**Research Report**

**Senior Seminar**

**Tomy Romero**

**Title:** Comparing and Contrasting Clustering in Machine Learning, algorithms DBSCAN and BIRCH.

**Abstract:**

**Hypothesis: Which Clustering algorithm performs better using the same dataset, DBSCAN or BIRCH.**

Machine learning is the use and development of computer systems that can adapt and learn without the following detailed clear instructions from users by using algorithms and statistical models. How do machine learning algorithms differ from one another? How do they compare? What attributes do these algorithms possess and how do they function.

In this research report I will talk about this and more by covering a special type of Machine Learning operation called clustering and how the algorithms DBSCAN and BIRCH represent this ideology.

**Introduction:**

In machine learning the practice of grouping examples are used as a first step to understanding a dataset. Grouping these unlabeled examples is called clustering. Clustering depends on unsupervised machine learning. Unsupervised learning in machine learning are types of algorithms that learns patterns from untagged data. The idea is to mimic behaviors like people do in real life.

This makes the machine build a concise representation of its world and then create imaginative content from it. In this report I will be tackling two clustering unsupervised algorithms called DBSCAN and BIRCH by going in depth on how they work, they’re applications, what sets them apart and they’re limitations. Credit for the data I used goes to github page: <https://gist.github.com/pravalliyaram/5c05f43d2351249927b8a3f3cc3e5ecf>

**Background:**

Data mining is a technique found in clustering which takes groups of unlabeled data and bases them on their differences and similarities. The clustering algorithms are used to process data that is unclassified, raw and takes them into groups that are then represented by structures and patterns that reside in the information. Clustering can be broken down into different categorized types such as probabilistic, overlapping, hierarchical and specifically exclusive. In exclusive clustering the grouping stipulates a data point that can only exist in a singular cluster. It is commonly referred to as hard clustering.

A good example of this type of clustering can be found when you look at the K-means algorithm. Overlapping is different from exclusive because it lets data points belong to more than one cluster including different degrees of membership. K-mean soft and fuzzy clustering is an example of this type. Hierarchical clustering (HCA) can be described in two ways; agglomerative or divisive. Hierarchical agglomerative isolates data points and separates the groups from the start until they are combined on a case of similarity until the cluster is achieved.

Hierarchical divisive clustering is the opposite of agglomerative with a different approach. With divisive, a single cluster is divided by the differences between data points. Because of this approach, it is not commonly used. Probabilistic models help solve density approximation other known as soft clustering problems. In this approach, data points are clustered depending on whether they belong to a particular distribution. A common method example of this clustering is called the Gaussian Mixture Model (GMM).

In this report the types of clustering I will be covering are high density clustering with the DBSCAN algorithm and hierarchical agglomerative clustering with the BIRCH algorithm.

**Method and Approach:**

The rationale behind the machine learning workflow is to gather data, preprocess the data, decide what model will be best for the data type, train and test the data model, and elevate the results. Both DBSCAN and BIRCH require the same method and approach of having the appropriate rationale behind the machine learning workflow before clustering can be performed. A range of activities include training data sets, cleaning the data, testing, assessing and interpretation of the results. All these methods will be practiced as I go through both algorithms in efforts to compare the two, finding suitable applications for them and testing their limitations and efficiency. These methods will be approached through practical application and through code.

The problem I will be encountering will be clustering therefore classification and regression algorithms will not be needed. I will be analyzing the performance of two clustering algorithms.

**Design and Implementation:**

To summarize these are the steps taken in the DBSCAN algorithm after the initial machine learning workflow of cleaning and training data is implemented.

The design and implementation of the DBSCAN algorithm can be described as (note: eps and MinPts are parameters used in the algorithm):

* Find all the neighbor points within eps
* identify the core points with more than MinPts neighbors.
* If core not already assigned to a cluster, create a new cluster.
* Find density connected points and assign them to the same cluster as the core point.
* Iterate through the remaining unvisited points in the dataset.

Pseudo Code:

Text

Description automatically generated

Here are the steps taken in the BIRCH algorithm after the initial machine learning workflow of cleaning data is implemented:

Birch converts data into a tree data structure, while the centroids are being read off the leaf.

Like in Agglomerative Clustering, the centroids can remain the final cluster centroid.

**Four Phases:**

* Scanning data into memory.
* Resize Data
* Global clustering.
* Refining clusters

**Evaluation and interpretation of results:**

A cluster is a set of data points collected because of their similarities. Grouping unlabeled examples is known as clustering. Since the examples are unlabeled, clustering leans on unsupervised machine learning. However, if the examples become labeled then the problem turns in a classification one. Clustering is used in various industries, including market segmentation, search result grouping, social network analysis, medical imaging, anomaly detection and image segmentation.

K-means clustering is a simple and well-known unsupervised machine learning algorithm. Its objective is to group data points together so that it highlights underlying patterns in the data. To accomplish its goal, the algorithm searches for a fixed number of clusters from a provided dataset. The method must define a target number k, which implies to the number of centroids needed in the dataset. Centroids are representations of the center of a cluster.

Therefore, this algorithm, identifies k number of centroids and then distributes each data point to its closest cluster while maintaining the size of the centroids small. In the title K-means, the word means is directed to what is known as averaging the data which in exchange finds the centroid. To process the data, the algorithm begins with a group of random centroids and these centroids are used as the points of initiation. K-means then starts to perform iterative computations to optimize the locations of the centroids.

It stops this process when the centroids have become stabilized, no change of values. Additionally, it also stops when the number of defined iterations has occurred. My implementation of K-means was written in python. Keep in mind that the same dataset will be used for not only K-means but also for the DBSCAN and BIRCH algorithms that are advanced methods that builds of K-means.

The data utilized is an example of a basic mall customer dataset. The process starts by reading the data from a CSV file. The library pandas takes this data and saves it to a data frame that we can take and manipulate to get the data we need. Data frames are two-dimensional data structures that are labeled with columns of diverse types. Visualize a spreadsheet or SQL table. These data frames are used with the panda’s object.

When the data has been read it now opens various features such as being able to investigate the info of the data at hand. Couple favorite methods include info, describe head. Each of these methods provide crucial information about the data so that a proficient model can be planned and made to fit the problem of clustering. Head provides the first data rows while describe talks about the mean, std, max, min. Info provides valuable stuff like the list of columns. The list of columns then gets locked into two values one representing X, and the other representing Y.

The means to deciding what columns to pick for those values come from interpretation of data and what answers we want to answer. Right now, I want to take the data of the example mall customers spreadsheet provided from GitHub, the columns included goes as customer ID, gender, age, annual income, and spending score. What does spending score mean? Well, the spending score represents the score provided to a customer by mall authorities, based upon the money spent and the actions of the customer. I want to use clustering to provide a pattern in the data based on the annual income of and spending score of customers.

Therefore, columns three (annual income) and four (spending score) were locked in with code and used to set as the X and Y values in order to build a scatter plot to visualize the data before clustering begins.

Chart, scatter chart

Description automatically generated

The following diagram is the result of the method described (code is attached to report).

After accessing the data, I want to run my first simplest form of clustering; K-means. The elbow method performs K-means clustering with the dataset for the range of values, for example 10 iterations, and then proceed to compute the value of k an average score of all the clusters.

Chart, line chart

Description automatically generated

WCSS represents the sum of squared distance between each point and the centroid. The elbow method gets its name because when the data is plotting the WCSS along with the k value look like an Elbow. When the number of clusters goes up, the WCSS goes down. Therefore, the WCSS value is biggest when K is equal to one.

**The DBSCAN Algorithm:**

DBSCAN is the next clustering algorithm. Unlike K-means, in DBSCAN clusters are arbitrary-shaped, which may contain noise or outliers, in contrast to k-means clusters which are spherical shaped. DBSCAN stands for Density Based Spatial Clustering of Applications with Noise. In this process, clusters are created from dense areas and apart from little to no density areas. Apart from K-mean, DBSCAN does not ask for several clusters at first.

There are two parameters that are needed. Eps is the radius of neighborhoods for each data point and minPts which is the minimum number of data points in a provided neighborhood. Data point p are randomly selected data points that are called core points if there is more than a minimum quantity of points inside a e-neighborhood of p. Border points are data points within e-neighborhood of P and has lower number of minimum points within the place.

Eps is the neighborhood around a data point, therefore, if the distance between two data points is less than or equal to Eps then they are neighbors. If the value is too small, then a large part of the data will be outliers. However, if the value is chosen too large then its clusters merge and most of the data will be in the same clusters.

MinPts are the minimum number of neighbors in the ep’s radius. Therefore, the larger the dataset, the larger the value must be. Furthermore, a point that is not considered a core or border point is known as a Noise point.

**Steps in DBSCAN clustering:**

* Choose point p randomly
* Find density reachable points from p using ε and minPts parameter
* Whereas if p is a core point, create a cluster
* If it is a border point, visit the next point in a dataset
* keep algorithm going until all points are visited

DBSCAN needs E and minPts parameters to cluster. Minimum points are simple to set, it should be four for two-dimensional datasets. For multidimensional datasets, minPts should be two times the number od dimensions. To determine the optimal E parameter, it must be as small as possible, we compute by finding the k-nearest neighbor distance (KNN) which is the average distance of every data point to its k-nearest neighbor.

Within my application, the same data was prepared for DBSCAN using the same columns of annual income and spending score. Data was fitted into a model then the same data was predicted to acquire the labels to map the graph. Using the PLT library we can visualize the results and determine whether the algorithm detected the clusters.

Chart, scatter chart

Description automatically generated

We have a successful cluster; all clusters were found and color coded.

Why choice DBSCAN? This algorithm is suitable for well-separated and compact clusters.

**The BIRCH Algorithm:**

K-means does not perform well or efficient when it comes to large datasets with limited resources such as memory and CPU. Therefore, regular cluster algorithms don’t scale to running time and quality with large sets. Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) is an algorithm that can cluster large datasets. It achieves this by generating a compact and small summary of the large dataset it uses to keep as much information as possible. The small summary is clustered instead of the large dataset.

BIRCH can often match other clustering algorithms by creating the summary those methods to use. There is one drawback, the algorithm can only compute metric attributes. What are metric attributes? They are values that can be represented by Euclidean space with no categorical attributes. BIRCH has two features, the clustering and tree clustering feature.

The algorithm summarizes its large data sets into dense smaller regions which are Clustering Features. Clustering Features (CF) are written as ordered triples such as (N, LS, SS) where N is the amount of data points in the cluster. Where LS is the linear sum of data points and SS is what data points’ squared sum. A clustering feature can also be composed of other clustering features.

Clustering Trees (CT) is the compact representation of everything just talked about. In a clustering tree there are leaf nodes that each contain a sub-cluster. In every entry the tree holds a pointer to a child node. A clustering tree entry is made up from the sum of the entries into the child nodes. Each child node has a maximum number of entries which is referred to as the threshold.

The branching factor is the parameter that indicates the maximum number of clustering tree sub-clusters in the nodes. The last parameter,” n\_clusters” is the amount of clusters that are to be returned after the algorithm is complete. If it is set to None, the final step of clustering is not computed, and the intermediate clusters are brought back.

**Steps for BIRCH:**

* Find a model, in my case I will be using the same Mall Customers Data
* Fit the Data
* Predict the Data
* Create a Scatter Plot and Visualize the Results

Chart, scatter chart

Description automatically generated

Implementation of BIRCH, successful clustering completed

**Evaluation and performance of Algorithms:**

What are some limitations that DBSCAN and BIRCH have? Well for starters DBSCAN can’t cluster large differences in densities with its datasets. Combinations of minPts and Eps cannot be chosen correctly for all clusters. Therefore, picking a good eps value can be hard if the data is not fully understood. DBSCAN is not well suited to define clusters if the clusters are different in the sense of densities.

If we recall, BIRCH limitations come from its ability to only process metric attributes. What are some advantages that these two algorithms possess? An advantage of BIRCH is its ability to dynamically and incrementally cluster multi-dimensional metric data points. Often BIRCH only requires a single scan of its dataset. Advantages of DBSCAN included the ability to not require a specific number of clusters in advance. It operates well with arbitrary clusters and is vigorous to outliers and detecting them.

Although these pros and cons are not directly related to each other it does show us the identities of the two algorithms and what they have to offer. To determine which algorithm is more efficient I went about two methods, the timing of the algorithms, and the Silhouetter Score. The rationale behind the timing is to see which algorithm performs clustering at a faster rate. To conduct this experiment, I set time functions in python during the duration of algorithms ten time and calculated the average time to provide the most accurate result.

The Silhouette Score is a measurement used to analysis the performance of a clustering algorithm. How does it work? It uses the compactness of specific clusters also known as the intra cluster distance and the separation within the clusters which is the inter cluster distance. It takes these attributes to measure the representative overall score and it determines how well the clustering algorithm has operated.

The function Silhouette will calculate the mean Silhouette Coefficient of each sample using the mean nearest cluster distance and intra cluster distance.

S=(b−a)/max(a,b)S=(b−a)/max(a,b)

A is the intra-cluster distance, b is the mean nearest cluster distance. The Silhouette Score ranges from -1 to 1. When the score is closer to one, it shows that its better at clustering, so it is good practice to look for scores that are not close to negative one.

**Results:**

|  |  |  |  |
| --- | --- | --- | --- |
| Round | K-Means | DBSCAN | BIRCH |
| 1 | 1.990 | 1.323 | 1.323 |
| 2 | 2.668 | 0.925 | 1.063 |
| 3 | 1.830 | 1.394 | 0.323 |
| 4 | 1.763 | 1.215 | 1.331 |
| 5 | 1.920 | 1.130 | 0.962 |
| 6 | 2.016 | 1.320 | 1.189 |
| 7 | 1.872 | 1.337 | 1.561 |
| 8 | 2.192 | 1.176 | 1.197 |
| 9 | 2.591 | 1.235 | 1.222 |
| 10 | 2.205 | 1.278 | 1.294 |

**DBSCAN Silhouetter Score: 0.114**

**BIRCH Silhouetter Score: 0.323**

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Sum | Median | Average |
| **K-means** | 21.047 | 2.003 | 2.1047 |
| **DBSCAN** | 12.333 | 1.2565 | 1.2333 |
| **BIRCH** | 11.455 | 1.2095 | 1.1455 |

**Conclusion and Future Work:**

The algorithm that performs better using the same dataset is BIRCH. This claim is supported by two factors, the data of the time/performance. Not only did BIRCH perform faster but it also had more accurate results. The time data of BIRCH shows the lowest numbers out of three clustering algorithms, K-means, DBSCAN and itself. BIRCH got the lowest sum, median and average of the test runs. We can conclude that BIRCH performed fastest.

DBSCAN got a Silhouette score of 0.114 which was less than half of BIRCH’s score; 0.323. We can conclude that BIRCH was able to cluster the same dataset with more accurate results that were close to the optimal score of one. Therefore, my hypnosis is answered but my work is not done. There are more opportunities for research in the future because although BIRCH won in the two categories, there is a chance and metric to test it along with the other clustering algorithms.

I have compared the two algorithms and their performances, but it is important to note just like mentioned earlier that each one has a specific specialty and purpose with its own pros and cons. Leaving the door open to further research using different datasets that can be ran with both algorithms so that further tests can be run.

**Appendices:**

* Code of all three algorithms provided using the Mall Customers Dataset.
* Mall Customers CSV file dataset
* Txt File of performance results

**References:**

IBM Cloud Education. (n.d.). What is unsupervised learning? IBM. Retrieved November 20, 2022, from

Self-organizing maps - kohonen maps. GeeksforGeeks. (2022, February 16). Retrieved November 20, 2022, from <https://www.geeksforgeeks.org/self-organising-maps-kohonen-maps/>

DBSCAN clustering in ML: Density based clustering. GeeksforGeeks. (2022, August 24). Retrieved November 20, 2022, from <https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/>

Siddiquiamir. (2022, January 15). Python-clustering-tutorials/mall\_customers.CSV at main · Siddiquiamir/python-clustering-tutorials. GitHub. Retrieved November 30, 2022, from https://github.com/siddiquiamir/Python-Clustering-Tutorials/blob/main/Mall\_customers.csv

Google. (n.d.). What is clustering? &nbsp2;|&nbsp; machine learning &nbsp;|&nbsp; google developers. Google. Retrieved November 20, 2022, from <https://developers.google.com/machine-learning/clustering/overview#:~:text=In%20machine%20learning%20too%2C%20we,relies%20on%20unsupervised%20machine%20learning>.

Workflow of a machine learning project - towards data science. (n.d.). Retrieved November 20, 2022, from <https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94>

Birch in Data Mining - Javatpoint. www.javatpoint.com. (n.d.). Retrieved November 20, 2022, from <https://www.javatpoint.com/birch-in-data-mining>

How DBSCAN works and why should we use it? (n.d.). Retrieved November 20, 2022, from <https://towardsdatascience.com/how-dbscan-works-and-why-should-i-use-it-443b4a191c80>

Benzer, N. J. (2022, May 25). Balanced iterative reducing and clustering using heirarchies(birch). Medium. Retrieved November 20, 2022, from <https://medium.com/@noel.cs21/balanced-iterative-reducing-and-clustering-using-heirachies-birch-5680adffaa58>

Bedre, R. (2022, May 1). DBSCAN in python (with example dataset). Data science blog. Retrieved November 29, 2022, from <https://www.reneshbedre.com/blog/dbscan-python.html>

Kumar, A. (2021, June 21). Kmeans silhouette score with python examples - dzone. dzone.com. Retrieved November 30, 2022, from <https://dzone.com/articles/kmeans-silhouette-score-explained-with-python-exam>

Github Data: <https://gist.github.com/pravalliyaram/5c05f43d2351249927b8a3f3cc3e5ecf>