Machine Learning Pipeline

I. Introduction

The objective of this study is to use machine learning techniques to predict customer churn for credit card companies. The application background is in the financial industry, specifically in the credit card sector. Customer attrition is a significant issue for credit card companies, as losing customers can have a negative impact on revenue and profitability. By analyzing the patterns and trends in the data, the study aims to identify factors that contribute to customer churn and provide effective solutions to reduce it. The findings of this study can help credit card companies develop targeted retention strategies and improve customer satisfaction.

The machine learning tasks involved in this problem are exploratory data analysis, feature engineering, and predictive modelling. The first step is to analyse the data and identify the factors that contribute to customer attrition. Exploratory data analysis was performed to examine the data distribution and visualize the relationships between features. This helps to understand the variables that affect customer behaviour, such as demographics, usage patterns, and credit scores.

Once the relevant variables are identified, the next step is feature engineering, where these variables are transformed or combined to create features that can be used in predictive modelling. This process involve in data cleaning, standardization, and feature engineering techniques.

Finally, three predictive models Random Forest, Support Vector Machine, and K-Nearest Neighbors classifiers are used to build models that can predict which customers are likely to leave the portfolio. Hyperparameters were tuned using a grid search method. Confusion matrices, ROC & AUC were used to evaluate the models, and it was concluded that Random Forest performed the best. By comparing the area under the ROC curve for different models, we can determine which model is the most effective in predicting customer churn. The most significant features were determined using the Random Forest model, with the top three features being the total transaction amount in the previous 12 months, the total transaction count in the previous 12 months, and the total revolving balance on the credit card.

II. Data and preliminary analysis

A. Data Overview

The credit card holders' information data provided contains information about credit card customers, along with various demographic and account-related variables. The variable with their descriptions are given in Table 1.

In the given dataset, there are 10127 rows representing unique customers and 21 columns providing various information about each customer. Out of the 21 columns, 5 columns contain categorical data, and the remaining 16 columns contain numerical data.

Categorical columns provide information about the customer's demographics, such as gender, marital status, education, income category, and type of card. These columns contain discrete values that represent specific categories, rather than numerical values that can be mathematically manipulated.

On the other hand, numerical columns provide information about the customer's credit card usage, such as credit limit, balance, transaction amounts, and card utilization ratio. These columns contain continuous or discrete numerical values that can be mathematically manipulated.

TABLE I. Feature Description

Features	Description
Client number	Unique customer identifier
Attrition flag	This variable track whether the customer closed their account or not. If the account is closed, the value is 1. Otherwise, it is 0.
Age	Customer's age in years.
Gender	M for male, F for female
Number of dependents	Number of dependents of the customer.
Educational Qualification	Educational qualification of the account holder (High School, College, Graduate, Post-Graduate, Doctorate, Uneducated, Unknown).
Marital status	Marital status of the account holder (married, single, divorced, unknown).
Annual Income Category	Annual income category of the account holder (< $40K, $40K - 60K, $60K - $80K, $80K-$120K, > $120K, unknown).
Type of card	Type of card (Blue, Silver, Gold, Platinum).
Months on book	Period of relationship with bank in months.
Total relationship count	Total number of products held by the customer.
Months inactive	Number of months inactive in the last 12 months.
Number of contacts in the last 12 months	Number of contacts made with the customer in the last 12 months.
Credit limit on the credit card	Credit limit on the credit card.
Total revolving balance on the credit card	Total revolving balance on the credit card.
Open to Buy Credit Line (average of last 12 months)	Open to buy credit line (average of last 12 months).
Change in transaction amount (Q4 over Q1)	Change in transaction amount from Q1 to Q4.
Total transaction amount (last 12 months)	Total transaction amount in the last 12 months.
Total transaction count (last 12 months)	Total transaction count in the last 12 months.
Change in transaction count (Q4 over Q1)	Change in transaction count from Q1 to Q4.
Average card utilization ratio	Average card utilization ratio.

The target variable for this analysis is the "Attrition_Flag", which is an internal event variable that tracks whether the customer closed their account or not. The original dataset had the Attrition_Flag variable as an object data type, meaning it contained string values of "Attrited Customer" or "Existing Customer". However, to perform machine learning tasks on this variable, it needed to be converted to a binary format

where "Attrited Customer" is represented by 1 and "Existing Customer" is represented by 0. Therefore, the Attrition_Flag variable was recoded to facilitate modeling and analysis.

As part of the analysis, we will be using various machine learning techniques to predict customer attrition and identify the key factors that contribute to it. By analyzing the various features provided in the dataset, we hope to gain insights into the behavior and preferences of the customers and leverage this knowledge to develop effective retention strategies for the business.

B. Exploratory Data Analysis

```
Data columns (total 21 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   CLIENTNUM                 10127 non-null  int64
 1   Attrition_Flag            10127 non-null  object
 2   Customer_Age              10127 non-null  int64
 3   Gender                    10127 non-null  object
 4   Dependent_count           10127 non-null  int64
 5   Education_Level           10127 non-null  object
 6   Marital_Status            10127 non-null  object
 7   Income_Category           10127 non-null  object
 8   Card_Category             10127 non-null  object
 9   Months_on_book            10127 non-null  int64
 10  Total_Relationship_Count  10127 non-null  int64
 11  Months_Inactive_12_mon    10127 non-null  int64
 12  Contacts_Count_12_mon     10127 non-null  int64
 13  Credit_Limit              10127 non-null  float64
 14  Total_Revolving_Bal       10127 non-null  int64
 15  Avg_Open_To_Buy           10127 non-null  float64
 16  Total_Amt_Chng_Q4_Q1      10127 non-null  float64
 17  Total_Trans_Amt           10127 non-null  int64
 18  Total_Trans_Ct            10127 non-null  int64
 19  Total_Ct_Chng_Q4_Q1       10127 non-null  float64
 20  Avg_Utilization_Ratio     10127 non-null  float64
dtypes: float64(5), int64(10), object(6)
memory usage: 1.6+ MB
```

Upon checking the dataset, it was observed that there were no null values or duplicates present in the data. However, it was observed that some of the attributes have the value "Unknown" instead of missing values. This technique of assigning null values a separate class (Unknown) in categorical variables is a useful approach for handling missing values in such variables. This indicates that the dataset is clean and ready for analysis.

To begin the analysis, we segregated the dataset into categorical and continuous variables, considering the churn customers as the dependent variable and others as independent variables. Then, we examined the variables using various statistical metrics. The findings revealed the absence of a linear relationship between variables, prompting us to employ non-linear models for predicting customer churn.

The CLIENTNUM column contains unique customers identity numbers, which does not provide any valuable insights into predicting customer churn. Hence, it was removed from the dataset to avoid any unnecessary noise in the model. The updated dataset now consists of 14 numerical independent variables. To understand the distribution of each variable, a histogram was created for each of them.

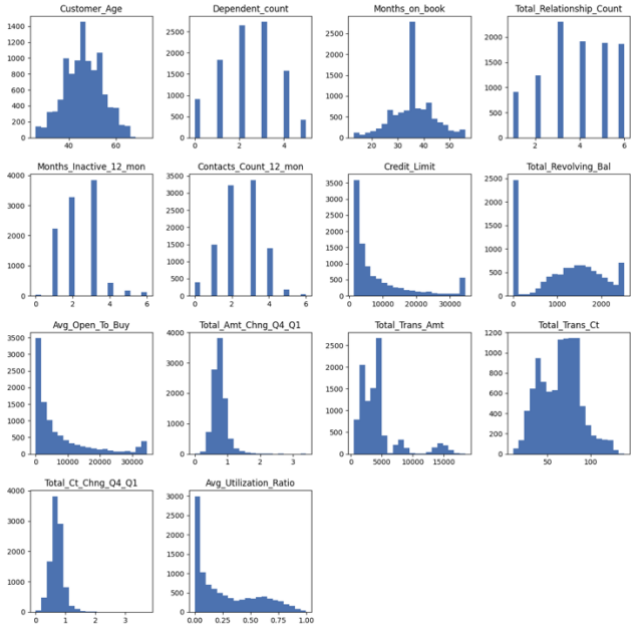


Fig. 1. Numerical features distribution Graph

The distribution of ages for the customers in the dataset appears to follow a normal distribution. This assumption of normality allows us to use the age feature in further analysis and modeling. From the distribution of dependent count, we can infer that much of the dependent count follows normal distribution, but there may be some outliers on the higher end, causing a slight right skew in the distribution. The distribution of months the customer is part of bank has a low peak and is more spread out than a normal distribution. The distribution of total transactions (Last 12 months) is multimodal, indicates that there may be distinct groups or clusters in the data. It would be worthwhile to explore these clusters and identify the factors that contribute to the different modes in the distribution. All the other features are having slight skewness in their distributions.

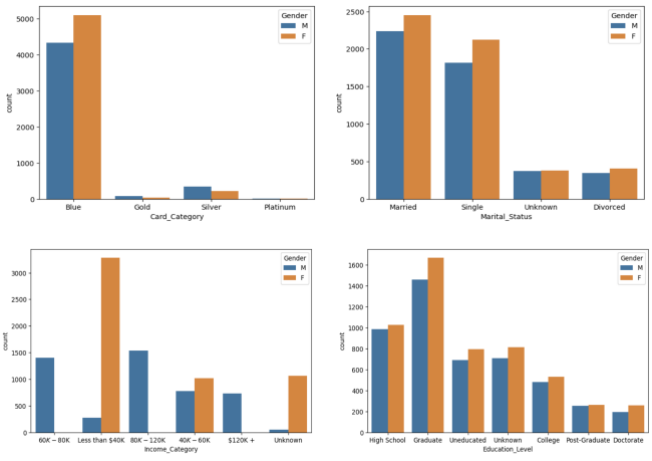


Fig. 2. Catagorical Features Distribution Graph

From the above figures we can infer that many customers with an unknown education status do not have any formal education, suggesting that over 70% of customers in our dataset have attained some level of education. Moreover, around 35% of customers have achieved a higher level of education. Also, female samples are slightly more than male samples. If we check the card category, we can see the general

customers are using Blue card. Its mean, most of them uses the basic card.

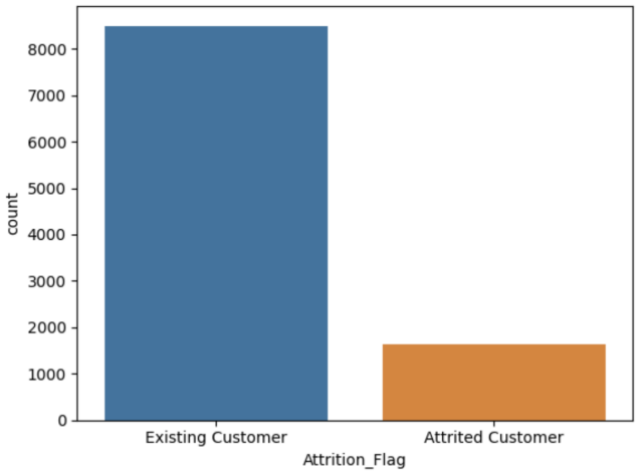


Fig. 3. Attrition Flag Histogram

It can be observed that the proportion of data samples that belong to churn customers is only 16%. This indicates that the data set is imbalanced, which can cause problems when building a prediction model. To correct this problem, the SMOTE (Synthetic Minority Over-sampling Technique) method will be used. SMOTE generates synthetic samples of the minority class (in this case, churn customers) to balance the dataset and avoid overfitting. With a balanced dataset, the prediction model can be developed with higher accuracy and reliability.

III. Methods

A. Random Forest

Random Forest is a machine learning algorithm used for classification and regression tasks. The Random Forest algorithm is based on fitting multiple classification trees to a given dataset and aggregating the predictions from all the trees. According to Breiman (2001), the algorithm starts by selecting numerous bootstrap samples from the dataset. In a typical bootstrap sample, about 63% of the original observations appear at least once, while observations in the original data that do not appear in a bootstrap sample are referred to as out-of-bag observations. For each bootstrap sample, a classification tree is constructed, but at each node, only a small number of randomly selected variables (e.g., the square root of the number of variables) are available for binary partitioning. The trees are fully grown, and each tree predicts the out-of-bag observations. The predicted class of an observation is determined by the majority vote of the out-of-bag predictions for that observation, with ties being randomly resolved [1].

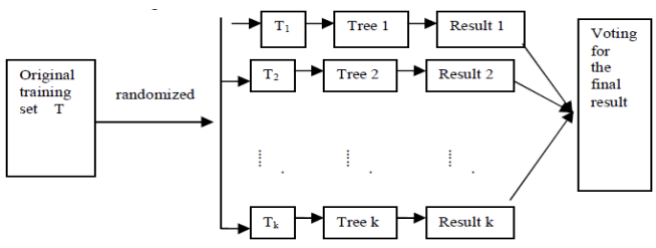


Fig. 4. Random Forest schematic [2]

In the context of Random Forest applied to classification data, it is essential to understand that the Gini index is frequently employed to determine how nodes on a decision tree branch. The formula used in this index utilizes the class and probability to determine the Gini of each branch on a node, ultimately determining which of the branches is more likely to occur. The Gini index formula is expressed as follows:

$$\text{Gini} = 1 - \sum_{i=1}^{c} p_i^2$$

In the above formula, pi denotes the relative frequency of the class you are observing in the dataset, while c represents the number of classes. By leveraging the Gini index, the Random Forest algorithm can identify and select optimal decision trees to enhance the accuracy of its predictions.

Random Forests algorithm can also use entropy as a measure to determine how nodes should branch in a decision tree. Entropy uses the probability of a certain outcome to calculate the degree of impurity at a given node. The entropy formula is more mathematically intensive than the Gini index formula, as it involves a logarithmic function. The formula for entropy is as follows:

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_2 P(x_i)$$

In the above formula H(X) is the entropy of the random variable X, P(x_i) is the probability of x_i occurring, and n is the number of possible outcomes [3].

B. Support Vector Machine

Support vector machine is a supervised machine learning algorithm that can be used for regression or classification. The main idea behind SVM is to find the best possible decision boundary that can separate the different classes in the given dataset. The decision boundary is constructed by selecting a subset of the training data, called support vectors, which are the data points closest to the decision boundary.

The goal of SVM is to maximize the margin between the decision boundary and the closest support vectors. The margin is defined as the distance between the decision boundary and the closest support vectors from each class. The optimal decision boundary is the one that maximizes the margin.

The formula below represents the classifier for the linear case,

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} y_i (x^T x_i) + b\right)$$

In the case of non-linear classification, the classifier function is as below:

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} y_i \, K(x, x_i) + b\right)$$

The kernel function K (xi, xj) is calculated from the below equation, where the dot product of feature vectors of xi and xj is taken and transformed using a kernel:

$$K(xi, xj) = \phi(xi)^T \phi(xj)$$

The most used kernel functions are:

Radial basis kernel: $K(xi, xj) = \exp\left(-\gamma\left|\left|xi - xj\right|\right|^2\right)$

Polynomial kernel: $K(xi, xj) = (xi^T xj + r)^d$

Sigmoid kernel: $K(xi, xj) = \tanh(\gamma xi^T xj + r)$ [4]

C. K-Nearest Neighbors

K-nearest neighbor (KNN) is a supervised machine learning algorithm that can be used for classification and regression tasks. In the KNN algorithm, the output of a new query instance is classified based on the K number of nearest neighbors in the training set.

The choice of K in the KNN algorithm is important because it determines how many neighbors are considered to make a classification decision. If K is too small, the classification decision may be overly influenced by the noise in the data. If K is too large, the classification decision may be biased towards the most common classes in the data. The optimal value of K can be determined by cross-validation or other model selection techniques.

In the KNN algorithm, the distance between the query instance and the training instances can be measured using various distance metrics such as Euclidean, Manhattan, and cosine distance. Additionally, the algorithm can be modified to weigh the contributions of the K nearest neighbors differently based on their distance from the query instance.

Manhattan Distance Formula: $d_1(p, q) = \sum_{i=1}^{n} |p_i - q_i|$

where p and q are two n-dimensional points.

Euclidean Distance Formula: $\sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$

where p and q are two n-dimensional points

Cosine Distance Formula: $\text{cosine\_distance}(x, y) = 1 - \frac{\langle x,y \rangle}{|x||y|}$

where x and y are vectors [5].

IV. Experiments

In preparation for developing a predictive classification model, it is necessary to process the data in a way that enables it to be used as training and testing data for the model. Therefore, this section will delve into the techniques employed in feature engineering, the algorithms utilized for constructing the classification models, and the evaluation strategies employed to measure model performance.

A. Feature Engineering

The dataset has a class distribution of 16% for customers leaving the bank and 86% for those staying. To address the class imbalance in the dataset, we are applying Synthetic Minority Over-sampling Technique (SMOTE), a widely used method for upsampling the minority class. By using SMOTE, we can generate synthetic samples for the minority class to achieve a more balanced class distribution, which can potentially improve the performance of the machine learning model.

Undersampling techniques can be used to balance the class distribution in imbalanced datasets, but it has a potential drawback of reducing the sample size of majority class and therefore potentially losing important information from the data. In cases where the majority class is under-represented, this can result in a significant reduction in the number of observations, which in turn could lead to a loss of important features and patterns that are important for the classification model.

SMOTE (Synthetic Minority Over-sampling Technique) works by generating new synthetic samples in the minority class by selecting examples that are close in the feature space. SMOTE first selects a minority class instance a at random and finds its k nearest minority class neighbors. The synthetic instance is then created by choosing one of the k nearest neighbors b at random and connecting a and b to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances a and b [6].

When dealing with categorical variables, one common technique is to use one-hot encoding, also known as dummy variable encoding. This technique involves converting each categorical value in a column to a new column and assigning a binary value (0 or 1) to indicate whether a particular value is present. This way, categorical variables can be represented as numerical data that can be used by machine learning algorithms.

In our analysis, we are utilizing one-hot encoding to convert categorical variables into a format that can be used for further analysis. Specifically, we are using the Dummy Variable Encoding technique available in the Python library pandas to create a binary representation of categorical variables in our dataset. This technique will help us to avoid assigning any numerical order or rank to the categorical variables and enable us to use them in our machine learning models.

In order to improve the performance of classification algorithms such as SVM and KNN, it is essential to scale the data. Therefore, standardization technique is applied to the data, which rescales the data by centering it around the mean and scaling it by the standard deviation. By doing so, we can ensure that all features are on the same scale and prevent certain features from dominating the others. Standardization can help to improve the accuracy and stability of these classification algorithms. We standardized the data using the StandardScaler() function from the scikit-learn library, which transforms each feature to have zero mean and unit variance.

After applying the feature engineering techniques mentioned above, the dataset has undergone significant transformations resulting in an increase of data points from 10127 rows and 21 columns to 17000 rows and 33 columns. This is due to the creation of new features through techniques such as one-hot encoding, which expands the number of columns in the dataset to capture categorical variables. The increase in data points and columns will provide a more comprehensive view of the data for modeling purposes. We can see the resulting dataset output in “Fig. 5”.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17000 entries, 0 to 16999
Data columns (total 36 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Customer_Age                    17000 non-null  int64
 1   Gender                          17000 non-null  int64
 2   Dependent_count                 17000 non-null  int64
 3   Months_on_book                  17000 non-null  int64
 4   Total_Relationship_Count        17000 non-null  int64
 5   Months_Inactive_12_mon          17000 non-null  int64
 6   Contacts_Count_12_mon           17000 non-null  int64
 7   Credit_Limit                    17000 non-null  float64
 8   Total_Revolving_Bal             17000 non-null  int64
 9   Avg_Open_To_Buy                 17000 non-null  float64
 10  Total_Amt_Chng_Q4_Q1            17000 non-null  float64
 11  Total_Trans_Amt                 17000 non-null  int64
 12  Total_Trans_Ct                  17000 non-null  int64
 13  Total_Ct_Chng_Q4_Q1             17000 non-null  float64
 14  Avg_Utilization_Ratio           17000 non-null  float64
 15  Education_Level_College         17000 non-null  uint8
 16  Education_Level_Doctorate       17000 non-null  uint8
 17  Education_Level_Graduate        17000 non-null  uint8
 18  Education_Level_High School     17000 non-null  uint8
 19  Education_Level_Post-Graduate   17000 non-null  uint8
 20  Education_Level_Uneducated      17000 non-null  uint8
 21  Education_Level_Unknown         17000 non-null  uint8
 22  Marital_Status_Divorced         17000 non-null  uint8
 23  Marital_Status_Married          17000 non-null  uint8
 24  Marital_Status_Single           17000 non-null  uint8
 25  Marital_Status_Unknown          17000 non-null  uint8
 26  Income_Category_$120K +         17000 non-null  uint8
 27  Income_Category_$40K - $60K     17000 non-null  uint8
 28  Income_Category_$60K - $80K     17000 non-null  uint8
 29  Income_Category_$80K - $120K    17000 non-null  uint8
 30  Income_Category_Less than $40K  17000 non-null  uint8
 31  Income_Category_Unknown         17000 non-null  uint8
 32  Card_Category_Blue              17000 non-null  uint8
 33  Card_Category_Gold              17000 non-null  uint8
 34  Card_Category_Platinum          17000 non-null  uint8
 35  Card_Category_Silver            17000 non-null  uint8
dtypes: float64(5), int64(10), uint8(21)
memory usage: 2.3 MB
```

Fig. 5. After Feature Engineering

B. *Modelling*

After performing feature engineering. To evaluate the performance of our models, we employed k-fold cross-validation technique. We applied this method to Random Forest, Support Vector Machine, and K-Nearest Neighbors models, and obtained their respective evaluation results. The models used in the analysis were initially executed using their default parameters.

Classifier	CrossVal Score Means	CrossVal errors
RandomForestClassifier	0.978412	0.002238
SupportVectorMachine	0.946412	0.004297
KNeighborsClassifier	0.913294	0.004481

```
                    Algo  CrossVal_Score_Means  CrossValerrors
0  RandomForestClassifier              0.978412        0.002238
1    SupportVectorMachine              0.946412        0.004297
2    KNeighborsClassifier              0.913294        0.004481
```

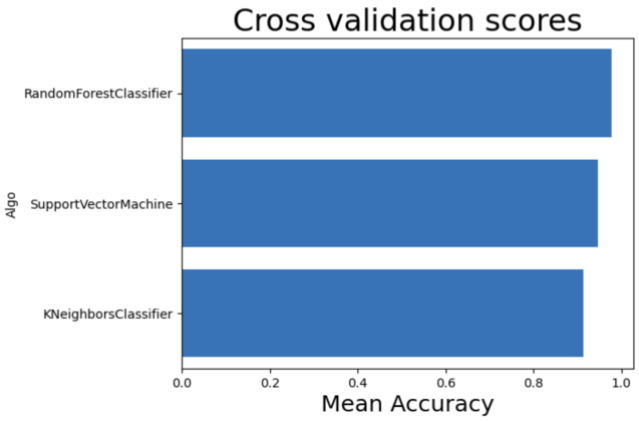
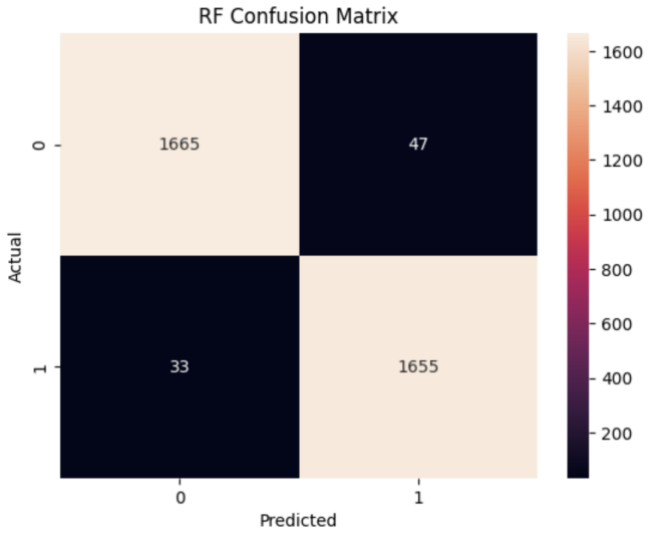
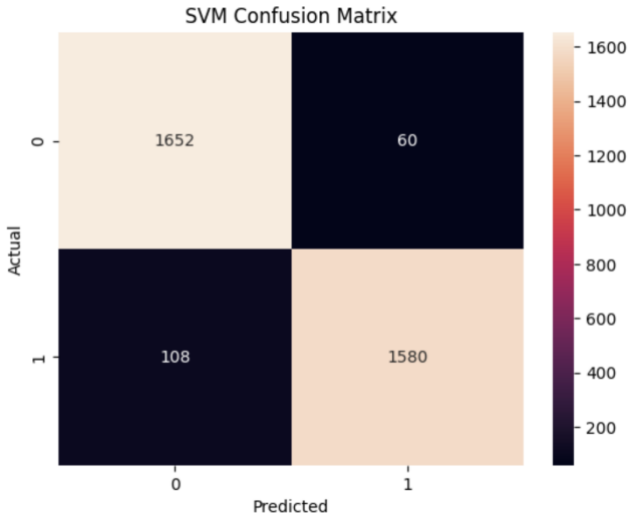
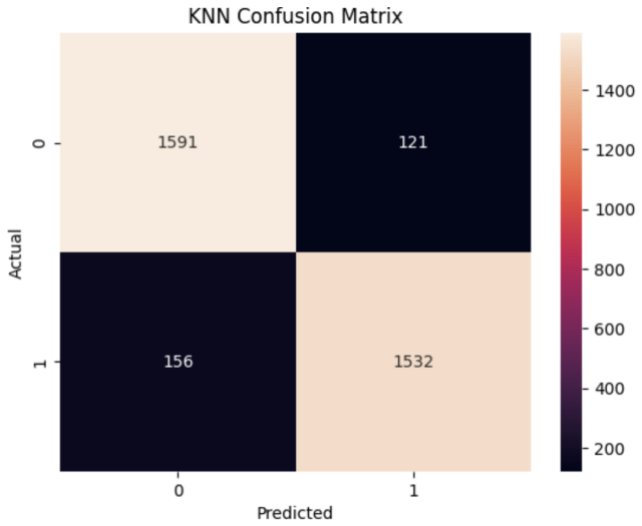


Fig. 6. *Cross validation Scores*

Confusion Matrix with default parametes







C. *Hyper Parameter tuning*

Hyperparameter tuning refers to the process of finding the optimal combination of hyperparameters for a given machine learning algorithm to achieve the best performance on the test set. Hyperparameters are adjustable parameters that control the learning process and affect the behaviour of the machine

learning model. Tuning these hyperparameters can greatly impact the performance of the model.

There are various techniques to perform hyperparameter tuning, such as grid search, random search, and Bayesian optimization. Grid search involves trying out all possible combinations of hyperparameters within a predefined range, while random search involves randomly sampling hyperparameters from the predefined range. Bayesian optimization involves using a probabilistic model to predict the performance of different hyperparameters and selecting the best ones based on the model's predictions [7].

In our study, we used grid search to tune the hyperparameters for the Random Forest, SVM, and KNN models. We defined a range of values for each hyperparameter and tried out all possible combinations using cross-validation to find the optimal set of hyperparameters that would give the best performance on the test set.

After tuning the parameters, the result were show in the below figures for Random Forest classifier.

```
Random Forest Best Params: {'criterion': 'entropy', 'max_depth': 150, 'n_estimators': 1700}
```

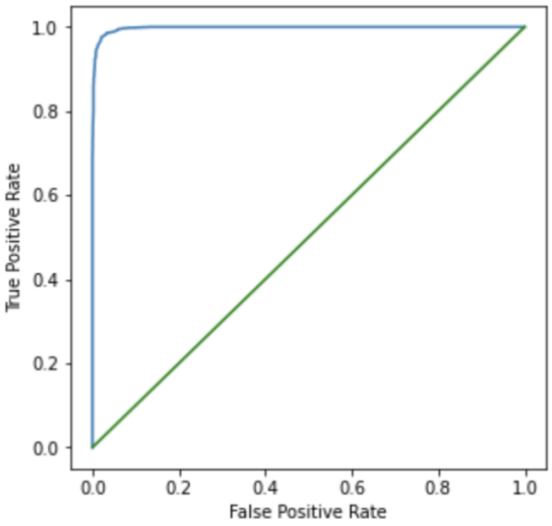
```
RandomForestClassifier
Accuracy: 0.9758823529411764
              precision    recall  f1-score   support

           0       0.91      0.93      0.92      1712
           1       0.93      0.91      0.92      1688

    accuracy                           0.92      3400
   macro avg       0.92      0.92      0.92      3400
weighted avg       0.92      0.92      0.92      3400

[[1665   47]
 [  35 1653]]
```

ROC curve - RandomForestClassifier



There is very minute decrease in accuracy as the range selected for tuning is very less but Random Forest give better accuracy compared to other two models

We can see the feature importance from the random forest classifer model in the figures below.

```
Total_Trans_Amt                0.188288
Total_Trans_Ct                 0.183916
Total_Revolving_Bal            0.095357
Total_Ct_Chng_Q4_Q1            0.082961
Avg_Utilization_Ratio          0.059218
Total_Relationship_Count       0.057431
Total_Amt_Chng_Q4_Q1           0.054479
Credit_Limit                   0.032993
Avg_Open_To_Buy                0.032639
Customer_Age                   0.027280
Marital_Status_Married         0.022974
Months_on_book                 0.019295
Months_Inactive_12_mon         0.017710
Contacts_Count_12_mon          0.013923
Education_Level_Graduate       0.013740
Marital_Status_Single          0.012338
Dependent_count                0.011588
Education_Level_High School    0.010817
Income_Category_$60K - $80K    0.008894
Gender                         0.008548
Education_Level_Uneducated     0.007295
Income_Category_$80K - $120K   0.007120
Income_Category_$40K - $60K    0.006971
Education_Level_Unknown        0.005438
Income_Category_Less than $40K 0.005067
Marital_Status_Unknown         0.004687
Income_Category_Unknown        0.003734
Education_Level_Post-Graduate  0.001868
Education_Level_Doctorate      0.001614
Card_Category_Silver           0.001343
Card_Category_Gold             0.000430
Card_Category_Platinum         0.000044
dtype: float64
```

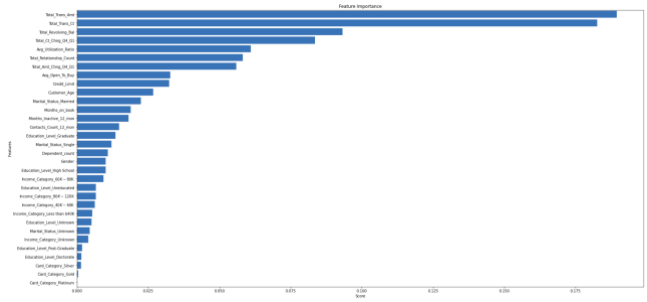


Fig. 7. *Feature Importance plot and values*

After tuning the parameters, the result was show in the below figures for Support Vector Machine.

```
SVM Best Params: {'C': 10.0, 'gamma': 0.1, 'kernel': 'rbf'}
```

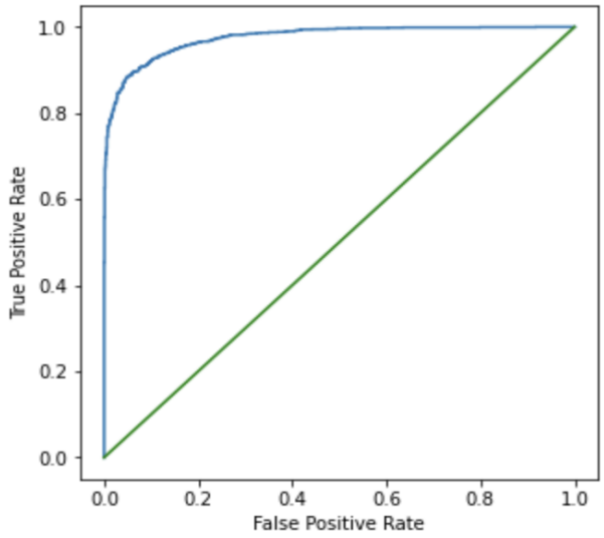
```
SupportVectorMachine
Accuracy: 0.8917647058823529
[[1681   31]
 [ 337 1351]]
              precision    recall  f1-score   support

           0       0.83      0.98      0.90      1712
           1       0.98      0.80      0.88      1688

    accuracy                           0.89      3400
   macro avg       0.91      0.89      0.89      3400
weighted avg       0.90      0.89      0.89      3400
```

For SVM after the model accuracy decreased. If we tune with random search cv and the selecting the parameter ranges for grid search cv will give us better results

ROC curve - KNN



After tuning the parameters, the result was show in the below figures for SVM. This model got better after tunning params.

```
KNN Best Params: {'metric': 'manhattan', 'n_neighbors': 2, 'weights': 'uniform'}
```

```
K-KNeighborsClassifier
Accuracy: 0.9441176470588235
[[1628   84]
 [ 106 1582]]
              precision    recall  f1-score   support

           0       0.94      0.95      0.94      1712
           1       0.95      0.94      0.94      1688

    accuracy                           0.94      3400
   macro avg       0.94      0.94      0.94      3400
weighted avg       0.94      0.94      0.94      3400
```

ROC curve - RandomForestClassifier

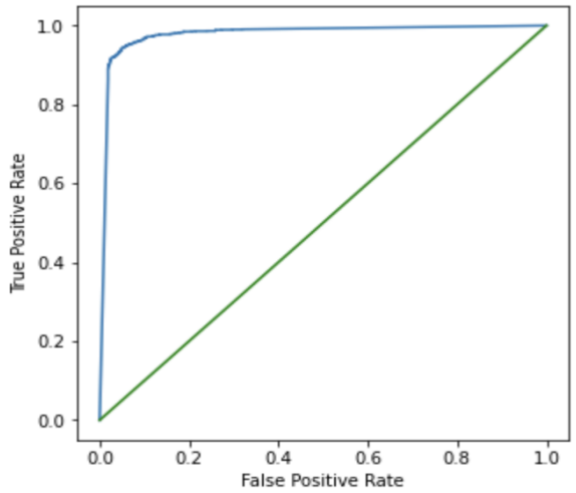


Figure below is the ROC scores of models after parameter tuning

	Algo	roc_auc_score
0	RandomForestClassifier	0.975906
1	SupportVectorMachine	0.891124
2	KNeighborsClassifier	0.939537

V. Reflection

In this study, our aim was to predict customer churn in a bank's credit card service. To achieve this, we analysed a dataset consisting of 10,000 observations containing relevant features such as age, salary, marital status, credit card limit, etc. We performed data pre-processing and applied three classification models: Random Forest, Support Vector Machine, and K-Nearest Neighbors, using 5-fold cross-validation. We improved the accuracy of these models by tuning their hyperparameters and evaluated their performance using ROC & AUC curves and confusion matrices. Our analysis revealed that Random Forest had the strongest predictive ability. Furthermore, we identified three features that had the greatest impact on customer churn prediction using this model. Overall, our study provides valuable insights into predicting customer churn in the credit card service industry.

Although grid search cv improved the performance of our models, we observed a slight dip in the accuracy due to the limited range of parameter values we could search, as our computational power was limited. However, it is important to note that hyperparameter tuning and feature selection are not guaranteed to always improve the performance of the models, as the optimal values of hyperparameters and feature importance depend on the specific dataset and the problem at hand.

To achieve better performance, we suggest using a combination of random search cv and grid search cv to optimize the models. Additionally, feature selection based on feature importance from random forest can also be used to further enhance the performance of the models.

The dataset provided to us was imbalanced, and we had to generate synthetic data points to balance the proportion of existing and attrited customers. However, this issue could have been avoided if a balanced dataset was provided. Furthermore, we used three models to make predictions. Nevertheless, utilizing ensemble learning to combine the strengths of multiple models can result in improved performance.

References

[1] Cutler, D. R., Edwards, T. C., Beard, K. H., Cutler, A., Hess, K. T., Gibson, J., & Lawler, J. J. (2007). Random forests for classification in ecology. *Ecology*, *88*(11), 2783–2792. https://doi.org/10.1890/07-0539.1.

[2] Schott, M. (2020, February 27). *Random Forest Algorithm for Machine Learning*. Medium. Retrieved March 17, 2023, from https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb.

[3] Liu, Y., Wang, Y., & Zhang, J. (2012). New Machine Learning Algorithm: Random forest. *Information Computing and Applications*, 246–252. https://doi.org/10.1007/978-3-642-34062-8_32.

[4] ÜNLÜ, K. D. (2021). Predicting credit card customer churn using support vector machine based on Bayesian optimization. *Communications Faculty Of Science University of Ankara Series A1Mathematics and Statistics*, *70*(2), 827–836. https://doi.org/10.31801/cfsuasmas.899206.

[5] *Distance metrics used in KNN* . KDnuggets. (n.d.). Retrieved March 17, 2023, from https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html.

[6] He, H., & Ma, Y. (n.d.). In Imbalanced learning: Foundations, algorithms, and applications (p. 47). essay.

[7] GERON, A. (2019). End-to-End Machine Learning Project. In Hands-on machine learning with scikit-learn, Keras, and tensorflow: Concepts, tools and techniques to build Intelligent Systems (2nd ed., p. 79). essay,