# Data Science Analysis
# Loan default prediction

Mukunda Reddy
ai21btech1021

## Introduction

Delving into this case study presents a profound exploration of real-world problem-solving through the synergy of Exploratory Data Analysis and Machine Learning within the realm of financial services. Specifically, it offers a unique lens into risk analytics, a cornerstone in the banking and financial sector, elucidating how data-driven methodologies are leveraged to mitigate financial risks inherent in lending operations.

Within this narrative, LendingClub emerges as a pioneering force, headquartered in the bustling tech hub of San Francisco, California. Its groundbreaking initiatives include being the first peer-to-peer lending entity to attain SEC registration for its offerings, alongside pioneering loan trading on a secondary market. As the global leader in its domain, LendingClub embodies the convergence of technological innovation and financial acumen, epitomizing the transformative potential of fintech.

## Dataset

### Data Collection

The dataset, available at the provided link , encapsulates a comprehensive repository of loan information curated over several years, dating back to the inception of the lending company. It meticulously documents various attributes pertinent to loans, including but not limited to borrower details, loan terms, repayment history, and crucially, the classification of loan defaulters. This rich dataset serves as a treasure trove for understanding the dynamics of lending practices and assessing risk factors associated with borrowers. Its longevity and depth enable profound insights into the evolution of lending trends and patterns over time.

Here's an summary of all attributes and their meanings in the dataset:

| **LoanStatNew** | **Description** |
|---|---|
| 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. |
| term | The number of payments on the loan. |
| int_rate | Interest Rate on the loan |
| installment | The monthly payment owed by the borrower if the loan originates. |
| grade | LC assigned loan grade |
| sub_grade | LC assigned loan subgrade |
| emp_title | The job title supplied by the Borrower when applying for the loan.* |
| emp_length | Employment length in years. |
| home_ownership | provided by the borrower during registration or obtained from the credit report. |
| annual_inc | The self-reported annual income provided by the borrower during registration. |
| verification_status | Indicates if income was verified by LC, not verified, or if the income source was verified |
| issue_d | The month which the loan was funded |
| loan_status | Current status of the loan |
| purpose | A category provided by the borrower for the loan request. |
| title | The loan title provided by the borrower |
| addr_state | The state provided by the borrower in the loan application |
| 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. |
| earliest_cr_line | The month the borrower's earliest reported credit line was opened |
| open_acc | The number of open credit lines in the borrower's credit file. |
| pub_rec | Number of derogatory public records |
| revol_bal | Total credit revolving balance |
| revol_util | Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit. |
| total_acc | The total number of credit lines currently in the borrower's credit file |
| initial_list_status | The initial listing status of the loan. |
| application_type | Indicates whether the loan is an individual application or a joint application with two co-borrowers |
| mort_acc | Number of mortgage accounts. |
| pub_rec_bankruptcies | Number of public record bankruptcies |

## Exploratory Data Analysis

In examining the loan-status column, it becomes evident that the dataset presents an inherent imbalance. Specifically, the distribution reveals a substantial discrepancy between the counts of 'Fully Paid' instances, totaling 318,357, and 'Charged Off' instances, amounting to 77,673. This imbalance necessitates careful consideration during the classification process to ensure equitable representation and accurate predictive performance.

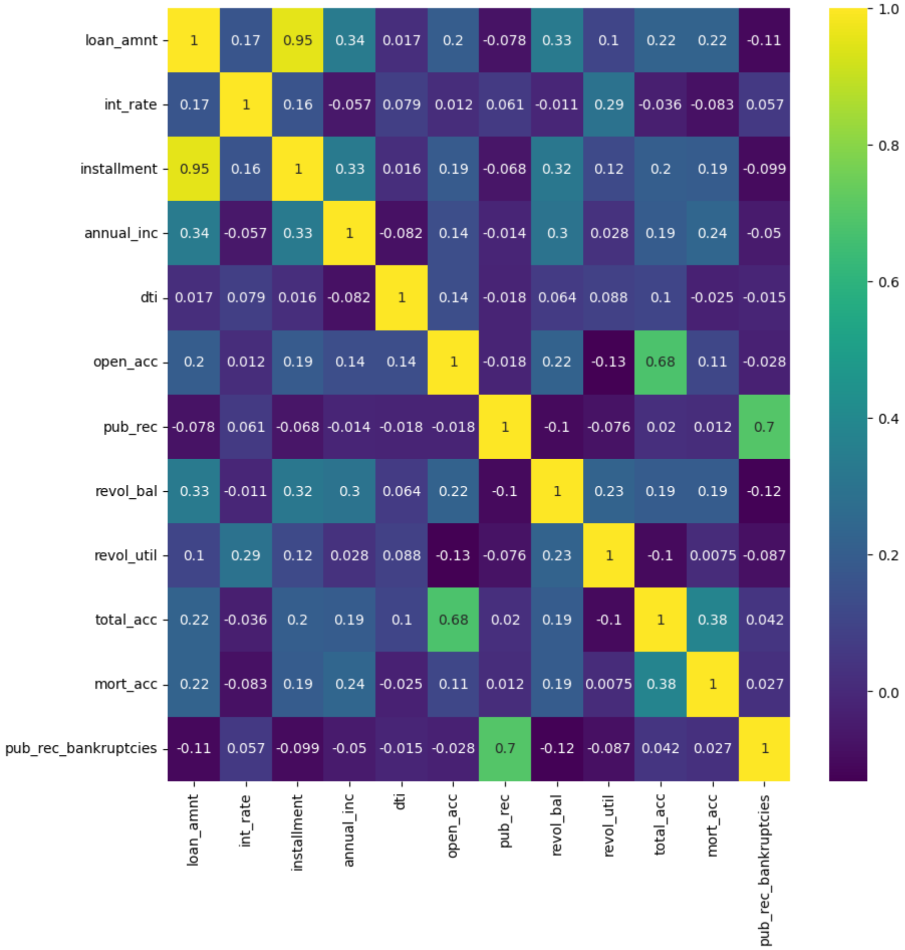


Figure 1: Pearson Correlation among numerical features

Upon examining the numerical features within the dataset, we observe correlations among them, shedding light on potential relationships and dependencies. Notably, we can see there is strong correlation between installment and loan-amount which is understandable.also there is slight correlation between pub-rec and pub-rec-bankrupcies.

For each categorical variable, it is imperative to discern the nature and diversity of its unique categories. Here's a summary of the distinct categories present in the dataset for each categorical variable:

| **Column** | **Unique Values** |
|---|---|
| term | [’36 months’, ’60 months’] |
| grade | [’B’, ’A’, ’C’, ’E’, ’D’, ’F’, ’G’] |
| sub_grade | [’B4’, ’B5’, ’B3’, ’A2’, ’C5’, ’C3’, ’A1’, ’B2’, ’C1’, ’A5’, ’E4’, ’A4’, ’A3’, ’D1’, ’C2’, ’B1’, ’D3’, ’D5’, ’D2’, ’E1’, ’E2’, ’E5’, ’F4’, ’E3’, ’D4’, ’G1’, ’F5’, ’G2’, ’C4’, ’F1’, ’F3’, ’G5’, ’G4’, ’F2’, ’G3’] |
| emp_title | [’Marketing’, ’Credit analyst ’, ’Statistician’, ... ’Michael’s Arts & Crafts’, ’licensed bankere’,’Gracon Services’] |
| emp_length | [’10+ years’, ’4 years’, ’< 1 year’, ’6 years’, ’9 years’, ’2 years’ ’3 years’, ’8 years’, ’7 years’, ’5 years’, ’1 year’, nan] |
| home_ownership | [’RENT’, ’MORTGAGE’, ’OWN’, ’OTHER’, ’NONE’, ’ANY’] |
| verification_status | [’Not Verified’, ’Source Verified’, ’Verified’] |
| initial_list_status | [’w’, ’f’] |
| application_type | [’INDIVIDUAL’, ’JOINT’, ’DIRECT_PAY’] |
| loan_status | [’Fully Paid’, ’Charged Off’] |
| purpose | [’vacation’, ’debt_consolidation’, ’credit_card’, ’home_improvement’, ’small_business’, ’major_purchase’, ’other’, ’medical’, ’wedding’, ’car’, ’moving’, ’house’, ’educational’, ’renewable_energy’] |
| title | [’Vacation’, ’Debt consolidation’, ’Credit card refinancing’, ... ’Credit buster ’, ’Loanforpayoff’, ’Toxic Debt Payoff’] |
| earliest_cr_line | [’Jun-1990’, ’Jul-2004’, ... ] |
| address | [’0174 Michelle Gateway Mendozaberg, OK 22690’, ’1076 Carney Fort Apt. 347 Loganmouth, SD 05113’, ...] |

Additional analysis is conducted within the codebase to explore the relationship between the categorical variables and our focal target variable, loan_status, with the aim of discerning their potential utility in predictive modeling. This scrutiny seeks to ascertain whether these categorical attributes harbor any discernible predictive value concerning loan status.

# Preprocessing Data

## Missing Values

| **Column** | **Missing Values (%)** |
|---|---|
| emp_title | 5.789208 |
| emp_length | 4.621115 |
| title | 0.443148 |
| revol_util | 0.069692 |
| mort_acc | 9.543469 |
| pub_rec_bankruptcies | 0.135091 |

- Removing the `title` column due to redundancy.

- Eliminating the `emp_length` column, as its predictive value has been deemed negligible in prior analysis.
- Dropping the `emp_title` column due to its excessively large number of unique values, leading to computational inefficiency during encoding.
- Discarding the `revol_util` and `pub_rec_bankruptcies` columns since the proportion of missing values is less than 0.2%.
- Addressing missing values in the `mort_acc` column by imputing them with expected values based on the corresponding `total_acc` category.
- Removing the `grade` column, as its information is effectively captured by the more granular `sub_grade` column.
- Omitting the `issue_d` column, as it pertains to loan issuance and is thus irrelevant for predictive modeling.

## Categorical to numerical convertion

In data preprocessing, we extracted only the pincode from the 'address' column. The 'term' feature was recoded into integers, with its two categories replaced by 36 and 60. For 'loan_status,' we adopted a binary encoding scheme, assigning 1 to 'Fully Paid' and 0 to 'Charged Off,' aligning with our classification labels.

As for 'sub_grade,' we employed ordinal encoding due to the inherent order among its categories.

Moreover, 'verification_status,' 'address', 'purpose', 'initial_list_status', 'application_type', and 'home_ownership' underwent one-hot encoding using 'get_dummies', ensuring a comprehensive representation of categorical information.

## Resampling

Initially we have the label distribution as majority : 222340,minority : 54313

1. **Initial Split**: The original dataset is split into training and testing sets using a 70-30 ratio, with a random state of 42 for reproducibility.
2. **Identification of Majority and Minority Classes**: The majority and minority classes within the training set are identified based on the 'loan_status' column.
3. **Downsampling of Majority Class**: The majority class instances are downsampled to approximately two-thirds of their original count without replacement, ensuring a balanced representation.
4. **Upsampling of Minority Class**: The minority class instances are upsampled to 1.2 times their original count with replacement to augment their representation.
5. **Concatenation and Randomization**: The downsampled majority and upsampled minority sets are concatenated into a single resampled training dataset, which is then randomized to shuffle the instances.

6. **Observing Class Distribution**: The counts of each class in the resampled training set are printed to verify the effectiveness of the resampling technique.

7. **SMOTE Oversampling**: Further oversampling is applied using the Synthetic Minority Over-sampling Technique (SMOTE) to enhance the minority class representation. The sampling_strategy parameter controls the ratio of minority to majority class after resampling.in this case we took 80/100 ratio.

After doing resampling on the imbalanced dataset we have Minority samples : 118580, Majority samples:148226, Next we normalize the dataset.

# Models

## Logistic regression with L2 regularization

Hyperparameter Tuning: we perform hyperparameter tuning for a logistic regression model with L2 regularization (ridge regularization). The hyperparameter being tuned is the regularization strength $C$, which controls the inverse of the regularization parameter. A list of $C$ values is specified as $C\_values = [0.01, 0.1, 1, 5, 10]$.

Cross-Validation: For each value of $C$ in the $C\_values$ list, a LogisticRegression model is created with the corresponding $C$ value and the penalty set to $'l2'$ (ridge regularization) and solver set to $'liblinear'$ (a solver suitable for small datasets). The model's performance is evaluated using 5-fold cross-validation on the training data ($x\_train$ and $y\_train$).

Best Hyperparameter Selection: After evaluating all $C$ values, we find the best $C$ value A new LogisticRegression model is created with the best_C value, and it is fitted on the entire training data ($x\_train$ and $y\_train$).

Model Evaluation: The plot_roc_and_confusion_matrix function is called, which plots the receiver operating characteristic (ROC) curve and the confusion matrix for the trained model on the test data ($x\_test$ and $y\_test$).

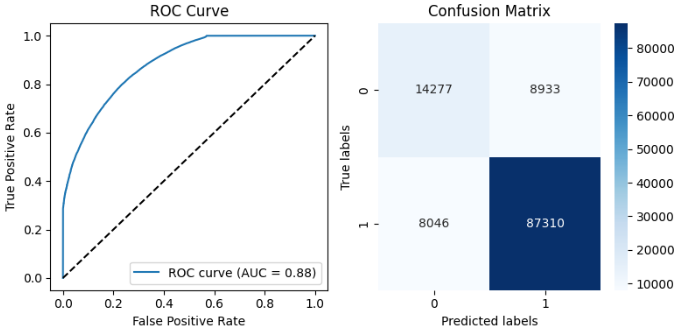


Figure 2: Logistic regression

The ROC analysis indicates a favorable area under the curve(AUC) value of 0.88, suggesting robust performance of the model, particularly concerning label 1. This label benefits from an ample number of samples, enabling the model to learn and generalize effectively. Conversely, label 2 exhibits a relatively lower accuracy.To address this class imbalance and enhance performance on label 2, techniques such as sample generation and bootstrap sampling explored. In summary, while the model performs admirably on label 1, efforts to bolster the representation of label 2 through targeted sampling techniques further enhance overall predictive performance.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.64 | 0.62 | 0.63 | 23210 |
| 1 | 0.91 | 0.92 | 0.91 | 95356 |
| accuracy | | | 0.86 | 118566 |
| macro avg | 0.77 | 0.77 | 0.77 | 118566 |
| weighted avg | 0.85 | 0.86 | 0.86 | 118566 |

## Random Forest Classification

performed hyperparameter tuning for a Random Forest Classifier using GridSearchCV. The hyperparameters being tuned are the number of trees in the forest (n_estimators) and the maximum depth of each tree (max_depth).

The grid search was performed by calling the fit method of the GridSearchCV object, passing the training data (*x_train* and *y_train*). This method exhaustively evaluated all parameter combinations in the grid using 4-fold cross-validation and selected the best combination based on the highest mean accuracy score across the folds.

Finally, created a new RandomForestClassifier instance, passing the best hyperparameter values obtained from the grid search. This new model was then fitted on the entire training data (*x_train* and *y_train*) using the optimized hyperparameters.

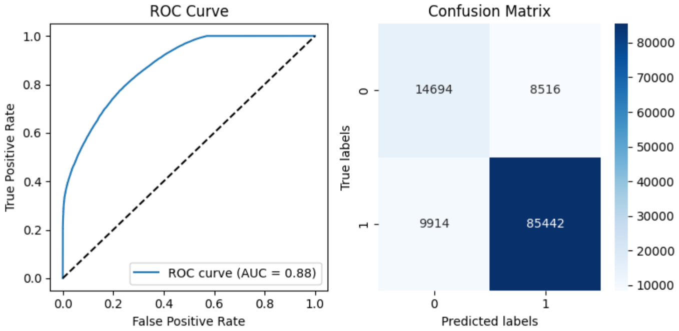


Figure 3: Random Forest

The AUC score of 0.88 achieved by the model closely aligns with that of the logistic regression classifier, indicating comparable performance between the two models. This similarity is further corroborated by the confusion matrix analysis, where the distribution of predicted labels closely mirrors that of the logistic regression model.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.60 | 0.63 | 0.61 | 23210 |
| 1 | 0.91 | 0.90 | 0.90 | 95356 |
| accuracy | | | 0.84 | 118566 |
| macro avg | 0.75 | 0.76 | 0.76 | 118566 |
| weighted avg | 0.85 | 0.84 | 0.85 | 118566 |

## XGBoost

we instantiated an XGBClassifier with following parameters n_estimators=100: This parameter sets the number of decision trees to be used in the XGBoost ensemble model. The higher the number of estimators, the more complex the model can become, potentially leading to better performance but also risking overfitting.

After creating the XGBClassifier instance with the specified parameters, we call the fit method and pass the training data (x_train and y_train). During the training process, XGBoost iteratively constructs an ensemble of decision trees, where each new tree is trained to correct the errors made by the previous trees. This process continues until the specified number of estimators (100 in this case) is reached or another stopping criterion is met.

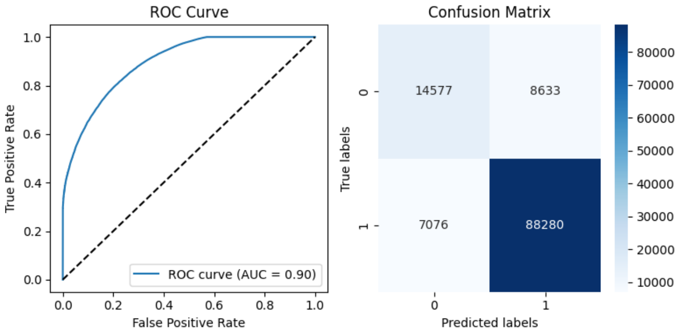


Figure 4: xgboost

The XGBoost model achieves a superior AUC score of 0.90 compared to the aforementioned logistic regression and random forest models. This improvement is evident not only in the AUC metric but also reflected in the confusion matrix analysis, where the XGBoost model demonstrates enhanced predictive performance.

One key factor contributing to this improvement is the inherent sophistication and optimization techniques embedded within the XGBoost algorithm. Unlike random forest, XGBoost employs a gradient boosting framework that iteratively improves upon the weaknesses of preceding models, thereby refining its predictive accuracy.

Additionally, the parameter configuration chosen for XGBoost, particularly the decision to expand the ensemble with 100 trees, contributes significantly to its superior performance. By increasing the number of trees in the ensemble, XGBoost can capture more nuanced patterns and relationships within the data, leading to improved generalization and discrimination between classes.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.67 | 0.63 | 0.65 | 23210 |
| 1 | 0.91 | 0.93 | 0.92 | 95356 |
| accuracy | | | 0.87 | 118566 |
| macro avg | 0.79 | 0.78 | 0.78 | 118566 |
| weighted avg | 0.86 | 0.87 | 0.87 | 118566 |

# Conclusion

While comparing the accuracy scores of the three models (XGBoost - 0.87, logistic regression - 0.86, and random forest - 0.84) may suggest a ranking in terms of performance, it's essential to acknowledge that accuracy alone might not provide a comprehensive evaluation, particularly in the presence of imbalanced datasets. Models may exhibit inflated accuracy if they disproportionately predict the majority class.

In scenarios where one class dominates the dataset, accuracy might not effectively capture the model's true predictive ability, especially if it tends to favor the majority class. Thus, while accuracy serves as a primary metric, it's imperative to complement it with other evaluation measures, such as precision, recall, and F1-score, to gain a more nuanced understanding of the model's performance, especially concerning minority classes.

In summary, while accuracy provides an initial insight into model performance, it's crucial to interpret it alongside other evaluation metrics, particularly in imbalanced datasets, to obtain a more accurate assessment of a model's effectiveness.

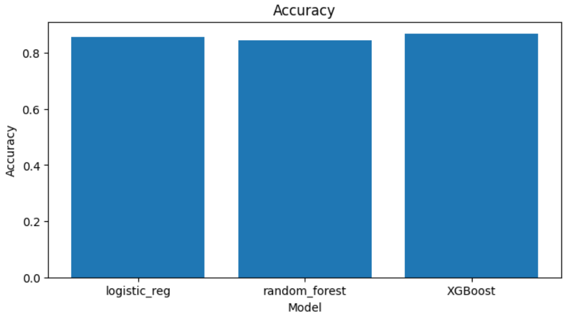


Figure 5: comparing accuracy

Upon examining the precision and recall scores for individual classes, we observe the following insights:

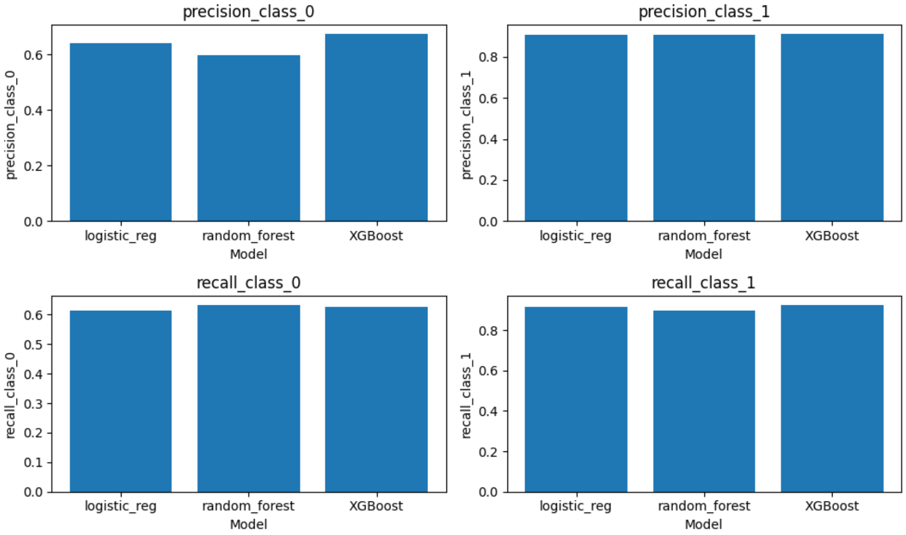
$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

For label 1 (majority class), all three models demonstrate comparable precision and recall scores, hovering around 0.91 and 0.92, respectively. This parity suggests that all models exhibit consistent performance in predicting instances belonging to the majority class.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Conversely, for label 0 (minority class), XGBoost outperforms logistic regression and random forest in terms of precision, followed by logistic regression and then random forest. This indicates that XGBoost exhibits superior precision in correctly identifying instances of the minority class.

In terms of recall, logistic regression and random forest yield similar performance, while logistic regression slightly lags behind. Despite this minor discrepancy, XGBoost maintains its lead in precision for the minority class.



Overall, considering both precision and recall, as well as the notably faster training time of XGBoost compared to the other models, XGBoost emerges as the preferred choice for this classification task. Its superior performance, particularly in identifying instances of the minority class, underscores its effectiveness in handling imbalanced datasets and maximizing predictive accuracy.

The codebase can be found Here.