DATA MINING PROJECT REPORT

ABSTRACT

This problem is about loan defaulters forecasting. For this, I have used Lending Club Loan Dataset avaiable on Kaggle. Lending Club is a peer-to-peer lending platform.
The dataset contains loan data for all loans issued through the 2018-2020, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information.Using necessary data, I have developed a machine learning model to predict loan defaulters.
This ML model is able to predict 61% of the defaulters. It has also improved Return on Investment (ROI) of loans.

We establish that the rate and interest rate changes with grade and year loan default. Based on the following aggregations features:

i. Default rate by grade
ii. Default rate by year
iii. Interest rate by grade
iv. Interest rate by year

Indicates that default rate highly correlated with grade. Riskier the grade more is the chance to default default rate do not show any pattern with issue year, but it seems to relate with the economic conditions prevailing at that time. Also after 2019 the loans issued on site increased which also increased the loan defaulters. 2021 has peak default rate.
Interest rate is highly correlated with grade. Riskier the grade more is the interest rate.
Average interest rate is increasing. It is at peak in 2019 and 2020 which may be a reason for peak default rate in 2020.

Table of Contents

Summary of Features

1) **addr_state** -The state provided by the borrower in the loan application
2) **annual_inc**:- The self-reported annual income provided by the borrower during registration
3) **annual_inc_joint**-The combined self-reported annual income provided by the co-borrowers during registration
4) **application_type**- Indicates whether the loan is an individual application or a joint application with two co-borrowers
5) **collection_recovery_fee**-post charge off collection fee
6) **collections_12_mths_ex_med** -Number of collections in 12 months excluding medical collections
7) **delinq_2yrs**-The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
8) **desc-** Loan description provided by the borrower
9) **dti-** A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower's self-reported monthly income.
10) **dti_joint**- A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income
11) **earliest_cr_line** -The month the borrower's earliest reported credit line was opened
12) **emp_length**- Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
13) **emp_title** The job title supplied by the Borrower when applying for the loan.*
14) fico_range_high-The upper boundary range the borrower's FICO at loan origination belongs to.

15) **fico_range_low**-The lower boundary range the borrower's FICO at loan origination belongs to.
16) **funded_amnt**-The total amount committed to that loan at that point in time.
17) **funded_amnt_inv**-The total amount committed by investors for that loan at that point in time.
18) **Grade- LC** assigned loan grade
19) **home_ownership**-The home ownership status provided by the borrower during registration. Our values are: RENT, OWN, MORTGAGE, OTHER.
20) **Id- A unique LC** assigned ID for the loan listing.
21) **initial_list_status**-The initial listing status of the loan. Possible values are – W, F
22) inq_last_6mths-The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
23) **installment**-The monthly payment owed by the borrower if the loan originates.
24) **int_rate**-Interest Rate on the loan
25) is_inc_v Indicates if income was verified by LC, not verified, or if the income source **was verified**
26) issue_d-The month which the loan was funded
27) last_credit_pull_d-The most recent month LC pulled credit for this loan
28) **last_fico_range_high**-The upper boundary range the borrower's last FICO pulled belongs to.
29) **last_fico_range_low**-The lower boundary range the borrower's last FICO pulled belongs to.
30) **last_pymnt_amnt**-Last total payment amount received
31) **last_pymnt_d**-Last month payment was received
32) **loan_amnt**-The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value.
33) **loan_status**-Current status of the loan
34) **member_id**-A unique LC assigned Id for the borrower member.
35) **mths_since_last_delinq**-The number of months since the borrower's last delinquency.
36) **mths_since_last_major_derog**-Months since most recent 90-day or worse rating
37) **mths_since_last_record**-The number of months since the last public record.
38) **next_pymnt_d**-Next scheduled payment date
39) **open_acc** The number of open credit lines in the borrower's credit file.
40) **out_prncp**-Remaining outstanding principal for total amount funded
41) **out_prncp_inv**-Remaining outstanding principal for portion of total amount funded by investors
42) **policy_code**-publicly available policycode=1 new products not publicly available policycode=2
43) **pub_rec**-Number of derogatory public records
44) **purpose-**A category provided by the borrower for the loan request.
45) **pymnt_plan**-Indicates if a payment plan has been put in place for the loan
46) recoveries-post charge off gross recovery
47) **revol_bal** Total credit revolving balance
48) **revol_util**-Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
49) sub_grade-LC assigned loan subgrade

50) term-The number of payments on the loan. Values are in months and can be either 36 or 60.
51) Title-The loan title provided by the borrower
52) total_acc The total number of credit lines currently in the borrower's credit file
53) total_pymnt-Payments received to date for total amount funded
54) total_pymnt_inv-Payments received to date for portion of total amount funded by investors
55) total_rec_int-Interest received to date
56) total_rec_late_fee-Late fees received to date
57) total_rec_prncp-Principal received to date
58) url-URL for the LC page with listing data.
59) verified_status_joint-Indicates if the co-borrowers' joint income was verified by LC, not verified, or if the income source was verified
60) zip_code The first 3 numbers of the zip code provided by the borrower in the loan application.
61) open_acc_6m-Number of open trades in last 6 months
62) open_il_6m-Number of currently active installment trades
63) open_il_12m-Number of installment accounts opened in past 12 months
64) open_il_24m-Number of installment accounts opened in past 24 months
65) mths_since_rcnt_il Months since most recent installment accounts opened
66) total_bal_il-Total current balance of all installment accounts
67) il_util-Ratio of total current balance to high credit/credit limit on all install acct
68) open_rv_12m-Number of revolving trades opened in past 12 months
69) open_rv_24m-Number of revolving trades opened in past 24 months
70) max_bal_bc-Maximum current balance owed on all revolving accounts
71) all_util Balance to credit limit on all trades
72) total_rev_hi_lim Total revolving high credit/credit limit
73) inq_fi Number of personal finance inquiries
74) total_cu_tl Number of finance trades
75) inq_last_12m Number of credit inquiries in past 12 months
76) acc_now_delinq The number of accounts on which the borrower is now delinquent.
77) tot_coll_amt Total collection amounts ever owed
78) tot_cur_bal Total current balance of all accounts

Data Cleaning and Preprocessing

Data preprocessing involves the transformation of the raw dataset into an understandable format. Preprocessing data is a fundamental stage in data mining to improve data efficiency. The data preprocessing methods directly affect the outcomes of any analytic algorithm. Data preprocessing is generally carried out in 7 simple steps:

Steps In Data Preprocessing:

1. Gathering the data
2. Import the dataset & Libraries
3. Dealing with Missing Values
4. Divide the dataset into Dependent & Independent variable
5. dealing with Categorical values
6. Split the dataset into training and test set
7. Feature Scaling

For this, I proceeded with Data cleaning and preprocessing as follows

i. Selected only Fully Paid and Defaulted loans for training model.
ii. Train-Test Split
iii. Created a pipeline for data preprocessing. See 'Data Cleaning' notebook for detailed approach.
iv. Standard Scaling
v. Over-Sampling using SMOTE : I have not used this in final model, as ML models I have used consider class imbalance.
vi. I have not removed features with high VIF, as model is uderfitting and performing well after including these features

Load the data

- Selected only Fully Paid and Defaulted loans for training model.
- Train-Test Split
- Created a pipeline for data preprocessing. See 'Data Cleaning' notebook for detailed approach.
- Standard Scaling
- Over-Sampling using SMOTE : I have not used this in final model, as ML models I have used consider class imbalance.
- I have not removed features with high VIF, as model is uderfitting and performing well after including these features.

```
data=pd.read_csv('loan.csv')

# Select only default or paid loans. Ignore current loans
default=['Charged Off','Late (31-120 days)','Default',
            'Does not meet the credit policy. Status:Charged Off','Late (16-30 days)',
               'In Grace Period']
paid=['Fully Paid',
             'Does not meet the credit policy. Status:Fully Paid']

data['loan_status']=data['loan_status'].apply(lambda x: 'Default' if x in default else x)
```

i. Load dataset using pandas read_csv method in variable df and give file path as filepath_or_buffer=path, compression='zip'and low_memory = False
ii. Store all the features(independent values) in a variable called X
iii. Store the target variable (loan_status) in a variable called y
iv. Split the dataframe into X_train,X_test,y_train,y_test using train_test_split() function. Use test_size = 0.25 and random_state = 4
v.

Data cleaning

When working with multiple data sources, there are many chances for data to be incorrect, duplicated, or mislabeled. If data is wrong, outcomes and algorithms are unreliable, even though they may look correct. *Data cleaning* is the process of changing or eliminating garbage, incorrect, duplicate, corrupted, or incomplete data in a dataset. There's no such absolute way to describe the precise steps in the data cleaning process because the

i. Find the sum of null values for each column and store it in a variable col
ii. Find the features with more than 61% missing data and add it to a variable col_drop
iii. Find columns which contains only one unique value using.nunique() and append it to the variable col_drop
iv. Drop the features stored in the variable col_drop from X_trainand X_test
v. After initial data cleaning, we're left with 30 columns. Now I did analysis of each column

Multiclass to binary class

Observation

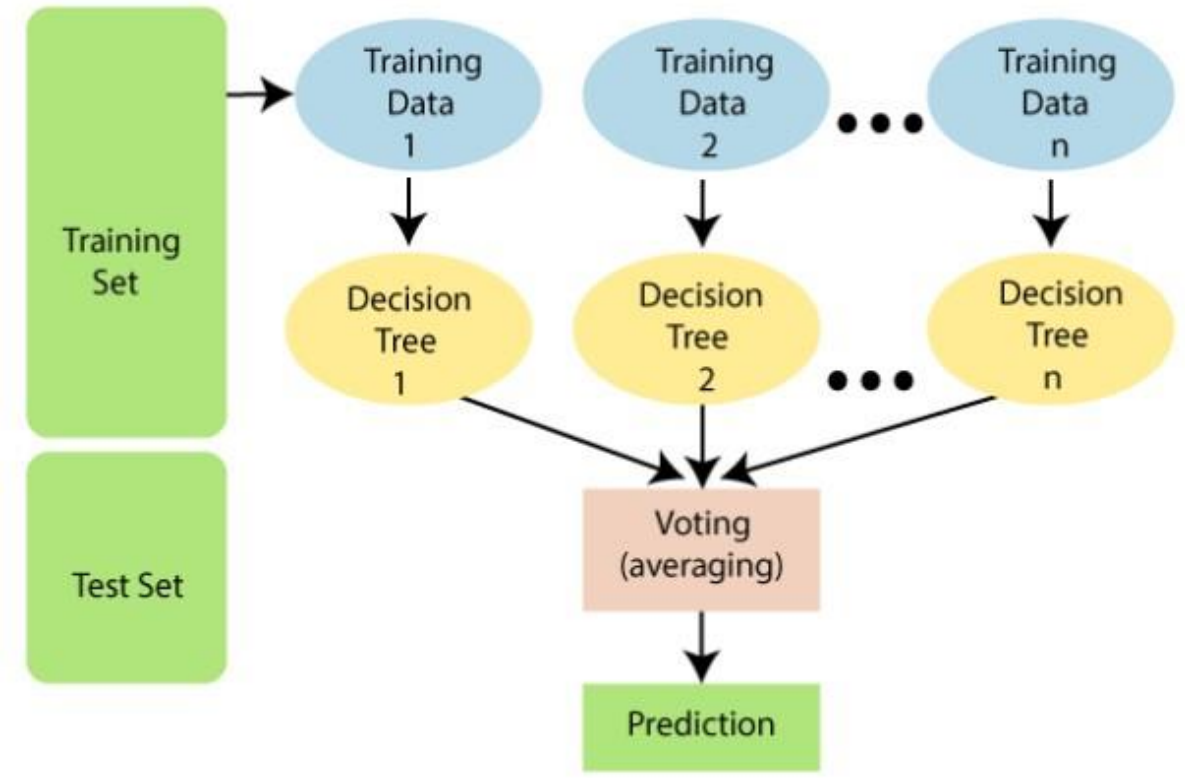
- I calculated percent null values in each column.
- Columns with null values : 'emp_length', 'revol_util', 'collections_12_mths_ex_med', 'tot_coll_amt', 'tot_cur_bal', 'total_rev_hi_lim'

- As 'emp_length' is ordinal variable, I encoded it according to value instead of creating dummy binary variables. 'emp_length' less that one year was set to 0 and that greater than 10 years was set to 10.
- 'tot_coll_amt','tot_cur_bal','total_rev_hi_lim' have missing values till Sept-2019 date.
- Other variables do not show any pattern in missing values
- We thus find that in the target variable loan_status there are six classes. We want to convert these six classes to two classes. So we put Fully Paid and Current as one class as it is very unlikely that these will be defaulters and the rest to the other class.
- INPUT
- df.loan_status.value_counts()

The numerical variables are stored in num and categorical variables are stored in cat.
Fill the missing values in X_train and X_test with mean for numerical variable and mode for categorical variables.
Label Encode categorical variables in X_train and X_test.

Supervised Learning (Random Forest)

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

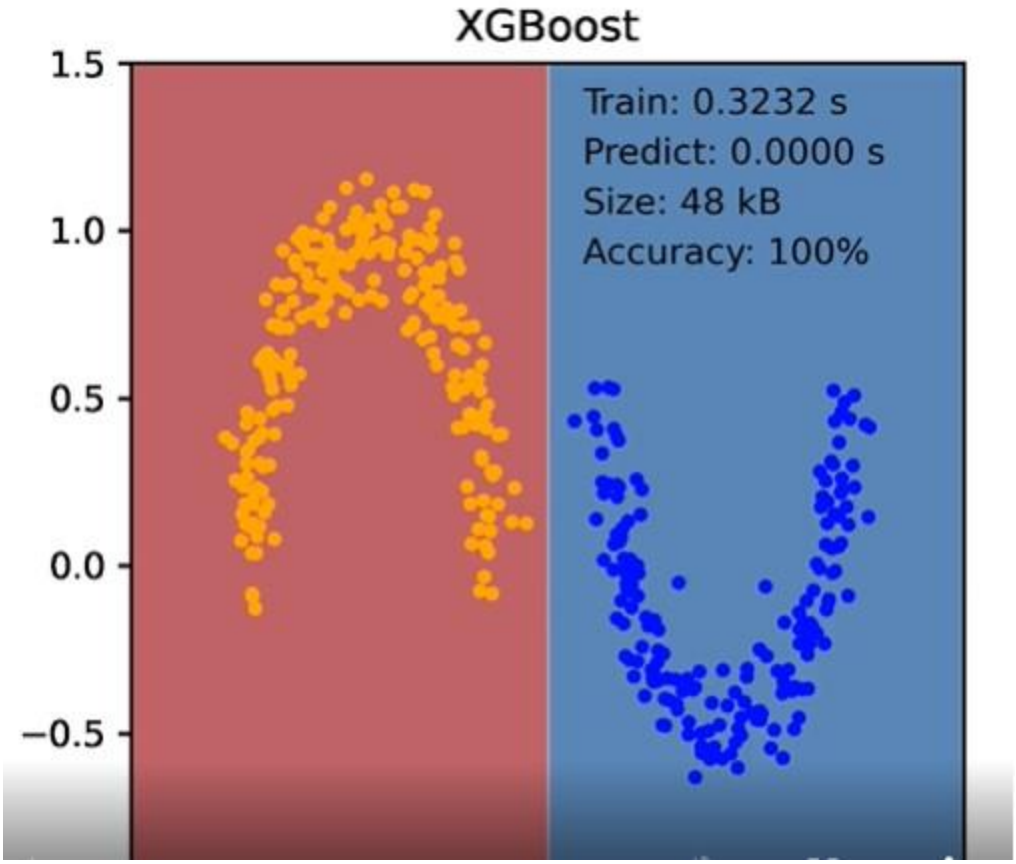


As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.For this, project, I proceeded to implement the algorithm as follows:

i. Instantiate RandomForestClassifier to a variable rf with random_state =42,max_depth=2andmin_samples_leaf=5000.
ii. Fit the model on X_train and y_train.
iii. Store the f1_score in variable f1, precision_score in variable precision, recall_score in variable recall and roc_auc_scorein roc_auc. **Note: Write rocaucscore(ytest, ypred)
iv. Print the confusion_matrix and classification_report.
v. Predict the probability for the X_test == 1 and store the result in y_pred_proba.
vi. Use metrics.roc_curve to calculate the fpr and tpr and store the result in variables fpr, tpr,.
vii. Calculate the auc score of y_test and y_pred_proba and store it in variable called auc.
viii. Plot auc curve of 'auc' using the line plt.plot(fpr,tpr,label="Random Forest model, auc="+str(auc)).

Unspervised Learning (XGBoostClassifier)

XGBoost is a gradient boosting package that implements a gradient boosting framework. The algorithm is scalable for parallel computing. In addition to Python, it is available in C++, Java, R, Julia, and other computational languages. XGBoost has gained attention in machine learning competitions as an algorithm of choice for classification and regression.



i. Instantiate XGBoostClassifier and store it to a variable xgb with parameter learning_rate = 0.0001
ii.
iii. Fit the model on X_train and y_train.
iv. Store the values predicted by model for X_test in a variable y_pred.
v. Store the f1_score in variable f1, precision_score in variable precision, recall_score in variable recall and roc_auc_scorein roc_auc.
vi. Print the confusion_matrix and classification_report.
vii. Predict the probability for the X_test == 1 and store the result in y_pred_proba.
viii. Use metrics.roc_curve to calculate the fpr and tpr and store the result in variables fpr, tpr,_.
ix. Calculate the auc score of y_test and y_pred_proba and store it in variable called auc.
x. Plot auc curve of 'auc' using the line plt.plot(fpr,tpr,label="XGBoost model, auc="+str(auc)).

Model Evaluation and Testing

Evaluating Model (On Test Set)

Model is able to predict 61% of the defaulters.
When we look at in the below we can appreciate how the **Return on investment (ROI)** we see how model can help in real business.

To calculate ROI, I've used total loan repayed and amount funded as loan. These variables leak future information and I have not used these variables for training. Data used to calculate ROI includes fully paid, charged off and loans in grace period. It does not include Current loans.

- ROI = 'total_payment' / 'funded_amount
- ROI without using model : -4.57
- ROI after using model : 2.22

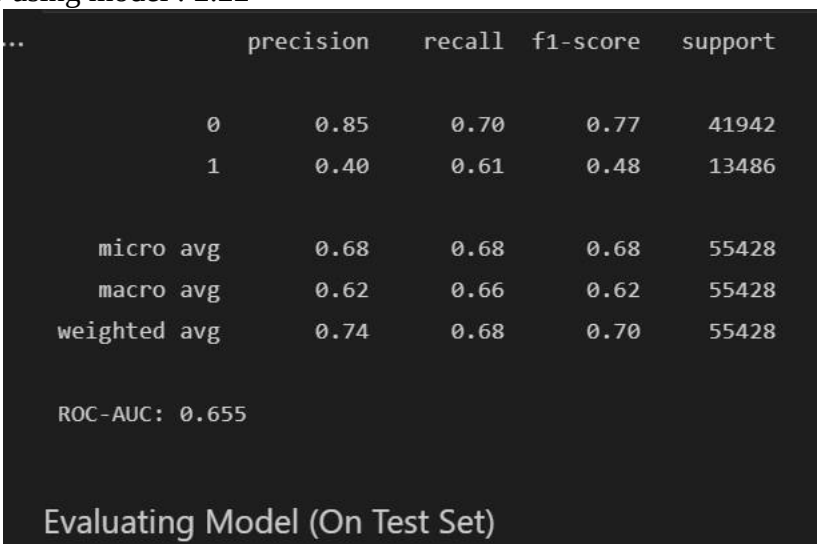


Figure 1

Learning Curve

Learning Curve and Scope for Improvement

From learning curve, it is clear that model is the under-fit. Even if we supply more data, it will improve model minimally.
Also confusion matrix printed above shows that 30% of the fair loans were predicted as default. This model can be improved by adding more features.

Grade-wise Analysis of ROI:

A table below shows grade-wise analysis of loans. It shows ROI, % of loans selected and % of loans defaulted from selected loans.

	ROI	% Picked	% Default	ROI_w/o_model	% Picked_w/o_model	% Default_w/o_model
A	0.0250712	99.9774	8.64002	0.025071	100	8.63806
B	0.0200738	96.1866	16.1057	0.0152713	100	16.7564
C	0.017765	53.9985	18.7983	-0.0447373	100	25.9779
D	0.0349775	24.597	20.8825	-0.096151	100	34.7948
E	0.0703003	7.08117	22.2561	-0.143831	100	42.5518
F	-0.0234227	3.26508	22.0339	-0.16447	100	47.316
G	0.0961287	2.6694	38.4615	-0.163535	100	

* ROI for every grade has improved.
* ROI for grades C, D, E, G became positive after using model.

Conclusion

We have achieved the general goal of using supervised and unsupervised learning to predict the rate of loan defaulters using Random Forest, Gradient Bosting Algorithm and the Kmeans classifier. We have established that default rate highly correlated with grade. Riskier the grade more is the chance to default default rate do not show any pattern with issue year, but it seems to relate with the economic conditions prevailing at that time.

Also after 2019 the loans issued on site increased which also increased the loan defaulters. 2021 has peak default rate.
Interest rate is highly correlated with grade. Riskier the grade more is the interest rate.
Average interest rate is increasing. It is at peak in 2019 and 2020 which may be a reason for peak default rate in 2020

References

1.1. Vishal Morde. (2019, April 8). XGBoost Algorithm
https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

1.2. Aburashid Yusuf. (2021, September 9). Lending Club Loan
https://www.kaggle.com/faressayah/lending-club-loan-defaulters-prediction

1.3. Basil Saji (2018, December 19)

https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/#h2_5

1.4. Javapoint.com (2021, October 11). K-Means Clustering Algorithm
https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning