CREDIT CARD DEFAULT PREDICTION

Data Set: Default of credit card clients dataset

1 Abstract

In order to achieve the highest accuracy, precision recall, and F1-scores, a range of techniques, including logical data analysis, one-hot encoding for data cleaning, PCA, LDA for feature reduction, outlier detection and removal utilizing z-scores, were used or tried in this project. Additionally, we have carried out a number of feature analysis, data exploration, and visualization techniques. Overall, seven distinct models have been attempted and tested along with a standard classifier such the nearest means classifier and simple system analysis. we can claim that "*Support Vector Machine*", "*Naïve Bayes*", "*Logistic Regression*" are the best models to predict defaulters in credit card data.

2 Introduction

2.1 Problem Assessment and Goals

The credit card default dataset gives information about the payment records, demographic factors, and, bill statements of credit card clients in Taiwan from April 2005 to September 2005. The dataset aims to determine whether the customer will be a defaulter in the next month based on the various predictive analysis. 23 features related to a customer influence the decision of them making a payment in the next month. There are two classes in the target variable "default payment next month"; 1 denotes that the person will be in default the following month, and 0 denotes the opposite. The ordinal measures that indicate the payment delay in months are displayed in the columns "PAY_0," "PAY_1," etcetera. "BILL_AMT" shows the bill statement of a particular month. The amount paid in a given month is displayed in the column "PAY_AMT". Similarly, Age, Marriage, and other nominal features have a major effect on the analysis.

3.1 Dataset Usage

- The dataset contains 27000 rows and 24 columns. The last column, "default payment next month", is out target column consisting of the labels we are ought to predict.
-

Number of Constraints	Degrees of Freedom
27000	23

- We have used the train-test split module of sklearn to create a training and validation set respectively. These sets were used to evaluate the initial and final performances after the data cleaning techniques applied on the dataset. Validation set was used towards the end to evaluate the performance of each classifier. Additionally, we also used k-fold cross validation to find the best model through multiple iterations, and used the same to train the dataset.
- The TEST dataset was used only once, Once we cleaned and scaled the train dataset, classifier models were trained and implemented on the test data.

3.2 Pre-processing and Visualization

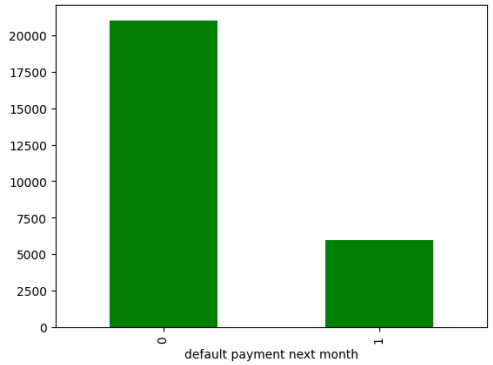
- <u>Running baseline models on unmodified original dataset:</u>

MODEL NAME	VALIDATION SET ACCURACY	VALIDATION SET F1-SCORE
TRIVIAL SYSTEM	0.66	0.656
NEAREST MEAN CLASSIFIER	0.53	0.5712

- <u>Detecting aberrations in the dataset</u>:

1. Finding null/missing data- We implemented 'isna()' method from Pandas to check whether there is missing information in the DataFrame. In the result, it was found that there aren't any empty cells in the dataset.

2. Check for count for class labels- The count of class labels is imbalanced in the dataset. Number of datapoints with class "0" accounts to 21028 and with class "1" accounts to 5972.

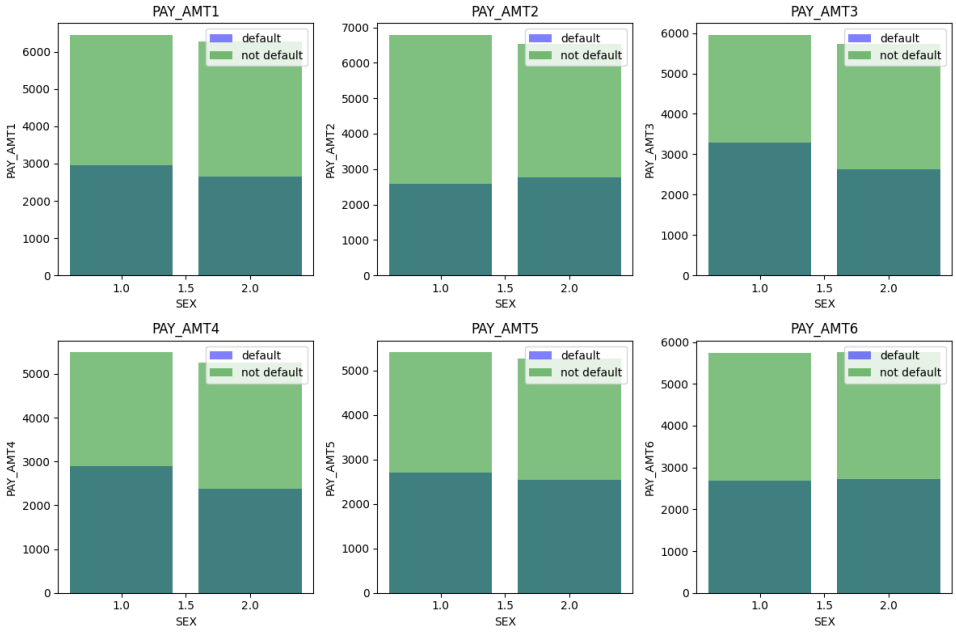



```
Percent of class 1 data:- 77.88148148148149
Percent of class 2 data:- 22.118518518518517
```

- Cross-Tab Study and Observations: -

1. To check relation between gender, payment due and target–

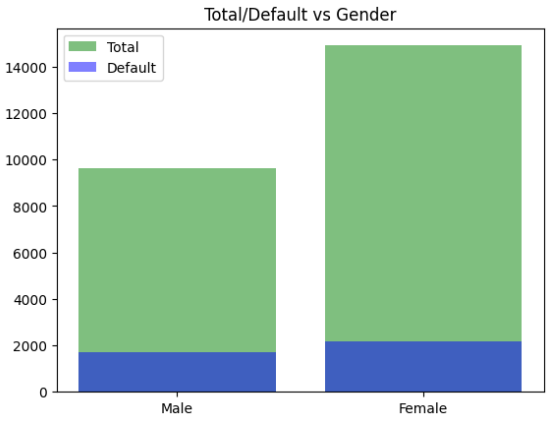
   - We are denoting 1.0 for Men and 2.0 for Women on the x-axis. The y-axis gives the comparison in the amount of payment dues for men and women in different months. We have plotted PAY_AMT1, PAY_AMT2 etcetera on the y-axis.
   - It is clear from below plots that men have larger payment amount due than women over six months. Hence it is possible that more men could default than women in this case, it would be conspicuous if we plot a cross-tab of gender with target alone



   - Percentage of male and female defaulting-

default payment next month	0	1	total	percentage_default
SEX				
1	7942	1700	9642	17.631197
2	12731	2175	14906	14.591440

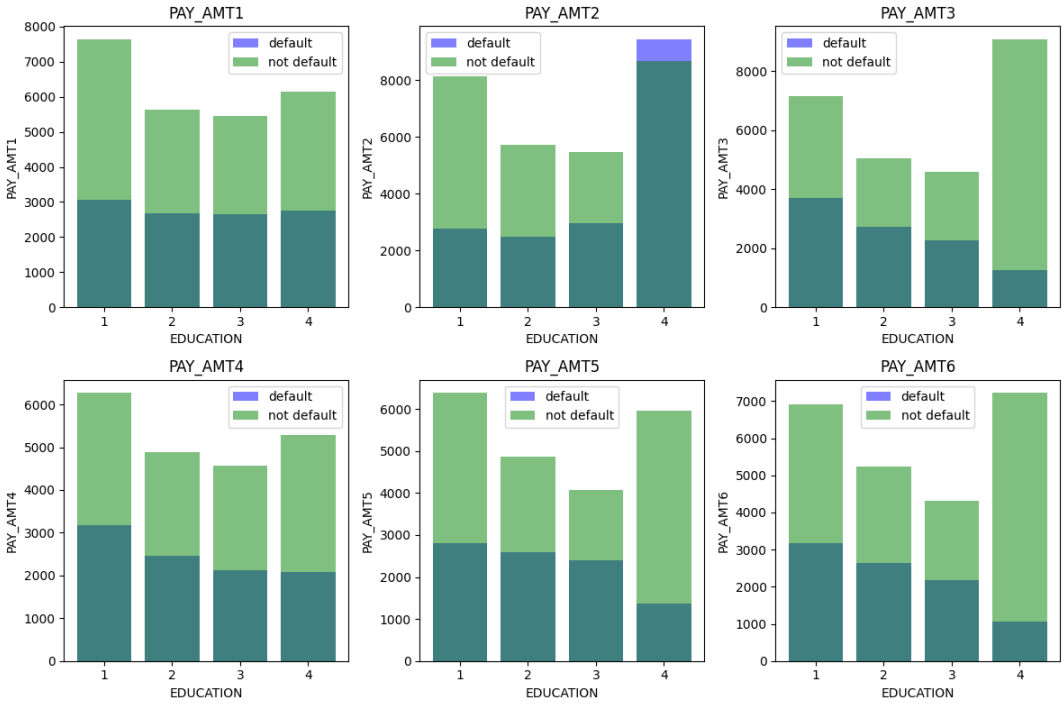
- Analysis of defaulted on total number of men and women-



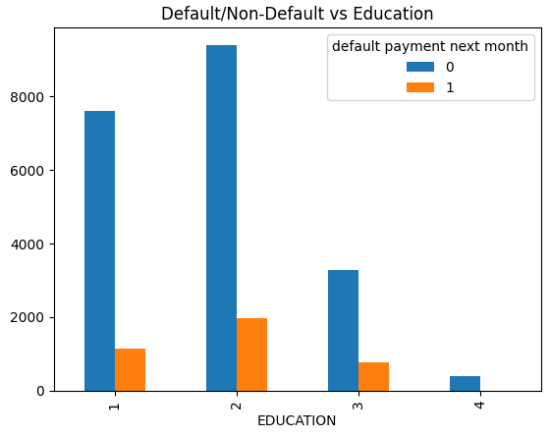
It is quite evident that men are defaulting with a total percentage of 17.63% and women are defaulting with a percentage of 14.59%. The above graph might show that women are defaulting more, But the ratio of women is quite higher than men i.e., w:m = 1.54:0

2. <u>To check relation between education, payment due and target-</u>

- There are four main categories of "Education"; Education (1 = graduate school; 2 = university; 3 = high school; 4 = others).

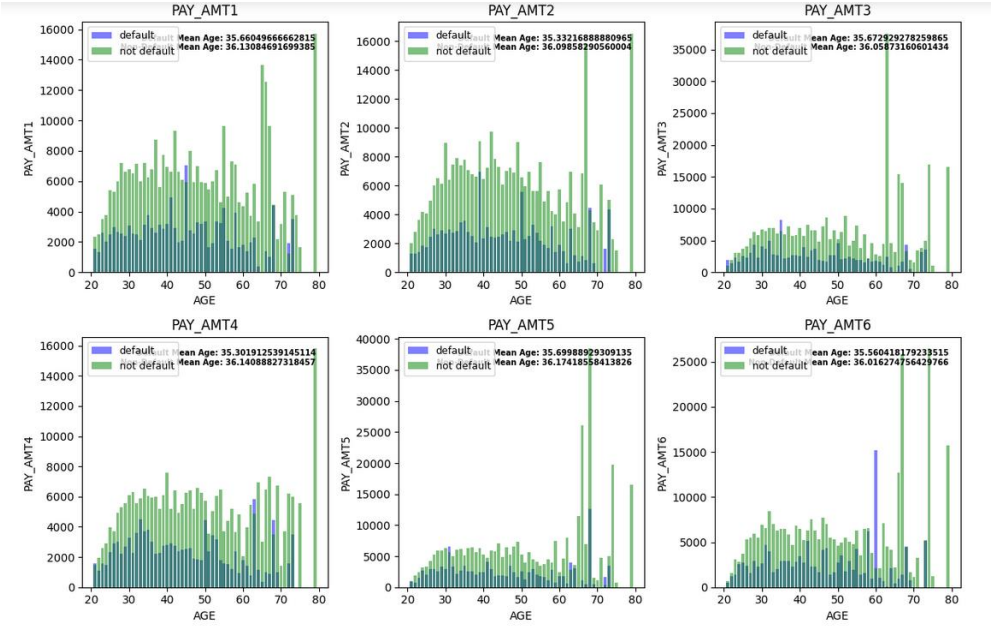


From these plots, it's quite cogent that higher the education level, lower the dues. However, for the category of others (i.e., 4), we can see vacillating behaviour for default and non-default. It must be so because this category consists of customers that can be even unemployed currently or lost their jobs. We would still carry out the cross-tab visualization and univariate analysis by comparing them to the total for each category.

- 

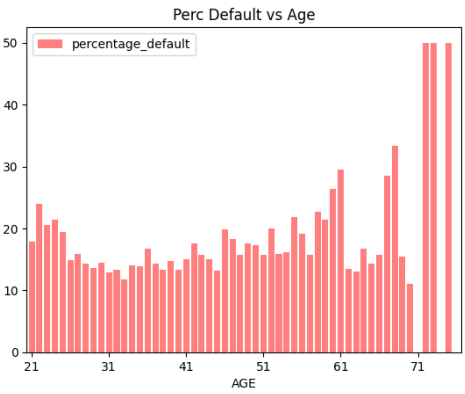
If we consider the education order from 3 i.e., high school to 1 i.e., graduate school, we see a decline in defaulters. It is quite obvious as only people with some education status would be earning. Furthermore, the amount of loan borrowers decreases as education level decreases, again quite obvious factor in this case. We are still not sure about others category as it can contain customers from any random background, such as laid-off employee.

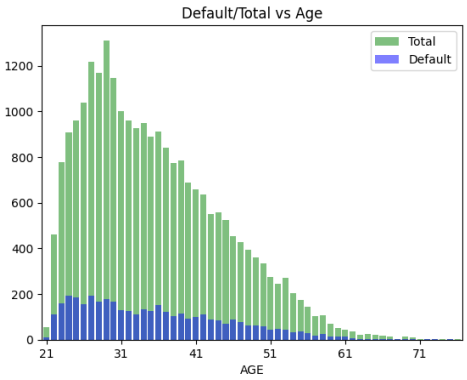
3. <u>To check relation between age, payment due and target-</u>

- 

A neat observation can be drawn that the mean of this distribution is more towards the younger side i.e., age group of 30 for both defaulter and non-defaulters. We can get a clear idea about percentages by cross-tab view and univariate analysis. It could be observed that customers of the age 30-35 then to take loan and default more than elderly people.

▪





The observations from below two plots can be drawn as follows: -The defaulter and total vs age groups indicates that the defaulters decrease the age. The reason is quite obvious that a lot younger people have taken loan and hence they tend to default more than old people. Whereas in the percentage graph we could observe that there is an almost uniform distribution of default percentages over the age group.

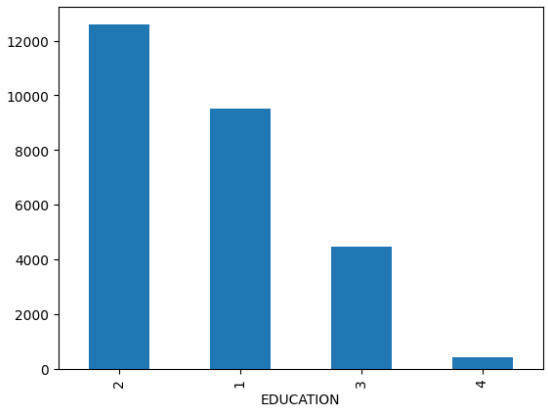
3.3 Feature Engineering

- Feature Engineering for "EDUCATION" column: -

As per description in https://archive-beta.ics.uci.edu/dataset/350/default+of+credit+card+clients categories 0,5 and 6, are to be grouped together as others. Now we shall apply a function to the education column that will assign 0,5,6 datapoints to the category 4. As described by the dataset owners that category 4 is others.

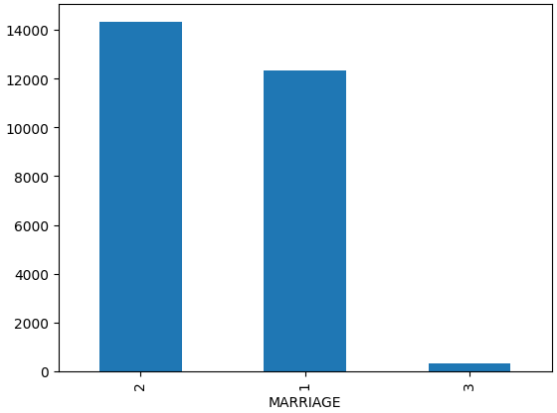
The count of categories in our newly constructed column-

EDUCATION	
1	9520
2	12596
3	4464
4	420



- Feature Engineering for "MARRIAGE" column: -
  Furthermore, we also observe that marital status has categories only 1,2 and 3. As per dataset description X4: Marital status (1 = married; 2 = single; 3 = others). Whereas from our graphs, it's quite conspicuous we have an extra category 0. We shall assign this 0 to others as well. First let's get the count of all the marriage categories.

MARRIAGE	
1	12328
2	14340
3	332



- Other changes for robustness of the dataset and prediction analysis-
  1. We replace all the negative values by zeros first, as the PAY columns have some negative values. Here Zeros or negative values doesn't matter, it still means that the customer have no payment amount due.
  2. Remove illogical rows where Pay is 0 but default is 1, basically it says the customer has paid all of their deus but still they will default.
  3. Remove illogical rows where Pay is 0 but default is 1, basically it says the customer has dues remaining, and yet they won't default.

3.4 Feature dimensionality adjustment

- ONE HOT ENCODING (FEATURE EXPANSION)-
  - We performed One Hot Encoding for Nominal Features such as "Marriage" and "Education". It is because gender and marriage are strictly nominal features.
  - The dimensions of the new dataset are 24548 rows × 29 columns.

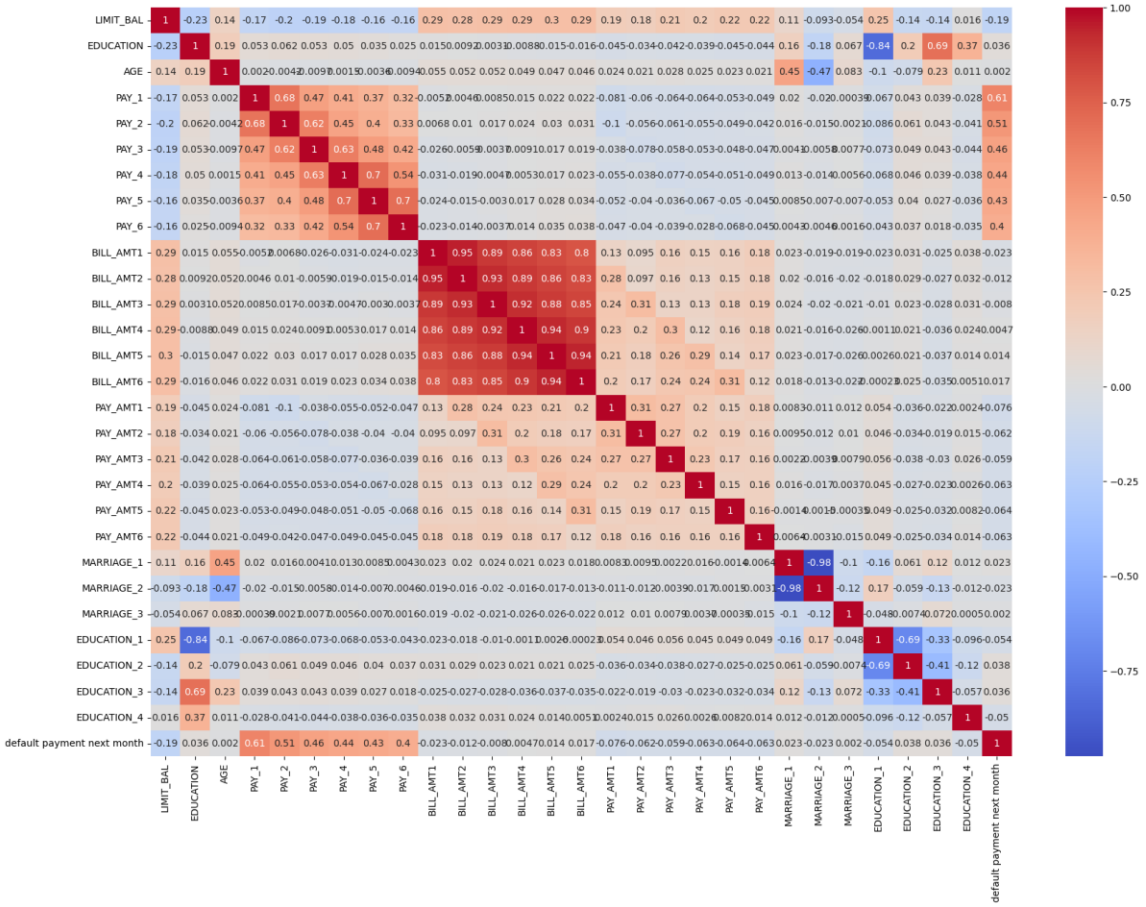
Number of Constraints	Degrees of Freedom
24548	28

- LINEAR DISCRIMINANT ANALYSIS (LDA)-

  The LDA is performed using the LinearDiscriminantAnalysis class from scikit-learn. Since it's a binary classification problem, only one component is selected for dimensionality reduction, which is specified using the n_components parameter. The fit_transform() method is used to fit the LDA model to the data and transform the data into a lower-dimensional space

Number of Constraints	Degrees of Freedom
22093	1

- CORRELATION HEATMAP-

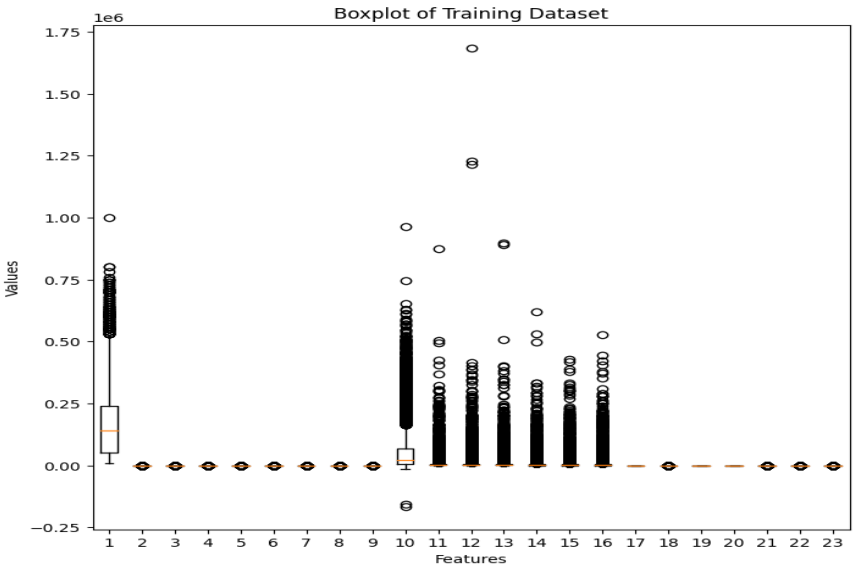


We defined a function “get_least_correlated_features” which finds the degree of similarity and dependence between the features and the target column.

```
['MARRIAGE_2', 'BILL_AMT3', 'BILL_AMT5', 'BILL_AMT6', 'BILL_AMT4', 'BILL_AMT2']
```

these features have a correlation of **85%** and above. That implies, even if we merge these two columns into one column or drop one of the columns, the change wouldn’t affect the target label.

- <u>Outlier detection and deletion (ROWS DELETION)-</u>



We have used an elliptic envelope outlier detection model is used to identify outliers in the dataset. An instance of the EllipticEnvelope model is created. The predict () method is called on the same model instance to predict the outliers in the dataset based on the fitted model. The outliers are identified as -1, while the inliers are identified as 1. Boolean indexing is used to select the rows of the input features and labels dataframes where the outlier prediction is equal to 1, indicating an inlier. The resulting data is stored in the variable's 'features' and 'labels'. After the deletion of outliers, we have 22093 rows and 23 columns in the new dataset.

Number of Constraints	Degrees of Freedom
22093	22

3.5 Training, Classification or Regression, and Model Selection

Baseline analysis-

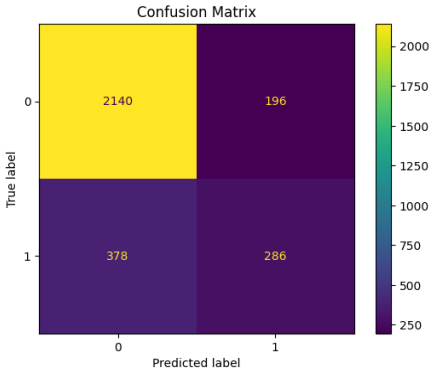
1. **Nearest Means Classifier**: -

Implementation: -

This was a baseline classifier that uses mean of each class to classify the datapoints. For NMC we created a separate NearestMeansClassifier.py file and implemented a class name NearestMeansClassifer. We only used NumPy for this purpose.

Results: - The best Test metrices were observed after performing SMOTE. The confusion matrix and classification report are as follows: -

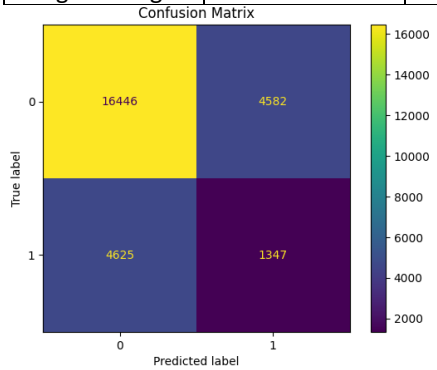
	precision	recall	F1-score	support
0.0	0.85	0.92	0.88	2336
1.0	0.59	0.43	0.50	664
accuracy	-	-	0.81	3000
macro avg	0.72	0.67	0.69	3000
weighted avg	0.79	0.81	0.80	3000



2. **Trivial System**: - It was is a naïve solution in which we generated samples randomly using the relative frequencies of each class and created a baseline for the rest of our processing. This system was implemented in a separate Trivial.py file and used in our code to judge the initial performance of our dataset. The classification report and confusion matrix for Validation/Test set are: -

	precision	recall	F1-score	support
0.0	0.78	0.78	0.78	21028
1.0	0.23	0.23	0.23	5972
accuracy			0.66	27000

macro avg	0.50	0.50	0.50	27000
weighted avg	0.66	0.66	0.66	27000



Classifiers applied on cleaned datasets-

1. **ARTIFICIAL NEURAL NETWORKS**-

Libraries and tech-stack-

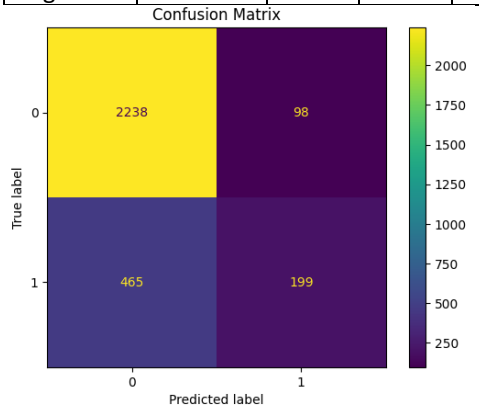
Most modules are subclasses of PyTorch library, pandas and imblearn and numpy

Implementation-

The data loaders batch and shuffle samples using parallel processing. The training loop runs for 500 epochs, iterating over data batches. Input features and labels are loaded onto the device, and a forward pass computes model predictions. Loss is calculated using nn.CrossEntropyLoss, and backpropagation is performed. The optimizer updates model parameters based on gradients. The model's performance is evaluated on the training dataset, tracking correct predictions and total samples. Predicted labels are saved, and the model score, confusion matrix, and classification report are displayed using scikit-learn's metrics module.

Results:- The best Test metrices were obtained before SMOTE on the dataset, we obtain the following classification report, confusion matrix: -

	precision	recall	F1-score	support
0.0	0.83	0.96	0.89	2336
1.0	0.67	0.30	0.41	664
accuracy	-	-	0.81	3000
macro avg	0.75	0.63	0.65	3000
weighted avg	0.79	0.81	0.78	3000



Parameters: Batch size - Train: 6007, Test: 375

PARAMETERS USED-

Learning rate	Degrees of Freedom	Number of Constraints	Epochs

0.1	1	22093	500

2. **Logistic Regression**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy.

Implementation-

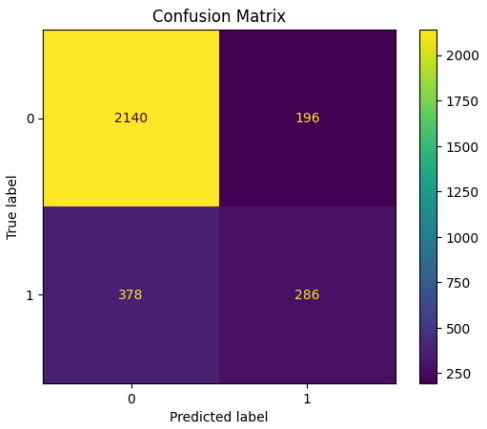
Hyperparameters are defined in a dictionary, including regularization penalty, inverse of strength C, solver algorithm, and max iterations. Grid search cross-validation tests all combinations, splitting data into five parts. Logistic regression learns weights to maximize likelihood. SMOTE counters class imbalance.

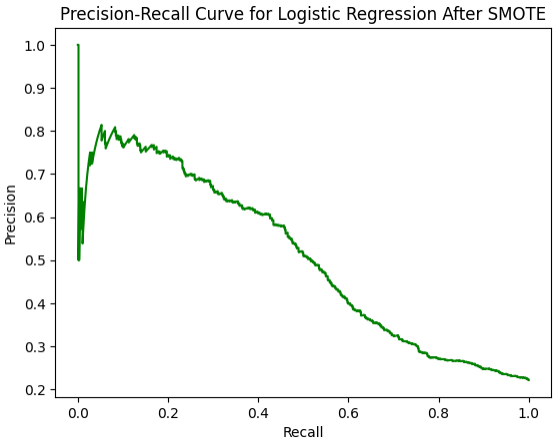
Results:- The best Test metrices were obtained after perform SMOTE on the dataset, we obtain the following classification report, confusion matrix and precision recall curve: -

	precision	recall	F1-score	support
0.0	0.8	0.92	0.88	2336
1.0	0.59	0.43	0.50	664
accuracy	-	-	0.81	3000
macro avg	0.72	0.67	0.69	3000
weighted avg	0.79	0.81	0.80	3000

PARAMETERS USED-

Penalty	Degrees of Freedom	Number of Constraints	Max iterations
L1	1	27547	100





3. **K-NEAREST NEIGHBORS**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy

KNeighborsClassifier, GridSearchCV, f1_score, classification report, confusion matrix, and ConfusionMatrixDisplay, SMOTE are some of the important libraries/modules that are used in this code.

Implementation-

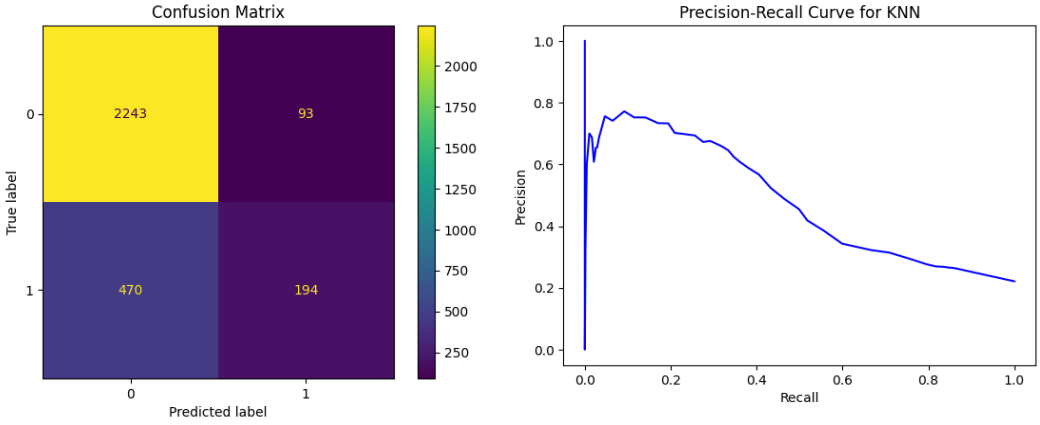
Using KNeighborsClassifier() and GridSearchCV(), hyperparameters are tuned for KNN with neighbors in range (1, 50). Best parameters are used to fit the model. Small K captures fine patterns but is susceptible to outliers; large K is stable but may miss patterns. SMOTE counters class imbalance.

Results:- The best Test metrices were obtained before performing SMOTE on the dataset, we obtain the following classification report, confusion matrix and precision recall curve: -

	precision	recall	F1-score	support
0.0	0.83	0.96	0.89	2336
1.0	0.68	0.29	0.41	664
accuracy	-	-	0.81	3000
macro avg	0.75	0.63	0.65	3000
weighted avg	0.79	0.81	0.78	3000

PARAMETERS USED-

N-neighbours	Degrees of Freedom	Number of Constraints
49	1	22093



4. **<u>SUPPORT VECTOR MACHINE (SVM)</u>**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy

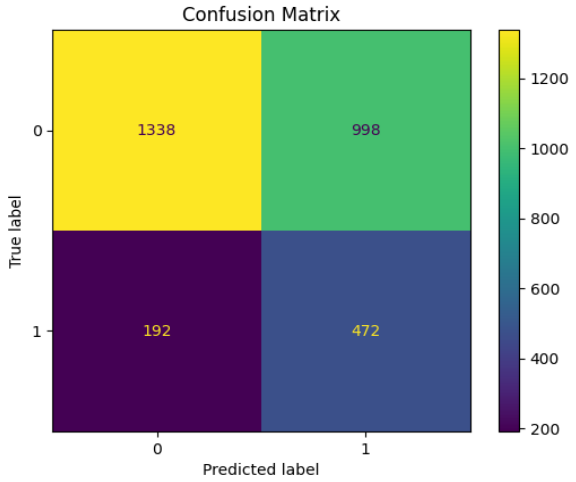
SVC, GridSearchCV, f1_score, classification report, confusion matrix, and ConfusionMatrixDisplay, SMOTE are some of the important libraries/modules that are used in this code.

Implementation-

SVM determines the hyperplane maximizing margin between classes. Using GridSearchCV, hyperparameters 'C', 'gamma', and 'kernel' are tuned. Best parameters are used for the final model, generating a classification report. SMOTE counters class imbalance.

Results:- The best Test metrices were obtained before performing SMOTE on the dataset, we obtain the following classification report, confusion matrix: -

	precision	recall	F1-score	support
0.0	0.83	0.96	0.89	2336
1.0	0.66	0.30	0.41	664
accuracy	-	-	0.81	3000
macro avg	0.74	0.63	0.65	3000
weighted avg	0.79	0.81	0.78	3000



5. **NAÏVE BAYES**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy

GaussianNB, precision_recall_curve, GridSearchCV, f1_score, classification report, confusion matrix, and ConfusionMatrixDisplay, SMOTE are some of the important libraries/modules that are used in this code.

Implementation-

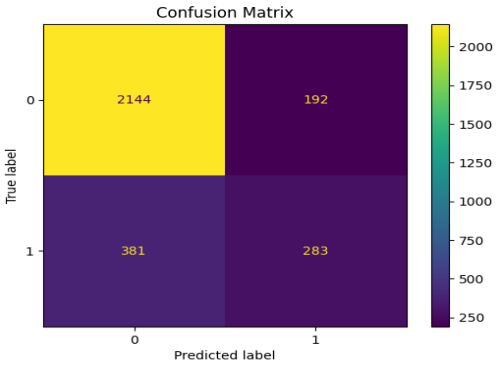
GNB represents classes as Gaussian distributions, calculating and estimating mean and variance from training data. Using GridSearchCV, hyperparameters 'var_smoothing' is tuned and SMOTE counters class imbalance.

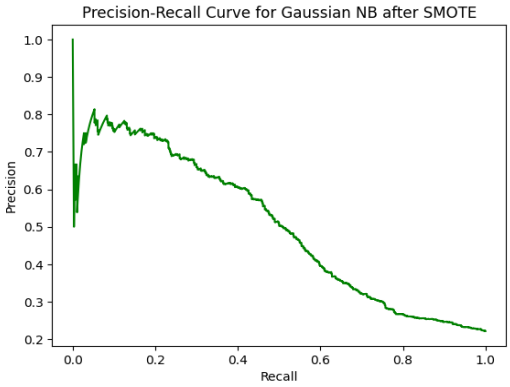
Results:- The best Test metrices were obtained after performing SMOTE on the dataset, we obtain the following classification report, confusion matrix and precision recall curve: -

	precision	recall	F1-score	support
0.0	0.85	0.92	0.88	2336
1.0	0.66	0.30	0.41	664
accuracy	-	-	0.81	3000
macro avg	0.72	0.67	0.69	3000
weighted avg	0.79	0.81	0.80	3000

PARAMETERS USED-

Variance Smoothing	Degrees of Freedom	Number of Constraints
1e-10	1	31220





6. **Random-Forest Classifier**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy

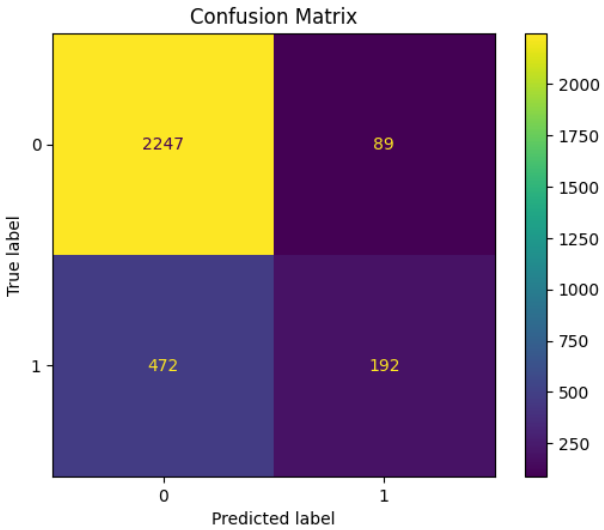
Matplotlib, RandomForestClassifier, RandomizedSearchCV, confusion_matrix, classification_report, ConfusionMatrixDisplay, precision_recall_curve, imblearn

Implementation-

Random Forest classifier uses ensemble learning to classify the data. Ensemble learning is where we use n_estimators or n-decision trees. It then collects a vote for accuracy from every tree and then chooses the majority vote. It is not necessary that more number of estimators means better classification every time.

Results:- The best Test metrices were obtained before performing SMOTE on the dataset, we obtain the following classification report, confusion matrix and precision recall curve: -

	precision	recall	F1-score	support
0.0	0.83	0.96	0.89	2336
1.0	0.68	0.29	0.41	664
accuracy	-	-	0.81	3000
macro avg	0.75	0.63	0.65	3000
weighted avg	0.79	0.81	0.78	3000



PARAMETERS USED-

n_estimators	Degrees of Freedom	Number of Constraints	min_samples_leaf	Max depth	min_samples_split
137	1	22093	7	4	12

7. **Radial-Basis Network**-

Libraries and tech-stack-

Most modules are subclasses of sklearn library, pandas and imblearn and numpy

rbf_kernel, KFold, shuffle, f1_score, classification report, confusion matrix, and ConfusionMatrixDisplay, SMOTE are some of the important libraries/modules that are used in this code.

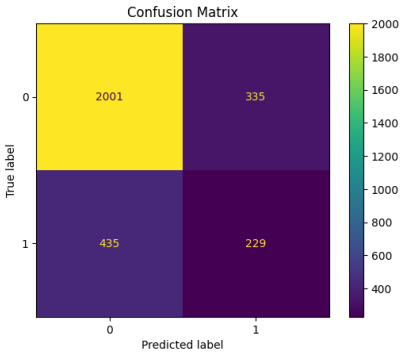
Implementation-

The RBN algorithm is a type of artificial neural network that uses radial basis functions as activation functions to model complex nonlinear relationships between input and output variables.

From "RadialBasisNetwork" class, methods calc_random_centers and calc_kmeans_centers that compute the basis function centers using random sampling or k-means clustering, respectively. "cluster_center_gamma_tuning" function loops over all combinations of gamma and M, and for each combination, loops over each fold of the cross-validation, trains a RadialBasisNetwork model on the training data, and computes the RMSE on both the training and test data for that fold. SMOTE is applied to counter the problem of class imbalance.

Results:- The best Test metrices were obtained before performing SMOTE on the dataset, we obtain the following classification report, confusion matrix: -

	precision	recall	F1-score	support
0.0	0.82	0.86	0.84	2336
1.0	0.41	0.34	0.37	664
accuracy	-	-	0.81	3000
macro avg	0.61	0.60	0.61	3000
weighted avg	0.73	0.74	0.74	3000



PARAMETERS USED-

Cluster Center	Degrees of Freedom	Number of Constraints	Best Gamma
300	1	22093	10

4.1 ANALYSIS REPORT

- From section (3) we could draw out a brief analysis report as follows for all the 9 machine learning systems

SR.NO	Model Name	F1-Score Train	Accuracy Train	F1-Score Test	Accuracy Test	(Smote) test F-1 score	(Smote) test accuracy	SMOTE suitable?	D.O.F	N.O.C

1	Trivial System	0.8	0.76	0.66	0.66	0.68	0.7	NO	23	27000
2	Nearest Means Classifier	0.89	0.89	0.80	0.81	0.8	0.81	YES	1	27547
3	Logistic Regression	0.9	0.9	0.80	0.81	0.8	0.81	YES	1	27547
4	Knn	0.89	0.9	0.78	0.81				1	22093
5	Radial Basis Network	0.89	0.9	0.74	0.81	0.36	0.37	NO	1	22093
6	Artificial Neural Network	0.73	0.74	0.78	0.81	0.78	0.79	NO	1	22093
7	Naïve Bayes	0.89	0.89	0.80	0.81	0.8	0.813	YES	1	31220
8	SVM	0.89	0.9	0.78	0.81	0.8	0.81	YES	1	22093
9	Random Forest	0.89	0.89	0.78	0.81	0.47	0.45	NO	1	22093

1. Here, we have implemented SMOTE to counter the issue of unbalanced data labels. But oversampling may boost or affect the performance of the model as seen in the table.
2. In *RandomForest classifier*, *RadialBasis Network*, *ANN* and *Trivial system*, the accuracy of test data after applying SMOTE has decreased the test accuracy to a noticeable extent. Hence, we will conduct analysis without oversampling in such models.
3. According to various factors like F1-score, precision, accuracy and recall, we can claim that "*Support Vector Machine*", "*Naïve Bayes*", "*Logistic Regression*" are the best models to predict defaulters in credit card data.
4. The problem of overfitting on the train data is observed in "*K-nearest neighbors*" and "*RadialBasis Network*".

5. Libraries used-

sklearn, matplotlib, scipy, Pandas, Numpy,imblearn were most primary, important libraries used.

6 Summary and conclusions

The analysis gives us a brief representation of whether a person is going to default in the next month. By defining various features such as f1-score, precision and recall, we understand the functionality of the classification models. We tried various techniques to solve data imbalance issue by using *Optuna* and *SMOTE*, the method gave better results in some of the suitable classifiers. We learnt the deformities in the data through examination of each and every feature and engineered a modified dataset to predict our target efficiently. We implemented Z-score, Boxplots and EllipticEnvelope method to detect and remove outliers. Also, PCA and LDA were tried to test if dimensionality reduction improves the robustness of the model.

References

[1]. https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset is referred to understand and get ideas on exploratory data analysis