NATIONAL ECONOMICS UNIVERSITY
FACULTY OF MATHEMATICAL ECONOMICS

REPORT ON PREDICTING DEFAULT OF CREDIT CARD CLIENTS

Student: Kieu Thanh Tam

Student ID: 11219286

Class: DSEB 63

Course name: Risk Analytics

Hanoi, April 14, 2024

INTRODUCTION

1. Background of the Study

The study on predicting credit card client defaults involves comprehensive research and analysis leveraging the knowledge acquired from the Risk Analytics course with the aim of developing accurate models or algorithms capable of forecasting the likelihood of credit card users failing to make their payments. This area of study primarily focuses on evaluating individuals' creditworthiness and reducing risk for credit card issuers and lenders.

The background of the study mainly involves recognizing the importance of credit default prediction in effectively managing risk within consumer lending operations. By predicting which customers are at the highest risk of defaulting on their credit card accounts, issuers can take proactive steps to minimize risk and exposure, leading to a better customer experience and sound business economics.

2. Objectives of the Study

The study aims to develop a robust model to assist commercial banks in assessing the likelihood of customer default, effectively segmenting customers based on their default probability, and classifying the signals influencing default. By leveraging customer signals and information, the study seeks to enable credit institutions to devise tailored lending and investment strategies.

The specific goal include:

- Constructing a highly applicable model for predicting default probability.
- Evaluating and analyzing the impact of various signals on customer default.
- Formulating actionable solutions and recommendations for credit institutions' lending strategies tailored to each customer segment based on default probability.

3. Scope of the Study

The scope of the study in predicting default of credit card clients involves examining various factors and models to develop accurate predictions. The study aims to analyze the relationship between input features, such as credit amount, gender, education, marital status, age, and payment history, and the occurrence of default in credit card clients.

THEORICAL BACKGROUND

I. Theory of the techniques

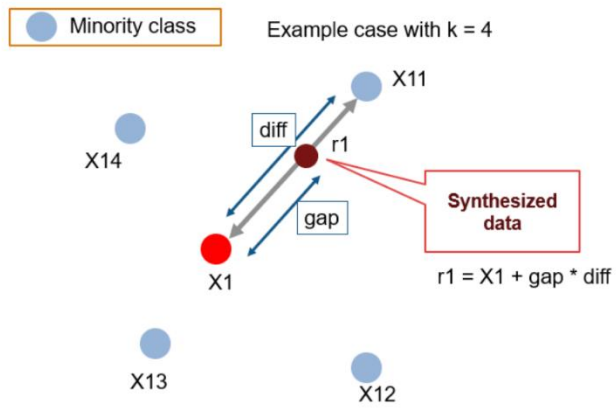
1. Model sampling.

1.1 SMOTE (Synthetic Minority Oversampling Technique)

SMOTE stands for Synthetic Minority Over-sampling Technique, a popular data augmentation technique used in machine learning, specifically in the field of imbalanced classification. This algorithm helps to address the overfitting problem posed by random oversampling. The main goal of SMOTE is to create synthetic samples of the minority class by interpolating between existing minority class samples. This helps to balance the class distribution and improve the performance of machine learning models in predicting the minority class.

Working Procedure

- The total number of oversampled observations, N, is set initially. In general, the binary class distribution is selected to be 1:1. However, this can be changed depending on the situation. The iterative process then begins by selecting an active class instance at random.
- The KNNs for that instance (by default 5) are then obtained. Finally, to interpolate the new composite individuals, N of these K people are chosen.
- The KNNs for that instance (by default 5) are then obtained. Finally, N people are chosen from among these K to interpolate the new composite individuals. The difference in distance between the feature vector and its neighbors will be determined using any distance measure.
- This difference is now multiplied by any random value in the range (0,1) and added to the feature vector. The illustration below depicts this:



Drawbacks: The main limitation of SMOTE is that it can introduce noise into the data through synthetic instances, especially if the number of nearest neighbors is set too high. Moreover, SMOTE may not be effective in scenarios where the minority class instances are tightly clustered or when there are only a few instances in the minority class.

Figure 1: The difference in distance between the feature vector and its neighborhoods

1.2 UnderSampling and Oversampling

Under sampling refers to the process of reducing the number of observations of the majority group to the same number as the minority group. Under sampling has the advantage of being able to make sample balance quickly and readily without the use of a simulation algorithm. However, it has the disadvantage of reducing the sample size greatly.

Oversampling is a data augmentation technique utilized to address class imbalance problems in which one class significantly outnumbers the others. It aims to rebalance training data distribution by amplifying the volume of instances that belong to the under-represented class.

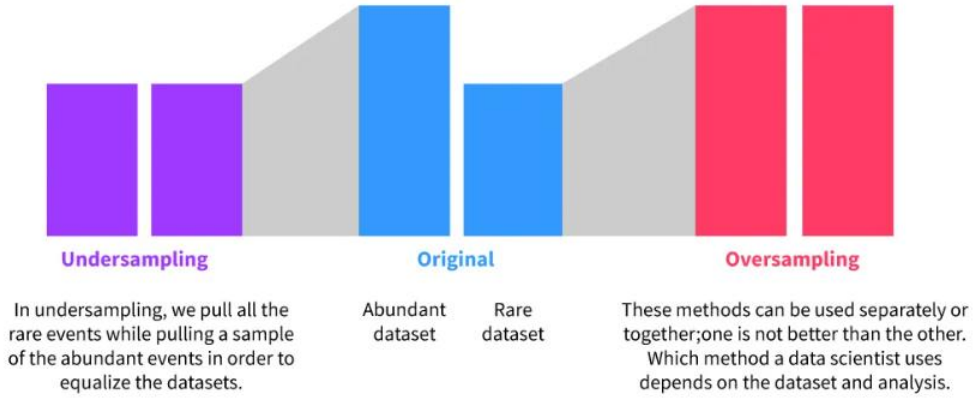


Figure 2: Resembling techniques

2. Evaluation metrics

2.1 Accuracy, Precision, Recall and F1-score

Evaluation metrics for classification compare the predicted class with the actual class of the samples. To facilitate this analysis, a confusion matrix is commonly used, where each element represents the count of instances where the actual class is one value and the predicted class is another. In the case of binary classification, such as our scenario, the confusion matrix becomes simpler to construct and interpret. If the two classes being predicted are labeled as True and False, the confusion matrix can be represented as shown in Table 1.

	Predicted (1)	**Predicted (0)**
Actual (1)	TP	FN
Actual (0)	FP	TN

Table 1: Binary classification confusion matrix

The confusion matrix consists of four basic characteristics (numbers) that are used to define the measurement metrics of the classifier. These four numbers are:

1. TP (True Positive): TP represents the number of patients who have been properly classified to have malignant nodes, meaning they have the disease.

2.TN (True Negative): TN represents the number of correctly classified patients who are healthy.

3.FP (False Positive): FP represents the number of misclassified patients with the disease but actually they are healthy. FP is also known as a *Type I error*.

4.FN (False Negative): FN represents the number of patients misclassified as healthy but actually they are suffering from the disease. FN is also known as a *Type II error*.

Different metrics can be used depending on the task, the data imbalance and other factors. While dealing with classification tasks, these are some of the most used ones.

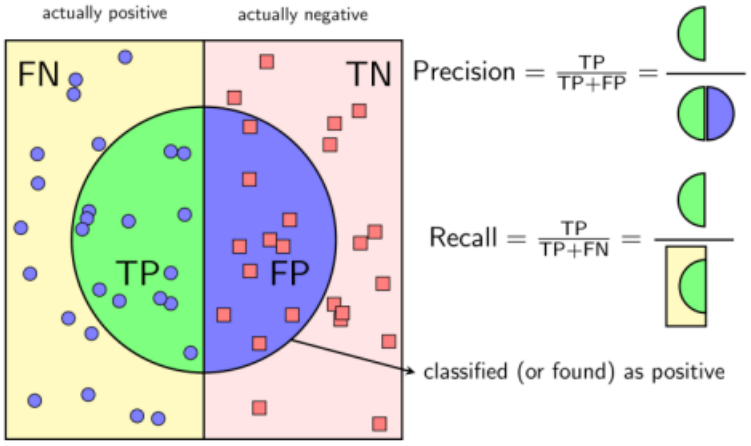


Figure 3: Precision and Recall

1. Accuracy: measures how often a model correctly predicts the outcome. In other words, accuracy calculates the proportion of correct predictions made by the model out of the total number of predictions

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

2. Precision: measures he accuracy of a model's predictions for the positive class. It quantifies how often the model correctly predicts instances belonging to the positive class out of all instances it predicted as positive.

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

3. Recall: measures the model's effectiveness in capturing and identifying the true positives among all the positive instances

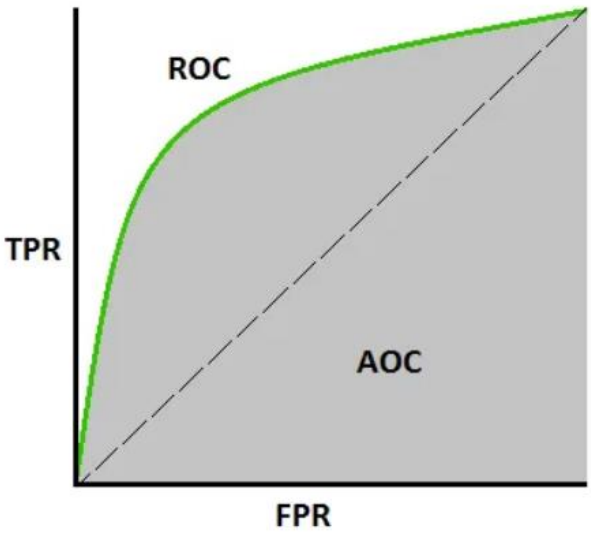
$$Recall = \frac{TP}{TP + FN}$$

4. F1 – score: is a metric used to evaluate the performance of a model in machine learning. Unlike accuracy, which treats all classes equally, the F1-Score considers the importance of imbalanced datasets, where the rare class is more significant than the majority class..

The F1-Score combines precision and recall to provide a balanced measure of a model's performance. It represents the harmonic mean between precision and recall, ensuring that both metrics are taken into account. A higher F1-Score indicates a better balance between precision and recall, indicating a more reliable model performance.

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

2.2 ROC – AUC curve



The AUC-ROC curve is a performance evaluation tool used in classification problems. It measures the model's performance at different threshold settings for classifying instances. The ROC (Receiver Operating Characteristic) curve is a graphical representation of the model's ability to distinguish between classes. The AUC (Area Under the Curve) represents the degree of separability, indicating how well the model can differentiate between the two classes. A higher AUC indicates that the model is better at correctly predicting instances of class 0 as 0 and instances of class 1 as 1. The ROC curve is plotted by comparing the True Positive Rate (TPR) against the False Positive Rate (FPR), where TPR is on the y-axis and FPR is on the x-axis..

Figure 4: Defining terms used in AUC and ROC Curve

5. TPR (True Positive Rate) / Recall /Sensitivity

$$TPR\ (True\ Positive\ Rate)\ /\ Recall\ /Sensitivity = \frac{TP}{TP+FN}$$

6. Specificity

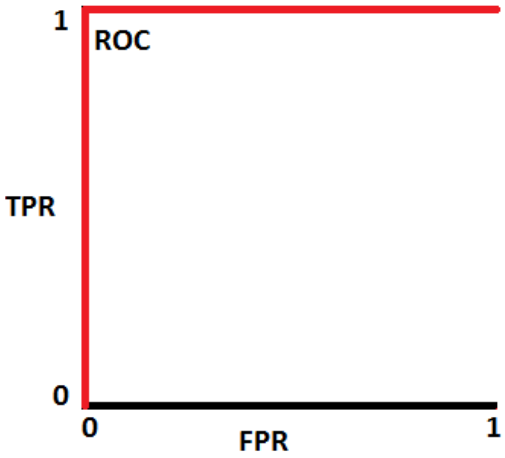
$$Specificity = \frac{TN}{TN + FP}$$

7. FPR

$$FPR = 1 - Specificity$$

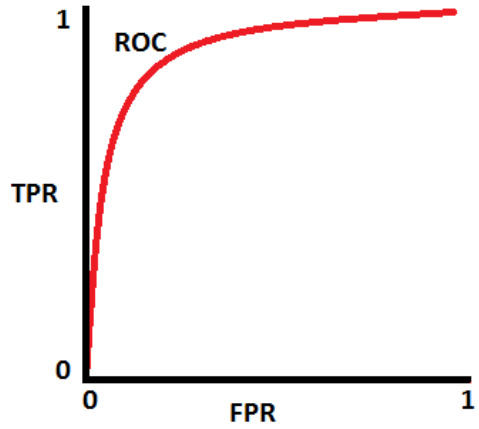
$$= \frac{FP}{TN + FP}$$

An excellent model has AUC been closer to the 1 indicating a high measure of separability. A poor model has an AUC close 0 signifying that it has the lowest measure of separability. In fact, this means that the model's predictions are completely inverted, where it wrongly identifies class 0 as class 1 and class 1 as class 0. When the AUC value is 0.5, it indicates that the model lacks any ability to separate the classes.



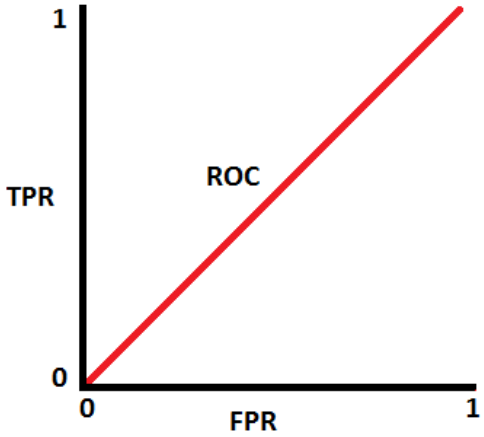
When two curves don't overlap at all means model has an ideal measure of separability. It is perfectly able to distinguish between positive class and negative class.

Figure 5: The classifier when AUC =1



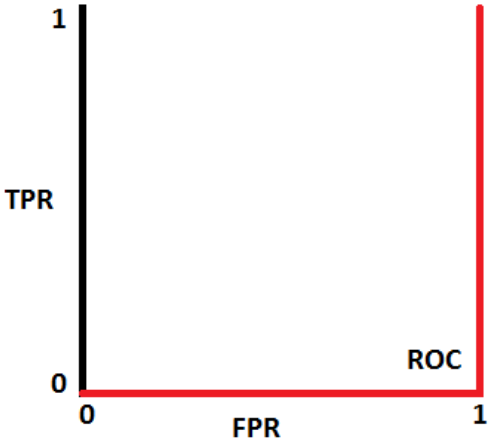
When two distributions overlap, we introduce type 1 and type 2 errors. Depending upon the threshold, we can minimize or maximize them. When AUC is 0.7, it means there is a 70% chance that the model will be able to distinguish between positive class and negative class.

Figure 6: The classifier when 0.5 < AUC < 1



This is the worst situation. When AUC is approximately 0.5, the model has no discrimination capacity to distinguish between positive class and negative class.

Figure 7: The classifier when AUC = 0.5



When AUC is approximately 0, the model is reciprocating the classes. It means the model is predicting a negative class as a positive class and vice versa.

Figure 8: The classifier when AUC = 0

II. Theory of the model Machine Learning

1. Logistic Regression

Logistic regression is a powerful supervised machine learning algorithm commonly employed for binary classification problems where the target variable is categorical. It can be seen as a variant of linear regression specifically tailored for classification tasks. In logistic regression, a logistic function is utilized to model the relationship between the input variables and a binary output variable. This logistic function constrains the predicted values to fall within the range of 0 and 1.

The main distinction between linear regression and logistic regression lies in their output ranges. While linear regression produces continuous values, logistic regression produces probabilities between

0 and 1. Moreover, logistic regression does not assume a linear relationship between the input and output variables, allowing it to capture non-linear patterns and better handle complex classification problems.

$$Logistic\ function = \frac{1}{1 + e^{-x}}$$

The logistic function will always produce an S-shaped curve between 0 and 1, as shown in Figure 9.

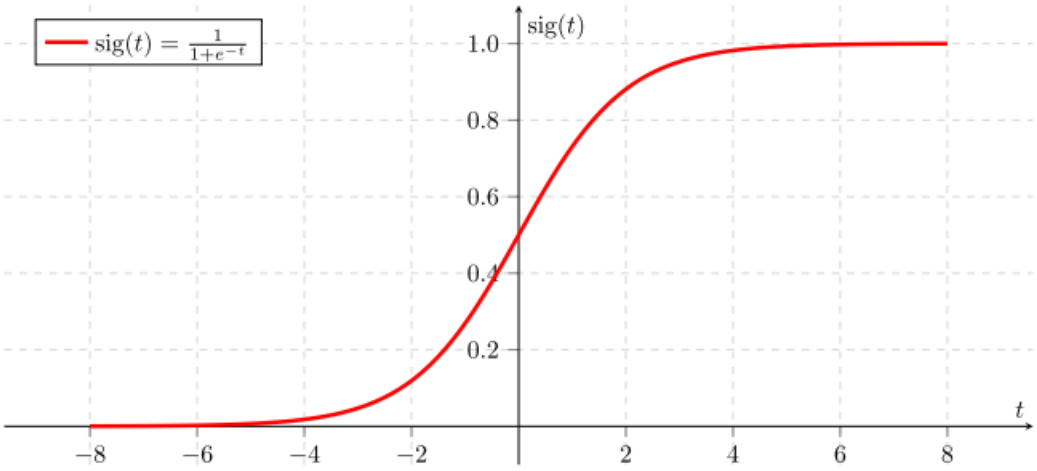


Figure 19: Sigmoid activation function

As opposed to linear regression where MSE or RMSE is used as the loss function, logistic regression uses a loss function referred to as "maximum likelihood estimation (MLE)" which is a conditional probability. If the probability is greater than 0.5, the predictions will be classified as class 0. Otherwise, class 1 will be assigned. Before going through logistic regression derivation, let's first define the logit function. Logit function is defined as the natural log of the odds. A probability of 0.5 corresponds to a logit of 0, probabilities smaller than 0.5 correspond to negative logit values, and probabilities greater than 0.5 correspond to positive logit values. It is important to note that as illustrated in Fig. 5.17, logistic function ranges between 0 and 1 (P∈ [0,1]) while logit function can be any real number from minus infinity to positive infinity (P∈ [−∞, ∞]).

$$odds = \frac{P}{1-P} \rightarrow logit(P) = \ln\left(\frac{P}{1-P}\right)$$

Let's set logit of P to be equal to mx +b, therefore:

$$logit(P) = mx + b \rightarrow\ mx + b = \ln(\ \frac{P}{1-P})$$

$$\left(\frac{P}{1-P}\right) = e^{mx+b} \rightarrow P = \frac{e^{mx+b}}{1 + e^{mx+b}} \rightarrow P = \frac{1}{1 + e^{-(mx+b)}}$$

2. Decision Tree

A decision tree is a supervised machine learning algorithm used for categorization or prediction based on a set of questions and their corresponding answers. It consists of three main parts: decision nodes, chance nodes, and end nodes. Decision nodes represent choices, chance nodes represent probabilities, and end nodes represent outcomes.

The decision tree works by splitting nodes into multiple sub-nodes based on certain conditions or attributes. Each node represents a question or split point, and the leaf nodes represent the possible answers or outcomes. The tree is constructed by selecting attributes and conditions that will produce the tree, and it can be pruned to remove irrelevant branches that may affect accuracy.

There are two main types of decision trees: categorical and continuous. Categorical decision trees classify data into distinct categories, while continuous decision trees predict outcomes based on multiple variables.

Decision trees have various applications, including customer recommendation engines and identifying risk factors for depression. They provide a simple view of complex processes and can easily map nonlinear relationships. However, they may not always provide clear-cut answers and require careful pruning to avoid overfitting.

Gini impurity (IG) and entropy (IH) are the most commonly used splitting criteria in binary decision trees. Defining as $p(i|t)$ the proportion of the examples that belong to class i for a particular node t, we can write the entropy as:

$$Gini\ Impurity: I_G(t) = -\sum_{i=1}^{c} p(i|t)(1 - p(i|t))$$

$$Entropy: -\sum_{i=1}^{c} p(i|t) \log_2 p(i|t)$$

3. Random Forest

Random forest is a machine learning algorithm that combines the predictions of multiple decision trees to make more accurate predictions. It is widely used due to its simplicity, flexibility, and ability to handle both classification and regression tasks. It operates in these steps:

STEP 1: Randomly select *k* features from the total m features, where $k \ll m$

STEP 2: Among the "*k*" features, calculate the node "*d*" using the best split point.

STEP 3: Split the node into daughter nodes using the best split.

STEP 4: Repeat 1 to 3 steps until the "*p*" number of nodes has been reached.

STEP 5: Build the forest by repeating steps 1 to 4 for "*n*" number of times to create "*n*" number of trees.

Random Forest classifier employs bagging techniques, utilizing decision tree classifiers as its base learners. A Random Forest comprises multiple trees, where each tree independently generates its own classification prediction. The final decision of the model is determined by aggregating the predictions from all the trees and selecting the class that receives the highest number of votes

Pros:

- Diversity: Each decision tree in the Random Forest considers only a subset of features, leading to diverse models. This diversity helps to improve the overall accuracy and robustness of the model
- Overfitting Prevention: Random Forest effectively mitigates the problem of overfitting, which occurs when a model performs well on the training data but poorly on unseen data. By using multiple decision trees and aggregating their predictions, Random Forest reduces the risk of overfitting
- Parallelization: The construction of each decision tree in Random Forest can be done independently, allowing for parallelization. This means that Random Forest can take advantage of multi-core processors and significantly speed up the training process
- Stability: Random Forest's predictions are based on majority voting or averaging, which adds stability to the model. Even if some decision trees make incorrect predictions, the majority of trees can still provide accurate results

4. Support Vector Machine

Support Vector Machines (SVMs) are considered among the most effective classification algorithms in modern machine learning [19]. When used for classification tasks, SVMs are supervised learning methods that construct an hyperplane that maximizes the margin between two classes in the feature space.

A hyperplane in a space H endowed with a dot product $\langle \cdot, \cdot \rangle$ is described by the set:

$\{x \in \mathrm{H} | < w, x > + b = 0$ where w ∈ H and b ∈ R

Such a hyperplane naturally divides H into two half-spaces and hence can be used as the decision boundary of a binary classifier: $\{x \in \mathrm{H} | < w, x > + b \geq 0$ and $\{x \in \mathrm{H} | < w, x > + b \leq 0$

Given a set $X = [x_1, \ldots, x_m]$ the margin is the distance of the closest point in X to the hyperplane:

$$min_{i=1,\ldots,m} \frac{|< w, x_i >|}{||w||}$$

Since the parametrization of the hyperplane is not unique, we set

$$min_{i=1,\ldots,m} |< w, x_i > + b| = 1$$

the margin simply becomes $\frac{1}{||w||}$

Let $S = [(x_1, y_1), \ldots, (x_m, y_m)]$ be a training set of examples, where each $x_i \in H$ and $y_i \in \{\pm 1\}$. Our aim is to find a linear decision boundary parameterized by (w, b) such that $\langle w, x_i \rangle + b \geq 0$ whenever $y_i = +1$ and $\langle w, x_i \rangle + b < 0$ whenever $y_i = -1$. The SVM solution is the separating hyperplane with the maximum geometric margin, as it is the safest choice. The problem of maximizing the margin can be written as:

$$min_{w,b,\xi} \frac{1}{2} ||w||^2$$
$$s.t. y_i(< w, x_i > + b) \geq 1 \; \forall i$$

5. Gradient Boosting

Gradient boosting is an ensemble technique in machine learning that combines the predictions of multiple weak learners, such as decision trees, in a sequential manner. Its goal is to enhance the overall predictive performance by iteratively optimizing the weights of the models based on the errors from previous iterations. This iterative process gradually reduces prediction errors and improves the accuracy of the model.

Step 1: Initialize model with a constant value.

$F_0(x) = argmin \sum_{i=1}^{n} L(y_i, \gamma)$

Step2: For m = 1 to M:The whole step2 processes from 2–1 to 2–4 are iterated *M* times. *M* denotes the number of trees we are creating, and the small *m* represents the index of each tree.

Step 2–1: Compute residuals $r_{im} = -\left[\frac{\partial L(y_i,\ F(x_i))}{\partial\ F(x_i)}\right]_{F(x)=F_{m-1}(x)}$ $for\ i = 1, \dots, n$

Step 2–2: Train regression tree with features x against γ and create terminal node reasons R_{jm} for j = $1,\dots, J_m$

Step 2-3: Compute $\gamma_{im} = argmin\ \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)\ for\ j = 1, \dots \dots, J_m$

Step 2- 4: Update the model.

$$F_m(x) = F_{m-1}(x) + v \sum_{j=1}^{J_m} \gamma_{jm} 1\left(x \in R_{jm}\right)$$

6. XG Boosting

XGBoost is an ensemble learning algorithm that falls under the gradient boosting framework in machine learning. It utilizes decision trees as base learners and incorporates regularization techniques to improve the generalization of the model. XGBoost is well-known for its computational efficiency, ability to analyze feature importance, and its capability to handle missing values. It is widely used for various tasks including regression, classification, and ranking.

Step 1: Initialize the model: The algorithm starts by initializing the model with an initial prediction. This initial prediction is usually a simple average or a constant value.

Step 2: Calculate the residuals: The algorithm calculates the difference between the actual target values and the predicted values from the previous step. These differences are called residuals.

Step 3: Build a weak learner: XGBoost uses decision trees as weak learners. A decision tree is built to predict the residuals from the previous step. The tree is constructed by recursively splitting the data based on certain features and thresholds.

Step 4: Update the model: The predictions from the weak learner are combined with the previous predictions to update the model. The update is done by adding a fraction of the predictions from the weak learner to the previous predictions.

Step 5: Repeat steps 2-4: Steps 2-4 are repeated iteratively to build multiple weak learners and update the model. Each new weak learner focuses on correcting the errors made by the previous weak learners

Step 6: Regularization: XGBoost incorporates regularization techniques to prevent overfitting. Regularization helps in controlling the complexity of the model and improves its generalization ability

Step 7: The final prediction is obtained by combining the predictions from all the weak learners. The predictions from each weak learner are weighted based on their performance and contribution to the overall model.

DATA

1. Dataset description

The Default of Credit Card Clients dataset contains 30 000 instances of credit card status collected in Taiwan from April 2005 to September 2005. The dataset employs the binary variable default payment next month as response variable. It indicates if the credit card holders will be defaulters next month (Yes = 1, No = 0). In detail, for each record (namely, each client) we have demographic information, credit data, history of payments and bill statements. To be more precise, the following is the complete list of all the 23 predictors.

- Client personal information

  1. ID: ID of each client

  2. LIMIT_BAL: Amount of given credit (in New Taiwan dollars): it includes both the individual consumer credit and his/her family (supplementary) credit

  3. SEX: 1 = male, 2 = female

  4. EDUCATION: 1 = graduate school; 2 = university; 3 = high school; 4 = others.

  5. MARRIAGE: Marital status, 1 = married; 2 = single; 3 = others.

  6. AGE: Age in years.

- History of past payments from April to September 2005, i.e., the delay of the past payment referred to a specific month:

  7. PAY_0: Repayment status in September, 2005. (scale same as above)

  8. PAY_2: Repayment status in August, 2005. (scale same as above)

  9. PAY_3: Repayment status in July, 2005. (scale same as above)

  10. PAY_4: Repayment status in June, 2005. (scale same as above)

  11. PAY_5: Repayment status in May, 2005. (scale same as above)

  12. PAY_6: Repayment status in April, 2005. (scale same as above)

The measurement scale for the repayment status is: -1 = pay duly; 1 = payment delay for one month; 2 = payment delay for two months; ...; 8 = payment delay for eight months; 9 = payment delay for nine months and above.

- Amount of bill statement (in New Taiwan dollars), i.e. a monthly report that credit card companies issue to credit card holders in a specific month:

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

  14. BILL_AMT2: Amount of bill statement in August, 2005. (NT dollar)

  15. BILL_AMT3: Amount of bill statement in July, 2005. (NT dollar)

  16. BILL_AMT4: Amount of bill statement in June, 2005. (NT dollar)

  17. BILL_AMT5: Amount of bill statement in May, 2005. (NT dollar)

  18. BILL_AMT6: Amount of bill statement in April, 2005. (NT dollar)

- Amount of previous payment (in New Taiwan dollars):

  19. PAY_AMT1: Amount of previous payment in September, 2005. (NT dollar)

  20. PAY_AMT2: Amount of previous payment in August, 2005. (NT dollar)

  21. PAY_AMT3: Amount of previous payment in July, 2005. (NT dollar)

  22. PAY_AMT4: Amount of previous payment in June, 2005. (NT dollar)

  23. PAY_AMT5: Amount of previous payment in May, 2005. (NT dollar)

  24. PAY_AMT6: Amount of previous payment in April, 2005. (NT dollar)

  25. default.payment.next.month: Default payment (1=yes, 0=no)

In figure 10, we can understand what the data looks like. The target in the dataset is column default.payment.next.month.

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default.payment.next.month
0	1	20000.0	2	2	1	24	2	2	-1	-1	...	0.0	0.0	0.0	0.0	689.0	0.0	0.0	0.0	0.0	1
1	2	120000.0	2	2	2	26	-1	2	0	0	...	3272.0	3455.0	3261.0	0.0	1000.0	1000.0	1000.0	0.0	2000.0	1
2	3	90000.0	2	2	2	34	0	0	0	0	...	14331.0	14948.0	15549.0	1518.0	1500.0	1000.0	1000.0	1000.0	5000.0	0
3	4	50000.0	2	2	1	37	0	0	0	0	...	28314.0	28959.0	29547.0	2000.0	2019.0	1200.0	1100.0	1069.0	1000.0	0
4	5	50000.0	1	2	1	57	-1	0	-1	0	...	20940.0	19146.0	19131.0	2000.0	36681.0	10000.0	9000.0	689.0	679.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29995	29996	220000.0	1	3	1	39	0	0	0	0	...	88004.0	31237.0	15980.0	8500.0	20000.0	5003.0	3047.0	5000.0	1000.0	0
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-1	...	8979.0	5190.0	0.0	1837.0	3526.0	8998.0	129.0	0.0	0.0	0
29997	29998	30000.0	1	2	2	37	4	3	2	-1	...	20878.0	20582.0	19357.0	0.0	0.0	22000.0	4200.0	2000.0	3100.0	1
29998	29999	80000.0	1	3	1	41	1	-1	0	0	...	52774.0	11855.0	48944.0	85900.0	3409.0	1178.0	1926.0	52964.0	1804.0	1
29999	30000	50000.0	1	2	1	46	0	0	0	0	...	36535.0	32428.0	15313.0	2078.0	1800.0	1430.0	1000.0	1000.0	1000.0	1

30000 rows × 25 columns

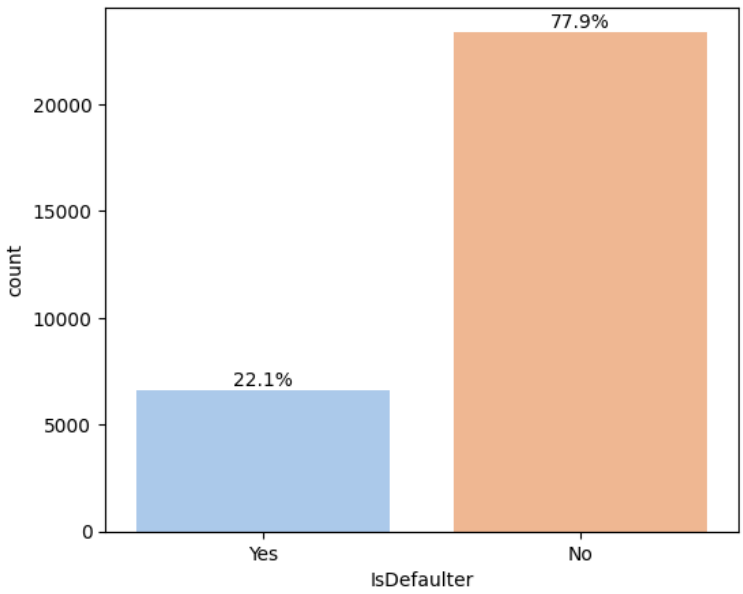
Figure 10: Original dataset from UCI machine learning repository through pandas framework.

2. Data cleaning

So looking at the values present in the attributes some changes have to be done:

- Attribute EDUCATION should present only one of those values: 1,2,3,4; but in the dataset some records have values 0,5,6. Because 0,5,6 we don't have any description so we add up them in group 4: others
- Attribute MARRIAGE should present only one of those values: 1,2,3; but in the dataset some records have value 0. As the same as the EDUCATION, we add up the value 0 to the group 3: others
- Attributes PAY_N should present only one of those values: -1,1,2,3,4,5,6,7,8,9; but in the dataset some records have value -2 and 0.

In first two cases since there is an attribute which represent the Other class (respectively 3 for marriage and 4 for education), all the attributes not-known are mapped in that category. In the last case, in order to use this attributes as a numerical attribute, and not a categorical one, all the values -2 and -1 are mapped in 0. In this way PAY N will indicate for how many months the payment was delayed



For the feature PAY_N, BILL_ATMN, PAY_ATMN, default.payment.next.month we change name of these columns for simplicity and better understanding.

- Attributes PAY_0: PAY_1
- Attributes default.payment.next.month : IsDefaulter

3. Data Analysis

3.1 Target features

From Figure 3 it is possible to see the distribution of the target variable default payment next month. It clearly shows an imbalance towards the 0 class (i.e. no default), with around 78% of the

whole dataset. This imbalance problem will make classification models focusing on the majority class overlooking the minority class if not addressed.

Figure 11: Countplot of IsDefaulter

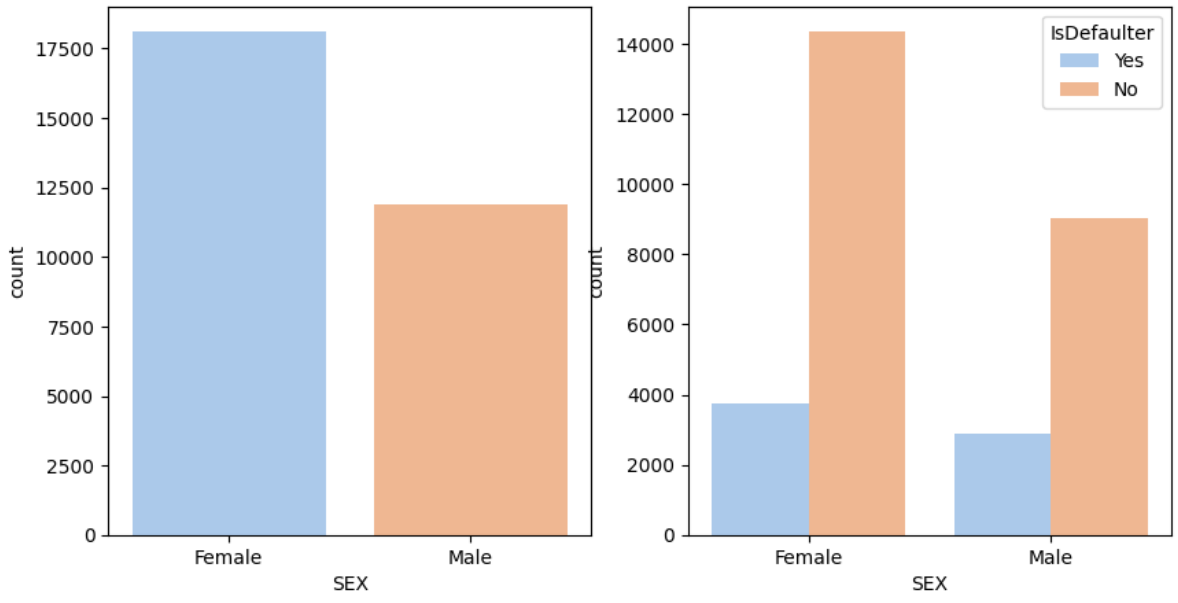
3.2 Categorical features

Regarding the categorical features SEX, EDUCATION and MARRIAGE showed in Figure 4 and counts in Table 1.

a) SEX: There are much more females than males in the dataset. In particular, Males have a slightly higher chance of defaulting compared to females (0.24% vs 0.21%). However, the proportion of defaulters and non-defaulters is consistent across both genders.

a) EDUCATION: The percentage of defaulters and non-defaulters varies across different education levels. Defaulters primarily concentrate among students at university and graduate school, with rates of 50% and 30% respectively, compared to the total number of defaulters. However, when comparing the number of defaulters to the total count within each education level, high school emerges as the predominant category, accounting for 25%, followed closely by university at 24%.

b) MARRIAGE: The number of defaulters in both the Single and Married categories is approximately equal, with counts of 3329 and 3192 respectively. Nevertheless, when considering the ratios, all categories are approximately equal, with a ratio of 21, 24, and 26 respectively.



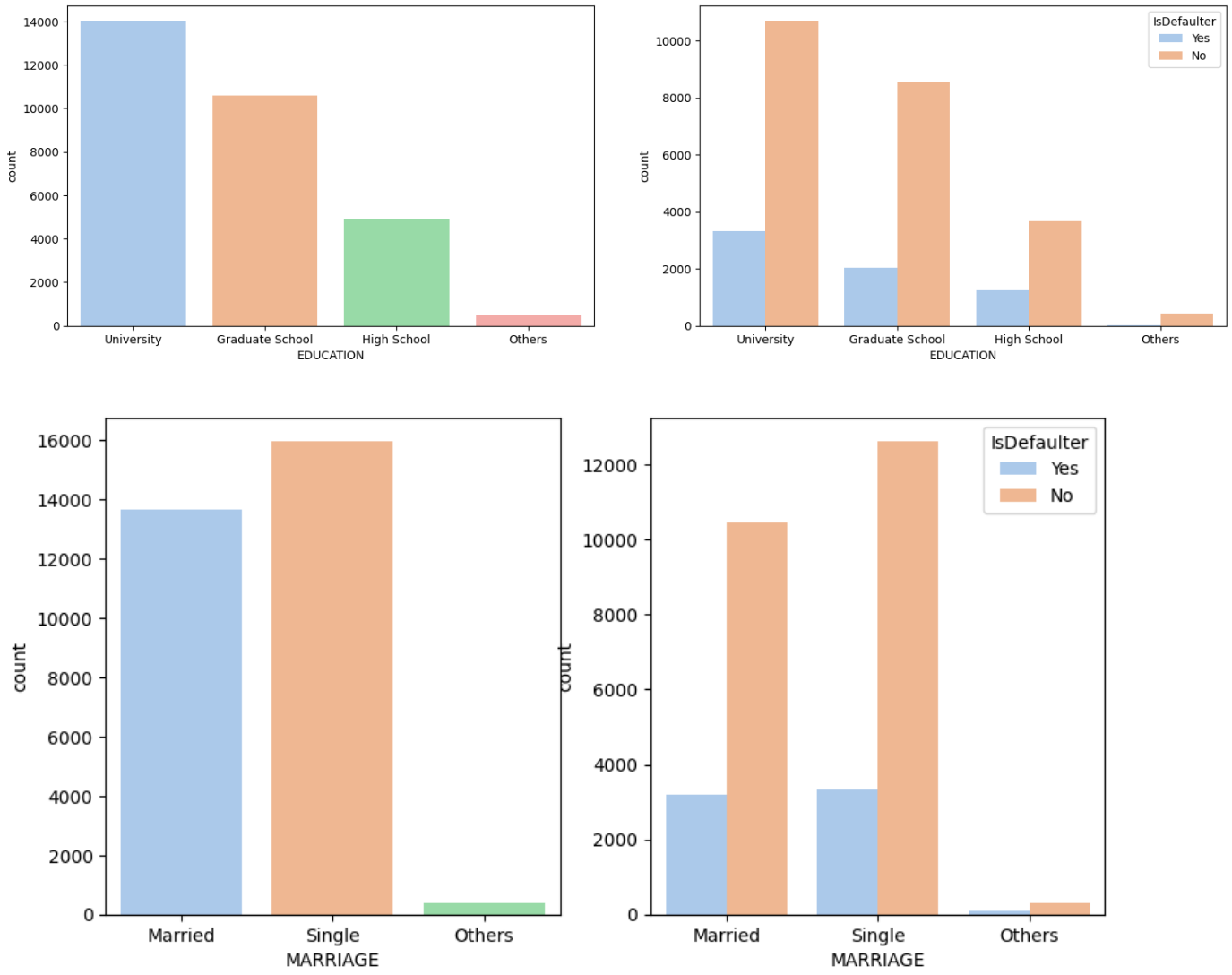


Figure 12: Countplot of SEX, EDUCATION and MARRIAGE grouped by DEFAULT class.

attribute	value	count	defaulters	(%)
SEX	Female	17.855	3.744	20,96%
	Male	11.746	2.861	24,35%
EDUCATION	University	14.024	3.329	23,73%
	Graduate school	10.581	2.036	19,24%
	High school	4.873	1.233	25,30%
	Other	123	7	5,70%
MARRIAGE	Single	15.806	3.329	21,06%
	Married	13.477	3.192	23,68%
	Others	318	84	26,4%

Table 2: Value counts for SEX, EDUCATION and MARRIAGE feature

We still have to inspect the payment status feature PAY_N, boxplots below shown in Figure 5 is very useful. Clients who delay payment by one month or less have fewer credit card defaults. I.e. a greater discriminatory power is held the repayment status in September, PAY 1, than the repayment status in the other months.

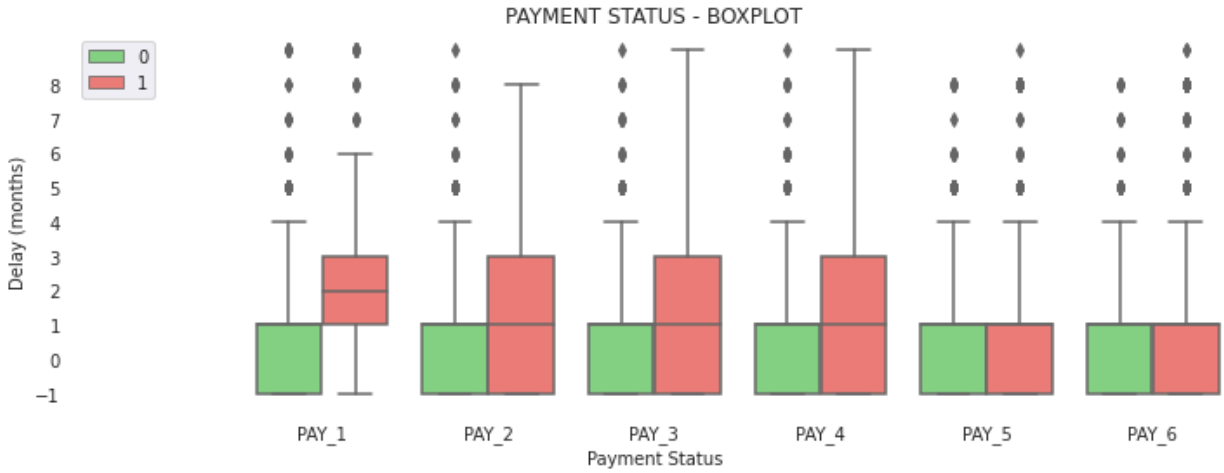


Figure 13: Boxplots of PAY_N grouped by DEFAULT class

3.3 Numerical features

In statistics, the Kernel Density Estimation (KDE) is a fairly well known technique for estimating the probability density function in a non-parametric way (i.e. it does not assume any underlying distribution). So, for the following continuous feature, we explored their KDE plots. Observing Figure 14 it is possible to notice that most of the default come from credits with a lower LIMIT BAL (i.e. credit amount), in particular they are observed in a range among a few thousands Taiwanese dollars to around $140000. The customers above this threshold are more likely to repay their debts.

For the feature AGE, is recognized as a crucial feature of credit card default, with younger individuals often exhibiting a higher propensity for default compared to their older counterparts. This trend may stem from various factors including financial instability, limited experience in managing finances, and a heavier debt burden among younger age groups.

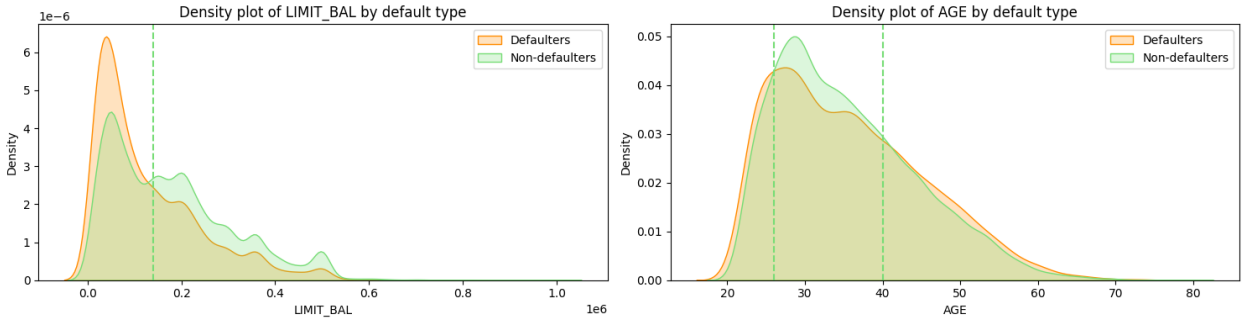


Figure 14: KDE plots of LIMIT BAL and AGE grouped by DEFAULT class

3.4 Other features

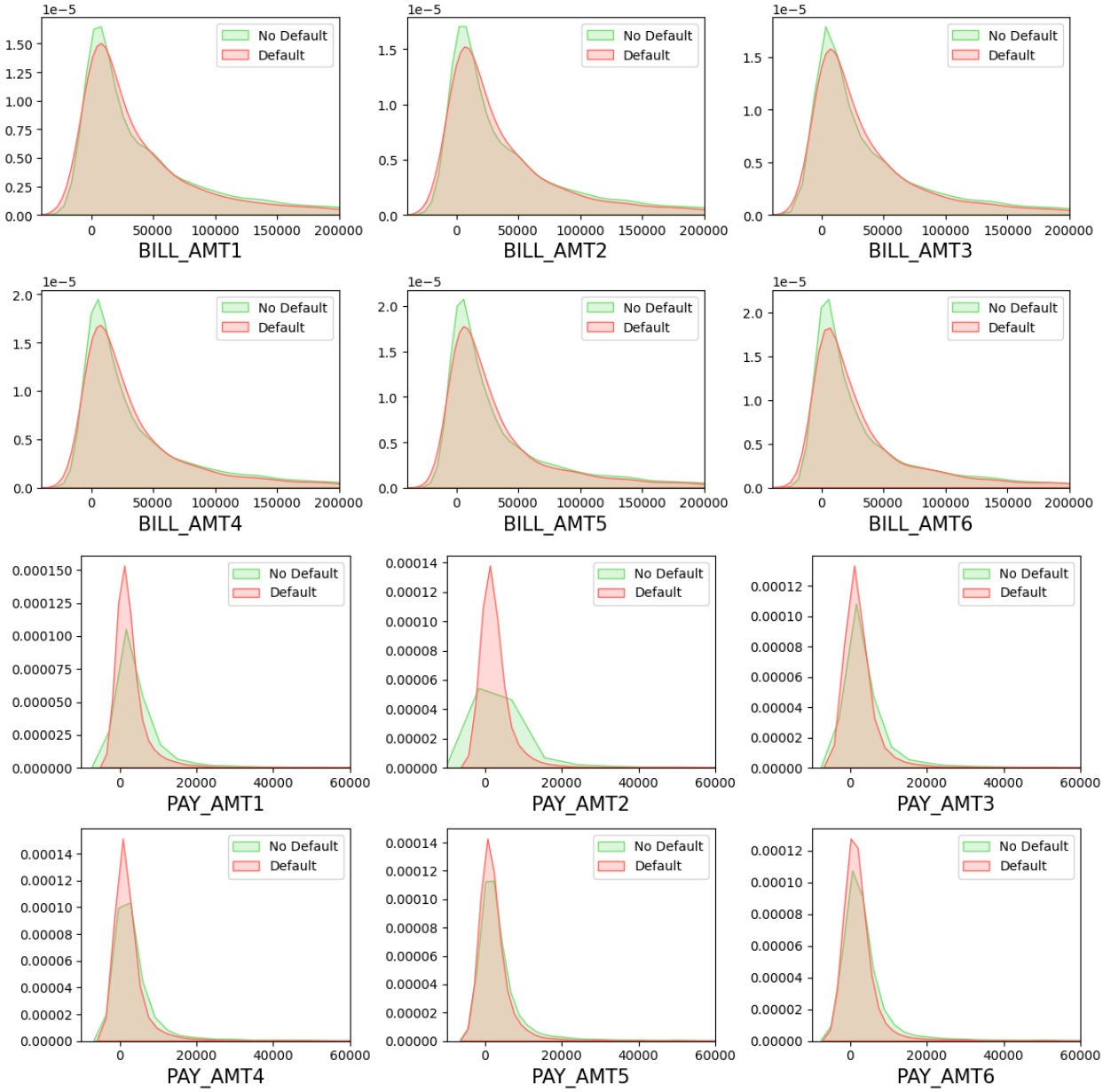


Figure 15: Plots of BILL_ATMn and PAY_ATMs grouped by DEFAULT class

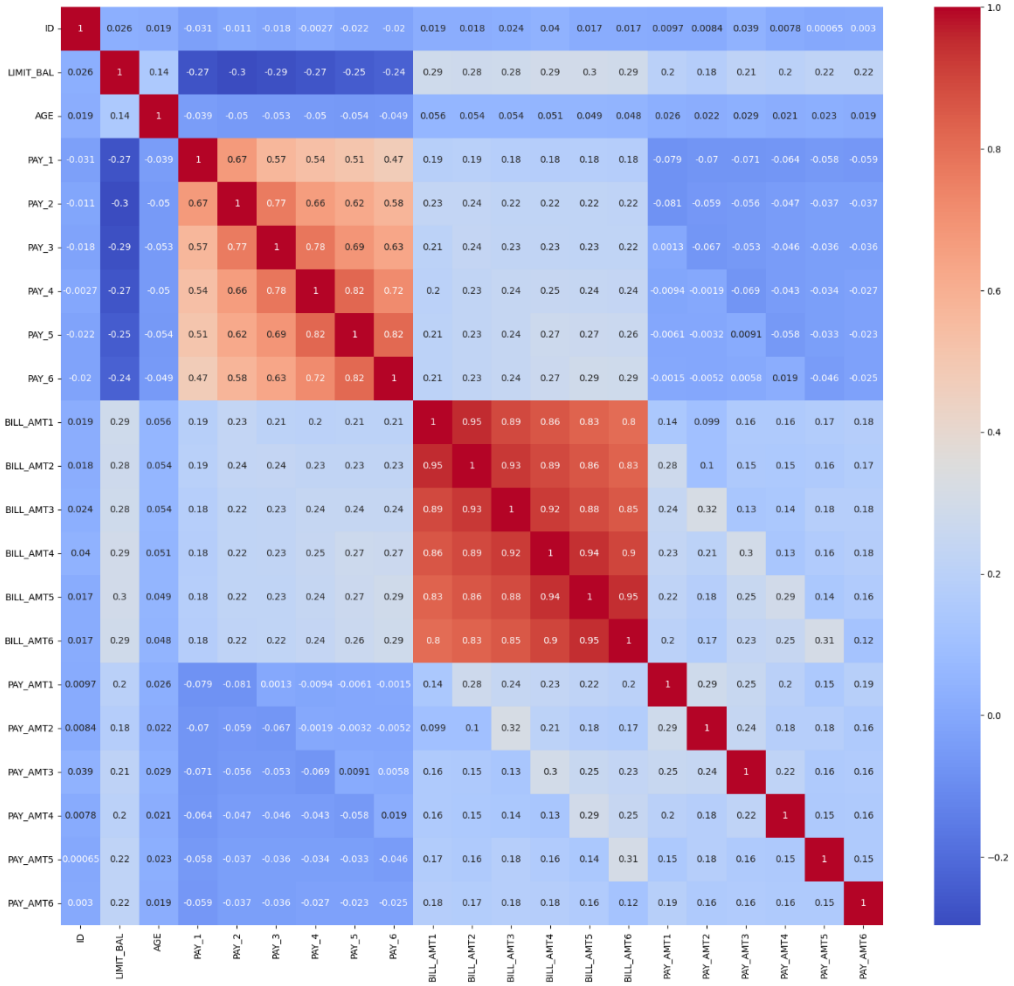
3.5 Correlation

Correlation is a statistical term describing the degree to which two random variables move in coordination with one another. Intuitively, if they are moving in the same direction, then those variables are defined with a positive correlation, or vice versa we define that with a negative correlation. The Pearson Correlation (ρ) is one of the most used linear correlation measures

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \quad (1)$$

The value of Pearson's Correlation Coefficient range is [-1, 1].

- +1 means that they are strongly correlated
- 0 means no correlation, the two random variables are statistically independent, but it is not true the opposite, because they may have a non-linear relationship.
- -1 means that there is a negative correlation (inverse proportion)



High values of this correlation coefficient with respect to the target is a synonym of data redundancy, so it could be helpful to drop those columns. In Figure 7 is given a graphical representation of the Person Correlation with a Heatmap, where each cell (i, j) represents the Person Correlation between the random variables Xi and Xj.

From Figure 16, we can observe an "internal" correlation among the groups of features such as BILL_ATM, PAY_N. Figure 16 : Heatmap correlation

We can also notice that there is no feature with a strong relationship with the target. In fact, there are 15 features with an absolute value of the correlation below than 0.1 and none of the remaining ones have a greater correlation than 0.29.

4. Data Preprocessing

4.1 Outliers and Anomaly detection

Outliers or anomalies refer to observations that deviate significantly from the rest of the data, raising suspicions that they were generated by a different process. There are several common reasons for the presence of outliers:

- An entity may appear different because it belongs to a distinct category or class.
- Occasionally, we may record values that are far from the usual patterns, albeit with a lower probability.
- Technical or human errors can also contribute to the occurrence of outliers.

The presence of a substantial number of outliers can have a significant impact on the performance of models. Therefore, it is a common practice to train models both with and without outliers to assess their influence. Various techniques exist for identifying and removing outliers, including graphical representations like boxplots.

In Figure 8, the boxplots for the variables BILL_ATM and PAY_ATM are presented, along with a description of how outliers were determined for each variable. The BILL ATM variables represent the amount of bill statements during different months. Since these variables can be considered repeated observations of the same variable per cardholder, they were analyzed using the same boxplot for outlier identification. To minimize information loss resulting from outlier removal, a cutoff value was established based on the overall trend observed in all six variables.

For the BILL ATM variables, amounts exceeding 1,000,000 or falling below the minimum value (170,000 in BILL ATM4) were classified as outliers. The same approach was applied to the PAY ATM variables. The upper cutoff point for these variables was determined by the maximum value in PAY_ATM4 (621,000). Any observation surpassing this threshold was considered an outlier. As no observation had a value lower than 0, there were no outliers based on this criterion. After excluding the outlier values, the dataset used for training and validation consisted of 29,993 samples (with 7 observations being dropped).

It should be noted that no samples were eliminated as outliers because the available literature on the dataset did not provide information on this matter, and the domain knowledge was insufficient.

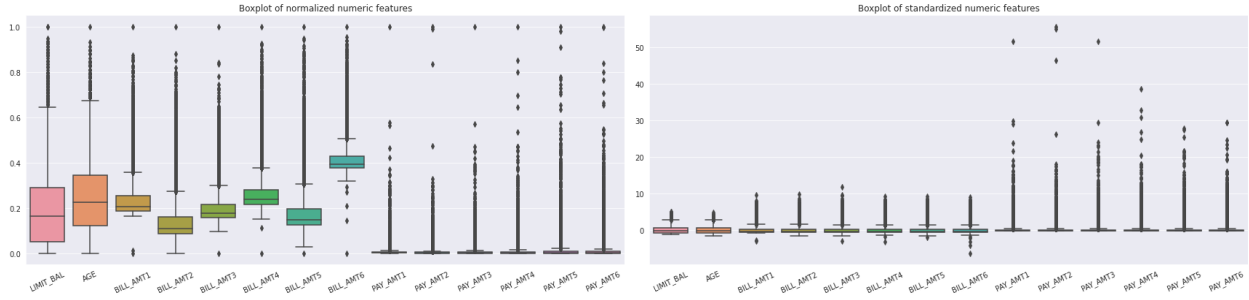


Figure 17: Box-plot for PAY_ATM and BILL_ATM Variables

4.2 Data train-test split

To evaluate the performance of a classification algorithm, is to divide the dataset into two partitions, called training and test set. The training set is used to fit the machine learning model, whereas the test set is used to evaluate the fit machine learning model. In this case, we decide to divide the dataset into 3 ways, 80%,70% and 50% is used for the training procedure and the remaining 20%, 30%, and 50% with respectively for testing.

4.3 Imbalance data

Before inputting the dataset into the training model, it is important to address the issue of imbalanced data. In this study, we use three different ways to handle the data: SMOTE, Undersampling and Oversampling. Each method, when used to train the model, yields different results, which will be presented in the following section.

FINDING AND CONCLUSION

I, Findings

Train-test split with train = 0.8 and test = 0.2

a. SMOTE

		Accuracy		Precision	Recall	F1-score	AUC
		Train	Test				
Logistic Regression	Base Model	0.859	0.863	0.789	0.927	0.852	0.871
	Tunned Model	0.859	0.863	0.789	0.927	0.852	0.871

Decision Tree	Base Model	1	0.813	0.819	0.809	0.814	0.813
	Tunned Model	0.847	0.847	0.774	0.906	0.835	0.854
Random Forest	Base Model	0.999	0.879	0.840	0.910	0.874	0.881
	Tunned Model	0.850	0.845	0.804	0.876	0.839	0.848
SVM	Base Model	0.870	0.866	0.788	0.934	0.855	0.875
	Tunned Model	0.870	0.866	0.788	0.934	0.855	0.875
Gradient Boosting	Base Model	0.865	0.867	0.798	0.926	0.857	0.874
	Tunned Model	0.967	0.876	0.823	0.921	0.869	0.880
XG Boosting	Base Model	0.909	0.870	0.808	0.922	0.861	0.876
	Tunned Model	0.999	0.880	0.832	0.920	0.874	0.883

Table 3: The result of train-test split with train = 0.8 and test = 0.2 with SMOTE

b. UnderSampling

		Accuracy		Precision	Recall	F1-score	AUC
		Train	Test				
Logistic Regression	Base Model	0.708	0.704	0.570	0.777	0.658	0.719
	Tunned Model	0.700	0.700	0.557	0.780	0.650	0.718
Decision Tree	Base Model	1	0.625	0.597	0.632	0.614	0.625
	Tunned Model	0.711	0.701	0.619	0.740	0.674	0.707
Random Forest	Base Model	0.999	0.701	0.613	0.745	0.672	0.708
	Tunned Model	0.730	0.717	0.625	0.766	0.689	0.725
SVM	Base Model	0.716	0.706	0.563	0.788	0.657	0.724
	Tunned Model	0.716	0.706	0.563	0.788	0.657	0.724
Gradient Boosting	Base Model	0.730	0.712	0.622	0.758	0.683	0.719
	Tunned Model	0.790	0.707	0.622	0.750	0.680	0.714
XG Boosting	Base Model	0.906	0.684	0.620	0.711	0.662	0.687
	Tunned Model	0.989	0.693	0.619	0.727	0.669	0.698

Table 4: The result of train-test split with train = 0.8 and test = 0.2 with UnderSampling

c. Oversampling

		Accuracy		Precision	Recall	F1-score	AUC
		Train	Test				

Logistic Regression	Base Model	0.705	0.712	0.590	0.781	0.672	0.725
	Tunned Model	0.705	0.712	0.590	0.781	0.672	0.726
Decision Tree	Base Model	1.000	0.881	0.966	0.825	0.890	0.892
	Tunned Model	0.780	0.735	0.735	0.734	0.735	0.735
Random Forest	Base Model	1.000	0.940	0.972	0.914	0.942	0.942
	Tunned Model	0.752	0.745	0.676	0.784	0.726	0.750
SVM	Base Model	0.719	0.718	0.602	0.784	0.681	0.731
	Tunned Model	0.762	0.739	0.655	0.787	0.715	0.746
Gradient Boosting	Base Model	0.727	0.728	0.665	0.761	0.710	0.732
	Tunned Model	1.000	0.927	0.966	0.897	0.930	0.930
XG Boosting	Base Model	0.879	0.816	0.822	0.813	0.817	0.816
	Tunned Model	0.998	0.930	0.970	0.898	0.933	0.933

Table 5: The result of Train-test split with train = 0.8 and test = 0.2 with OverSampling

+) With SMOTE 0.8: Among the baseline models, the Random Forest classifier demonstrates the highest test accuracy, precision, F1 score, and AUC. However, the baseline models of Random Forest and Decision Tree exhibit a significant disparity between their train and test accuracy, indicating overfitting. By employing cross-validation and hyperparameter tuning, the XG Boost model achieves the highest test accuracy score of 88.80% and an AUC of 0.883. The utilization of cross-validation and hyperparameter tuning effectively mitigates the risk of overfitting and enhances the overall performance of the model.

+) With UnderSampling 0.8: Comparing to the SMOTE 0.8, the score of UnderSampling is much lower. The score witnesses the overfitting of Decision Tree, Random Forest, and XG Boosting. After using tuning, Random Forest shows the highest test accuracy score of 0.717 and AUC is 0.725.

+) With OverSampling 0.8: From the table, we can clearly observe the noticeable differences in scores among the models when using oversampling to address the imbalance and when using the original models before and after hyperparameter tuning.The baseline models, Random Forest and Decision Tree, exhibit significantly higher scores for metrics such as test accuracy (94% for Random Forest) and precision (0.972 for Random Forest), as well as a high AUC score (0.942 for Random Forest). However, these scores decrease substantially after hyperparameter tuning.

In contrast, Gradient Boosting and XGBoosting show significant improvements after hyperparameter tuning. Particularly, Gradient Boosting witnesses a remarkable increase in precision

score from 0.665 to 0.966, while XGBoosting shows an increase from 0.822 to 0.97. XGBoosting demonstrates the highest scores with cross-validation and hyperparameter tuning.

From these observations, it can be concluded that oversampling the imbalanced data and employing cross-validation and hyperparameter tuning can enhance the performance of most models, except for Decision Tree and Random Forest, which experience a decrease in scores.

We also experimented with training and testing the models using different train-test split ratios, such as 0.7-0.3 and 0.5-0.5, while employing techniques like SMOTE, Undersampling, and Oversampling to address the imbalanced data. However, consistently, the 0.8-0.2 train-test split ratio yielded the highest results.

II, Conclusion

In summary, our analysis highlights key factors influencing credit card default. Gender imbalance exists, with males having a slightly higher default rate. Education level, particularly among university and graduate students, plays a significant role in default rates. Marital status has minimal impact. Lower credit amounts are associated with higher default likelihood, while higher credit limits indicate better repayment records. Age is a crucial factor, with younger individuals exhibiting higher default propensity. These insights aid credit card issuers in effectively managing credit risk.

From the above results, using SMOTE to address the imbalanced data yields high and consistent scores across the models (both tuned and base models). Undersampling, on the other hand, produces the lowest scores among the three techniques. Oversampling leads to uneven results among the models, with significant score discrepancies. Gradient Boosting and XGBoosting achieve remarkably high scores, while Logistic Regression performs relatively poorly. Decision Tree exhibits overfitting issues, even after balancing the data.

The best scores are obtained with a train-test split ratio of 0.8-0.2, while the worst scores are observed with a 0.5-0.5 split ratio.

Among all the models tested, Random Forest and Gradient Boosting demonstrate the best performance consistently.

In summary, through multiple trials, Random Forest and Gradient Boosting consistently exhibit the highest performance. SMOTE is effective in balancing the data and improving model performance. The choice of the train-test split ratio significantly impacts the scores.

APPENDIX

1. Ruilin Liu (2018), “Machine Learning Approaches to Predict Default of Credit Card Clients”, *Modern Economy*, Vol.9 No.11
2. Chester Wang, HanChen Wang, Qurat-ul-Ain Azim, Renee Kwon;(2022), “Credibility Classification of Credit Card Clients”,
3. Roweida Mohammed, Jumanah Rawashdeh and Malak Abdulla,(2020), “Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results”, *International Conference on Information and Communication Systems (ICICS)*
4. Ying Chen, Ruirui Zhang,(2021) ”Research on Credit Card Default Prediction Based on k-Means SMOTE and BP Neural Network”, *Complexity*, Volume 2021, Article ID 6618841, 13 pages
5. Sheikh Rabiul Islam, W. Eberle, S. Ghafoor, (2018), “Credit Default Mining Using Combined Machine Learning and Heuristic Approach”, *Computer Science*
6. Rohini Srivastava, ... Basant Kumar,(2023), “Classification model of machine learning for medical data analysis”, *Statistical Modelling in Machine Learning.*
7. Uduak A. Umoh, ... Emmanuel E. Nyoho, (2022), “ Fuzzy-machine learning models for the prediction of fire outbreaks: A comparative analysis”, *Artificial Intelligence and Machine Learning for EDGE Computing.*
8. Chrysovalantis Gaganis, Panagiota Papadimitri, Fotios Pasiouras & Menelaos Tasiou,(2022), “Social traits and credit card default: a two-stage prediction framework”, *Annals of Operation Research* , Volume 325, pages 1231–1253, (2023)
9. Olalekan J. Awujoola, ... Olayinka R. Adelegan,(2023), “Genomic data science systems of Prediction and prevention of pneumonia from chest X-ray images using a two-channel dual-stream convolutional neural network”, *Data Science for Genomics*, Chapter 13