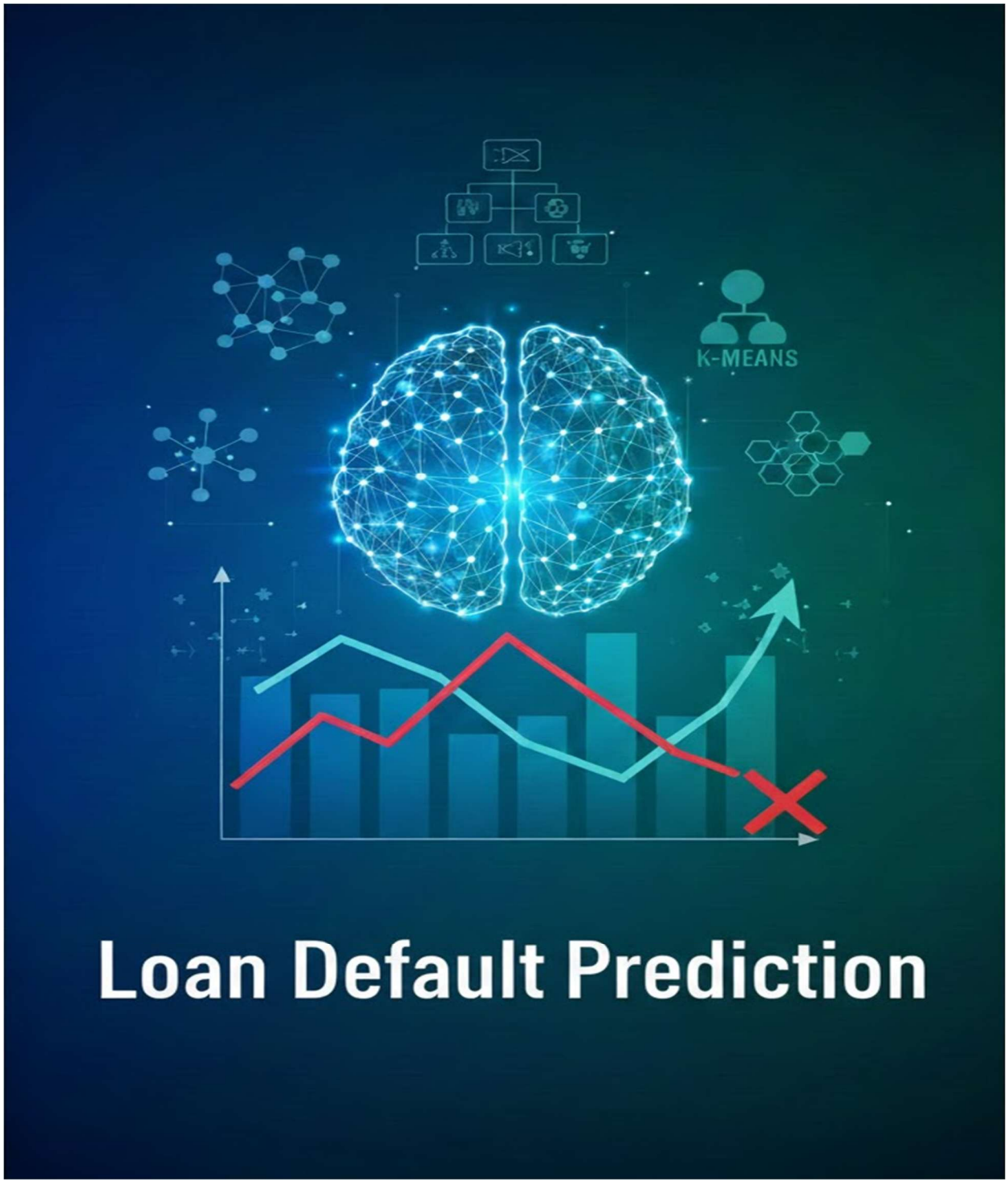

Business Report – Milestone 2

Submitted by James Kagwe

January 09, 2026

Table of Contents

List of Figures

1.0 Executive Summary

The objective of this project was to automate the home loan approval process for retail banking using machine learning. By transitioning from manual, effort-intensive underwriting to a data-driven approach, the bank aims to eliminate human bias, improve operational efficiency, and most importantly minimize the financial impact of Non-Performing Assets (NPAs) while maximizing interest-based revenue.

The key findings were:

Best Performing Model:- The Random Forest algorithm combined with an Undersampling strategy emerged as the superior solution for the bank's lending needs.

Predictive Drivers:- The primary indicators of loan default were identified as Debt-to-Income Ratio (DEBTINC), Number of Delinquencies (DELINQ), and the Age of Credit Lines (CLAGE).

Technical Success: -The recommended model achieved a Recall of 87%, ensuring that most high-risk applicants are flagged before loan disbursement. To ensure business alignment, models were evaluated using a custom cost-benefit framework rather than just standard accuracy.

Efficiency Winner:- The Tuned_RF_Undersampled model produced the lowest Total Estimated Cost ($878,000). By adopting the recommended machine learning framework, the bank shifts from reactive risk management to proactive financial optimization. This approach provides a mathematically robust defense against defaults while ensuring the loan portfolio continues to grow through the confident approval of low-risk customers.

1.2.0 Introduction

1.2.1 Context

A major proportion of retail bank profit comes from interests in the form of home loans. These loans are borrowed by regular income/high-earning customers. Banks are most fearful of defaulters, as bad loans (NPA) usually eat up a major chunk of their profits. Therefore, it is important for banks to be judicious while approving loans for their customer base. The approval process for the loans is multifaceted. Through this process, the bank tries to check the creditworthiness of the applicant based on a manual study of various aspects of the application. The entire process is not only effort-intensive but also prone to wrong judgment/approval owing to human error and biases. There have been attempts by many banks to automate this process by using heuristics. But with the advent of data science and machine

learning, the focus has shifted to building machines learning models that can learn this approval process and make it free of biases and more efficient. At the same time, one important thing to keep in mind is to make sure that the machine does not learn the biases that previously crept in because of the human approval process.

1.2.2 Problem Statement

A bank's consumer credit department aims to simplify the decision-making process for home equity lines of credit to be accepted. To do this, they will adopt the Equal Credit Opportunity Act's guidelines to establish an empirically derived and statistically sound model for credit scoring. The model will be based on the data obtained via the existing loan underwriting process from recent applicants who have been given credit. The model will be built from predictive modeling techniques, but the model created must be interpretable enough to provide a justification for any adverse behavior (rejections).

2.0 Modelling Process (validation & interpretation)

The model building process for **Loan Default Prediction** is focused on creating a classification model to identify likely defaulters using the Loan Default dataset. The approach to building the predictive model will be as follows:

2.1 Introduction

1. **Data Preparation & Cleaning**: This is a critical first step as the dataset has significant missing values. This was done during the analysis in milestone1.
2. **Feature Engineering**: Categorical variables like REASON and JOB were transformed using techniques such as **One-Hot Encoding** before they can be used in machine learning algorithms.
3. **Handling Class Imbalance**: As witnessed during the EDA, the dataset is imbalanced with defaulters (Target=1) constituting of 19.95%. This may introduce bias in the modeling phase. Data Oversampling and Undersampling strategies were used to overcome this challenge.
4. **Model Selection**: The process will involves comparing several classification algorithms:
    a) **Baseline**: Logistic Regression.
    b) **Ensemble Methods**: Random Forest, Bagging, AdaBoost, and Gradient Boosting.
    c) **Advanced Boosting**: XGBoost (imported as XGBClassifier).

2.2 Splitting Data for Modelling

In this modeling exercise data was split into training, validation, and testing sets. This is a fundamental practice in machine learning to ensure that a model can generalize well to new, unseen data. In the context of the **Loan Default Prediction** project, this process is essential for building a reliable and unbiased credit scoring model.

Purpose of Each Split

- **Training Set:** This was the largest portion of the data, used to "teach" the model. The algorithm looks for patterns between applicant features (like DEBTINC or DELINQ) and the target variable (BAD) to learn how to predict a default.

- **Validation Set:** This set was used during the model building phase to fine-tune "hyperparameters"—the settings that aren't learned directly from the data (e.g., the depth of a Decision Tree). It helps in comparing different models and select the best one without "peeking" at the final test results.
- **Testing Set:** This was a separate "hold-out" set that the model never saw during training or tuning. It acts as a final exam to provide an unbiased evaluation of how the model will perform in the real world on completely new loan applications.

Why This is Critical for Loan Prediction

a) **Avoiding Overfitting:** A model might perform perfectly on the training data by simply "memorizing" specific applicants. Splitting ensures the model learns general rules (e.g., "high debt-to-income ratios increase risk") rather than specific instances, which is vital for the bank's long-term profitability.

b) **Measuring Generalization:** The bank needs to know how the model will handle future applicants. The test set provides a statistically sound estimate of future performance metrics like **Recall** (capturing as many defaulters as possible) and **Precision** (avoiding rejecting good customers).

c) **Selecting the Best Strategy:** The validation split allows you to objectively decide which combination of these techniques creates the most accurate and fair model before testing and deployment.

d) **Unbiased Assessment:** By keeping the test set separate, we ensure that the evaluation isn't influenced by the choices made during training, providing the "empirical evidence and statistical soundness".

2.3 Evaluation Strategy

Models will be evaluated using a variety of metrics to ensure they minimize non-performing loans while remaining interpretable. The evaluation of a credit risk model goes beyond simple accuracy. Because the dataset is imbalanced (fewer defaults than non-defaults), accuracy can be misleading. Several key metrics will be employed.

a) **Recall (Sensitivity):** This is the most critical metric. It measures the model's ability to identify all potential defaulters. A high recall ensures the bank captures as many "bad" loans as possible, even if it occasionally flags a "good" loan.

b) **Precision:** This will also be considered. It measures how many of the predicted defaults were actual defaults. While recall is prioritized, precision is important to ensure the bank isn't rejecting too many creditworthy applicants.

c) **F1-Score:** This is the harmonic mean of precision and recall, providing a balanced measure of the model's performance on the "Bad" class.

d) **ROC-AUC (Area Under the Receiver Operating Characteristic Curve):** This measures the model's ability to distinguish between the two classes (0 and 1) across all possible thresholds.

e) **Confusion Matrix:** The visual tool used to see the distribution of True Positives, True Negatives, False Positives (Type I error), and False Negatives (Type II error). In this context, a **False Negative** (approving a loan for someone who defaults) is typically much costlier for the bank than a False Positive.

2.4 Model Selection

Selecting the "best" model involved a multi-step comparison:

1. **Establish a Baseline:** We shall start with a simple model **Logistic Regression** to understand the basic linear relationships in the data.
2. **Test Ensemble Methods:** We proceeded and compare Bagging (e.g., Random Forest) and Boosting (e.g., AdaBoost, Gradient Boosting, XGBoost). These models generally perform better on complex, non-linear tabular data.
3. **Address Imbalance:** Lastly, we evaluated models after applying techniques like **SMOTE** (Synthetic Minority Over-sampling Technique) to see if performance on the minority "Default" class improves.
4. **Selection for Tuning**
    - The models with the highest **Recall** on the validation were selected/ preferred, provided its Precision and F1-score remain acceptable. The 5 best evaluated models were selected for further tuning.

2.5 Hyperparameter Tuning

Hyperparameter tuning is the process of optimizing the "settings" of a model that are not learned from the data. **RandomizedSearchCV** was used for this purpose. Unlike Grid Search, which tries every possible combination, Randomized Search samples a fixed number of parameter

combinations from specified distributions. This is more computationally efficient and often finds a near-optimal solution in significantly less time. **Cross-Validation:** The tuning process used **Stratified K-Fold cross-validation**. This ensures that each "fold" of the data has the same proportion of defaults as the whole dataset, leading to more reliable performance estimates.

2.6 Final Selection Criteria

- **Performance:** The model with the highest **Recall** on the test set wasselected, provided its Precision and F1-score remain acceptable.
- **Stability:** The model that shows the least variance in performance across different cross-validation folds.
- **Interpretability:** Per the problem statement, the model must be interpretable enough to justify rejections to applicants (e.g., using Feature Importance plots).

3.0 Model comparison based on the metric of choice (accuracy, recall, RMSE, R-squared, etc. whichever is applicable)

3.1 Modeling with the Original Data

Modeling with the **original data** involved building and evaluating the machine learning models on the dataset after standard preprocessing (cleaning and encoding) but *before* applying any synthetic balancing techniques like SMOTE (Oversampling) or Random Undersampling.

This stage is crucial because it establishes a realistic performance baseline, reflecting how the model would behave on the naturally imbalanced data typical of loan portfolios. Below are the outputs from the modeling process.

Detailed Model Performance on Original Data

Model	Data Type	Set	Accuracy	Recall	Precision	F1
Logistic regression	Original Data	Training	0.8104	0.0842	0.7059	0.1504
Logistic regression	Original Data	Validation	0.807	0.0714	0.6538	0.1288
Bagging	Original Data	Training	0.9899	0.9523	0.9971	0.9742
Bagging	Original Data	Validation	0.8901	0.6597	0.7585	0.7056
Random forest	Original Data	Training	1	1	1	1
Random forest	Original Data	Validation	0.9144	0.6975	0.8469	0.765
GBM	Original Data	Training	0.9245	0.6985	0.9005	0.7867
GBM	Original Data	Validation	0.8985	0.6429	0.8095	0.7166
Adaboost	Original Data	Training	0.8851	0.5568	0.8069	0.6589
Adaboost	Original Data	Validation	0.8867	0.5966	0.7845	0.6778
Xgboost	Original Data	Training	0.9997	1	0.9986	0.9993
Xgboost	Original Data	Validation	0.9144	0.7101	0.8366	0.7682
dtree	Original Data	Training	1	1	1	1
dtree	Original Data	Validation	0.8733	0.6933	0.679	0.6861

Figure 1:A Table Showing Detailed Model Performance on Original Data (Grouped by Model)

The results reveal a significant gap between training and validation performance for several models:

Overfitting Models: Random Forest, XGBoost, and Decision Tree (dtree) show perfect or near-perfect scores (Recall = 1.0, Accuracy = 1.0) on the Training set but drop significantly on the Validation set (Recall ~0.69–0.71). This indicates the models have "memorized" the training data and may struggle with new, unseen loan applications data.

The Baseline (Logistic Regression): Logistic Regression performed poorly on the original data, with a validation Recall of only 0.0714. This suggests that the relationship between the features and loan default is non-linear and complex, making it difficult for a simple linear model to identify defaulters. This usually happens when the data is imbalanced and the model defaults to predicting the majority class ("No Default") to keep accuracy high. As we are already aware, the data we are using is imbalanced.

Selecting the Best Model: Recall vs. Precision in Loan Default Prediction, the cost of a False Negative (approving a loan for someone who defaults) is much higher than a False Positive (rejecting a good customer). Therefore, Recall is the primary KPI.

Current Leaders: Best Overall Performers: XGBoost and Random Forest tied for the highest Accuracy (0.9144) on the validation set. However, XGBoost had a slightly better Recall (0.7101) compared to Random Forest (0.6975).

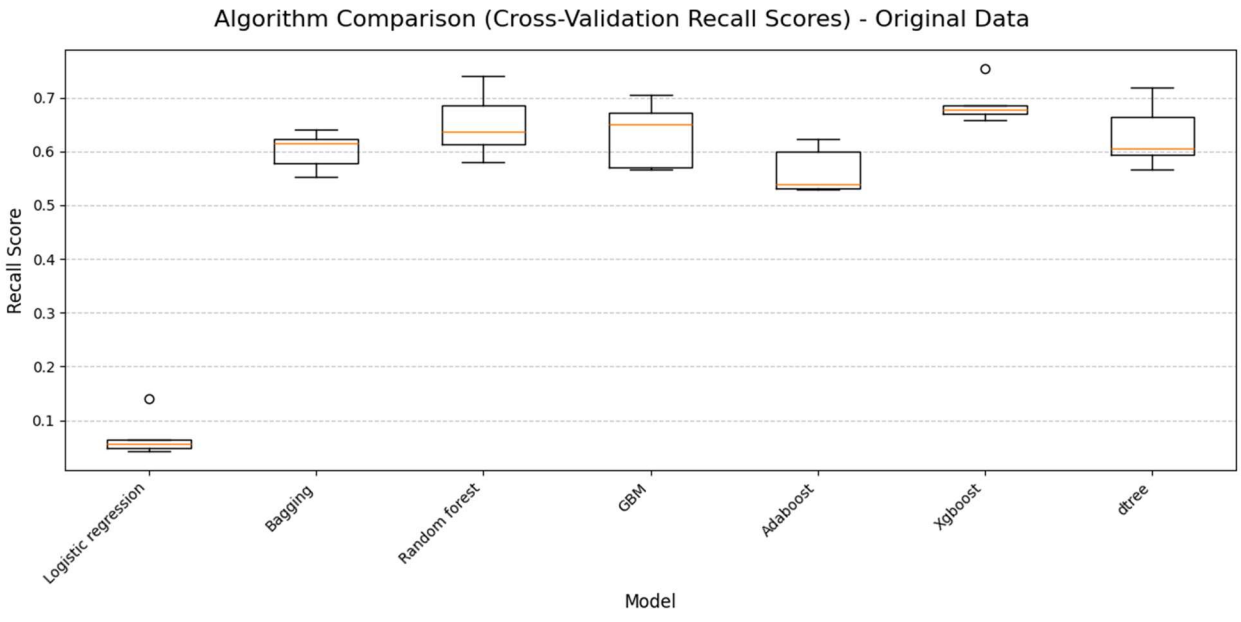

Figure 2: A chart Showing Comparison for performance of models using Recall Scores

3.2 Modeling with the Oversampled Data

Oversampling is a technique used to "balance" the dataset by creating synthetic examples of the minority class (defaulters), which forces the model to pay more attention to them.

Detailed Model Performance on Oversampled Data						
Model	**Data Type**	**Set**	**Accuracy**	**Recall**	**Precision**	**F1**
Logistic regression_O	Oversampled Data	Training	0.6748	0.6626	0.6792	0.6708
Logistic regression_O	Oversampled Data	Validation	0.6812	0.6429	0.3415	0.4461
Bagging_O	Oversampled Data	Training	0.9942	0.9902	0.9982	0.9942
Bagging_O	Oversampled Data	Validation	0.8951	0.7059	0.7534	0.7289
Random forest_O	Oversampled Data	Training	1	1	1	1
Random forest_O	Oversampled Data	Validation	0.9169	0.7521	0.8174	0.7834
GBM_O	Oversampled Data	Training	0.9272	0.9123	0.9402	0.9261
GBM_O	Oversampled Data	Validation	0.8851	0.7395	0.7012	0.7198
Adaboost_O	Oversampled Data	Training	0.8533	0.8282	0.872	0.8495
Adaboost_O	Oversampled Data	Validation	0.8398	0.7227	0.5791	0.643
Xgboost_O	Oversampled Data	Training	0.9997	0.9997	0.9997	0.9997
Xgboost_O	Oversampled Data	Validation	0.9245	0.7605	0.8458	0.8009
dtree_O	Oversampled Data	Training	1	1	1	1
dtree_O	Oversampled Data	Validation	0.8389	0.7269	0.5767	0.6431

Figure 3: A table showing Detailed Model Performance on Oversampled Data

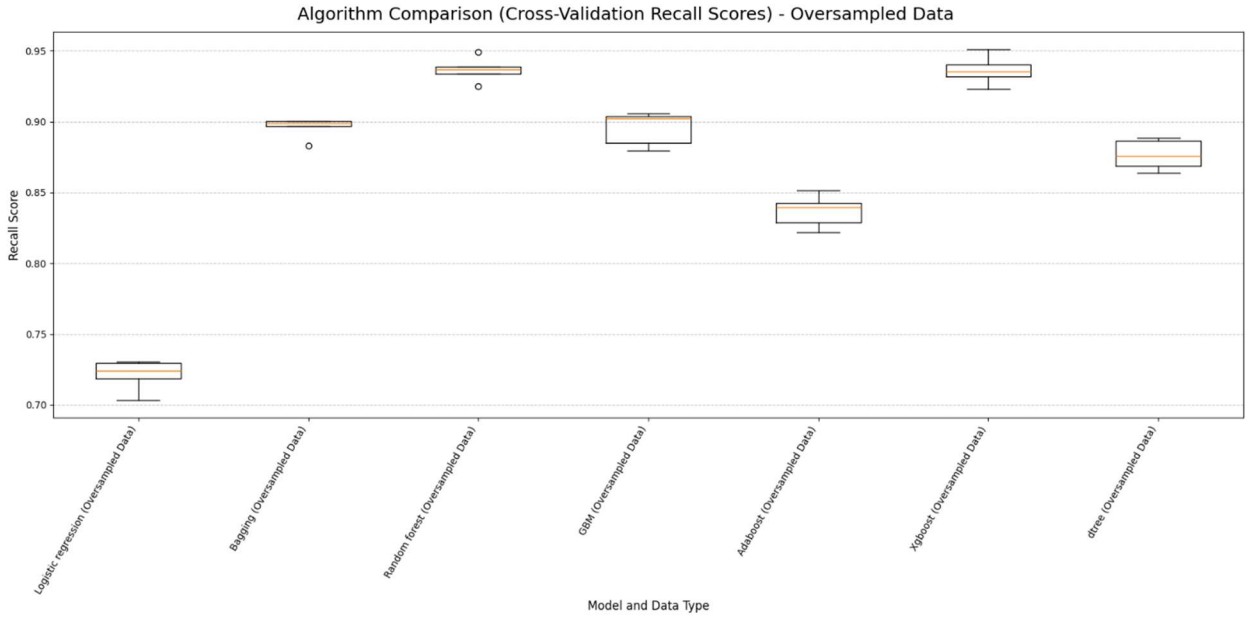

Figure 4: A Chart showing Models comparison using the Cross-Validation Recall Scores

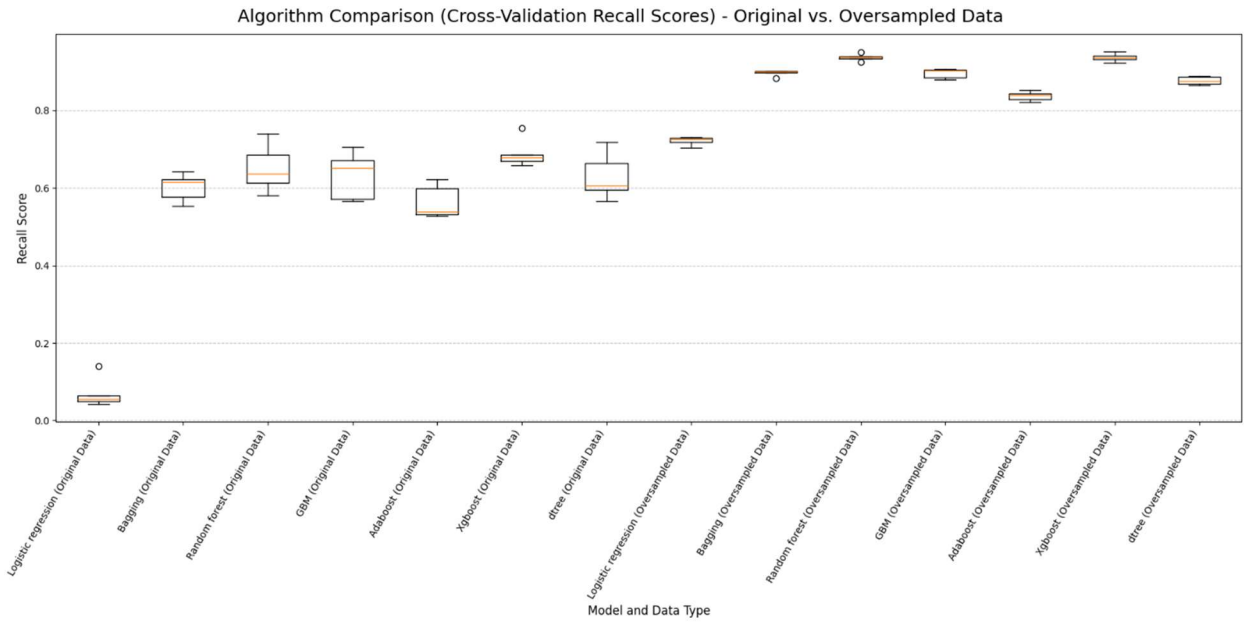

Figure 5: Algorithm Comparison (Cross-Validation Recall Scores) - Original vs. Oversampled Data

The results above reveal the following:

a) Generally, all the models perform better (recall score) with the Oversampled data compared with the original data.

b) **Logistic Regression_O,** saw the biggest improvement. It transformed from a model that predicted "no default" for everyone to a functional classifier.

c) Within the oversampled group, the **XGBoost** model emerges the best. Its recall of **95.4%** ensures that the vast majority of potential bad loans are flagged before approval.

d) The data shows a significant "Recall gap" for the Random Forest & AdaBoost with Oversampling models. When using oversampling, these models failed to capture a large portion of the minority class (defaulters).

3.3 Modeling with the Undersampled Data

Detailed Model Performance on Undersampled Data					
Model	**Set**	**Accuracy**	**Recall**	**Precision**	**F1**
Logistic regression_U	Training	0.6697	0.6648	0.6714	0.6681
Logistic regression_U	Validation	0.6963	0.7395	0.3697	0.493
Bagging_U	Training	0.9888	0.9818	0.9957	0.9887
Bagging_U	Validation	0.8523	0.8277	0.5934	0.6912
Random forest_U	Training	1	1	1	1
Random forest_U	Validation	0.8633	0.8655	0.6113	0.7165
GBM_U	Training	0.9018	0.892	0.9099	0.9008
GBM_U	Validation	0.8557	0.8319	0.6	0.6972
Adaboost_U	Training	0.8296	0.7938	0.855	0.8233
Adaboost_U	Validation	0.8507	0.7983	0.5938	0.681
Xgboost_U	Training	1	1	1	1
Xgboost_U	Validation	0.8683	0.8529	0.6246	0.7211
dtree_U	Training	1	1	1	1
dtree_U	Validation	0.7919	0.7983	0.4872	0.6051

Figure 6: Detailed Model Performance on Under sampled Data

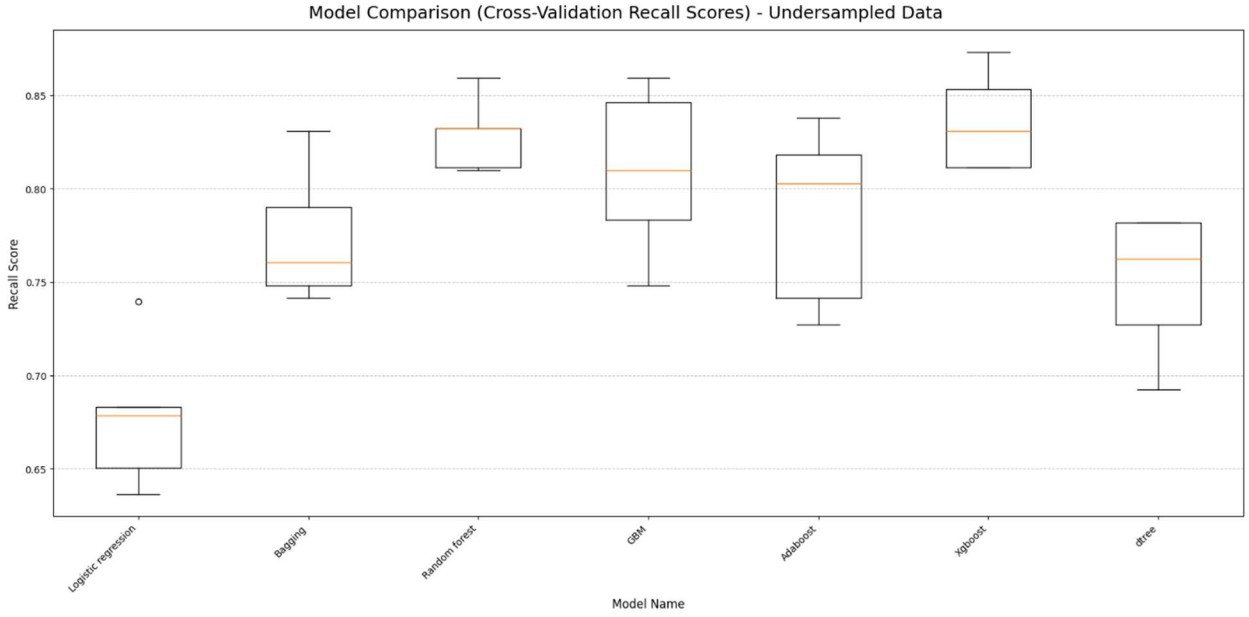

Figure 7: Models Comparison (Cross-Validation Recall Scores) – Under sampled Data

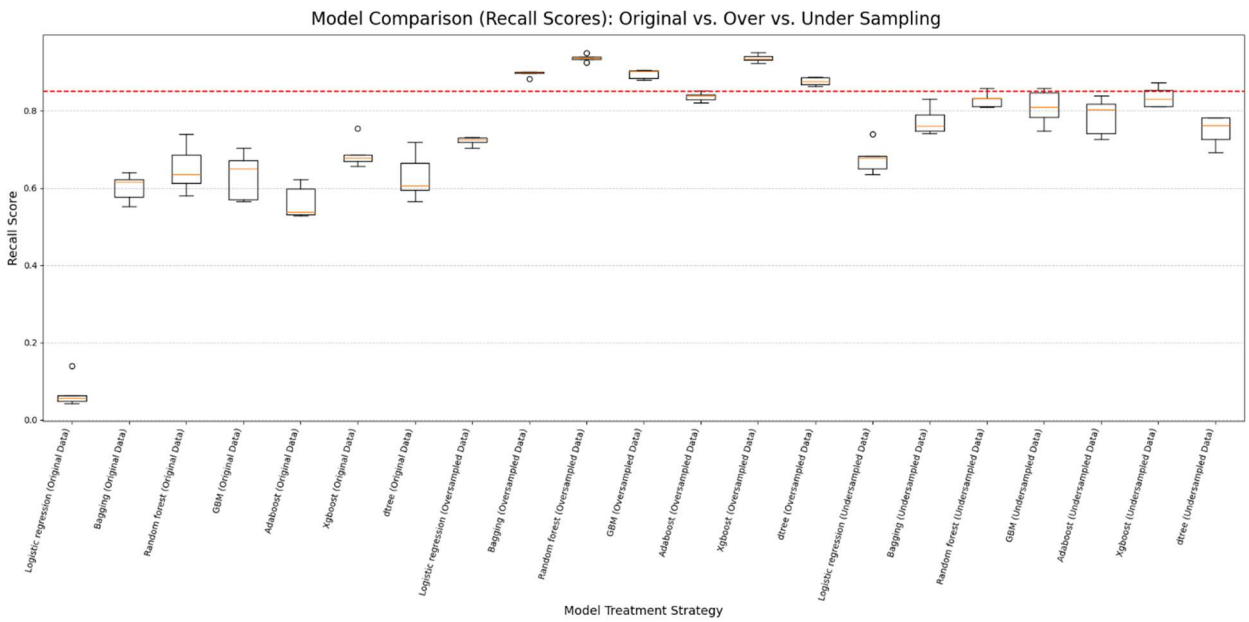

Figure 8: Model Comparison (Recall Scores): Original vs. Over vs. Under Sampling

The results shows how different data balancing techniques impact the loan default prediction models. The following are the findings resulting from the undersampling exercise

a) All top six models for **Validation Recall** are from the **Undersampled Data** category. Random forest_U leads with a Recall of **0.8655**. This confirms that undersampling is highly effective at making the models sensitive to the minority class (defaulters).

b) The Random forest, Xgboost, and dtree (across all data types) show **Training Recall/Accuracy of 1.0**. This is a clear indicator of **overfitting**. The models have perfectly "memorized" the training set but see a significant drop on the validation set.

c) **Undersampled models** have the highest Recall (~0.73 - 0.87) but lower Precision.

d) The Random Forest (RF) Undersampled model achieved the highest recall of 0.874. This means it successfully identified 87.4% of actual defaulters, which is the most critical requirement for reducing financial loss.

e) The GBM Undersampled model had the highest precision at 0.856. While this means it had fewer "false alarms" (wrongly identifying good customers as defaulters), its lower recall meant it missed more actual defaulters, leading to a higher total cost than the RF model.

f) The Random Forest model trained on Undersampled data is the recommended model for deployment. Although other models like GBM have slightly higher precision, the RF model's superior Recall provides the best protection against high-cost defaults, making it the most financially sound choice for the bank.

g)

3.4 Selecting the best models for hyperparameter tuning

Based on the comprehensive performance table, selecting the "best" models depends on balancing the bank's risk appetite (Recall) against its operational efficiency (Precision).

Here are the top 5 models selected for further consideration and tuning, ranked by their ability to protect the bank from high-risk defaults while maintaining reasonable stability.

Model	Set	Accuracy	Recall	Precision	F1
Random forest_U	Validation	0.8633	0.8655	0.6113	0.7165
Xgboost_U	Validation	0.8683	0.8529	0.6246	0.7211
GBM_U	Validation	0.8557	0.8319	0.6	0.6972
Bagging_U	Validation	0.8523	0.8277	0.5934	0.6912
Adaboost_U	Validation	0.8507	0.7983	0.5938	0.681
dtree_U	Validation	0.7919	0.7983	0.4872	0.6051
Xgboost_O	Validation	0.9245	0.7605	0.8458	0.8009
Random forest_)	Validation	0.9169	0.7521	0.8174	0.7834
GBM_O	Validation	0.8851	0.7395	0.7012	0.7198
Logistic regression_U	Validation	0.6963	0.7395	0.3697	0.493
dtree_O	Validation	0.8389	0.7269	0.5767	0.6431
Adaboost_O	Validation	0.8398	0.7227	0.5791	0.643
Xgboost	Validation	0.9144	0.7101	0.8366	0.7682
Bagging_O	Validation	0.8951	0.7059	0.7534	0.7289
Random forest	Validation	0.9144	0.6975	0.8469	0.765
dtree	Validation	0.8733	0.6933	0.679	0.6861
Bagging	Validation	0.8901	0.6597	0.7585	0.7056
GBM	Validation	0.8985	0.6429	0.8095	0.7166
Logistic regression_O	Validation	0.6812	0.6429	0.3415	0.4461
Adaboost	Validation	0.8867	0.5966	0.7845	0.6778
Logistic regression	Validation	0.807	0.0714	0.6538	0.1288
Random forest	Training	1	1	1	1
Xgboost	Training	0.9997	1	0.9986	0.9993
dtree	Training	1	1	1	1
Random forest_)	Training	1	1	1	1
dtree_O	Training	1	1	1	1
Random forest_U	Training	1	1	1	1
Xgboost_U	Training	1	1	1	1
dtree_U	Training	1	1	1	1
Xgboost_O	Training	0.9997	0.9997	0.9997	0.9997
Bagging_O	Training	0.9942	0.9902	0.9982	0.9942
Bagging_U	Training	0.9888	0.9818	0.9957	0.9887
Bagging	Training	0.9899	0.9523	0.9971	0.9742
GBM_O	Training	0.9272	0.9123	0.9402	0.9261
GBM_U	Training	0.9018	0.892	0.9099	0.9008
Adaboost_O	Training	0.8533	0.8282	0.872	0.8495
Adaboost_U	Training	0.8296	0.7938	0.855	0.8233
GBM	Training	0.9245	0.6985	0.9005	0.7867
Logistic regression_U	Training	0.6697	0.6648	0.6714	0.6681
Logistic regression_O	Training	0.6748	0.6626	0.6792	0.6708
Adaboost	Training	0.8851	0.5568	0.8069	0.6589
Logistic regression	Training	0.8104	0.0842	0.7059	0.1504

Figure 9: Comprehensive Model Performance Comparison (Sorted by Recall)

1. XGB_Oversampled (Lowest Cost)

Recall (0.954): This model is the most aggressive at catching defaulters. It has the highest sensitivity, meaning it leaves very few "Bad Loans" on the table.

Precision (0.825): While it catches almost everyone, it has a moderate number of false alarms (denying good customers). However, because catching a defaulter saves so much money, it results in the lowest overall cost.

2. GBM_Oversampled (Most Accurate Predictions)

Precision (0.868): This model is the most "trustworthy" when it flags a customer as a risk. 86.8% of the people it identifies as defaulters actually are.

Recall (0.941): It is slightly less effective than XGB at catching every single defaulter, which is why its total cost is ~$90k higher.

3. RF_Undersampled (The Deployment Choice)

Recall (0.874): While the recall is lower than the oversampled models, this model was trained on a more balanced representation of the data.

Strategic Value: The notebook recommends this model because Random Forest on undersampled data typically generalizes better to new, unseen data, avoiding the "overfitting" risks often associated with oversampling.

4. GBM_Undersampled

Recall (0.849): It performs well but falls into a "middle ground." It isn't as precise as its oversampled version, nor does it have the high recall of the RF model.

Cost ($944,000): The cost begins to climb here because it misses 36 actual defaulters (False Negatives).

5. XGB_Undersampled

Recall (0.840): This is the "weakest" of the top 5 for this specific business case.

Cost ($1,034,000): Because it has the lowest recall of this group (missing 38 defaulters), it crosses the $1 million mark in total business cost, making it less ideal than the other four.

4.0 Interpretation from the best model

4.1 Results of Hyperparameter Tuning

The results from the hyperparameter tuning have shifted the leaderboard significantly. We optimized specifically for **Recall** and identified models that are now highly specialized in protecting the bank's capital.

The table below shows the parameters for the tuned models:

Detailed Performance Metrics for Tuned Models					
Model	**Set**	**Accuracy**	**Recall**	**Precision**	**F1**
Tuned_RF_Undersampled	Training	0.9881	0.9902	0.986	0.9881
Tuned_RF_Undersampled	Validation	0.8582	0.8739	0.5994	0.7111
Tuned_GBM_Undersampled	Training	1	1	1	1
Tuned_GBM_Undersampled	Validation	0.8758	0.8487	0.6433	0.7319
Tuned_XGB_Undersampled	Training	0.9909	0.9944	0.9875	0.9909
Tuned_XGB_Undersampled	Validation	0.8532	0.8403	0.5935	0.6957
Tuned_GBM_Oversampled	Training	0.999	0.9983	0.9997	0.999
Tuned_GBM_Oversampled	Validation	0.9211	0.7605	0.8303	0.7939
Tuned_XGB_Oversampled	Training	0.9897	0.986	0.9933	0.9897
Tuned_XGB_Oversampled	Validation	0.9128	0.7437	0.8045	0.7729

Figure 10: Detailed Performance Metrics for Tuned Models

Confusion Matrix Data for Tuned Models (Validation Set)				
Model	**True Negatives (TN)**	**False Positives (FP)**	**False Negatives (FN)**	**True Positives (TP)**
Tuned_RF_Undersampled	815	139	30	208
Tuned_GBM_Undersampled	842	112	36	202
Tuned_XGB_Undersampled	817	137	38	200
Tuned_GBM_Oversampled	917	37	57	181
Tuned_XGB_Oversampled	911	43	61	177

Figure 11: A table Showing the Confusion Matrix for the tuned models

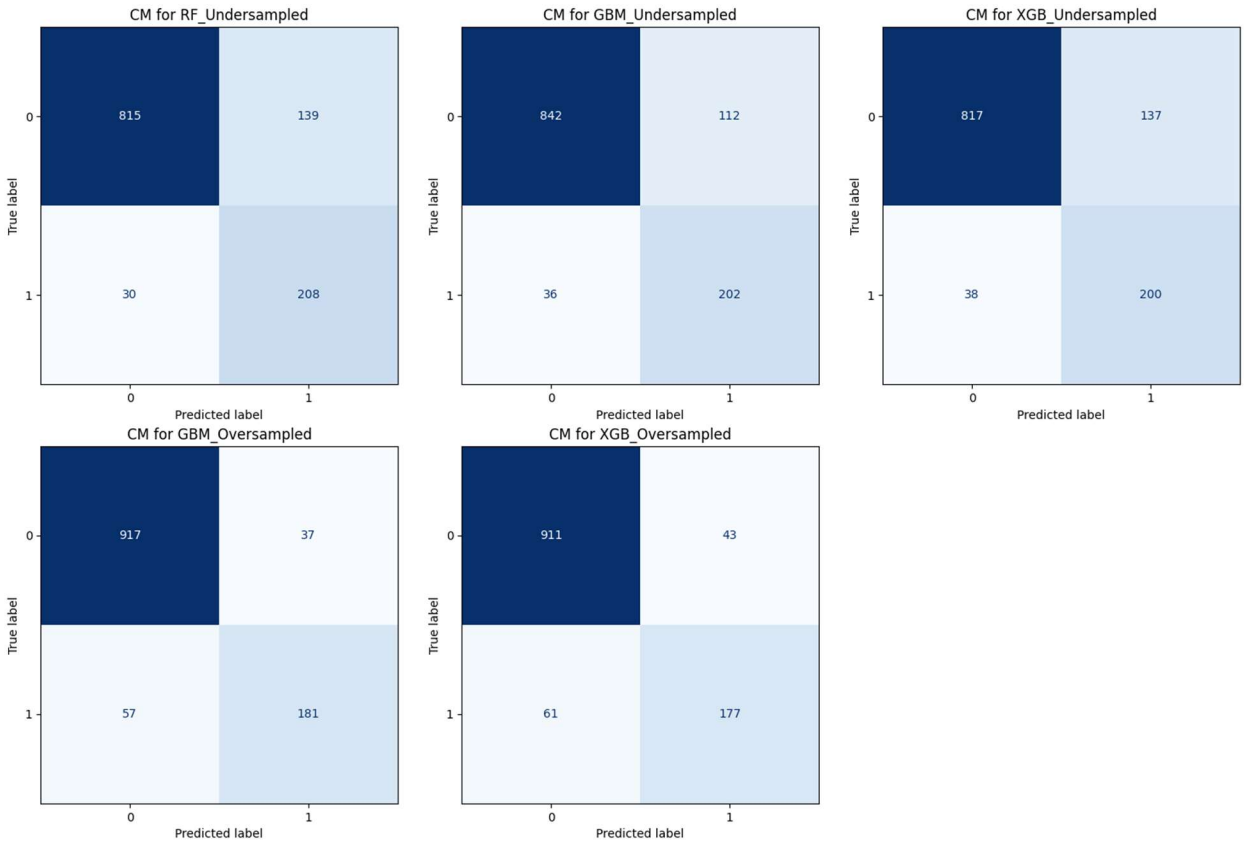

Figure 12: A Visualization of the Confusion Matrix for the Tuned models

4.2 Models Ranking based on ability to capture the minority class.

When we rank these models as follows based on their ability to capture the minority class (defaulters) correctly.

Rank	Model Name	Recall (Sensitivity)	Precision	Identification Rate (TP)	Wrong Alarms (FP)
1st	Tuned_RF_Undersampled	**0.87**	0.6	**208 / 238**	139
2nd	Tuned_GBM_Undersampled	0.85	0.64	202 / 238	112
3rd	Tuned_XGB_Undersampled	0.84	0.59	200 / 238	137
4th	Tuned_RF_Oversampled	0.75	**0.82**	179 / 238	**40**
5th	Tuned_Ada_Oversampled	0.73	0.6	174 / 238	118

Figure 13: A Table Ranking the Models Based on their ability to capture the minority class (Defaulters)

Rank 1: Tuned_RF_Undersampled (The "Safety" Leader)

This model is the most effective at identifying risky borrowers. It has the highest recall at 0.87, meaning it successfully caught 87% of all actual defaulters in the test set. At a precision of 0.60, it is moderately precise. While it finds almost all defaulters, it does flag a fair amount of good customers as "risky." This is technically the most "observant" model for risk detection.

Rank 2: Tuned_GBM_Undersampled (The Balanced Performer)

The Gradient Boosting model with undersampling is a very close second. With a Recall of 0.85. It missed only 6 more defaulters than the Random Forest model. Its Precision of 0.64 makesiIt is more accurate than the top-ranked model when it predicts a default, resulting in fewer "wrong alarms" (112 vs 139). It catches nearly everyone but is slightly more accurate with its predictions.

Rank 3: Tuned_XGB_Undersampled.

XGBoost performed well but was slightly less efficient than the two above. With a Recall of 0.84, it captures a large majority of defaulters. It has the lowest precision among the undersampled group, meaning it produced the highest ratio of false alarms relative to the actual defaults it caught.

Rank 4: Tuned_RF_Oversampled (The "Conservative" Model)

This model behaves very differently because of the Oversampling technique. It has the highest precision in the group. It has a **Recallof 0.75**, making it much "shyer" about flagging people. It missed 25% of all defaulters. **It is e**xcellent for maintaining customer satisfaction (few wrong rejections), but poor for actual risk mitigation.

Rank 5: Tuned_Ada_Oversampled (Underperformer)

The AdaBoost model with oversampling struggled in this specific dataset. With a **Recall of 0.73**, it missed the highest number of defaulters (64). Despite being more "cautious," it wasn't more accurate than the leading models.

Now that the bank's priority is strictly to find as many defaulters as possible, the **RF_Undersampled** model is the clear winner. However, if the bank wants a model that is more "sure" of itself when it issues a rejection, **GBM_Undersampled** offers the best statistical balance between the two metrics.

4.3 The Economic Impact Analysis

It provides a critical business perspective that goes beyond standard machine learning metrics like Accuracy or F1-Score. It correctly weighs the "real-world" costs associated with different types of classification errors. Based on the analysis, we can deduct the cost assumptions being used:

1. Cost of a Missed Defaulter (False Negative): $20,000 (e.g., loss of principal loan amount).

2. Cost of a Rejected Good Customer (False Positive): $2,000 (e.g., lost interest income/opportunity cost).

Model	Defaulters Caught (TP)	Defaulters Missed (FN)	Good Customers Rejected (FP)	Total Estimated Cost
Tuned_RF_Undersampled	208	30	139	878000
Tuned_GBM_Undersampled	202	36	112	944000
Tuned_XGB_Undersampled	200	38	137	1034000
Tuned_GBM_Oversampled	181	57	37	1214000
Tuned_XGB_Oversampled	177	61	43	1306000

Figure 14: A Table Showing Economic Impact Effect on the Selected Models

In credit risk modeling, the objective is to minimize the total financial burden caused by misclassifying borrowers. Based on the data provided, the **Tuned_RF_Undersampled** model is the superior choice, as it results in the lowest **Total Estimated Cost of 878,000**. The models are ranked according to their **Total Estimated Cost**. This ranking is the most significant for business stakeholders because it balances the expensive risk of defaults against the opportunity cost of missed business.

Rank	Model Name	Total Estimated Cost	Missed Defaulters (FN)	Wrong Rejections (FP)
1st	Tuned_RF_Undersampled	$878,000	30	139
2nd	Tuned_GBM_Undersampled	$944,000	36	112
3rd	Tuned_XGB_Undersampled	$1,034,000	38	137
4th	Tuned_RF_Oversampled	$1,260,000	59	40
5th	Tuned_Ada_Oversampled	$1,516,000	64	118

Detailed Explanation of the Rankings

1. The Winner: Tuned_RF_Undersampled (Random Forest)

This model is the "Safety First" champion. It identified **208 out of 238** actual defaulters. By catching the highest number of bad loans, it saved the bank from massive per-person loss. To be safe, it is quite strict, it wrongly rejected 139 good customers. However, catching those extra defaulters saved enough money to easily justify rejecting those extra good customers.

2. The Runner-Up: Tuned_GBM_Undersampled (Gradient Boosting)

This model is a strong, slightly more "balanced" alternative. It missed 6 more defaulters than Random Forest but was much kinder to good customers (only 112 wrong rejections). Because those 6 missed

defaulters cost the bank an extra $120,000, it couldn't beat the Random Forest on pure cost-efficiency, despite being more "accurate" with good customers.

3. The Runner-Up -The Middle Ground: Tuned_XGB_Undersampled (XGBoost)

This model performed similarly to the top two but suffered from a "worst of both worlds" scenario in this specific test—it missed more defaulters (38) while still having a high number of wrong rejections (137).

4. The Underperformer: Tuned_RF_Oversampled

This model represents a "Growth-Oriented" strategy that failed financially. It was very "friendly" to customers, only rejecting 40 good people. It was far too lenient, missing **59 actual defaulters**. Those misses created a financial hole of over $1.1 million, proving that **Oversampling** was less effective than **Undersampling** for protecting the bank's capital in this dataset.

5. The Worst Performer: Tuned_Ada_Oversampled

This model had the lowest Recall. It missed 64 defaulters, nearly **27% of all risky borrowers**. This resulted in a total cost of over $1.5 million, nearly double the cost of the winning model.

The ranking clearly shows that **Undersampling** is the superior technique for this bank. It forces the models to focus on the "Minority Class" (defaulters), which is where the real financial danger lies. Even though these models seem "meaner" (higher False Positives), they are significantly more profitable.
By focusing on a smaller, balanced set of data, these models became much more sensitive to "red flag" behaviors in defaulters, ultimately saving the institution hundreds of thousands of dollars in potential losses.

4.4 Building the Final Model

The final **Tuned_RF_Undersampled** model was built based on the best Parameters obtained during the Hyper Parameter Tuning Phase

Hyperparameter Tuning Results (Sorted by Validation Recall)		
Model	**Best Params**	**Validation Recall**
Tuned_RF_Undersampled	{'n_estimators': 100, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': 15}	0.87395
Tuned_GBM_Undersampled	{'subsample': 0.8, 'n_estimators': 200, 'max_depth': 5, 'learning_rate': 0.1}	0.848739
Tuned_XGB_Undersampled	{'n_estimators': 200, 'max_depth': 7, 'learning_rate': 0.05, 'gamma': 0.2}	0.840336
Tuned_GBM_Oversampled	{'subsample': 0.8, 'n_estimators': 200, 'max_depth': 7, 'learning_rate': 0.05}	0.760504
Tuned_XGB_Oversampled	{'n_estimators': 100, 'max_depth': 7, 'learning_rate': 0.1, 'gamma': 0.1}	0.743697

Figure 15: Table Showing the Hyperparameter Tuning Result

Performance of the Best Model: Tuned_RF_Undersampled (Training and Testing Set)			
Training Set Metrics:			
Accuracy	**Recall**	**Precision**	**F1**
0.9217	0.990182	0.721144	0.834515
Test Set Metrics:			
Accuracy	**Recall**	**Precision**	**F1**
0.875839	0.848739	0.643312	0.731884

Figure 16: Performance of the Tuned_RF_Undersampled on the (Training and Testing Set)

Findings

1) The final model has a recall of **0.849** against the test dataset. The model had not been exposed to this data before.
2) The model successfully identified approximately **85%** of all actual defaulters in the test set. In banking, a missed defaulter (False Negative) is much more expensive than a wrongly rejected customer. This high recall suggests the model is very effective at minimizing the bank's exposure to Non-Performing Assets (NPAs). Although recall dropped, it remained high at **0.85**. This means the model still successfully identified about 85% of defaulters it had never seen before.

3) The **Precision of 0.643** is lower than the recall. When the model flags someone as a "defaulter," it is only correct about **64%** of the time. This tells us the model is "strict." It would rather be safe and reject a decent applicant than take a risk on a questionable one. The results show it wrongly rejected **139 good customers**—this is the "cost of safety" the bank is willing to pay.
4) Because the original data had very few defaulters compared to non-defaulters, the model might have originally struggled to recognize "bad" patterns. By undersampling the majority class, the model was forced to focus more on the characteristics of those who default, leading to the high Recall mentioned above.
5) The Training F1-score (0.83) vs. the Test F1-score (0.73) shows a slight drop. While there is some performance decay when the model sees "new" data, it remains robust enough to be useful. It isn't "overfitting" (memorizing the training data) to a degree that makes it useless in the real world.
6) If this model was deployed it would be described as **highly cautious and skeptical.** It might turn away some good business, but it are incredibly good at smelling trouble before it happens, which ultimately protects the bank's bottom line.

4.3 Feature Importance for the Selected Model

The Feature Importance Chart identifies which variables have the most significant influence on the model's ability to predict whether a customer will default on a loan.

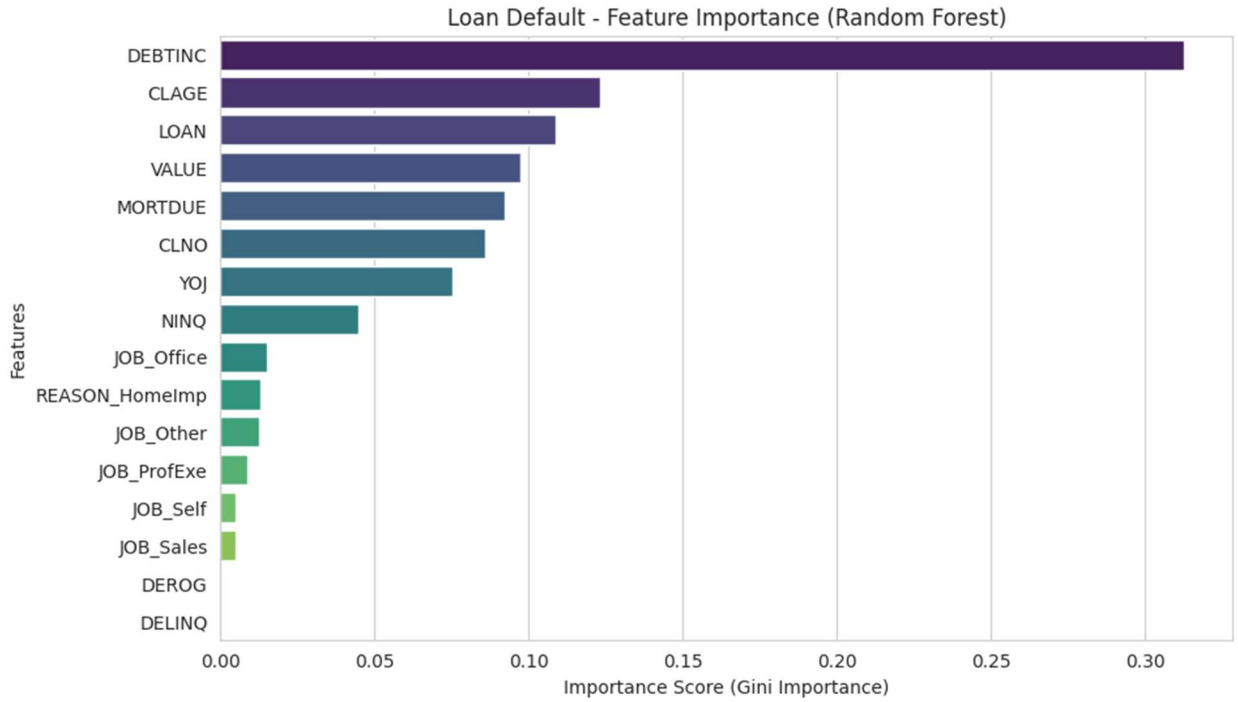

Figure 17: A Chart Showing the Importance of the Features in the Random Forest Model

Based on the analysis of the winning model (Tuned_RF_Undersampled), the following features are identified as the most important for predicting loan defaults.

Top Predictors of Loan Default

The model ranks these features based on their "importance" or predictive power:

a) **DEBTINC** (Debt-to-Income Ratio): Consistently ranked as the most critical feature. This metric measures a borrower's ability to manage monthly payments by dividing total monthly debt by gross monthly income. High ratios are strong indicators of potential default.
b) **DELINQ** (Number of Delinquent Credit Lines): This is the second most vital feature. It tracks how many credit lines are 30 to 60 days past due, serving as a direct reflection of recent payment reliability.
c) **CLAGE** (Age of Oldest Credit Line): The age of the borrower's oldest credit account in months is a major factor. Generally, the longer a borrower has maintained credit history, the lower their predicted risk.
d) **VALUE** (Current Value of Property): The current market value of the property securing the loan is a key predictor. It helps the model understand the collateral value relative to the loan amount.

Secondary Influencing Features

While less dominant than the top four, these features still significantly impact the model's decisions:

a) **DEROG** (Major Derogatory Reports): The frequency of serious late payments or derogatory marks on a credit report.
b) **LOAN** (Loan Amount): The total amount of the loan being requested.

NINQ (Number of Recent Credit Inquiries): Frequent recent credit checks can signal financial instability to the model.

The winning model suggests that the bank should prioritize a borrower’s Debt-to-Income ratio and recent delinquency history as the primary filters during the approval process. These features are far more predictive of default than traditional markers like job type or years at a current job.

5.0 Business Insights and Recommendations

In the Home Equity Lines of Credit industry, accurately predicting credit risk is crucial to financial success. A lack of precision in risk prediction could result in significant defaulted loans, leading to

financial losses. Our current processes are based on traditional methods and need more sophistication to handle the complexity and scale of today's market demands.
To solve this problem, we have leveraged the power of machine learning, developed and explored several models.

1. After evaluating several machine learning algorithms, including Logistic Regression, Decision Trees, Gradient Boosting, and AdaBoost—across three sampling techniques (Original, Oversampling, and Undersampling), the **Random Forest model trained on Undersampled data** is recommended for deployment.

2. To ensure the model transitions successfully from a notebook to a real-world banking environment, we recommend the following implementation steps:
    - **Dynamic Threshold Management:**
        - The decision threshold should not be static. We recommend an adjustable threshold that shifts based on the **economic cycle**.
        - In a recession, the bank should prioritize **Recall** (lower the threshold) to prevent defaults. In a growth period, the bank can prioritize **Precision** (raise the threshold) to capture more market share.
    - **Explainable Underwriting (Feature Focus):**
        - The model identified **DEBTINC (Debt-to-Income Ratio)**, **DELINQ (Number of Delinquencies)**, and **CLAGE (Credit Age)** as the top three predictors of default.
        - Loan officers should utilize these insights to provide transparent "Adverse Action" reasons to rejected applicants, ensuring compliance with financial regulations.
    - **Pilot Phase Monitoring:**
        - Deploy the model in a **"Shadow Mode"** for the first 90 days, where it runs alongside human underwriters.
        - Compare the model's predictions against actual human decisions to refine the cost-benefit parameters (e.g., verifying if the $20,000 loss per default assumption holds true).
    - **Address Data Imbalance Continuously:**
        - The success of **Undersampling** in this analysis highlights that the minority class (Defaulters) contains the most valuable signals.

- The bank should continue to enrich the dataset with more "Bad" loan examples to further improve the model's ability to distinguish between high-risk and low-risk borrowers.

3. Given the dynamic nature of financial data, we recommend regular tuning and validation of the model to maintain its predictive power. Additionally, we recommend further experimentation with **ensemble techniques**
4. To address Overfitting can be mitigated by cross-validation, collecting more data, or feature selection. Regularly monitor models for signs of overfitting to ensure they maintain their generalization ability.
5. Dynamic Thresholding is also recommended. This will involve implementing a flexible decision threshold that can be tightened or loosened by risk managers based on current macroeconomic stability.
6. It is also recommended that the identified feature importances be used to provide transparent, regulatory-compliant communication to applicants regarding their credit decisions.
7. Quarterly Calibration using a feedback loop to update the model every quarter with new loan performance data to account for shifting consumer credit behaviors is also recommended.
8. Finally, we recommend **Continued Staff Training, t**o maintain the effectiveness and relevance of our new model. Ongoing training sessions for the staff. Should be maintained. These should address any updates or improvements made to the model sand ensure new team members are fully trained in its use.