import required library's

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

Load the datasets

In [2]:
```
loan_data = pd.read_csv('loan-data.csv')
loan_data.head()
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Terr
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.

In [3]:
```
loan_data.shape
```

Out[3]:
```
(614, 13)
```

In [4]:
```
loan_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 [5]:
```
loan_data.describe()
```

Out[5]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

In [6]:
```
null = loan_data.isnull().sum()
```

In [7]:
```
petcentage = (null/len(loan_data))*100
petcentage
```

Out[7]:
```
Loan_ID              0.000000
Gender               2.117264
Married              0.488599
Dependents           2.442997
Education            0.000000
Self_Employed        5.211726
ApplicantIncome      0.000000
CoapplicantIncome    0.000000
LoanAmount           3.583062
Loan_Amount_Term     2.280130
Credit_History       8.143322
Property_Area        0.000000
Loan_Status          0.000000
dtype: float64
```

In [8]:
```
loan_data['Loan_Status'].unique()
```

Out[8]:
```
array(['Y', 'N'], dtype=object)
```

In [9]:
```
# crosstab create a cross-tabulation or contingency table between two columns from a DataFrame
pd.crosstab(loan_data['Credit_History'],loan_data['Loan_Status'], margins=True)
```

Out[9]:

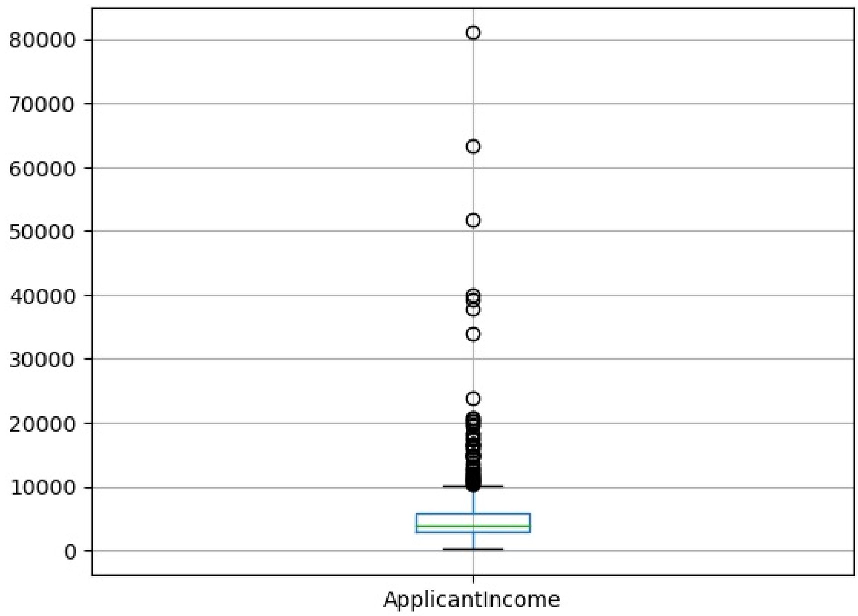
Loan_Status	N	Y	All
Credit_History			
0.0	82	7	89
1.0	97	378	475
All	179	385	564

In [10]:
```
loan_data['ApplicantIncome'].value_counts()
```

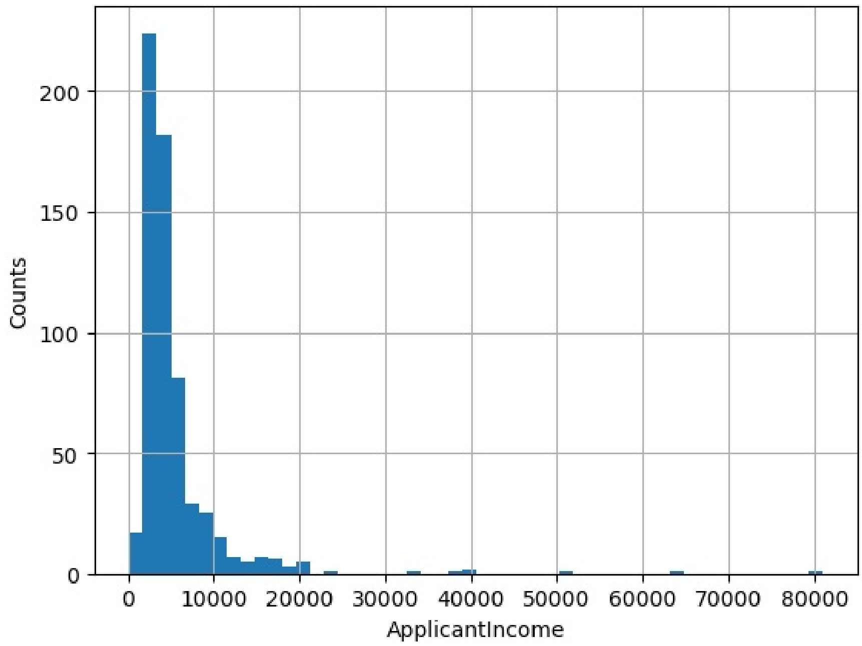
Out[10]:
```
2500    9
4583    6
6000    6
2600    6
3333    5
       ..
3244    1
4408    1
3917    1
3992    1
7583    1
Name: ApplicantIncome, Length: 505, dtype: int64
```

In [11]:
```
loan_data.boxplot(column='ApplicantIncome')
```

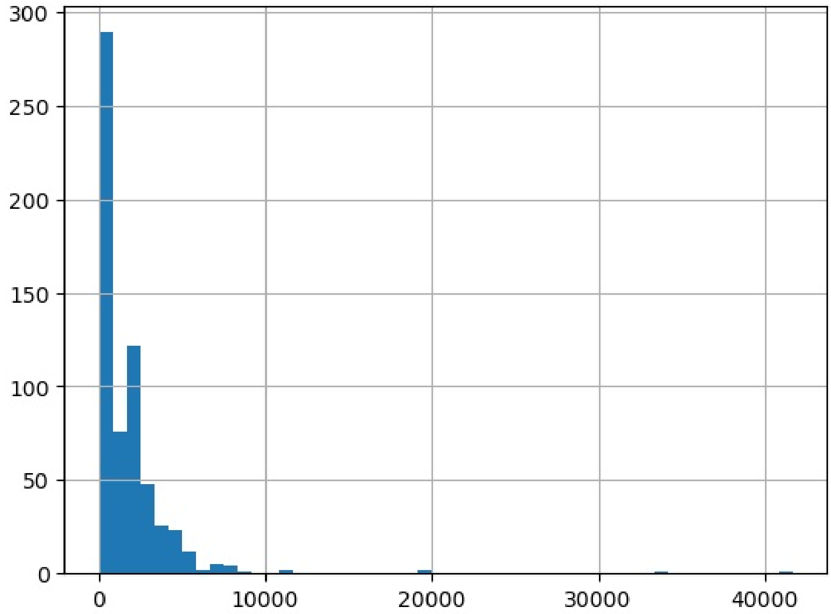
Out[11]:
```
<Axes: >
```



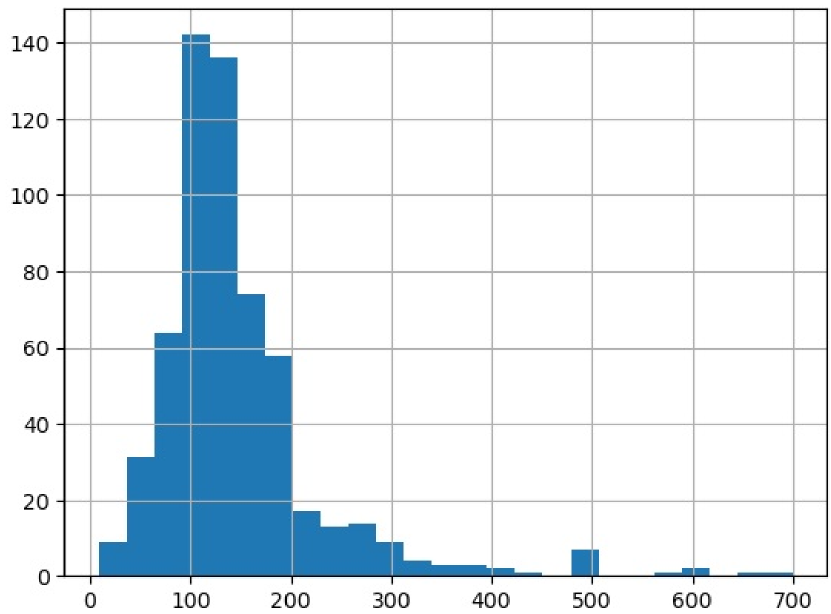
In [12]:
```
loan_data['ApplicantIncome'].hist(bins=50)
plt.xlabel('ApplicantIncome')
plt.ylabel('Counts')
plt.show()
```



In [13]:
```
loan_data['CoapplicantIncome'].hist(bins=50)
```
Out[13]: <Axes: >

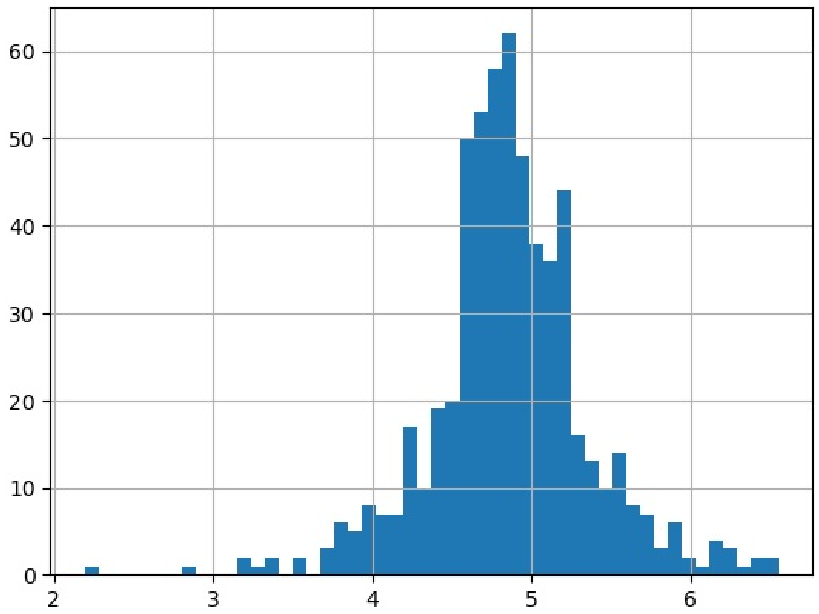


In [14]:
```
loan_data['LoanAmount'].hist(bins=25)
```
Out[14]: <Axes: >



In [15]:
```python
#make log value of  LoanAmount
loan_data['LoanAmount_log'] = np.log(loan_data['LoanAmount'])
loan_data['LoanAmount_log'].hist(bins=50)
```

Out[15]:
```
<Axes: >
```



In [16]:
```python
loan_data.isnull().sum()
```

Out[16]:
```
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
LoanAmount_log       22
dtype: int64
```

In [17]:
```python
loan_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 14 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
 13  LoanAmount_log     592 non-null    float64
dtypes: float64(5), int64(1), object(8)
memory usage: 67.3+ KB
```

fill null values

categrical value fill with mode value of data

In [18]:
```python
loan_data['Gender'].fillna(loan_data['Gender'].mode()[0], inplace=True)
```

In [19]:
```python
loan_data['Married'].fillna(loan_data['Married'].mode()[0], inplace=True)
```

In [20]:
```python
loan_data['Dependents'].fillna(loan_data['Dependents'].mode()[0], inplace=True)
```

In [21]:
```python
loan_data['Self_Employed'].fillna(loan_data['Self_Employed'].mode()[0], inplace=True)
```

In [22]:
```python
loan_data['Loan_Amount_Term'].fillna(loan_data['Loan_Amount_Term'].mode()[0], inplace=True)
```

In [23]:
```python
loan_data['Credit_History'].fillna(loan_data['Credit_History'].mode()[0], inplace=True)
```

numerical value fill with mean value of data

In [24]:
```python
# numerical value
loan_data.LoanAmount = loan_data.LoanAmount.fillna(loan_data.LoanAmount.mean())
loan_data.LoanAmount_log = loan_data.LoanAmount_log.fillna(loan_data.LoanAmount_log.mean())
```

In [25]:
```python
loan_data.isnull().sum()
```

Out[25]:
```
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
LoanAmount_log       0
dtype: int64
```

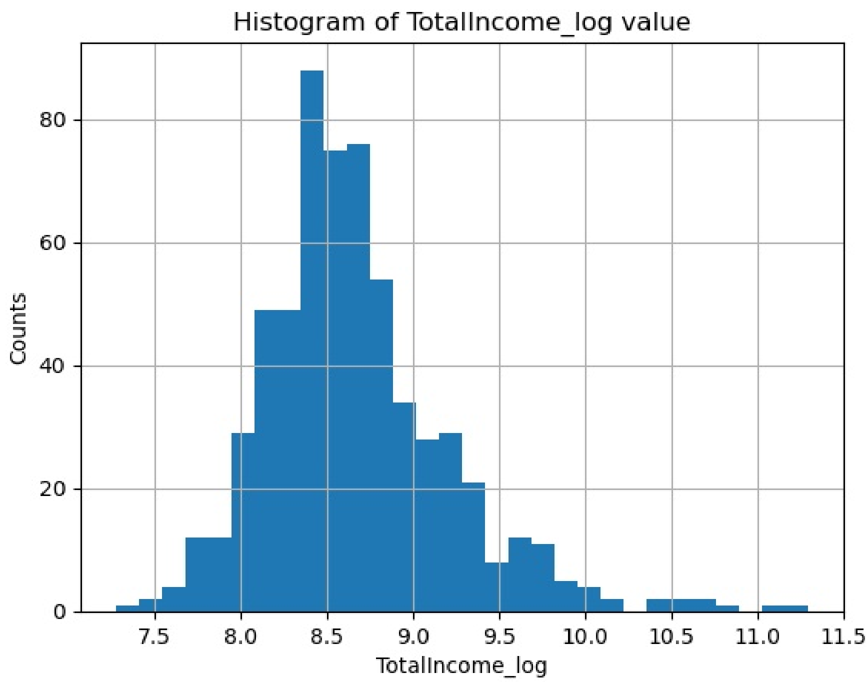
In [26]:
```python
#TotalIncome  is the combnation of ApplicantIncome and CoapplicantIncome
loan_data['TotalIncome'] = loan_data['ApplicantIncome'] + loan_data['CoapplicantIncome']

# log value of TotalIncome column
loan_data['TotalIncome_log'] = np.log(loan_data['TotalIncome'])
```

In [27]:
```python
loan_data.head()
```

Out[27]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Terr
0	LP001002	Male	No	0	Graduate	No	5849	0.0	146.412162	360.
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.000000	360.
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.000000	360.
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.000000	360.
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.000000	360.

In [28]:
```python
loan_data['TotalIncome_log'].hist(bins=30)
plt.title('Histogram of TotalIncome_log value')
plt.xlabel('TotalIncome_log')
plt.ylabel('Counts')
plt.show()
```



In [29]:
```python
x = loan_data.iloc[:,np.r_[1:5,9:11,13:15]].values #colums[1,2,3,4,9,10,13,14]
```

```python
y = loan_data.iloc[:,12].values
```

In [30]:
```python
loan_data.head(2)
```

Out[30]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Terr
0	LP001002	Male	No	0	Graduate	No	5849	0.0	146.412162	360.
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.000000	360.

In [31]:
```python
x
```

Out[31]:
```
array([['Male', 'No', '0', ..., 1.0, 4.857444178729352, 5849.0],
       ['Male', 'Yes', '1', ..., 1.0, 4.852030263919617, 6091.0],
       ['Male', 'Yes', '0', ..., 1.0, 4.189654742026425, 3000.0],
       ...,
       ['Male', 'Yes', '1', ..., 1.0, 5.53338948872752, 8312.0],
       ['Male', 'Yes', '2', ..., 1.0, 5.231108616854587, 7583.0],
       ['Female', 'No', '0', ..., 0.0, 4.890349128221754, 4583.0]],
      dtype=object)
```

In [32]:
```python
y
```

Out[32]:
```
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', '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', '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', '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)
```

In [33]:
```python
# splitting value for training model
from sklearn.model_selection import train_test_split
```

In [34]:
```python
x_train , x_test, y_train, y_test = train_test_split(x,y,test_size=0.2, random_state=0)
```

In [35]:
```python
print(x_train)
```

```
[['Male' 'Yes' '0' ... 1.0 4.875197323201151 5858.0]
 ['Male' 'No' '1' ... 1.0 5.278114659230517 11250.0]
 ['Male' 'Yes' '0' ... 0.0 5.003946305945459 5681.0]
 ...
 ['Male' 'Yes' '3+' ... 1.0 5.298317366548036 8334.0]
 ['Male' 'Yes' '0' ... 1.0 5.075173815233827 6033.0]
 ['Female' 'Yes' '0' ... 1.0 5.204006687076795 6486.0]]
```

In [36]:
```
from sklearn.preprocessing import LabelEncoder
label_x = LabelEncoder()
label_x
```

Out[36]:
```
▾ LabelEncoder
LabelEncoder()
```

In [37]:
```
# convert categorical to numerival between 0 to 5 range
for p in range(0,5):
    x_train[:,p] = label_x.fit_transform(x_train[:,p])
```

In [38]:
```
x_train[:,7] = label_x.fit_transform(x_train[:,7])
```

In [39]:
```
x_train
```

Out[39]:
```
array([[1, 1, 0, ..., 1.0, 4.875197323201151, 267],
       [1, 0, 1, ..., 1.0, 5.278114659230517, 407],
       [1, 1, 0, ..., 0.0, 5.003946305945459, 249],
       ...,
       [1, 1, 3, ..., 1.0, 5.298317366548036, 363],
       [1, 1, 0, ..., 1.0, 5.075173815233827, 273],
       [0, 1, 0, ..., 1.0, 5.204006687076795, 301]], dtype=object)
```

In [40]:
```
label_y = LabelEncoder()
y_train = label_y.fit_transform(y_train)
```

In [41]:
```
y_train
```

Out[41]:
```
array([1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1,
       1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
       0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1,
       1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1,
       1, 1, 1, 0, 1, 0, 1])
```

In [42]:
```
x_test
```

Out[42]:
```
array([['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.430816798843313,
        7085.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.718498871295094,
        4230.0],
       ['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.780743515792329,
        10039.0],
       ['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.700480365792417,
        6784.0],
       ['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.574710978503383,
        3875.0],
       ['Male', 'Yes', '0', 'Not Graduate', 180.0, 0.0, 5.10594547390058,
        6058.0],
       ['Male', 'Yes', '3+', 'Graduate', 180.0, 1.0, 5.056245805348308,
        6417.0],
       ['Male', 'No', '0', 'Graduate', 360.0, 1.0, 6.003887067106539,
        12876.0],
       ['Male', 'No', '0', 'Graduate', 360.0, 0.0, 4.820281565605037,
        5124.0],
       ['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.852030263919617,
        5233.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.430816798843313,
        2917.0],
       ['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.553876891600541,
        2895.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 5.634789603169249,
        8333.0],
       ['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.4638318050256105,
        8667.0],
       ['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.564348191467836,
        14880.0],
       ['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.204692619390966,
        3875.0],
```

```
['Male', 'No', '1', 'Not Graduate', 360.0, 1.0, 5.247024072160486,
 4311.0],
['Male', 'No', '0', 'Not Graduate', 360.0, 1.0, 4.882801922586371,
 3946.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.532599493153256,
 2500.0],
['Male', 'Yes', '0', 'Not Graduate', 360.0, 0.0,
 5.198497031265826, 4787.0],
['Female', 'Yes', '0', 'Graduate', 360.0, 0.0, 4.787491742782046,
 6085.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.962844630259907,
 4765.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.68213122712422,
 7550.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.10594547390058,
 11500.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.060443010546419,
 4521.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 5.521460917862246,
 8069.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 5.231108616854587,
 8724.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.231108616854587,
 11333.0],
['Male', 'Yes', '3+', 'Graduate', 360.0, 0.0, 4.852030263919617,
 4680.0],
['Female', 'No', '0', 'Graduate', 360.0, 0.0, 4.634728988229636,
 5000.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.429345628954441,
 9083.0],
['Male', 'No', '0', 'Not Graduate', 360.0, 1.0, 3.871201010907891,
 4885.0],
['Male', 'Yes', '1', 'Not Graduate', 360.0, 1.0,
 4.499809670330265, 5100.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.19295685089021,
 9734.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.857444178729352,
 8235.0],
['Female', 'Yes', '0', 'Not Graduate', 360.0, 0.0,
 5.181783550292085, 5386.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.147494476813453,
 5717.0],
['Male', 'No', '0', 'Not Graduate', 360.0, 1.0, 4.836281906951478,
 4592.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.852030263919617,
 6250.0],
['Male', 'Yes', '2', 'Not Graduate', 360.0, 1.0, 4.68213122712422,
 3917.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.382026634673881,
 3244.0],
['Male', 'Yes', '3+', 'Graduate', 360.0, 0.0, 4.812184355372417,
 5900.0],
['Male', 'Yes', '2', 'Graduate', 120.0, 1.0, 2.833213344056216,
 2385.0],
['Male', 'Yes', '1', 'Not Graduate', 360.0, 1.0,
 5.062595033026967, 5783.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.330733340286331,
 3858.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 5.231108616854587,
 12083.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.7535901911063645,
 3750.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.74493212836325,
 4547.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.852030263919617,
 6091.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.941642422609304,
 6500.0],
['Male', 'Yes', '3+', 'Not Graduate', 360.0, 1.0,
 4.30406509320417, 3173.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.867534450455582,
 7083.0],
['Male', 'Yes', '0', 'Not Graduate', 360.0, 1.0,
 4.672828834461906, 4300.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.857444178729352,
 5505.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.718498871295094,
 3798.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.556828061699537,
 10916.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.553876891600541,
 4492.0],
['Male', 'No', '0', 'Not Graduate', 360.0, 1.0, 4.890349128221754,
 6216.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.123963979403259,
 5532.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.787491742782046,
 4191.0],
['Female', 'No', '0', 'Graduate', 360.0, 0.0, 4.919980925828125,
```

```
 11117.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 5.365976015021851,
 10000.0],
['Male', 'Yes', '0', 'Not Graduate', 360.0, 1.0, 4.74493212836325,
 4567.0],
['Female', 'No', '0', 'Graduate', 360.0, 0.0, 4.330733340286331,
 3510.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.890349128221754,
 5935.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 5.752572638825633,
 11580.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.075173815233827,
 6166.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.912654885736052,
 4897.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.204006687076795,
 6873.0],
['Male', 'No', '0', 'Not Graduate', 360.0, 1.0, 4.564348191467836,
 5484.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.204692619390966,
 6979.0],
['Female', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.867534450455582,
 2928.0],
['Male', 'Yes', '2', 'Not Graduate', 360.0, 1.0,
 5.056245805348308, 5398.0],
['Male', 'Yes', '1', 'Not Graduate', 180.0, 1.0,
 4.919980925828125, 6608.0],
['Female', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.969813299576001,
 5126.0],
['Male', 'Yes', '0', 'Not Graduate', 300.0, 1.0,
 4.820281565605037, 5297.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.499809670330265,
 35673.0],
['Male', 'No', '3+', 'Graduate', 360.0, 1.0, 5.768320995793772,
 15500.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.718498871295094,
 9703.0],
['Female', 'No', '0', 'Graduate', 360.0, 0.0, 4.7535901911063645,
 4166.0],
['Female', 'No', '0', 'Graduate', 480.0, 1.0, 4.727387818712341,
 4328.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 6.214608098422191,
 18333.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 5.267858159063328,
 7441.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.231108616854587,
 7583.0],
['Male', 'No', '0', 'Graduate', 480.0, 1.0, 4.2626798770413155,
 3069.0],
['Male', 'Yes', '0', 'Graduate', 12.0, 1.0, 4.709530201312334,
 7482.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.700480365792417,
 4801.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.298317366548036,
 11179.0],
['Male', 'No', '1', 'Graduate', 180.0, 1.0, 4.727387818712341,
 3667.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.6443908991413725,
 4545.0],
['Female', 'Yes', '0', 'Not Graduate', 360.0, 1.0,
 4.605170185988092, 3572.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.30406509320417,
 3167.0],
['Male', 'Yes', '1', 'Graduate', 84.0, 1.0, 5.147494476813453,
 7283.0],
['Male', 'Yes', '3+', 'Graduate', 300.0, 0.0, 5.19295685089021,
 7167.0],
['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.2626798770413155,
 2900.0],
['Male', 'No', '0', 'Not Graduate', 180.0, 0.0, 4.836281906951478,
 5454.0],
['Male', 'No', '0', 'Graduate', 180.0, 1.0, 5.1647859739235145,
 6950.0],
['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.969813299576001,
 5762.0],
['Male', 'Yes', '2', 'Not Graduate', 360.0, 1.0,
 4.394449154672439, 5093.0],
['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 5.231108616854587,
 9538.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 5.351858133476067,
 7977.0],
['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 4.605170185988092,
 3539.0],
['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.787491742782046,
 10819.0],
['Male', 'No', '0', 'Graduate', 180.0, 1.0, 4.787491742782046,
 10383.0],
['Male', 'Yes', '3+', 'Graduate', 360.0, 1.0, 4.852030263919617,
 5703.0],
```

```
       ['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.8283137373023015,
        4950.0],
       ['Male', 'No', '0', 'Not Graduate', 360.0, 1.0,
        4.6443908991413725, 4750.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.477336814478207,
        3410.0],
       ['Male', 'Yes', '0', 'Not Graduate', 360.0, 1.0,
        4.553876891600541, 3849.0],
       ['Male', 'Yes', '3+', 'Not Graduate', 180.0, 1.0,
        4.394449154672439, 3522.0],
       ['Male', 'No', '0', 'Graduate', 360.0, 1.0, 5.298317366548036,
        6400.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.90527477843843,
        3418.0],
       ['Male', 'No', '0', 'Graduate', 480.0, 1.0, 4.727387818712341,
        3750.0],
       ['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 4.248495242049359,
        3900.0],
       ['Male', 'Yes', '0', 'Not Graduate', 360.0, 0.0,
        5.303304908059076, 5558.0],
       ['Male', 'Yes', '0', 'Graduate', 180.0, 0.0, 4.499809670330265,
        4949.0],
       ['Female', 'No', '0', 'Graduate', 360.0, 1.0, 4.430816798843313,
        4292.0],
       ['Male', 'No', '0', 'Graduate', 360.0, 1.0, 4.897839799950911,
        4269.0],
       ['Male', 'Yes', '2', 'Graduate', 360.0, 1.0, 5.170483995038151,
        7100.0],
       ['Male', 'Yes', '3+', 'Graduate', 360.0, 1.0, 4.867534450455582,
        13746.0],
       ['Male', 'Yes', '0', 'Graduate', 360.0, 1.0, 6.077642243349034,
        14583.0],
       ['Male', 'Yes', '3+', 'Not Graduate', 180.0, 0.0,
        4.248495242049359, 4611.0],
       ['Male', 'Yes', '1', 'Graduate', 360.0, 1.0, 4.564348191467836,
        3428.0]], dtype=object)
```

In [43]:
```python
for z in range(0,5):
    x_test[:,z] = label_x.fit_transform(x_test[:,z])
```

In [44]:
```python
x_test[:,7] = label_x.fit_transform(x_test[:,7])
```

In [45]:
```python
x_test
```

Out[45]:
```
array([[1, 0, 0, 0, 5, 1.0, 4.430816798843313, 85],
       [0, 0, 0, 0, 5, 1.0, 4.718498871295094, 28],
       [1, 1, 0, 0, 5, 1.0, 5.780743515792329, 104],
       [1, 1, 0, 0, 5, 1.0, 4.700480365792417, 80],
       [1, 1, 2, 0, 5, 1.0, 4.574710978503383, 22],
       [1, 1, 0, 1, 3, 0.0, 5.10594547390058, 70],
       [1, 1, 3, 0, 3, 1.0, 5.056245805348308, 77],
       [1, 0, 0, 0, 5, 1.0, 6.003887067106539, 114],
       [1, 0, 0, 0, 5, 0.0, 4.820281565605037, 53],
       [1, 1, 0, 0, 5, 1.0, 4.852030263919617, 55],
       [0, 0, 0, 0, 5, 1.0, 4.430816798843313, 4],
       [1, 1, 1, 0, 5, 1.0, 4.553876891600541, 2],
       [0, 0, 0, 0, 5, 1.0, 5.634789603169249, 96],
       [1, 1, 2, 0, 5, 1.0, 5.4638318050256105, 97],
       [1, 1, 0, 0, 5, 1.0, 4.564348191467836, 117],
       [1, 1, 1, 0, 5, 1.0, 4.204692619390966, 22],
       [1, 0, 1, 1, 5, 1.0, 5.247024072160486, 32],
       [1, 0, 0, 1, 5, 1.0, 4.882801922586371, 25],
       [0, 0, 0, 0, 5, 1.0, 4.532599493153256, 1],
       [1, 1, 0, 1, 5, 0.0, 5.198497031265826, 44],
       [0, 1, 0, 0, 5, 0.0, 4.787491742782046, 71],
       [1, 1, 0, 0, 5, 1.0, 4.962844630259907, 43],
       [1, 1, 2, 0, 5, 1.0, 4.68213122712422, 91],
       [1, 1, 2, 0, 5, 1.0, 5.10594547390058, 111],
       [1, 1, 0, 0, 5, 1.0, 4.060443010546419, 35],
       [1, 1, 1, 0, 5, 1.0, 5.521460917862246, 94],
       [1, 0, 0, 0, 5, 1.0, 5.231108616854587, 98],
       [1, 1, 0, 0, 5, 1.0, 5.231108616854587, 110],
       [1, 1, 3, 0, 5, 0.0, 4.852030263919617, 41],
       [0, 0, 0, 0, 5, 0.0, 4.634728988229636, 50],
       [1, 1, 0, 0, 5, 1.0, 5.429345628954441, 99],
       [1, 0, 0, 1, 5, 1.0, 3.871201010907891, 46],
       [1, 1, 1, 1, 5, 1.0, 4.499809670330265, 52],
       [1, 1, 0, 0, 5, 1.0, 5.19295685089021, 102],
       [1, 1, 0, 0, 5, 1.0, 4.857444178729352, 95],
       [0, 1, 0, 1, 5, 0.0, 5.181783550292085, 57],
       [1, 1, 0, 0, 5, 1.0, 5.147494476813453, 65],
       [1, 0, 0, 1, 5, 1.0, 4.836281906951478, 39],
       [1, 1, 0, 0, 5, 1.0, 4.852030263919617, 75],
       [1, 1, 2, 1, 5, 1.0, 4.68213122712422, 24],
       [0, 0, 0, 0, 5, 1.0, 4.382026634673881, 9],
       [1, 1, 3, 0, 5, 0.0, 4.812184355372417, 68],
       [1, 1, 2, 0, 2, 1.0, 2.833213344056216, 0],
```

```
       [1, 1, 1, 1, 5, 1.0, 5.062595033026967, 67],
       [1, 0, 0, 0, 5, 1.0, 4.330733340286331, 21],
       [1, 0, 0, 0, 5, 1.0, 5.231108616854587, 113],
       [1, 1, 1, 0, 5, 1.0, 4.7535901911063645, 18],
       [0, 0, 0, 0, 5, 1.0, 4.74493212836325, 37],
       [1, 1, 1, 0, 5, 1.0, 4.852030263919617, 72],
       [1, 0, 0, 0, 5, 1.0, 4.941642422609304, 78],
       [1, 1, 3, 1, 5, 1.0, 4.30406509320417, 8],
       [1, 1, 0, 0, 5, 1.0, 4.867534450455582, 84],
       [1, 1, 0, 1, 5, 1.0, 4.672828834461906, 31],
       [1, 0, 0, 0, 5, 1.0, 4.857444178729352, 61],
       [1, 1, 0, 0, 5, 1.0, 4.718498871295094, 19],
       [1, 1, 0, 0, 5, 1.0, 5.556828061699537, 107],
       [1, 1, 0, 0, 5, 1.0, 4.553876891600541, 34],
       [1, 0, 0, 1, 5, 1.0, 4.890349128221754, 74],
       [1, 1, 2, 0, 5, 1.0, 5.123963979403259, 62],
       [1, 0, 0, 0, 5, 1.0, 4.787491742782046, 27],
       [0, 0, 0, 0, 5, 0.0, 4.919980925828125, 108],
       [0, 0, 0, 0, 5, 1.0, 5.365976015021851, 103],
       [1, 1, 0, 1, 5, 1.0, 4.74493212836325, 38],
       [0, 0, 0, 0, 5, 0.0, 4.330733340286331, 13],
       [1, 1, 2, 0, 5, 1.0, 4.890349128221754, 69],
       [1, 1, 1, 0, 5, 1.0, 5.752572638825633, 112],
       [1, 1, 0, 0, 5, 1.0, 5.075173815233827, 73],
       [1, 0, 0, 0, 5, 1.0, 4.912654885736052, 47],
       [1, 1, 0, 0, 5, 1.0, 5.204006687076795, 81],
       [1, 0, 0, 1, 5, 1.0, 4.564348191467836, 60],
       [1, 0, 0, 0, 5, 1.0, 4.204692619390966, 83],
       [0, 1, 0, 0, 5, 1.0, 4.867534450455582, 5],
       [1, 1, 2, 1, 5, 1.0, 5.056245805348308, 58],
       [1, 1, 1, 1, 3, 1.0, 4.919980925828125, 79],
       [0, 1, 0, 0, 5, 1.0, 4.969813299576001, 54],
       [1, 1, 0, 1, 4, 1.0, 4.820281565605037, 56],
       [1, 0, 0, 0, 5, 1.0, 4.499809670330265, 120],
       [1, 0, 3, 0, 5, 1.0, 5.768320995793772, 118],
       [1, 1, 2, 0, 5, 1.0, 4.718498871295094, 101],
       [0, 0, 0, 0, 5, 0.0, 4.7535901911063645, 26],
       [0, 0, 0, 0, 6, 1.0, 4.727387818712341, 33],
       [1, 1, 1, 0, 5, 1.0, 6.214608098422191, 119],
       [0, 0, 0, 0, 5, 1.0, 5.267858159063328, 89],
       [1, 1, 2, 0, 5, 1.0, 5.231108616854587, 92],
       [1, 0, 0, 0, 6, 1.0, 4.2626798770413155, 6],
       [1, 1, 0, 0, 0, 1.0, 4.709530201312334, 90],
       [1, 1, 0, 0, 5, 1.0, 4.700480365792417, 45],
       [1, 1, 2, 0, 5, 1.0, 5.298317366548036, 109],
       [1, 0, 1, 0, 3, 1.0, 4.727387818712341, 17],
       [1, 1, 1, 0, 5, 1.0, 4.6443908991413725, 36],
       [0, 1, 0, 1, 5, 1.0, 4.605170185988092, 16],
       [1, 0, 0, 0, 5, 1.0, 4.30406509320417, 7],
       [1, 1, 1, 0, 1, 1.0, 5.147494476813453, 88],
       [1, 1, 3, 0, 4, 0.0, 5.19295685089021, 87],
       [0, 0, 0, 0, 5, 1.0, 4.2626798770413155, 3],
       [1, 0, 0, 1, 3, 0.0, 4.836281906951478, 59],
       [1, 0, 0, 0, 3, 1.0, 5.1647859739235145, 82],
       [1, 0, 0, 0, 5, 1.0, 4.969813299576001, 66],
       [1, 1, 2, 1, 5, 1.0, 4.394449154672439, 51],
       [1, 1, 1, 0, 5, 1.0, 5.231108616854587, 100],
       [1, 1, 0, 0, 5, 1.0, 5.351858133476067, 93],
       [1, 1, 0, 0, 5, 1.0, 4.605170185988092, 15],
       [1, 1, 2, 0, 5, 1.0, 4.787491742782046, 106],
       [1, 0, 0, 0, 3, 1.0, 4.787491742782046, 105],
       [1, 1, 3, 0, 5, 1.0, 4.852030263919617, 64],
       [1, 0, 0, 0, 5, 1.0, 4.8283137373023015, 49],
       [1, 0, 0, 1, 5, 1.0, 4.6443908991413725, 42],
       [0, 0, 0, 0, 5, 1.0, 4.477336814478207, 10],
       [1, 1, 0, 1, 5, 1.0, 4.553876891600541, 20],
       [1, 1, 3, 1, 3, 1.0, 4.394449154672439, 14],
       [1, 0, 0, 0, 5, 1.0, 5.298317366548036, 76],
       [0, 0, 0, 0, 5, 1.0, 4.90527477843843, 11],
       [1, 0, 0, 0, 6, 1.0, 4.727387818712341, 18],
       [1, 1, 2, 0, 5, 1.0, 4.248495242049359, 23],
       [1, 1, 0, 1, 5, 0.0, 5.303304908059076, 63],
       [1, 1, 0, 0, 3, 0.0, 4.499809670330265, 48],
       [0, 0, 0, 0, 5, 1.0, 4.430816798843313, 30],
       [1, 0, 0, 0, 5, 1.0, 4.897839799950911, 29],
       [1, 1, 2, 0, 5, 1.0, 5.170483995038151, 86],
       [1, 1, 3, 0, 5, 1.0, 4.867534450455582, 115],
       [1, 1, 0, 0, 5, 1.0, 6.077642243349034, 116],
       [1, 1, 3, 1, 3, 0.0, 4.248495242049359, 40],
       [1, 1, 1, 0, 5, 1.0, 4.564348191467836, 12]], dtype=object)
```

In [46]:
```
label_y = LabelEncoder()
y_test = label_y.fit_transform(y_test)
```

In [47]:
```
y_test
```

Out[47]:
```
array([1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 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, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1,
       1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1])
```

In [48]:
```
#standardization of model
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)
```

DecisionTreeClassifier

In [59]:
```
from sklearn.tree import DecisionTreeClassifier
DTClassifier = DecisionTreeClassifier(criterion='entropy', random_state=0)
DTClassifier.fit(x_train,y_train)
```

Out[59]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [60]:
```
y_pred = DTClassifier.predict(x_test)
y_pred
```

Out[60]:
```
array([0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1,
       1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1])
```

In [61]:
```
from sklearn.metrics import accuracy_score
accuracy_score = accuracy_score(y_pred,y_test)
print('The accuracy score of DecisionTree is : ',accuracy_score)
```

```
The accuracy score of DecisionTree is :  0.7073170731707317
```

GaussianNB

In [62]:
```
from sklearn.naive_bayes import GaussianNB
nbclassifier = GaussianNB()
nbclassifier.fit(x_train, y_train)
```

Out[62]:
```
▾ GaussianNB
GaussianNB()
```

In [63]:
```
y_pred = nbclassifier.predict(x_test)
y_pred
```

Out[63]:
```
array([1, 1, 1, 1, 1, 0, 1, 1, 0, 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, 1, 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])
```

In [64]:
```
from sklearn.metrics import accuracy_score
accuracy_score = accuracy_score(y_pred,y_test)
print('The accuracy score of naive bases is: ', accuracy_score)
```

```
The accuracy score of naive bases is:  0.8292682926829268
```

In [ ]: