Homework1 Report / Seniha Sena Topkaya / 131044020

As part of this assignment, the db-scan algorithm is implemented according to the algorithm in the book. Diabetes patients' data was used as sample data set (**data.txt**). The data set includes 4 different properties (date, time, code and value). The code data from these features is clustered. These codes show blood glucose results measured under different conditions:

The Code field is deciphered as follows:

33 = Regular insulin dose

34 = NPH insulin dose

35 = UltraLente insulin dose

48 = Unspecified blood glucose measurement

57 = Unspecified blood glucose measurement

58 = Pre-breakfast blood glucose measurement

59 = Post-breakfast blood glucose measurement

60 = Pre-lunch blood glucose measurement

61 = Post-lunch blood glucose measurement

62 = Pre-supper blood glucose measurement

63 = Post-supper blood glucose measurement

64 = Pre-snack blood glucose measurement

65 = Hypoglycemic symptoms

66 = Typical meal ingestion

67 = More-than-usual meal ingestion

68 = Less-than-usual meal ingestion

69 = Typical exercise activity

70 = More-than-usual exercise activity

71 = Less-than-usual exercise activity

72 = Unspecified special event

The **Pseudocode** specified for the algorithm in the textbook is as follows:

```
 #DBSCAN
#mark all objects as unvisited;
#do
#randomly select an unvisited object p;
#mark p as visited;
#if the -neighborhood of p has at least MinPts objects
#       create a new cluster C, and add p to C;
#let N be the set of objects in the -neighborhood of p;
#for each point p0 in N
#       if p0 is unvisited
#               mark p0 as visited;
#       if the -neighborhood of p0 has at least MinPts points,
#               add those points to N;
#       if p0 is not yet a member of any cluster, add p0 to C;
#               end for
#               output C;
#       else mark p as noise;
#until no object is unvisited;
```

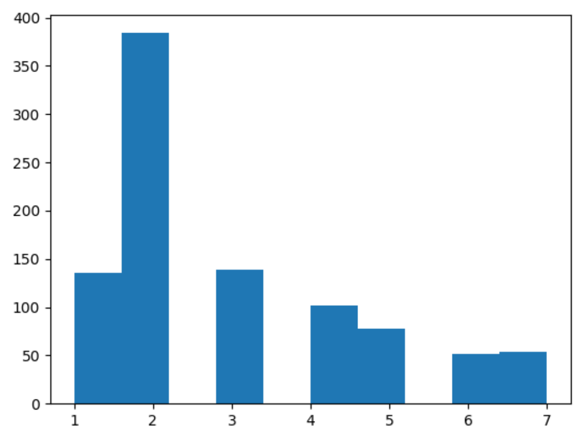
Eps is the radius of the circle to be determined, and MinPts is the minimum number of neighbors in this circle.

Test:

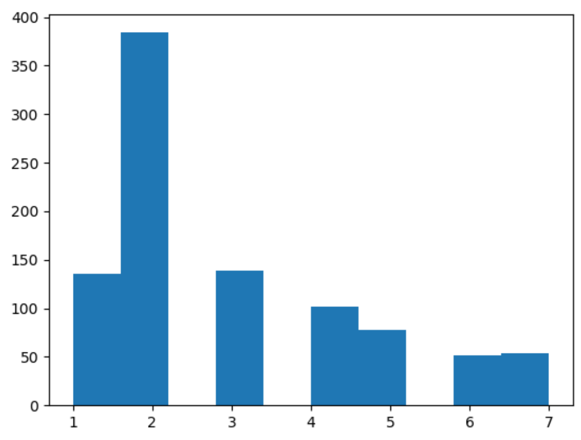
Variables are assigned different values and results are visualized using histogram graphics. The different values given affected the number of clusters and the size of the clusters.

Test result graphs and comments are as follows:

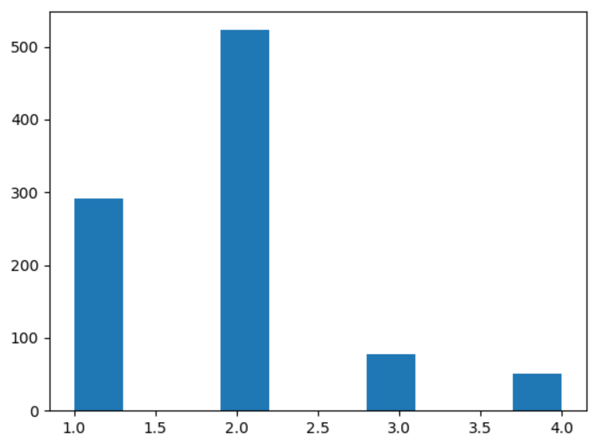
Eps = 1 && MinPts = 1;



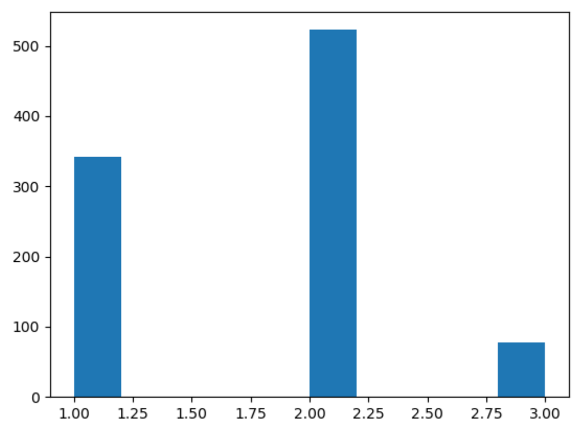
Eps = 1 && MinPts = 5;



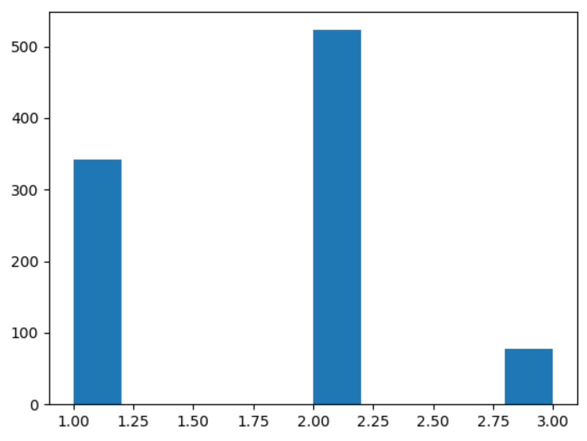
Eps = 3 && MinPts = 5;



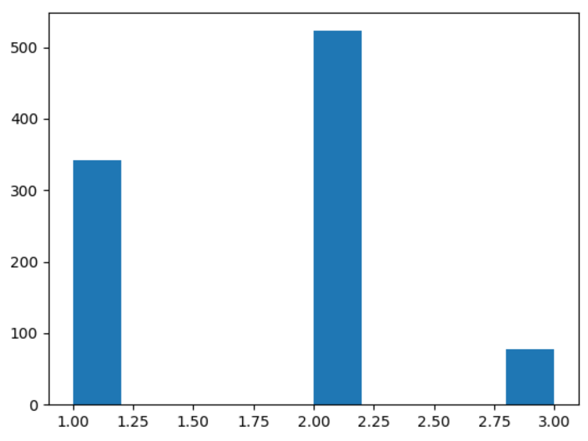
Eps = 5 && MinPts = 1;



Eps = 5 && MinPts = 3;



Eps = 5 && MinPts = 5;



As can be seen from the graphs, the number of clusters decreased as the Eps value increased.

When **Eps = 1, MinPts = 5**, the number of clusters formed was **7**.

When **Eps = 3, MinPts = 5**, the number of clusters formed was **4**.

When **Eps = 5, MinPts = 5**, the number of clusters formed was **3**.

Cluster creation or cluster expansion condition was to provide the threshold value in MinPts neighbor number.

When **Eps = 1, MinPts = 1**, the number of clusters formed was **7**.

When **Eps = 5, MinPts = 3**, the number of clusters formed was **3**.

When **Eps = 5, MinPts = 5**, the number of clusters formed was **3**.

Changing the MinPts value did not affect the number of clusters much, possibly due to the dataset used.

For automatic determination of Eps and MinPts values;

Eps should be chosen according to the distance of the data set. We can use a knn (K-Nearest Neighbors) graph to find it. In the **KNN** algorithm, we tell you how many nearest neighbors will be calculated. The K value will directly affect the result. If K is 1, the probability of overfitting will be very high. If it is too large, it will give very general results.

A minimum of MinPts can be obtained from a number of dimensions (D) in the data set, such as **MinPts ≥ D + 1**.