In [31]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [32]:
```python
Data=pd.read_csv("LoanPrediction(ML).csv")
Data
```

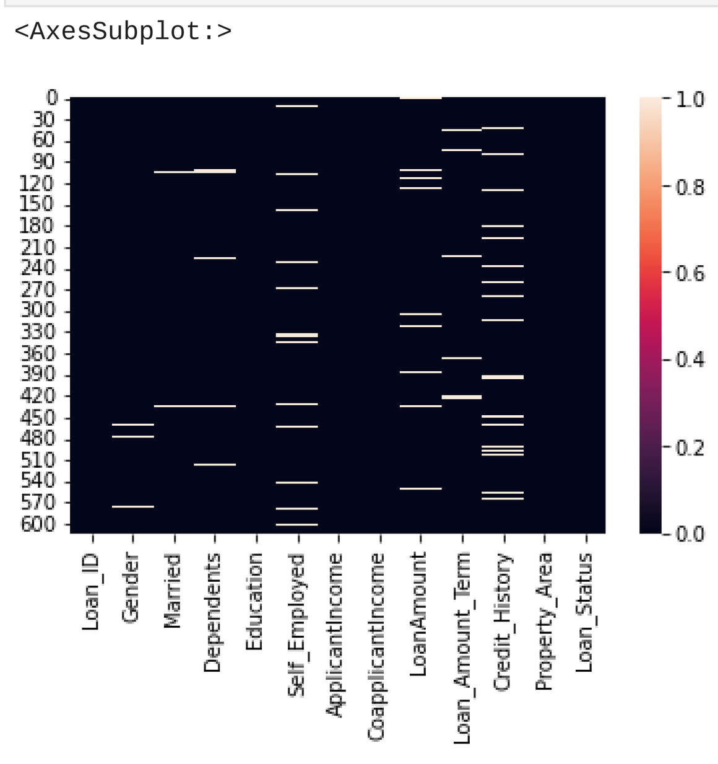
Out[32]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y
...	...	...	...	...	...	...	...	...	...	...	...	...	...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	1.0	Rural	Y
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	1.0	Rural	Y
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	1.0	Urban	Y
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	1.0	Urban	Y
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0	Semiurban	N

614 rows × 13 columns

In [33]:
```python
#Checking Null values
sns.heatmap(Data.isnull())
```

Out[33]: <AxesSubplot:>



In [34]:
```python
#Checking Null values with respect to sections
Data.isnull().sum()
```

Out[34]:
```
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 [35]:
```python
#Checking Non-null values for each section
Data.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 [73]:
```python
#Changing the Text into Int values
Data['Gender'].replace(['Male','Female'],[0,1],inplace=True)
Data['Dependents'].replace(['3+'],[3],inplace=True)
Data['Loan_Status'].replace(['Y','N'],[1,0],inplace=True)
Data['Married'].replace(['Yes','No'],[1,0],inplace=True)
Data['Self_Employed'].replace(['Yes','No'],[1,0],inplace=True)
Data['Property_Area'].replace(['Urban','Semiurban','Rural'],[3,2,1],inplace=True)
Data['Education'].replace(['Graduate','Not Graduate'],[1,0],inplace=True)
```

In [74]:
```python
#Filling the Null values
```

In [75]:
```python
Data['Gender'].value_counts()
```

Out[75]:
```
0.0    489
1.0    125
Name: Gender, dtype: int64
```

In [76]:
```python
Data.Gender=Data.Gender.fillna('0')
```

In [77]:
```python
Data['Married'].value_counts()
```

Out[77]:
```
1.0    401
0.0    213
Name: Married, dtype: int64
```

In [78]:
```python
Data.Married=Data.Married.fillna('1')
```

In [79]:
```python
Data['Dependents'].value_counts()
```

Out[79]:
```
0    360
1    102
2    101
3     51
Name: Dependents, dtype: int64
```

In [80]:
```python
Data.Dependents=Data.Dependents.fillna('0')
```

In [81]:
```python
Data['Self_Employed'].value_counts()
```

Out[81]:
```
0.0    500
1.0     82
0       32
Name: Self_Employed, dtype: int64
```

In [82]:
```python
Data.Self_Employed=Data.Self_Employed.fillna('0')
```

In [83]:
```python
Data.LoanAmount=Data.LoanAmount.fillna(Data.LoanAmount.mean())
```

In [84]:
```python
Data['Loan_Amount_Term'].value_counts()
```

Out[84]:
```
360.0    512
180.0     44
480.0     15
0         14
300.0     13
240.0      4
84.0       4
120.0      3
60.0       2
36.0       2
12.0       1
Name: Loan_Amount_Term, dtype: int64
```

In [85]:
```python
Data.Loan_Amount_Term=Data.Loan_Amount_Term.fillna('360')
```

In [86]:
```python
Data['Credit_History'].value_counts()
```

Out[86]:
```
1.0    475
0.0     89
1       50
Name: Credit_History, dtype: int64
```

In [87]:
```python
Data.Credit_History=Data.Credit_History.fillna('1')
```

In [88]:
```python
#Again Checking if Null Vaues
Data.isnull().sum().sum()
```

Out[88]: 0

In [89]:
```python
Data.head(n=5)
```

Out[89]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	0.0	0.0	0	1	0.0	5849	0.0	146.412162	360.0	1.0	3	1
1	LP001003	0.0	1.0	1	1	0.0	4583	1508.0	128.000000	360.0	1.0	1	0
2	LP001005	0.0	1.0	0	1	1.0	3000	0.0	66.000000	360.0	1.0	3	1
3	LP001006	0.0	1.0	0	0	0.0	2583	2358.0	120.000000	360.0	1.0	3	1
4	LP001008	0.0	0.0	0	1	0.0	6000	0.0	141.000000	360.0	1.0	3	1

In [90]:
```python
#Data Visualization
x=Data[['Married','Dependents','Education','Self_Employed','ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','Credit_History','Property_Area']]
y=Data['Loan_Status']
```

In [91]:
```python
x.head(n=4)
```

Out[91]:

	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	0.0	0	1	0.0	5849	0.0	146.412162	360.0	1.0	3
1	1.0	1	1	0.0	4583	1508.0	128.000000	360.0	1.0	1
2	1.0	0	1	1.0	3000	0.0	66.000000	360.0	1.0	3
3	1.0	0	0	0.0	2583	2358.0	120.000000	360.0	1.0	3

In [92]:
```python
y.head(n=4)
```

Out[92]:
```
0    1
1    0
2    1
3    1
Name: Loan_Status, dtype: int64
```

In [93]:
```python
from sklearn.model_selection import train_test_split
```

In [94]:
```python
X_Train,X_Test,Y_Train,Y_Test=train_test_split(x,y,test_size=0.25,random_state=0)
```

In [95]:
```python
from sklearn.linear_model import LogisticRegression
```

In [96]:
```python
My_mdl=LogisticRegression()
```

In [97]:
```python
My_mdl.fit(X_Train,Y_Train)
```

Out[97]: LogisticRegression()

In [98]:
```python
Y_Pred=My_mdl.predict(X_Test)
```

In [99]:
```python
Y_Pred
```

Out[99]:
```
array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0],
      dtype=int64)
```

In [100…
```python
from sklearn.metrics import accuracy_score
```

In [101…
```python
print('So the model accuracy is ::::',accuracy_score(Y_Test, Y_Pred)*100 )
```

```
So the model accuracy is :::: 83.11688311688312
```

In [ ]: