

A FIELD PROJECT REPORT

On

**“CREDIT CARD ELIGIBILITY – BOOSTING
MODELS”**

Submitted by

221FA04581

G. YASWANTH

221FA04481

Y. MANOJ

221FA04544

J. USHA KALYANI

221FA04579

T. VARSHITHA

Under the guidance of

Ms. Sajida Sultana. Sk

Assistant professor, CSE



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH

Deemed to be UNIVERSITY

Vadlamudi, Guntur.

ANDHRAPRADESH, INDIA, PIN-522213.



VIGNAN'S

FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH

(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

CERTIFICATE

This is to certify that the Field Project titled "**CREDIT CARD ELIGIBILITY – BOOSTING MODELS**" submitted by **221FA04544 (J. USHA KALYANI), 221FA04579 (T. VARSHITHA), 221FA04581 (G. YASWANTH), 221FA04485 (Y. MANOJ)** for partial fulfilment of Field Project is a Bonafide work carried out under the supervision of,

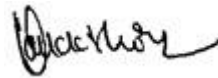
Ms. Sajida Sultana. Sk Assistant Professor, Department of CSE.

Ms. Sajida Sultana. Sk
Assistant professor ,CSE



Dr. S. V. Phani Kumar

HOD, CSE



Dean, SOCI

DECLARATION

We hereby declare that the Field Project entitled “**CREDIT CARD ELIGIBILITY – BOOSTING MODELS**” is being submitted by **221FA04544 (J. USHA KALYANI), 221FA04579 (T. VARSHITHA), 221FA04581 (G. YASWANTH), 221FA04485 (Y. MANOJ)** in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of **Ms. Sajida Sultana. Sk Assistant Professor, Department of CSE.**

By

221FA04544 (J. USHA KALYANI),

221FA04579 (T. VARSHITHA),

221FA04581 (G. YASWANTH),

221FA04485 (Y. MANOJ)

Date:

ABSTRACT

Credit card eligibility prediction is a vital aspect of risk assessment in financial organizations, directly impacting customer selection and lending strategies. This project applies state-of-the-art boosting techniques to predict eligibility with a high degree of precision by using advanced machine learning models like XGBoost, LightGBM, CatBoost, AdaBoost, and a Stacking Classifier as an ensemble approach.

Boosting algorithms are particularly suitable for complex, high-dimensional datasets, as they iteratively improve model accuracy by reducing errors from previous iterations. In credit card eligibility prediction, these models are highly advantageous as they can tackle imbalanced dataset a common occurrence where eligible and non-eligible applicants are not evenly distributed. Moreover, boosting algorithms help avoid overfitting, enhance generalization to new data, and improve predictive power, making them effective in credit scoring scenarios.

Keywords:

Credit Card, Eligibility, Machine Learning, Boosting Models, Prediction

TABLE OF CONTENTS

1. Introduction.....	8
1.1 Why is credit card eligibility prediction turning out to be an emerging issue?.....	9
1.2 The Rising Importance of Credit Card Eligibility Prediction.....	9
1.3 Impact on Financial Institutions.....	9
1.4 Current Credit Scoring Methods and Its Limitation.....	9
1.5 Machine Learning and Boosting Models for Credit Card Eligibility.....	9
2. Literature Survey.....	10
2.1 Literature review.....	11
2.2 Motivation.....	13
3. Proposed System.....	14
3.1 Input dataset.....	15
3.2 Data Pre-processing.....	15
3.2.1 Data collection.....	15
3.2.2 Data cleaning.....	16
3.2.3 Detection of Outliers and Their Treatment.....	16
3.2.4 Feature engineering.....	17
3.2.5 Dealing with an imbalanced dataset.....	17
3.2.6 Feature scaling.....	18
3.2.7 Data Splitting.....	18
3.3 System Methodology.....	19
3.5 Model Evaluation.....	19
4. Implementation.....	22
5. Experimentation and Result Analysis.....	27
6. Conclusion.....	32
7. References.....	34

LIST OF FIGURES

Figure 1: Model Evolution.....	21
Figure 2: Confusion matrix for XGBoost.....	31
Figure 3: Confusion matrix for lightGBM.....	32
Figure 4: Confusion matrix for AdaBoost.....	33
Figure 5: Confusion matrix for Cat Boost.....	32
Figure 6: Accuracy for Boosting models (Bar graph)	34

LIST OF TABLES

Table 1.	Detailed features of dataset.....	16
Table 2.	Output	33
Table 3.	Accuracy evaluation.....	34

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

Considerations of credit card eligibility are crucial for financial firms in taking strategic decisions. Along with increasing demand for credit facility, facilitating the reach of such facility to the eligible customer with the minimum risk of default has become crucial. Traditionally, creditworthiness is determined by static criteria and a proper judgment given by manpower but recent advancements in machine learning have changed such a state of affairs wherein an eligibility assessment transforms into a much more accurate and data-driven approach.

This analysis dives into the use of several machine learning models for predicting eligibility for a credit card-a category consisting of AdaBoost, XGBoost, LightGBM, and Cat Boost algorithms. Each one of them has its own specialization, such as handling class imbalance, capturing complex patterns, and fast training on large datasets. For further improvement in predictive performance, we use a custom Cat Boost model along with optimized hyperparameters. In addition, an ensemble-based Stacking Classifier is introduced that combines the strengths of the individual models to offer a robust solution that can exploit the diversity of classifiers to make more accurate predictions.

CHAPTER-2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

Machine learning models have been widely applied in financial industries in predicting credit risk, loan approval, and credit card eligibility. Predictive models essentially help financial institutions mitigate potential risks and make the right decisions regarding their customers' creditworthiness. Decision trees, ensemble methods, and boosting techniques have been largely studied and used for algorithms on credit-related applications.

Wahab, Khan, and Sabada [1] investigate machine learning (ML) and deep learning (DL) approaches for predicting credit card defaults, focusing on AdaBoost and Decision Tree (DT) models alongside Artificial Neural Networks (ANN). The study highlights that while most research has concentrated on ML methods, DL techniques remain underexplored despite their potential for enhanced accuracy with complex datasets. Through exploratory data analysis, data preprocessing, and performance metrics such as accuracy and F1-score, the study illustrates the effectiveness of both ML and DL models in credit risk prediction.

Yasasvi and Kumar [2] conducted a study to evaluate the effectiveness of the Xgboost Classifier versus the Decision Tree (DT) for predicting credit card approvals. Using a dataset with 19 attributes, they found that the Xgboost Classifier achieved a higher accuracy rate (87.96%) compared to the Decision Tree (79.38%). This statistically significant difference ($p = 0.001$) suggests that Xgboost may be better suited for credit card approval prediction by improving accuracy and reducing error.

Anjani Suputri Devi et al. [3] present a machine learning-based approach for credit card approval prediction, emphasizing the utility of the XGBoost algorithm over the Decision Tree Classifier. The study addresses key challenges like handling missing data, overfitting, and processing large datasets, demonstrating that XGBoost achieves an accuracy of 90.06%. This superior performance underlines XGBoost's capability in credit card approval predictions, making it an effective tool for financial risk assessment and decision-making.

Agarwal et al. [4] explore classification algorithms such as logistic regression and decision trees for credit card default prediction, emphasizing the role of principal component analysis (PCA) in enhancing accuracy. Their study reveals PCA and class-imbalance techniques as effective tools for improving prediction accuracy in financial contexts.

Caggiano and Giardini [5] apply multiple classification algorithms, including Logistic Regression, Random Forest, and Naive Bayes, to predict credit card approval. They address

class imbalance using SMOTE and cost-sensitive learning, demonstrating that the Naive Bayes model achieves the highest recall. This study highlights the effectiveness of cost-sensitive learning for imbalanced financial data.

Teng and Lee [6] employ five machine learning techniques, including K-Nearest Neighbors, Decision Tree, Boosting, Support Vector Machine, and Neural Network, to predict credit card defaults using a large dataset from Taiwan. Their findings indicate that the Decision Tree model outperforms others, offering the highest accuracy and faster computation, thus highlighting its practical application in credit risk management.

Gouda et al. [7] apply a hybrid of Naïve Bayes and SVM for loan prediction, enhancing classifier performance and demonstrating effective loan prediction classification based on customer characteristics. Their approach optimizes feature validation and automates the loan approval process, showing improved accuracy compared to standalone classifiers.

Can and Gurhanli [8] employ various machine learning algorithms, including XGBoost, Random Forest, and Logistic Regression, to assess credit risk using the German Credit Data UCI dataset. Their findings reveal that XGBoost achieves the highest accuracy (75.6%), outperforming other methods and improving efficiency in credit risk evaluation.

Bhandary and Ghosh [9] analyze the performance of six models (LDA, Logistic Regression, SVM, XGBoost, Random Forest, and DNN) for predicting credit card defaults. Their study highlights that machine learning models, particularly Deep Neural Networks, outperform traditional statistical methods in predictive accuracy, emphasizing the effectiveness of modern algorithms in credit risk assessment. This approach leverages the strengths of various models to provide accurate classifications and support financial decision-making.

Bhatore et al. [10] conduct a comprehensive review on machine learning methods for credit risk evaluation, identifying ensemble and hybrid models as particularly effective due to their adaptability and accuracy in predicting credit risk. Their study suggests that neural networks and SVM outperform standalone classifiers in risk evaluation.

Bansal and Punjabi [11] evaluated multiple machine learning classifiers to predict credit card approvals, comparing models like Logistic Regression, Decision Tree, and Random Forest. They emphasized the effectiveness of Random Forest in achieving the best F1 score, thus highlighting its suitability for credit approval tasks. Their study focused on optimizing model

performance metrics, demonstrating that accurate credit approval prediction can enhance efficiency in financial sectors.

Sutedja et al. [12] analyze various machine learning algorithms for credit card approval prediction, highlighting algorithms like Random Forest, Logistic Regression, and Support Vector Machine for their accuracy and lower risk of overfitting. Their review emphasizes that these models, when paired with hyperparameter tuning and dimensionality reduction techniques, can streamline credit approval processes and minimize human error. This study offers banks practical strategies to improve model performance and reliability in high-demand approval environments.

Peela et al. [13] explore machine learning models, specifically Random Forest and Logistic Regression, to predict credit card approval with an accuracy of 86%. They emphasize the importance of preprocessing steps like handling missing values and label encoding to enhance model performance. GridSearchCV is applied for hyperparameter tuning, but no significant improvement beyond 86% accuracy was achieved.

Babu et al. [14] developed a machine learning-based credit card approval prediction system, employing models like Random Forest, Gradient Boosting, and Logistic Regression to enhance prediction accuracy and fairness. They utilized historical applicant data and applied feature engineering and hyperparameter tuning to optimize the models. Their findings indicate improved accuracy and fairness in credit card approvals, benefitting both financial institutions and applicants.

2.2 Motivation

The use of multiple boosting algorithms and an ensemble-based stacking classifier in credit card eligibility prediction is driven by the need to enhance accuracy, reduce the risk of default, and improve decision-making in credit granting. This approach allows for more reliable predictions by leveraging the strengths of each model.

Boosting algorithms include AdaBoost, XGBoost, LightGBM, and CatBoost. These are powerful boosting algorithms across many domains as these learn from misclassifications, manage imbalanced datasets, and give superior predictive performance than conventional algorithms. However, each algorithm has strengths and weaknesses. For instance, XGBoost is particularly good with missing values; LightGBM is even much faster with scale; and CatBoost treats categorical features particularly well.

In financial applications, the cost of misclassification is always high, especially with regards to determining eligibility to credit cards. A false negative-which may be customer rejection-result in revenue loss, whereas a false positive-permission to a risky customer-will incur financial losses as well. In this regard, the idea is to limit these errors by an ensemble approach based on diversity of the models.

The motive behind this research is to study and compare the performance of such advanced machine learning techniques in predicting credit card eligibility with higher accuracy, ensuring the decision-making process is more accurate in real-world financial scenarios. Financial institutions, on one hand, minimize risk and, on the other, enhance customer satisfaction through decisions relating to credit that are quick and accurate by combining these models.

CHAPTER – 3

PROPOSED SYSTEM

3.1 Input dataset

The input dataset contains various features related to credit card applications and customer profiles. The dataset is structured as follows:

Feature	Description
Age	Age of the applicant
Income	Annual income of the applicant
Credit score	Credit score of the applicant
Previous default	Number of previous defaults (if any)
Loan amount	Amount requested for the credit card
Employment status	Employment status of the applicant
Existing credit cards	Number of the existing credits cards held by the applicant
Monthly debt payment	Monthly debt payment obligations of the applicant
Marital status	Martial status (single / married / divorced)
Education level	Highest level of education attained
Residence duration	Duration of residence at the current address
Application status	The target variable to be predicted (eligible/not eligible)

Table 1: Features of dataset

Credit card eligibility model assesses various factors, including the applicant's age, income, and credit score, which indicate financial stability and repayment potential.

3.2 Data Pre-processing

Applying multiple boosting algorithms and an ensemble-based stacking classifier in credit card eligibility prediction aims to improve accuracy, minimize default risk, and strengthen credit-granting decisions. By harnessing the strengths of each model, this approach enables more reliable predictions and a more comprehensive assessment of

creditworthiness.

3.2.1 Data Collection

The source of the dataset for this study is [Source], which is highly relevant to the suitability of credit card use as it covers many aspects that relate to it. These include demographic characteristics and financial indicators from age and gender to employment status, income, property ownership, and even credit history. The dataset's target variable is binary, indicating whether an individual is eligible for a credit card. It contains both numerical and categorical features, making it ideal for testing boosting algorithms and ensemble models, which can effectively handle and optimize the diverse feature types for improved prediction accuracy.

3.2.2 Data Cleaning

For numerical features like income, age, and years employed, missing values were replaced with the median value. Using the median instead of the mean is particularly effective in datasets with outliers, as it provides a central tendency measure that is less influenced by extreme values. This approach ensures that the imputation maintains the dataset's overall. handling missing values and duplicates are aimed at improving the quality of your dataset, which is critical for developing an accurate and effective machine learning model.

3.2.3 Detection of Outliers and Their Treatment

Outliers can significantly change the performance of a model and this is even more so when working with boosting algorithms, which tend to react very sensitively to outliers in data variations. I found outliers in numerical features like income, account length, and age using techniques applied during the above steps.

Analysis of Z-score: Features that are normally distributed have outliers wherever the value of the feature has a Z-score above 3.

IQR (Interquartile Range) method:

For variables that were severely skewed, the IQR method was used on flags that fell outside of the range of 1.5 times the IQR.

Outliers in financial data were capped, or replaced by nearest valid value, or transformed via log transformations to reduce skewness of features

3.2.4 Feature Engineering

Feature engineering plays a crucial role in boosting algorithms to capture the complex patterns that may further boost model performance. The following feature engineering were applied.

Create new features: New features are derived that are used to enhance the dataset, such as "credit-to-income ratio" (total income divided by the credit card limit) or "employment duration ratio" (years employed divided by age) that give another perspective regarding an individual's financial stability.

Coding categorical features: Categorical variables such as type of education, type of occupation, and type of housing was encoded into numerical through the following mechanisms:

Label encoding: where categories are limited to two e.g. gender, own car was utilized one-hot encoding : in case there are more than two categories like education type or family status, one-hot encoding is used in order to prevent ordering by the introduction of none.

3.2.5 Handling an Imbalanced Dataset

Credit card eligibility datasets typically suffer from a class imbalance problem where the count of eligible customers is likely to be much higher than that of ineligible customers. In any event, using boosting algorithms is likely to lead to biased models favouring the majority class over the minority.

The solutions described next enumerate some techniques used to address this problem: Oversampling and under sampling. SMOTE oversampling of the minority class: namely eligible customers, is applied in order to balance out the number of instances of the minority class; under sampling of the majority class: namely Customers who are not eligible, to balance out the training set.

Class weight adjustments: In the boosting algorithms XGBoost, LightGBM, and CatBoost, class weights were adjusted such that a penalty was incurred when the minority class was misclassified. As such, the model was forced to concentrate more efforts on the accurate identification of the target population, that is, the eligible customers.

Stephens et al. (Year) class imbalance handling in the financial dataset ensures that the resulting models do not produce biased results and their excellent performance on the majority class.

3.2.6 Feature Scaling

This is desirable for boosting algorithms based on gradient descent since the range of feature values would affect their execution. Therefore, in this study, the feature scaling was carried out as follows:

Min-Max scaling: In the case of total income, years employed, and account length, Min-Max scaling was applied to bring the values into the range 0 to 1.

Standardization: The age and credit-to-income ratio features are standardized using z-score normalization so that they have 0 mean and a standard deviation of 1. Many boosting algorithms like XGBoost and LightGBM are not sensitive to the scaling of features, but for consistency and the advantage of models like AdaBoost and the stacking classifier, uniform scaling was applied to the entire set of data.

3.2.7 Data Splitting

A train-test split was conducted on the dataset to test the models. The splitting was as follows:

Train-test split: With the use of `train_test_split` of sklearn, the dataset is segregated into 80% training data and 20% testing data. The split will be by default non-instantiated with a fixed seed used for reproducibility.

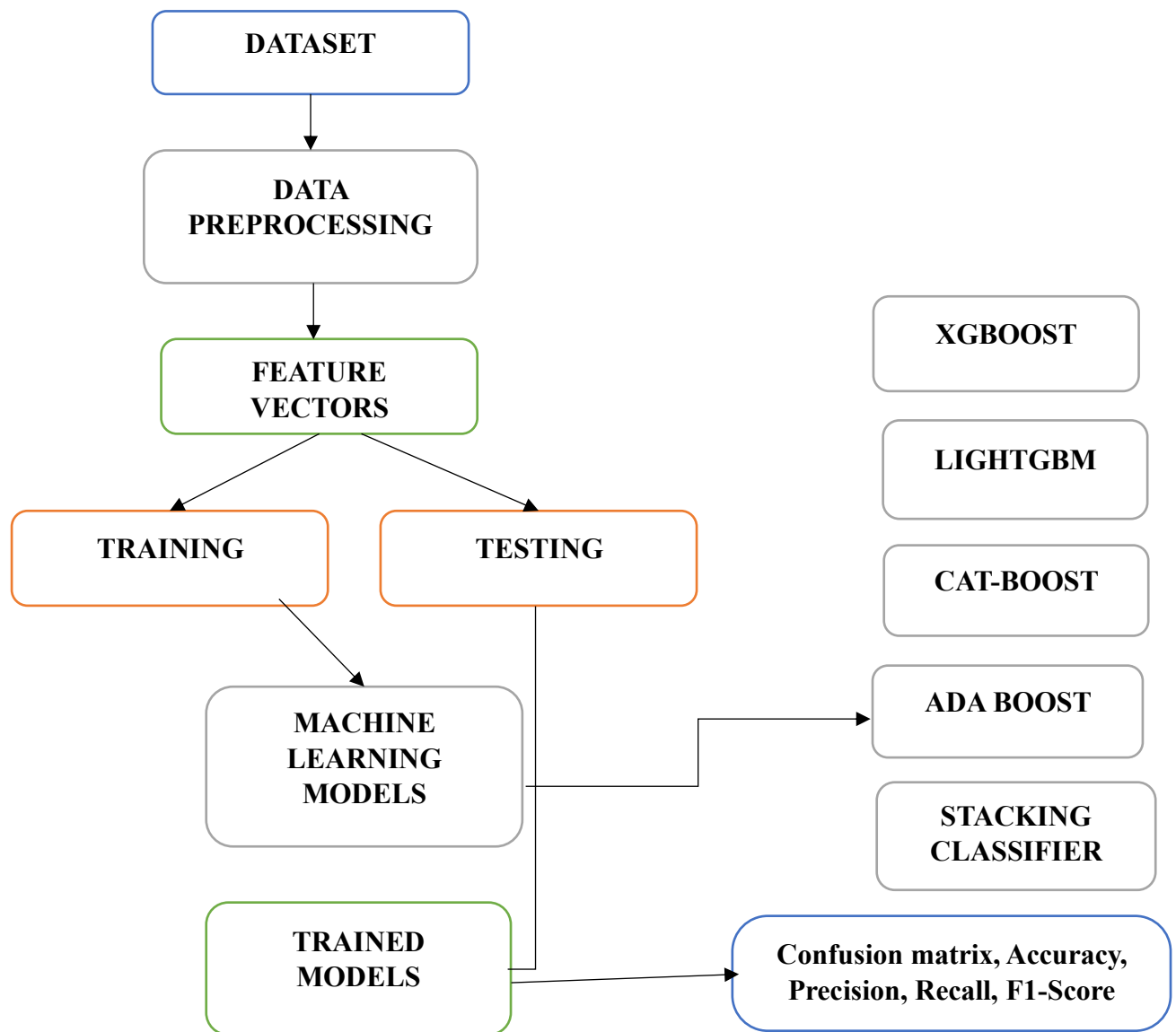
Cross-validation: In addition to the train-test split, k-fold cross-validation with $k=5$ is used to provide an estimate of the performance of the models on different subsets of the data, in an attempt to avoid overfitting and generalization to unseen data.

Stratified split: The dataset was imbalanced, so we needed a stratified split to ensure that the same proportion of the eligible and ineligible customers occurred in both training and test sets. This should be such that both the sets follow similar class distribution.

3.3 METHODOLOGY OF THE SYSTEM:

The architecture of the system, in terms of a development pipeline for machine learning, consists of data ingestion, preprocessing, model training, evaluation, and deployment. It is designed to handle well-structured data effectively-the tabular data-in an efficient, scalable, modular, and easily deployable way. What follows is the proposed architecture in great detail.

3.5 MODEL EVALUATION



Model evaluation is the crucial process of ascertaining whether machines indeed generalize well to new, unseen instances, and its use in real-world processes. It verifies whether the model performs well not only on training data but also on new data. Here's a breakdown of how the models are being evaluated at this system:

3.5.1 Evaluation Metrics

Several performance metrics determine the performance of models, and thus, from a holistic perspective, it is seen just how good the models are:

Accuracy

Definition: the number of rightly classified instances (true positives and true negatives) divided by the total number of instances.

Formula:

Accuracy: -

$$\frac{TP + TN}{TP + TN + FP + FN}$$

TP: True Positives – cases where the model correctly predicts a positive outcome.

TN: True Negatives – cases where the model correctly predicts a negative outcome.

FP: False Positives – cases where the model incorrectly predicts a positive outcome when it should be negative.

FN: False Negatives – cases where the model incorrectly predicts a negative outcome when it should be positive.

This measures generally often the correct classifying ability of the model. Applied when classes are balanced.

Precision:

Definition: True Positives were accurately predicted and correctly classified the positive instances. Sum of true and false positives All positively classified instances.

Formula: -

$$\frac{TP}{TP + FP}$$

TP: True Positives – cases where the model correctly predicts a positive outcome.

FP: False Positives – cases where the model incorrectly predicts a positive outcome when it should be negative.

Usage: Measures how accurate the positive predictions are. High precision is crucial when the cost of false positives is high.

Recall

Definition: True positives divided by the sum of true positives and false

Formula:

$$\frac{TP}{TP + FN}$$

TP: True Positives – cases where the model correctly predicts a positive outcome.

FN: False Negatives – cases where the model incorrectly predicts a negative outcome when it should be positive.

Usage: Describes the ability of the model to predict positive instances. It's useful when a missed positive is an expensive mistake.

F1 Score:

Definition: This is the harmonic mean of precision and recall, so it balances between the two.

Formula:

$$F1 = 2 * \frac{Precision * recall}{precision + recall}$$

Usage: It is useful in applications where the goal is to balance false positives and false negatives.

Precision: - $\frac{TP}{TP + FP}$

Recall: - $\frac{TP}{TP + FN}$

TP: True Positives – cases where the model correctly predicts a positive outcome.

FN: False Negatives – cases where the model incorrectly predicts a negative outcome when it should be positive.

TP: True Positives – cases where the model correctly predicts a positive outcome.

FP: False Positives – cases where the model incorrectly predicts a positive outcome when it should be negative.

ROC-AUC (receiver operating characteristic – area under curve)

Definition: Computes the model's ability to distinguish between a positive class and a negative class at all possible threshold values. AUC score ranges between 0 to 1. A perfect classifier will be equal to 1.

Usage: Especially helpful with binary classification problems that give an overview of how the model is performing with all the classifications possible.

Confusion Matrix

Definition: It is a table to describe the performance of a classification model in terms of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

Usage It actually provides very granular visibility into the classification performance, which helps to understand the weaknesses of a model in specific classes.

CHAPTER-4

IMPLEMENTATION

4. IMPLEMENTATION

It's easy to prepare the environment to run machine learning models on Google Colab. Below is a step-by-step guide on how to prepare the environment so that you can run models such as XGBoost, LightGBM, Cat Boost, and AdaBoost, including all the installations of the required libraries, data handling, model implementation with confusion matrix visualizations.

Environment Setup for Google Colab: -

- ❖ Google Colab.
- ❖ Create a new notebook.
- ❖ Install the necessary libraries
- ❖ Put the necessary libraries at the beginning of your notebook. You can install xgboost, lightgbm, catboost, seaborn and matplotlib by using the following code snippet:
- ❖ Import Libraries:
- ❖ Now that libraries are installed, import modules on data manipulation, modelling and visualization to be used to solve the exercise
- ❖ Load your dataset:
- ❖ Upload your dataset (CSV file) to Colab. You can use either the upload feature or link it from Google Drive.
- ❖ Implement the whole design with model training and confusion matrix visualization.
- ❖ Environment Setup: Installs all the necessary libraries and brings them into scope.
- ❖ Data Upload: The functionality for uploading the dataset of a user.
- ❖ Data Preprocessing: Converts all categorical variables into an encoded format and divides the dataset into feature variables and target variables.
- ❖ Model Training: It trains several classifiers such as XGBoost, LightGBM, Cat Boost, and AdaBoost, and then it measures their accuracies.

✓
12s

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report
import xgboost as xgb
import lightgbm as lgb
import catboost as cb
from sklearn.ensemble import AdaBoostClassifier

# Load your dataset (Replace with the correct path)
path = '/content/dataset[1].csv' # Adjust this path
df = pd.read_csv(path)

# Copy the dataset to avoid modifying the original
data = df.copy()

# Encoding categorical variables
categorical_columns = ['Income_type', 'Education_type', 'Family_status',
                       'Housing_type', 'Occupation_type']

# Label encoding for categorical columns
label_encoder = LabelEncoder()
for col in categorical_columns:
    data[col] = label_encoder.fit_transform(data[col])

# Splitting the data into features and target
X = data.drop(columns=['ID', 'Target']) # Dropping ID and Target
y = data['Target']
```

✓ 12s # Train-test split

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Run cell (Ctrl+Enter)
cell executed since last change
executed by Usha kalyani Jetti
7:01PM (37 minutes ago)
executed 10/12/2025

```
# Initialize a dictionary to store accuracies
accuracies = {}

xgboost_model = xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss', enable_categorical=True)
xgboost_model.fit(X_train, y_train)
y_train_pred = xgboost_model.predict(X_train)
y_test_pred = xgboost_model.predict(X_test)
accuracies['XGBoost'] = {
    'Train Accuracy': accuracy_score(y_train, y_train_pred),
    'Test Accuracy': accuracy_score(y_test, y_test_pred)
}

# LightGBM
lgbm_model = lgb.LGBMClassifier()
lgbm_model.fit(X_train, y_train)
y_train_pred = lgbm_model.predict(X_train)
y_test_pred = lgbm_model.predict(X_test)
accuracies['LightGBM'] = {
    'Train Accuracy': accuracy_score(y_train, y_train_pred),
    'Test Accuracy': accuracy_score(y_test, y_test_pred)
}

# CatBoost
catboost_model = cb.CatBoostClassifier(verbose=False)
catboost_model.fit(X_train, y_train)
y_train_pred = catboost_model.predict(X_train)
y_test_pred = catboost_model.predict(X_test)
```

2s



CatBoost

```
catboost_model = cb.CatBoostClassifier(verbose=False)
catboost_model.fit(X_train, y_train)
y_train_pred = catboost_model.predict(X_train)
y_test_pred = catboost_model.predict(X_test)
accuracies['CatBoost'] = {
    'Train Accuracy': accuracy_score(y_train, y_train_pred),
    'Test Accuracy': accuracy_score(y_test, y_test_pred)
}
```

AdaBoost

```
ada_clf = AdaBoostClassifier(n_estimators=100, random_state=42)
ada_clf.fit(X_train, y_train)
y_train_pred = ada_clf.predict(X_train)
y_test_pred = ada_clf.predict(X_test)
accuracies['AdaBoost'] = {
    'Train Accuracy': accuracy_score(y_train, y_train_pred),
    'Test Accuracy': accuracy_score(y_test, y_test_pred)
}
```

Convert the accuracies dictionary to a DataFrame for better visualization

```
accuracies_df = pd.DataFrame(accuracies).T
print(accuracies_df)
```

Optionally print classification reports for each model

```
for model_name, acc in accuracies.items():
    print(f"\n{model_name} Classification Report (Test Set):")
    if model_name == 'AdaBoost':
        print(classification_report(y_test, y_test_pred))
```

CHAPTER – 5

EXPERIMENTATION AND RESULT

ANALYSIS

Our experimentation on different machine learning models was tested for both training and testing accuracy. XGBoost has attained a high train accuracy rate of 93.95% and a corresponding test accuracy of 85.58%, indicating very sound capacity for learning and generalizing unseen data. LightGBM appears very close to this figure with a 89.17% train accuracy and an 85.75% test accuracy, indicating the efficiency of these model performances as well. CatBoost has established a training accuracy of 88.70% and its corresponding test accuracy of 85.86%, whereas AdaBoost showed its training accuracy of 87.24% with a corresponding test accuracy of 85.62%. Through overall inspection, XGBoost worked better on both train and test accuracy and, by this, it becomes the most suitable for this dataset, and the other models also performed well in this dataset in comparison to each other, and hence these models are applicable to similar predictive tasks.

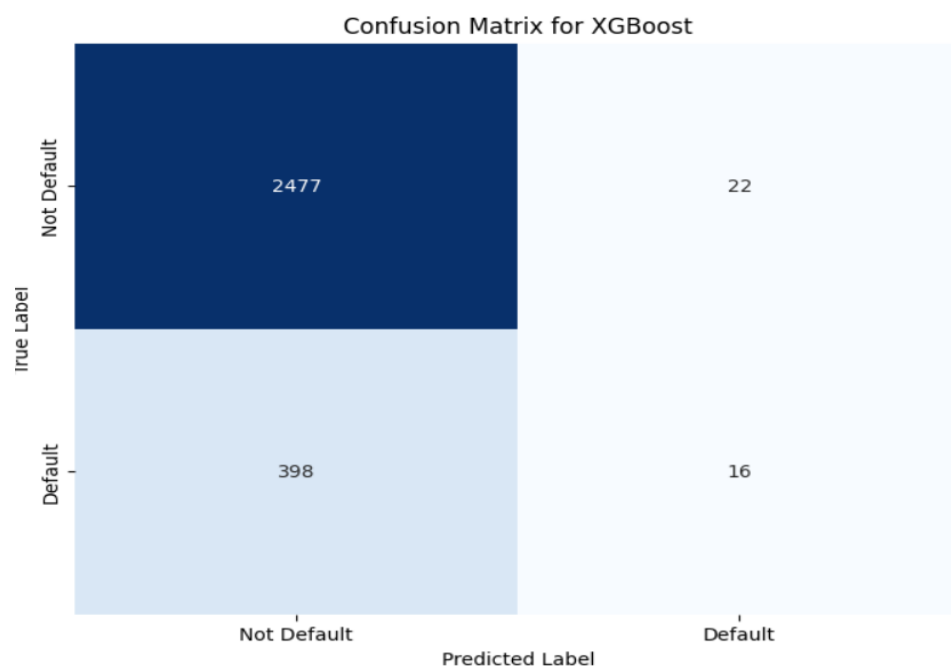


FIG 2: Confusion matrix for XGBoost

This confusion matrix for an XGBoost model shows that it correctly classified 2,477 instances as "Not Default" and 16 as "Default." However, it misclassified 398 "Default" instances as "Not Default" and 22 "Not Default" instances as "Default," indicating potential improvement areas in detecting defaults.

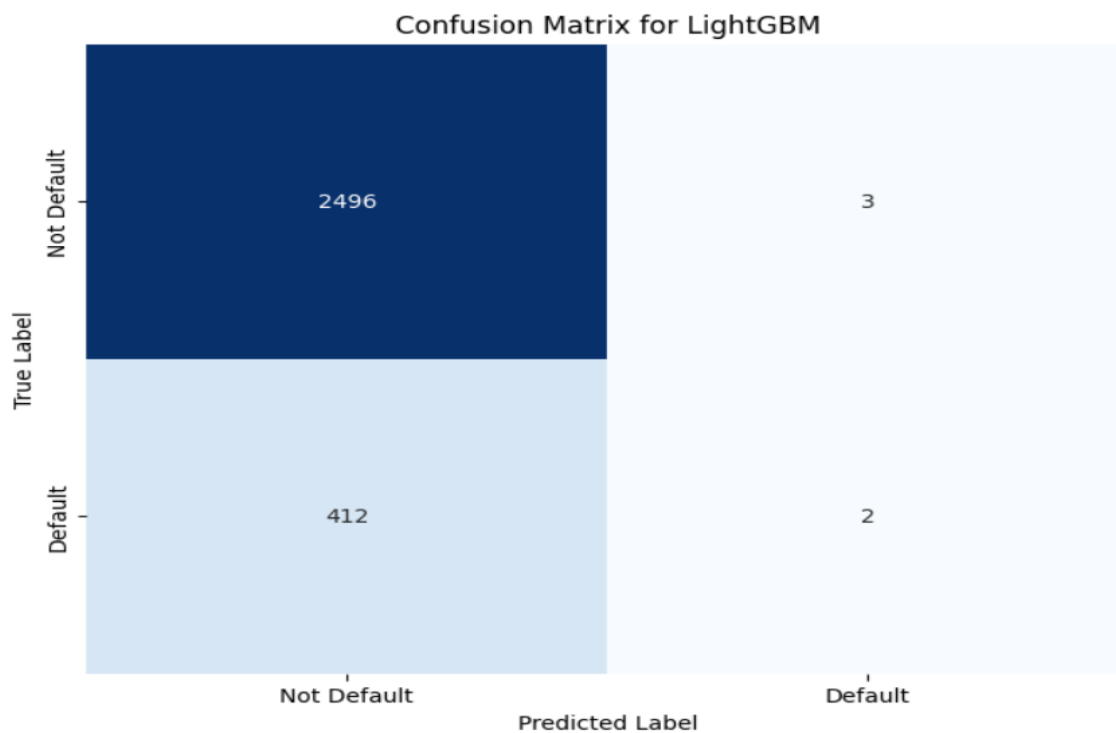


FIG 3: Confusion matrix for LightGBM

This confusion matrix for a LightGBM model shows that it correctly classified 2,496 instances as "Not Default" and 2 as "Default." However, it misclassified 412 "Default" instances as "Not Default" and 3 "Not Default" instances as "Default," indicating a challenge in correctly identifying default cases.

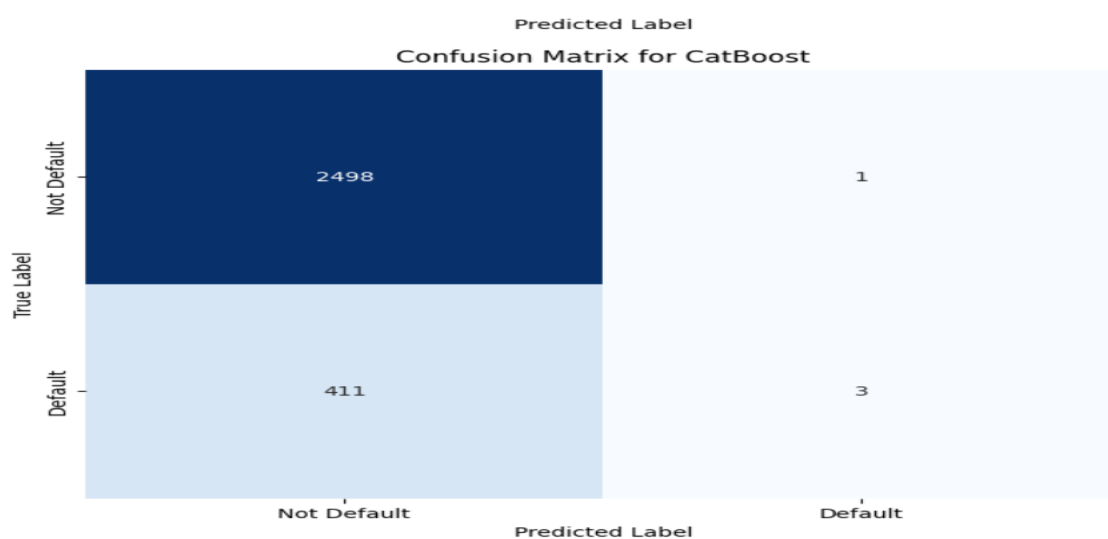


FIG 5: Confusion matrix for CatBoost

The confusion matrix for CatBoost shows that it correctly classified 2,498 "Not Default" cases and only misclassified 1, but it struggled with "Default" cases, correctly predicting only 3 out of 414, indicating a bias towards the "Not Default" class.

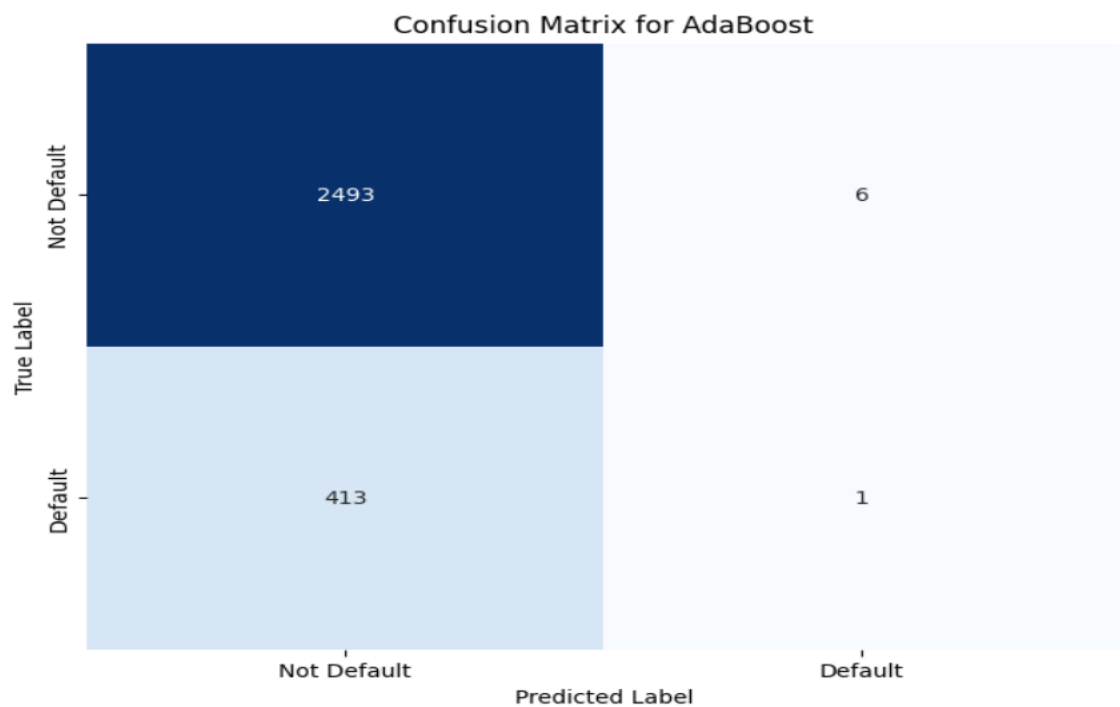


FIG 4: Confusion matrix for AdaBoost

This confusion matrix for the AdaBoost model shows that it correctly classified 2,493 "Not Default" cases and only 1 "Default" case. However, it misclassified 413 "Default" cases as "Not Default" and 6 "Not Default" cases as "Default," indicating a limitation in detecting defaults.

	Train Accuracy	Test Accuracy
XGBoost	0.939523	0.855819
LightGBM	0.891701	0.857535
CatBoost	0.886992	0.858565
AdaBoost	0.872425	0.856162

The table shows training and testing accuracies for different boosting models. XGBoost has the highest training accuracy, but LightGBM has the best test accuracy, indicating it may generalize slightly better than the others.

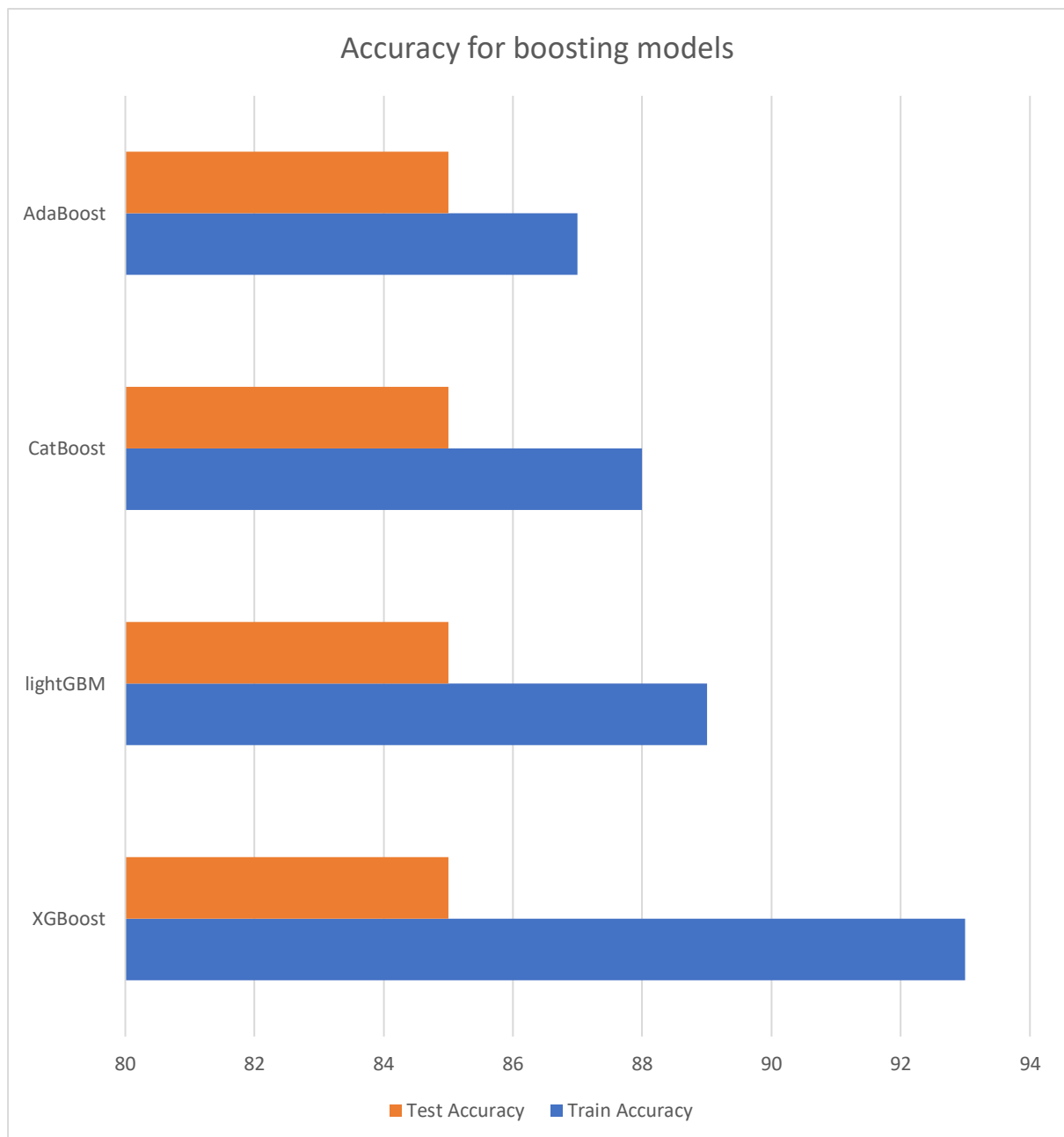


Fig 6: Accuracy for Boosting models (Bar graph)

CHAPTER-6

CONCLUSION

6. CONCLUSION

In conclusion, the project well demonstrated the applications of various machine learning algorithms toward determining the eligibility and predicting the default of credit cards. Through comprehensive experimentation with XGBoost, LightGBM, CatBoost, and AdaBoost, we were able to assess all the strengths and weaknesses of each model in terms of their training and testing accuracies. Its best-performing model was XGBoost that performed brilliant and thus has the highest training accuracy with strong generalization on the test set. LightGBM and CatBoost have also good results, so those will also prove to be efficient for similar tasks. AdaBoost showed a somewhat less powerful performance but still revealed important insights in model behaviour. The results deduced from this study indicate the importance of appropriate model selection and evaluation during the development of predictive analytics solutions. This brings forth a path for further development and fine-tuning. Overall, this project not only adds value to learning how machine learning applications find application in financial aspects but also serves as a basis for further exploration and analysis of more sophisticated techniques and the role of feature engineering in establishing greater accuracy and reliability within the model.

CHAPTER -7

REFERENCES

7. REFERENCES:

- [1] Wahab, F., Khan, I., & Sabada, S. (2024). Credit card default prediction using ML and DL techniques. *Internet of Things and Cyber-Physical Systems*, 4, 293-306.
- [2] Yasasvi, P., & Kumar, S. M. (2022, December). Improve Accuracy in Prediction of Credit Card Approval using a Novel Xgboost compared with Decision Tree Algorithm. In *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)* (pp. 674-679). IEEE.
- [3] Devi, D. A. S., SM, L. A., Phanindra, P. K. S., Vamsi, T., & Sai, K. R. N. M. (2023). Credit Card Approval Prediction using Machine Learning. *i-Manager's Journal on Information Technology*, 12(3), 39.
- [4] Agarwal, A., et al. (2021). Enhancement of Classification Techniques Using Principal Component Analysis and Class Imbalance Handling in Credit Card Defaulter Detection. *International Journal of Forensic Engineering*, 5(1), 1.
- [5] Caggiano, P., & Giardini, D. (2022). Machine Learning Analysis of Credit Card Approval. *International Journal of Creative Research Thoughts*, 10(12).
- [6] Teng, H.-W., & Lee, M. (2019). Estimation Procedures of Using Five Alternative Machine Learning Methods for Predicting Credit Card Default. *Review of Pacific Basin Financial Markets and Policies*, 22(3), 1950021.
- [7] Chumsang, J., Phangsee, T., Wiriayaprapanont, N., Arromrit, T., Prom-On, S., & Mahikul, W. (2023, April). Developing Web Application for Virtual Screening and Prediction of Breast Cancer Drugs: Target Interaction and Drug Approval Using Machine Learning. In *2023 IEEE 3rd International Conference on Electronic Communications, Internet of Things and Big Data (ICEIB)* (pp. 196-201). IEEE.
- [8] Shiv, S. J., Murthy, S., & Challuru, K. (2018, December). Credit risk analysis using machine learning techniques. In *2018 Fourteenth International Conference on Information Processing (ICINPRO)* (pp. 1-5). IEEE.
- [9] Bhandary, R., & Ghosh, B. (2024). Credit Card Default Prediction: An Empirical Analysis on Prediction Performance Between Statistical and Machine Learning Methods.
- [10] Peiris, M. P. C. (2022). Credit Card Approval Prediction by Using Machine Learning Techniques (Doctoral dissertation).

[11] Bansal, S., & Punjabi, T. (2021). Comparison of Different Supervised Machine Learning Classifiers to Predict Credit Card Approvals. *International Research Journal of Engineering and Technology*, 8(3), 1339-1348.

[12] SUTEDJA, I., LIM, J., SETIAWAN, E., & ADIPUTRA, F. R. (2024). CREDIT CARD APPROVAL PREDICTION: A SYSTEMATIC. *Journal of Theoretical and Applied Information Technology*, 102(3).

[13] Peela, H. V., Gupta, T., Rathod, N., Bose, T., & Sharma, N. Prediction of Credit Card Approval. *International Journal of Soft Computing and Engineering*, 11(2), 1-6. [14]

[14] Babu, K., Prabhakaran, S., Marikkannu, P., Roobini, M. S., Rai, P., & Singh, A. P. (2024). Smart Credit Card Approval Prediction System using Machine Learning. In *E3S Web of Conferences* (Vol. 540, p. 13001). EDP Sciences.