

Loan Eligibility Prediction

BHAGAVATULA SRIHARSHA

10-05-2022

Abstract

In the banking system, banks have a variety of products to provide, but credit lines are their primary source of revenue. As a result, they will profit from the interest earned on the loans they make. Loans, or whether customers repay or default on their loans, affect a bank's profit or loss. The bank's Non-Performing Assets will be reduced by forecasting loan defaulters. As a result, further investigation into this occurrence is essential. Because precise forecasts are essential for benefit maximisation, it's crucial to analyse and compare the various methodologies. The logistic regression model is an important predictive analytics tool for detecting loan defaulters.

1. Problem Statement

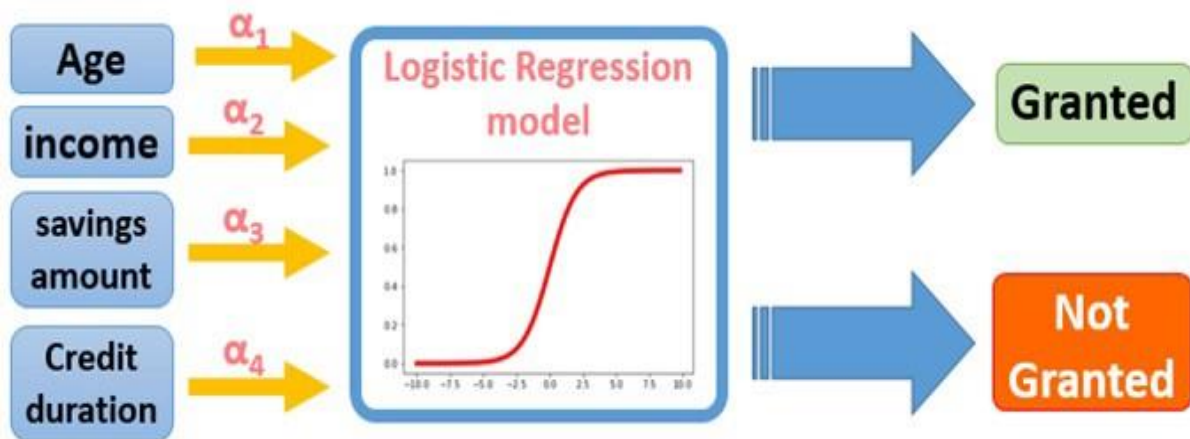
Loans are the core business of banks. The main profit comes directly from the loan's interest. The loan companies grant a loan after an intensive process of verification and validation. Even after this process the banks are not assured with the loan recovery. They face a huge loss as the borrower delays or completely ignores to pay the loan taken from the bank.

This may arise due to many reasons some of them would be misinterpretation or favouritism with the borrower or collectively can be considered as the human error. To overcome this problem we can develop a model which predicts whether the borrower would pay the loan on time using Machine Learning.

When seen from the borrower's side even, There are many instances where the borrower is capable of repaying the loan promptly but the loan gets rejected due to some orthodox decisions where the loan providing is decided on some single factor.

2. Business Need Assessment

As mentioned above, Loans are the primary income for the banks as their main profit comes directly from the loan's interest. If the Machine Learning model is able to predict the extent of the loan recovery from the individual efficiently, Then the banks would have a better profits as they would assess the data of the borrower and check the prediction of the model, decide whether to provide loan to the borrower or not.



3. Target Specification and Characterisation

The target audience are the banks which provide loan. Most of the banks would be willing to provide as much as loan as possible which would increase their profit but the recovery is always the biggest fear which makes them a bit more conscious in providing the loan.

Factors considered in this model are-

- Gender(M or F)
- Marital status(Yes or No)
- Dependents(number of persons dependent)
- Education(education of the applicant)
- Self-employed(Yes or No)
- co applicant income(income of the co applicant)
- Loan amount(required amount of the borrower)
- Loan amount term(term duration)
- Credit history(ability to repay the loan)
- Property area(urban, semi urban or rural)

4. External Search

The model uses [dataset](#) from a repository in GitHub which consists of 614 rows of data which shows on what basis the loan was provided to the applicant. Based on this data the model would predict whether the applicant(new data) is eligible to the loan or not.

```
In [6]: df=pd.read_csv("train_data.csv")
df
```

```
Out[6]:
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	F
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	
...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	1.0	
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	1.0	
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	1.0	
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	1.0	
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0	

614 rows × 13 columns

5. Bench marking alternate products

There are many independent loan eligibility predicting firms and banks also maintain their own system to predict whether an applicant is eligible to receive the loan or not. One such independent Organisation is “[allcloud.in](#)”. It provides various functionalities in which “Loan Originating system” is such program which gives streamlines process across various channels and products thus enabling banks to issue loans and manage underwriting services in a few minutes, absolutely paperless.

6. Applicable Patents

The organisation “[allcloud.in](#)” is an open source software and it is free to integrate with any other platforms.

7. Applicable Constraints

- All necessary libraries in python like pandas, numpy, scikitlearn, matplotlib to execute the code cell is required
- A cloud space to store, update and retrieve the datasets
- An AI/ML expert body to monitor the working of the model and improvise it timely for better performance
- An easy to use UI for the client

8. Business Opportunity

If a company provides better prediction of loan eligibility than the banks then most of them would prefer using the services from the company as their profits increases as the loan defaulters are kept as minimum as possible.

This would increase the company’s engagement with the banks and hence a good revenue is generated as the banks trust the predictions of the company.

9. Concept Generation

The term “Loan” is both give and take thing, Banks need to keep providing loans to increase the income of it and the applicant would require loan for his/her personal reasons. Hence an easy way to help both banks and applicants in providing and getting loans respectively would be by using this machine learning model.

10. Concept Development

Firstly, we collect datasets of all types of applicants, remove all the unwanted data and train the machine learning model with that refined data. Once the model is trained we can take the values from the user(applicant details are entered by the bank) and give an output whether the applicant is eligible for the loan or not.

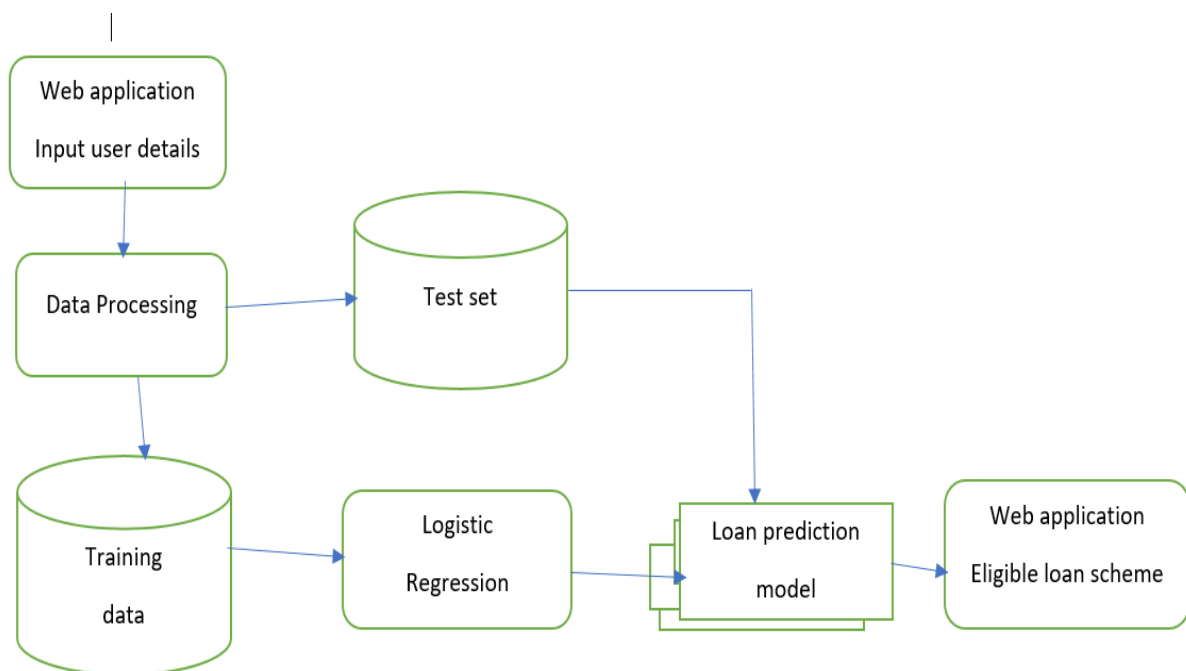
We can divide our project into two phases

First phase: In this phase, our model is more helpful to banks than the applicants as it only predicts whether the applicant is eligible for getting the loan or not.

Second phase: In this phase, the model takes the input, gives the output(whether the applicant is eligible for the loan or not) and also gives the possible ways for the applicant to get the loan(by decreasing the loan amount or increasing the interest etc).

11. Final Product Prototype

Schematic Diagram: The schematic diagram for the above model can be



Product Details

- Working

The machine learning model works using logistic regression which classifies the details entered by the user(applicant details) into whether the applicant can receive the loan or not.

- Data Sources

We use datasets for training and testing the model from Kaggle

- Algorithms

We use Logistic Regression algorithm as it gives the best accuracy among all other algorithms when tested with same test data set(83%)

- Team required to develop

To develop this project we need a team which consists of members from Machine Learning field and banking field where majority can be members from Machine Learning field.

CODE IMPLEMENTATION/VALIDATION ON SMALL SCALE

Initially we import all the necessary libraries required for the code implementation and read the csv file containing the dataset.

Understanding the data

We have 12 independent variables and 1 target variable, i.e. Loan_Status in the training dataset.

```
In [3]: import numpy as np
import pandas as pd
import matplotlib as plt

In [7]: df=pd.read_csv("train_data.csv")
df
```

Out[7]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0
...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	1.0
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	1.0
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	1.0
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	1.0
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0

614 rows x 13 columns

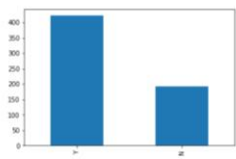
```
In [8]: df.columns
Out[8]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount', 'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'], dtype='object')
```

Missing value imputation

Let's list out feature-wise count of missing values.

```
In [14]: df['Loan_Status'].value_counts(normalize=True)
Out[14]: Y    0.687296
N    0.312704
Name: Loan_Status, dtype: float64

In [15]: df['Loan_Status'].value_counts().plot.bar()
Out[15]: <AxesSubplot>
```



```
In [16]: df.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
dtype: int64
```

There are missing values in Gender, Married, Dependents, Self_Employed, LoanAmount, Loan_Amount_Term, and Credit_History features.

We can consider these methods to fill the missing values:

- For numerical variables: imputation using mean or median
- For categorical variables: imputation using mode

```
In [17]: df['Gender'].fillna(df['Gender'].mode()[0], inplace=True)
df['Married'].fillna(df['Married'].mode()[0], inplace=True)
df['Dependents'].fillna(df['Dependents'].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 [18]: df['Loan_Amount_Term'].value_counts()

Out[18]:
360.0    512
180.0     44
480.0     15
300.0     13
84.0       4
240.0       4
120.0       3
36.0        2
60.0         2
12.0         1
Name: Loan_Amount_Term, dtype: int64

In [20]: df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0], inplace=True)
df['LoanAmount'].fillna(df['LoanAmount'].median(), inplace=True)

In [21]: df.isnull().sum()

Out[21]:
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 [20]: df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0], inplace=True)
df['LoanAmount'].fillna(df['LoanAmount'].median(), inplace=True)

In [21]: df.isnull().sum()

Out[21]:
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 [23]: df=df.drop('loan_ID',axis=1)
```

Project Building

We can remove ‘loan_ID’ as it does not have any effect in the prediction, and in fact eliminating all the unwanted data would give a better and optimised solutions.

```
In [21]: df.isnull().sum()

Out[21]:
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 [23]: df=df.drop('loan_ID',axis=1)
```

We will use scikit-learn (sklearn) for making different models which is an open source library for Python. It is one of the most efficient tools which contains many inbuilt functions that can be used for modelling in Python.

Consider the “Gender” variable. It has two classes, Male and Female.

As logistic regression takes only the numerical values as input, we have to change male and female into a numerical value.

We will divide the data into two parts

X- All the columns except the target column(type=numpy array)

y- Target variable(type=numpy array)

[illegible]

The dataset has been divided into training and validation part. Let us import LogisticRegression and fit the logistic regression model.

[illegible]

To take the input from the user for predicting the loan eligibility, we declare eleven variables of type float and take input from the user

After taking the input from the user, we create an array(numpy array) and place those input values in it.

By using this array we can use ‘model.predict(array)’ to get the prediction.

```

In [60]: asf=float(input("Enter the gender of the applicant(male=1,female=0):"))
         bsf=float(input("Enter the Marital status of the applicant(married=1,unmarried=0):"))
         csf=float(input("Enter the number of dependents on the applicant:"))
         dsf=float(input("Enter the educational qualification of the applicant(Graduate=1,Not Graduate=0):"))
         esf=float(input("Enter whether the applicant is self employed or not(yes=1,no=0):"))
         fsf=float(input("Enter the applicant's income:"))
         gsf=float(input("Enter co-applicant's income:"))
         hsf=float(input("Enter the loan amount:"))
         isf=float(input("Enter the loan amount term:"))
         jsf=float(input("Enter the credit history:"))
         ksf=float(input("Enter the property area of the applicant(urban=1, semi urban=0, rural=-1):"))
         x=np.array([a,b,c,d,e,f,g,h,i,j,k])
         model.predict(x)

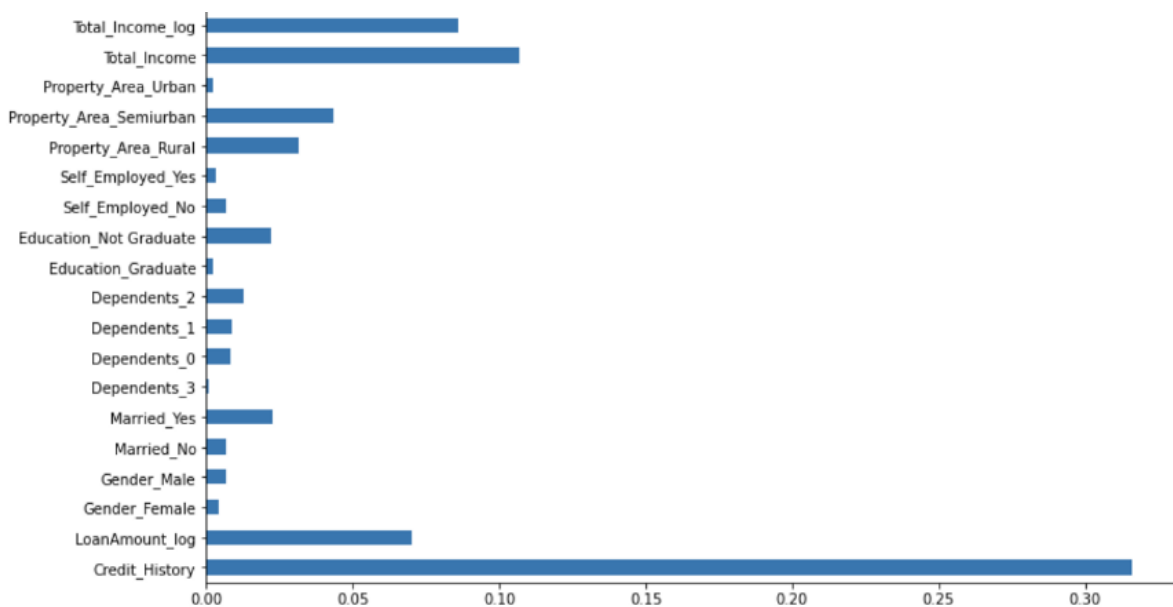
Enter the gender of the applicant(male=1,female=0):1
Enter the Marital status of the applicant(married=1,unmarried=0):1
Enter the number of dependents on the applicant:1
Enter the educational qualification of the applicant(Graduate=1,Not Graduate=0):1
Enter whether the applicant is self employed or not(yes=1,no=0):0
Enter the applicant's income:4583
Enter co-applicant's income:1508
Enter the loan amount:128
Enter the loan amount term:360
Enter the credit history:1
Enter the property area of the applicant(urban=1, semi urban=0, rural=-1):-1

Out[60]: array(['Y'], dtype=object)

```

Comparing the Dependencies

Out of all the features, The feature 'Credit_History' is the most important and the prediction is more dependent on it.



Please visit my [GitHub](#) repository for the code execution part

References-

<https://towardsdatascience.com/predict-loan-eligibility-using-machine-learning-models-7a14ef904057>

<https://github.com/mridulrb/Predict-loan-eligibility-using-IBM-Watson-Studio>

Conclusion

This model is very useful in the banking sector for swift processing of loan eligibility for the applicants.