Thera Bank's Dataset

Machine Learning

Marios Kyriacou

Contents

1 Introduction

Thera Bank's management plans to explore methods to convert its liability customers into personal loan customers while keeping them as deposits. Last year, the bank ran a campaign aimed towards liability clients, which saw a respectable conversion rate of more than 9%. Thus, the retail marketing department has been forced to develop more targeted programs to boost success rates on a limited budget. This dataset contains information on 5000 costumers. Customer demographic information, like age , income, the customers' connection with the bank, and the customers' reaction to the previous personal loan campaign are all included in the dataset. Moreover, only 480 of the 5000 clients accepted the personal loan given to them in the last campaign.

This report investigates the many correlations between the dataset's attributes and graphs their distribution through the concept of exploratory data analysis **(EDA)**. This study uses a variety of supervised and unsupervised Machine Learning **(ML)** approaches to answer different questions depending on the dataset. Primarily, deciding when a customer will accept a personal loan or not **(Supervised)** is one of the most difficult challenges using this data. The decision tree algorithm **(DT)** plays a crucial role in this question to determine whether or not a customer would accept the loan. Secondly, we use the Support Vector Machine method **(SVM)** to classify if a client has a certificate of deposit account with the bank or not **(Supervised)**. Finally, the Clustering algorithm is used to categorize customers according to their yearly income and average monthly spending **(Unsupervised)**.

The most fundamental aspect of each algorithm is the construction of models and comparisons with other algorithms, an approach known as ensemble learning. Therefore, this research will use the Thera Bank dataset to go through several machine learning concepts and answer various questions.

1.1 Dataset

The "Bank Personal Loan Modelling" dataset, available on the Kaggle website, has 5000 cases with 14 attributes. The dataset contains seven numeric variables ('Age','CC Avg','ID','Income', 'Mortgage','Zip Code','Experience'), two categorical variables ('Education', 'Family'), and five Boolean variables ('CD Account', 'Credit Card', 'Online', 'Personal Loan', 'Securities Account')[Kaggle, 2021]. In Table 1 the description of the each attribute is illustrated.

Attributes	**Description**
ID	Customer ID
Age	Customer's age in completed years.
Experience	Years of professional experience.
Income	Annual income of the customer ($000).
ZIP Code	Home Address ZIP code.
Family	Family size of the customer.
CCAvg	Avarage spending on credit cards per month ($000).
Education	Education Level:1. Undergrad 2. Graduate 3.Advanced/Professional
Mortgage	Value of house mortgage if any ($000).
Personal Loan	If the customer accept the personal loan offered in the last campaign or not.
Securities Account	If the customer have a securities account with the bank or not.
CD Account	If the customer customer have a certificate of deposit (CD) account with the bank or not.
Online	If the customer use internet banking facilities or not.
Credit card	If the customer use a credit card issued by any other Bank or not.

Table 1: The dataset's Information.

2 Exploratory Data Analysis

Exploratory data analysis (EDA) is an important and required initial step before applying any machine learning model. EDA is a method of analyzing data gathered from many sources [Milo and Somech, 2020]. Addressing missing values, removing duplicates, outliers treatment, normalizing and scaling, and encoding categorical data are all key concerns of EDA [Admin, 2020].

2.1 Dataset Description and Missing Values

Before proceeding to the following phases of EDA, it is necessary first to understand the various data types and other statistics that relate to our data [Edureka, 2021]. Understanding the dataset helps to develop statistical analysis, which summarizes the dataset's mean value, distribution, and shape.

	ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
0	1	25	1	49	91107	4	1.6	1	0	0	1	0	0	0
1	2	45	19	34	90089	3	1.5	1	0	0	1	0	0	0
2	3	39	15	11	94720	1	1.0	1	0	0	0	0	0	0
3	4	35	9	100	94112	1	2.7	2	0	0	0	0	0	0
4	5	35	8	45	91330	4	1.0	2	0	0	0	0	0	1

Figure 1: The first 5 rows of the dataset.

	ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000	5000.000000	5000.000000
mean	2500.500000	45.338400	20.104600	73.774200	93152.503000	2.396400	1.937938	1.881000	56.498800	0.096000	0.104400	0.06040	0.596800	0.294000
std	1443.520003	11.463166	11.467954	46.033729	2121.852197	1.147663	1.747659	0.839869	101.713802	0.294621	0.305809	0.23825	0.490589	0.455637
min	1.000000	23.000000	-3.000000	8.000000	9307.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000
25%	1250.750000	35.000000	10.000000	39.000000	91911.000000	1.000000	0.700000	1.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000
50%	2500.500000	45.000000	20.000000	64.000000	93437.000000	2.000000	1.500000	2.000000	0.000000	0.000000	0.000000	0.00000	1.000000	0.000000
75%	3750.250000	55.000000	30.000000	98.000000	94608.000000	3.000000	2.500000	3.000000	101.000000	0.000000	0.000000	0.00000	1.000000	1.000000
max	5000.000000	67.000000	43.000000	224.000000	96651.000000	4.000000	10.000000	3.000000	635.000000	1.000000	1.000000	1.00000	1.000000	1.000000

Figure 2: The description of the dataset.

The data description includes a variety of metrics such as count, mean, standard deviation, minimum and maximum values, first (Q_1) and third (Q_3) quartile for each attribute. Since each attribute includes 5000 recordings, we can see no missing values in the dataset. Furthermore, Figure 2 assists us in displaying certain key facts for the 'Experience,' 'Education,' and 'Personal Loan' attributes. The 'Experience' column has a minimum value of -3, which has no relevance. Hence those observations are replaced by the value of 0, meaning that the customers have no experience. Further, we can observe that the 'Education' feature has a mean value of 1.8810, which is close to 2, indicating that most customers have graduated. The mean value for 'Personal Loan' is 0.0960, displaying that the dataset is unbalanced since the majority values are zero rather than one. Finally, we remove 'ID' and 'Age' attributes

since they will not affect the upcoming models.

As previously stated, the dataset contains no missing values; this is supported by the graph below.

Figure 3: The missing values.

2.2 IQR Method and Features Distributions

One of the essential parts of EDA is identifying and detecting the dataset's outliers. Outliers are values "that lie quite far from the middle of the distribution in either direction", according to Mendenhall [Last and Kandel, 2021]. Outliers might have resulted from a data-gathering error, or they can be a hint of data variability [Sharma, 2018]. The most essential reason to eliminate outliers is that the existence of outliers affects most statistical approaches, leading to a poor ML model [Wijaya, 2020].

Outliers can be identified in various ways. In this report we'll apply box plots to identify the outliers and Inter Quartile Range **(IQR)** method to eliminate them. Note that since the dataset is imbalanced, the outliers of 'Personal Loan', 'Security Accounts', 'CD Account', 'Online' and 'Credit Card' are detecting the cases of the minority class. Thus, we will not identify them as outliers since we will end up with only one class.

```
for i,feature  in enumerate(features):
    plt.figure(figsize=(25,25))
    plt.subplot(14,2,i+1)
    sns.boxplot(x=data[str(feature)], width= 0.8, linewidth = 1.2, whis =0.1, fliersize= 0.8)
    plt.subplot(14,2,i+2)
    sns.distplot(data[str(feature)])
    plt.show()
```

Figure 4: The outliers code.

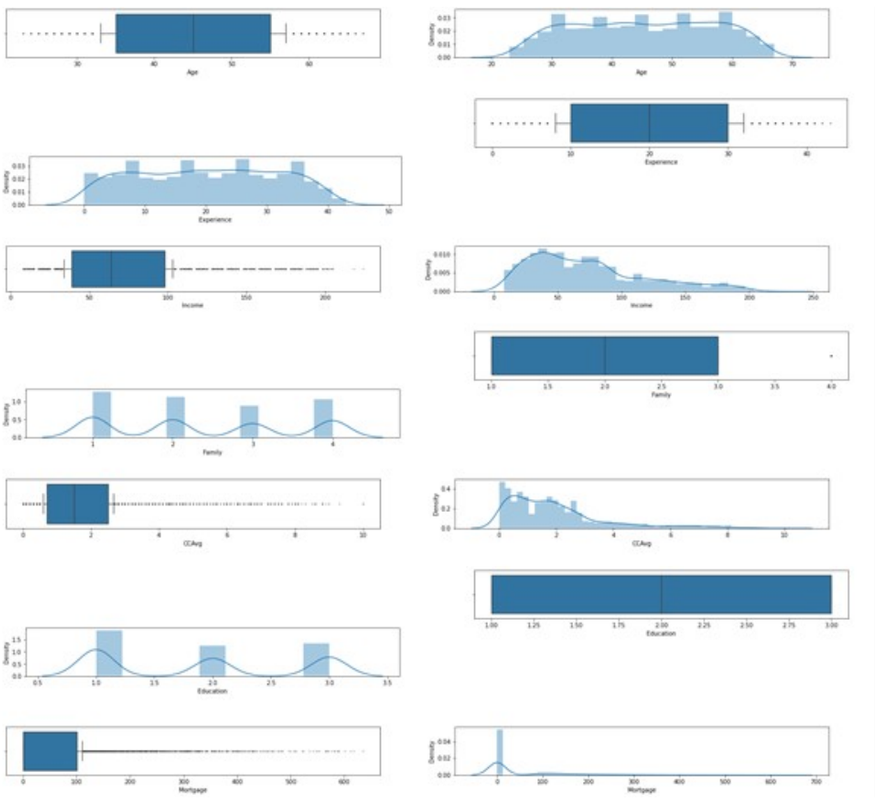


Figure 5: The outliers of the numeric features.

According to Figure 5, 'Income' ,'CC Avg' and 'Mortgage' are the three features that outliers may notice with the naked eye. Our initial suspicions regarding the outliers of these three features may be confirmed using the IQR method:

$$IQR = Q_3 - Q_1$$

The IQR is a statistical data preprocessing technique which recognizes extreme values that may produce "noise" and lead to poor results [Bougoudis,Demertzis and Iliadis, 2016]. Note that outliers in **numerical** datasets are discovered using the IQR method [Nair, 2019].

To start with, after the computation of the IQR value, we must establish the normal data range with lower and higher limits of $Q1 - 1.5 * IQR$ and $Q3 + 1.5 * IQR$, respectively. Outliers are data points that fall outside this range and should be eliminated from the dataset [Taylor and Courtney, 2020].

```
#Copy the original dataset to find the outliers for the numeric columns
dt = data.copy()
dt.drop([ 'Family', 'Personal Loan', 'Securities Account', 'CD Account', 'Online', 'CreditCard'] ,axis=1,inplace=True) #drop the numeric columns
#For each numeric column apply IQR method
out = []
for i in dt.columns:
    Q1 = dt[i].quantile(0.25)
    Q3 = dt[i].quantile(0.75)
    IQR = Q3 - Q1
    outliers = dt[(dt[i] < (Q1 - 1.5 * IQR)) | (dt[i] > (Q3 + 1.5 * IQR))].index  # find the index of the outliers
    out.extend(outliers)

data.drop(out, axis=0,inplace=True ) # drop the outliers from the original data
```

Figure 6: IQR method in Python.

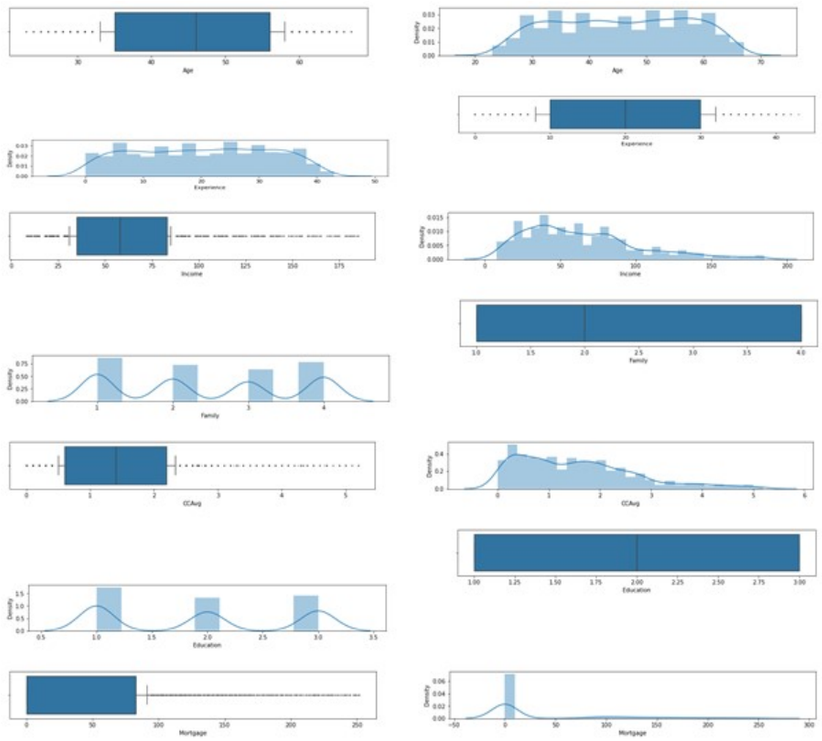


Figure 7: Features after IQR method.

To ensure that the outliers have been eliminated, we must first compare the scale of each feature before and after applying the IQR method. As Figure 5 illustrates, the 'Income' attribute was initially scaled from 0 to 200. However, after implementing the IQR method, the scale of this attribute is now 0 to 175, as seen in Figure 7. Hence, all the numeric attributes have been eliminated based on the IQR approach.

2.3 Hypothesis Testing and Features Distributions

Feature distribution is another necessary analysis that we need to run to identify whether the values of an attribute are centered or distributed [PELTARION, 2020]. Machine learning models learn from data; thus, distributions are crucial. If we build a ML model using incorrect data (i.e., skewness data, etc.), the model will learn incorrectly, and the given results would be poor [Vieira and Gomes, 2010]. Therefore, feature distribution is used to identify if the samples follows the normal distribution, as such the distributions of the training and test sets [Malik, 2019].

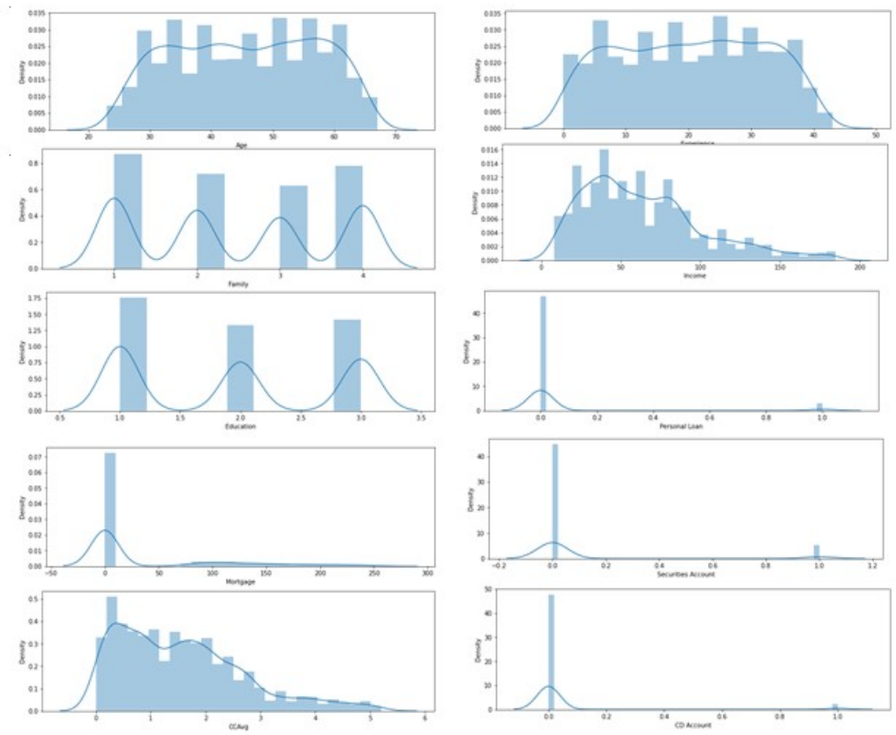


Figure 8: The Distributions of the Features.

According to Figure8, there are indications that 'Age' and 'Experience' are normally distributed. Moreover, the imbalance of the binary columns is obvious. Therefore, hypothesis testing is required to determine if a feature is normally distributed.

Hypothesis testing is a statistical method in which an analyst examines a hypothesis about the population of a sample and determines whether or not a hypothesis should be rejected [MAJASKI, 2021]. In our report we check if the **numerical** attributes follow the Normal Distribution **N∼(0,1)** at 5% significant level. The Normal (Gaussian) distribution is a symmetric probability distribution, indicating that data frequently occur close to the mean [CHEN, 2021]. The probability distribution function is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp^{\frac{x-\mu}{\sigma}}$$

where σ is the standard deviation and μ is the mean of the sample [Jain and Singh, 2003].

To identify if a sample follow the Normal distribution, this report uses the Quantile-Quantile plots **(QQ-plots)** and the Kolmogorov–Smirnov **(KS)** test. QQ-plot is a graphical method of comparing the quantiles of two samples that are placed against each other, illustrating whether their distributions are similar or not [Chaudhary, 2019]. On the other hand, the $K-S$ test compares the data to a known distribution and determines whether or not the two samples are the same [Bhalla, 2019]. Even though the test is commonly used to determine whether or not your data is normally distributed [Glen, 2016].

In our case the null hypothesis H_0 indicates if an attribute follow the Normal Distribution $N(0,1)$, while the alternate hypothesis H_1 indicates if a sample is not normally distributed.

```
def Hypothesis_test (test):
  attribute = data[str(test)]

  # Convert a pandas.Series to list
  print('\n'+('*'*35)+' Convert a pandas.Series to list '+ ('*'*35))
  print('The type of Income feature:',type(attribute))
  attribute_list = attribute.values
  print('The type of Income feature:',type(attribute_list))

  #Normalize the 'Income' attribute
  print('\n'+('*'*35)+' Normalize the attribute '+ ('*'*35))

  min_attribute = min(attribute_list)
  max_attribute = max(attribute_list)
  for i, x in enumerate(attribute_list):
      attribute_list[i] = (x-min_attribute) / (max_attribute-min_attribute)
  print('The min %.3f and max  %.3f after normalization' %(min(attribute_list), max(attribute_list)))
  print('\n'+('*'*35)+'QQ Plot '+ ('*'*35))
  #QQ plot
  sm.qqplot(attribute_list, line ='45')
  py.show()

  #KS Test
  print('\n'+('*'*35)+'KS Test '+ ('*'*35))
  print(stats.kstest(attribute_list, 'norm'))
```

Figure 9: The function of Hypothesis testing in Python.

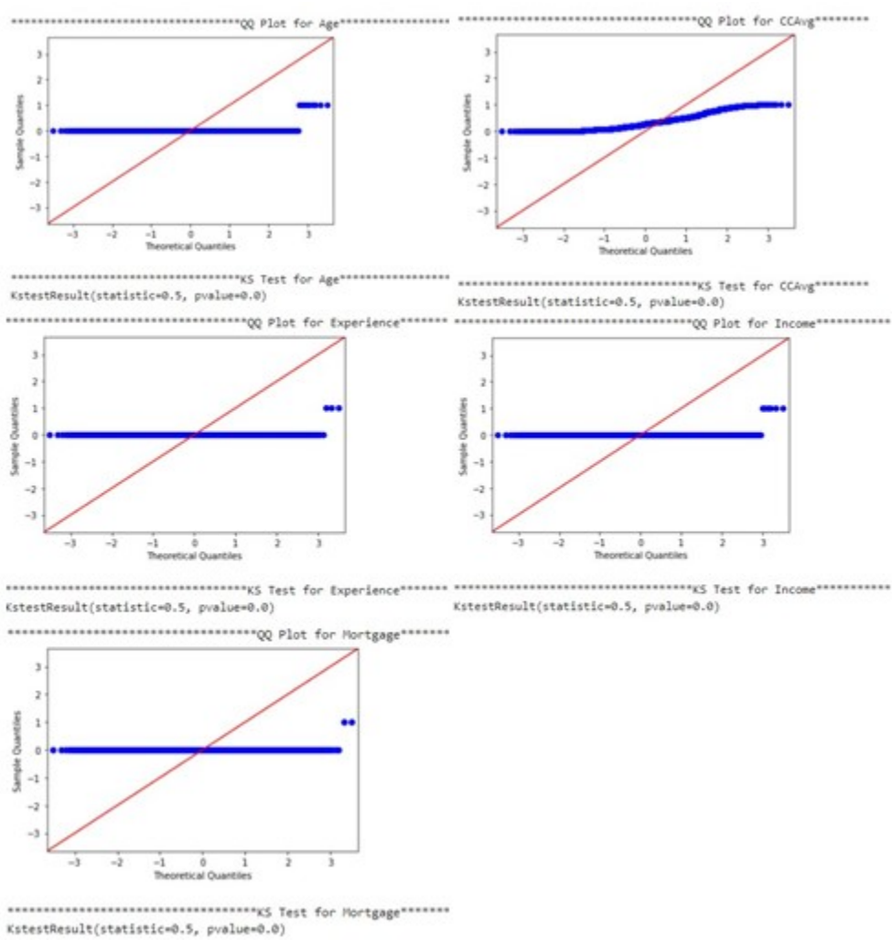


Figure 10: QQ-Plots and KS test.

As shown in Figure 10, each plot illustrates the points of an attribute. It is obvious that the points are not lying in the red line that passes through the origin $(0,0)$. Therefore, we have our first indication that our attributes will not follow the Normal Distribution $N(0,1)$.

By using the $K-S$ test we can confirm this indication. The p-value given from the $K-S$ test of each attribute is equal to zero which is less than 0.5. Consequently, we reject the null hypothesis H_0, meaning that the attributes are not normally distributed.

2.4 Correlation Matrix

A correlation matrix is used to identify the coefficients of a two-variable relationship [Bock, 2019]. The coefficients are scaled from -1 to 1, which corresponds to a perfect negative or positive linear correlation, respectively. Notably, the zero value indicating that the two variables have no linear correlation [ZACH, 2020].

Figure 11: The correlation coefficient for each pair of features.

As Figure 11 shown, "Experience" and "Age" attributes has coefficient equal to 0.99, indicating the highly **positively association**. In other words, the more experience a person has, the older he or she becomes and vice versa. Furthermore, the correlation between "Experience" and "Education" is 0.007249, meaning that they are unrelated. Hence, the two attributes have a **poor correlation**, so a person with a high degree of education does not necessarily means that have experience.

3 Methodology

EDA and the solution of three distinct questions based on Thera Bank's dataset are the two significant aspects of this report.This session represents the methodology that guide this report to solve the three scenarios.

3.1 Supervised

The main issue of a supervised machine learning classification problem is the imbalance dataset; this issue can be confirmed by Figure 8. Imbalanced data refers to datasets in which the target class has an unequal distribution of observations [Mazumder, 2021]. In this study, the Synthetic Minority Over-sampling Technique **(SMOTE)** method and other methods are applied to overcome this problem. In addition, the Grid-Search library, feature selection and 10-fold cross validation will assist us in optimizing each model's results, and the last phase of this report will compare the models and their results using **Ensemble learning**.

3.1.1 Feature Selection and Cross Validation

Feature Selection is one of the most crucial aspects of optimizing an ML model; and more specifically, the Filter Method. Feature selection aims to determine the optimal collection of features that allow the development of functional models [Gupta, 2020]. The Filter method utilizes the correlation coefficients metric to rate each feature and then chooses the one with the highest-ranking [Mazumder, 2019].

Moreover, the **Random Forest** (RF)model, which is a supervised model and consists of several decision trees, is compared with the feature selection results [MALATOOCTOBER, 2021]. Note that from each decision tree is given a value of a feature based on its capacity, to boost leaves purity [Dubey, 2018].

Cross-Validation **(CV)** is a re-sampling technique used to assess machine learning models on a dataset, and examine how well a model generalizes to unknown data [SQLRelease, 2021]. The input dataset is separated into 10-folds; then, the model is trained on nine folds before being tested on the holdout set; lastly, the method is repeated ten times [Brownlee, 2018]. It is worth noting that GridSearchCV aims to use 10-fold CV to identify the optimal hyper-parameters to the most accurate model [SQLRelease, 2021].

3.1.2 SMOTE Method

SMOTE is an oversampling method is used for both supervised scenarios, aimed at resolving the problem of imbalance [Dye, 2020]. By randomly duplicating minority class samples, the SMOTE technique redistributes class distribution [Joshi, 2021]. The algorithm begins by selecting a minority class as the input vector. Then it identifies its k closest neighbors and selects them one of these neighbors by placing a synthetic point somewhere along the line between the point under consideration and its selected neighbor. This process is performed until the dataset is evenly distributed [Brownlee, 2020].

3.1.3 Random Under and Over Sampling

Random Over Sampling **(ROS)** and Random Under Sampling **(RUS)** are two distinct methods that assist in solving the problem of an imbalance dataset in an **SVM** model. RUS is a technique for removing random samples from the majority class [Admin, 2020], while RUS has the disadvantage of eliminating possibly useful samples. ROS is a method that replicates minority class samples randomly, so its drawback is to cause a poor generalization to the test set related to overfitting [Pykes, 2020]. Both procedures are continued until the majority and/or minority classes are dispersed equally [Admin, 2020].

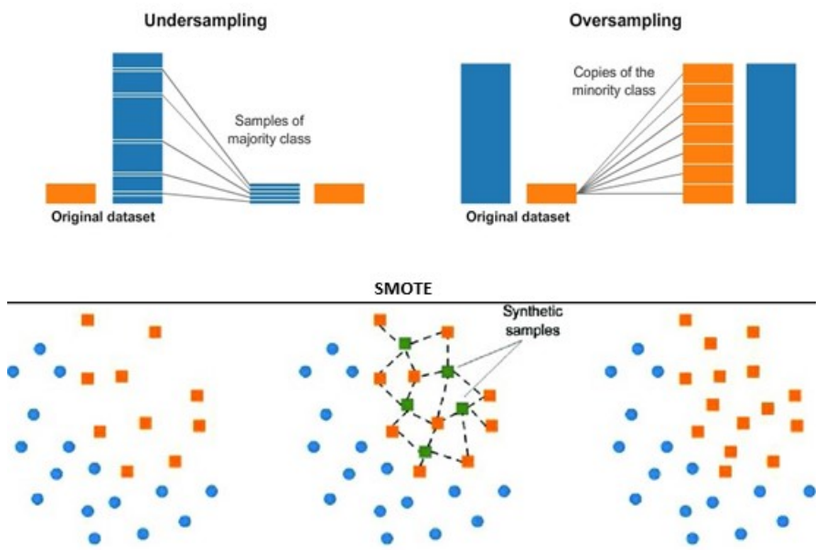


Figure 12: The difference between SMOTE method, Random Under and Over Sampling.

3.1.4 Cost-Complexity Pruning

Pruning is a machine learning data approach that decreases the size of a **DT** model by deleting non-critical parts of the tree [Das, 2020]. Therefore, pruning is one of the strategies used to overcome the problem of overfitting; this report uses pruning to avoid overfitting in a DT model [Arora, 2020].

Cost-Complexity pruning provides another option to control the size of a tree. This technique uses a parameter ($a \geq 0$) known as the complexity parameter, which controls the depth of a decision tree model [Singh, 2020]. As α rises, the number of nodes removed grows, indicating that cost complexity has reached a limit.

3.1.5 Ensemble Learning

Ensemble learning is an approach to machine learning models that combines predictions from different models to improve predictive performance [Brownlee, 2020]. Ensemble algorithms aim to integrate various classifiers into a meta-classifier that outperforms each classifier in terms of generalization [Brownlee, 2021]. The two main types of ensemble methods are sequential and parallel ensemble techniques. Sequential ensemble techniques are used to construct models in sequential order. However, models are produced in parallel using parallel ensemble approaches [Rajbangshi, 2020].

Hence, for the first unsupervised problem, we use **Voting classifier** and Receiver operating characteristic **(ROC)** curves as ensemble methods, and for the second unsupervised problem, we apply Adaptive Boosting **(AdaBoost)** Classifier.

A Voting Classifier is an ML model that learns from an ensemble of models and predicts class based on the highest likelihood of the outcome [Kuwar, 2019]. The AdaBoost algorithm creates a model by assigning equal weight to all samples of the dataset. Then, the algorithm gives a higher weight to the incorrectly classified samples. In the next model, all points with increased weights are more important. It will continue to train models until a minor error is returned [Saini, 2021].

An ROC curve is created which displays a classification model's performance overall categorization levels. True Positive Rate ($TPR = TP/(TP + FN)$) and False Positive Rate ($FPR = FP/(FP + TN)$) are plotted on a ROC curve [Bhandari, 2020].

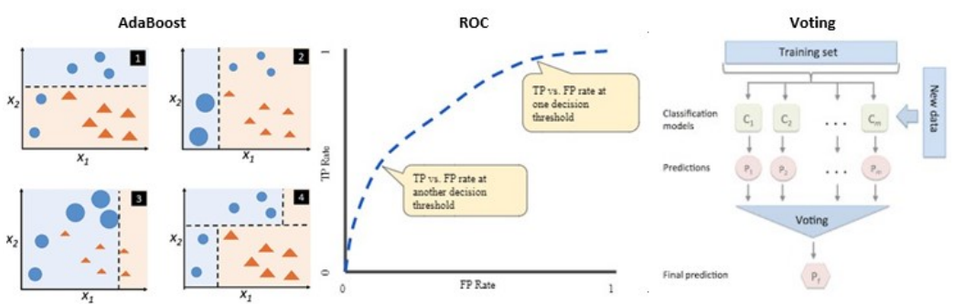


Figure 13: The Ensemble methods.

3.2 Unsupervised

To continue with, we deal with the unsupervised problems using the **Silhouette** and **Elbow** methods. These methods optimize the clusters created by the **k-means** algorithm, while the **Knee Locator** method verify that the amount of clusters is accurate.

3.2.1 The Elbow Method

Determining the correct number of clusters is significant in an unsupervised problem, and the elbow method is used [Bonaros, 2019]. The method locates the ideal number of clusters by calculating the within-cluster sum of squares to measure within-cluster variation [Gupta, 2021]. A plot consisting of the explained variation of the amount number of clusters is created. Hence, we identify the elbow of the curve as the number of sets to use [Kleine, 2021].

3.2.2 Silhouette Method

The Silhouette Coefficient is a statistic measure determining how similar data points are within a cluster compared to other clusters [Chaudhary, 2020]. The Silhouette Coefficient's value is between -1 and 1 [Banerji, 2021]. Note that 1 indicates that clusters are well separated, 0 suggests that clusters are indifferent and -1 indicates that clusters are misallocated [Bhardwaj, 2020].

The formula that deters the silhouette coefficient for a given data point is:

$$S_i = \frac{b_i - a_i}{max(b_i, a_i)}$$

where S_i is the silhouette coefficient of the samples,
a_i is the average distance between i and all other data points in the cluster to which i,
b_i is the average distance between i and all clusters to which i does not belong [Kumar, 2020].

3.2.3 Knee Locator

The Knee Locator method is used to ensure that the Elbow method's desired number of clusters is appropriate.The method denotes the point at which a greater k stops providing useful information and makes distinguishing clusters more complex [MWITI, 2020]. Unfortunately, this critical point is not always simple to recognize on some datasets [Clarke, 2021]. The Knee Locator method is also used to locate the knee point numerically, instead of examining this manually [Kleine, 2021].

4 Results

This section will examine the implications of each model to answer the three central questions based on Thera's Bank dataset. Table 2 refers to the definitions that guide our results.

	Definitions	**Formulas**
Confusion Matrix	A matrix that displays the performance of an algorithm.	
Accuracy	The number of correctly predicted sample points out of all the data points is accuracy.	$\frac{TP+TN}{TP+FP+FN+TN}$
Recall	The percentage of real positive values that are correctly identified is referred to as recall.	$\frac{TP}{TP+FN}$
Precision	The percentage of positive identifications that were actually correct is precision.	$\frac{TP}{TP+FP}$
F1 score	The weighted average of Precision and Recall is the F1 Score.	$\frac{2*(Recall*Precision)}{(Recall+Precision)}$

Table 2: Important Definitions [Joshi, 2016].

4.1 Decision Tree

The Decision Tree algorithm assists to classify whether a customer will accept a personal loan or not. A DT is a model with nodes and edges that can solve classification problems. Each node represents an outcome or is used to make a choice [Sakkaf, 2020].

The primary step is to verify the dataset's balance using the target column (Personal Loan). The number of customers who receive a personal loan against those who do not, demonstrate an imbalanced dataset, since only 273 customers out of 5000 accept it, as seen in Figure 14.

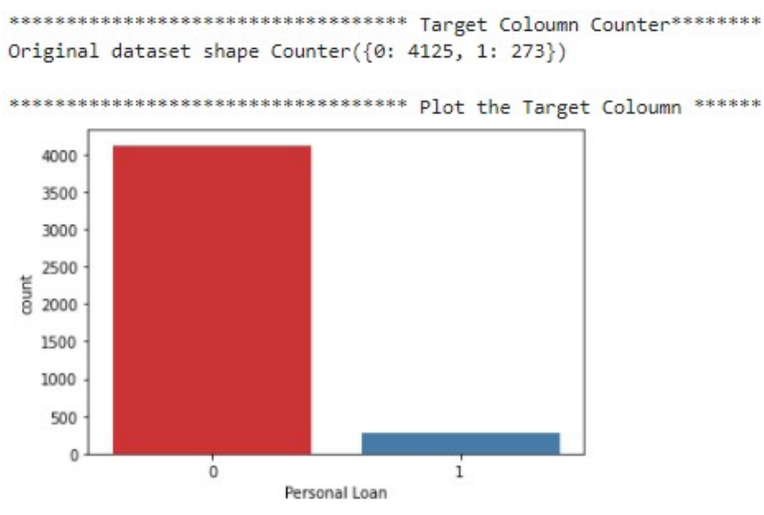


Figure 14: Personal Loan Counter.

According to Figure 16, each attribute is grouped by the 'Personal Loan' and the imbalance of the dataset is verified, since the number of clients who accept the loan is fewer than the total number of customers who will not.

```
for plot in feture_name:
  sns.histplot(decision_tree, x=plot, hue='Personal Loan', multiple='stack', palette = 'Set1')
  plt.show()
```

Figure 15: Python code for the target.

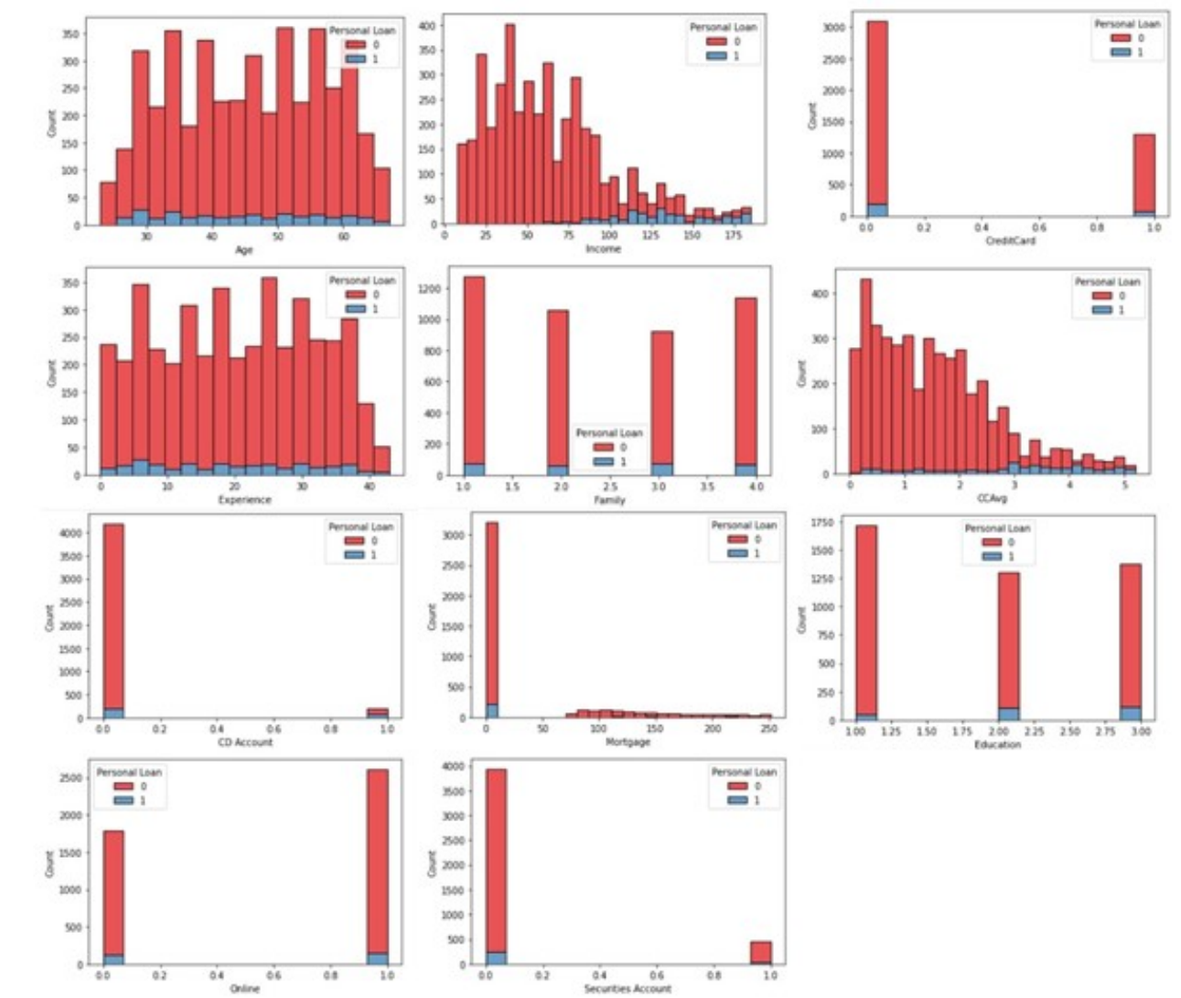


Figure 16: Plot each attribute based on the target Personal Loan.

As shown in Figure 16, all attributes have a sliding percentage of clients who will accept the personal loan. For instance, the 'Credit Cart' feature contains two types that reflect whether a client uses a credit card provided by another bank or not. As indicated in $\alpha_{1,3}$ position of the plots, no more than 250 customers who do not have a credit card from another bank will take the loan. And, if some customers do have a credit card from another bank, the number of customers who will get the loan will be fewer than 250.

The following results were obtained by the **train-test-split** method with 70% training samples and a simple DT model (without the optimum hyperparameters, CV, SMOTE method).

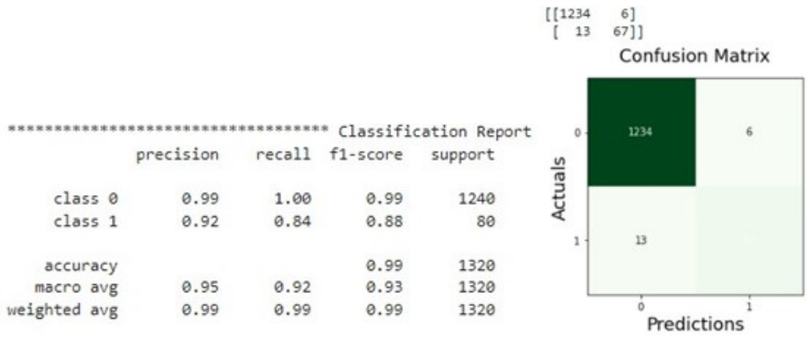


Figure 17: Classification report for a simple DT model.

The classification report shows DT accuracy's value equal to 0.99 and based on the definitions in Table 2, this indicates that the model correctly identifies 1234 out of 1320 customers. Recall, precision, and f1-score, on the other side, are poor in comparison to accuracy. The recall value, for instance, displays that 0.84 percent of true positive values are correctly identified. This is related to the imbalance dataset since the number of clients who accept the loan (class 1) is fewer than those who do not (class 0). Hence, our DT model need to be improved to achieve higher recall, precision, and f1-score values.

Feature selection and Grid-Search with cross-validation is used to find more accurate results according to the significant features and the appropriate hyper-parameters of the model.

```
#Identify input features having high correlation with target variable
import numpy as np
importances = Features.apply(lambda x: x.corr(Target))
indices = np.argsort(importances)
print(importances[indices])

#Plot the results
plt.title('Correlation with Target')
plt.barh(range(len(indices)), importances[indices], color='g', align='center')
plt.yticks(range(len(indices)), [feture_name[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```

Figure 18: Python code for feature selection.

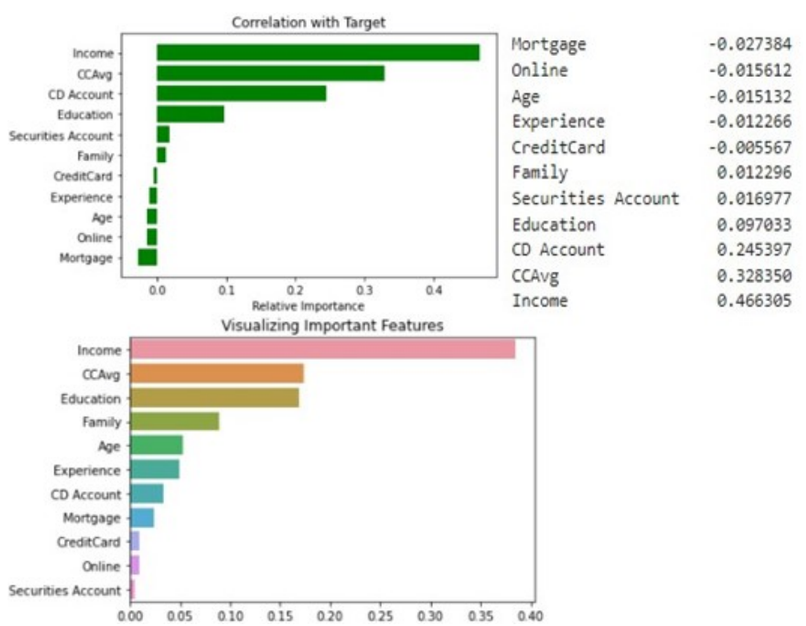


Figure 19: Correlation of each feature with the target.

Figure 19 demonstrates the connection between each attribute and the target column. As is shown, the strongest correlation with the target has the 'Income' attribute (0.46), while the weakest connection is reached by the 'Credit Card' variable (-0.005). The Random Forest model provides a more detailed field of which attribute has a poor relationship with the target. Hence, by comparing the two methods, we can conclude that 'Online', 'CreditCard' and 'Securities Account' features have less correlation with the target. As a result, GridSearch is applied with only the 'best' features to identify the hyper-parameters. Note that the results give the 'entropy' as the best criterion used, and the ideal maximum depth of the DT as nine.

The following results were obtained by 10-fold **Cross-validation** with the optimal hyper-parameters for the DT model.

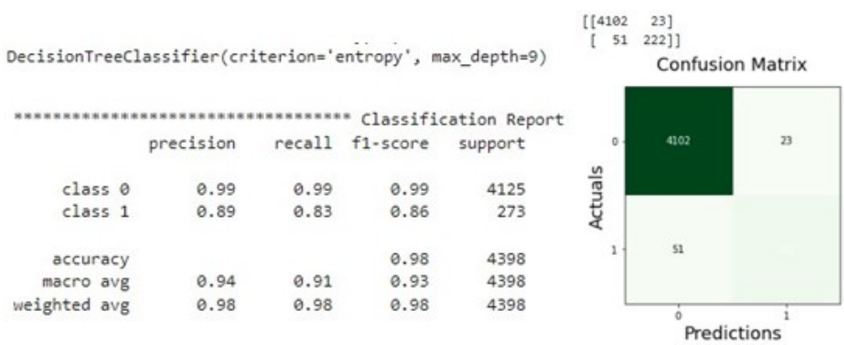


Figure 20: The results of DT with CV.

Figure 20 demonstrates that while DT accuracy remains high (0.98), precision and recall scores have dropped to 0.89 and 0.83, respectively. Consequently, the f1-score drops from 0.88 to 0.86. It is worth mentioning that the model's performance remains unchanged due to CV employing all of the dataset samples and the difference between the pre-optimized and post-optimized values is not significant. These findings show that even though we use the optimal hyper-parameters, the results are roughly similar to the model created in the first place. However, Cost-Complexity Pruning and SMOTE method are applied to build new models and to be compared.

The SMOTE method balances both the attributes and the target, meaning that the customers who accept the loan would be equal to those who do not. Therefore, the cases in the dataset are doubled, as Figure 21 represents.

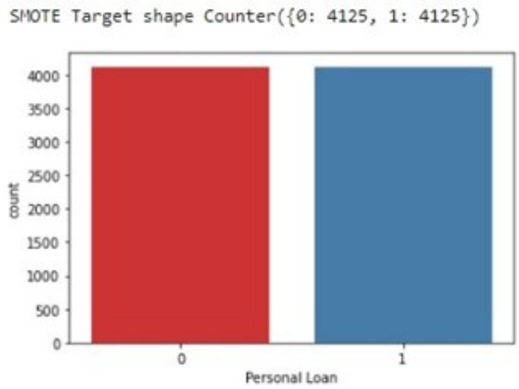


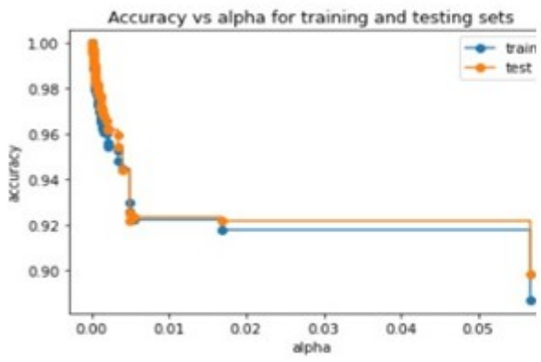
Figure 21: The target with SMOTE method.

To continue with, using the new dataset and after the feature selection with CV, the RF algorithm concludes to the same attributes that have low correlation with the target. Hence, determining the best hyper-parameters is necessary, including also the new parameter called complexity parameter, α. As Figure 22 illustrates, the best criterion is entropy, the optimal depth is 14, and the α value equals to 0.000222.

```
The best score is:  0.9867878787878788
The best estimator is:  DecisionTreeClassifier(ccp_alpha=0.00022222222222222223, criterion='entropy',
                        max_depth=14)
```

Figure 22: GridSearch with CV and SMOTE method.

This report employs the Cost-Complexity Pruning method to ensure that the alpha value is 0.000222.




```
ccp_alphas:
 [0.00000000e+00 8.02349739e-05 8.06285073e-05 9.69134174e-05
 9.69696970e-05 1.01010101e-04 1.03896104e-04 1.09090909e-04
 1.14081996e-04 1.14832536e-04 1.15864528e-04 1.16550117e-04
 1.16550117e-04 1.17647059e-04 1.17695941e-04 1.17748918e-04
 1.17936118e-04 1.18686869e-04 1.19303269e-04 1.20331358e-04
 1.20721384e-04 1.29292929e-04 1.55844156e-04 1.57713466e-04
 1.61616162e-04 1.61616162e-04 1.61616162e-04 1.61616162e-04
 1.81818182e-04 1.93939394e-04 2.02812438e-04 2.09859792e-04
 2.15488215e-04 2.15833549e-04 2.18295487e-04 2.22414622e-04
 2.24685883e-04 2.26783969e-04 2.29798501e-04 2.31884058e-04
 2.32727273e-04 2.34604106e-04 2.38910848e-04 2.67442684e-04
 2.85502080e-04 2.85936286e-04 2.96442688e-04 3.03030303e-04
 3.16160919e-04 3.18181818e-04 3.27431122e-04 3.46320346e-04
 3.46320346e-04 3.47931139e-04 3.66431482e-04 3.69482787e-04
 3.76674306e-04 3.83616384e-04 3.96694215e-04 4.13737374e-04
 4.15584416e-04 4.15584416e-04 4.31372549e-04 4.40030558e-04
 4.43788365e-04 4.48276617e-04 4.54545455e-04 4.55858064e-04
 4.71585152e-04 4.77272727e-04 4.92527664e-04 5.90909091e-04
 6.49471459e-04 6.57493152e-04 7.15151515e-04 7.44207949e-04
 7.63601696e-04 7.98585433e-04 8.38913256e-04 9.31289304e-04
 1.15295648e-03 1.20492738e-03 1.22431764e-03 1.27281932e-03
 1.40438871e-03 1.45858586e-03 1.59887469e-03 1.86903117e-03
 2.10083555e-03 2.19912490e-03 3.35367173e-03 3.40133478e-03
 3.89053324e-03 4.88578584e-03 4.91306520e-03 5.55923548e-03
 1.68267466e-02 5.67604416e-02]
```

Figure 23: Alpha Values.

Figure 23 displays that when the alpha value is close to zero, the tree performs relative 100% training accuracy and close 100% testing accuracy. However, as α value increases, the accuracy decreases. Moreover, by examining α values with high accuracy, we can determine that the superior α value is the same as that of GridSearch. The ideal hyper-parameters have been identified, and the optimal DT model has the following results when 10-fold CV with the SMOTE method is used:

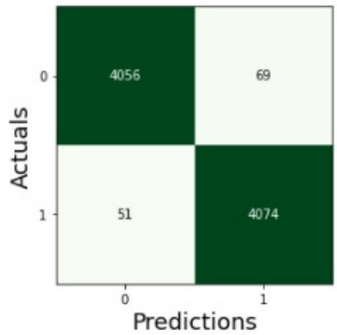


Figure 24: The confusion matrix with all the applying methods.

********** Classification Report **********

	precision	recall	f1-score	support
class 0	0.99	0.98	0.99	4125
class 1	0.98	0.99	0.99	4125
accuracy			0.99	8250
macro avg	0.99	0.99	0.99	8250
weighted avg	0.99	0.99	0.99	8250

****

Accuracy	Recall	Precision	F1-score
0.986667	0.987864	0.978469	0.986731
0.985455	0.990291	0.985542	0.986699
0.983030	0.978155	0.982968	0.981774
0.985455	0.985437	0.987835	0.985437
0.978182	0.983010	0.973621	0.979444
0.984242	0.992736	0.980815	0.984356
0.990303	0.995157	0.987981	0.991556
0.986667	0.983051	0.990244	0.987864
0.989091	0.980630	0.992683	0.989064
0.986667	0.992736	0.987893	0.987864

Figure 25: The results after applying all the methods on the dataset.

In Figure 25, the results of the SMOTE method using the hyper-parameters are displayed, consisting of the accuracies, recalls, precisions, and f1-scores for each fold, and also the mean of them, which correspond to the result given by the 10-fold CV. As is shown, all the values are higher due to the True Positives and False Negatives values are 4056 and 4074, respectively, out of 8250 samples, according to the confusion matrix in Figure 24. Furthermore, the False Positives sample count is 51, while the True Negatives sample count is 69.

The visualization of the Decision Tree:

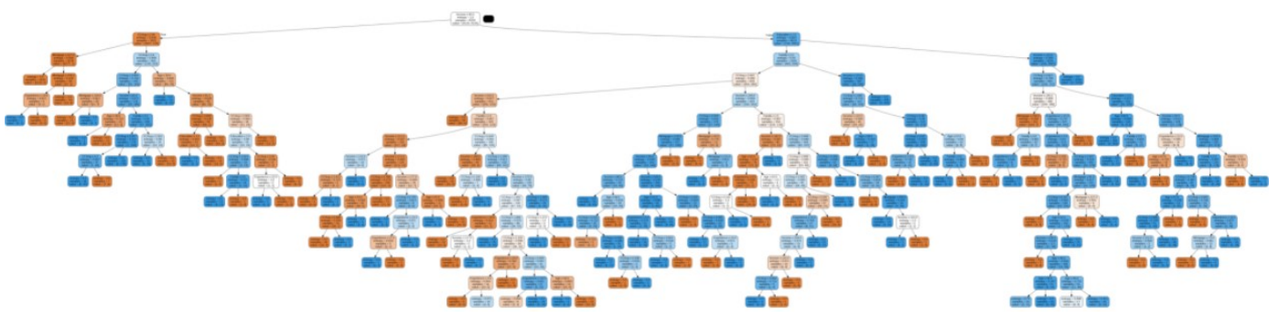


Figure 26: Visualise the DT.

Comparing the DT model against other ML algorithms is the final phase in solving the first scenario. This report uses the features and target column obtained by the SMOTE method to identify the optional hyperparameters for each algorithm to make a more accurate comparison between them.

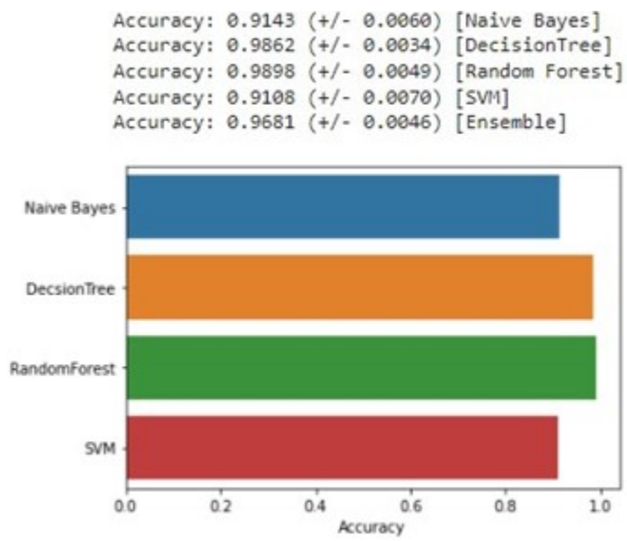


Figure 27: Ensemble Learning for the first scenario.

After applying 10-fold cross-validation to train each model, RF has the highest accuracy (0.9898) and surpasses the other algorithms in this dataset. The decision tree method has a similar accuracy (0.9862) to RF. Furthermore, the accuracies of the Naive Bayes (NB) and SVM algorithms are 0.9143 and 0.9108, respectively, suggesting that if one of these algorithms is used to solve the first scenario, it will perform poorly in comparison to the DT and RF algorithms. Finally, the accuracy of the Voting Classifier is 0.9681, demonstrating that the algorithm classifies the sample using both classes.

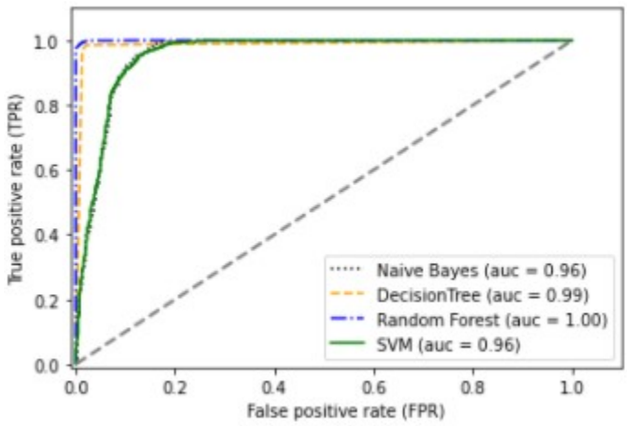


Figure 28: ROC curve.

The RF classifier performs well on the test dataset ($AUC = 1.00$), as seen by the ROC curve in Figure 28. Furthermore, as shown in Figure 27, the DT classifier performs well ($AUC = 0.99$) on the same dataset, considering the low standard deviation. In this dataset, however, NB and SVM perform poorly.

4.2 Support Vector Machine

The second scenario uses SVM algorithm to determine whether or not a client has a certificate of deposit account (CD-account) with the bank. The SVM classifier divides data

points using the hyperplane with the most significant margin; hence, SVM determines an ideal hyperplane that assists in the classification of new data points [Navlani, 2019]. The second scenario uses a similar strategy as the first scenario.

Figure 29 displays the dataset's balance using the target column. The number of customers who have a CD-account with the bank or not demonstrates an imbalanced dataset, since only 207 customers out of 5000 have a CD-account.

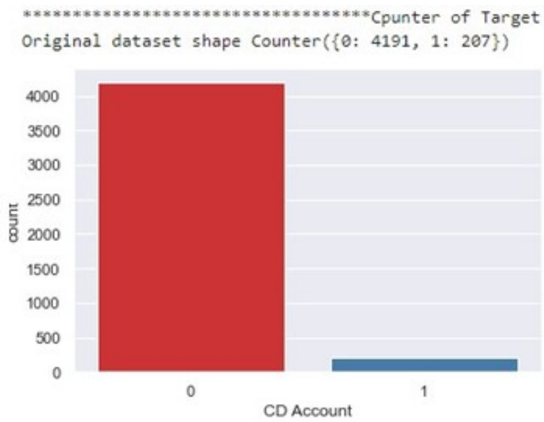


Figure 29: CD Account counter.

By grouping, each feature with the target column verifies the signs of an imbalance dataset once again. Figure 30 illustrates that all attributes have a sliding percentage of clients who have a CD-account with the bank.

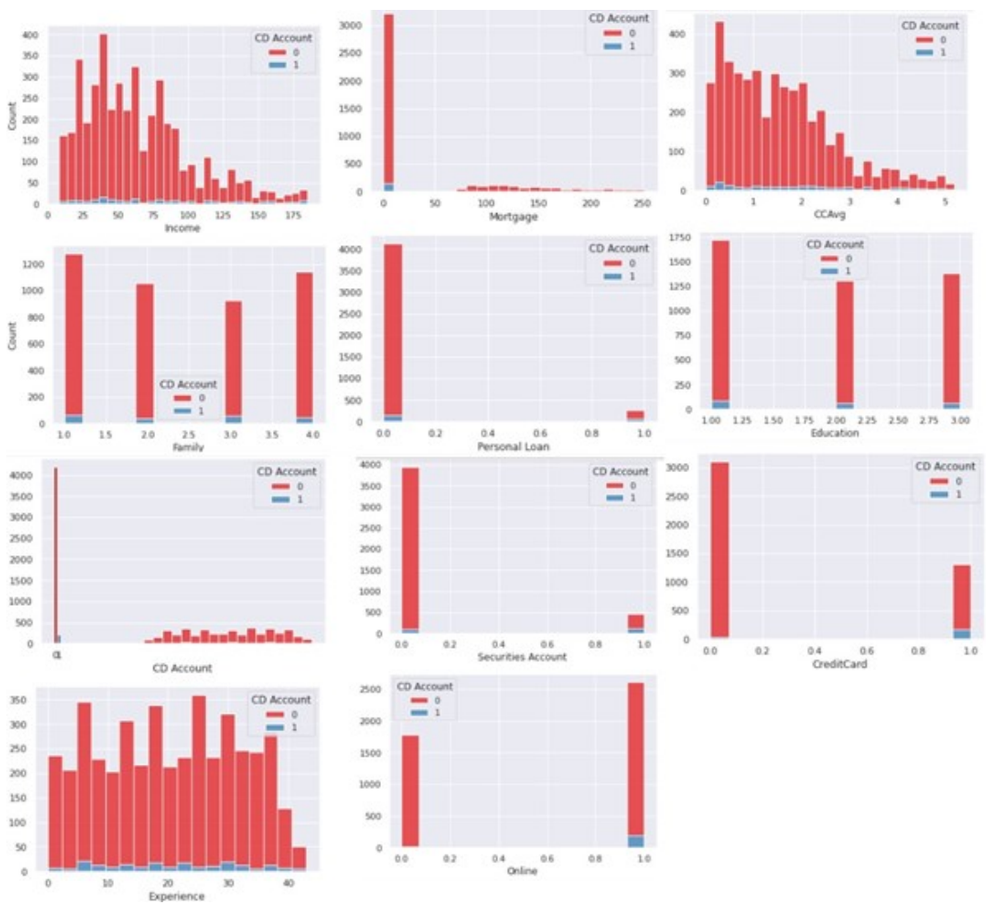


Figure 30: Plot each attribute based on target.

The train-test-split method obtained the following results with 70% training and 30% testing samples and applied them in a simple SVM model.

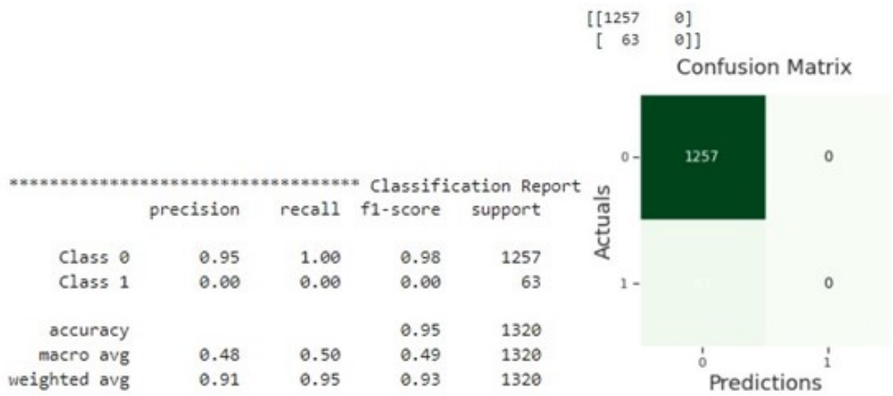


Figure 31: Classification Report and Confusion Matrix for a simple SVM model.

In Figure 31, the accuracy score is 0.95, suggesting that the SVM model correctly classifies 1257 out of 1320 customers who do not have a CD-account. On the other hand, the recall, precision, and f1-score are all zero, indicating that the SVM model does not categorize any customers with a bank CD-account. This can be justified because only a few customers have a bank's CD-account, as Figure 29 shows. Hence, the SVM algorithm must be evaluated.

To start with, feature selection is the first step of the model evaluation.

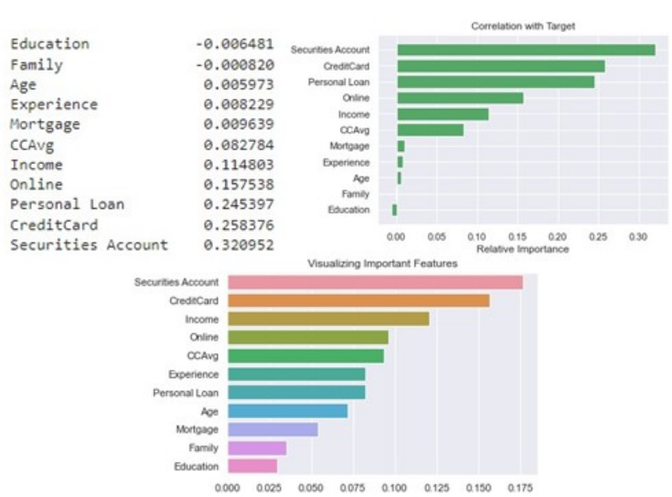


Figure 32: Feature Selection with a Filter Method and RF model for SVM.

As shown in Figure 32, the 'Securities Account' attribute has the highest association with the target with a value of 0.320952, while the 'Education' and 'Family' attributes have the poorest correlation with the target having values equal to -0.006481 and -0.000820, respectively. Meanwhile, the RF model reveals that the target column has the least connection with 'Education' and 'Family'. Consequently, we may conclude that 'Education' and 'Family' should be not considered in the model. Finally, after identifying the best features, Gridsearch with 10-fold CV determines the optimal hyper-parameters to build the model. Therefore, linear

kernels with C value equal to 0.1 and γ value equal to 1 are the optimal hyper-parameters for the SVM model, as Figure 33 shows.

```
********** The Best Hyperparameters via GridSearch and 10 fold CV
The best hyperparameters are:  {'C': 0.1, 'gamma': 1, 'kernel': 'linear'}
```

Figure 33: The optimal Hyperparameters for SVM.

Next, we implement a 10-fold Cross-validation using the optimal hyper-parameters for the SVM model and the strongly correlated features with the target.

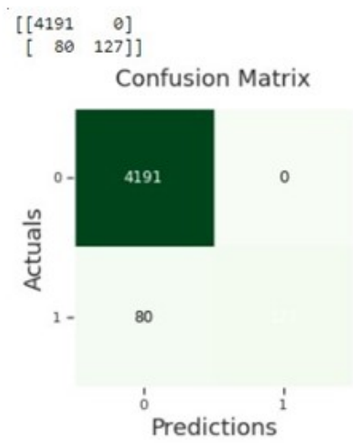


Figure 34: Confusion matrix with 10-fold CV and the optimal hyperparameters.

The confusion matrix in Figure 34 displays a significant improvement compared to the confusion matrix in Figure 31. Hence, the True Positive value is 4191 since CV employs 70% of the dataset's samples. However False Negative value is 127, indicating that 127 customers with a CD account with the bank are identified correctly. Instead, as seen in Figure 31, the model fails to properly predict any costumers having a bank CD account.

	precision	recall	f1-score	support
Class 0	0.98	1.00	0.99	4191
Class 1	1.00	0.61	0.76	207
accuracy			0.98	4398
macro avg	0.99	0.81	0.88	4398
weighted avg	0.98	0.98	0.98	4398

	Accuracy	Recall	Precision	F1-score
0	0.979545	0.550000	1.0	0.709677
1	0.977273	0.523810	1.0	0.687500
2	0.988636	0.761905	1.0	0.864865
3	0.986364	0.714286	1.0	0.833333
4	0.977273	0.523810	1.0	0.687500
5	0.979545	0.571429	1.0	0.727273
6	0.977273	0.523810	1.0	0.687500
7	0.981818	0.619048	1.0	0.764706
8	0.984055	0.650000	1.0	0.787879
9	0.986333	0.700000	1.0	0.823529

Figure 35: The Classification Report for SVM.

SVM's accuracy has improved to 0.98, while precision and recall have increased to 1 and 0.61, respectively, as Figure 35 illustrates. As a result, the f1-score jumps from 0 to 0.76. Since the prior results (Figure 31) were not greater than 0, these findings suggest that feature selection, GridSearch, and CV have assessed the model. However, the recall, accuracy, and f1-score values are insufficient. This report will employ SMOTE, Random over and under sampling to study if the model can assess more.

ROS (Random Over Sampling) and RUS (Random Under Sampling) are applied to the features and the target to evaluate the model. The following results were obtained by applying Random over and under sampling methods with a simple SVM model.

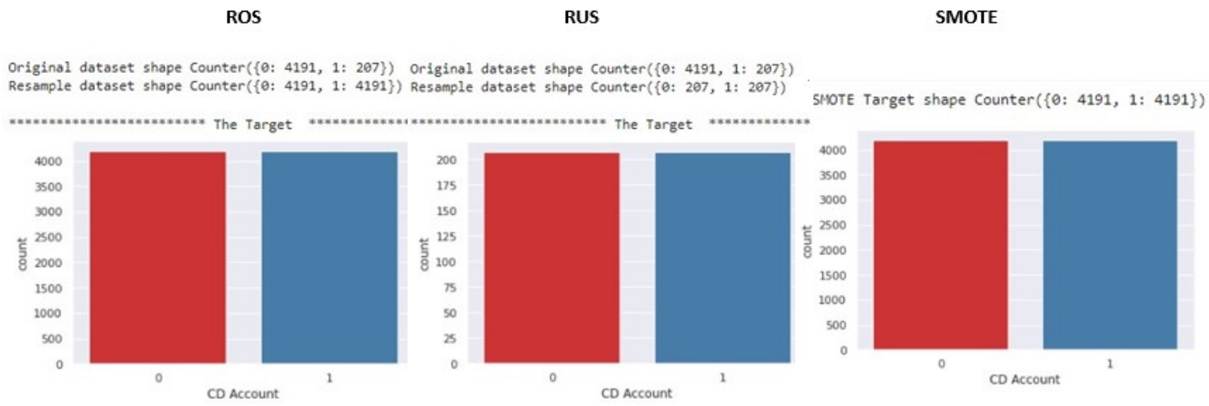


Figure 36: ROS and RUS target column.

Clients who have a CD-account are treated equally to those who do not, as seen in Figure 36. However, SMOTE and ROS split the dataset's samples in half, giving 8382 samples in total. On the other hand, RUS's dataset has just 414 samples, which represent customers who have or do not have a CD account.

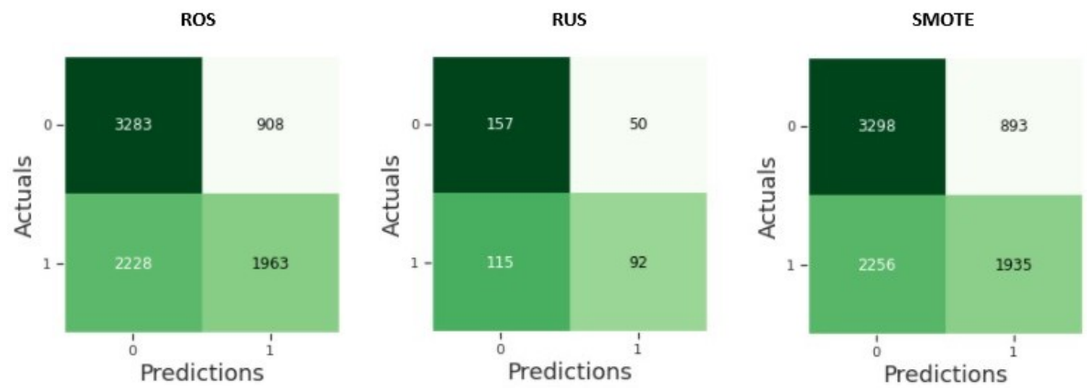


Figure 37: The confusion matrices for ROS, RUS and SMOTE method.

ROS

	precision	recall	f1-score	support
Class 0	0.60	0.78	0.68	4191
Class 1	0.68	0.47	0.56	4191
accuracy			0.63	8382
macro avg	0.64	0.63	0.62	8382
weighted avg	0.64	0.63	0.62	8382

RUS

	precision	recall	f1-score	support
Class 0	0.58	0.76	0.66	207
Class 1	0.65	0.44	0.53	207
accuracy			0.60	414
macro avg	0.61	0.60	0.59	414
weighted avg	0.61	0.60	0.59	414

SMOTE

	precision	recall	f1-score	support
Class 0	0.59	0.79	0.68	4191
Class 1	0.68	0.46	0.55	4191
accuracy			0.62	8382
macro avg	0.64	0.62	0.61	8382
weighted avg	0.64	0.62	0.61	8382

Figure 38: The classification reports ROS and ROS, RUS and SMOTE method.

None of the approaches optimize the model, as seen in Figures 37 and 38. Both the ROS and SMOTE methods aim to increase the number of minority samples. As a result, the outcomes of those methods differ slightly. The ROC, RUS, and SMOTE accuracies are 0.63, 0.60, and 0.62, respectively; the poor accuracies are due to many mispredicted samples. For instance, the ROS method correctly predicts 5246 ($TP + FN$) samples for both categories, whereas 3136 ($TP + FN$) samples out of 8382 are incorrectly predicted. Furthermore, RUS has the lowest accuracy and the lowest precision, recall, and f1-score values. Due to the small number of samples, the accuracy, recall, and f1-score are 0.65, 0.4, and 0.53, respectively. Since the model correctly predicts just 249 ($TP + FN$) observations, the remaining ones are misclassified. As a result, the SVM model is not performing the best model to solve this scenario. Therefore, ensemble learning will assist in the identification of the best model to handle this scenario.

When there is an imbalance in the observations between the two classes, reviewing precision and recall is beneficial; hence, the precision and recall calculations are only focused on the correct prediction of the minority class [Brownlee, 2018].

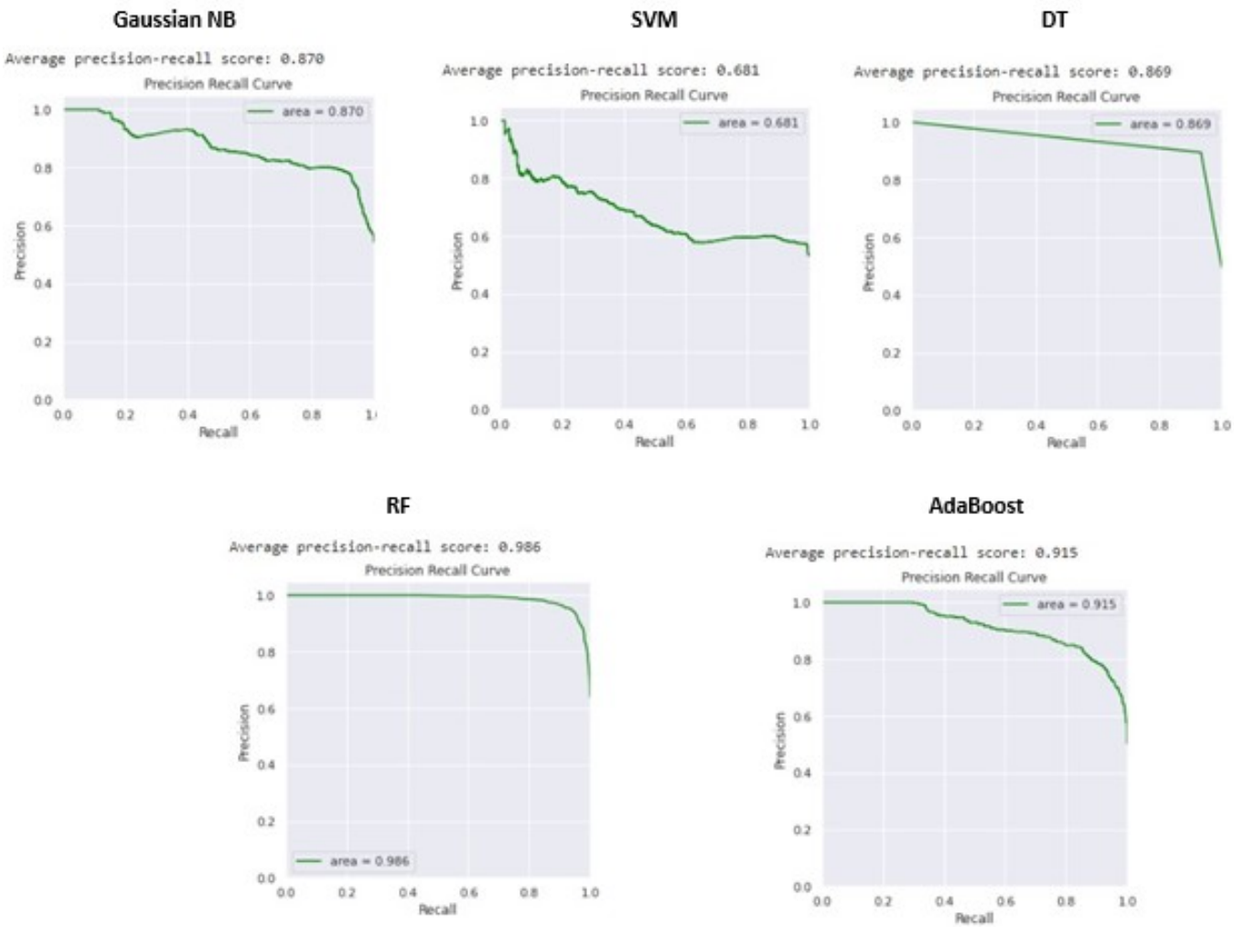


Figure 39: Ensemble Learning for the Second Scenario.

**********The Ensemble DataFrame*******

	f1score	precisionscore	recallscore	avg_pre_rec
GaussianNB	0.771620	0.802620	0.742926	0.865381
SVC	0.553271	0.655592	0.478577	0.672180
DT	0.921608	0.907524	0.936136	0.880977
RF	0.939130	0.918794	0.960388	0.985225
AdaBoost	0.849749	0.813609	0.889248	0.910099

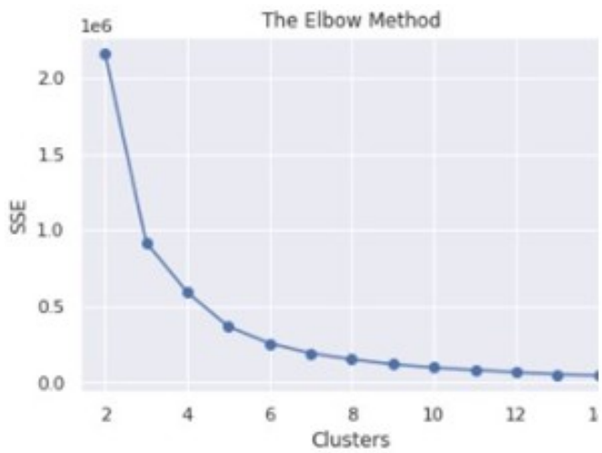
Figure 40: Recall, Precision,f1-score.

Figure 39 and 40 represents the average precision-recall score for the Gaussian NB, SVM, DT, RF, and AdaBoost classifiers, which are 0.865, 0.672, 0.880, 0.985, and 0.910, respectively. Hence, the RF model obtained the highest score, indicating that the positive predicted samples outnumbered misclassified samples significantly. Furthermore, the DT and Gaussian NB classifiers perform magnificently on this dataset. Hence, we conclude that the choice of using the SVM model to solve the second scenario was incorrect since it has the lowest score and it does not perform well for this dataset.

4.3 Clustering

Customers are classified using the K-Means algorithm based on their annual income and typical monthly expenditure, regardless of whether they have a securities account with the bank. The K-means algorithm divides the dataset into K clusters, with each data point belonging to just one [Dabbura, 2018].

The attributes 'Income,' 'CCAvg,' and 'Security Account' will address the third scenario. The Elbow method indicates that as k rises, the SSE (sum of square error) inside the cluster decreases. In order to determine the k value we choose the number where the SSE begins to decrease rapidly. Hence, the number of clusters is not clearly illustrated in this case because the SSE starts to drop between the **fourth** and **fifth** clusters, as seen in Figure 41. However, the difference in SSE between the two potential cluster numbers is slight, since if k equals four, the SSE is $5.877559e+05$, but if k equals five, the SSE is $3.648253e+05$.



Cluster	sse
2	2.156372e+06
3	9.115092e+05
4	5.877559e+05
5	3.648253e+05
6	2.505367e+05

Figure 41: The elbow method.

The Silhouette coefficient and Knee locator is another way to determine the correct amount of groups. Based on the Silhouette coefficient in Figure 42, there is a sign that five is the optimal amount of clusters since the coefficient for five clusters (0.563) is higher than the coefficient for four clusters (0.558). However, the difference between the two coefficients is not significant, so the solution for this dilemma comes from the Knee Locator method that displays the most appropriate value for the clusters $(k = 5)$.

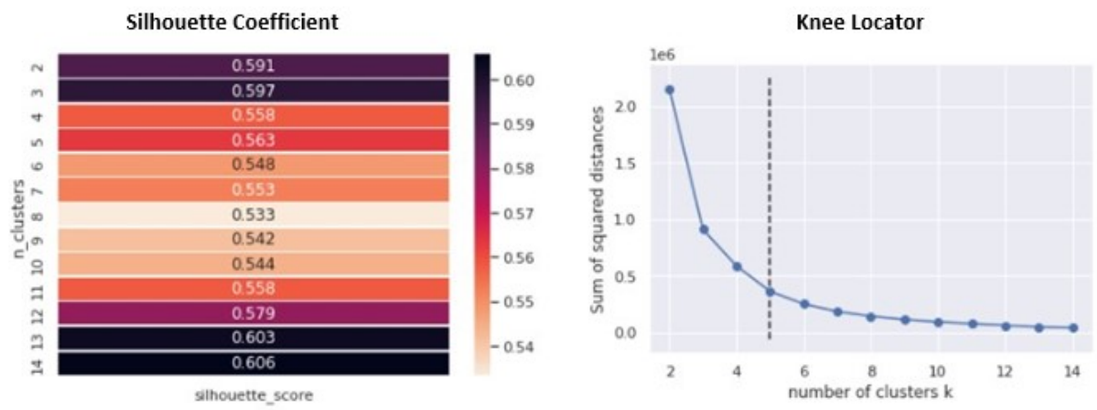


Figure 42: The silhouette coefficient and Knee Locator.

The final step of this scenario is to analyze the clusters, that have been created based on the above methods and compare them with the amount of clients that have a security account with the bank or not.

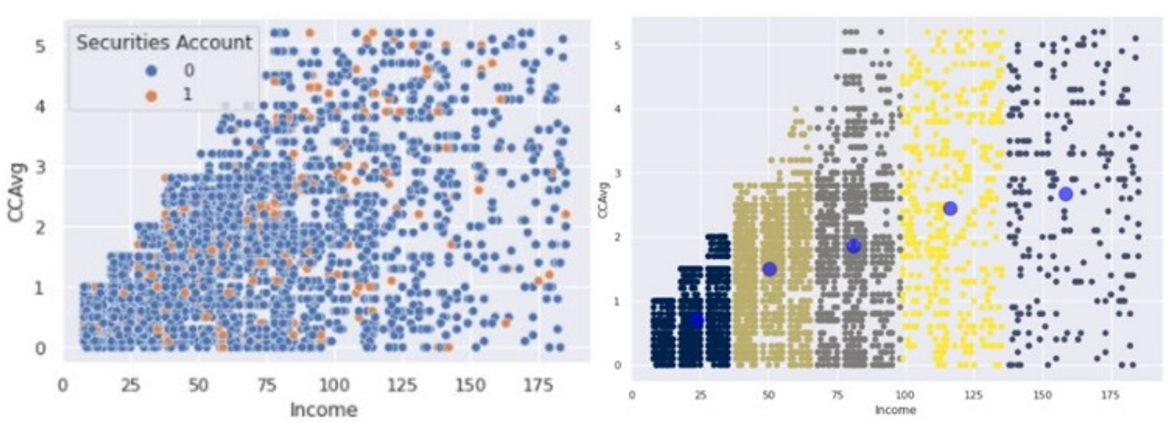


Figure 43: The Clusters and the solution of the second scenario.

The clients are split into five groups based on their similarities, as the right-hand side of Figure 43 displays. For instance, the first group has clients whose annual income is not higher than $30000, and most of the clients have average spending on credit cards monthly equal to or lower than $2000. Therefore, this group's majority of clients will not have a security account with the bank, as confirmed from the left-hand side of Figure 43. The second group's annual income ranges from $30000 to $70000, and their customers' average monthly credit card expenditure is no more than $4000. As a result, the second cluster has a similar number of costumers with bank security accounts. Continuing along this line, the 3^{rd}

and 4^{th} groups consists of the most clients who have a security account with the bank because the clients in those two clusters have yearly incomes ranging from \$70000 to \$100000 and \$100000 to \$130000, respectively. Furthermore, neither group's average monthly spending surpasses \$55000. Consequently, such customers choose to open a bank security account since their monthly expenses are low compared to their yearly income.

5 Conclusion

The management of Thera Bank intends to look at ways to transform its clients into personal loan customers while maintaining them as deposits. This report examines three distinct situations based on Thera Banks' dataset and several ways for evaluating the solution for each scenario. The decision Tree model assist to determine whether or not a consumer will accept a personal loan. The SVM model then identifies whether or not a customer has a bank certificate of deposit account. Finally, regardless of whether they have a securities account with the bank, customers are categorized using the K-Means algorithm based on their yearly income and normal monthly consumption.

Please find the link for Python code:

https://colab.research.google.com/drive/1SAmF8d8DR4qYijFzod2Sg_xj0Pn90s_v?usp=sharing

Please find the link for One Drive file:

https://mailbcuac-my.sharepoint.com/:f:/g/personal/marios_kyriacou_mail_bcu_ac_uk/EgX2zkcJlJxFohcy9ZGCeuYBtiFQYgYb_2-5RwlvXuhuOA?e=3YFWyh

References

[Kaggle, 2021] Kaggle. (2021) Bank Personal Loan Modelling. Available at: https://www.kaggle.com/krantiswalke/bank-personal-loan-modelling [Accessed 18 December 2021].

[Milo and Somech, 2020] Milo, T. and Somech, A., 2020, June. Automating exploratory data analysis via machine learning: An overview. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (pp. 2617-2622).

[Admin, 2020] Admin.J (2020) Exploratory Data Analysis(EDA) in Python!. Available at: https://www.analyticsvidhya.com/blog/2020/08/exploratory-data-analysiseda-from-scratch-in-python/[Accessed 18 December 2021].

[Edureka, 2021] edureka (2021) Exploratory Data Analysis(EDA) in Python!. Available at: https://www.edureka.co/blog/exploratory-data-analysis-in-python/need[Accessed 18 December 2021].

[Last and Kandel, 2021] Last, M. and Kandel, A., 2001, November. Automated detection of outliers in real-world data. In Proceedings of the second international conference on intelligent technologies (pp. 292-301).

[Sharma, 2018] Sharma.N (2018) Ways to Detect and Remove the Outliers. Available at: https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba [Accessed 18 December 2021]

[Wijaya, 2020] Wijaya.C.Y (2020) Outlier — Why is it important?. Available at: https://towardsdatascience.com/outlier-why-is-it-important-af58adbefecc [Accessed 18 December 2021].

[Bougoudis,Demertzis and Iliadis, 2016] Bougoudis, I., Demertzis, K. and Iliadis, L., 2016. HISYCOL a hybrid computational intelligence system for combined machine learning: the case of air pollution modeling in Athens. Neural Computing and Applications, 27(5), pp.1191-1206.

[Taylor and Courtney, 2020] Taylor, Courtney. (2020). What Is the Interquartile Range Rule?. Available at: https://www.thoughtco.com/what-is-the-interquartile-range-rule-3126244 [Accessed 19 December 2021].

[Nair, 2019] Nair.P (2019) Hands-on : Outlier Detection and Treatment in Python Using 1.5 IQR rule. Available at: https://medium.com/@prashant.nair2050/hands-on-outlier-detection-and-treatment-in-python-using-1-5-iqr-rule-f9ff1961a414 [Accessed 19 December 2021].

[PELTARION, 2020] PELTARION (2020) Feature distribution. Available at:https://peltarion.com/knowledge-center/documentation/datasets-view/edit-an-imported-dataset-for-use-in-experiments/feature-distribution [Accessed 20 December 2021].

[Vieira and Gomes, 2010] Vieira, E.S. and Gomes, J.A., 2010. Citations to scientific articles: Its distribution and dependence on the article features. Journal of Informetrics, 4(1), pp.1-13.

[Malik, 2019] Malik.F (2019) Ever Wondered Why Normal Distribution Is So Important?. Available at:https://medium.com/fintechexplained/ever-wondered-why-normal-distribution-is-so-important-110a482abee3[Accessed 22 December 2021].

[MAJASKI, 2021] MAJASKI.C (2021) Hypothesis Testing. Available at:https://medium.com/fintechexplained/ever-wondered-why-normal-distribution-is-so-important-110a482abee3[Accessed 22 December 2021].

[CHEN, 2021] CHEN.J (2021) Normal Distribution. Available at:https://www.investopedia.com/terms/n/normaldistribution.asp[Accessed 23 December 2021].

[Jain and Singh, 2003] Jain.S.K., Singh V.P. (2003) Water Resources Systems Planning and Management. Available at:https://www.sciencedirect.com/topics/earth-and-planetary-sciences/normal-density-functionsp[Accessed 24 December 2021].

[Wilk and Gnanadesikan, 1968] Wilk, M.B. and Gnanadesikan, R., 1968. Probability plotting methods for the analysis for the analysis of data. Biometrika, 55(1), pp.1-17.

[Chaudhary, 2019] Chaudhary M K, (2019) qqplot (Quantile-Quantile Plot) in Python. Available at: https://www.geeksforgeeks.org/qqplot-quantile-quantile-plot-in-python/[Accessed 02 January 2022].

[Glen, 2016] Glen.S, (2016) Kolmogorov-Smirnov Goodness of Fit Test. Available at: https://www.statisticshowto.com/kolmogorov-smirnov-test/[Accessed 02 January 2022].

[Bhalla, 2019] Bhalla.C, (2019) CALCULATE KS STATISTIC WITH PYTHON. Available at: https://www.listendata.com/2019/07/KS-Statistics-Python.html/[Accessed 02 January 2022].

CALCULATE KS STATISTIC WITH PYTHON Deepanshu Bhalla 5 Comments Python

[Bock, 2019] Bock.T, (2019) What is a Correlation Matrix?. Available at: https://www.displayr.com/what-is-a-correlation-matrix/[Accessed 23 December 2021].

[Glen, 2019] Glen.S, (2016) What is a Correlation Matrix?. Available at: https://www.statisticshowto.com/correlation-matrix/[Accessed 23 December 2021].

[ZACH, 2020] ZACH, (2020) How to Read a Correlation Matrix. Available at: https://www.statology.org/how-to-read-a-correlation-matrix/[Accessed 23 December 2021].

[Mazumder, 2021] Mazumder.S, (2021) 5 Techniques to Handle Imbalanced Data For a Classification Problem. Available at: https://www.analyticsvidhya.com/blog/2021/06/5-techniques-to-handle-imbalanced-data-for-a-classification-problem/[Accessed 21 December 2021].

[Mazumder, 2019] Khandelwal.R, (2019) Feature selection in Python using the Filter method. Available at: https://towardsdatascience.com/feature-selection-in-python-using-filter-method-7ae5cbc4ee05[Accessed 26 December 2021].

[Gupta, 2020] Gupta.A, (2020) Feature Selection Techniques in Machine Learning. Available at: https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning/[Accessed 26 December 2021].

[SQLRelease, 2021] SQLRelease, (2021) Introduction to k-fold Cross-Validation in Python. Available at: https://sqlrelease.com/introduction-to-k-fold-cross-validation-in-python[Accessed 23 December 2021].

[Brownlee, 2018] Brownlee J, (2018) A Gentle Introduction to k-fold Cross-Validation. Available at: https://machinelearningmastery.com/k-fold-cross-validation/[Accessed 23 December 2021]

[SQLRelease, 2021] SQLRelease, (2021) An introduction to GridSearchCV and RandomizedSearchCV. Available at: https://sqlrelease.com/an-introduction-to-gridsearchcv-and-randomizedsearchcv[Accessed 23 December 2021]

[Joshi, 2016] Joshi R, (2016) Accuracy, Precision, Recall F1 Score: Interpretation of Performance Measures. Available at: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/[Accessed 24 December 2021]

[Dye, 2020] Dye S, (2020)How to Handle SMOTE Data in Imbalanced Classification Problems. Available at: https://towardsdatascience.com/how-to-handle-smote-data-in-imbalanced-classification-problems-cf4b86e8c6a1[Accessed 25 December 2021]

[Joshi, 2021] Tyagi K, (2021) ML | Handling Imbalanced Data with SMOTE and Near Miss Algorithm in Python. Available at: https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/[Accessed 25 December 2021]

[Brownlee, 2020] Brownlee J, (2020)SMOTE for Imbalanced Classification with Python. Available at: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/[Accessed 25 December 2021]

[MALATOOCTOBER, 2021] MALATOOCTOBER G, (2021)Feature selection with Random Forest. Available at: https://www.yourdatateacher.com/2021/10/11/feature-selection-with-random-forest/[Accessed 25 December 2021]

[Dubey, 2018] Dubey A, (2018)Feature Selection Using Random forest. Available at: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f[Accessed 25 December 2021]

[Arora, 2020] Arora S, (2020)Let's Solve Overfitting! Quick Guide to Cost Complexity Pruning of Decision Trees. Available at: https://www.analyticsvidhya.com/blog/2020/10/cost-complexity-pruning-decision-trees/[Accessed 25 December 2021]

[Singh, 2020] Singh R, (2020)A practical approach to Tree Pruning using sklearn | Decision Trees. Available at: https://ranvir.xyz/blog/practical-approach-to-tree-pruning-using-sklearn/[Accessed 25 December 2021]

[Das, 2020] Das A, (2020)Decision Tree Classifier and Cost Computation Pruning using Python. Available at: https://towardsdatascience.com/decision-tree-classifier-and-cost-computation-pruning-using-python-b93a0985ea77[Accessed 25 December 2021]

[Admin, 2020] Admin J, (2020)10 Techniques to deal with Imbalanced Classes in Machine Learning. Available at: https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/[Accessed 25 December 2021]

[Admin, 2020] Brownlee J, (2020)Random Oversampling and Undersampling for Imbalanced Classification. Available at: https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification//[Accessed 25 December 2021]

[Pykes, 2020] Pykes P, (2020)Oversampling and Undersampling. Available at: https://towardsdatascience.com/oversampling-and-undersampling-5e2bbaf56dcf[Accessed 25 December 2021]

[Brownlee, 2020] Brownlee J, (2020)Why Use Ensemble Learning?. Available at: https://machinelearningmastery.com/why-use-ensemble-learning/[Accessed 26 December 2021]

[Brownlee, 2021] Brownlee J, (2021)A Gentle Introduction to Ensemble Learning Algorithms. Available at: https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/[Accessed 26 December 2021]

[Lutins, 2017] Lutins E, (2017)Ensemble Methods in Machine Learning: What are They and Why Use Them?. Available at: https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f[Accessed 26 December 2021]

[Rajbangshi, 2020] Rajbangshi A, (2020)Types of Ensemble methods in Machine learning. Available at: https://towardsdatascience.com/types-of-ensemble-methods-in-machine-learning-4ddaf73879db[Accessed 26 December 2021]

[Kuwar, 2019] Kuwar A, (2019) ML|Voting Classifier using Sklearn. Available at: https://www.geeksforgeeks.org/ml-voting-classifier-using-sklearn/[Accessed 28 December 2021]

[Saini, 2021] Saini A, (2021)AdaBoost Algorithm–A Complete Guide for Beginners. Available at: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/[Accessed 28 December 2021]

[Bhandari, 2020] Bhandari A, (2020)AUC-ROC Curve in Machine Learning Clearly Explained. Available at: https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/[Accessed 28 December 2021]

[Gupta, 2021] Gupta A, (2021)Elbow Method for optimal value of k in KMeans. Available at: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/[Accessed 28 December 2021]

[Kleine, 2021] Kleine D, (2021)Detecting knee- / elbow points in a graph. Available at: https://towardsdatascience.com/detecting-knee-elbow-points-in-a-graph-d13fc517a63c[Accessed 28 December 2021]

[Bonaros, 2019] Bonaros B, (2019)K-Means Elbow Method Code For Python. Available at: https://predictivehacks.com/k-means-elbow-method-code-for-python/[Accessed 28 December 2021]

[MWITI, 2020] MWITI D, (2020)The Beginners Guide to Clustering Algorithms and How to Apply Them in Python. Available at: https://cnvrg.io/clustering-

algorithms/?gclid=Cj0KCQiA2ZCOBhDiARIsAMRfv9JGhF0x89Jcvn4441P0uoIiGrh8z3I93t89J6S7aERAMTDoPJLzCUYaArMDEALw$_{w}cB$[$Accessed 28 December 2021$]

[Clarke, 2021] Clarke M, (2021)How to use knee point detection in k means clustering. Available at: https://practicaldatascience.co.uk/machine-learning/how-to-use-knee-point-detection-in-k-means-clustering[Accessed 28 December 2021]

[Banerji, 2021] Banerji A, (2021)K-Mean: Getting The Optimal Number Of Clusters. Available at: https://www.analyticsvidhya.com/blog/2021/05/k-mean-getting-the-optimal-number-of-clusters/[Accessed 28 December 2021]

[Bhardwaj, 2020] Bhardwaj A, (2020)Silhouette Coefficient. Available at: https://towardsdatascience.com/silhouette-coefficient-validating-clustering-techniques-e976bb81d10c[Accessed 28 December 2021]

[Kumar, 2020] Kumar A, (2020)KMeans Silhouette Score Explained with Python Example. Available at: https://vitalflux.com/kmeans-silhouette-score-explained-with-python-example/[Accessed 28 December 2021]

[Chaudhary, 2020] Chaudhary M, (2020)Silhouette Analysis in K-means Clustering. Available at: https://medium.com/@cmukesh8688/silhouette-analysis-in-k-means-clustering-cefa9a7ad111[Accessed 28 December 2021]

[Sakkaf, 2020] Sakkaf S, (2020)Decision Trees: ID3 Algorithm Explained. Available at: https://towardsdatascience.com/decision-trees-for-classification-id3-algorithm-explained-89df76e72df1[Accessed 29 December 2021]

[Dabbura, 2018] Dabbura I, (2018)K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Available at: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a1[Accessed 30 December 2021]

[Navlani, 2019] Navlani A, (2019)Support Vector Machines with Scikit-learn. Available at: https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python[Accessed 30 December 2021]

[Brownlee, 2018] Brownlee J, (2018)How to Use ROC Curves and Precision-Recall Curves for Classification in Python. Available at: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/[Accessed 30 December 2021]