**SUPERVISED MACHINE LEARNING- CLASSIFICATION**

**ON**

**CREDIT CARD DEFAULT PREDICTION**



**Submitted by:**

**Shreesha K, Data Science Trainee, Almabetter**

**E-mail:** [**shreeshasrisha@gmail.com**](mailto:shreeshasrisha@gmail.com)

**ABSTRACT**

With the advancement of technology, banks have started to use new strategies to improve their market share. They provide various new services to their customers. A credit card is one such service where a customer can use a certain amount of money and then he is supposed to pay it back. To improve their market shares, banks provide credit cards to their customers without checking their backgrounds. Moreover, many customers use their credit cards beyond their repayment capacity, resulting in higher debt accumulation. It is a greater task for banks to identify risky and non-risky customers. In this project, I have built a supervised ML classification model, which can predict risky and non-risky customers. This model can predict the potential defaulters (who fail to repay within a certain period), thereby helping the banks to take necessary actions.

***Keywords: Credit Card, Debt Accumulation, Supervised ML Model.***

1. **PROBLEM STATEMENT**

This project is aimed at predicting the case of customers default payments in Taiwan. From the perspective of risk management, the result of predictive accuracy of the estimated probability of default will be more valuable than the binary result of classification - credible or not credible clients. We can use the K-S chart to evaluate which customers will default on their credit card payments.

Below there are the description of the attributes that will be used in our model for better understanding of the data:

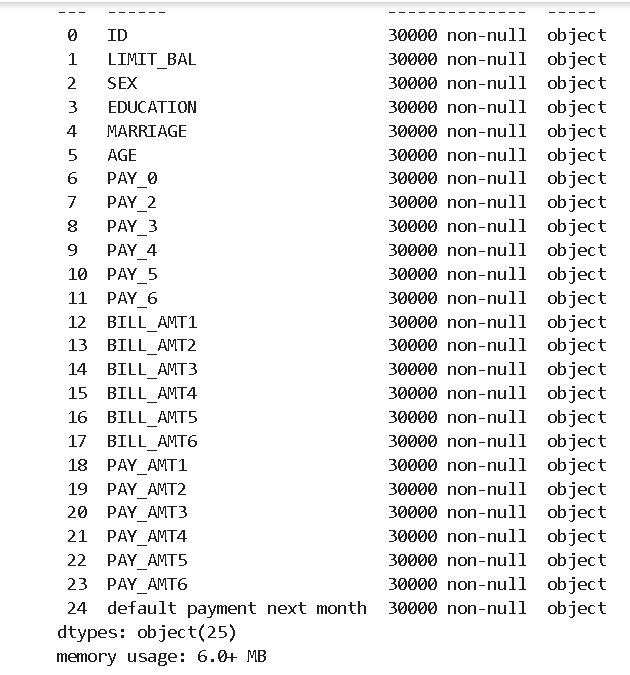
* ID: ID of each client
* LIMIT\_BAL: Amount of given credit in NT dollars (includes individual and family/supplementary = credit)
* SEX: Gender (1=male, 2=female)
* EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)
* MARRIAGE: Marital status (1=married, 2=single, 3=others)
* AGE: Age in years
* PAY\_0: Repayment status in September, 2005 (-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)
* PAY\_2: Repayment status in August, 2005 (scale same as above)
* PAY\_3: Repayment status in July, 2005 (scale same as above)
* PAY\_4: Repayment status in June, 2005 (scale same as above)
* PAY\_5: Repayment status in May, 2005 (scale same as above)
* PAY\_6: Repayment status in April, 2005 (scale same as above)
* BILL\_AMT1: Amount of bill statement in September, 2005 (NT dollar)
* BILL\_AMT2: Amount of bill statement in August, 2005 (NT dollar)
* BILL\_AMT3: Amount of bill statement in July, 2005 (NT dollar)
* BILL\_AMT4: Amount of bill statement in June, 2005 (NT dollar)
* BILL\_AMT5: Amount of bill statement in May, 2005 (NT dollar)
* BILL\_AMT6: Amount of bill statement in April, 2005 (NT dollar)
* PAY\_AMT1: Amount of previous payment in September, 2005 (NT dollar)
* PAY\_AMT2: Amount of previous payment in August, 2005 (NT dollar)
* PAY\_AMT3: Amount of previous payment in July, 2005 (NT dollar)
* PAY\_AMT4: Amount of previous payment in June, 2005 (NT dollar)
* PAY\_AMT5: Amount of previous payment in May, 2005 (NT dollar)
* PAY\_AMT6: Amount of previous payment in April, 2005 (NT dollar)
* default.payment.next.month: Default payment (1=yes, 0=no)

1. **I****NTRODUCTION**

The objective of this project is to identify the customers who might become the default in the next month. As the first step, we checked the distribution of variables and checked for any unusual data in the dataset. Then data refining is carried out by which the attributes names and data types had been modified appropriately. Some columns were modified and we created separate bins for the values of such columns. We checked the effect of different features on the target variable. By Exploratory Data analysis, we got some important facts about the dataset. By plotting the heatmap, we checked the correlation between different features involved in the dataset. It was found that some of the columns were irrelevant and do not cause significant changes in the pattern of the data. Such Columns were modified appropriately. By using the train\_test split and cross-validation technique, we fitted different machine-learning algorithms into the data. We used different ML classification algorithms such as Random Forest, Decision Tree, KNN, and Logistic Regression for predicting the output. To handle the class imbalance, we used SMOTE (Synthetic Minority Oversampling Technique) to resample the data and thereby improving the model performance. By using different performance evaluation metrics, we evaluated the performance of each model and compared them. We trained each model with and without SMOTE and compared how they perform in each case. After comparing the models, we decided on the best model for analysis by predetermined criteria. We drew certain inferences from the project and summarized the same.

1. **DATA HANDLING AND FEATURE ENGINEERING**
   1. **Data loading and Checking**

As discussed earlier, this dataset contains details about the customers with their credit limit, repayment status, etc. along with the status of defaulted or not.



We loaded the data from the drive using pandas.read\_csv function. Our data had 30000 rows and 25 Columns.

This dataset contains the data of datatype object.

* 1. **Checking for Null values and Duplicates**

Null or missing values are the direct results of errors or inefficiency in data recording. If they are left untreated, our ML models will result in an error. Hence, they need to be treated in the initial stages.

Fortunately, this dataset did not contain Null Values and we were not required to handle them.

Duplicate entries are redundant and they unnecessarily increase the complexity. It is important to remove such observations.

It was observed that there were no such duplicate entries in the dataset and all were unique.

Some of the numerical values were also present in the ‘object’ format. In the later stages, we converted them into ‘int’ datatype.

* 1. **EDA**

As the first step, we checked the proportion of defaulted and non-defaulted customers. We observed that 77% percent of the customers have not defaulted. We checked the distribution of credit limit and age to find any unusual data points and outliers. We segregated the customers based on their education, marital status, and gender. It was observed that most of the defaulted customers were females. Customers with higher education tend to default more. Marital status did not have a great impact on the default rate. We also categorized the customers based on their age and observed that elder people did not tend to default compared to younger people. We checked whether the education level had any effect on the credit limit. It was noted that people with higher education levels got higher credit limits. The customers who did not default had higher credit limits than their counterparts. Overall, these analyses provided a good insight into how these independent features affect our target variable.

* 1. **Handling categorical features**

It is important to encode the categorical features into numerical values otherwise the scikit library would not recognize them. We used one-hot encoding to produce binary digits such as 0 and 1 to convert categorical columns into numeric ones.

* 1. **Handling class imbalance**

The dataset contained many not-default customers and a small proportion of the customers have defaulted. It results in class imbalance and the model’s accuracy will be hampered. We used SMOTE (Synthetic Minority Oversampling Technique) to oversample the minority category. This technique produces duplicate values of minority classes resulting in the equal size of both categories.

* 1. **Fitting Different Models**

We used many ML classification models to get better results. We used Random Forest, Logistic Regression, KNN, and Decision Tree algorithms. We tried each algorithm with and without SMOTE to compare the performance in each case. For better results and to avoid overfitting, we performed hyperparameter tuning.

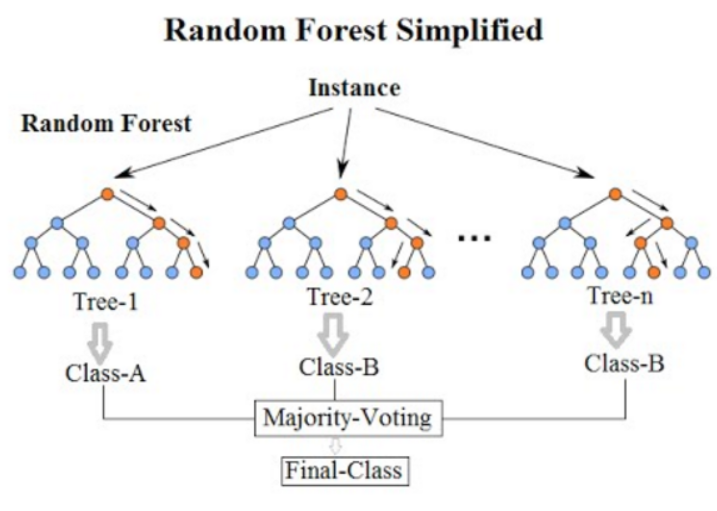
1. **ALGORITHMS**

Our target variable is discreet, i.e., 0

and 1. This is a case of Machine Learning classification problem.

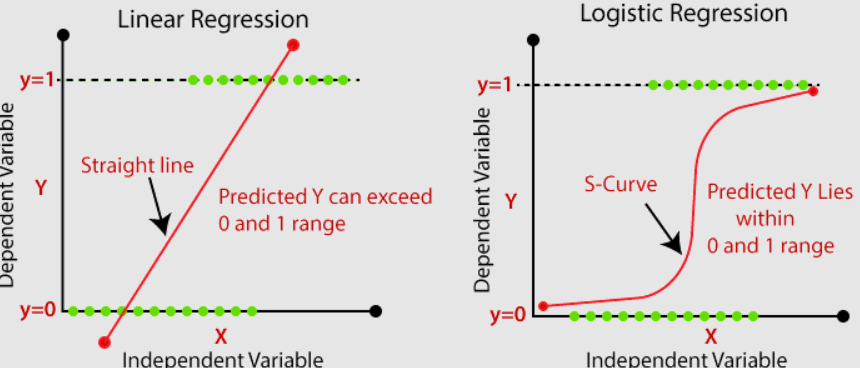
* 1. **Random Forest**

Random Forest is a classifier algorithm that trains several decision trees on various subsets of the parent dataset and produces the final output by taking the average of the outputs of all trees. It uses an ensemble methodology to produce the outcome. During splitting of nodes, observations and features will be selected randomly.

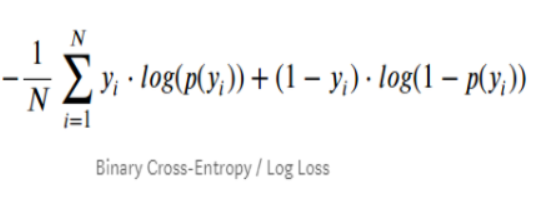


* 1. **Logistic Regression**

Logistic Regression is used to build ML models where the dependent variable is discreet and dichotomous. It fits an S-shaped sigmoid curve into the dataset and makes predictions depending on that. It is highly accurate in those cases where there is a linear relationship between features. It is less calculation intensive. On the downside, it is not too accurate if the distribution of data is too complex.

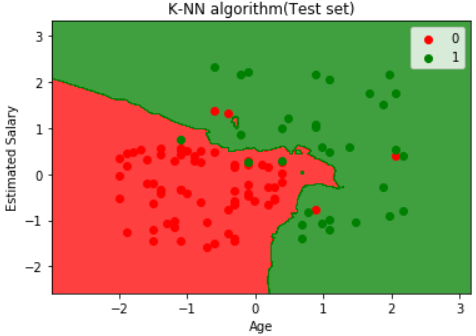


This model minimizes the value of Log Loss function, which is given by,



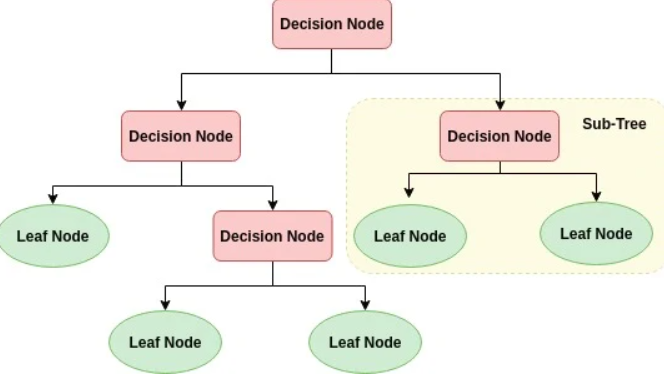
* 1. **K- Nearest Neighbors (KNN)**

KNN uses the similarity of the features to predict the value of new data points. The new data point is assigned a value based on how closely it matches its neighboring points. It is a lazy learner algorithm and does not work well with missing or noisy data. It performs poorly on relatively higher dimensional datasets.



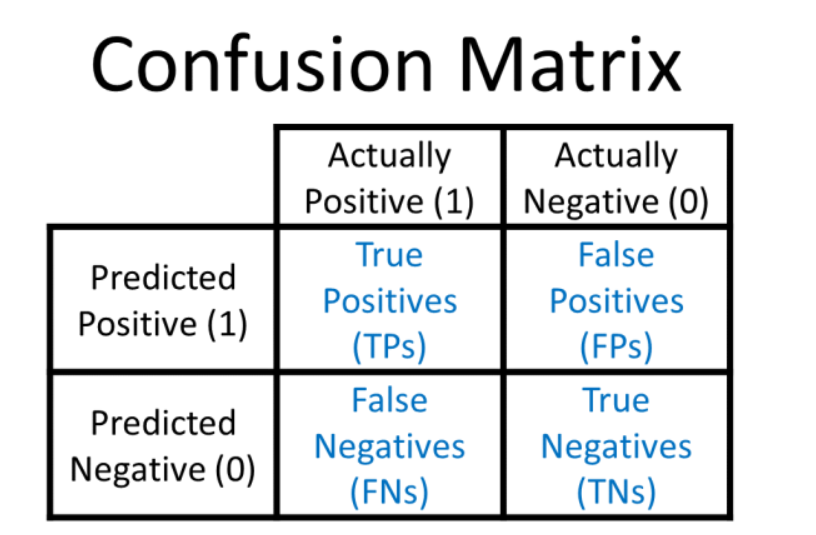
* 1. **Decision Tree**

A decision tree is a supervised ML model where data is continuously split based on certain parameters. It contains decision nodes and leaves. The data is split into decision nodes and leaf nodes contain the final output. It is highly prone to overfitting if not tuned well.



1. **PERFORMANCE PARAMETERS**
   1. **Confusion Matrix**

It indicates the extent to which the ML algorithm can classify the datapoints into specific classes. It contains two axes, namely, actual classes and predicted classes. The TP, TN indicate the correctly classified datapoints, whereas FP, FN represent the incorrectly classified datapoints.



* 1. **Precision and Recall**

Precision is the ratio of correctly classified positive datapoints to the total number of positive predictions.

It can be expressed as **TP/(TP+ FP).**

When our main focus in on False positives, this parameter is useful.

Recall is the ratio of correctly classified positive datapoints to the total number of positive datapoints.

It is given by **TP/(TP+FN).**

When our focus is on False negatives, we use recall.

* 1. **Accuracy**

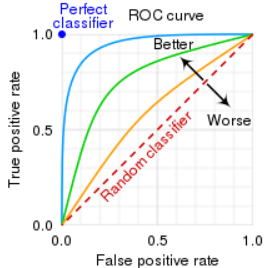
Accuracy is the ratio of the number of correct predictions to the total number of predictions. However, in the case of imbalanced dataset its applicability is limited.

Accuracy is given by

**TP+TN/(TP+TN+FP+FN)**

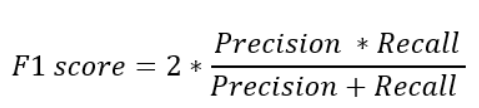
* 1. **Area Under ROC Curve**

It is the curve used to predict the efficiency of the model. It has two axes. Its Y axis is True Positive Rate (Proportion of positive datapoints correctly classified, also called recall). On the X axis, it has False Positive Rate (Proportion of negative datapoints incorrectly classified). Higher the area under curve, more is the model performance.



* 1. **F1 Score**

F1 score combines the precision and recall of a classification model by taking their harmonic mean. A higher value of F1 Score indicates the higher values of precision and Recall. It is given by



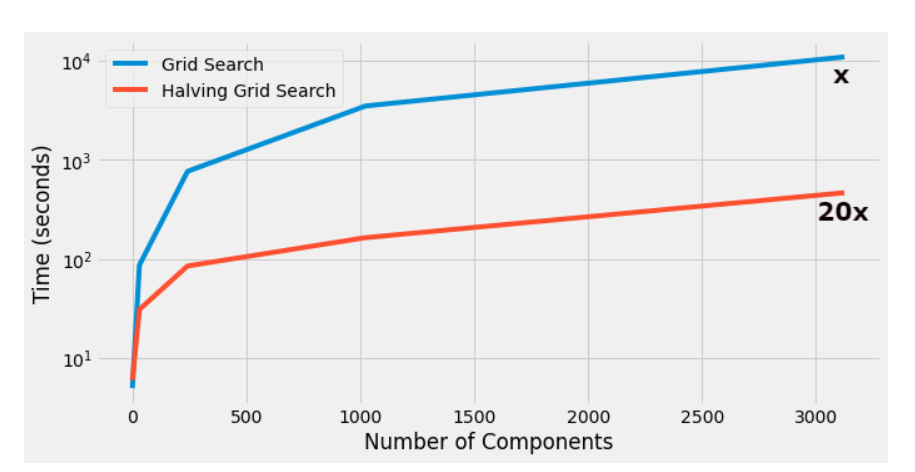
1. **CROSS VALIDATION AND HYPERPARAMATER TUNING**

Cross Validation is a method by which the model is trained by using the subset of the dataset and tested by using the complementary subset. It is performed to ensure that the model catches the correct pattern of the data and becomes robust to the random noises within the data.

Hyperparameters are those variables that affect the way by which an ML model gets trained. By adjusting these hyperparameters we can modify the performance, stability, and interpretation of the model. The commonly used methods for this purpose are GridSearchCV and RandomSearchCV. Being a large dataset, Gridsearch Cross validation takes too much time. So sometimes we have used HalvingGridsearchCV, which is a little bit inaccurate, but multiple times faster than the GridSearchCV. GridsearchCV evaluates the performance of the model for all the combinations of different hyperparameters. On the other hand, RandomSearchCV chooses these combinations randomly, and obtaining the best model will be a hit or miss.



For larger datasets, GridsearchCV results in a lot of possible combinations of parameters. It takes a lot of time. An approach towards reducing the time is by using HalvingGridsearchCV. It follows a successive halving approach. It first trains t a subset of the data for all possible combinations. Then it finds the best-performing combinations. A larger subset of the data is trained using these top combinations. As time progresses, the parameter counts decreases, and the size of the training data increases. Since it follows a successive halving approach, its time complexity is too less compared to the conventional methods.



1. **CONCLUSION**

We have loaded the data, treated the NULL values, performed feature engineering, performed exploratory data analysis, and built ML models. After the evaluation, by analyzing the performance parameters, we drew some conclusions.

* There was an imbalance in the target variable. It was handled by SMOTE (Synthetic Minority Oversampling Technique).
* Accuracy is not a good performance parameter in this problem where the data is imbalanced. We cannot classify a defaulter as a non-defaulter by mistake, i.e., we need to keep an eye on the number of false negatives. Hence, recall is a better performance parameter.
* We have used Random Forest, Logistic Regression, KNN, and Decision Tree algorithms. Amongst these models, the performance of Logistic Regression was quite poor.
* Logistic Regression had a heavy imbalance in the recall and precision score, which got balanced after implementing smote. It required further hyperparameter tuning.
* Decision Tree and Random Forest algorithms performed better with and without SMOTE. Random Forest had a recall score of 93% (for class 0), and 83% (for class 1), and Decision Tree had a recall score of 77% (for class 0), and 91% (for class 1).
* The performance of all models generally got better with SMOTE.
* Repayment Status in September, Repayment Status in August and Gender were the important features.

1. **REFERENCES**

* GeeksForGeeks
* AnalyticsVidya
* TowardsDataScience