Loan Status Prediction

Discription:

The objective of the model is predicting the loan status of a person, whether the loan would be

Inputs and the output:

Loan_id, gender, marital status, dependents, education, self_employed, applicantincome, coapplicantincome, loan amount, loan_amount_term, credit history, property_area

Output is: Loan_Status

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings('ignore')
```

loading the dataset

In [2]:
```
df=pd.read_csv("C:\\Users\\ASWINI\\Desktop\\DSP\\archive\\train_u6lujuX_CVtuZ9i (1).csv")
```

In [3]:
```
df
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coappl
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	
5	LP001011	Male	Yes	2	Graduate	Yes	5417	
6	LP001013	Male	Yes	0	Not Graduate	No	2333	
7	LP001014	Male	Yes	3+	Graduate	No	3036	
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	
20	LP001043	Male	Yes	0	Not Graduate	No	7660	
21	LP001046	Male	Yes	1	Graduate	No	5955	
22	LP001047	Male	Yes	0	Not Graduate	No	2600	
23	LP001050	NaN	Yes	2	Not Graduate	No	3365	
24	LP001052	Male	Yes	1	Graduate	NaN	3717	
25	LP001066	Male	Yes	0	Graduate	Yes	9560	
26	LP001068	Male	Yes	0	Graduate	No	2799	
27	LP001073	Male	Yes	2	Not Graduate	No	4226	
28	LP001086	Male	No	0	Not Graduate	No	1442	
29	LP001087	Female	No	2	Graduate	NaN	3750	
...	...	...	...	...	...	...	...	
584	LP002911	Male	Yes	1	Graduate	No	2787	
585	LP002912	Male	Yes	1	Graduate	No	4283	
586	LP002916	Male	Yes	0	Graduate	No	2297	
587	LP002917	Female	No	0	Not Graduate	No	2165	
588	LP002925	NaN	No	0	Graduate	No	4750	
589	LP002926	Male	Yes	2	Graduate	Yes	2726	
590	LP002928	Male	Yes	0	Graduate	No	3000	
591	LP002931	Male	Yes	2	Graduate	Yes	6000	
592	LP002933	NaN	No	3+	Graduate	Yes	9357	
593	LP002936	Male	Yes	0	Graduate	No	3859	
594	LP002938	Male	Yes	0	Graduate	Yes	16120	
595	LP002940	Male	No	0	Not Graduate	No	3833	
596	LP002941	Male	Yes	2	Not Graduate	Yes	6383	
597	LP002943	Male	No	NaN	Graduate	No	2987	
598	LP002945	Male	Yes	0	Graduate	Yes	9963	
599	LP002948	Male	Yes	2	Graduate	No	5780	
600	LP002949	Female	No	3+	Graduate	NaN	416	
601	LP002950	Male	Yes	0	Not Graduate	NaN	2894	
602	LP002953	Male	Yes	3+	Graduate	No	5703	
603	LP002958	Male	No	0	Graduate	No	3676	
604	LP002959	Female	Yes	1	Graduate	No	12000	
605	LP002960	Male	Yes	0	Not Graduate	No	2400	
606	LP002961	Male	Yes	1	Graduate	No	3400	
607	LP002964	Male	Yes	2	Not Graduate	No	3987	
608	LP002974	Male	Yes	0	Graduate	No	3232	
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 [4]:
```
df.head()
```

Out[4]:

	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 [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
Loan_ID              614 non-null object
Gender               601 non-null object
Married              611 non-null object
Dependents           599 non-null object
Education            614 non-null object
Self_Employed        582 non-null object
ApplicantIncome      614 non-null int64
CoapplicantIncome    614 non-null float64
LoanAmount           592 non-null float64
Loan_Amount_Term     600 non-null float64
Credit_History       564 non-null float64
Property_Area        614 non-null object
Loan_Status          614 non-null object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.4+ KB
```

In [6]:
```
#no.of rows and columns
df.shape
```

Out[6]:
```
(614, 13)
```

In [7]:
```
df.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]:
```
# droping the missing values
df=df.dropna()
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
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 [10]:
```
df.shape
```

Out[10]:
```
(480, 13)
```

In [11]:
```
#statistical measures
df.describe()
```

Out[11]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	480.000000	480.000000	480.000000	480.000000	480.000000
mean	5364.231250	1581.093583	144.735417	342.050000	0.854167
std	5668.251251	2617.692267	80.508164	65.212401	0.353307
min	150.000000	0.000000	9.000000	36.000000	0.000000
25%	2898.750000	0.000000	100.000000	360.000000	1.000000
50%	3859.000000	1084.500000	128.000000	360.000000	1.000000
75%	5852.500000	2253.250000	170.000000	360.000000	1.000000
max	81000.000000	33837.000000	600.000000	480.000000	1.000000

In [12]:
```
#label encoding
df.replace({"Loan_Status":{'N':0,'Y':1}},inplace=True)
```

In [13]:
```
df.head()
```

Out[13]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
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	

In [14]:
```
#dependent column value
df['Dependents'].value_counts()
```

Out[14]:
```
0     274
2      85
1      80
3+     41
Name: Dependents, dtype: int64
```

In [15]:
```
#replacing the value of 3+ to 4
df=df.replace(to_replace='3+', value=4)
```

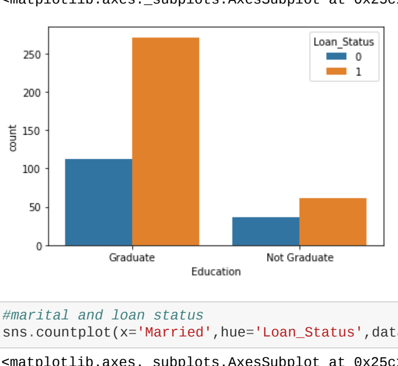
In [16]:
```
#dependent values
df['Dependents'].value_counts()
```

Out[16]:
```
0    274
2     85
1     80
4     41
Name: Dependents, dtype: int64
```

Data visualization

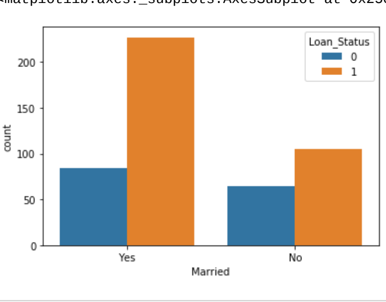
In [17]:
```
#Education and loan status
sns.countplot(x='Education', hue='Loan_Status',data=df)
```

Out[17]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c15b01198>
```



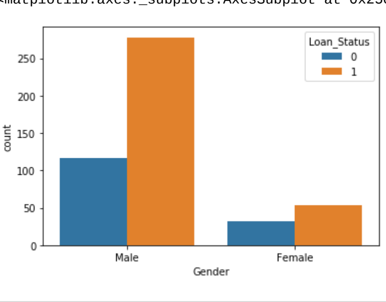
In [18]:
```
#marital and loan status
sns.countplot(x='Married',hue='Loan_Status',data=df)
```

Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c15bfdc18>
```



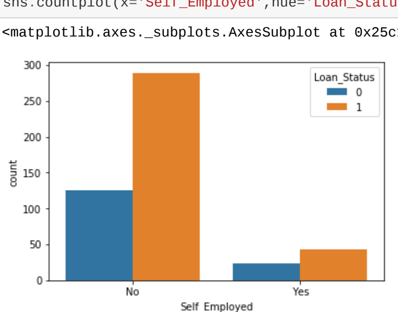
In [19]:
```
#gender and loan status
sns.countplot(x='Gender',hue='Loan_Status',data=df)
```

Out[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c15c6bbe0>
```



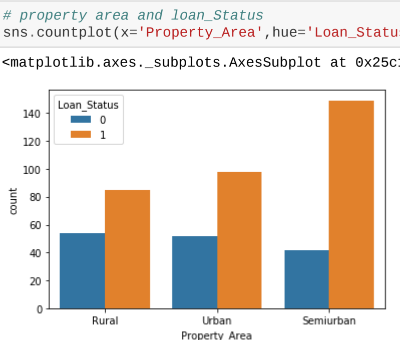
In [20]:
```
#self employed and loan statis
sns.countplot(x='Self_Employed',hue='Loan_Status',data=df)
```

Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c15cc4898>
```



In [21]:
```
# property area and loan_Status
sns.countplot(x='Property_Area',hue='Loan_Status',data=df)
```

Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x25c15d30828>
```



In [22]:
```
#convert categorical columns to numerical values
df.replace({'Married':{'No':0,'Yes':1},'Gender':{'Male':1,'Female':0},'Education':{'Graduate':1,'Not Graduate':0},'Self_Employed':{'No':0,'Yes':1},'Property_Area':{'Rural':0,'Semiurban':1,'Urban':2}},inplace=True)
```

In [23]:
```
df.head()
```

Out[23]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica
1	LP001003	1	1	1	1	0	4583	
2	LP001005	1	1	0	1	1	3000	
3	LP001006	1	1	0	0	0	2583	
4	LP001008	1	0	0	1	0	6000	
5	LP001011	1	1	2	1	1	5417	

In [24]:
```
df['Property_Area'].value_counts()
```

Out[24]:
```
1    191
2    150
0    139
Name: Property_Area, dtype: int64
```

In [25]:
```
#Seperating data and label
x=df.drop(columns=['Loan_ID','Loan_Status'],axis=1)
y=df['Loan_Status']
```

In [26]:
```
print(x)
print(y)
```

```
     Gender  Married Dependents  Education  Self_Employed  ApplicantInco
me  \
1         1        1          1          1              0             45
83
2         1        1          0          1              1             30
00
3         1        1          0          0              0             25
83
4         1        0          0          1              0             60
00
5         1        1          2          1              1             54
17
6         1        1          0          0              0             23
33
7         1        1          4          1              0             30
36
8         1        1          2          1              0             40
06
9         1        1          1          1              0            128
41
10        1        1          2          1              0             32
00
12        1        1          2          1              0             30
73
13        1        0          0          1              0             18
53
14        1        1          2          1              0             12
99
15        1        0          0          1              0             49
50
17        0        0          0          1              0             35
10
18        1        1          0          0              0             48
87
20        1        1          0          0              0             76
60
21        1        1          1          1              0             59
55
22        1        1          0          0              0             26
00
25        1        1          0          1              1             95
60
26        1        1          0          1              0             27
99
27        1        1          2          0              0             42
26
28        1        0          0          0              0             14
42
31        1        0          0          1              0             31
67
32        1        0          1          1              1             46
92
33        1        1          0          1              0             35
00
34        1        0          4          1              0            125
00
37        0        1          0          1              0             36
67
38        1        0          0          1              0             41
66
39        1        0          0          0              0             37
48
..      ...      ...        ...        ...            ...
...
575       1        1          2          1              0             31
59
577       1        0          0          1              0             32
29
578       1        1          1          1              0             17
82
580       1        1          2          1              0             65
40
581       1        0          0          1              0             18
36
582       0        1          0          1              0             31
66
584       1        1          1          1              0             27
87
585       1        1          1          1              0             42
83
586       1        1          0          1              0             22
97
587       0        0          0          0              0             21
65
589       1        1          2          1              1             27
26
590       1        1          0          1              0             30
00
591       1        1          2          1              1             60
00
593       1        1          0          1              0             38
59
594       1        1          0          1              1            161
20
595       1        0          0          0              0             38
33
596       1        1          2          0              1             63
83
598       1        1          0          1              1             99
63
599       1        1          2          1              0             57
80
602       1        1          4          1              0             57
03
603       1        0          0          1              0             36
76
604       0        1          1          1              0            120
00
606       1        1          1          1              0             34
00
607       1        1          2          0              0             39
87
608       1        1          0          1              0             32
32
609       0        0          0          1              0             29
00
610       1        1          4          1              0             41
06
611       1        1          1          1              0             80
72
612       1        1          2          1              0             75
83
613       0        0          0          1              1             45
83

     CoapplicantIncome  LoanAmount  Loan_Amount_Term  Credit_History  \
1               1508.0       128.0             360.0             1.0
2                  0.0        66.0             360.0             1.0
3               2358.0       120.0             360.0             1.0
4                  0.0       141.0             360.0             1.0
5               4196.0       267.0             360.0             1.0
6               1516.0        95.0             360.0             1.0
7               2504.0       158.0             360.0             0.0
8               1526.0       168.0             360.0             1.0
9              10968.0       349.0             360.0             1.0
10               700.0        70.0             360.0             1.0
12              8106.0       200.0             360.0             1.0
13              2840.0       114.0             360.0             1.0
14              1086.0        17.0             120.0             1.0
15                 0.0       125.0             360.0             1.0
17                 0.0        76.0             360.0             0.0
18                 0.0       133.0             360.0             1.0
20                 0.0       104.0             360.0             0.0
21              5625.0       315.0             360.0             1.0
22              1911.0       116.0             360.0             0.0
25                 0.0       191.0             360.0             1.0
26              2253.0       122.0             360.0             1.0
27              1040.0       110.0             360.0             1.0
28                 0.0        35.0             360.0             1.0
31                 0.0        74.0             360.0             1.0
32                 0.0       106.0             360.0             1.0
33              1667.0       114.0             360.0             1.0
34              3000.0       320.0             360.0             1.0
37              1459.0       144.0             360.0             1.0
38              7210.0       184.0             360.0             1.0
39              1668.0       110.0             360.0             1.0
..                 ...         ...               ...             ...
575              461.0       108.0              84.0             1.0
577             2739.0       110.0             360.0             1.0
578             2232.0       107.0             360.0             1.0
580                0.0       205.0             360.0             1.0
581            33837.0        90.0             360.0             1.0
582                0.0        36.0             360.0             1.0
584             1917.0       146.0             360.0             0.0
585             3000.0       172.0              84.0             1.0
586             1522.0       104.0             360.0             1.0
587                0.0        70.0             360.0             1.0
589                0.0       106.0             360.0             0.0
590             3416.0        56.0             180.0             1.0
591                0.0       205.0             240.0             1.0
593             3300.0       142.0             180.0             1.0
594                0.0       260.0             360.0             1.0
595                0.0       110.0             360.0             1.0
596             1000.0       187.0             360.0             1.0
598                0.0       180.0             360.0             1.0
599                0.0       192.0             360.0             1.0
602                0.0       128.0             360.0             1.0
603             4301.0       172.0             360.0             1.0
604                0.0       496.0             360.0             1.0
606             2500.0       173.0             360.0             1.0
607             1411.0       157.0             360.0             1.0
608             1950.0       108.0             360.0             1.0
609                0.0        71.0             360.0             1.0
610                0.0        40.0             180.0             1.0
611              240.0       253.0             360.0             1.0
612                0.0       187.0             360.0             1.0
613                0.0       133.0             360.0             0.0

     Property_Area
1                0
2                2
3                2
4                2
5                2
6                2
7                1
8                2
9                1
10               2
12               2
13               0
14               2
15               2
17               2
18               0
20               2
21               2
22               1
25               1
26               1
27               2
28               2
31               2
32               0
33               1
34               0
37               1
38               2
39               1
..             ...
575              2
577              2
578              0
580              1
581              2
582              1
584              0
585              0
586              2
587              1
589              1
590              1
591              1
593              0
594              2
595              0
596              0
598              0
599              2
602              2
603              0
604              1
606              1
607              0
608              0
609              0
610              0
611              2
612              2
613              1

[480 rows x 11 columns]
```

Train Test Split

In [27]:
```
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.1, stratify=y, random_state=2)
```

In [28]:
```
print(x.shape, x_train.shape, x_test.shape)
```

```
(480, 11) (432, 11) (48, 11)
```

Training the model

Support Vector Machine Model

In [29]:
```
classifier=svm.SVC(kernel='linear')
```

In [30]:
```
#training the support vector model
classifier.fit(x_train,y_train)
```

Out[30]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

Model Evaluation

In [31]:
```
#accuracy score on training data
x_train_prediction=classifier.predict(x_train)
training_data_accuracy=accuracy_score(x_train_prediction, y_train)
```

In [32]:
```
print('Accuracy on training data:',training_data_accuracy)
```

```
Accuracy on training data: 0.7986111111111112
```

In [33]:
```
#accuracy score on testing data
x_test_prediction=classifier.predict(x_test)
testing_data_accuracy=accuracy_score(x_test_prediction, y_test)
```

In [34]:
```
print('Accuracy on testing data:',testing_data_accuracy)
```

```
Accuracy on testing data: 0.8333333333333334
```

Making predictive system

In [41]:
```
input_data=(1,0,0,1,0,6000,0.0,141.0,360.0,1.0,2)

#change the input data to a numpy array

input_data_as_numpy_array = np.asarray(input_data)

#reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = classifier.predict(input_data_reshaped)
print(prediction)
```

```
[1]
```

In [ ]:

S180401

B.Aswini

In [ ]: