Teesside University

School of Computing, Engineering & Digital Technologies

MACHINE LEARNING MODULE (CIS4035-N)

CREDIT CARD DEFAULT PREDICTION USING MACHINE LEARNING ALGORITHMS

An Approach to Handling Imbalanced Dataset

EZEMA, CHUKWUJEKWU JOSEPH
B1190869

20/05/2022

ABSTRACT

Credit Card Default Predication, like every other financial or bank-related problem, is associated with an imbalanced dataset – which usually affects the performance of machine learning models in the financial sector. In this research, 5 classification models have been applied: Logistic Regression, KNN, Gradient Boost, Random Forest, XGB (Extreme Gradient Boost), and a deep learning model (Artificial Neural Networks) to effectively demonstrate the influence of imbalance on the result and how to handle it with different sampling techniques. Thus, Feature Scaling, Resampling, GridSearchCV, Cross-Validation, Model Evaluation, SMOTE oversampling technique, and PCA was applied to handle the skewness of the imbalance. However, in order not to be deceived by high accuracies as expected, the evaluation metrics used were strictly by the macro average, precision-recall scores, and areas under curve scores. Random Forest and ANN had the best scores at 60.7% AUC on SMOTE samples and 67.2% on normal samples for each category respectively, while the bagging methods (Gradient Boost and XG Boost) had the overall best performance across all the experiments. The prediction was limited by lack of time to experiment similar datasets under the same condition to critically analyze the difference.

Table of Contents

1.0 INTRODUCTION

Since the advent of digital banking in this contemporary time, the growing rate of credit card transactions in the banking sector has caused some operational risk measures to be applied by financial institutions [9]. Credit Card has become one of the most used financial products over the years, thus, credit providers are beginning to face huge, uncontrollable amount of credit card delinquencies [1]. The biggest threat faced by both credit card clients and their providers is that, regardless of the financial buoyance of most card users, they overspend on credit and cash-card debts [2]. The purpose of credit card default prediction is to minimize the risks associated with giving loans via credit cards by predicting whether a credit card applicant will default or not - from the historical credit data of customers [10]. However, such predictions do not come easily guaranteed as there are factors that can affect the creditworthiness of a customer; such as the amount in debt, number of existing credits, frequency of overdue payments, income, financial dependents, age, family status, etc. Solutions to some of these issues can be provided by a robust application of machine learning algorithms [20].

In Literature Review, various machine learning models have been proposed by so many research papers for credit card default prediction - showing different results, data strategies, evaluation metrics, and conclusions. The most widely used algorithms for credit card default predictions are Logistic Regression (LR) [1][3], Decision Tree (DT) [1][3], Random Forest (RF) [1][2][3][4][5], K-Nearest Neighbours (KNN) [2], Support Vector Machine (SVM) [2], Gradient Boost Decision Tree (GBDT) [4][5], XG Boost (XGB) [5], and AdaBoost [2][3][5]. Different studies show that some algorithms can perform more than others in the same experiment: RF, best at an accuracy of 82.12% for weighted models [3], and 82% compared to LR and DT [1]; AdaBoost, best at 88% accuracy compared to six others [2]; GBDT tops the performance chart at 88.7% accuracy for an experiment carried out with 3 different imbalanced datasets with K-means SMOTE resampling method outperforming other resampling techniques [4], and 78% best performance achieved under AUC [5].

Credit Card Default Predictions are mostly classification problems [19]. And in most classification problems, handling an imbalanced dataset is key to improving machine learning models [4]. In addition, the results of Alam, T.M. et al [4] show that classifier models for the balanced datasets performed far better than the unbalanced. More so, the datasets with oversampling techniques (K-means SMOTE, SMOTETomek) have better performance than the under-sampling methods (NearMiss, Random Undersampling).

Therefore, with regards to the above studies, this paper proposes to address the issue of the imbalanced dataset in credit card default prediction; and to effectively lessen the threats posed by poor prediction models across the banking and consumer lending sectors, by using five machine learning algorithms: Logistic Regression, Gradient Boost Decision Tree, K-Nearest Neighbours, Random Forest, and XGBoost; to a 90% performance level each across all metrics. This is to improve the application of machine learning algorithms in other credit card predictions.

1.1 Data Description

The dataset was gotten from a public domain dataset repository of [Kaggle](#) by Seanny @rikdifos - who created the dataset in 2019 and updated it 2 years ago for machine learning projects. The expected update frequency was yearly.

Two Datasets:

1. Application record (contains general information about the applicant, such as applicant gender, DOB, education type, assets that applicant had, etc.)
2. Credit record (contains applicant's loan payment records)

Why this dataset?

- Dataset is real from an undisclosed bank to Kaggle Database and is maintained yearly
- It serves as a good example for experimenting with imbalanced dataset cases
- Dataset has basic features of credit card default prediction from related works [1]
- 90% features for modeling similar datasets like in loan applications, mortgages, etc.

2.0 DATA EXPLORATION AND FEATURE ENGINEERING

2.1 Load and Clean Data

To begin this predictive analysis, the dataset was explored to a full understanding of what the dataset was all about; the nature of the contents, and how to prepare them for machine learning modeling. The following steps were taken in exploring the data:

Import and Install Libraries: The appropriate libraries were put into one place in each section for ease of running the code and to be certain necessary libraries for the experiment were put on.

Load and Clean Data: The main folder contained two original datasets – the application file and the credit file. The Application File contained 438,557 instances and 18 features. 47 duplicated rows were removed, and 134,203 null values, which were up to 30% from the Occupation Type column, were replaced with 'Unknown'. However, the Credit File had no duplicated row nor null value from the 1,048,575 instances and 3 features. Although, it contained only 45,985 unique IDs. This implies each ID contains variable features.

Column Description

application_record.csv

#	Feature name	Explanation	Remarks
1	ID	Client number	
2	CODE_GENDER	Gender	
3	FLAG_OWN_CAR	Is there a car	
4	FLAG_OWN_REALTY	Is there a property	
5	CNT_CHILDREN	Number of children	
6	AMT_INCOME_TOTAL	Annual income	
7	NAME_INCOME_TYPE	Income category	
8	NAME_EDUCATION_TYPE	Education level	
9	NAME_FAMILY_STATUS	Marital status	
10	NAME_HOUSING_TYPE	Way of living	
11	DAYS_BIRTH	Birthday	Count backwards from current day (0), -1 means yesterday

Fig. 1. A snippet of the Application Columns

```
In [16]: # Displaying first-and-last-5 rows of the credit data
credit
```

Out[16]:

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	X
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	C
...	...	...	...
1048570	5150487	-25	C
1048571	5150487	-26	C
1048572	5150487	-27	C
1048573	5150487	-28	C
1048574	5150487	-29	C

1048575 rows × 3 columns

Fig. 2. A snippet of the Credit File

2.2 Feature Engineering

At this stage, the two datasets were still prepared differently. This was to avoid loss of data or misrepresentation of features separately summarized and prepared before joining them.

<u>To Integer Function:</u>

Out[11]:

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE
0	5008804	M	1	1	0	427500.0	Working
1	5008805	M	1	1	0	427500.0	Working
2	5008806	M	1	1	0	112500.0	Working
3	5008808	F	0	1	0	270000.0	Commercial associate
4	5008809	F	0	1	0	270000.0	Commercial associate

To Label Encoding Function:

- ✓ Converted the multivariate categorical data to similar labels for 1-hot encoding
- ✓ Dropped some unwanted columns before applying the dummy feature function

Hot Encoding Function:

- ✓ Converted the labeled categorical data to features
- ✓ Renamed the columns for better recognition of what they represented

```
df2 = hot_encoding(df1)
df2
```

Out[13]:

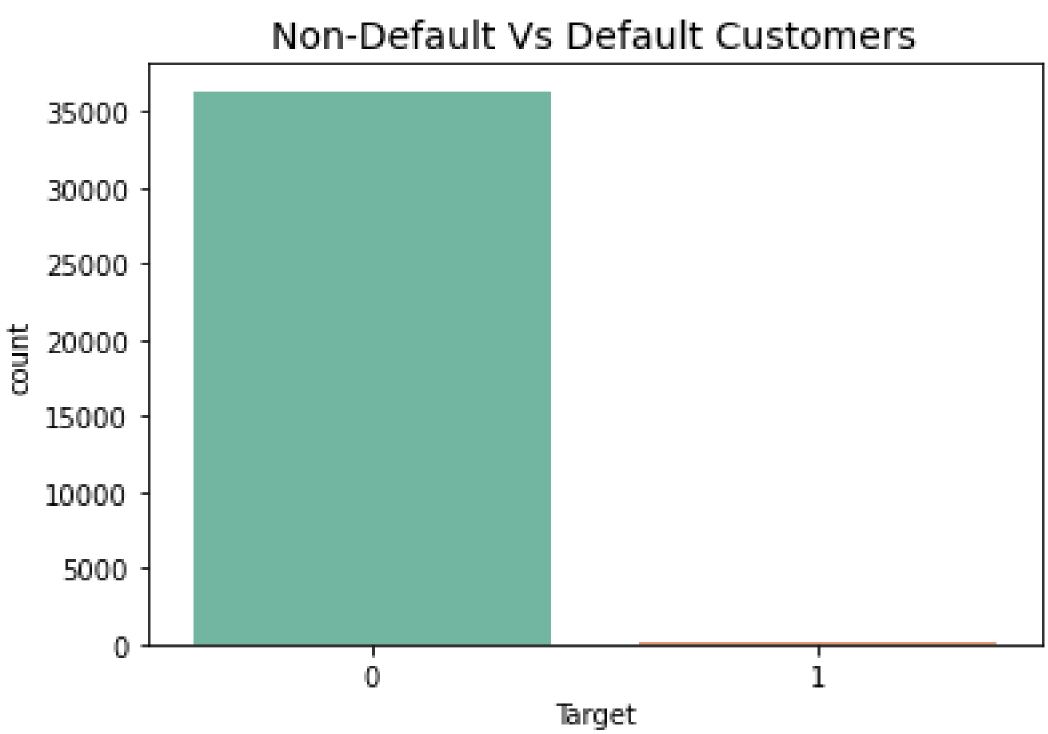
	ID	Car	Realty	Income	MobilePhone	WorkPhone	OtherPhone	Email	FamilySize	Age	Experience	Household	Gender_F	Gender_M
0	5008804	1	1	427500.0	1	1	0	0	2	33	13	2	0	1
1	5008805	1	1	427500.0	1	1	0	0	2	33	13	2	0	1
2	5008806	1	1	112500.0	1	0	0	0	2	59	4	2	0	1
3	5008808	0	1	270000.0	1	0	1	1	1	53	9	1	1	0
4	5008809	0	1	270000.0	1	0	1	1	1	53	9	1	1	0

Transform Target Function:

- ✓ Categorized Target Class from credit dataset - "Default (1)" or "Not_Default(0)" on a new column, Target

The datasets were merged here after the above transformation of the Target class.

```
Shape after transforming to Target Class: (36457, 26)
Number of Non-Default Class = 36334
Number of Default Class = 123
Percent of Non-Default Class = 99.66%
Percent of Default Class = 0.34%
```



2.3. Data Visualization

2.3.1. Visualizing Outliers across the Numerical Features

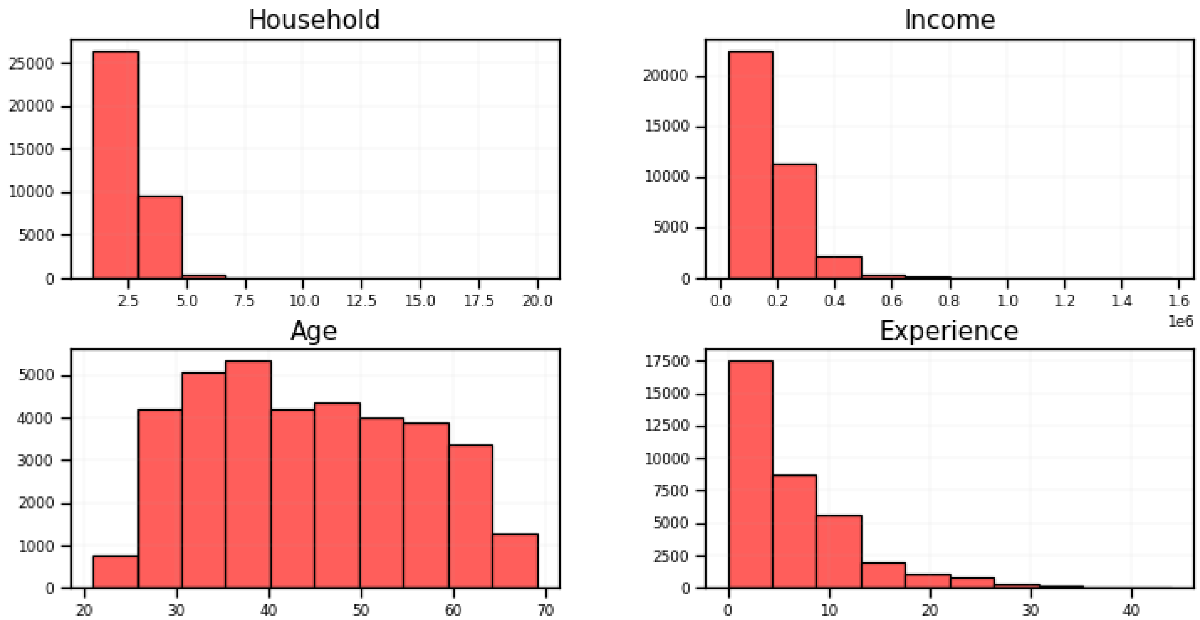
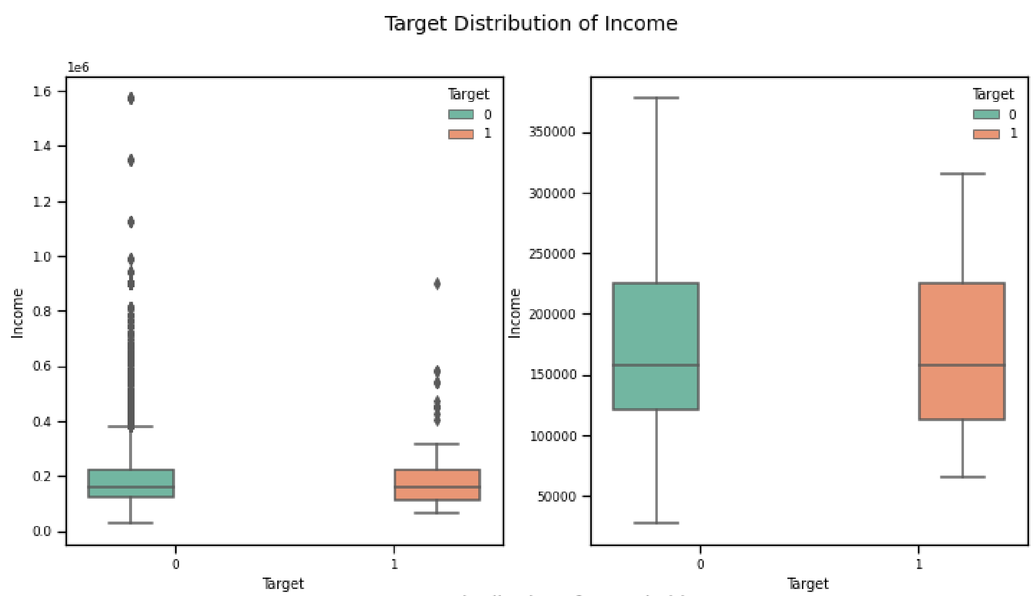
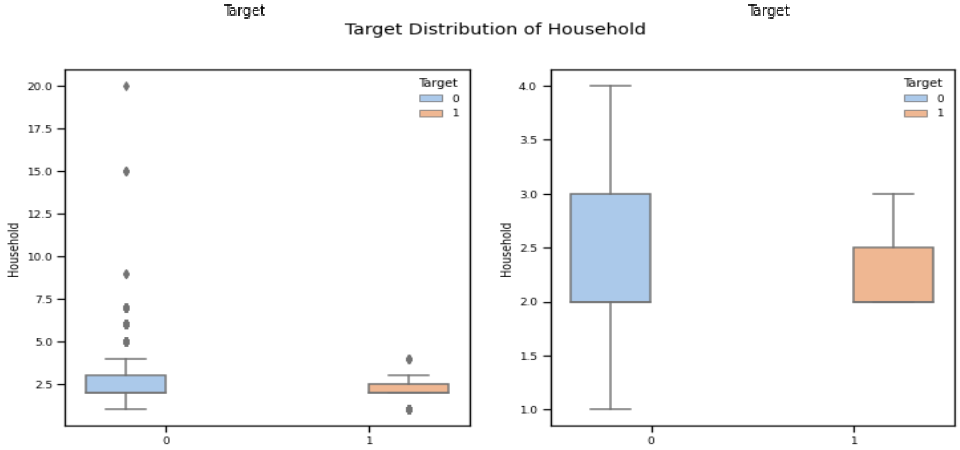


Fig. 3. Outliers across the Quantitative Features





2.3.2. Visualizing Observations across the Categorical Features

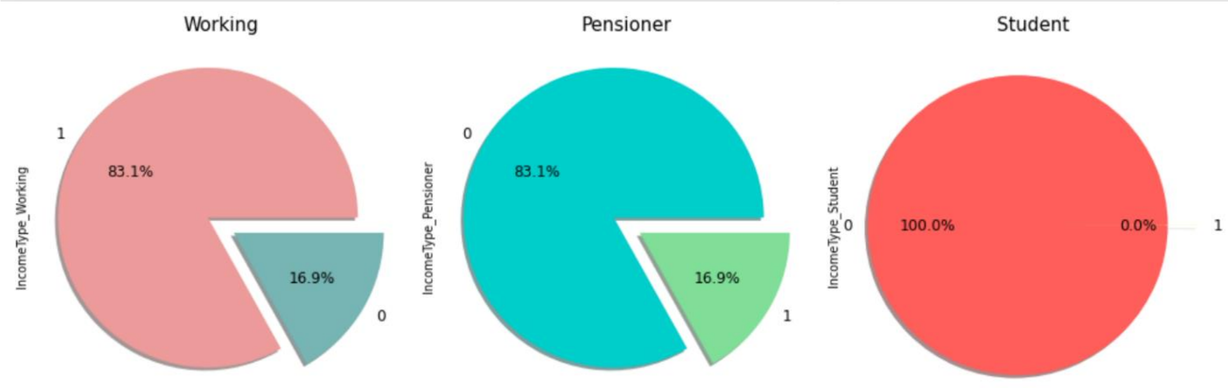


Fig. 4. Observation across Income Type

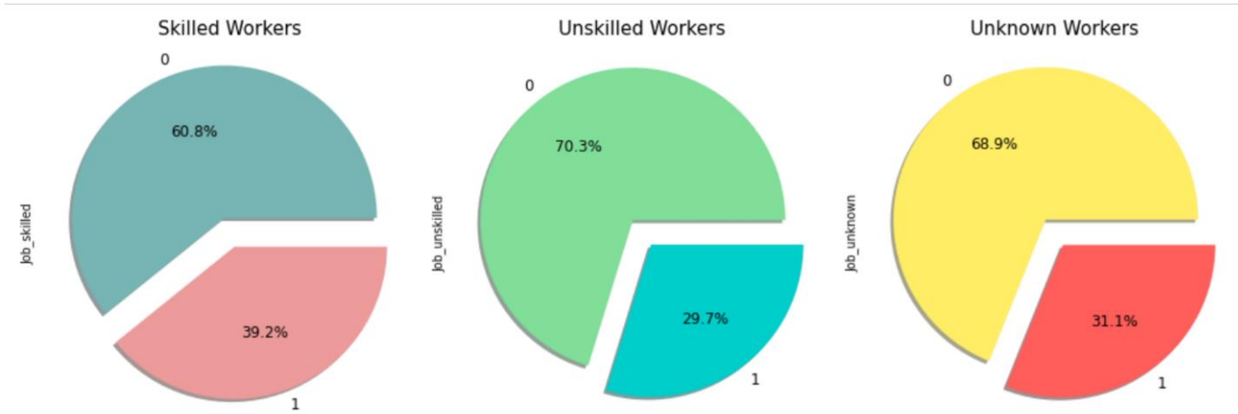


Fig. 5. Observation across Job Type

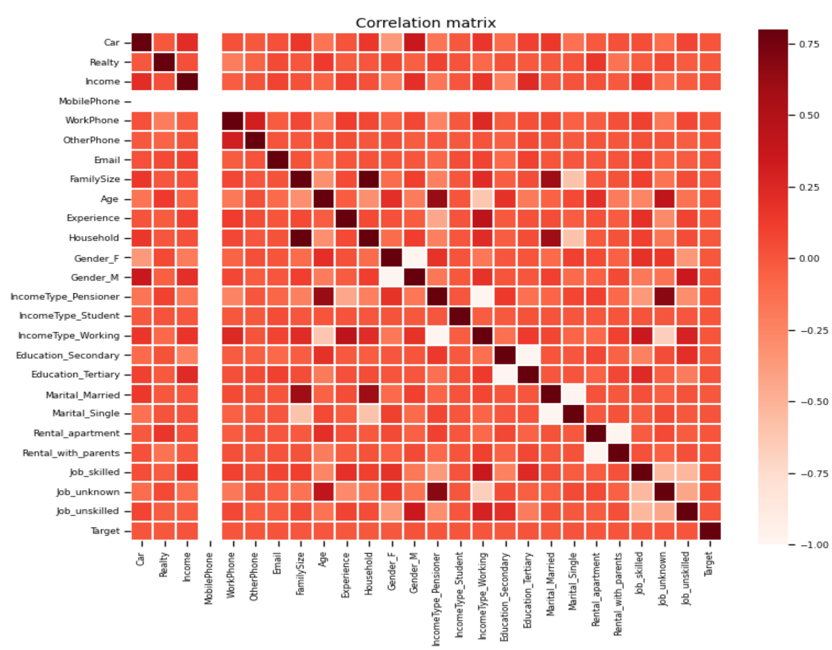


Fig. 6. Correlation Matrix before feature selection

Reflection: Firstly, the target distribution has shown that there is a high imbalance in the dataset. Secondly, the continuous variables have variable mean and number scales. To enhance the model, the values will be scaled within a common range. The Income and Household contain outliers which will be avoided when scaling by using a StandardScaler method instead of MinMaxScaler. The Student Column revealed no data and will be dropped alongside the MobilePhone that did not correlate with other features and the Target.

3.0 FEATURE SELECTION AND DATA PREPARATION

3.1 Feature Scaling

StandardScaler: The StandardScaler calculates the probability of a score occurring within the normal distribution. It compares two scores that are from different normal distributions. The standard score does this by converting (in other words, standardizing) scores in a normal distribution to z-scores – which has an average of 0 and a standard error of 1, and usually between the range of -3 to 3 (mostly covers 99% of data) into what becomes a standard normal distribution [6],[7].

$$x_{std} = \frac{x-\mu}{\sigma}$$

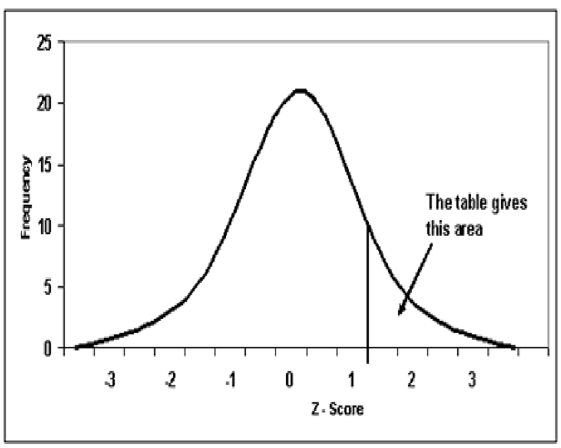


Fig. 7. The standard score probability

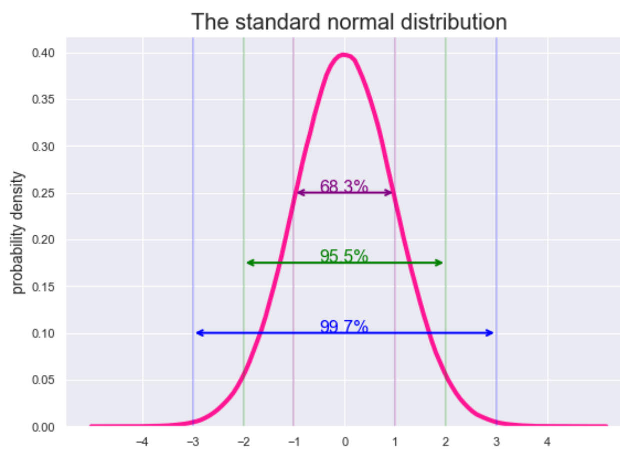


Fig. 7a. The standard score probability graph

```
In [26]: """ Step 3: Prepare and select best features for the training"""
         # 3.1 Scale the numeric columns using Standard Scaler

         # copy of datasets
         df = credit_app.copy()
         for i in num_col:

             # fit on training data column
             scale = StandardScaler().fit(df[[i]])

             # transform the training data column
             df[i] = scale.transform(df[[i]])

         df
```

Out[26]:

	Car	Realty	Income	MobilePhone	WorkPhone	OtherPhone	Email	FamilySize	Age	Experience	Household	Gender_F	Gender_M
0	1	1	2.365845	1	1	0	0	2	-0.976304	0.997704	-0.217308	0	1
1	1	1	2.365845	1	1	0	0	2	-0.976304	0.997704	-0.217308	0	1
2	1	1	-0.728827	1	0	0	0	2	1.283786	-0.372036	-0.217308	0	1

3.2. Feature Sampling Strategy

Resample Data:

The dataset was reduced because of train time. So, n_sample was preselected from the main data to train and evaluate the best output first.

```
In [28]: # 3.3. Get sample data
         # To reduce training time, resample data
         n_sample = 5000
         random_state = 42

         non_default = df[df['Target'] == 0].sample(n_sample, random_state=random_state) #resample only from the majority class
         default =  df[df['Target'] == 1]

         # Merge 2 subset
         sample = non_default.append(default).sample(frac=1, random_state=random_state).reset_index(drop=True)
         y = sample["Target"].values

         print("After resampling, Number of Default Instances : {}".format(sum(y==1)))
         print("After resampling, Number of Non-Default Instances : {}".format(sum(y==0)))

         sample
```

```
After resampling, Number of Default Instances : 123
After resampling, Number of Non-Default Instances : 5000
```

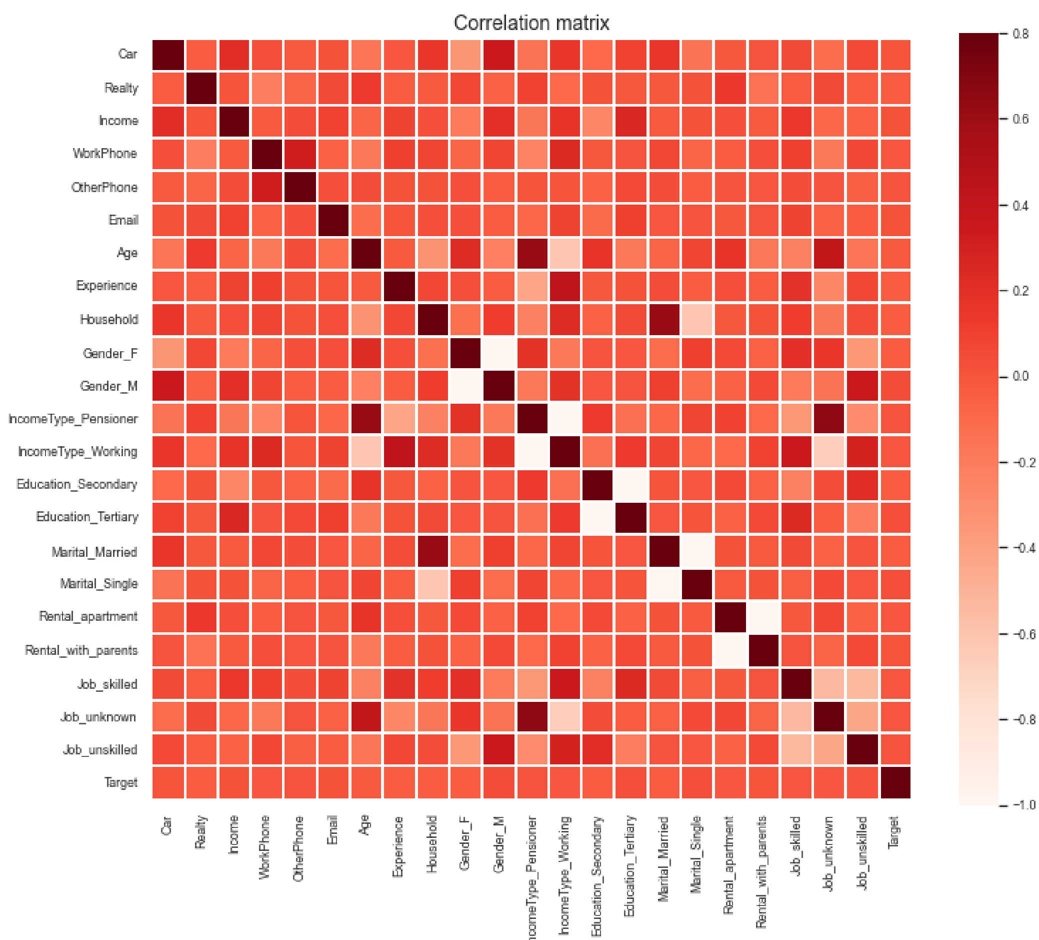


Fig. 8. Correlation Matrix after feature selection

3.3 Test Train Split

Attributes:

```
For the resample data of 5000 non-default instances:
Shape of X_train dataset:  (4098, 22)
Size of y_train dataset:  4098
Shape of X_test dataset:  (1025, 22)
Size of y_test dataset:  1025
----------------------------------------------------
Number of Default Instances in train set : 98
Number of Non-Default Instances in train set : 4000
Number of Default Instances in test set : 25
Number of Non-Default Instances in test set : 1000
```

3.4 SMOTE Oversampling Technique

Attributes:

```
In [31]: # 3.6 Train_Test Split samples for SMOTE oversampling
         # Using SMOTE as oversampling Technique for handling imbalanced dataset

         sm = SMOTE(sampling_strategy='minority', random_state=random_state)
         X_train_sm, y_train_sm = sm.fit_resample(X_train, y_train)
         print('The number of target class before oversampling: {}'.format(y_train.shape[0]))
         print('The number of target class after oversampling: {}'.format(y_train_sm.shape[0]))
         print("----------------------------------------------------")
         print("\nAfter OverSampling, counts of default: {}".format(sum(y_train_sm==1)))
         print("After OverSampling, counts of non-default: {}".format(sum(y_train_sm==0)))

         The number of target class before oversampling: 4098
         The number of target class after oversampling: 8000
         ----------------------------------------------------

         After OverSampling, counts of default: 4000
         After OverSampling, counts of non-default: 4000
```

4.0 MODELLING AND EXPERIMENTS

4.1 The Classifiers

As mentioned above, this project embarked on 5 classification models and a deep learning model – all from supervised learning types of machine learning algorithms.

Logistic Regression (LR): This is one of the most used classification algorithms for financial or banking problems [8]. It measures the probability of a class prediction between 0 and 1. The model was chosen because of its efficiency and simplicity of application. It has attributes like "liblinear," and "sag," for the solver parameter, which was applied to find the best functioning model for the perfect result.

K-Nears Neighbours Classifier (KNN): KNN learns based on similarity, unlike LR which uses linear separation between two classes [17]. Hence, the name 'nearest neighbour' – where k is the number of neighbours. The GridsearchCV uses the most suitable for n_neighbours to optimize or tune the best fit for the model [14]. In this project, the k was given the range of 2 to 8 according to how it suited each of the experiments.

Random Forest (RF): Random Forest is an ensemble technique that uses multiple trees, unlike Decision Tree, to classify a problem [12]. It is one of the most used algorithms for classification because of how it handles skewness in some data – especially in imbalanced datasets. RF had the best accuracy of 82.12% for weighted models [3], and 82% compared to LR and DT [1].

Gradient Boosting Decision Tree (GBDT): This model can also be called Gradient Boosting (GB). It is the advanced level of Decision Tree and Random Forest classifiers to address the limitations of a single tree or random forest (ensemble) by using multiple trees to classify a target class much better than RF because it learns independently at every iteration more than RF [15]. It was used in this project in the place of Decision Tree to get experiments on which was better for the dataset. GB DT tops the performance chart at 88.7% accuracy [4].

Extreme Gradient Boosting (XGB): XGB was designed to improve the gradient boosting models with speed and accuracy [13]. Bagging methods use very high complex parameters that take time to build; for example, high n_estimators for big data can take too long to train. But XGB focuses on train time reduction and model performance. In this project, max depth was used to optimize the model – as is the same with all other tree-based models.

Artificial Neural Networks (ANN): This can simply also be called Neural Networks. It uses a network of functions (parameters) to learn like in the human brain [11][18]. In this work, different epochs will be tested in each experiment. Neural Networks most times are used for cases of external data to manage the capacity and the performance of the model [16]. Therefore, it was only used in this work for experiment purposes.

4.2 Experiments

Six experiments were carried out, each with either of the different samples: baseline, SMOTE, and PCA samples - to compare and demonstrate the best sampling method suitable for an imbalanced dataset in classification prediction.

Exp 1a – Trained and Tested the 5 Classifiers using baseline sample (without oversampling)

Exp 1b – Trained and Tested the 5 Classifiers using PCA feature selection of the baseline

Exp 2a – Trained and Tested the 5 Classifiers using SMOTE samples from the oversampling

Exp 2b – Trained and Tested the 5 Classifiers using PCA feature selection of the SMOTE

Exp 3a – Trained and Tested the ANN model using a baseline sample (without oversampling)

Exp 3b – Trained and Tested the ANN model using SMOTE samples from the oversampling

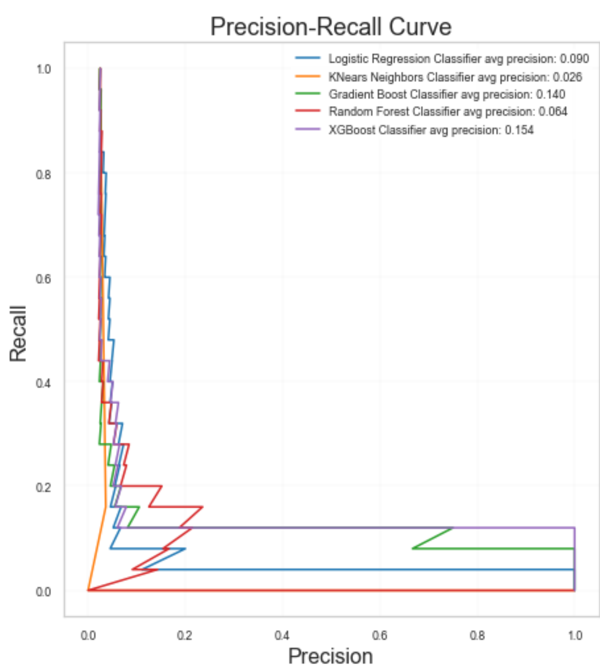
5.0 RESULTS

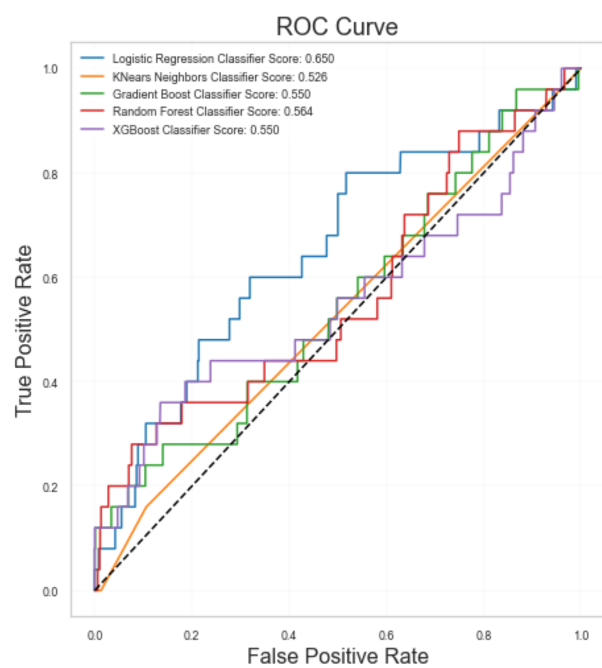
In classification problems, the evaluation metric is very important in a case where there is an imbalance. In the case of an imbalanced dataset, accuracy is not the best form of evaluation. Instead, AUC-ROC, Precision-Recall Score, and Macro Average expose the underperforming models accordingly.

Exp 1a: Classifiers on the baseline sample (without oversampling)

CLASSIFIER	ACCURACY	WEIGHTED F1-SCORE	MACRO F1-SCORE	P-R CURVE	ROC CURVE
Log. Regression	0.98	0.96	0.49	0.090	0.650
KNN	0.98	0.97	0.53	0.026	0.526
Random Forest	0.98	0.96	0.49	0.064	0.564
Gradient Boost	0.98	0.97	0.67	0.140	0.550
XG Boost	0.98	0.97	0.63	0.154	0.550

Table 1: Experiment 1a Result

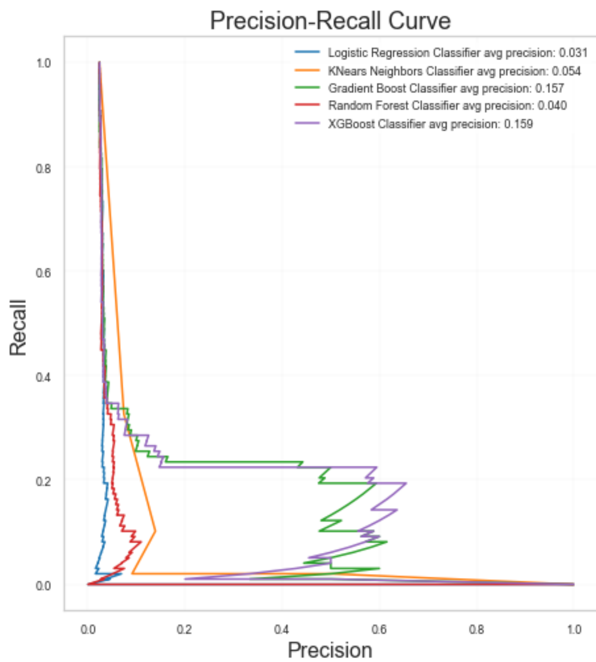


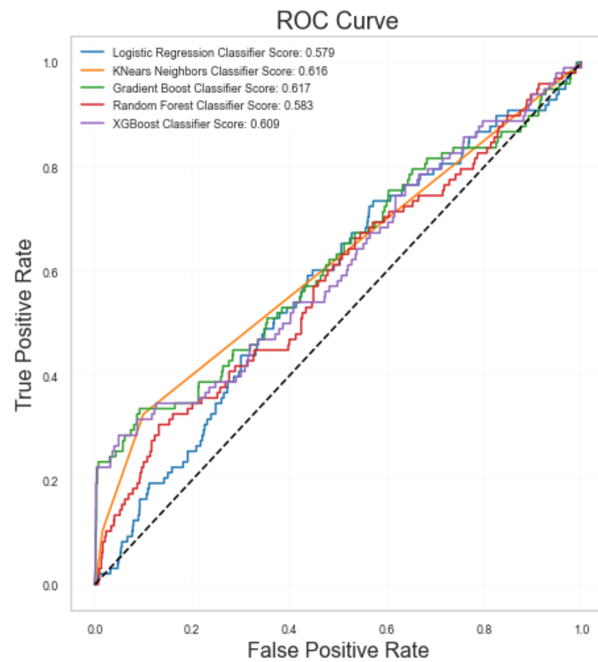


Exp 1b: Classifiers on PCA feature selection of the baseline

CLASSIFIER	ACCURACY	WEIGHTED F1-SCORE	MACRO F1-SCORE	P-R CURVE	ROC CURVE
Log. Regression	0.98	0.96	0.49	0.031	0.579
KNN	0.98	0.96	0.49	0.054	0.616
Random Forest	0.98	0.96	0.49	0.040	0.583
Gradient Boost	0.98	0.97	0.62	0.157	0.617
XG Boost	0.98	0.97	0.62	0.159	0.609

Table 2: Experiment 1b Result

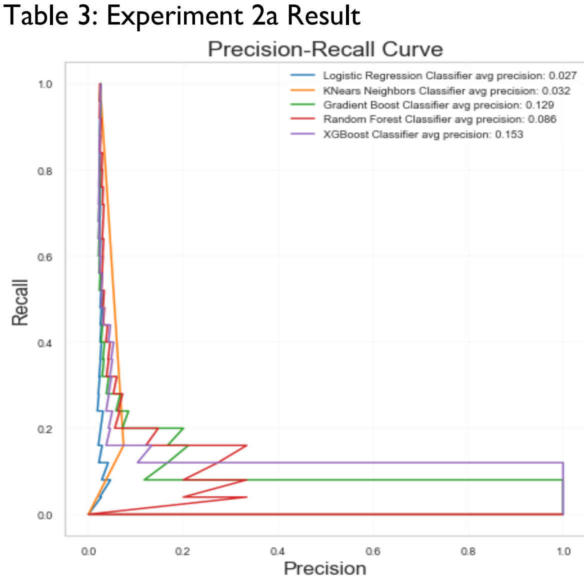


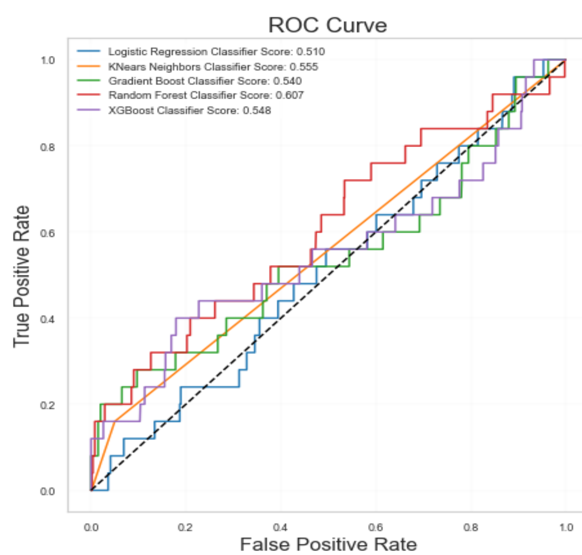


Exp 2a: Classifiers on SMOTE samples from the oversampling

CLASSIFIER	ACCURACY	WEIGHTED F1-SCORE	MACRO F1-SCORE	P-R CURVE	ROC CURVE
Log. Regression	0.88	0.92	0.50	0.027	0.510
KNN	0.96	0.96	0.58	0.032	0.555
Random Forest	0.84	0.89	0.50	0.086	0.607
Gradient Boost	0.96	0.96	0.58	0.129	0.540
XG Boost	0.97	0.97	0.60	0.153	0.548

Table 3: Experiment 2a Result

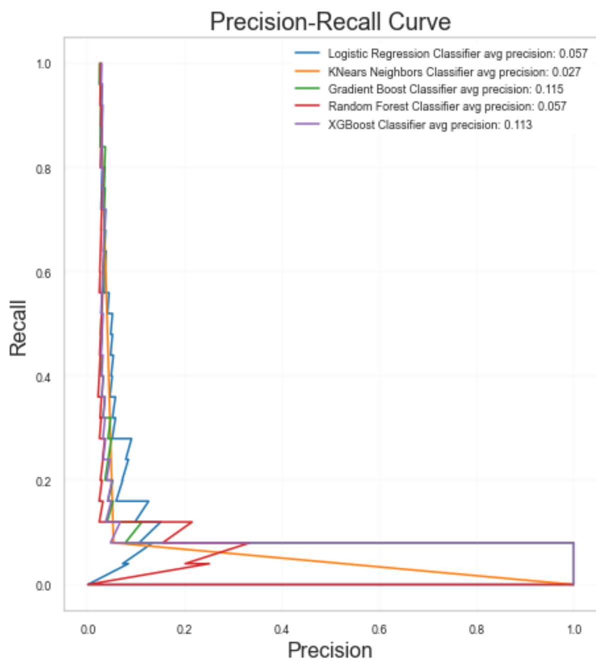


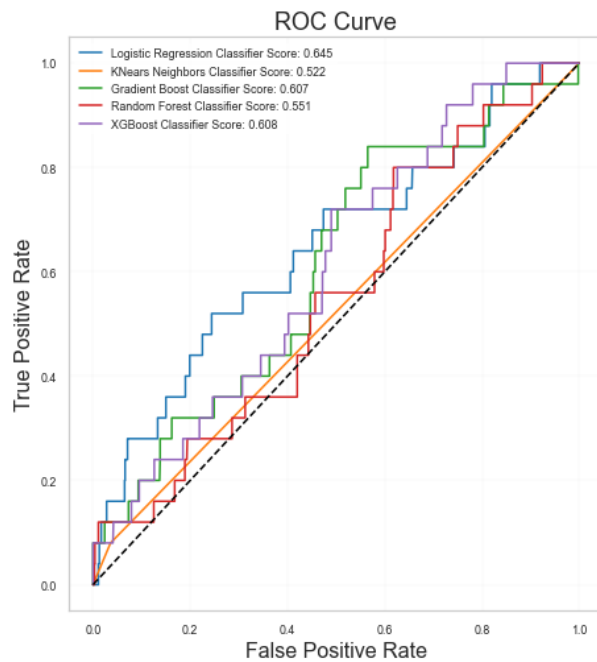


Exp 2b: Classifiers on PCA feature selection of the SMOTE samples

CLASSIFIER	ACCURACY	WEIGHTED F1-SCORE	MACRO F1-SCORE	P-R CURVE	ROC CURVE
Log. Regression	0.64	0.76	0.43	0.057	0.645
KNN	0.95	0.95	0.56	0.027	0.522
Random Forest	0.72	0.82	0.45	0.057	0.551
Gradient Boost	0.91	0.93	0.53	0.115	0.607
XG Boost	0.94	0.95	0.54	0.113	0.608

Table 4: Experiment 2b Result

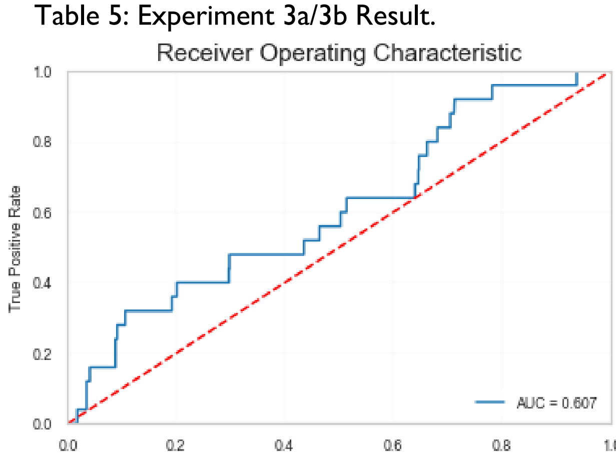


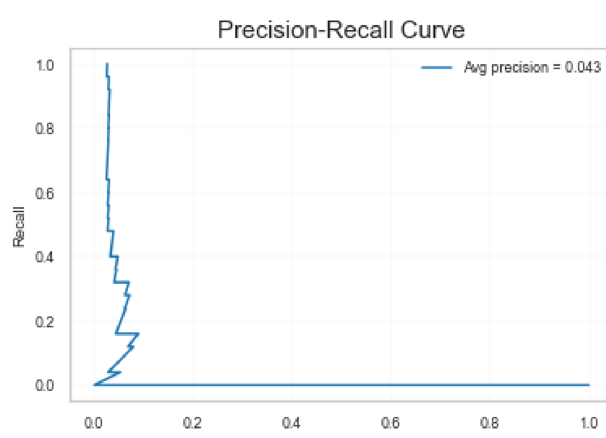


Exp 3a/3b: Artificial Neural Network Classification Model

SAMPLE	ACCURACY	WEIGHTED F1-SCORE	MACRO F1-SCORE	P-R CURVE	ROC CURVE
Baseline Sample	0.98	0.96	0.49	0.043	0.657
SMOTE Sample	0.91	0.93	0.53	0.043	0.522

Table 5: Experiment 3a/3b Result.





6.0 REFLECTIONS AND CONCLUSION

What this project intended to show was how to approach the influence of an imbalanced dataset in classification problems for credit card default predictions. However, it can be inferred that the results didn't show a true representation of high performance as intended. And this is justified by the high degree of imbalance in the data. The macro f1-score showed average performance across the two-class predictions. Thus, demonstrated the models put all input into one class – the majority class.

The dataset was properly cleaned and preprocessed. To reduce training time, a resampling was carried out, this was also to reduce the adverse effect of imbalance in the data. The data were scaled using StandardScaler. The normal dataset samples proved to be quite good on the models, with less impact by SMOTE techniques and PCA feature selection. GridSearchCV was used to get the best-fit parameters on each of the models, which was evident in the model evaluation of 98% for at least 4 out of 5 models, with the least at 97%.

In conclusion, just as in the case of Alam, T.M. et al., the models could be trained and tested under the same procedure on entirely new separate datasets to critically analyze the differences and how the origin of the dataset could be the source of the problem [4]. Nevertheless, this project was limited by a lack of time to experiment on other similar datasets.

7.0 REFERENCES

[1] Y. Sayjadah *et al.* (2018) 'Credit Card Default Prediction using Machine Learning Techniques', - *2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA)*. DOI: 10.1109/ICACCAF.2018.8776802.

[2] M. A. Ullah *et al.* (2018) 'Predicting Default Payment of Credit Card Users: Applying Data Mining Techniques', - *2018 International Conference on Innovations in Science, Engineering and Technology (ICISET)*. DOI: 10.1109/ICISET.2018.8745571

[3] Yu, Y. (2020) 'The application of machine learning algorithms in credit card default prediction', *2020 International Conference on Computing and Data Science (CDS).* IEEE.

[4] Alam, T.M. et al. (2020) 'An investigation of credit card default prediction in the imbalanced datasets', IEEE Access, 8, pp. 201173-201198.

[5] A. Bačová and F. Babič (2021) 'Predictive Analytics for Default of Credit Card Clients', - *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. DOI: 10.1109/SAMI50585.2021.9378671

[6] https://statistics.laerd.com/statistical-guides/standard-score.php#:~:text=The%20standard%20score%20(more%20commonly,are%20from%20different%20normal%20distributions.

[7] https://towardsdatascience.com/data-normalization-with-pandas-and-scikit-learn-7c1cc6ed6475

[8] Teng, H. and 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), pp. 1950021

[9] Butaru, F. *et al.* (2016) 'Risk and risk management in the credit card industry', *Journal of Banking & Finance,* 72, pp. 218-239

[10] Chehrazi, N. and Weber, T.A. (2015) 'Dynamic Valuation of Delinquent Credit-Card Accounts', *Management Science,* 61(12), pp. 3077-3096.

[11] Ebiaredoh-Mienye, S.A., Esenogho, E. and Swart, T.G. (2021) 'Artificial neural network technique for improving prediction of credit card default: A stacked sparse autoencoder approach', *International Journal of Electrical and Computer Engineering (Malacca, Malacca),* 11(5), pp. 4392.

[12] Faraj, A.A., Mahmud, D.A. and Rashid, B.N. (2021) 'Comparison of Different Ensemble Methods in Credit Card Default Prediction', *UHD Journal of Science and Technology,* 5(2), pp. 20-25.

[13] Gao, J., Sun, W. and Sui, X. (2021) 'Research on Default Prediction for Credit Card Users Based on XGBoost-LSTM Model', *Discrete Dynamics in Nature and Society,* 2021, pp. 1-13.

[14] Leow, M. and Crook, J. (2016) 'The stability of survival model parameter estimates for predicting the probability of default: Empirical evidence over the credit crisis', *European Journal of Operational Research,* 249(2), pp. 457-464.

[15] Mesri, K., Tahseen, I. and Ogla, R. (2021) 'Default on a credit prediction using decision tree and ensemble learning techniques', *Journal of Physics: Conference Series,* 1999(1), pp. 12121.

[16] Sun, T. and Vasarhelyi, M.A. (2018) 'Predicting credit card delinquencies: An application of deep neural networks', *Intelligent Systems in Accounting, Finance & Management,* 25(4), pp. 174-189.

[17] Teng, H. and 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), pp. 1950021

[18] W. A. Chishti and S. M. Awan (2019) 'Deep Neural Network a Step by Step Approach to Classify Credit Card Default Customer', *- 2019 International Conference on Innovative Computing (ICIC).* DOI: 10.1109/ICIC48496.2019.8966723.

[19] Yontar, M., Namli, ÖH. and Yanik, S. (2020) 'Using machine learning techniques to develop prediction models for detecting unpaid credit card customers', *Journal of Intelligent & Fuzzy Systems,* , pp. 1-15.

[20] A. Bindal and S. Chaurasia (2018) 'Predictive Risk Analysis For Loan Repayment of Credit Card Clients', *- 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT).* DOI: 10.1109/RTEICT42901.2018.9012366