How DBSCAN works ?

Define the essential terms

1. **Eps (Epsilon) :** This is a **parameter** that defines the radius of a neighborhood around a data point. All points within this radius are considered neighbors of the point.

2. **Min_samples :** This is a **parameter** that defines the minimum number of data points (including the point itself) required within an eps radius for a point to be considered a **core point**.

3. **Core Point :** A data point is a **core point** if its eps neighborhood contains at least min_samples data points. These points are considered to be in the "interior" of a cluster.

4. **Border Point :** A data point is a **border point** if it has fewer than min_samples within its eps neighborhood, but it is located within the eps radius of a **core point**. Border points are on the edge of a cluster.

5. **Noise Point :** A data point is a **noise point** (or outlier) if it is neither a **core point** nor a **border point**. It is a point that does not belong to any cluster and exists in a low-density region of the data space.

Steps of DBSCAN ?

The working mechanism of DBSCAN can be described as follows:

- **Initialization** : This step involves setting the two critical parameters that the algorithm relies on to define a dense region: **eps** (the radius) and **min_samples** (the minimum number of points).

- **Core Point Identification** :This is where the algorithm starts its search. It randomly selects a point and checks if it's "dense" enough to be the starting point of a cluster. If the number of neighbors within its radius meets the min_samples threshold, the point is labeled a **core point**, and a new cluster is initiated.

- **Cluster Expansion** : From the initial core point, the algorithm begins to expand the cluster. It moves to a neighbor and checks its density. Points that are dense enough become new core points and expand the cluster further. Points that are in a core point's neighborhood but not dense enough themselves are labeled **border points**.

- **Cluster Completion** : The algorithm continues this expansion process, moving from one core point to the next, until all reachable points within that cluster have been identified and labeled. This ensures the entire high-density region is captured in a single cluster.

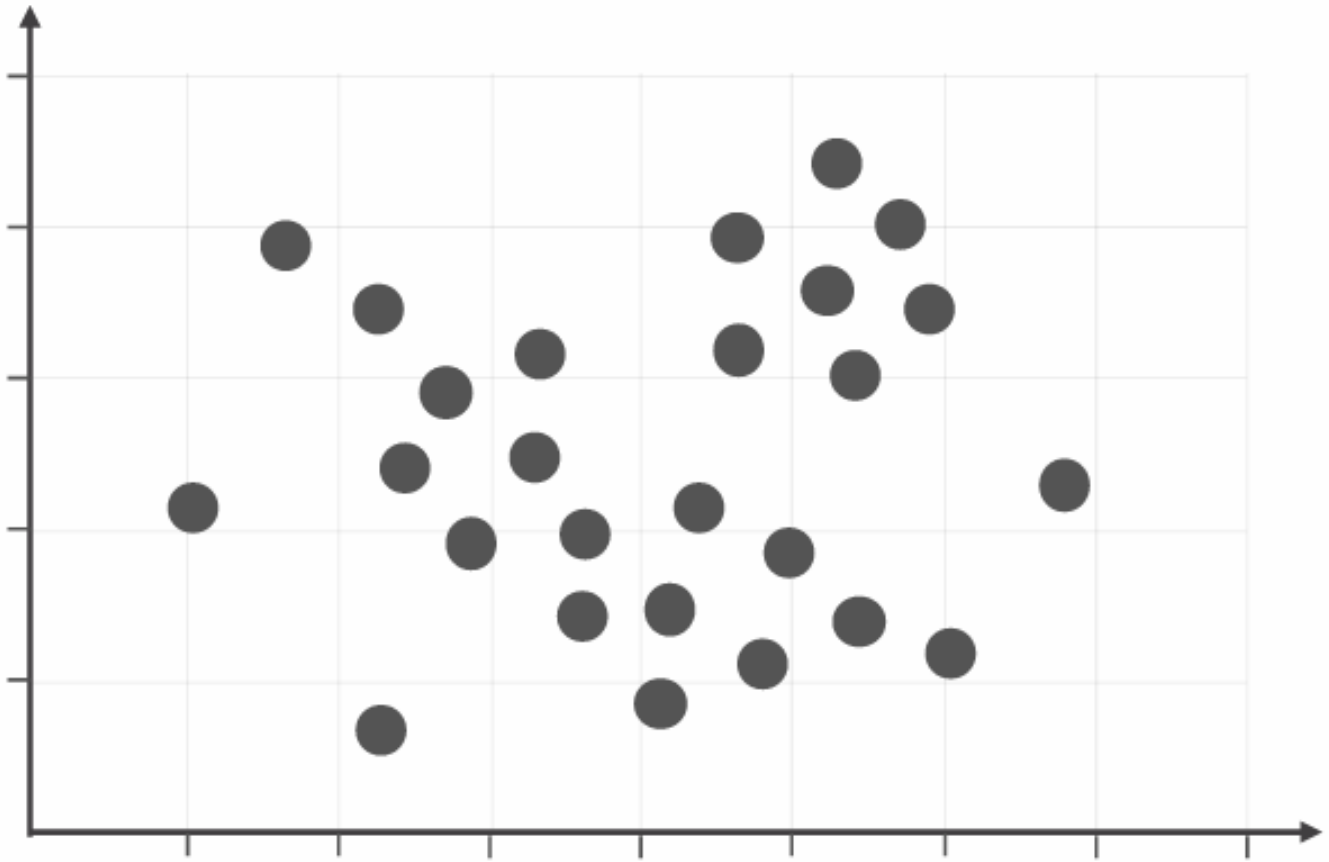
- **Iteration** : After a cluster is fully formed, the algorithm moves on to another unlabeled, randomly selected point in the dataset and repeats the process from step 2. This continues until all data points have been visited and assigned a label (either a cluster ID or "noise").

Step 1 - Initialization

Step 1 - Initialization

This step involves setting the two critical parameters that the algorithm relies on to define a dense region: **eps** (the radius) and **min_samples** (the minimum number of points).

Step 1 - Select a radius (eps) and minimum number of points (min_samples)



eps = 0.75
min_samples = 4

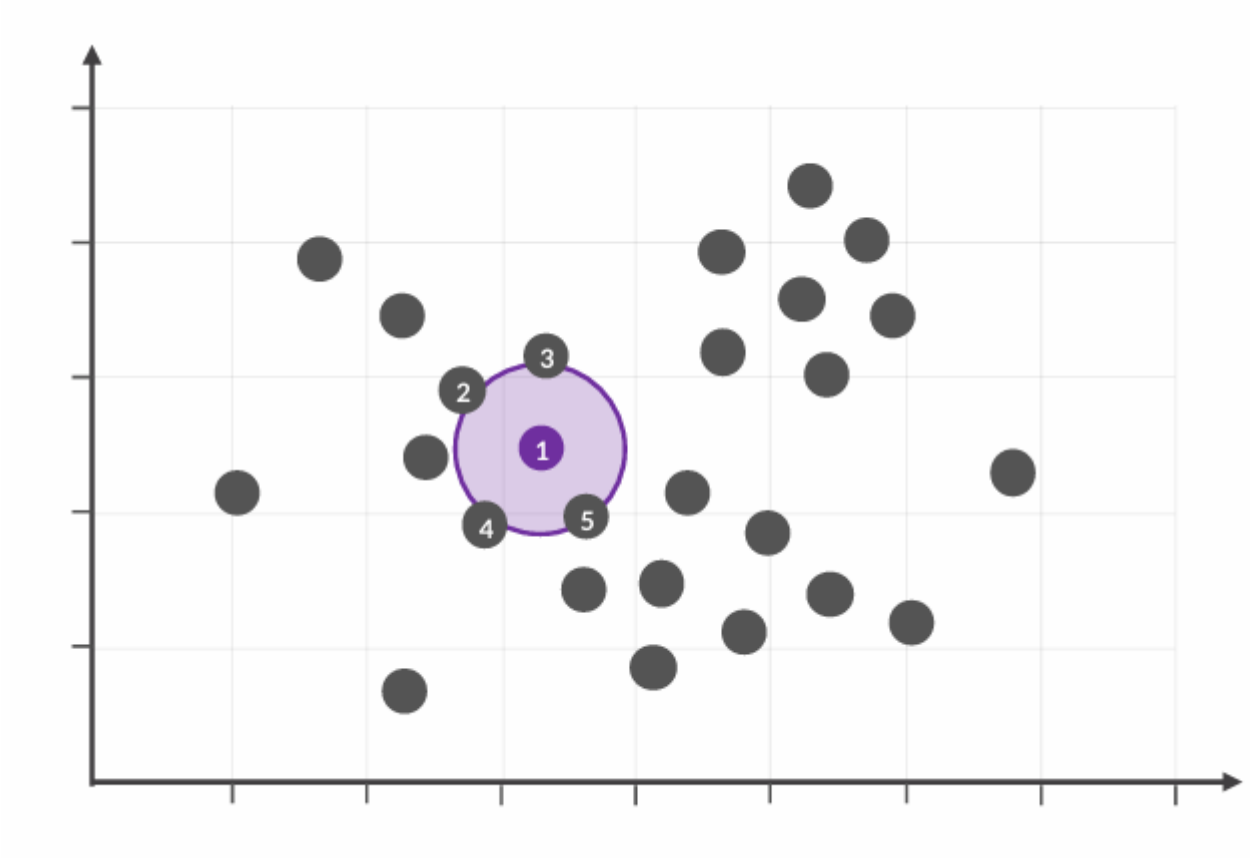
Step 2 – Core Point Identification

Step 2 – Core Point Identification

This is where the algorithm starts its search. It randomly selects a point and checks if it's "dense" enough to be the starting point of a cluster. If the number of neighbors within its radius meets the min_samples threshold, the point is labeled a **core point**, and a new cluster is initiated.

Step 2.1 - Select a point at random and count the points within its radius

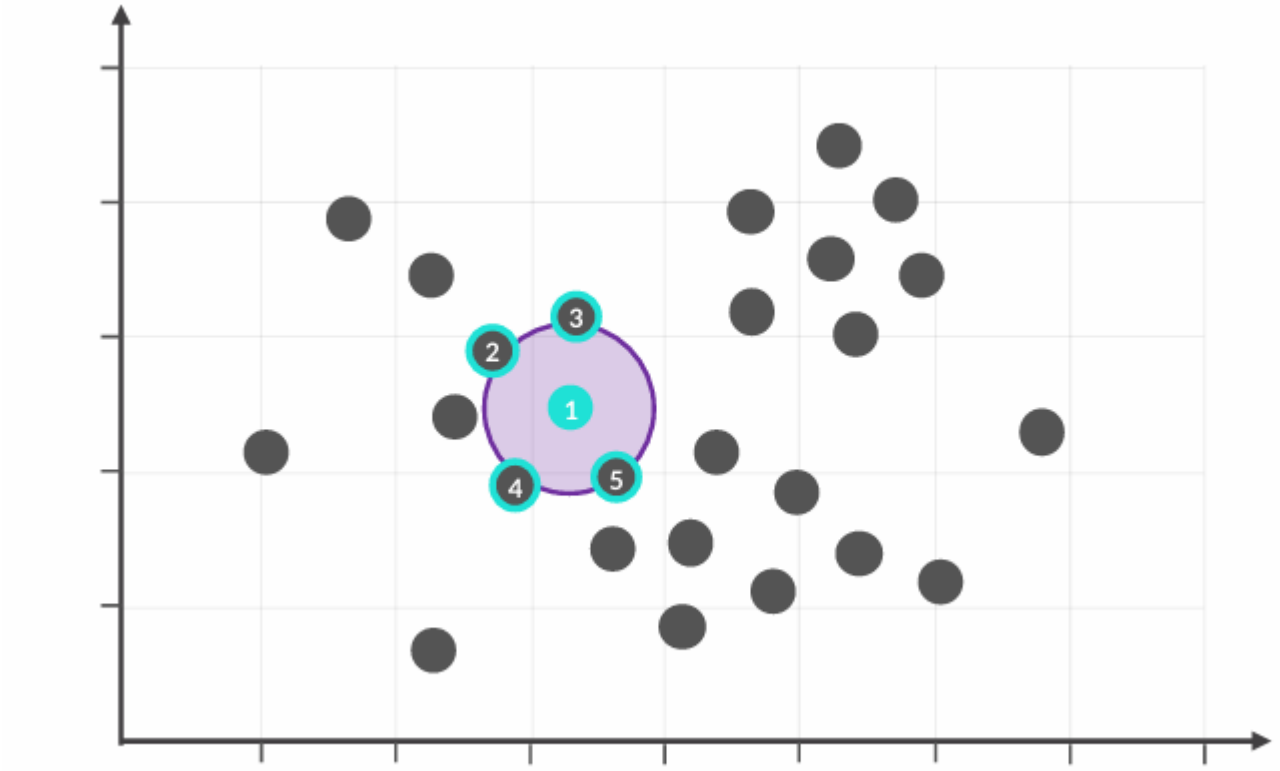
- If it's greater than or equal to the "min_samples", then start a cluster, label the point as a core point, and mark the points within its radius as a neighbor



eps = 0.75
min_samples = 4

Step 2.2 - Select a point at random and count the points within its radius

• If it's greater than or equal to the "min_samples", then start a cluster, label the point as a core point, and mark the points within its radius as a neighbor



eps = 0.75
min_samples = 4

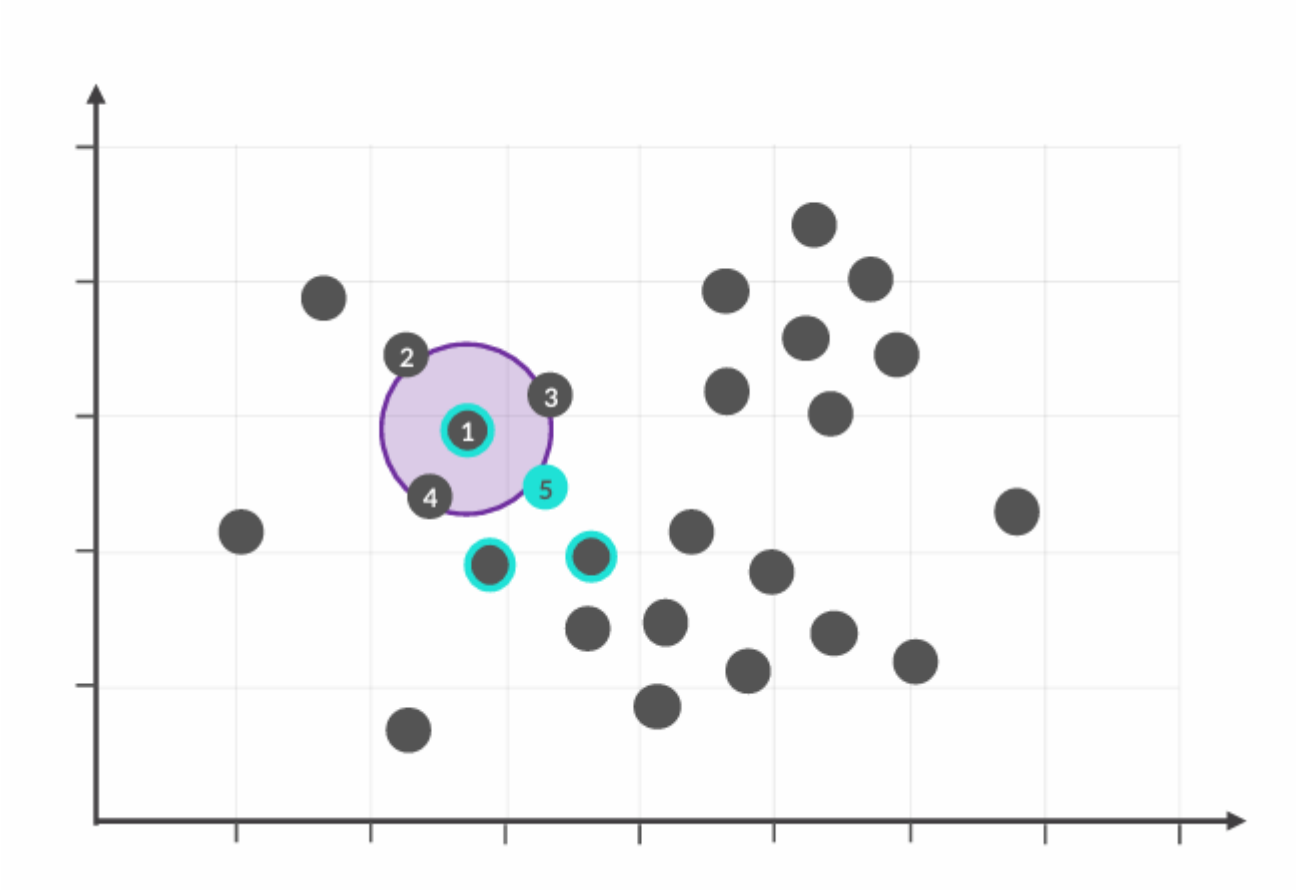
Step 3 – Cluster Expansion

Step 3 – Update Step

From the initial core point, the algorithm begins to expand the cluster. It moves to a neighbor and checks its density. Points that are dense enough become new core points and expand the cluster further. Points that are in a core point's neighborhood but not dense enough themselves are labeled **border points**.

Step 3.1 - Move to a neighbor and count the points within its radius

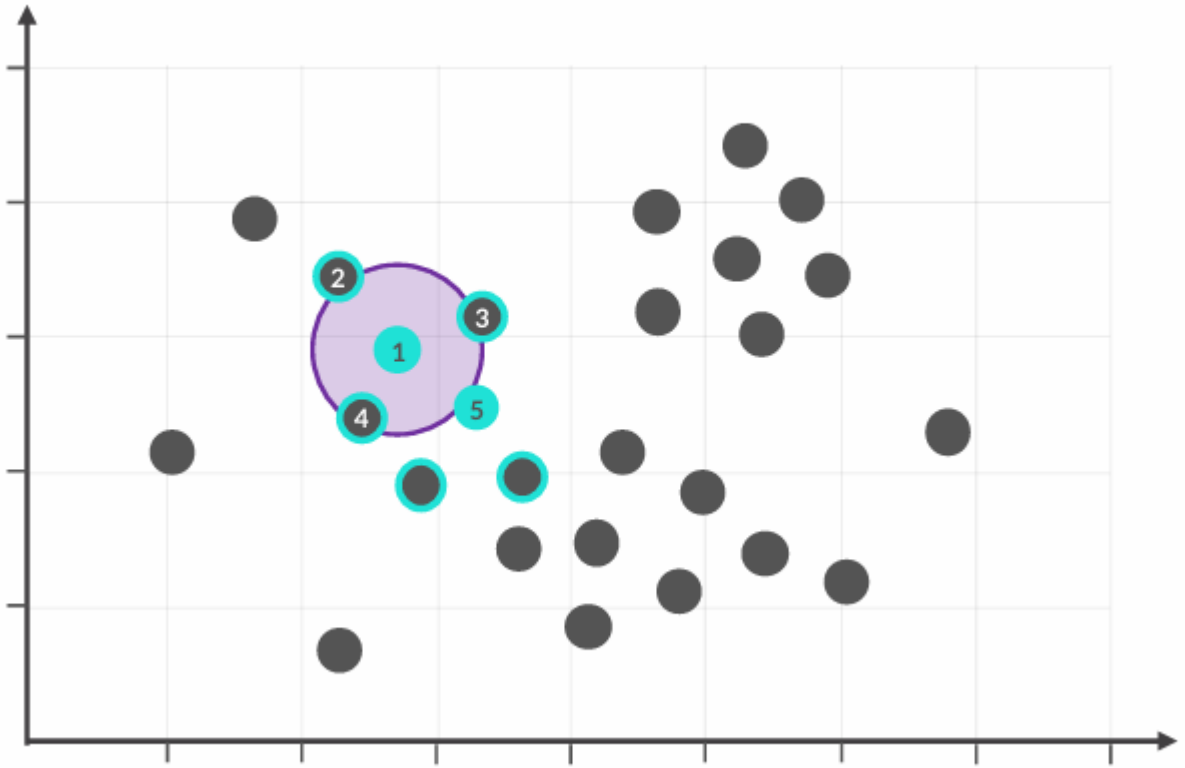
- If it's greater than or equal to the "min_samples", label it as a core point, and mark its neighbors



eps = 0.75
min_samples = 4

Step 3.2 - Move to a neighbor and count the points within its radius

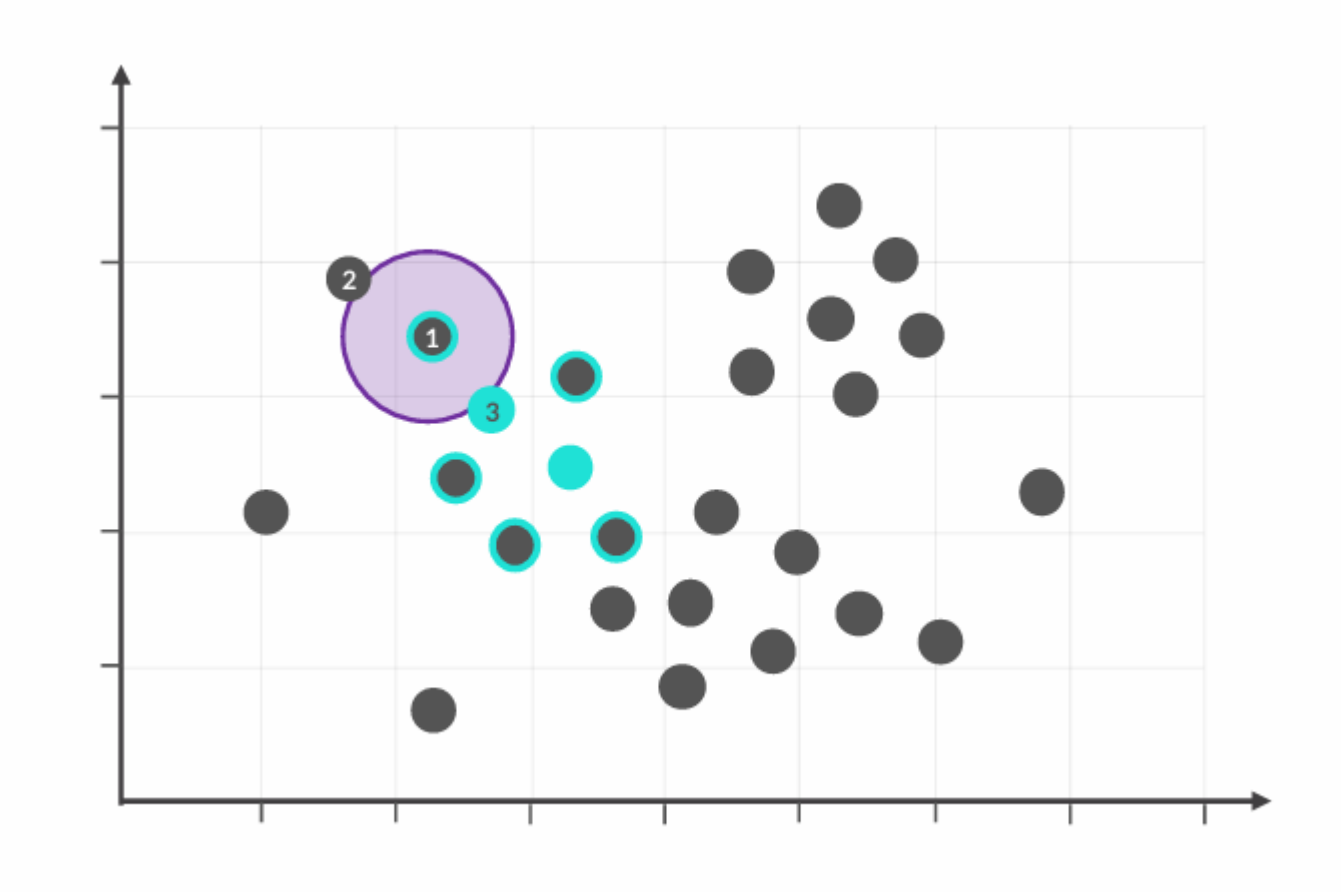
- If it's greater than or equal to the "min_samples", label it as a core point, and mark its neighbors



eps = 0.75
min_samples = 4

Step 3.3 - Move to a neighbor and count the points within its radius

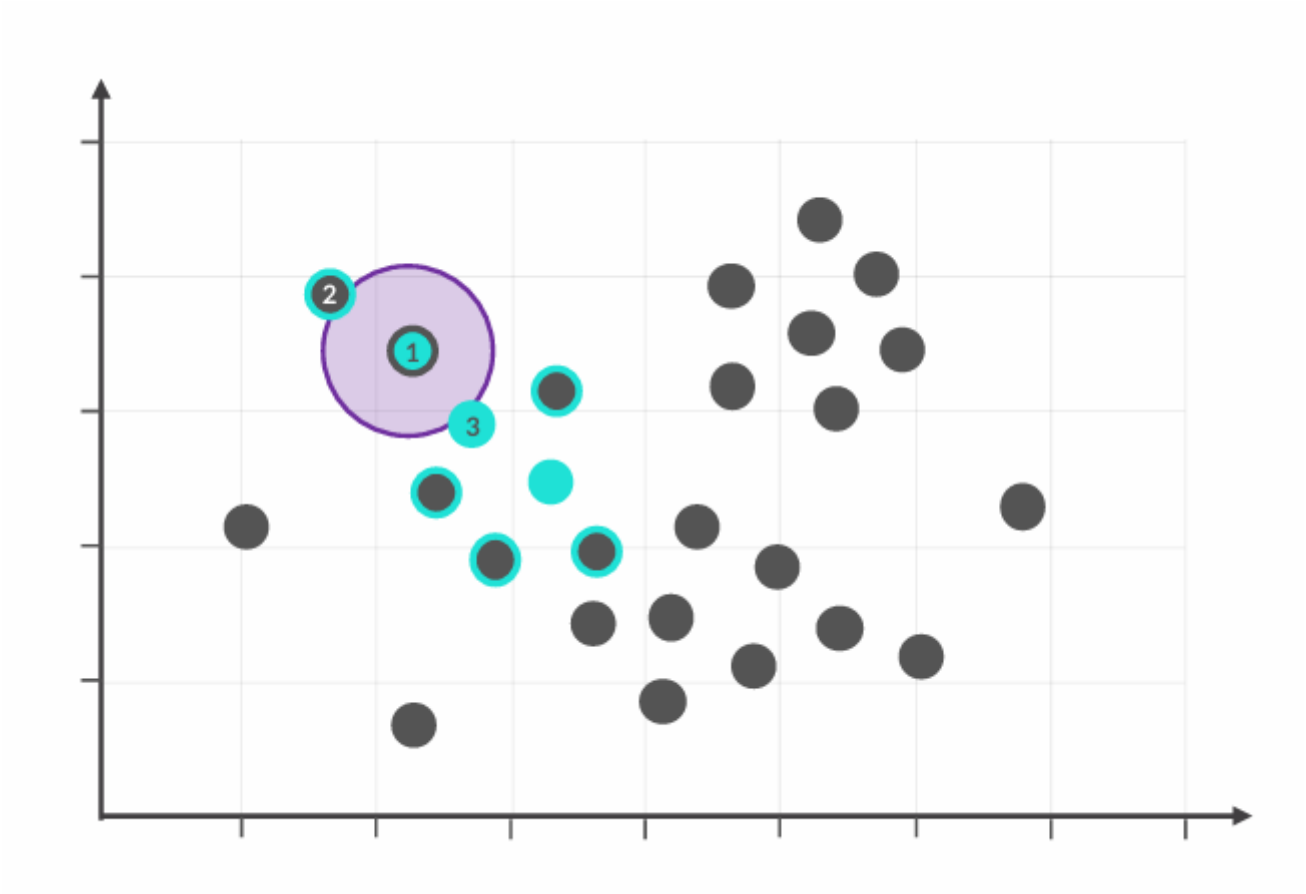
- If it's less than "min_samples", but at least one of them is a core point, label it as a border point



eps = 0.75
min_samples = 4

Step 3.4 - Move to a neighbor and count the points within its radius

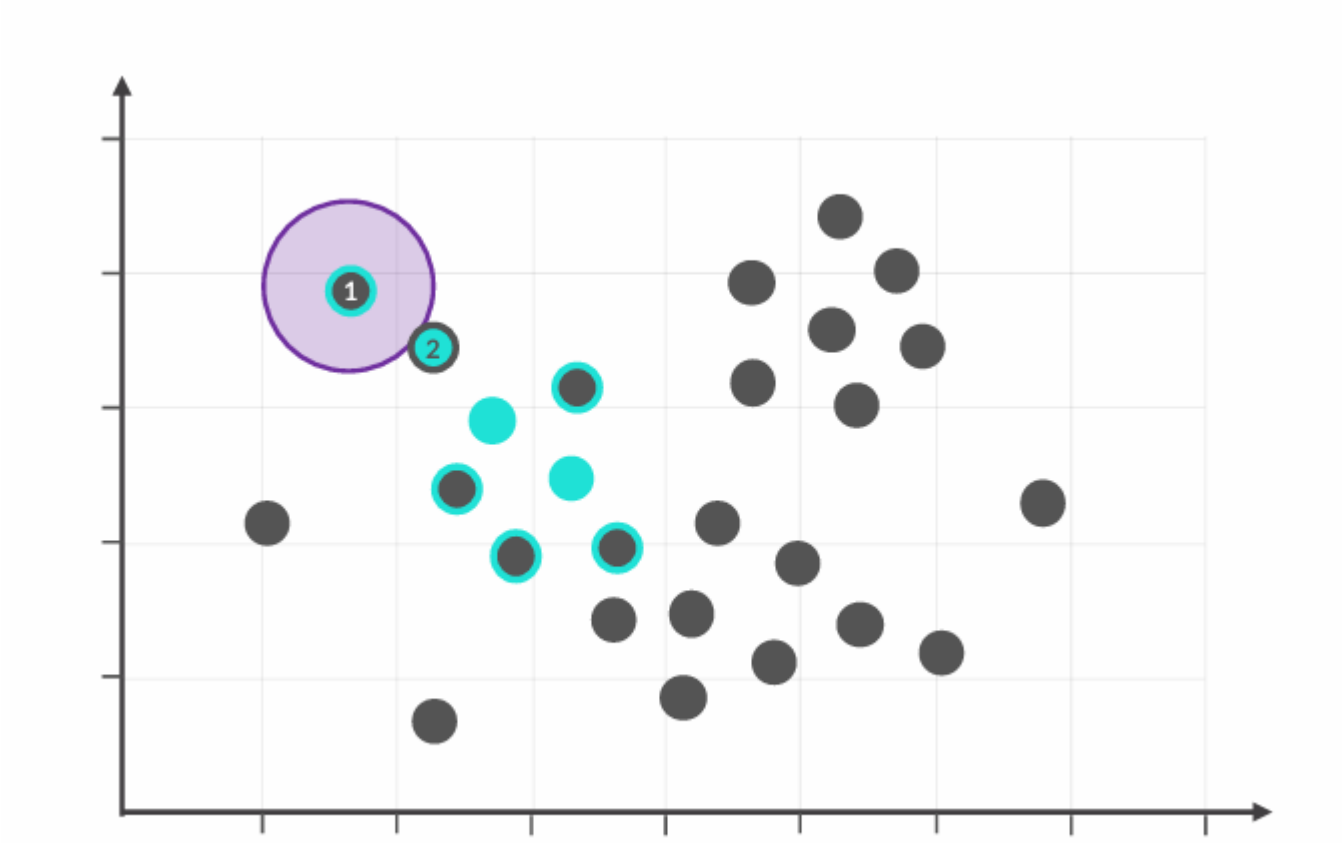
- If it's less than "min_samples", but at least one of them is a core point, label it as a border point



eps = 0.75
min_samples = 4

Step 3.5 - Move to a neighbor and count the points within its radius

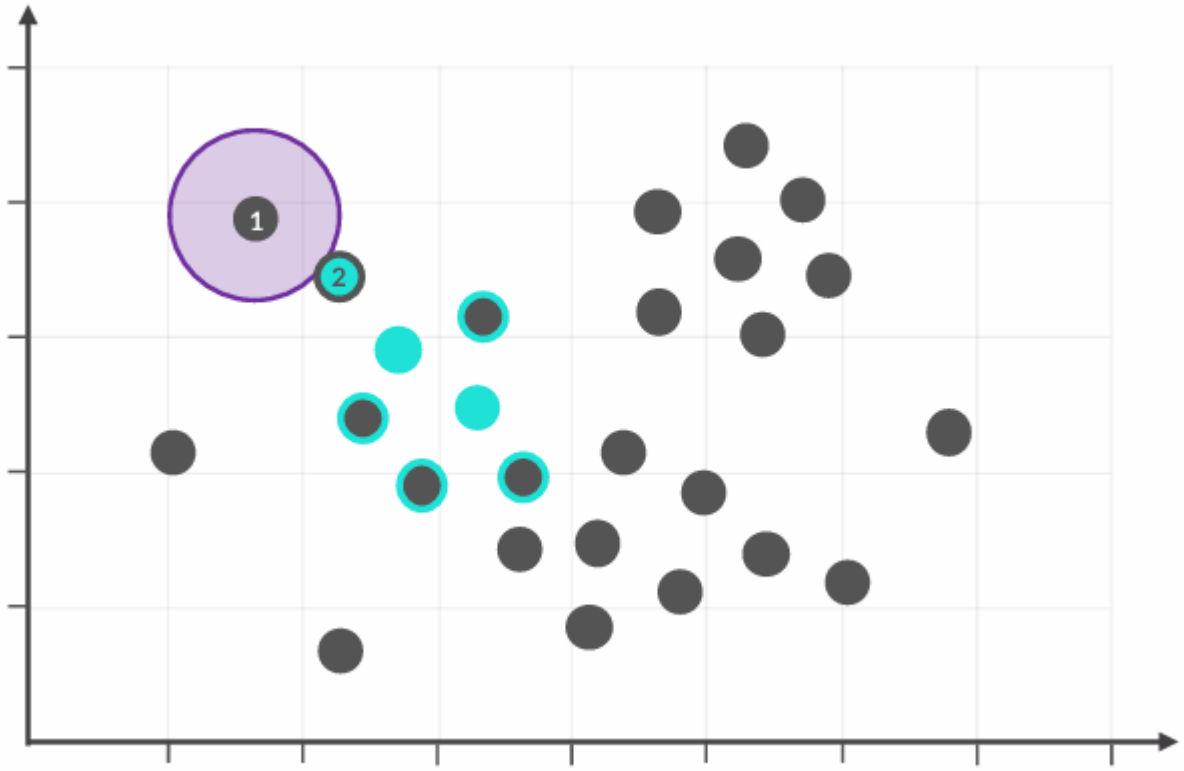
- If it's less than "min_samples", and none of them is a core point, label it as a noise point



eps = 0.75
min_samples = 4

Step 3.6 - Move to a neighbor and count the points within its radius

- If it's less than "min_samples", and none of them is a core point, label it as a noise point



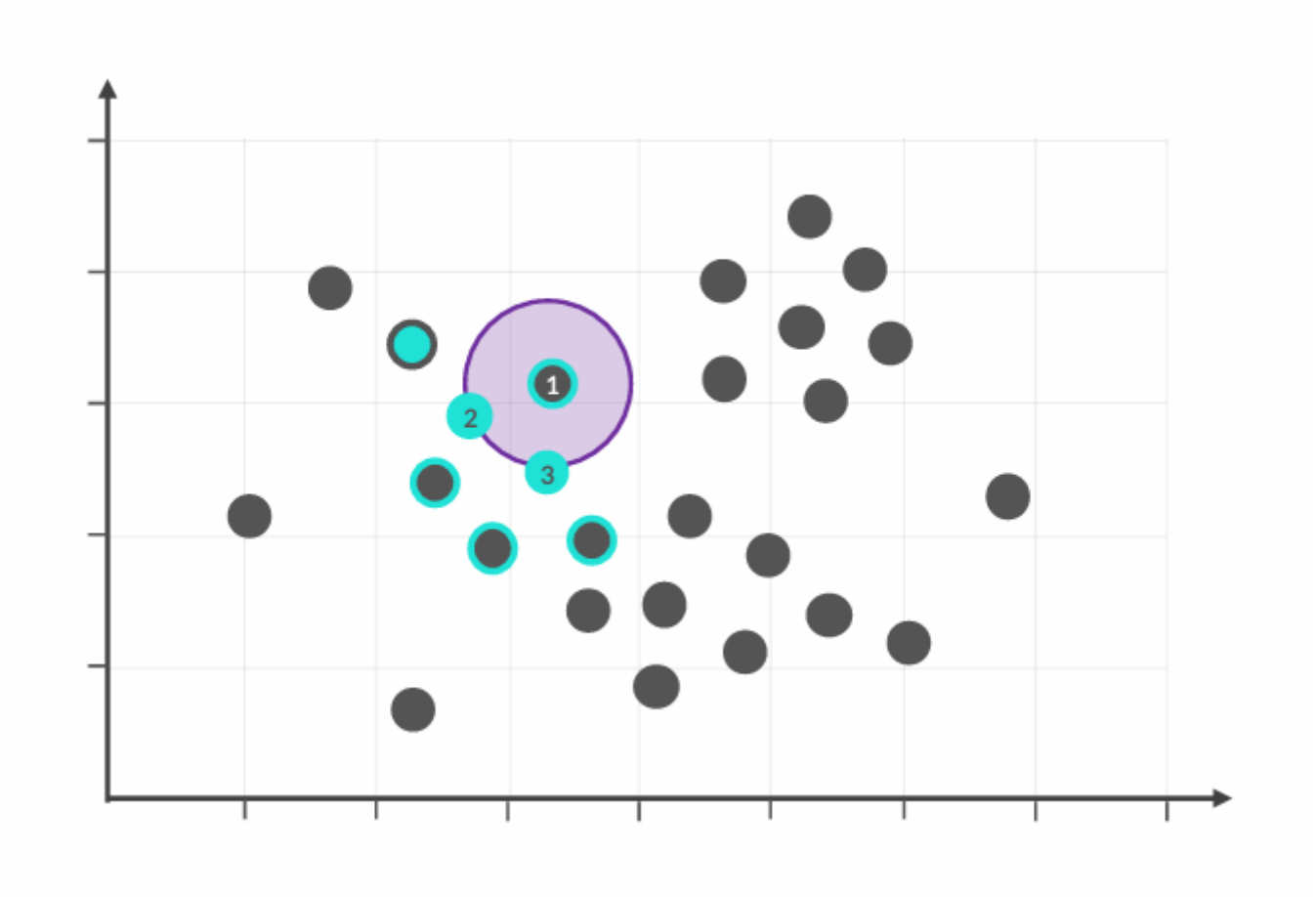
eps = 0.75
min_samples = 4

Step 4 - Cluster Completion

Step 4 - Cluster Completion

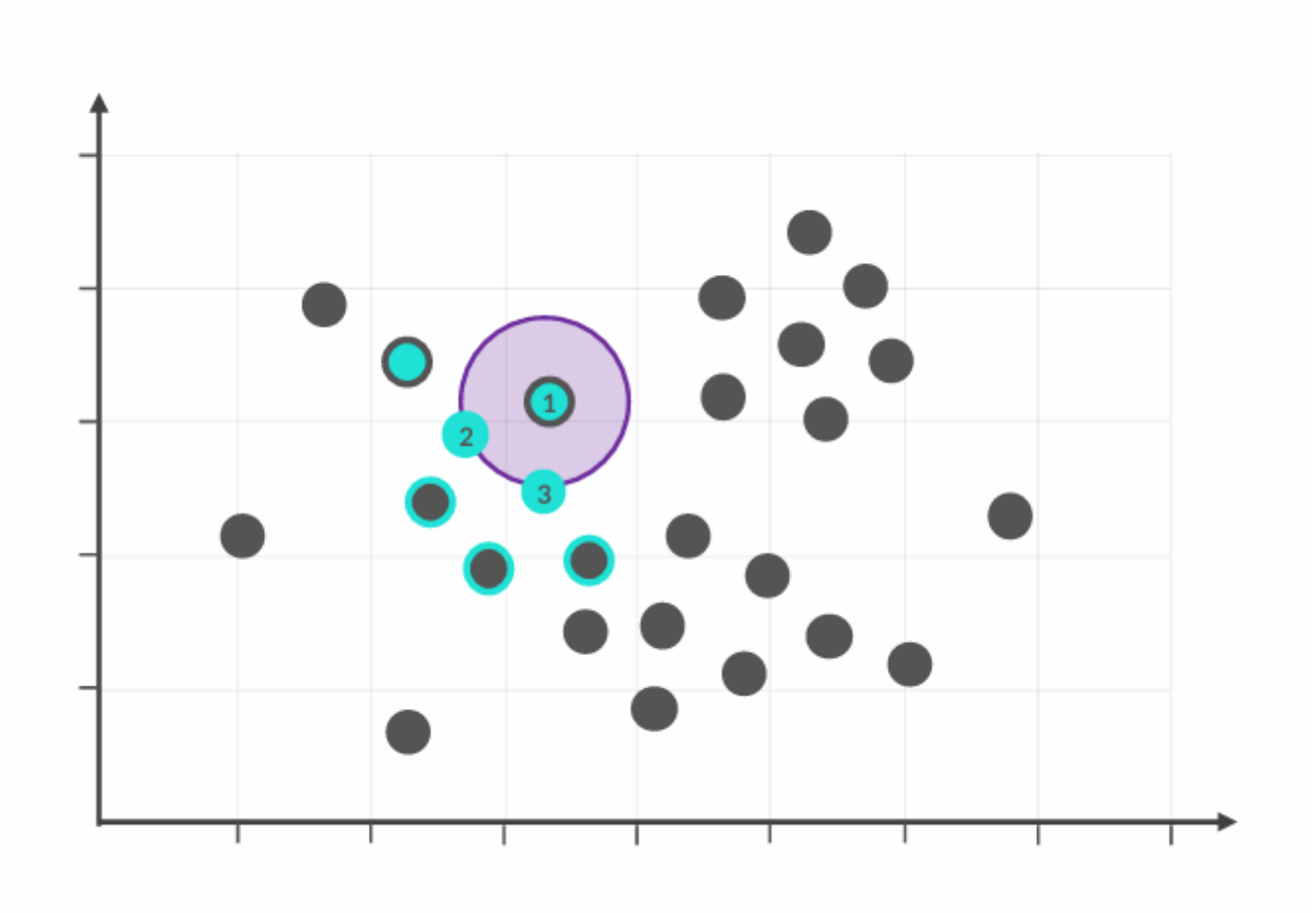
The algorithm continues this expansion process, moving from one core point to the next, until all reachable points within that cluster have been identified and labeled. This ensures the entire high-density region is captured in a single cluster.

Step 4.1 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



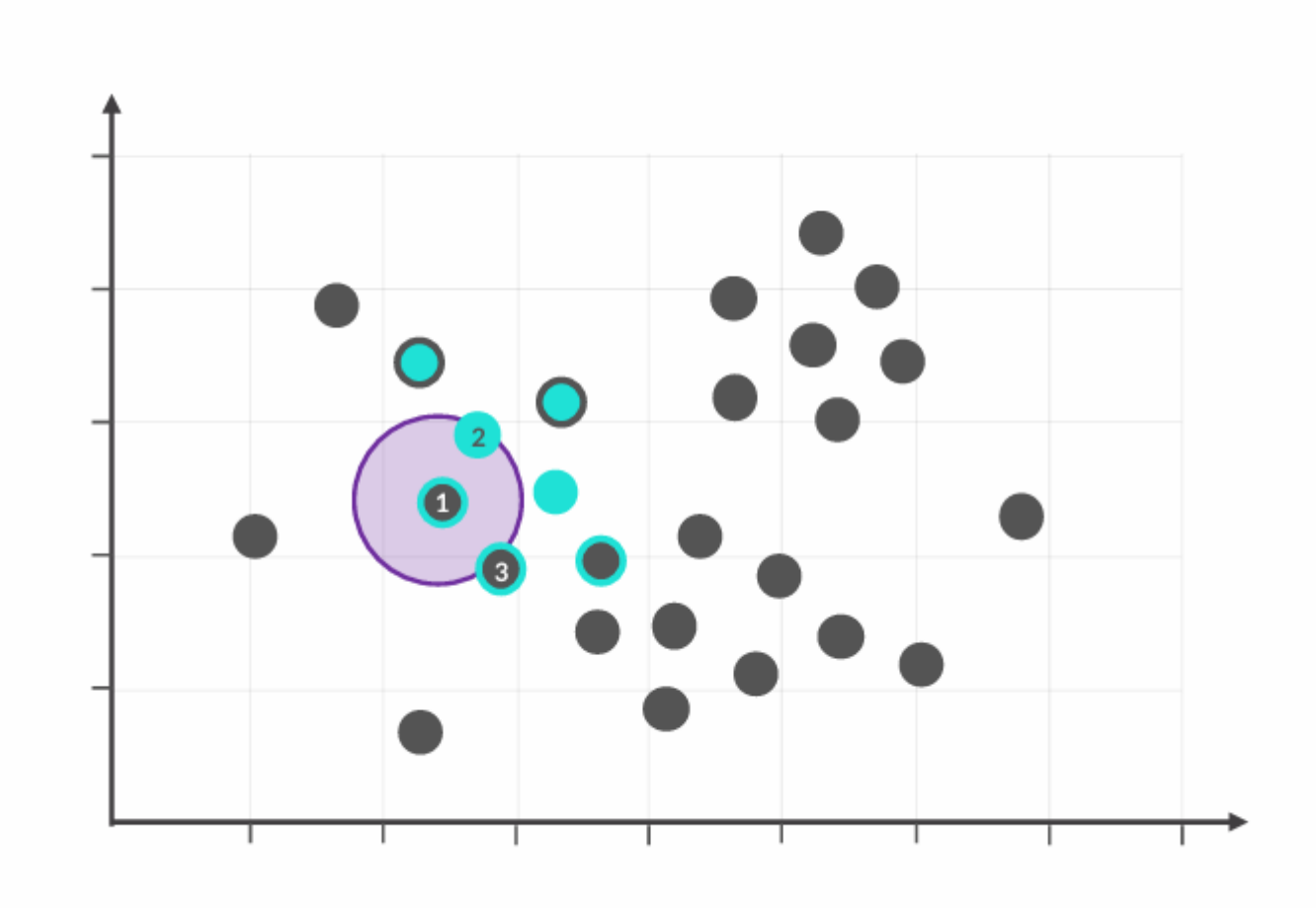
eps = 0.75
min_samples = 4

Step 4.2 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



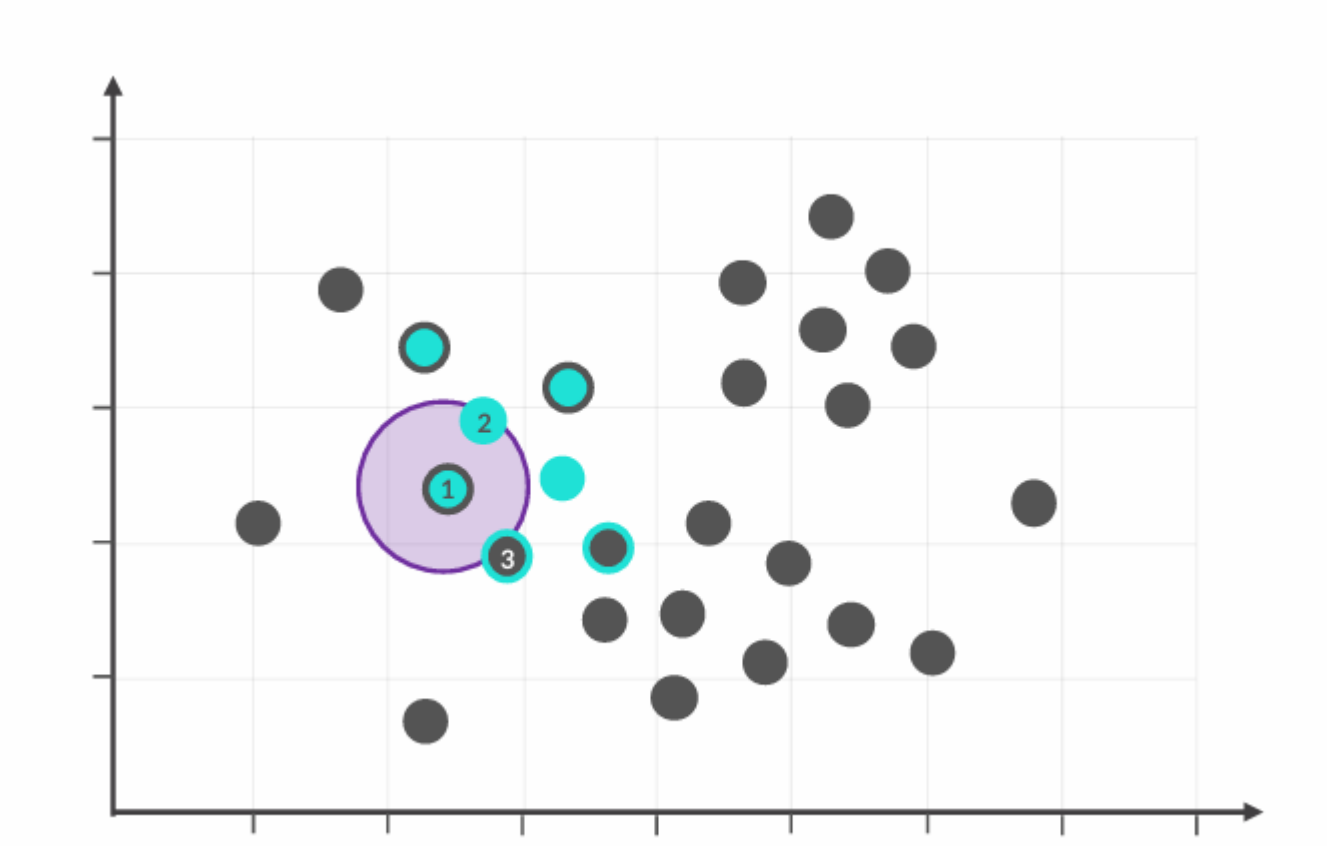
eps = 0.75
min_samples = 4

Step 4.3 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



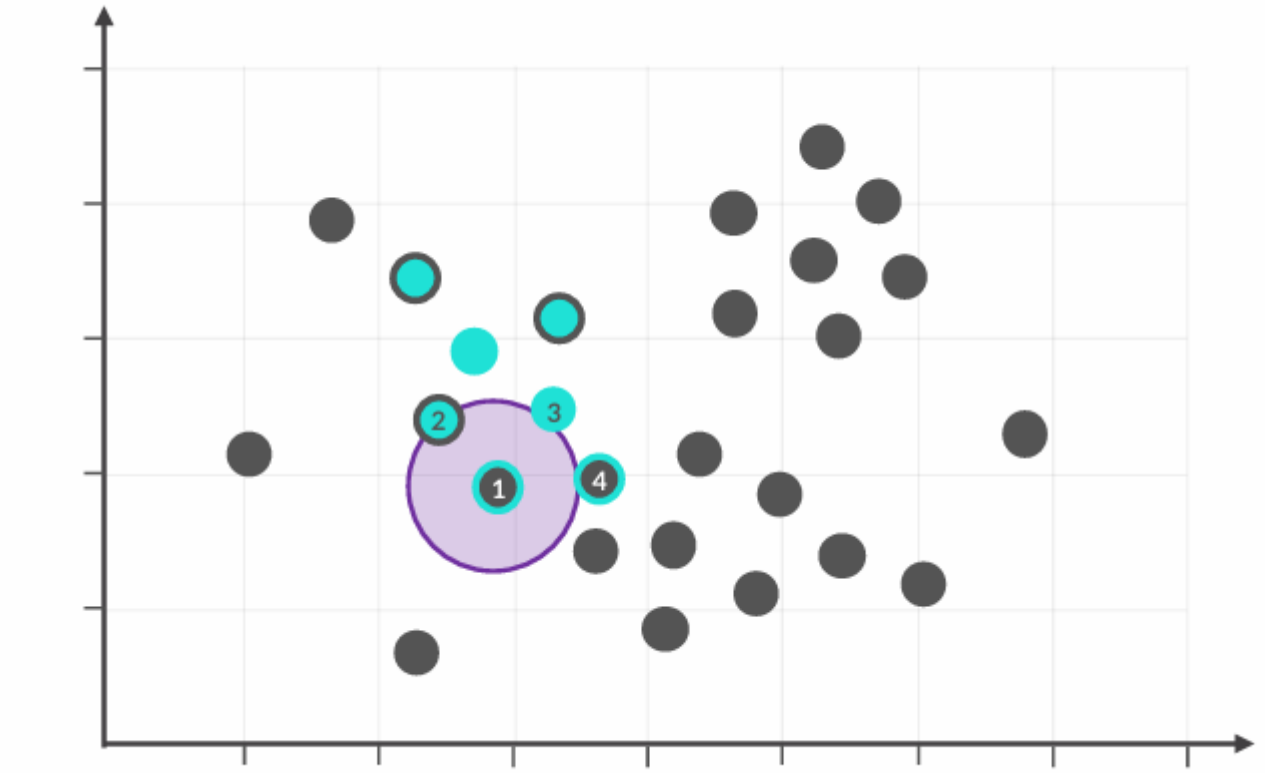
eps = 0.75
min_samples = 4

Step 4.4 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



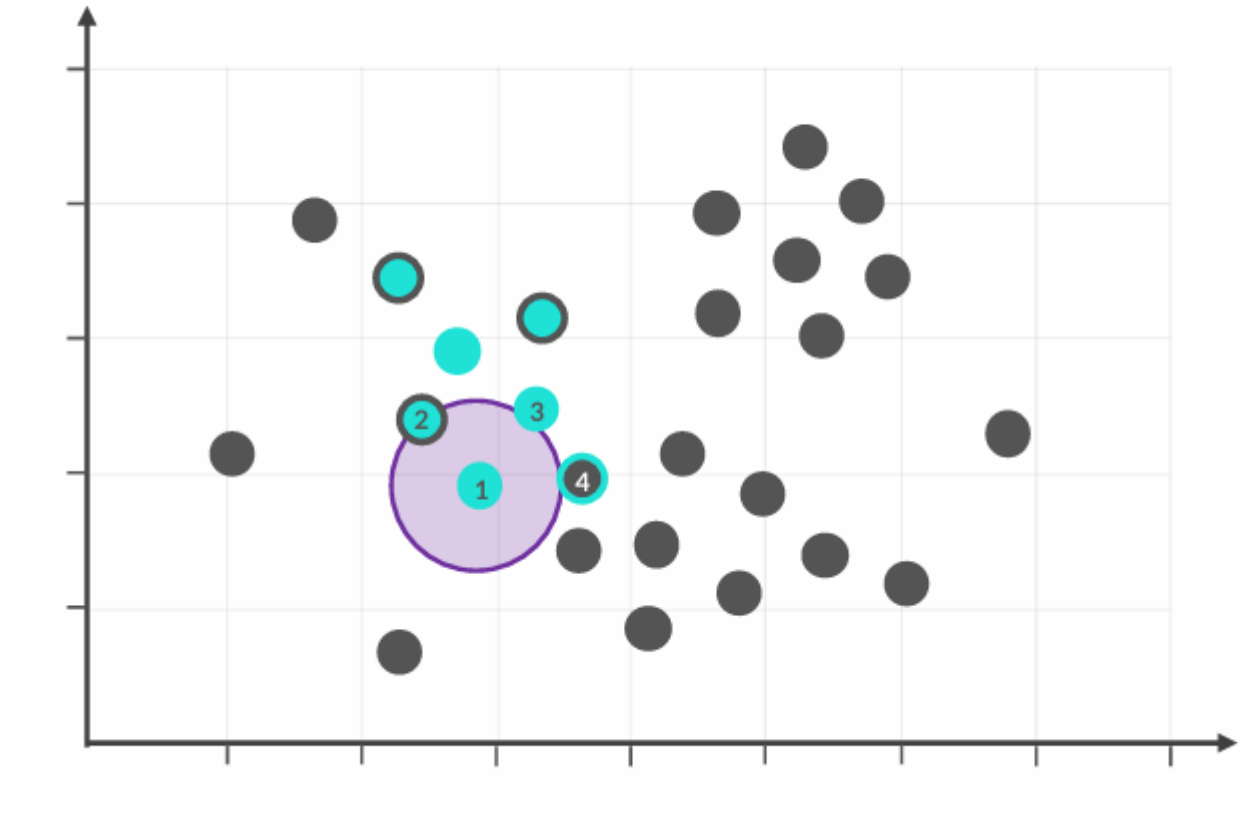
eps = 0.75
min_samples = 4

Step 4.5 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



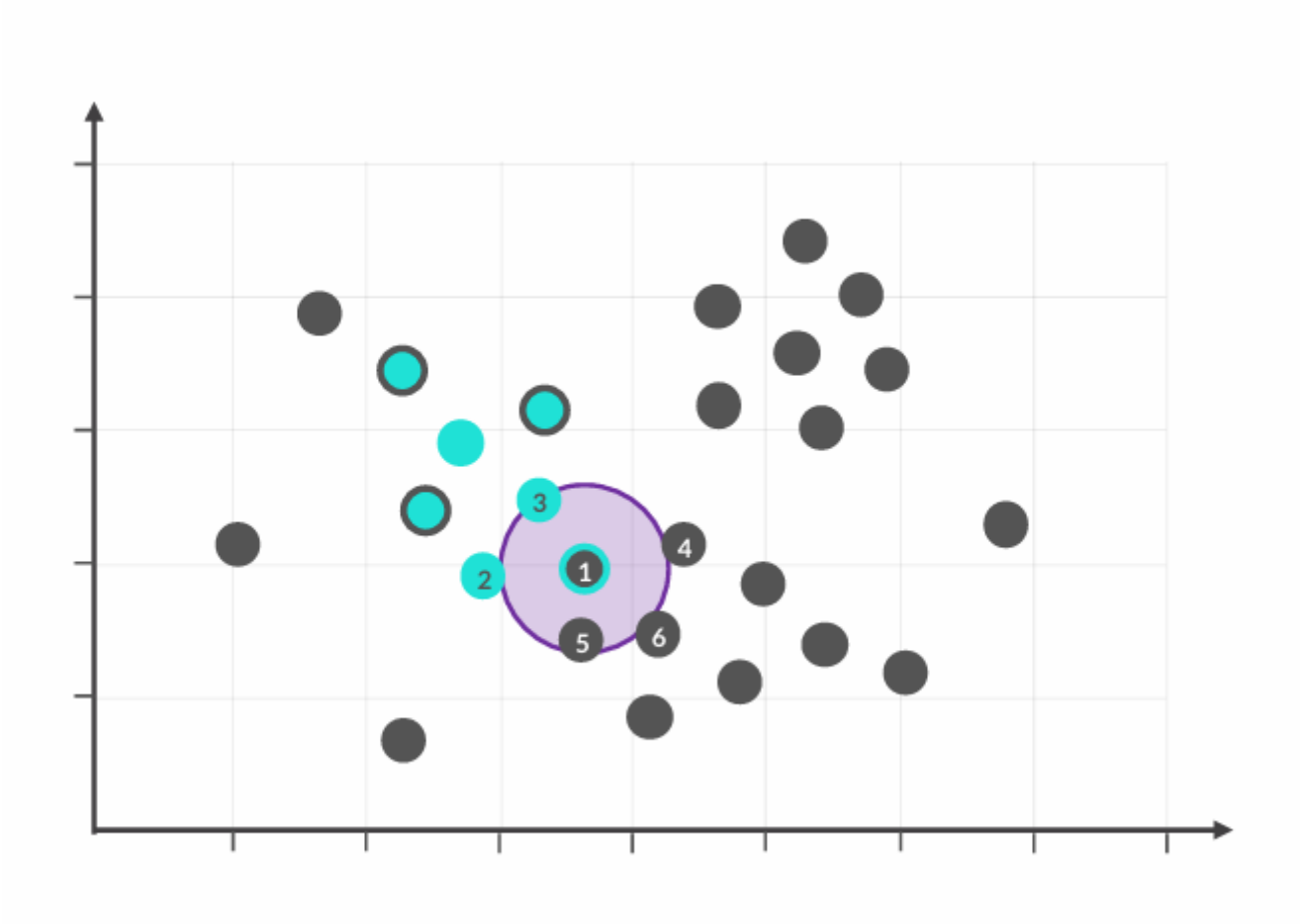
eps = 0.75
min_samples = 4

Step 4.6 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



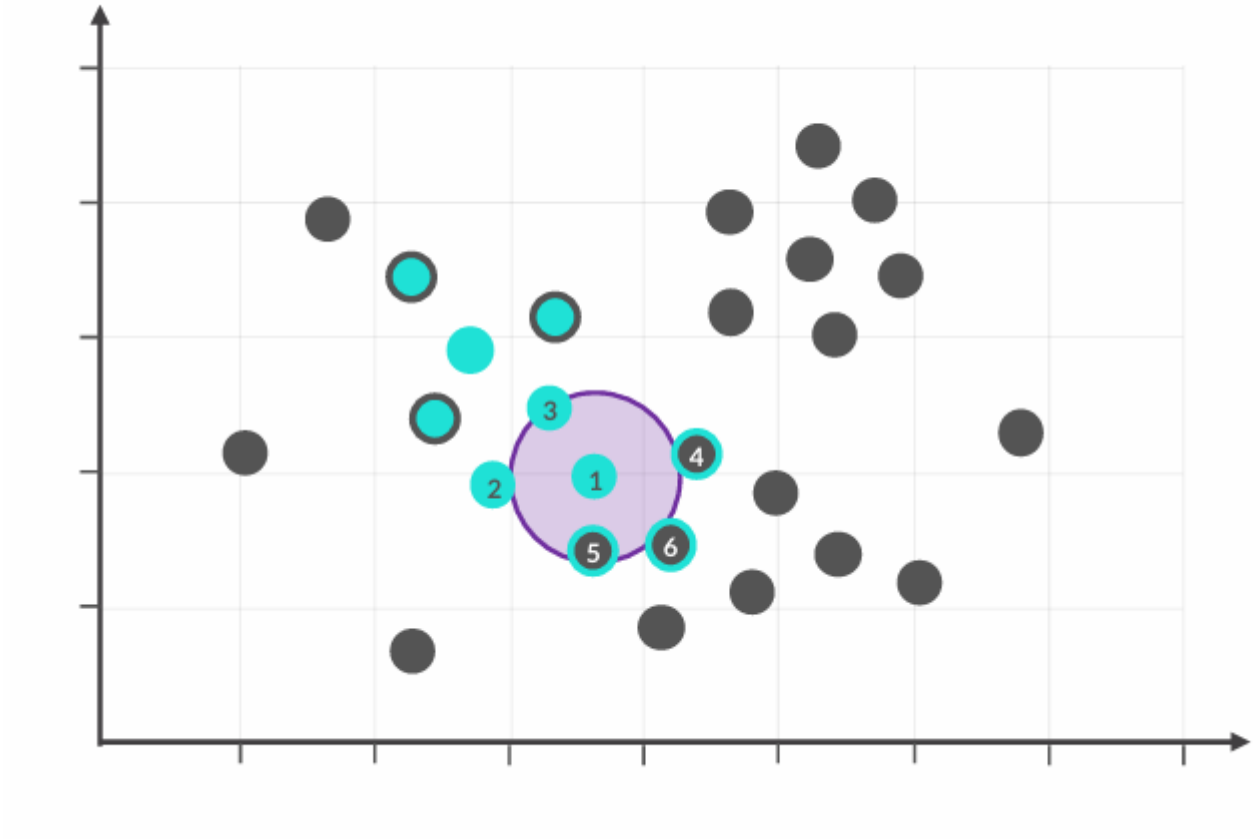
eps = 0.75
min_samples = 4

Step 4.7 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



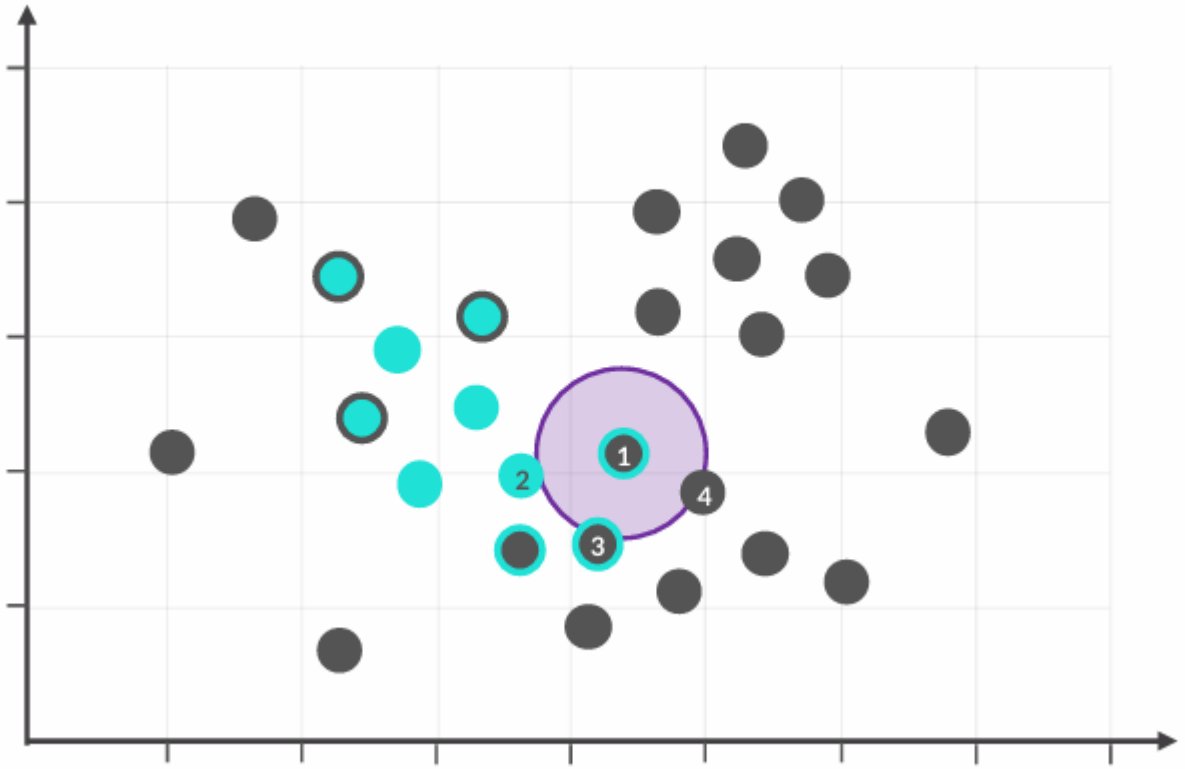
eps = 0.75
min_samples = 4

Step 4.8 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



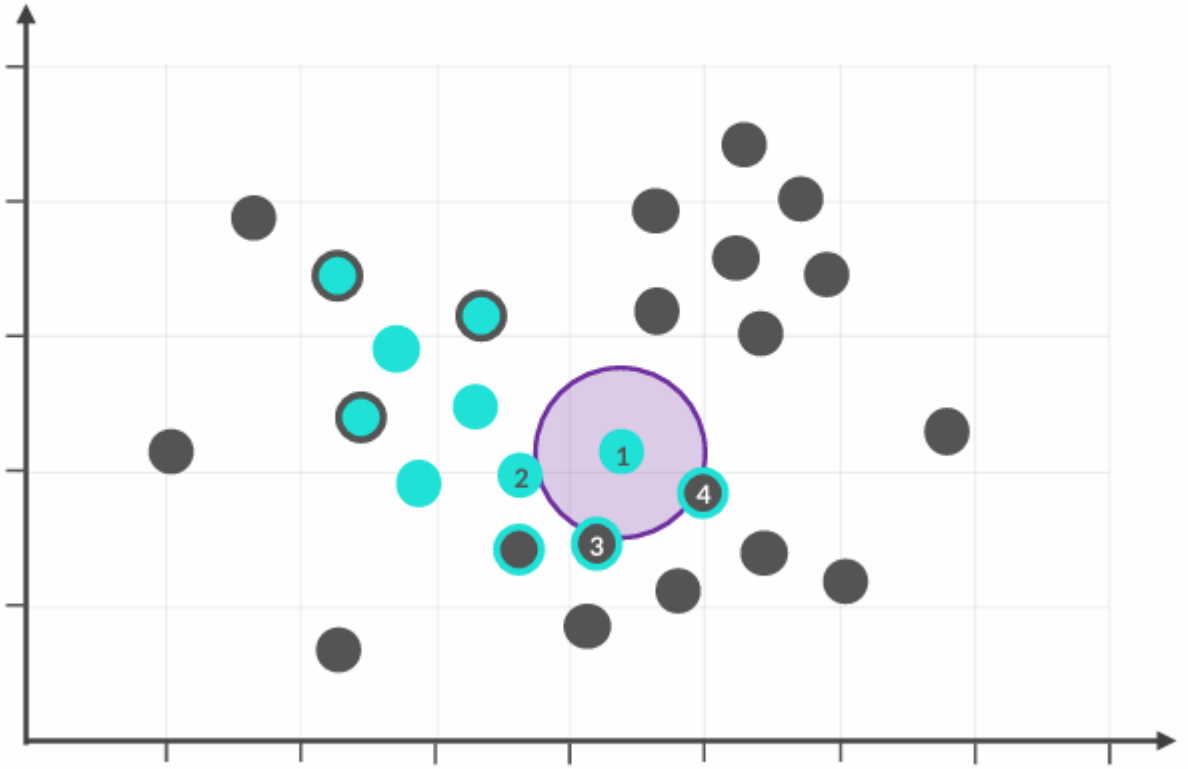
eps = 0.75
min_samples = 4

Step 4.9 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



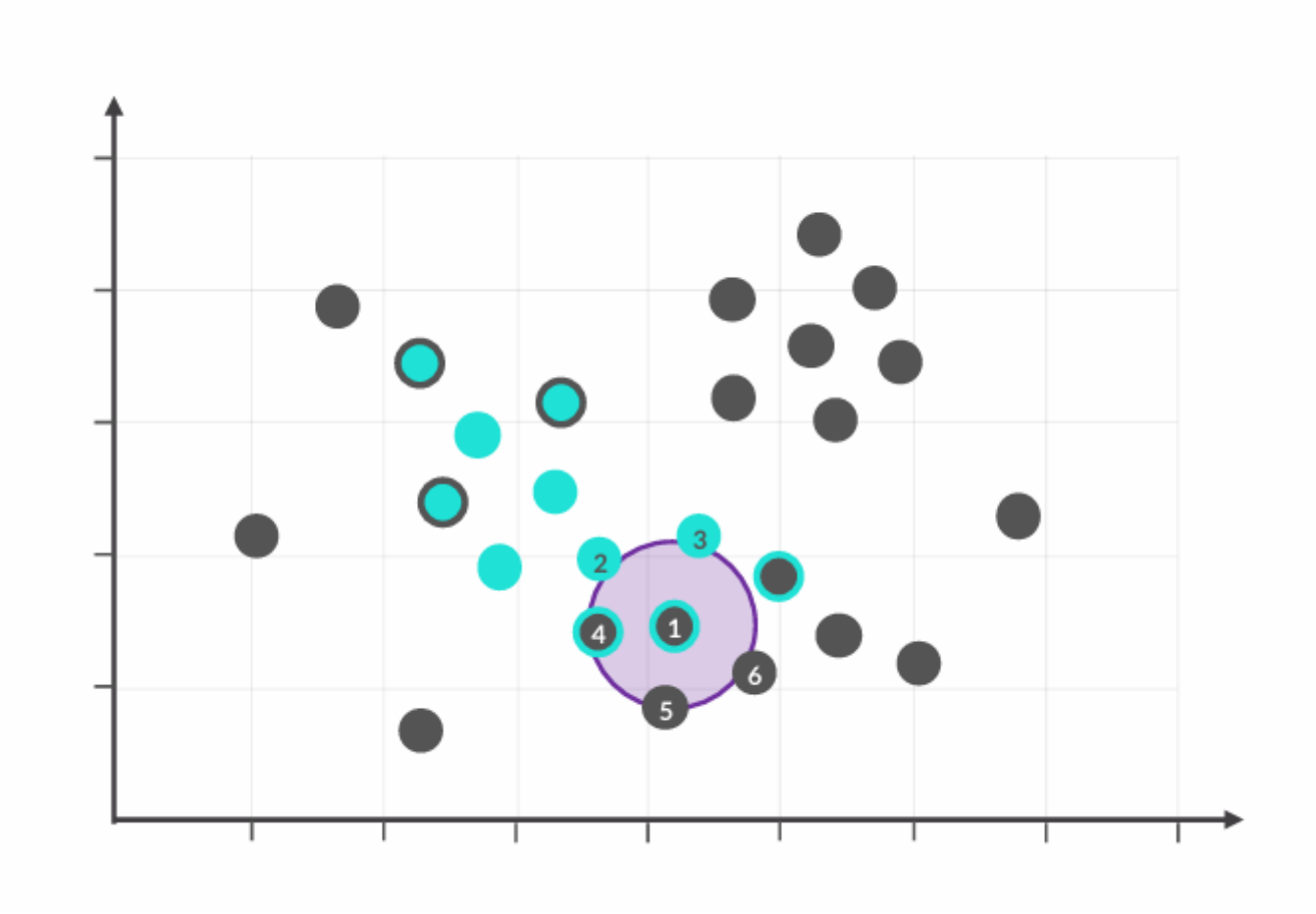
eps = 0.75
min_samples = 4

Step 4.10 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



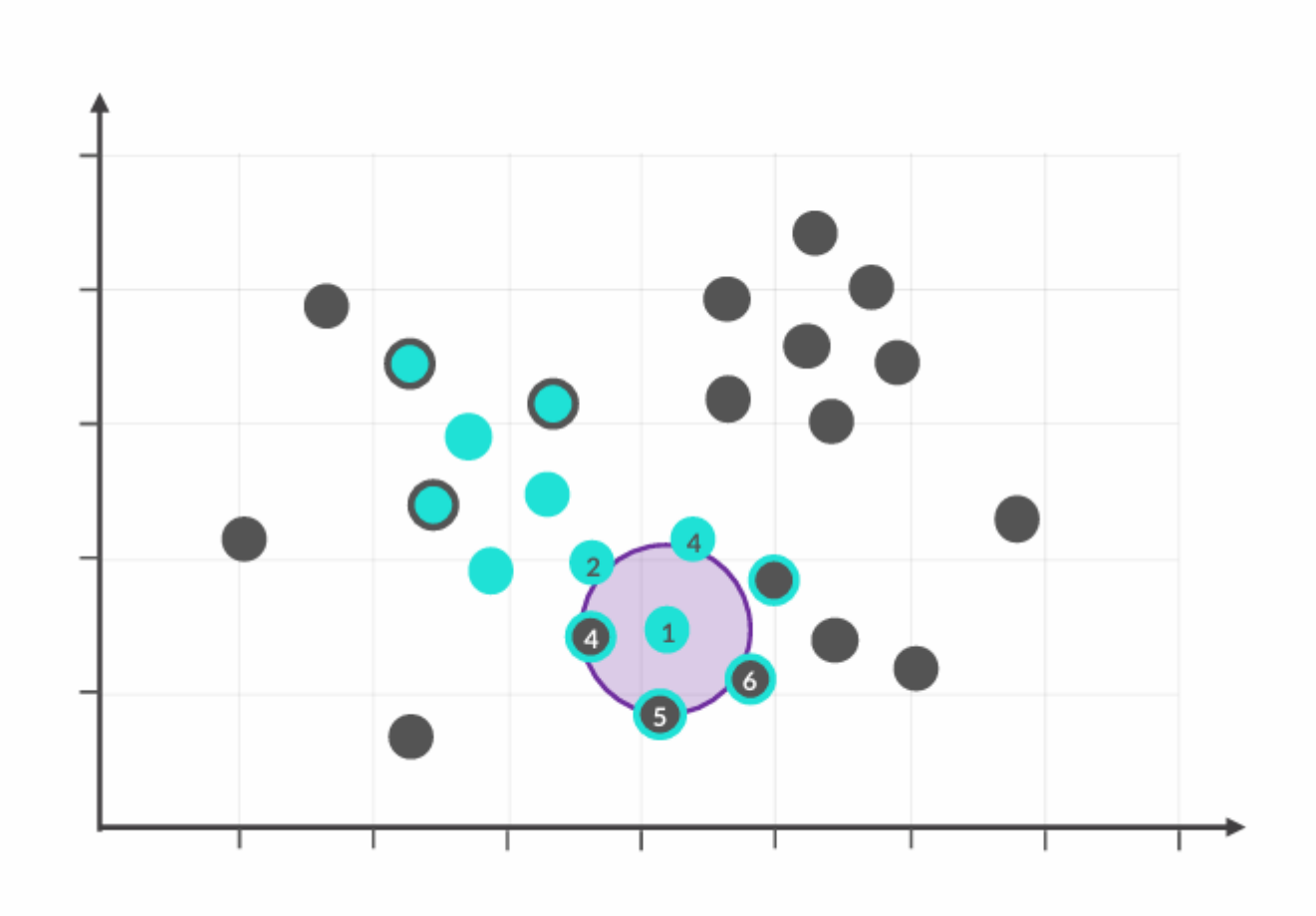
eps = 0.75
min_samples = 4

Step 4.11 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



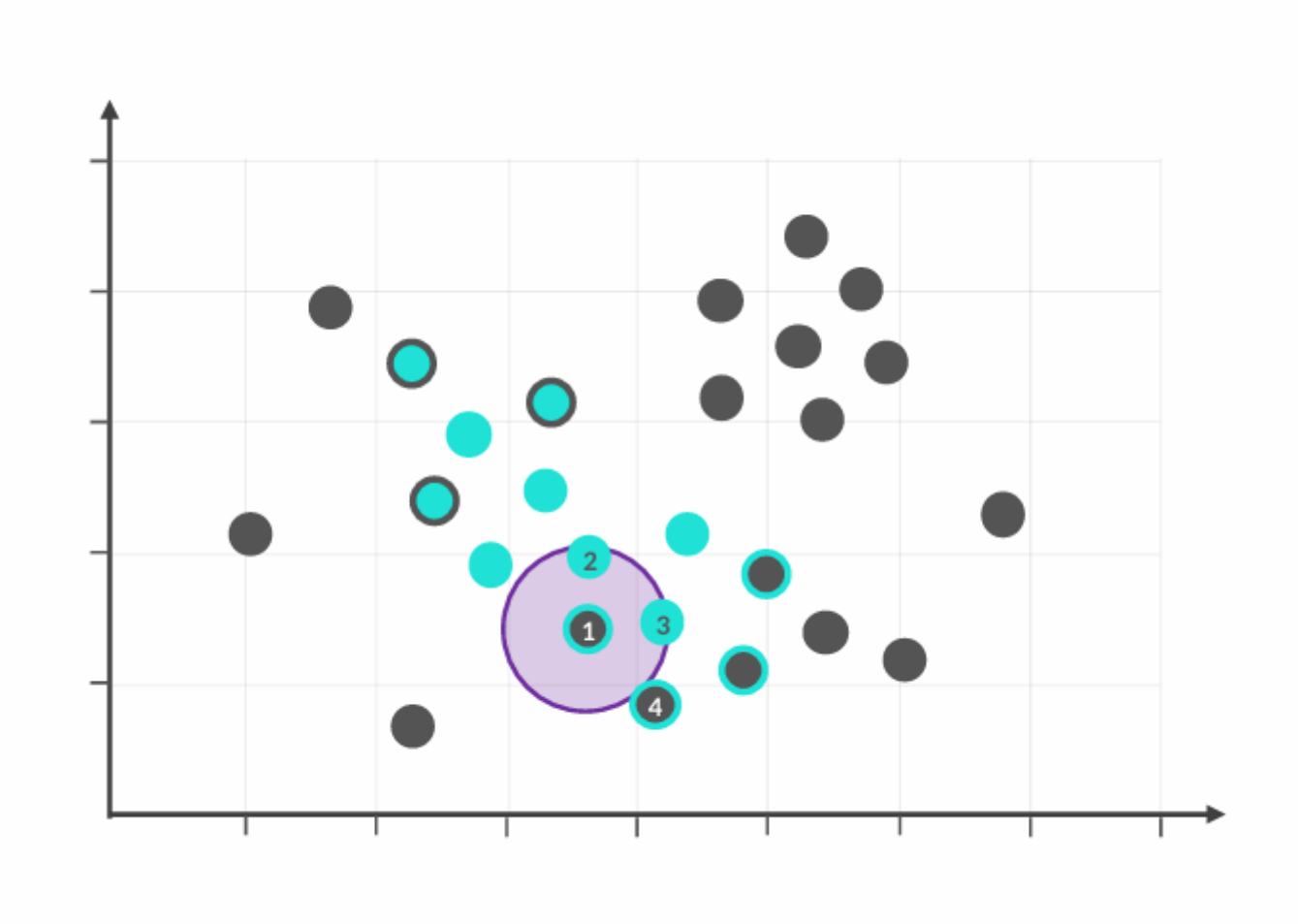
eps = 0.75
min_samples = 4

Step 4.12 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



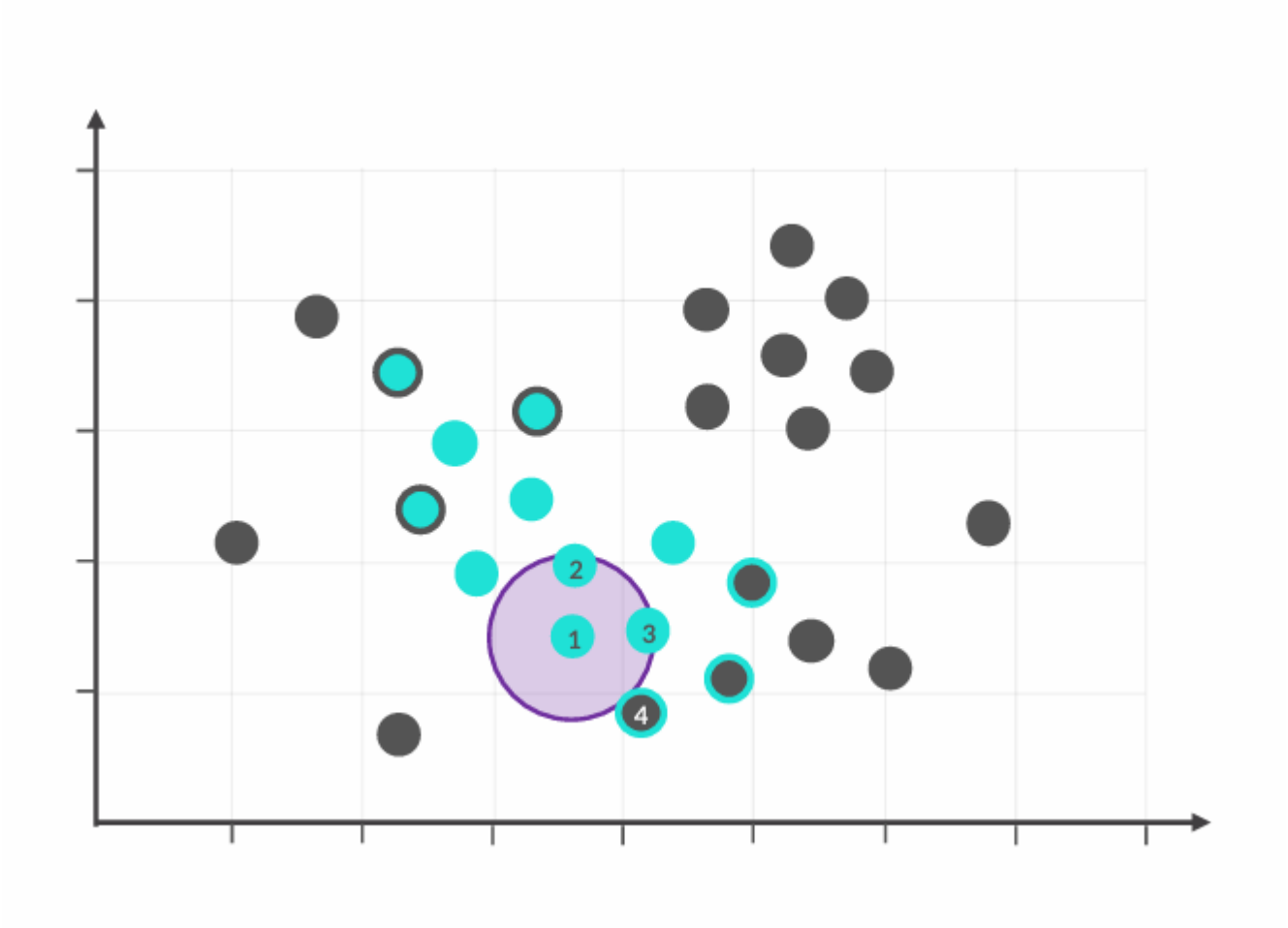
eps = 0.75
min_samples = 4

Step 4.13 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



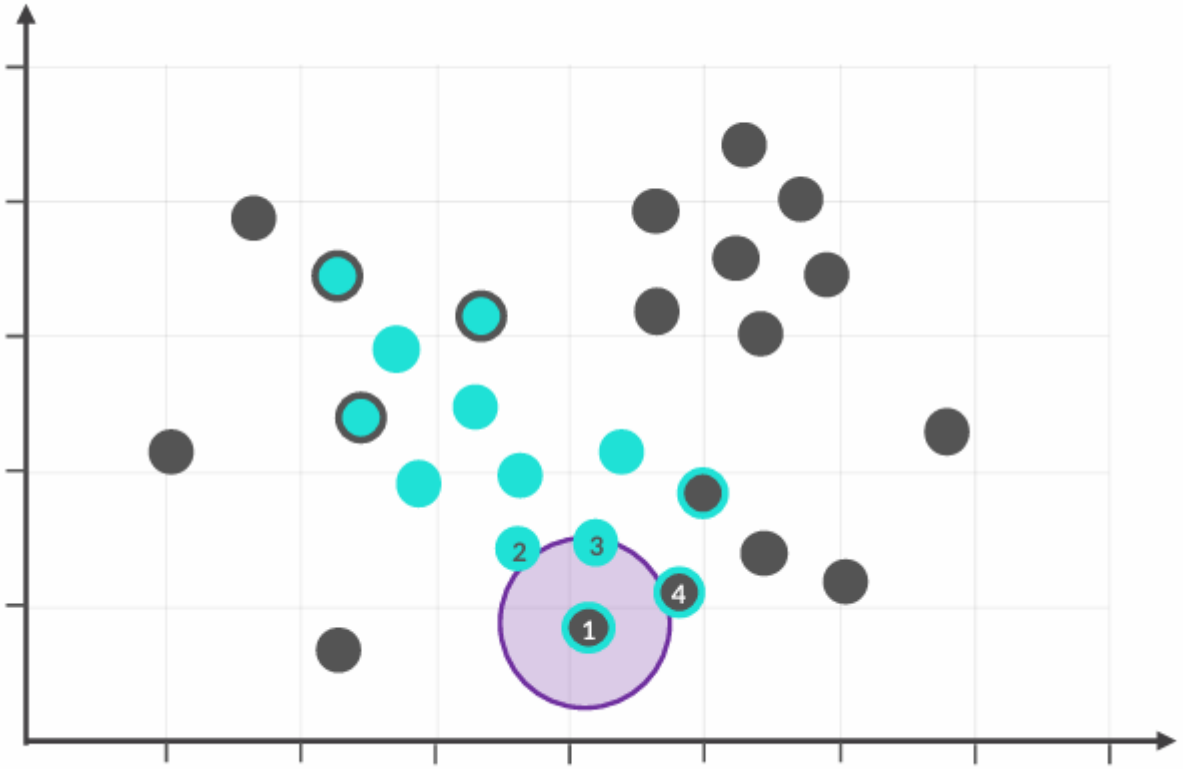
eps = 0.75
min_samples = 4

Step 4.14 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



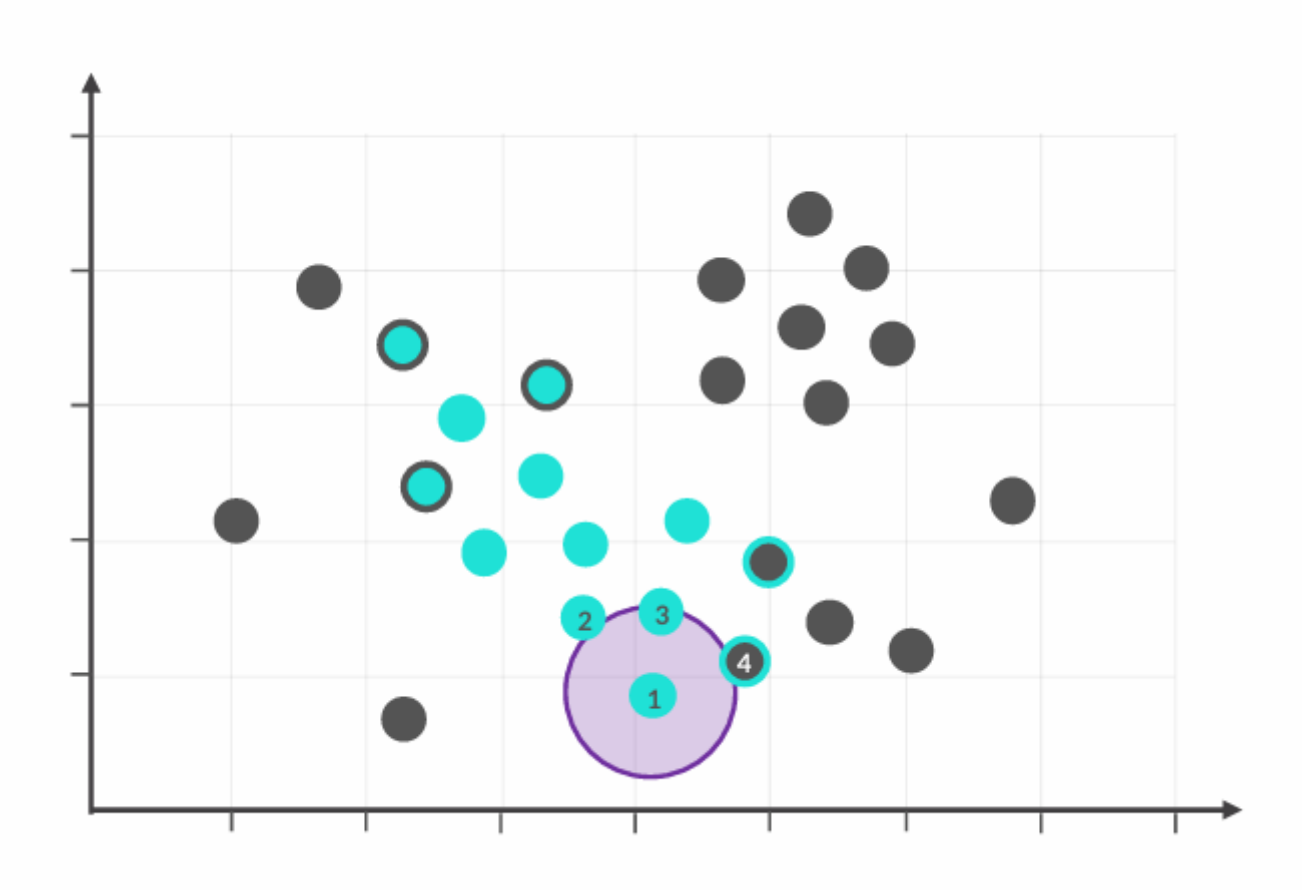
eps = 0.75
min_samples = 4

Step 4.15 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



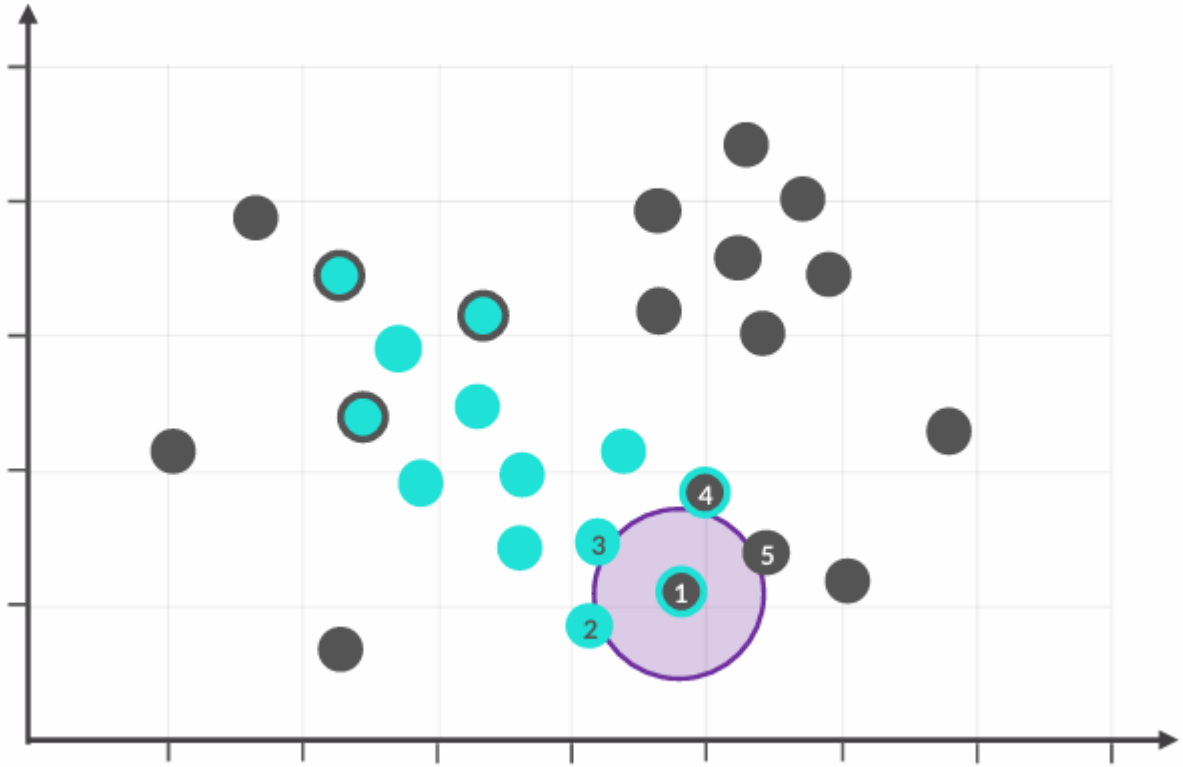
eps = 0.75
min_samples = 4

Step 4.16 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



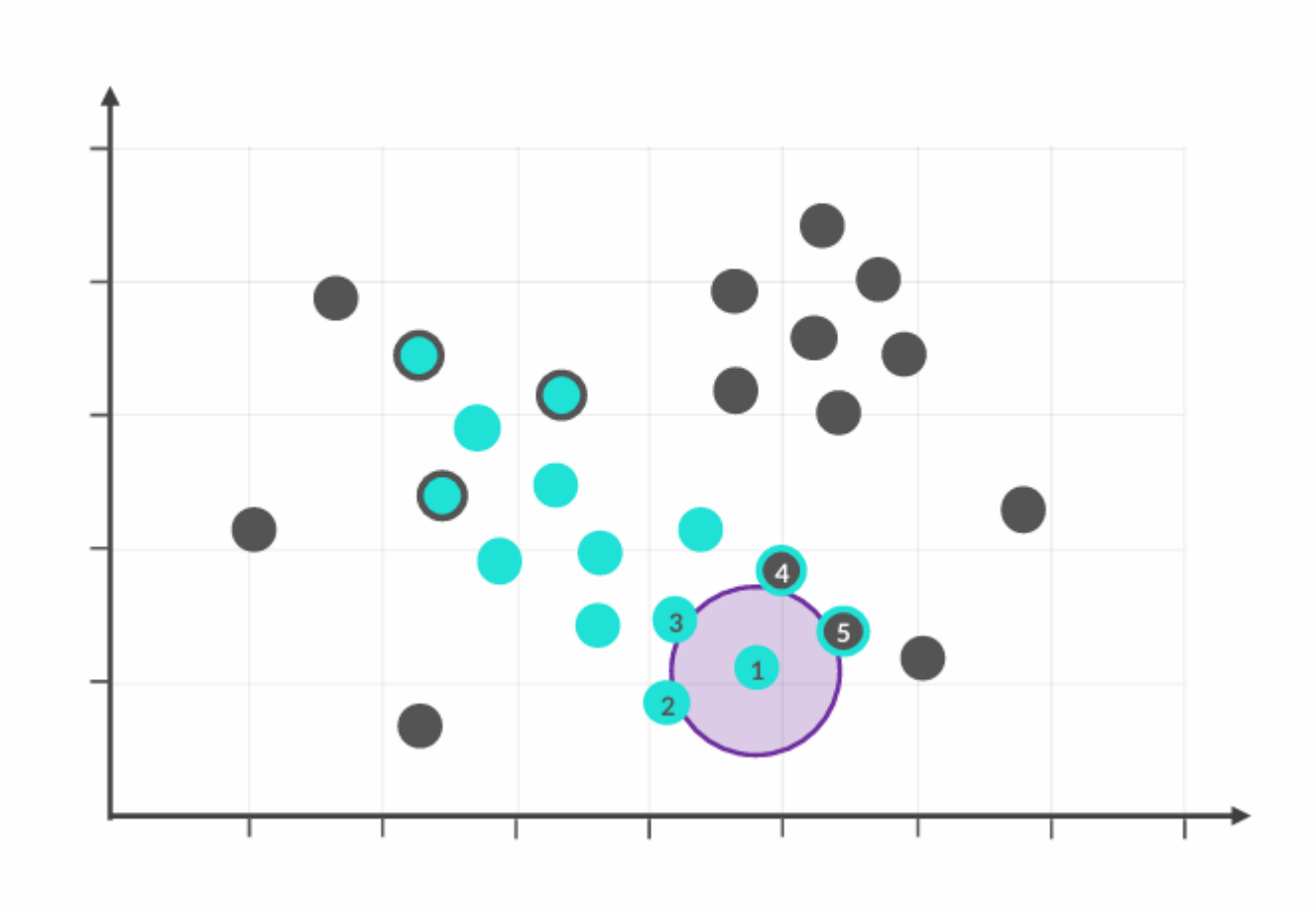
eps = 0.75
min_samples = 4

Step 4.17 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



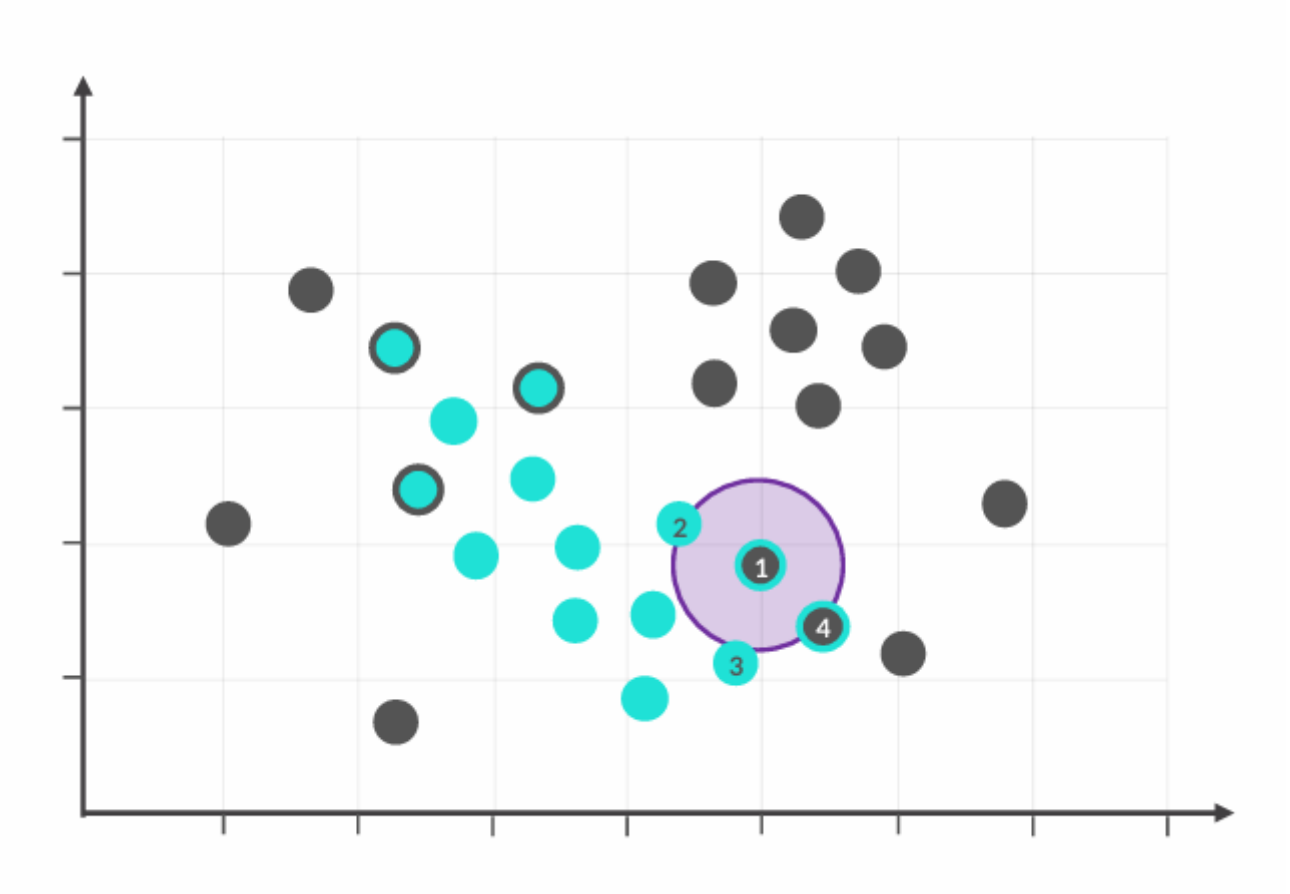
eps = 0.75
min_samples = 4

Step 4.18 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



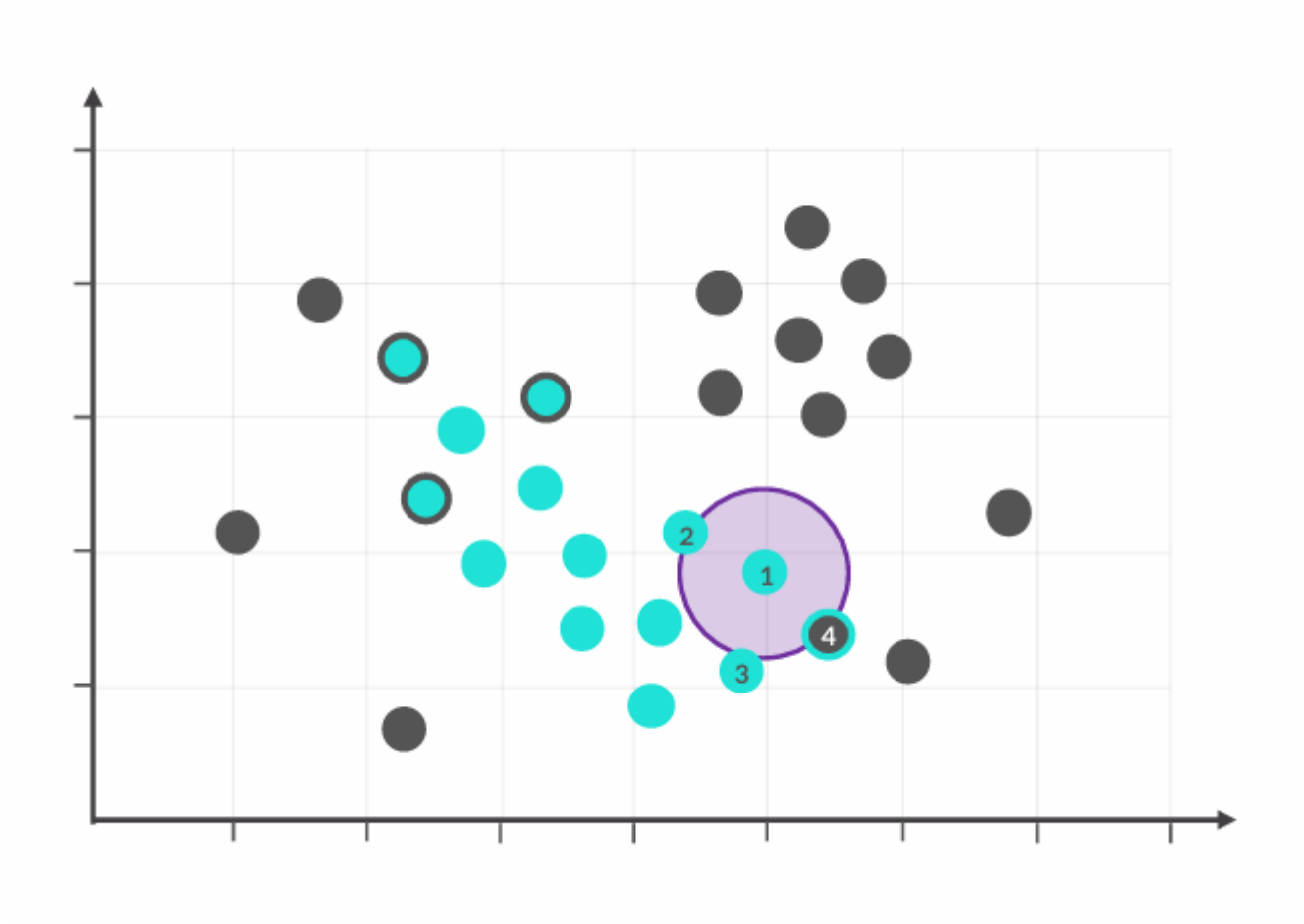
eps = 0.75
min_samples = 4

Step 4.19 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



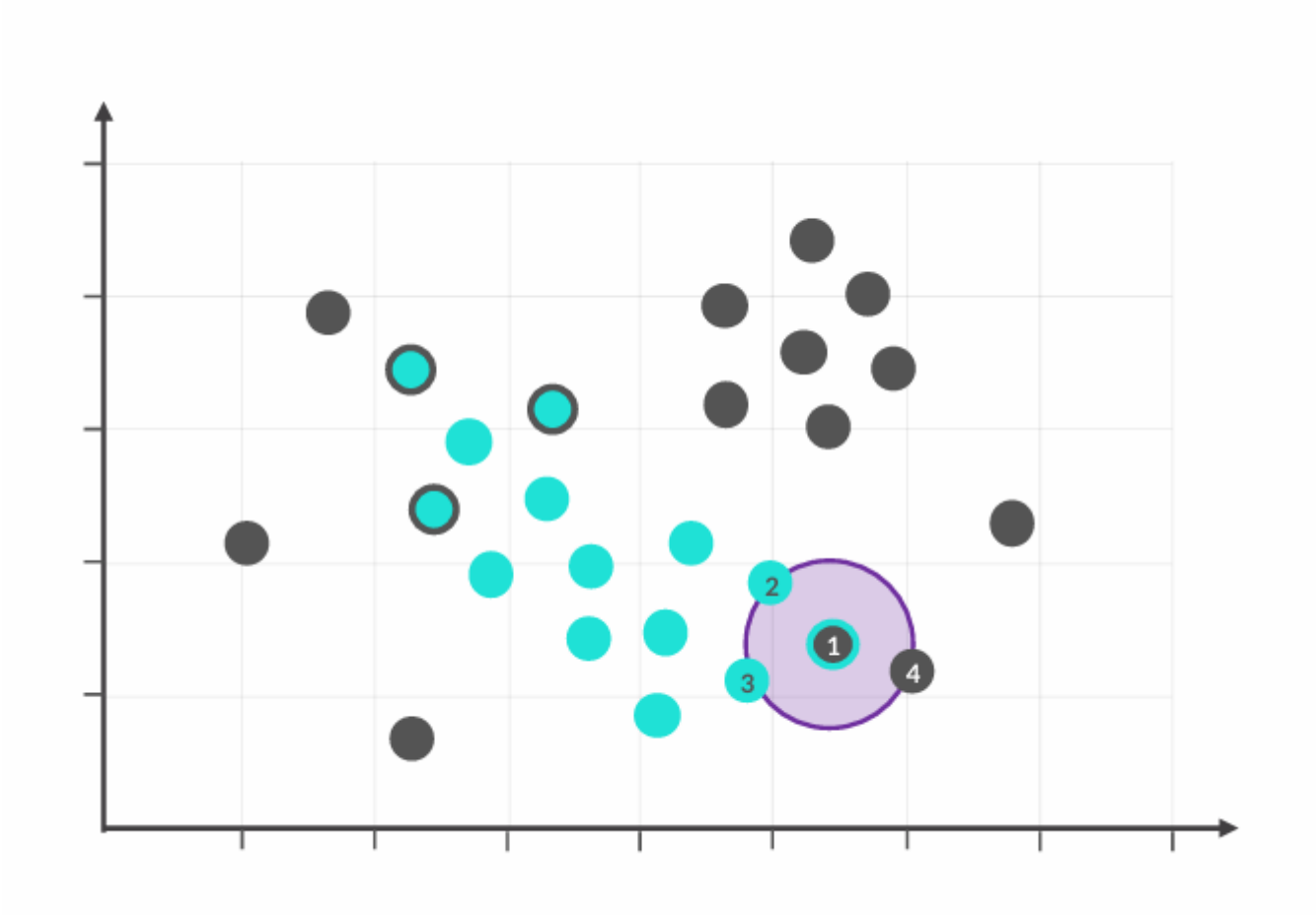
eps = 0.75
min_samples = 4

Step 4.20 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



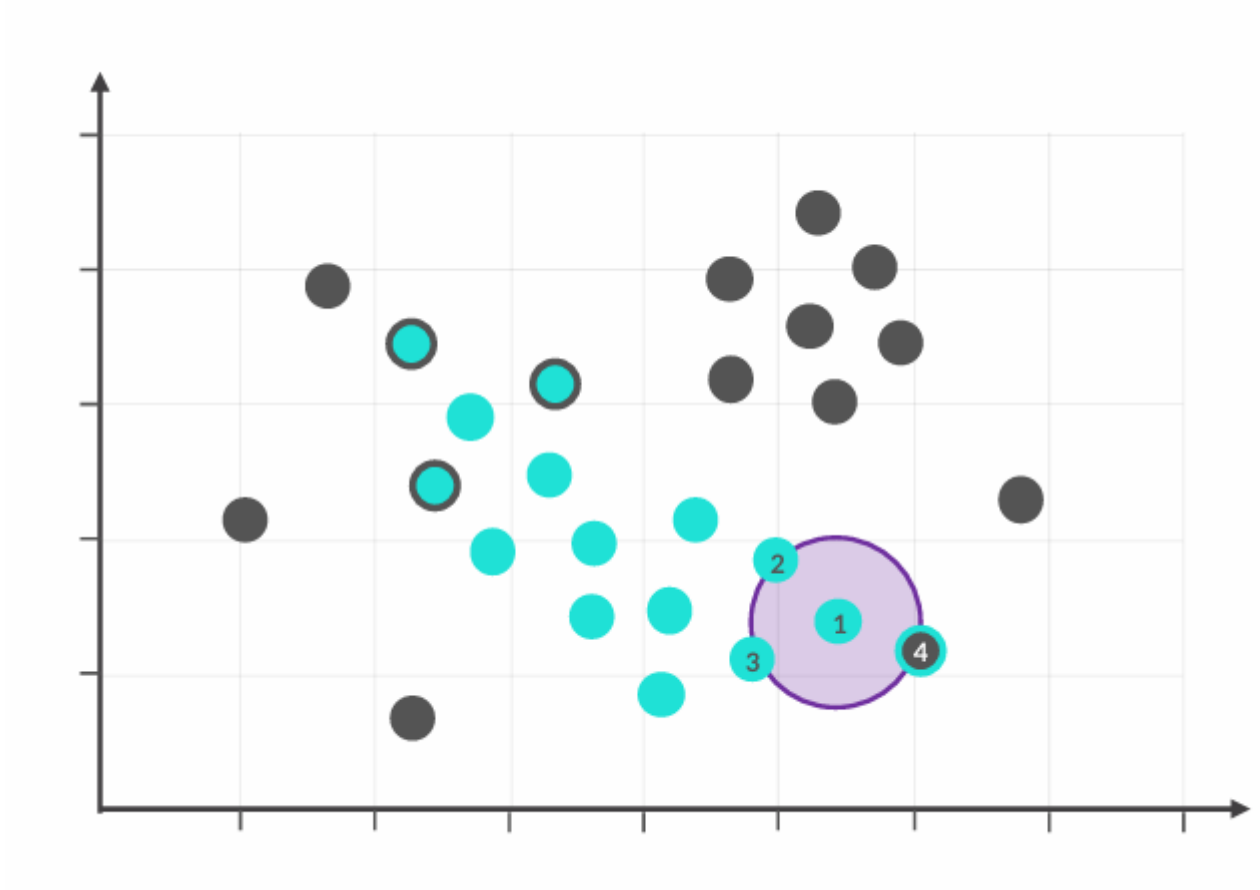
eps = 0.75
min_samples = 4

Step 4.21 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



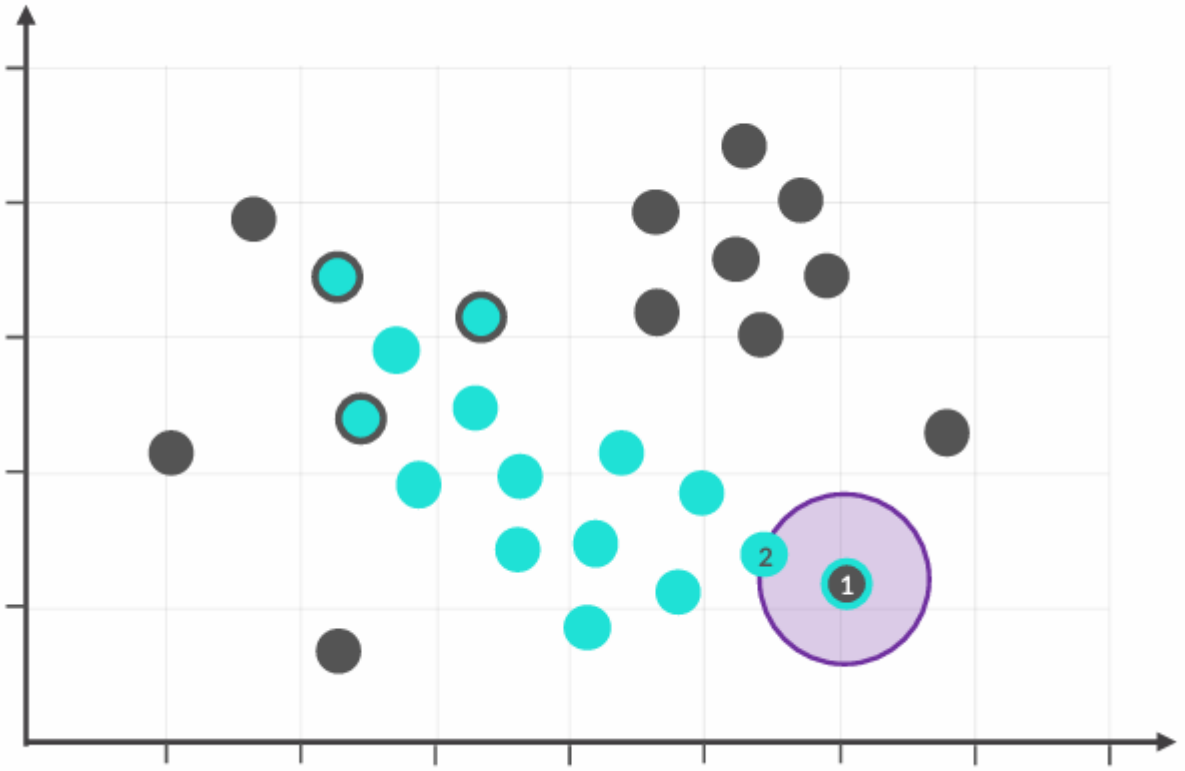
eps = 0.75
min_samples = 4

Step 4.22 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



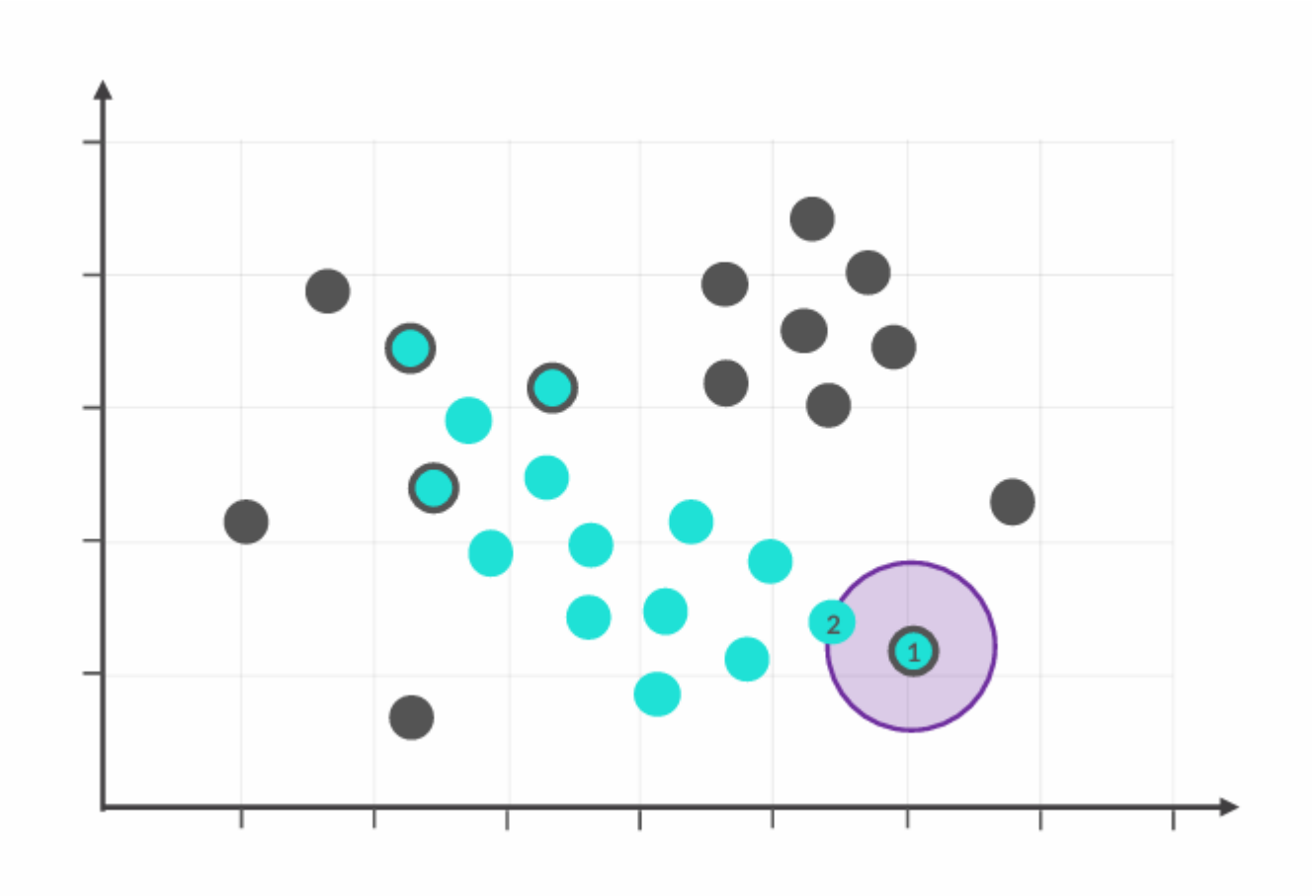
eps = 0.75
min_samples = 4

Step 4.23 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



eps = 0.75
min_samples = 4

Step 4.24 - Move on to another neighbor and continue this process until all the points are labeled within the cluster



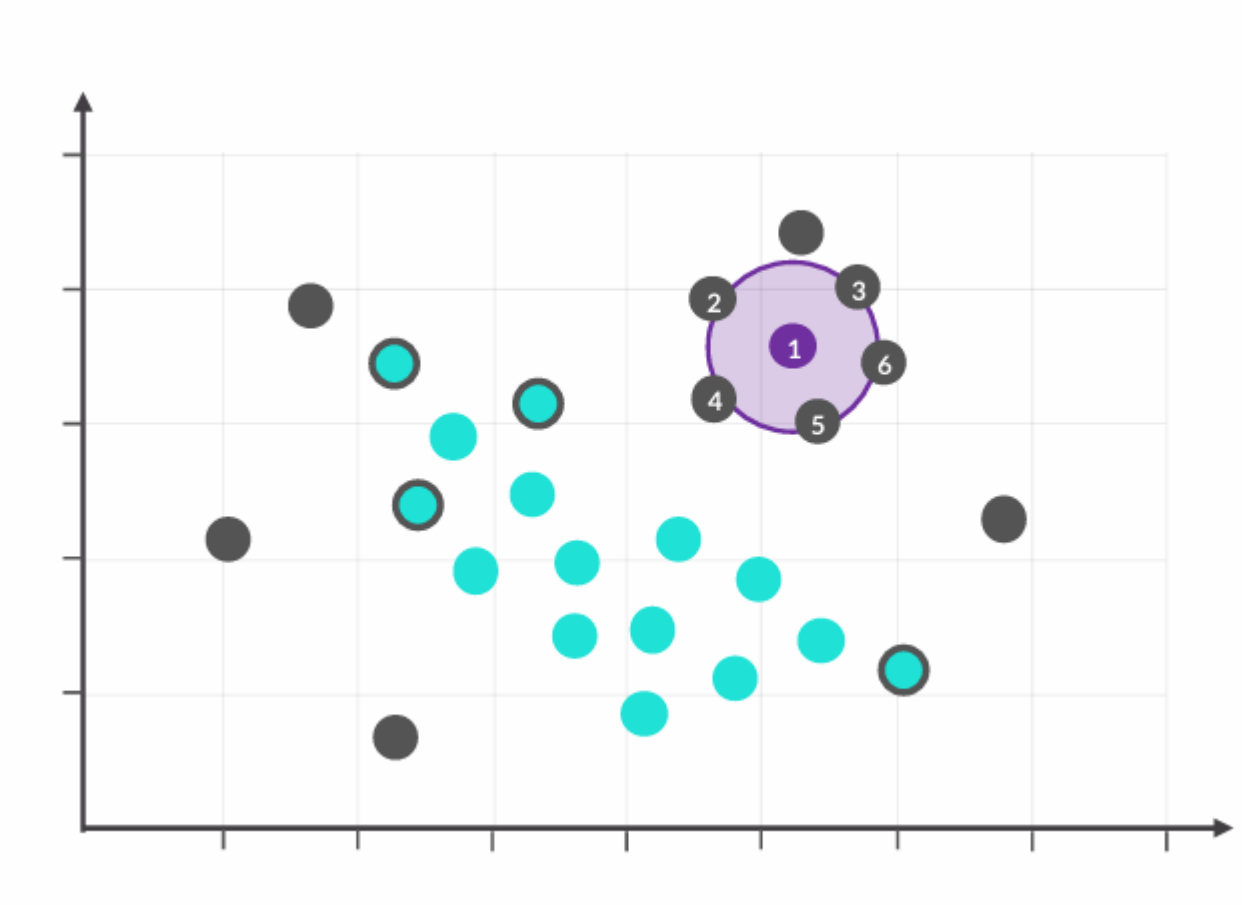
eps = 0.75
min_samples = 4

Step 5 - Iteration

Step 5 - Iteration

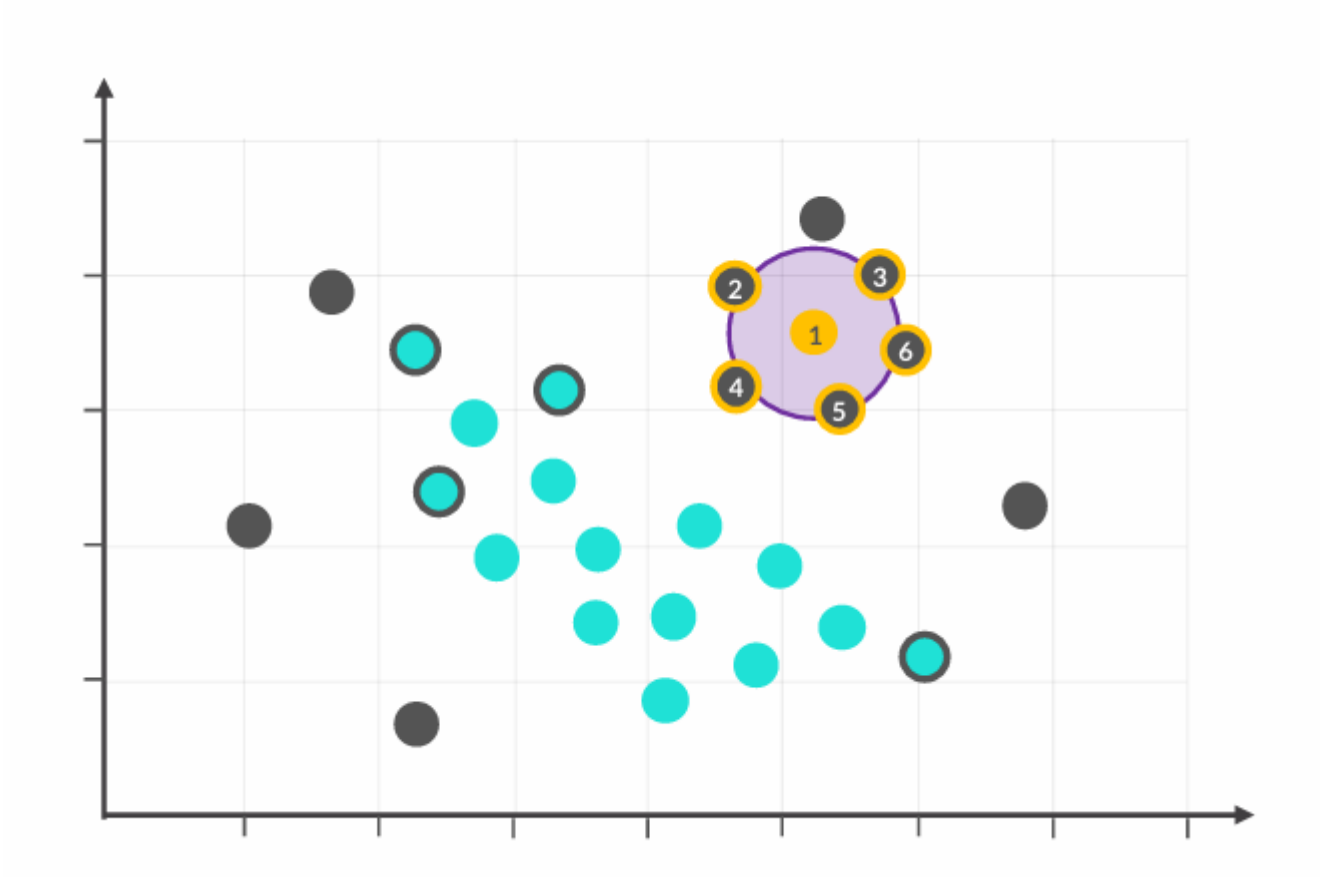
After a cluster is fully formed, the algorithm moves on to another unlabeled, randomly selected point in the dataset and repeats the process from step 2. This continues until all data points have been visited and assigned a label (either a cluster ID or "noise").

Step 5.1 - Move on to another random point and repeat the same steps



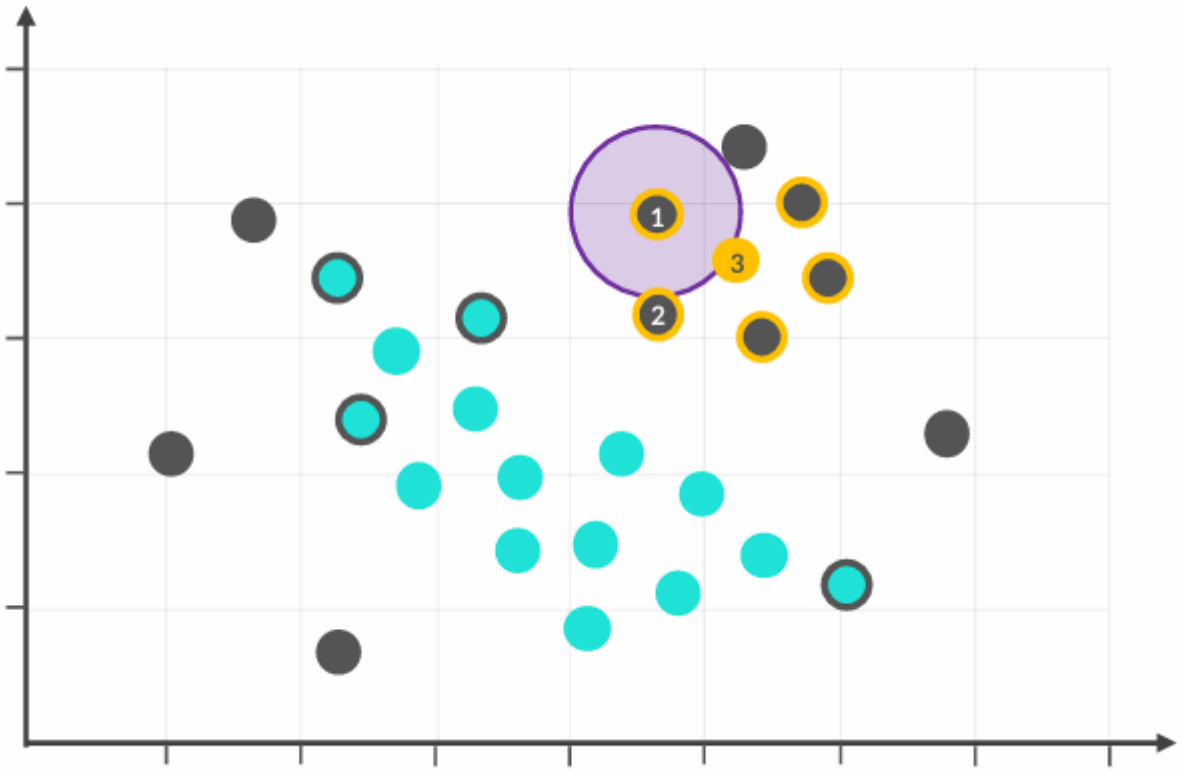
eps = 0.75
min_samples = 4

Step 5.2 - Move on to another random point and repeat the same steps



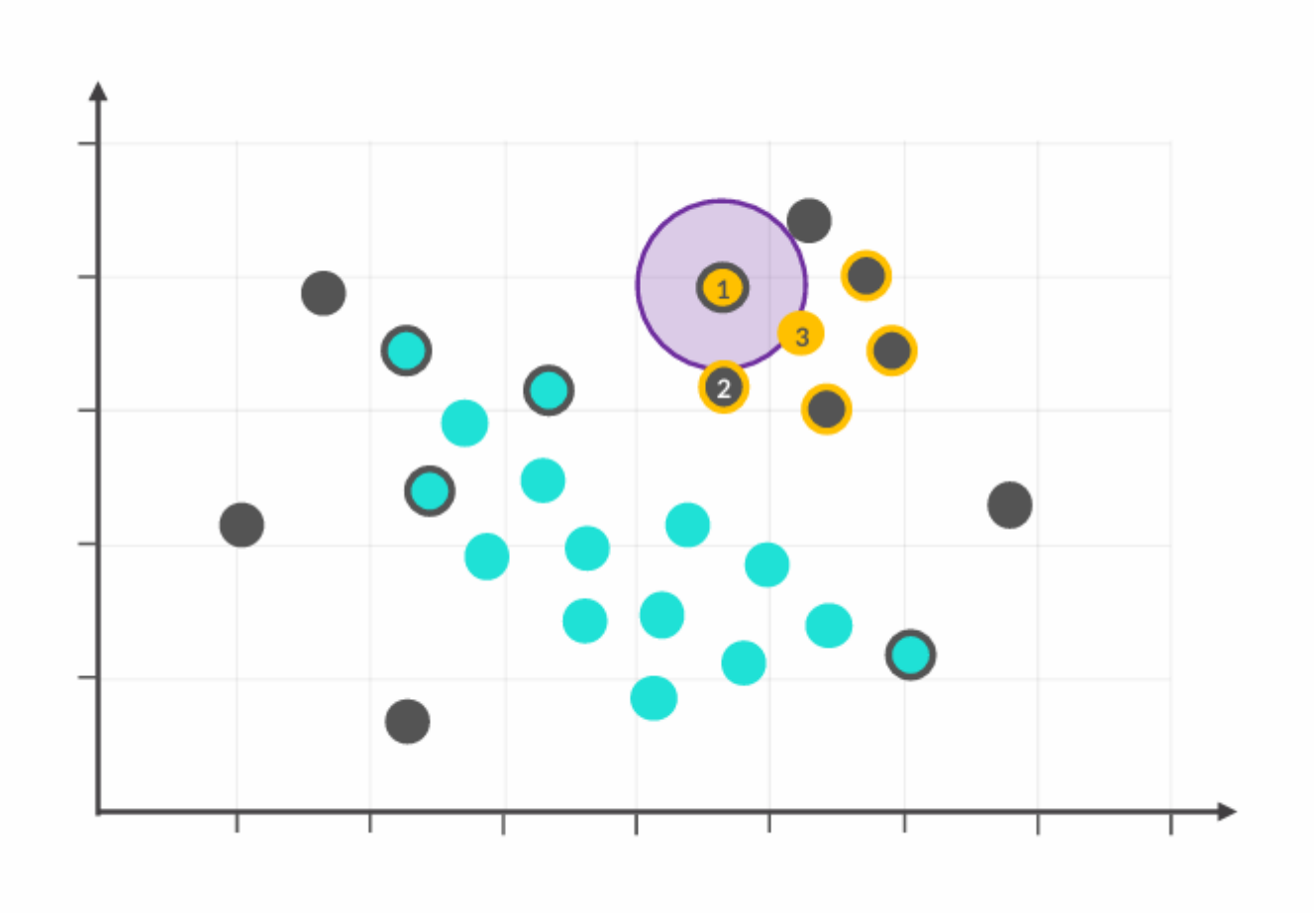
eps = 0.75
min_samples = 4

Step 5.3 - Move on to another random point and repeat the same steps



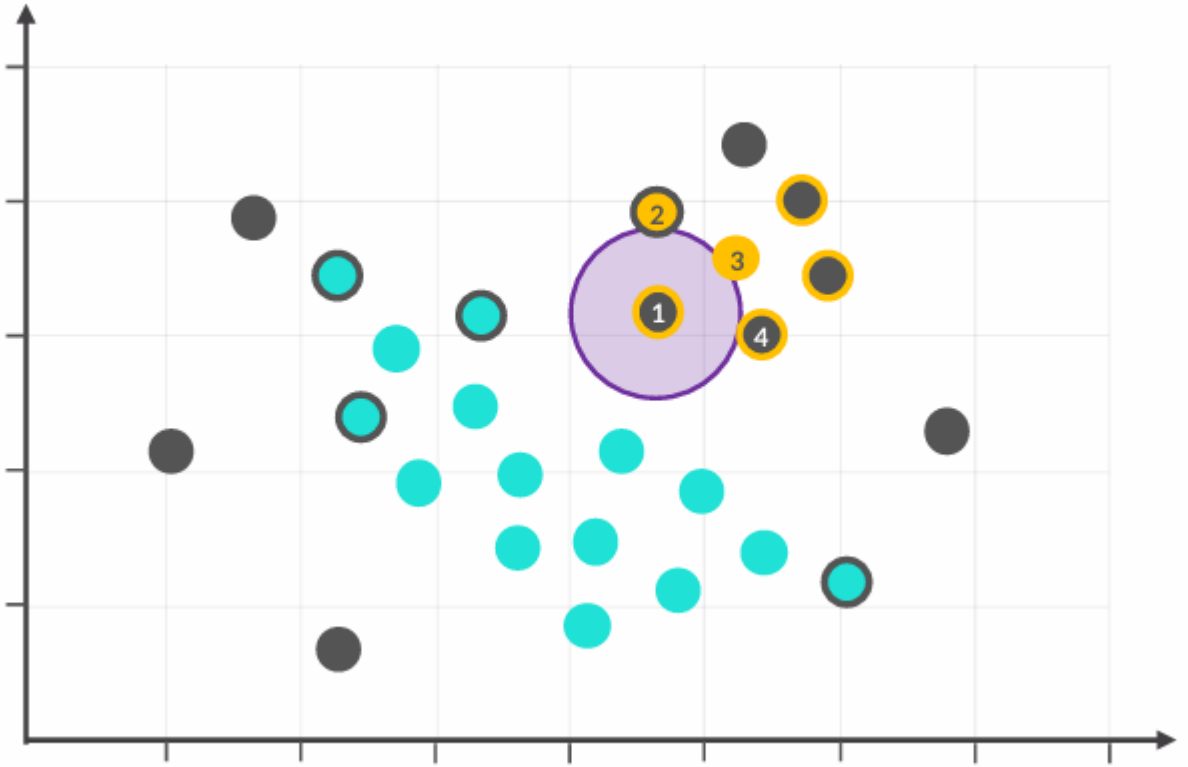
eps = 0.75
min_samples = 4

Step 5.4 - Move on to another random point and repeat the same steps



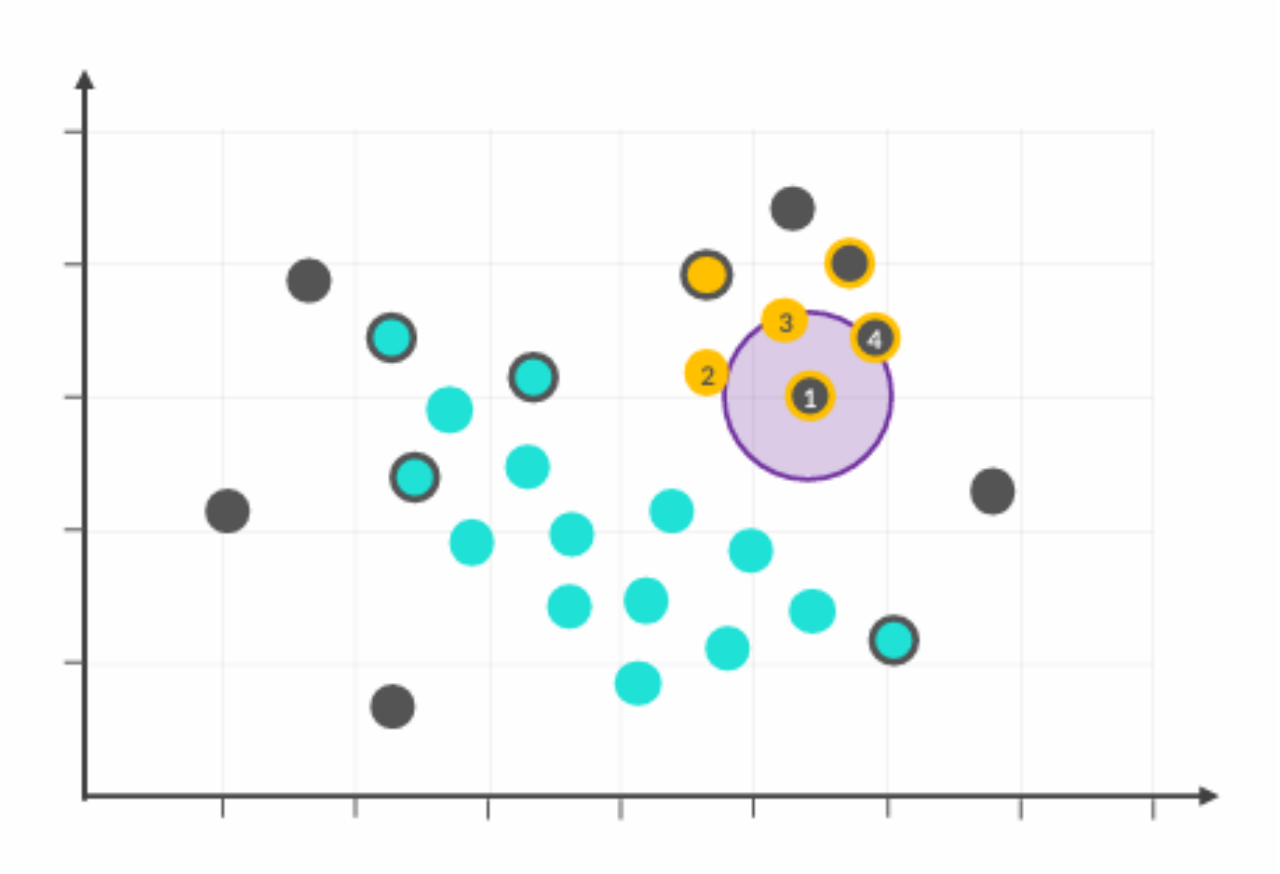
eps = 0.75
min_samples = 4

Step 5.5 - Move on to another random point and repeat the same steps



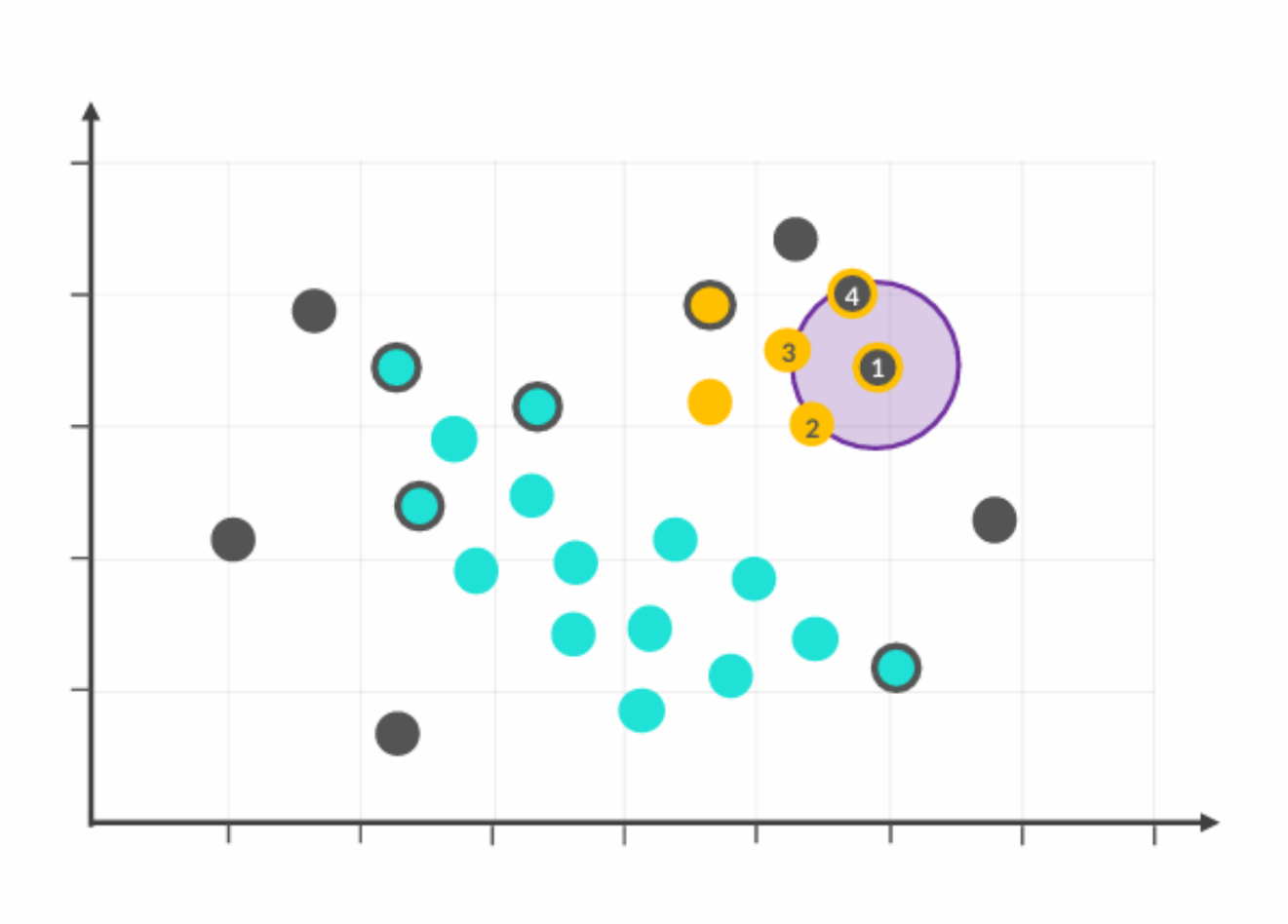
eps = 0.75
min_samples = 4

Step 5.6 - Move on to another random point and repeat the same steps



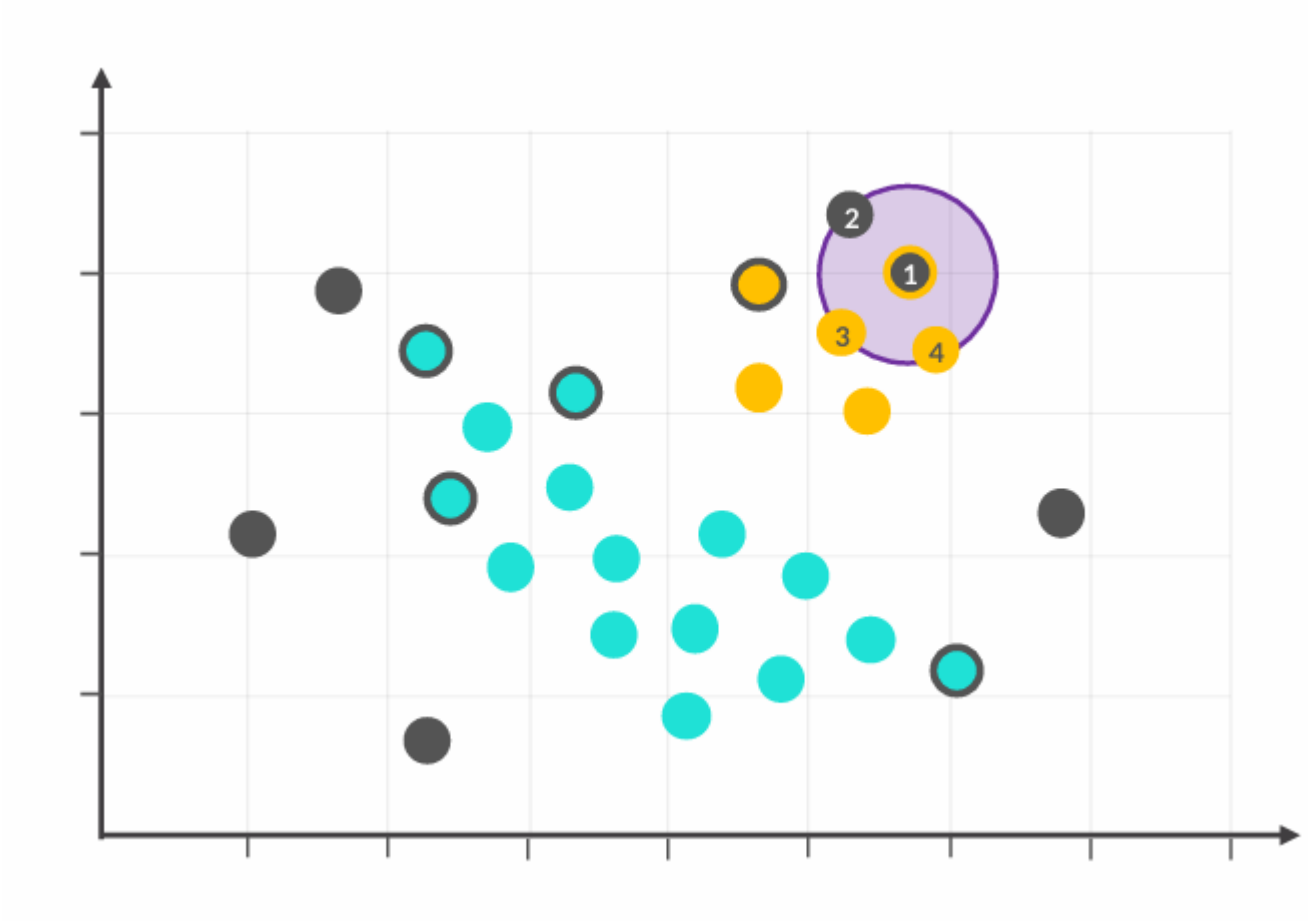
eps = 0.75
min_samples = 4

Step 5.7 - Move on to another random point and repeat the same steps



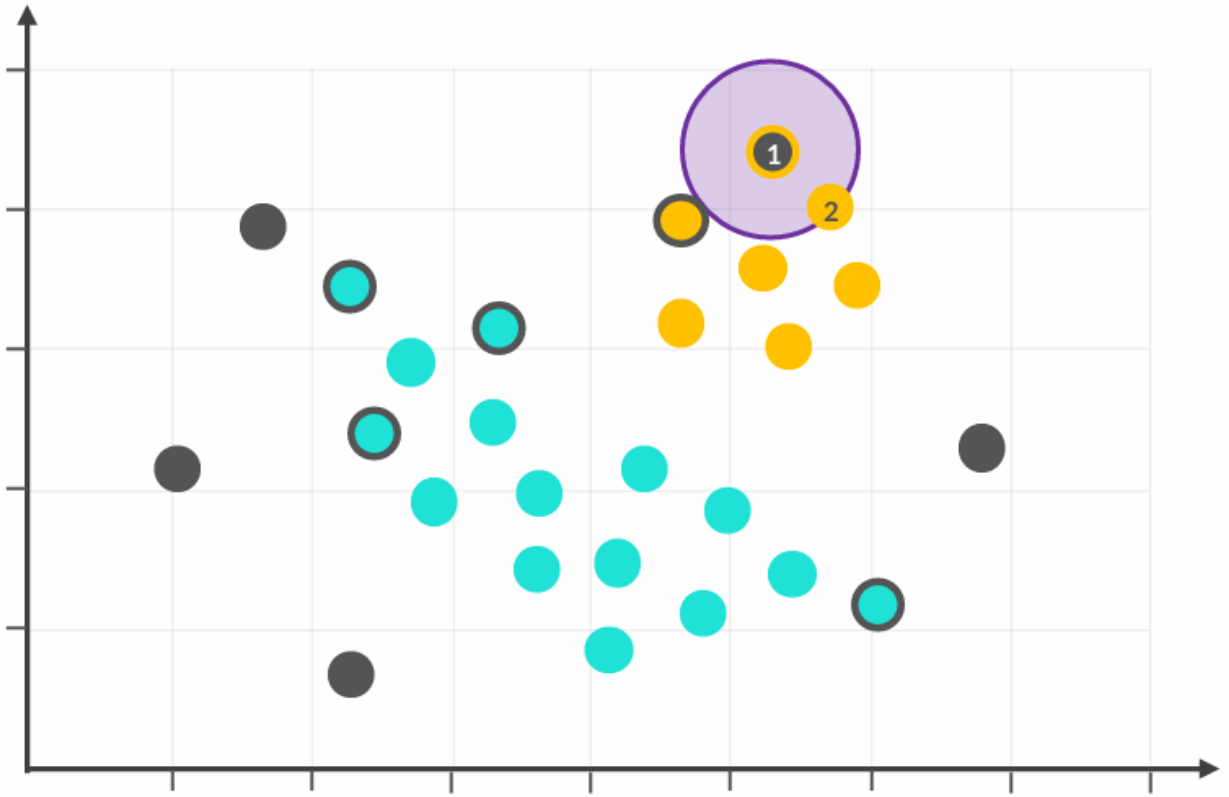
eps = 0.75
min_samples = 4

Step 5.8 - Move on to another random point and repeat the same steps



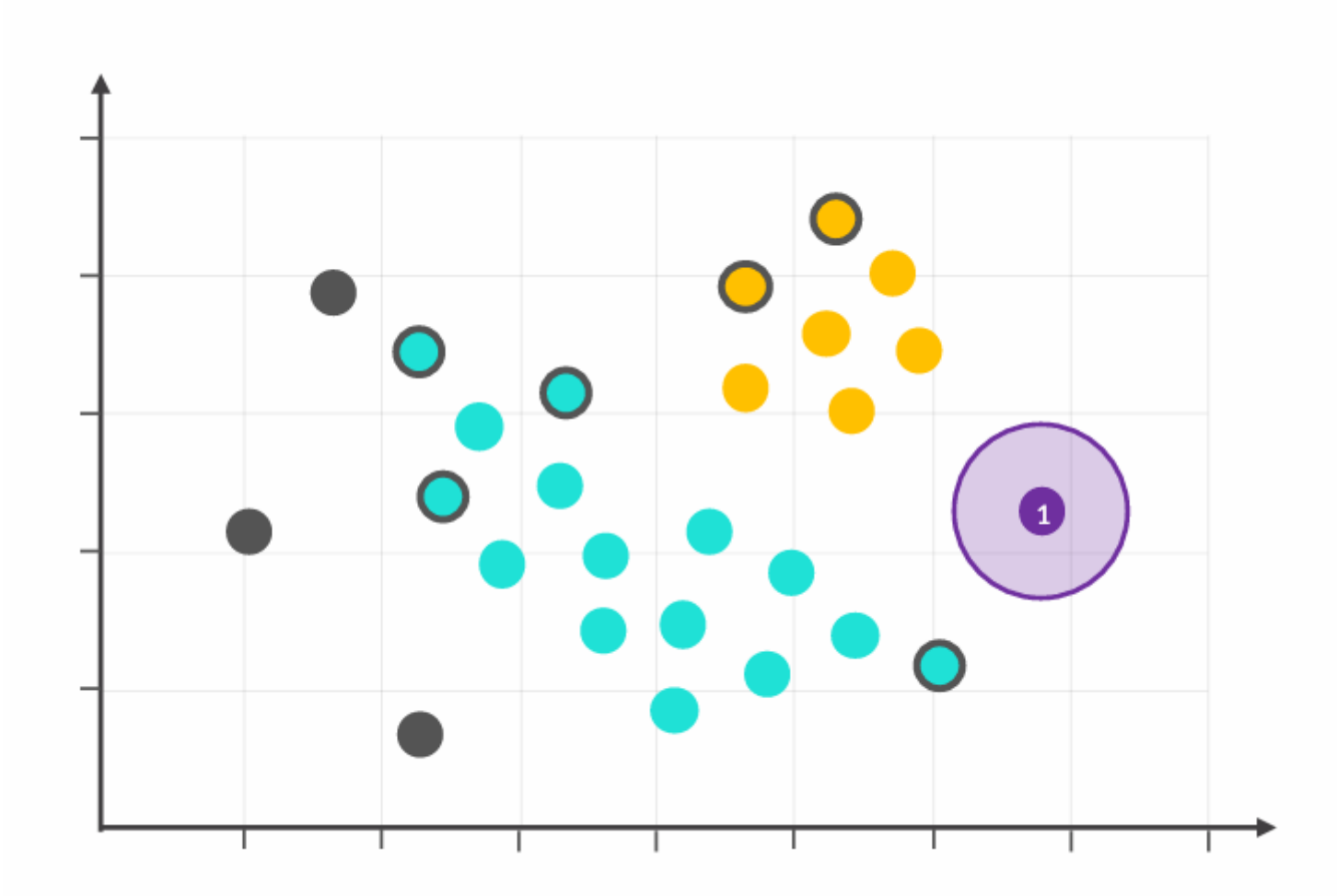
eps = 0.75
min_samples = 4

Step 5.9 - Move on to another random point and repeat the same steps



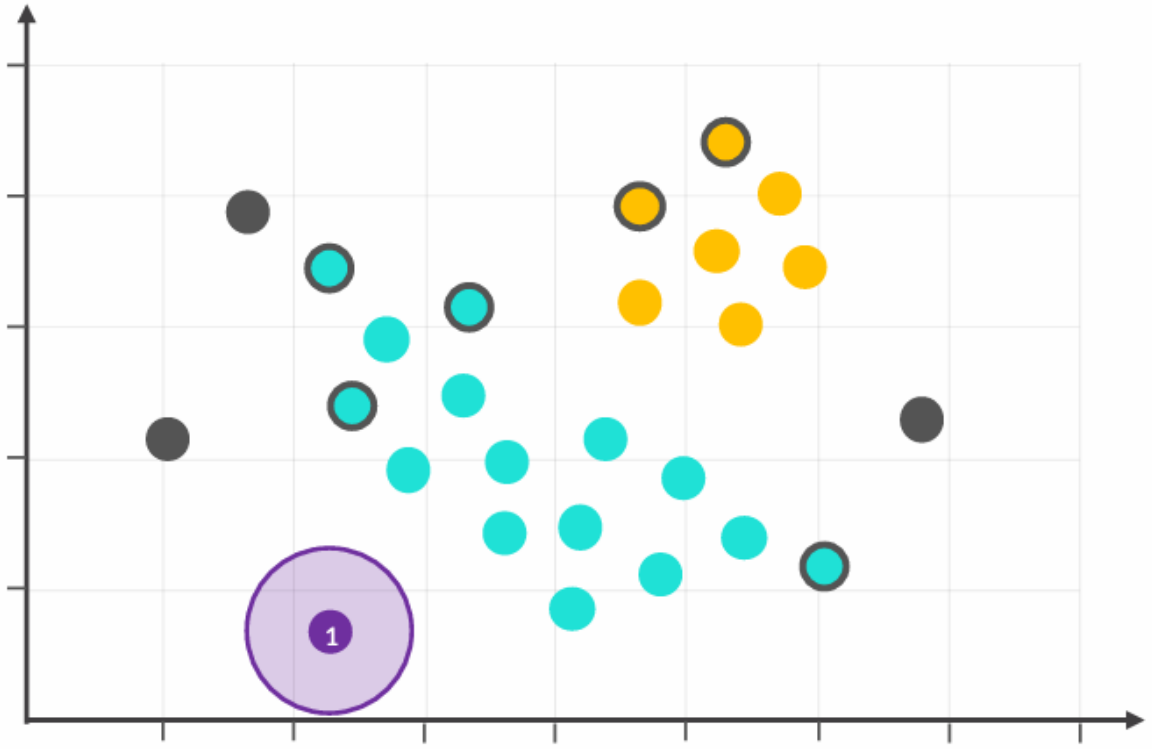
eps = 0.75
min_samples = 4

Step 5.10 - Move on to another random point and repeat the same steps



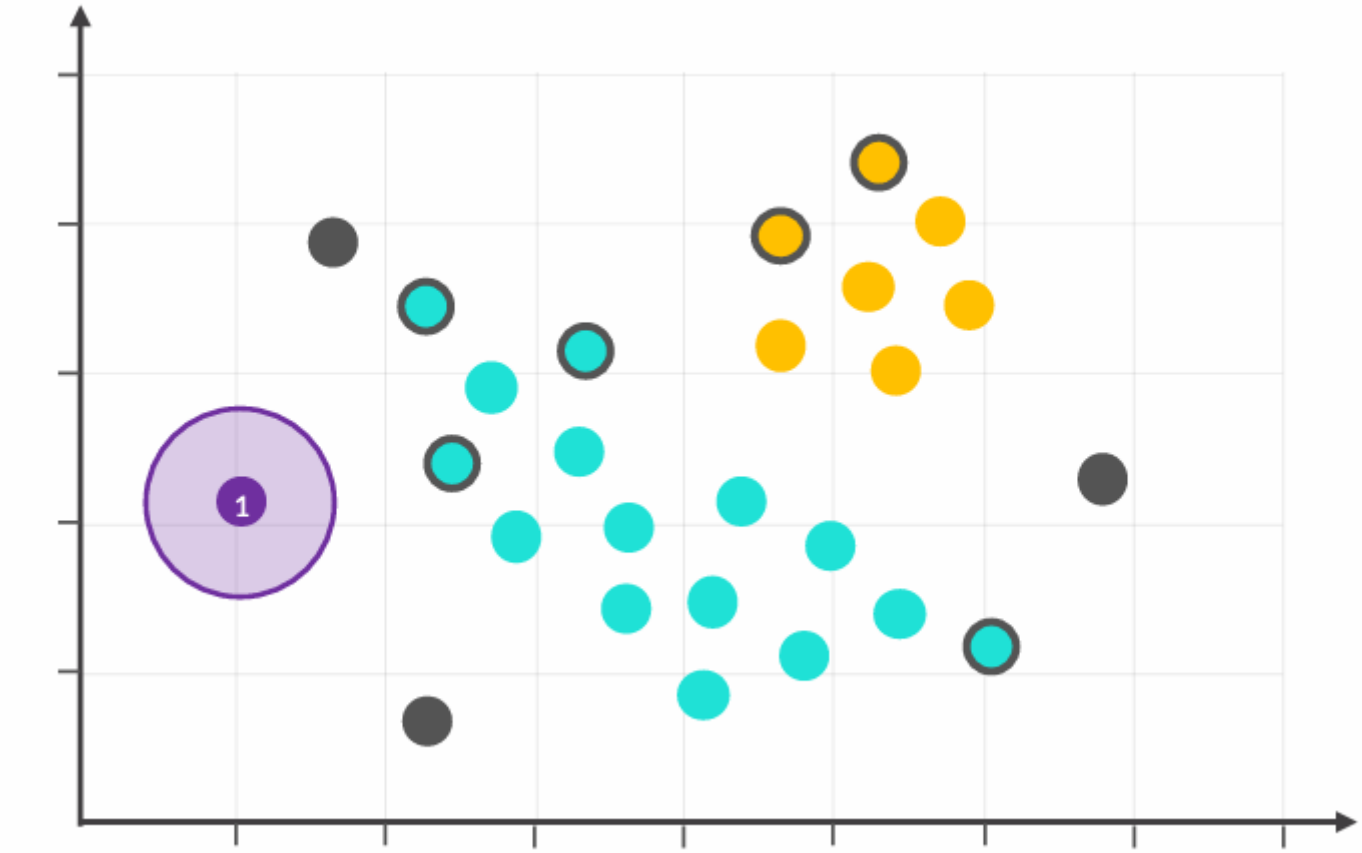
eps = 0.75
min_samples = 4

Step 5.11 - Move on to another random point and repeat the same steps



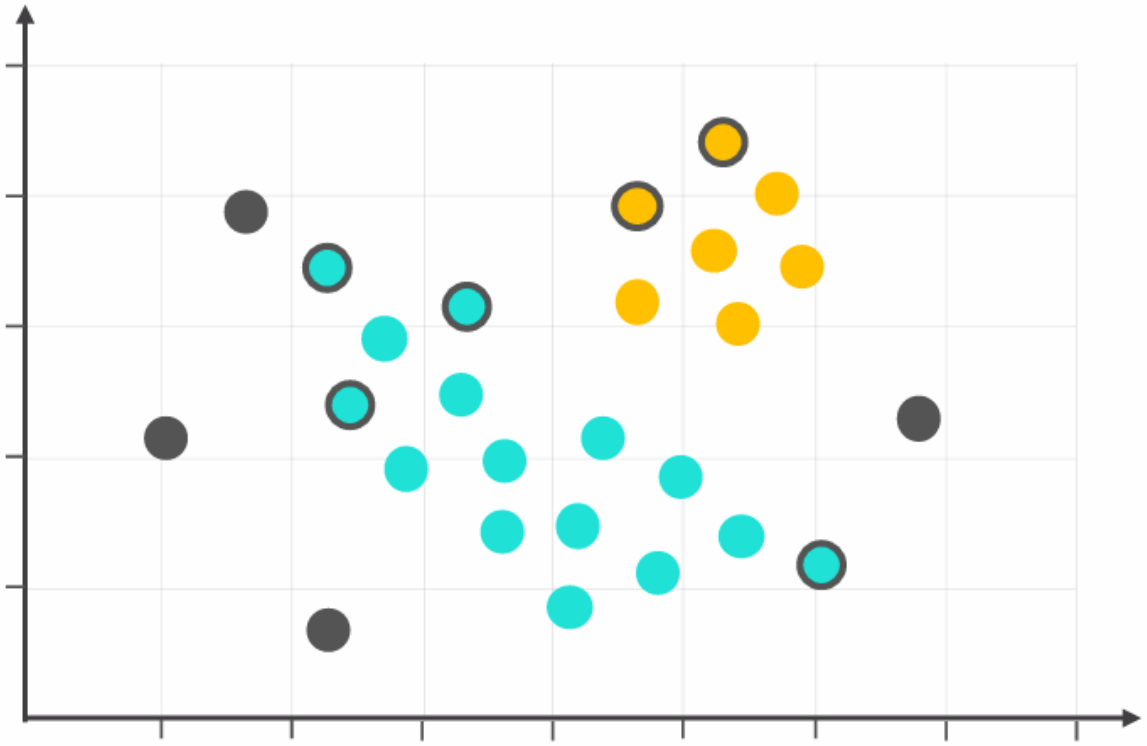
eps = 0.75
min_samples = 4

Step 5.12 - Move on to another random point and repeat the same steps



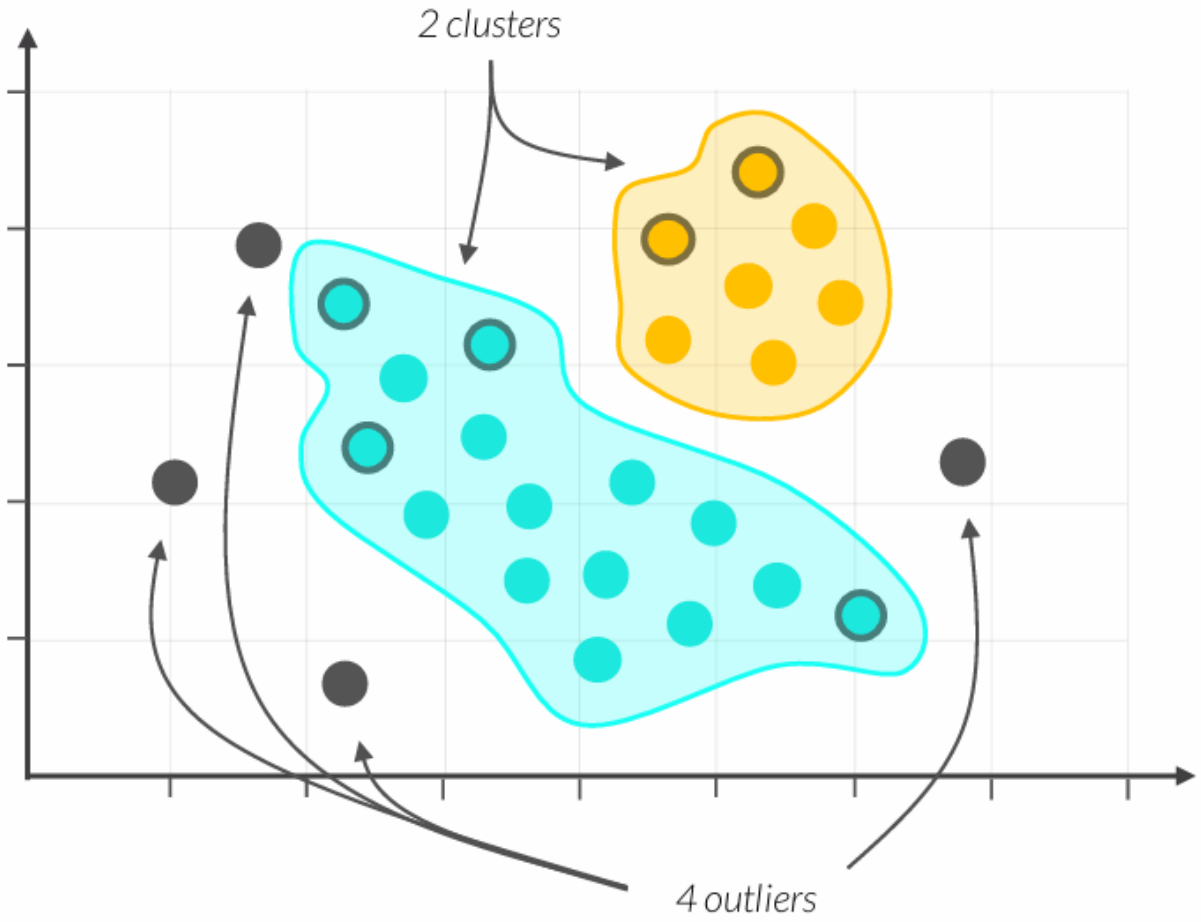
eps = 0.75
min_samples = 4

Step 5.13 - Move on to another random point and repeat the same steps



eps = 0.75
min_samples = 4

Step 5.14 - Move on to another random point and repeat the same steps



eps = 0.75
min_samples = 4

- **Core Points:** The points with a **solid color other than black** are core points. These are the points that have a high density of neighbors and form the center of a cluster.

- **Border Points:** The points with **outer black borders** are border points. These points are on the edge of a cluster and are within the radius of a core point but don't have enough neighbors to be core points themselves.

- **Noise Points/Outliers:** The **black-colored points** are noise points. They are in low-density regions and do not belong to any cluster.

Summarization - How it works ?

The working mechanism of DBSCAN can be described as follows:

1. Select a radius (eps) and a minimum number of points (min_samples)

2. Within a scatterplot, select a point at random and count the points within its radius

- If it's greater than or equal to the "min_samples", then start a cluster, label the point as a core point, and mark the points within its radius as a neighbor
- If it's less than the "min_samples", label the point as a noise point and move to step 5

3. Move to a neighbor and count the points within its radius

- If it's greater than or equal to the "min_samples", label it as a core point, and mark its neighbors
- If it's less than the "min_samples", but at least one of them is a core point, label it as a border point
- If it's less than the "min_samples", and none of them is a core point, label it as a noise point

4. Continue with another neighbor until all the points are labeled within the cluster

5. Move on to another random point and repeat the same steps

LET’S GO