**OPTICS – Ordering Points to Identify Clustering Structure**

**CS 512 Project Report**

**Group 14**

**Topic:**

Using and visualizing OPTICS clustering algorithm to Identify Patterns in Mall Customers’ Spending Habits

**Team members**:

● Rishikesh Gawade (NetID: rkg63, Email: [rkg63@scarletmail.rutgers.edu](mailto:rkg63@scarletmail.rutgers.edu))

● Pavankumar Prakash Savanur (NetID: ps1149, Email: [ps1149@scarletmail.rutgers.edu](mailto:ps1149@scarletmail.rutgers.edu))

● Ekta Dhobley (NetID: ed560, Email: <mailto:ed560@scarletmail.rutgers.edu>)

**Project Description**:

Visualizing OPTICS (Ordering Points To Identify the Clustering Structure), a deterministic algorithm for an iterative clustering of data points for cluster analysis and outlier detection.

● OPTICS, like DBSCAN, is a density-based spatial clustering algorithm that iteratively identifies dense regions among points and groups them together to form distinct clusters.

● It works on the concept of forming a neighborhood given the threshold for maximum distance between neighbors and the number of points required to form a cluster.

● It is a widely used density-based clustering technique as it has the capability, unlike DBSCAN, to deal with varying density in any point space, and effectively forms clusters out of the points.

● We aim to visualize the cluster formation process to demonstrate how the points move between clusters as we try to reach convergence.

**Data Description:**

Mall Customer Segmentation Data:

<https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python>

The dataset is in CSV format with 200 records. This is a static dataset which contains information about mall customers with interesting features like age, annual income, and spending score. The data is at rest and will fit locally in a desktop.

**The table below represents what the data represents and how we used it to fit our algorithm.**

**Table

Description automatically generated**

**Snippet of data:**

**Table

Description automatically generated**

**Data Transformation:**

We explored the data and checked the distribution of it to check if it needed any transformation, like normalization or filtering. The annual income is normally distributed, so is the spending score and the age. Thus, we don’t need normalization. Also, as no factors correlate with the Gender much, we didn’t need to encode the gender to its numerical representation. Finally, we checked if the data had any missing values in any relevant columns and eliminated those with NaN’s or empty values.

**Parameters:**

1. DB: A database object containing the data in which we aim to identify clusters. This can be a result set, data frame, map, or any other tabular representation of data.
2. Eps: Called epsilon, this parameter is a threshold for the maximum core distance (radius) any core point can have.
3. minPts: Called minPoints, this is a hyperparameter that determines the strength of the ϵ-neighborhood of points for the point to become a core point.

**Pseudocode for the OPTICS Algorithm:**

1. We will use two lists, one is an ordered list which is empty initially, and maintain another empty list for storing neighbors of the processed points in the order of their reachability distances.
2. The main loop of the OPTICS algorithm describes the optics function. The function can be called with the database/data frame, epsilon value and minimum amount of points value.
3. For each point in the database, we will set the reachability-distance to UNDEFINED, we will compute it later.
4. For the unprocessed points (p) we will perform a set of following steps:
   * 1. We will get the neighbors of that point p.
     2. We mark p as processed.
     3. We push p to the ordered list.
     4. We will now look at the core-distance of p. If the core-distance of p is not undefined, that means it is the core point, therefore we will explore it further. If it is not the core point, we move on to the next unprocessed point.
        1. For the core points, we will initialize a priority queue, which the update function (discussed later) will order the queue based on reachability distance.
        2. In the ordered priority queue, the points with the lowest reachability point are covered first, for each point we get its neighbors. We the mark the point as processed, and then add it to the ordered list.
        3. We will extend the priority queue if it is a core point, as the clusters might be close to each other and likely belong to the same bigger cluster.
        4. Extending the priority queue means that more points are added to the reachability-distance ordered Seeds list (priority queue).
5. Update function:
   1. In the update function, the core distance is calculated for the point for which its neighbors are passed. So we know what is the minimum distance to keep the neighborhood true.
   2. For each point in the neighborhood, the reachability distance is undefined if the point is not processed before. So we compute the reachability distance, this value we add to the priority queue at that distance.
   3. If the reachability distance of the point is already, we update the queue and move the already set reachability distance forward, if the newly computed reachability distance is lower than the earlier one.
   4. During extension of the priority queue, the order in the queue is continuously changing, according to the reachability distance.
   5. The outliers define the threshold or cut off points for forming different clusters.

**Running the Algorithm on the aforementioned dataset:**

1. Initial State:
2. Midway state, while determining the clusters among the ordered points:
3. Final State:

**Complexity:**

Time complexity:

* Best and average case: O(nlogn)

As the algorithm searches for the neighbors of any point, the neighborhood search typically costs O(logn). Doing this for all the points until they are processed, the total average costs goes up to O(nlogn)

* Worst case: O(n2)

As the ϵ acts as a hyperparameter for determining the clusters, if it is greater than the maximum distance between any pairs of points in the dataset, then the neighborhood search costs O(n) time. Doing this for all ‘n’ points will shoot the algorithm’s time complexity to O(n2).

This complexity is similar to DBSCAN algorithm, however, a constant slowdown factor of 1.6 is observed for OPTICS in comparison to DBSCAN.

Space complexity:

While searching the neighborhood and trying to explore the neighborhood of the neighboring core point of any given point, we may end up storing all n points at once in the priority queue, which, when exhausted, takes the algorithm to the next unprocessed point. Thus, while processing any particular point, the algorithm will store at most all the n points. Thus, the space complexity is O(n).

**Analysis:**

Which attributes give an idea of the purchasing behavior of the customers?

Applying the correlation function on the dataset, we found that the spending score is positively correlated with the annual income and negatively correlated with the age of the customers, suggestive of the fact that young adults who earn might be spending relatively more while shopping, as compared to the older customers. However, these correlations are not very strong. Using statistical analytical tools like the boxplot, we think the presence of outliers are affecting the correlations.

Can these spending habits be categorized in any way?

We used the simple scatterplot in 2d first, to plot the Age vs Spending Score comparison, then the Annual Income vs Spending Score, and ultimately a 3d plot to plot all three features. The points, each representing a customer, tend to show a grouping behavior, as the once with low age and high annual income salary were quite close to each other, while the older people with lower annual salary were also pretty close to each other.

How can clustering help identify these groups and even exclude from the groups the customers with unusually high or low spending behavior?

OPTICS algorithm adapts to points with varying density in space and accordingly identifies clusters of interest, while even eliminating the outliers that don’t share a strong neighborhood with the clusters or groups of points around them. Using this while tuning the hyperparameters like epsilon and minPts, the algorithm effectively finds groups of customers with similar spending habits and groups them into a cluster. The shopping mall or the ecommerce business owners can then target any of these groups with offers to engage them more into shopping.

**Conclusion:**

Upon exploring the dataset using simple correlation functions, bar plots and scatterplots, we deduced that the data had outliers as well as the points that were not closely packed together depicted similar behavior. Thus, to address such groups of varying density, we needed to implement and run the data through a density-based algorithm, with a special emphasis on identifying clusters among points having varying density, while also making sure that the outliers were excluded from such clusters.

Thus, the OPTICS clustering algorithm most accurately fits the description of the dataset, and helped us identify relevant groups of customers with respect to their age, income and overall spending behavior.