Loan Status Prediction using Machine Learning

August 2, 2023

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

[2]:
```
data=pd.read_csv('train_u6lujuX_CVtuZ9i (1).csv')
```

[3]:
```
data.head()
```

[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

[4]:
```
data.shape
```

[4]:
```
(614, 13)
```

[5]:
```
data.describe()
```

[5]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
count	614.000000	614.000000	592.000000	600.00000

```
mean       5403.459283        1621.245798  146.412162         342.00000
std        6109.041673        2926.248369   85.587325          65.12041
min         150.000000           0.000000    9.000000          12.00000
25%        2877.500000           0.000000  100.000000         360.00000
50%        3812.500000        1188.500000  128.000000         360.00000
75%        5795.000000        2297.250000  168.000000         360.00000
max       81000.000000       41667.000000  700.000000         480.00000

       Credit_History
count      564.000000
mean         0.842199
std          0.364878
min          0.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          1.000000
```

[6]:
```
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
```

[7]:
```
data.isnull().sum()
```

[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
```

[8]:
```
#Droping The Missing Values
```

[9]:
```
data=data.dropna()
```

[10]:
```
data.head()
```

[10]:
```
     Loan_ID Gender Married Dependents     Education Self_Employed  \
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

   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
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

   Credit_History Property_Area Loan_Status
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
```

[11]:
```
data.isnull().sum()
```

[11]:
```
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
```

[12]:
```
from sklearn.preprocessing import LabelEncoder
```

[13]:
```
obj=LabelEncoder()
```

[14]:
```
Loan_Status=data['Loan_Status']
```

[15]:
```
Loan_Status=obj.fit_transform(Loan_Status)
data["Loan_Status"]=Loan_Status
```

[16]:
```
Loan_Status
```

[16]:
```
array([0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1,
       0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1,
       0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0])
```

[17]:
```
obj.classes_
```

[17]:
```
array(['N', 'Y'], dtype=object)
```

[18]:
```
data['Dependents'].value_counts()
```

[18]:
```
0     274
2      85
1      80
3+     41
Name: Dependents, dtype: int64
```

[19]:
```
#Replaceing the value of 3+ to 4
```

[20]:
```
data=data.replace(to_replace='3+' ,value='4')
```

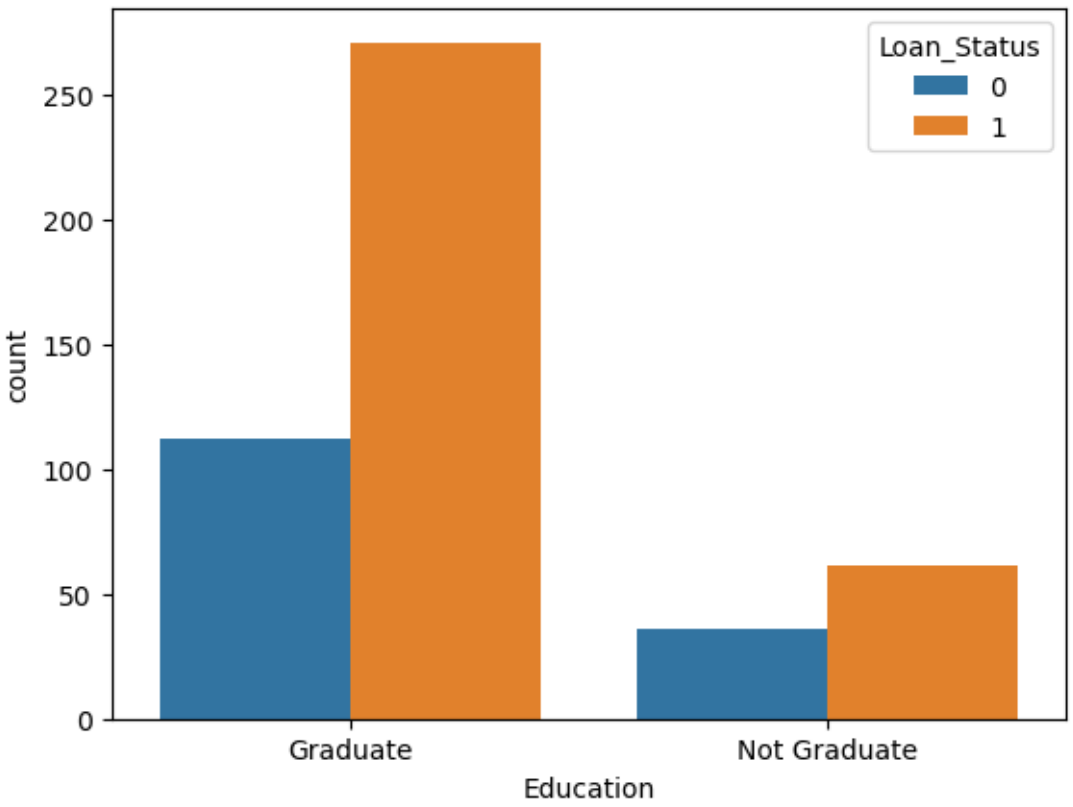
[21]:
```
data['Dependents'].value_counts()
```

[21]:
```
0    274
2     85
1     80
4     41
Name: Dependents, dtype: int64
```

[22]:
```
# Education and loan status
```

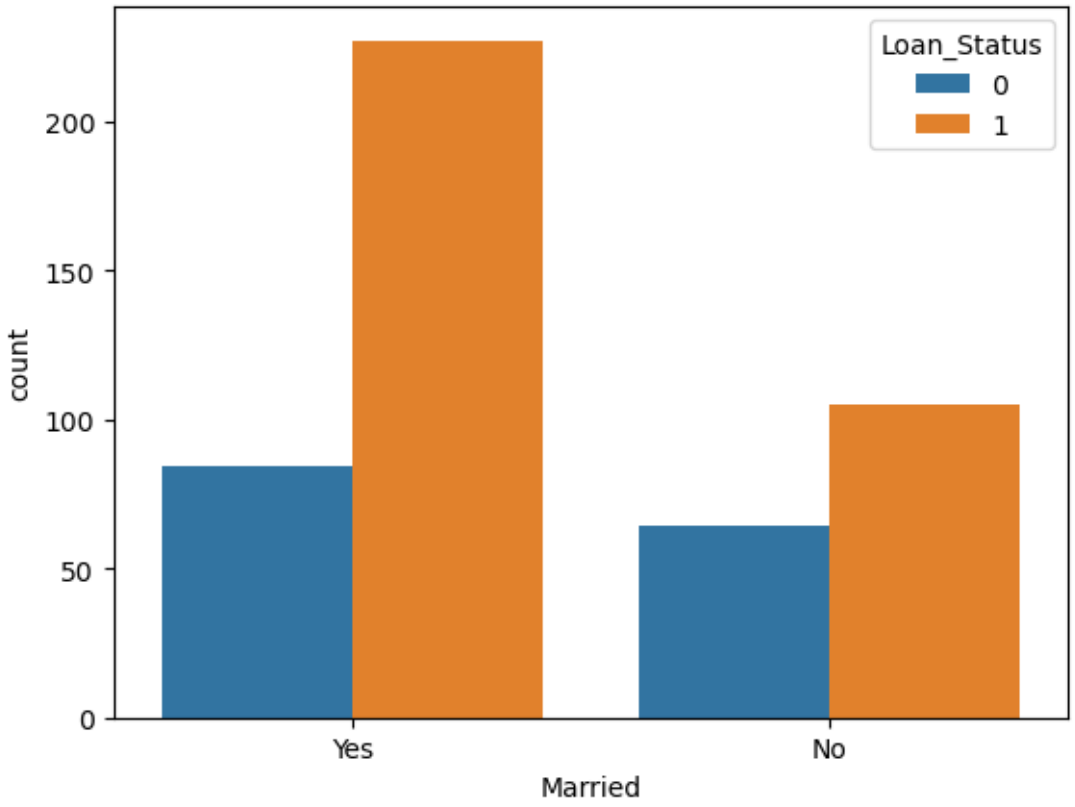
[23]:
```
sns.countplot(data=data,x='Education',hue='Loan_Status')
```

[23]:
```
<Axes: xlabel='Education', ylabel='count'>
```



[24]:
```
sns.countplot(data=data,x='Married',hue='Loan_Status')
```

[24]:
```
<Axes: xlabel='Married', ylabel='count'>
```



[25]:
```
#convert categorical columns to numerical values
```

[26]:
```
Gender=data['Gender']
```

[27]:
```
Gender=obj.fit_transform(Gender)
data['Gender']=Gender
```

[28]:
```
obj.classes_
```

[28]:
```
array(['Female', 'Male'], dtype=object)
```

[29]:
```
Married=data['Married']
```

[30]:
```
Married=obj.fit_transform(Married)
data['Married']=Married
```

[31]:
```
obj.classes_
```

[31]:
```
array(['No', 'Yes'], dtype=object)
```

[32]:
```
Self_Employed=data['Self_Employed']
```

[33]:
```
Self_Employed=obj.fit_transform(Self_Employed)
data['Self_Employed']=Self_Employed
```

[34]:
```
obj.classes_
```

[34]:
```
array(['No', 'Yes'], dtype=object)
```

[35]:
```
Property_Area=data['Property_Area']
```

[36]:
```
Property_Area=obj.fit_transform(Property_Area)
data['Property_Area']=Property_Area
```

[37]:
```
obj.classes_
```

[37]:
```
array(['Rural', 'Semiurban', 'Urban'], dtype=object)
```

[38]:
```
Education=data['Education']
```

[39]:
```
Education=obj.fit_transform(Education)
data["Education"]=Education
```

[40]:
```
obj.classes_
```

[40]:
```
array(['Graduate', 'Not Graduate'], dtype=object)
```

[41]:
```
data.head()
```

[41]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	\
1	LP001003	1	1	1	0	0	
2	LP001005	1	1	0	0	1	
3	LP001006	1	1	0	1	0	
4	LP001008	1	0	0	0	0	
5	LP001011	1	1	2	0	1	

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	\
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	

	Credit_History	Property_Area	Loan_Status

```
1             1.0              0              0
2             1.0              2              1
3             1.0              2              1
4             1.0              2              1
5             1.0              2              1
```

[42]:
```
x=data.drop(columns=(['Loan_ID','Loan_Status']),axis=1)
```

[43]:
```
x.head()
```

[43]:
```
   Gender  Married Dependents  Education  Self_Employed  ApplicantIncome  \
1       1        1           1          0              0             4583
2       1        1           0          0              1             3000
3       1        1           0          1              0             2583
4       1        0           0          0              0             6000
5       1        1           2          0              1             5417

   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

   Property_Area
1              0
2              2
3              2
4              2
5              2
```

[44]:
```
y=data.Loan_Status
```

[45]:
```
y.head()
```

[45]:
```
1    0
2    1
3    1
4    1
5    1
Name: Loan_Status, dtype: int32
```

[46]:
```
from sklearn.model_selection import train_test_split
```

[47]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

[48]:
```
x_train.shape
```

[48]:
```
(336, 11)
```

[49]:
```
x_test.shape
```

[49]:
```
(144, 11)
```

[50]:
```
from sklearn.svm import SVC
model=SVC(kernel='linear')
```

[51]:
```
model.fit(x_train,y_train)
```

[51]:
```
SVC(kernel='linear')
```

[52]:
```
pred=model.predict(x_train)
```

[53]:
```
pred
```

[53]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 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, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 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, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 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, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 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, 1, 1, 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, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 0])
```

[61]:
```
from sklearn.metrics import accuracy_score
```

[62]:
```
accuracy_score=accuracy_score(y_train,pred)
```

[63]:
```
accuracy_score
```

[63]:
```
0.7857142857142857
```

[ ]:
```

```