**Detailed Case Study on Fake Notes Classification**

**DATASET: BANK NOTE AUTHENTICATION**

**ALGORITHM: 1) SUPPORT VECTOR MACHINE**

**2) LOGISTIC REGRESSION**

**By: Yogesh Agrawal**

**INTRODUCTION OF TOPIC (DATA SET PROBLEM)**

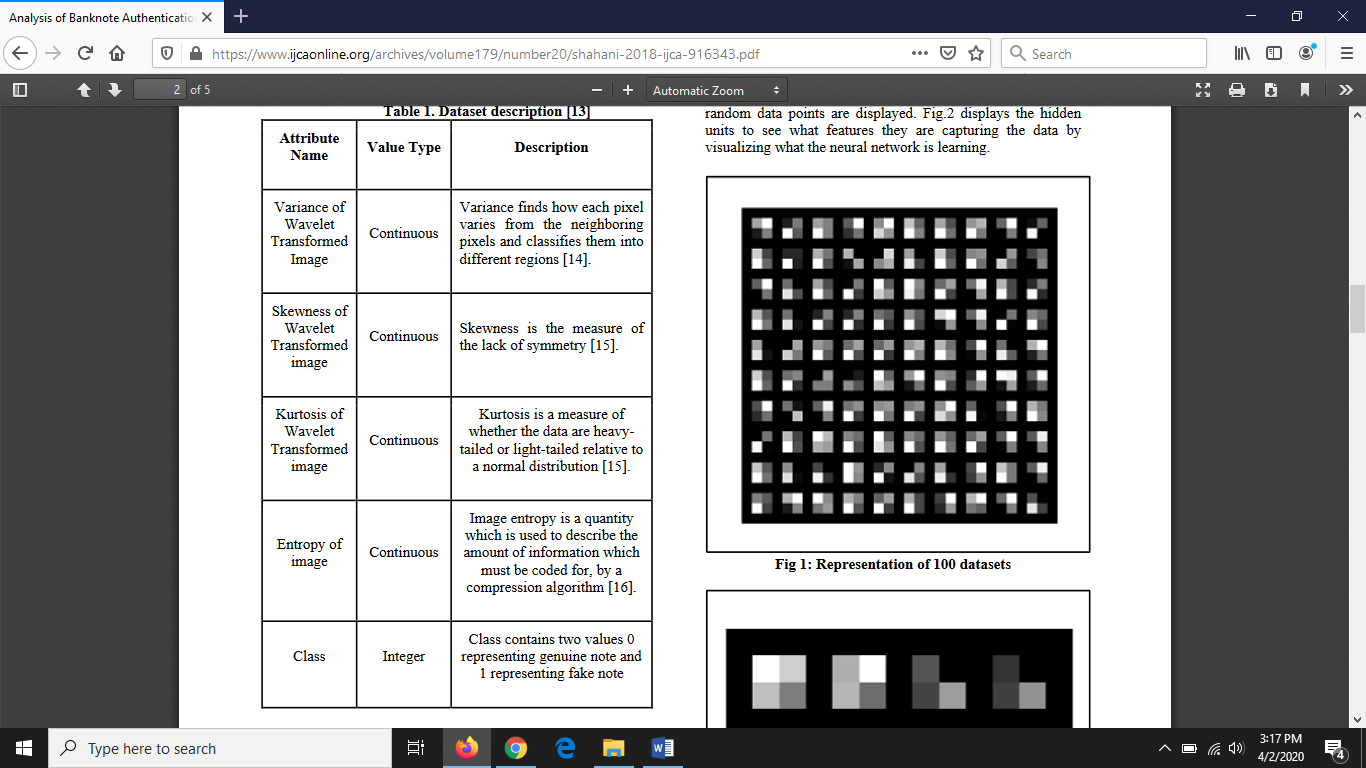
**BANK NOTE AUTHENTICATION**

Banknotes are one of the most important assets of a country and are used to carry out financial activities. Some people introduce fake notes which bear a resemblance to original notes to create discrepancies of the money in the financial market and they are created illegally for various motives. There has been a drastic increase in the rate of fake notes in the market and since the fake notes are created in all denominations, the financial market of the country is brought to a low level.

To continue with smooth cash transactions, entry of forged banknotes in circulation should be conserved. It is difficult for humans to identify between true and fake banknotes because they have a lot of similar features and are created with great accuracy to look like a genuine note. Fake notes are created with precision, hence there is need for an efficient algorithm which accurately predicts whether a banknote is genuine or not.

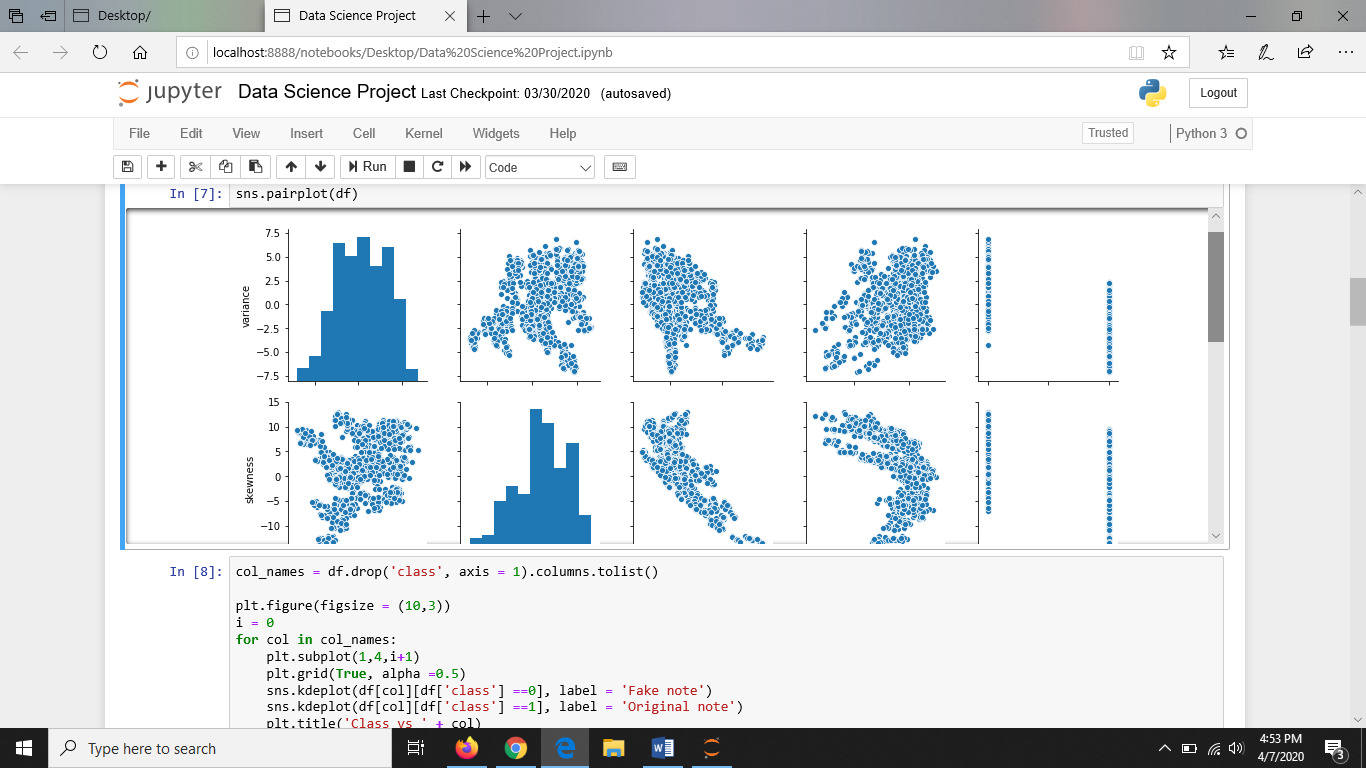
Hence, there is a need for banks and ATM machines to implement a system that classifies a note as genuine or fake. A recognition system must be installed to detect legitimacy of the note.

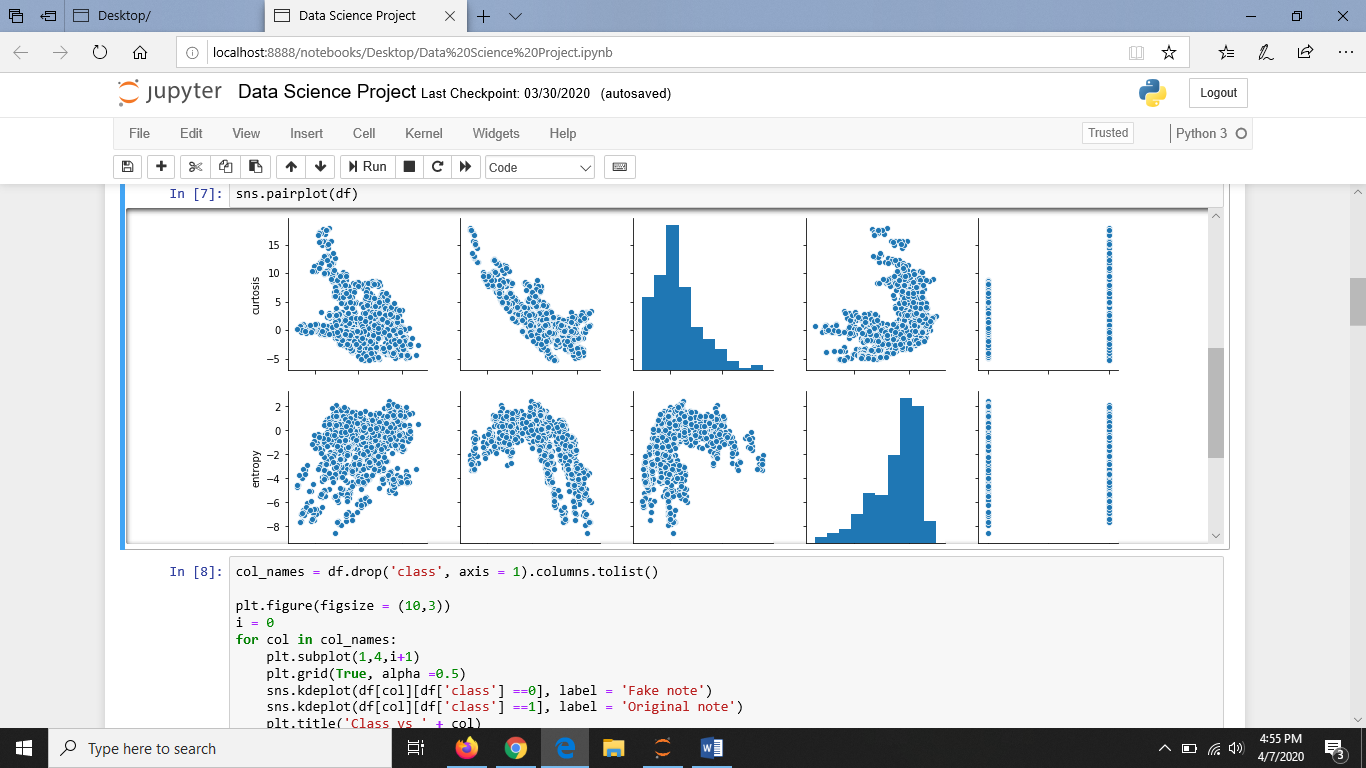
The data set provided to us has been extracted from genuine and counterfeit banknote images.

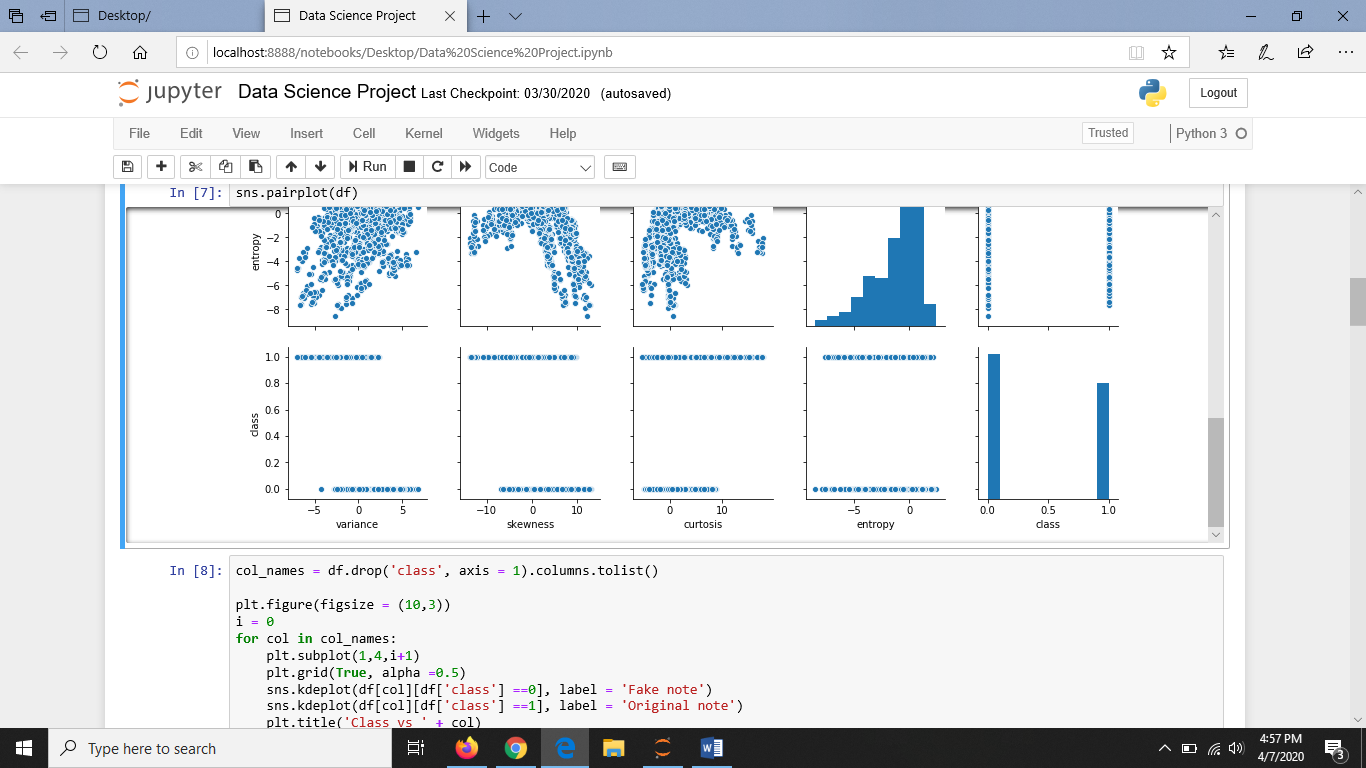


**DATA VISUALISATION**

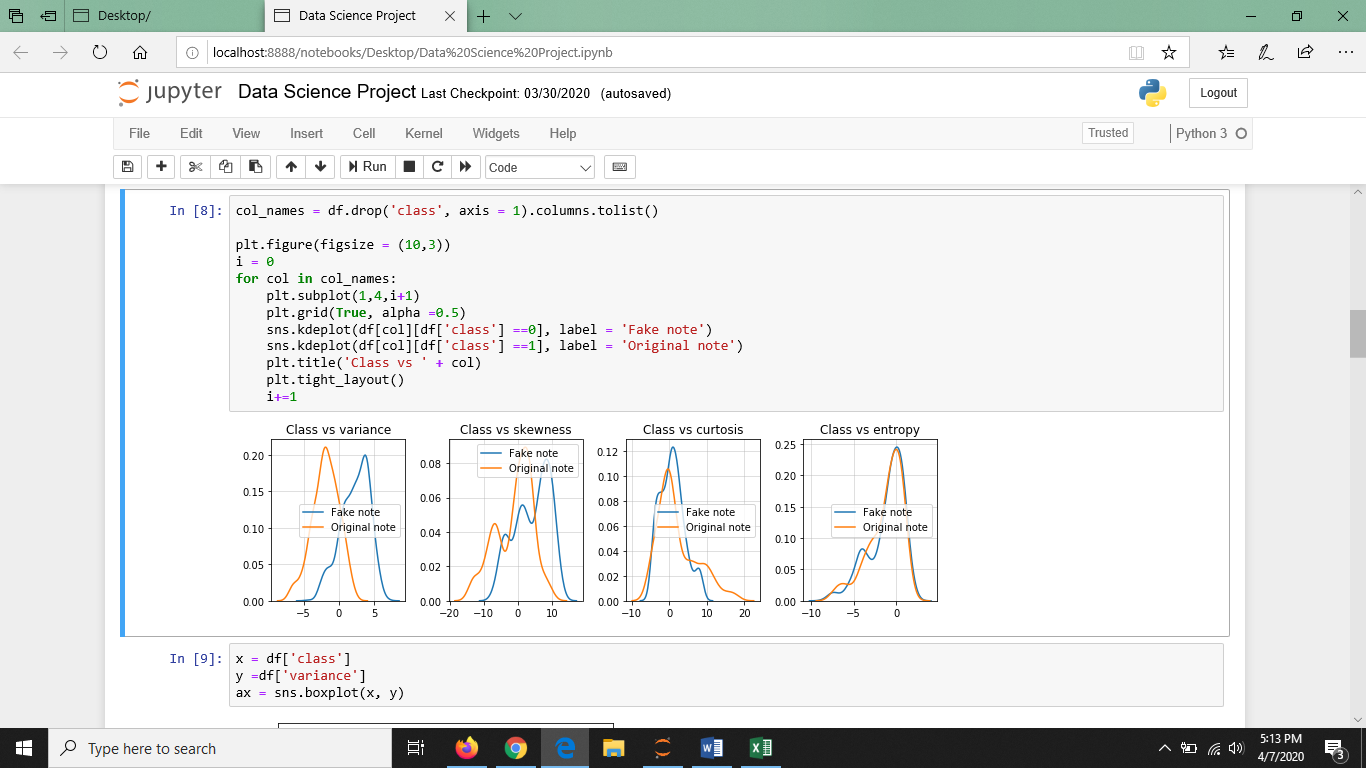
1. **Pair plot**: This is a plot between the different fields of data set with each other. All the same fields have a graph which is a histogram as the values are same and overlapping whereas the graph between two different values is a scatter plot since the values in both the fields can be varied greatly and this shows the range over which the values are scattered.



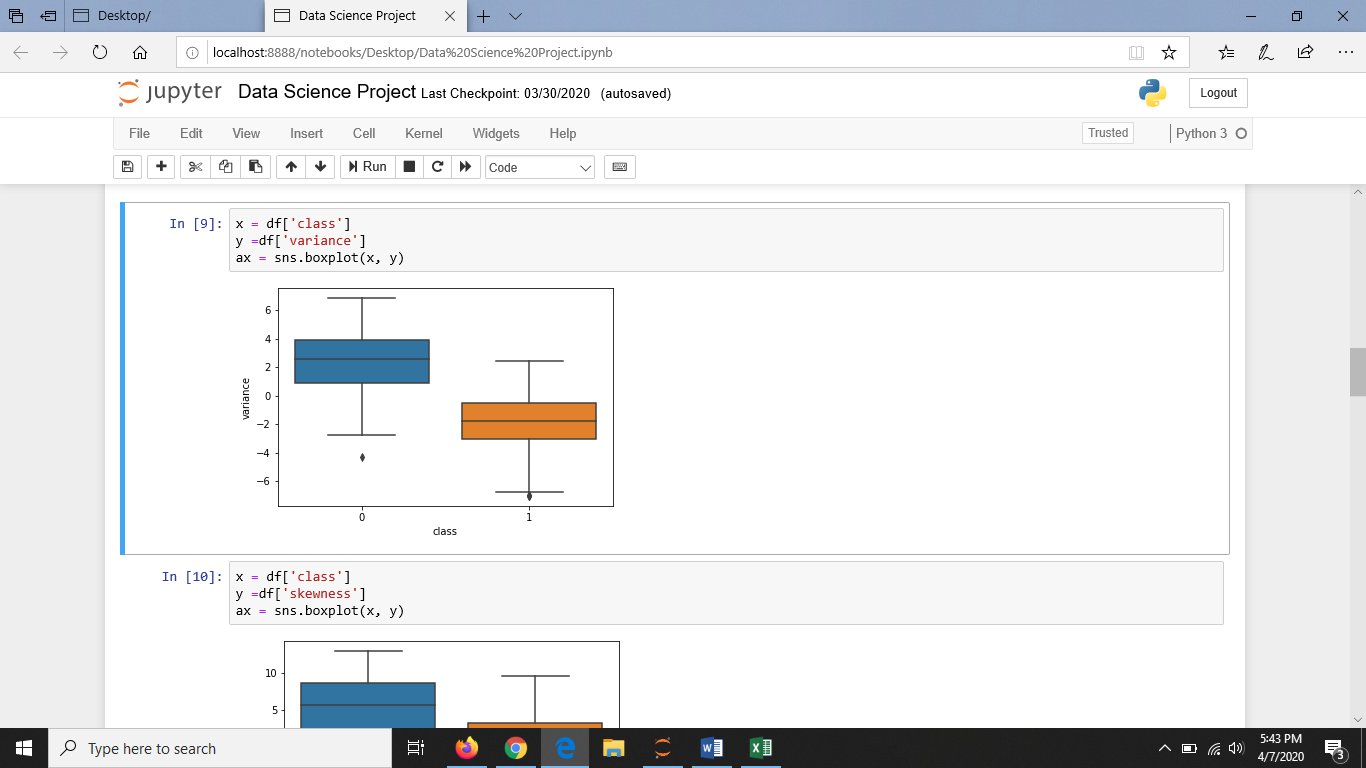


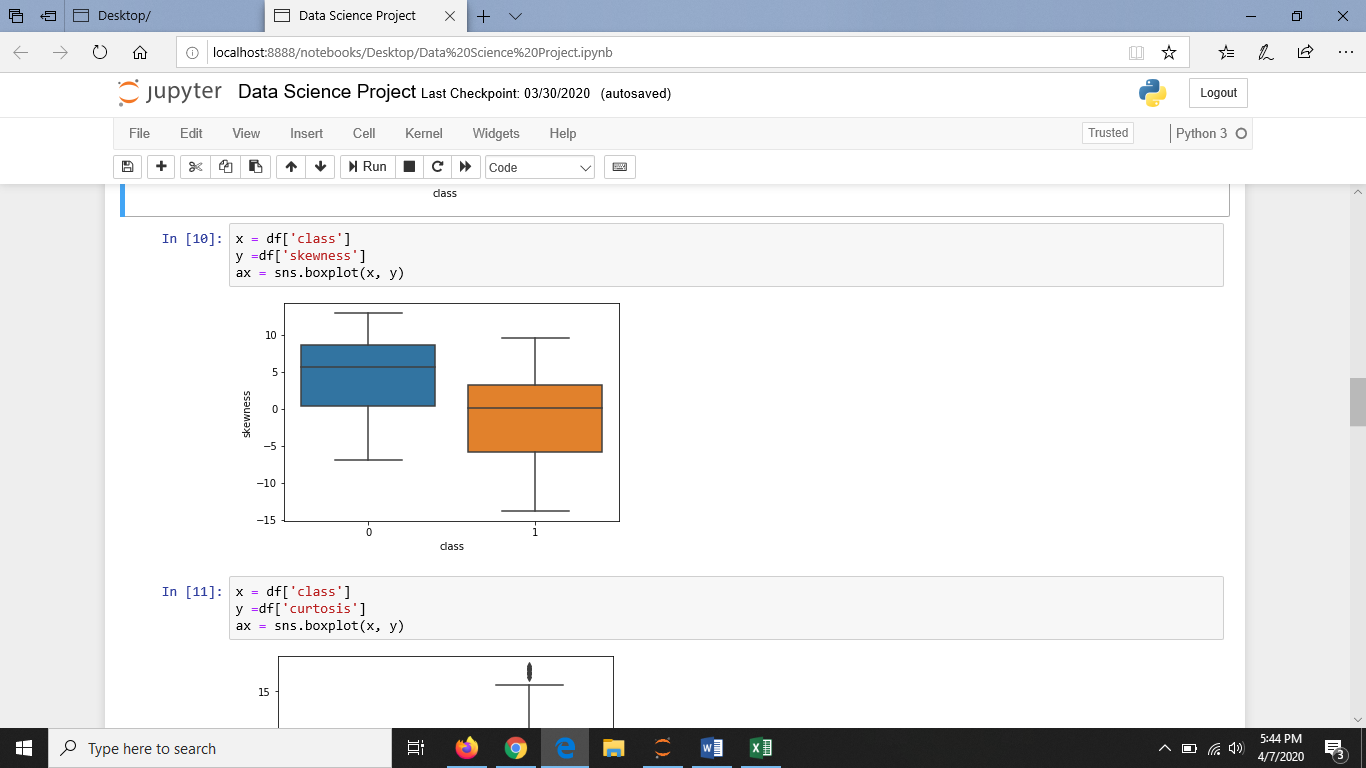


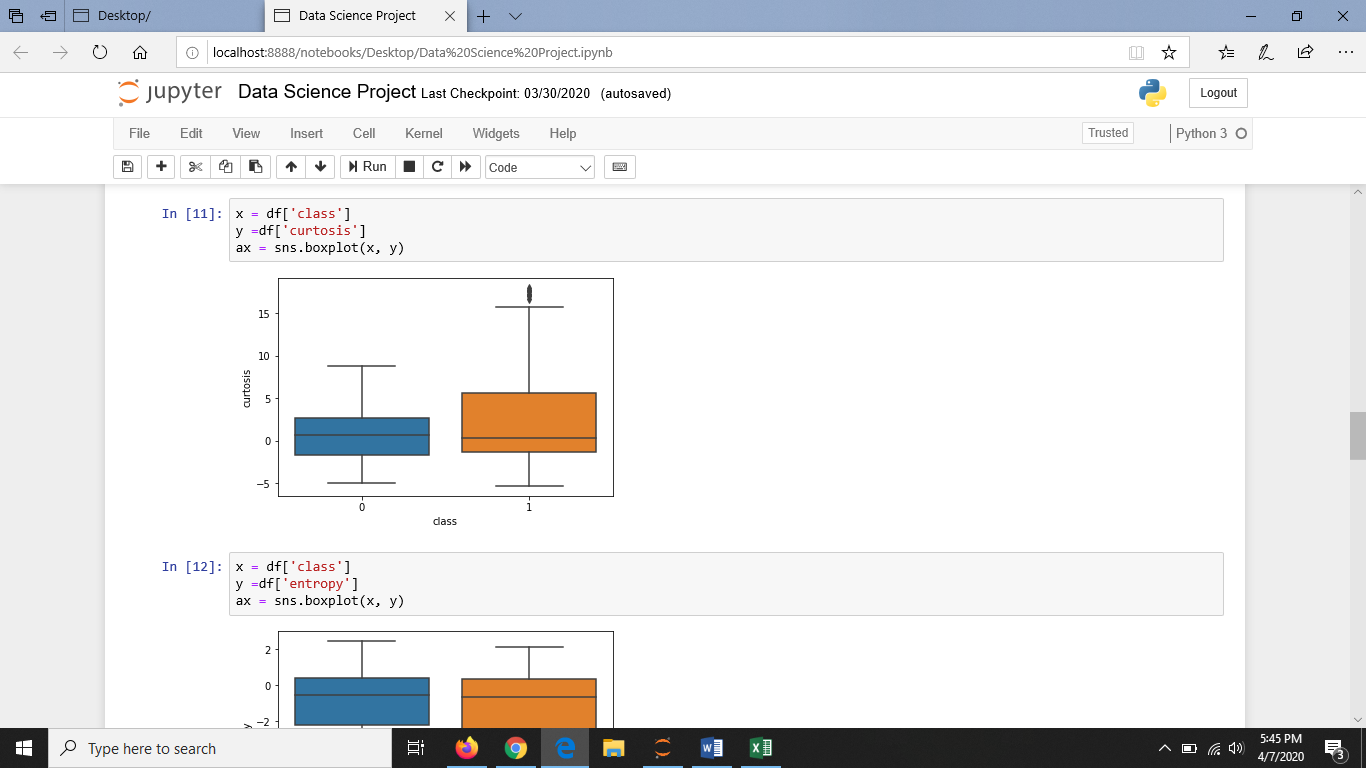
1. **KDE Plot (Kernel Density Estimate)**: It is used for visualizing the Probability Density of a continuous variable. It depicts the probability density at different values in a continuous variable. Here, it compares values of each field with the value of class field i.e, 0 and 1 which shows the probability density of a fake note and an original note.

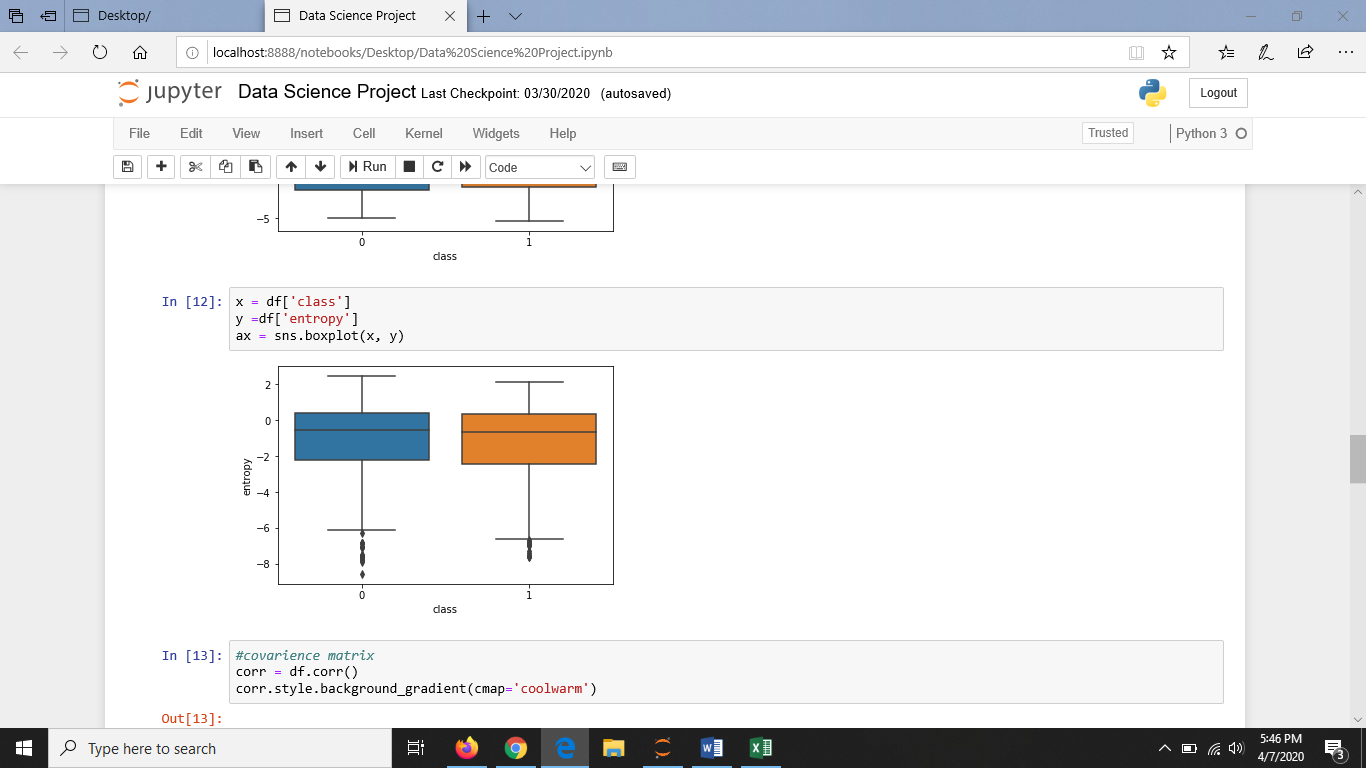


1. **BOX PLOT**: It displays the distribution of data based on the five number summary: minimum, first quartile, median, third quartile, and maximum. The dots in the graph describes those values which are the outliers and lie beyond the inner fences. Here, fake note and original notes are plotted with respect to every field so that we know the range in which either of the type of notes lie for that particular field and can easily understand the difference between them by visualization. It can be further used to identify the types of notes.









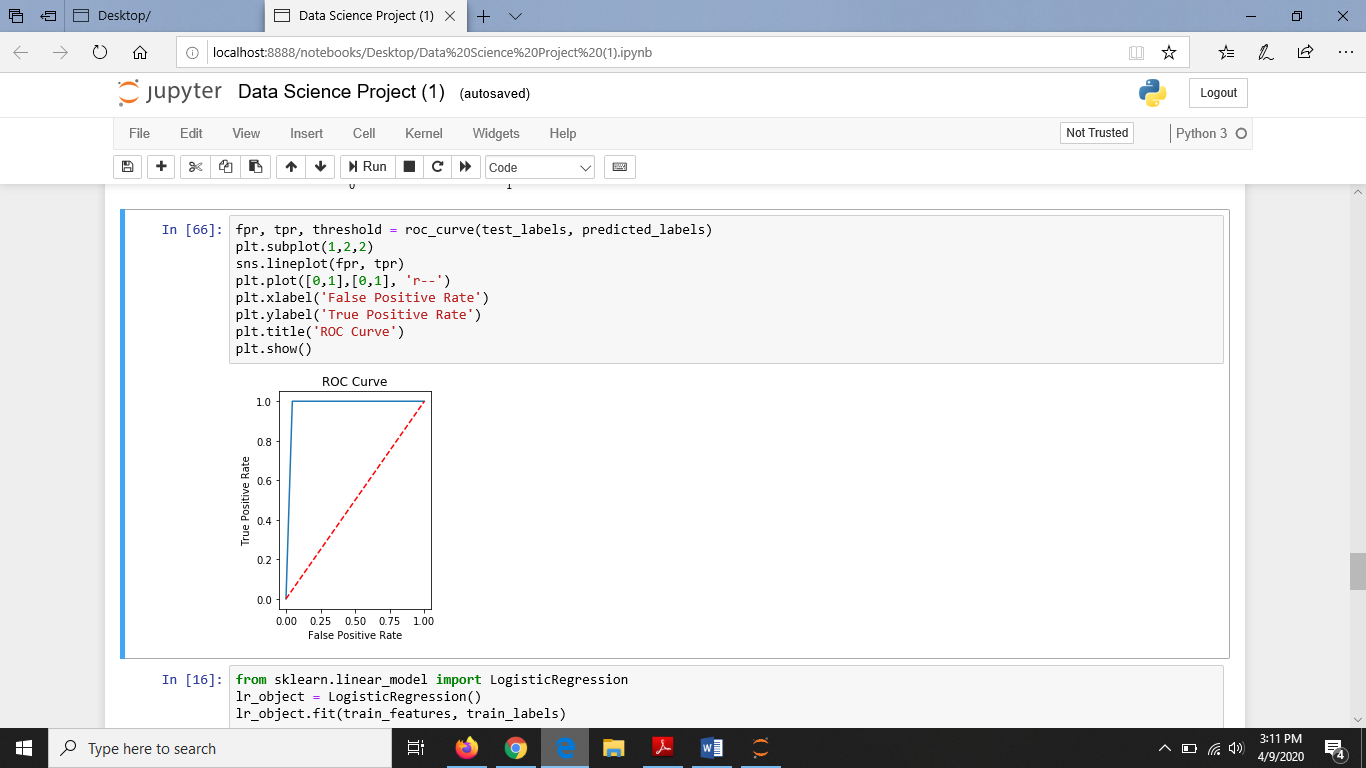
**ALGORITHM ON WHICH THEY ARE WORKING WITH GRAPHS**

1. **SVM**

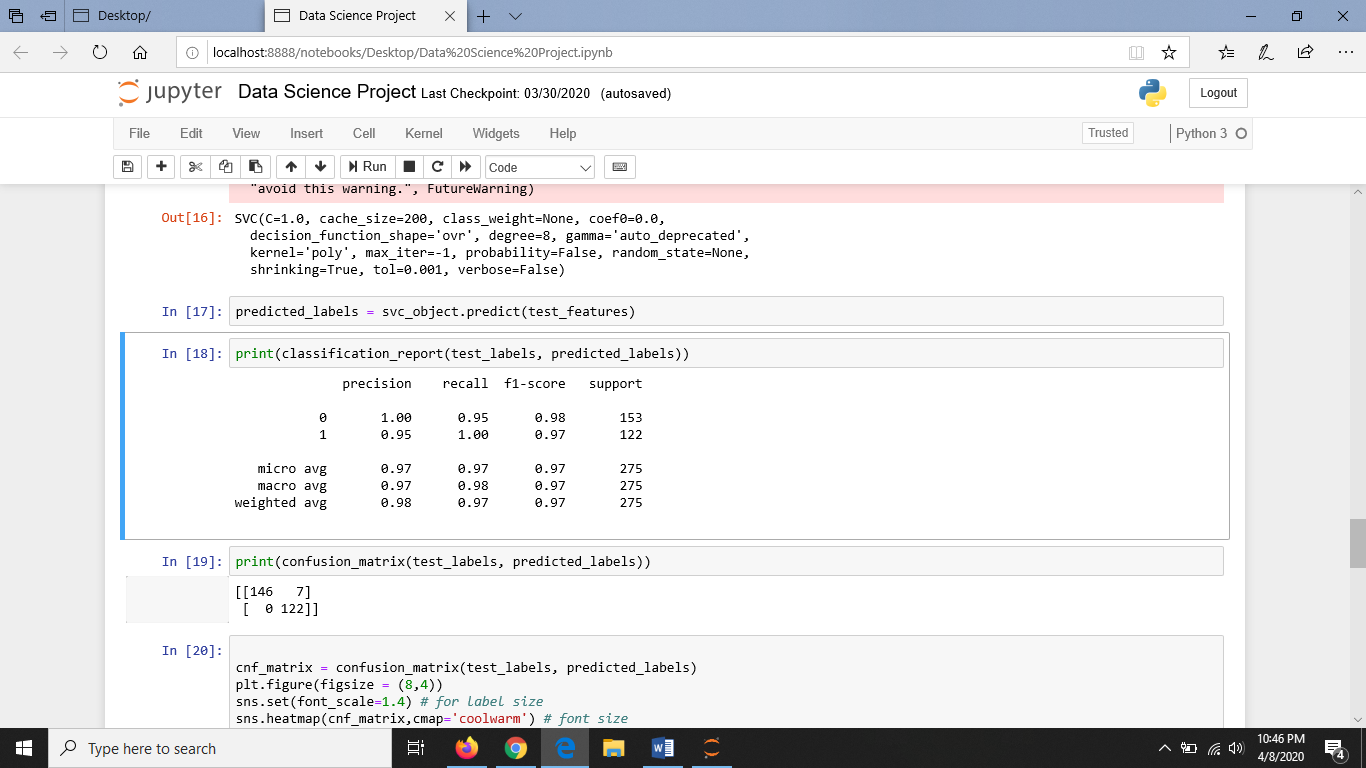
Support Vector Machines are supervised learning models that evaluate the data and recognize the patterns to classify the data. It creates a decision boundary to separate the two classes in the data. In SVM, each data item is plotted on the graph and then classification is performed to find the hyper plane that differentiates the two classes.

SVM uses a kernel function that projects the data from a lower-dimensional space to a higher-dimensional space. This is done to make the non-linearly separable data into linearly separable.

Kernel functions are used in SVM as SVM does not perform well with huge datasets. For implementation purpose, linear kernel is used which is especially used for classification where there are a few features and the number of test cases is large. In the model, as the dataset has linearly separable data, the linear kernel finds the linear margin that separates the two regions in the graph. This decision boundary is chosen such that it is maximally far away from each data point.



1. For Test:Train Ratio= 20:80

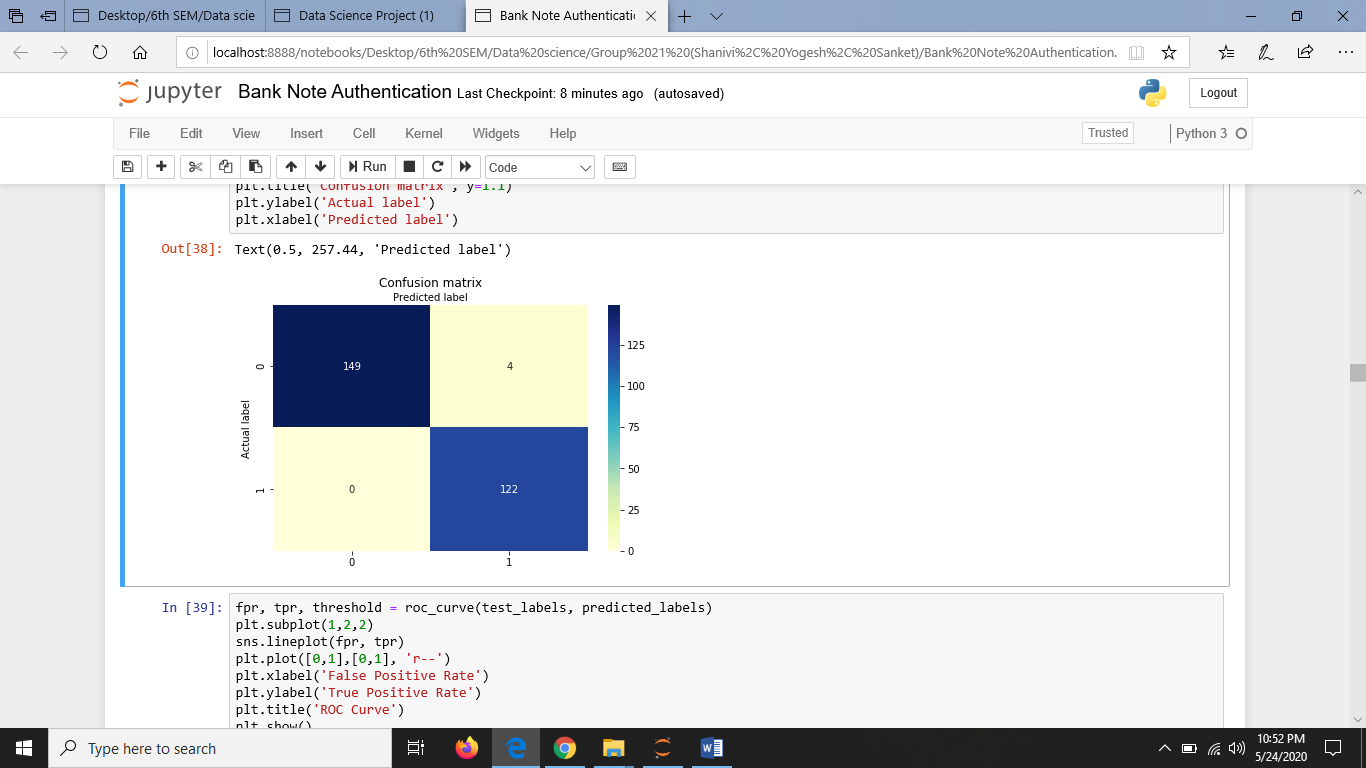
 The classification report is as follow

Accuracy: 0.9854545454545455

Sensitivity : 0.9738562091503268

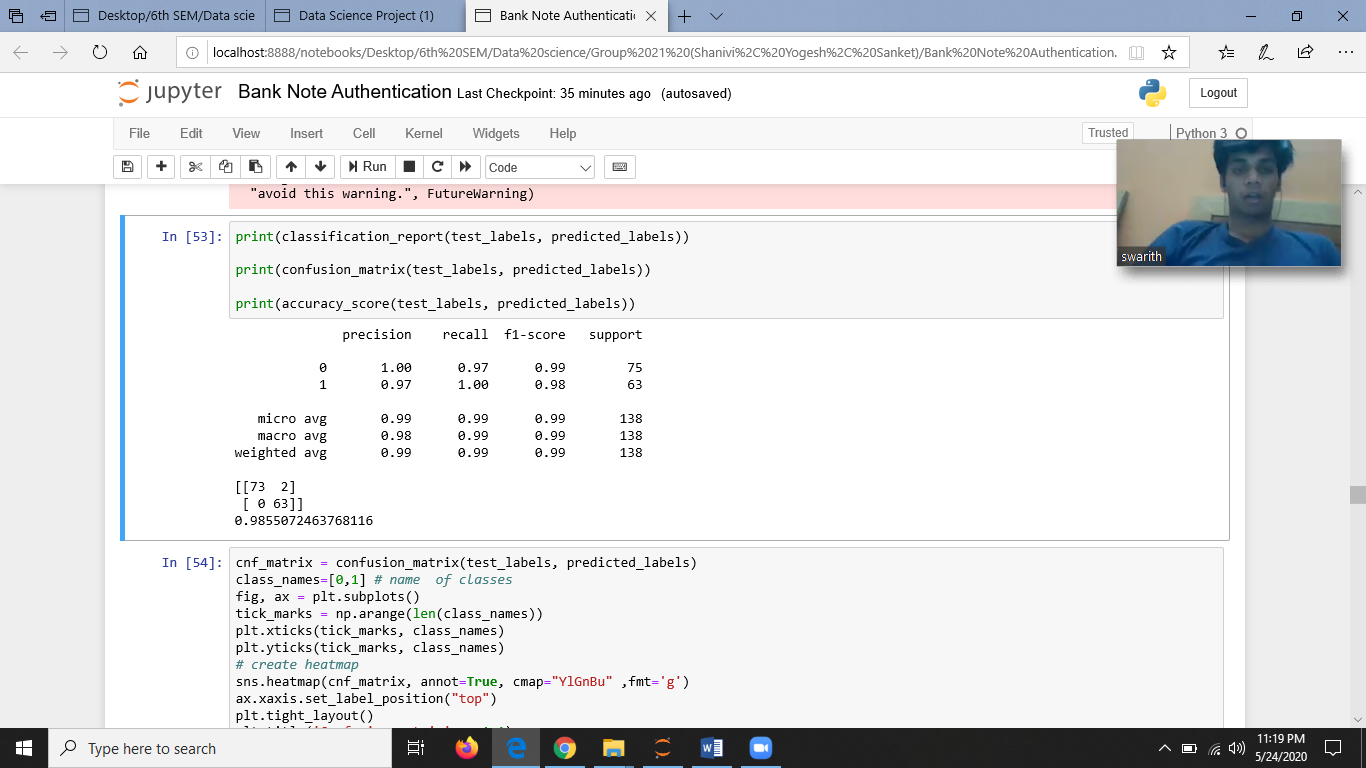
Specificity : 1.0

MCC: 0.9710510486161736



1. For Test:Train Ratio= 10:90

The classification report is as follows:

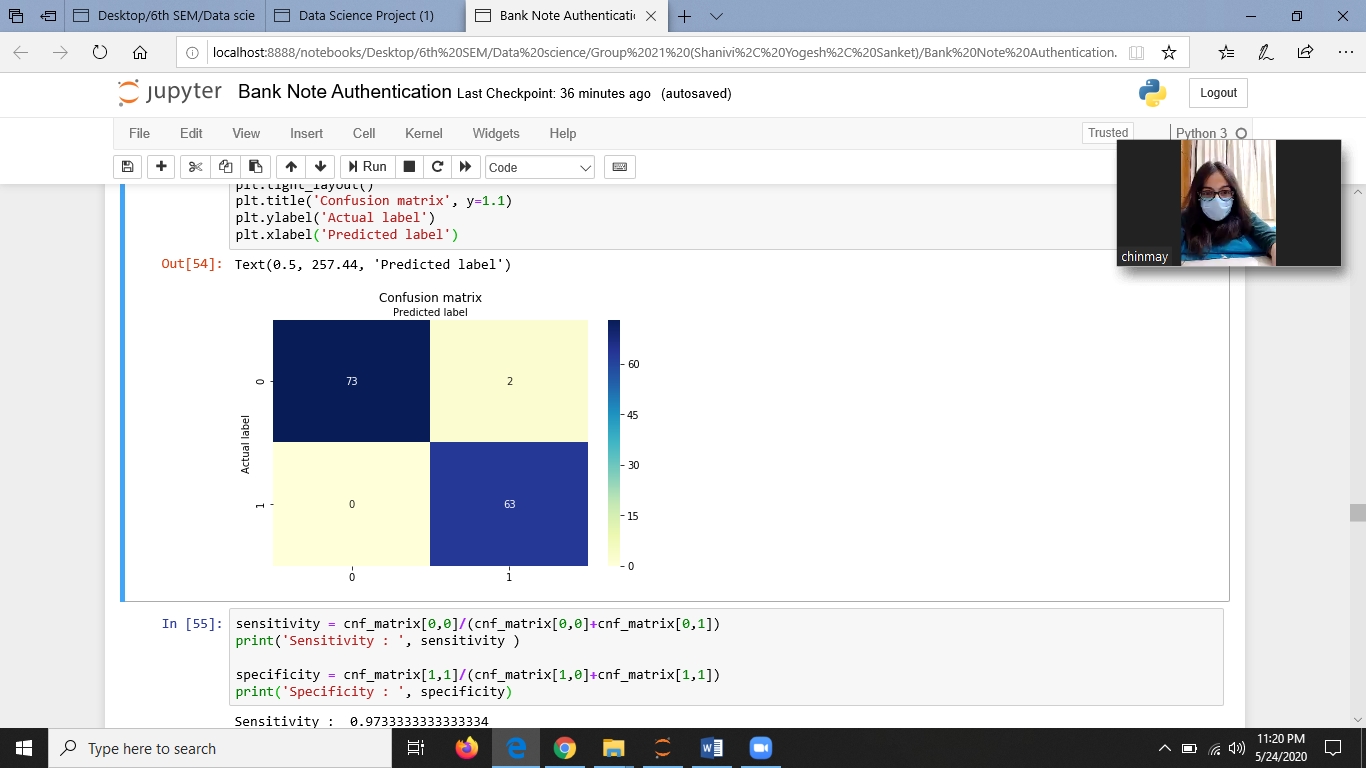


Accuracy: 0.9855072463768116

Sensitivity : 0.9733333333333334

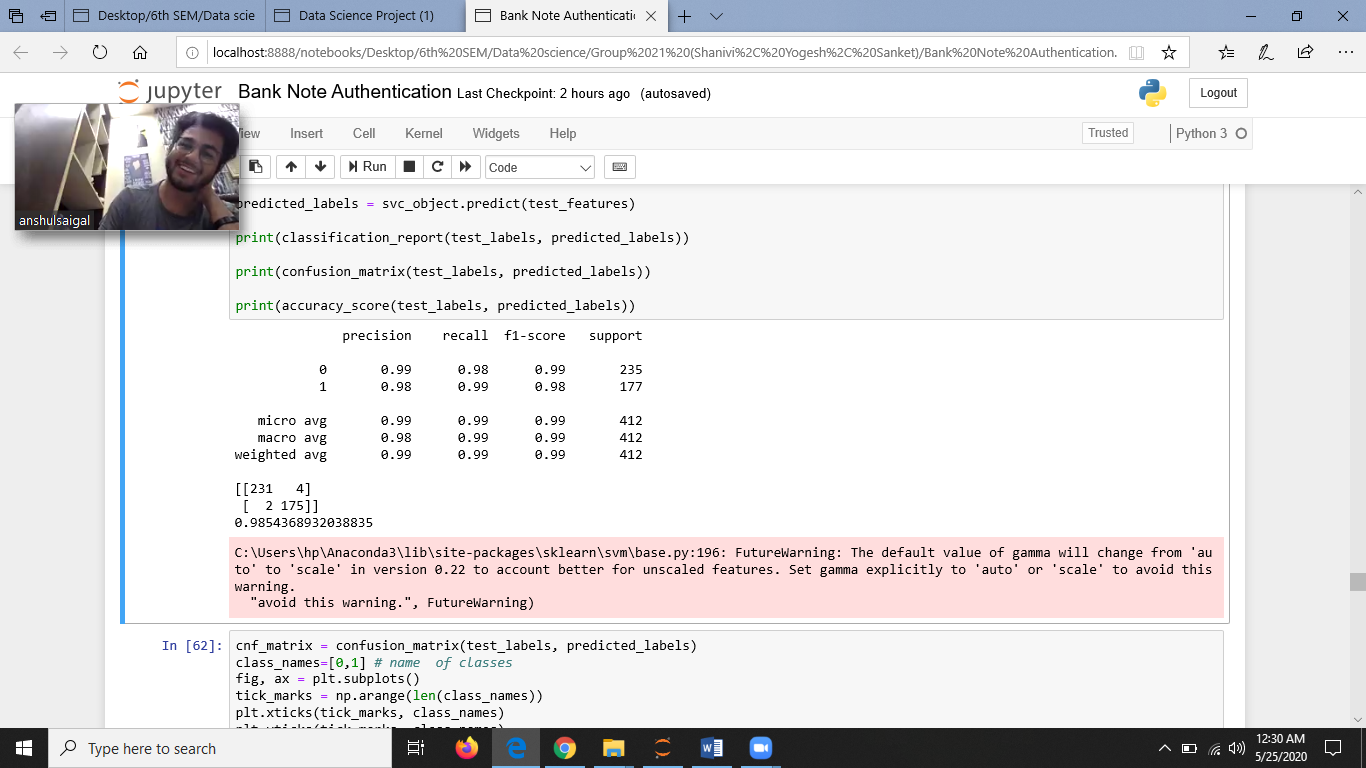
Specificity : 1.0

MCC: 0.9712798851951046



1. For Test:Train Ratio= 30:70

The classification report is as follows:

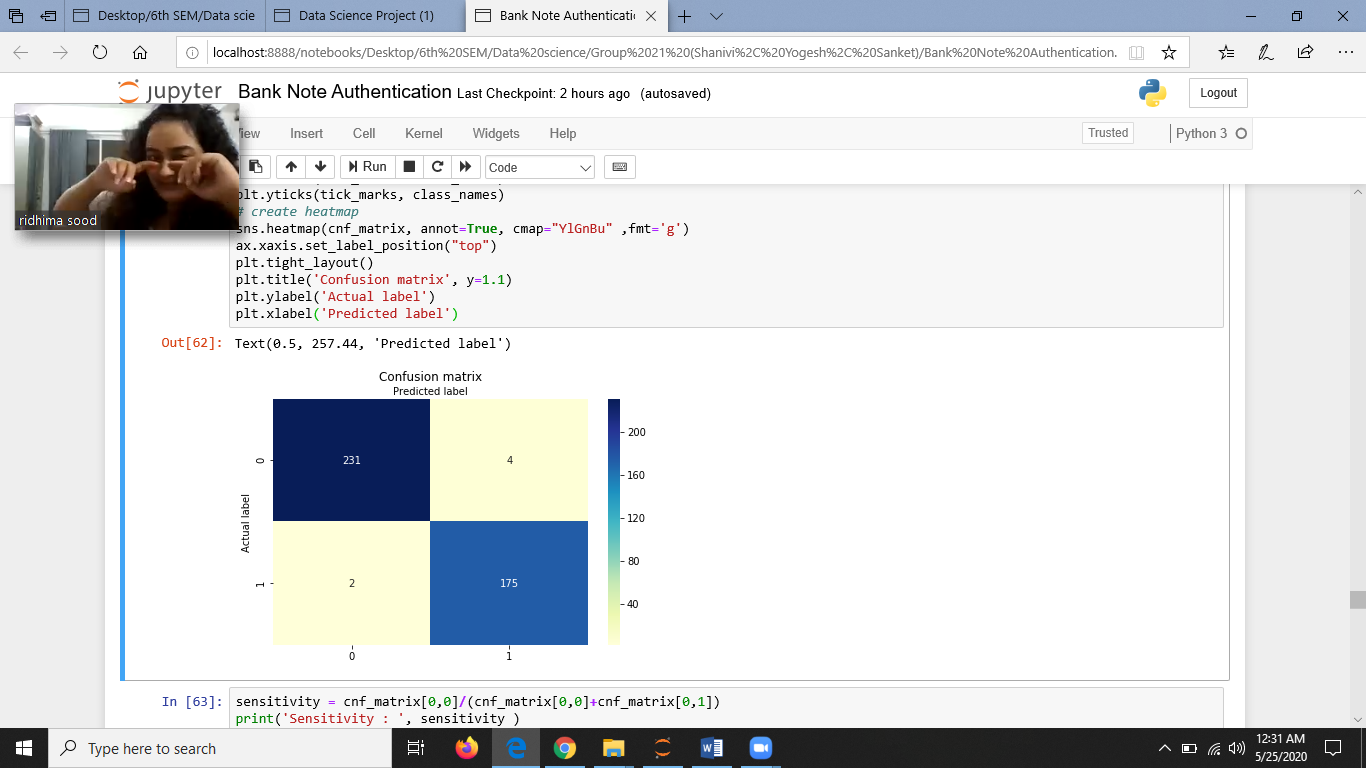


Accuracy: 0.9854368932038835

Sensitivity : 0.9829787234042553

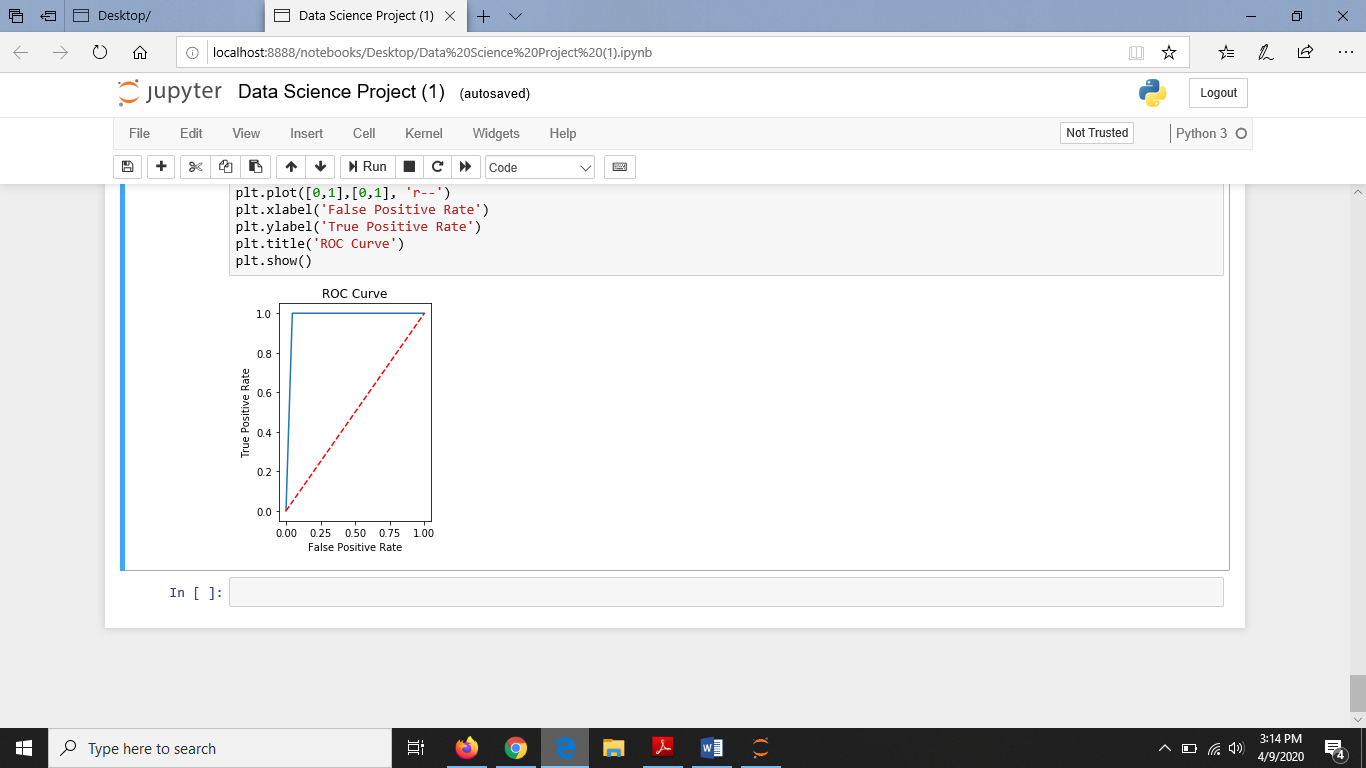
Specificity : 0.9887005649717514

MCC: 0.9703737372657699



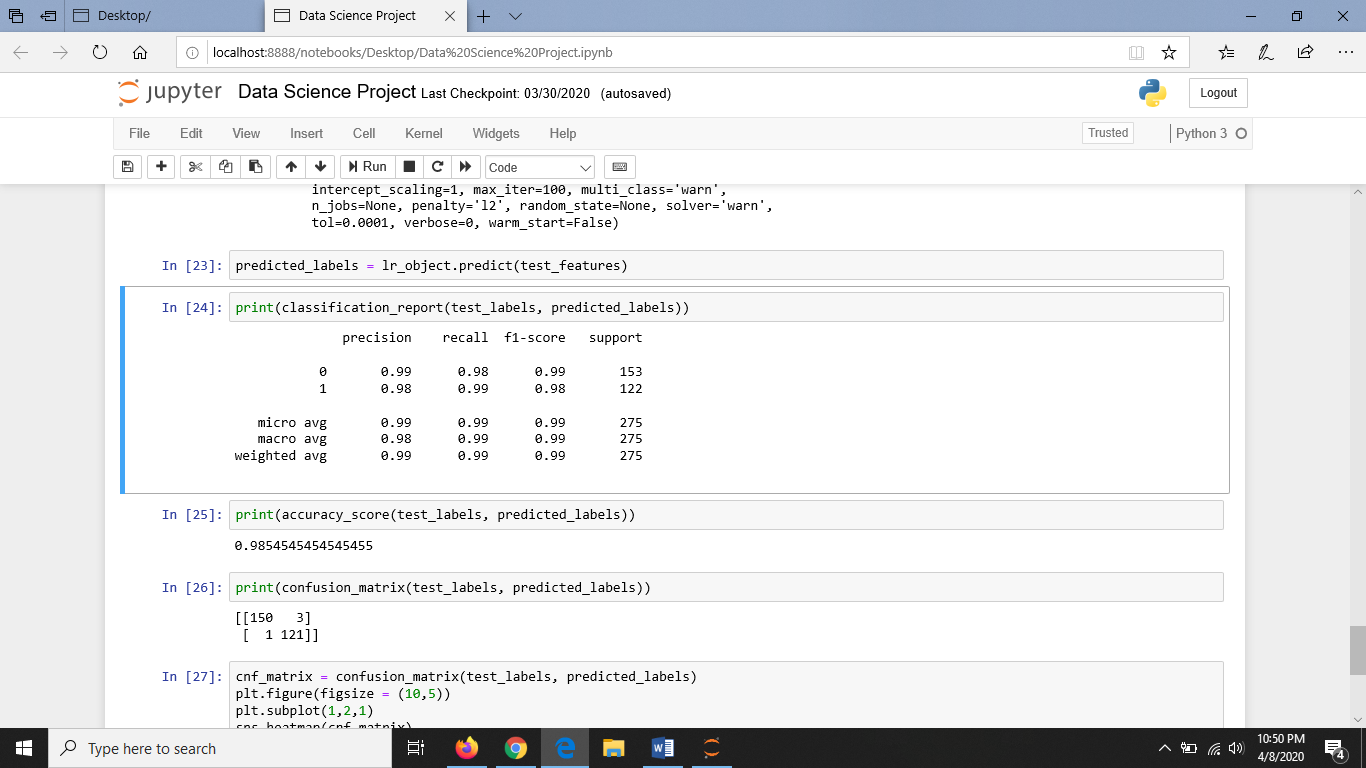
1. **LOGISTIC REGRESSION**

Logistic regression is a classification model that is very easy to implement but performs very well on linearly separable classes. It is one of the most widely used algorithms for classification in industry. The logistic regression model is a linear model for binary classification that can be extended to multiclass classification via the OvR technique.



1. For Test:Train Ratio= 20:80

The classification report is as follows:

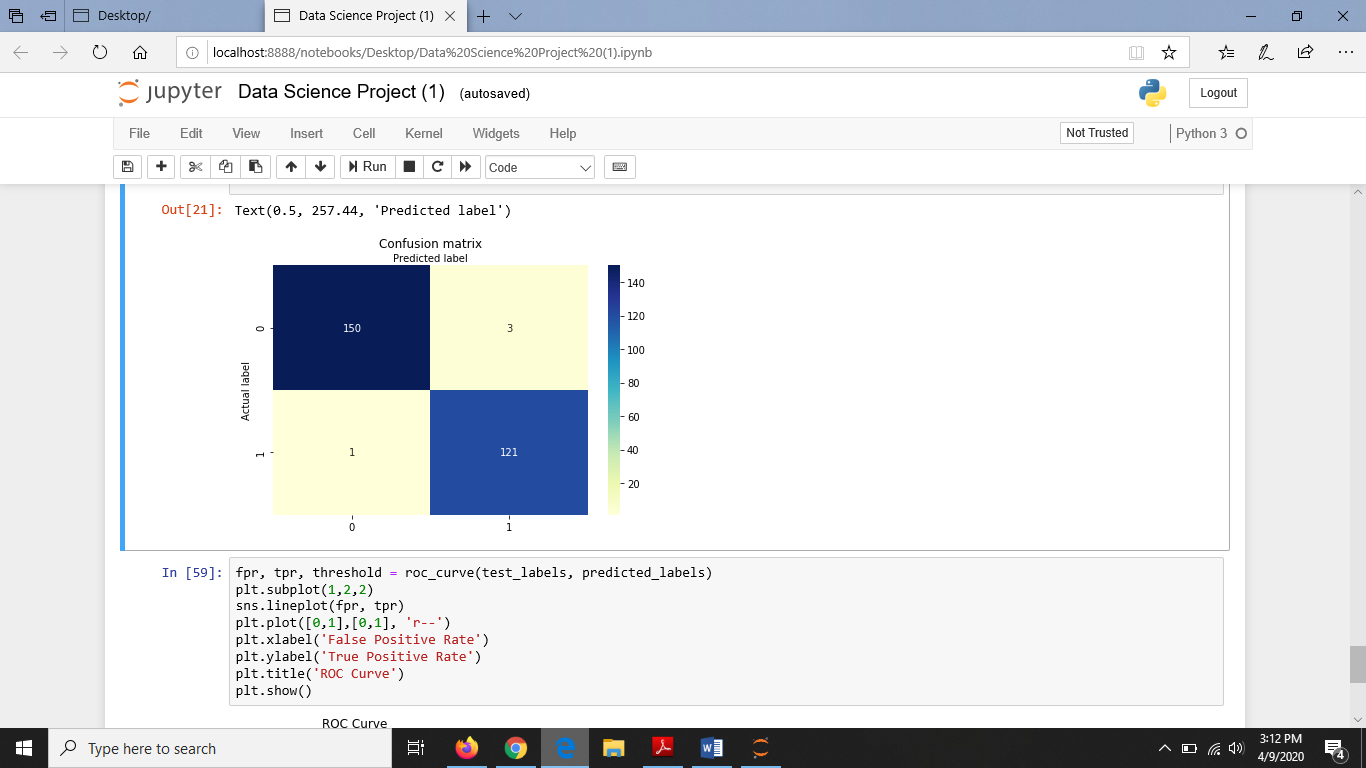


Accuracy: 0.9854545454545455

Sensitivity : 0.9803921568627451

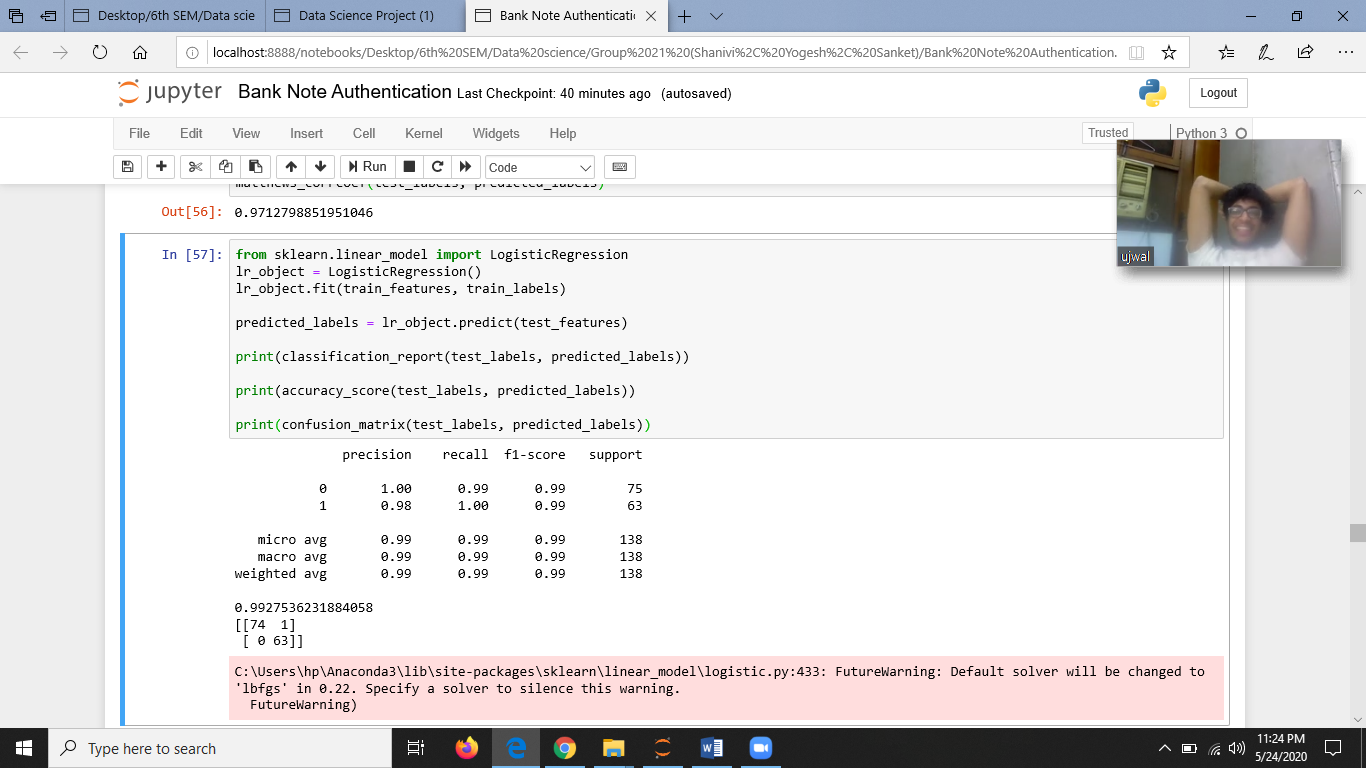
Specificity : 0.9918032786885246

MCC: 0.9706885174306206



1. For Test:Train Ratio= 10:90

The classification report is as follows:

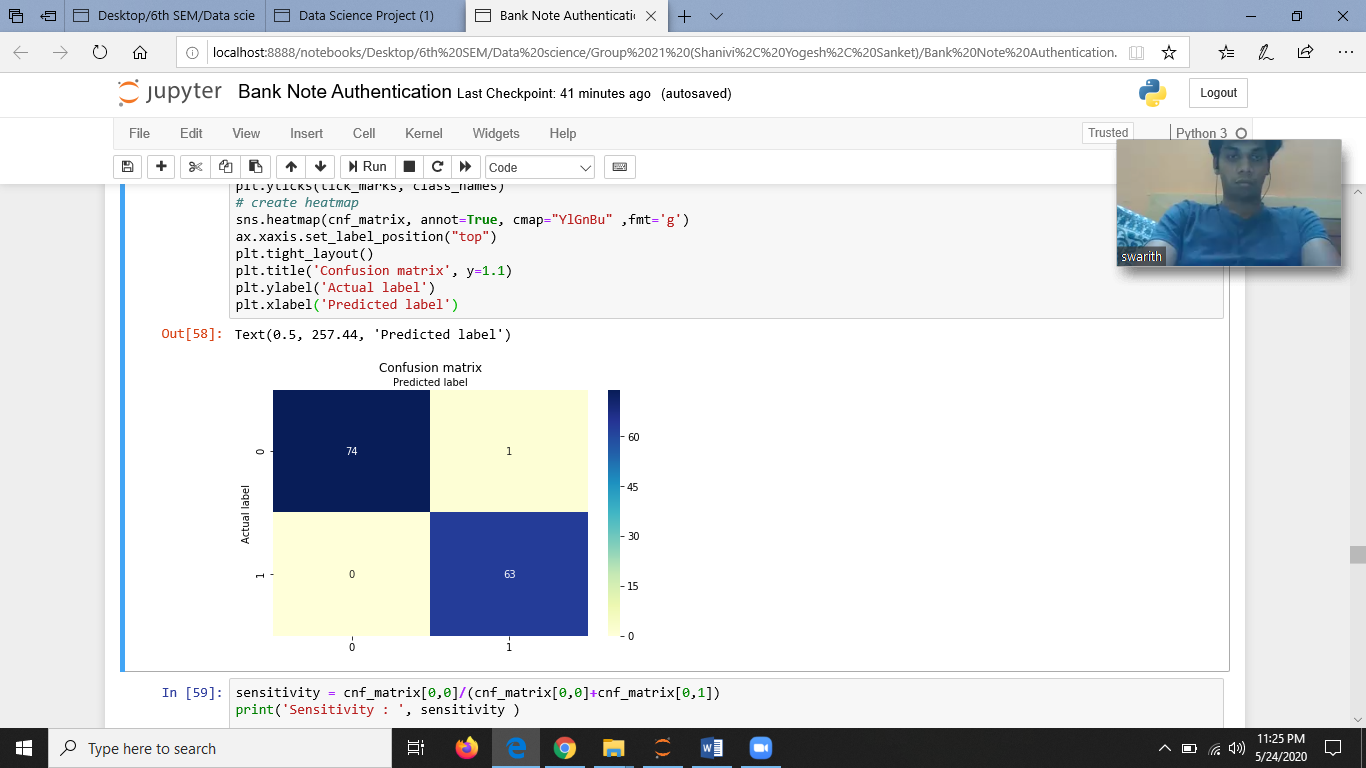


Accuracy: 0.9927536231884058

Sensitivity : 0.9866666666666667

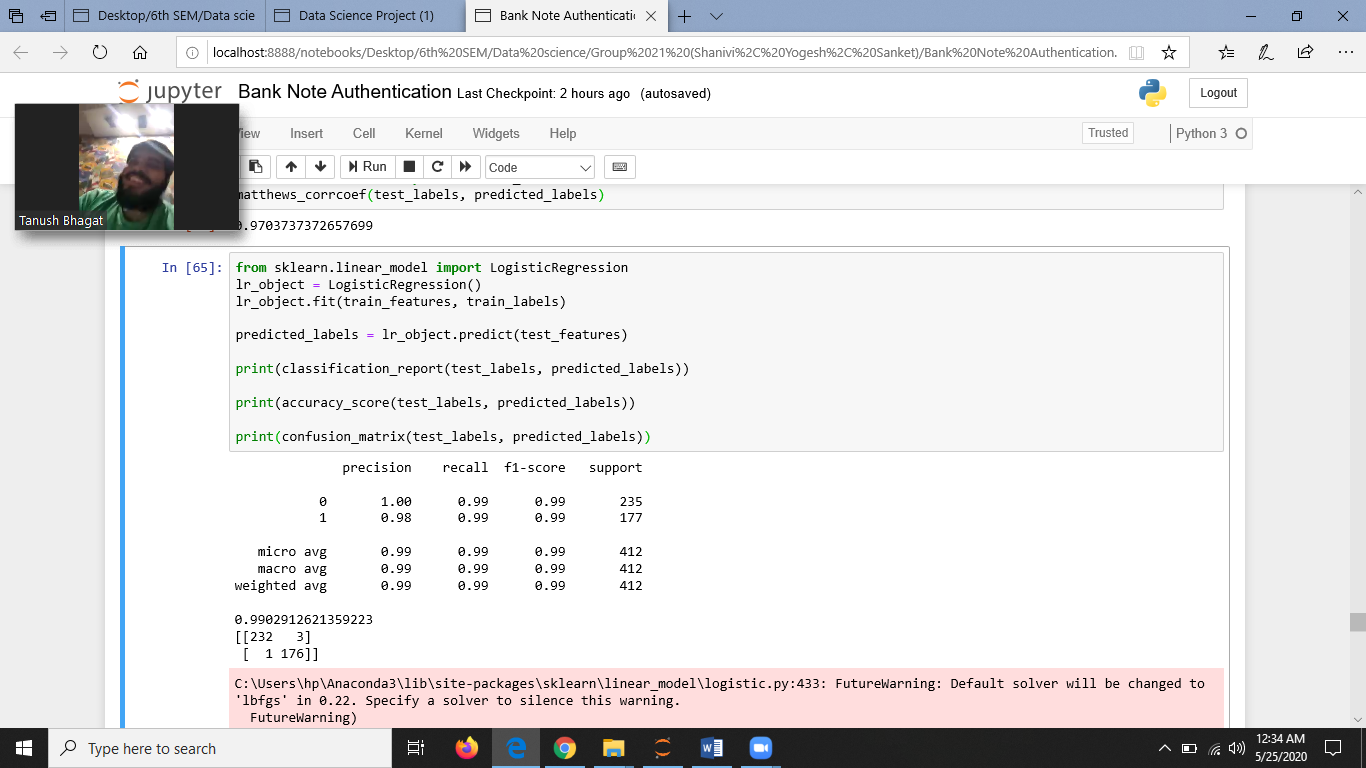
Specificity : 1.0

MCC:0.9855201672213512



1. For Test:Train Ratio= 30:70

The classification report is as follows:

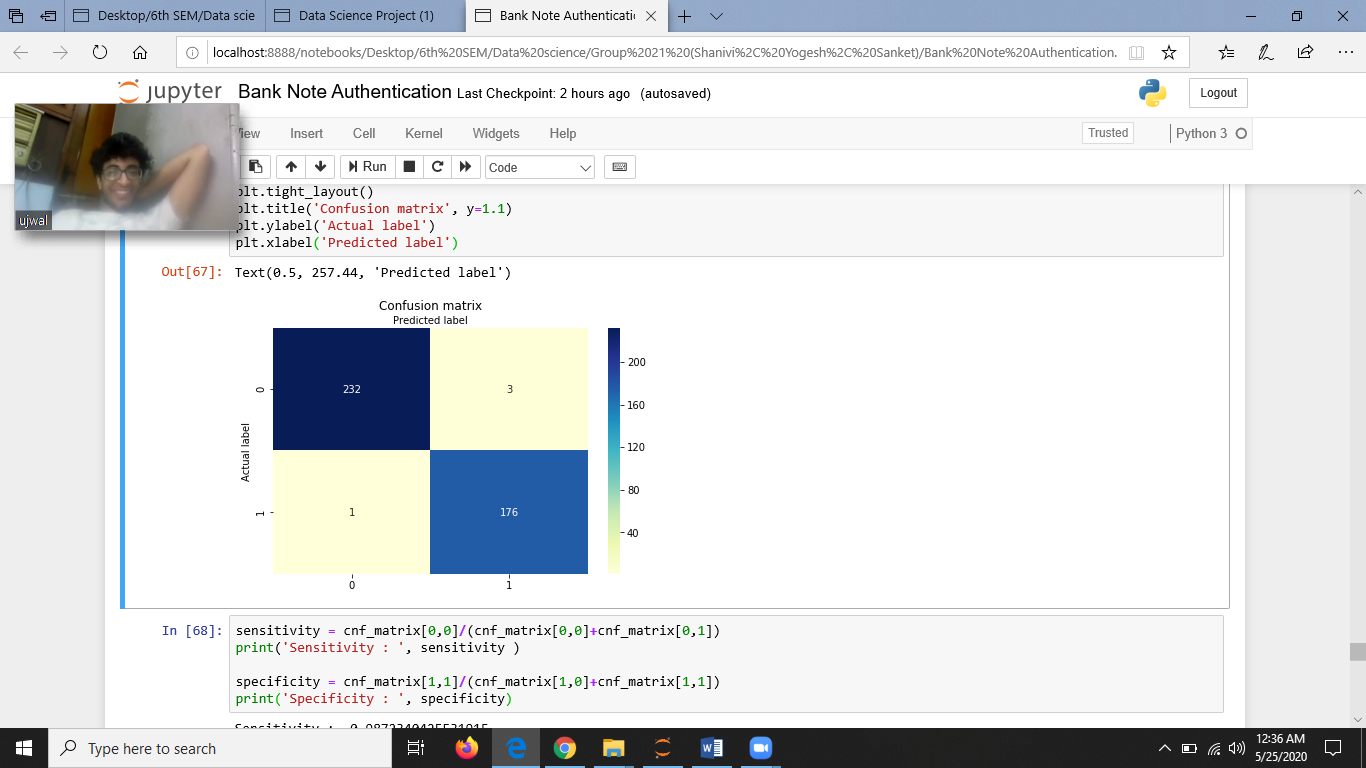


Accuracy: 0.9902912621359223

Sensitivity: 0.9872340425531915

Specificity: 0.9943502824858758

MCC: 0.9802654654928401



**SUMMARY :**

1. For Test:Train Ratio= 20:80

|  |  |  |
| --- | --- | --- |
| **Algorithms→**  **Measures↓** | **SVM** | **Logistic Regression** |
| **Specificity** | 1.0 | 0.9918032786885246 |
| **Sensitivity** | 0.9738562091503268 | 0.9803921568627451 |
| **Accuracy** | 0.9854545454545455 | 0.9854545454545455 |
| **Precision** | 0.96825397 | 0.97580695 |
| **FPR** | 0.00524934 | 0.00393701 |
| **FNR** | 0 | 0.00163934 |
| **NPV** | 1 | 0.99337748 |
| **FDR** | 0.03174603 | 0.02419355 |
| **F1- Score** | 0.98387097 | 0.98374009 |
| **MCC** | 0.9710510486161736 | 0.9706885174306206 |

1. For Test:Train Ratio= 10:90

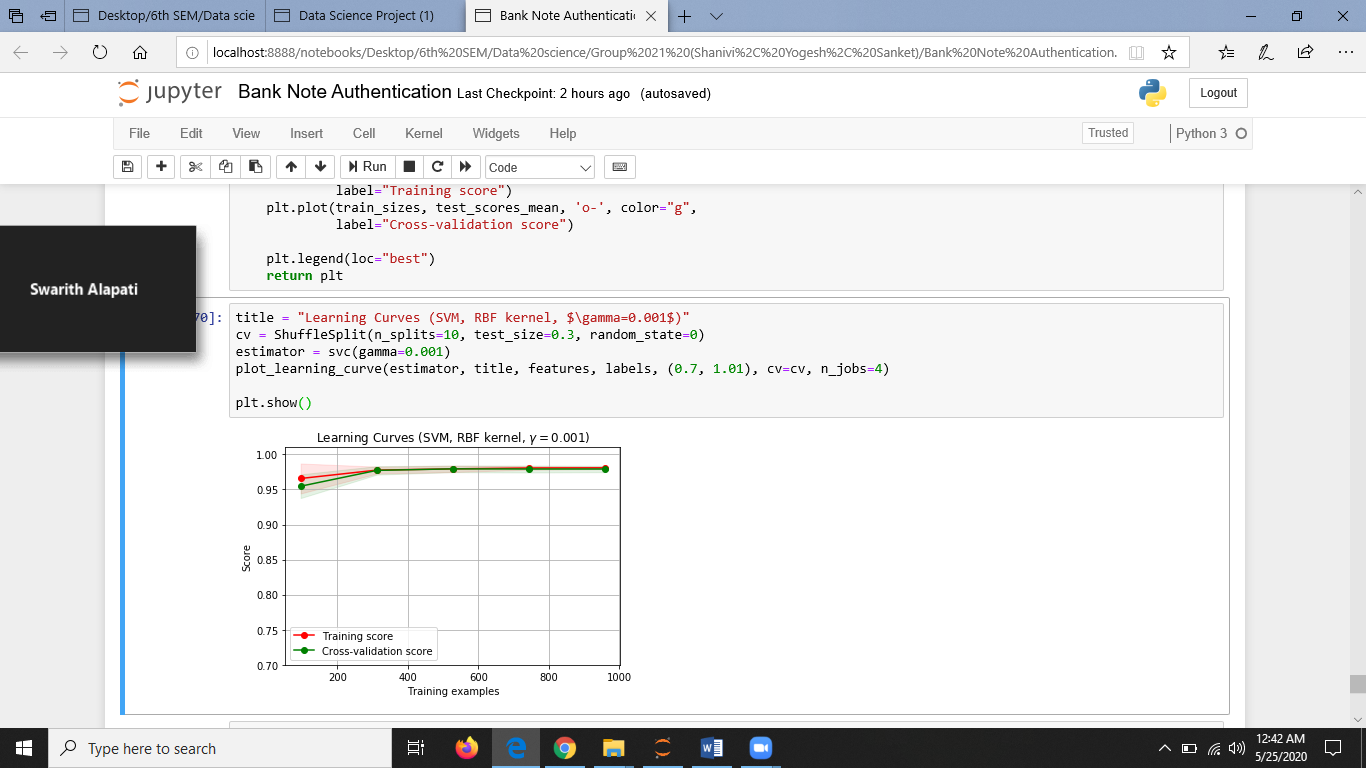
|  |  |  |
| --- | --- | --- |
| **Algorithms→**  **Measures↓** | **SVM** | **Logistic Regression** |
| **Specificity** | 1.0 | 1.0 |
| **Sensitivity** | 0.9733333333333334 | 0.9866666666666667 |
| **Accuracy** | 0.9855072463768116 | 0.9927536231884058 |
| **Precision** | 0.984375 | 0.984375 |
| **FPR** | 0.00262467 | 0.00131234 |
| **FNR** | 0 | 0 |
| **NPV** | 1 | 1 |
| **FDR** | 0.03076923 | 0.015625 |
| **F1- Score** | 0.99212598 | 0.99212598 |
| **MCC** | 0.9712798851951046 | 0.9855201672213512 |

1. For Test:Train Ratio= 30:70

|  |  |  |
| --- | --- | --- |
| **Algorithms→**  **Measures↓** | **SVM** | **Logistic Regression** |
| **Specificity** | 0.9887005649717514 | 0.9943502824858758 |
| **Sensitivity** | 0.9829787234042553 | 0.9872340425531915 |
| **Accuracy** | 0.9854368932038835 | 0.9902912621359223 |
| **Precision** | 0.97765363 | 0.98324022 |
| **FPR** | 0.00524934 | 0.00393701 |
| **FNR** | 0.00327869 | 0.00163934 |
| **NPV** | 0.99141631 | 0.99570815 |
| **FDR** | 0.02234637 | 0.01675978 |
| **F1- Score** | 0.98314606 | 0.98876404 |
| **MCC** | 0.9703737372657699 | 0.9802654654928401 |

**LEARNING CURVES:**

**SVM**



**LOGISTIC REGRESSION**

