In [1]:

```
import numpy as np
```

In [2]:

```
import pandas as pd
```

In [3]:

```
import seaborn as sns
```

In [4]:

```
import matplotlib.pyplot as plt
```

In [5]:

```
df1=pd.read_csv(r'C:\Users\anith\OneDrive\Documents\anil.csv')
```

In [6]:

```
df1
```

Out[6]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantInc |
|---|---|---|---|---|---|---|---|
| **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 [7]:

```
df1.isnull().sum()
```

Out[7]:

```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [8]:

```
df1.dropna(inplace=True)
```

In [10]:

```
df1.isnull().sum()
```

Out[10]:

```
Loan_ID              0
Gender               0
Married              0
Dependents           0
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
dtype: int64
```

In [11]:

```
df1
```

Out[11]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantI |
|---|---|---|---|---|---|---|---|
| **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 |
| **5** | LP001011 | Male | Yes | 2 | Graduate | Yes | 5417 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **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 |

480 rows × 13 columns

In [12]:

```
df1.reset_index(inplace=True)
```

In [13]:

```
df1
```

Out[13]:

| | index | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | Appli |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | LP001003 | Male | Yes | 1 | Graduate | No | 4583 |
| **1** | 2 | LP001005 | Male | Yes | 0 | Graduate | Yes | 3000 |
| **2** | 3 | LP001006 | Male | Yes | 0 | Not Graduate | No | 2583 |
| **3** | 4 | LP001008 | Male | No | 0 | Graduate | No | 6000 |
| **4** | 5 | LP001011 | Male | Yes | 2 | Graduate | Yes | 5417 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **475** | 609 | LP002978 | Female | No | 0 | Graduate | No | 2900 |
| **476** | 610 | LP002979 | Male | Yes | 3+ | Graduate | No | 4106 |
| **477** | 611 | LP002983 | Male | Yes | 1 | Graduate | No | 8072 |
| **478** | 612 | LP002984 | Male | Yes | 2 | Graduate | No | 7583 |
| **479** | 613 | LP002990 | Female | No | 0 | Graduate | Yes | 4583 |

480 rows × 14 columns

In [14]:

```
df1=df1.replace({'Gender':{'Male':1,'Female':0},'Married':{'Yes':1,'No':0},'Education':{'Gra
```

In [15]:

```
df1
```

Out[15]:

| | index | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | Appli |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | LP001003 | 1 | 1 | 1 | 1 | 0 | 4583 |
| **1** | 2 | LP001005 | 1 | 1 | 0 | 1 | 1 | 3000 |
| **2** | 3 | LP001006 | 1 | 1 | 0 | 0 | 0 | 2583 |
| **3** | 4 | LP001008 | 1 | 0 | 0 | 1 | 0 | 6000 |
| **4** | 5 | LP001011 | 1 | 1 | 2 | 1 | 1 | 5417 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **475** | 609 | LP002978 | 0 | 0 | 0 | 1 | 0 | 2900 |
| **476** | 610 | LP002979 | 1 | 1 | 3+ | 1 | 0 | 4106 |
| **477** | 611 | LP002983 | 1 | 1 | 1 | 1 | 0 | 8072 |
| **478** | 612 | LP002984 | 1 | 1 | 2 | 1 | 0 | 7583 |
| **479** | 613 | LP002990 | 0 | 0 | 0 | 1 | 1 | 4583 |

480 rows × 14 columns

In [16]:

```
df1['Dependents'].unique()
```

Out[16]:

```
array(['1', '0', '2', '3+'], dtype=object)
```

In [17]:

```
df1['Dependents']=df1['Dependents'].replace(to_replace='3+',value=4)
```

In [18]:

```
df1['Dependents']=df1['Dependents'].astype('int')
```

In [19]:

```
df1['Dependents'].unique()
```

Out[19]:

```
array([1, 0, 2, 4])
```

In [20]:

```
X = df1.drop(columns=["Loan_ID", "Loan_Status"], axis=1)
```

In [52]:

```
Y = df1["Loan_Status"]
```

In [26]:

```
Y
```

Out[26]:

```
0      N
1      Y
2      Y
3      Y
4      Y
      ..
475    Y
476    Y
477    Y
478    Y
479    N
Name: Loan_Status, Length: 480, dtype: object
```

In [23]:

```
from sklearn. linear_model import LogisticRegression
from sklearn .model_selection import train_test_split
```

In [24]:

```
lr=LogisticRegression()
```

In [25]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, stratify=Y, random_s
```

In [27]:

```
X_train.shape
```

Out[27]:

```
(384, 12)
```

In [29]:

```
Y_train.shape
```

Out[29]:

```
(384,)
```

In [30]:

```
X_test.shape
```

Out[30]:

```
(96, 12)
```

In [31]:

```
Y_test.shape
```

Out[31]:

```
(96,)
```

In [32]:

```
from sklearn.linear_model import LogisticRegression
```

In [33]:

```
lr=LogisticRegression()
```

In [34]:

```
lr.fit(X_train,Y_train)
```

```
C:\Users\anith\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceW
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[34]:

```
LogisticRegression
LogisticRegression()
```

In [35]:

```
from sklearn.metrics import accuracy_score
```

In [36]:

```
pred=lr.predict(X_test)
```

In [37]:

```
accuracy_score(Y_test,pred)
```

Out[37]:

```
0.8020833333333334
```

In [41]:

```
Y_test.astype('str')
```

```
Out[41]:
         159    Y
         429    Y
         400    N
         324    Y
         213    Y
               ..
         156    Y
         267    Y
         473    Y
         424    Y
         365    N
         Name: Loan_Status, Length: 96, dtype: object
```