Data science Project Report on

**Banknote Authentication System**

Submitted to Manipal University, Jaipur

Towards the partial fulfillment for the Award of the Degree of

**BACHELORS OF TECHNOLOGY**

In Information Technology

2019-2020

By

Himanshu Mittal (179302059)



Under the guidance of

Dr. Anju Yadav

**Department of Information Technology**

**School of Computing and Information Technology**

**Manipal University Jaipur, Rajasthan**

**CERTIFICATE**

Date: 24/05/2020

This is to certify that the Data science project titled “Banknote Authentication System using Decision Tree and Naïve Bayes” is a record of the Bonfide work done by “Himanshu Mittal (179302059)”, submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology B. Tech in Information Technology of Manipal University Jaipur, during the academic year of 2019-20

Dr. Anju Yadav Dr. Pankaj Vyas

Assistant Professor HOD

Department of IT Department of IT

MUJ MUJ

# ACKNOWLEDGEMENT

# Apart from the efforts of myself, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project. I would like to show my greatest appreciation to Dr. Anju Yadav. I can’t say thank you enough for his tremendous support and help. I feel motivated and encouraged every time I attend his meeting. Without his encouragement and guidance this project would not have materialized. The guidance and support received from other professors was vital for the success of the project. I am grateful for their constant support and help

# Sincerely,

Himanshu Mittal (179302059)

# ABSTRACT

Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data.  Data science is related to data mining, deep learning and big data.

Data science is a "concept to unify statistics, data analysis, machine learning and their related methods" in order to "understand and analyze actual phenomena" with data. It uses techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, and information science. Turing award winner Jim Gray imagined data science as a "fourth paradigm" of science (empirical, theoretical, computational and now data-driven) and asserted, "Everything about science is changing because of the impact of information technology" and the data deluge.

Banknotes are one of the most important assets of a country. Some miscreants introduce fake notes, which bear a resemblance to original note to create discrepancies of the money in the financial market. It is difficult for humans to tell true and fake banknotes apart especially because they have many similar features. Forged notes are created with precision, hence there is need for an efficient algorithm, which accurately predicts whether a banknote is genuine, or not.

This paper proposes machine-learning techniques to evaluate authentication of banknotes. Supervised learning algorithms such as Decision Tree Classification and Naïve Bayes Classification are used for differentiating genuine banknotes from forged ones. The study also shows the comparison of these algorithms in classification of banknotes

# CONTENT

**Certificate 2**

**Acknowledgement 3**

**Abstract 4**

**Chapter 01 INTRODUCTION 6**

**1.1 Problem Statement 6**

**1.2 Methodology 7**

**Chapter 02 ALGORITHMS USED 8**

**2.1 Decision Tree Classification 8**

**2.2 Naïve Bayes Classification 17**

**Chapter 03 IMPLEMENTATION OF DECISION TREE 20**

**Chapter 04 IMPLEMENATATION OF NAÏVE BAYES 33**

**Chapter 05 RESULT 47**

**Chapter 06 REFERENCE 54**

# INTRODUCTION

**Problem Statement:**

**Banknote Authentication**

Banknotes are currencies used by any nation to carryout financial activities and are every countries asset, which every nation wants it (bank note) to be genuine. Lot of miscreants induces fake notes into the market, which resemble exactly the original note. Hence, there is a need for an efficient authentication system, which predicts accurately whether the given note is genuine, or not.

**Data set:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variance** | **Skewness** | **Curtosis** | **Entropy** | **Class** |
| 3.6216 | 8.6661 | -2.8073 | -0.44699 | 0 |
| 4.5459 | 8.1674 | -2.4586 | -1.4621 | 0 |
| 3.866 | -2.6383 | 1.9242 | 0.10645 | 0 |
| 3.4566 | 9.5228 | -4.0112 | -3.5944 | 0 |
| 0.32924 | -4.4552 | 4.5718 | -0.9888 | 0 |
| 4.3684 | 9.6718 | -3.9606 | -3.1625 | 0 |

**METHODOLOGY**

* First objective of the project is to collect banknote dataset from Kaggle
* Before we can import our dataset and perform analysis, we need to import a few libraries
* Once we import the libraries, the next step is to load the dataset into our application. To do so, we used the “read\_csv ()” function of the Pandas library, which reads dataset that is in the CSV format
* To see how the dataset actually looks, we can use the “head ()” function of the Pandas data frame and to see the statistical details of data, we can use “describe ()” function of pandas data frame:
* We will segregate the variables into x and y using the preprocessing library of sklearn and apply train\_test\_split on x and y to train the dataset. We will also normalize the values of x variable to make the scale of each value equal.

**Decision Tree Classifier**

* Then we have to select the appropriate maximum depth of tree by plotting the graph between train accuracy and test accuracy
* We will build the model by importing Decision Tree classification from tree library in sklearn and then by creating an object of it and fitting the train data into this model.
* We will calculate the evaluation metrics of model using Accuracy, Loss graph, AUC

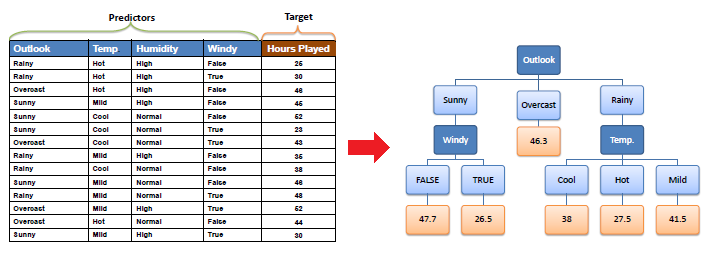
**Naïve Bayes Classifier**

* At last,We will build the model by importing GaussianNB from naïve\_bayes library in sklearn and then by creating an object of it and fitting the train data into this model
* We will calculate the evaluation metrics of model using Accuracy, Loss graph, AUC

**ALGORITHMS USED**

* **Decision tree regression**

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree, which corresponds to the best predictor, called root node. Decision trees can handle both categorical and numerical data.

****

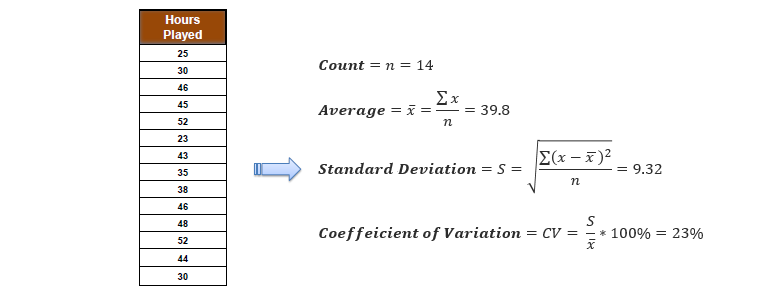
**Decision tree algorithm**

The core algorithm for building decision trees called ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with Standard Deviation Reduction.

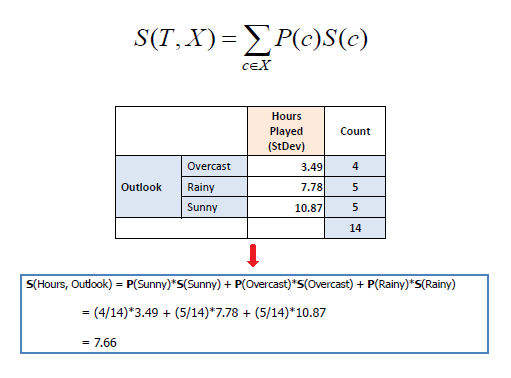
**Standard Deviation**

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). We use standard deviation to calculate the homogeneity of a numerical sample. If the numerical sample is completely homogeneous, its standard deviation is zero.

* Standard deviation for one attribute:

****

* **Standard deviation for two attributes (target and predictor):**

****

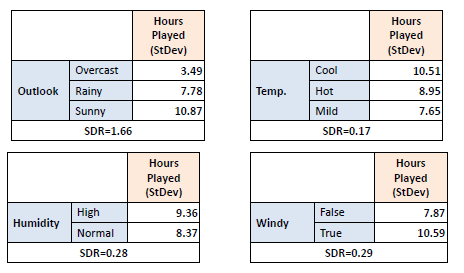
**Standard Deviation Reduction**

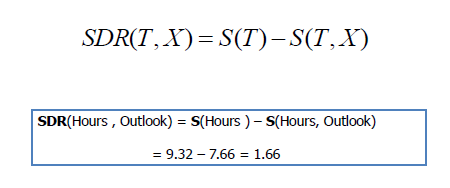
The standard deviation reduction is based on the decrease in standard deviation after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest standard deviation reduction (i.e., the most homogeneous branches).

* **Step 1:** The standard deviation of the target is calculated.

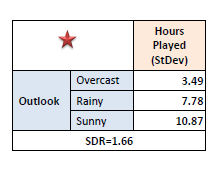
Standard deviation (Hours Played) = 9.32

* **Step 2:** The dataset is then split on the different attributes. The standard deviation for each branch is calculated. The resulting standard deviation is subtracted from the standard deviation before the split. The result is the standard deviation reduction.

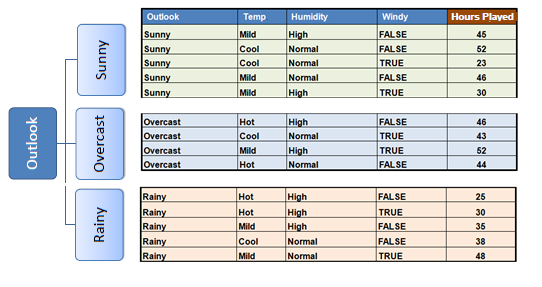
****

****

* **Step 3:** The attribute with the largest standard deviation reduction is chosen for the decision node.

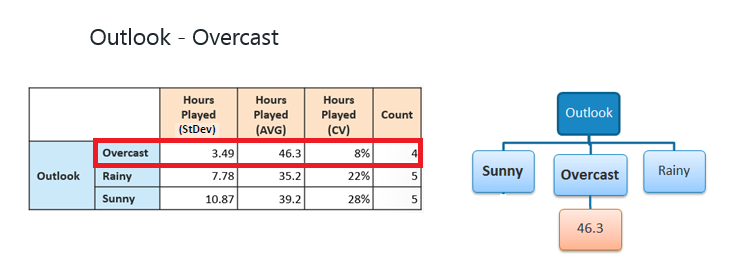
****

* **Step 4a:** The dataset is divided based on the values of the selected attribute. This process is run recursively on the non-leaf branches, until all data is processed.

****

In practice, we need some termination criteria. For example, when coefficient of deviation (CV) for a branch becomes smaller than a certain threshold (e.g., 10%) and/or when too few instances (n) remain in the branch (e.g., 3).

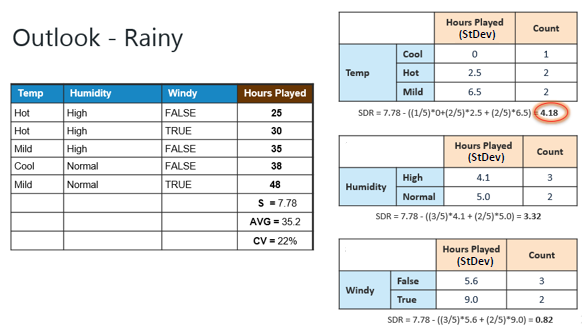
* **Step 4b:** "Overcast" subset does not need any further splitting because its CV (8%) is less than the threshold (10%). The related leaf node gets the average of the "Overcast" subset.

****

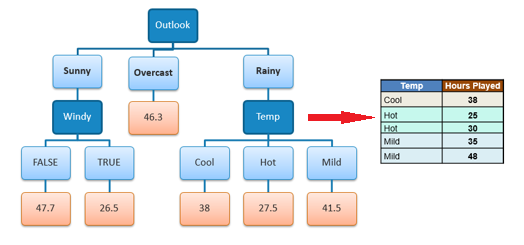
* **Step 4c:** However, the "Sunny" branch has a CV (28%) more than the threshold (10%) which needs further splitting. We select "Windy" as the best node after "Outlook" because it has the largest SDR.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Because the number of data points for both branches (FALSE and TRUE) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node. |  |  |
|  |  |  |

* **Step 4d:** Moreover, the "rainy" branch has a CV (22%) which is more than the threshold (10%). This branch needs further splitting. We select "Windy" as the best node because it has the largest SDR.

****

Because the number of data points for all three branches (Cool, Hot and Mild) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node.

****

When the number of instances is more than one at a leaf node we calculate the average as the final value for the target

**Decision tree types**

Decision trees used in [data mining](https://en.wikipedia.org/wiki/Data_mining) are of two main types:

* [Classification tree](https://en.wikipedia.org/wiki/Classification_tree) analysis is when the predicted outcome is the class (discrete) to which the data belongs.
* Regression tree analysis is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).

The term Classification and Regression Tree (CART) analysis is an [umbrella term](https://en.wikipedia.org/wiki/Umbrella_term) used to refer to both of the above procedures, first introduced by [Breiman](https://en.wikipedia.org/wiki/Leo_Breiman) et al. in 1984. Trees used for regression and trees used for classification have some similarities - but also some differences, such as the procedure used to determine where to split.

Some techniques, often called *ensemble* methods, construct more than one decision tree:

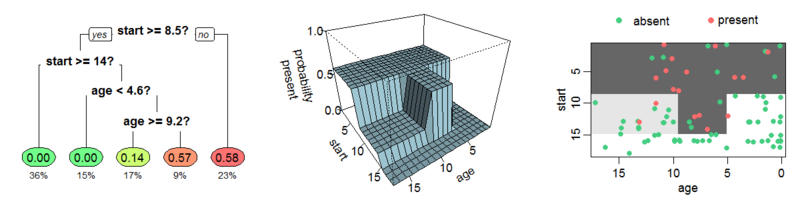
* [Boosted trees](https://en.wikipedia.org/wiki/Gradient_boosted_trees) incrementally building an ensemble by training each new instance to emphasize the training instances previously mis-modeled. A typical example is [AdaBoost](https://en.wikipedia.org/wiki/AdaBoost). These can be used for regression-type and classification-type problems.
* [Bootstrap aggregated](https://en.wikipedia.org/wiki/Bootstrap_aggregating) (or bagged) decision trees, an early ensemble method, builds multiple decision trees by repeatedly resampling training data with replacement, and voting the trees for a consensus prediction.
  + A [random forest](https://en.wikipedia.org/wiki/Random_forest) classifier is a specific type of [bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating)
* Rotation forest – in which every decision tree is trained by first applying [principal component analysis](https://en.wikipedia.org/wiki/Principal_component_analysis) (PCA) on a random subset of the input features.

A special case of a decision tree is a [decision list](https://en.wikipedia.org/wiki/Decision_list), which is a one-sided decision tree, so that every internal node has exactly 1 leaf node and exactly 1 internal node as a child (except for the bottommost node, whose only child is a single leaf node). While less expressive, decision lists are arguably easier to understand than general decision trees due to their added sparsity, permit non-greedy learning methods and monotonic constraints to be imposed.

**Decision tree learning** is the construction of a decision tree from class-labeled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node.

There are many specific decision-tree algorithms. Notable ones include:

* [ID3](https://en.wikipedia.org/wiki/ID3_algorithm) (Iterative Dichotomiser 3)
* [C4.5](https://en.wikipedia.org/wiki/C4.5_algorithm) (successor of ID3)
* [CART](https://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees_.28CART.29) (Classification And Regression Tree)[[4]](https://en.wikipedia.org/wiki/Decision_tree_learning#cite_note-bfos-4)
* [Chi-square automatic interaction detection](https://en.wikipedia.org/wiki/Chi-square_automatic_interaction_detection) (CHAID). Performs multi-level splits when computing classification trees.[[12]](https://en.wikipedia.org/wiki/Decision_tree_learning#cite_note-12)
* [MARS](https://en.wikipedia.org/wiki/Multivariate_adaptive_regression_splines): extends decision trees to handle numerical data better.
* Conditional Inference Trees. Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid overfitting. This approach results in unbiased predictor selection and does not require pruning

****

**An example tree which estimates the probability of** [**kyphosis**](https://en.wikipedia.org/wiki/Kyphosis) **after surgery, given the age of the patient and the vertebra at which surgery was started. The same tree is shown in three different ways. Left the colored leaves show the probability of kyphosis after surgery, and percentage of patients in the leaf. Middle the tree as a perspective plot. Right Aerial view of the middle plot. The probability of kyphosis after surgery is higher in the darker areas**

* **Naive Bayes algorithm**

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of [feature](https://en.wikipedia.org/wiki/Feature_vector) values, where the class labels are drawn from some finite set. There is not a single [algorithm](https://en.wikipedia.org/wiki/Algorithm) for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is [independent](https://en.wikipedia.org/wiki/Independence_(probability_theory)) of the value of any other feature, given the class variable. For example, a fruit may be considered an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible [correlations](https://en.wikipedia.org/wiki/Correlation_and_dependence) between the color, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) setting. In many practical applications, parameter estimation for naive Bayes models uses the method of [maximum likelihood](https://en.wikipedia.org/wiki/Maximum_likelihood); in other words, one can work with the naive Bayes model without accepting [Bayesian probability](https://en.wikipedia.org/wiki/Bayesian_probability) or using any Bayesian methods.

**Bayes’ Theorem**

Bayes’ Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes’ theorem is stated mathematically as the following equation:



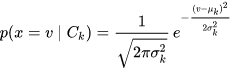
Where A and B are events

* We are trying to find probability of event A; given the event, B is true. Event B is also termed as evidence.
* P (A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance (here, it is event B).
* P (A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

### Gaussian naive Bayes

When dealing with continuous data, a typical assumption is that, the continuous values associated with each class are distributed according to a [normal](https://en.wikipedia.org/wiki/Normal_distribution) (or Gaussian) distribution. For example, suppose the training data contains a continuous attribute, we first segment the data by the class, and then compute the mean and [variance](https://en.wikipedia.org/wiki/Variance#Estimating_the_variance) of in each class. Let be the mean of the values in  associated with class *Ck*, and let be the [Bessel corrected variance](https://en.wikipedia.org/wiki/Bessel%27s_correction) of the values in  associated with class *Ck*. Suppose we have collected some observation value  Then, the probability distribution of  given a class  can be computed by plugging  into the equation for a [normal distribution](https://en.wikipedia.org/wiki/Normal_distribution) parameterized by and 

That is,



**Types of Naive Bayes Classifier:**

## **Multinomial Naive Bayes:**

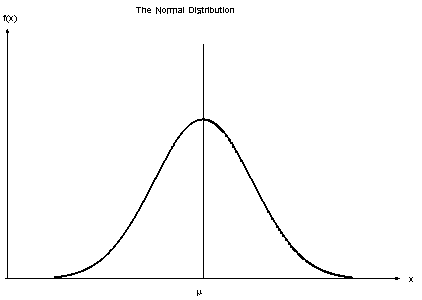
This is mostly used for document classification problem, i.e. whether a document belongs to the category of sports, politics, technology etc. The features/predictors used by the classifier are the frequency of the words present in the document.

## **Bernoulli Naive Bayes:**

This is similar to the multinomial naive Bayes but the predictors are Boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

## **Gaussian Naive Bayes:**

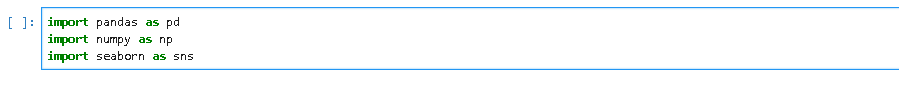
When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a Gaussian distribution.



**IMPLEMENTATION OF DECISION TREE**

**IMPORTING REQUIRED LIBRARIES**

Before we can import our dataset and perform analysis, we need to import a few libraries. The following script is used to import libraries:

****

**LOADING THE DATASET**

Once we import the libraries, the next step is to load the dataset into our application. To do so, we used the “read\_csv()” function of the Pandas library, which reads dataset that is in the CSV format

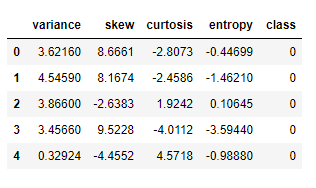
The following script loads the dataset into “banknote\_dataset” data frame:

****

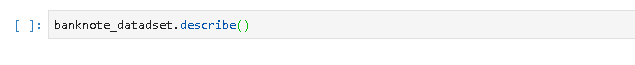
**DATA ANALYSIS**

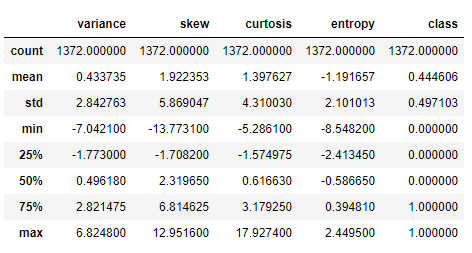
To see how the dataset actually looks, we can use the “head()” function of the Pandas data frame:

The "head()" function returns the first five rows of the dataset as shown below:****

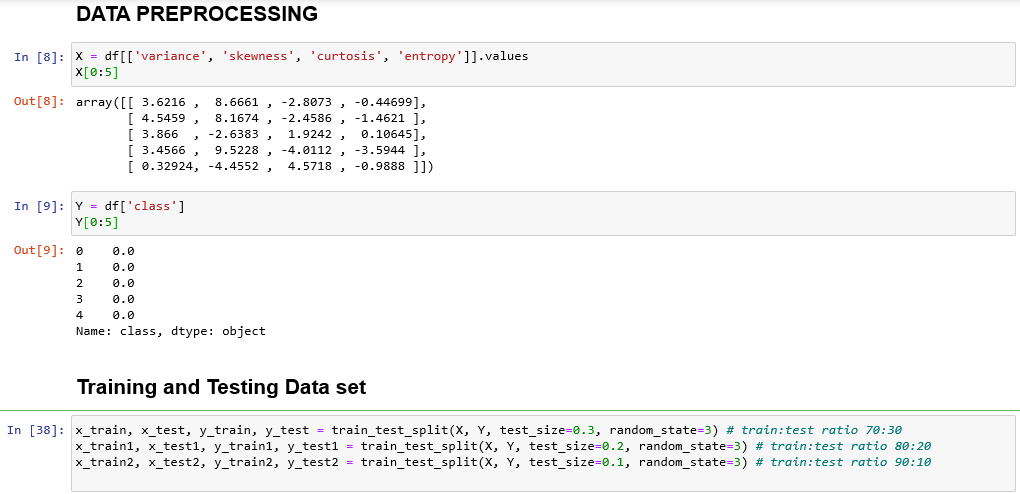


To see the statistical details of the data, the “describe()” function can be used, which returns the mean, count, standard deviation, quartile information and maximum values for each column

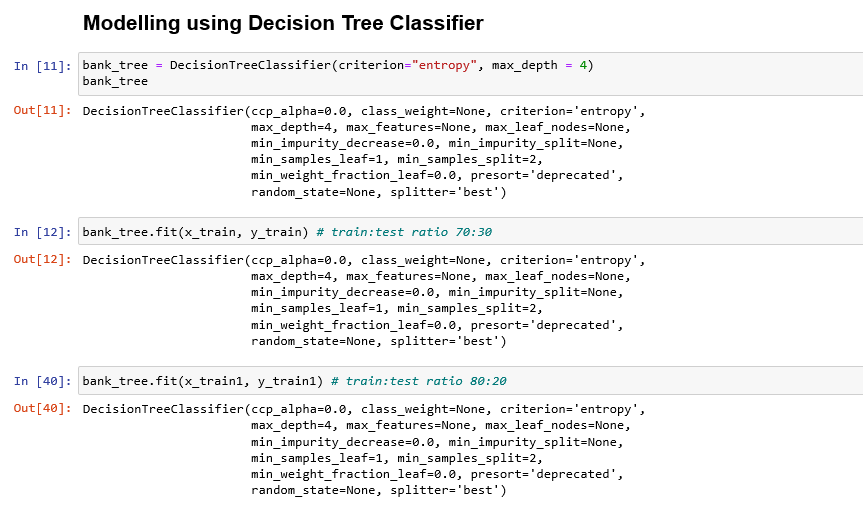
****

****

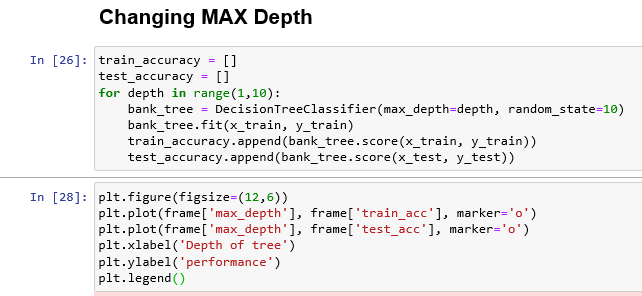
**DATA PREPROCESSING AND TRAINING/TESTING DATASET**

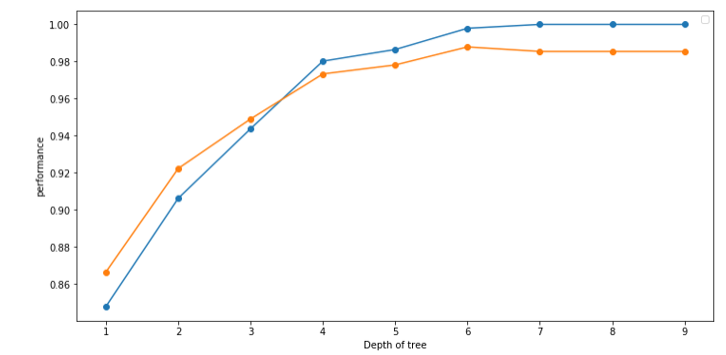
****

**BUILDING MODEL**

****

**SELECTING APPROPRIATE MAXIMUM DEPTH**

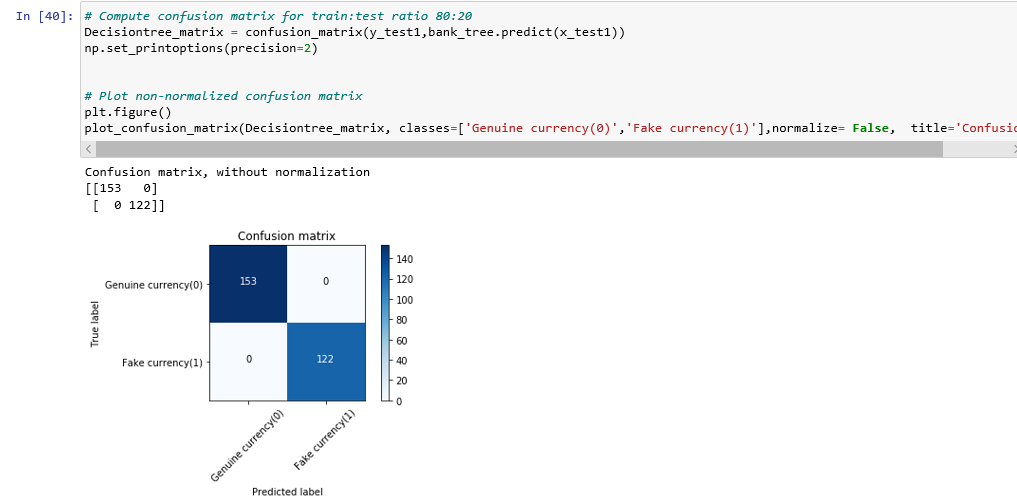
****

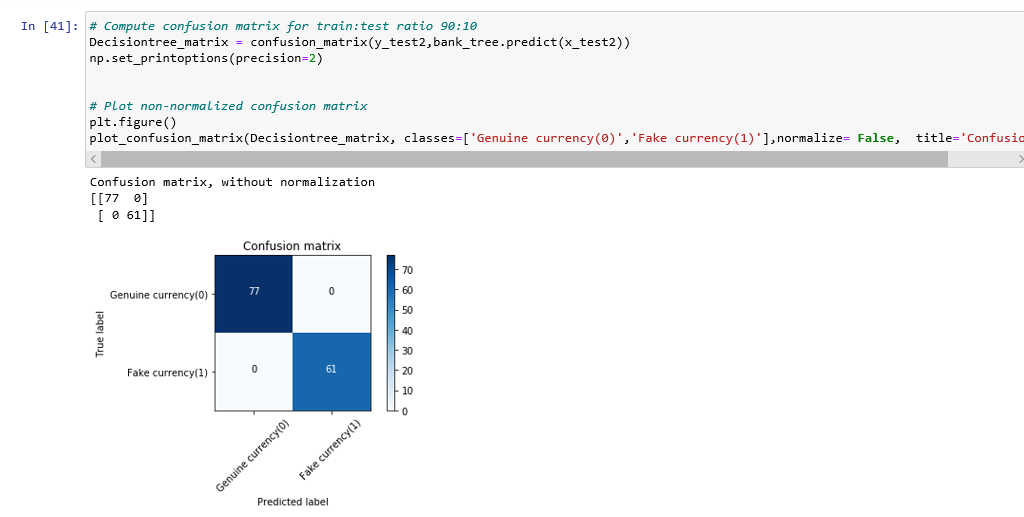
****

**This graph shows the curve of train and test set accuracy when set to a particular depth. This model’s performance becomes high at depth 9, so a 9 is selected as maximum depth of tree**

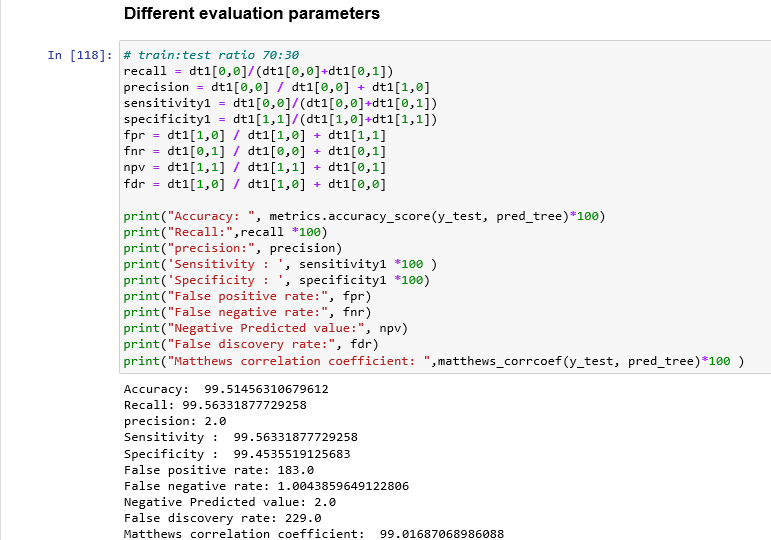
**CONFUSION MATRIX**

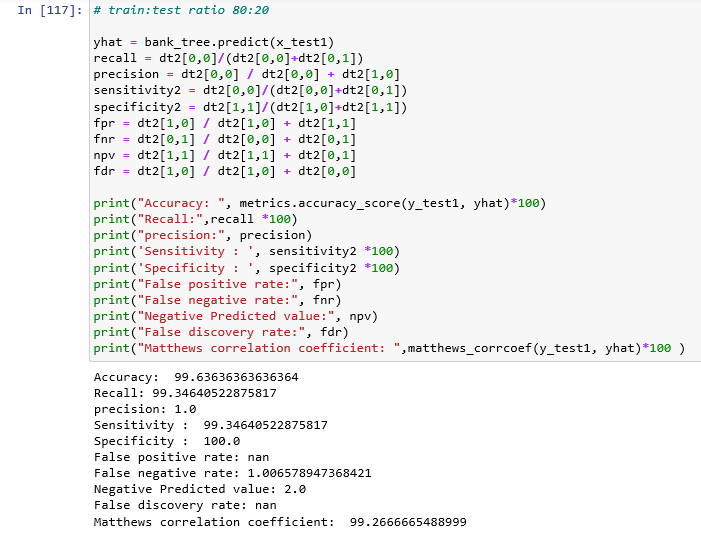
****

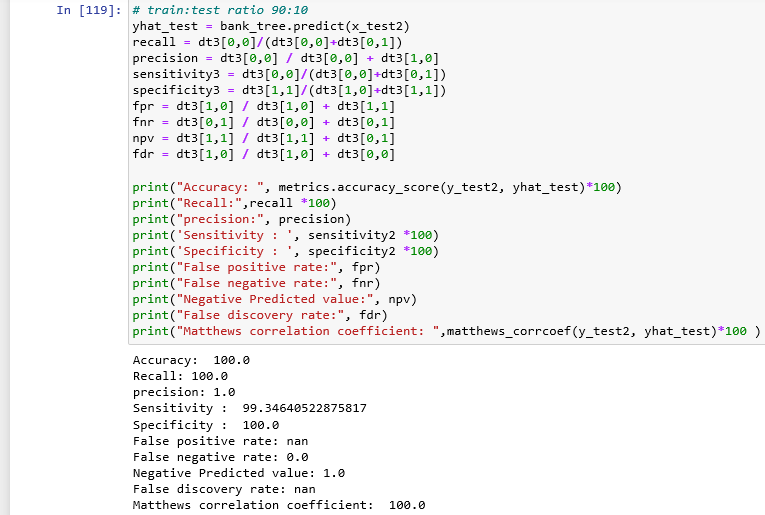
****

****

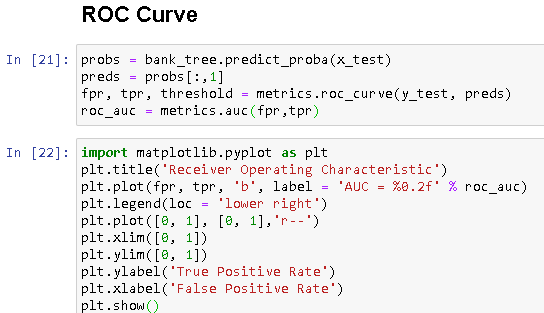
**EVALUATION METRICS**

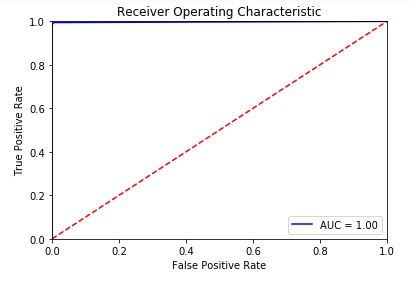






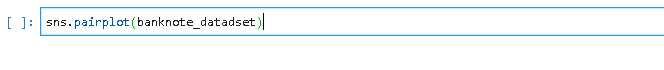
**ROC CURVE**

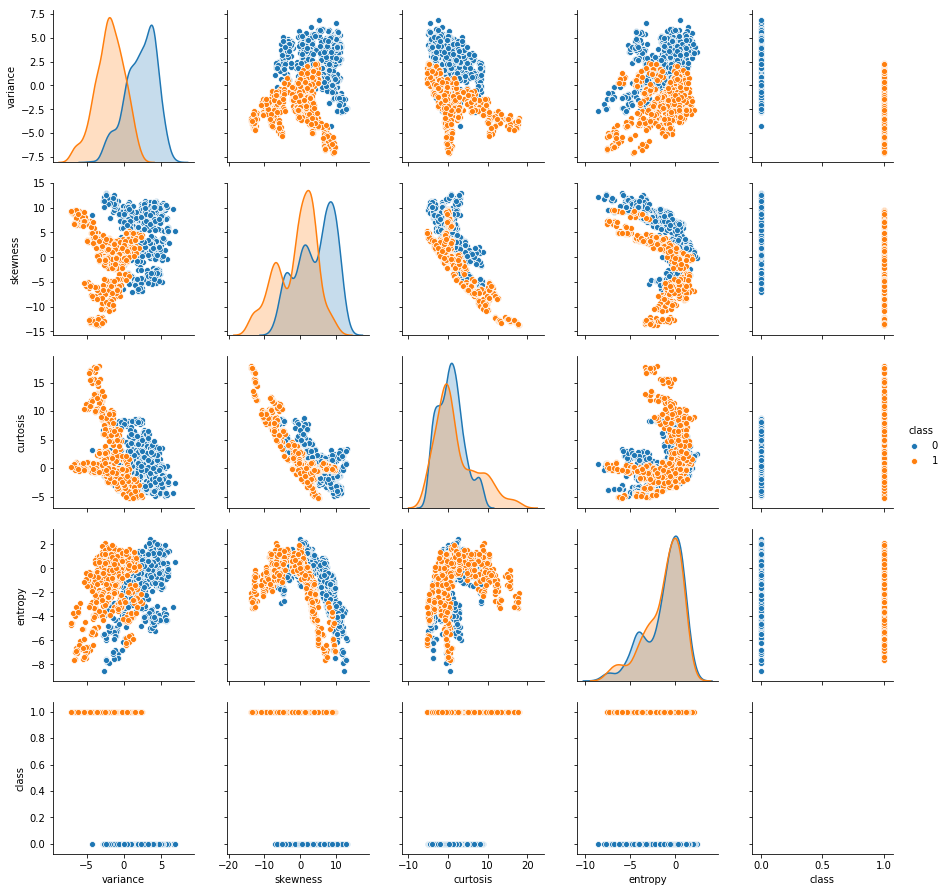
****

****

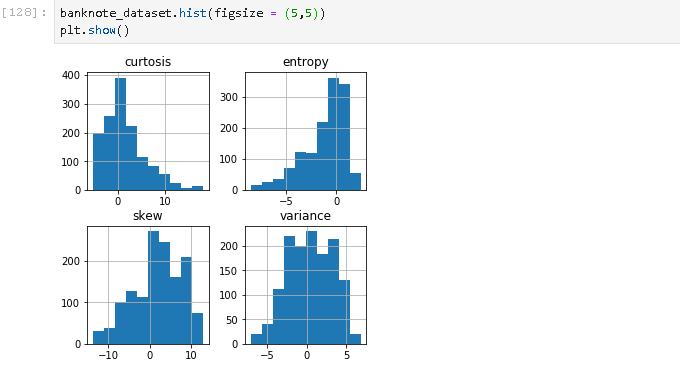
* This is the ROC curve between true positive rate and false positive rate
* Area under curve is 1.00

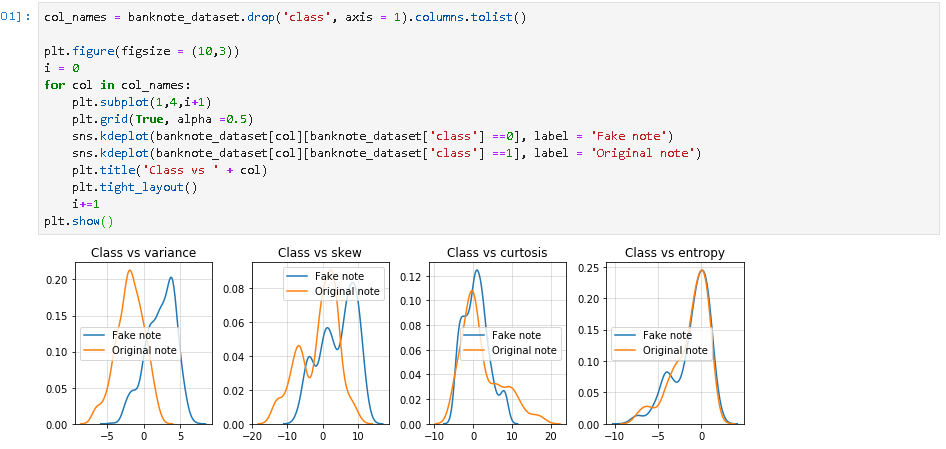
**DATA VISUALIZATION**

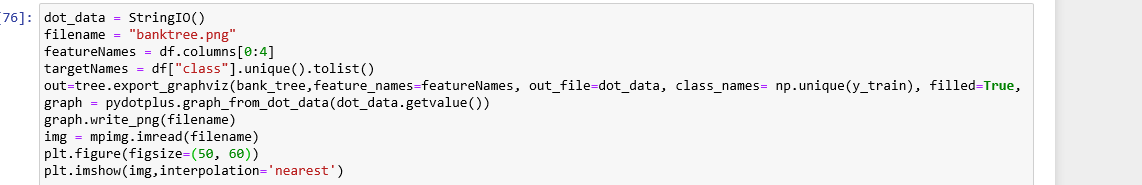


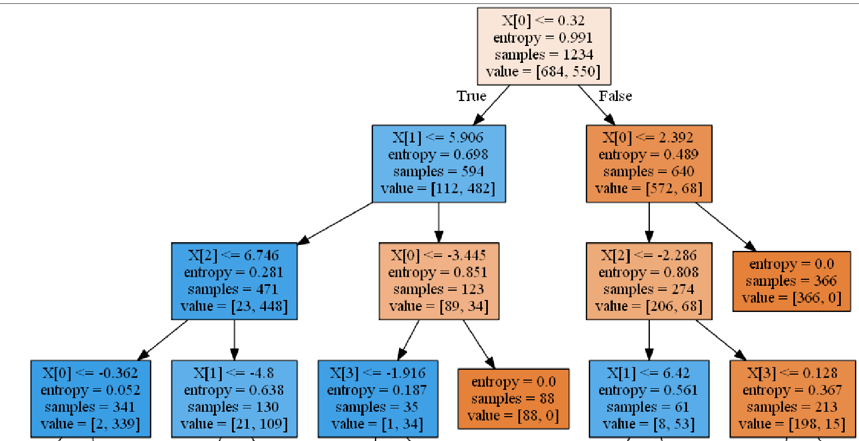
****

It is visible from the output that entropy and variance have a slight linear correlation. Similarly, there is an inverse linear correlation between the curtosis and skew. Finally, we can see that the values for curtosis and entropy are slightly higher for real banknotes, while the values for skew and variance are higher for the fake banknotes

****

****

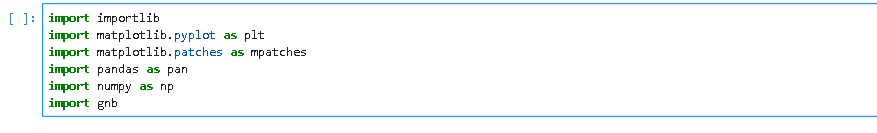
****

****

**IMPLEMENTATION OF NAIVE BAYES**

**IMPORTING REQUIRED LIBRARIES**

Before we can import our dataset and perform analysis, we need to import a few libraries. The following script is used to import libraries:

****

**LOADING THE DATASET**

Once we import the libraries, the next step is to load the dataset into our application. To do so, we used the “read\_csv ()” function of the Pandas library, which reads dataset that is in the CSV format

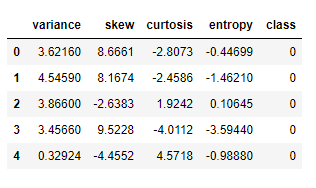
The following script loads the dataset into “banknote\_dataset” data frame:

****

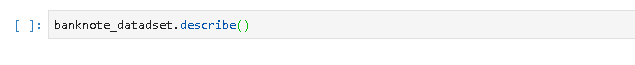
**DATA ANALYSIS**

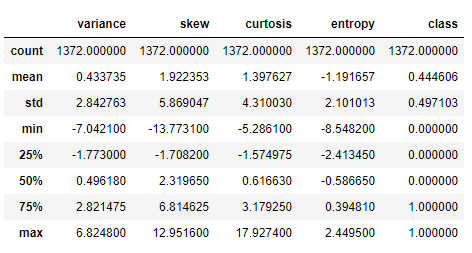
To see how the dataset actually looks, we can use the “head ()” function of the Pandas data frame: The "head ()" function returns the first five rows of the dataset as shown below:

****

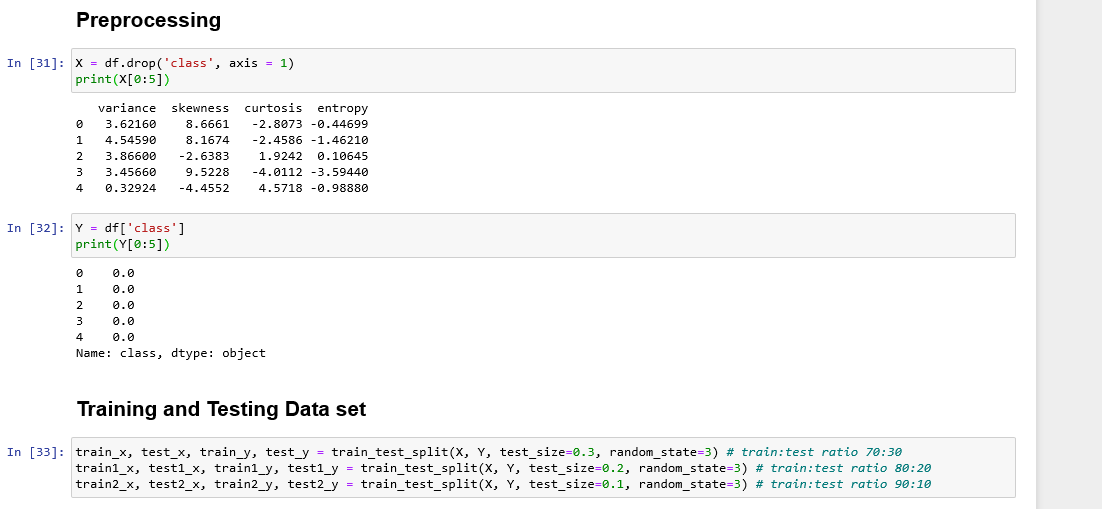


To see the statistical details of the data, the “describe ()” function can be used, which returns the mean, count, standard deviation, quartile information and maximum values for each column

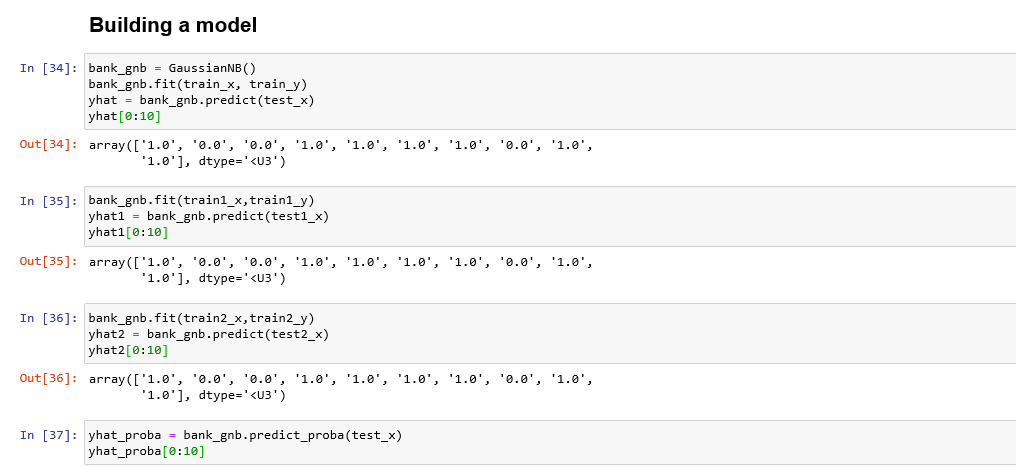
****

****

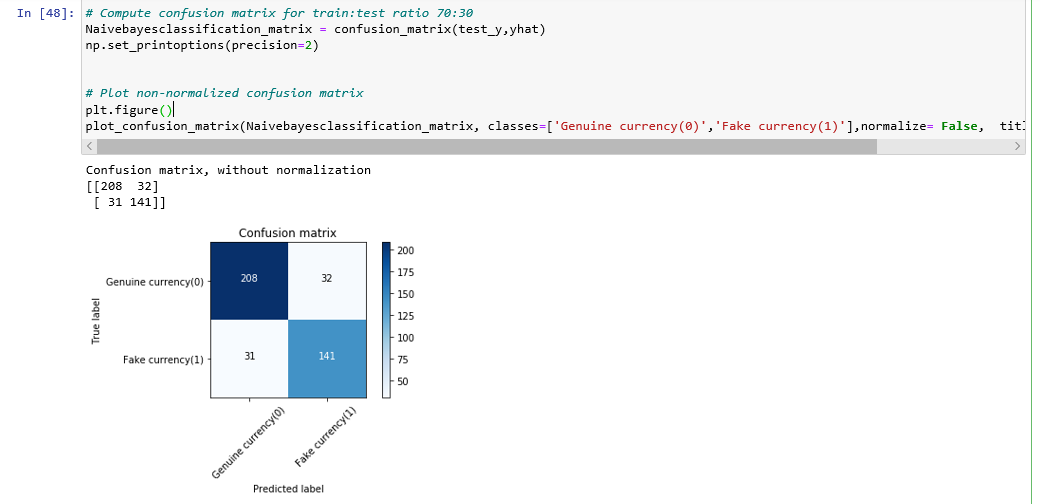
**DATA PREPROCESSING AND TRAINING/TESTING DATASET**

****

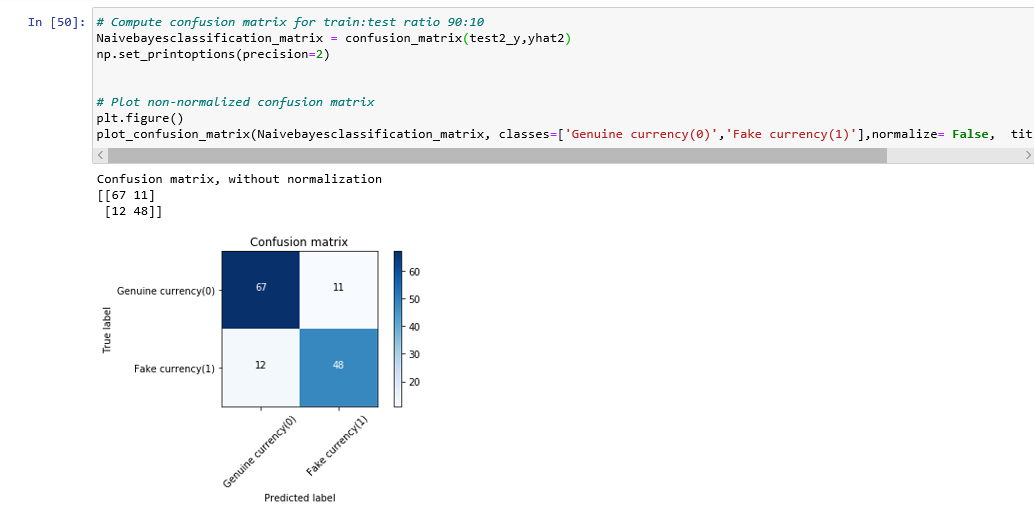
**BUILDING MODEL**

****

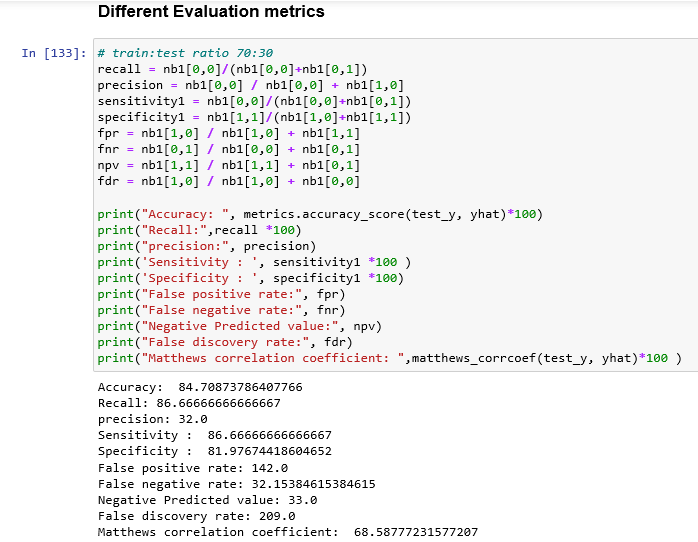
**CONFUSION MATRIX**

****

****

****

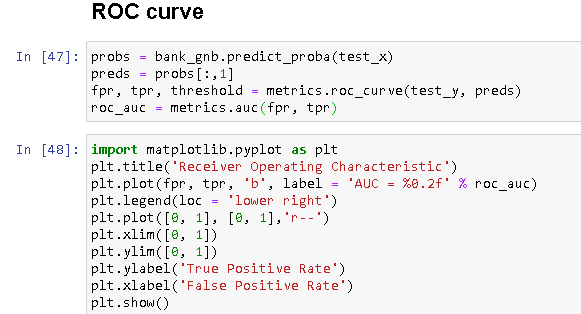
**EVALUATION METRICS**

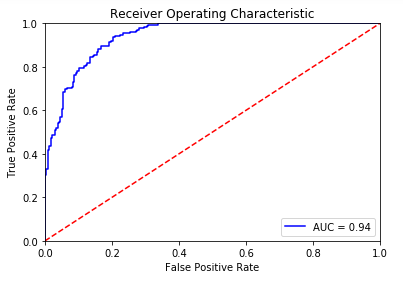
****

****

****

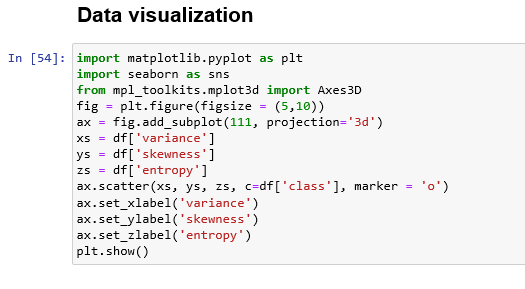
**ROC CURVE**

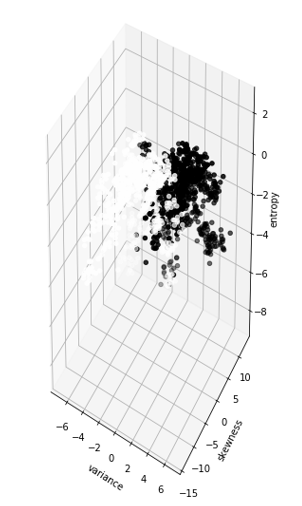
****

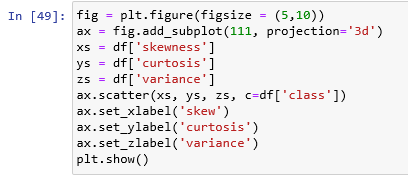
****

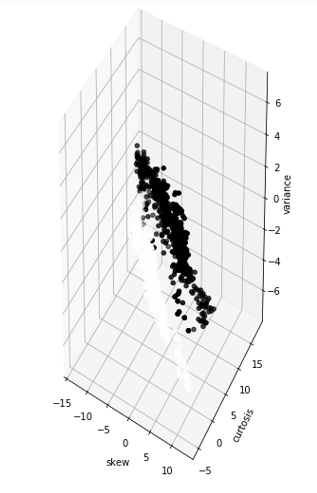
* This is the ROC curve between true positive rate and false positive rate
* Area under curve is 0.94

**DATA VISUALIZATION**

****

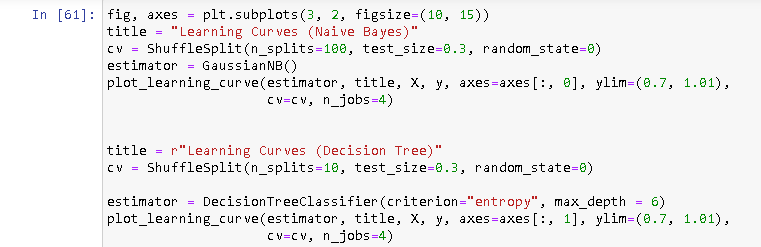
****

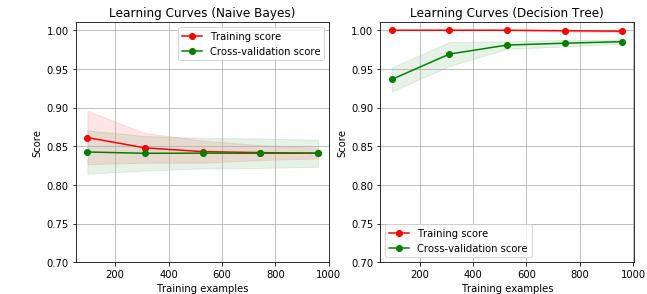
****

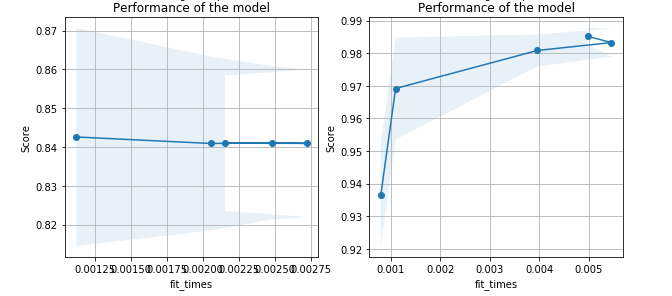
****

**LEARNING CURVE AND PERFORMANCE OF MODEL**

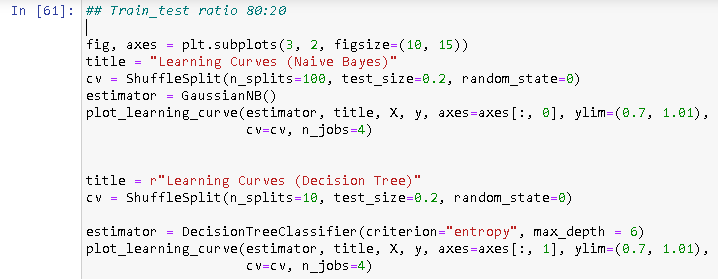
**Train-test ratio 70:30**

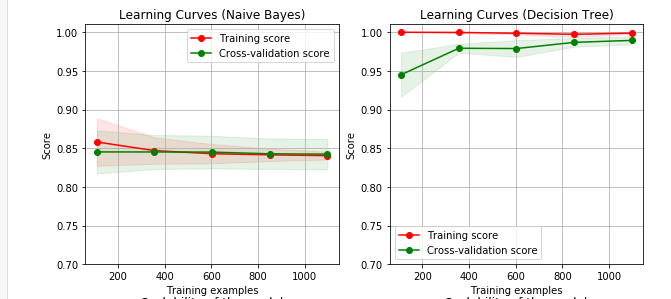
****

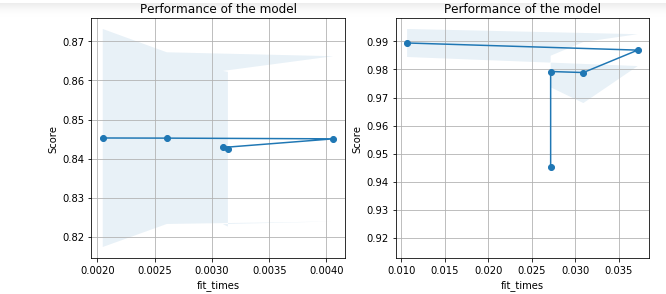
****

****

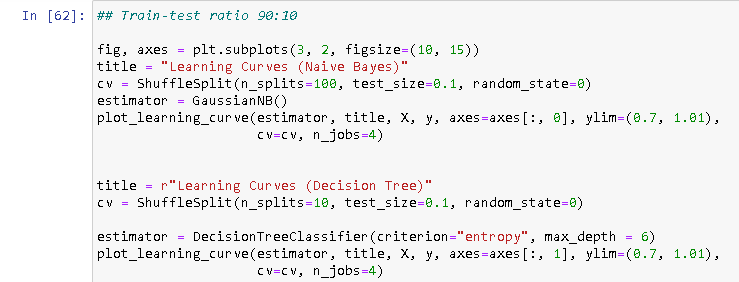
**Train-test ratio 80:20**

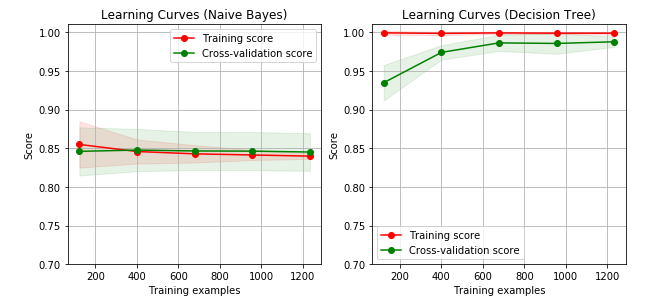
****

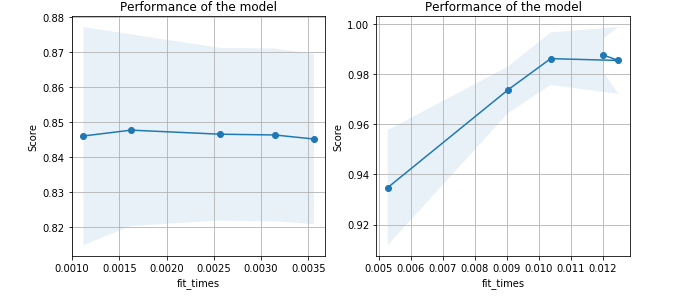
****

****

**Train-test ratio 90:10**

****

****

****

**RESULT**

**Evaluation metrics for decision tree**

|  |  |  |  |
| --- | --- | --- | --- |
| **Decision Tree**  **Measures** | **Train test ratio 1** | **Train test ratio 2** | **Train test ratio 3** |
| **Specificity** | 99.4535519125683 | 100.0 | 100.0 |
| **Sensitivity** | 99.56331877729258 | 99.34640522875817 | 99.34640522875817 |
| **Accuracy** | 99.51456310679612 | 99.63636363636364 | 100.0 |
| **Precision** | 2.0 | 1.0 | 1.0 |
| **FPR** | 183.0 | nan | nan |
| **FNR** | 1.0043859649122806 | 1.006578947368421 | 0.0 |
| **NPV** | 2.0 | 2.0 | 1.0 |
| **FDR** | 229.0 | nan | nan |
| **MCC** | 99.01687068986088 | 99.2666665488999 | 100.0 |

From the observation, we can conclude that Accuracy of the model is highest and constant throughout our testing of dataset on all train-test ratio

**Evaluation metrics for Naïve Bayes**

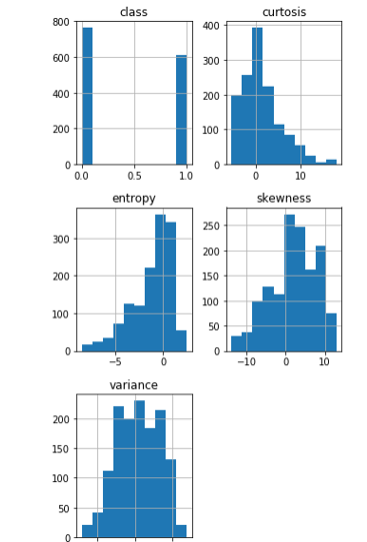
|  |  |  |  |
| --- | --- | --- | --- |
| **Naïve Bayes**  **Measures** | **Train test ratio 1** | **Train test ratio 2** | **Train test ratio 3** |
| **Specificity** | 81.97674418604652 | 83.17757009345794 | 80.0 |
| **Sensitivity** | 86.66666666666667 | 88.09523809523809 | 85.8974358974359 |
| **Accuracy** | 84.70873786407766 | 86.18181818181819 | 83.33333333333334 |
| **Precision** | 32.0 | 19.0 | 13.0 |
| **FPR** | 142.0 | 90.0 | 49.0 |
| **FNR** | 32.15384615384615 | 20.135135135135137 | 11.164179104477611 |
| **NPV** | 33.0 | 21.0 | 12.0 |
| **FDR** | 209.0 | 149.0 | 68.0 |
| **MCC** | 68.58777231577207 | 71.04002527659058 | 66.03161074375328 |

From the observation, we can conclude that Accuracy of the model is lowest and constant throughout our testing of dataset on all train-test ratio

**DATA VISUALIZATION**

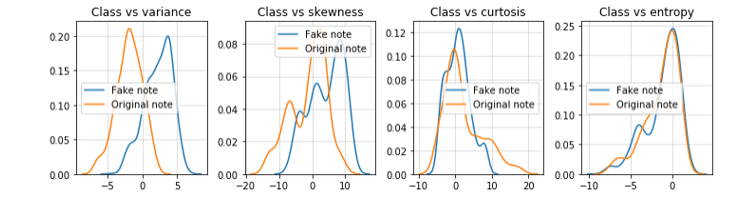
* **Decision Tree Classification**

**Histogram**

****

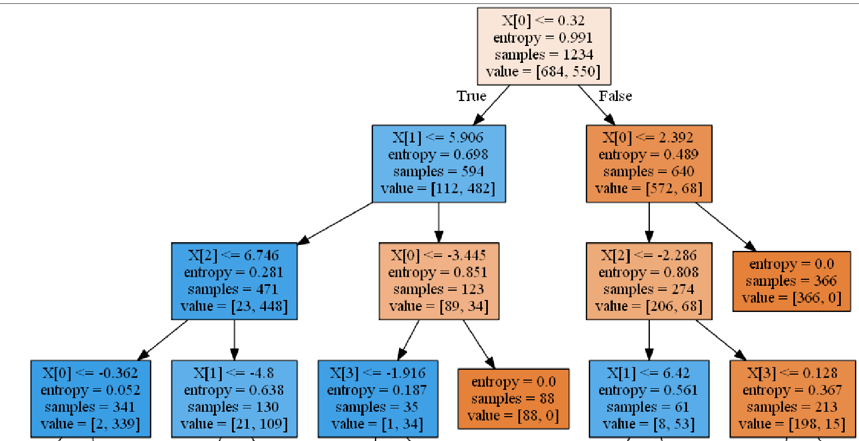
This is a histogram of every feature of banknote datasets

**Grid plot**

****

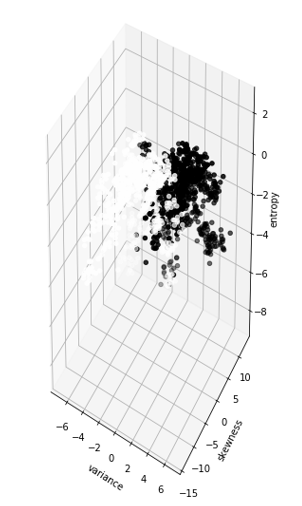
In the above plot comparison has been made between every feature of dataset and a grid plot has been made of fake note (denoted by blue) and original note (denoted by orange).

**Decision Tree**

****

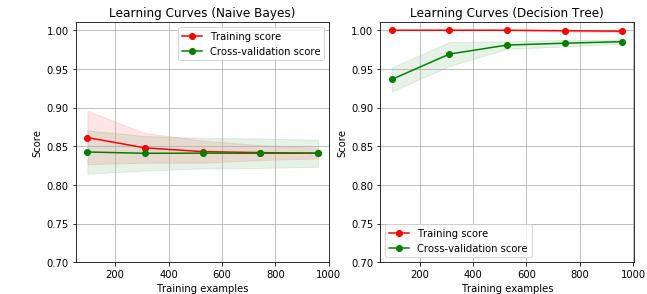
* This is a decision tree model of our dataset. In this tree in each node criterion “entropy”, samples and value of each variable is calculated
* Criterion is a function, which measures the purity of nodes. Supported criteria are “Gini” for the Gini impurity and “entropy” for the information gain.
* Lower the “entropy”, higher the homogeneity of nodes
* **Naïve Bayes classification**

**3-D Scatter plot between variance, skewness and entropy**

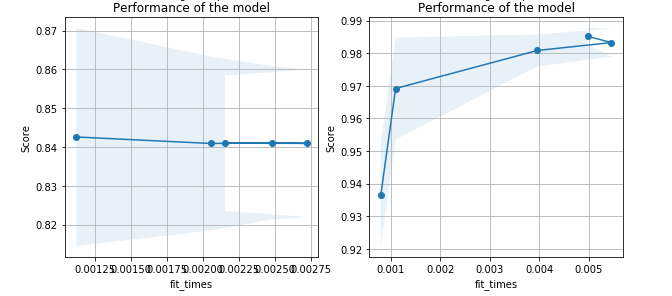
****

* This is a 3-D scatter plot between variance, skewness and entropy. This scatter plot is plotted based on target variable, which is “class”.
* Black points in this plot denotes genuine notes and White points in this plot denotes Fake notes

**Learning curve and Performance of both models**

****

This is the learning curve of Decision Tree classification and naïve Bayes classification. From both the graphs we can predict the accuracy of train data and test data on different training data

****

This is the performance of both the models. In this graph accuracy score of training data is plotted against the Fit\_times of training data

Fit\_time: The time for fitting the estimator on the train set for each CV split.

**REFERENCE**

* machinelearningmastery.com
* Geek for Geeks.com
* TowardsDatascience.com
* wikipedia.org/wiki/Decision\_tree\_learning
* towardsdatascience.com/decision-tree-classification
* scikitlearn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier