Course: **T301 Data Mining**

Project: **Credit Risk**

Introduction

Problem Statement: Lending involves the provision of funds, often in the form of a loan, to individuals, businesses, or other entities, with the anticipation that the borrowed capital will be returned with interest or in adherence to the conditions laid out in a loan agreement. This financial practice is a widespread and pivotal element of the economy.

It critical to address the loan defaulting risk: Loan default takes place when a borrower does not adhere to the specified terms and conditions outlined in the loan agreement. This often involves a failure to make timely payments or a breach of other contractual commitments. The perpetual chance that a borrower might fail to meet or cease making payments leads to accurately assessing credit risk and it is crucial.

Probable solution: Credit risk analytics transforms both historical and projected data into actionable analytical insights, empowering financial institutions to evaluate risk and determine lending and account management strategies. One approach is the integration of credit risk modeling into their decision-making processes.

Plan to explain the to stake holders he solution: To Explain the integration of credit risk modelling into their decision-making processes

Explain what credit risk is all about and detail the risk of loss due to a borrower's failure to repay a loan or meet contractual obligations.

Outline how the current Credit Risk is being assessed using SME knowledge or under writer's review process and natural biases

Explain the prediction or risk analysis gap that can be addressed by using data analytics risk modelling.

Provide a high-level modeling task such as collecting data, performing Exploratory Data Analysis (EDA), utilizing predictive classification models, Validation and refining the same as the new data is received

List the input data that will be used (both existing, new and derived ones like)

Point out the benefits of using the model like

- Enhanced Decision-making
- Reducing Losses
- Improving profitability
- Staying ahead of competition
- Automation and Scalability
- Meeting regulatory obligations by eliminating bias and reducing human errors

- Staying consistent and maintaining integrity
- Evolving solution that adapts to the new data
- Vetted solution, how the model will be implemented in a Phased approach, merging model process with the existing process and cut over
- Call out automation across the business / industry domain and the need to improvise
- What is the Return of investment, like reducing bad debt, optimizing loan pricing and improving investors' confidence?
- how credit risk aligns with the organization's vision or goals if applicable.

Dataset Identified: Dataset from the Kaggle is identified Credit Risk Dataset

Feature Name	Description
person_age	Age of the person
person_income	Annual Income of the person
person_home_ownership	Home ownership - whether person owns a homeor mortage or rented
person_emp_length	Employment length (in years)
loan_intent	Loan intent - intent of the loan
loan_grade	Loan grade - sclae based on credit worthiness
loan_amnt	Loan amount
loan_int_rate	Interest rate for the loan
loan_status	Loan status - credit approval
loan_percent_income	Percent income - % of income represented by loan
cb_person_default_on_file	Historical default (previous default history from Bureau)
cb_preson_cred_hist_length	Credit history length

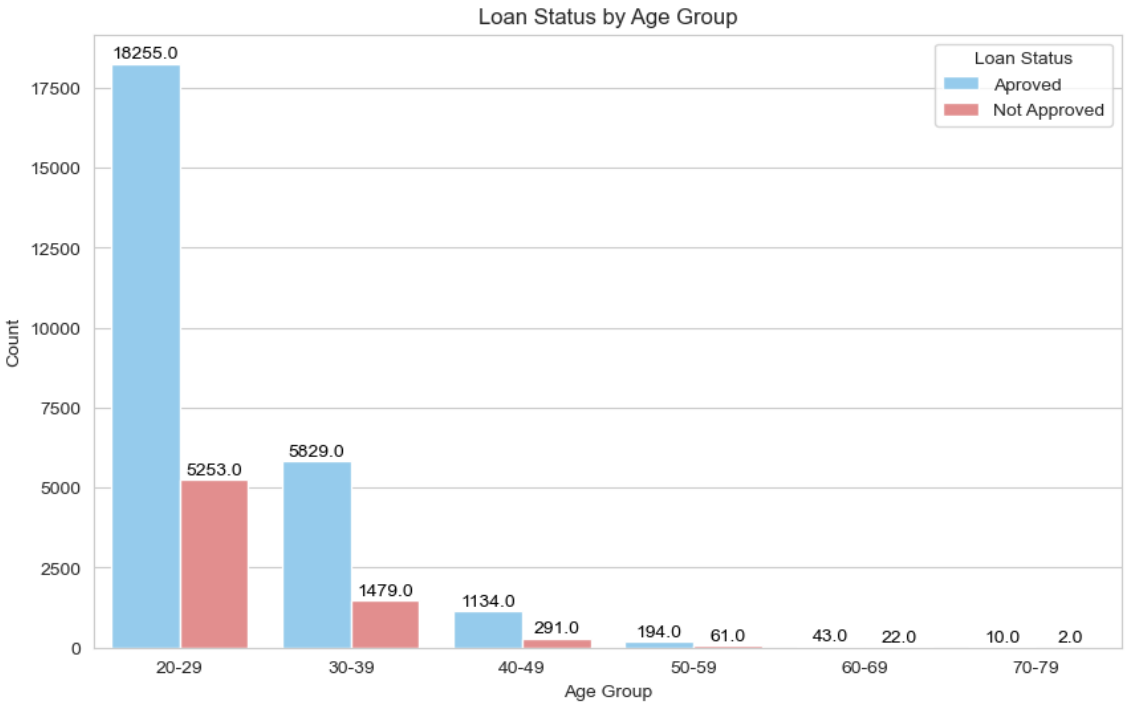
This is a simulated data based on credit bureau data https://www.kaggle.com/datasets/laotse/credit-risk-dataset/data

Summary of Milestones 1-3

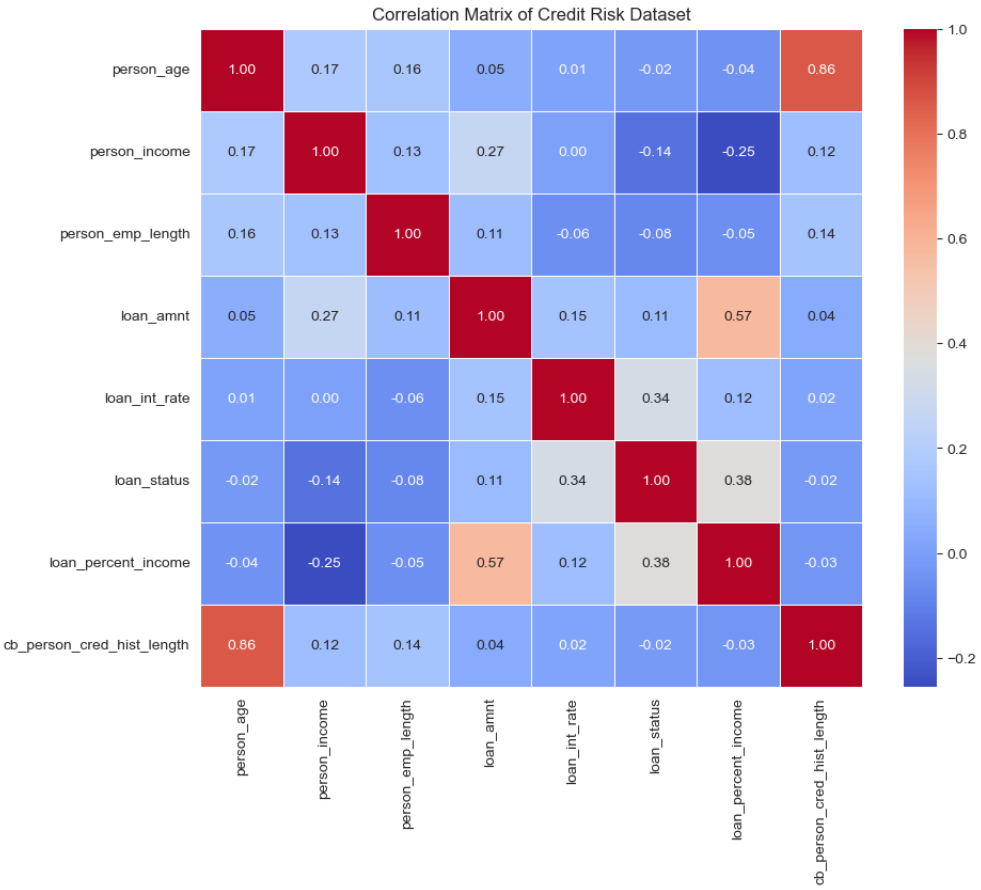
EDA: Exploratory Data Analysis (EDA) involves analyzing and visualizing datasets to understand their underlying structure, identify patterns, detect anomalies, and test hypotheses before applying any modeling techniques.

Some of the analysis done are:

- Identifying independent and Target features from the dataset
- Analyzing Features like, finding the Age frequency in the data
- **Grouping and listing** the history of loan approvals done



- Finding the **correlation between the features** to understand their influence over each other



Handling Outliers to avoid them influencing both positively or negatively and impacting prediction accuracy. Hence the below features outliers were excluded post iterations so that data other key features values are not lost

- Person's age
- Person's income

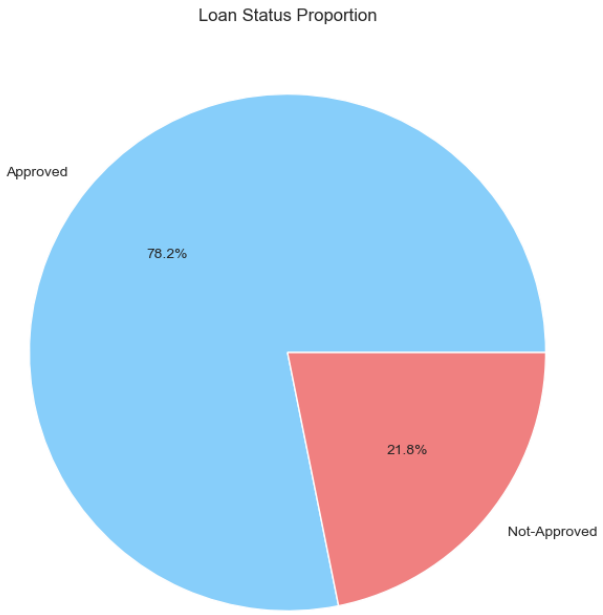
Normalize and transform values of features so that the magnitude does not influence the predictions and bringing al the values to a level playing field. All the features holding numerical values were normalized using min-max scaler that will transform all values within 0 to 1, without losing the variance across.

Derive new features: Created a new feature from interest rate to classify them as 'Low', 'Medium and 'High' based on the range. 0 to 10% is classified as 'Low', 10% to 15% as 'Medium' and above 15% as 'High'.

Perform cleanup: If some of the features has missing values then median values were used to fill the missing values

To aid the model use the non-numerical categorical features, they were converted to numbers using **one hot encoding**.

Prior to Modelling the refined dataset, it was identified that target variable had an unbalanced data and hence might end up with overfitting.



Dataset had ~78% Approved cases records and ~22% non-approved cases records. To balance the same SMOTE was used. SMOTE (Synthetic Minority Over-sampling Technique) is a popular technique used to address the issue of imbalanced datasets in machine learning. Post that the both approved and not-approved had equal proportion.

Train and evaluate the model

Following are the steps to train the model

- **Split the data for training and test**: Divide the dataset into a training set, used to train the model, and a test set, used to evaluate the model's performance on unseen data. This split ensures that the model's accuracy is tested on data it has not encountered during training, providing an unbiased evaluation.

- **Choose an Algorithm to train the model:** Select an appropriate machine learning algorithm based on the characteristics of the data. The choice of algorithm affects how well the model can learn patterns from the data. Following were the algorithms used as the target feature had binary classification
    - Logistic Regression
    - Support Vector Machine
    - Random Forest Classifier
    - XGBClassifier

- **Fit the model**: Train the selected algorithm on the training data by fitting the model to learn the relationship between the input features and the target variable. This process involves adjusting the model's parameters to minimize the error in predictions.

- **Evaluate the model and find the metrics of each model**: Assess the model's performance on the test set using evaluation metrics such as accuracy, precision, recall, and F1-score, these metrics provide insight into how well the model generalizes to new data.

- **Use hyper tuning parameters to fine tune and identify best optimal model**: Adjust the model's hyperparameters (e.g., learning rate, regularization strength) through techniques like cross-validation to improve its performance. Hyperparameter tuning helps in finding the best combination of parameters that result in the most accurate model.

- Use GridSearchCV to find the optimal model using hyper tuning model parameter grid: Implement GridSearchCV to exhaustively search through a specified grid of hyperparameters,

evaluating the model's performance for each combination. This method systematically identifies the best hyperparameter values that yield the highest model performance.

Evaluate the metrics for optimal models

- Find the evaluation metrics from the optimal model: Once the optimal model is identified using GridSearchCV, evaluate its performance on the test set using the same metrics as before. These metrics indicate how well the tuned model performs in comparison to the initial model.

Conclusion

Findings from across the optimal models

Model Name	Accuracy	Precision	Recall	F1 Score	True Pos	True Neg	False Pos	False Neg
Logsitic Regression	0.81	0.56	0.77	0.65	1128	4197	329	858
Support Vector Machine	0.86	0.69	0.74	0.71	1081	4571	376	484
Random Forest Classifier	0.91	0.83	0.75	0.79	1099	4842	358	213
XGBooster Classifier	0.92	0.90	0.73	0.80	1071	4938	386	117

Evaluation Metrics

The following metrics are chosen for evaluating the performance of classification models due to their ability to provide comprehensive insights:

Accuracy: Measures the ratio of correctly predicted instances to the total instances. This metric gives a quick snapshot of the model's overall performance across all classes.

Precision: Represents the ratio of correctly predicted positive observations to the total predicted positives. Precision is crucial when the cost of false positives is high. For example, in spam detection, high precision ensures that fewer legitimate emails are incorrectly marked as spam.

Recall: Indicates the ratio of correctly predicted positive observations to all observations in the actual positive class. Recall is essential when the cost of false negatives is high. For instance, in disease screening, high recall ensures that most actual positive cases are identified, reducing the risk of missing true positive cases.

F1 Score: The harmonic mean of precision and recall. The F1 score balances both precision and recall, making it useful as a single metric for evaluating model performance, particularly in cases of class imbalance.

Confusion Matrix: A table that describes the performance of a classification model by displaying the true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The

confusion matrix provides detailed insights into the model's performance, helping to diagnose specific issues, such as whether the model is more prone to false positives or false negatives.

These metrics collectively offer a well-rounded evaluation of a model's performance, highlighting different aspects and helping to identify areas for improvement.

Based on the evaluation metrics, the XGBoost Classifier excels in several aspects:

Accuracy: The XGBoost Classifier has a high accuracy of 92%, indicating that it correctly predicted 92% of the test values. The Random Forest Classifier is close behind with 91% accuracy.

Precision: With a precision of 90%, the XGBoost Classifier demonstrates that it correctly identified 90% of the positive predictions, effectively reducing the false positive rate.

Recall: The recall score for XGBoost is 73%, which is lower than the optimal standard and slightly less than Logistic Regression's recall of 77%. This indicates that while XGBoost has a higher false negative rate, it is still competitive.

F1 Score: The F1 score, balancing precision and recall, is 80% for XGBoost, higher than the other models, suggesting it better identifies positive cases.

Confusion Matrix: The confusion matrix shows that the XGBoost Classifier has higher true positive and true negative predictions compared to other models, indicating better overall prediction capability.

In conclusion, the XGBoost Classifier outperforms other optimized models, demonstrating superior accuracy, precision, and balanced performance.

The comprehensive evaluation indicates that the XGBoost Classifier is the most optimized model for this classification task, providing the best balance among accuracy, precision, recall, and F1 score. Its ability to effectively handle class imbalances and minimize both false positives and false negatives makes it the preferred choice. However, the Random Forest Classifier is also a strong performer, with metrics only slightly lower than XGBoost. Support Vector Machine and Logistic Regression are suitable for specific scenarios where their particular strengths (higher recall for Logistic Regression and balanced performance for SVM) are advantageous.

Overall, selecting the right model depends on the specific needs of the application, whether prioritizing reducing false positives, minimizing false negatives, or balancing both effectively.

Way Forward

The optimal XGBoost model has an accuracy of 0.92 or 92% and following are ways to improve and also the challenges

Ways to improve: Create or identify new features post discussing with SME and use the same.

Challenge: Feature engineering requires a deep understanding of the data and problem domain,

Ways to improve: Use the new data and re-run the model and evaluate the same

Challenge: Sometimes new data might introduce more noise and degrade the model

Ways to improve: Fine tune Hyperparameters such as learning rate, max depth etc. and find the optimal model and train it and evaluate.

Challenge: Hyperparameter tuning can be computationally expensive and time-consuming,