Statement of Work V2

AIDI 1002-02 AI Algorithms 1

Project Title: Loan Prediction Problem

Professor – Marcos Bittencourt

Email: - marcos.bittencourt@dcfaculty.mycampus.ca

Submitted by:

Biraj Prajapati (100766706)

Phase 01: -

1) Rational Statement: Loan Prediction Problem

2) Business Problem in Brief:
   Find out the person is eligible to acquire loan based on their qualifications, employment, earning, dependent, their dependent's income, credit history, their loan amount, and loan term. Create a machine learning model to generate loan approval from person's information.

3) Data Source: https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset

Loan_ID : Unique Loan ID

Gender : Male/ Female

Married : Applicant married (Y/N)

Dependents : Number of dependents

Education : Applicant Education (Graduate/ Under Graduate)

Self_Employed : Self employed (Y/N)

ApplicantIncome : Applicant income

CoapplicantIncome : Coapplicant income

LoanAmount : Loan amount in thousands of dollars

Loan_Amount_Term : Term of loan in months

Credit_History : credit history meets guidelines yes or no

Property_Area : Urban/ Semi Urban/ Rural

Loan_Status : Loan approved (Y/N) this is the target variable

Image 1 (Source: towardsdatascience)

4) Data Requirement:

Gender, married, dependents, education, self-employed, applicant income, co-applicant income, loan amount, loan term, credit history, property area and loan status.

```
<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
```

5) Data Assumption:

- Decision tree will have high accuracy than random forest, and logistic regression
- Loan status is extremely reliant on credit history.
- Linear relationship between columns.

6) Data Limitations and Constraints:

Dataset only got 613 entries in train dataset and 366 entries in test dataset. Dataset got some missing values in some columns, which need to be cleaned.

7) Test Process:

Data Cleaning → EDA → Feature Engineering → Preprocessing → Modeling → Model testing

Phase 02: -

a) EDA:

⇨ Table of training dataset

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

614 rows × 13 columns

⇨ Numerical data description of training dataset

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

⇨ Attributes type and their values count of training dataset

```
<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
```

b) Data Cleaning:

- Both training and test dataset contains missing values. I'll give explanation about how to resolve missing values in training dataset.
- Here is the screenshot of total missing values by attributes.

```
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
```

- I'll use mode function to add highly used values of that specific columns, but with 'LoanAmount' column using mode function will create an issue. That is they have more unique value and with less in bulk.

```
120.0    20
110.0    17
100.0    15
187.0    12
160.0    12
         ..
570.0     1
300.0     1
376.0     1
117.0     1
311.0     1
Name: LoanAmount, Length: 203, dtype: int64
```

- In this case I'll use median function to cover missing values for LoanAmount tribute.
- Below code is utilized to resolve null values.

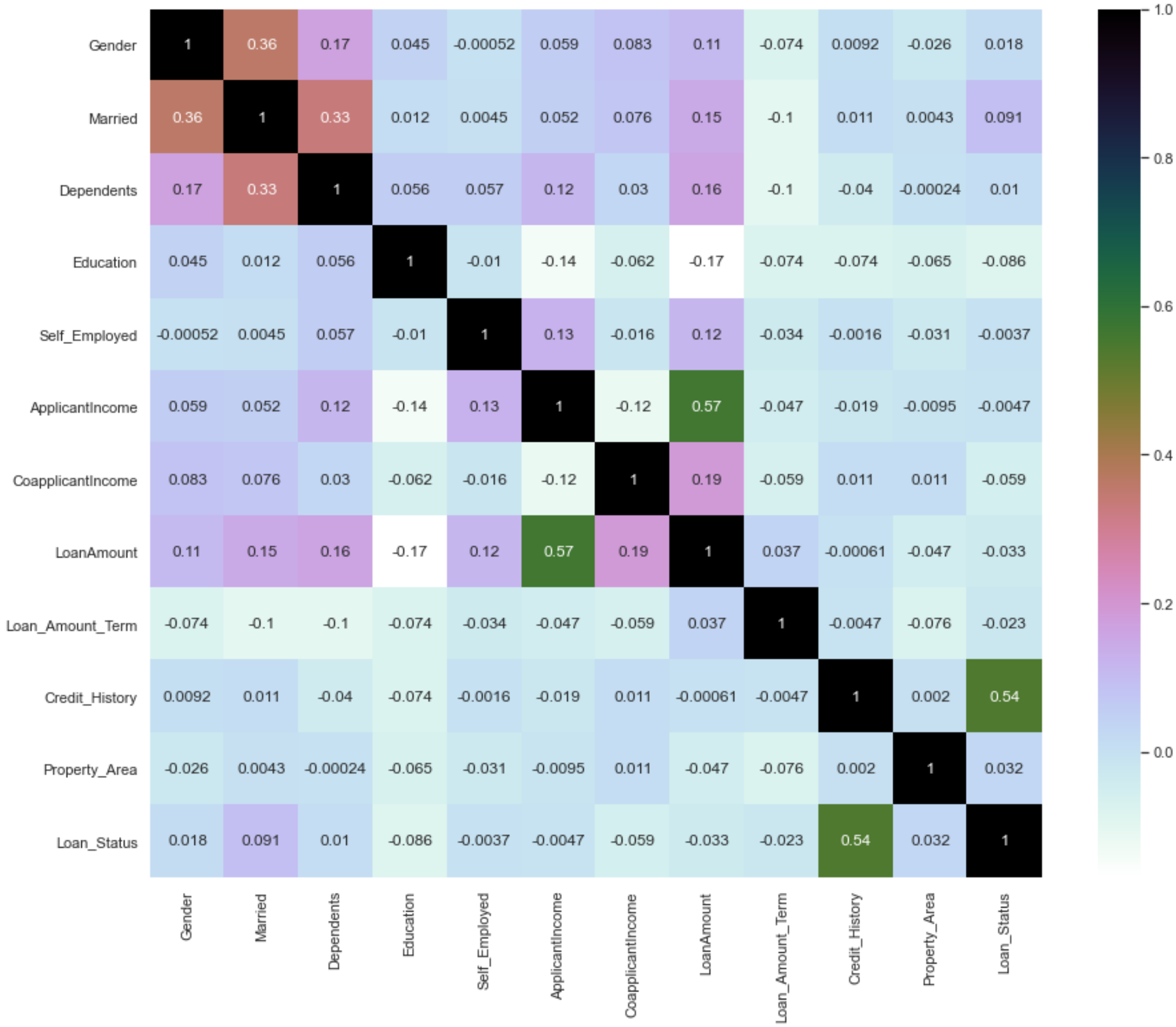
```
ds_loan['Gender'].fillna(ds_loan['Gender'].mode()[0], inplace = True)
ds_loan['Married'].fillna(ds_loan['Married'].mode()[0], inplace = True)
ds_loan['Dependents'].fillna(ds_loan['Dependents'].mode()[0], inplace = True)
ds_loan['Self_Employed'].fillna(ds_loan['Self_Employed'].mode()[0], inplace = True)
ds_loan['LoanAmount'].fillna(ds_loan['LoanAmount'].median(), inplace = True)
ds_loan['Loan_Amount_Term'].fillna(ds_loan['Loan_Amount_Term'].mode()[0], inplace = True)
ds_loan['Credit_History'].fillna(ds_loan['Credit_History'].mode()[0], inplace = True)
```

- The table output after cleaning the train dataset.

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

c) Correlation:

⇨ Below is the correlation of all columns after changing categorical data into numerical.



d) Benefits of Feature Engineering:

We can find useful variables to get good predictive model output. It is valuable to resolve missing values by imputed data cleaning and to solve data redundancy by filtering out the feature selection. We can also automate feature engineering by using algorithms.

References:

Image 1 source: https://towardsdatascience.com/ml-basics-loan-prediction-d695ba7f31f6

Dataset source: https://www.kaggle.com/altruistdelhite04/loan-prediction-problem