

## APPENDIX 1



# CLASSIFICATION OF REAL AND FAKE BANK NOTES

## PROJECT REPORT

### CA1

*Submitted by*

**CECILIA SAGWATI**

**12020304**

**INT578: ADVANCED MACHINE LEARNING**

*Submitted to*

**DR. DHANPRATAP SINGH**

**FEBRUARY, 2022**

## APPENDIX 2

### TABLE OF CONTENTS

<b>Appendix .....</b>	<b>0</b>
Cover page .....	1
Table of contents .....	2
Declaration .....	3
Acknowledgement .....	3
List of figures .....	3
Abbreviation .....	4
<b>CHAPTER1 .....</b>	<b>5</b>
2.0 Introduction .....	5
2.1 Abstract and Objective .....	5
<b>CHAPTER2 .....</b>	<b>7</b>
3.1 Pictorial Description of the project .....	7
<b>CHAPTER3 .....</b>	<b>7</b>
3.1 Project Implementation .....	8
<b>CHAPTER4 .....</b>	<b>23</b>
4.0 Conclusion.....	23

## **APPENDIX 3**

### **DECLARATION**

I, Cecilia Sagwati, hereby declare that the work presented herein is genuine work originally done by me and it has not been published or submitted elsewhere for the requirement of a degree programme.

## **APPENDIX 4**

### **ACKNOWLEDGEMENT**

I would like to express my sincere gratitude to Dr. Dhanpratap Singh of Lovely Professional University, for teaching me machine learning language which enabled me to undertake this project. Machine Learning knowledge is very important because in the society that we live in, there are so many complex problems that can be solved easily by machine learning algorithms.

## **APPENDIX 5**

### **LIST OF FIGURES**

Fig1: Data Analysis using seaborn plot .....	12
Fig2. Heatmap .....	13
Fig3. Screenshot of SVM model training, prediction and evaluation .....	15
Fig4. Screenshot of perceptron model training, prediction and evaluation .....	16
Fig5. Screenshot of Logistic Regression model training, prediction and evaluation .....	17
Fig6. Screenshot of Decision Tree model training, prediction and evaluation .....	18
Fig7. Screenshot of K-Nearest neighbors model training, prediction and evaluation .....	19
Fig8. Graphical representation of model evaluation summary during testing .....	21
Fig9. Graphical representation of model evaluation summary during training .....	22

## LIST OF ABBREVIATIONS

SVC	Support Vector Machine Model
KNN	K-Nearest Neighbor Model
IDE	Integrated Development Environment
PPTN	Perceptron model
LR	Logistic Regression Model
DT	Decision Tree Model

# CHAPTER1

## TITLE: CLASSIFICATION OF REAL AND FAKE BANK NOTES

### 1.0 INTRODUCTION

#### 1.1 ABSTRACT AND OBJECTIVE

This is a supervised machine learning project regarding authenticating banknotes. It is a binary classification, and the main objective of the project is to classify real and fake banknotes denoted by 0 and 1 in the dataset. When depositing money in the bank, the bank teller loads the money in a machine to verify whether a banknote is real or fake. This verification task can also be performed by a machine learning algorithm. This project report presents how machine learning models were trained to classify real and fake banknotes.

The dataset which I used to train the models was downloaded from UCI online repository through web link <https://archive.ics.uci.edu/ml/machine-learning-databases/00267/>.

This dataset contains 4 features that were extracted from images of real and fake banknotes. The features were digitalized by an industrial camera, and wavelet transform tool, was used to extract the features from the said banknotes' images. The features of the banknote dataset are:

- Variance of wavelet of transformed image denoted as **variance of wavelet**
- Skewness of wavelet of transformed image denoted as **skewness of wavelet**
- curtosis of wavelet of transformed image denoted as **curtosis of wavelet**
- entropy of transformed image denoted as **entropy of image**

The fifth column in the dataset is "class" which contains class of a banknote denoted as either 0 or 1. The banknote dataset has 1372 instances, 762 for class 0 and 610 for class 1. Thus, its shape is 1372 by 5. For purposes of training and testing, the dataset was split in the ratio of 70 to 30. Thus, the training dataset denoted as data\_train had 960 instances of banknotes whereas the testing dataset denoted as data\_test had 412 instances of banknotes.

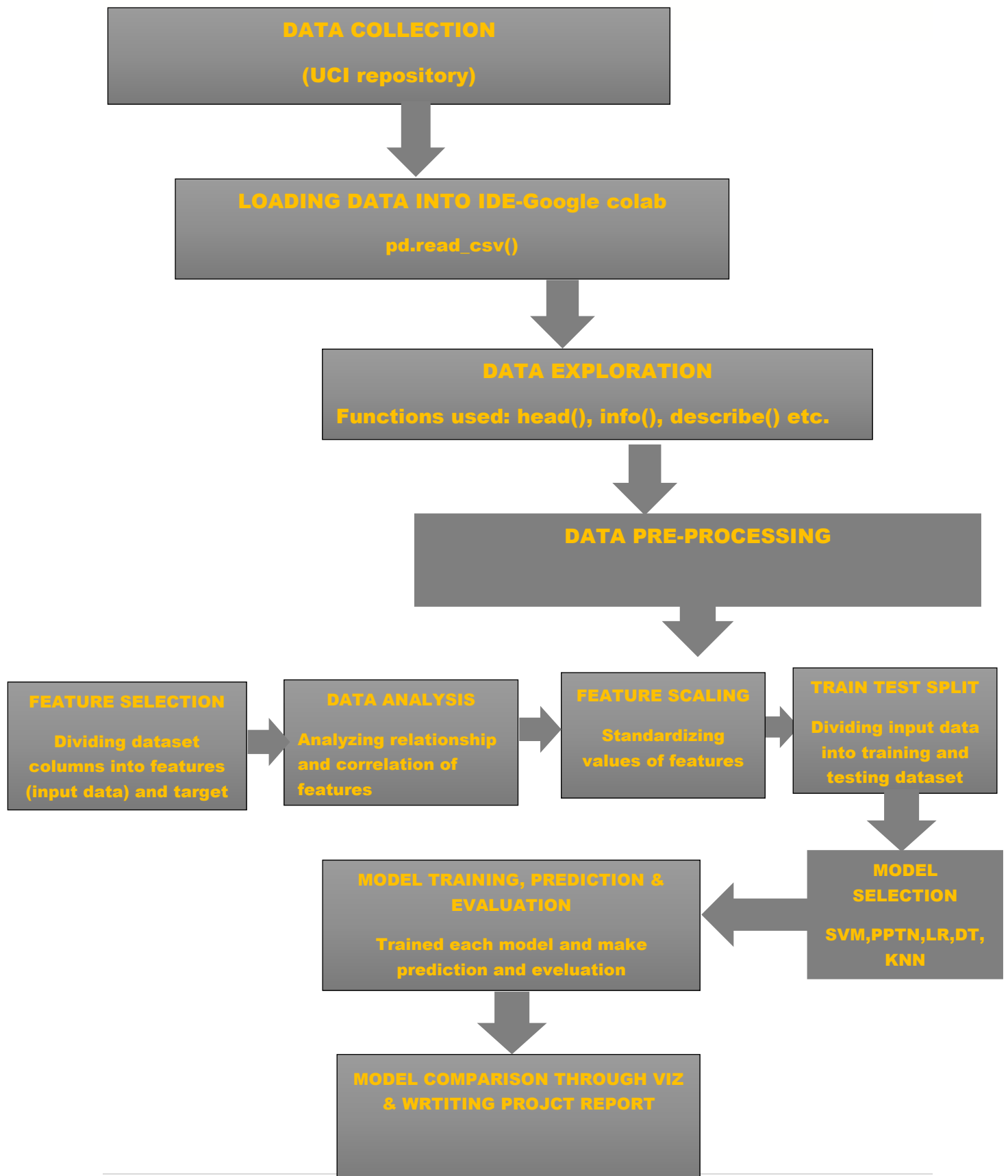
I classified the banknotes using 5 machine learning classification models namely Support Vector Machine, Perceptron, Logistic Regression, Decision Tree and K-Nearest Neighbors. **The machine learning model with 100% accuracy during both training and testing was Support Vector Machine model.** The accuracy for the rest of the models during testing and

training was also good as it was above 96%, which means there were few misclassified banknotes. **Perceptron model** had 8 misclassified banknotes for class 0 and its testing and training accuracy was 98.06% and 99.48% respectively. **Logistic Regression model** had testing accuracy of 98.3% and training accuracy of 98.13% with 7 misclassified banknotes for class 0. **Decision Tree model** had testing accuracy of 97.82% and training accuracy of 100% with 6 misclassified banknotes for class 0 and 3 misclassified banknotes for class 1. Finally, **K-Nearest Neighbors** model had testing accuracy of 99.76% and training accuracy of 99.9% with 1 misclassified banknote for class 0.

Based on the evaluation report of the models, it was concluded that class 0 banknotes of this dataset are more difficult to classify than class 1. Thus, out of the 5 trained models, only **Decision Tree model** had misclassified banknotes (3) in class 1 and it performed poorly in comparison with the other models. The rest of the models classified class 1 bank notes correctly. **Support Vector Machine model** was recommended as the best model for classifying the banknotes in the dataset followed by **K-Nearest Neighbors model** and at last **Decision Tree model**.

# CHAPTER2

## 2.1 DESCRIPTION OF THE PROJECT



# CHAPTER 3

## 3.0 PROJECT IMPLEMENTATION

### 3.1 DATA COLLECTION

The dataset which I used to train the models was downloaded from UCI online repository through web link <https://archive.ics.uci.edu/ml/machine-learning-databases/00267/>.

This dataset contains 4 features that were extracted from images of real and fake banknotes. The features were digitalized by an industrial camera and, wavelet transform tool, was used to extract the features from the said banknotes' images. The features of the banknote dataset are:

- Variance of wavelet of transformed image denoted as **variance of wavelet**
- Skewness of wavelet of transformed image denoted as **skewness of wavelet**
- curtosis of wavelet of transformed image denoted as **curtosis of wavelet**
- entropy of transformed image denoted as **entropy of image**

### 3.2 LOADING DATA INTO IDE (INTEGRATED DEVELOPMENT ENVIRONMENT)

I carried out this machine learning project on google colab which is an online IDE for machine learning using python language. I loaded the dataset on google colab and printed the shape of the data set, and also the first 6 rows of the dataset with the below command lines:

```
[55] ## LOADING DATASET
import pandas as pd
dataset=pd.read_csv('/content/drive/MyDrive/CA1ProjectDatasetCeciliaSagwati112020304.csv')
print('data shape = ',dataset.shape)

data shape = (1372, 5)
```

```
## DATA EXPLORATION: CHECKING THE FIRST 6 RECORDS
dataset.head(6)
```

	Variance of Wavelet	skewness of Wavelet	curtosis of Wavelet	entropy of image	class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
2	3.86600	-2.6383	1.9242	0.10645	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0
5	4.36840	9.6718	-3.9606	-3.16250	0

As shown in the above screenshot, there are 1372 rows and 5 columns and the first 6 rows are also shown in this screenshot.



### 3.3 DATA EXPLORATION

In order to explore the dataset, I used different functions as shown and discussed below:

The **info()** function was used to show information about the dataset in terms of total number of columns and the column names, count of values which are not null and the data type. Hence, this data had 1372 instances of the banknotes and 5 columns, including the class/target column. The data types of the first 4 columns were decimal values and that of last column was integer.

```
✓ 0s dataset.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1372 entries, 0 to 1371
Data columns (total 5 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   Variance of Wavelet    1372 non-null  float64
 1   skewness of Wavelet    1372 non-null  float64
 2   curtosis of Wavelet    1372 non-null  float64
 3   entropy of image       1372 non-null  float64
 4   class                  1372 non-null  int64   
dtypes: float64(4), int64(1)
memory usage: 53.7 KB
```

**describe()** function, provides count of the records and summary statistics for each column of the dataset such as mean, standard deviation, minimum value, the 25<sup>th</sup> (first) quartile, 50<sup>th</sup> (second) quartile and 75<sup>th</sup> (third) quartile as well as maximum value. These statistics are called 5 number summary. On the second cell, I used **isna()** together with **sum()** function. The **isna()** checks if there are **missing values** and the **sum()** add the missing values, as you can see in the screenshot, the dataset had no missing values.

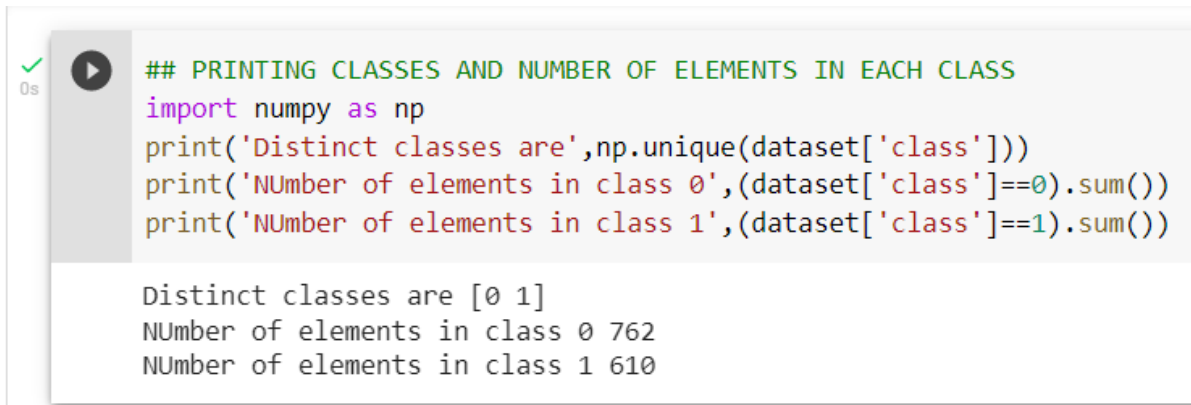
```
✓ 0s ## SUMMARY STATISTICS OF THE DATA
dataset.describe()

   Variance of Wavelet  skewness of Wavelet  curtosis of Wavelet  entropy of image  class
count      1372.000000         1372.000000         1372.000000         1372.000000  1372.000000
mean           0.433735           1.922353           1.397627          -1.191657    0.444606
std            2.842763           5.869047           4.310030           2.101013    0.497103
min           -7.042100          -13.773100          -5.286100          -8.548200    0.000000
25%           -1.773000          -1.708200          -1.574975          -2.413450    0.000000
50%            0.496180           2.319650           0.616630          -0.586650    0.000000
75%            2.821475           6.814625           3.179250           0.394810    1.000000
max            6.824800          12.951600          17.927400           2.449500    1.000000

✓ 0s [5] ## DATA PRE_PROCESSING: CHECKING MISSING VALUES
dataset.isna().sum()

Variance of Wavelet    0
skewness of Wavelet    0
curtosis of Wavelet    0
entropy of image       0
class                  0
```

I used print() and unique() function to print distinct classes of the class column. As u can see, there were two classes, class 1 and class 0. I also printed number of elements in each class using print() and sum() function. As u can see, class 0 had 762 elements while class 1 had 610 elements.



```
## PRINTING CLASSES AND NUMBER OF ELEMENTS IN EACH CLASS
import numpy as np
print('Distinct classes are',np.unique(dataset['class']))
print('NUmber of elements in class 0',(dataset['class']==0).sum())
print('NUmber of elements in class 1',(dataset['class']==1).sum())
```

Distinct classes are [0 1]  
NUmber of elements in class 0 762  
NUmber of elements in class 1 610

## 4.0 DATA PRE-PROCESSING

I pre-processed the data and converted it into the right format for training and testing a machine learning algorithm. To achieve this, I used different functions as shown below:

### 4.1 FEATURE SELECTION

Machine learning algorithm requires data with a demarcation of independent variables which are also called independent features and dependent variable which is the label to be predicted and it is also called target or class label. Thus, class labels are predicted using independent features. Hence, I separated the data using index location method, and I stored the independent features in a variable called 'data' and the dependent feature /target in a variable called 'target'. I then printed the 5 records of each variable using head() function.

```
[6] ## FEATURE SELECTION: INPUT DATA
data=dataset.iloc[:,0:4]
data.head(5)
```

	Variance of Wavelet	skewness of Wavelet	curtosis of Wavelet	entropy of image
0	3.62160	8.6661	-2.8073	-0.44699
1	4.54590	8.1674	-2.4586	-1.46210
2	3.86600	-2.6383	1.9242	0.10645
3	3.45660	9.5228	-4.0112	-3.59440
4	0.32924	-4.4552	4.5718	-0.98880

```
## FEATURE SELECTION :TARGET DATA
target=dataset.iloc[:,4]
target.head(5)
```

```
0    0
1    0
2    0
3    0
4    0
Name: class, dtype: int64
```

I printed the shape of the input data and the target data as show below:

```
## SHAPE OF INPUT AND TARGET DATA
import numpy as np
print('Input data shape ',data.shape)
print('Target data shape ',target.shape)
```

```
Input data shape = (1372, 4)
Target data shape = (1372,)
```

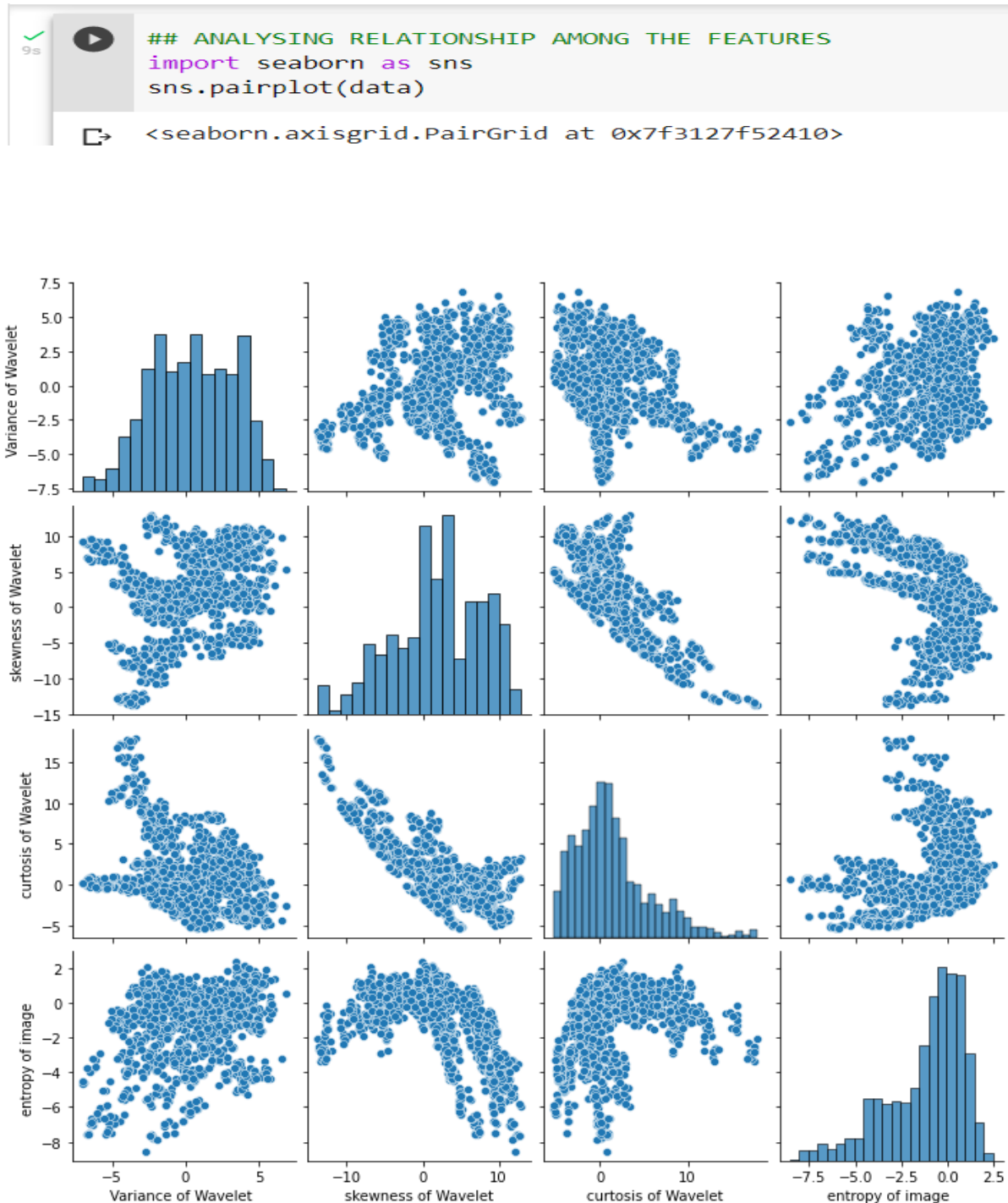
## 4.2 DATA ANALYSIS

### pairplot() in seaborn(sns) library

I imported seaborn library in order to use pairplot() function which plot pairwise relationships of the columns in a dataset. As you can see in fig1 below, the pairplot function produced a grid in which the features were plotted against each other such that each feature had a shared y-axis across a single row and a shared x-axis across a single column.

After analysing the above plot, it was discovered that the skewness of wavelet and curtosis of wavelet has an inverse linear relationship (negative correlation), meaning when value of one variable decreases, the value of the other variable increases and vice versa. Whereas entropy of image and variance of wavelet have a non-inverse linear relationship (positive correlation), meaning when value of one variable increases, the value of the other variable also increases and vice versa.

**Fig1: Data Analysis using seaborn plot**



### **figure(), corr(), heatmap () functions**

For correlation analysis, the value '+1' means highly positive correlation while '-1' means highly negative correlation and '0' means no correlation. I also used **corr()** function to find correlation between the features of the input data, **figure()** function to change the size of the plot and **heatmap()** function to produce the plot shown using the output of the **corr()** function.

Since I already imported the seaborn library in the previous plot, I just Imported matplotlib library. I executed the codes listed below and the output was as shown below:

**Fig2: Heatmap**

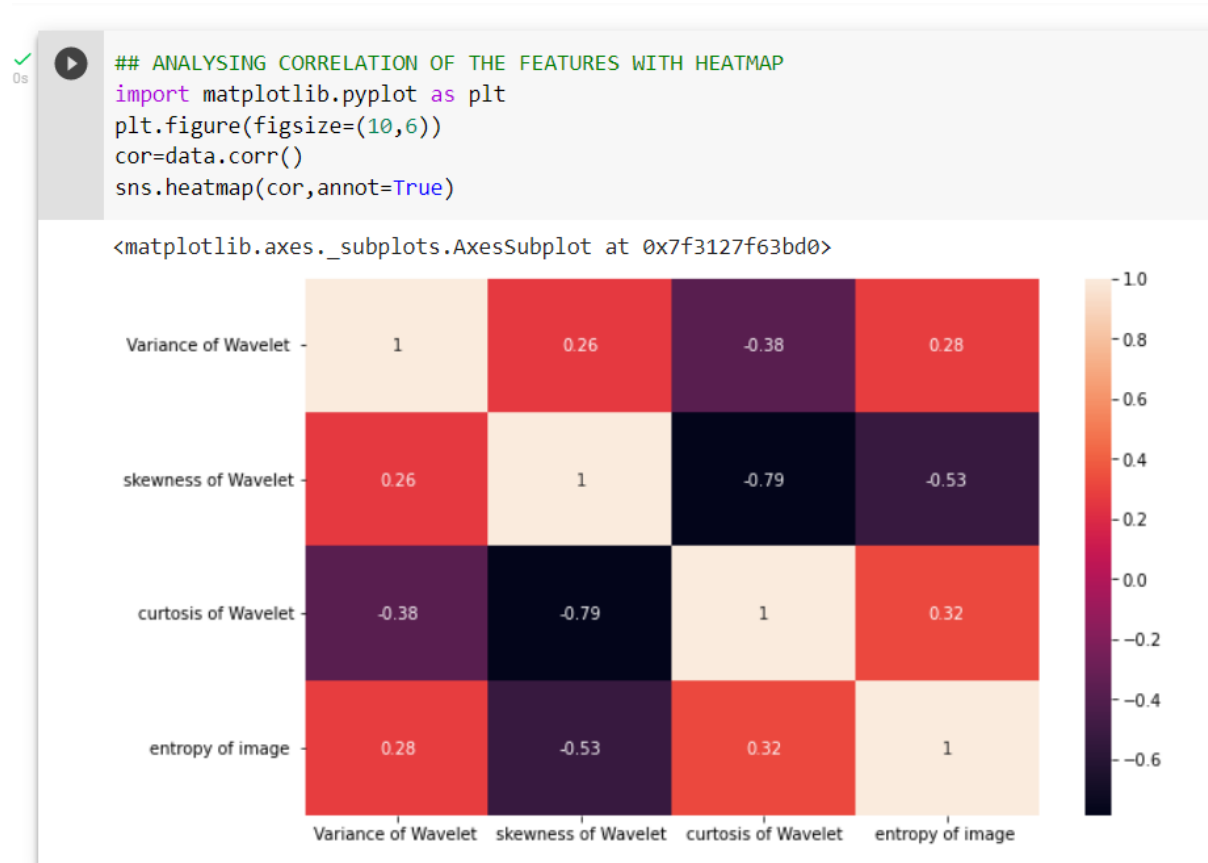


Fig2 shows that skewness of wavelet and curtosis of wavelet have a high negative correlation (-0.79) and skewness of wavelet and entropy of image also have a negative correlation of -0.53. Hence, it can be concluded that variance of wavelet is an important feature in determining class of the banknote because it has low correlation with the other features.

### 4.3 FEATURE SCALING

Feature scaling helps to standardize the data so that there is low variation between the values of the features. I standardized the input data with StandardScaler() function, fit() and transform(). The fit() function estimate the mean and standard deviation of the data, while the transform() function replaces the data values with the values estimated by the fit().

## 4.4 DIVIDING DATA INTO TRAINING AND TESTING DATASET

I divided input data and the target data into training and testing datasets using train-test-split() function from sklearn. I reserved 30% of the input and target data for testing while the remaining 70% was used for training. The training dataset is used to train the machine learning model while the testing dataset is used to evaluate the performance of the trained model. Reserving some data for testing is very important as it enables evaluation of performance of the model on unseen data. I also printed the shape of training and testing data.

### FEATURE SCALING AND TRAIN-TEST-SPLIT COMMANDS

```
[41] ## FEATURE SCALING
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
sc.fit(data)
data_std=sc.transform(data)

## SPLITTING INPUT DATA AND TARGET DATA INTO TRAINING AND TESTING SET
from sklearn.model_selection import train_test_split
data_train,data_test,target_train,target_test=train_test_split(data_std,target,test_size=0.3,random_state=0)
print('Training datashape =',data_train.shape,'and Testing dataset =',data_test.shape)

Training datashape = (960, 4) and Testing dataset = (412, 4)
```

## 5.1 MODEL SELECTION

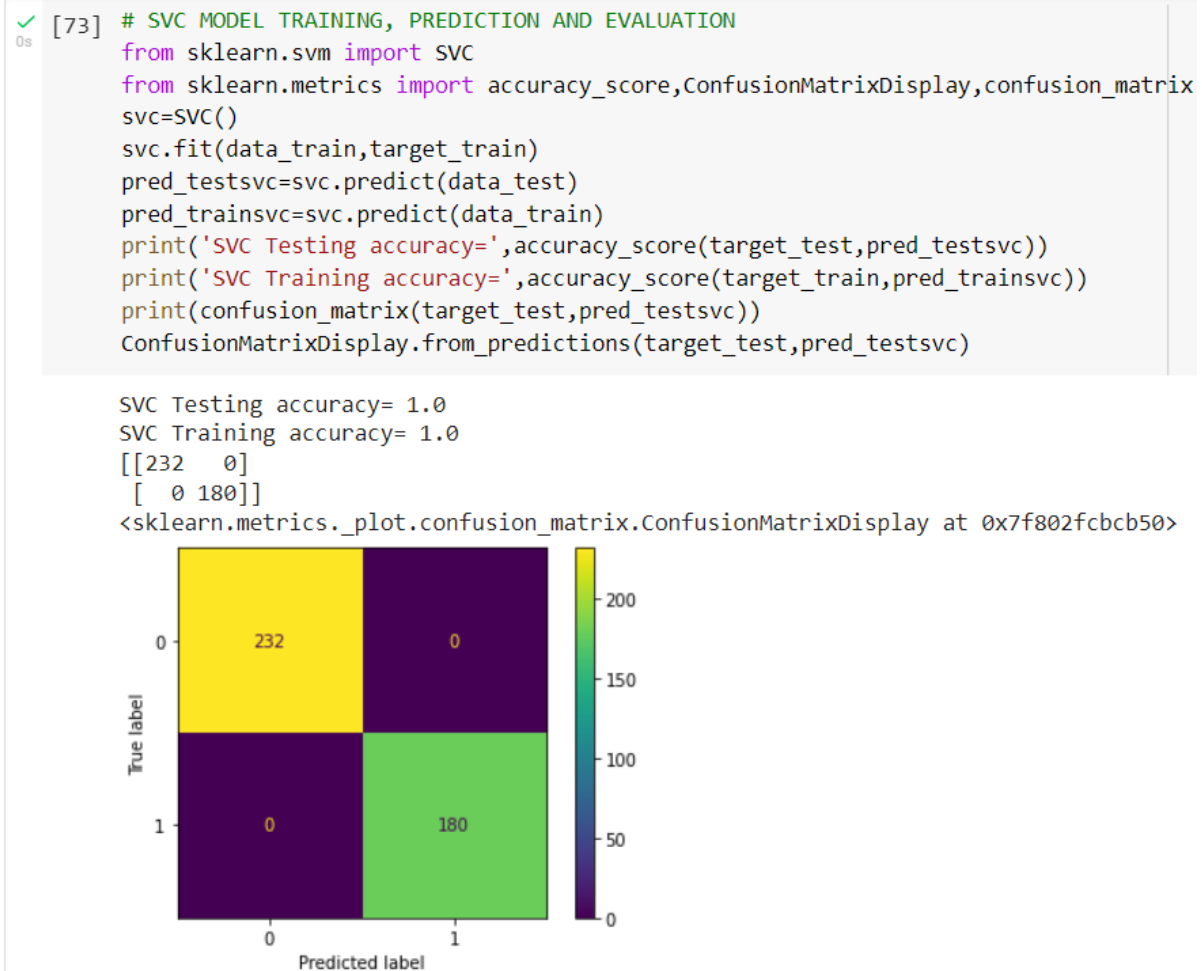
I selected 5 machine learning classifiers namely Support Vector Machine (SVM), Perceptron, Logistic Regression, Decision Tree and K-Nearest Neighbors (KNN). I wanted to train these models and compare their performance, in order to choose the best model among the 5 models.

## 6.1 MODEL TRAINING, PREDICTION AND EVALUATION

I trained each model separately using respective model function and fit() function. After training the model I evaluated performance of the model on the testing dataset by predicting class labels of the testing dataset using predict(). I also measured performance of the model during both testing and training using accuracy\_score() function. I also used confusion\_matrix() and ConfusionMatrixDisplay() metric functions to show and display the output of the predicted labels against the true labels. The commands for training, prediction and evaluation of each model is as shown in the section below:

## SUPPORT VECTOR MACHINE (SVM) MODEL TRAINING, PREDICTION AND EVALUATION

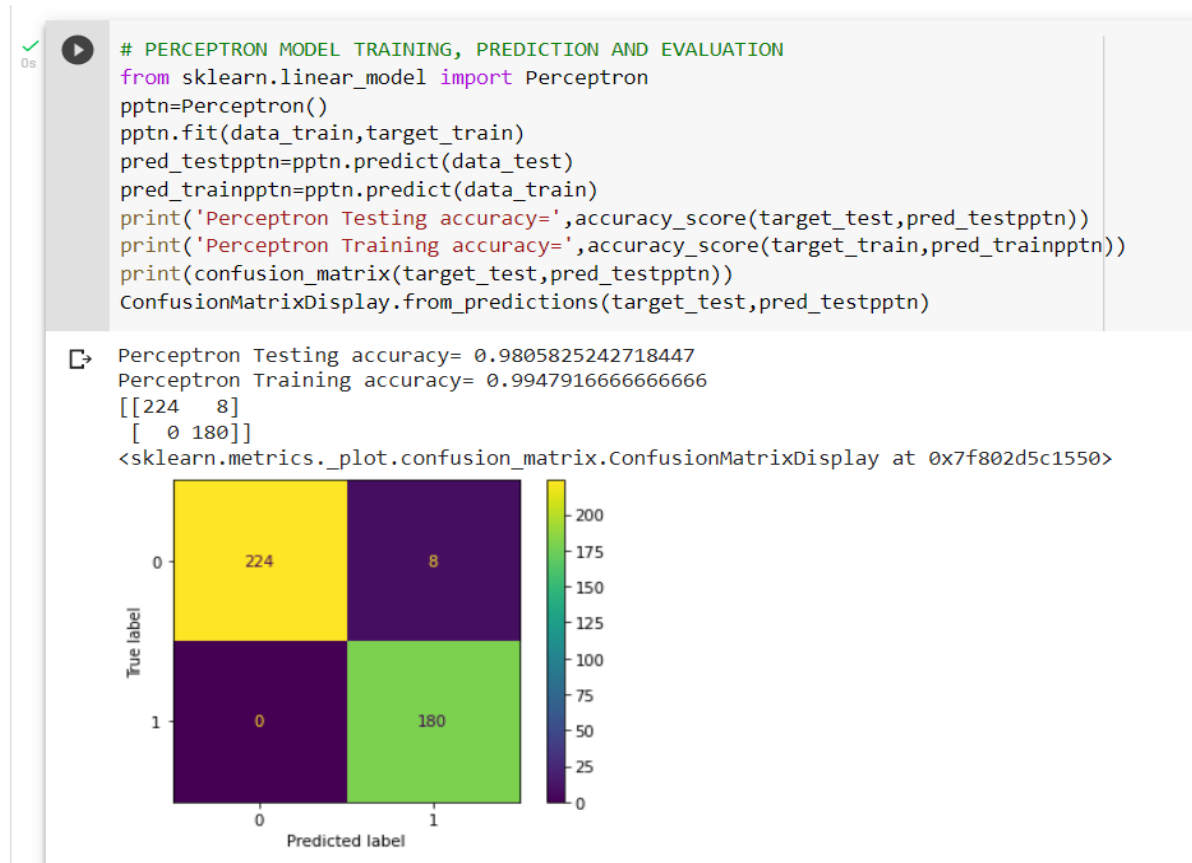
**Fig3: Screenshot of SVM model training, prediction and evaluation**



As shown in fig3 accuracy of Support Vector Machine model was 100% during both testing and training. Hence, there were no misclassified banknotes.

## PERCEPTRON MODEL TRAINING, PREDICTION AND EVALUATION

**Fig4: Screenshot of perceptron model training, prediction and evaluation**

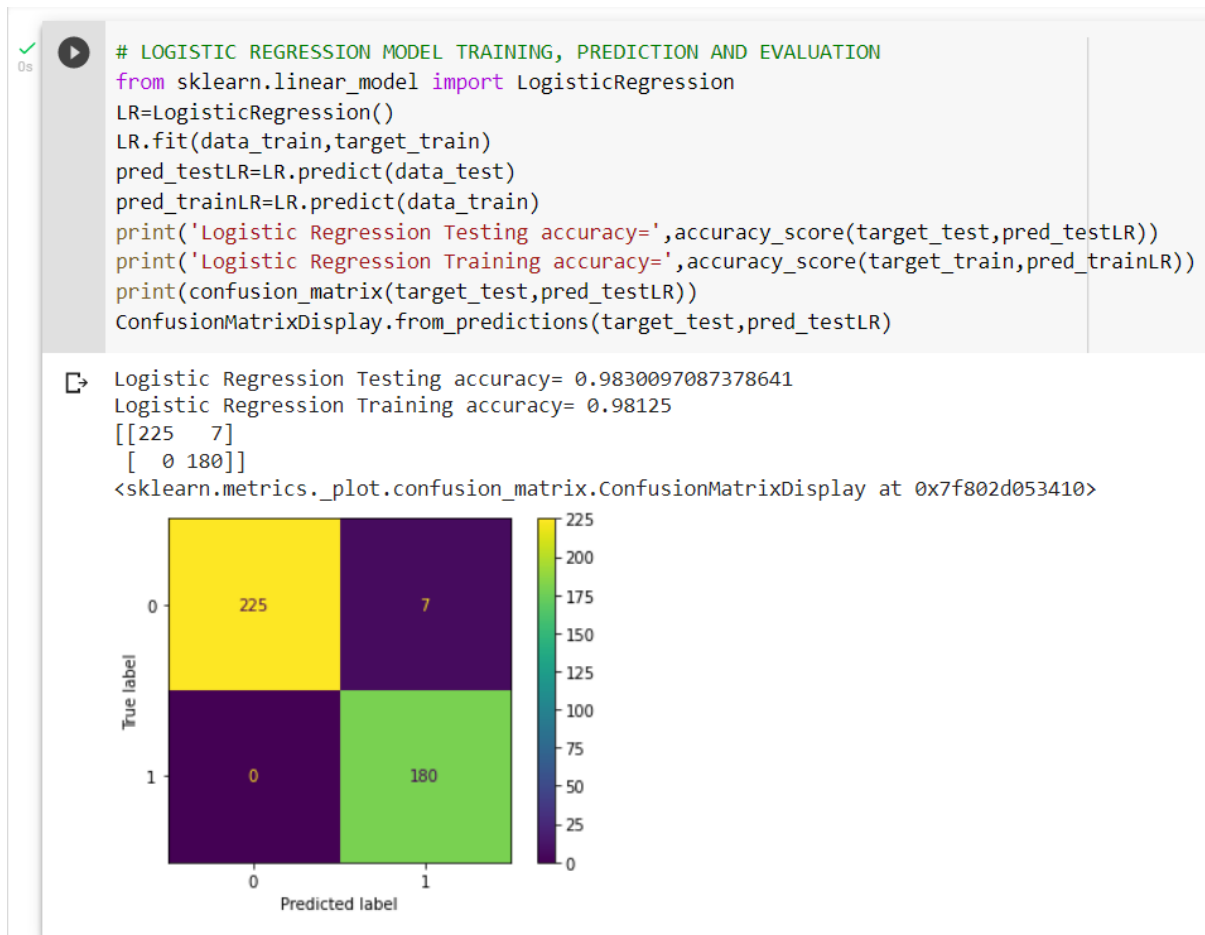


As shown in figure 4, testing accuracy of perceptron model was 98.06% and training accuracy was 99.48%. There were 8 misclassified labels of class 0 during testing. Hence, class 0 labels were short of 8 banknotes which were classified as class 1.



## LOGISTIC REGRESSION MODEL TRAINING, PREDICTION AND EVALUATION

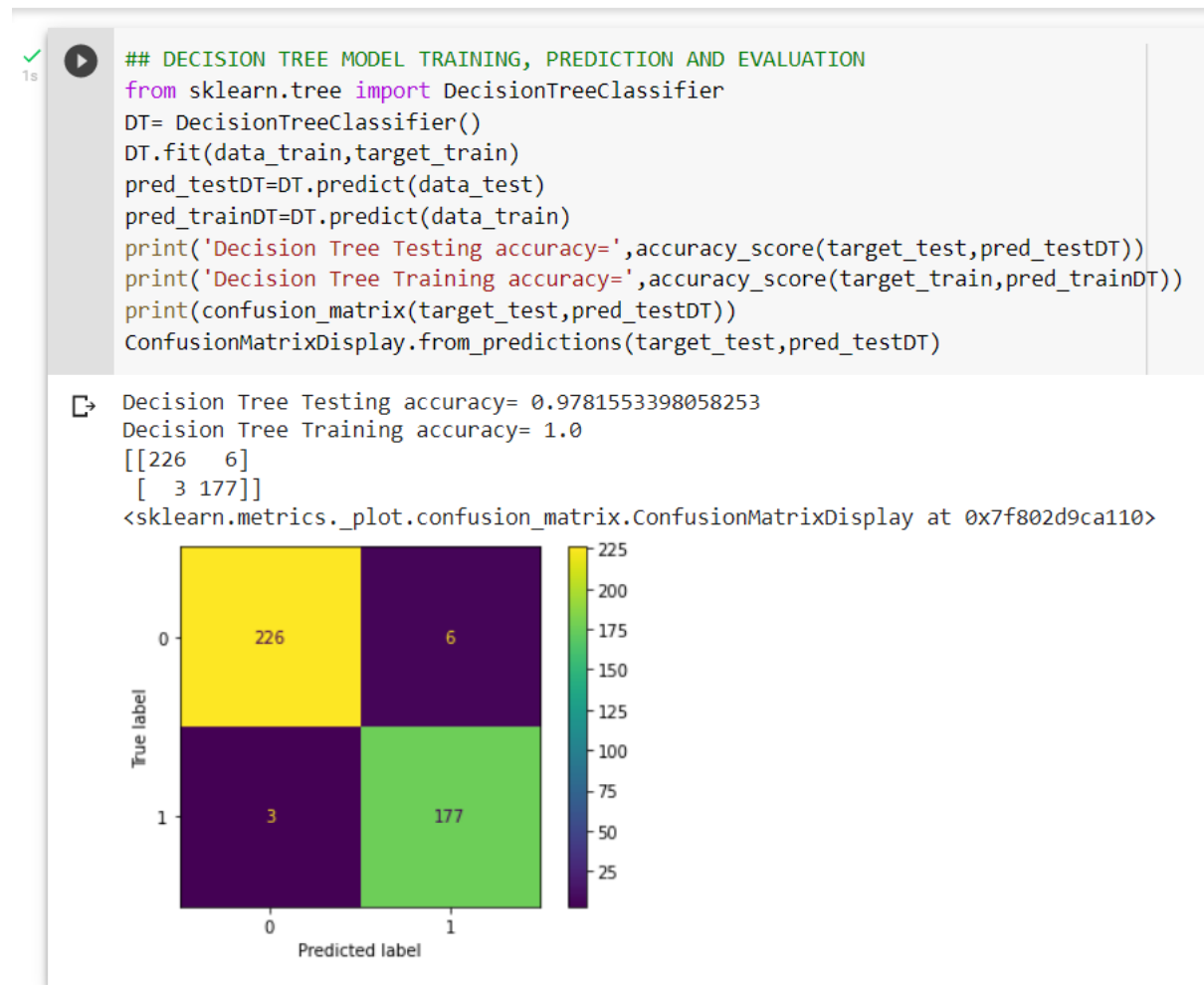
**Fig5: Screenshot of Logistic Regression model training, prediction and evaluation**



As shown in figure 5, accuracy of Logistic Regression during testing was 98.30% and 98.13% during training. There were 7 misclassified labels of class 0. Thus, 7 labels of class 0 were classified as class 1 resulting in a short fall of 7 labels in class 0.

## DECISION TREE MODEL TRAINING, PREDICTION AND EVALUATION

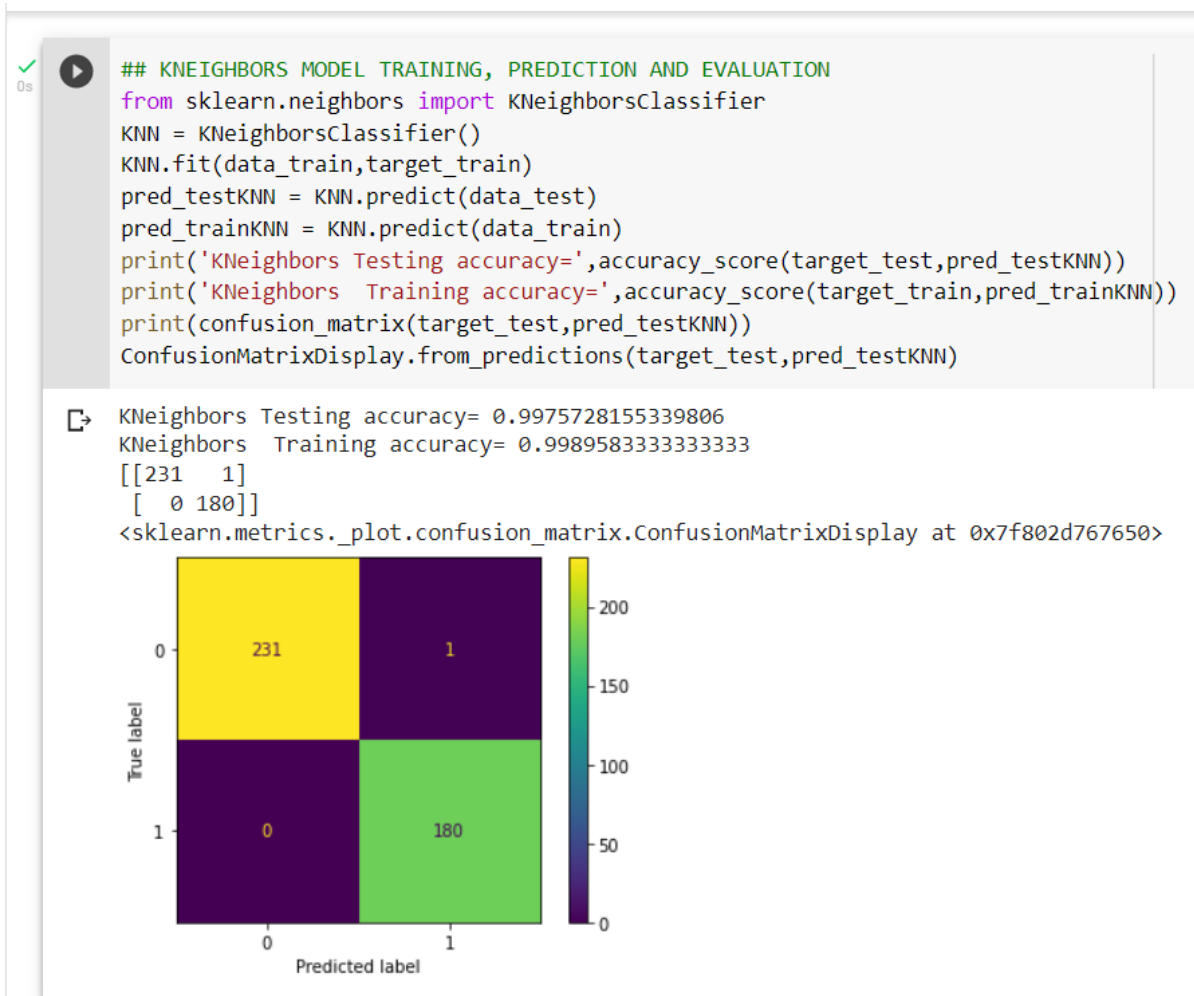
**Fig6: Screenshot of Decision Tree model training, prediction and evaluation**



As shown in fig6 above, performance of Decision Tree model during testing was 97.82% while its performance during training was 100%. There were 6 misclassified labels of class 0 and 1 misclassified label of class 1. This is the only model with a misclassified label in class 1.

## K-NEIGHBORS MODEL TRAINING, PREDICTION AND EVALUATION

**Fig7: Screenshot of K-Nearest neighbors model training, prediction and evaluation**



As shown in fig7, performance of K-Nearest Neighbors model was 99.76% during testing and 99.9% during training. There was only 1 misclassified label in class 0. Performance of this model is very close to performance of Support Vector Machine Model.

## 7.1 MODEL EVALUATION SUMMARY DURING TESTING

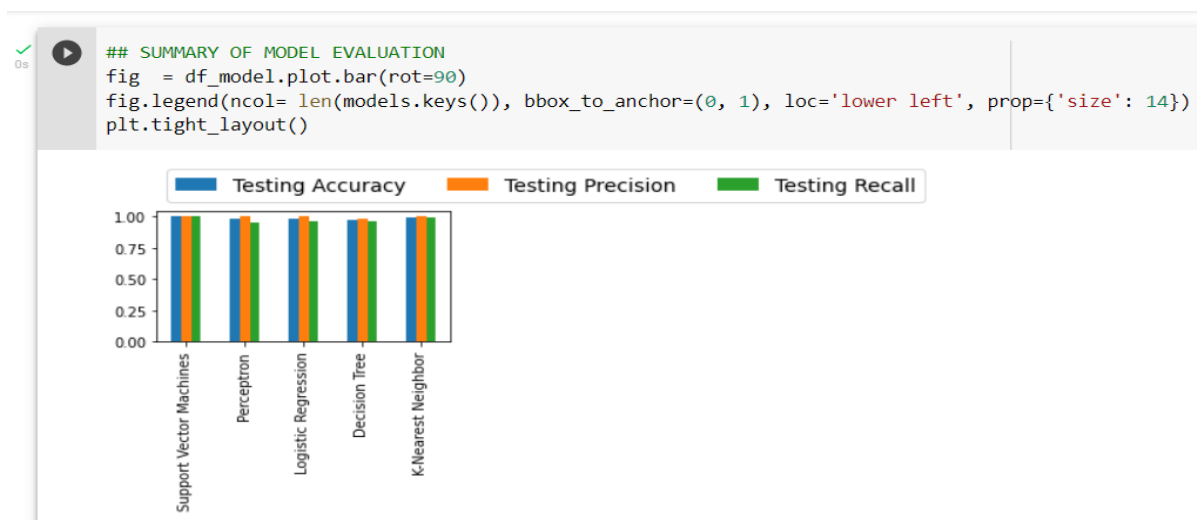
This is the summary of evaluation of all the 5 models during testing. Support Vector Machine model classified the class labels correctly and had 100% accuracy, precision and recall. The second-best model in terms of performance, is K-Nearest Neighbor with an accuracy of 99.76%, precision of 100% and a recall of 99.45%. Generally, all the models performed very well as their performance was above 96% in respect of all the metrics used to evaluate them. Decision tree model was the one with lowest performance.

```
0s  ## MODEL EVALUATION SUMMARY: ACCURACY,PRECISION AND RECALL
from sklearn.metrics import accuracy_score, precision_score, recall_score
models={}
models['Support Vector Machines'] = SVC()
models['Perceptron'] = Perceptron()
models['Logistic Regression'] = LogisticRegression()
models['Decision Tree'] = DecisionTreeClassifier()
models['K-Nearest Neighbor'] =KNeighborsClassifier()
from sklearn.metrics import accuracy_score, precision_score, recall_score
accuracy, precision, recall = {}, {}, {}
for key in models.keys():
    # Fit the classifier model
    models[key].fit(data_train, target_train)
    # Testing Prediction
    predictions = models[key].predict(data_test)
    # Calculate Accuracy, Precision and Recall Metrics
    accuracy[key] = accuracy_score(predictions, target_test)
    precision[key] = precision_score(predictions, target_test)
    recall[key] = recall_score(predictions, target_test)
```

```
0s [79] ## CREATING A DATAFRAME OF MODEL EVALUATION SUMMARY
df_model = pd.DataFrame(index=models.keys(), columns=['Testing Accuracy', 'Testing Precision', 'Testing Recall'])
df_model['Testing Accuracy'] = accuracy.values()
df_model['Testing Precision'] = precision.values()
df_model['Testing Recall'] = recall.values()
df_model
```

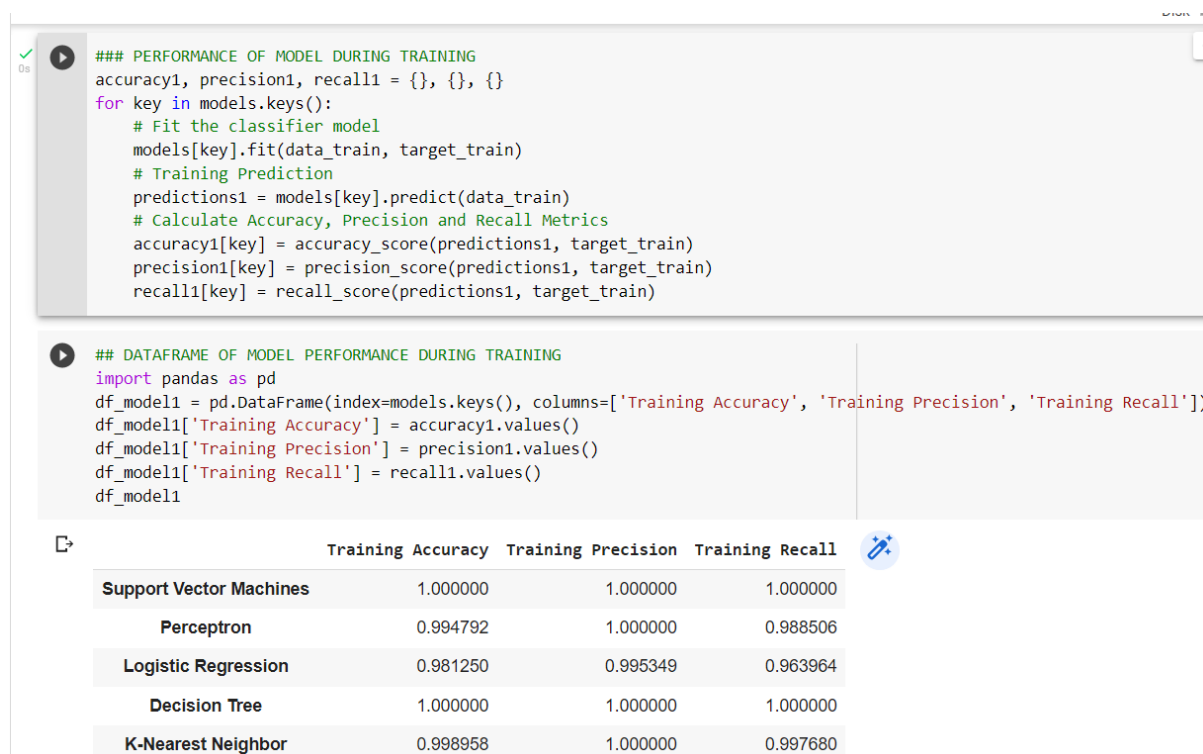
	Testing Accuracy	Testing Precision	Testing Recall
Support Vector Machines	1.000000	1.000000	1.000000
Perceptron	0.980583	1.000000	0.957447
Logistic Regression	0.983010	1.000000	0.962567
Decision Tree	0.978155	0.983333	0.967213
K-Nearest Neighbor	0.997573	1.000000	0.994475

**Fig8: Graphical representation of model evaluation summary during testing**

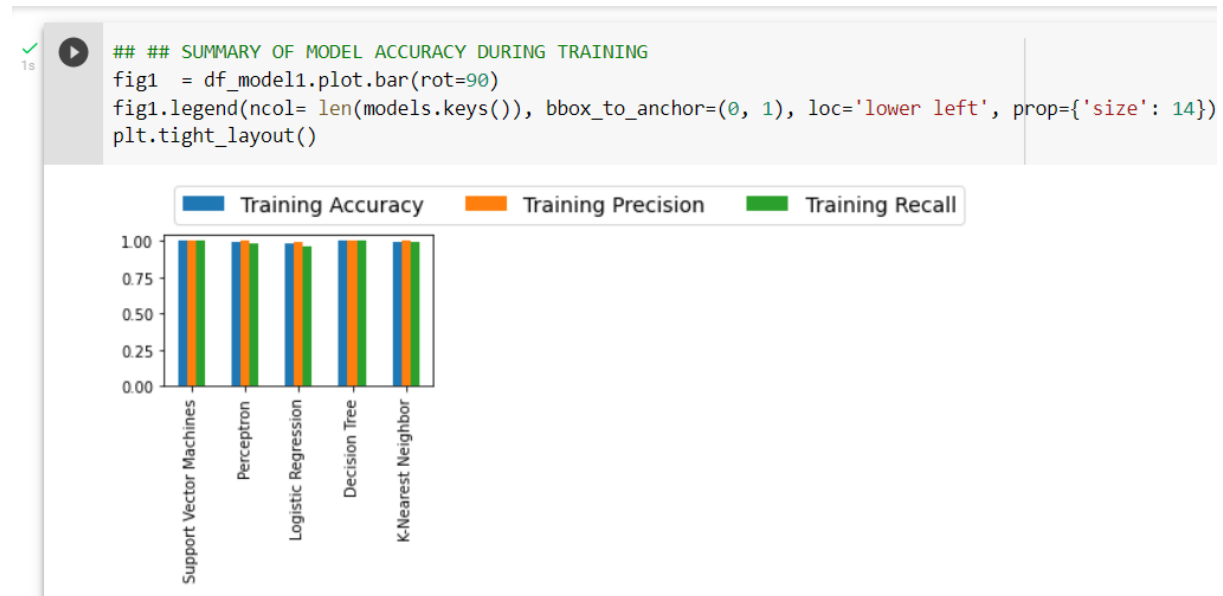


## 8.1 MODEL EVALUATION SUMMARY DURING TRAINING

This is the summary of evaluation of all the 5 models during training. Support Vector Machine model classified the class labels correctly and had 100% accuracy, precision and recall. The second-best model in terms of performance, is K-Nearest Neighbor with an accuracy of 99.9%, precision of 100% and a recall of 99.77%. Generally, all the models performed very well as their performance was above 97% in respect of all the metrics used to evaluate them. Logistic Regression model was the one with lowest performance during training.



**Fig9: Graphical representation of model evaluation summary during training**



# CHAPTER4

## CONCLUSION

All the models generally performed very well during both testing and training. Support Vector machine (SVM) model was on position 1 in terms of performance, while Decision tree model was at last. Hence, I recommend SVM as the best classifier for classifying the banknotes in the dataset seconded by KNN. There were many misclassified labels in class 0, therefore, it can be concluded that class 0 labels are more difficult to classify than class 1 labels. The google colab link for this project is shown below:

### Google colab link

<https://colab.research.google.com/drive/1rNCwskwB0XHXcCnpacgMOJOP1-YBcvYT?usp=sharing>