Assignment-8

August 24, 2024

1 Name:Krupanidhi Jena

2 Reg NO:2141010032

3 Sec:16(p)

4 B.tech (CSE)

5 importing libraries

[162]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[164]:
```
df=pd.read_csv('Training Dataset.csv')
df.head(30)
```

[164]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
0	LP001002	Male	No	0	Graduate	No	
1	LP001003	Male	Yes	1	Graduate	No	
2	LP001005	Male	Yes	0	Graduate	Yes	
3	LP001006	Male	Yes	0	Not Graduate	No	
4	LP001008	Male	No	0	Graduate	No	
5	LP001011	Male	Yes	2	Graduate	Yes	
6	LP001013	Male	Yes	0	Not Graduate	No	
7	LP001014	Male	Yes	3+	Graduate	No	
8	LP001018	Male	Yes	2	Graduate	No	
9	LP001020	Male	Yes	1	Graduate	No	
10	LP001024	Male	Yes	2	Graduate	No	
11	LP001027	Male	Yes	2	Graduate	NaN	
12	LP001028	Male	Yes	2	Graduate	No	
13	LP001029	Male	No	0	Graduate	No	
14	LP001030	Male	Yes	2	Graduate	No	
15	LP001032	Male	No	0	Graduate	No	
16	LP001034	Male	No	1	Not Graduate	No	
17	LP001036	Female	No	0	Graduate	No	
18	LP001038	Male	Yes	0	Not Graduate	No	

```
19  LP001041    Male     Yes          0      Graduate           NaN
20  LP001043    Male     Yes          0  Not Graduate            No
21  LP001046    Male     Yes          1      Graduate            No
22  LP001047    Male     Yes          0  Not Graduate            No
23  LP001050     NaN     Yes          2  Not Graduate            No
24  LP001052    Male     Yes          1      Graduate           NaN
25  LP001066    Male     Yes          0      Graduate           Yes
26  LP001068    Male     Yes          0      Graduate            No
27  LP001073    Male     Yes          2  Not Graduate            No
28  LP001086    Male      No          0  Not Graduate            No
29  LP001087  Female      No          2      Graduate           NaN

    ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0              5849                0.0         NaN             360.0
1              4583             1508.0       128.0             360.0
2              3000                0.0        66.0             360.0
3              2583             2358.0       120.0             360.0
4              6000                0.0       141.0             360.0
5              5417             4196.0       267.0             360.0
6              2333             1516.0        95.0             360.0
7              3036             2504.0       158.0             360.0
8              4006             1526.0       168.0             360.0
9             12841            10968.0       349.0             360.0
10             3200              700.0        70.0             360.0
11             2500             1840.0       109.0             360.0
12             3073             8106.0       200.0             360.0
13             1853             2840.0       114.0             360.0
14             1299             1086.0        17.0             120.0
15             4950                0.0       125.0             360.0
16             3596                0.0       100.0             240.0
17             3510                0.0        76.0             360.0
18             4887                0.0       133.0             360.0
19             2600             3500.0       115.0               NaN
20             7660                0.0       104.0             360.0
21             5955             5625.0       315.0             360.0
22             2600             1911.0       116.0             360.0
23             3365             1917.0       112.0             360.0
24             3717             2925.0       151.0             360.0
25             9560                0.0       191.0             360.0
26             2799             2253.0       122.0             360.0
27             4226             1040.0       110.0             360.0
28             1442                0.0        35.0             360.0
29             3750             2083.0       120.0             360.0

    Credit_History Property_Area Loan_Status
0              1.0         Urban           Y
1              1.0         Rural           N
```

```
2              1.0         Urban           Y
3              1.0         Urban           Y
4              1.0         Urban           Y
5              1.0         Urban           Y
6              1.0         Urban           Y
7              0.0     Semiurban           N
8              1.0         Urban           Y
9              1.0     Semiurban           N
10             1.0         Urban           Y
11             1.0         Urban           Y
12             1.0         Urban           Y
13             1.0         Rural           N
14             1.0         Urban           Y
15             1.0         Urban           Y
16             NaN         Urban           Y
17             0.0         Urban           N
18             1.0         Rural           N
19             1.0         Urban           Y
20             0.0         Urban           N
21             1.0         Urban           Y
22             0.0     Semiurban           N
23             0.0         Rural           N
24             NaN     Semiurban           N
25             1.0     Semiurban           Y
26             1.0     Semiurban           Y
27             1.0         Urban           Y
28             1.0         Urban           N
29             1.0     Semiurban           Y
```

[166]:
```
df.shape
```

[166]:
```
(614, 13)
```

6 Null value finding and removing them by using fillna or dropna

[169]:
```
df.isnull().sum()
```

[169]:
```
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
```

df.dtypes

[172]:
```
df_filled=df.dropna(subset=['Married'])
```

7 Encoding categorical values into numerical value for preparing for predictive modeling using label encoder

[175]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df_filled['Loan_ID']=le.fit_transform(df_filled['Loan_ID'])
df_filled['Gender']=le.fit_transform(df_filled['Gender'])
df_filled['Married']=le.fit_transform(df_filled['Married'])
df_filled['Dependents']=le.fit_transform(df_filled['Dependents'])
df_filled['Education']=le.fit_transform(df_filled['Education'])
df_filled['Self_Employed']=le.fit_transform(df_filled['Self_Employed'])
df_filled['Property_Area']=le.fit_transform(df_filled['Property_Area'])
```

```
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Loan_ID']=le.fit_transform(df_filled['Loan_ID'])
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Gender']=le.fit_transform(df_filled['Gender'])
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:5:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Married']=le.fit_transform(df_filled['Married'])
```

```
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Dependents']=le.fit_transform(df_filled['Dependents'])
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Education']=le.fit_transform(df_filled['Education'])
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:8:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Self_Employed']=le.fit_transform(df_filled['Self_Employed'])
C:\Users\krupa\AppData\Local\Temp\ipykernel_3224\4133105678.py:9:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_filled['Property_Area']=le.fit_transform(df_filled['Property_Area'])
```

[177]:
```
df_filled
```

[177]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
0	0	1	0	0	0	0	
1	1	1	1	1	0	0	
2	2	1	1	0	0	1	
3	3	1	1	0	1	0	
4	4	1	0	0	0	0	
..	…	…	…	…	…	…	
609	606	0	0	0	0	0	
610	607	1	1	3	0	0	
611	608	1	1	1	0	0	
612	609	1	1	2	0	0	
613	610	0	0	0	0	1	

```
     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0               5849                0.0         NaN             360.0
1               4583             1508.0       128.0             360.0
2               3000                0.0        66.0             360.0
3               2583             2358.0       120.0             360.0
4               6000                0.0       141.0             360.0
..               …                  …          …                 …
609             2900                0.0        71.0             360.0
610             4106                0.0        40.0             180.0
611             8072              240.0       253.0             360.0
612             7583                0.0       187.0             360.0
613             4583                0.0       133.0             360.0

     Credit_History  Property_Area Loan_Status
0               1.0              2           Y
1               1.0              0           N
2               1.0              2           Y
3               1.0              2           Y
4               1.0              2           Y
..              …              …         …
609             1.0              0           Y
610             1.0              0           Y
611             1.0              2           Y
612             1.0              2           Y
613             0.0              1           N

[611 rows x 13 columns]
```

[179]:
```
df_filled=df_filled.fillna(df_filled[['Gender','Dependents','Self_Employed']].
↪mode())
```

[181]:
```
df_filled1=df_filled.
↪fillna(df_filled[['LoanAmount','Loan_Amount_Term','Credit_History']].mean())
```

[183]:
```
df_filled1.isnull().sum()
```

[183]:
```
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
```

8 splitting the data for use in the model

[186]:
```
x=df_filled1.iloc[:, :-1].values
y=df_filled1.iloc[:, -1].values
```

[188]:
```
x
```

[188]:
```
array([[  0.,   1.,   0., …, 360.,   1.,   2.],
       [  1.,   1.,   1., …, 360.,   1.,   0.],
       [  2.,   1.,   1., …, 360.,   1.,   2.],
       …,
       [608.,   1.,   1., …, 360.,   1.,   2.],
       [609.,   1.,   1., …, 360.,   1.,   2.],
       [610.,   0.,   0., …, 360.,   0.,   1.]])
```

[190]:
```
y
```

[190]:
```
array(['Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y',
       'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y',
       'N', 'N', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N',
       'N', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N',
       'N', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'N', 'N', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'N', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y',
       'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y',
       'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'N', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y',
       'Y', 'N', 'N', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'N', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y',
```

```
       'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y',
       'N', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y',
       'N', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N',
       'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'N',
       'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'N', 'N', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'N', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'Y', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y',
       'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N',
       'Y', 'N', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'N', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y', 'N', 'N',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y', 'Y',
       'Y', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N'],
      dtype=object)
```

9 Train test split

[193]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

[195]:
```
x_train
```

[195]:
```
array([[ 90.        ,   1.        ,   1.        , …, 360.        ,
           1.        ,   1.        ],
       [530.        ,   1.        ,   0.        , …, 360.        ,
           0.84135472,   1.        ],
       [449.        ,   1.        ,   1.        , …, 360.        ,
           0.        ,   0.        ],
       …,
       [359.        ,   1.        ,   1.        , …, 360.        ,
           1.        ,   1.        ],
       [192.        ,   1.        ,   0.        , …, 360.        ,
           1.        ,   1.        ],
       [559.        ,   1.        ,   1.        , …, 360.        ,
           1.        ,   0.        ]])
```

[197]:
```
y_train
```

[197]:
```
array(['Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N',
       'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'N',
       'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'N', 'N',
       'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'N', 'Y',
       'N', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'N', 'Y', 'N', 'N', 'Y',
       'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N',
       'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'N', 'N', 'Y', 'Y', 'Y',
       'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'N', 'Y', 'N', 'Y',
       'Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'N',
       'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y',
       'Y', 'N', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y',
       'N', 'N', 'N', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N',
       'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y',
       'N', 'N', 'Y', 'N', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'N',
       'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'Y', 'N', 'N', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'N', 'N',
       'Y', 'Y', 'N', 'N', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'N', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y'], dtype=object)
```

10 modeling using randomforest classifier because its accuracy score is higher than other classification models for this dataset it has almost 80% accuracy result meanwhile others have less accurate for this paticular dataset

[200]:
```
from sklearn.ensemble import RandomForestClassifier
classifier=RandomForestClassifier(n_estimators=100,criterion='entropy',random_state=0)
classifier.fit(x_train,y_train)
```

[200]:
```
RandomForestClassifier(criterion='entropy', random_state=0)
```

[202]:
```
y_pred=classifier.predict(x_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1),y_test.
  ↪reshape(len(y_test),1)),1))
```

```
[['Y' 'N']
 ['Y' 'N']
 ['Y' 'Y']
 ['Y' 'N']
 ['Y' 'Y']
 ['N' 'N']
 ['Y' 'Y']
 ['Y' 'N']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['N' 'N']
 ['Y' 'Y']
 ['Y' 'N']
 ['Y' 'Y']
 ['Y' 'Y']
 ['N' 'N']
 ['N' 'N']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']
 ['N' 'N']
 ['Y' 'Y']
 ['Y' 'Y']
 ['Y' 'Y']]
```

```
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['N' 'N']
['Y' 'Y']
['Y' 'N']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'N']
['Y' 'Y']
['N' 'N']
['Y' 'N']
['Y' 'N']
['Y' 'Y']
['Y' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'N']
['Y' 'Y']
['Y' 'N']
['N' 'N']
['N' 'N']
```

```
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['N' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'N']
['Y' 'N']
['N' 'N']
['N' 'N']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['N' 'N']
['Y' 'Y']
['Y' 'N']
['Y' 'N']
['N' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'Y']
['Y' 'N']
['Y' 'N']
['Y' 'Y']
['Y' 'N']
['N' 'N']
['Y' 'Y']
['Y' 'N']
['Y' 'N']
['Y' 'Y']
['Y' 'Y']
['Y' 'N']
['Y' 'Y']]
```

[204]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
print(cm)
```

```
accuracy_score(y_test,y_pred)
```

```
[[23 22]
 [ 1 77]]
```

[204]: 0.8130081300813008

11 applying the model to the dataset that they given and predicting the result

[207]:
```
df_test=pd.read_csv('Test Dataset.csv')
df_test
```

[207]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
0	LP001015	Male	Yes	0	Graduate	No	
1	LP001022	Male	Yes	1	Graduate	No	
2	LP001031	Male	Yes	2	Graduate	No	
3	LP001035	Male	Yes	2	Graduate	No	
4	LP001051	Male	No	0	Not Graduate	No	
..	…	…	…	…	…	…	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	
363	LP002975	Male	Yes	0	Graduate	No	
364	LP002980	Male	No	0	Graduate	No	
365	LP002986	Male	Yes	0	Graduate	No	
366	LP002989	Male	No	0	Graduate	Yes	

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	\
0	5720	0	110.0	360.0	
1	3076	1500	126.0	360.0	
2	5000	1800	208.0	360.0	
3	2340	2546	100.0	360.0	
4	3276	0	78.0	360.0	
..	…	…	…	…	
362	4009	1777	113.0	360.0	
363	4158	709	115.0	360.0	
364	3250	1993	126.0	360.0	
365	5000	2393	158.0	360.0	
366	9200	0	98.0	180.0	

	Credit_History	Property_Area
0	1.0	Urban
1	1.0	Urban
2	1.0	Urban
3	NaN	Urban
4	1.0	Urban
..	…	…
362	1.0	Urban

```
363             1.0         Urban
364             NaN     Semiurban
365             1.0         Rural
366             1.0         Rural

[367 rows x 12 columns]
```

[209]:
```
df_filledd=df_test.dropna(subset=['Married'])
```

[211]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df_filledd['Loan_ID']=le.fit_transform(df_filledd['Loan_ID'])
df_filledd['Gender']=le.fit_transform(df_filledd['Gender'])
df_filledd['Married']=le.fit_transform(df_filledd['Married'])
df_filledd['Dependents']=le.fit_transform(df_filledd['Dependents'])
df_filledd['Education']=le.fit_transform(df_filledd['Education'])
df_filledd['Self_Employed']=le.fit_transform(df_filledd['Self_Employed'])
df_filledd['Property_Area']=le.fit_transform(df_filledd['Property_Area'])
```

[213]:
```
df_filledd=df_filledd.
 ↪fillna(df_filledd[['Gender','Dependents','Self_Employed']].mode())
```

[215]:
```
df_filledd1=df_filledd.
 ↪fillna(df_filledd[['LoanAmount','Loan_Amount_Term','Credit_History']].mean())
```

[217]:
```
test_pred=classifier.predict(df_filledd1)
test_pred.reshape(len(df_filledd1),1)
```

```
C:\Users\krupa\anaconda3\Lib\site-packages\sklearn\base.py:486: UserWarning: X
has feature names, but RandomForestClassifier was fitted without feature names
  warnings.warn(
```

[217]:
```
array([['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['N'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['Y'],
       ['N'],
       ['Y'],
       ['Y'],
```

['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],

```
['N'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
```

```
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['N'],
['N'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
```

```
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
```

```
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
```

```
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['N'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['N'],
```

```
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
```

```
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['N'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y'],
['Y']], dtype=object)
```

12 thankyou Sir

[ ]: