# DBSCAN

# DBSCAN

- DBSCAN - Density-based spatial clustering of applications with noise is a powerful technique which can be used for clustering and outlier detection.
- Let’s review what this section will cover!

# DBSCAN

- Section Overview:
  - Intuition of DBSCAN
  - DBSCAN vs. K-Means Clustering
  - DBSCAN Hyperparameters Theory
  - DBSCAN Hyperparameters Coding
  - Outlier Project Exercise
  - Project Solutions

# Let’s get started!

# DBSCAN

Theory and Intuition

# DBSCAN

- DBSCAN stands for Density-based spatial clustering of applications with noise.
- Let’s review a brief history of the algorithm and then explore an intuition based approach to understanding how it works.

# DBSCAN

- **1972: Robert F. Ling published a closely related algorithm in "***The Theory and Construction of k-Clusters***" with an expected run time of O($n^3$).**
- **This means that as n number of points grows, the run time of the algorithm grows cubically!**

# DBSCAN

- 1996: Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu proposed the modern version of DBSCAN with a runtime of $O(n^2)$.
- 2014: DBSCAN was awarded the test of time award at the leading data mining conference, SIGKDD.

# DBSCAN

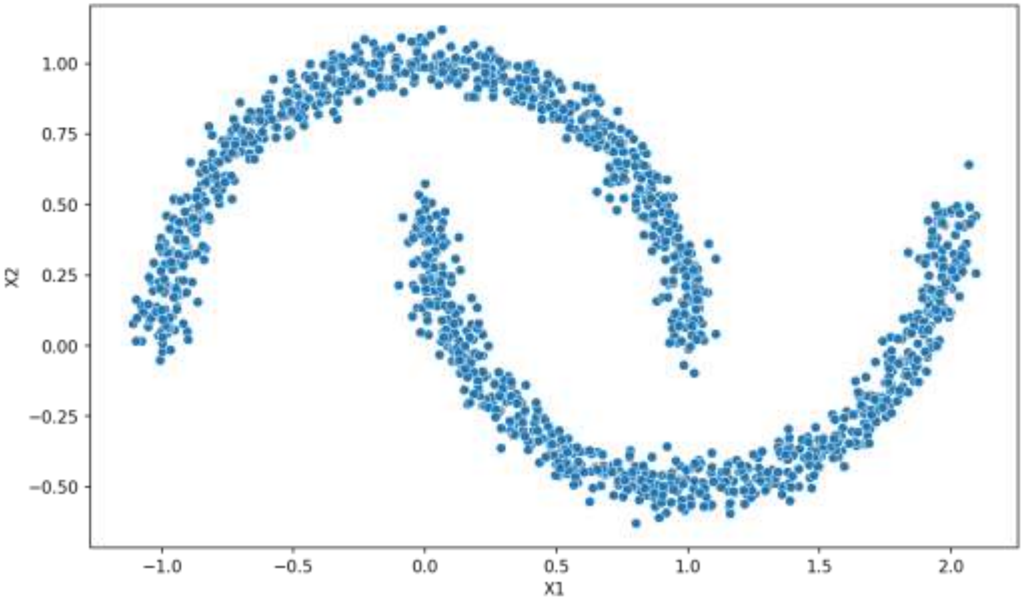
- Questions to consider:
    - How does DBSCAN work?
    - Advantages and disadvantages of DBSCAN?
    - How does it deal with outliers and noise?

# DBSCAN

- DBSCAN Key Ideas
  - DBSCAN focuses on using **density** of points as its main factor for assigning cluster labels.
  - This creates the ability to find cluster segmentations that other algorithms have difficulty with.

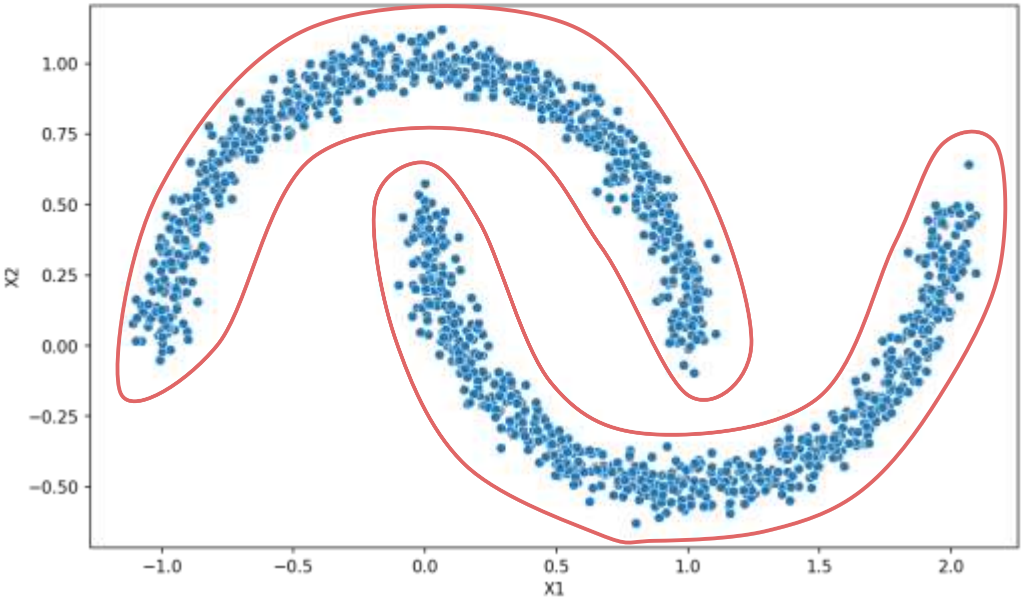
# DBSCAN

- Consider the following data set:



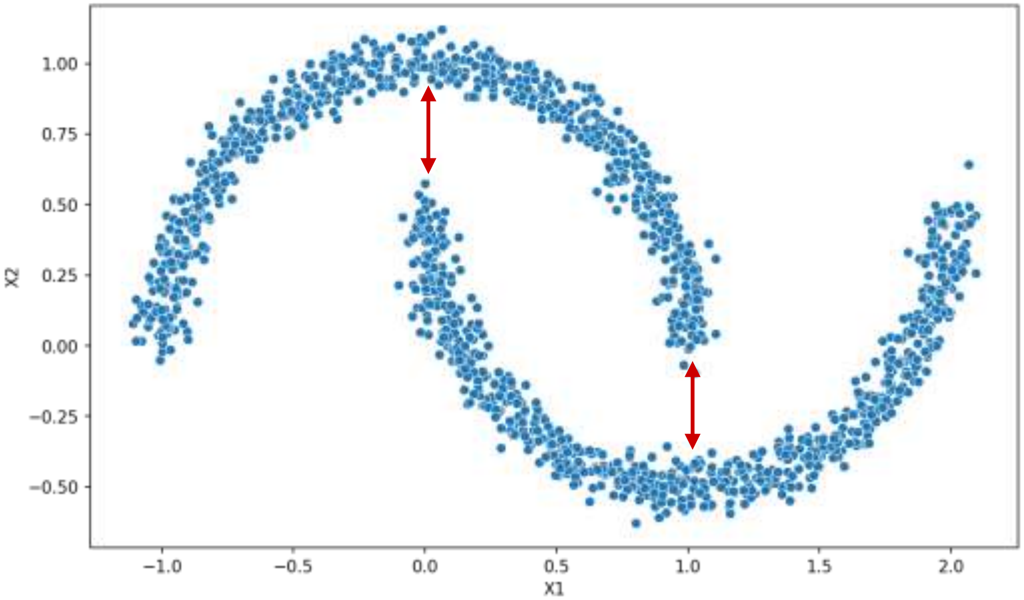
# DBSCAN

- Cleary two “moon” shaped clusters:



# DBSCAN

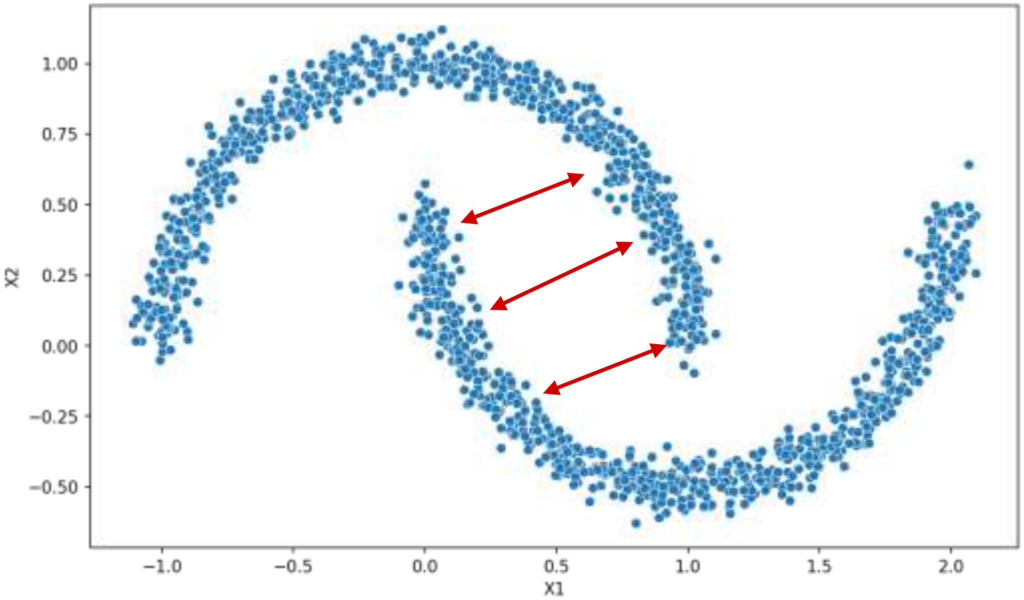
- But distance based clustering has issues:




PIERIAN DATA

# DBSCAN

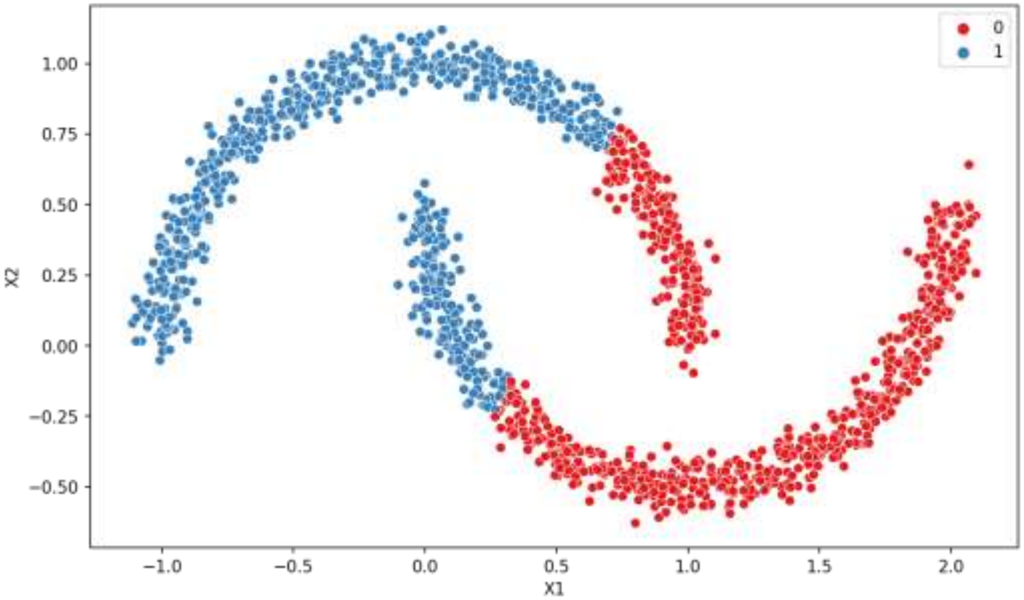
- But distance based clustering has issues:




PIERIAN DATA

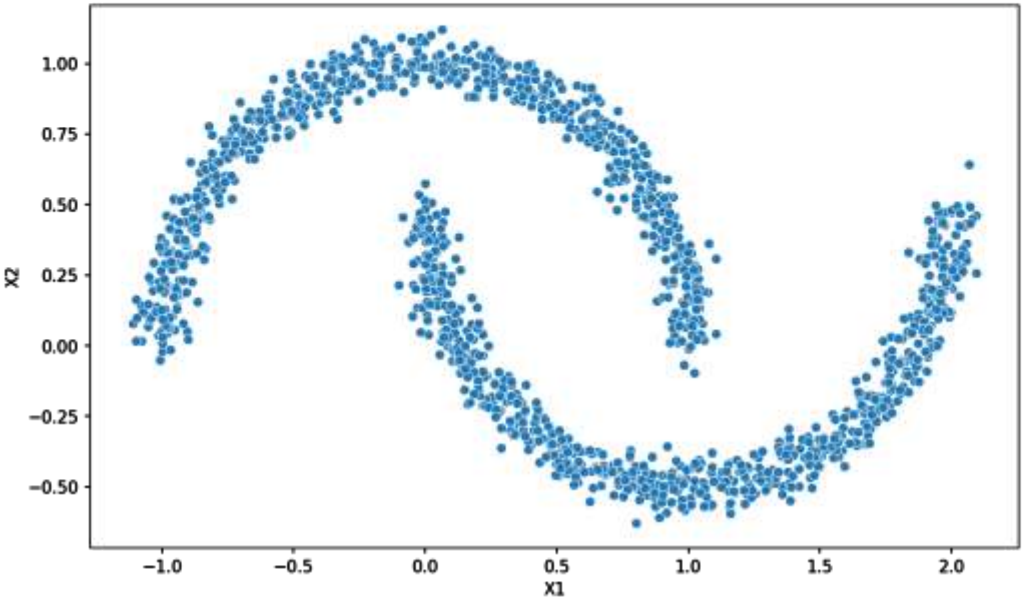
# DBSCAN

- Results of K-Means:



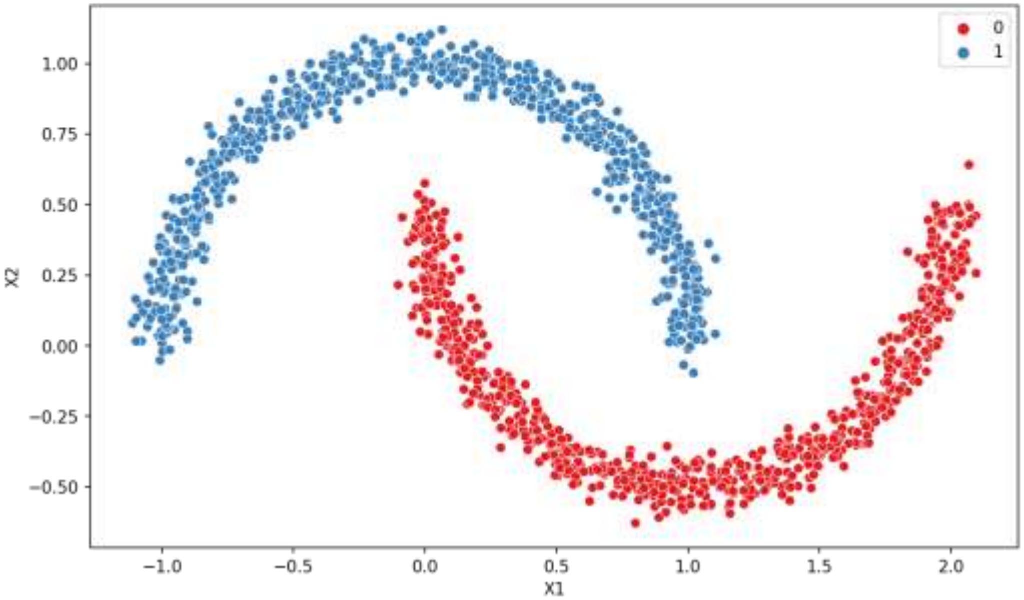
# DBSCAN

- Results of DBSCAN:



# DBSCAN

- Results of DBSCAN:



# DBSCAN

- DBSCAN iterates through points and uses two key hyperparameters (epsilon and minimum number of points) to assign cluster labels.
- Unlike K-Means, it focuses on density as the main factor for cluster assignment of points.

# DBSCAN

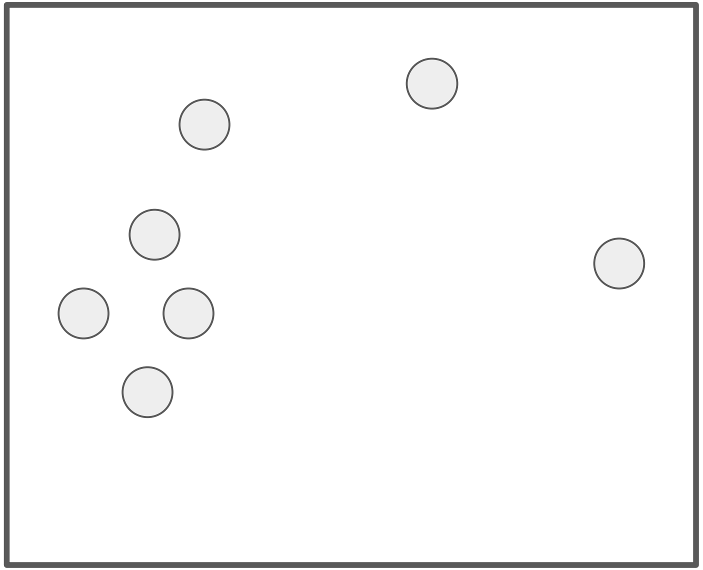
- DBSCAN Key Hyperparameters:
    - Epsilon:
        - Distance extended from a point.
    - Minimum Number of Points:
        - Minimum number of points in an epsilon distance.

# DBSCAN

- DBSCAN Point Types:
  - Core
  - Border
  - Outlier

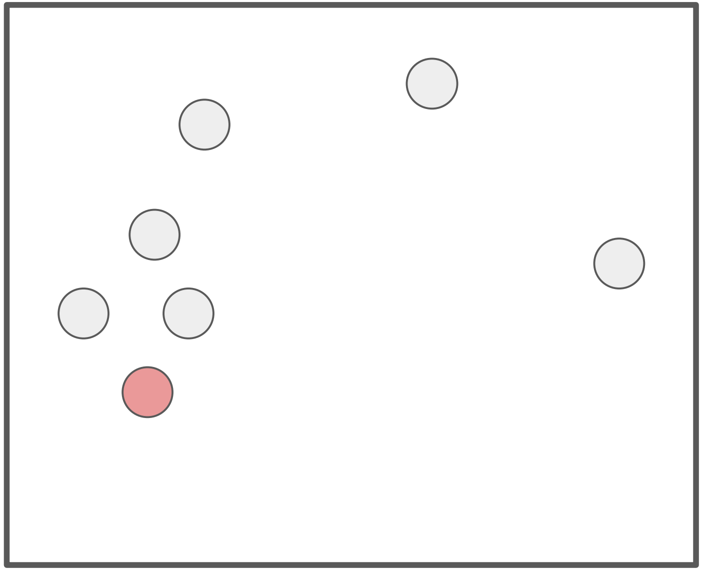
# DBSCAN

- DBSCAN Point Types:
    - Core
    - Border
    - Outlier



# DBSCAN

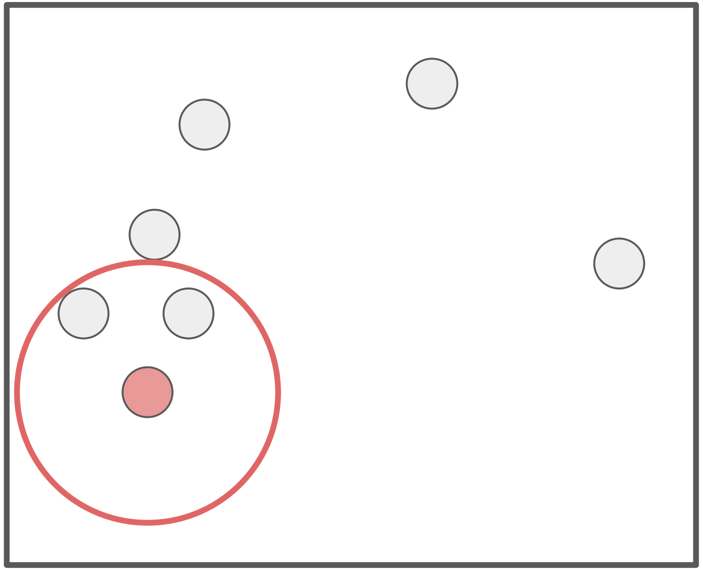
- DBSCAN Point Types:
    - Core



# DBSCAN

- DBSCAN Point Types:
  - Core

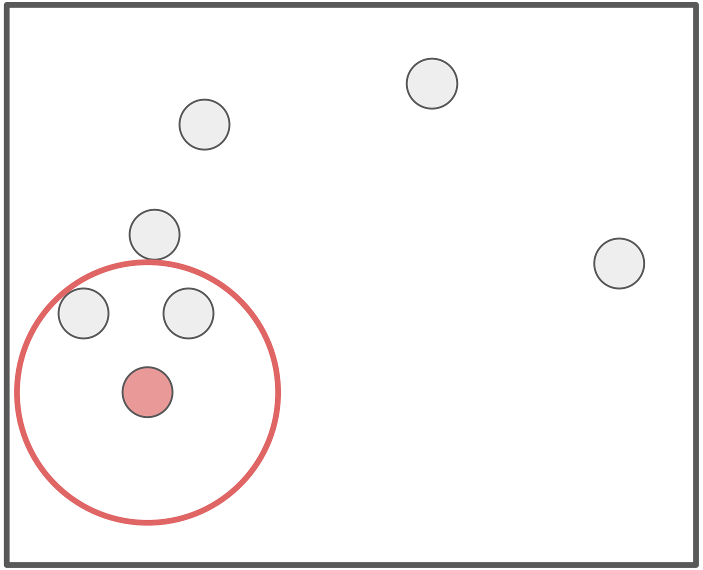
$\varepsilon = 1$



# DBSCAN

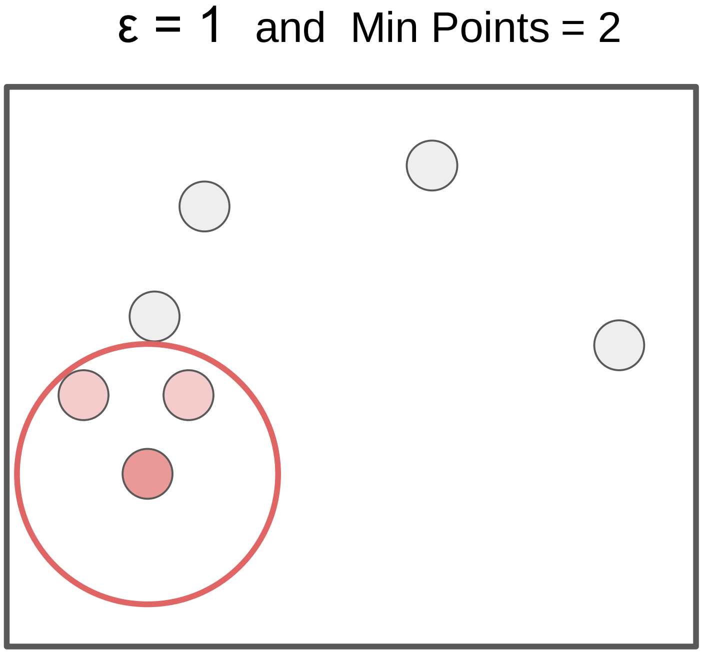
- DBSCAN Point Types:
  - Core

ε = 1  and  Min Points = 2



# DBSCAN

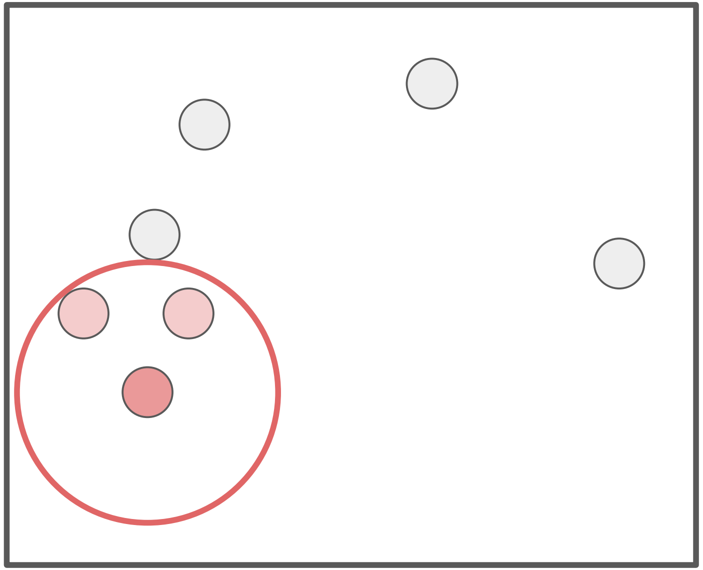
- DBSCAN Point Types:
  - Core



# DBSCAN

- DBSCAN Point Types:
  - Core:
    - Point with min. points in epsilon range.

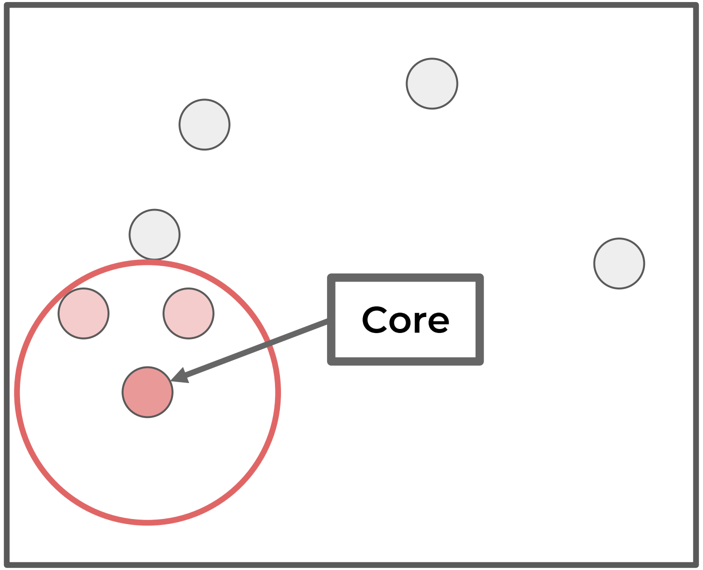
ε = 1 and Min Points = 2



# DBSCAN

- DBSCAN Point Types:
  - Core:
    - Point with min. points in epsilon range.

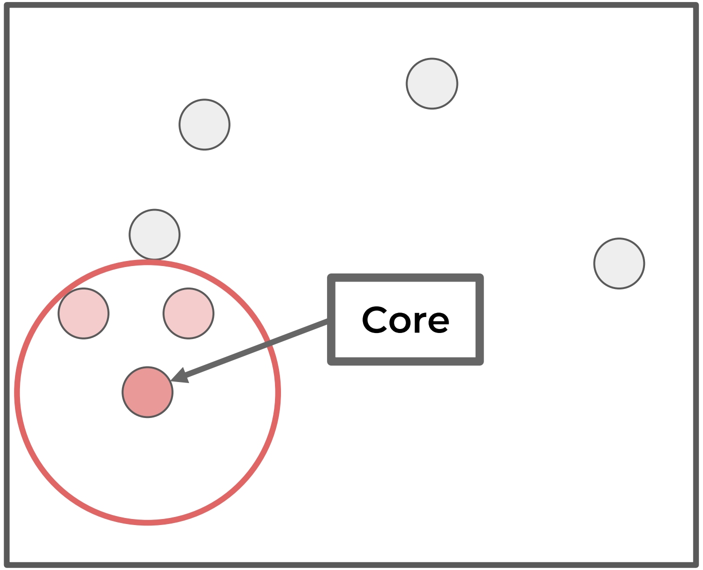
ε = 1 and Min Points = 2



# DBSCAN

- DBSCAN Point Types:
  - Core:
    - Point with min. points in epsilon range (including itself).

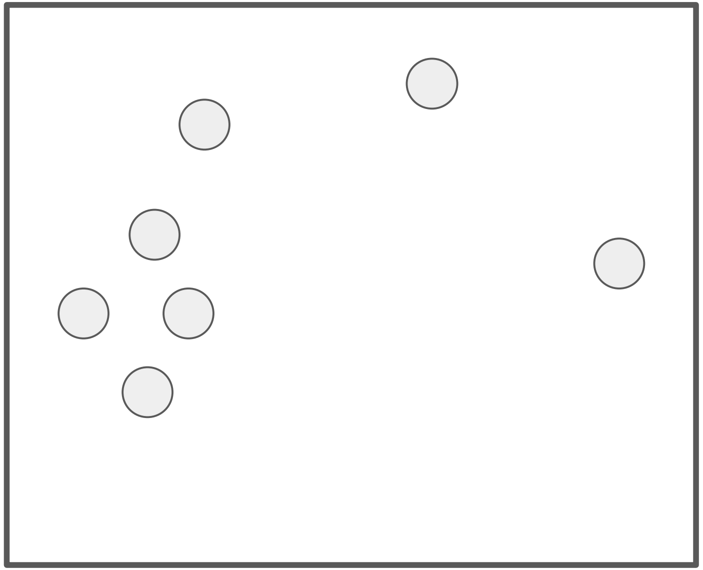
ε = 1 and Min Points = 3



# DBSCAN

- DBSCAN Point Types:
  - Border:
    - In epsilon range of core point, but does not contain min. number of points.

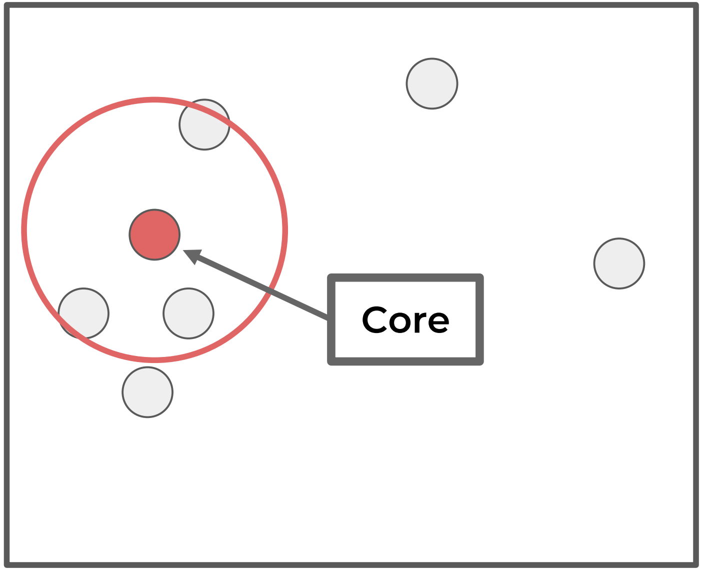
ε = 1 and Min Points = 3



# DBSCAN

- DBSCAN Point Types:
  - Border:
    - In epsilon range of core point, but does not contain min. number of points.

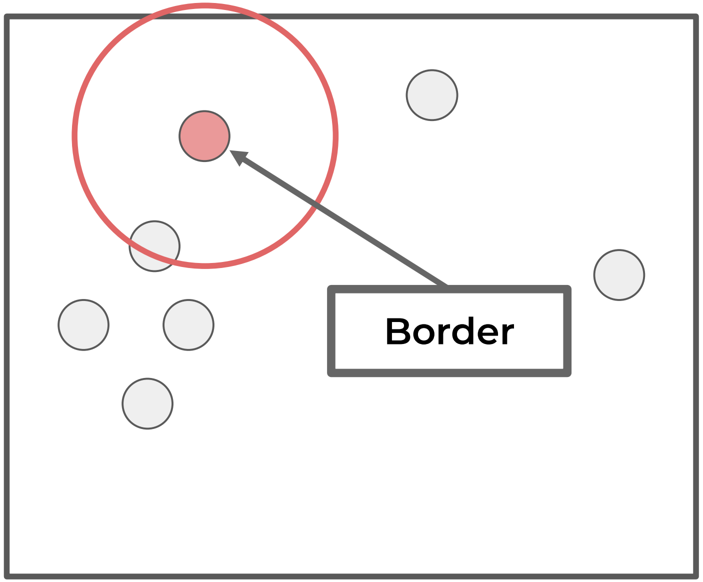
ε = 1  and  Min Points = 3



# DBSCAN

- DBSCAN Point Types:
  - Border:
    - In epsilon range of core point, but does not contain min. number of points.

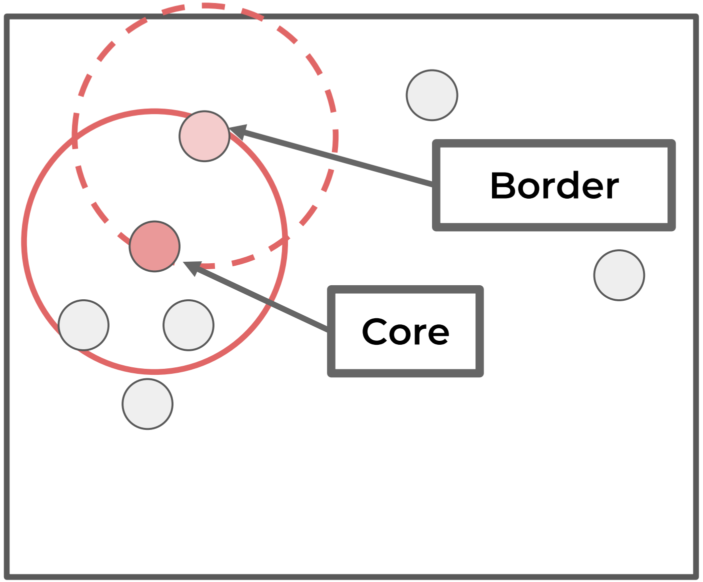
$\varepsilon = 1$ and Min Points = 3



# DBSCAN

- DBSCAN Point Types:
  - Border:
    - In epsilon range of core point, but does not contain min. number of points.

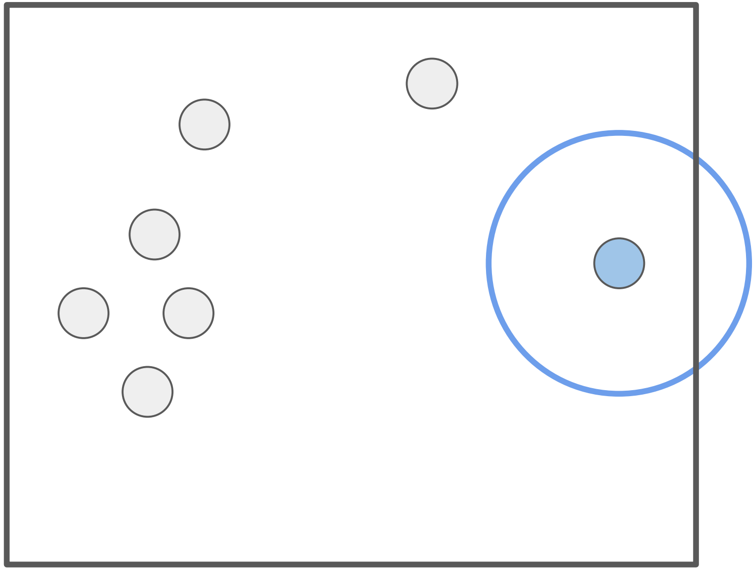
ε = 1 and Min Points = 3



# DBSCAN

- DBSCAN Point Types:
  - Outlier:
    - Can not be "reached" by points in a cluster assignment.

ε = 1 and Min Points = 3



# DBSCAN

- We will discuss neighborhoods, epsilon, and minimum number of points in further detail later on, but let's review the actual process of DBSCAN for assigning clusters.

# DBSCAN

- DBSCAN Procedure:
  - Pick a random point not yet assigned.
  - Determine the point type.
  - Once a **core** point has been found, add all directly reachable points to the same cluster as core.
  - Repeat until all points have been assigned to a cluster or as an outlier.

# DBSCAN

- Let’s explore a useful visualization of the procedure!

# DBSCAN

Coding Example on Data Sets

# DBSCAN

- Let’s explore how DBSCAN compares to K-Means clustering on some unique data sets to get an intuitive understanding of the density based approach of DBSCAN versus a distance based clustering approach of K-Means.

# DBSCAN

Key Hyperparameters

# DBSCAN

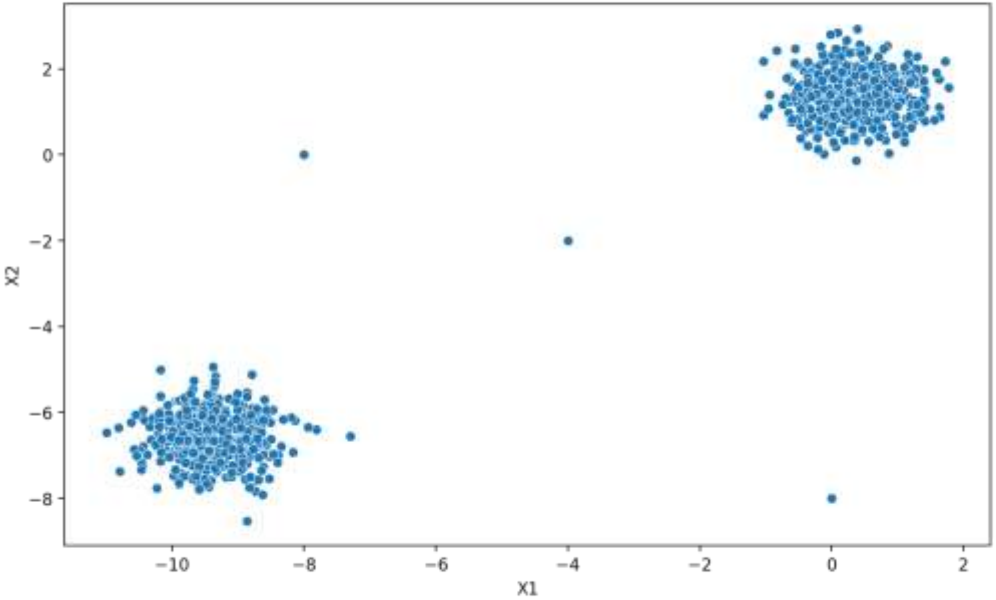
- As we’ve seen already, there are two key hyperparameters to consider for DBSCAN:
  - Epsilon:
    - Distance extended from a point to search for Min. Number of Points.
  - Min. Number of Points:
    - Min. Number of Points within Epsilon distance to be a core point.

# DBSCAN

- Adjusting these hyperparameters have two main outcomes:
    - Changing number of clusters.
    - Changing what is an outlier point.

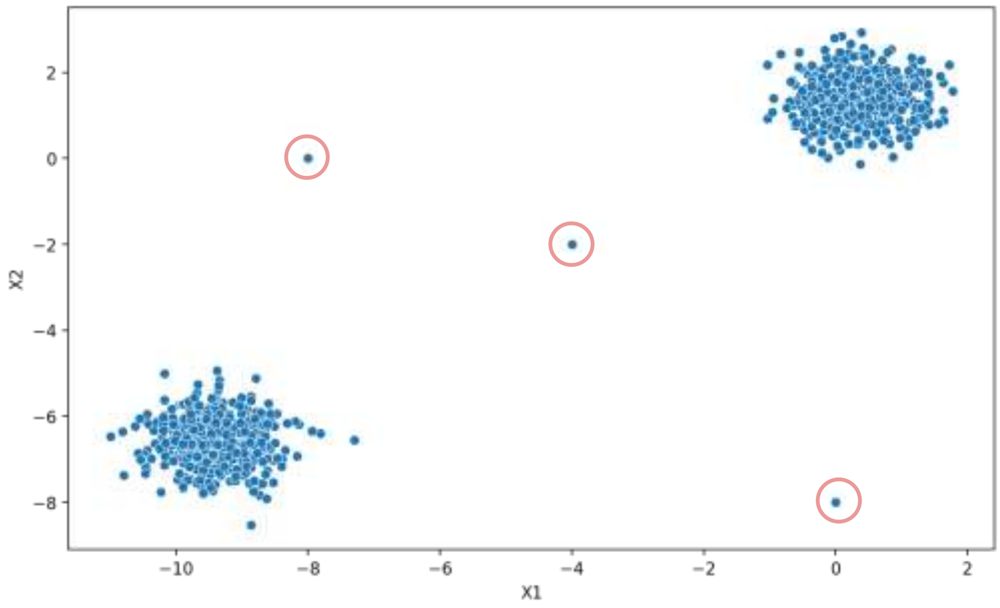
# DBSCAN

- Example Data Set:



# DBSCAN

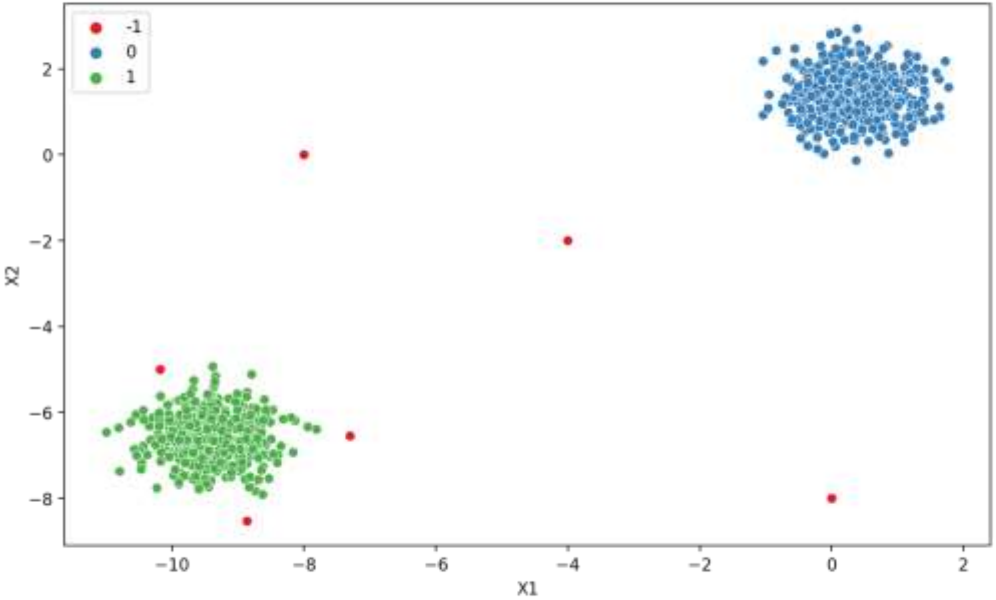
- Example Data Set:




PIERIAN DATA

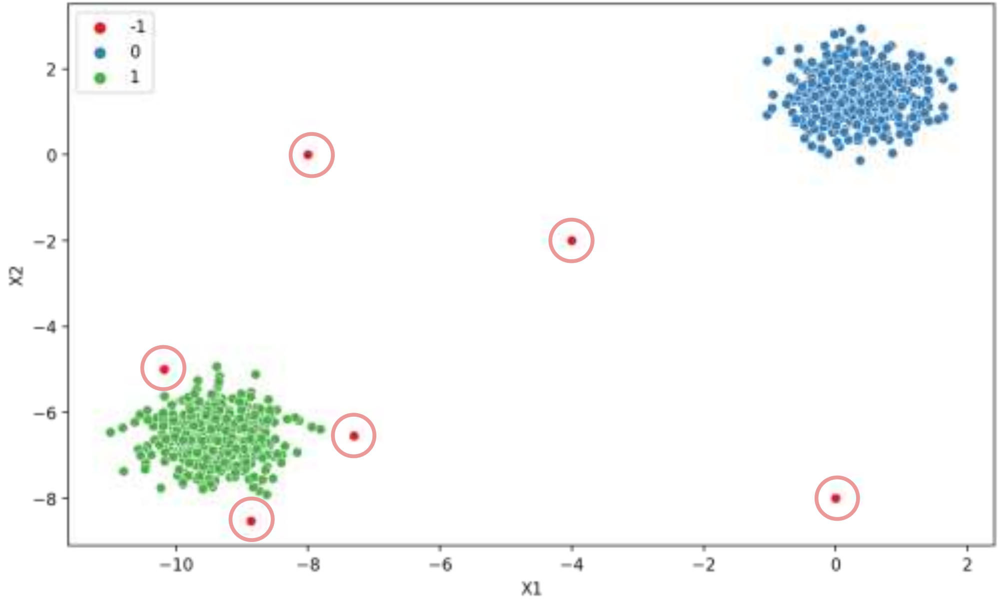
# DBSCAN

- **DBSCAN Results:**



# DBSCAN

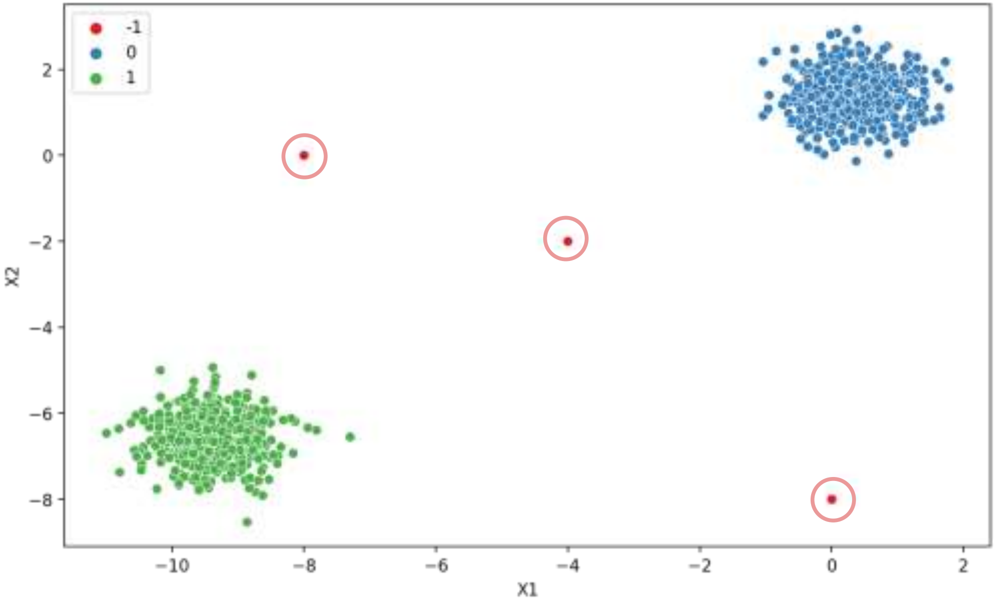
- DBSCAN Results:




PIERIAN DATA

# DBSCAN

- DBSCAN Results:



# DBSCAN

- Epsilon Intuition:
    - Increasing epsilon allows more points to be **core** points which also results in more **border** points and less outlier points.
    - Imagine a huge epsilon, all points would be within the neighborhood and classified as the same cluster!

# DBSCAN

- Epsilon Intuition:
  - Decreasing epsilon causes more points not to be in range of each other, creating more unique clusters.
  - Imagine a tiny epsilon, the range would not extend far out enough to come into contact with any other points!

# DBSCAN

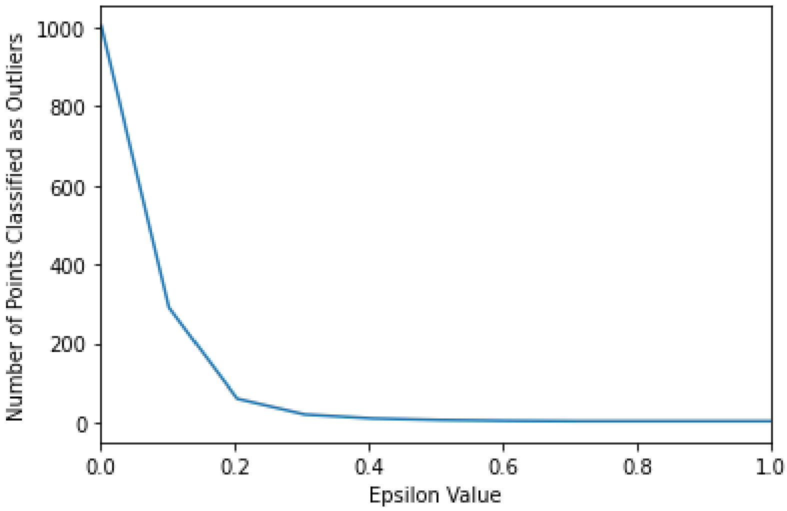
- Methods for finding an epsilon value:
    - Run multiple DBSCAN models varying epsilon and measure:
        - Number of Clusters
        - Number of Outliers
        - Percentage of Outliers

# DBSCAN

- Methods for finding an epsilon value:
    - Extremely dependent on the particular data set and domain space.
    - Requires user to have some expectation or intuition about number of clusters and relative percentage of outliers.

# DBSCAN

- Plot "elbow/knee" diagram comparing epsilon values:



# DBSCAN

- Minimum Number of Samples/Points:
    - Number of samples in a neighborhood for a point to be considered as a **core** point (including the point itself).

# DBSCAN

- Min. Number of Samples Intuition:
    - Increasing to a larger number of samples needed to be considered a core point, causes more points to be considered unique outliers.

# DBSCAN

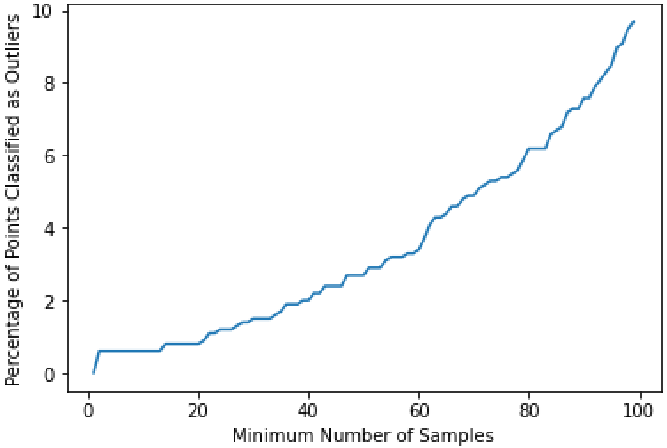
- Min. Number of Samples Intuition:
    - Imagine if min. number of samples was close to total number of points available, then very likely all points would become outliers.

# DBSCAN

- Choosing Min. Number of Samples:
  - Test multiple potential values and chart against number of outliers labeled.

# DBSCAN

- Choosing Min. Number of Samples:
  - Test multiple potential values and chart against number of outliers labeled.

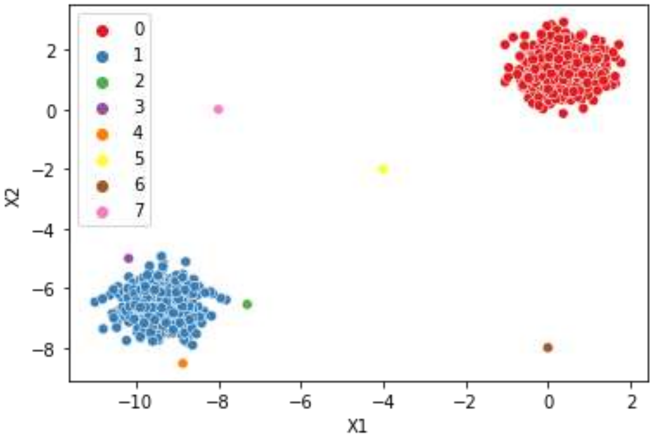


# DBSCAN

- Min. Number of Samples Note:
  - Useful to increase to create potential new small clusters, instead of complete outliers.

# DBSCAN

- Min. Number of Samples Note:
    - Useful to increase to create potential new small clusters, instead of complete outliers.



# DBSCAN

- Let’s continue by exploring hyperparameters with code and data examples!

# DBSCAN

Hyperparameter Search

# DBSCAN

Project Exercise Overview

# DBSCAN

Project Exercise Solutions