

ediction-of-loan-using-logisticreg

December 14, 2023

```
[5]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
[6]: data=pd.read_csv(r"E:\xlsx\loandata1.csv")
```

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

```
[7]:
```

	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	

	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	

	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

```
[10]: 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

```
[8]: data.shape
```

```
[8]: (614, 13)
```

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

```
[12]:
```

	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

```
[13]: list(data)
```

```
[13]: ['Loan_ID',
      'Gender',
      'Married',
      'Dependents',
      'Education',
      'Self_Employed',
      'ApplicantIncome',
      'CoapplicantIncome',
      'LoanAmount',
      'Loan_Amount_Term',
      'Credit_History',
      'Property_Area',
      'Loan_Status']
```

```
[14]: data.isna().sum()
```

```
[14]: 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
```

```
[15]: data['LoanAmount']=data['LoanAmount'].fillna(data['LoanAmount'].mean())
```

```
[16]: data['Credit_History']=data['Credit_History'].fillna(data['Credit_History'].
      ↳median())
```

```
[17]: data.dropna(inplace=True)
```

```
[18]: data.isna().sum()
```

```
[18]: 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

```

```
[19]: data.shape
```

```
[19]: (542, 13)
```

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

```

[20]:   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

      ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0                5849                0.0   146.412162             360.0
1                4583             1508.0   128.000000             360.0
2                3000                0.0    66.000000             360.0
3                2583             2358.0   120.000000             360.0
4                6000                0.0   141.000000             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

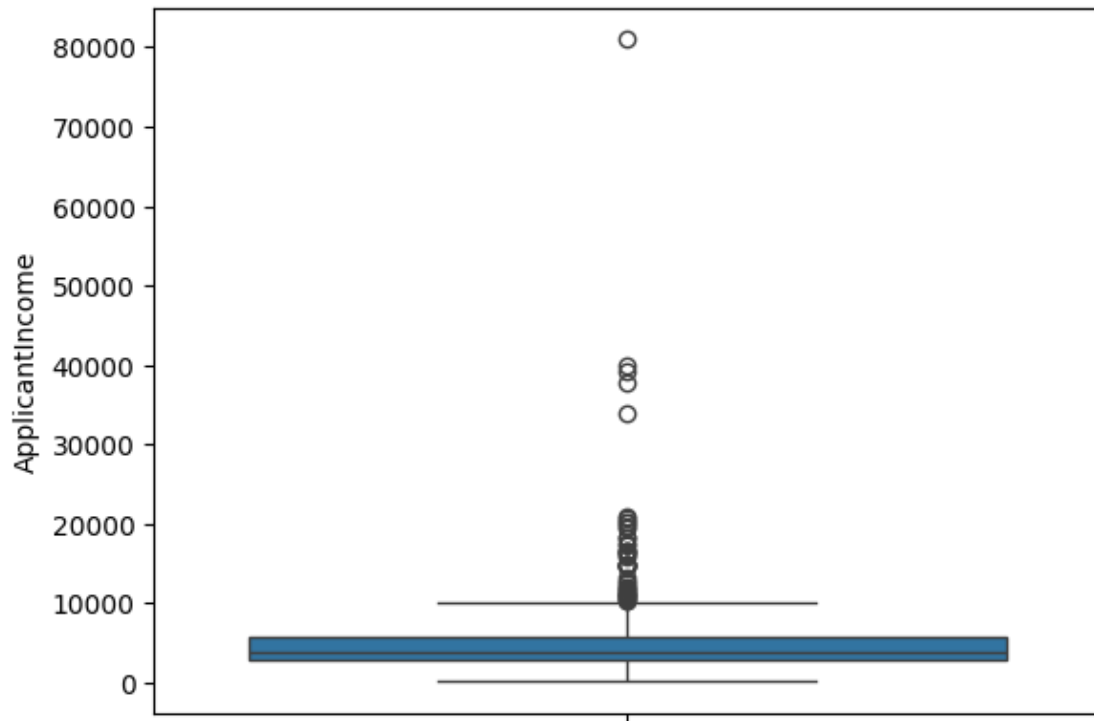
```

```

[21]: import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(data.ApplicantIncome)

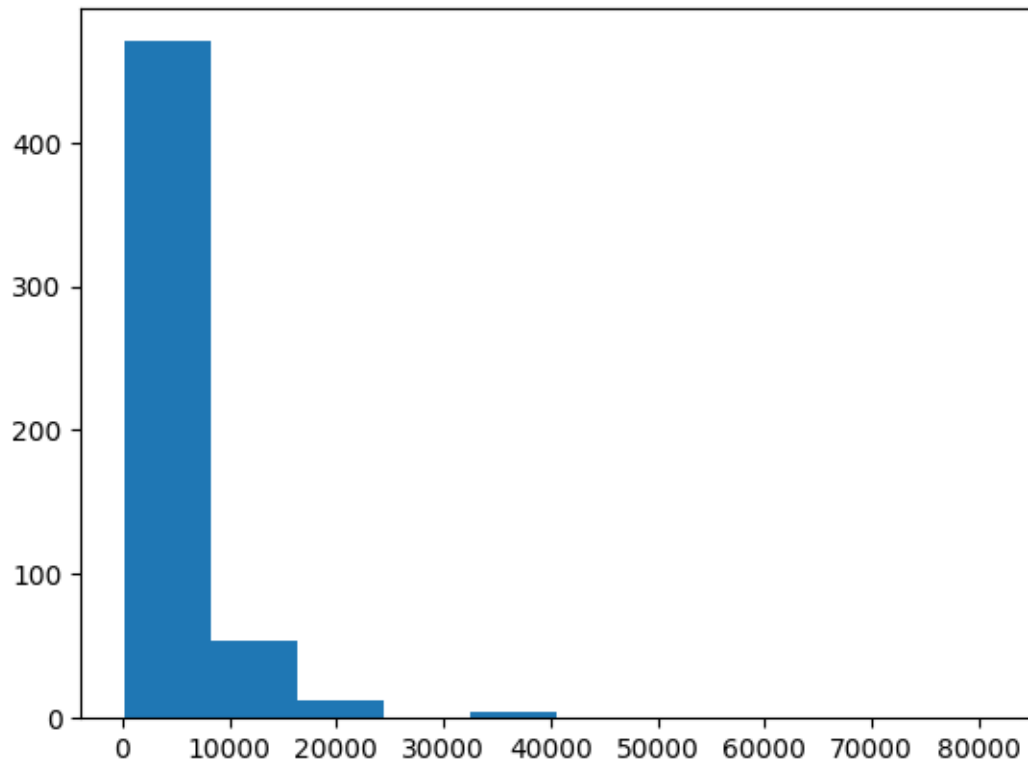
```

```
[21]: <Axes: ylabel='ApplicantIncome'>
```



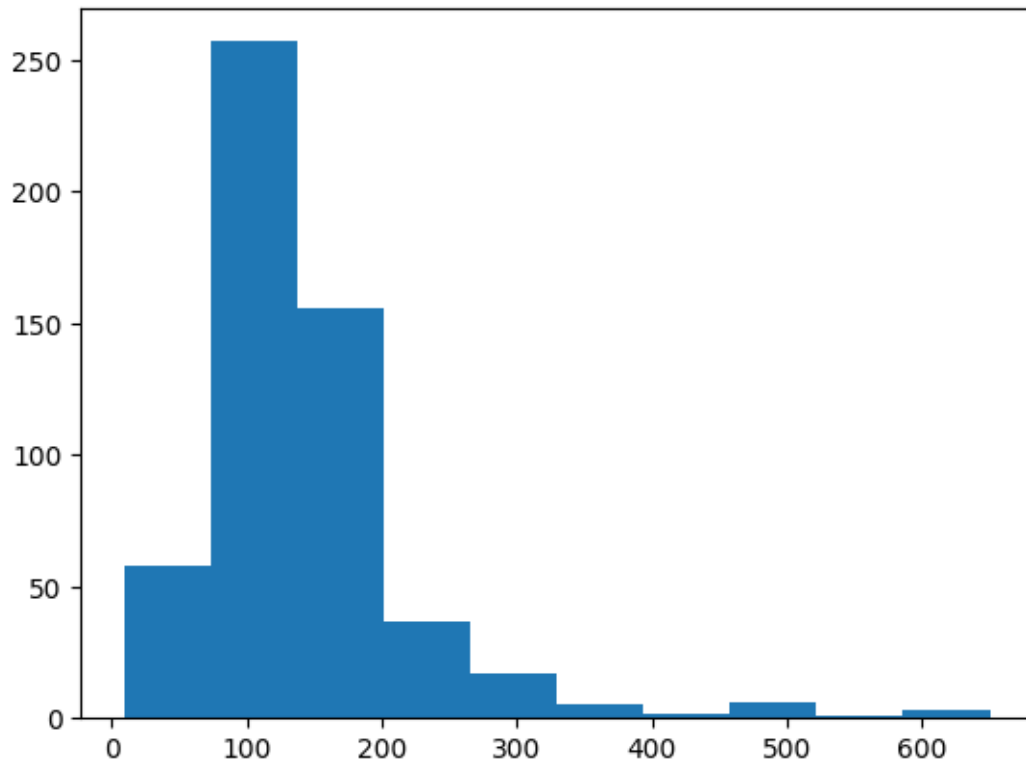
```
[22]: plt.hist(data['ApplicantIncome'])
```

```
[22]: (array([470., 54., 13., 0., 4., 0., 0., 0., 0., 1.]),
      array([ 150., 8235., 16320., 24405., 32490., 40575., 48660., 56745.,
              64830., 72915., 81000.]),
      <BarContainer object of 10 artists>)
```



```
[23]: plt.hist(data['LoanAmount'])
```

```
[23]: (array([ 58., 257., 156.,  37.,  17.,   5.,   2.,   6.,   1.,   3.]),  
      array([  9. ,  73.1, 137.2, 201.3, 265.4, 329.5, 393.6, 457.7, 521.8,  
            585.9, 650. ]),  
      <BarContainer object of 10 artists>)
```



```
[24]: data['Loan_Status'].replace('Y',1,inplace=True)
      data['Loan_Status'].replace('N',0,inplace=True)
```

```
[26]: data['Loan_Status'].value_counts()
```

```
[26]: Loan_Status
      1    376
      0    166
      Name: count, dtype: int64
```

```
[27]: data.Gender=data.Gender.map({'Male':1,'Female':0})
      data['Gender'].value_counts()
```

```
[27]: Gender
      1    444
      0     98
      Name: count, dtype: int64
```

```
[28]: data.Married=data.Married.map({'Yes':1,'No':0})
      data['Married'].value_counts()
```

```
[28]: Married
      1    355
      0    187
      Name: count, dtype: int64
```

```
[29]: data.Dependents=data.Dependents.map({'0':0,'1':1,'2':2,'3+':3})
      data['Dependents'].value_counts()
```

```
[29]: Dependents
      0    309
      1     94
      2     94
      3     45
      Name: count, dtype: int64
```

```
[30]: data.Education=data.Education.map({'Graduate':1,'Not Graduate':0})
      data['Education'].value_counts()
```

```
[30]: Education
      1    425
      0    117
      Name: count, dtype: int64
```

```
[31]: data.Self_Employed=data.Self_Employed.map({'Yes':1,'No':0})
      data['Self_Employed'].value_counts()
```

```
[31]: Self_Employed
      0    467
      1     75
      Name: count, dtype: int64
```

```
[32]: data.Property_Area=data.Property_Area.map({'Urban':2,'Rural':0,'Semiurban':1})
      data['Property_Area'].value_counts()
```

```
[32]: Property_Area
      1    209
      2    174
      0    159
      Name: count, dtype: int64
```

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

```
[33]:   Loan_ID  Gender  Married  Dependents  Education  Self_Employed  \
0  LP001002      1        0           0          1              0
1  LP001003      1        1           1          1              0
2  LP001005      1        1           0          1              1
3  LP001006      1        1           0          0              0
```


4	LP001008	1	0	0	1	0
---	----------	---	---	---	---	---

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term \
0	5849	0.0	146.412162	360.0
1	4583	1508.0	128.000000	360.0
2	3000	0.0	66.000000	360.0
3	2583	2358.0	120.000000	360.0
4	6000	0.0	141.000000	360.0

	Credit_History	Property_Area	Loan_Status
0	1.0	2	1
1	1.0	0	0
2	1.0	2	1
3	1.0	2	1
4	1.0	2	1

```
[34]: data.shape
```

```
[34]: (542, 13)
```

```
[35]: data['Credit_History'].value_counts()
```

```
[35]: Credit_History
1.0    468
0.0    74
Name: count, dtype: int64
```

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

```
[52]: x
```

```
[52]:
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome \
0	1	0	0	1	0	5849
1	1	1	1	1	0	4583
2	1	1	0	1	1	3000
3	1	1	0	0	0	2583
4	1	0	0	1	0	6000
..
609	0	0	0	1	0	2900
610	1	1	3	1	0	4106
611	1	1	1	1	0	8072
612	1	1	2	1	0	7583
613	0	0	0	1	1	4583

	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History \
0	0.0	146.412162	360.0	1.0

1	1508.0	128.000000	360.0	1.0
2	0.0	66.000000	360.0	1.0
3	2358.0	120.000000	360.0	1.0
4	0.0	141.000000	360.0	1.0
..
609	0.0	71.000000	360.0	1.0
610	0.0	40.000000	180.0	1.0
611	240.0	253.000000	360.0	1.0
612	0.0	187.000000	360.0	1.0
613	0.0	133.000000	360.0	0.0

	Property_Area
0	2
1	0
2	2
3	2
4	2
..	...
609	0
610	0
611	2
612	2
613	1

[542 rows x 11 columns]

```
[40]: from sklearn.model_selection import train_test_split
      from sklearn import metrics
      x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

```
[46]: from sklearn.linear_model import LogisticRegression
      Classifier=LogisticRegression()
      Classifier.fit(x_train,y_train)
```

```
[46]: LogisticRegression()
```

```
[47]: y_pred=Classifier.predict(x_test)
      y_pred
```

```
[47]: array([1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 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, 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, 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, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1,
            1, 0, 1, 1, 1, 1, 1, 1, 1], dtype=int64)
```

```
[48]: from sklearn.metrics import confusion_matrix  
      confusion_matrix(y_test,y_pred)
```

```
[48]: array([[ 16,  31],  
          [  0, 116]], dtype=int64)
```

```
[49]: from sklearn.metrics import accuracy_score  
      accuracy_score(y_test,y_pred)
```

```
[49]: 0.8098159509202454
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```