**UNIVERSITY OF WESTMINSTER**

**Msc Big Data Technologies**

Module Code: 7BUIS008W.2

Module Title: Data Mining and Machine Learning

Module Leader: Dr. Panagiotis Chountas

Module Tutor: Dr. Natalia Yerashenia

Assignment: Coursework 2

Student ID: W1848650

Student Name: Mohammadamin Shams

Table of Contents

[1 Abbreviations 2](#_Toc125310213)

[2 Facebook Data Analysis 3](#_Toc125310214)

[2.1 Data Analysis 3](#_Toc125310215)

[2.1.1 Understanding The Data and pre-processing and choosing input and output features 4](#_Toc125310216)

[2.2 Data Preparation 5](#_Toc125310217)

[2.2.1 Missing Values Treatment 5](#_Toc125310218)

[2.2.2 Outliers and Data Type Treatment 6](#_Toc125310219)

[2.2.3 Feature Selection and splitting the data 7](#_Toc125310220)

[2.2.4 Standardizing the Data 8](#_Toc125310221)

[2.2.5 Split Input and Output to Train and Test Dataset 8](#_Toc125310222)

[2.3 Creating Ensemble 9](#_Toc125310223)

[2.3.1 K-Nearest Neighbours 9](#_Toc125310224)

[2.3.2 Logistic Regression 11](#_Toc125310225)

[2.3.3 Support Vector Machine 12](#_Toc125310226)

[2.3.4 Random Forest 14](#_Toc125310227)

[2.3.5 Ensemble Voting Classifier 15](#_Toc125310228)

[2.4 Tuning and Performance Measurements of the Algorithms and Creating Ensemble Classifier with Tuned Models 16](#_Toc125310229)

[2.4.1 Grid Search on KNN 17](#_Toc125310230)

[2.4.2 Grid Search on Logistic Regression 17](#_Toc125310231)

[2.4.3 Grid Search on SVM 18](#_Toc125310232)

[2.4.4 Grid Search on Random Forest 18](#_Toc125310233)

[2.4.5 Grid Search with Ensemble 19](#_Toc125310234)

[3 References 21](#_Toc125310235)

[4 Appendix 21](#_Toc125310236)

[4.1 All the Box Plots for Finding Outliers for Task A 21](#_Toc125310237)

# Abbreviations

|  |  |
| --- | --- |
| SVM | Support Vector Machine |
| KNN | K-Nearest Neighbours |
| LR | Logistic Regression |
| ROC Curve | Receiver Operating Characteristic |
| AUROC | Area Under Receiver Operating Characteristic |

# Facebook Data Analysis

This part of report presents analysis for different interaction metrics that a company gets from consumers based on their social media activities on Facebook. This data is obtained from a cosmetic company about data available around their posts on Facebook.

In the first part of the report, we try to understand the data and then use several features before the post is published and try to predict another feature based on how people interact with them, with the help of supervised machine learning classifiers. In our model we use KNN classifier, random forest classifier, support vector machine classifier and logistic regression classifier. After the different models are built for our classification problem, then we try to create ensemble model which will combine these models to reduce the generalization error for each model. If each model is diverse and independent, the prediction error of the ensemble model decreases.

In the second task we try to tune our models and form better models by changing hyperparameters and parameters of models to predict the desired value and this part is done by the help of grid search algorithm. Then again, we will create our ensemble model this time with tuned models for better prediction.

In the next parts, extra information for each task and the code for them with extra details is provided.

## Data Analysis

First step to create a classifier is to understand the data. In order to do that, it is necessary to start with preliminary steps to get a better knowledge of the dataset provided and also, we should remedy dataset before using it in our algorithms to get a better result.

The first part before any analysis is running importing necessary libraries and modules for the different functions needed to be used. The first three cells in google colab (file attached in the appendix) is dedicated to this part. Then we need to import our dataset with the help of pandas which makes it possible for us to read csv files.

Graphical user interface, text, application

Description automatically generated

Figure Importing the dataset

### Understanding The Data and pre-processing and choosing input and output features

#### Description of the Dataset

The raw dataset has 18 variables according to csv file provided. These features capture data related to the posts that a company published on social media. In our dataset we have information about 500 posts published by this company. From these 18 features, data of the seven of them can be collected before the post is published in Facebook. Other 11 is measured when the post is out, and they are the metrics that Facebook uses to measure the interactions a post gets. The features are explained in the table below with a small description about each based on the article provided in blackboard.

|  |  |  |
| --- | --- | --- |
| Feature | Data type | Description |
| Page total likes | Numeric | Number of the people who liked your page |
| Type | String of letters | Type of content (Link, Photo, Status, Video) |
| Category | Numeric | Manual content characterization: action (special offers and contests), product (direct advertisement, explicit brand content), and inspiration (non-explicit brand related content |
| Post Month | Numeric | Month of the post was published |
| Post Weekday | Numeric | Weekday that post was published |
| Post Hour | Numeric | Hour the post was published |
| Paid | 0 or 1 | If the company paid Facebook for advertisement |
| Lifetime Post Total Reach | Numeric | The number of people who saw a page post |
| Lifetime Post Total Impressions | Numeric | Impressions are the number of times a post  from a page is displayed, whether the post is  clicked or not |
| Lifetime Engaged Users | Numeric | The number of people who clicked anywhere  in a post |
| Lifetime Post Consumers | Numeric | The number of people who clicked anywhere  in a post |
| Lifetime Post Consumptions | Numeric | The number of clicks anywhere in a post |
| Lifetime Post Impressions by people who have liked your Page | Numeric | Total number of impressions just from people who have liked a page |
| Lifetime Post reach by people who like your Page | Numeric | The number of people who saw a page post because they have liked that page |
| Lifetime People who have liked your Page and engaged with your post | Numeric | The number of people who have liked a Page and clicked anywhere in a post |
| Comments | Numeric | Number of comments on the post |
| Likes | Numeric | Number of likes on the post |
| Shares | Numeric | Number of times the post was shared |
| Total Interactions | Numeric | Sum of likes, shares, comments |

Table Features in the dataset

This data is gathered by Facebook for company and a group of scientists to analyse the Facebook metrics and to predict and get insight from the table above. In their work they used 7 first columns for input of their models to predict other 11. The approach that they took is reasonable because prior to the post being published the first 7 columns can be created and based on them, the other 11 can be predicted and then the accuracy of each model can be calculated. They did all of this and found out the best column to predict with least error was “lifetime people who have liked your page and engaged with your post” and the second one was “lifetime post consumers”. The second one with the least error based on the description provided is related to each post published. “Lifetime post consumers” is a better candidate for predicting and testing models on it, and also finding other metrics of models. In this report the approach that has been taken is like the article and instead of predicting a continuous value with regression for “lifetime post consumers” we try to predict the range of this feature and classify it. To fit our classification model this feature is categorized into 2 equal groups in terms of number of observations after treating the null and outliers.

## Data Preparation

First understand the data type in each column and preliminary information like number of columns and rows about the dataset.

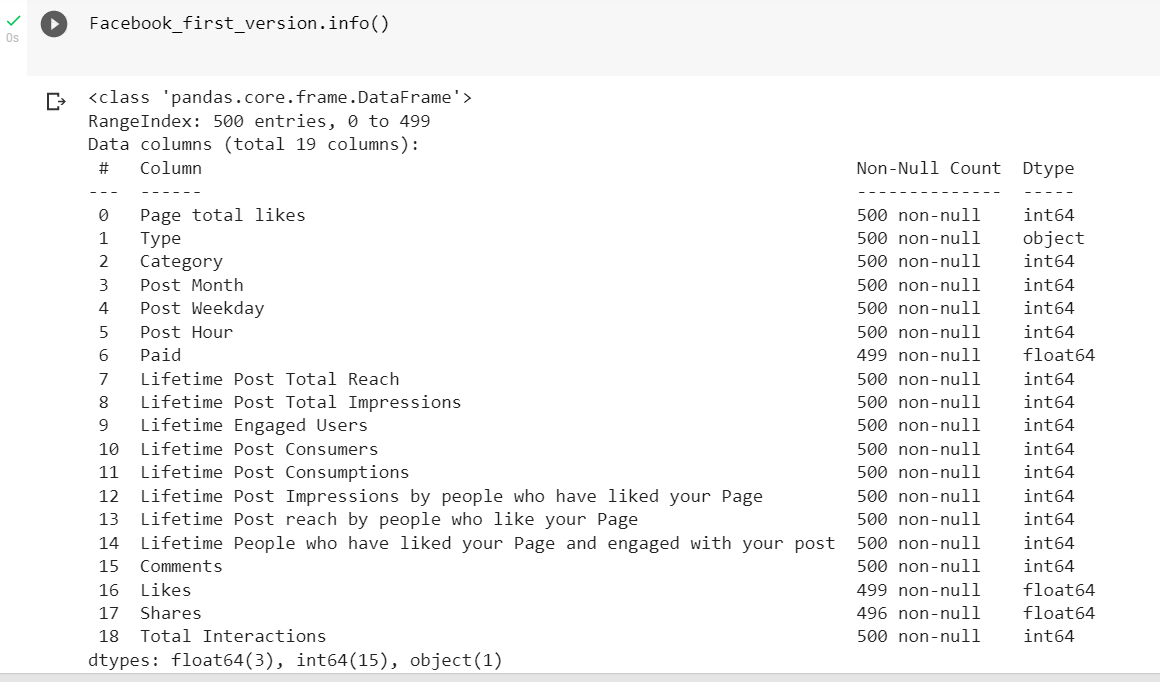


Figure Data Types in the dataset

We also can get a good report of the data with running profile report by importing pandas profiling. Second thing needs to be done is see the null values and treat them accordingly.

### Missing Values Treatment

Based on the number of missing values in each column, best approach of dealing with them is to drop those rows with null values. Its can be done with the help of dropna() function on our dataset.

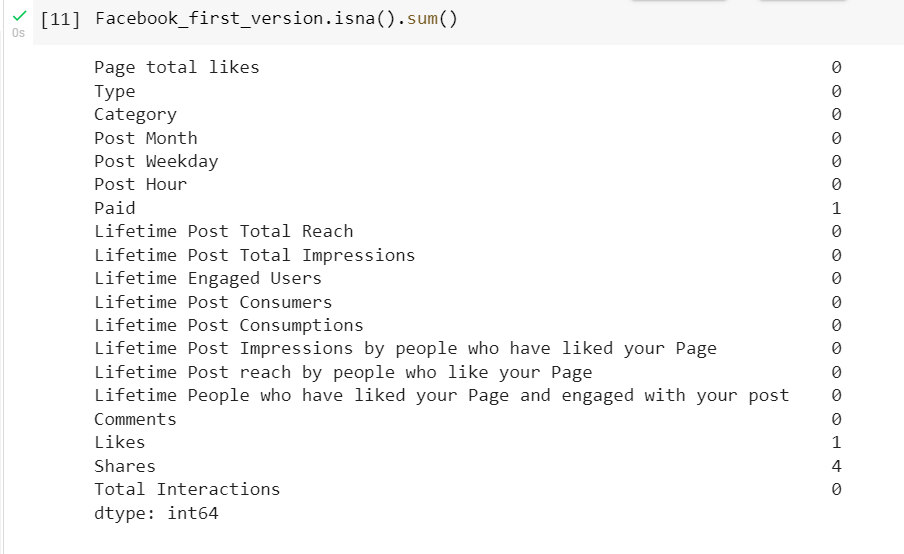


Figure Null values in different columns

### Outliers and Data Type Treatment

In order to give the data to our classifying models the input should be numeric, but Type column is a string telling the type of each post and values are photo or video or status or link, this should be changed to numeric values. There is two ways to do this we can use getdummies function and create four different hot encoded columns and add them to our dataset or changing it into one column by the help of NumPy library and use conditions and values to create a new column. Both are used but the second one is used as an input for our classifiers.

A picture containing graphical user interface

Description automatically generated

Figure Change Type column from text objects to numeric values

To get better results with machine learning algorithms the data should not have outliers. One of the ways to do this objective is plotting boxplot for each column to see outliers out of 1.5 IQR (interquartile range), then define a function to detect outliers in columns and then we can delete them from the original dataset. By running this on all features of dataset around 150 outliers is detected which is a high number compared to number of all 495 rows after dropping null values. To address this issue only outlier’s detection function is used on the input and output features explained in the previous part. Which will only indicate there is only 37 outliers in our chosen features for classifying models, and they are around 10 percent of the dataset. After detecting outliers, with the help of the list of indexes created for them, they are going to be deleted from dataset. Boxplot for all the features is included in the appendix. The code and only the chosen outcome for our model is provided in the figure below.

Graphical user interface, text, application, email

Description automatically generatedChart, box and whisker chart

Description automatically generated

Figure Boxplot function

A picture containing graphical user interface

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

Figure Outlier detection and removing them

### Feature Selection and splitting the data

Based on the approach elaborated in the article, inputs for our models are the features that we have a prior knowledge before post publication, and they are listed in the table below.

|  |  |
| --- | --- |
| Feature | Data type |
| Page total likes | Numeric |
| Type to numeric | Numeric |
| Category | Numeric |
| Post Month | Numeric |
| Post Weekday | Numeric |
| Post Hour | Numeric |
| Paid | Numeric |

Table Inputs for classifier models

Choosing these features for machine learning algorithms are meaningful because a company can gather this data before publishing the post and based on these, they can see the consumers engagement with their post. In the article it is mentioned that based on the predicting models that they have created, the best feature for prediction based on the feature being meaningful and related to only one post and lowest error rate for their regression models was “Lifetime Post Consumers”. This feature is used for prediction purposes and because we must build a classification ensemble, it is needed to change it to two groups with different ranges and assign discrete values for those ranges and then create our models. The median of this feature is used to divide observations into two groups as shown below.

Text

Description automatically generated with medium confidence

Figure Target value change to categorical value

Now we need to create our input(X) and target value(y) for our models and put them into 2 datasets called X for inputs and y for outputs. There are several ways to form our input and output. We can create X and y by using column names from dataset or create them by dropping some columns with drop function from pandas and then keep desired ones for our analysis. Both ways are used but one of them is provided below.

A picture containing text

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

Figure Input and output for classifier models

### Standardizing the Data

Because we used page total likes as one of the features for our analysis and by observing the values of that column and compare it to other input features and see the difference in ranges of them, its obvious that for a better result in our machine learning algorithms data should be scaled. Scaling help so the machine learning algorithms and deep learning algorithms do not saturate too fast.

Graphical user interface, text, application

Description automatically generated

Figure Scaling the input of the model

### Split Input and Output to Train and Test Dataset

Now we need to split our data into train and test set, so we would be able to train our models with training data and use test data as unseen data for measuring the performance of our classifiers and see how well they perform. First method to split the data is holdout method and we used it by importing train\_test\_split from sklearn. We form our train and test set by 70% for training set and 30% for testing set. Using random\_state equal to an integer number ensures that we get the same split each time we use it and also using stratify to make sure that the training data has proportion of both values of our output.

Second method to create test and train set is using K-Fold method with 5 folds to split the data into train and test set.

Graphical user interface, text, application

Description automatically generated

Figure Train and test split with two ways

## Creating Ensemble

Now we formed our data by splitting it to input and target values, we are ready to create our models. First step is to define our 4 classifiers and train them with the test data. In this report we used supervised machine learning models and that means model will learn from the data that is labelled before. They are k-nearest neighbour classifier, support vector machine, random forest, and logistic regression. After we trained our model with the train set, they are ready to predict the outcome by using test input, with the prediction they made we can see how well they performed by comparing outcome from the models with the actual target values of the test set.

Then we can create our ensemble model which is a voting classifier and use the models created and predict target by counting each vote from classifiers and then decide on the outcome. Ensemble can use hard voting that means the majority of votes create the outcome or soft voting and that means using probabilities for creating the outcome.

Now let’s start creating each model and train them for predicting the target value.

### K-Nearest Neighbours

K-Nearest Neighbours is a supervised machine learning algorithm which works by taking a data point and looking at the ‘k’ nearest data points around it and assign it to majority of the ‘k’ closest neighbours. In this report we use Scikit-learn machine learning library and build our model with it. We build our model and put number of neighbours at 3. When the model is created and trained with the training set, we can predict the target value for the test set with predict function and then evaluate the performance of the machine.

Graphical user interface, text, application, email

Description automatically generated

Figure KNN model and its accuracy

Then we can see the accuracy of our model by comparing real test values with the prediction of model. Accuracy is simply the ratio of correctly predicted observations to the total observations and our model has 76% accuracy. Other evaluation metrics are recall, precision. Precision is the ratio of correctly predicted positive observations (in our case it means posts that had less than 520 consumers engaged with the post and clicked on it) to the total predicted positive observations. Precision of our model is 75% and it means 75% of the predicted posts with consumers less than 520 are predicted right. Recall is the ratio of correctly predicted positive observations (in our case it means posts that had more than 520 consumers engaged with the post and clicked on it) to all observations in actual class. In our case recall is 76% and it means out of all the posts with more than 520 consumers 76 percent of them are detected. We also can see the confusion matrix which will summary a prediction of results, in the figure below we can see what confusion matrix will do and then there is the code for getting one for our model.

Chart

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Figure Confusion matrix

we have two classes so our confusion matrix is 2\*2 (and we have two classes 0 and 1 (0=posts with consumers (who click on the post) below 520, 1= posts with consumers more than 520)) and we can understand 52 and 53 was the accurate prediction and 17 and 16 was the wrong prediction in different groups. We also can see the classification report which will give us recall and precision and F1 score (The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0) for both labels(0and1).

Table

Description automatically generated

Figure KNN classification report

We trained our KNN model with our test data from holdout split method now let’s see the accuracy with k-fold splitting method.

Graphical user interface, text, application

Description automatically generated

Figure KNN with k-fold split

As we can see the accuracy got lower and became 70%.

### Logistic Regression

Logistic regression is another supervised classifying machine learning algorithm, and it will model the probability of a discrete output, with a given input variable and this model is one of the useful methods for classification problems.

Now like KNN model let’s create our model and evaluate the outcome with metrics described in KNN model.

Graphical user interface, text, application, email

Description automatically generated

Figure Logistic regression accuracy

We can see the accuracy of LR is 75 percent and it is a good accuracy.

Table

Description automatically generated

Figure Logistic regression evaluation

We can see other parameters like recall and F1score and precision for both groups in the figure above and from the confusion matrix we can see our model predicted 51 and 53 of predictions was right and 18 and 16 was wrong based on (0,1) labels respectively.

Now let’s use the train and test sets from k-fold split method and see the accuracy of our model with those input and output.

Graphical user interface, text, application

Description automatically generated

Figure Logistic regression with k-fold split

The accuracy score with k-fold splitting method is lower than holdout method exactly like KNN.

### Support Vector Machine

Another algorithm for prediction of values that can be used in both classification and regression problems is support vector machine. Support vector machine for classification problems tries to find the best plane that divide the data into different categories with maximum margin between data points for different categories. SVM to be able to separate the categories uses different kernels, the kernels used in this report for SVM are polynomial, gaussian, and sigmoid kernels.

#### Polynomial Kernel

Graphical user interface, text, application, email

Description automatically generatedTable

Description automatically generated

Figure SVM with polynomial kernel evaluation

#### Gaussian Kernel

Table

Description automatically generated with medium confidence

Figure SVM with gaussian kernel evaluation

#### Sigmoid Kernel

Table

Description automatically generated with medium confidence

Figure SVM with sigmoid kernel evaluation

Based on the figures above we can see the gaussian kernel performed better and had the best accuracy amongst other with 80%. With the confusion matrix we can see that 55 and 56 from 1 and 0 label for post consumers were predicted right respectively.

Now let’s use k-fold split outputs for our best SVM model and see the accuracy from the figure below.

Graphical user interface, text, application, email

Description automatically generated

Figure SVM with Gaussian and k-fold split

As it can be seen again, we have a lower accuracy with k-fold split than holdout split

### Random Forest

Random forest classifier will use several decision trees and combine them together. Random forest itself is one of the ensemble algorithms and based on the vote of different trees will create the output and you can mention the depth of trees you want to be combined for creating the random forest classifier. Now let’s build our random forest and test it.

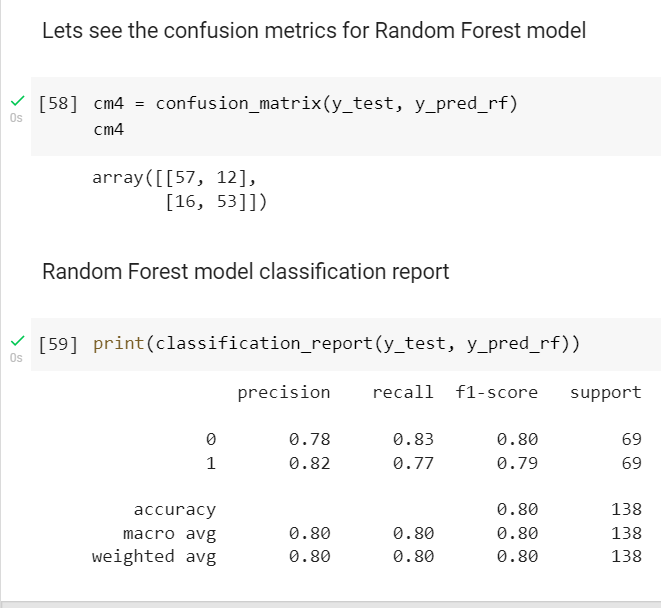
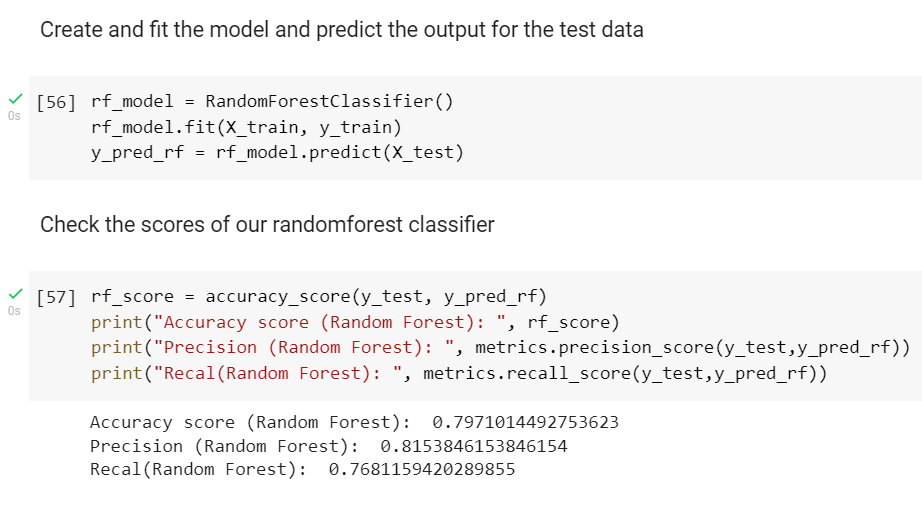


Figure Random Forest model evaluation

Random forest model accuracy is 79% and have a better precision for predicting posts with more than 520 consumers and it has a good accuracy. Before we create our ensemble lets create random forest with k-fold split too and see if the outcome will be better or not.

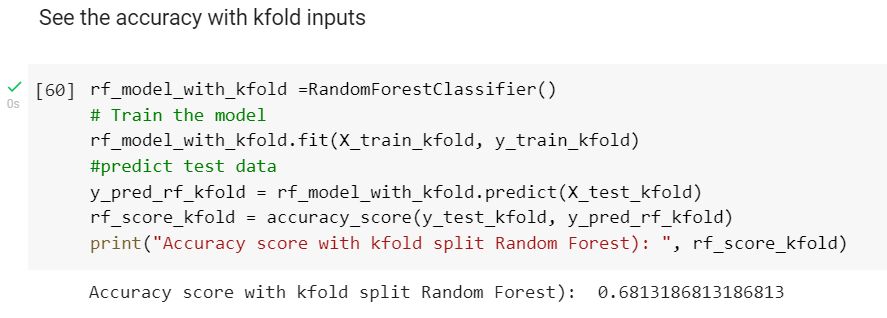


Figure Random Forest evaluation with k-fold split

Like all other models the accuracy drops, and it seems that k-fold with these parameters is not a good approach to split the data and holdout version is a better approach for splitting the data.

### Ensemble Voting Classifier

Before we create our ensemble model lets see the accuracy of each model with holdout and k-fold method for splitting data in the figure below. One thing needs to be mentioned is all the models every time we run the code has the same output but random forest because it’s an ensemble every time generates another accuracy but we can see that each time, it’s in the range of 75% to 79%. Also, we can see that the SVM with gaussian kernel creates a better prediction for our test data.

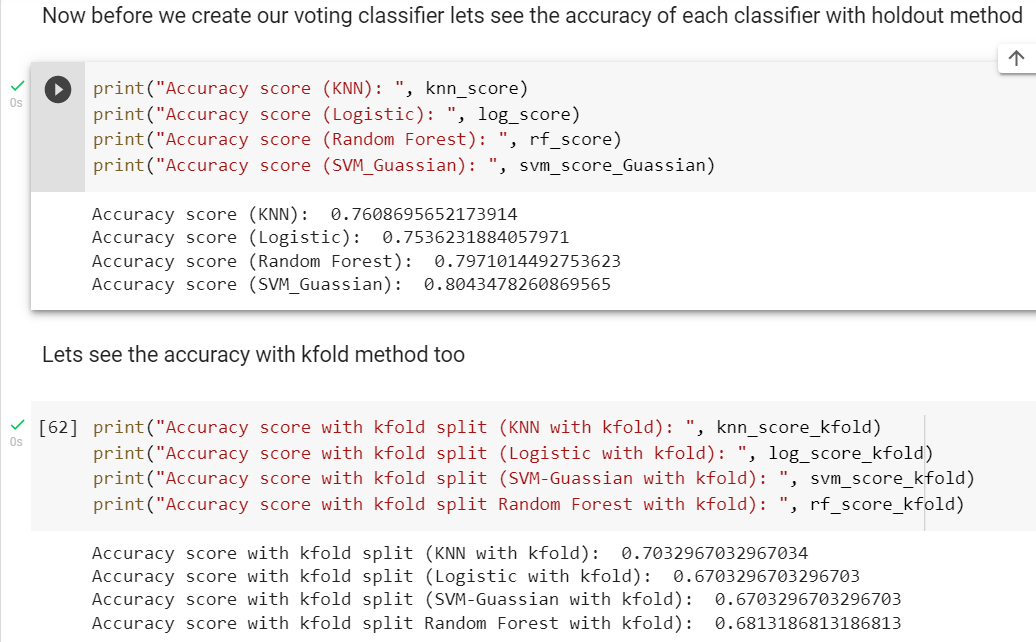
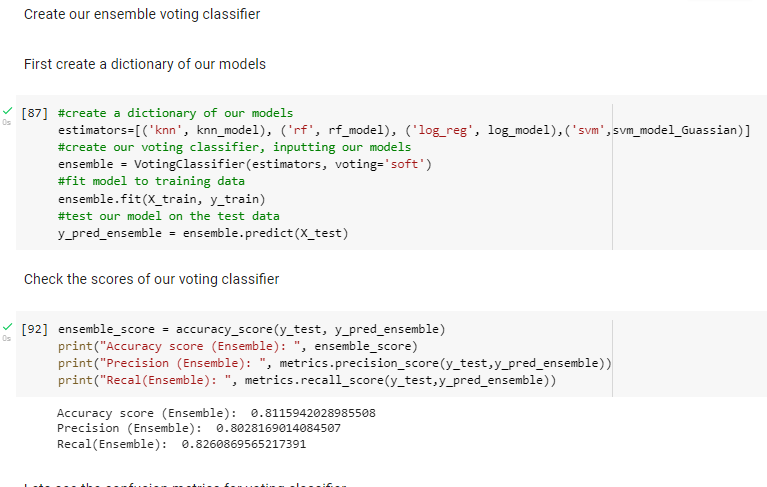


Figure Accuracy of all models

Now it’s time to create the ensemble model. Ensemble method will combine several models together to predict a better outcome and reduce the generalization error for each model. If each model is diverse and independent the prediction error of the ensemble model decreases. First, we need to create an array of estimators and put the name of models we created on that array for the voting classifier. Then we need to put the voting method that is used by voting classifier, there is 2 options hard and soft voting. Hard voting entails picking the prediction with the highest number of votes, whereas soft voting entails combining the probabilities of each prediction in each model and picking the prediction with the highest total probability. We use soft voting because there is a problem with roc curve when we want to plot it for our ensemble model with hard voting.

Table

Description automatically generated

Figure Ensemble model evaluation

Our voting classifier was able to create a better accuracy for our classification problem and done better compared to each of the models, but this is not always the case and, in our case, our classifier was able to create a better target value. Other values and confusion matrix and classification report is also provided above.

Let’s create our ensemble mode with our k-fold splitting method to see the outcome.

Graphical user interface, text, application, email

Description automatically generated

Figure Ensemble evaluation with k-fold split

Our k-fold splitting method with our voting classifier have the best accuracy of 87% and this fact is interesting because every time we used k-fold split method the accuracy was lower than holdout split method but, in the ensemble, we have a better result with k-fold splitting method.

## Tuning and Performance Measurements of the Algorithms and Creating Ensemble Classifier with Tuned Models

Tuning parameters is when we go through an algorithm to find the best parameters for the model to improve the accuracy. GridSearchCV goes through our model several times and train it with range of different parameters mentioned by us and find the best parameters with the highest accuracy.

### Grid Search on KNN

On our KNN model we can use grid search by changing the numbers of k neighbours for classifying of a point.

A picture containing text

Description automatically generated

Figure KNN with grid search

We can see that the number of neighbours for best outcome is 7 and we will get 78% accuracy in this way which is higher than the default value of neighbours (5) which was 76% that we got without grid search.

### Grid Search on Logistic Regression

For our logistic regression model, we will form array of values we want to change and see the outcome of them on the accuracy and choose the best one. One of the values is regularization parameter for penalizing the coefficient values for reducing the chance of overfitting. C is another parameter we can change which will define relative strength of regularization.

Text

Description automatically generated

Figure Logistic regression with grid search

After grid search, we can see the penalty should be l2 and the value for C should be 1 to get the best result. Accuracy is got higher from 75% to 78% with grid search and other metrics increased too.

### Grid Search on SVM

Grid search on SVM can change C parameter and gamma. The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. Gamma parameter defines how far the influence of a single training example reaches.

A picture containing text

Description automatically generated

Figure SVM with grid Search

After grid search the best value for gamma is 0.1 and best value for C is 1 and we can see the accuracy got even better from 80% to 82%.

### Grid Search on Random Forest

For tuning random forest algorithm, we can change the number of estimators and it is the number of trees being used for the prediction.

Text

Description automatically generated

Figure Random Forest with grid search

With the 200 for estimators, accuracy got a bit lower and fell to 77% from 79% Again it is worth mentioning that each time we run the algorithm, numbers will change for the random forest and the accuracy is in a better range with grid search.

### Grid Search with Ensemble

Now we create the ensemble model, this time with the tuned classifiers to see the performance of the ensemble.

Graphical user interface, text, application

Description automatically generated

Figure Ensemble with grid search

The accuracy score for the model is 81% and it is same as before grid search, another point worth mentioning is our best SVM model has a higher accuracy than ensemble model.

#### Metric Measurement on Ensemble

##### Accuracy Score

Accuracy is simply the ratio of correctly predicted observations to the total observations. We can get this score by using accuracy\_score and put test output and predicted output to get the accuracy. In the figure above the code is provided. For our Ensemble classifier model accuracy is 81%.

##### Confusion Matrix

In the figure below we can see a confusion matrix and it will give us a good understanding of what is confusion matrix. Confusion matrix is 2\*2 matrix, if target has two different values. The diagonal values show the accurate predictions and non-diagonal values shows us inaccurate predictions. Confusion matrix can be achieved with a one line code after importing confusion\_matrix function from sklearn.metrics library. Also, we can create a heatmap easily (but there was a problem with my outcome with heatmap and I couldn’t get those) instead of raw confusion matrix. Our model predicted 55 and 57 accurate predictions, 14 and 12 was inaccurate predictions. So, for class 0 (Posts with consumers less than 520) model predicted 55 accurate predictions but it missed 12. For class 1 (Posts with consumers more than 520) our model predicted 57 right targets and missed 14.

Chart

Description automatically generated

Figure Confusion Matrix

##### Precision

Precision quantifies the number of positive class predictions that belong to the class. We need to know that we can create precision for all labels of the outcome with using the description above with minor changes.

##### Recall

Recall quantifies the number of positive class predictions made from all positive examples in the dataset. Like precision we can get this value for all of the labels.

##### F1 Score

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0.

It is possible to get all these values for our model with a single line of code by classification report and see these values for our 0 and 1 labels.

##### The ROC Curve and AUROC

The receiver operating characteristic curve is a graph visualizing the performance of a classification model at all classification thresholds. This curve plots two parameters: true positive rate (TPR) and false positive rate (FPR) of a classifier against each other. Again, we can plot this with both of our labels by changing the meaning of positive and negative, we can plot it for all labels. And a good ROC curve should stay far away from diagonal line. We can get ROC plot with the help of Scikitplot library and with the metrics.plot\_roc line we can plot our ROC curve and also it will calculate AUROC score. AUROC score measures the performance of the classifier with area under the curve, this value is between one and 0.5 and it should be closer to 1 to show a better classifier in our case AUROC is 0.88 for both classes and also the code is provided in fig 31 and the ROC curve is at fig 33.

Chart, line chart

Description automatically generated

Figure ROC curve for ensemble

# References

Moro, S., Rita, P. and Vala, B. (2016). Predicting social media performance metrics and evaluation of the impact on brand building: A data mining approach. Journal of Business Research, 69(9), pp.3341–3351.

‌

# Appendix

## All the Box Plots for Finding Outliers for Task A

