American International Theism University Scientific Research Journal (AITUSRJ)
ISSN (E): 2578-3882**, ISSN (P):** 2578-3874

10.63094/AITUSRJ.25.4.2.3

A Comparative Study of MATLAB-Based Classification Algorithms for Loan Approval Prediction

Maheem Khowaja
Department of Robotics and Artificial Intelligence, SZABIST University Karachi, Pakistan
msds24101153@szabist.pk

Laraib Awan
Department of Robotics and Artificial Intelligence, SZABIST University Karachi, Pakistan
msds24101115@szabist.pk

Mughair Aslam Bhatti
Department of Robotics and Artificial Intelligence, SZABIST University Karachi, Pakistan
dr.mughair@szabist.edu.pk

Yusra Saeed
Department of Robotics and Artificial Intelligence, SZABIST University Karachi, Pakistan
yusra.saeed@szabist.edu.pk

Abstract **–** This work presents a detailed comparison of different machine learning algorithms with the help of MATLAB by using Classification Learner to predict loan approvals. We were using applicant data on income, credit history, and level of education and we tried everything: decision trees, neural networks. We have found on consideration significant sacrifices on accuracy against the ease of computation that can guide financial institutions in selecting the appropriate model according to their requirements.

Index Terms **–** Loan approval prediction, machine learning, Classification Learner, MATLAB, ensemble methods, neural networks, SVM, k-nearest neighbors, decision trees, training accuracy, testing accuracy, overfitting, generalization, model comparison, credit risk modeling, financial analytics, computational efficiency.

I. INTRODUCTION

The accurate prediction of LOAN approval is an imperative mission in financial institutions as errors may either result in the loss of lending opportunities or an intolerable level of credit risk. Conventionally, several banks have been making use of rule-based decision systems, which can be interpreted, although they tend to fail to capture the complex, non-linear relationships among Australian bank applicant data. Machine learning models, in turn, constitute more data-driven variants with the potential to reveal multifaceted figures and enhance the reliability of decisions. Driven by such a possibility, we have been able to provide a thorough studyon benchmarking with MATLAB Classification Learner by assessing the effectiveness of different algorithm classifications using actual loan application data. We aimed to identify that trade predictive accuracy and computational efficiency to serve as scalable and risk aware foundations for lending decision making. The research will be based on the current reality of the banking environment and technology. We strictly simulated the normal loan screening procedure that is currently being carried out by the institution by considering the following attributes of the applicants: level of income, employment history, education background and credit score. The goal was straightforward aiming at building a machine learning-based system, which is not only correct and computationally efficient but also equitable. An important aspect in overcoming this challenge is to make sure that not well qualified applicants will be wrongfully rejected because of the bias caused by models or simplified decision rules. Our method achieves this through modeling real-world conditions and constraints and therefore ensures that the gap between theoretical validity and practical implementation can be closed translating to credit decisions made by lenders that are more swift, fair and reliable.

II. LITERATURE REVIEW

A vast number of studies have been performed on credit scoring and loan approval using various models including logistic regression, decision tree, support vector machines (SVM), and neural networks as models used in machine learning. [1] Many of these attempts have grown the field; however, much extant work either tests a small set of algorithms (leading to model-specific performance findings), or attempts to benchmark and compare models between platforms, datasets, or preprocessing, precluding meaningful, real-world inference about model performance.[2]

An investigation of the algorithms undertaken by our paper helps to fill this gap because it investigates most available classification algorithms on a huge scale in a controlled way through the Classification Learner app in MATLAB. [3] We made use of a standardized framework to maintain homogeneous data processing, feature processing, cross-validation, and performance indicators. This is how one can use an apple to apples comparison without the influence of the noise that can be caused by the variability of the experimental conditions.[4]

In addition to this, our study, contrary to numerous studies in academia, does not focus on the correctness of theoretical accuracy as the primary factor but considers the feasibility of implementation considered the financial institutions like computational cost, training time,

interpretability, and resistance to the class imbalance. [5] Trade-offs between training and testing accuracy to reveal the risk of overfitting or under fitting models were also examined where the presented models are likely to be deployed in high-stakes settings such as loan approvals.[6]

Our work contributes to a holistic study of benchmarking bringing together the concepts of breadth, depth, operational. relevance which can be considered as bridging the divide between the research and the practice in the industry. The results may contribute to the choice of individual machine learning models by data science teams, credit risk officers and financial engineers looking to find not only accurate models but also practical ones to produce in the case of contemporary lending environments.[7]

III. METHODOLOGY

The specifics of our methodology were aimed at emulating the experience of professionals in the level of non- programming domain experts (data analyst, business decision- maker, etc.) who do not write code but still require the machine learning to be applied. To this end, the MATLAB Classification Learner App was employed to enable non-programming domain experts to conduct machine learning experimentation without writing custom code. The main stages of our process are the following:

A. Data Preparation

We started with the loading and sufficient data processing of a real-life loan application dataset with the help of the in-built tools in MATLAB which deal with data sorting. This included:

- Interactive imputation tools one may use to clean missing values
- Preprocessing: Categorical variables insertion through selected options in the app
- Standardizing numeric attributes like income and credit scores to make it compatible with the model
- Model validation of data quality by means of integrated visualization and summary statistics panels
- There was no need for a custom code or other external packages for this step.

B. Dataset Partitioning

The most useful way the data was split by using automatic stratified splitting (in the app under the Split Data button) would be to maintain the ratio of approved to declined applications. We used:

80 percent of the training data

20 percent testing

We were also able to implement the k-fold cross-validation (k = 5) in the app to have a reliable analysis of the generalization output of each of the models.

C. Training and Comparison of Models

There are over 30 different classifiers which we trained and tested using the Classification Learner in MATLAB:

- Tree-based models - Fine Tree, Medium Tree, Coarse Tree
- Linear and Quadratic: discriminant analysis
- Gaussian and Kernel Naive Bayes
- Support Vector Machines (SVM): Linear, Quadratic,Cubic Gaussian
- k-Nearest Neighbors (k-NN):Fine, Weighted, Cosine, Cubic, and so on.
- Logistic Regression
- Ensemble choices: Bagged Trees, Boosted Trees,Reboot, Subspace
- Shallow Neural Networks

The model training was introduced under the same conditions, in terms of the standard pipeline of the app, and it was possible to compare the models fairly, avoiding any manual parameter tuning.

The models were trained in the same conditions, as the standard pipeline is used in the app, and so the comparisons could be fair and could be replicated without tuning its parameters manually.

Performance and Measures

To evaluate the individual models, we employed the number of built-in performance measures:

- Accuracy
- F1-Score
- Precision and Recall
- Area under the Curve (AUC)
- Confusion Matrices

Besides, the Classification Learner gave able visual comparisons of:

- Training / Testing Accuracy (of overfitting)
- Training time of model
- Latency (change speed)

This enabled us to not only assess its predictive power, but also practicability in the context of real banking systems.

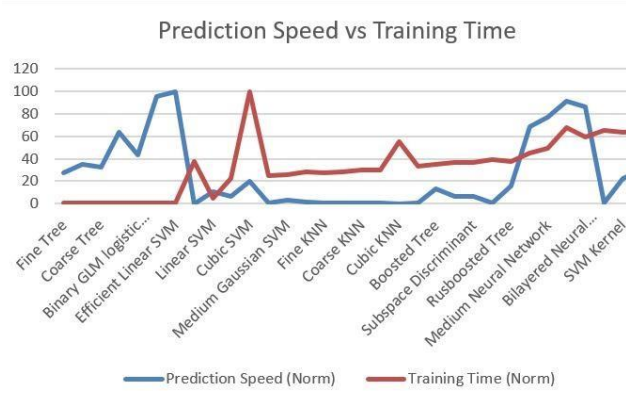

Figure 1 Prediction Speed vs. Training Time Across Classifiers: Normalized comparison showing fast predictors often have lower training times, with notable spikes for specific SVM variants.

D. Reporting and visualization

We created graphs using the visualization features that are already provided in the app:

- Parallelogram coordinate plots
- ROC and Scatter curves
- Comparison graphs of accuracy

These assisted in establishing the superior models in terms of visualization and performance, without OR the exportation of outcomes and creation of customized visualizations.

E. No-Code Approach: Re-Buildability and Re-Usability

While using our methodology, peculiarity is a completely

non-code method. All models, tests and visualizations could be created based on GUI-based controls and this process can be repeated by groups that do not have any programming skills. This is also well comfortable in the operations set ups of financial institutions where the domain experts can utilize tools such as MATLAB without having to write up the code.

This approachable but intense method even enabled us to mimic the real-world, non-programmer-friendly process of model evaluation and its artifacts, which is why our results are of maximum interest to companies that want to implement the use of AI in their credit deaccessioning projects and make this process scale and low-barrier

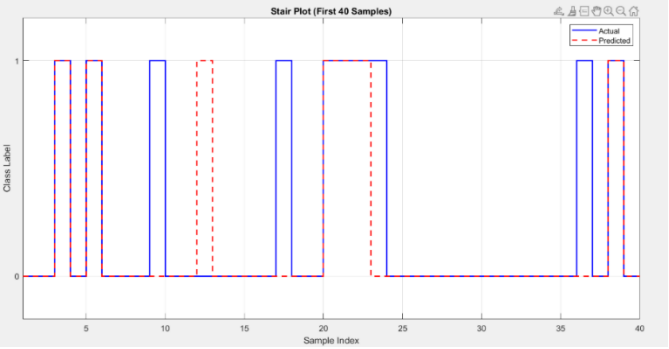

Figure 2 Stair Plot

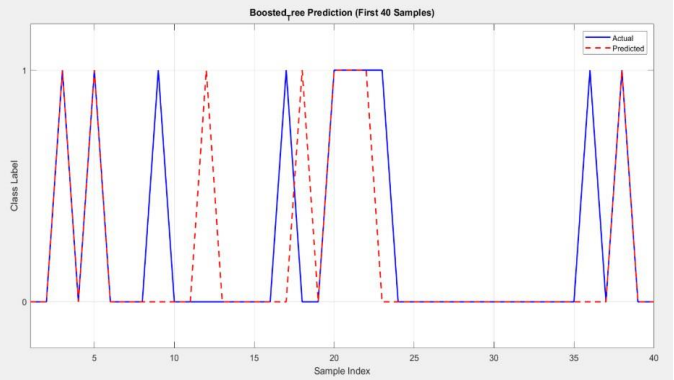

Figure 3 Boosted Tree Prediction

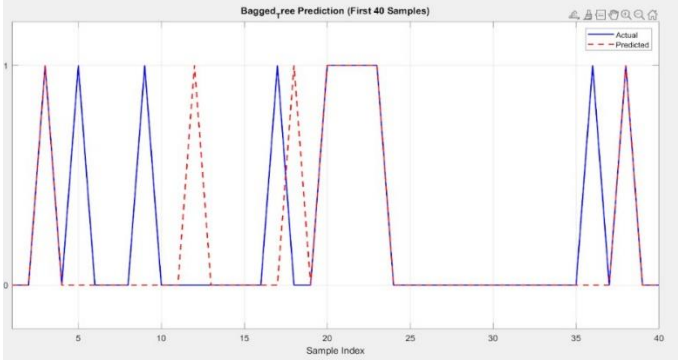

Figure 4 Bagged Tree Prediction

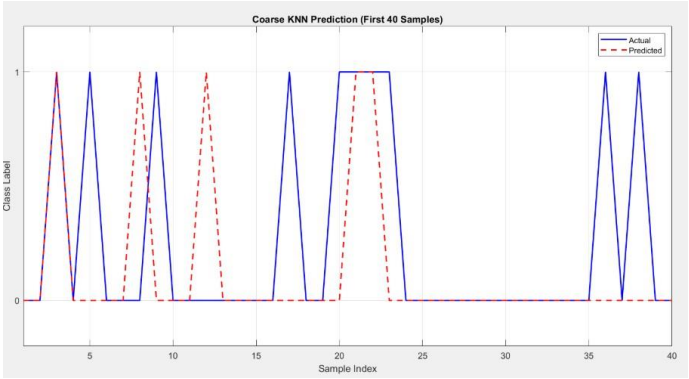

Figure 5 Coarse KNN Prediction

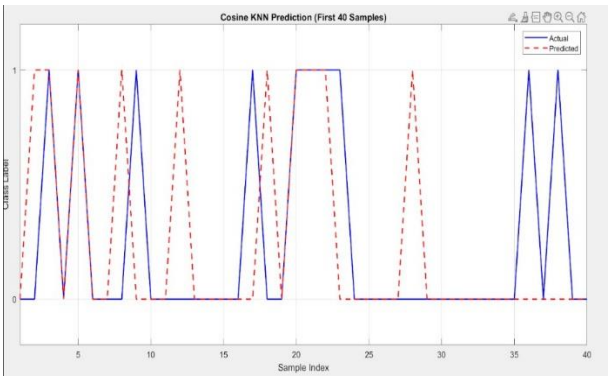

Figure 6 Cosine KNN Prediction

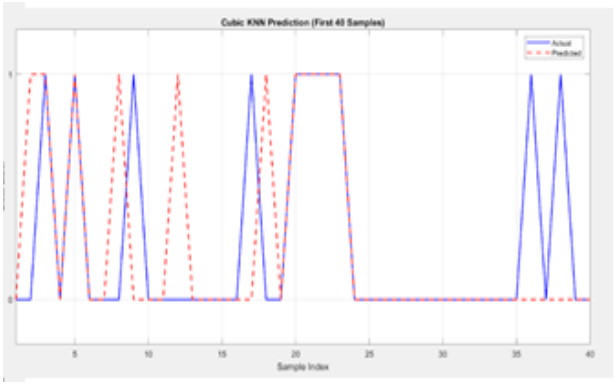

Figure 9 Cubic KNN Prediction

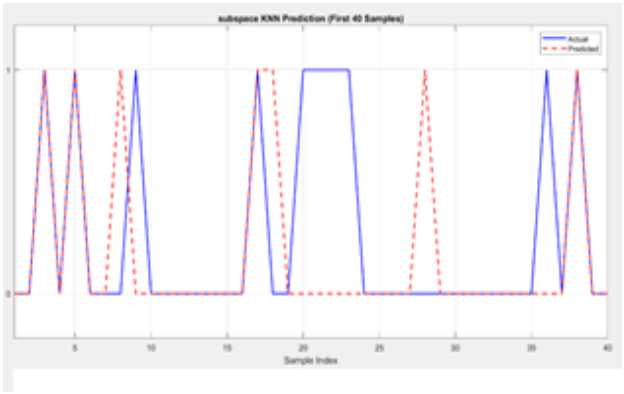

Figure 7 Subspace KNN Predictions

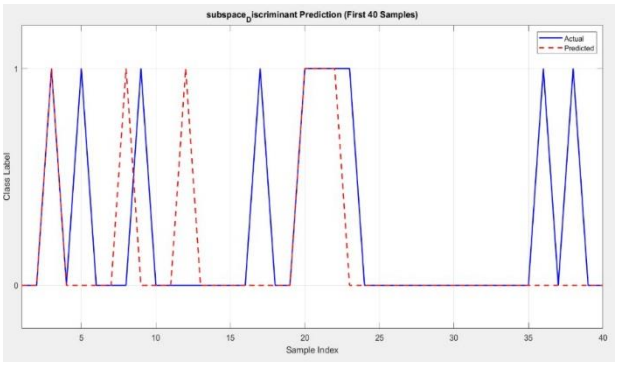

Figure 10 Subspace In discriminant Prediction

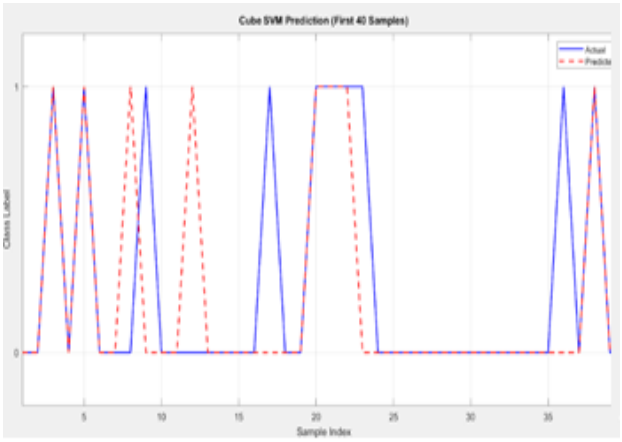

Figure 8 Cube SVM Prediction

IV. Experimental Setup

To have consistency, transparency and consequently a practical relevance, we settled on a simplified form of experimentation with the Classification Learner App in MATLAB. The emphasis was placed on the realistic process of model selection that an average model analyst or decision maker would follow as it would be conducted in a financial institution.

Model Configuration

We made the setting deliberately naive and reproducible: Settings that have been used, MATLAB has the default hyper parameters that were used to train most of the models. This is because it mirrors the field environment settings, whereby the domain experts may use pre-configured settings in a bid to run a fast experiment. Minimal Tuning, only custom tuning was used when a model always performed unsatisfactorily, or it was necessary to adjust input format. No code or optimization scripts were used, and all adjustments could be introduced with the help of the graphic interface.

Data Partitioning

To comply with an impartial and fair assessment, MATLAB was used to randomly divide the dataset by 80 percent training and 20 percent testing subset with option of stratified sampling. Training set, 5-fold cross validation

was adopted to obtain stronger performance estimates when training the models. This served to ensure that models were not merely memorizing training data and so the fact that they could easily determine how effectively each model would generalize to previously unseen loan applications.

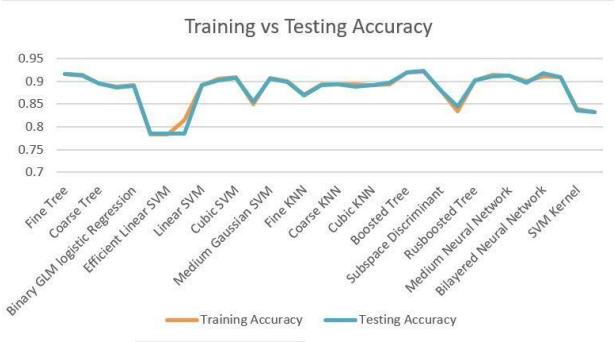

Figure 11 Training vs. Testing Accuracy Across Classifiers: Normalized comparison showing close alignment for most models, with minor generalization gaps around tree-based and SVM variants.

Evaluation Metrics

To give an all-round conclusion of model performance and ease of practical use, we traced the following important parameters of each of the classifiers:

- Training and testing accuracy:
- Time to Train
- Speed in Prediction

V. Results and Discussions

Our assessment showed that there were levels of performance of the classification algorithms where there was a distinct trade-off between being accurate, fast and being sensitive to classes. Although the overall accuracy scores had given a superficial, overall indicator of correctness, a more detailed analysis at the level of each class and the prediction behavior shown here demonstrated significant limitations including how models performed with denied loan applications (Class 1). Distinct performance variations emerged across the classification models. Ensemble methods demonstrated superior performance. Ensemble methods were the best and they included Bagged Trees and Boosted Trees, where they recorded a testing accuracy of approximately 92%. Nonetheless, upon closer examination of the class wise performance one will find a critical weakness to such models in that even though they provided high overall accuracy, they only managed to recognize around 25% of the denied applications (Class 1). It brings out an important finding which is that they did a great job on approvals (Class 0) but always misclassified high-risk applicants, an important weakness in practice when it comes to making lending decisions. The same trend was observed in other three variations with these findings substantiating the idea that such a class imbalance or feature overlap seriously affects the sensitivity of a model concerning denied cases.

On the contrary end, models such as Subspace k-NN had poor performance as they were only able to detect 35.8 percent of the rejected cases. Even one of these methods was not able to detect any rejected loans effectively reducing it to a single level predictor. Such failures highlight how one should consider beyond overall accuracy: any sensitivity to minority classes, such as lending, which are high-stakes areas.

There were also observed unique trade-offs of speed and accuracy. As an example, the Linear SVM was lightning fast with the ability to conduct up to 220,000 applications per second. This, however, was at the expense of accuracy especially on denied cases. Neural networks provided more temperate answers including average training time and above- average accuracy, yet they have also demanded more running time in proportion to less complex models. Such differences are paramount in deciding between real-time system model and batch processing system model.

Model behavior was also explained by visual prediction graphs. The Boosted and Bagged Tree models gave clean and equals patterns of predictions in comparison to actual results. However, prediction mistakes were significant (especially in samples 12-18) throughout all models. The similarity in the pattern also indicates a shared problem with the classifiers, which are probably because of edge cases or applicant profiles that have ambiguously defined features. In comparison, the graph of the Subspace k-NN prediction was so turbulent that in most cases the results fluctuate ragingly, particularly between sample 5 and 22-which characterizes its inferior predictive stability.

Several interesting patterns arose in the entire set of models. To begin with, none of the algorithms were effective in classifying denied applications properly. Second, although tree-based models were coherent overall and effective across the board, they showed its own biased pattern-IT systematically missed some patterns, particularly those related to types of applicants. Third, the ensemble of applicants that the samples under 12 to 18 referred to were also problematic to almost all the models, thus, it might be necessary to further study the features at hand or collect more data. Lastly, even some of the models largely abandoned the cases that were denied as indicative of inappropriateness to be implemented in risk-sensitive credit conditions. Overall, the findings depict that, although the ensemble trees are the best candidates regarding the overall accuracy and convenience compilation, they, and other models, run into serious problems when measured against how well they can distinguish between high-risk and low-risk applicants. To be able to implement it in the real world, accuracy should be calibrated with class sensitivity, the speed of the computation, and uniformity across a variable applicant profile.

VI. RECOMMENDATIONS

Based on this study, some major suggestions can be made to finance institutions interested in the use of machine learning in loan approval systems. To begin with, the issue of class imbalance ought to be given priority. The uniformity of poor performance of the diverse models in predicting denied applications (Class 1) indicates that such methods as SMOTE (Synthetic Minority Oversampling Technique), under- sampling, or manipulation of class weights may considerably improve the accuracy of the model by achieving a better rate of predicting risking applicants. Devoid of this, even accurate models will fail in areas that count the most, that is, reducing credit risk.

Second, the choices of models should be informed by the priorities of the institution regarding its operations. In cases where predictive accuracy or precision is of the utmost importance to an organization, Boosted Trees and Bagged Trees methods of ensemble remain the most popular.

But when throughput is of utmost priority as in high volume, automated lending environments, linear SVM has the fastest throughput, and may process hundreds of thousands of applications per second. The models such as Cubic SVM offer a good balance between the performance and the speed and thus are suitable to any institution that desires such aspect.

Third, closer examination should be given to profiles of the applicants that questioned the several models many times, especially those which were related with the sample 12 18. Such data values repeated within different algorithms that were performing reasonably well, meaning that this data could be exhibiting some problems like feature overlap, unconstructed data variables, or inaccurate data labelling of some extra issues. A purer audit of this subset can indicate issues in the data quality that, once rectified, would benefit the ability of the model to generalize on edge cases.

VII. CONCLUSIONS

Loan approval prediction has no single model. Frame of frame of Physical Frame (U) Individual machine learning techniques have a distinct set of strengths and trade-offs. Tree based ensembles have high general accuracy and stability, however poor at detection denied applications, which is typically the most important application in lending. Less sophisticated models can be faster but less accurate and others such as neural networks or SVMs can be slower, needing training and fine-tuning.

Visual analysis, perhaps, contributed the most to the comprehension. Prediction graphs showed us underlying behavioral trends in data, which would have been missed with the correct metrics alone, e.g., it was clear that there were consistent failures in ranges of the sample, which were not visually apparent with the correct metrics alone. Such visual diagnostics became significant to comprehend not only the quality of a model's performance but also its locations and reasons of failure.

After all, the optimal model is a matter of priorities of an institution: accuracy, speed, fairness, or interpretability. What is important though is that these models are not fixed. They should be constantly monitored, retrained, and measured in order to remain efficient in constantly changing real world scenarios. The promise of machine learning could revolutionize one aspect of loan decision-making, and the answer is no: it should be used responsibly, with ambition as well as restraint.

REFERENCES

[1] Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. European Journal of Operational Research, 247(1), 124–136. [https://doi.org/10.1016/j.ejor.2015.05.030]

[2] Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Systems with Applications, 39(3), 3446–3453. [https://doi.org/10.1016/j.eswa.2011.09.033]

[3] Zhou, Z.-H. (2012). Ensemble Methods: Foundations and Algorithms. Chapman and Hall/CRC. (A foundational text explaining bagging, boosting, and ensemble learning strategies used in your study.)

[4] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W.P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321–357. [https://doi.org/10.1613/jair.953]

[5] MATLAB Documentation. (2024). Classification Learner App. The MathWorks, Inc. [https://www.mathworks.com/help/stats/classification-learner-app.html]

[6] Han, J., Pei, J., & Kamber, M. (2011). Data Mining: Concepts and Techniques (3rd ed.). Morgan Kaufmann. (Includes chapters on supervised learning, evaluation metrics, and classification performance.)

[7] Thomas, L. C. (2009). Consumer Credit Models: Pricing, Profit and Portfolios. Oxford University Press. (Covers credit scoring systems and real-world decision strategies for financial institutions.)