GENERAL ASSEMBLY

# DBSCAN

*Matt Brems*

*Data Science Immersive, GA*

# LEARNING OBJECTIVES

- By the end of this lesson, students should be able to:
    - Describe the effect of `epsilon` and `min_points` on DBSCAN.
    - Implement DBSCAN.
    - Identify advantages and disadvantages of DBSCAN.

# K-MEANS

- In unsupervised learning, one strategy is to cluster observations into groups
- Observations in the same group are more similar than observations in different groups
- So far, you've learned how to cluster using *k*-Means

# K-MEANS

- What are the pros/cons to using *k*-Means?

# DBSCAN

- There's another method of clustering that can sidestep some of the disadvantages of *k*-Means: **DBSCAN**

  - Density-Based Spatial Clustering of Applications with Noise

  - We can detect areas of high and low density
    - Areas of high density will become a cluster
    - Areas of low density will be **not** clustered/regarded as *noise*

# DBSCAN

- DBSCAN requires you to specify two hyperparameters:
  - **`min_samples`**: the minimum number of points needed to form a cluster.
  - **`epsilon`**: the “searching” distance when attempting to build a cluster.

# HOW DOES DBSCAN WORK?

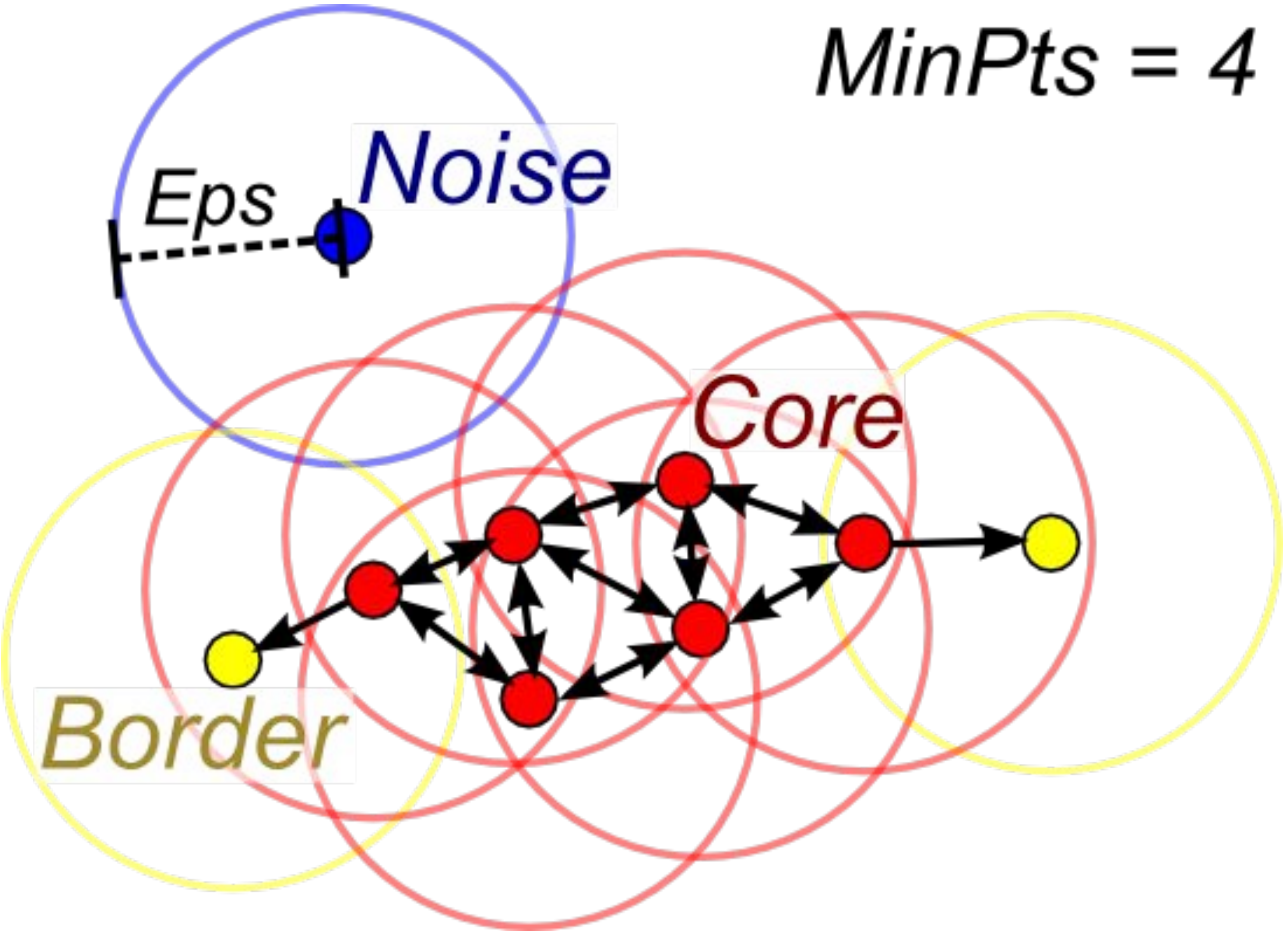
```
DBSCAN(DB, distFunc, eps, minPts)
   C = 0
   for each point P in database DB
      if label(P) ≠ undefined then continue
      Neighbors N = RangeQuery(DB, distFunc, P, eps)
      if |N| < minPts then
         label(P) = Noise
         continue
      C = C + 1
      label(P) = C
      Seed set S = N \ {P}
      for each point Q in S
         if label(Q) = Noise then label(Q) = C
         if label(Q) ≠ undefined then continue
         label(Q) = C
         Neighbors N = RangeQuery(DB, distFunc, Q, eps)
         if |N| ≥ minPts then
            S = S ∪ N
```

Source: https://en.wikipedia.org/wiki/DBSCAN#Algorithm

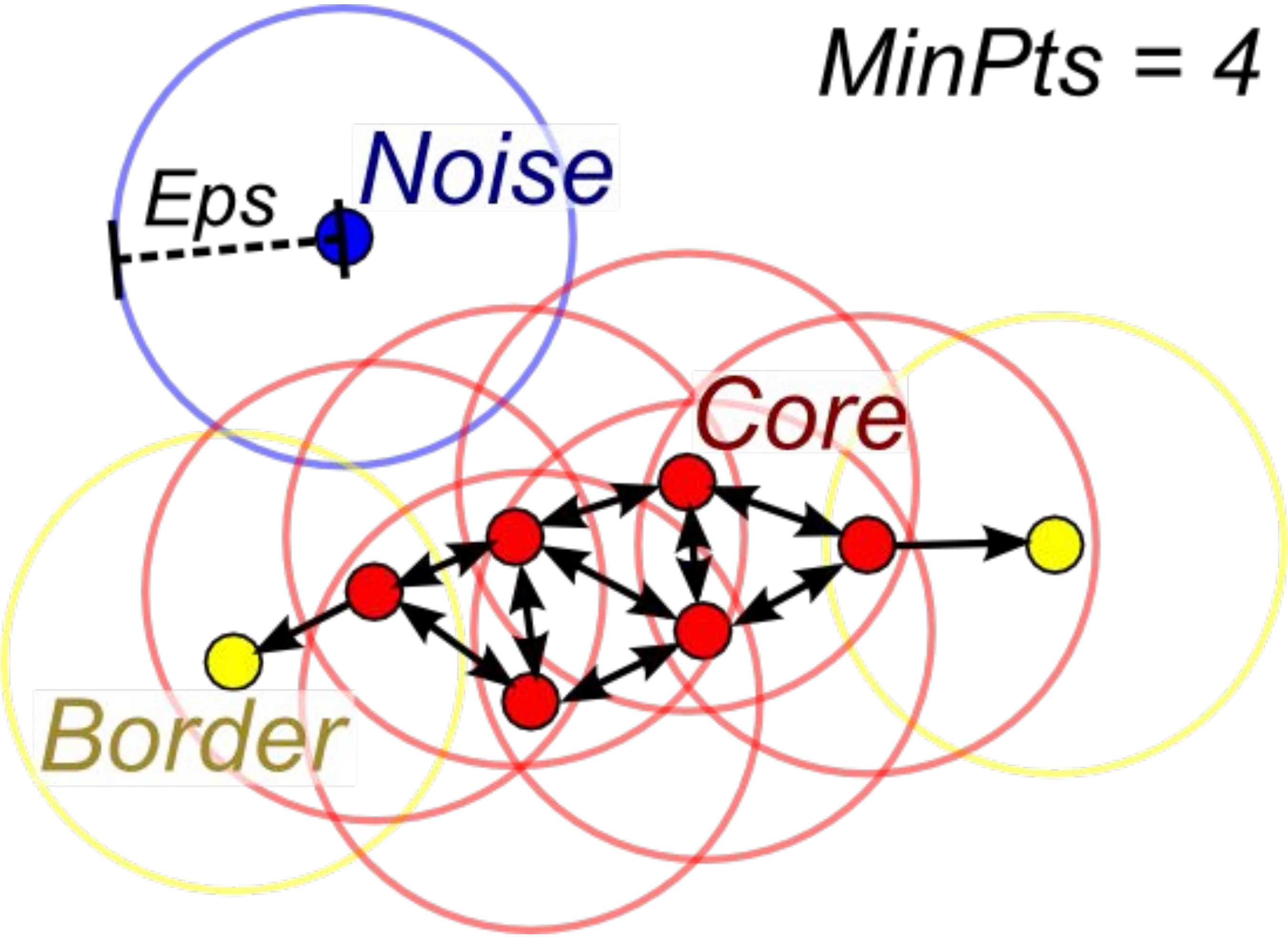
# VISUALIZING DBSCAN

- https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

# VISUALIZING DBSCAN



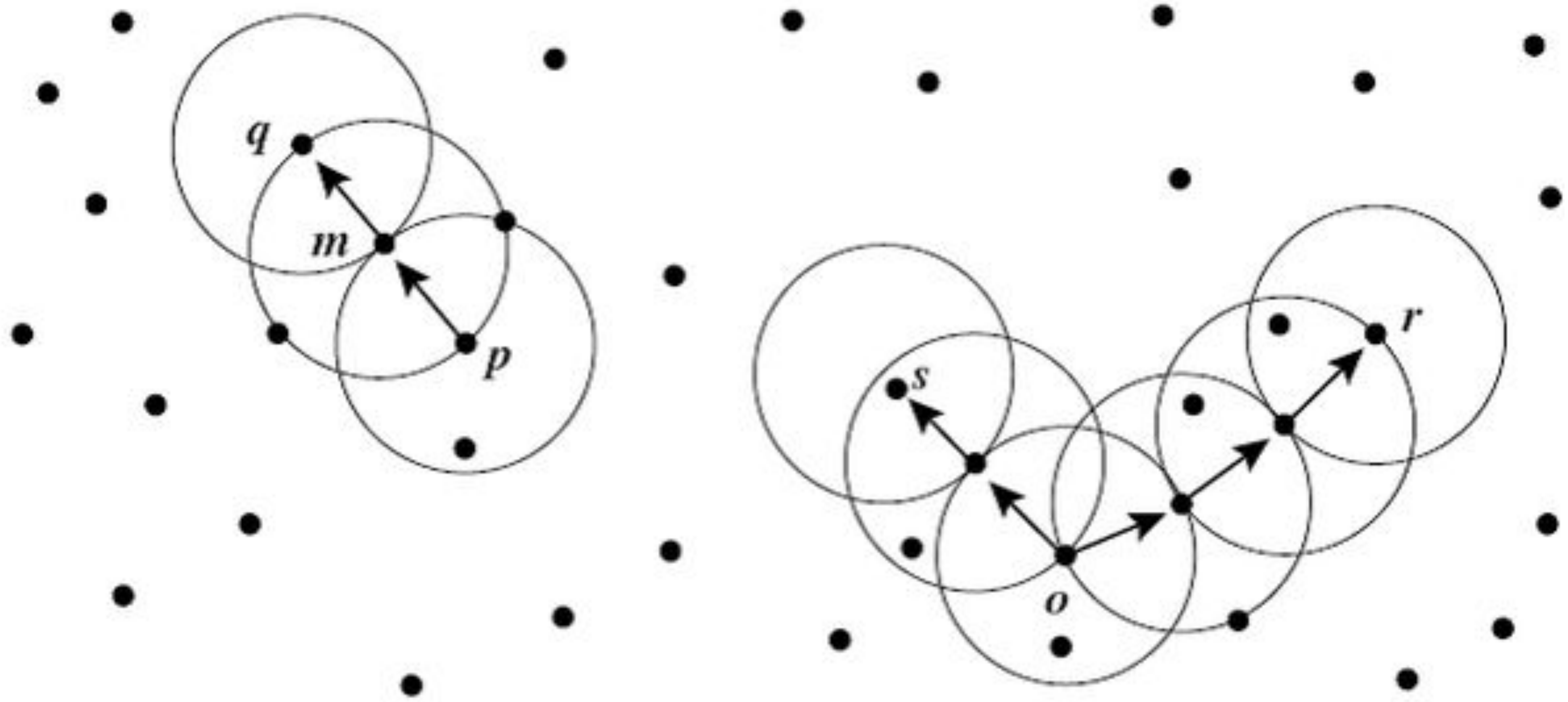
# VISUALIZING DBSCAN



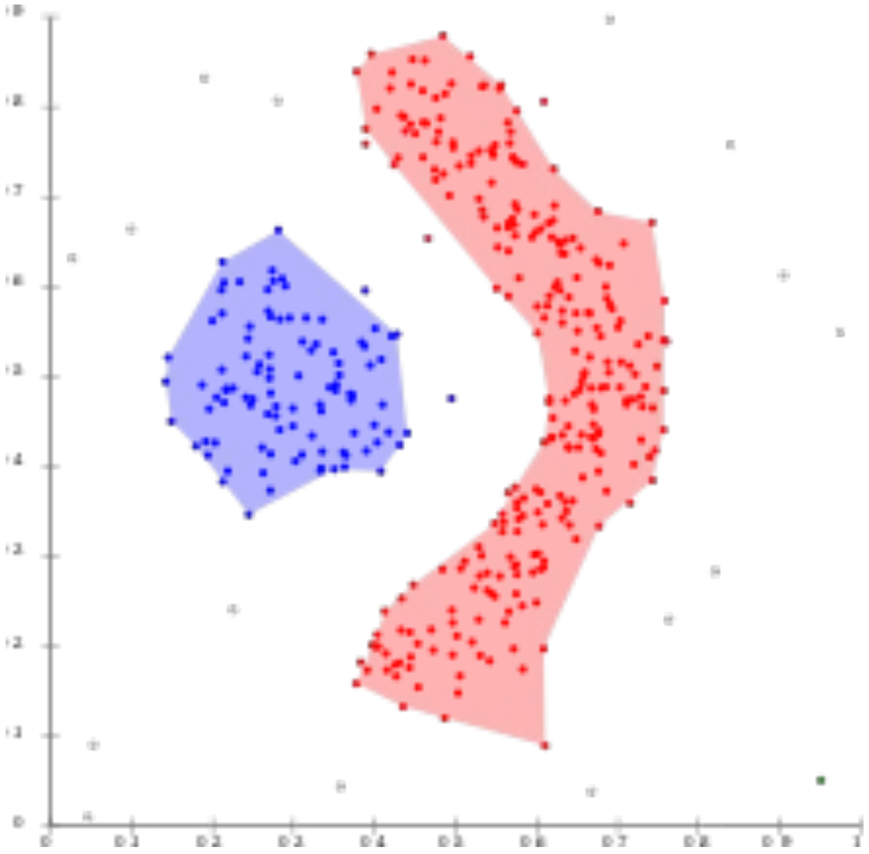

- **Core points**: Points inside a cluster that have at least `min_samples` points within `epsilon`.
- **Border points**: Points inside a cluster that <u>do not</u> have at least `min_samples` points within `epsilon`.
- **Noise:** Points that belong to no cluster.

## WHY DBSCAN?

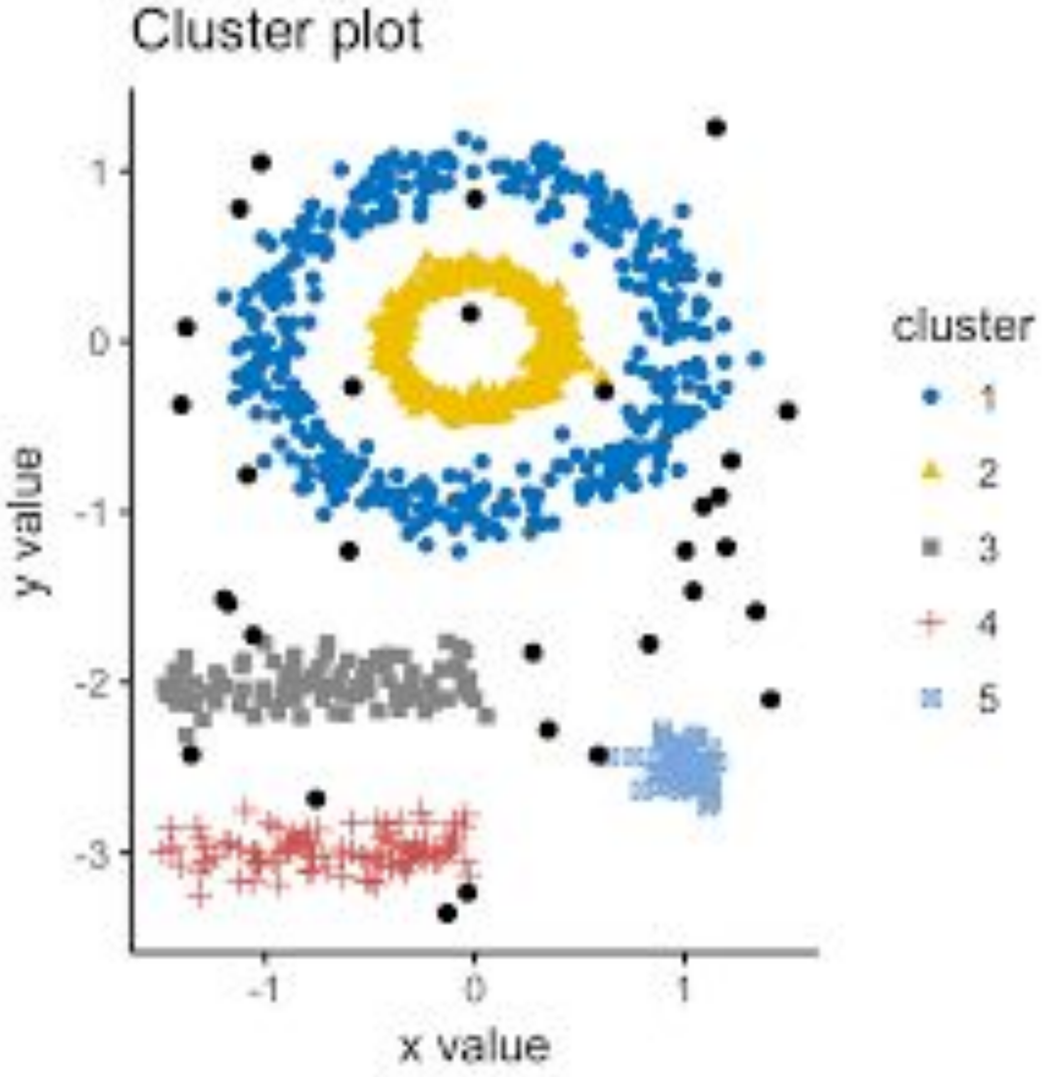
- DBSCAN allows us to detect some cluster patterns that *k*-Means might not be able to detect.



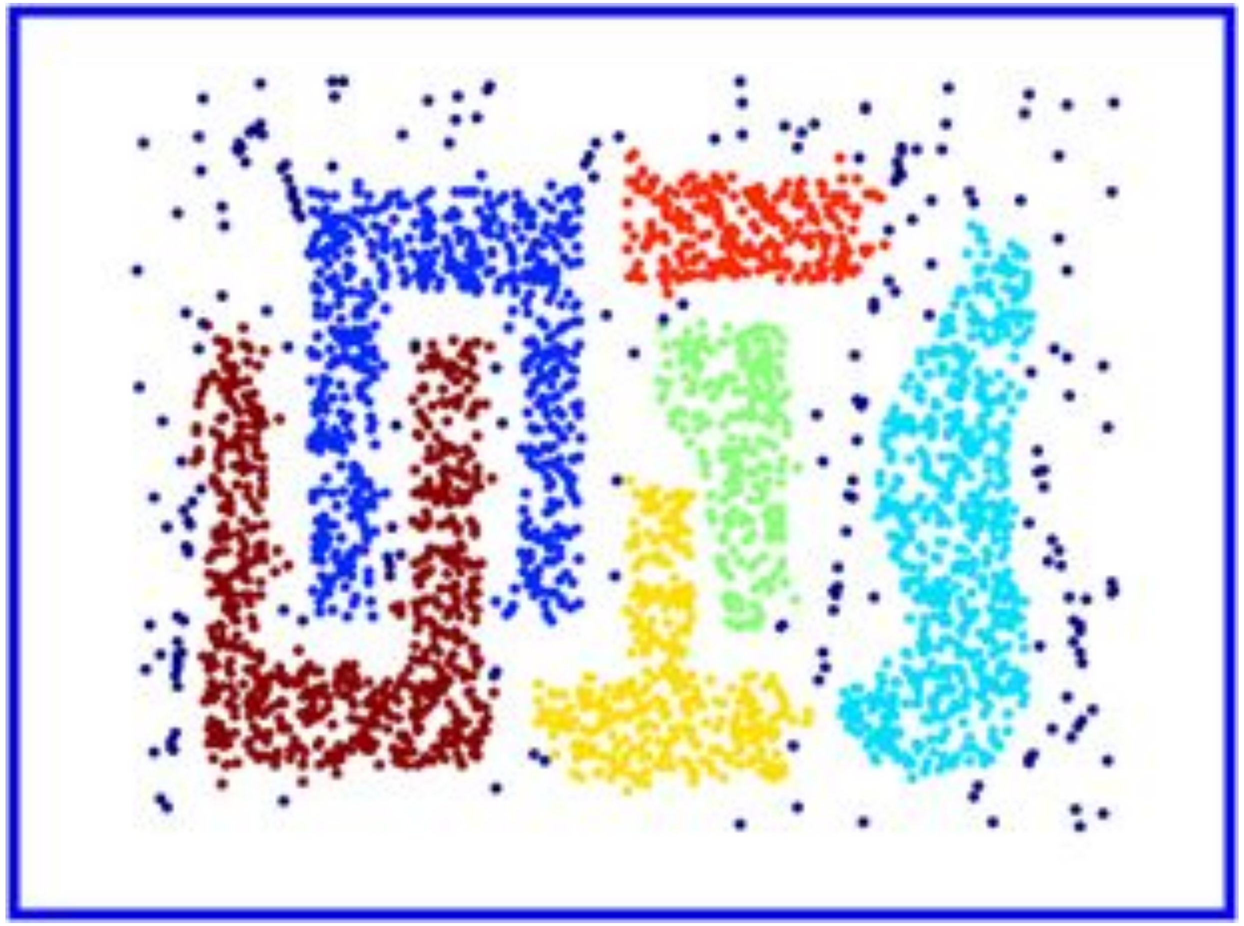
# WHY DBSCAN?



# WHY DBSCAN?



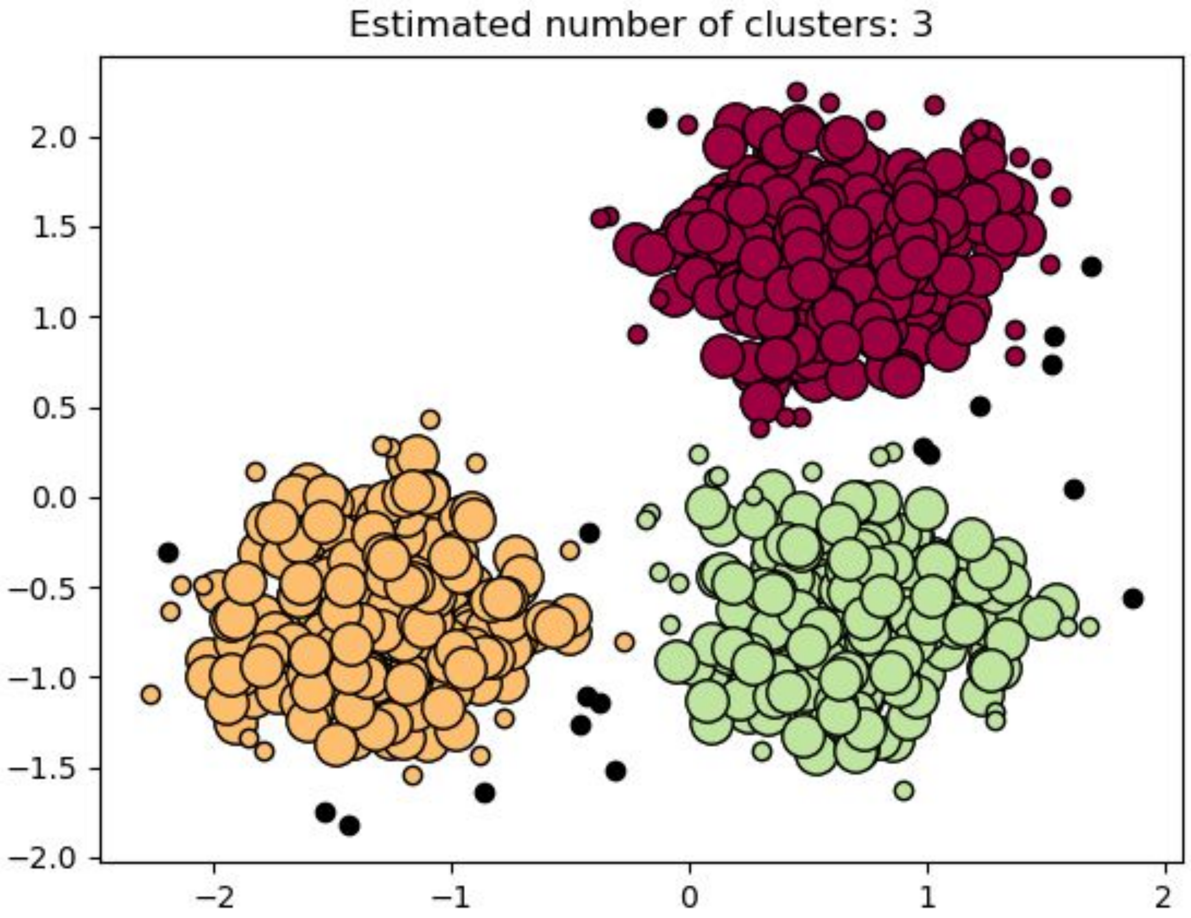
# WHY DBSCAN?



# WHY DBSCAN?

- DBSCAN allows us to detect some cluster patterns that k-Means might not be able to detect.
- We don’t need to pre-specify the number of clusters; the algorithm will determine how many clusters are appropriate given fixed `min_samples` and `epsilon` values.
  - This is particularly valuable when we are clustering data in more than two or three dimensions.
- Not every point is clustered!
  - Good for **identifying outliers**.

# WHY DBSCAN?



# DISADVANTAGES OF DBSCAN

- DBSCAN requires us to tune two parameters.

- DBSCAN works well when clusters are of a different density than the overall data, but does not work well when the clusters themselves are of varying density.
  - Fixed `epsilon`.