],  
[0.      , 1.      ],  
[0.      , 1.      ],  
[0.      , 1.      ]])
```

```
In [21]: logr.predict_proba(observation)[0][0]
```

```
Out[21]: 0.0
```