Loan Approval Prediction

Abhinay Gupta Chhavi Kirtani Rachit Jaiswal
Indraprastha Institute of Information Technology Delhi
Okhla Industrial Estate, Phase III, near Govind Puri Metro Station, New Delhi, Delhi 110020

Abstract

The banking sector has evolved a lot with many people applying for loans. Loan approval is a very important process for banking organizations, as loan repayment is a major contributing factor in the bank's financial statement. By automating a machine learning model, both the banking organization and the person applying could be benefited. The model would predict how likely a person is to be able to pay back the loan in a given duration of time, and hence will help the bank decide if the loan should be approved. This whole process would be automated and hence would reduce the amount of manual work done by employees in the bank. We first experimented with easier models like Logistic regression, Decision Tree ,Random Forest, Gaussian Naive Bayes, Support Vector Machine and then progressed to complex models such as ADAboost, XGboost and Neural Networks. Experimental tests found that the Random Forest has the best accuracy, precision and AUC in making the right decision to approve or reject the loan request of the customers. We also experimented with ensemble methods, such as combining random forest and svm together to check for better performance. It was found that Random Forest still performs better than the ensemble methods.
Github Link for the project: `https://github.com/chhavikirtani2000/Loan-Prediction`
Keywords: Loan data, Data Analysis, Feature Selection, Machine Learning

1. Introduction

The most critical task for the banking sector is deciding how risky is it to provide loan to the borrower i.e., will the borrower be able to repay loan or not.
Today many financial companies perform regress process of verification and validation but still there is no surety whether the borrower is the right applicant who can be trusted or not, in terms of repaying loan.
Lending money to inappropriate borrowers who doesn't repay the loan will make the industry run in loss, as the major source of assets for the banking industry comes from the profit generated from the interest which is paid by the borrower. The prime focus in banking sector is to invest their assets in safe hands. Our aim is to provide efficient, easy method to filter the deserving applicants. With features such as annual income and interest rates, we can determine if the borrower is eligible for the loan using machine learning approaches. Through this technique, we can fully automate the whole process of verification and validation of applicants. Loan prediction using machine learning can calculate the weight of each feature in the training data and then the same features can be processed in the test data with respect to their predicted weights. The drawback of this technique is that by assigning different weights to each feature, our prediction depends on all features, but sometimes loan approval prediction doesn't need to evaluate complex feature representation, it can focus on few features and give near accurate results.

2. Literature Survey

In **[1]** Vimala, Sharmili K.C. (2018) 'Prediction of Loan Risk using Naive Bayes and Support Vector Machine', *International Conference on Advancements in Computing Technologies* - ICACT, 4(2), pp. 110 – 113
This paper focused on evaluating the performance of Naive Bayes and SVM combined. The paper provided an interesting result where the combination of the two methods gave better accuracy and time consumption.

In **[2]** Yiyun Liang, Xiaomeng Jin , Zihan Wang. (2019) 'Loanliness: Predicting Loan Repayment Ability by Using Machine Learning Methods', *7th International Conference on Information Technology and Quantitative Management*, 162, pp. 503-513
This paper applied methods like Random Forest, Logistic Regression, Naive Bayes, LightGBM and Neural Networks, and provided good insights on class imbalance and missing values.

In **[3]**]Arun, K., Ishan, G. and Sanmeet, K., 2016. Loan Approval Prediction based on Machine Learning Approach. *IOSR J. Comput. Eng*, 18(3), pp.18-21.

The model calculates the weight of each features taking part in loan processing and on new test data same features

are processed with respect to their associated weights. Six ML classification models have been used for prediction like Decision Tree, Random Forest, SVM etc.

3. Dataset: Dataset details with data preprocessing techniques.

The dataset is taken from LendingClub.com [4], a website that connects borrowers and investors over the internet.

3.1. DataSet Description

3.1.1 Attributes/features description

credit_policy: If the borrower meets the credit criteria
purpose: The purpose of the loan
Int_rate: The interest rate of the loan
Installment: Monthly installment borrower will owe
Log_annual_inc: Natural log of the annual income
Dti: Debt to income ratio of the borrower
Fico: The FICO credit score
Days_with_cr_line: The number of days the borrower had a credit line
Revol_bal: The borrower's revolving balance
Revol_util: The borrower's revolving line utilization rate.
Inq_last_6mths: The borrower's number of inquiries by creditors in the last 6 months
Delinq_2yrs: The number of times borrower was 30+ days past the due date of payment in the past 2 years
Pub_rec: Total Number of derogatory records
Not_fully_paid: Indicates if the loan was not fully paid.

3.1.2 Data Analysis

- The dataset has 9578 Rows and 14 columns (13 features/input and 1 output)
- There are no duplicate rows.
- DataSet Types: 4 Categorical features, 9 Numerical Features
- Null/Missing values: 6 features had missing values across all data points. 1 of those features were categorical. The other 5 were numerical.

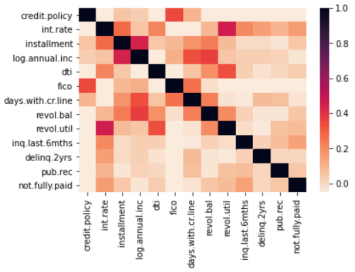

Figure 1. Correlation between features

3.2. Data Preprocessing

3.2.1 Handling missing data:

The missing data is replaced by the feature mean for numerical data and feature mode for categorical data.

3.2.2 One Hot Encoding:

Four of the features from our dataset have categorical values. It is important to deal with such variables to avoid the loss of important variables in a model. One hot encoding is the technique which is done to convert categorical values from features into dummy variables.

3.2.3 Data Scaling:

If different features have values in varying ranges, one feature might dominate over the other. To handle this, we used RobustScaler technique for standardisation. RobustScaler centers the data around the median and scale it using interquartile range.

3.2.4 Handling Data Imbalance:

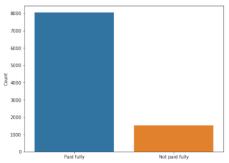

Figure 2. Class Counts

Our data set is highly imbalanced because 81% of the total samples have paid the loan which influences a learning algorithm during training by making the decision rule biased towards the majority class by implicitly learns a model that optimizes the predictions based on the majority class in the dataset. [5] So, to increase the sample case of minority class we used upsampling. Up-sampling is the process of randomly duplicating observations from the minority class in order to reinforce its signal.

3.2.5 Feature Selection:

We performed feature importance analysis using Extra Tree Classifier to see which factors are relevant to the for our result.After analysing,we found that features pub.rec and purpose do not play an important role for determining if the loan was repaid or not.

4. Methodology, model details

4.1. Methodology

Exploratory Data analysis is performed in order to get the idea of the preprocessing measures that are needed to be

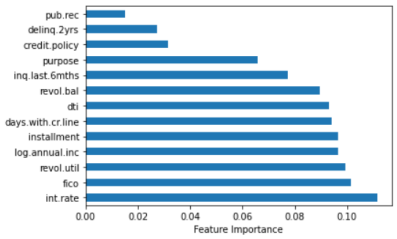

Figure 3. Feature Importance

taken.

4.1.1 Handling missing values

Missing Values and categorical values are properly handled where missing values are replaced by mean for numerical data and categorical values are replaced with mode of their respective column.

4.1.2 Splitting the dataset into Train and Validation sets:

The data is highly imbalanced, as can be seen below: To handle this, we decided to use stratified sampling method to split the data in such a way that every class is represented equally in both train and test set.This ensures the model learns about every class it needs to evaluate in test set.

4.1.3 Handling Data Imbalance:

In the train set, still the representation of paid class is large because of which there is a chance of biasness. To handle this imbalance, we used the method of upsampling the minority class to increase the sample case of minority class.

4.1.4 Training and testing models:

We used scikit implementation of the models to get the results. Since, we are dealing with imbalanced data for classification, we calculated various metrics such as precision, recall, AUC score along with accuracy to consider the performance, and performed cross-validation to get the best results.

4.1.5 Hyper-parameter tuning:

We performed grid search to evaluate the best hyper-parameters for models.

4.2. Model details

We have implemented the following models: [6]

1. Logistic Regression:

It is a supervised learning algorithm that uses weights and bias to predict the output. It assumes linearity of classification problem. It uses the formula in figure 2 to give the probability of a given data point belonging to an allocated class. (where bi=weights to predict)

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_p x_p)}}$$

Figure 4. Logistic Regression

2. Naive Bayes:

Naive Bayes is a simple technique which assumes all of the input attributes to be relatively independent. It uses the probability theory to classify data. We use Gaussian Naive Bayes, in which continuous attributes are considered to be distributed according to a Gaussian distribution. The conditional probability is given by:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Figure 5. Conditional probability

The final prediction is made using the following general formula, where P(B/A) is calculated using the above formula.

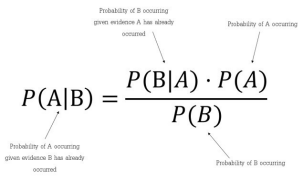

Figure 6. Naive Bayes formula

3. Decision Tree:

Decision tree is a supervised machine learning algorithm which could be used to solve classification and regression problems. It uses tree representation to predict the output of an unclassified point. External nodes represent attributes and leaf node in a tree represents class labels. In Decision Tree, best attribute is chosen at every step to perform the split on the tree. Techniques like information gain etc are used to perform the split. This process is then repeated until all the data points are split.

4. Random Forest Classifier:

Random Forest is a supervised learning algorithm. It can be seen as an ensemble of decision trees with bagging method. A bagging method is a combination of learning models, where each model is trained in parallel on random model to improve the overall result. The Random Forest algorithm randomly selects observations and features to build several decision trees and then takes the average of the results. Unlike decision trees, random forest prevents overfitting as it creates random subsets of the features and constructs smaller sub trees. Random forest can be seen like a

cross-validation algorithm performed for decision trees.

5. Support Vector Machine:

It is a supervised learning algorithm which uses a separating hyperplane. In other words, given labeled training data, the algorithm outputs an optimal hyperplane which classifies new examples. In two-dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side. In multi-dimensional space, the separation of the class is a hyperplane.

6. K-nearest neighbours[7]:

KNN is a supervised machine learning technique which could be used for both classification and regression problems. KNN assumes that similar things are close to each other.The intuition is that the algorithm plots the data points on a graph and locates some clusters or groups. To classify a data point in test set, the algorithm checks what groups does its k-nearest neighbours belong to.

7. AdaBoost [8] :

Boosting algorithms are powerful techniques to solve complex problems. Adaboost combines multiple "weak learners" and develops and strong classifier. The weak classifiers in Adaboost are the decision trees with a single split and Adaboost improves upon these classifiers by putting more weights on instances that are difficult to classify and less weight on instances that are already handled. This way Adaboost works on the "weakness" of weak classifiers to build a stronger classifier. The final prediction is the weighted majority vote.

Adaboost is different from Random forest as Adaboost is an ensemble method based on boosting, and random forest is based on bagging.

In bagging, individual models are trained in parallel on random subset of data, whereas in boosting, models are trained in a sequential way, each learning from the mistakes made by the previous model.

AdaBoost is capable of overfitting in datasets with less noise due to complexity in the model's nature and has only a few hyperparameters that need to be tuned to improve model performance.

8. XGBoost [8]:

eXtreme Gradient Boosting is similar to Random forest as it also uses individual models and perform sequential learning. The individual models used are similar to decision trees, however, each leaf node in them are assigned continuous scores which are summed up to provide the final prediction. For each iteration i, a tree t is grown and scores w are calculated to predict the certain outcome y.

9. Neural Networks [9]:

The neural network type we used is Sequential, which allows to build a model layer by layer where each layer has weights associated with it. By using the add() function, we added two layers to our model, along with an output layer. Our layer type is Dense, which is a standard layer type that works for most cases. In a dense layer, all nodes in the previous layer connect to the nodes in the current layer. Model complexity and hence the capacity can be increased by increasing the number of layers and the number of nodes in each layer. The activation function we used is ReLU or Rectified Linear Activation, which is defined by the following formula.

Relu(x) = 0 if $x < 0$

Relu(x) = x if $x >= 0$

The last layer is the output layer. It only has one node, which gives us the predicted probability of the positive class.

Ensemble Methods: We tried combining models together to check if any particular combination gives good results. We used two methods to combine models together.

1. Hard Voting: To classify a sample data point, the ensemble classifier chooses the class predicted by majority of the models.

2. Soft Voting: To classify a sample data point, the ensemble classifier chooses the class on the basis of average probability over all the models combined.

5. Results and analysis

5.1. Results

All the following results are reported after manual hyperparameter tuning and cross validation.

Best overall accuracy, precision and AUC score was given by **Random Forest.**

Accuracy: 0.9733
AUC score: 0.994
Precision score: 0.9568
F1 score: 0.97354

We trained complex models such as ADAboost, XGboost and a Neural Network with 2 hidden layers. The results for these methods are given below:

ADAboost:
Accuracy: 0.89838
AUC score: 0.898
Precision score: 0.8344
F1 score: 0.90647

XGboost:
Accuracy: 0.6954067
AUC score: 0.767
Precision score: 0.6335858
F1 score: 0.704359

Neural Network:
Accuracy: 0.907364
AUC score: 0.92
Precision score: 0.91
F1 score: 0.90

Linear SVM vs Gaussian kernel based SVM:

Model	Accuracy	AUC
Linear SVM	0.63	0.609
Gaussian kernel based SVM	0.68	0.675

Considering ensembled models, best overall accuracy and AUC score was given by combining Support Vector and Random Forest models together, using soft voting.
Soft Voting Accuracy: 0.9134497172362921
Auc score: 0.9134650682615894
A summary of results obtained from all the models:

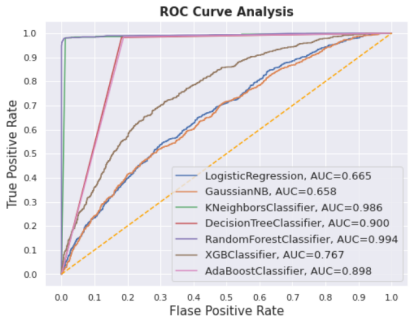

Figure 7. ROC curve

Model	Accuracy	Precision	AUC
Logistic Regression	0.61329	0.5695	0.665
Gaussian Naive Bayes	0.6107	0.571	0.658
K-nearest neighbours	0.806	0.72153	0.986
Decision Tree	0.9	0.83754	0.9
Random Forest	0.9733	0.9568	0.994
Adaboost	0.89838	0.8344	0.898
XGboost	0.6954067	0.6335858	0.767
Neural Network	0.907364	0.91	0.92

Hyper-parameter evaluation using grid-search for some good performing models is given below
Random Forest:
bootstrap: True
max depth (longest path between the root node and the leaf node for all trees': 60
max features: auto
min samples leaf: 5
min samples split: 10
n estimators (number of trees to build): 500
Adaboost:
algorithm=SAMME.R
base estimator=Decision Tree Classifier
learning rate=0.98
n estimators=2
K-nearest neighbours:
algorithm=auto
leaf size=30
metric=manhattan
n neighbors=11
weights='distance'
ROC curves for individual models have been shown in figure 5.

5.2. Analysis

Random Forest gives the best accuracy, precision and AUC score. As the cost of false positive is high, we prefer to look at precision more than recall.

5.2.1 Random Forest

The reason why Random Forest performed the best is because there are not many categorical attributes in the dataset. A possible weakness of random forests is that they are biased in favor of attributes with more levels. With less categorical data, Random Forest performed better compared to other techniques.

5.2.2 Evaluation Metric

As has been analysed by recent studies in machine learning, AUC is considered a better metric for comparison between models than accuracy, as accuracy is highly dependent on how the classes are balanced. For this reason, we have focused more on AUC score.

5.2.3 Ensemble techniques

Ensemble technique combining SVM and Random Forest using soft voting gives good results. Reason could be that the data points incorrectly classified by SVM were correctly classified by Random Forest. SVM normally performs better for binary classification and as SVM relies on the concept of distance, pre-processing steps such as one-hot encoding and standardization improves performance of SVM. Random Forest is well suited for a data with a mixture of categorical and numerical. Our current dataset is a mixture of categorical and numerical and the problem at hand is a binary classification problem, and hence, combining SVM and Random Forest averaged out the probabilities and the points at which SVM didn't perform well were covered by Random Forest and vice versa.

5.2.4 Random Forest vs Neural Network

Neural Networks work well when the data available at hand is large, which isn't the case for loan prediction applications. For a small datasets, Neural Network can decimate the interpretability of the features to a point where they become meaningless [10]. Predicting if a loan should be approved or not requires the features to be meaningful, and

hence using Neural Networks makes the model more complex than it needs to be, which in turn decreases the performance.

5.2.5 Random Forest vs ADAboost

In random forests, too much complexity while training can lead to overfitting, and hence less features are suitable for Random Forests, which is the case here. Also, Random Forest is less affected by noise in the dataset and it generalizes better because the generalization error reaches a limit with an increasing number of trees being formed [11](according to the Central Limit Theorem). This reduces the variance, which leads to better performance and generalization.

5.2.6 ADAboost vs XGboost

XGboost performed worse than ADAboost for the same reason why Neural Networks performed worse than Random Forest. Also, tuning of hyper-parameters in XGboost algorithms is computationally extensive. Finding the right hyper-parameters for XGboost is difficult, and in our case, is not required.

5.2.7 Linear SVM vs Gaussian kernel based SVM

Linear SVM performs worse than Gaussian kernel based SVM, suggesting that the classification problem at hand is a non-linear problem.

5.2.8 SVM vs Logistic Regression

SVM and Logistic Regression both give similar AUC. SVM gives slightly better AUC score and gives a better Accuracy as compared to Logistic Regression as Support Vector can perform a trick (SVM with Gaussian Kernel) to classify non-linear models better than other models. The fact that SVM performs better than logistic regression indicates that the problem at hand is a non-linear classification, and for non-linear classification, SVM transforms input features into a higher dimensional space. In higher dimensions, classification becomes more efficient using SVM than in a lower dimension.

5.2.9 K-nearest neighbours vs SVM

K-nearest neighbours performed better than SVM as KNN is automatically non-linear and works well even if the number of data points are large. As can be seen above, the difference in the performance of non-linear SVM and linear SVM is not too large, this indicates that the data is high dimensional, but not by a huge amount. KNN performs well when there are a lot of data points in lower dimensions, as is in this case.

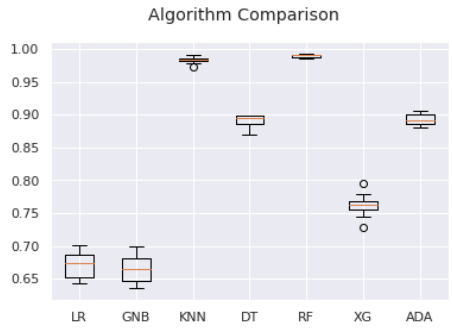

Figure 8. Algorithm Comparison

6. Conclusion

6.1. Conclusion

Loan approval is a crucial process in the banking industry as inability to repay the loan causes huge losses to the economy of the country. Hence, we tried developing a system that would automatically and efficiently predict the loan approval for people applying for loans.
In this project, we first clean the dataset and perform exploratory data analysis. The strategies to deal with both missing values and imbalanced data sets were discussed. The outliers were handled using boxplots and zscore. Then we propose nine machine learning models for prediction: Random Forest, Logistic Regression, Support Vector Machine, Decision Tree, ADAboost, XGboost, Neural Networks, K-nearest neighbours and Naive Bayes and experimented with some ensemble models. Through experiments it is found that Random Forest best fits the data set with highest accuracy and AUC score. We can also conclude that for loan prediction problem, using ensembeled techniques or complex techniques like neural networks and ADAboost doesn't give better performance than individual random forest model.

6.2. Work Left

We have completed all the work we planned to do and have provided an in depth analysis of different methods.

6.3. Individual Contribution

Chhavi Kirtani: Implementing the machine learning algorithms, Analysis of different methods, performed Pre-processing, Exploratory data analysis
Abhinay Gupta: Exploratory data analysis, Pre-processing, Neural Network implementation
Rachit Jaiswal: Neural Network implementation, Pre-processing
All work was equally divided and discussed together.

References

[4] indra90/predicting-loan-repayment Kaggle
[5] Chang Han (2019) 'Loan Repayment Prediction Using Machine Learning Algorithms' UCLA open access publications pp. 3-33
[6] Imad Dabbura (2018) 'predicting loan repayment' towardsdatascience
[7] Onel Harrison (2018) 'Machine Learning Basics with the K-Nearest Neighbors Algorithm' towardsdatascience
[8] Julia Nikulski (2020) 'The Ultimate Guide to AdaBoost, random forests and XGBoost' towardsdatascience
[9] Eijaz Alibhai (2018) 'Building A Deep Learning Model using Keras' towardsdatascience
[10] James Montentes (2020) '3 Reasons to Use Random Forest Over a Neural Network–Comparing Machine Learning versus Deep Learning' towardsdatascience
[11] Julia Nikulski (2020) 'The Ultimate Guide to AdaBoost, random forests and XGBoost' towardsdatascience