TABLE OF CONTENTS

Title **Page No.**

List of Figures

1.0.INTRODUCTION

The selected dataset provided in this report is gathered from a telecommunications company in line with its customer retention program. The focus of the program is to examine the total number of existing and churned customers of the company. The resultant information will further be used to predict the number of customers that are most likely to churn. The company is therefore interested in building a classification model for churn prediction, which will help to achieve the objective of minimizing the churn rate and consequently enhance customer retention. Identifying the factors that would likely cause a customer to churn, will therefore be of key importance.

The dataset is stored in a file titled, *"Telco_Churn.csv".*** Relevant information about the ***"Telco_Churn.csv"*** dataset can be found in the appendix of this report**

2.0.CRITICAL ANALYSIS

The *Telco_Churn* dataset represents a classification problem. The steps taken to construct a classification model includes Data Preparation, Model Hyperparameter Tuning, Model Evaluation, Model Feature Selection and Model Comparison.

2.1. Data Preparation

The dataset (***Telco_Churn.csv***) was first loaded into pandas (a data analytics library in python) useful for data manipulation, imported as ***pd***. This process supports flexibility as many datatypes can be read using the pandas function (*pd.read_csv*) to create a dataframe. The following steps were further taken to prepare the data:

- All columns in the dataset were displayed to see the necessary components needed to build a good model. The pandas function (*pd.set_option)* was used to achieve the display of these columns. The dataframe with the name "dataset" was printed to check for the number of rows and columns, non-null count (i.e., missing data) and datatypes (object, float or int). The dataset consisted of 7032 rows and 20 columns (variables). The range of the numerical features in the dataset was also analysed for notable differences to know whether there is a need for normalization or not.
- A total of fourteen (14) categorical columns were converted to numerical columns using both the converter function and the map function. This is necessary because the presence of objects or texts would alter the algorithm learning process.
- Two (2) other nominal columns (*InternetService* and *PaymentMethod*) were read into the pandas function (*pd.get_dummies*) to divide the columns into different parts and assign numeric values (1 or 0) to each part. This is necessary because of the presence of three or more textual labels that cannot be ordered mathematically in each of these columns.
- The dummy columns and the entire dataset were further passed into a new dataframe named *final_data*.
- The variables in *"final_data"* was then divided into features set, "X" (independent variables) and label set, "Y" (target variable) by dropping the target column from the list of columns in the dataset and defining it as a label set.

- The numerical features were further normalized because of the wide gap in the range of these features. The *StandardScalar()* function imported from scikit learn library was used to achieve this by transforming the data in a way that all columns have a mean of zero and variance of one. This resulted in values between -1 and +1 stored in "*X_scaled".*

2.2. Model Hyperparameter Tuning (Random Forest Model)

The normalized dataset was passed into the random forest classifier model. The *pipeline* function imported from imblearn open-source library, shows the sequence of steps required to be implemented on the training set which are as follows:

- Balancing of the training set using the synthetic minority oversampling technique (SMOTE) which calculates the nearest points among the minority class and adds synthetic samples to the minority class to ensure that the artificial sets are not entirely different from the training set. Training set must be balanced to ensure balanced learning while the test set is left unbalanced to depict the true picture of the dataset.
- Classification of the balanced training set was based on the entropy criterion, which uses variables with the highest information gain score to split the data. The total amount of variables in the dataset is further regulated with the use of *"max_depth",* which takes the square root of the variables for classification. Also, *"random_state"* is set to 1 (can be any figure), to ensure uniformity of result when code is re-run.

The hyperparameter (i.e., parameter with unknown value) here is the number of decision trees needed to build the random forest model. This hyperparameter is represented by *"n_estimators"* and a series of numbers from 100 – 400 was used to determine the optimal tree size that gives the best model score. Due to the presence of this hyperparameter, a good model should not be evaluated on a single test set (i.e., the use of a single decision tree is unreliable). Therefore, a 5-fold cross validation was implemented using a grid search function *"GridSearchCV"* imported from Scikit learn library. Grid search is a mechanism for performing hyperparameter tuning, where different tree sizes are used to make predictions for the random forest model. The dataset was trained using *"gd_sr.fit"* and the model that gave the best score (0.598) was selected with its corresponding tree size (n_estimators - 200). The random forest classifier model with the best score and corresponding tree size (200) was further tuned by passing it into a random forest classifier *"rfc.fit".*

2.3. Model Evaluation (Random Forest Model)

The goal of this telecommunication company is to evaluate and predict the number of churned customers as a guide for the recently launched customer retention program. However, there are two major evaluation metrics to assess (False Positive and False Negative Prediction errors), which further determines how the overall model will be built.

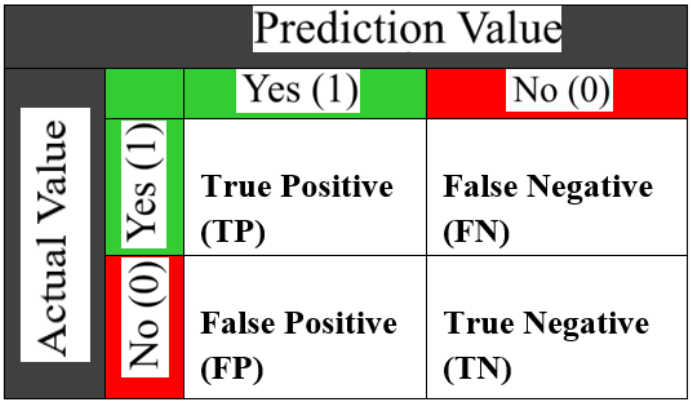

Figure 1. Confusion Matrix

- **False Positive (FP):** This refers to a scenario where an existing or active customer is predicted to have churned.
- **False Negative (FN):** This refers to a scenario where a churned customer is predicted to still be an active customer

In this case, the False Negative (FN) scenario is more fatal because the goal of the company is to detect the number of churned customers; therefore, it is important to minimize omissions as much as possible. A False Negative (FN) prediction error would mean that the company is still taking records and acknowledging the existence of customers who no longer use their products and services i.e. churned customers. This needs to be reduced to a bare minimum in the prediction model for model accuracy. Therefore, ***"recall"*** scoring metric is used in this model to minimize False Negatives. The higher the recall score, the lower the number of False Negatives and vice versa.

2.4. Model Feature Selection

It is important to minimize the number of variables needed to build a model. Although, the random forest model is rarely prone to overfitting. Variable selection is done in relation to its level of significance, and this can be displayed by printing the features importance list (*"featimp").*

The features importance list returned 24 variables in order of their significance. The top three variables (listed below) with the highest scores were chosen to redefine the feature set.

TotalCharges	0.188454
MonthlyCharges	0.174930
Tenure	0.164157

The redefined feature set was further normalized to ensure a balance in the range of the numerical features. The random forest classifier was tuned using the selected significant variables and the model was rebuilt following the sequence of the previous steps. Upon completion of the algorithm learning process, the best score for the random forest model was **"0.564"** with a corresponding decision tree size (*n_estimator*) of **"100".**

2.5. Model Hyperparameter Tuning (Support Vector Classification)

The Support Vector Classifier (SVC) model aims at solving an optimization problem by maximizing the margin (hyperplane) between data points. The *pipeline* function imported from imblearn, showing the sequence of steps to be implemented on the training set is used in the construction of the support vector classifier model. Synthetic Minority Oversampling Technique (SMOTE) is also used to balance the training set, while the dataset is classified to determine what side of the hyperplane the data points fall on.

There are two hyperparameters in the support vector classifier model which are:

- The kernel function to use (linear, polynomial, radial basis function (rbf), sigmoid)
- The regularization parameter "C" (i.e., the amount of permissible slack data points within and beyond the margin).

A kernel function (*classification__kernel*) is simply used to transform data that is not linearly separable from an n-dimensional space to a high dimensional space. Although, domain knowledge can be used to determine which kernel function to apply, it is great to run the different types of kernel functions through grid search, to see which kernel function works best for the model objective.

The regularization parameter "C" (*classification__C*) is used to avoid overfitting and allow for model generalization. It regularizes the number of slack data points allowed outside or within the margin. Therefore, the larger the regularization parameter, the lower the number of allowed slack data points and vice versa. The use of the regularization parameter makes the support vector classifier model a soft margin model.

The hyperparameters can further be tuned using *"GridSearchCV()"* to determine the function and parameter that works best for the support vector classifier model**.**

2.6. Model Evaluation (Support Vector Classifier Model)

The evaluation metric for this model is based on the minimization of false negative prediction error because the focus is to reduce the omission of churned customers as much as possible. Therefore, the ***"recall"*** scoring metric is implemented in the support vector classifier model.

The variables in the SVC model were trained and the printed result is as follows:

{'classification__C': 0.001, 'classification__kernel': 'poly'}
0.9839486172241259

Interpretation

The best model performance score is **"0.983**" and the regularization parameter (**0.001**) shows the permission of a large number of slack data points outside or within the margin and that the sum of slack distances is large. Also, the best fitting kernel function is the polynomial kernel function which is mostly associated to high dimensional and dense data.

2.7. Model Comparison

The models constructed for the *Telco_Churn* classification problem will be compared based on two factors:

- **Model Performance:** The random forest model recorded a performance score of **"0.564"** while the support vector classifier model recorded a performance score of **"0.983".** This shows that the SVC model has done a great job in minimizing the false negative prediction error by over 90%, while the random forest model barely minimized false negatives by about 50%. Therefore, the SVC model is a better performing model.

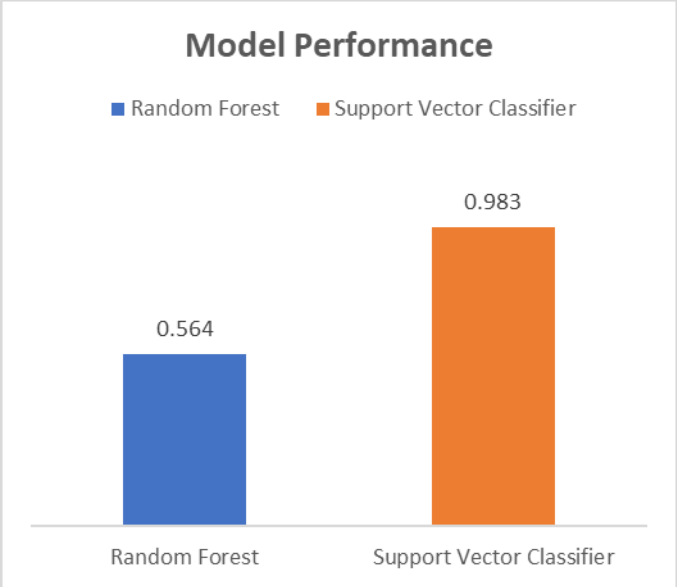

Figure 2. Model Performance

- **Model Interpretability:** The SVC model is a rigid model i.e., a black box model with no logical explanation. Although, the SVC model is a better performing model, it has poor model interpretability. Random forest model on the other hand, has a higher model interpretability i.e., the relationship between variables is defined and inferences can be drawn using the features importance list to state which variables have a greater influence on the prediction result.

3.0. CONCLUSION

The preferred model to adopt for the *Telco_Churn* classification problem is relatively subjective. This means that the objective of the telecommunications company will determine the model that best fits the company's interest. However, in consideration of the focused customer retention program, the random forest model is a good recommendation because it gives a better explanation on likely features or factors that causes a customer to churn. The random forest model was able to make a prediction, backed up with valid interpretability. This model gave an average performance in reducing the false negative prediction error but possesses a logical overview on the predicted result.

However, if the company objective becomes volatile or the course of the customer retention program changes to a performance centric goal, then the support vector classifier model is highly recommended as it gives an excellent performance in churn prediction.

4.0.APPENDIX

Detailed information of the ***"Telco_churn.csv"*** dataset:

Number of Instances: 7032
Total number of Variables: 20
Number of independent variables: 19
Number of target variable: 1

Independent Variables

s/n	Variable	Labels
1	gender	Female, Male
2	SeniorCitizen	Int (0,1)
3	Partner	Yes, No
4	Dependents	Yes, No
5	tenure	Int (0 - 72)
6	PhoneService	Yes, No
7	MultipleLines	Yes, No, No phone service
8	InternetService	DSL, Fibre optic, No
9	OnlineSecurity	Yes, No, No internet service
10	OnlineBackup	Yes, No, No internet service
11	DeviceProtection	Yes, No, No internet service
12	TechSupport	Yes, No, No internet service
13	StreamingTV	Yes, No, No internet service
14	StreamingMovies	Yes, No, No internet service
15	Contract	Month-to-month, One year, Two year
16	PaperlessBilling	Yes, No
17	PaymentMethod	Bank transfer (automatic), Credit card (automatic), Electronic check, Mailed check
18	MonthlyCharges	Int (18.25 - 118.75)
19	TotalCharges	Int (18.8 - 8684.8)

Target Variable

1	Churn	Yes, No

REFERENCE

Dataset is publicly available on Kaggle
https://www.kaggle.com/blastchar/telco-customer-churn