CS 4710-01 Programming Project: DBSCAN

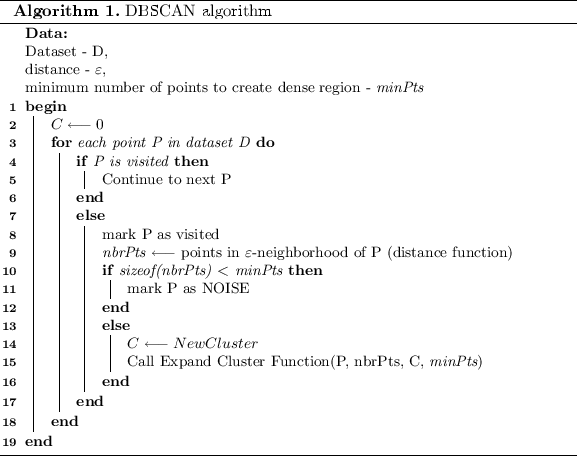
Jacob MacLeod

Dr. Chung

**Description of Algorithm and Dataset**

For this project, I chose to program a clustering algorithm, specifically DBSCAN. DBSCAN is a density-based clustering algorithm, so it can be used for clusters of arbitrary shape. Clusters are dense regions of data points. Clustering algorithms are known as unsupervised learning, meaning that it is learning by observations, rather than learning by examples. DBSCAN starts with an arbitrary point, and if the point is surrounded by enough other points, the points surrounding it are put in a cluster with it.

The dataset I chose to use for this project was the 1990 US Census Data from the UCI Machine Learning Repository. This dataset contains a lot of lines, each of which represents one person, and each line has a lot of numbers in it. These integers represent the values of all the attributes for each person. There were a lot of attributes in the dataset, representing the different pieces of information collected about each person by the census.



**Description of Implementation**

I implemented the DBSCAN algorithm in Java, using a greatly reduced version of the dataset. The first time I tried to use the whole dataset, the program would not run without running out of memory. I then cut the number of objects down to 1500. In my Java program, I started by marking all objects as unvisited, then I would randomly select one object, check if it is unvisited, and then, if the object is unvisited, I checked to see whether it was part of a cluster or if it was noise. If the point is part of a cluster, then all the points within its epsilon-neighborhood would be checked to see if they have enough support. If the points within the neighborhood had enough points, those points would be marked as being in the neighborhood, and the points in the neighborhood would be added to the cluster if they were not already in a cluster. This process is repeated for all objects in the neighborhood until all of them have been visited. The randomly selected points are either marked as part of a cluster or as noise until no object is unvisited.

**Source Codes**

**Results**

I programmed the DBSCAN algorithm in Java and wrote out the sizes of each of the clusters to the console. I did not indicate which clusters each object was part of. If I had, I would be able to test how well the clusters are separated. As far as I can tell, the total number of objects printed out in clusters and in noise matches the total number of objects in the file.

**Discussion**

I expected programming this algorithm to be difficult, and it was. I learned how to translate an algorithm from pseudocode to a real programming language, and that will probably be very useful. It took a long time for me to get the program to run correctly. For example, earlier I had a loop within a loop, both of which were never able to meet their end condition, so I had infinite loops. It was hard to figure out how to program the algorithm correctly. In the future, I plan to improve this algorithm by indicating which cluster each object is in and testing how well each cluster is separated.