Loan defaulter classification system

Mansi Lakhani (40197791), Apekshaba Gohil (40203058), Akshaya Barat Bushan (40220139), Anitha Ramakrishnan (40231724)

Abstract- Accurately assessing the creditworthiness of borrowers is vital for financial institutions to mitigate the risk of defaults. This project presents a loan defaulter prediction system utilizing a Kaggle dataset with over 39,000 records and 111 features. The initial dataset analysis identified three primary challenges: high dimensionality, poor data quality, and significant data imbalance. To address these issues, a comprehensive data preprocessing phase was conducted, encompassing column filtering and selection, handling null values, target variable selection, outlier detection and removal, and relevant feature selection. Additionally, sampling techniques were applied to balance the dataset. Three models were constructed: a supervised decision tree, a semi-supervised decision tree, and deep neural networks. The models were trained on the training data and evaluated using the test data. Extensive evaluations were performed to assess their performance. Hyperparameter tuning and optimization techniques were employed, resulting in remarkable test accuracy exceeding 95% for all three models. This research contributes to the development of a reliable loan defaulter prediction system, empowering financial institutions to make informed decisions and reduce financial risks.

I. INTRODUCTION

When approving loans, financial institutions need to assess the creditworthiness of borrowers to minimize the risk of defaults. The ability to accurately identify potential defaulters can help lenders make informed decisions, manage their loan portfolios effectively, and maintain the stability of their financial operations. The problem of predicting loan defaulters is crucial for many reasons including,

1. Risk Mitigation: Financial institutions can proactively manage and mitigate the risk of financial losses associated with unpaid loans.
2. Responsible Lending: Predicting defaulters allows lenders to ensure responsible lending practices by evaluating borrowers' creditworthiness and offering loans to those who are more likely to repay them.

The selected problem application posed several challenges,

1. High Dimension and Poor Data Quality: The dataset comprises over 39,000 records and 111 features, characterized by numerous missing and null values.
2. Feature Selection: An essential task is identifying relevant features that accurately captured the borrower's creditworthiness, while minimizing the inclusion of unnecessary or redundant variables.
3. Imbalanced Data: The dataset exhibits a significant class imbalance, with the "Fully paid" class representing 85% of the data and the "Charged off" class comprising only 15%.

In the literature, various techniques have been proposed to address these challenges in predicting loan defaulters. These include data preprocessing techniques, feature selection methods, and handling imbalanced data The advantage of these existing solutions include improved data quality, reduced dimensionality, and enhanced representation of minority classes.

While these existing solutions for predicting loan defaulters offer advantages, they also have their drawbacks. Data preprocessing techniques may inadvertently result in information loss. Feature selection methods, while reducing dimensionality, may overlook important features that could affect prediction accuracy. Balancing techniques for imbalanced data, if not applied carefully, can introduce biases or lead to overfitting. It is essential to be mindful of these limitations when implementing these approaches in order to develop a reliable loan defaulter prediction system.

This report aims to overcome these challenges and provide an effective solution to the loan defaulter prediction problem. The methodology begins with a comprehensive data preprocessing phase, null value handling, outlier detection and removal, and feature selection techniques. The dataset is then balanced using appropriate sampling techniques to address the class imbalance issue. Then the data is split into train and test data.

In this study, three models were implemented to predict loan defaulters: a supervised decision tree, a semi-supervised decision tree, and deep neural networks. These models were trained on the carefully preprocessed dataset and underwent extensive evaluation using a range of metrics including accuracy, precision, F1 score, and ROC curve analysis to comprehensively assess their performance. Based on that, hyperparameter tuning and optimization techniques were employed to further enhance the models' performance and robustness.

Promising preliminary results were obtained, with the proposed models achieving test accuracies exceeding 95%. By addressing challenges related to high dimensionality, poor data quality, and imbalanced data, this research aims to provide a reliable loan defaulter prediction system. The outcomes of this study have the potential to empower financial institutions in making informed lending decisions, mitigating risks, and ensuring the overall stability of their lending portfolios.

II. METHODOLOGIES

A. DATASET

The dataset utilized in this project was obtained from Kaggle. The dataset contains information related to loan applications and borrowers' details. It consists of 39,000 samples with 111 features. The features include borrower attributes such as annual income, employment length, loan amount, loan status, credit history, and various financial indicators. Additionally, the dataset provides information on delinquencies, credit utilization, loan descriptions, and other factors that can help predict loan defaulters.

The dataset consists of various types of features. It includes numerical features such as loan amount and interest rate, categorical features like home ownership and loan status, text features such as the title and description of the loan, and date and time features such as the last payment date. These different types of features provide diverse information that can be used to analyze borrowers' creditworthiness and make informed decisions regarding loan approvals and risk management.

The following is the overview of the data preprocessing steps:
1. Column Filtering and Selection: Columns with all NULL values were removed, resulting in the elimination of 54 columns. Additionally, irrelevant columns were dropped, reducing the dataset to 45 relevant features.
2. Handling Null Values: Rows with any missing values were removed from the dataset, resulting in the removal of 1,894 rows.
3. Target Variable Selection: The target variable is the "Loan Status," which initially had three classes: Fully Paid, Charged Off, and Current. The "Current" class was deemed irrelevant for the classification task and was therefore discarded. Consequently, the dataset shape was modified to (36,725, 45).
4. Outlier Detection and Removal: Outliers In the "Annual Income" column were identified using the Quantile method. Outliers exceeding the 95th percentile were removed from the dataset. However, no outliers were removed for columns such as Loan Amount, Interest Rate, and Total Payment, as they exhibited a mostly continuous distribution.
5. Feature Selection: Several features, including “id”, “application_type”, “policy_code”, “initial_list_status”, "installment", and "pymnt_plan" were determined to be unimportant for the goal of the problem and were discarded.
6. Data Splitting: The dataset was divided into training and testing sets, with 13,418 records allocated for training and 6,609 records for testing.
7. Over Sampling and Under Sampling: The initial class distribution of the target variable, Loan Status, showed a significant imbalance, with 85.66% classified as Fully Paid and only 14.34% as Charged Off. To address this, random under sampling was applied to the majority class, and Synthetic Minority Over-sampling Technique for Nominal and Continuous (SMOTENC) was used to augment the training dataset. The resulting class distribution in the training data became more balanced, with Fully Paid accounting for 59.88% and Charged Off for 40.12%.

B. SUPERVISED DECISION TREES

The decision tree architecture plays a crucial role in understanding the model's behavior and interpretability. Here is an overview of the decision tree's architecture and its performance metrics:
1. Depth: The decision tree was constructed with a depth of 3 initially. This choice of depth helps strike a balance between capturing important features and avoiding overfitting.
2. Splitting Criterion: The entropy was employed to determine the best feature to split the data at each decision node. By utilizing entropy, the decision tree identifies the most informative features that maximize the separation of the target classes. The “recoveries” feature was the most informative with the highest entropy which was selected as the root. Then features like “total_rec_prncp” (Principal received to date) and “funded_amont” and “last_credit_pull” were selected as they have better entropy than other features.
3. Random State: The random state hyperparameter of the decision tree classifier was set to its default value of 0. This parameter controls the randomness of the estimator. It ensures that the features are randomly permuted at each split, even when the splitter is set to "best". By fixing the random state to an integer value, a deterministic behavior is achieved during the fitting process. The value 0 is commonly used and believed to yield good performance.
4. Model Training: A supervised decision tree model was trained on a training dataset comprising 71% of the total sampled data, using the parameters mentioned above.
5.Testing and Prediction: The remaining split of test data, which accounts for 29% of the total sampled data.

C. SEMI SUPERVISED DECISION TREES

The semi-supervised learning approach was employed to utilize both labeled and unlabeled data for training the model. The following steps were performed:

1. Division of Training Dataset into Labeled and Unlabeled Subsets:
After preprocessing and balancing, the training dataset with approximately 16k records was divided into labeled and unlabeled subsets. 20% of the preprocessed training data (3324 records) was set aside as the labeled dataset and the remaining 80% (13424 records) of the training data was considered the unlabeled dataset.
2. Training a Decision Tree Model:
The labeled dataset was used to train the decision Tree model. We used our knowledge from the supervised decision tree implementation to get the optimal hyperparameters. We set entropy as the splitting criterion and the depth at 5.
3. Pseudo-Labeling of Unlabeled Data:
The trained Decision Tree model was then applied to the unlabeled dataset to assign pseudo-labels. The class label predictions made by the model for the unlabeled data were considered as pseudo-labels.
4. Confidence Threshold for Pseudo-Labels:
To ensure high-confidence pseudo-labels, only predictions with probability exceeding 90% were considered. These predictions were treated as high-confidence labels.
5. Incorporating Pseudo-Labels into Labeled Data:

The high-confidence pseudo-labels were added to the labeled data, along with their corresponding predicted pseudo-labels. This process was iteratively repeated until all the unlabeled data was assigned pseudo-labels and incorporated into the labeled dataset.

6. Final Model Training:

After incorporating pseudo-labels into the labeled dataset, the entire labeled dataset, including the initially labeled and newly pseudo-labeled instances, were used to train the Decision Tree model again.

7. Test Set Prediction:

Finally, the trained model was applied to the same final test set used for evaluating other models to ensure a fair comparison and assess its performance.

By leveraging the available labeled data along with the pseudo-labeled data generated through the semi-supervised learning process, the aim was to improve the model's performance by effectively utilizing the information contained in the unlabeled dataset.

D. Deep Neural Networks

The selected DNN model has a sequential architecture, meaning that the layers are stacked sequentially on top of each other. The components of the DNN architecture are described below:

1. Input Layer: The input layer has 78 units/neurons, and the activation function used is ReLU (Rectified Linear Unit). Additionally, a dropout layer is applied with a dropout rate of 0.2 which sets a fraction of input units to 0 during training, which helps prevent overfitting.

2. Hidden Layers: The hidden layer has 39 units/neurons, and again, the ReLU activation function is used. Dropout with a rate of 0.2 is applied. Another hidden layer is added with 19 units/neurons, ReLU activation and 0.2 dropout.

3. Output Layer: The output layer consists of a single neuron, as this is a binary classification problem (loan default or not). The activation function used is sigmoid, which squashes the output between 0 and 1, representing the probability of loan default.

The model is compiled using the Adam optimizer for optimization, which has gained prominence due to its capacity to efficiently converge on a diverse range of problems. The loss function chosen is binary_crossentropy, suitable for binary classification tasks, and the metric used for evaluation is Testing accuracy, F1-score, precision, and recall.

The selected DNN model is suitable for the practice of loan default classification as the model consists of multiple hidden layers, allowing it to capture complex relationships and patterns in the data. The use of the ReLU activation function in the hidden layers helps introduce non-linearity, enabling the model to learn more complex decision boundaries. The addition of dropout layers helps prevent overfitting by randomly dropping out a fraction of the neurons during training enabling network to learn more robust and generalized representations. The sigmoid activation function in the output layer is appropriate for binary classification, providing a probability-like output that can be interpreted as the likelihood of loan default. The number of neurons in each layer has been chosen based on the complexity of the problem and the size of the dataset. The gradual reduction in the number of neurons in the hidden layers (78, 39, 19) allows the model to progressively learn higher-level features while keeping the computational complexity manageable. Total 641,433 trainable parameters were trained in the deep neural network architecture.

Early stopping method was used while training monitoring validation loss and the model stopped training after 31 epochs with a total computational complexity of 58.034 sec for both training and validation phases.

E. Optimization algorithm

The choice of an optimization algorithm for a specific problem depends on the unique characteristics of the data involved. In our project, we employed the grid search technique to optimize the hyperparameters, including experimenting with different learning rates (0.001, 0.01, and 0.1) and batch sizes (32, 64, and 256). By training the Deep Neural Network (DNN) on the dataset using these combinations, our goal was to identify the best set of hyperparameters that would yield optimal performance.

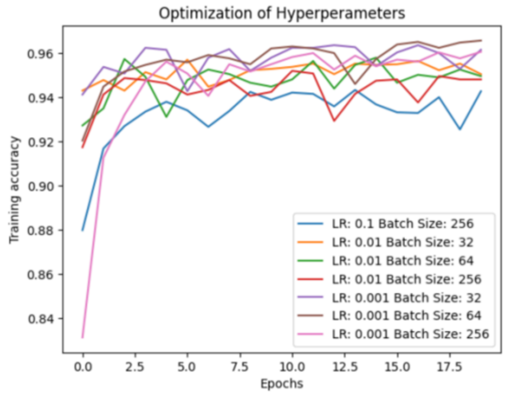

Figure A: Optimization of Hyperparameters

Figure A shows the experiment of learning rate accuracy with 20 epochs on different learning rate and batch sizes. The best hyperparameters were batch size of 64 and 0.001 Learning rate with 96% testing accuracy with a fixed 20 epoch.

To evaluate the performance of the proposed method, we conducted a comparative analysis of different models using various metrics such as F1 score, Testing accuracy, Precision, and Recall.

In our research study, we employed the Adam Optimizer for optimization, which has become widely recognized for its effectiveness in converging efficiently on various problem types. One advantageous aspect of utilizing the Adam optimizer is its capability to handle gradients that are either noisy or sparse. By mitigating the influence of such gradients, Adam enables the model to converge more effectively, leading to improved optimization outcomes.

Regularization method like dropout was added to help prevent overfitting and improve the generalization of your model. Early stopping was used while training the model to automatically determine the optimal stopping point during model training monitoring the test accuracy.

III. Results

A. Supervised Decision trees

In this section, we present a summary of the performance of the semi-supervised decision tree model on the testing data. The evaluation encompasses various metrics including the confusion matrix, precision, recall, accuracy, F1 score, and ROC curve. The confusion matrix provides an overview of the model's predictions by categorizing them into true positive (TP), true negative (TN), false positive (FP), and false negative (FN) outcomes. These metrics collectively offer insights into the effectiveness and reliability of the model's performance.

	Before Sampling		After Sampling	
Classes/ Evaluation metric	Fully Paid	Charged off	Fully Paid	Charged off
Precision	94 %	92 %	97 %	100 %
Recall	99 %	69 %	100 %	89 %
F1 score	96 %	79 %	98 %	94 %
Accuracy	94 %		97 %	

Table1. Performance matrix of test data for Decision tree

The supervised decision tree model outperformed the control in predicting loan repayment status, achieving a high accuracy of 0.97 on testing data. It accurately identified loan defaulters and non-defaulters, with precision scores of 1.00 and 0.97 for "Charged Off" and "Fully Paid" classes, respectively. The model's reliability was further reinforced with recall scores of 0.89 and 1.00 for "Charged Off" and "Fully Paid" classes.

While the results demonstrate the Accuracy of 0.97 of the supervised decision tree models, there is room for further improvement to enhance the reliability of its predictions.

Additional experiments were conducted to address this objective.

Hyperparameter Tuning: focusing on the "max_depth" parameter, significantly impacts supervised decision tree model performance. Finding the optimal value can lead to near-perfect accuracy in identifying loan defaulters, enhancing system reliability. Fine-tuning the hyperparameter results in nearly 99% accuracy, indicating the model's high reliability and accurate predictions on prospective defaulters.

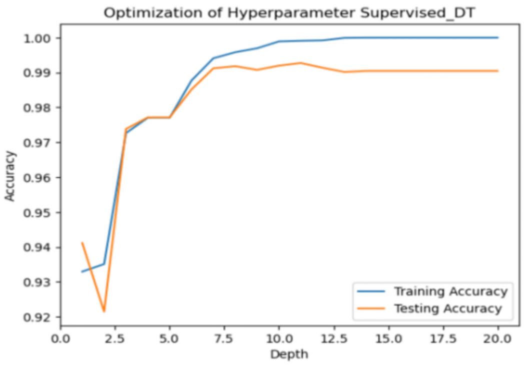

Figure 1: Optimization of Hyperparameter for Supervised Decision tree

At the Depth of 11 the model shows the highest accuracy of 99% as we can see in the above plot of optimization of hyperparameter.

After the setting max_depth to 11, splitting criterion as entropy, and random splitter to 0 the improved results are as follow.

	Before Tuning		After Tuning	
Classes/ Evaluation metric	Fully Paid	Charged off	Fully Paid	Charged off
Precision	97 %	100 %	99 %	99 %
Recall	100 %	89 %	100 %	98 %
F1 score	98 %	94 %	100 %	99 %
Accuracy	97 %		99 %	

Table2. Performance matrix of test data

The model demonstrated a high capacity to distinguish between loans that had been defaulted and loans that had been fully repaid. These findings imply that the Supervised decision tree model can aid financial institutions in loan risk assessment.

B. Semi Supervised Decision trees

The performance of the semi-supervised decision tree model on the testing data is summarized in this section. The evaluation includes the confusion matrix, Precision, recall, accuracy, F1 score and ROC curve. The confusion matrix showcases the model's predictions in terms of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) outcomes.

Test output / Predicted class	Real class	
	"Charged off"	"Fully Paid"
"Charged off"	1554	79
"Fully Paid"	10	4966

Table3. Performance matrix of test data for Semi-Supervised technique

The results demonstrate that the semi-supervised decision tree model achieved a high accuracy of 0.98 on the testing data, indicating its effectiveness in predicting loan repayment status. With precision scores of 0.99 for both classes and recall scores of 0.96 and 1.00 for "Charged Off" and "Fully Paid" classes respectively, the model showcased a strong ability to accurately identify loans that defaulted and loans that were fully repaid. These findings suggest that the semi-supervised decision tree model holds promise as a valuable tool for financial institutions in assessing loan risks and making informed lending decisions.

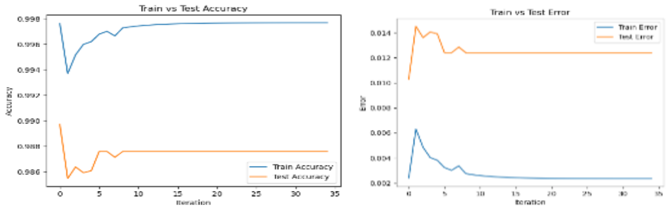

Figure 2a: Train Vs Test Accuracy, Figure2b: Test vs Train Error

In Figure 2a, the accuracy is plotted against the number of iterations, representing the impact of incorporating a portion of unlabeled data with pseudo labels into the labeled data. The Figure 2b gives an idea about error on training and testing data over the same iterations. The trend in the figure suggests that as more iterations are performed and unlabeled data with pseudo labels are added to the labeled data, the accuracy tends to increase. This implies that leveraging unlabeled data can be beneficial in improving the model's performance. The gap between the test and train curves is perceptible, although the disparity is minimal, amounting to mere fractions of a unit (0.001).

C. Deep Neural Networks

The performance of the deep neural network model on the testing data is summarized in this section. When training deep neural networks, it is common to monitor and compare the performance of the model on both the training and test datasets. The assessment consists of various metrics such as the confusion matrix, Precision, recall, accuracy, F1 score, and ROC curve. The confusion matrix displays the model's predictions by categorizing them into true positive (TP), true negative (TN), false positive (FP), and false negative (FN) results.

Test output / Predicted class	Real class	
	"Charged off"	"Fully Paid"
"Charged off"	1440	62
"Fully Paid"	29	5004

Table4. Performance matrix of test data for DNN

The results depicted a 96% accuracy and recall on test dataset and 94% recall for "Charged off" and 98% for "Fully paid". It evaluated F1 score of 95% for "Charged off" and 97% for "Fully paid". Comparing the training and test accuracies helps to assess whether the model is overfitting or underfitting. The training accuracy of the model is 99% and validation accuracy is 96%. Ideally, both accuracies should be close and relatively high. If the training accuracy is significantly higher than the test accuracy, it suggests overfitting, where the model is memorizing the training data and not generalizing well to new data.

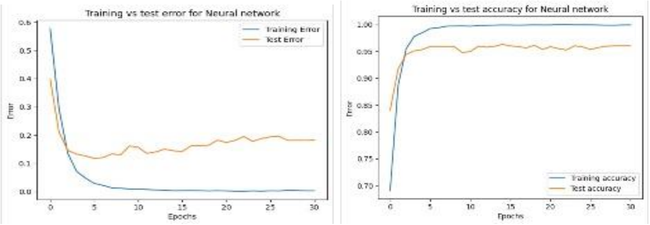

Figure 3a: Train Vs Test Error, Figure3b: Test vs Train Accuracy

In Figure 3b, the test accuracy is plotted against the number of epochs. In 3a, the test error is plotted. Overall, when comparing the metrics with Decision tree models we can conclude that we achieve higher accuracies and results using decision tree models as we have many categorical features. As categorical data can be effectively handled by Decision trees and require less data preprocessing compared to neural networks.

D. Ablative study

In our study, we conducted hyperparameter tuning for semi-supervised decision trees by adjusting the tree depth within the range of 3 to 10. Notably, at a depth of 3, the model achieved an accuracy of 97%, which notably improved to 98% when the depth was increased to 5. Furthermore, augmenting the depth to 7 yielded an even higher accuracy of 99%. However, when we pushed the depth to 10, the model exhibited clear signs of overfitting. Thus, based on our findings, we conclude that the optimal depth range for the decision trees lies between 5 and 7. Moreover, the min_samples_leaf parameter effectively complemented the depth parameter, particularly when set to 3.

Evaluation metric/ Model	Supervised Decision trees		Semi - Supervised Decision trees	Deep Neural Networks	
Accuracy	99%		99%	96%	
Precision	99%		99%	96%	
Recall	98%		98%	98%	
F1 Score	99%		98%	97%	
Confusion matrix	1606	27	1566 67	1440	62
	21	4955	7 4969	29	5004

Table 1: Evaluation metrics for all the models on test dataset.

We also experimented hyperparameter tuning for Deep neural network using grid search technique for hyperparameters like learning rate and batch size. We performed a comparative analysis of multiple models, employing a range of metrics like F1 score, testing accuracy, precision, and recall.

The provided table presents a comprehensive comparison of different models, revealing that decision trees consistently outperform other models on the test data. With an impressive performance, decision trees consistently demonstrate superior accuracy when compared to alternative models. This significant performance advantage further solidifies the suitability and efficacy of decision trees for the given dataset, making them the top-performing model choice for predictive tasks.

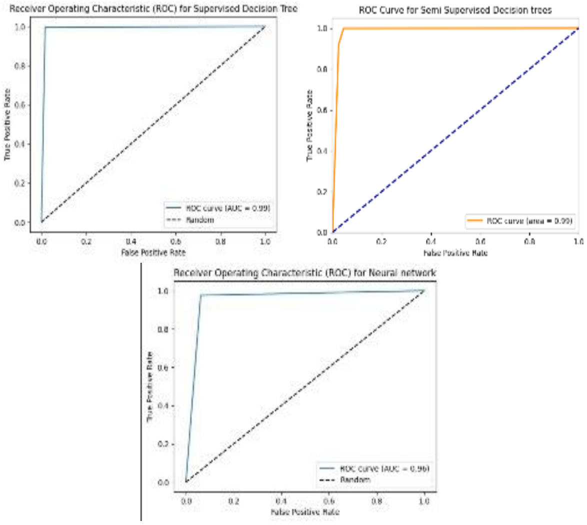

Figure 4: ROC Curves for all three models

The ROCs curvers of all three models are added here for comparison.The TSNE Visualization for all three models are added in the below figure.

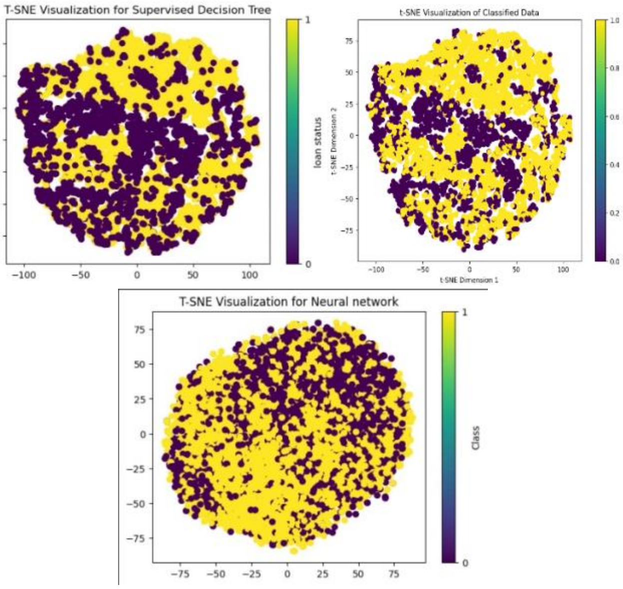

Figure 5: t-NSE data visualization for all three models

In our experiments, it became evident that both supervised and semi-supervised decision trees outperformed neural networks in terms of accuracy on the test data. The decision trees displayed an exceptional accuracy rate of 99%, while the neural networks fell slightly behind at 96%. This substantial discrepancy in performance can be attributed to several factors related to the dataset itself. Firstly, the dataset primarily consists of categorical features organized in a tabular format. When we encoded this data for the neural networks, some valuable information was inevitably lost, thus impacting their predictive capability. Additionally, the dataset's simplicity, despite its high dimensionality, rendered the intricate structure of neural networks overly complex for the task at hand. Even with the incorporation of regularization techniques, the neural networks failed to match the remarkable accuracy achieved by the decision trees. These findings highlight the importance of considering the nature of the dataset and the complexity of the modeling approach when selecting the appropriate algorithm for a given task.

Furthermore, the superiority of decision trees over neural networks can also be attributed to their inherent interpretability and simplicity. Decision trees provide explicit rules and decision paths, allowing for easier understanding and explanation of the underlying logic driving the predictions.

Moreover, decision trees tend to handle imbalanced class distributions more effectively compared to neural networks. Decision trees have mechanisms such as sampling that can mitigate the impact of class imbalance, ensuring more balanced predictions. While neural networks have gained popularity for their ability to learn complex patterns and generalize well to diverse data, the specific characteristics of the dataset used in our experiments favored the decision tree models.

IV. Reference

[1] Dataset: “Loan Classification dataset” from Kaggle. https://www.kaggle.com/datasets/abhishek14398/loan-dataset

[2] Gupta, Kanishk and Chakrabarti, Binayak and Ansari, Aseer Ahmad and Rautaray, Siddharth Swarup and Pandey, Manjusha, Loanification - Loan Approval Classification using Machine Learning Algorithms (April 24, 2021). Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2021

[3] Rokach, L., Maimon, O. (2005). Decision Trees. In: Maimon, O., Rokach, L. (eds) Data Mining and Knowledge Discovery Handbook. Springer, Boston, MA.

[4] Mehul Madaan *et al* 2021 *IOP Conf. Ser.: Mater. Sci. Eng.* **1022** 012042

[5] Decision tree classifier: https://scikitlearn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[6] Neural network architecture: https://keras.io/api/layers/

[7] Feature importance: https://machinelearningmastery.com/calculate-feature-importance-with-python/

[8] I O Eweoya *et al* 2019 *J. Phys.: Conf. Ser.* **1299** 012037

[9] A. Goyal and R. Kaur, “Accuracy Prediction for Loan Risk Using Machine Learning Models”.

[10] Hsieh, N. C., & Hung, L. P. (2010). A data driven ensemble classifier for credit scoring analysis. Expert systems with Applications, 37(1), 534-545.

[11] Aslam U, Aziz H I T, Sohail A and Batcha N K 2019 An empirical study on loan default prediction models *Journal of Computational and Theoretical Nanoscience* **16** 3483-8

[12] Li Y 2019 Credit risk prediction based on machine learning methods *The 14th Int. Conf. on Computer Science & Education (ICCSE)* 1011-3

[13] Ahmed M S I and Rajaleximi P R 2019 An empirical study on credit scoring and credit scorecard for financial institutions *Int. Journal of Advanced Research in Computer Engineering & Technol. (IJARCET)* **8** 275-9

[14] Zhu L, Qiu D, Ergu D, Ying C and Liu K 2019 A study on predicting loan default based on the random forest algorithm *The 7th Int. Conf. on Information Technol. and Quantitative Management (ITQM)* **162** 503-13

[15] Ghatasheh N 2014 Business analytics using random forest trees for credit risk prediction: a comparison study *Int. Journal of Advanced Science and Technol.* **72** 19-30

[16] Madane N and Nanda S 2019 Loan prediction analysis using decision tree *Journal of The Gujarat Research Society* **21** 214-21

[17] Amin R K, Indwiarti and Sibaroni Y 2015 Implementation of decision tree using C4.5 algorithm in decision making of loan application by debtor (case study: bank pasar of yogyakarta special region) *The 3rd Int. Conf. on Information and Communication Technol. (ICoICT)* 75-80

[18] Jency X F, Sumathi V P and Sri J S 2018 An exploratory data analysis for loan prediction based on nature of the clients *Int. Journal of Recent Technol. and Engineering (IJRTE)* **7** 176-9

[19] Alshouiliy K, Alghamdi A and Agrawal D P 2020 AzureML based analysis and prediction loan borrowers creditworthy *The 3rd Int. Conf. on Information and Computer Technologies (ICICT)* **1** 302-6

[20] Addo P M, Guegan D and Hassani B 2018 Credit risk analysis using machine and deep learning models Risks 6 p 38.

[21] Kim, Aleum, and Sung-Bae Cho. "An ensemble semi-supervised learning method for predicting defaults in social lending." *Engineering applications of Artificial intelligence* 81 (2019): 193-199.

[22] Li, Meixuan, Chun Yan, and Wei Liu. "The network loan risk prediction model based on Convolutional neural network and Stacking fusion model." *Applied Soft Computing* 113 (2021): 107961.

[23] Quinlan, J.R. Induction of decision trees. Mach Learn 1, 81–106 (1986). https://doi.org/10.1007/BF00116251

[24] Lin, C., Qiao, N., Zhang, W. *et al.* Default risk prediction and feature extraction using a penalized deep neural network. *Stat Comput* 32, 76 (2022). https://doi.org/10.1007/s11222-022-10140-z