[[1]](#footnote-2)

F1-scores and performance of three different clustering algorithms on Iris dataset

Krishna Mathur

*Abstract*—This report applies machine learning algorithm to classify the data in the Iris dataset. Classification is done using clustering algorithms and further the results of these algorithms are compared on the basis of F1- score and accuracy.

*Index Terms*— Accuracy, Clustering algorithms, DBSCAN, F-1 score, Hierarchical clustering, Iris dataset, KNN.

# Introduction

M

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. Machine learning is a type of AI that allows computer programs to adjust when exposed to new data, in effect, “learning” without being explicitly programmed. Machine learning is similar to data mining in which databases are examined by humans to produce new information and insight. Machine learning algorithms are often categorized as supervised or unsupervised.

1. Supervised machine learning – These algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
2. Unsupervised machine learning – These algorithms are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

# Clustering Algorithms

It is basically a type of [unsupervised learning method](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) . An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.  
Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

For Example - The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.

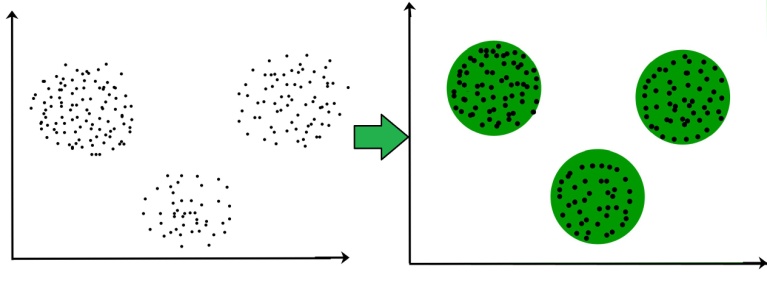


Fig.1

## K-means clustering algorithm –

  It is the simplest unsupervised learning algorithm that solves clustering problem. K-means algorithm partition n observations into k clusters where each observation belongs to the cluster with the nearest mean serving as a prototype of the cluster. The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood calculating the distance between points on a graph. There are other ways of calculating distance, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

The KNN Algorithm -

1. Load the data.
2. Initialize K to your chosen number of neighbors.
3. For each example in the data.
   1. Calculate the distance between the query example and the current example from the data.
   2. Add the distance and the index of the example to an ordered collection.
4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
5. Pick the first K entries from the sorted collection.
6. Get the labels of the selected K entries.
7. If regression, return the mean of the K labels.
8. If classification, return the mode of the K labels.

To select the K that’s right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm’s ability to accurately make predictions when it’s given data it hasn’t seen before. Sometimes it’s quite possible that, we might be choosing a initial K center points in such a way that the algorithm gives a false positive model. This can be avoided using K++ means. To solve the random initialization of K we use WCSS (Within Cluster Sum of Squares). In WCSS we calculate the sum of squares of the distance of each data point in cluster 1 from their center point C1. Let’s say there are 3 points in cluster 1 (c1p1, c1p2, c1p3).

[dist(C1, c1p1) ]² + [dist(C1, c1p2)]² + [dist(C1, c1p3)]²

This is cluster 1 sum of squares. Similarly we do the same for C2 & C3. Now, we add the sum of all 3 ‘clusters sum of squares’ to get WCSS. WCSS always decreases with the increase in the number of clusters. However, it should be noted that, the rate of drop in WCSS starts to drop as we increase the number of clusters. This would be our hint. We need to stop at the number of clusters from where the rate of drop in WCSS doesn’t drop substantially (in other words, the rate of drop is very less). This is sometimes also termed as Elbow method.

## Hierarchical clustering

As the name implies, hierarchical clustering is an algorithm for constructing cluster hierarchies. The algorithm starts with all the data assigned to one of their own clusters and then joins the two most recent clusters to the same cluster. Finally, when there is only one cluster left, the algorithm ends. Hierarchical Clustering uses the distance based approach between the neighbor data-points for clustering. Each data point is linked to its nearest neighbors.  There are two ways you can do Hierarchical clustering Agglomerative that is bottom-up approach clustering and Divisive uses top-down approaches for clustering. Hierarchical clustering starts by treating each observation as a separate cluster. Then, it repeatedly executes the following two steps:

1- Identify the two clusters that are closest together,

2 - Merge the two most similar clusters. This continues until all the clusters are merged together. This is illustrated in the diagrams below.

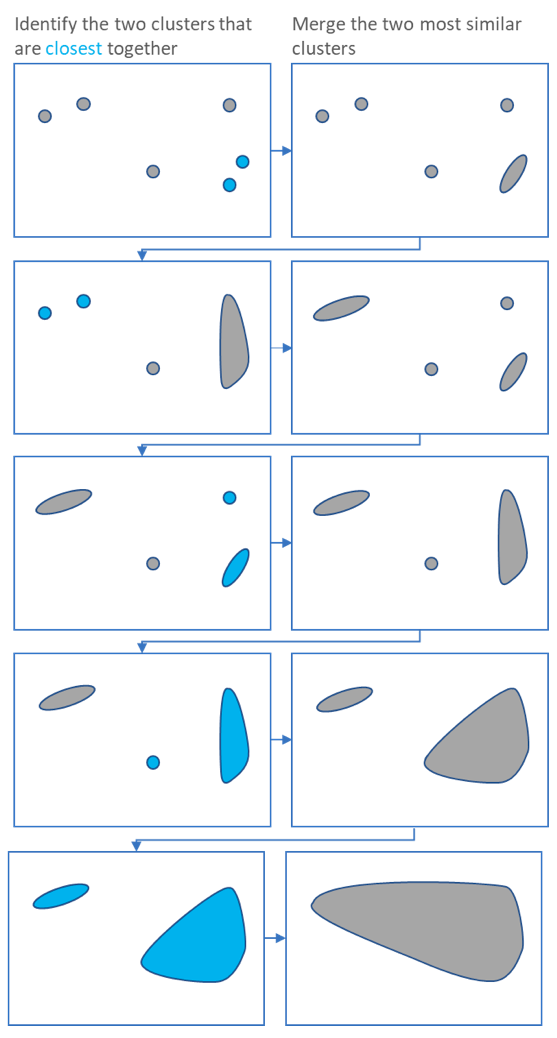


Fig.2

The main output of Hierarchical Clustering is a [dendrogram](https://www.displayr.com/what-is-dendrogram/), which shows the hierarchical relationship between the clusters:

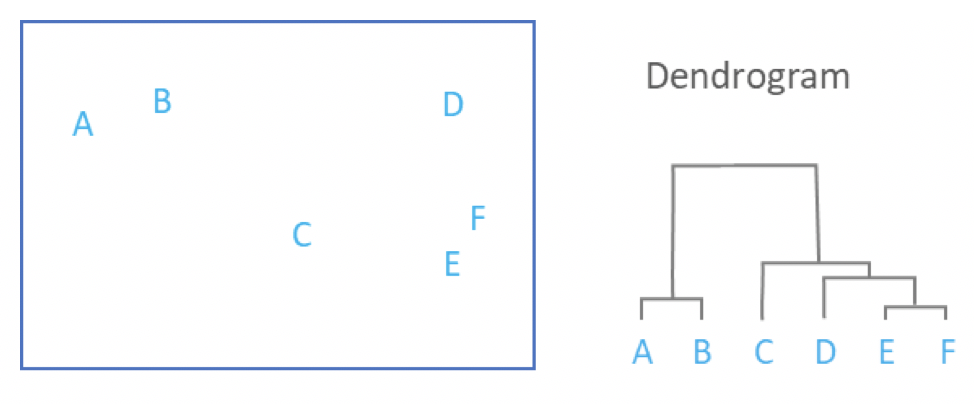


Fig.3

Based on the Fig.2 above the Hierarchical clustering algorithm steps are -

1. In the initial step, we calculate the proximity of individual points and consider all the six data points as individual clusters.
2. In step two, similar clusters are merged together and formed as a single cluster.
3. We again calculate the proximity of new clusters and merge the similar clusters to form new clusters. We repeat this step until only one cluster is left.

The Hierarchical clustering Technique can be visualized using a Dendrogram (Fig.3). A Dendrogram is a tree-like diagram that records the sequences of merges or splits.

## DBSCAN

DBSCAN stands for Density Based Spatial Clustering of Applications with Noise. It is a popular unsupervised learning method utilized in model building and machine learning algorithms. DBSCAN is a clustering method that is used in machine learning to separate clusters of high density from clusters of low density. Given that DBSCAN is a density based clustering algorithm, it does a great job of seeking areas in the data that have a high density of observations, versus areas of the data that are not very dense with observations. DBSCAN can sort data into clusters of varying shapes as well, another strong advantage. DBSCAN works as such:

* Divides the dataset into n dimensions
* For each point in the dataset, DBSCAN forms an n dimensional shape around that data point, and then counts how many data points fall within that shape.
* DBSCAN counts this shape as a cluster. DBSCAN iteratively expands the cluster, by going through each individual point within the cluster, and counting the number of other data points nearby. Take the graphic below for an example:

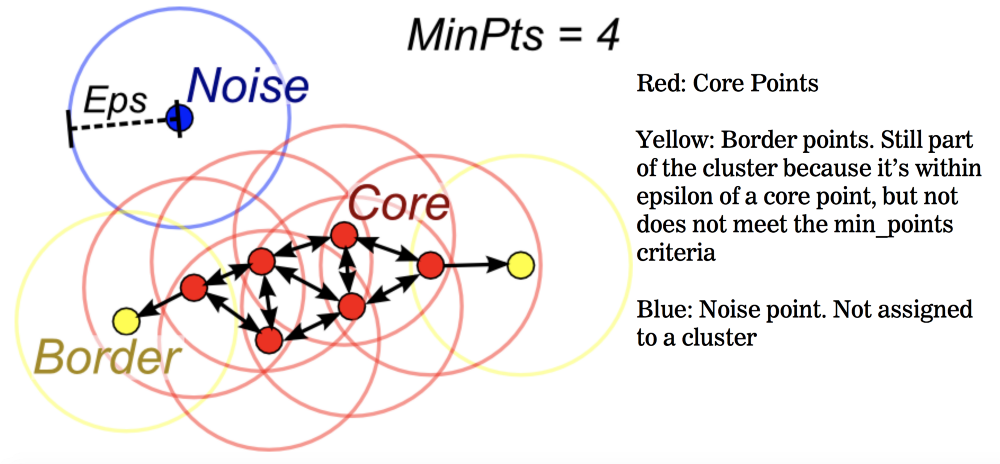


Fig.4

DBSCAN algorithm requires two parameters –

1. Eps : It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to ‘eps’ then they are considered as neighbors. If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters will merge and majority of the data points will be in the same clusters. One way to find the eps value is based on the k-distance graph.
2. MinPts: Minimum number of neighbors (data points) within eps radius. Larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1. The minimum value of MinPts must be chosen at least 3.

DBSCAN Algorithm –

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPts neighbors.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.  
   A point a and b are said to be density connected if there exist a point c which has a sufficient number of points in its neighbors and both the points a and b are within the eps distance. This is a chaining process. So, if b is neighbor of c, c is neighbor of d, dis neighbor of e, which in turn is neighbor of a implies that b is neighbor of a.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

# DATASET

In this report we have taken the iris dataset from Scikit learn (machine learning library) in which iris dataset is already inbuilt.

Dataset Information: The dataset contain 150 sample data in it. The dataset has three classes of data that are Setosa, Versicolor and Virginica each having 50 sample data.

Numbers of attributes in the datasets are:

4 numeric attributes, predictive attribute (class of iris plant) and the class attribute information.

1. Sepal length in cm
2. Sepal width in cm
3. Petal length in cm
4. Petal width in cm

*A. Iris Setosa*



Fig.5

## B. Iris Versicolor



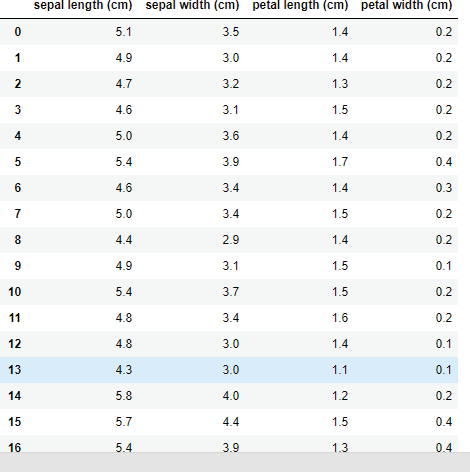
Fig.6

## C. Iris Virginica



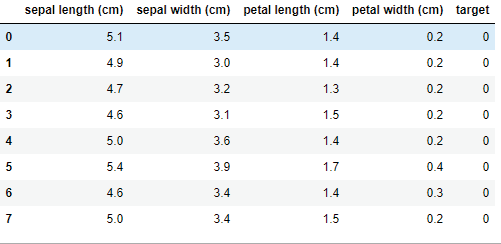
Fig.7

TABLE I. ATTRIBUTES OF THE DATASET THAT IS SEPAL LENGTH, SEPAL WIDTH, PETAL LENGTH AND PETAL WIDTH



Above table shows the attributes of the dataset that is Sepal length, Sepal width, Petal length and Petal width. A dataset contain value of all attribute. As the dataset is already preprocessed so we don’t need to do data preprocessing. Now we decide target variable that is 0,1,2

TABLE II.



In this table we have included the target variable. Here the target variable 0,1,2 corresponds to Iris Setosa, Iris Versicolor, Iris Virginica respectively.

# Implementation

In order to implement the clustering algorithm on the Iris dataset we need to follow these steps –

1. Load the Iris dataset
2. Create object of that clustering algorithm.
3. Fit the data into the object.
4. Evaluate the model.

We used Google colaboratory to build the model. Initially we load the iris dataset from Scikit learn library which is common step for all the algorithms and then store the data in the pandas dataframe.

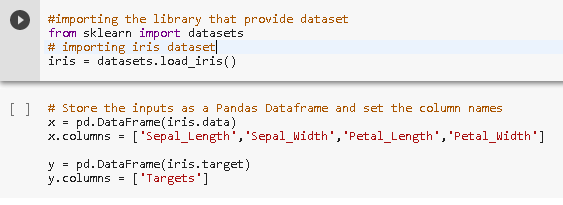


Fig.8

Visualizing the data of iris dataset so that we can see the clusters after performing the algorithm on the dataset

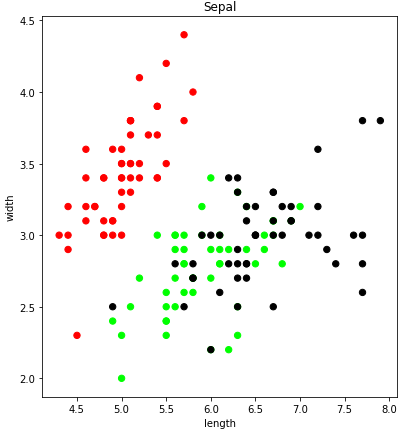


Fig.9

In this plot we have Sepal length on the x-axis and Sepal width on the y-axis and the colors Red, Green and black belongs to 3 different types of iris i.e., Versicolor, Setosa, Virginica.

In next plot we have petal length on x-axis and petal width on y-axis. In this plot we can see that we can visualize the clusters so we will be comparing the plot of petal length-petal width with our model predicted clusters.

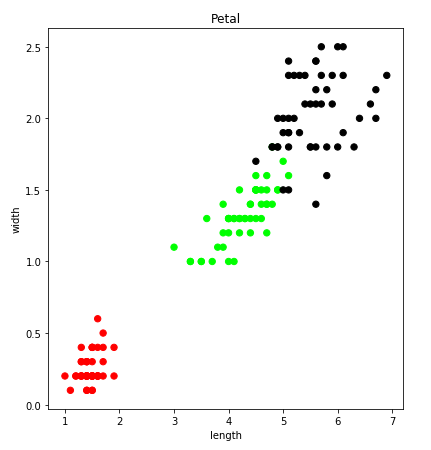


Fig.10

## KNN model

Creating the the object and fitting data in the object.

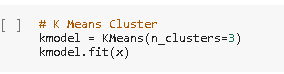


Fig.11

Plotting the graph predicted by KNN model –

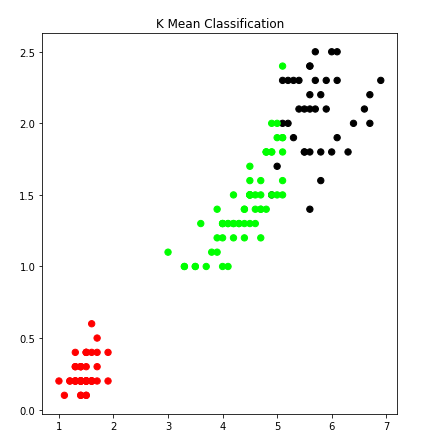


Fig.12

## Hierarchical model –

We will first create a dendogram to see the clusters and then fir the number of cluster in model object.

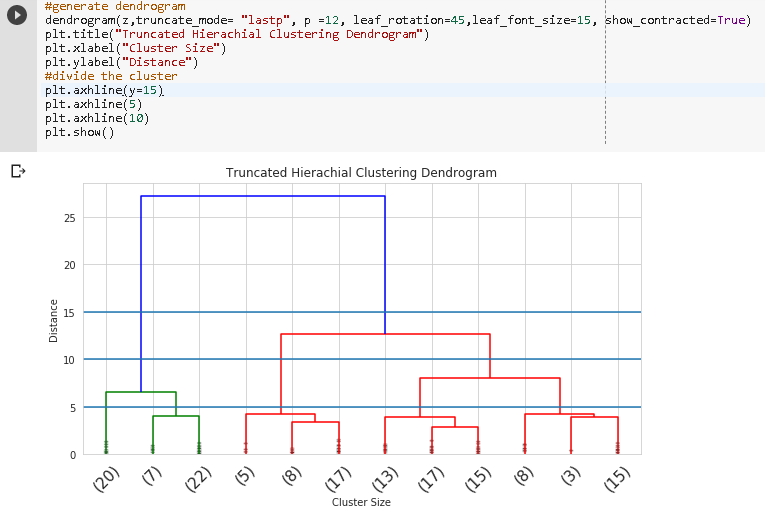
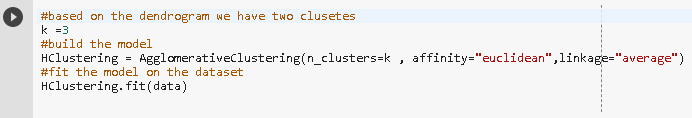


Fig.13

Now we will fir the model with 3 clusters.



Plotting the graph predicted by Hierarchical model –

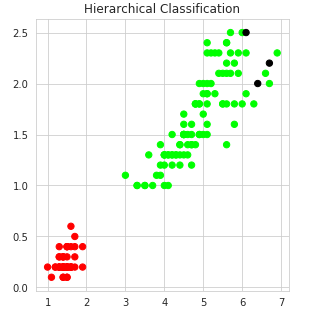


Fig.14

## DBSCAN model-

Creating the object and fitting the data.



Plotting the graph predicted by DBSCAN –

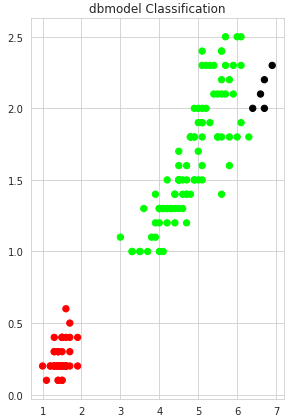


Fig.15

# Evaluation

We evaluate the model to check wheather the model is properly working or not. And the other purpose to evaluate the model is to modify the model and to get the better result. The metrics that you choose to evaluate your machine learning algorithms are very important.

Choice of metrics influences how the performance of machine learning algorithms is measured and compared. They influence how you weight the importance of different characteristics in the results and your ultimate choice of which algorithm to choose.

Classification problems are perhaps the most common type of machine learning problem and as such there are a myriad of metrics that can be used to evaluate predictions for these problems. In this paper we have used the classification Accuracy metrics and F-1 Scores.

Classification accuracy is the number of correct predictions made as a ratio of all predictions made.

This is the most common evaluation metric for classification problems, it is also the most misused. It is really only suitable when there are an equal number of observations in each class (which is rarely the case) and that all predictions and prediction errors are equally important, which is often not the case.

F1 score combines precision and recall relative to a specific positive class -The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

Confusion Matrix is a performance measurement for machine learning classification. It is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values.

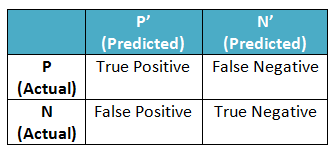


Fig.16

True Positive(TP) – You predicted Positive and it’s True.

True Negative(TN) – You predicted Negative and it’s True.

False Positive(FP) – You predicted Positive and it’s False.

False Negative(FN) – You predicted Negative and it’s False.

Accuracy is calculated –

Accuracy = TP+TN/(TP+TN+FP+FN)

Recall - Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (small number of FN).

Recall = TP/(TP+FN)

Precision - To get the value of precision we divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (small number of FP).

Precision = TP/(TP+FP)

F- measure - Since we have two measures (Precision and Recall) it helps to have a measurement that represents both of them. We calculate an F-measure which uses Harmonic Mean in place of Arithmetic Mean as it punishes the extreme values more.  
The F-Measure will always be nearer to the smaller value of Precision or Recall.

F-measure = 2\*(Recall \* Precision)/(Recall + Precision)

Plotting the graph to compare Accuracy of these models-

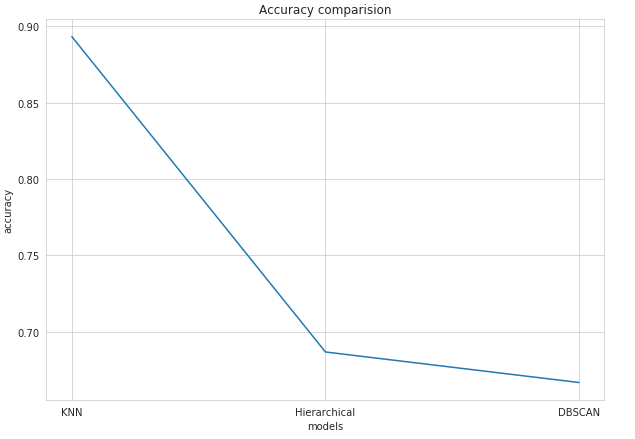


Fig.17

Plotting the Graph to compare F1-Score –

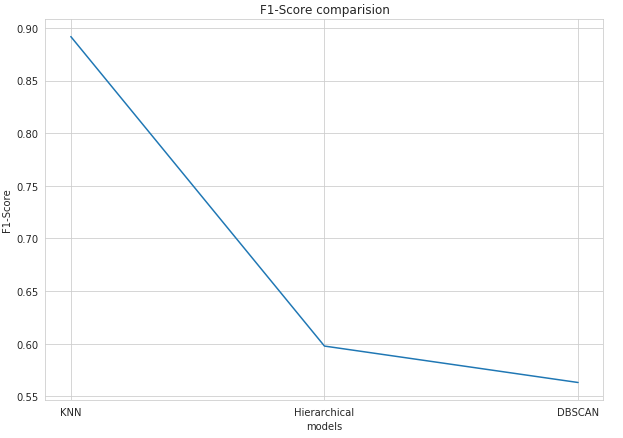


Fig.18

# CONCLUSION

I tried to build the models that are able to recognize the iris species accurately on the basis of 3 classes, but some sample provides the misclassified result. All the models are able to Classify 2 Classes with higher percentage of accuracy. But the accuracy of KNN model is much better overall in terms of classifying all 3 classes and same in the case of F1-Score.

1. [↑](#footnote-ref-2)