Loan Approval Prediction Using Machine Learning

Team 6 - Titans

Jennifer Nghi Nguyen - 010872316

Ayush Wattal - 015245490

Mohini Patil - 015359188

Ritu Bhamrah - 015324478

Department of Applied Data Science, San Jose State University

DATA 245 – Machine Learning Technologies

Dr. Shih Yu Chang

December 4, 2021

Abstract

The asset cost increase requires high capital for which people need loans from banks. The various loans offered by banks are house loans, education loans, personal loans, etc. The loaning process is crucial for banks as they generate money in the form of interest. If people do not pay back the principal amount, banks can face major losses. Thus, it is very important to understand the payback capacity of a loanee. This project focuses on using machine learning classification techniques to solve the loan approval prediction problem. The banking data is sensitive data which makes it hard to get the real-time data of bank customers. For this project, the data had been collected from the online resource Kaggle which is labeled data. Data cleaning and exploration has been performed before applying the machine learning techniques - Logistic regression, Decision tree, AdaBoost classifier, and Gradient boosting classifier. For validation, Stratified k fold is followed after hyper-optimization of parameters. The model which outperformed all the other models in all the evaluation metrics after Stratified K- Fold Validation is a Tuned Logistic Regression model with an overall accuracy of 81.11 percent. The highest AUC provided by the logistic regression technique is 0.737 and it also performed the best in confusion metrics with less False positives and overall better in True negative and True Positive. In the future, the model results could be leveraged by banks and customers using the interface to predict loan approval based on input data.

Table of Contents

Presentation Link: https://www.youtube.com/watch?v=rq9Wplk5Acg

1. Problem Definition

1.1 Project Background and Target Problem

According to the Federal Reserve, American household debt has reached $14.6 trillion in the spring of 2021, and from a 2021 report from CNBC, the average American has $90,460 in debt. Therefore, we can clearly see that loans play a big part in our lives and our economy. However, there are always risks associated with default loans. One example is the great recession of 2007 to 2009 which is known as the second-worst economic crisis in US history. The main culprit for the recession is the collapse of the mortgage market due to default high-risk housing loans which were granted to those who previously weren't qualified which resulted in floods of new homebuyers and then led to the bursting of the 2000s housing bubble. Consequently, it was extremely hard for loan borrowers to pay back due to combinations of high interests and falling home prices. Therefore, in 2007, the largest mortgage provider, New Century Financial declared bankruptcy and the subprime mortgage crisis had started.

With that in mind, our project aims to set up a foundation prediction model which minimizes the risks of defaulting loans on those who may have difficulty paying back in the future.

1.2 Literature Survey

Paper 1: Fraud prediction in bank loan administration using decision tree

Authors: I. O. Eweoya, A. A. Adebiyi, A. A. Azeta and Angela E Azeta.

Goals and Dataset

The research aims to implement a machine learning model to predict fraud using the decision tree method. The dataset in this paper is a credit dataset with 5000 records with 9

attributes of gender, employment status, income, tracking of last three payments, the balance of loan taken.

Models and Evaluation

The model is evaluated using accuracy score, confusion matrix, AUC, and ROC curves.

Results

The results show that the model has an accuracy of 75.9% with a high true positive ratio and an AUC = 0.63.

Paper 2: Loan Prediction Using Logistic Regression in Machine Learning

Authors: S. Sreesouthry, A. Ayubkhan, M. Mohamed Rizwan, D. Lokesh, K. Prithivi Raj.

Goals and Dataset

This paper aims to develop a model for a credit scoring system using a machine learning classification model. The dataset includes descriptive data about clients including gender, dependents, incomes, loan amount, etc.

Models and Evaluation

The chosen model for this paper is logistic regression combined with min-max normalization. The model is evaluated using an accuracy and confusion matrix.

Results

The results show that the model archives an accuracy of 77% and the confusion matrix indicates that those with the lowest credit score will be rejected a loan, those with low income and the low loan amount will be likely to be granted.

Paper 3: Predicting Bank Loan Default with Extreme Gradient Boosting

Authors: Rising Odegua

Goals and Dataset

The author aimed to implement an effective foundation for identifying risky clients from a large set of loan applicants using predictive modeling.

Three datasets were used for this paper:

- Demographic data: loan applicant's descriptions and details.
- Performance data: further details and a flag indicates if an applicant is a good or bad debtor.
- Previous loan data: all previous loan applications.

Models and Evaluation

Only the Extreme Gradient Boosting (XGboost) model is applied in this paper, and the author used accuracy, recall, precision F1-score, and ROC as metrics for evaluation

Results

The final result shows that the XG boost achieved 79% of accuracy, 97% of precision, 79% of recall, and 87% of F1 score. Finally, the author concluded that XGBoost was successfully applied for defaulting loan prediction and identifying risky clients from a large pool of loan applicants.

Paper 4: Prediction of loan risk using naive Bayes and support vector machine

Authors: Vimala S.1, Sharmili K.C

Goals and Dataset

The authors discussed the prediction of loan risk models using a combined model of Naive Bayes and support vector machine techniques.

Models and Evaluation

The authors use Naive Bayes, support vector machine, and a combined model of Naive Baes and support vector machine. The combined model works as followed:

- Step 1: Calculate probabilities of each features' distinct value given class values using Naive Bayes. Then, from this, build a Naive Bayes model
- Step 2: Predict the mode in step 1 with the dataset
- Step 3: Apply prediction results to the support vector machine
- Step 4: Find the prediction values from the model in the previous step with the original dataset

The authors used accuracy scores and execution times as measures of evaluation

Results

Naive Bayes has the lowest accuracy score and takes the most time to execute. Support vector machine model and the combined model have similar accuracy scores, but the combined model has the fastest execution time.

Paper 5: Credit scoring analysis using weighted k nearest neighbor

Authors: M A Mukid, T Widiharih, A Rusgiyono, A Prahutama.

Goals and Dataset

This paper examines methods for the credit scoring system which evaluate the credit risk of loan applicants. The authors consider the weighted k nearest neighbor (WKNN) model with different kernels for this problem. The dataset used in this paper is from an Indonesian bank consisting of 948 clients and 184 of them are marked as bad customers. Variables in this dataset are age, working experience, total income, other loans, net income, interaction with the bank, saving, and debt ratio.

Models and Evaluation

This paper evaluates the performances of the k nearest neighbor model and its modified version. The modified version works as follow:

- Step 1: Find k+1 nearest neighbors to x based on distance.
- Step 2: Standardize k smallest distance.
- Step 3: Transform results from step 2 with any kernel function into weights.
- Step 4: Predict the class output by a weighted majority of the k nearest neighbors.

Some considered kernels are Epanechnikov, biweight, triweight, gauss, inverse.

The models are evaluated by sensitivity, specificity, and accuracy measures from true positive, true negative, false positive, and false negative.

Results

The final evaluation shows that the Gaussian kernel and rectangular kernel archives better performance with 82.4% of accuracy for both.

2. Project Objectives and Management

The project objectives and management section include project objectives, scope, methodology, project resources, and dataset details.

2.1 Project Objective

This project aims to predict loan approval based on the client's data such as gender, credit history, the applicant and co-applicant income, dependents, loan amounts, loan periods, etc. We will apply multiple classification machine learning models to solve this problem. Then, we will validate and evaluate chosen models and pick the one with the best performance to be the final prediction model. Following classification, machine learning modes are chosen for the problem: decision trees, logistic regression, AdaBoost, and gradient boosting. Users of interest for this project will be Banks, Common Public (Loan requester).

2.2 Project Scope

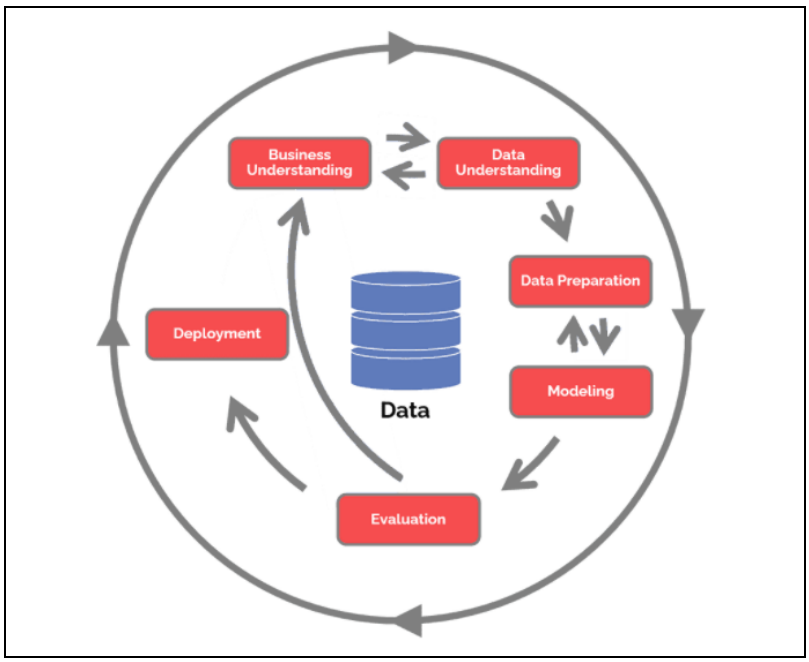
- Apply CRISP-DM methodology for a machine learning project
- Perform data exploration with multiple EDA tools
- Apply and tune various machine learning models
- Validate and evaluate model's performances

2.3 Methodology

The methodology used for this project is CRISP-DM. Figure 1 depicts the main phases of this project. Note that, deployment phase will be excluded from this project.

Figure 1

CRISP- DM



Note. The figure shows the CRISP-DM. This figure is taken from https://www.datascience-pm.com/crisp-dm-2/

Table 1

CRISP- DM activities

Phase	Activities
Business Understanding	Learn about problems, identify objectives, conduct literature surveys on available technologies and solutions, collect original datasets.
Data Understanding	Perform data exploration with multiple EDA tools, survey data problems.
Data Preparation	Handle data problems, perform feature selection and data transformation for modeling
Modeling	Apply and tune chosen models
Evaluation	Validate and evaluate all models. Compare and justify for the final best model.

Note. Deployment is excluded since it does not belong in the scope of the project.

2.4 Project Resource Requirements & Plan

The tools used for this project are as below:

- **Zoom calls** - This is used for project discussions, roadblocks, etc.
- **Google collab** - This is used to have shared access to Python code.
- **Google sheets** - This is used to maintain the tasks assigned to different team members.
- **Google Docs** - This is used to have shared reporting work access

- **Lucid chart -** This is used to create the flow charts and architecture diagrams for the project.

2.5 Dataset

We will use the Loan Prediction Problem Dataset from Kaggle (Chatterjee, 2019). https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset/metadata

3. Multidisciplinary Analysis

The multidisciplinary analysis will cover the exploration done on data, cleaning performed on the dataset, different models proposed with their justification, and the platform used for these models.

3.1 Exploration

The dataset used for this project has 614 rows and 13 columns.

Table 2

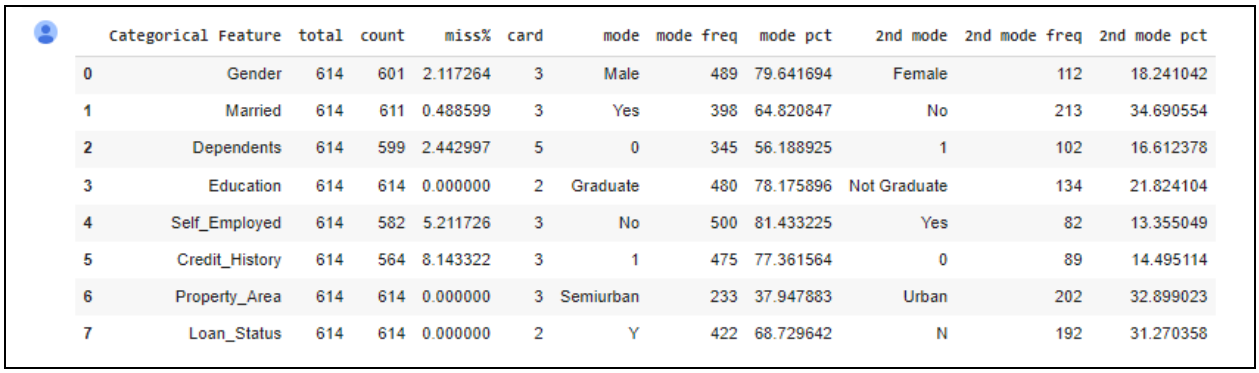
Different Parameters, Data Type, and Description

Parameter	**Data Type**	**Description**
Loan_ID	String	Unique loan id for the user
Gender	String	Male/Female
Married	String	Yes/No
Dependents	Float	Number of people dependent on the applicant
Education	String	Graduate/Not Graduate
Self_Employed	String	Yes/No
ApplicantIncome	Integer	Applicant's income
CoapplicantIncome	Float	Co-applicant income
LoanAmount	Float	Amount of the loan being applied for by the user
Loan_Amount_Term	Float	Loan amount term in months
Credit_History	Float	0/1 whether meets the credit history guidelines
Property_Area	String	Urban/Semi-urban/Rural
Loan_Status	String	Yes/No Loan approved (Target feature)

Note. The figure shows the various parameters for the dataset like loan_id, gender, married, dependents, etc. with their data types.

Figure 2

Categorical Data Quality Report Before Cleaning

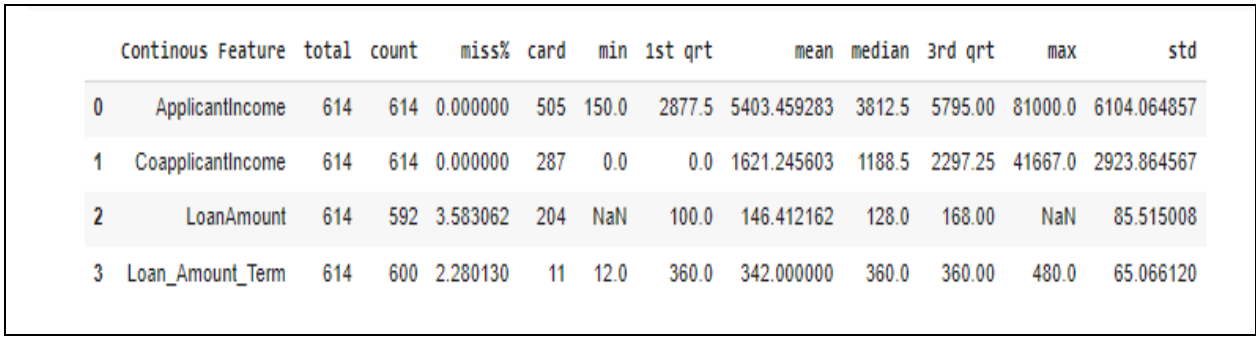


	Categorical Feature	total	count	miss%	card	mode	mode freq	mode pct	2nd mode	2nd mode freq	2nd mode pct
0	Gender	614	601	2.117264	3	Male	489	79.641694	Female	112	18.241042
1	Married	614	611	0.488599	3	Yes	398	64.820847	No	213	34.690554
2	Dependents	614	599	2.442997	5	0	345	56.188925	1	102	16.612378
3	Education	614	614	0.000000	2	Graduate	480	78.175896	Not Graduate	134	21.824104
4	Self_Employed	614	582	5.211726	3	No	500	81.433225	Yes	82	13.355049
5	Credit_History	614	564	8.143322	3	1	475	77.361564	0	89	14.495114
6	Property_Area	614	614	0.000000	3	Semiurban	233	37.947883	Urban	202	32.899023
7	Loan_Status	614	614	0.000000	2	Y	422	68.729642	N	192	31.270358

Note. The figure shows the categorical data quality report. The report shows that the gender, married, dependents, self-employed, and credit history have missing values.

Figure 3

Continuous Data Quality Report Before Cleaning

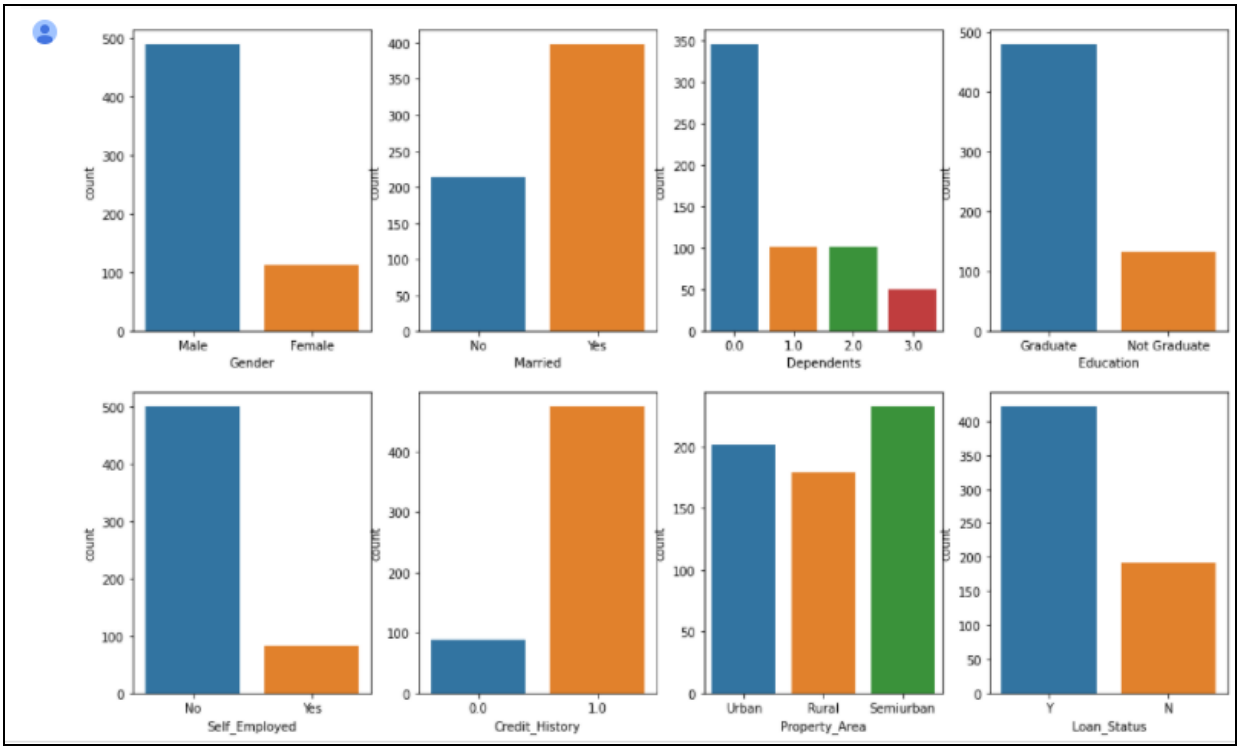


	Continous Feature	total	count	miss%	card	min	1st qrt	mean	median	3rd qrt	max	std
0	ApplicantIncome	614	614	0.000000	505	150.0	2877.5	5403.459283	3812.5	5795.00	81000.0	6104.064857
1	CoapplicantIncome	614	614	0.000000	287	0.0	0.0	1621.245603	1188.5	2297.25	41667.0	2923.864567
2	LoanAmount	614	592	3.583062	204	NaN	100.0	146.412162	128.0	168.00	NaN	85.515008
3	Loan_Amount_Term	614	600	2.280130	11	12.0	360.0	342.000000	360.0	360.00	480.0	65.066120

Note. The figure shows the continuous data quality report. The report shows the missing values for the features loan amount and loan amount term.

Figure 4

Data Exploration with Bar Graph



Note. The figure shows the exploration through bar graphs for categorical features gender, married, dependents, education, self-employed, credit history, property area, loan status.

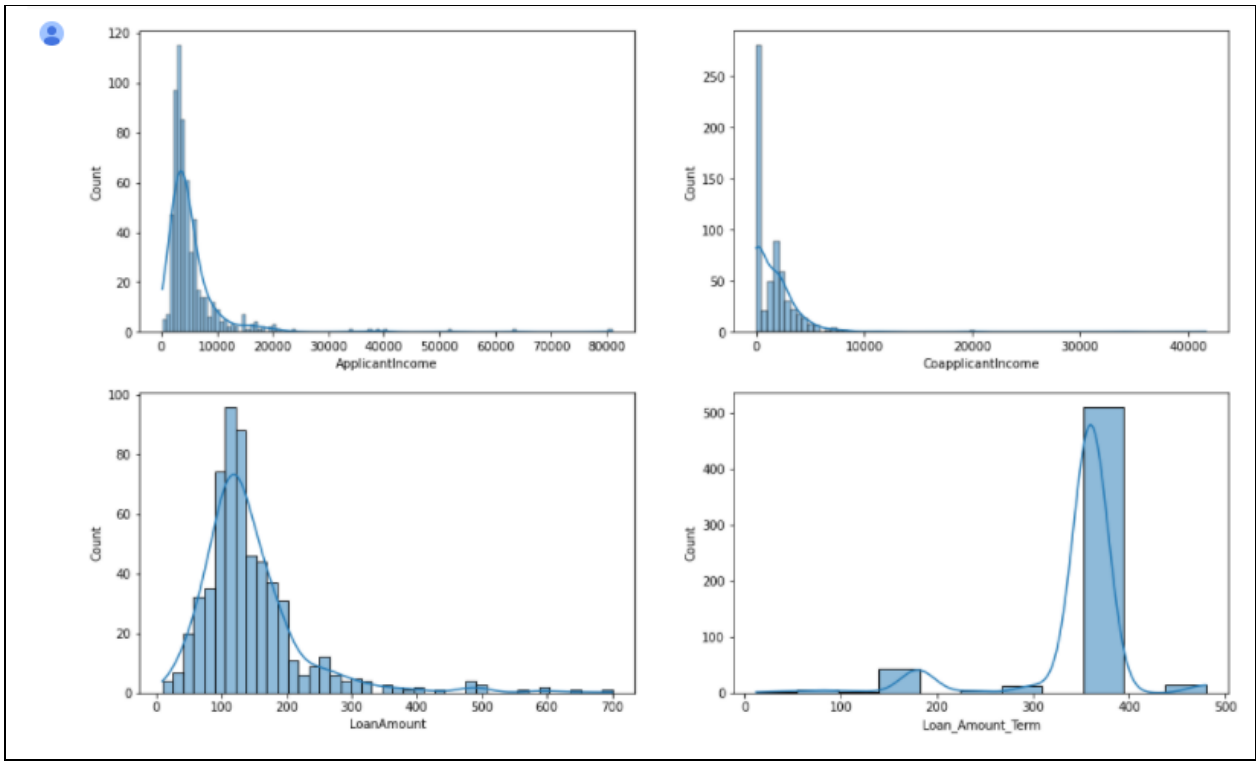
From the graph we can see below:

- There are more male applicants than females.
- Most applicants are married.
- Loan applicants with 0 dependents are seen more in the dataset.
- Applicants with three or more dependents are very few.
- Most applicants are graduates.
- Most applicants are not self-employed.

- Most applicants had a good credit history.
- There are more applicants living in semi-urban areas than urban and rural.
- The bank approved loans for most of the applicants.

Figure 5

Distribution Analysis for Various Continuous Fields



Note. The figure shows the distribution of the applicant's income, co-applicants income, loan amount, loan amount term.

From the distribution we can see below:

- The applicant's income range is 10 to 80000. The data is right-skewed. Most applicants have income in the range of 10 to 10000.
- The co-applicant income ranges from 10 to 40000 and the distribution is right-skewed. Most co-applicants have income in the range of 10 to 10000

- The loan amount range is approximately 10 to 700. The distribution is right-skewed. Most values lie in the range of 80 to 180.
- Most loans in our dataset had a 360 months loan amount term.

3.2 Data Preparation and Cleaning

The data cleaning had been performed on the data in the following ways:

Figure 6

Data Cleaning for Categorical Features

```
[ ] #filling the null values of categorical features with the its mode
    dataset ['Gender'] = dataset ['Gender'].fillna(dataset['Gender'].mode()[0])
    dataset ['Married'] = dataset ['Married'].fillna(dataset['Married'].mode()[0])
    dataset ['Dependents'] = dataset ['Dependents'].fillna(dataset['Dependents'].mode()[0])
    dataset ['Education'] = dataset ['Education'].fillna(dataset['Education'].mode()[0])
    dataset ['Self_Employed'] = dataset ['Self_Employed'].fillna(dataset['Self_Employed'].mode()[0])
    dataset['Credit_History'] = dataset['Credit_History'].fillna(dataset['Credit_History'].mode()[0])
    dataset ['Property_Area'] = dataset ['Property_Area'].fillna(dataset['Property_Area'].mode()[0])
    dataset ['Loan_Status'] = dataset ['Loan_Status'].fillna(dataset['Loan_Status'].mode()[0])
```

Note. The above figure shows the python code for replacing missing values of categorical features with their mode.

Figure 7

Data Cleaning for Continuous Features

```
[ ] #filling the null values of continuous features with the its mean
    dataset['ApplicantIncome'] = dataset['ApplicantIncome'].fillna(dataset['ApplicantIncome'].mean())
    dataset['CoapplicantIncome'] = dataset['CoapplicantIncome'].fillna(dataset['CoapplicantIncome'].mean())
    dataset['LoanAmount'] = dataset['LoanAmount'].fillna(dataset['LoanAmount'].mean())
    dataset['Loan_Amount_Term'] = dataset['Loan_Amount_Term'].fillna(dataset['Loan_Amount_Term'].mean())
```

Note. The above figure shows the python code for replacing missing values of continuous features with their mean.

Figure 8

Categorical Data Quality Report After Cleaning

	Categorical Feature	total	count	miss%	card	mode	mode freq	mode pct	2nd mode	2nd mode freq	2nd mode pct
0	Gender	614	614	0.0	2	Male	502	81.758958	Female	112	18.241042
1	Married	614	614	0.0	2	Yes	401	65.309446	No	213	34.690554
2	Dependents	614	614	0.0	4	0	360	58.631922	1	102	16.612378
3	Education	614	614	0.0	2	Graduate	480	78.175896	Not Graduate	134	21.824104
4	Self_Employed	614	614	0.0	2	No	532	86.644951	Yes	82	13.355049
5	Credit_History	614	614	0.0	2	1	525	85.504886	0	89	14.495114
6	Property_Area	614	614	0.0	3	Semiurban	233	37.947883	Urban	202	32.899023
7	Loan_Status	614	614	0.0	2	Y	422	68.729642	N	192	31.270358

Note. The figure shows the data quality report for categorical data after cleaning. The data in the figure is clean with no missing values.

Figure 9

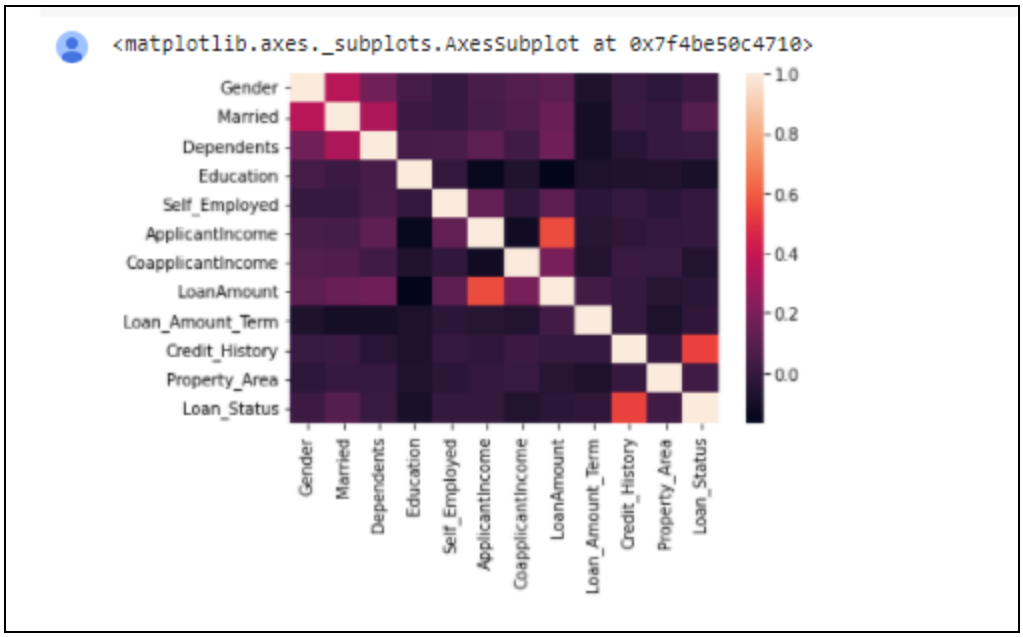
Continuous Data Quality Report After Cleaning

	Continous Feature	total	count	miss%	card	min	1st qrt	mean	median	3rd qrt	max	std
0	ApplicantIncome	614	614	0.0	505	150.0	2877.50	5403.459283	3812.5	5795.00	81000.0	6104.064857
1	CoapplicantIncome	614	614	0.0	287	0.0	0.00	1621.245603	1188.5	2297.25	41667.0	2923.864567
2	LoanAmount	614	614	0.0	204	9.0	100.25	146.412162	129.0	164.75	700.0	83.969005
3	Loan_Amount_Term	614	614	0.0	11	12.0	360.00	342.000000	360.0	360.00	480.0	64.320047

Note. The figure shows the data quality report for continuous data after cleaning. The data in the figure is clean with no missing values.

Figure 10

Heatmap for Correlation Between Various Features



Note. The figure shows a heatmap for correlation between different features of the dataset. The figure shows a good positive correlation between applicant income and loan amount. It also shows a good positive correlation between credit history and loan status.

3.3 Data Transformation

- Encoding
    - Encoding is used to convert categorical variables into numerical values to fit the data to ML models
    - Label Encoder encodes labels with a value 0 or 1
    - In one hot encoding, each categorical value convert into a new column and assign a value of 1 or 0

Figure 11

Label Encoder

```
[29] le = LabelEncoder()
     dataset['Gender'] = le.fit_transform(dataset['Gender'].astype(str))
     dataset['Married'] = le.fit_transform(dataset['Married'].astype(str))
     dataset['Education'] = le.fit_transform(dataset['Education'].astype(str))
     dataset['Self_Employed'] = le.fit_transform(dataset['Self_Employed'].astype(str))
     dataset['Property_Area'] = le.fit_transform(dataset['Property_Area'].astype(str))
     dataset['Loan_Status'] = le.fit_transform(dataset['Loan_Status'].astype(str))
```

Note. The label encoding is used for the features in the dataset with 2 values.

Figure 12

One Hot Encoding

```
[34] # Using one hot encoding for Property_Area and Dependents features

     ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [2, 10])], remainder='passthrough')
     X = np.array(ct.fit_transform(X))
```

Note. One hot encoding is done on features with 3 or more than 3 values.

- Feature Scaling
    - Feature scaling is used for the model to train fast and converge quickly
    - Standardization is used so that values are centered around mean with a unit standard deviation

Figure 13

Standardization Technique

```
[36] sc = StandardScaler()
     X_train = sc.fit_transform(X_train)
     X_test = sc.transform(X_test)
```

Note. The data is normalized using the standardization technique.

3.4 Model Proposals with Model Comparison and Justification

This is a supervised binary classification problem in which our models will predict either yes or no for loan approval prediction status. The proposed models for this project are:

- Logistic regression

It is a supervised classification algorithm. The target variable (y) will have categorical values. The descriptive features (X) can be either continuous or discrete. This model predicts the probability of a given target variable. Logistic regression models the data using sigmoid function whereas linear regression uses the linear function. In logistic regression, the model predicts the P(Y=1) as a function of X. In this model, the linear function is used as an input to the logistic or sigmoid function.

Based on the number of values in the target feature, we can classify logistic regression as:

1. Binomial: In this type of regression, the target variable can have only two possible values, either 0 or 1. Example, win or loss, pass or fail, etc.
2. Multinomial: In this type, the target variable can have 3 or more possible values without any order. An example of such can be disease A vs. disease B vs. disease C.
3. Ordinal: In this type, target variables contain ordered values. For example, the student can get first, second, or third in the class.

As the dataset we are using contains binary target values, we are using binomial logistic regression.

Figure 14

Logistic Equation

$$h_\theta(x) = g(\theta^T x) where 0 \leq h_\theta \leq 1$$

Note. The above figure shows the equation for the logistic equation. The image is taken from https://www.tutorialspoint.com/machine_learning_with_python/classification_algorithms_logistic_regression.htm

Figure 15

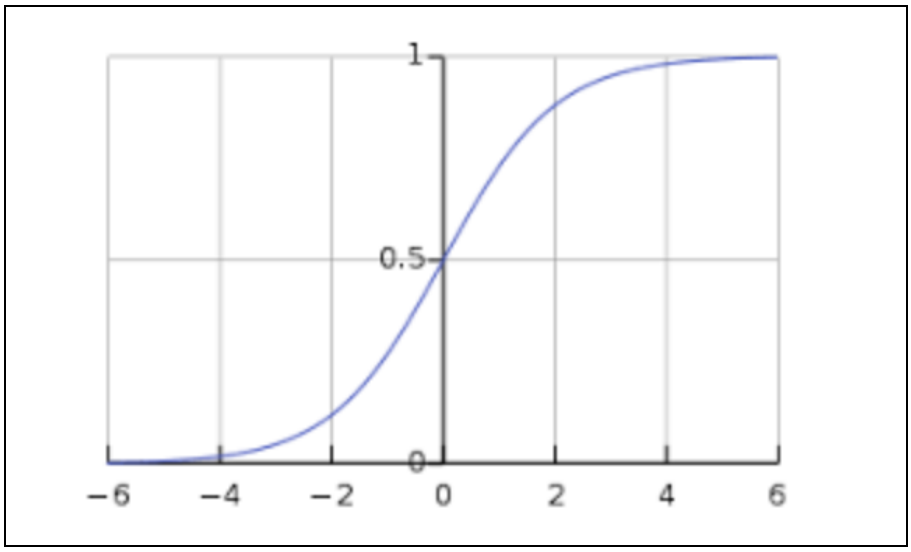
Logistic or Sigmoid Function

$$g(z) = \frac{1}{1 + e^{-z}} where z = \theta^T x$$

Note. In the above figure, g is the logistic or sigmoid function. The image is taken from https://www.tutorialspoint.com/machine_learning_with_python/classification_algorithms_logistic_regression.htm

Figure 16

Sigmoid Curve



Note. The sigmoid curve is shown above, as we can see y values are between 0 and 1, and it crosses the c axis at 0.5. The image is taken from https://www.tutorialspoint.com/machine_learning_with_python/classification_algorithms_logistic_regression.htm

The target labels can be divided into positive or negative. The target label comes under the probability of positive class if it lies between 0 and 1. For our implementation, we are considering the label as positive if it is ≥ 0.5, otherwise negative. We also need to define a loss function to measure how well the algorithm performs using the weights, represented by theta as, $h=g(X\theta)$.

Figure 17

Loss Function

$$J(\theta) = \frac{1}{m}.(-y^T log(h) - (1-y)^T log(1-h))$$

Note. The image shows how we can calculate the loss function. The image is taken from https://www.tutorialspoint.com/machine_learning_with_python/classification_algorithms_logistic_regression.htm

Here, our main goal is to minimize the cost function by increasing or decreasing the weights. With the help of derivatives of the loss function with respect to each weight, we will be able to know which parameters should have high weight and which should have a small weight.

Figure 18

Gradient Descent Equation

$$\frac{\delta J(\theta)}{\delta \theta_j} = \frac{1}{m} X^T (g(X\theta) - y)$$

Note. The image describes how the gradient descent equation to help loss will change if we changed the parameters.

- Decision tree classifier

According to the book Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies by John D. Kelleher, Brian Mac Namee, and

Aoife Darcy, a Decision tree consists of a root node, interior nodes, and leaf nodes where interior nodes are tested on descriptive features and each leaf node is a predictive output of the target class. Tests carried out on each feature are information gain which calculations are dependent on impurity measures of either entropy or Gini index and remainder. Figure 19, Figure 20, Figure 21, and Figure 22 depict calculations for entropy, Gini index, remainder, and information gain. Plus, the decision tree method works on both categorical and continuous data. Tree pruning methods can be used to help with overfitting problems.

The advantages of the decision tree:

- Easy to interpret.
- Suitable for both categorical and continuous data.
- Demonstrate interactions between features.

The disadvantages of the decision tree:

- Can create complex trees if the dataset is too large.
- Eager learning which retrains are required and can be costly.

Figure 19

Entropy Calculation

$$H(t, D) = \sum_{t \in levels(t)} (P(t = l)\, X\, \log_2(t = l))$$

Note. The formula for calculation of entropy.

Figure 20

Gini Index Calculation

$$Gini(t, \mathcal{D}) = 1 - \sum_{l \in levels(t)} P(t = l)^2$$

Note. Figure 20 shows the formula for calculating the gini index.

Figure 21

Remainder Calculation

$$rem(d, \mathcal{D}) = \sum_{l \in levels(d)} \underbrace{\frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|}}_{\text{weighting}} \times \underbrace{H(t, \mathcal{D}_{d=l})}_{\substack{\text{entropy of} \\ \text{partition } \mathcal{D}_{d=l}}}$$

Note. The entropy part can be replaced by the Gini index calculation.

Figure 22

Information gain calculation

$$IG(d, \mathcal{D}) = H(t, \mathcal{D}) - rem(d, \mathcal{D})$$

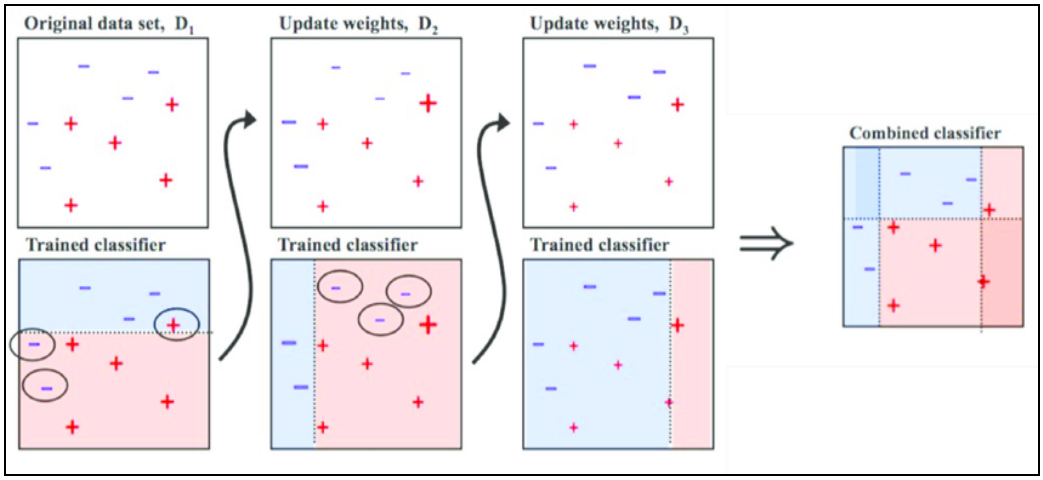
Note. Figure 22 shows the formula for calculating information gain.

- AdaBoost classifier

AdaBoost is also called the Adaptive Boosting algorithm. It is an ensemble model based on Boosting technique, where we use multiple weak classifiers to make a strong one as shown in Figure 23.

Figure 23

AdaBoost Classifier Flow



Note. Weight is getting updated after each iteration and finally, all classifiers are combined.

As compared to other boosting algorithms, AdaBoost focuses on altering the weights of input data based on the previous model's predictions. The weight is updated based on total error, using total error we hence calculate the alpha value.

$$\alpha = \frac{1}{2}\ln\left(\frac{1 - Total\ Error}{Total\ Error}\right)$$

Then this alpha value is used to update the weights in each iteration.

$$w_i(Updated\ Weight) = w_{i-1} \times e^{\pm\alpha}$$

Positive alpha if the data is classified incorrectly and negative in case of correct classification. The iteration is continued till the model reaches the desired level of accuracy.

- Gradient Boosting classifier

Gradient Boosting Algorithm is an ensemble model that uses boosting techniques. It is based on a decision tree classifier. It focuses on combining multiple weak classifiers to make a single strong classifier. It follows the Boosting Ensemble technique, where the previous model's output is used as an input for the next model. Gradient boosting Algorithm uses Gradient Descent function (shown in equation) to optimize the loss of the output then uses this optimized output as the input for the next iteration.

$$Y_{i+1} = Y_i - learning\ rate * \frac{d}{dx} f(Y_i)$$

Where Y_i is the previous input.

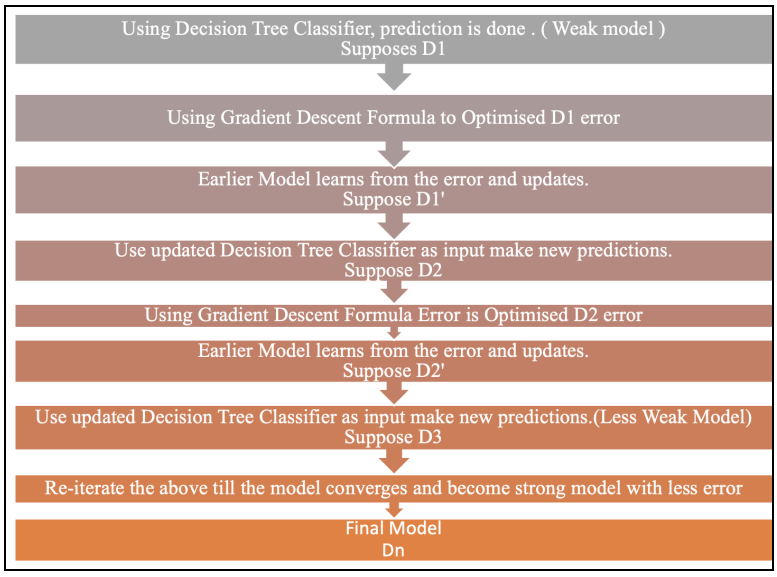
Y_{i+1} is new input

$F(Y_i)$ is output of previous input

Figure 24 shows the broad level flow of Gradient Boosting Algorithm

Figure 24

Flow showing Gradient Boosting Algorithm



Note. General Flow of Gradient Boosting Algorithm

Table 3

Strengths, Limitations, and Justification for Logistic Regression

Strengths	Limitations	Justification
Easy to implement, efficient to train	Can lead to overfitting	Large enough to provide faster results
Can be extended to multinomial regression & uses a probabilistic approach	Assumption of linearity between the dependent and independent variable	Widely used for similar issues
Measures are given as coefficients and provide us with direction with positive and negative values.	Works only on linearly separable data	The data used for the project is linear data which should provide good results

Note. The strengths and limitations for a logistic regression model are from https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/

Table 4

Strengths, Limitations, and Justification for Decision Tree Classifier

Strengths	Limitations	Justification
Less time for data pre-processing	Sensitive to change in data which affects the structure of the tree	
Can work with continuous and categorical data	Complex calculations	The data for the project contains categorical and continuous data
Easy to interpret the results due to eager learning nature	Higher time to train	

Note. The strengths and limitations for decision tree model are from https://learning.oreilly.com/library/view/machine-learning-with/9781787121515/697c4c5f-1109-4058-8938-d01482389ce3.xhtml

Table 5

Strengths, Limitations, and Justification for AdaBoost Classifier

Strengths	Limitations	Justification
Less prone to overfitting	Cannot perform well if data is noisy & has outliers	Used to improve the performance of prediction

Note. The strengths and limitations for AdaBoost are from https://www.analyticsvidhya.com/blog/2021/06/adaboost-a-brief-introduction-to-ensemble-learning/

Table 6

Strengths, Limitations, and Justification for Gradient Boosting Classifier

Strengths	Limitations	Justification
It has flexibility and provides various hyperparameter tuning options.	The continuous improvement of reducing errors can lead to overfitting	Used to decrease the bias error

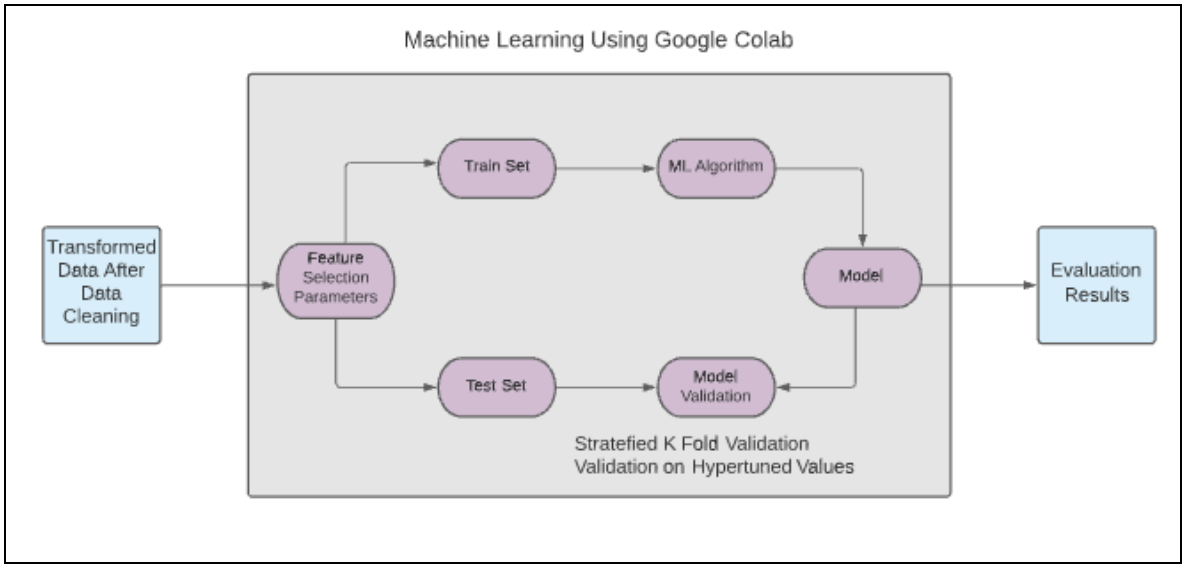
Note. The strengths and limitations for gradient boosting are from http://uc-r.github.io/gbm_regression#proscons

3.5 Model Support

The project is done using the shared access of Google colab between our team members. The raw data is imported in Google colab which will be cleaned and transformed. The transformed data will be used for machine learning. The figure shows that the training dataset is provided on which the algorithm learns and provides the model. We have also performed stratified k fold validation on hyper tuned parameters.

Figure 25

Model Support Flow Chart



Note. The figure shows the flow chart representation of the machine learning platform.

4. Results

In this section, we will be showing the results of different evaluation metrics used to evaluate the model.

4.1 Model Evaluation Methods

There are various model evaluation metrics used for the evaluation of loan approval prediction. The classification problem metrics used are as below:

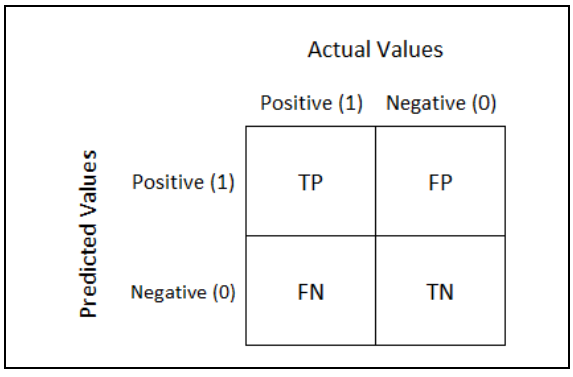
- Confusion matrix
- Overall Accuracy
- F1 score
- Receiver Operating Characteristics (ROC) curve and Area Under Curve (AUC)
- Model learning curve

Confusion matrix

A confusion matrix is a table that describes the performance of a classification model on a test set. It shows the correct and incorrect predictions for target levels.

Figure 26

Parameters of Confusion Matrix



Note. The figure is taken from https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 shows TP = true positives, TN = True negatives, FN = False negatives, and FP = false negatives.

Accuracy

Accuracy is the ratio of the number of correct predictions to the total number of predictions.

F1 score

F1-score is the harmonic mean of precision and recall. We are using an F1-score to compare the performance of different classification models.

Figure 27

F1 Score Formula

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

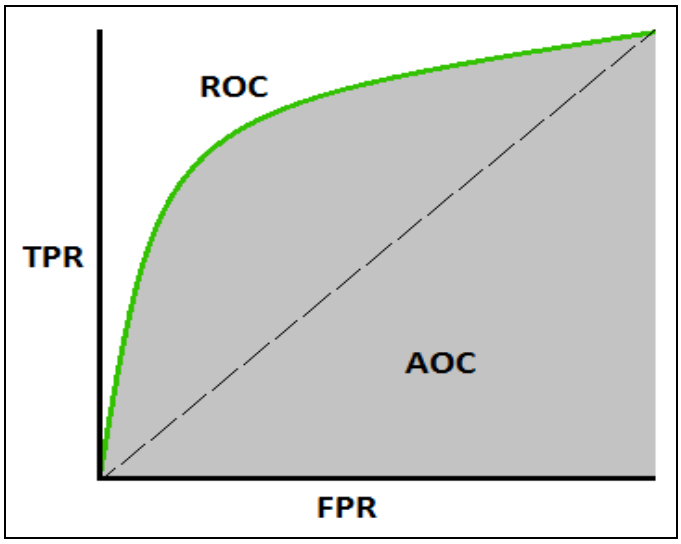
Note. The picture of the f1 formula is taken from the https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

ROC and AUC

ROC curve is a tradeoff between true positive rate (TPR), and false-positive rate (FPR). TPR is also called sensitivity and FPR is also called specificity. Using the ROC curve, we can observe the model's performance. For the model to be performed better, the ROC curve for the model should belong to the top left corner of the graph. AUC is used as a summary of the ROC curve.

Figure 28

AUC-ROC Curve



Note. The figure is taken from

https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Figure 29

TPR (True Positive Rate) / Recall /Sensitivity

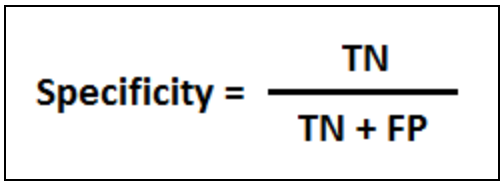
$$\text{TPR /Recall / Sensitivity} = \frac{TP}{TP + FN}$$

Note. The figure shows how we calculate the true positive rate. The figure is taken from

https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Figure 30

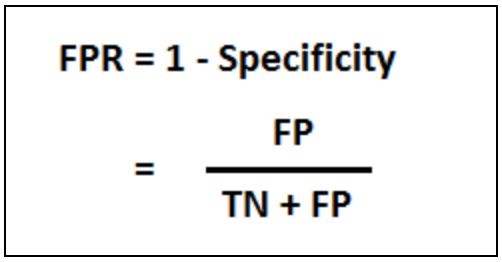
Specificity



Note. The figure shows the formula for specificity. The image is taken from https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Figure 31

FPR



Note. The figure shows the formula for FPR. The image is taken from https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Learning Curve

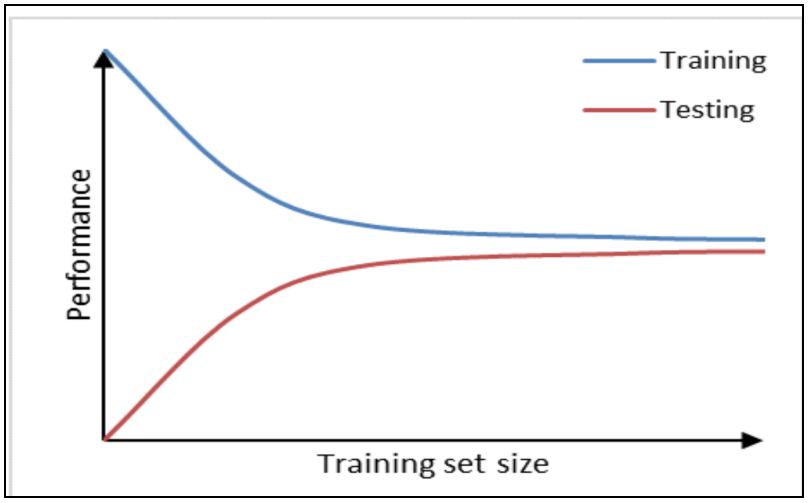
We have also used a learning curve to see model performance on the train set and validation set after stratified k fold. Figure 32 shows an example of the learning curve. In the learning curve plot, the y axis shows the performance of the model, and the x-axis shows the training set size.

The model performance is dependent on the size of the train and test set. The smaller train set can increase the performance of the model on the train set due to overfitting. Thus, the results provided on the test set would not be great. At the same time, if the size of the train set increases, it results in poor performance. Because of the large size of the train set, learning models will be difficult.

With the help of a learning curve, we will be able to understand the bias and variance for our loan approval prediction model showing us the overall performance. The results could help us to understand if we can improve the performance of the model by reducing the variance and the bias.

Figure 32

Learning Curve



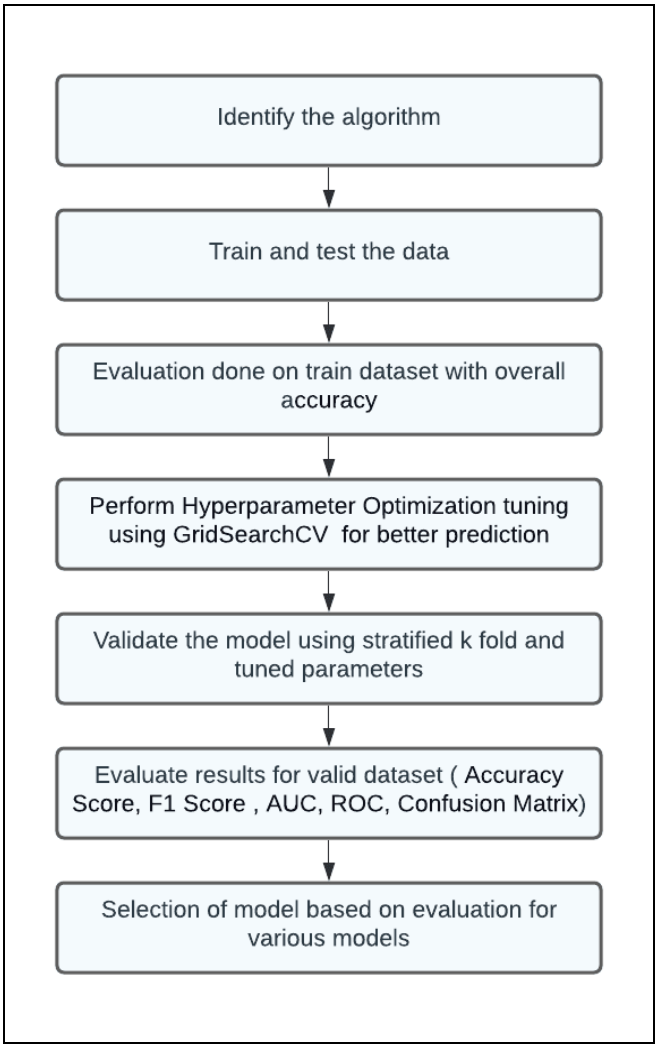
Note. The above figure shows a learning curve example. The image is taken from https://ohdsi.github.io/PatientLevelPrediction/articles/CreatingLearningCurves.html

4.2 Model Validation and Evaluation Steps

There are four models which will be used for loan prediction. This process will use steps as shown in Figure 33.

Figure 33

Flowchart showing steps followed for Validation and Evaluation



Note. The above figure shows the flow chart describing various steps performed for validation and evaluation.

4.3 Model Validation and Evaluation Results

The validation and evaluation for loan prediction are performed as shown below:

- Step 1 - Validate model results using training dataset for logistic regression, decision tree, AdaBoost classifier, and gradient boosting

- Step 2 - Hyperparameter tuning and showing best parameter for logistic regression, decision tree, AdaBoost classifier, and gradient boosting

- Step 3 - Evaluation was done using stratified k fold on tuned parameters for Logistic regression, decision tree, AdaBoost classifier, and gradient boosting

Table 7

Summary of Results of Model for Step 1

Model	Accuracy	F1 score	Confusion matrix	AUC
Logistic Regression	0.7964	0.8979	18 20 0 84	0.716
Decision Tree	0.6938	0.7738	23 15 23 61	0.664
AdaBoost	0.7068	0.8666	17 21 17 67	0.751
Gradient Boosting	0.7967	0.8691	19 19 3 81	0.688

Note. The table shows the accuracy, f1 score, confusion matrix, and AUC results on the training dataset.

Table 8

Hyperparameter Tuning and Best Parameter for Step 2

Model	Parameters	Best Parameter
Logistic Regression	solvers = ['newton-cg', 'lbfgs', 'liblinear'] penalty = ['l2'] c_values = [100, 10, 1.0, 0.1, 0.01]	solver = newton-cg, penalty = l2, c_values = 100
Decision Tree	criterion=["gini", "entropy"] splitter = ["best", "random"] max_features = ["auto", "sqrt", "log2"]	Criterion = gini, splitter = best, max_features = auto
AdaBoost	n_estimators = [10, 50, 100, 500, 1000, 5000] algorithm = ['SAMME', 'SAMME.R'] learning_rate = [0.0001, 0.001, 0.01, 0.1,1]	Algorithm = SAMME, learning_rate = 0.0001, n_estimators = 10
Gradient Boosting	n_estimators = [10, 100, 1000] learning_rate = [0.001, 0.01, 0.1] subsample = [0.5, 0.7, 1.0] max_depth = [3, 7, 9]	n_estimators = 1000 learning_rate = 0.001 subsample = 0.5 max_depth = 3

Note. The table shows the different parameter combinations used for different models. The table also shows the best parameters chosen from the given combinations.

Table 9

Summary of Results of Tuned model using Stratified k Fold Validation for Step 3

Model	Accuracy	F1 score	Confusion matrix	AUC
Logistic Regression	0.8111	0.877308	18 20 0 84	0.736842
Decision Tree	0.724670	0.798954	22 16 24 60	0.646617
AdaBoost	0.809490	0.876659	18 20 0 84	0.736842
Gradient Boosting	0.806238	0.874910	18 20 0 84	0.736842

Note. The table shows the accuracy, f1 score, confusion matrix, and AUC results after tuned stratified k fold.

5. Discussion

Table 10

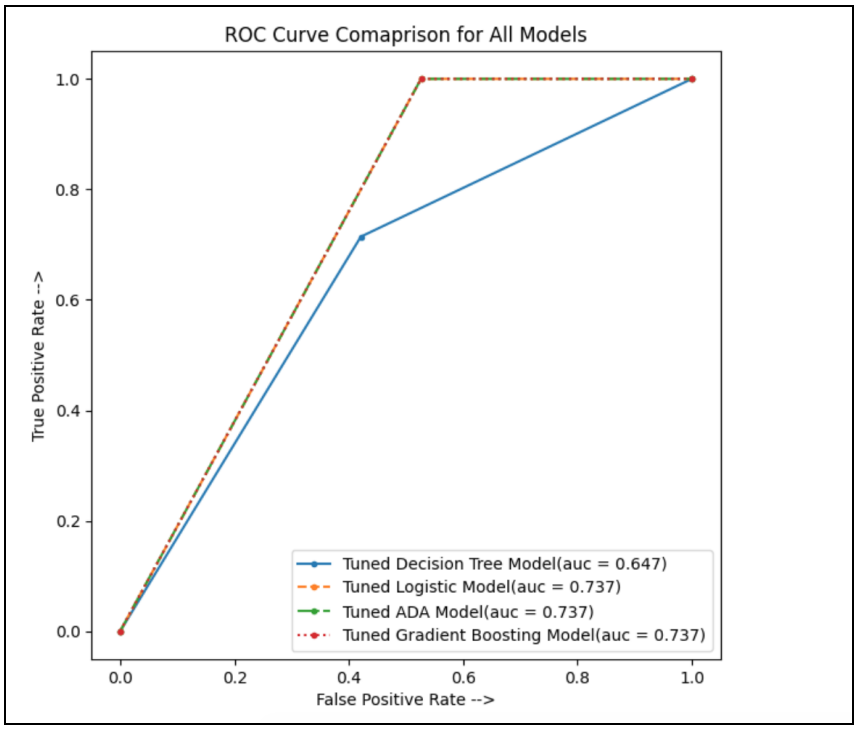
Comparison Summary Results for Various Models

	Model	accuracy_score	auc	F1 score
0	DecisionTreeClassifier	0.693802	0.665727	0.774979
1	Tuned DecisionTreeClassifier	0.724670	0.646617	0.798954
2	LogisticRegression	0.796481	0.736842	0.866395
3	Tuned LogisticRegression	0.811116	0.736842	0.877308
4	AdaBoostClassifier	0.706811	0.622494	0.792098
5	Tuned AdaBoost	0.809490	0.736842	0.876659
6	GradientBoostingClassifier	0.778609	0.771617	0.853400
7	Tuned GradientBoostingClassifier	0.807904	0.783521	0.875843

Note. The table shows the comparison of the results by different models. This table is from the Google colab code we ran for Loan prediction. Tuned logistic regression has the highest accuracy score of 0.8111116 and the highest f1 score with a value of 0.877308.

Figure 34

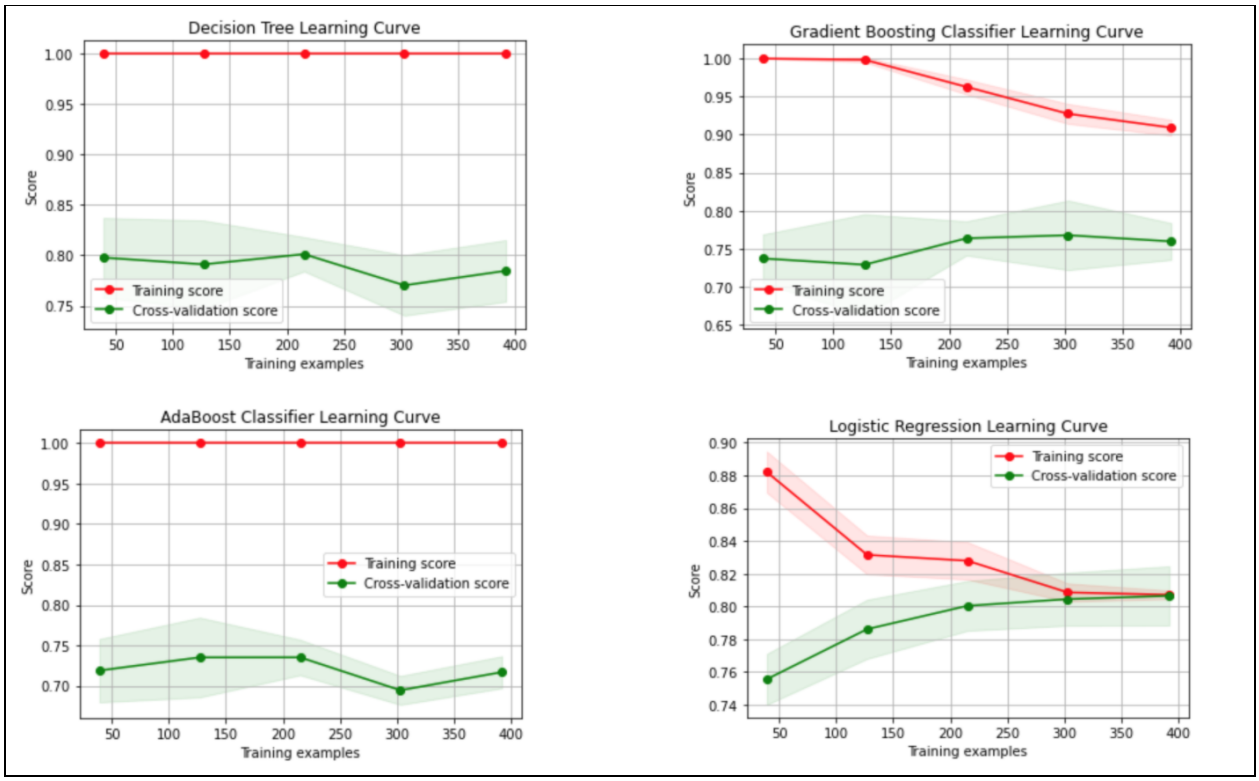
Comparison of ROC Curve and AUC for Various Tuned Models



Note. The figure shows the ROC curves and AUC for various models. We can see that the AUC value for logistic regression, AdaBoost, and gradient boosting is the same as 0.737.

Figure 35

Learning Curves for Various Models



Note. The above figure shows the learning curve for all the models. From this, we can see that the logistic regression model is performing well.

6. Evaluation and Reflection

Conclusion

As per the evaluation of all the models: Decision Tree Classifier, Logistic Regression, AdaBoost Classifier using metrics such as Accuracy Score, F1-Score, Confusion Matrix, Learning Curves, ROC, and AUC we can see that Logistic Regression Model performed the best among all the models, followed by AdaBoost classifier, GradientBoosting classifier and finally Decision Tree classifier.

Key Learnings

In this project, we proposed models that could help banks and customers to check if a person who has applied for a loan is eligible or not. For the development of this project, we followed the CRISP-DM process. Data was collected, clean, and transformed for model training and testing and finally, models were evaluated.

During the data collection and cleaning phase, we learned concepts related to analyzing categorical and collecting data, handling null values, and outliers. During the data transformation phase, we learned about entropy and information gain calculation for features, one-hot encoding, and label encoding.

When it came to modeling evaluation and validation, we learned about splitting a dataset into training and testing datasets, creating models, training and testing them, validating them using Stratified K-Fold Validation, improving model performance by tuning hyper-parameters and evaluating them based on various metrics such as Accuracy Score, F1-Score, Confusion Matrix, ROC and AUC.

Future Scope

The following points can be part of our future scope ：

- Increase the accuracy of models above 90 percent.
- Use Deep learning models to classify.
- Create an end-to-end data pipeline that can take real-time data and classify it.
- Create an interface using which banks and customers can use the model.
- Deploy the model as a part of an extensive software structure.

References

Chatterjee, D. (2019). *Loan Prediction Problem Dataset*. Kaggle. https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset/metadata

Mukid, M. A., Widiharih, T., Rusgiyono, A., & Prahutama, A. (2018, May). Credit scoring analysis using weighted k nearest neighbor. In *Journal of Physics: Conference Series* (Vol. 1025, No. 1, p. 012114). IOP Publishing.

Odegua, R. (2020). Predicting Bank Loan Default with Extreme Gradient Boosting. *arXiv preprint arXiv:2002.02011*.

Vimala, S., & Sharmili, K. C. (2018). Prediction of loan risk using naive bayes and support vector machines. In *Int Conf Adv Comput Technol (ICACT)* (Vol. 4, No. 2, pp. 110-113).

S. Sreesouthry, A. Ayubkhan, M. Mohamed Rizwan, D. Lokesh, K. Prithivi Raj. (2021). Loan Prediction Using Logistic Regression in Machine Learning. *Annals of the Romanian Society for Cell Biology*, 2790 –. Retrieved from https://www.annalsofrscb.ro/index.php/journal/article/view/2818

Kelleher, J., & Namee, B., & D'Arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics*. The MIT Press.

Eweoya, I. O., Adebiyi, A. A., Azeta, A. A., & Azeta, A. E. (2019, August). Fraud prediction in bank loan administration using decision tree. In *Journal of Physics: Conference Series* (Vol. 1299, No. 1, p. 012037). IOP Publishing.

https://www.businessinsider.com/what-caused-the-great-recession

https://www.debt.org/faqs/americans-in-debt/demographics/