Beyond the Banks: A Machine Learning Odyssey in Loan Default Prediction

Dhanashri Deshpande
deshpande.dha@northeastern.com
Northeastern University
Seattle, Washington, USA

Abhigna Balusani
balusani.a@northeastern.edu
Northeastern University
Seattle, Washington, USA

Shuyang Ren
ren.shuy@northeastern.edu
Northeastern University
Seattle, Washington, USA

Figure 1: The cost of loan defaults: A visual metaphor.

Abstract

Accurately identifying potential loan defaulters is crucial for reducing financial risk in banking and lending. This project develops a comprehensive machine learning pipeline to predict loan defaults using a variety of models and imbalance-handling techniques. We compare various models including Logistic Regression, SVM, Random Forest, and advanced methods like XGBoost, CatBoost, and a Stacking Classifier. The dataset is highly imbalanced, which we address using SMOTE, SMOTE-ENN, random oversampling, and undersampling. Our work demonstrates the effectiveness of combining resampling strategies with ensemble learning in loan default prediction.

1 Introduction

Since the invention of banking, accurately assessing the risk of loan defaults is critical for sustaining the financial health of lending institutions. Loan default—when borrowers fail to meet their repayment obligations—poses significant risks, including revenue losses, increased credit risk, and diminished investor confidence. Traditionally, banks have relied on conventional credit scoring systems and expert judgment to evaluate default risk, but with the arrival of the information age, computing, and machine learning, more sophisticated methods are available to provide more effective analysis and predictions on the large quantity of data in circulation. We seek to emulate such possibilities by performing a large set of modeling on a data set of bank-loan defaults. In the following sections, we detail the methodology and experiments undertaken, discuss the challenges and trade-offs encountered during model development, and evaluate the results in terms of various performance metrics. Ultimately, we wish to demonstrates how integrating machine learning into financial risk management can lead to smarter, data-informed lending decisions that benefit both financial institutions and their customers.

CS6140, Northeastern University
.

2 Related Work

Ali Albastaki, Ali Abdullatif [1] The paper evaluated several machine learning models for predicting loan application outcomes, including Logistic Regression, Decision Tree, Random Forest, K-Nearest Neighbors (KNN), and Neural Networks. In terms of overall accuracy, the Random Forest and Decision Tree models performed the best, both achieving an accuracy of 91.8%, closely followed by Logistic Regression at 91.8%. However, Logistic Regression demonstrated the strongest performance in identifying loan applications that resulted in defaults or charge-offs, with a sensitivity of 22.08%, followed by the Decision Tree model at 19.08%.

Alomari and Zakaria [2] The paper investigates how peer-to-peer loan applications on Lending Club can be classified as either paid off or defaulted. The methodology is divided into three main stages: data exploration to understand the structure and distribution of the dataset, data preprocessing to prepare the features for modeling, and a classification phase where several machine learning algorithms were tested and fine-tuned. Among the models evaluated, the Random Forest classifier performed best, achieving an accuracy of 71.7%. In addition to predictive modeling, the study also applies the Apriori algorithm for association rule mining, uncovering meaningful patterns related to borrower behavior and loan characteristics.

Turiel, J.D. and others [3] This paper proposes a two-phase machine learning framework to support more effective lending decisions. In the first phase, the model predicts whether a loan application is likely to be rejected. Among the algorithms tested, Logistic Regression performed best, achieving a recall of 77.4% on the test data. The second phase shifts focus to predicting defaults among approved loans. Here, Deep Neural Networks outperformed other models, with a recall of 72% in identifying defaults.The study also examined loans issued for small business purposes, finding that the rejection prediction model performed best when trained on the full dataset, while default prediction improved when trained specifically on the small business subset.

Zhao, Selena & Zou, Jiying [4] The paper presents a logistic regression model designed to predict loan defaults. Several versions of the model were built using different combinations of borrower and loan-related variables, including loan amount, interest rate, and credit balance. To evaluate performance, the models were assessed using AIC, AUC, and overall prediction accuracy. The final model identified revolving balance and credit utilization as significant predictors of default. With an accuracy of 69.5%, the results suggest that logistic regression can be an effective tool for uncovering key risk factors tied to loan repayment behavior.

These are some of the papers that focus on loan default prediction, each exploring the topic in different contexts and using various approaches.

3 Data Overview

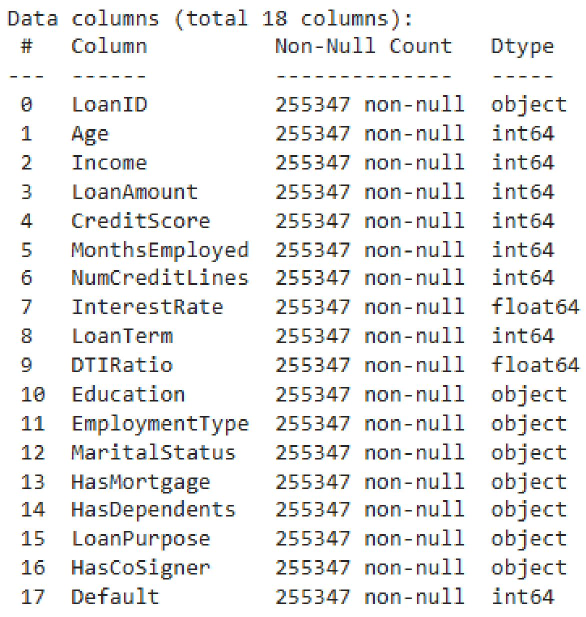
Our data was taken from kaggle which features data from Coursera's Loan Default Prediction Challenge. The dataset contains 255,347 rows and 18 columns in total.

3.1 Source

https://www.kaggle.com/datasets/nikhil1e9/loan-default/data

3.2 Features

Below are the features (columns) of the data set and their data type, where 'object' indicates categorical data while 'float64' or 'int64' indicate numeric data. Note that our target variable, Default, had binary data value (0 or 1) which signifies whether the loan has defaulted (with 1 = yes).



```
Data columns (total 18 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   LoanID          255347 non-null  object 
 1   Age             255347 non-null  int64  
 2   Income          255347 non-null  int64  
 3   LoanAmount      255347 non-null  int64  
 4   CreditScore     255347 non-null  int64  
 5   MonthsEmployed  255347 non-null  int64  
 6   NumCreditLines  255347 non-null  int64  
 7   InterestRate    255347 non-null  float64
 8   LoanTerm        255347 non-null  int64  
 9   DTIRatio        255347 non-null  float64
 10  Education       255347 non-null  object 
 11  EmploymentType  255347 non-null  object 
 12  MaritalStatus   255347 non-null  object 
 13  HasMortgage     255347 non-null  object 
 14  HasDependents   255347 non-null  object 
 15  LoanPurpose     255347 non-null  object 
 16  HasCoSigner     255347 non-null  object 
 17  Default         255347 non-null  int64  
```

Figure 2: Features of the dataset

3.3 Exploratory Analysis

A quick query of the dataset indicates that there were no outliers, no null values, and no duplicated values in the dataset. For that reason no rows were removed and no values were added or modified. We then performed a check for imbalance on the target variable 'Default' and noticed an imbalance for class 1 with only 11.61 percent. This is an issue that we will have to address below. We also generated boxplots and a correlation heatmap, which helped us infer that there are no outliers and no multicollinearity.

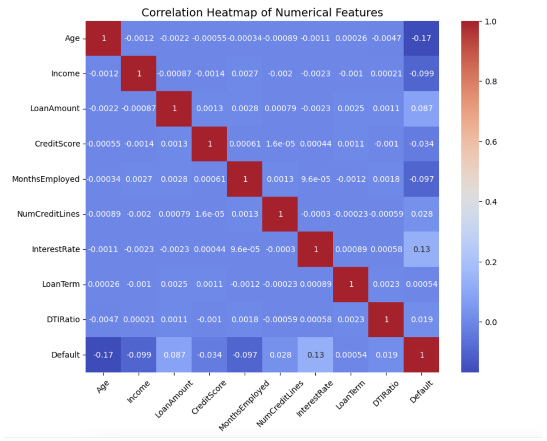


Figure 3: Correlation Heatmap

3.4 Pre-processing

The first step of data pre-processing was to drop the 'LoanID' as it was a unique identifier for each loan and not useful as a predictive feature. We then performed label encoding to transform the categorical data to numerical ones for building our model. To address the issue with imbalance in 'Default', we used SMOTE-ENN (Synthetic Minority Oversampling with Edited nearest neighbors), which first increased the representation of the minority class by generating synthetic samples, and then removed inconsistent or noisy examples from the newly augmented dataset. Finally, we performed feature scaling on the data for sensitive models such as support vector machines (SVM-Linear), multilayer perceptron (MLP), and Naïve Bayes. Originally, we used One-Hot encoding and SMOTE without ENN,explored over sampling and under sampling but later on switched over to using Label encoding and SMOTE+ENN which showed better performance in the models' predictive power. We will discuss those in detail in the sections further below.

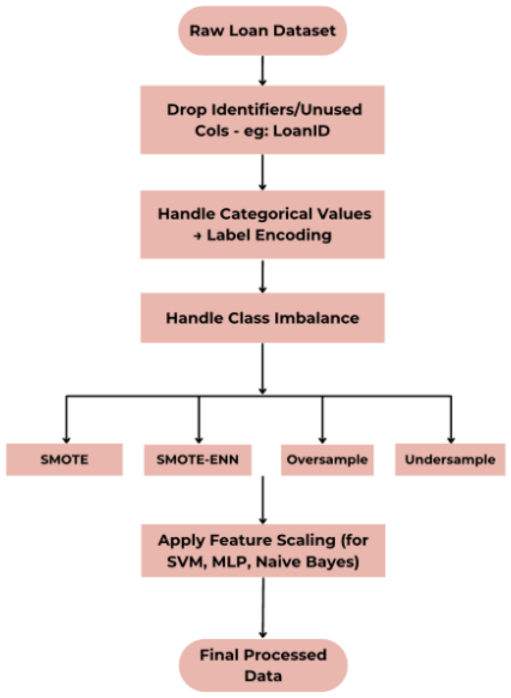

Raw Loan Dataset
Drop Identifiers/Unused Cols - eg: LoanID
Handle Categorical Values → Label Encoding
Handle Class Imbalance
SMOTE
SMOTE-ENN
Oversample
Undersample
Apply Feature Scaling (for SVM, MLP, Naive Bayes)
Final Processed Data

Figure 4: Data Workflow

The Data Workflow represents an overview of how the data was processed from its raw form to the data that was used to train our models, as well as the steps and procedures we took to prepare and refine it.

4 Methods

In this project, we experimented with a range of machine learning models for binary classification of loan defaults. We began by addressing class imbalance, then evaluated a sequence of models — from baseline learners to advanced ensembles. Evaluation metrics included **Accuracy, AUC, Precision, Recall**, and **F1-Score**.

We categorized our approach into:

- **Resampling Strategies**: SMOTE, SMOTE-ENN, Random Oversampling and Random Undersampling to address class imbalance.
- **Baseline Models**: Simple interpretable learners like Logistic Regression and Decision Trees.
- **Advanced Models**: Complex Models with enhanced learning capacity.
- **Stacked Ensemble**: Combined top models for better generalization.

4.1 Resampling Strategies

To mitigate class imbalance, we employed several resampling techniques:

- **SMOTE**: Generated synthetic samples to increase positive class. Improved recall but added noise.
- **SMOTE-ENN**: Combined SMOTE with Edited Nearest Neighbors to remove overlapping or noisy majority samples. Enhanced precision and reduced overfitting, especially for tree-based models.
- **Random Oversampling**: Involves duplicating instances from the minority class to balance the dataset. This helped improve the model's ability to detect minority class instances, leading to better F1-scores by enhancing recall with a slight trade-off in precision.
- **Random Undersampling**: Removes samples from the majority class to achieve class balance. This method significantly boosted recall for minority detection but reduced overall accuracy and precision due to the loss of useful majority class information.
- **Class Priors**: Adjusting class priors made the model more sensitive to the minority class. While this led to improved recall, it came at the cost of lower precision, indicating a tendency to overpredict the minority class.

SMOTE-ENN provided the best trade-off between recall and precision.

4.2 Baseline Models

We started with baseline models to identify the best performing model suited to our data. These models were further trained and evaluated using various sampling techniques (such as SMOTE and SMOTE-ENN) and performance-optimizing strategies like grid search, regularization, and class weighting. The results presented here reflect each model's best performance after applying these enhancements.

4.2.1 Naïve Bayes. Naïve Bayes is a probabilistic classifier based on Bayes' Theorem, assuming independence among features. It is fast, easy to implement, and performs well with high-dimensional data. However, in this case, it produced weaker results, with an accuracy of (0.67), AUC of (0.75), and particularly low precision (0.21), leading to a poor F1-score of (0.33). While it managed a moderate recall of (0.70), the overall low performance—especially in precision—makes it ill-suited for this data.

4.2.2 K-Nearest Neighbors. KNN is a non-parametric, instance-based learning algorithm that classifies samples based on the majority label of their nearest neighbors in feature space. Although intuitive and simple, its performance depends heavily on the choice of distance metric and the number of neighbors. In this evaluation, KNN achieved an accuracy of (0.65), AUC of (0.75), and similar to Naïve Bayes, showed poor precision (0.21) and F1-score (0.32). Despite a reasonable recall (0.73), the overall weak performance makes it unfit for the dataset.

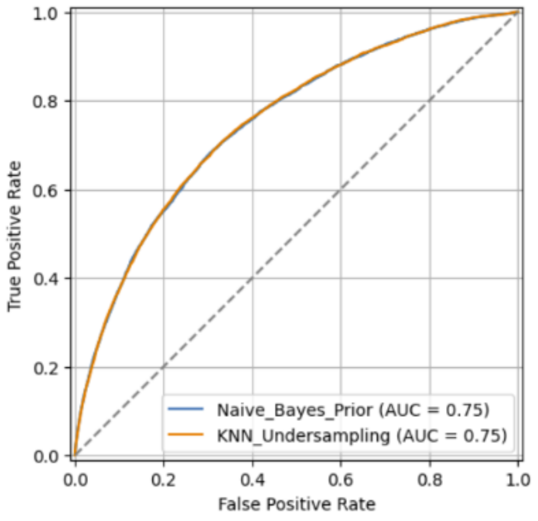


Figure 5: ROC Curve - Naïve Bayes,K-Nearest Neighbors

4.2.3 Logistic Regression. Logistic Regression is a linear model that estimates the probability of a binary outcome using a logistic function. It is known for its simplicity, efficiency, and interpretability, especially in linearly separable problems. In this case, the model achieved an accuracy of (0.75), AUC of (0.81), precision of (0.77), recall of (0.80), and an F1-score of (0.79). These results indicate that Logistic Regression performs reliably, particularly in this datasets . Its strong precision and balanced recall makes it a good choice.

4.2.4 Support Vector Machine. SVM is a margin-based classifier that seeks to find the optimal hyperplane separating classes by maximizing the margin between support vectors. It is particularly effective in high-dimensional spaces. In this dataset, SVM achieved an accuracy of (0.71) and an AUC of (0.78). While its precision was relatively low (0.69), it showed excellent recall (0.89), which is crucial in scenarios like this where minimizing false negatives is a priority. This tradeoff makes SVM suitable for our data, where catching all default cases is critical.

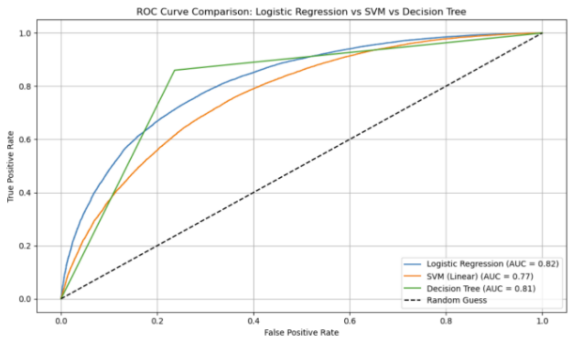


Figure 6: ROC Curve - Logistic Regression,Support Vector Machine,Decision Tree

4.2.5 Decision Tree. This is a tree-based model that recursively splits the data into subsets based on feature values, aiming to maximize information gain at each node. It is highly interpretable and easy to visualize. Decision Tree achieved the highest overall performance across all metrics, including accuracy (0.81), precision (0.83), recall (0.86), and F1-score (0.84). Its balanced results across these metrics suggest that it is effectively handling both false positives and false negatives, making this one of the best performers in baseline models.

The Decision Tree performed well compared to the other baseline models, but we have to acknowledge that even its best results might not be strong enough for real-world use. Since we are dealing with complex, non-linear relationships between features, there is a clear need to explore more advanced models that can better capture those patterns and deliver stronger predictive performance, especially in terms of precision and recall.

4.3 Advanced Models

After evaluating baseline models, we transitioned to more sophisticated algorithms capable of capturing complex non-linear patterns and offering stronger generalization under class imbalance. This section describes the advanced models used in this study, along with their performance characteristics and insights.

4.3.1 Random Forest. This is an ensemble method that constructs multiple decision trees using bootstrapped samples and aggregates their predictions. It balances performance and interpretability well. When trained with class weights and SMOTE-ENN resampling, it achieved strong recall and F1-score across imbalanced data scenarios. In our experiments, Random Forest delivered reliable results across all metrics: accuracy (0.83), precision (0.83), recall (0.83), and F1-score (0.83). Its symmetry in performance indicates balanced handling of both false positives and false negatives.

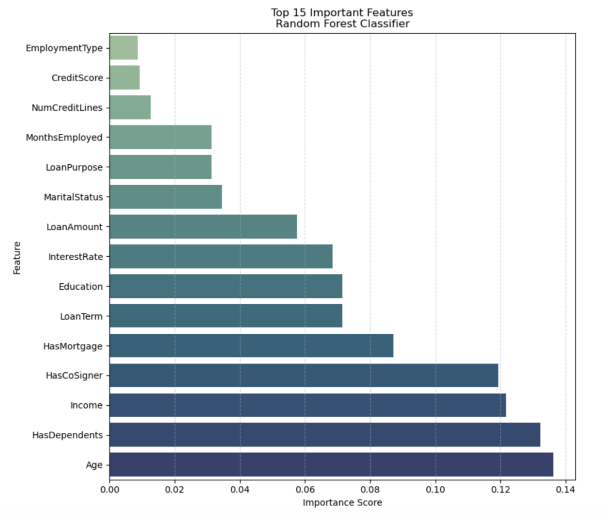


Figure 7: Top 15 Important Features Identified by the Random Forest Classifier

Feature Importance Analysis. Figure 2 shows the top 15 features ranked by their importance as determined by the Random Forest classifier. The most influential features in predicting the target variable include Age, HasDependents, Income, and HasCoSigner. These features received the highest importance scores, indicating their significant role in model decisions. On the other hand, variables like EmploymentType and NumCreditLines had relatively low importance in the model. Feature importance in Random Forest is derived from how much each feature decreases impurity across all trees in the forest.

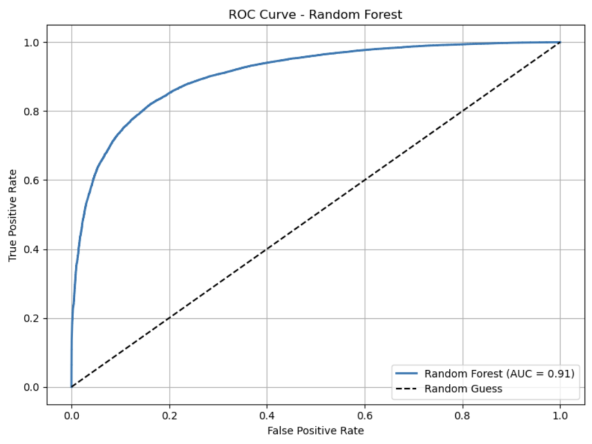


Figure 8: ROC Curve — Random Forest

4.3.2 XGBoost. XGBoost is a gradient boosting framework that combines decision trees in a sequential manner, with each new tree correcting the errors of its predecessors. It can handle missing values internally and performed well on high-dimensional data. It demonstrated excellent classification ability with an AUC of 0.96 and F1-score of 0.91. The model maintained strong precision (0.93) and recall (0.88), making it well-suited for applications requiring reliable separation between defaulters and non-defaulters.

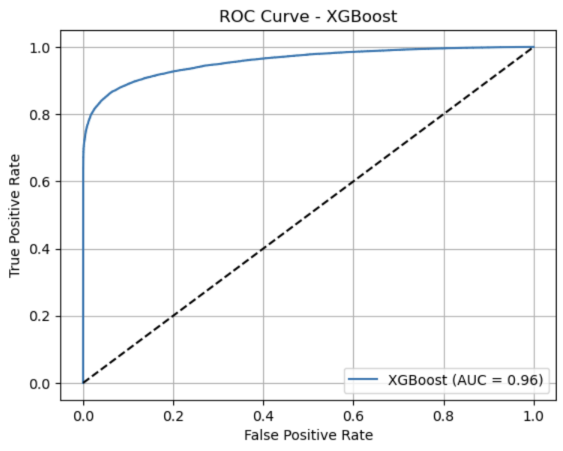


Figure 9: ROC Curve — XGBoost

4.3.3 CatBoost. This is a gradient boosting algorithm specifically optimized for categorical data. It introduces innovations like ordered boosting and minimal prediction shift, reducing overfitting while preserving generalization. CatBoost consistently achieved the highest performance among individual models — AUC (0.97), precision (0.97), recall (0.89), and F1-score (0.93). This robustness across metrics made it the top-performing single model.

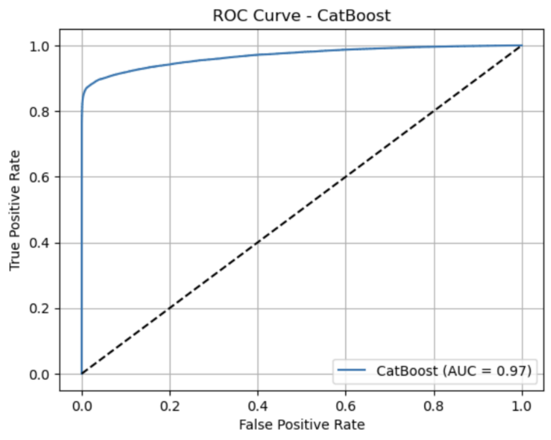


Figure 10: ROC Curve — CatBoost

4.3.4 Multilayer Perceptron (MLP). This is a fully connected feed-forward neural network. It required feature scaling and longer training times but was able to model complex feature interactions. MLP achieved competitive performance with an accuracy of 0.84, precision of 0.88, recall of 0.84, and F1-score of 0.86. Though it slightly lagged behind tree ensembles, it proved valuable as a diverse learner for ensembling.

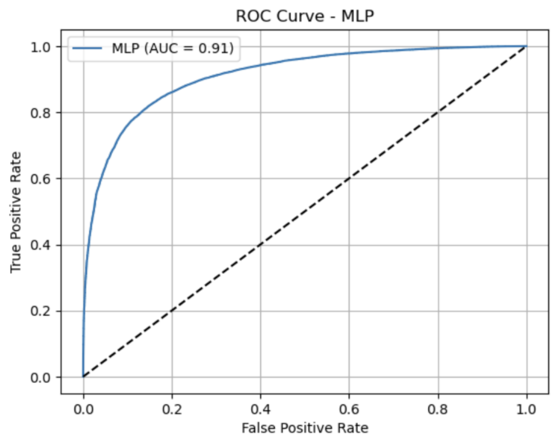


Figure 11: ROC Curve — MLP

These advanced models provided enhanced generalization, particularly when paired with SMOTE-ENN. Among them, **CatBoost emerged as the top-performing single model**, achieving the highest accuracy (0.92), AUC (0.97), precision (0.97), and F1-score (0.93). Its ability to handle categorical features natively and mitigate overfitting through ordered boosting contributed significantly to its superior performance. XGBoost also delivered strong results, particularly in AUC and precision, owing to its effective regularization and gradient boosting efficiency. MLP, while slightly trailing in F1-score, offered valuable learning of non-linear relationships that traditional tree models might overlook.

Given the strengths of individual models, especially the diversity in learning capabilities between tree-based methods and neural networks, we next explored **stacking ensembles** to leverage complementary strengths. By combining models like Random Forest, XGBoost, and CatBoost into a meta-learning architecture, stacking aims to further improve generalization and mitigate the weaknesses of any single model.

4.4 Stacking and Ensemble

Stacking is an ensemble learning technique that combines the strengths of multiple base models by training a meta-learner on their outputs. This approach aims to reduce generalization error by capturing complementary patterns learned by different algorithms.

In our pipeline, we used the following base learners:

- **Random Forest** – for stable, interpretable decision boundaries.
- **XGBoost** – for high-performance gradient boosting with regularization.
- **CatBoost** – for handling categorical variables and reducing overfitting.

The predictions (probabilities) from these models were passed to a **Logistic Regression** model acting as the meta-learner. This architecture allows the stacking model to learn how to optimally combine base-level outputs.

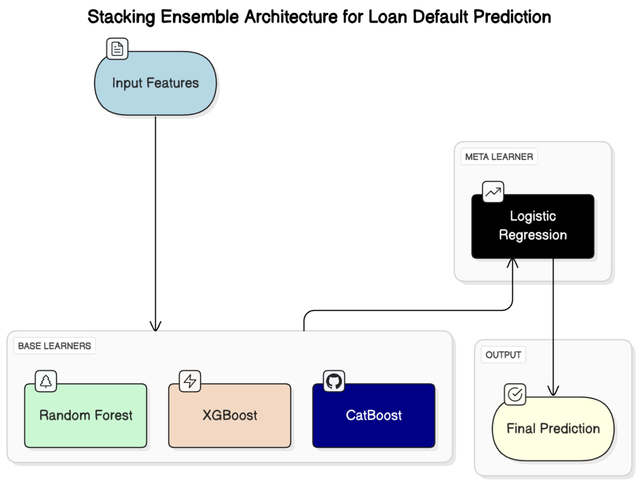


Figure 12: Stacking Ensemble Architecture for Loan Default Prediction

The stacking ensemble outperformed all individual models in overall metrics. It maintained high recall and precision in both customer-first and risk-first strategies, making it a well-rounded solution for real-world deployment.

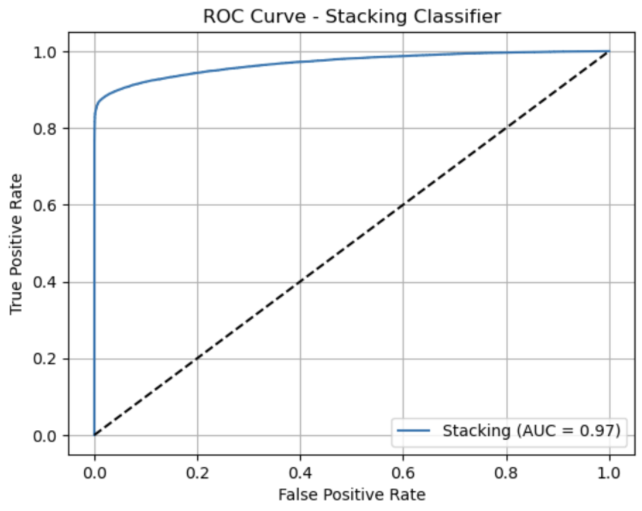


Figure 13: ROC Curve — Stacking Ensemble

5 Results

We evaluated a variety of models using standard classification metrics: Accuracy, AUC-ROC, Precision, Recall, and F1-Score. Results are summarized in Tables 1 and 2 for baseline and advanced models, respectively.

5.1 Key Highlights

- **CatBoost** achieved the best performance among individual models (AUC = 0.9681, F1 = 0.93).

- **Stacking Classifier** slightly outperformed CatBoost overall (AUC = 0.9689, Recall = 0.90).
- **Logistic Regression** served as a strong baseline with balanced metrics (AUC = 0.8165, F1 = 0.79).
- **KNN and Naïve Bayes** performed poorly in isolation but improved slightly with resampling strategies.

Table 1: Baseline Model Performance

Model	Accuracy	AUC	Precision	Recall	F1
Decision Tree	0.8199	0.8120	0.8354	0.8601	0.8476
Logistic Regression	0.7500	0.8165	0.7700	0.8000	0.7900
SVM (Linear)	0.7100	0.7806	0.6900	0.8900	0.7800
Naïve Bayes	0.6719	0.7499	0.2183	0.7070	0.3335
KNN	0.6507	0.7504	0.2108	0.7316	0.3272

Table 2: Advanced Model Performance

Model	Accuracy	AUC	Precision	Recall	F1
Random Forest (Tuned)	0.8300	0.9076	0.8300	0.8300	0.8300
XGBoost	0.8900	0.9576	0.9300	0.8800	0.9100
CatBoost	0.9200	0.9681	0.9700	0.8900	0.9300
MLP (Neural Net)	0.8400	0.9153	0.8800	0.8400	0.8600
Stacking Classifier	0.9200	0.9689	0.9600	0.9000	0.9300

5.2 Model Performance Comparison

To compare the performance across baseline and advanced models, we evaluated Accuracy, AUC-ROC, Precision, Recall, and F1-Score. The results are summarized in Table 1 for baseline models and Table 2 for advanced models. And Figure 14 gives the ROC comparison curve across various models we used.

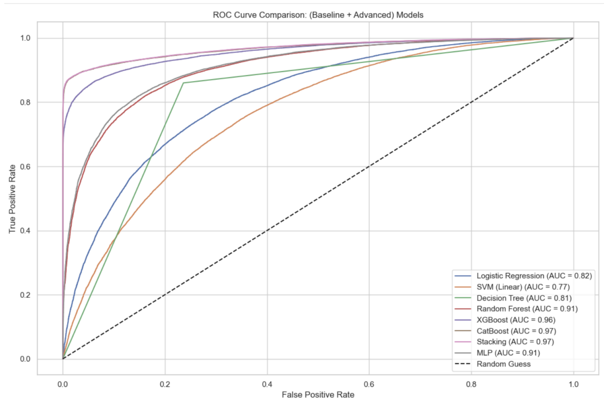


Figure 14: ROC Curve Comparison of Basic + Advanced Models

Figure 15 offers a holistic view of normalized performance. It shows that CatBoost and the Stacking Classifier dominate across all metrics, especially in F1-score and AUC. Baseline models like KNN and Naïve Bayes underperform significantly, especially on precision and AUC.

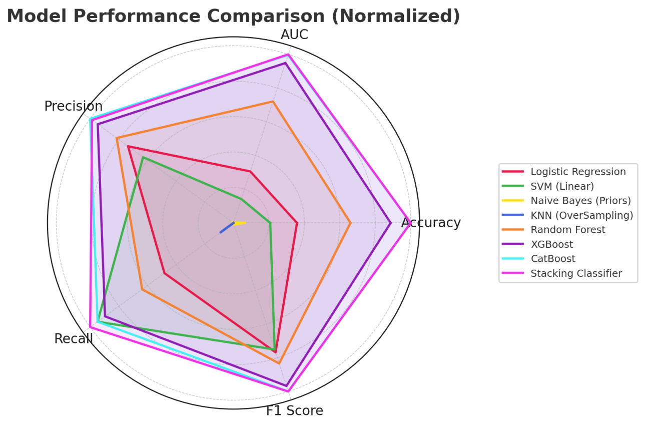


Figure 15: Radar Chart Comparing Model Performance (Normalized)

6 Discussion

6.1 Model Insights

The performance gap between models highlights key architectural differences:

- **CatBoost and XGBoost** performed strongly due to built-in regularization and effective handling of categorical features.
- **Stacking** offered superior generalization by leveraging model diversity from Random Forest, CatBoost, and XGBoost.
- **MLP** performed reasonably well but required more preprocessing and training time.

6.2 Precision–Recall Trade-offs

To better reflect real-world decision making in financial services, we evaluated models under **fixed threshold constraints**, one emphasizing high precision (Customer First) and another high recall (Risk First). Specifically, we fixed:

- **Precision at 0.99** for Customer-First evaluations
- **Recall at 0.90** for Risk-First evaluations

This allowed us to assess which models maintain strong complementary metrics under strict business requirements.

6.2.1 Customer-First (High Precision). This strategy prioritizes reducing false positives to avoid rejecting trustworthy applicants. The central question becomes:

> *"Given that we only flag people when we're 99% sure (high precision), how many actual defaulters do we catch?"*

As seen in Figure 16, CatBoost and Stacking classifiers maintained high recall (around 0.87) even under this strict precision threshold (0.99), making them ideal for customer-first banking products.

6.2.2 Risk-First (High Recall). This strategy aims to identify every potential defaulter, even at the cost of some false positives. The relevant question here is:

> *"If we catch 90% of defaulters (high recall), how precise are those flags?"*

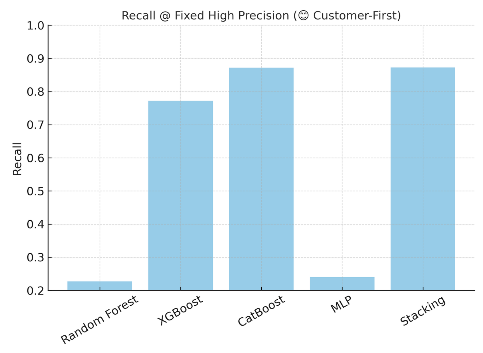


Figure 16: Recall at fixed high precision (Customer-First strategy). CatBoost and Stacking retain the highest recall when required to maintain 99% precision.

Figure 17 shows that both CatBoost and Stacking again lead, delivering precision above 95% even when aggressively maximizing recall (0.90). This makes them excellent choices for risk-first policies in high-stakes loan scenarios.

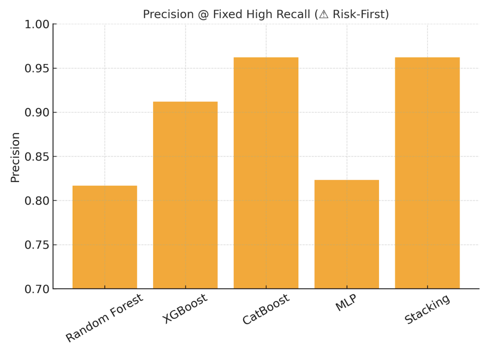


Figure 17: Precision at fixed high recall (Risk-First strategy). CatBoost and Stacking classifiers maintain high precision even when forced to catch 90% of defaulters.

7 Conclusion

We developed a robust ML pipeline for predicting loan defaults using advanced ensemble techniques and effective resampling strategies. CatBoost's strong performance stems from its native handling of categorical variables and built-in regularization, making it ideal for imbalanced, feature-rich credit datasets. Meanwhile, the stacking ensemble delivered the best overall results by combining diverse model strengths.

7.1 Key Findings

- **CatBoost** was the top-performing single model (AUC = 0.97, F1 = 0.93).
- **Stacking Classifier** surpassed individual models, offering flexibility and strong performance across business strategies.
- **SMOTE-ENN** significantly enhanced minority class recall.

7.2 Practical Implications

This work provides actionable insights for deploying machine learning in real-world loan risk assessment. Our pipeline supports data-driven decisions in:

- **Creditworthiness Prediction**: More precise approval screening
- **Fraud and Risk Detection**: Shared backbone for high-risk filtering
- **Probability of Default (PD)** estimation: Supporting credit rating systems

8 Future Work

For future work related to the findings of our project, we considered expanding our models to perform time-based validation for real-world simulation; incorporate cost-sensitive learning such as business impact and cost of fraud; and tuning hyperparameters using Optuna or GridSearchCV. In addition, by applying business intuitions, we can also further research ways to improve the model from the interest points of banks and find optimization among our models by giving more sophisticated risk-management paradigms.

Acknowledgments

The authors would like to thank Prof. Yifan Hu for the guidance and feedback throughout the project. No external funding was received for this work.

References

[1] Ali Albastaki and Ali Abdullatif. 2022. Loan Default Prediction System.
[2] Khalid Alomari and Nasir Zakaria. 2017. Classification of Peer-to-Peer Lending: A Machine Learning Approach. *International Journal of Advanced Computer Science and Applications* (2017).
[3] J.D. Turiel et al. 2020. A Two-Phase Machine Learning Framework for Lending Decisions. *Journal of Financial Data Science* (2020).
[4] Selena Zhao and Jiying Zou. 2021. Loan Default Prediction Using Logistic Regression. *Journal of Risk and Financial Management* (2021).

A GitHub

https://github.com/dhanashri-dotcom/bankloan-default-prediction/

B Dependencies

Run the following command to install the required dependencies for this project:

```
pip install pandas numpy matplotlib seaborn scikit-learn imbalanced-learn xgboost
catboost kagglehub
```

C Execution Method

We strongly recommend you to refer to the above listed GitHub's README for more detailed instructions and to run the main models for training and evaluation.