

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

```
In [2]: df=pd.read_csv(r"C8_loan-train.csv")
df
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Lo:
0	LP001002	Male	No	0	Graduate	No	5849	0.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	
...	
609	LP002978	Female	No	0	Graduate	No	2900	0.0	
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	

614 rows × 13 columns



```
In [3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Loan_ID               614 non-null   object
1   Gender                601 non-null   object
2   Married               611 non-null   object
3   Dependents            599 non-null   object
4   Education             614 non-null   object
5   Self_Employed         582 non-null   object
6   ApplicantIncome       614 non-null   int64
7   CoapplicantIncome     614 non-null   float64
8   LoanAmount            592 non-null   float64
9   Loan_Amount_Term      600 non-null   float64
10  Credit_History        564 non-null   float64
11  Property_Area         614 non-null   object
12  Loan_Status           614 non-null   object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

```
In [4]: df=df.dropna()
```

In [5]: `df.describe()`

Out[5]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	480.000000	480.000000	480.000000	480.000000	480.000000
mean	5364.231250	1581.093583	144.735417	342.050000	0.854167
std	5668.251251	2617.692267	80.508164	65.212401	0.353307
min	150.000000	0.000000	9.000000	36.000000	0.000000
25%	2898.750000	0.000000	100.000000	360.000000	1.000000
50%	3859.000000	1084.500000	128.000000	360.000000	1.000000
75%	5852.500000	2253.250000	170.000000	360.000000	1.000000
max	81000.000000	33837.000000	600.000000	480.000000	1.000000

In [6]: `df.columns`

Out[6]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'], dtype='object')

In [7]: `x=df[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History']]`
`y=df['Self_Employed']`
`x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)`
`lr=LogisticRegression()`
`lr.fit(x_train,y_train)`

Out[7]: LogisticRegression()

In [8]: `lr.predict(x_test)`

Out[8]: array(['No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No',
'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
'No'], dtype=object)

In [9]: `lr.score(x_test,y_test)`

Out[9]: 0.8055555555555556

In [10]: `from sklearn.preprocessing import StandardScaler`
`fs=StandardScaler().fit_transform(x)`
`logr=LogisticRegression()`
`logr.fit(fs,y)`

Out[10]: LogisticRegression()

```
In [16]: dff=pd.read_csv(r"C8_loan-test.csv")
dff
```

Out[16]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Lo:
0	LP001015	Male	Yes	0	Graduate	No	5720	0	
1	LP001022	Male	Yes	1	Graduate	No	3076	1500	
2	LP001031	Male	Yes	2	Graduate	No	5000	1800	
3	LP001035	Male	Yes	2	Graduate	No	2340	2546	
4	LP001051	Male	No	0	Not Graduate	No	3276	0	
...
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	1777	
363	LP002975	Male	Yes	0	Graduate	No	4158	709	
364	LP002980	Male	No	0	Graduate	No	3250	1993	
365	LP002986	Male	Yes	0	Graduate	No	5000	2393	
366	LP002989	Male	No	0	Graduate	Yes	9200	0	

367 rows × 12 columns

```
In [17]: dff=dff.dropna()
```

```
In [19]: o=dff[['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','Credit_History']
prediction=logr.predict(o)
print(prediction)

['Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
'Yes']
```

```
In [20]: logr.classes_
```

```
Out[20]: array(['No', 'Yes'], dtype=object)
```

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

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

```
In [22]: logr.predict_proba(o)[0][1]
```

```
Out[22]: 1.0
```

```
In [ ]:
```