CERTIFICATE

This is to certify that Project entitled ***Loan Approval Prediction*** *being submitted by* ***Akash Dubey*** *bearing University Reg. No.* ***1712010014*** *respectively for the award of degree of Bachelor of Technology in Computer Science & Engineering is a record bona fide Project work carried out by them under my supervision. The matter embodied in this project has not been submitted for the award of any other degree anywhere.*

HOD, CSE

Project Guide

ACKNOWLEDGEMENT

Our sincere thanks to Dr. Shachi Mall, Professor, and Head of Department of Computer Science & Engineering, Gandhi Engineering College, Bhubaneswar for this encouragement and valuable suggestions during the period of our Project work.

No words would suffice to express my regards and gratitude to Binayak Panda, Department of Computer Science & Engineering, for his inspiring guidance and constant encouragement, immense support, and help during the project work.

Date: ***Akash (1712010014)***

Place:

CONTENT

ABSTRACT

For our project we have decided to experiment, design, and implement a loan prediction problem. We will use the loan prediction dataset to automate the loan eligibility process(real time) based on customer details provided like Education, Loan amount, Credit history, Income etc. Since we have to classify whether the loan will get approved or not so this is a Classification problem which comes under Supervised Machine Learning.

The dataset we will be working on has 615 rows & 13 columns. We will need to preprocess the data, perform data cleaning & feature engineering and finally will be implementing models like Logistic Regression, Decision tree, Random Forest and XGBoost to check the accuracy of each model.

CHAPTER-1

INTRODUCTION

1.1 Motivation and Objective

Machine Learning has become increasingly popular tool in almost every industry out there be it Smartphone Industry, Healthcare or Banks. With Machine Learning we can achieve so much. We will be using machine learning algorithms along with some data analysis techniques in our project.

Our project focus is to use existing customer's details and analyze it further by applying a few machine learning techniques and predict which future applicant can be approved the loan.

1.2 Problem Statement

Dream Housing Finance Company deals in all home loans. Its customers first apply for home loan after that company validates the customer eligibility for loan**.** Since the process is time taking and tedious, the company decided to automate the loan approval process using machine learning.

1.3 Machine Learning

Machine Learning is an idea to learn from examples and experience, without being explicitly programmed.

"A computer program is said to learn from experience E with some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." -Tom M. Mitchel

1.3.1 Examples of Machine Learning

There are many examples of machine learning. Here are a few examples of classification problems where the goal is to categorize objects into a fixed set of categories.

Face detection: Identify faces in images (or indicate if a face is present).

Email filtering: Classify emails into spam and not-spam.

Medical diagnosis: Diagnose a patient as a sufferer or non-sufferer of some disease.

Weather prediction: Predict, for instance, whether or not it will rain tomorrow.

1.3.2 Need of Machine Learning

Machine Learning is a field which is raised out of Artificial Intelligence (AI). Applying AI, we wanted to build better and intelligent machines. But except for few mere tasks such as finding the shortest path between point A and B, we were unable to program more complex and constantly evolving challenges. There was a realization that the only way to be able to achieve this task was to let machine learn from itself. This sounds similar to a child learning from its self. So machine learning was developed as a new capability for computers. And now machine learning is present in so many segments of technology, that we don't even realize it while using it.

Finding patterns in data on planet earth is possible only for human brains. The data being very massive, the time taken to compute is increased, and this is where Machine Learning comes into action, to help people with large data in minimum time.

If big data and cloud computing are gaining importance for their contributions, machine learning as technology helps analyze those big chunks of data, easing the task of data scientists in an automated process and gaining equal importance and recognition.

The techniques we use for data mining have been around for many years, but they were not effective as they did not have the competitive power to run the algorithms. If you run deep learning with access to better data, the output we get will lead to dramatic breakthroughs which is machine learning.

1.3.3 Kinds of Machine Learning

There are three kinds of Machine Learning Algorithms.

a. Supervised Learning

b. Unsupervised Learning

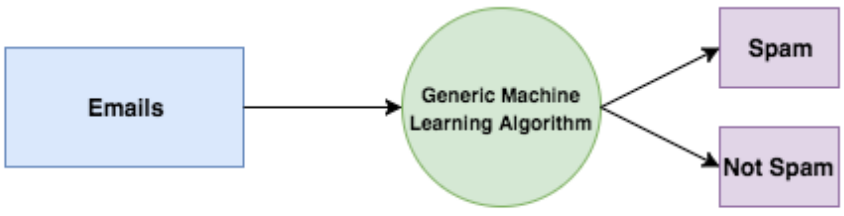
c. Reinforcement Learning

1.3.3.1 Supervised Learning

A majority of practical machine learning uses supervised learning.

In supervised learning, the system tries to learn from the previous examples that are given. (On the other hand, in unsupervised learning, the system attempts to find the patterns directly from the example given.)

Example:



Supervised learning problems can be further divided into two parts, namely classification, and regression.

Classification: A classification problem is when the output variable is a category or a group, such as “black” or “white” or “spam” and “no spam”.

Regression: A regression problem is when the output variable is a real value, such as “Rupees” or “height.”

1.3.3.2 Unsupervised Learning

In unsupervised learning, the algorithms are left to themselves to discover interesting structures in the data.

This is called unsupervised learning because unlike supervised learning above, there are no given correct answers and the machine itself finds the answers.

Unsupervised learning problems can be further divided into association and clustering problems.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as “people that buy X also tend to buy Y”.

Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

1.3.3.3 Reinforcement Learning

A computer program will interact with a dynamic environment in which it must perform a particular goal (such as playing a game with an opponent or driving a car). The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it continuously trains itself using trial and error method.

CHAPTER-2

PROCESS and SPECIFICATION REQUIREMENT

2.1 Process:

- Hypothesis Generation
- Understanding the Data
- Exploratory Data Analysis (EDA)
    - Univariate analysis
    - Bivariate analysis
- Missing Value and Outlier Treatment
- Feature Engineering
- Model Building
    - Logistic Regression
    - Decision Tree
    - Random Forest
    - XGBoost

2.2 Software Requirements:

- Python
- Anaconda (JUPYTER Notebook)

2.2.1 LIBRARIES USED:

- NumPy
- SciKit Learn
- Pandas
- MatplotLib
- Seaborn
- XGBoost

CHAPTER-3

METHODOLOGY

3.1 Hypothesis Generation

This is the first and foremost step which is performed even before looking at the data. It involves understanding the problem in detail by brainstorming as many factors as we can.

Below are some of the factors which I think can affect the Loan Approval (dependent variable for this loan prediction problem):

- **Salary**: Applicants with high income should have more chances of loan approval.
- **Previous history**: Applicants who have repaid their previous debts should have higher chances of loan approval.
- **Loan amount**: Loan approval should also depend on the loan amount. If the loan amount is less, chances of loan approval should be high.
- **Loan term**: Loan for less time period and less amount should have higher chances of approval.
- **EMI**: Lesser the amount to be paid monthly to repay the loan, higher the chances of loan approval.

3.2 Reading the data

Train file will be used for training the model, i.e. our model will learn from this file. It contains all the independent variables and the target variable.

Test file contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the test data.

```
train=pd.read_csv("train.csv")

test=pd.read_csv("test.csv")
```

3.3 Understanding the Data

Variable	Description
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

Loan_Amount_Term	Term of loan in months
Credit_History	Credit history meets guidelines
Property_Area	Urban/ Semi Urban/ Rural
Loan_Status	Loan approved (Y/N)

```
# Print data types for each variable
```

```
train. Types
```

```
Loan_ID                object
Gender                 object
Married                object
Dependents             object
Education              object
Self_Employed          object
ApplicantIncome         int64
CoapplicantIncome    float64
LoanAmount           float64
Loan_Amount_Term     float64
Credit_History       float64
Property_Area          object
Loan_Status            object
type: object
```

We can see there are three format of data types:

- object: Object format means variables are categorical. Categorical variables in our dataset are: Loan_ID, Gender, Married, Dependents, Education, Self_Employed, Property_Area, Loan_Status
- int64: It represents the integer variables. ApplicantIncome is of this format.
- float64: It represents the variable which have some decimal values involved. They are also numerical variables. Numerical variables in our dataset are: CoapplicantIncome, LoanAmount, Loan_Amount_Term, and Credit_History

3.4 Univariate Analysis

In this we examine each variable individual. For categorical features (Loan_ID, Gender, Married, Dependents etc.) frequency table or bar plots will be used which will calculate the number of each category in a particular variable. For numerical features, probability density plots can be used to look at the distribution of the variable.

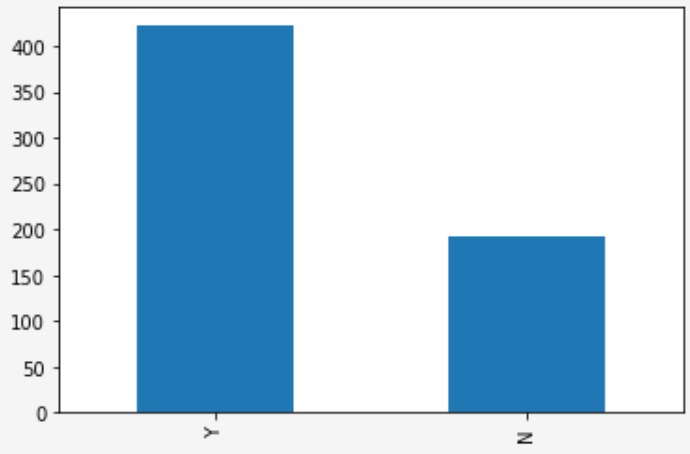
```
train['Loan Status'].value counts(normalize=True)
```

```
Y    0.687296

N    0.312704

Name: Loan_Status, dtype: float64
```

```
train['Loan_Status'].value_counts().plot.bar()
```



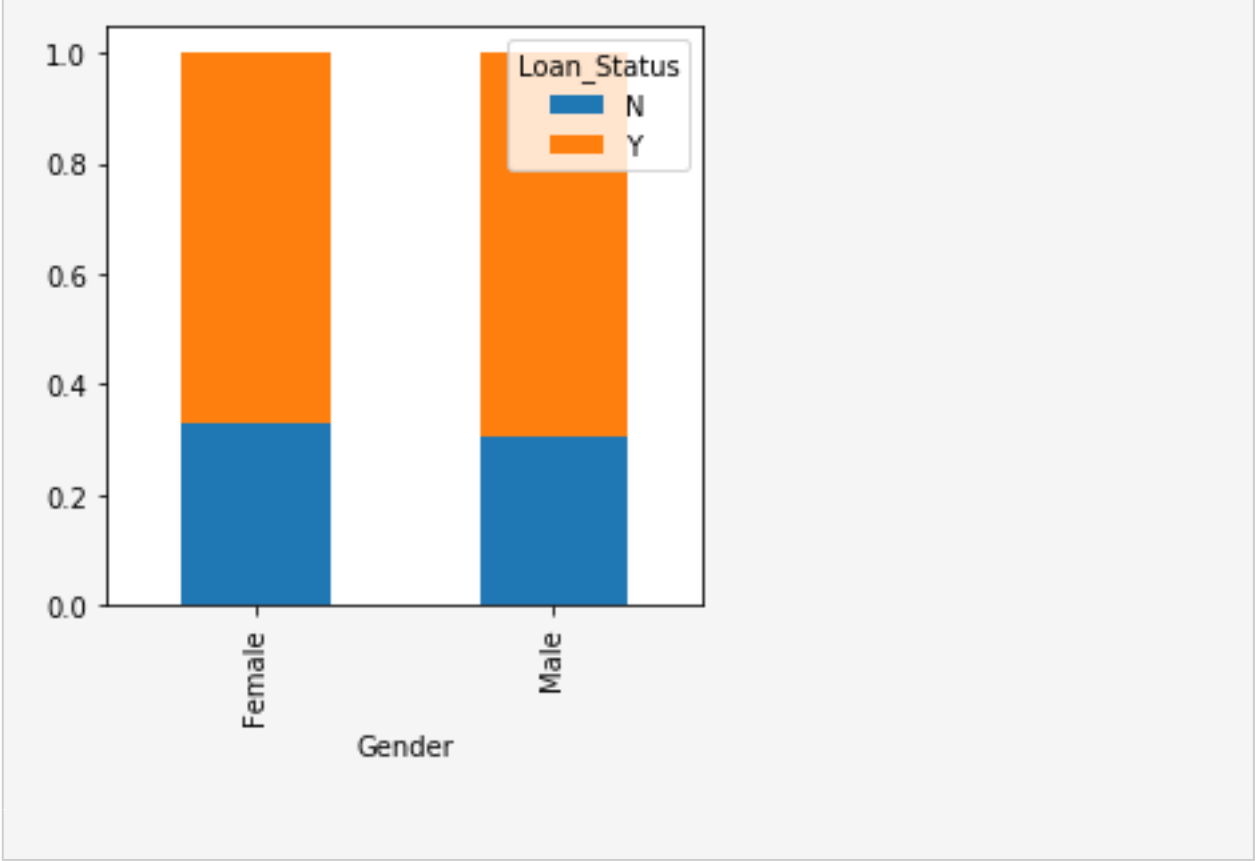
The loan of 422(around 69%) people out of 614 was approved.

3.5 Bivariate Analysis

After looking at every variable individually in univariate analysis, we will now explore them again with respect to the target variable.

```
Gender=pd.crosstab(train['Gender'],train['Loan Status'])
```

```
Gender.div(Gender.sum(1).astype(float), axis=0).plot(kind="bar",
stacked=True, figsize=(4,4))
```



It can be inferred that the proportion of male and female applicants is more or less same for both approved and unapproved loans.

3.6 Missing value and Outlier Treatment

3.6.1 Missing value Imputation

After exploring all the variables in our data, we can now impute the missing values and treat the outliers because missing data and outliers can have adverse effect on the model performance.

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

3.6.2 Outlier Treatment

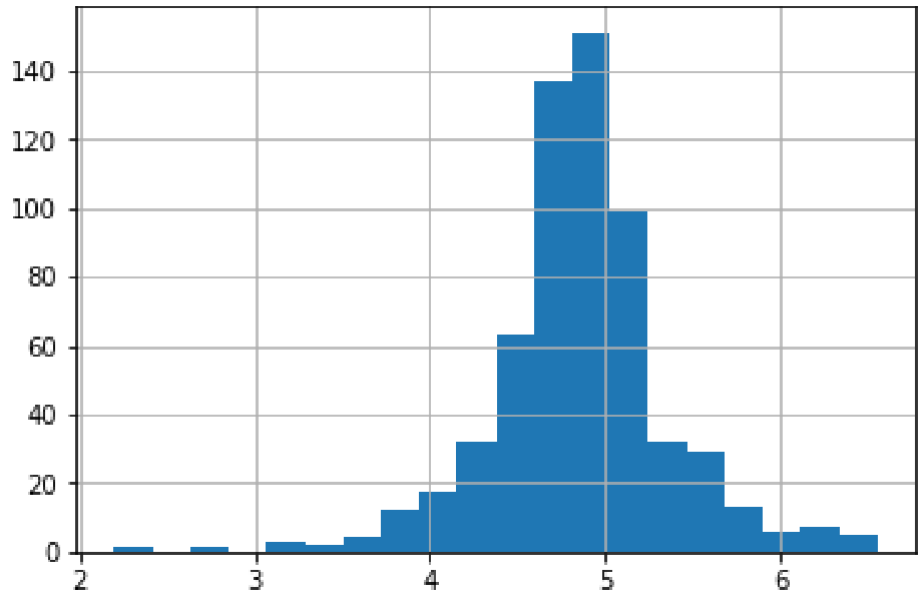
As we saw earlier in univariate analysis, LoanAmount contains outliers so we have to treat them as the presence of outliers affects the distribution of the data.

Due to these outliers bulk of the data in the loan amount is at the left and the right tail is longer. This is called right skew-ness. One way to remove the skew-ness is by doing the log transformation. As we take the log transformation, it does not affect the smaller values much, but reduces the larger values. So, we get a distribution similar to normal distribution.

```
train ['LoanAmount_log'] = np.log(train['LoanAmount'])
```

```
train['LoanAmount_log'].hist(bins=20)
```

```
test['LoanAmount_log'] = np.log(test['LoanAmount'])
```



Now the distribution looks much closer to normal and effect of extreme values has been significantly subsided.

3.7 Feature Engineering

Based on the domain knowledge, we can come up with new features that might affect the target variable. We will create the following three new features:

- **Total Income** - As discussed during bivariate analysis we will combine the Applicant Income and Coapplicant Income. If the total income is high, chances of loan approval might also be high.
- **EMI** - EMI is the monthly amount to be paid by the applicant to repay the loan. Idea behind making this variable is that people who have high EMI's might find it difficult to pay back the loan. We can calculate the EMI by taking the ratio of loan amount with respect to loan amount term.
- **Balance Income** - This is the income left after the EMI has been paid. Idea behind creating this variable is that if this value is high, the chances are high that a person will repay the loan and hence increasing the chances of loan approval.

```
train['Total_Income']=train['ApplicantIncome']+train['CoapplicantIncome']

test['Total_Income']=test['ApplicantIncome']+test['CoapplicantIncome']
```

We will add rest of features in similar way.

```
train['EMI']=train['LoanAmount']/train['Loan_Amount_Term']

test['EMI']=test['LoanAmount']/test['Loan_Amount_Term']

train['Balance Income']=train['Total_Income']-(train['EMI']*1000) # Multipl
y with 1000 to make the units equal

test['Balance Income']=test['Total_Income']-(test['EMI']*1000)
```

3.8 Model Building

After creating new features, we can continue the model building process. So we will start with logistic regression model and then move over to more complex models like Random Forest and XGBoost.

We will build the following models.

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost

Let’s prepare the data for feeding into the models.

```
X = train.drop('Loan Status',1)

y = train.Loan_Status      # Save target variable in separate dataset

#loan id don't have an effect on the outcome

train=train.drop('Loan_ID',axis=1)

test=test.drop('Loan_ID',axis=1)



"""Sklearn requires the target variable in a separate dataset.

so, we will drop our target variable from the train dataset and save it
in another dataset."""

x = train.drop('Loan_Status',1)

y = train.Loan_Status    # Save target variable in separate dataset



#As logistic regression takes only the numerical values as input, we ha
ve to change every categorical variable to continious
```

```
x=pd.get_dummies(x)

train=pd.get_dummies(train)

test=pd.get_dummies(test)



#we will use train_test_split function of sklearn to validate our predi
ctions

from sklearn.model_selection import train_test_split

x_train,x_cv,y_train,y_cv = train_test_split(x,y, test_size=0.3, random
_state=123)
```

In Train Test Split validation we split the training data into training and validating data as to evaluate our model. Since we don’ have the outcome for Test data so we can’t check its accuracy after predicting the outcome as we don’t have original outcomes to compare with.

Train Test Split is a validation technique used to solve the above issue and hence evaluate our model.

3.8.1 Logistic Regression

Here we build our model using the popular Logistic Regression. Logistic regression is an estimation of Logit function. Logit function is simply a log of odds in favor of the event.

```
from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score

LR=LogisticRegression()

LR.fit(x_train,y_train)




pred_cv=LR.predict(x_cv)
```

```
accuracy_score(y_cv,pred_cv)
```

```
0.7783783783783784
```

We got an accuracy score as 0.778 for this model.

3.8.2 Decision Tree

Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) based on most significant splitter / differentiator in input variables.

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable.

```
from sklearn import tree

DT = tree.DecisionTreeClassifier()

DT.fit(x_train, y_train)

pred_cv = DT.predict(x_cv)

accuracy_score(y_cv,pred_cv)
```

```
0.6702702702702703
```

We got an accuracy score of 0.6702 for this model.

3.8.3 Random Forest

Random Forest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) are combined to make a powerful prediction model.

For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model.

Final prediction can be a function of all the predictions made by the individual learners.

```
from sklearn.ensemble import RandomForestClassifier

RF = RandomForestClassifier()

RF.fit(x_train,y_train)

pred_cv=RF.predict(x_cv)

accuracy_score(y_cv,pred_cv)
```

```
0.7297297297297297
```

We got an accuracy score of 0.729 for this model.

3.8.4 XGBoost

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

```
from xgboost import XGBClassifier

XG= XGBClassifier()

XG.fit(x_train,y_train)

pred_cv=XG.predict(x_cv)

accuracy_score(y_cv,pred_cv)
```

```
0.7675675675675676
```

We got an accuracy score of 0.767 close to Logistic Regression.

We observe that Logistic Regression score is the highest and hence we will predict our Test data using Logistic Regression.

```
In [1360]: pred_test = LR.predict(test)

In [1361]: df=pd.DataFrame(pred_test)#converting the output to pandas dataframe

In [1364]: test.head(1)
```

Out[1364]:

	Dependents	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	LoanAmount_log	Total_Income	Total_Income_log	
0	0.0	6756.756757	0	33.783784	60.0	1.0	3.519981	6756.756757	8.818298	563.0

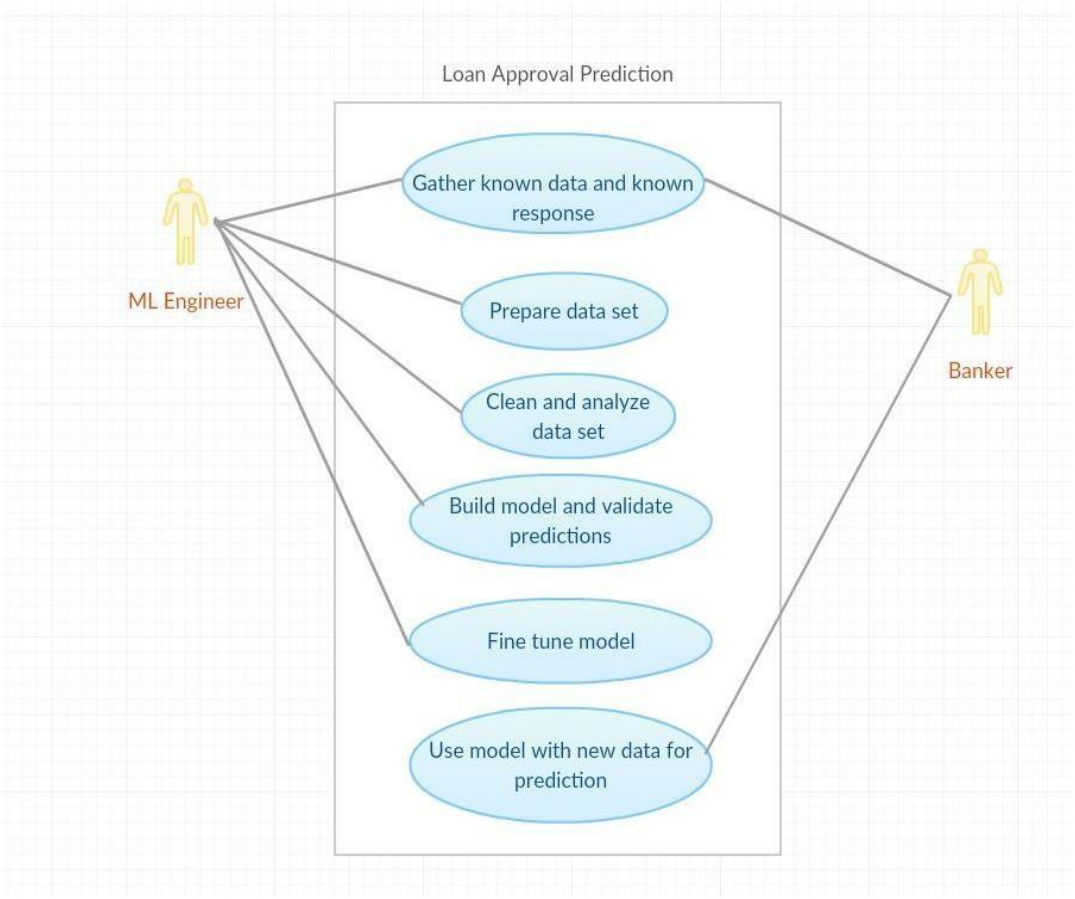
1 rows × 22 columns

```
In [1365]: df.head(1)
```

Out[1365]:

	0
0	1

3.9 Use Case Diagram



CHAPTER-4

CONCLUSION and FUTURE SCOPE

4.1 Conclusion

1. Out of all the classification algorithms used on the dataset, **Logistic Regression** algorithm gives the best overall prediction accuracy.
2. **Credit History, Balance Income, EMI, Property Area** are the most important factors for predicting the class of the loan applicant (whether the applicant would be‘approved’ or ‘not’).

3. In near future this module of prediction can be integrated with the module of automated processing system. The system is trained on old training dataset, in future software can be made such that new testing data can be used after certain time

4. We can train the XGBoost model using grid search to optimize its hyper parameters and improve the accuracy.

4.2 Future Scope

- Time Series Analysis can be done using the Loan data of several years, for prediction of the approximate time, when the client can default.
- Future analysis can be done on predicting the approximate Interest rates that the loan applicant is expected to be charged as per his profile, if his loan is approved. This can be useful for loan applicants, since some banks approve loans, but give very high interest rates to the customer.
- An app with proper UI can be built, which will take various inputs from the user like name, address, loan amount, loan duration, etc. and give a prediction of whether their loan application can be approved by the banks or not based on their inputs along with an approximate interest rates.

BIBLIOGRAPHY

1. Machine Learning Courses

1.2 Coursera(Andrew NG)

1.3 YouTube(SimpliLearn, Edureka)

2. Machine Learning Repositories

2.2 UCI

2.3 Kaggle

3. Websites

3.2 www.kdnuggets.com

3.3 www.analyticsvidhya.com

3.4 www.machinelearningmastery.com