Machine Learning Models Prediction of Credit Default Risk: algorithm suitability to balance accuracy and interpretability for financial decision-making

Mindaugas Kondrotas
The Bachelor of Science (B.Sc.) in Management and Technology
Student Nr. 03747728
Technical University of Munich, Germany

Micaelle Nogueira de Carvalho
The Bachelor of Science (B.Sc.) in Management and Technology
Student Nr. 03761870
Technical University of Munich, Germany

Chenxing Shen
The Bachelor of Science (B.Sc.) in Management and Technology
Student Nr. 03762550
Technical University of Munich, Germany

***Abstract*—Predicting credit default risk is crucial for financial decision-making. This study evaluates three machine learning models—SVM, Hybrid (Logistic Regression + Extra Trees Classifier), and LightGBM—on the UCI Default of Credit Card Clients dataset (30,000 observations) against the Logistic Regression. Models were trained with feature selection, encoding, and scaling, and assessed using ROC-AUC, Precision, Recall, and F1-score. The findings show a trade-off between accuracy and explainability, where LightGBM suits automated risk prediction, and the Hybrid Model offers a compromise for decision-making.**

Keywords—Credit risk, machine learning, classification models, financial decision-making, LightGBM, interpretability.

I. Introduction

In business and financial environments, credit risk assessment is critical and allows to predict customer credit risk optimizing lending decisions and effectively reducing default risk while improving overall operational efficiency. Therefore, the challenge is to obtain a precise and reliable method for credit risk assessment. Traditional techniques use heuristic-based assessments, which may lack accuracy [1], and static rules that fail to adapt to potentially evolving borrower behaviour [2]. The question is whether the group of the three machine learning (ML) models (Light Gradient Boosting Machine, Support Vector Machine and a Hybrid model combining Extra Trees Classifier and Logistic Regression) trained on diverse financial and demographic features, can outperform traditional methods, represented by Logistic Regression, in classifying borrowers based on their default risk, and if so, then which algorithm provides the most suitable balance of accuracy and interpretability. On the one hand, ML algorithms can process large-scale datasets efficiently, uncovering complex relationships between multiple financial indicators [3]. On the other hand, Python provides robust libraries [4], such as `scikit-learn`, `LightGBM`, and ensemble methods enabling the development of high-performing predictive models [5]. Therefore, classification accuracy potentially could be improved while ensuring reliable credit scoring using ML.

II. Dataset Description

A. Dataset Source and Characteristics

The dataset used was the *Default of Credit Card Clients* from UCI [6] containing 30,000 credit card account observations with 24 attributes and the final binary default attribute specifying whether the client defaulted on payment the next month. The first attribute was the unique identification of each credit account holder, followed by the attribute specifying the amount of credit granted. The following four attributes provided demographic detail related to each observation: gender, education level, marital status and age. The subsequent six attributes indicated payment delays in months from April to September (included) of 2005. Furthermore, the next six attributes contain the amount of bill statement - a monthly report issued to the credit account holders in months from April to September (included) of 2005. The succeeding six attributes indicate the amount of previous payments in months from April to September (included) of 2005. The final attribute is the variable, which indicates the default status in the following month, that the machine learning models were trained to predict in this study. A more detailed breakdown by attributes and their description of the dataset is included in the Appendix section *I. Dataset Details*.

In the dataset data, lower credit limits are associated with higher default rates [7]. The distributions of key variables of financial attributes like credit limits and bill amounts are notably skewed [8]. The amount of granted credit shows a right-skewed shape indicating most clients have lower credit limits, and progressively fewer clients have very high limits and both defaulting and non-defaulting customers exhibit this skewed credit limit distribution [9]. Moreover, past bill amounts and payment amounts share a similar skew with the majority of monthly bills and payments being relatively low, with a long tail toward large values resulting in distributions with a sharp peak near zero and a few very large outliers, indicating that while typical clients maintain moderate balances and payments, a small number of observations involve extremely high bills or payments [10]. In addition, the client base is skewed toward 60% female customers versus 40% male [11]. Therefore, all of these observations indicate that the dataset is somewhat imbalanced.

B. Data Preprocessing

The implementation of the following preprocessing steps was done before modelling to structure the dataset in a manner which facilitates effective training of the models used while mitigating any potential issues related to data inconsistency, scale differences, and categorical feature representation. Detailed information on software packages and libraries utilized for analysis is provided in Appendix section II. Resources.

The dataset was loaded using `pandas.read_excel()` within the function `get_dataset(filename)`, with the first row skipped using `skiprows=1`. There was no imputation or removal of missing values done as the dataset has no missing values. Then, the unique identification attribute, which has no predictive value, was removed in `prepare_data(data, encoding="ohe")`. Afterwards, the final attribute indicating the default status was extracted separately for supervised learning via `target = data["DEFAULT"]`. Categorical variables, if selected, were processed using one-hot encoding (OHE) in `features = pd.get_dummies(features)`.

This was done to ensure compatibility with machine learning algorithms and the categorical features had to be expanded into multiple binary columns. Alternatively, label encoding was implemented as an optional encoding method in `encoding="le"`, replacing categorical values with numerical labels where applicable.

To ensure that raw numerical attributes do not disproportionately influence models due to differences in magnitude, the numerical attributes were normalized using `StandardScaler()` from `sklearn.preprocessing`. This transformed all numerical attributes to have zero mean and unit variance, which is particularly beneficial for models sensitive to feature magnitudes, such as logistic regression and Support Vector Machines (SVM).

The dataset was then split into training and test sets using `train_test_split()` from `sklearn.model_selection`, with 80% of the data allocated for training (`X_train, y_train`) and the remaining 20% reserved for testing (`X_test, y_test`). The split was generated randomly but reproducibly, using `random_state=42`, ensuring consistent results across different executions. The training set was used for model fitting and validation, whereas the test set was used for evaluating the final model performance, ensuring that model evaluation was performed on unseen data.

III. Methodology

This section outlines the methodology used to evaluate the research question. It explains what machine learning models and evaluation metrics to assess performance were used as well as the experimental setup, data partitioning, feature processing, and optimization strategies employed.

A. Machine Learning Models

For predicting credit default risk, the following machine learning models were implemented and compared in Python.

- **Logistic Regression**: a basic linear model with easy to interpret coefficients that was used as a baseline to represent traditional methods. Implemented as `LogisticRegression()` with hyperparameter tuning via `GridSearchCV()`.
- **SVM**: a kernel-based classifier with the ability to captured non-linear patterns in the data was chosen to see whether predictive accuracy would improve. Implemented from `sklearn.svm` as `SVC(kernel="rbf", probability=True)`.
- **Hybrid**: a combination of **Logistic Regression** and **Extra Trees Classifier** - a high classification task performance method employing multiple randomized decision trees was selected to help examine the relevant factors for decision-making and result interpretability. Implemented from `sklearn.ensemble` as `ExtraTreesClassifier()`, this model generates feature probabilities which are then used as input to Logistic Regression.
- **Light Gradient Boosting Machine (LightGBM)**: generally considered an efficient and accurate tree algorithm, boosting the gradient, and well suited for larger datasets was included to capture complex feature interactions and non-linearities. Implemented as `LGBMClassifier()`.

Logistic regression, not an ML method, is generally favoured in credit risk assessment due to its ease of interpretability [12]. As it provides probabilistic predictions, it is generally deemed, in financial decision-making, suitable for direct interpretation, where the probability thresholds are used to guide lending approvals [12]. Furthermore, the model's coefficients contribute some insight on different default risk indicators [12]. Hence, it was used as a baseline model to represent the traditional techniques using heuristic-based assessments. The aim was to evaluate whether the three remaining more complex models significantly outperform logistic regression, justifying a shift from traditional methods to machine learning approaches.

B. Evaluation Metrics

The assessment of the models' performance was done on the test set by using multiple evaluation metrics to ensure a holistic evaluation of predictive accuracy and reliability. The comparison was done to ensure that the predictive patterns actually helped in evaluating the trade-off between a model's accuracy and its transparency and thus made economic sense.

A classification report was generated via `classification_report` from `sklearn.metrics` for (i) **precision** – to measure the fraction of predicted positive instances that are truly positive, (ii) **recall** – to measure the fraction of actual positive instances that were correctly identified (iii) **F1-score** – to obtain the harmonic mean of precision and recall. This provided a balanced measure for both default and non-default cases and overall accuracy. High recall score for the "default" class was considered indicative that the model is capturing most of the actual defaulters, while high precision indicated that the loans flagged by the model as "default" were in fact actual default cases. The F1-score summarizes recall and precision into a single number and was used to evaluate performance under class imbalance. Measuring accuracy provides an overall success rate, yet it was considered with caution if the default rate was very low as this measure can be misleading in imbalanced datasets such as the one used.

In addition, using `roc_auc_score` and `roc_curve` from `sklearn.metrics`, the Receiver Operating Characteristic Area Under the Curve (ROC-AUC) score was computed as `roc_auc = roc_auc_score(y_test, model.predict_proba(X_test)[:, 1])`. This was done to evaluate the model's trade-off between true positive and false positive rates and its ability to discriminate between default and non-default cases across all classification thresholds.

Furthermore, to assess interpretability, the contribution of features was considered. For the LightGBM model, feature importance scores were examined to determine the relative influence on the prediction outcome of each input feature. For logistic regression, the magnitude and sign of its coefficients indicating how each feature affects default risk were considered. Although the Extra Trees component of the Hybrid model could provide feature importances based on impurity (Gini importance), this was not explicitly used in this study due to incompatible metrics, stacking, and complex interactions.

C. Experimental Setup

The following methods, in addition to the ones already previously outlined, were used to ensure that the chosen models can be trained efficiently and deployed on larger datasets.

Each model was trained on the same training dataset and under the same conditions. For logistic regression, SVM, and Hybrid models scikit-learn library was used via the `LogisticRegression`, `SVC`, and `ExtraTreesClassifier` classes. The LightGBM model was implemented with the `lightgbm.LGBMClassifier` application programming interface (API). To avoid any data leakage from the training to the test set, preprocessing was integrated into the model training process using scikit-learn pipelines.

To mitigate overfitting and assess model generalizability, hyperparameter tuning was applied to Logistic Regression using `GridSearchCV` with a 5-fold cross-validation strategy. This process systematically searched for the best regularization strength (`C`) to optimize model performance. The search was guided by the ROC-AUC score, a metric suitable for imbalanced datasets, and was implemented as `GridSearchCV(LogisticRegression(), param_grid, cv=5, scoring="roc_auc")`.

Instead of grid search, LightGBM used early stopping to determine the optimal number of boosting iterations. This technique monitored the performance of the validation and automatically stopped the training when no further improvement could be detected, thus preventing overfitting. This was implemented using `callbacks=[early_stopping(100, verbose=True)]`. To improve computational efficiency of LightGBM, Python's garbage collector `gc.collect()` was used between training runs to free memory.

Each model was trained and evaluated under consistent conditions. While Logistic Regression used `GridSearchCV`, SVM and the Hybrid model were trained with default parameters. The Hybrid model combined Extra Trees for feature learning with Logistic Regression for classification. LightGBM was trained using its built-in gradient boosting optimization.

IV. Results and Discussion

A. Model Performance Comparison

Considering overall performance, the base-line Logistic Regression reference model and SVM performed similarly, and both were outperformed by LightGBM and the Hybrid models, as can be seen in the Table 1.

TABLE I. PERFORMANCE COMPARISON OF CREDIT DEFAULT PREDICTION MODELS

Overall Performance Table				
Model	***AUC Score***	***Precision***	***Recall***	***F1-Score***
Logistic Regression	0.7269	0.6920	0.2361	0.3521
SVM	0.7143	0.6781	0.3321	0.4458
Hybrid Model	0.7505	0.6298	0.2605	0.3685
LightGBM	0.7816[a]	0.4568	0.6283	0.5290

[a] Highest AUC Score. (*LightGBM: 0.7816*)

Fig. 1. Summary of the performance metrics for the evaluated machine learning models, including Logistic Regression, SVM, Hybrid Model (Extra Trees + Logistic Regression), and LightGBM. The models are compared based on their ROC-AUC score, Precision, Recall, and F1-Score. LightGBM achieved the highest ROC-AUC (0.78), indicating superior overall predictive performance, while Logistic Regression provided better interpretability but lower recall.

LightGBM model produced the highest ROC-AUC (0.78) score, indicating that it was best suited in distinguishing defaulters form non defaulters. The Hybrid (Extra Trees + Logistic Regression) performed well with an AUC of 0.75, suggesting it can provide a balance between predictive performance and interpretability. Whereas Logistic Regression (AUC = 0.73) and SVM (AUC = 0.71) exhibited a slightly lower predictive power, however, provided interpretability advantages.

In addition, when examining the result obtained in the ROC curve analysis, it can also be seen that LightGBM had the highest True Positive Rate (TPR) across all classification thresholds (Fig. 2), thus, the best discriminatory ability.

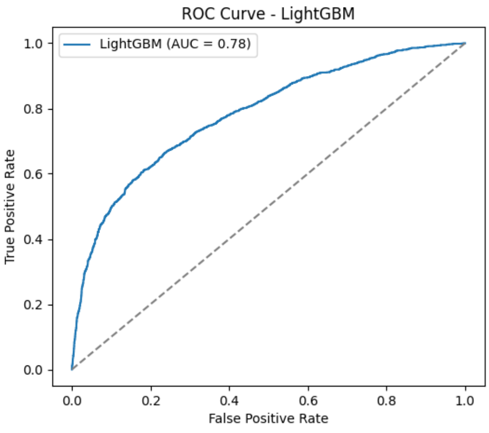

Fig. 2. ROC curve for the LightGBM model, achieving the highest AUC of 0.78, indicating superior performance in distinguishing between defaulting and non-defaulting clients

Again, the Hybrid model (Fig. 3) also performed better than both, SVM (Fig. 4) and Logistic Regression (Fig. 5), showing that ensemble learning can provide some added benefit.

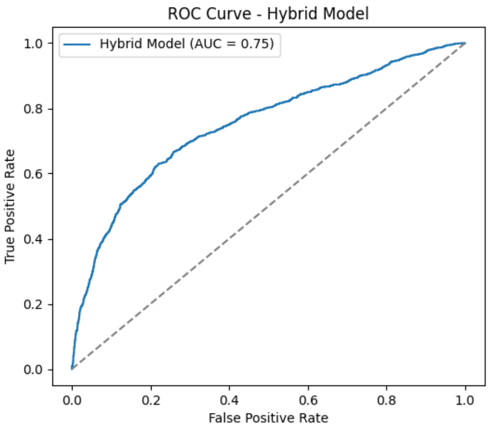

Fig. 3. ROC curve for the Hybrid Model (Extra Trees + Logistic Regression), with an AUC of 0.75, demonstrating improved predictive performance over the SVM and the Logistic Regression.

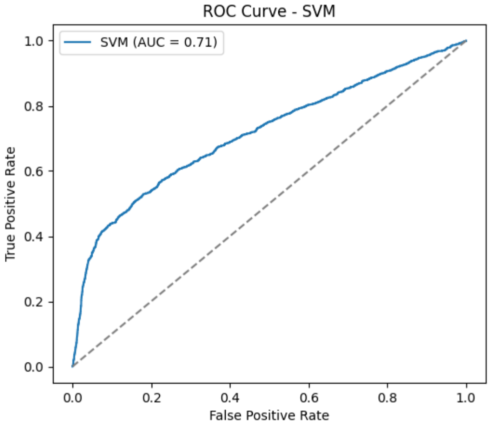

Fig. 4. ROC curve for the SVM model, showing an AUC of 0.71. This suggests slightly lower predictability than Logistic Regression.

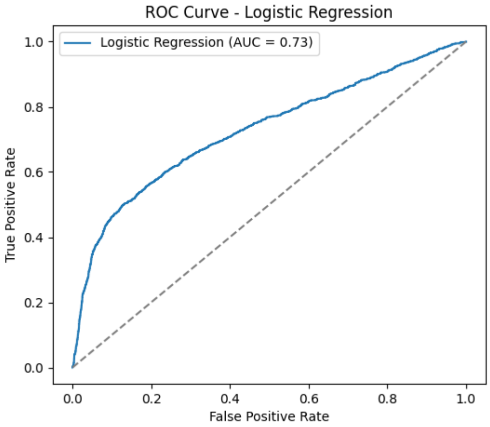

Fig. 5. ROC curve for the Logistic Regression model, illustrating the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR). The area under the curve (AUC) is 0.73, indicating moderate predictive performance.

SVM and Logistic Regression models exhibited resembling ROC curve shapes, which suggests that these models may have had difficulties dealing with the dataset's complex decision boundaries.

In addition, when examining each model's confusion matrix (see Appendix, section III. Confusion Matrices), once more, LightGBM (Fig. 7) demonstrated significantly higher recall for default cases, meaning it was more sensitive to predicting defaults correctly that the other models. The Hybrid model (Fig. 8) displayed a balance between precision and recall, which makes it a good choice for maintaining both interpretability and predictive performance. Logistic Regression (Fig. 9) had the lowest recall for detecting defaulters, meaning it misclassified defaults as non-defaults the most. SVM (Fig. 10) displayed similar results to Logistic Regression, though, it had a slightly better recall which seems come at the cost of increased false positive rate.

B. Interpretation of Results

The best performing model was LightGBM. It achieved the highest AUC score (0.78), thus, indicating superior predictive performance when compared to all other models used. This is postulated to be the result of the gradient boosting and optimized feature selection employed, which allowed it to capture non-linear relationships in the data. In addition, it identified more actual defaulters as can be seen from the fact that it exhibited higher recall while reducing false negatives.

Furthermore, when considering the importance of the observation attributes contained in the dataset, it can be seen that the two most influential features were `BILL_AMT1` (most recent bill amount) and `LIMIT_BAL` (credit limit), which suggests that recent credit behaviour may be strongly correlated with default risk, as in Fig. 6. However, payment history features had almost no importance except for `PAY_0` (Fig. 6.) which played a minor role.

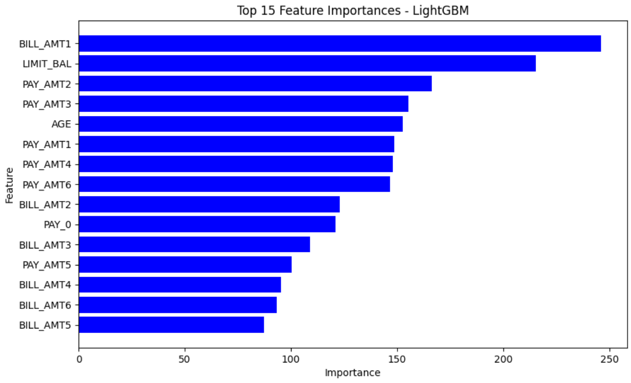

Fig. 6. Feature importance ranking of the LightGBM model, highlighting the top 15 most influential features. The `BILL_AMT1` (latest bill statement amount) and `LIMIT_BAL` (credit limit) were the strongest predictors of default risk, followed by repayment history variables (`PAY_AMT2`, `PAY_AMT3`).

Age (`AGE`) and the past bill amounts (`BILL_AMT2`, `BILL_AMT3`, `BILL_AMT4`, `BILL_AMT6`, `BILL_AMT5`) seem to have had less significant importance (Fig. 6), which indicates that demographic and historical financial behaviour were secondary factors, even though, they still played a role.

When considering model bias, overfitting, and generalization, clearly, LightGBM showed the highest generalization ability, outperforming other models consistently across all metrics. The slightly poorer performance displayed by SVM and Logistic Regression could be due to their reliance on linear decision boundaries, which may not capture the complex, non-linear relationships present in the dataset. On the other hand, the Hybrid model may have overfitted, due to it's the deep tree decision model employed in the Extra Trees Classifier, so it performed well on the training data as it probably simply memorised the specific details of the training set rather than learning general trend and thus subsequently failed to perform equally well on the test set. The reason for the better performance of the LightGBM is the boosting algorithm that it uses, which allows it to build many simple models and gradually improve them. In addition, the use of early stopping and regularization prevented overfitting in the LightGBM, ensuring the model generalizes well to unseen data.

In summary, Hybrid model, because of the use of Extra Tress Classifier, may have overfitted as it became to specialised on the training dataset, whereas, LightGBM avoided this issue by stopping the training when adding more trees was not improving performance on validation data, and, by controlling complexity which in turn makes it more generalizable and applicable for real-world credit default predictions. However, unlike Logistic Regression, which provides explicit coefficients, LightGBM only offers feature importance scores, which indicate general trends rather than the specific reasoning for its predictions. Therefore, LightGBM has the lowest interpretability since it relies on gradient boosting with multiple decision trees which makes it difficult to trace individual decisions.

V. Conclusion

A. *Summary of Findings*

The most effective model for distinguishing between defaulting and non-defaulting clients was LightGBM model which outperformed all other models with the highest ROC-AUC score (0.78). However, the Hybrid model was the most balanced in terms of accuracy and interpretability, with AUC score of 0.75. On the other hand, Logistic Regression (AUC = 0.73) and SVM (AUC = 0.71) showed lower performance in predictability but were useful nonetheless because of their interpretability. According to feature importance analysis, recent bill amounts (`BILL_AMT1`) and credit limits (`LIMIT_BAL`) were the strongest predictors of default risk, while secondary but significant role was also played by repayment history (`PAY_0`, `PAY_2`, `PAY_3`). Moreover, overfitting was observed in Hybrid model, likely as a result of the use of deep decision trees in the Extra Trees Classifier. On the contrary, with LightGBM overfitting was avoided due to the use of early stopping and regularization. Finally, the following three trade-offs were observed. First, highest accuracy but lowest interpretability was seen in LightGBM. Second was between the balanced performance and interpretability in the Hybrid model and the third was in Logistic Regression that was the most interpretable but had the lowest recall for default cases.

B. *Recommendations for Practical Applications*

According to the scope of this study, the best suited model for automated credit risk assessment, if predictive accuracy is of the highest priority, would be the LightGBM model. In addition, the Hybrid model would be a viable option for credit analysts who want to consider both reasonable accuracy and interpretability and perhaps further optimization may improve its generalization ability while maintaining its interpretability. Logistic Regression still relevant in the contexts where clear explanations of risk factors are needed, such as for regulatory compliance and decision-making transparency.

C. *Limitations of the Study*

The dataset had a class imbalance, which probably affected model performance. Even if ROC-AUC was used to address this, other balancing techniques (e.g., SMOTE or cost-sensitive learning) could possibly improve the results [13]. Furthermore, advanced tuning methods (e.g., Bayesian Optimization or Random Search) could have been used for further refinement of model performance [13]. Moreover, although basic preprocessing was implemented, additional steps such as feature selection stability testing that could enhance reliability were not used [13]. Finally, model evaluation was conducted on a single dataset, leaving the model's performance untested in real-world credit scoring systems, so future work should test the models on additional datasets from financial institutions.

References

[1] S. Lessmann, B. Baesens, H. V. Seow, and L. C. Thomas, “Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research,” *Eur. J. Oper. Res.*, vol. 247, no. 1, pp. 124–136, 2015, doi: 10.1016/j.ejor.2015.05.030.

[2] S. Bhatore, L. Mohan, and Y. R. Reddy, “Machine learning techniques for credit risk evaluation: A systematic literature review,” *J. Bank. Financ. Technol.*, vol. 4, no. 1, pp. 111–138, 2020, doi: 10.1007/s42786-020-00019-7.

[3] N. Suhadolnik, J. Ueyama, and S. Da Silva, “Machine learning for enhanced credit risk assessment: An empirical approach,” *J. Risk Financ. Manag.*, vol. 16, no. 12, Art. 496, 2023, doi: 10.3390/jrfm16120496.

[4] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[5] G. Ke *et al.*, “LightGBM: A highly efficient gradient boosting decision tree,” in *Adv. Neural Inf. Process. Syst. 30 (NeurIPS 2017)*, 2017, pp. 3146–3154

[6] I. Yeh, “Default of Credit Card Clients [Dataset],” UCI Machine Learning Repository, 2009. [Online]. Available: https://doi.org/10.24432/C55S3H, DOI:10.24432/C55S3H.

[7] M. Amendola, D. Gadler, R. Manetti, Gemma Martini, “An analysis of a Taiwanese Credit Card Dataset,” GitHub, 2019, [Online]. Available: https://danyele.github.io/lecture_notes/ReportGroup1.pdf. Accessed: Mar. 3, 2025.

[8] Ainslie, “Credit card default prediction analysis,” Kaggle, [Online]. Available: https://www.kaggle.com/code/ainslie/credit-card-default-prediction-analysis. Accessed: Mar. 3, 2025.

[9] S. W. Chu, “Classification- Default Payments of Credit Card Clients in Taiwan from 2005,” GitHub, [Online]. Available: https://github.com/Sarah-chu/Classification-Credit-Card-Default-Payment/blob/main/Default%20of%20Credit%20Card%20Clients.ipynb. Accessed: Mar. 3, 2025

[10] Guest Blog, “Logistic regression in R: A classification technique to predict credit card default,” DataScienceDojo, 2022, [Online]. Available: https://datasciencedojo.com/blog/logistic-regression-in-r-tutorial. Accessed: Mar. 3, 2025

[11] A. Reksi, “Credit Card (Default) Clients Profile,” Rpubs, 2020, [Online]. Available: https://rpubs.com/reksi/cc_default. Accessed: Mar. 3, 2025.

[12] J. N. Crook, D. B. Edelman, and L. C. Thomas, “Recent developments in consumer credit risk assessment,” *European Journal of Operational Research*, vol. 183, no. 3, pp. 1447–1465, 2007.

[13] I. Brown and C. Mues, "An experimental comparison of classification algorithms for imbalanced credit scoring datasets," Expert Systems with Applications, vol. 39, no. 3, pp. 3446–3455, 2012, doi: 10.1016/j.eswa.2011.09.033.

APPENDIX

I. DATASET DETAILS

Innitial attributes:

1) ID: categorical variable uniquely identifying each client account

2) LIMIT_BAL: amount of given credit in NT dollars incluing individual and family/supplementary credit.

Demographic attributes:

3) SEX: categorical gender variable (1 = male, 2 = female)

4) EDUCATION: categorical education level variable (1 = graduate school, 2 = university, 3 = high school, 4 = others, 5 = unknown, 6 = unknown)

5) MARRIAGE: categorical marital status variable (1 = married, 2 = single, 3 = others)

6) AGE: numerical age variable in years

The delay of the past payment in a specific month:

7) PAY_0: repayment status in September 2005 (-1 = pay duly, 1 = payment delay for one month, 2 = payment delay for two months, … , 9 = payment delay for nine months and above)

8) PAY_2: repayment status in August 2005 (same scale as before)

9) PAY_3: repayment status in July 2005 (same scale as before)

10) PAY_4: repayment status in June 2005 (same scale as before)

11) PAY_5: repayment status in May 2005 (same scale as before)

12) PAY_6: repayment status in April 2005 (same scale as before)

The amount of bill statement in a specific month:

13) BILL_AMT1: amount of bill statement in September, 2005 (NT dollar)

14) BILL_AMT2: amount of bill statement in August 2005 (NT dollar)

15) BILL_AMT3: amount of bill statement in July 2005 (NT dollar)

16) BILL_AMT4: amount of bill statement in June 2005 (NT dollar)

17) BILL_AMT5: amount of bill statement in May 2005 (NT dollar)

18) BILL_AMT6: amount of bill statement in April 2005 (NT dollar)

The amount of previous payment in a specific month:

19) PAY_AMT1: amount of previous payment in September 2005 (NT dollar)

20) PAY_AMT2: amount of previous payment in August 2005 (NT dollar)

21) PAY_AMT3: amount of previous payment in July 2005 (NT dollar)

22) PAY_AMT4: amount of previous payment in June 2005 (NT dollar)

23) PAY_AMT5: amount of previous payment in May 2005 (NT dollar)

24) PAY_AMT6: amount of previous payment in April 2005 (NT dollar)

The default attribute specifying whether the client defaulted on payment the next month:

25) default.payment.next.month: 1 = the credit card holders defaulted, 0 = the credit card holder did not default.

II. RESOURCES

The following section lists, in ascending alphabetical order, all the software packages, libraries, and tools that were used in data analysis and implementation in this study.

- **classification_report**: Scikit-learn Developers, "sklearn.metrics.classification_report," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html. Accessed: Mar. 14, 2025.
- **confusion_matrix**: Scikit-learn Developers, "sklearn.metrics.confusion_matrix," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html. Accessed: Mar. 14, 2025.
- **ExtraTreesClassifier**: Scikit-learn Developers, "sklearn.ensemble.ExtraTreesClassifier," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html. Accessed: Mar. 14, 2025.
- **Garbage Collector (gc):** Python Software Foundation, "gc — Garbage Collector interface," Python Standard Library. [Online]. Available: https://docs.python.org/3/library/gc.html. Accessed: Mar. 14, 2025.
- **GridSearchCV**: Scikit-learn Developers, "sklearn.model_selection.GridSearchCV," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. Accessed: Mar. 14, 2025.
- **KFold**: Scikit-learn Developers, "sklearn.model_selection.KFold," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html. Accessed: Mar. 14, 2025.
- **LabelEncoder**: Scikit-learn Developers, "sklearn.preprocessing.LabelEncoder," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html. Accessed: Mar. 14, 2025.
- **LightGBM (early_stopping)**: Microsoft Corporation, "lightgbm.early_stopping," LightGBM Documentation. [Online]. Available: https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.early_stopping.html. Accessed: Mar. 14, 2025.
- **LogisticRegression**: Scikit-learn Developers, "sklearn.linear_model.LogisticRegression," Scikit-learn Documentation. [Online]. Available: https://scikit-

learn.org/stable/modules/generated/sklearn.linear_model. LogisticRegression.html. Accessed: Mar. 14, 2025.

- **Matplotlib** (pyplot): Matplotlib Development Team, "matplotlib.pyplot," Matplotlib Documentation. [Online]. Available: https://matplotlib.org/stable/api/pyplot_summary.html. Accessed: Mar. 14, 2025.
- NumPy: NumPy Developers, NumPy Documentation. [Online]. Available: https://numpy.org/. Accessed: Mar. 14, 2025.
- **pandas**: pandas Development Team, pandas Documentation. [Online]. Available: https://pandas.pydata.org/. Accessed: Mar. 14, 2025.
- **PdfPages** (Matplotlib backend): Matplotlib Development Team, "matplotlib.backends.backend_pdf.PdfPages," Matplotlib Documentation. [Online]. Available: https://matplotlib.org/stable/api/backend_pdf_api.html. Accessed: Mar. 14, 2025.
- **roc_auc_score**: Scikit-learn Developers, "sklearn.metrics.roc_auc_score," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html. Accessed: Mar. 14, 2025.
- **roc_curve**: Scikit-learn Developers, "sklearn.metrics.roc_curve," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html. Accessed: Mar. 14, 2025.
- **Seaborn**: Seaborn Development Team, Seaborn: statistical data visualization. [Online]. Available: https://seaborn.pydata.org/. Accessed: Mar. 14, 2025.
- **StandardScaler**: Scikit-learn Developers, "sklearn.preprocessing.StandardScaler," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. Accessed: Mar. 14, 2025.
- **SVC**: Scikit-learn Developers, "sklearn.svm.SVC," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. Accessed: Mar. 14, 2025.
- **train_test_split**: Scikit-learn Developers, "sklearn.model_selection.train_test_split," Scikit-learn Documentation. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. Accessed: Mar. 14, 2025.classification_report:

III. Confusion Matrices

This section provides the set of confusion matrices generated by the code for each method used:

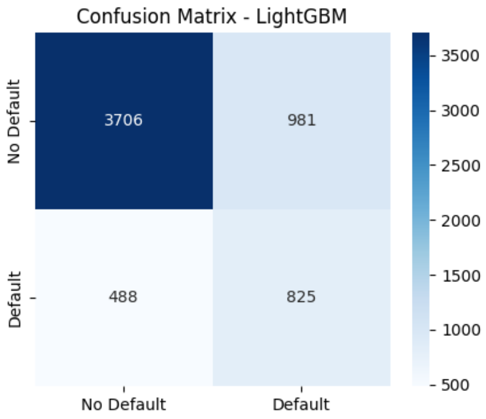

Fig. 7. Confusion matrix for LightGBM, revealing significantly **better recall** for identifying default cases compared to linear models, though at the cost of more false positives.

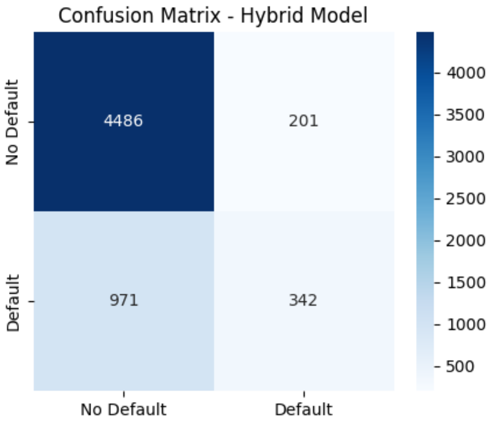

Fig. 8. Confusion matrix for the Hybrid Model, achieving a balance between precision and recall, with moderate performance across both classes.)

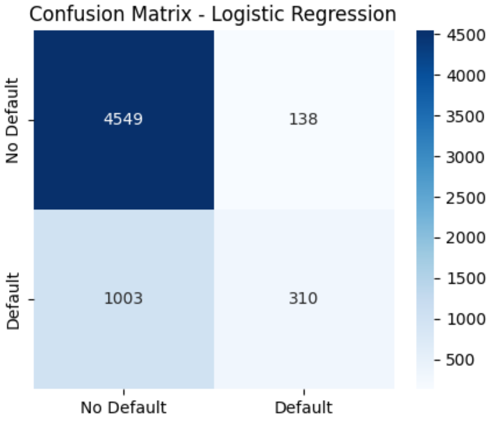

Fig. 9. Confusion matrix for Logistic Regression, indicating that while it correctly classifies most non-default cases, it misclassifies a substantial number of default cases, suggesting lower recall.

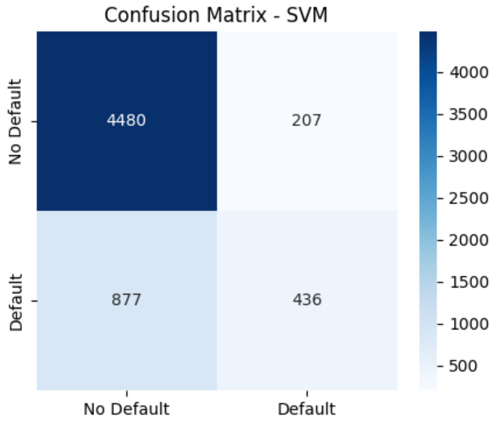

Fig. 10. Confusion matrix for SVM, showing similar classification patterns to Logistic Regression, with slightly better recall but higher false positives