Comparing Machine Learning Classification Models for the Prediction of Customer Churning

Fitzroy Meyer-Petgrave
Department of Data Science
Teesside University
Middlesbrough, England
a0384858@live.tees.ac.uk

***Abstract*— Predicting which customer may stop using a service within is period of time is a business strategy that can save time, money and company image. Various classification models have been designed to carry out this task but I believe there can be a more suitable model for every given task based on the type of dataset and classification problem.**

This project aims to get the best model for predicting customer churn from a given dataset by comparing 3 suitable classification models; K-Nearest Neighbor, Support Vector Machine and Random Forest. The results of this comparison showed that the Random Forest classifier model performed best in predicting customer from our given bank dataset of real clients.

I. Introduction

In business, customer churn is a term used to describe a situation where customers or subscribers stop doing business with a company or service. It is alternatively referred to as customer attrition. A measure of how many customers churn within a given period of time is known as churn rate, often evaluated monthly, quarterly, or annually.

Customers may churn for various reasons such as bad customer service, not enough value, no brand loyalty, etc. Whatever the reason may be, this is a nightmare to every business. A high customer churn rate impedes business growth by decreasing revenue and increasing cost of acquiring new customers.

Study has shown that it is 5 to 25 times more expensive to get new customers than retaining existing ones (Saleh, K., 2020). Therefore companies constantly try find ways to prevent this or bring it to the barest minimum. If a company is able to predict which customer is about to churn, they can take necessary steps to prevent it.

Some machine learning algorithms have been used to create models that can make this predictions given client data of both churned and existing customers. The client attributes are learned by the machine and used to make predictions based on characteristics and churning history of the clients.

Prediction Models Compared

Making predictions on which customer is likely to churn within a given period of time is a classification problem because the output variable is between 2 categories – “churn” or “not churn”. So for this research, 3 classification models suitable for categorising outputs will be compared.

A. K-Nearest neighbour (KNN)

The predictive model designed using KNN makes prediction by finding the distances between a query and all the examples in our dataset, selecting the specified number examples (K) closest to the query, then votes for the most frequent label.

B. Support Vector Machine (SVM)

This model can solve classification problems which may be linear or non-linear. So in this case the SVM algorithm creates a line or a hyperplane which separates the data into classes.

C. Random Forest

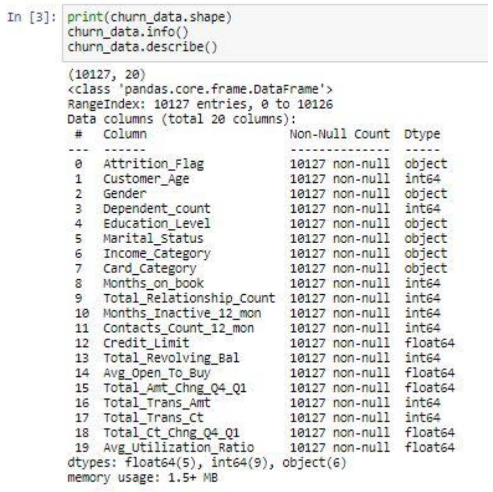
This model will make predictions by building multiple decision trees and merging them together to get a more accurate and stable prediction like a decision tree. However, random forest adds additional randomness to the model, while growing the trees.

Before we begin to make predictions, we want to explore our dataset and prepare it to make it readable by the selected models

II. Exploratory Data Analysis

The dataset we are exploring is a real bank data of 10,127 customers that use its credit card facilities with 21 features for each customer, including a description stating if a customer has churned or not.
Our dataset was originally sourced from;
https://leaps.analyttica.com/sample_cases/1
and publicly available for download at;
https://www.kaggle.com/sakshigoyal7/credit-card-customers.

Description of Attributes



```
In [3]: print(churn_data.shape)
        churn_data.info()
        churn_data.describe()

(10127, 20)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10127 entries, 0 to 10126
Data columns (total 20 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Attrition_Flag            10127 non-null  object 
 1   Customer_Age              10127 non-null  int64  
 2   Gender                    10127 non-null  object 
 3   Dependent_count           10127 non-null  int64  
 4   Education_Level           10127 non-null  object 
 5   Marital_Status            10127 non-null  object 
 6   Income_Category           10127 non-null  object 
 7   Card_Category             10127 non-null  object 
 8   Months_on_book            10127 non-null  int64  
 9   Total_Relationship_Count  10127 non-null  int64  
 10  Months_Inactive_12_mon    10127 non-null  int64  
 11  Contacts_Count_12_mon     10127 non-null  int64  
 12  Credit_Limit              10127 non-null  float64
 13  Total_Revolving_Bal       10127 non-null  int64  
 14  Avg_Open_To_Buy           10127 non-null  float64
 15  Total_Amt_Chng_Q4_Q1      10127 non-null  float64
 16  Total_Trans_Amt           10127 non-null  int64  
 17  Total_Trans_Ct            10127 non-null  int64  
 18  Total_Ct_Chng_Q4_Q1       10127 non-null  float64
 19  Avg_Utilization_Ratio     10127 non-null  float64
dtypes: float64(5), int64(9), object(6)
memory usage: 1.5+ MB
```

From the dataset descriptions, there are no null values, so we can easily visualize the relationship between the 'attrition_flag'(churn) and other variables to get more insight.

To create successful visualizations on our dataset we have to consider the data types of the different variables which are;

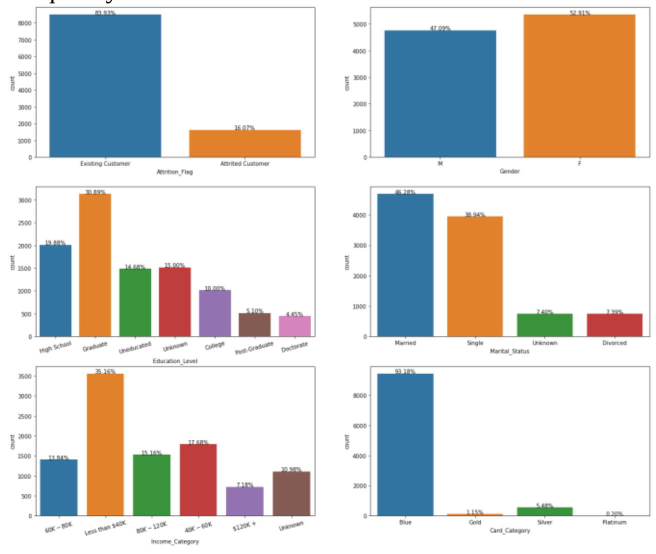
- Categorical(6) - 6 objects
- Numerical(14) - 5 float and 9 integers

So we would visualize the frequencies for the categorical and the relationship they may have with attrition using bar charts.

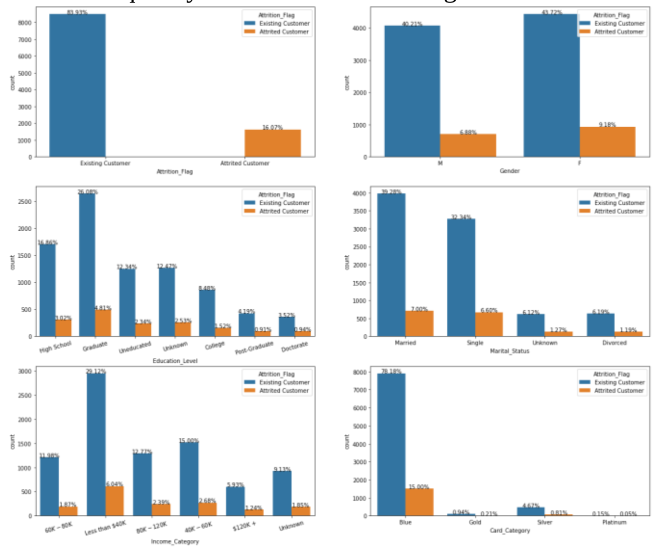
Next we visualize the frequencies of the numerical variables using histogram.

A. *Categorical Features Visualization*

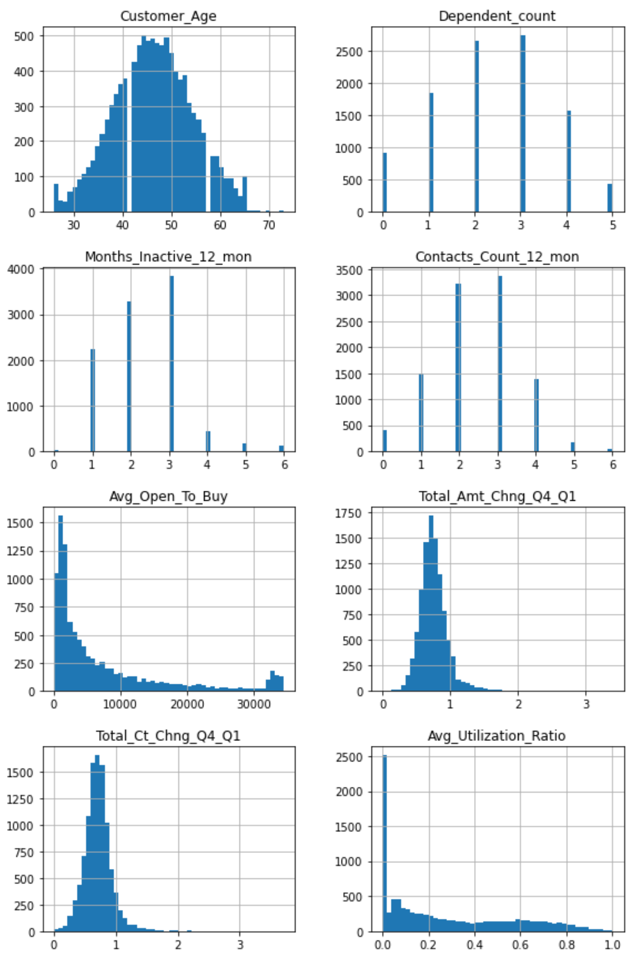
Frequency of Attributes

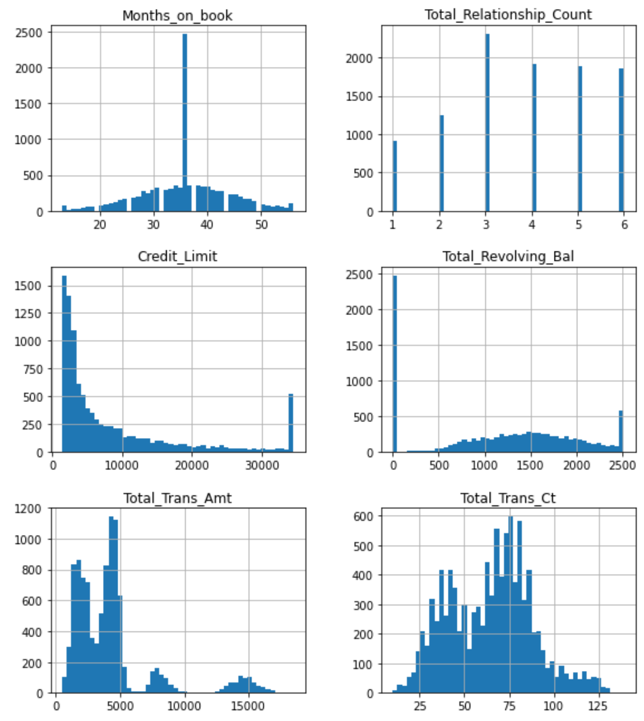


Feature frequency of churned and existing customers

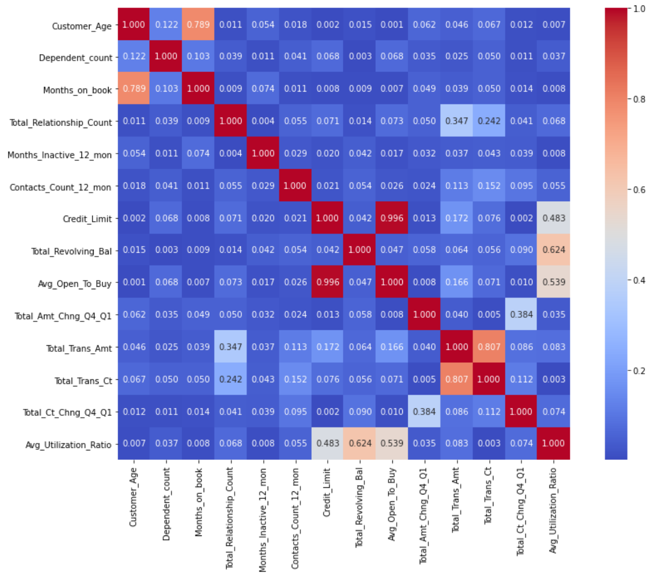


B. *Numerical Features Visualisation*





C. *Numerical Correlation*



From the visualisations, we can deduce the following

1. Our dataset for churned and existing customers is unbalanced – 16% and 84% respectively
2. Customer churning decreases as education level increases
3. Customers who earn between 60k and 80k have the highest churners

III. Feature Engineering

We need to label all the categorical variables to make them machine-readable. So we do this by encoding numerical values to each category in the different categorical variables. First we separate the categorical variables into their variable types which are nominal and ordinal.

Nominal variables have two or more categories which do not have any kind of order associated with them. In our dataset we have;

1. Attrition_Flag
2. Gender
3. Marital_status (contains 'unknown' values)

Ordinal variables have "levels" or categories with particular order associated with them. Ordinal variables in our dataset are;

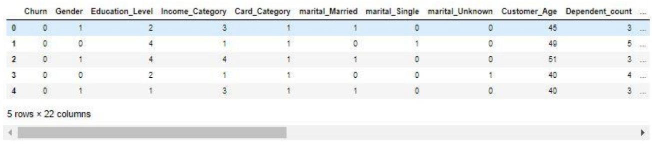
1. Education_level (contains 'unknown' values)
2. Card_category
3. Income_category (contains 'unknown' values)

Unknown values have not been removed to preserve the other important attributes of those clients which would help in the machine learning.

A. *Mapping*

The Nominal variables would be mapped using binary function, except the 'Marital_status' which contains 'unknown' values as shown in the frequency distribution. Therefore we map 'marital' using dummy function

Ordinal would be encoded by labelling them according to their respective levels using encoder. The 'unknown' values in the 'income_category' and 'education_level' variables would be labelled 'zero'.



	Churn	Gender	Education_Level	Income_Category	Card_Category	marital_Married	marital_Single	marital_Unknown	Customer_Age	Dependent_count	...
0	0	1	2	3	1	1	0	0	45	3	...
1	0	0	4	1	1	0	1	0	49	5	...
2	0	1	4	4	1	1	0	0	51	3	...
3	0	0	2	1	1	0	0	1	40	4	...
4	0	1	1	3	1	1	0	0	40	3	...

5 rows × 22 columns

B. *Feature Scaling*

As we can see from the table above, the values vary largely which would pose a challenge if we want to get good results from our training data. Therefore we use Feature scaling to normalize the data. It is essential for machine learning algorithms that calculate distances between data.

By applying feature scaling the range of all features would be normalized so that each feature contributes approximately proportionately to the final distance.

For this project, I would consider 3 scaling methods:

1. Standard Scaler
2. Power Transformer Scaler
3. MinMax Scaler

First, I split the data into Train and Test data, then apply the transformations.

C. *Decomposition*

Our dataset has many variables of which some of them might be irrelevant for our model to make prediction, which might cause us to have noisy data.

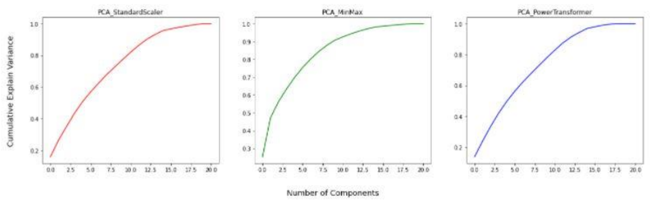
We need to extract the most important features thereby reducing redundancy and improving the execution time for our program. To do this we use **Principal Component Analysis (PCA).**

This PCA is meant to reduce the dimension of the dataset by preserving the maximal data variance in the dataset using our scaled training data;

X_train_s , **X_train_pt** , and **X_train_mm**

Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation "Fig. 1", even at the beginning of a sentence.

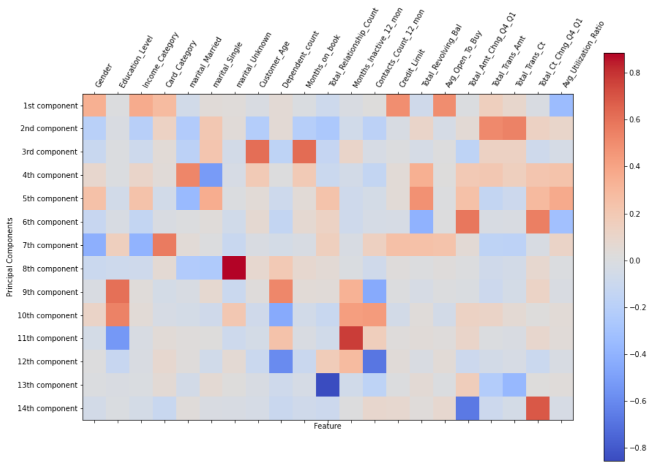
PCA Results



The above curves are the PCA results for 3 dataset we derive from Standard Scaler, MinMax and Power Transformer.

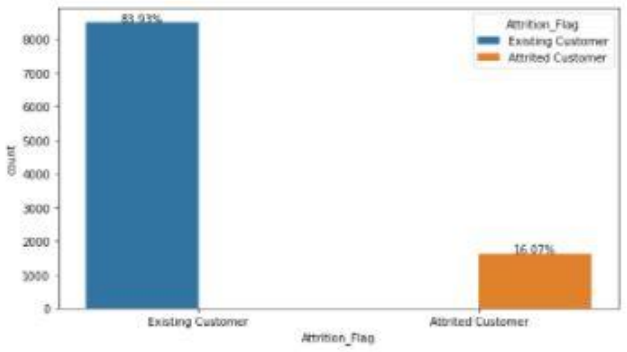
Then if we trace 0.9(90%) from the variance axis to the curve, the values for the number of components would be;

PCA_StandardScaler	14
PCA_MinMax	10
PCA_PowerTransformer	13



The heat map above was plotted using the result from the Standard Scaler – 14. This was used to ensure we get the most out of the data and reduce loss of valuable dataset properties. The map shows a high level of correlation among the principal components and the initial features, therefore they are good to make our predictions.

D. *Handling Imbalanced Dataset*



The bar chart above shows that our dataset is imbalanced with 84% existing customers and 16% churned.

So we handle this issue by applying KFold cross-validation and over-sampling (SMOTE) method to the dataset to ensure that we get the same split for each recall. This will balance the dataset by increasing the size of the rare sample (i.e. churned customers).
We use the imba_pipe function for KNN and SVM
Use imba_pipe_forest function for Random Forest because this model does not use scaled data or dimension reduction data.

IV. EXPERIMENT

First we generate an array of RFC recall score after KFold and resample, and then we use GridSearchCV to find the best hyperparameters and fit the model on our training set.

GridSearchCV is a function that helps to loop through predefined hyperparameters and fit the model on our training set. So all 3 models would be run first without and then with GridSearchCV.

Note: The standard scaled data was used for KNN and SVM, while unscaled data was used for Random Forest because data are not necessarily scaled when using this model.

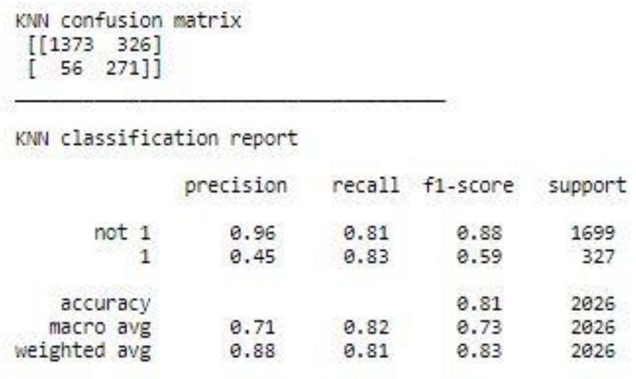
V. RESULTS

	KNN	**SVM**	**RFC**
Mean of array RFC recall score after KFold and resample	0.85	0.8	0.88
Grid best train recall score using GridSearchCV	0.86	0.81	0.91
Grid best test recall score using GridSearchCV	0.83	0.80	0.87

VI. EVALUATION METRICS

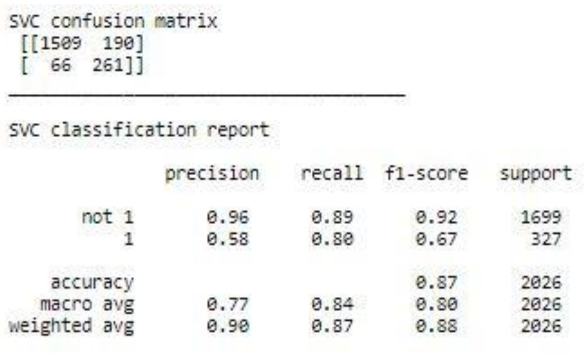
All 3 models are evaluated by printing out the respective confusion matrix and classification reports as shown below

A. *K-Nearest Neighbour (KNN)*



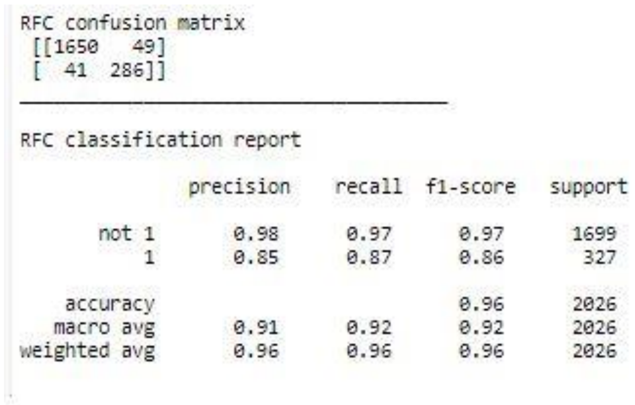
```
KNN confusion matrix
 [[1373  326]
 [  56  271]]
___________________________________________

KNN classification report

              precision    recall  f1-score   support

       not 1       0.96      0.81      0.88      1699
           1       0.45      0.83      0.59       327

    accuracy                           0.81      2026
   macro avg       0.71      0.82      0.73      2026
weighted avg       0.88      0.81      0.83      2026
```

B. *Support Vector Machine*



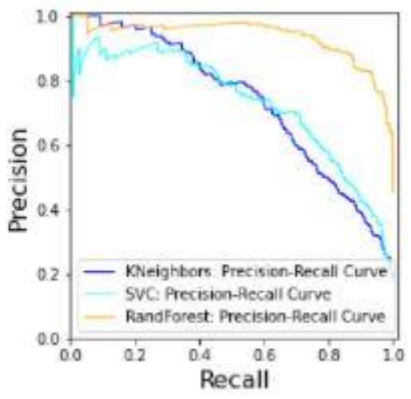
```
SVC confusion matrix
 [[1509  190]
 [  66  261]]
___________________________________________

SVC classification report

              precision    recall  f1-score   support

       not 1       0.96      0.89      0.92      1699
           1       0.58      0.80      0.67       327

    accuracy                           0.87      2026
   macro avg       0.77      0.84      0.80      2026
weighted avg       0.90      0.87      0.88      2026
```

C. *Random Forest Classifier*



```
RFC confusion matrix
 [[1650   49]
 [  41  286]]
________________________________________

RFC classification report
              precision    recall  f1-score   support

       not 1       0.98      0.97      0.97      1699
           1       0.85      0.87      0.86       327

    accuracy                           0.96      2026
   macro avg       0.91      0.92      0.92      2026
weighted avg       0.96      0.96      0.96      2026
```

D. *Precision Recall Curve*



E. *Receiver Operating Characteristic curve (ROC)*

ROC curves or Receiver Operating Characteristic curves illustrate the performance of a binary classifier. It is created by plotting the true positive rate (TPR) (or recall) against the false positive rate (FPR).
ROC curves on the X-axis show a classifier's False Positive Rate so that would go from 0 to 1.0, and on the Y-axis they show a classifier's True Positive Rate so that will also go from 0 to 1.0.
ROC curves are very help with understanding the balance between true-positive rate and false positive rate.

F. *AUC*

Model	**AUC Score**
KNN	0.91
SVM	0.92
Random Forest	0.99

VII. CONCLUSION

Random Forest has the best model compared to the other with all metrics score (precision, recall, f1, accuracy) has shown at least 85% accuracy. Also, Random Forest has AUC score of 99% that shows the model is learning the data well enough.

KNN and Support Vector Machine models are improving in their recall score, and have significant reduction in their precision score. This is important as we must be aware by getting a high recall score and a low precision score could be a sign that the model might simply be predicting customer that wants to churn their credit card and not customers that have churned already (increasing in False Positive).

VIII. REFERENCES

[1] Saleh, K. (2020). “Customer Acquisition vs. Retention Costs – Statistics and Trends”, Invesp, 11 November. Available at: https://www.invespcro.com/blog/customer-acquisition-retention/ (Accessed: May 5, 2021)

[2] Frankenfield, J. (2021). "Churn Rate" Investopedia US, 1 March. Web.

[3] Yıldırım, S. (2020). “K-Nearest Neighbors (kNN) — Explained”, Towards Data Science, 29 Feb. Available at: https://towardsdatascience.com/k-nearest-neighbors-knn-explained/ (Accessed: April 22, 2021).

[4] Yıldırım, S. (2020). “Support Vector Machine — Explained”, Towards Data Science, 7 Feb. Available at: https://towardsdatascience.com/support-vector-machine-explained-8d75fe8738fd/ (Accessed: April 22, 2021).

[5] Mujtaba, H. (2020). “Hyperparameter Tuning with GridSearchCV”, Great Learning, 29 Sept. Available at: https://www.mygreatlearning.com/blog/gridsearchcv/ (Accessed: May 6)

[6] F. Yamin, (2021). “Customer Credit Card Prediction”, *Kaggle*, March 2021. Available at: https://www.kaggle.com/yaef22/ (Accessed: April 20)