225229110

HARI PRASATH S

LOAN APPROVSL CLASSIFICATION USING SVM

Step1:Importing data

In [201]:
```
import pandas as pd
```

In [202]:
```
df=pd.read_csv('train_loan.csv')
df
```

8	LP001018	Male	Yes	2	Graduate	No	4006
9	LP001020	Male	Yes	1	Graduate	No	12841
10	LP001024	Male	Yes	2	Graduate	No	3200
11	LP001027	Male	Yes	2	Graduate	NaN	2500
12	LP001028	Male	Yes	2	Graduate	No	3073
13	LP001029	Male	No	0	Graduate	No	1853
14	LP001030	Male	Yes	2	Graduate	No	1299
15	LP001032	Male	No	0	Graduate	No	4950
16	LP001034	Male	No	1	Not Graduate	No	3596
17	LP001036	Female	No	0	Graduate	No	3510
18	LP001038	Male	Yes	0	Not Graduate	No	4887
19	LP001041	Male	Yes	0	Graduate	NaN	2600

In [203]:
```
df.head()
```

Out[203]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
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	

In [204]:
```
df.shape
```

Out[204]: (614, 13)

In [205]:
```
df.columns
```

Out[205]:
```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
      dtype='object')
```

In [206]:
```
df.dtypes
```

Out[206]:
```
Loan_ID               object
Gender                object
Married               object
Dependents            object
Education             object
Self_Employed         object
ApplicantIncome        int64
CoapplicantIncome    float64
LoanAmount           float64
Loan_Amount_Term     float64
Credit_History       float64
Property_Area         object
Loan_Status           object
dtype: object
```

In [207]:
```
df.info
```

```
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
..        ...     ...    ...        ...          ...          ...
584  LP002911    Male    Yes          1     Graduate           No
```

In [208]:
```
df.Self_Employed.value_counts
```

Out[208]:
```
<bound method IndexOpsMixin.value_counts of 0       No
1       No
2      Yes
3       No
4       No
5      Yes
6       No
7       No
8       No
9       No
10      No
11     NaN
12      No
13      No
14      No
15      No
16      No
17      No
18      No
19     NaN
20      No
21      No
22      No
23      No
24     NaN
25     Yes
26      No
27      No
28      No
29     NaN
      ...
584     No
585     No
586     No
587     No
588     No
589    Yes
590     No
591    Yes
592    Yes
593     No
594    Yes
595     No
596    Yes
597     No
598    Yes
599     No
600    NaN
601    NaN
602     No
603     No
604     No
605     No
606     No
607     No
```

```
608      No
609      No
610      No
611      No
612      No
613     Yes
Name: Self_Employed, Length: 614, dtype: object>
```

Step2:Data Cleaning

In [209]:
```
df['Dependents'].dtype
```

Out[209]:
```
dtype('O')
```

In [210]:
```
df['Dependents'].fillna("No_Dep",inplace = True)
df['Dependents']
```

Out[210]:
```
0           0
1           1
2           0
3           0
4           0
5           2
6           0
7          3+
8           2
9           1
10          2
11          2
12          2
13          0
14          2
15          0
16          1
17          0
18          0
19          0
20          0
21          1
22          0
23          2
24          1
25          0
26          0
27          2
28          0
29          2
        ...
584         1
585         1
586         0
587         0
588         0
589         2
590         0
591         2
592        3+
593         0
594         0
595         0
596         2
597    No_Dep
598         0
599         2
600        3+
601         0
602        3+
603         0
604         1
605         0
606         1
```

```
607        2
608        0
609        0
610       3+
611        1
612        2
613        0
Name: Dependents, Length: 614, dtype: object
```

In [211]:
```
df.Dependents[df.Dependents == '3+'] = 3
df.Dependents[df.Dependents == '1'] = 1
df.Dependents[df.Dependents == '2'] = 2
df.Dependents[df.Dependents == 'No_Dep'] =0

print(df)
```

```
     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
```

In [212]:
```
df.isnull().sum()
```

Out[212]:
```
Loan_ID               0
Gender               13
Married               3
Dependents            0
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 [213]:
```
df['Gender'].fillna(df['Gender'].mode()[0],inplace=True)
```

In [214]:
```
df[ 'Married'].fillna(df[ 'Married']. mode()[0],inplace=True)
df[ 'Dependents'].fillna(df[ 'Dependents'].mode()[0],inplace=True)
df['Education'].fillna(df[ 'Education'].mode()[0],inplace=True)
df['Self_Employed'].fillna(df['Self_Employed'].mode()[0],inplace=True)
df['Credit_History'].fillna(df['Credit_History'].mode()[0],inplace=True)
```

In [215]:
```
df.isnull().sum()
```

Out[215]:
```
Loan_ID               0
Gender                0
Married               0
Dependents            0
Education             0
Self_Employed         0
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History        0
Property_Area         0
Loan_Status           0
dtype: int64
```

In [216]:
```
df['Gender']
```

Out[216]:
```
0        Male
1        Male
2        Male
3        Male
4        Male
5        Male
6        Male
7        Male
8        Male
9        Male
10       Male
11       Male
12       Male
13       Male
14       Male
15       Male
16       Male
17     Female
18       Male
19       Male
20       Male
21       Male
22       Male
23       Male
24       Male
25       Male
26       Male
27       Male
28       Male
29     Female
        ...
584      Male
585      Male
586      Male
587    Female
588      Male
589      Male
590      Male
591      Male
592      Male
593      Male
594      Male
595      Male
596      Male
597      Male
598      Male
599      Male
600    Female
601      Male
602      Male
603      Male
604    Female
605      Male
606      Male
607      Male
```

```
608      Male
609    Female
610      Male
611      Male
612      Male
613    Female
Name: Gender, Length: 614, dtype: object
```

In [217]:
```
df['LoanAmount'].fillna(df['LoanAmount'].mean(), inplace=True)
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean(), inplace=True)
```

In [218]:
```
df.drop(['Loan_ID'], axis=1)
```

[218]:

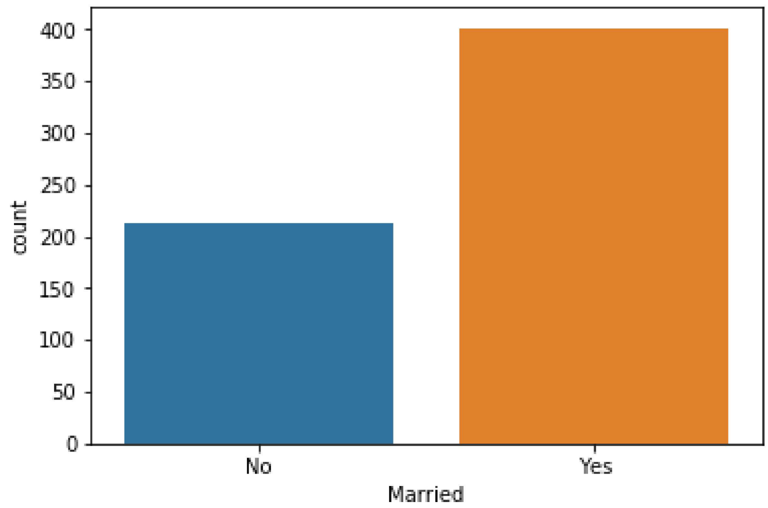
	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Loa
0	Male	No	0	Graduate	No	5849	0.0	14
1	Male	Yes	1	Graduate	No	4583	1508.0	12
2	Male	Yes	0	Graduate	Yes	3000	0.0	6
3	Male	Yes	0	Not Graduate	No	2583	2358.0	12
4	Male	No	0	Graduate	No	6000	0.0	14
5	Male	Yes	2	Graduate	Yes	5417	4196.0	26
6	Male	Yes	0	Not Graduate	No	2333	1516.0	9
7	Male	Yes	3	Graduate	No	3036	2504.0	15
8	Male	Yes	2	Graduate	No	4006	1526.0	16
9	Male	Yes	1	Graduate	No	12841	10968.0	34

Step3:Exploratory Data Analysis-Who got their loan approved

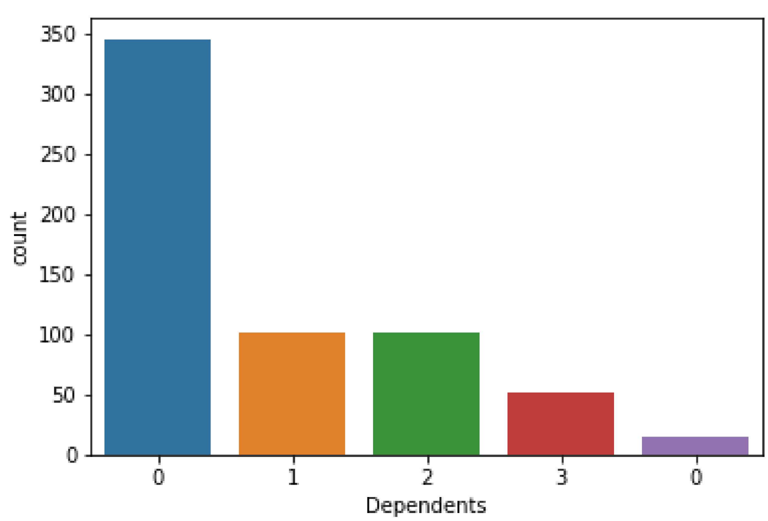
In [219]:
```
import matplotlib.pyplot as plt
```

In [220]:
```
import seaborn as sns
```

In [221]:
```
sns.countplot (x = 'Married', data =df)
```
Out[221]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1f9e2632978>
```

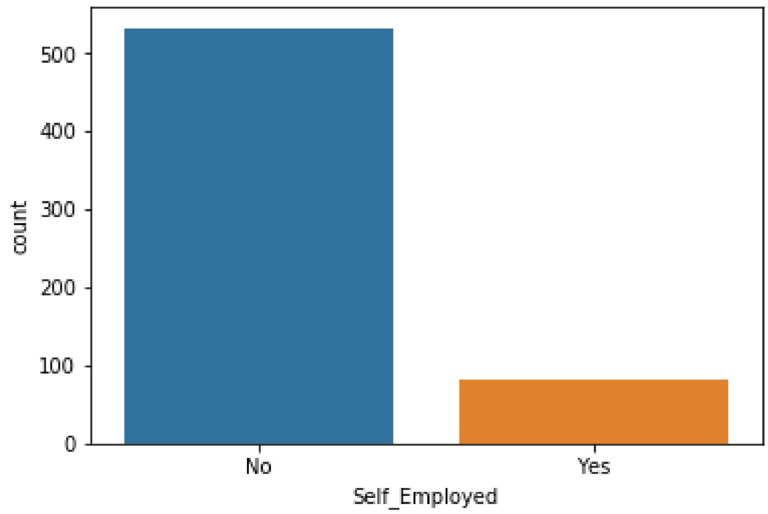


In [222]:
```
sns.countplot (x = 'Dependents', data =df)
```
Out[222]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1f9e27e6b70>
```



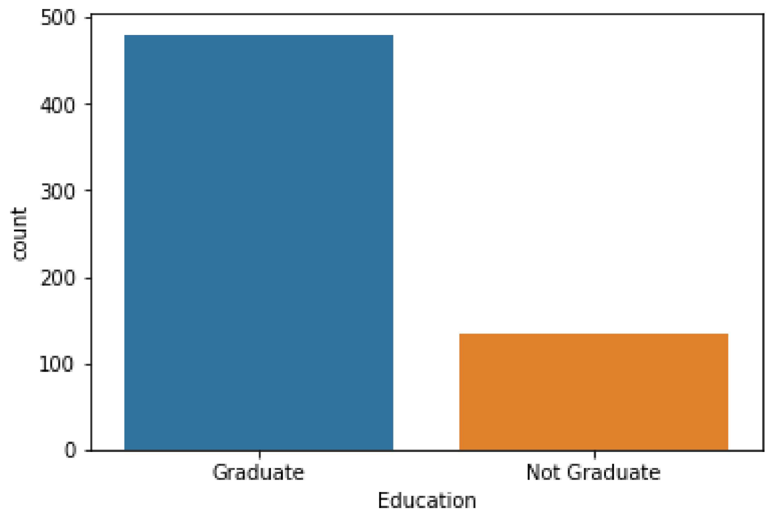
In [223]:
```
sns.countplot (x = 'Self_Employed', data =df)
```

Out[223]: <matplotlib.axes._subplots.AxesSubplot at 0x1f9e283f588>



In [224]:
```
sns.countplot (x = 'Education', data =df)
```

Out[224]: <matplotlib.axes._subplots.AxesSubplot at 0x1f9e285d898>



Step4:Extract X and y

In [225]:
```
X = df.drop(['Loan_Status'], axis =1)
```

In [226]:
```
X
```

503	LP002958	Male	No	0	Graduate	No	3676	4301.0	17
504	LP002959	Female	Yes	1	Graduate	No	12000	0.0	49
505	LP002960	Male	Yes	0	Not Graduate	No	2400	3800.0	14
506	LP002961	Male	Yes	1	Graduate	No	3400	2500.0	17
507	LP002964	Male	Yes	2	Not Graduate	No	3987	1411.0	15
508	LP002974	Male	Yes	0	Graduate	No	3232	1950.0	10
509	LP002978	Female	No	0	Graduate	No	2900	0.0	7
510	LP002979	Male	Yes	3	Graduate	No	4106	0.0	4
511	LP002983	Male	Yes	1	Graduate	No	8072	240.0	25
512	LP002984	Male	Yes	2	Graduate	No	7583	0.0	18
513	LP002990	Female	No	0	Graduate	Yes	4583	0.0	13

In [227]:
```
Y = df.pop('Loan_Status')
```

In [228]:
```
Y
```

Out[228]:
```
0      Y
1      N
2      Y
3      Y
4      Y
5      Y
6      Y
7      N
8      Y
9      N
10     Y
11     Y
12     Y
13     N
14     Y
15     Y
16     Y
17     N
18     N
19     Y
20     N
21     Y
22     N
23     N
24     N
25     Y
26     Y
27     Y
28     N
29     Y
      ..
584    N
585    N
586    Y
587    Y
588    Y
589    N
590    Y
591    N
592    Y
593    Y
594    Y
595    Y
596    N
597    N
598    Y
599    Y
600    N
601    Y
602    Y
603    Y
604    Y
605    N
606    Y
607    Y
```

```
608    Y
609    Y
610    Y
611    Y
612    Y
613    N
Name: Loan_Status, Length: 614, dtype: object
```

Step5:One Hot Encoding

In [229]:
```
X=pd.get_dummies(X)
```

In [230]:
```
X
```

Out[230]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Loa
0	5849	0.0	146.412162	360.0	1.0	
1	4583	1508.0	128.000000	360.0	1.0	
2	3000	0.0	66.000000	360.0	1.0	
3	2583	2358.0	120.000000	360.0	1.0	
4	6000	0.0	141.000000	360.0	1.0	
5	5417	4196.0	267.000000	360.0	1.0	
6	2333	1516.0	95.000000	360.0	1.0	
7	3036	2504.0	158.000000	360.0	0.0	
8	4006	1526.0	168.000000	360.0	1.0	
9	12841	10968.0	349.000000	360.0	1.0	
10	3200	700.0	70.000000	360.0	1.0	
11	2500	1840.0	109.000000	360.0	1.0	
12	3073	8106.0	200.000000	360.0	1.0	
13	1853	2840.0	114.000000	360.0	1.0	
14	1299	1086.0	17.000000	120.0	1.0	
15	4950	0.0	125.000000	360.0	1.0	
16	3596	0.0	100.000000	240.0	1.0	
17	3510	0.0	76.000000	360.0	0.0	
18	4887	0.0	133.000000	360.0	1.0	
19	2600	3500.0	115.000000	342.0	1.0	
20	7660	0.0	104.000000	360.0	0.0	
21	5955	5625.0	315.000000	360.0	1.0	
22	2600	1911.0	116.000000	360.0	0.0	
23	3365	1917.0	112.000000	360.0	0.0	
24	3717	2925.0	151.000000	360.0	1.0	
25	9560	0.0	191.000000	360.0	1.0	
26	2799	2253.0	122.000000	360.0	1.0	
27	4226	1040.0	110.000000	360.0	1.0	
28	1442	0.0	35.000000	360.0	1.0	
29	3750	2083.0	120.000000	360.0	1.0	
...	...	...	...	...	...	
584	2787	1917.0	146.000000	360.0	0.0	
585	4283	3000.0	172.000000	84.0	1.0	
586	2297	1522.0	104.000000	360.0	1.0	

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Loa
587	2165	0.0	70.000000	360.0	1.0	
588	4750	0.0	94.000000	360.0	1.0	
589	2726	0.0	106.000000	360.0	0.0	
590	3000	3416.0	56.000000	180.0	1.0	
591	6000	0.0	205.000000	240.0	1.0	
592	9357	0.0	292.000000	360.0	1.0	
593	3859	3300.0	142.000000	180.0	1.0	
594	16120	0.0	260.000000	360.0	1.0	
595	3833	0.0	110.000000	360.0	1.0	
596	6383	1000.0	187.000000	360.0	1.0	
597	2987	0.0	88.000000	360.0	0.0	
598	9963	0.0	180.000000	360.0	1.0	
599	5780	0.0	192.000000	360.0	1.0	
600	416	41667.0	350.000000	180.0	1.0	
601	2894	2792.0	155.000000	360.0	1.0	
602	5703	0.0	128.000000	360.0	1.0	
603	3676	4301.0	172.000000	360.0	1.0	
604	12000	0.0	496.000000	360.0	1.0	
605	2400	3800.0	146.412162	180.0	1.0	
606	3400	2500.0	173.000000	360.0	1.0	
607	3987	1411.0	157.000000	360.0	1.0	
608	3232	1950.0	108.000000	360.0	1.0	
609	2900	0.0	71.000000	360.0	1.0	
610	4106	0.0	40.000000	180.0	1.0	
611	8072	240.0	253.000000	360.0	1.0	
612	7583	0.0	187.000000	360.0	1.0	
613	4583	0.0	133.000000	360.0	0.0	

614 rows × 635 columns

Step6:Model Building

In [231]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
ss= StandardScaler()
```

In [232]:
```
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=.25,random_state=42)
```

In [233]:
```
X_train_ss=ss.fit_transform(X_train)
X_train_ss
```

Out[233]:
```
array([[-0.3670333 ,  0.08668355, -0.78687919, ..., -0.63101212,
        -0.79084872,  1.42348719],
       [-0.24621383,  0.34471599, -0.08548056, ...,  1.58475562,
        -0.79084872, -0.70250017],
       [ 0.26897235, -0.60393266, -0.26380224, ..., -0.63101212,
        -0.79084872,  1.42348719],
       ...,
       [-0.37301611, -0.60393266, -1.39317292, ..., -0.63101212,
        -0.79084872,  1.42348719],
       [ 0.7587316 , -0.60393266, -0.00925205, ..., -0.63101212,
         1.26446434, -0.70250017],
       [ 1.35751089, -0.60393266, -0.00925205, ..., -0.63101212,
        -0.79084872,  1.42348719]])
```

In [234]:
```
X_test_ss=ss.fit_transform(X_test)
X_test_ss
```

Out[234]:
```
array([[ 0.61611689, -0.46834278,  1.00517798, ..., -0.67292658,
         1.32287566, -0.69337525],
       [-0.13546144, -0.46834278, -0.16875232, ..., -0.67292658,
         1.32287566, -0.69337525],
       [-0.15797887, -0.07201228,  0.16665633, ...,  1.48604621,
        -0.75592895, -0.69337525],
       ...,
       [-0.3156009 , -0.46834278, -1.29476711, ..., -0.67292658,
         1.32287566, -0.69337525],
       [-0.12128963, -0.46834278, -0.20468897, ..., -0.67292658,
         1.32287566, -0.69337525],
       [-0.22364159, -0.46834278, -0.37239329, ..., -0.67292658,
        -0.75592895,  1.44222051]])
```

In [235]:
```
from sklearn.svm import LinearSVC
lvc=LinearSVC()
lvc.fit(X_train_ss,y_train)
l_pred=lvc.predict(X_test_ss)
```

In [236]:
```
l_pred
```

Out[236]:
```
array(['Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', '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', 'N', 'N', 'Y', 'Y',
       'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'N', 'Y', 'N', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'N',
       'N', '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', 'Y', 'N', 'Y', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'N', 'Y', 'Y',
       'N', 'Y', 'Y', 'N', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y',
       'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y'],
      dtype=object)
```

In [237]:
```
from sklearn.metrics import accuracy_score
lvc_acc=accuracy_score(y_test,l_pred)
print("lvc_acc_score : ",lvc_acc)
```

```
lvc_acc_score :  0.7532467532467533
```

In [238]:
```
from sklearn.metrics import confusion_matrix
c_mat=confusion_matrix(y_test,l_pred)
c_mat
```

Out[238]:
```
array([[18, 36],
       [ 2, 98]], dtype=int64)
```

In [239]:
```
from sklearn.metrics import classification_report
c_rep=classification_report(y_test,l_pred)
print(c_rep)
```

```
             precision    recall  f1-score   support

          N       0.90      0.33      0.49        54
          Y       0.73      0.98      0.84       100

avg / total       0.79      0.75      0.71       154
```

Step7:preformance Comparisons

In [240]:
```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(X_train_ss,y_train)
lr_pred=lr.predict(X_test_ss)

from sklearn.svm import LinearSVC
lvc=LinearSVC()
lvc.fit(X_train_ss,y_train)
l_pred=lvc.predict(X_test_ss)

from sklearn.metrics import accuracy_score
lvc_acc=accuracy_score(y_test,l_pred)
print("linear_svc_acc_score : ",lvc_acc)

from sklearn.metrics import accuracy_score
lr_acc=accuracy_score(y_test,lr_pred)
print("linear_reg_acc_score : ",lr_acc)
```

```
linear_svc_acc_score :  0.7532467532467533
linear_reg_acc_score :  0.7662337662337663
```

In [ ]: