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

Jyothsna Sai

AITS (http:\\www.ai-techsystems .com)

***Abstract*—This report will give information on performance of clusterinh algorithms on Irise dataset. Classification is done using clustering algorithms and further the results of these algorithms are compared on the basis of F1- score and accuracy.**

***KEYWORDS* :Clustering algorithms, DBSCAN, F-1 score, Agglomerative Clustering, Iris dataset, K-mean**

INTRODUCTION

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.

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..

CLUSTERING ALGORITHMS

Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group. In theory, data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features. Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields.

In Data Science, we can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into when we apply a clustering algorithm. Today, we're going to look at 5 popular clustering algorithms that data scientists need to know and their pros and cons!
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.

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.
    a. Calculate the distance between the query example and the current example from the data.

b. 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).

2.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:

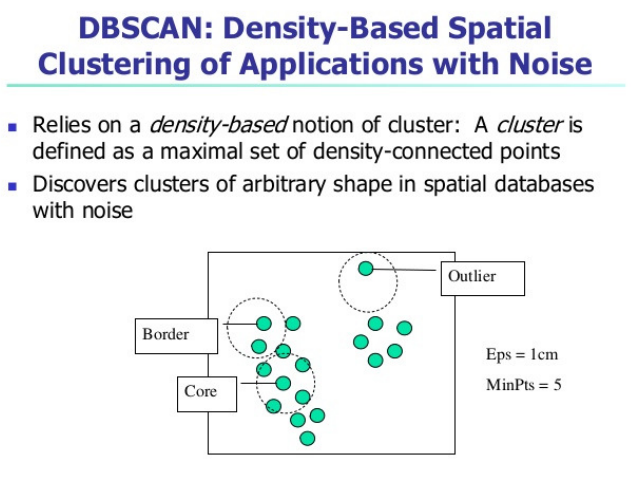


Fig.4

DBSCAN algorithm requires two parameters –

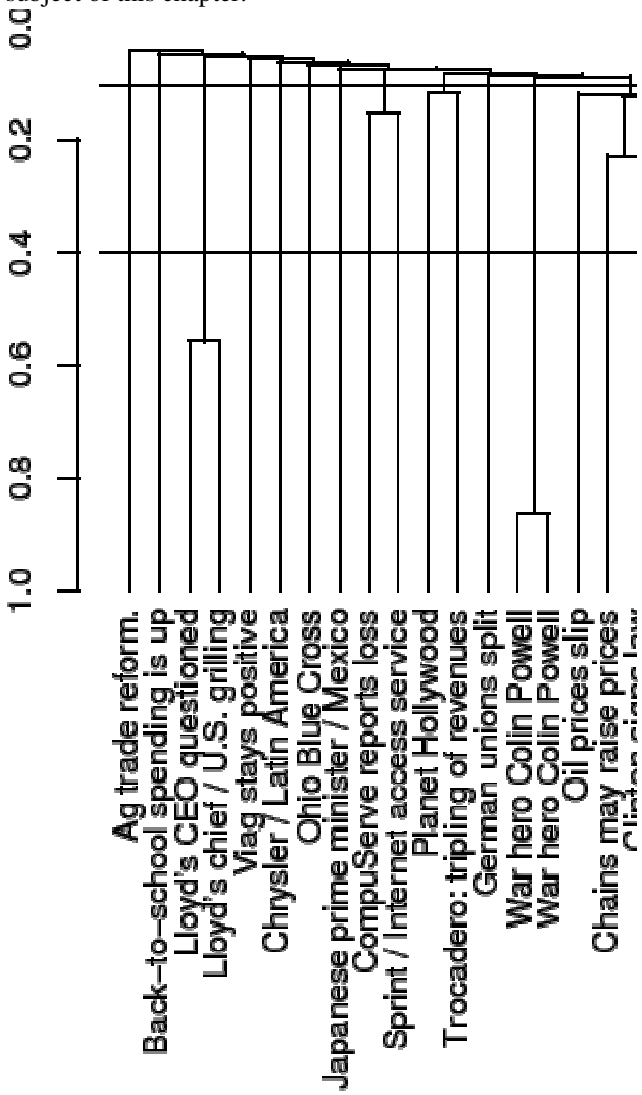
1. Eps : It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to 'eps' then they are considered as neighbors. If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters 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.

3.HIERARCHICAL AGGLOMERATIVE clustering

Algorithms are either top-down or bottom-up. Bottom-up algorithms treat each document as a singleton cluster at the outset and then successively merge (or *agglomerate*) pairs of clusters until all clusters have been merged into a single cluster that contains all documents. Bottom-up hierarchical clustering is therefore called *hierarchical agglomerative clustering* or *HAC* . Top-down clustering requires a method for splitting a cluster. It proceeds by splitting clusters recursively until individual documents are reached. See Section 17.6 . HAC is more frequently used in IR than top-down clustering and is the main subject of this chapter.



DATASET

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

Abourt dataset: 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

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	1
1	2	4.9	3.0	1.4	0.2	1
2	3	4.7	3.2	1.3	0.2	1
3	4	4.6	3.1	1.5	0.2	1
4	5	5.0	3.6	1.4	0.2	1

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

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.

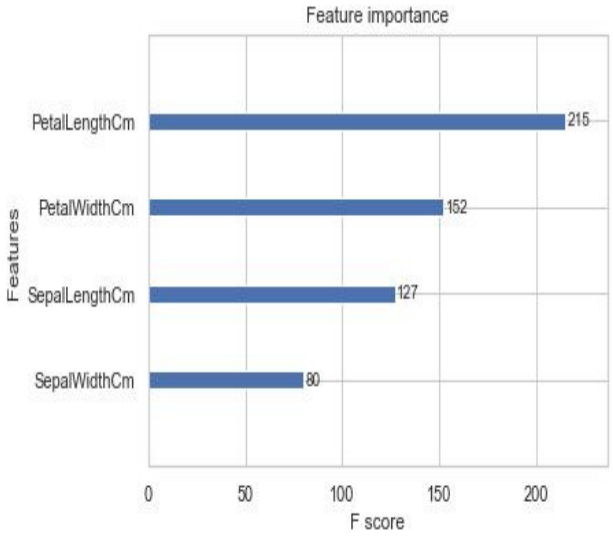
I used Anaconda navigator that is jupyter Notebook to build the model. Initially we imported the iris dataset and start farther.I used many libraries as shown below

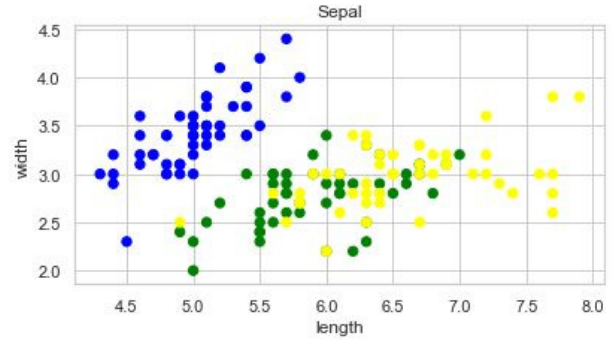
```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        from sklearn import datasets
        from sklearn.cluster import KMeans
        import sklearn.metrics as sm

In [2]: !pip install xgboost
        import xgboost as XGB

        Requirement already satisfied: xgboost in c:\programdata\anaconda3\lib\site-packages (0.90)
        Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from xgboost) (1.16.2)
        Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-packages (from xgboost) (1.2.1)

In [3]: from sklearn.cluster import AgglomerativeClustering

In [4]: from pylab import rcParams
        import seaborn as sb
```

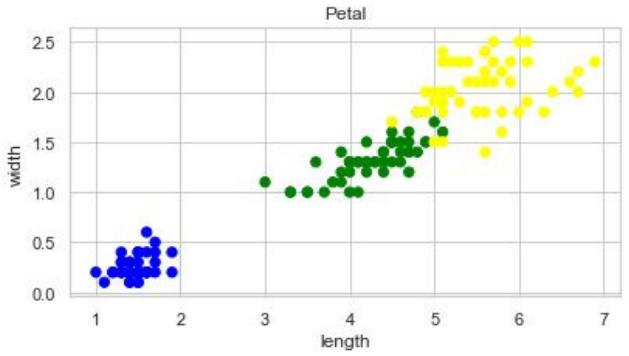
DATA VISUVAISATION

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.





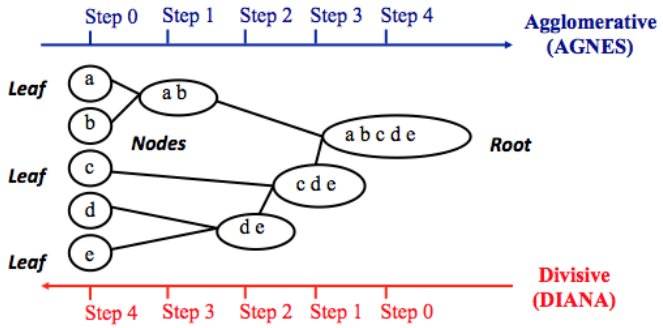


Agglomerative Clustering:

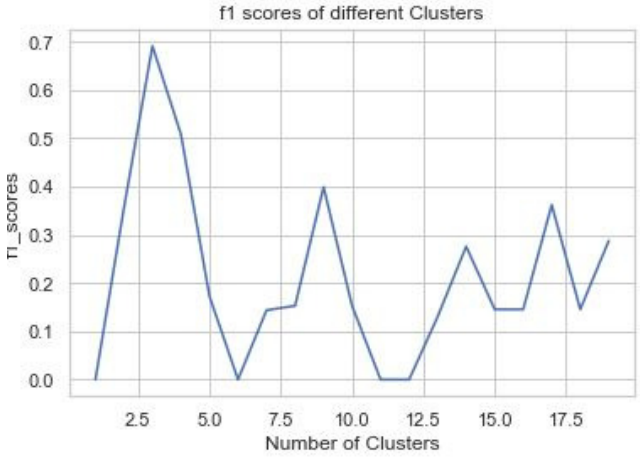
This algorithm is a hierarchal clustering algorithm. It recursively merges the pair of clusters that minimally increases a given linkage distance.

I have used the Boosted Gradient Descent to get insight of the feature importance and plotted the feature importance bar graph. From the graph is can be clearly seen that the Petal Length is the most important feature in the dataset.

Agglomerative clustering works in a "bottom-up" manner. That is, each object is initially considered as a single-element cluster (leaf). At each step of the algorithm, the two clusters that are the most similar are combined into a new bigger cluster (nodes). This procedure is iterated until all points are member of just one single big cluster (root) (see figure below).



Agglomerative Clustering- I have also used a loop to iterate over the number of clusters and used the best model for the final prediction.

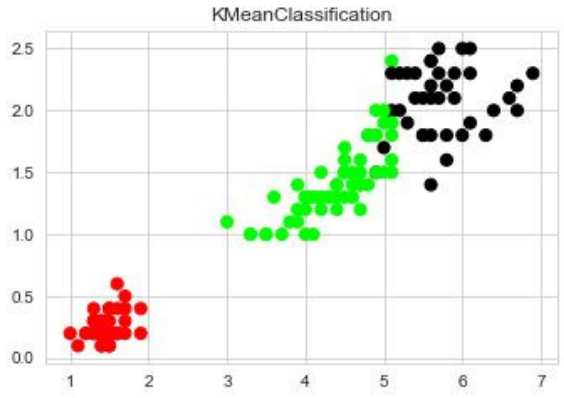


K-Means Algorithms gives good f1 score of 0. and an accuracy of 75.56%.
K-Means Algorithms gives good f1 score of 0. and an accuracy of 75.56%.

Agglomerative Algorithms gives good f1 score of 69.1% and an accuracy of 64.44%

KNN model:

Creating the the object and fitting data in the object

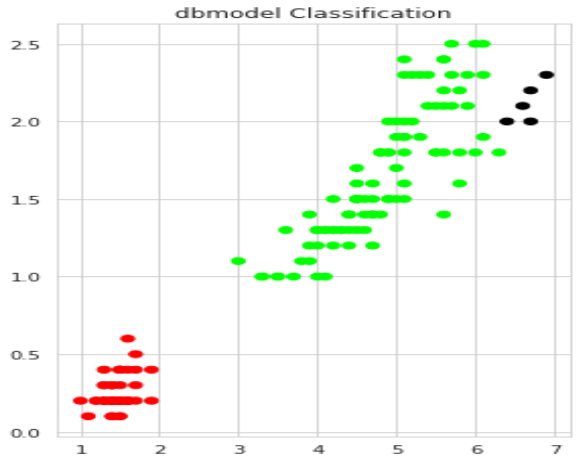


K-Means Algorithms gives good f1 score of 89.17 % and an accuracy of 89.33 %.

DBSCAN model-

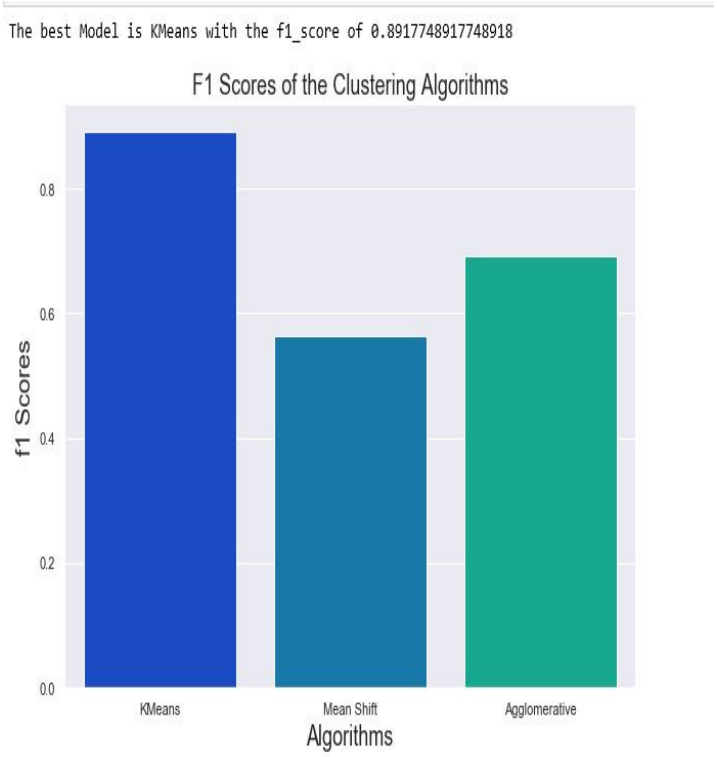
DBSCAN (**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise) is a popular **unsupervised** learning method utilized in model building and machine learning algorithms. Before we go any further, we need to define what an "unsupervised" learning method is. **Unsupervised** learning methods are when there is no clear objective or outcome we are seeking to find. Instead, we are clustering the data together based on the similarity of observations. To help clarify, let's take Netflix as an example**.**

DBSCAN Algorithms gives good f1 score of 89.17 % and an accuracy of 89.33 %.



CONCLUSION

I have plotted a bar graph that compares the f1 scores of the three clustering algorithms.



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.

. REFERENCES

1. IEEE conference paper: https://www.ieee.org/conferences/publishing/templates.html
2. Kaggle Iris Data Set : https://www.kaggle.com/ashishs0ni/iris-dataset

.