

## **INTRODUCTION**

Loan Distribution is the main business part of many banks. The main portion of banks income comes from the loan distributed to customers. These banks apply interest on loan which are distributed to customers.

The main objective of banks is to invest their assets in safe customers. Up to now many banks are processing loans after a process of verification and validation. But till now no bank can give surety that the customer who is chosen for loan application is safe or not. So to avoid this situation we introduced a system for the approval of bank loans known as Loan Prediction System Using Python.

Loan Prediction System is a software which checks the eligibility of a particular customer who is capable of paying loan or not. This system checks various parameters such as customer's marital status, income, expenditure and various factors. This process is applied for many customers of trained data set. By considering these factors a required model is built. This model is applied on the test data set for getting required output. The output generated will be in the form of yes or no. Yes indicates that a particular customer is capable of paying loan and no indicates that the particular customer is not capable of paying loan. Based on these factors we can approve loans for customers.

## **1.1 OVERVIEW**

### **Data Analysis for prediction of loan-based nature of clients**

The report main intention is to classify the nature of clients for loans. Depending upon the certain factors the report classifies the customers. Classification is done through exploratory data analyses [1].

Exploratory data analysis is a technique that analyzes and summaries the main features from training dataset.

### **Prediction of Loan Approval using Machine Learning Approach**

Machine learning [2] is a phenomenon in which analytical model is build from the trained model.

This model is applied on test data for providing of the accurate results.

Here the author used three algorithms for prediction of loan. They are

1. K Nearest Neighbor
2. Decision Tree
3. Random Forests

The main purpose of this report is to provide immediate and accurate results for the approval of loan to the eligible customers. In banking sector there will be n number of people who apply loans. It is difficult to check customer's eligibility through paper work. The system can provide accurate results for the n number of people.

### **Building the model using Random Forest Approach**

In this report we have discussed about credit risk and credit analysis. Banking sectors success mainly depends of credit risk analysis. In this report we have used Random Forest [3] approach to build the model. The use of Random Forest is because Random Forest Approach provides accurate results than the K Nearest Neighbor and Decision Tree.

## **Ensemble model survey for loan prediction**

In this report author has used Random Forest approach for building a model. In this report two or more classifiers are combined together and identify a perfect model for loan prediction.

Ensemble method compares two or more models and identifies a perfect model from two or more models for better loan prediction which makes banking sector to make a right choice for approval of loan application.

## **1.2 PURPOSE**

### **Existing System**

Till now loans are processed by various banks through pen and paperwork. When the large no of customers apply for bank loan these bank take lot of time to approve their loan. After approval of loan by the banks, there is no surety that the chosen applicant is capable of paying loan or not. Many banks use their own software's for the loan approval. In existing system we use datamining algorithms for the loan approval; this is the old technique for the approval of loan. Multiple data sets are combined and form a Generalized datasets, and different machine learning algorithms are applied to generate results. But these techniques are not up to the mark. Due to this huge banks are suffering from financial crises. To resolve this issue we introduce a new way for approval of loans.

### **Proposed System**

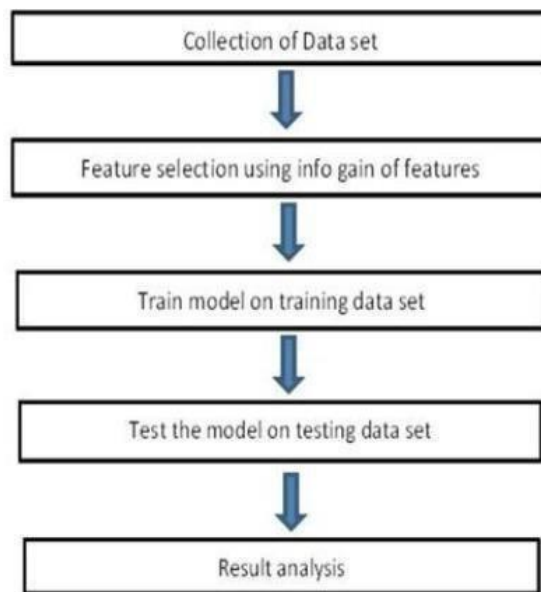
Loan Approval System is software used for approval of loan in banking sector. In this proposed system we have used machine learning algorithm. Machine Learning is process in which a symmetric model is build from the existing dataset; this model is applied for the testing of the new dataset. The system consists of trained dataset and test dataset. The trained dataset is used for construction of model. This model is applied on testing dataset for the required result. We have used Ensemble approach for building of the model.

Random forest algorithm uses this ensemble approach and builds a model from the existing training dataset.

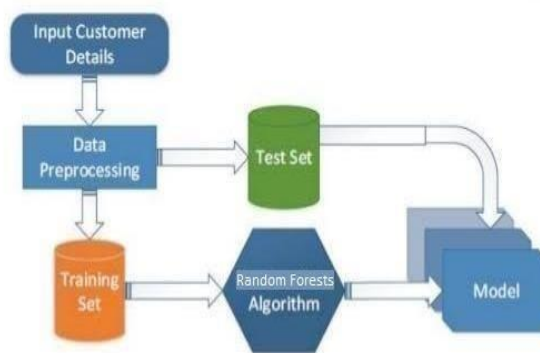
# PROBLEM DEFINITION AND DESIGN THINKING

## 2.1 EMPATHY MAP

We have used ensemble learning for building of the system.



## 2.2 IDEATION MAP



Architecture of Proposed Model

## RESULT

To protect system from unauthorized access, we have created admin login module for security purpose. It consists of username and password. By providing the valid username and password we can access the system.

The admin home consists of two options:

- Loan Approval Prediction
- Classifier performance

Loan Approval Prediction is chosen for prediction of loan status whereas classifier performance gives the prediction results of three algorithms:

- K Nearest Neighbour
- Decision Tree
- Random Forest


The train data consists of various attributes such as salary, marital status, loan accounts, and loan repayments on time etc. According to these factors we build a required model for loan prediction.

The test data consist of various attributes such as salary, marital status, loan accounts except the loan approval status. The loan approval status is obtained. When we deploy the test data to the model which is built from the trained data.

The loan status is obtained after the deployment of test data to the model which is built from the trained data using Random Forest Algorithm. The loan status consists of Customer id and loan status. It indicates for a particular customer loan is approved or not. If loan status is Y (Yes) then the customer is eligible for approval of loan and if it is N (No) then the


customer is not eligible for approval of loan.






### Data Analysis

- Size: 400 Kb
- Shape: 613 Rows & 13 Columns with 1 column as Target
- Data Source: DPhi
- Platform: Jupyter & Google Colab



### Data Visualization

- Matplotlib & Seaborn to visualize.
- SweetViz for reporting.



### Default Prediction

- Get your loan application approval chances by providing few necessary informations.

Predict

The director of SZE bank identified that going through the loan applications to filter the people who can be granted loans or need to be rejected is a tedious and time-

Loan Application



Visualization

## Get Knowledge about Loan Dataset

- Select the report to show the EDA through the whole Data set
- Select JoinReport to show the EDA through Train Data set

Whole Data set Report

Train Data Report



Few Insights.

## Using Machine Learning Model : Logistic Regression

AI System tries to predict the people who can be granted loans or need to be rejected

- We see that the most correlate variables are:
  - ApplicantIncome with LoanAmount
  - Credit\_History with Loan\_Status.
  - LoanAmount is also correlated with CoapplicantIncome

## Feature Engineering and using models LR, Decision Tree, Random Forest

After using Feature engineering with the introduction of some new features like

- Total Income:** As discussed during bivariate analysis we will



[Home](#) >> Predict Loan Application Approval

Please Fill in the details to get your chances of Loan Approval.

|   |  |
|---|--|
| Gender  | Applicant Income: The amount of income the applicant earns:  |
| Marital status of the applicant:                                | Co-applicant Income: The amount of income the co-applicant earns:                                  |
| Education level of the applicant:                               | Loan Amount (\$5 to \$1,000):  |
| Dependents: No. of people dependent on the applicant (0,1,2,3+) | Loan Amount Term: The no. of days over which the loan will be paid (in Days)                       |
| Self-employed (Yes, No)   | Credit History of a borrower's responsible repayment of debts (1- has all debts paid, 0- not paid) |
|   | Property_Area  |

MAKE PREDICTION



MAKE PREDICTION

**Applicant Income** : 250000.0  
**Co-Applicant Income** : 500000.0  
**Loan Amount** : 100000.0  
**Loan Amount Term** : 25000  
**Credit History** : 0  
**Gender** : MALE  
**Marital Status** : NO  
**Education Level** : GRADUATED  
**No of Dependents** : 1  
**Self Employment** : YES  
**Property Area** : RURAL

Loan Decision:

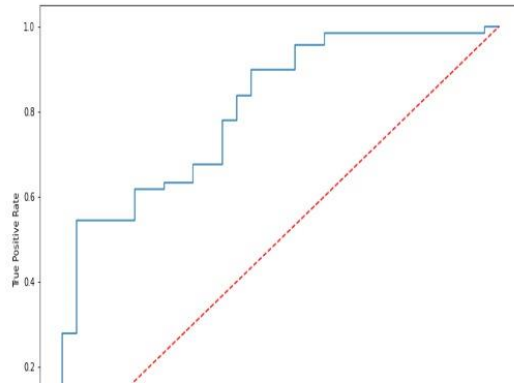
😞😞 Unfortunately your Loan Application has been Denied 😞😞



## FI SCORE ROC AUC

### ROC AUC Score

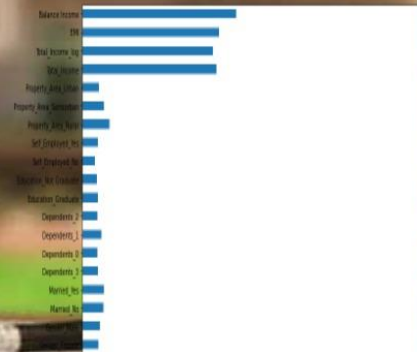
The following graph shows the ROC AUC Curve for Logistic Regression Machine Learning Model. It gives the best AUC.



## Features

### Random Forest Feature Importance After Feature Engineering

The following graph gives the feature importance to predict the Loan Application's approval.



## **ADVANTAGES AND DISADVANTAGES**

### **4.1 ADVANTAGES**

- \* Performance and accuracy of the algorithms can be calculated and compared.
- \* Class imbalance can be dealt with machine learning approaches
- \* One of the primary benefits of using machine learning for credit scoring is its accuracy.

Unlike human manual processing, ML-based models are automated and less likely to make mistakes. This means that loan processing becomes not only faster but more accurate, too, cutting costs on the whole

### **4.2 DISADVANTAGES**

- \* They had proposed a mathematical model and machine learning algorithms were not used.
- \* Class Imbalance problem was not addressed and the proper measure were not taken.

## **APPLICATIONS**

### **5.1 APPLICATIONS**

- This system can employed in all the mobile phones.
- Used in laptop.
- Used in Personal Computer.
- Used in emails.

## **CONCLUSION**

### **6.1 CONCLUSION**

From the proper view of analysis this system can be used perfect for detection of clients who are eligible for approval of loan. The software is working perfect and can be used for all banking requirements. This system can be easily uploaded in any operating system. Since the technology is moving towards online, this system has more scope for the upcoming days. This system is more secure and reliable. Since we have used Random Forest Algorithm the system returns very accurate results. There is no issue if there are many no. of customers applying for loan. This system accepts data for N no. of customers. In future we can add more algorithms to this system for getting more accurate results.

## **FUTURE SCOPE**

### **7.1 FUTURE SCOPE**

1. The current system only provides personal loan facilities to customers, but in future our loan prediction model can be created for predicting the eligibility of different types of loans like home, education, health, travel, property.
2. Similarly in our current system, the data of the customer is predicted based on the attributes that are mentioned in data set and it is stored in our database. But in future we can use the same data from the database to train and predict the output with better accuracy.
3. This model can be used to compare various machine learning algorithm generated prediction models and the model which will give higher accuracy will be chosen as the prediction mode

## APPENDIX

### 8.1 SOURCE CODE

```
import flask
```

```
import pickle
```

```
import pandas as pd
```

```
import numpy as np
```

```
from sklearn.preprocessing import StandardScaler
```

```
#load models at top of app to load into memory only one time
```

```
with open('models/loan_application_model_lr.pickle', 'rb') as f:
```

```
    clf_lr = pickle.load(f)
```

```
# with open('models/knn_regression.pkl', 'rb') as f:
```

```
#    knn = pickle.load(f)
```

```
ss = StandardScaler()
```

```
genders_to_int = {'MALE':1,
```

```
                  'FEMALE':0}
```

```
married_to_int = {'YES':1,
```

```
                  'NO':0}
```

```
education_to_int = {'GRADUATED':1,  
                    'NOT GRADUATED':0}
```

```
dependents_to_int = {'0':0,  
                     '1':1,  
                     '2':2,  
                     '3+':3}
```

```
self_employment_to_int = {'YES':1,  
                           'NO':0}
```

```
property_area_to_int = {'RURAL':0,  
                         'SEMIRURAL':1,  
                         'URBAN':2}
```

```
app = flask.Flask(__name__, template_folder='templates')
```

```
@app.route('/')  
  
def main():
```

```
    return (flask.render_template('index.html'))
```

```
@app.route('/report')
```

```
def report():
```

```
    return (flask.render_template('report.html'))
```

```
@app.route('/jointreport')
```

```
def jointreport():
```

```
    return (flask.render_template('jointreport.html'))
```

```
@app.route('/Loan_Application', methods=['GET', 'POST'])
```

```
def Loan_Application():
```

```
    if flask.request.method == 'GET':
```

```
        return (flask.render_template('Loan_Application.html'))
```

```
    if flask.request.method == 'POST':
```

```
        #get input
```

```
        #gender as string
```

```
        genders_type = flask.request.form['genders_type']
```

```
        #marriage status as boolean YES: 1 , NO: 0
```

```
        marital_status = flask.request.form['marital_status']
```

```
        #Dependents: No. of people dependent on the applicant (0,1,2,3+)
```

```
        dependents = flask.request.form['dependents']
```

```
        #dependents = dependents_to_int[dependents.upper()]
```

```
        #education status as boolean Graduated, Not graduated.
```

```
        education_status = flask.request.form['education_status']
```

```
        #Self_Employed: If the applicant is self-employed or not (Yes, No)
```

```
        self_employment = flask.request.form['self_employment']
```

```
        #Applicant Income
```

```
        applicantIncome = float(flask.request.form['applicantIncome'])
```

```
        #Co-Applicant Income
```

```
        coapplicantIncome = float(flask.request.form['coapplicantIncome'])
```

```
        #loan amount as integer
```

```
        loan_amnt = float(flask.request.form['loan_amnt'])
```



```
#term as integer: from 10 to 365 days...  
term_d = int(flask.request.form['term_d'])  
# credit_history  
credit_history = int(flask.request.form['credit_history'])  
# property are  
property_area = flask.request.form['property_area']  
#property_area = property_area_to_int[property_area.upper()]
```

```
#create original output dict  
output_dict= dict()  
output_dict['Applicant Income'] = applicantIncome  
output_dict['Co-Applicant Income'] = coapplicantIncome  
output_dict['Loan Amount'] = loan_amnt  
output_dict['Loan Amount Term']=term_d  
output_dict['Credit History'] = credit_history  
output_dict['Gender'] = genders_type  
output_dict['Marital Status'] = marital_status  
output_dict['Education Level'] = education_status  
output_dict['No of Dependents'] = dependents  
output_dict['Self Employment'] = self_employment  
output_dict['Property Area'] = property_area
```

```
x = np.zeros(21)
```

```
x[0] = applicantIncome  
x[1] = coapplicantIncome  
x[2] = loan_amnt  
x[3] = term_d
```

```
x[4] = credit_history
```

```
print('-----this is array data to predict-----')
```

```
print('X = '+str(x))
```

```
print('-----')
```

```
pred = clf_lr.predict([x])[0]
```

```
if pred==1:
```

```
    res = '🎉🎉Congratulations! your Loan Application has been
```

```
Approved! 🎉🎉'
```

```
else:
```

```
    res = '😞😞Unfortunately your Loan Application has been Denied 😞😞'
```

```
#render form again and add prediction
```

```
return flask.render_template('Loan_Application.html',
```

```
    original_input=output_dict,
```

```
    result=res,)
```

```
# temp = pd.DataFrame(index=[1])
```

```
# temp['genders_type'] = genders_to_int[genders_type.upper()]
```

```
# #marriage status as boolean YES: 1 , NO: 0
```

```

# temp['marital_status'] = married_to_int[marital_status.upper()]
# #Dependents: No. of people dependent on the applicant (0,1,2,3+)
# temp['dependents'] = dependents_to_int[dependents.upper()]
# #education status as boolean Graduated, Not graduated.
# temp['education_status'] = education_to_int[education_status.upper()]
# #Self_Employed: If the applicant is self-employed or not (Yes, No)
# temp['self_employment'] = self_employment_to_int[self_employment.upper()]
# #Applicant Income
# temp['applicantIncome'] = applicantIncome
# #Co-Applicant Income
# temp['coapplicantIncome'] = coapplicantIncome
# #loan amount as integer
# temp['loan_amnt'] = loan_amnt
# #term as integer: from 10 to 365 days...
# temp['term_d'] = term_d
# # credit_history
# temp['credit_history'] = credit_history
# # property are
# temp['property_area'] = property_area_to_int[property_area.upper()]

# temp['loan_amnt_log']=np.log(temp['loan_amnt'])

# Feature Engineering :
#temp['Total_Income']= temp['applicantIncome']+temp['coapplicantIncome']
#temp['Total_Income_log'] = np.log(temp['Total_Income'])
#temp['EMI']= temp['loan_amnt']/temp['term_d']
#temp['Balance Income'] = temp['Total_Income']-(temp['EMI']*1000)

# Columns to drop and afterward Predict up on the feature engineered columns
#temp = temp.drop(['applicantIncome', 'coapplicantIncome', 'loan_amnt',

```

```
'term_d'], axis=1)
```

*# Credit\_History is the most important feature followed by Balance Income, Total Income, EMI.*

*# So, feature engineering helped us in predicting our target variable.*

*# #make prediction*

```
if __name__ == '__main__':  
    app.run(debug=True)
```