COSC 3337 : Data Science I

N. Rizk

College of Natural and Applied Sciences

Department of Computer Science

University of Houston

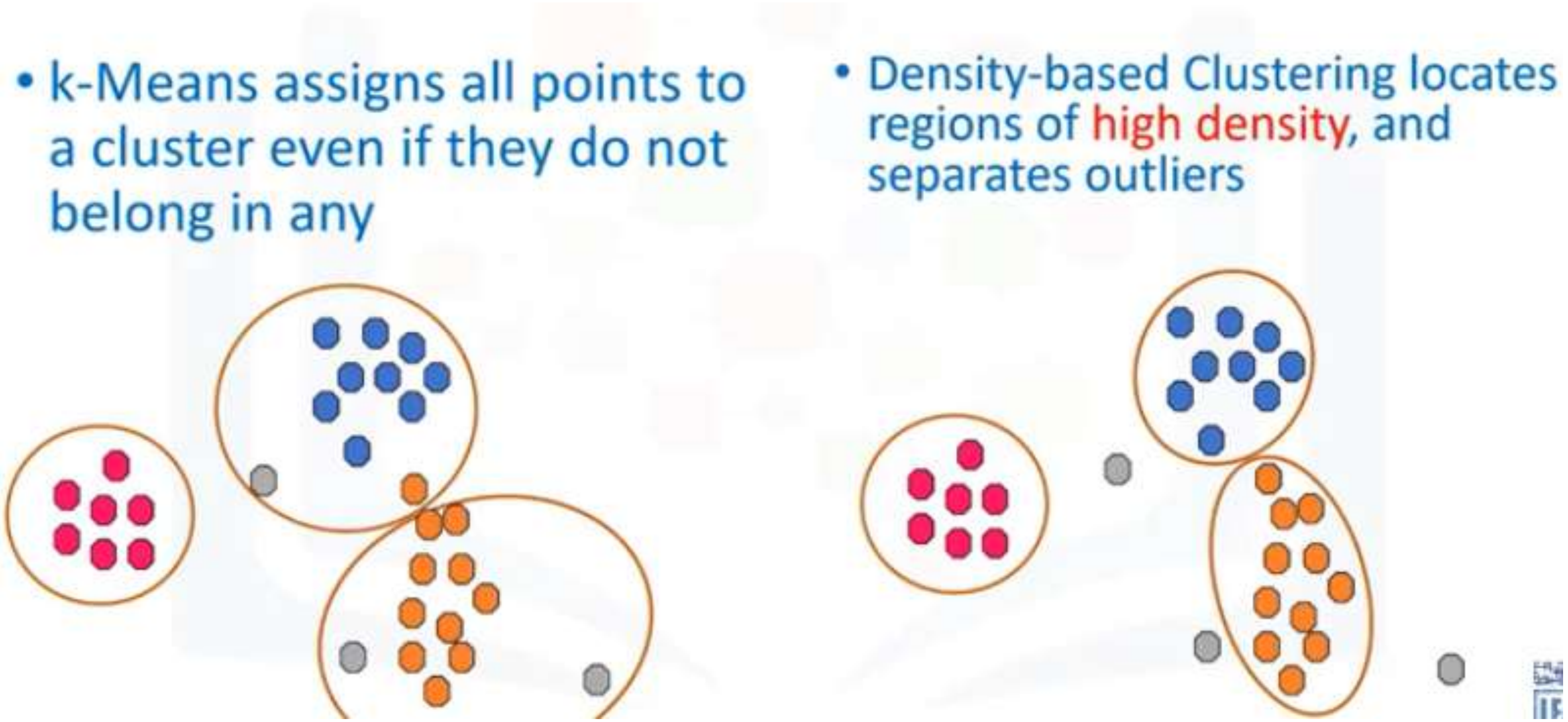
Density-based Approaches

- Why Density-Based Clustering methods?
  - Discover clusters of arbitrary shape.
  - Clusters – Dense regions of objects separated by regions of low density
- DBSCAN – the first density based clustering
- OPTICS – density based cluster-ordering
- DENCLUE – a general density-based description of cluster and clustering
- HDBSCAN Instead of taking an epsilon value as a cut level for the dendrogram, a different approach is taken
- AFFINITY PROPAGATION is a clustering algorithm based on the concept of "message passing" between data points

DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Proposed by Ester, Kriegel, Sander, and Xu (KDD96)
- Relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points.
- Discovers clusters of arbitrary shape in spatial databases with noise

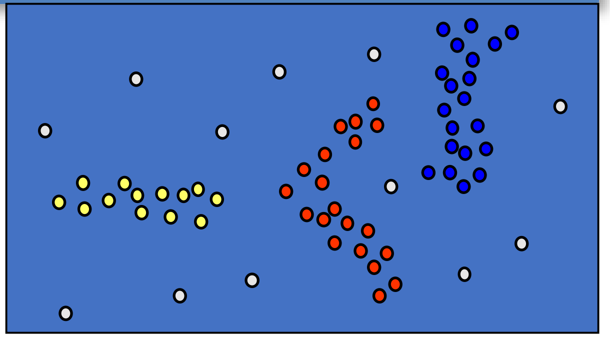
Kmeans vs DBscan



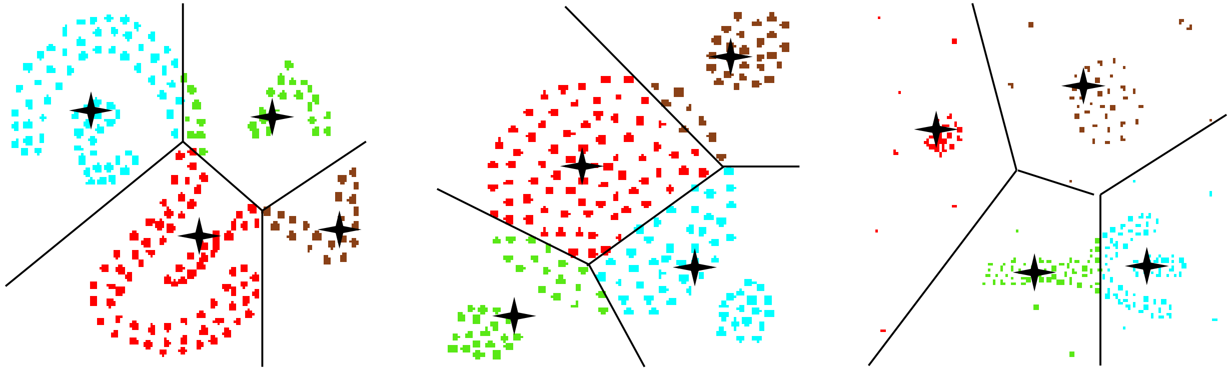
Density-Based Clustering

- *Basic Idea*:

  Clusters are dense regions in the data space, separated by regions of lower object density



- Why Density-Based Clustering?



Results of a *k*-medoid algorithm for *k*=4

Different density-based approaches exist (see Textbook & Papers)
Here we discuss the ideas underlying the DBSCAN algorithm

Density Based Clustering: Basic Concept

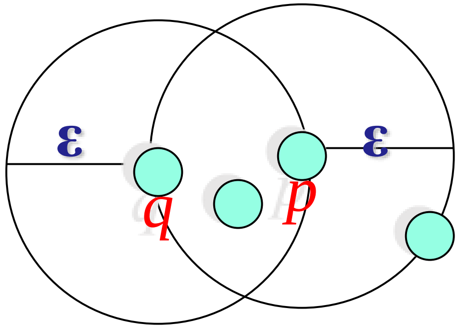
- Intuition for the formalization of the basic idea
  - For any point in a cluster, the local point density around that point has to exceed some threshold
  - The set of points from one cluster is spatially connected
- Local point density at a point *p* defined by two parameters
  - ε – radius for the neighborhood of point p:
    $N_{\varepsilon}(p) := \{q \text{ in data set } D \mid dist(p, q) \leq \varepsilon\}$
  - *MinPts* – minimum number of points in the given neighbourhood $N(p)$

ε-Neighborhood

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p):\{q \mid d(p,q) \leq \varepsilon\}$$

- “High density” - ε-Neighborhood of an object contains at least *MinPts* of objects.

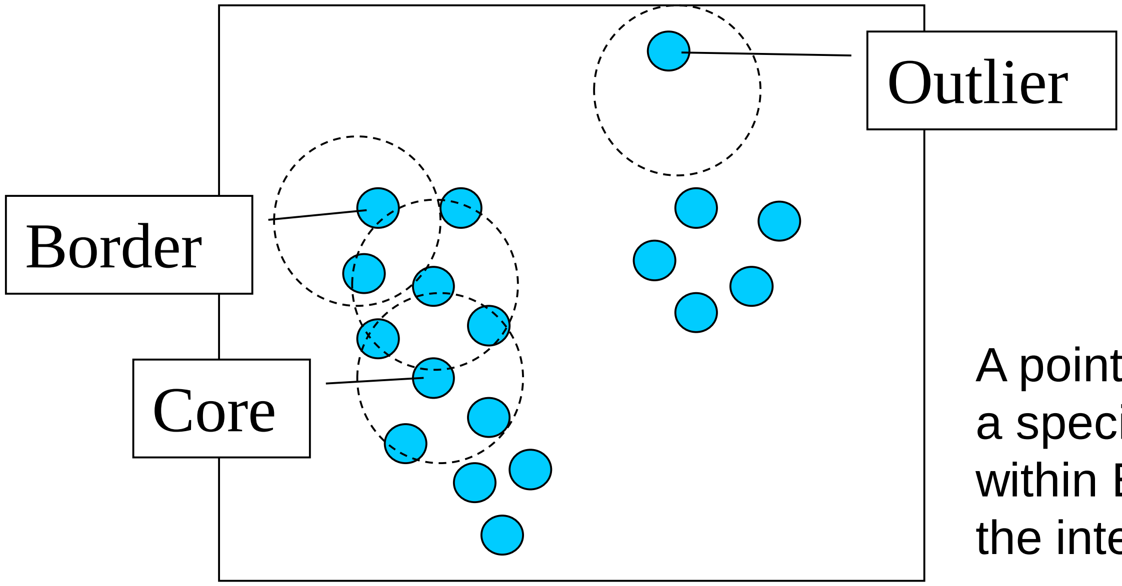


ε-Neighborhood of *p*
ε-Neighborhood of *q*

Density of p is “high” (MinPts = 4)

Density of q is “low” (MinPts = 4)

Core, Border & Outlier



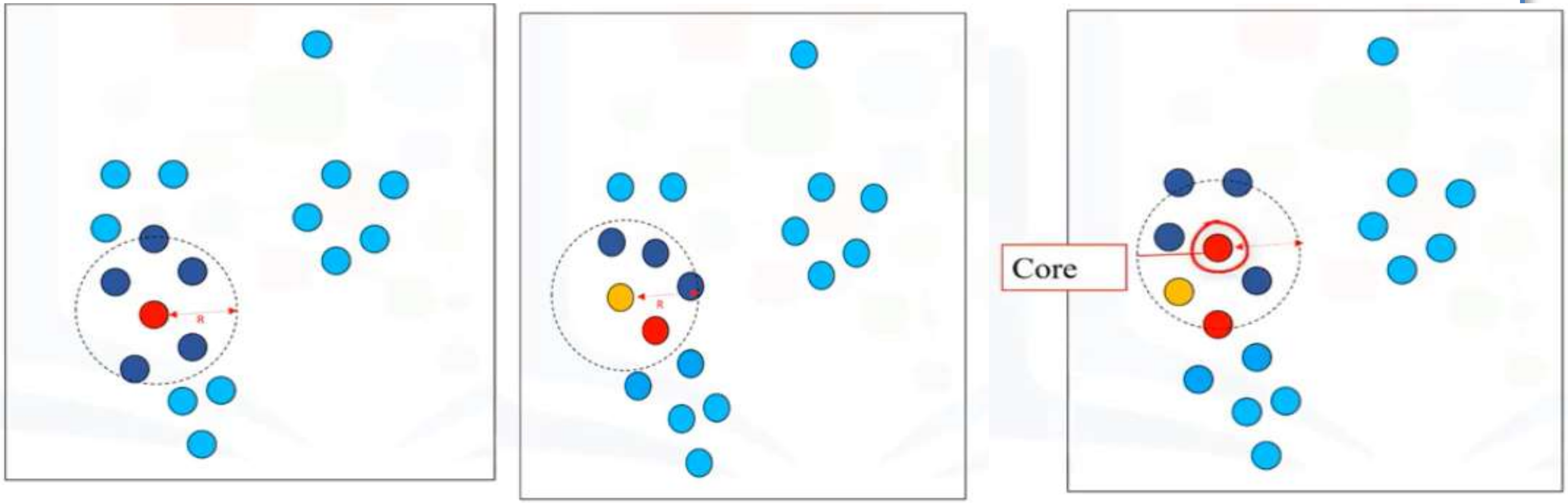
Given ε and *MinPts*, categorize the objects into three exclusive groups.

A point is a core point if it has more than a specified number of points (MinPts) within Eps These are points that are at the interior of a cluster.

ε = 1unit, MinPts = 5

A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

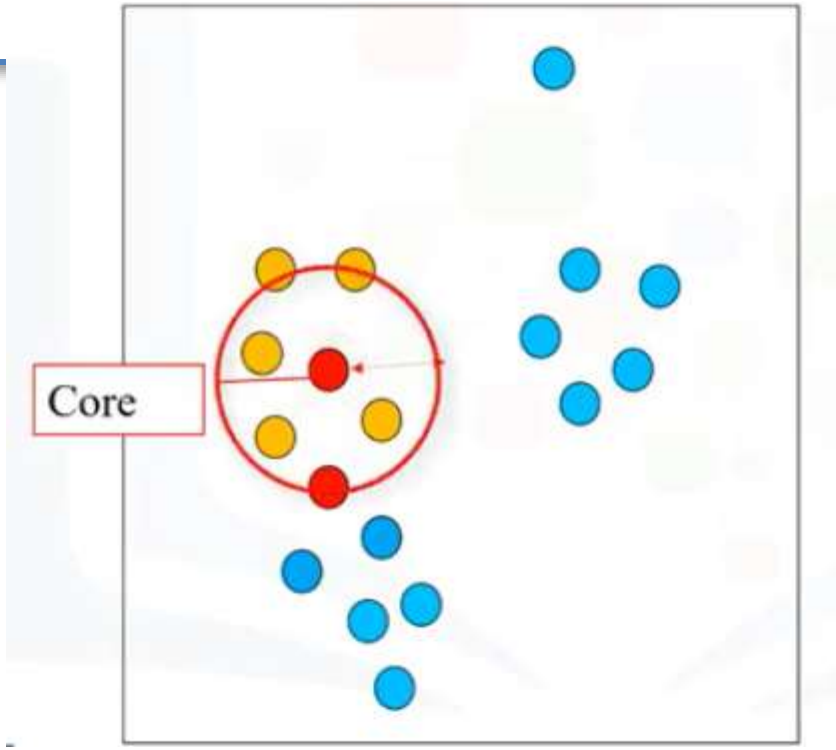


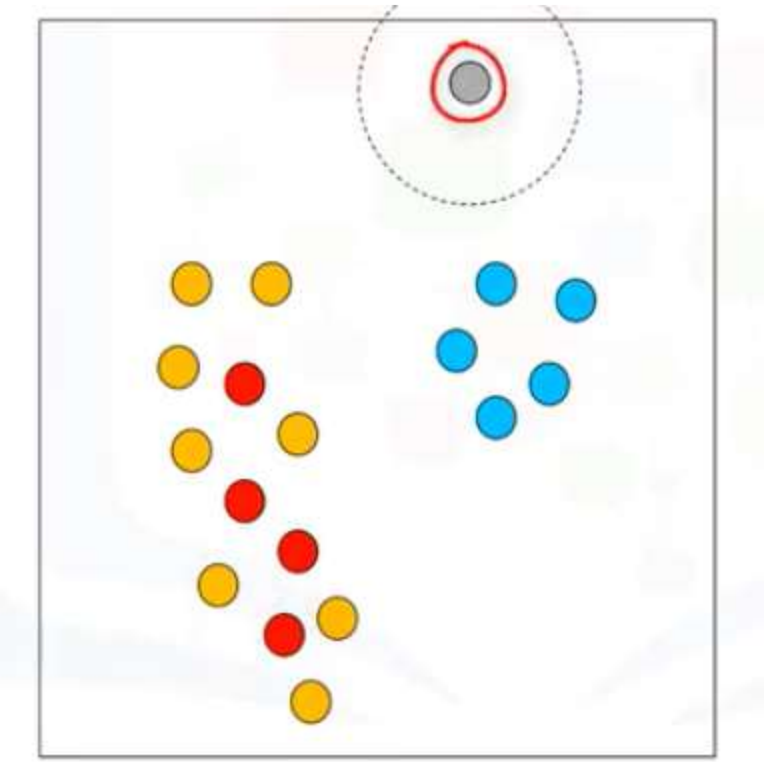
Border point if

1- less than Minpoints within radius

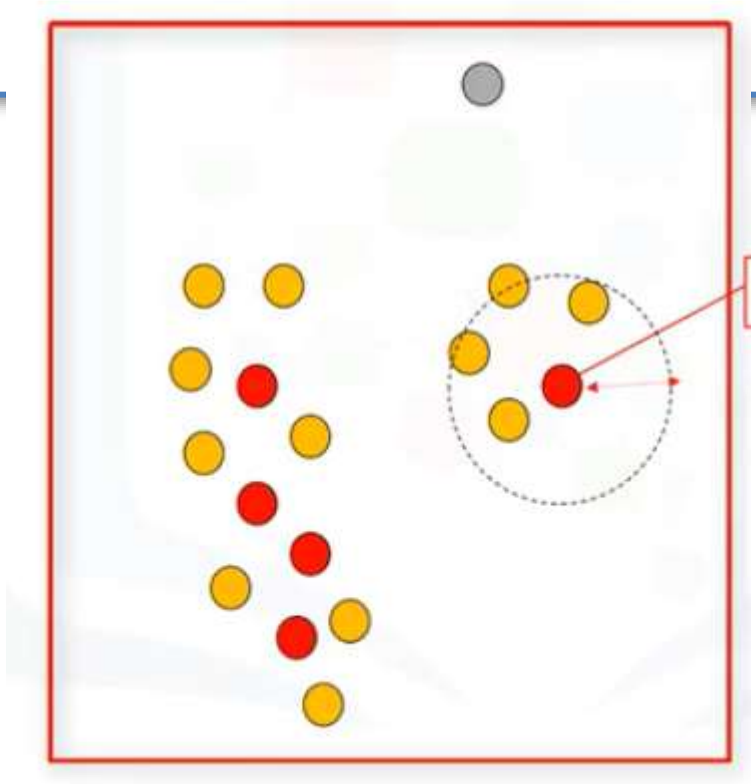
or

2- reachable from core point



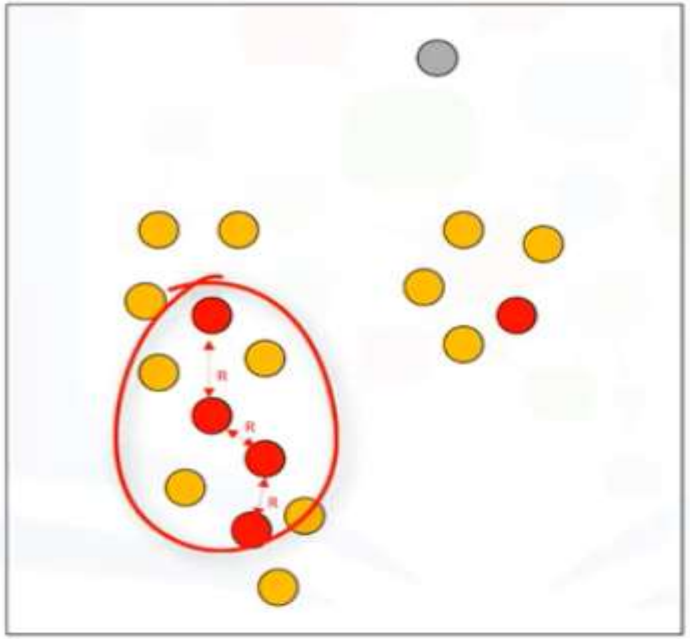


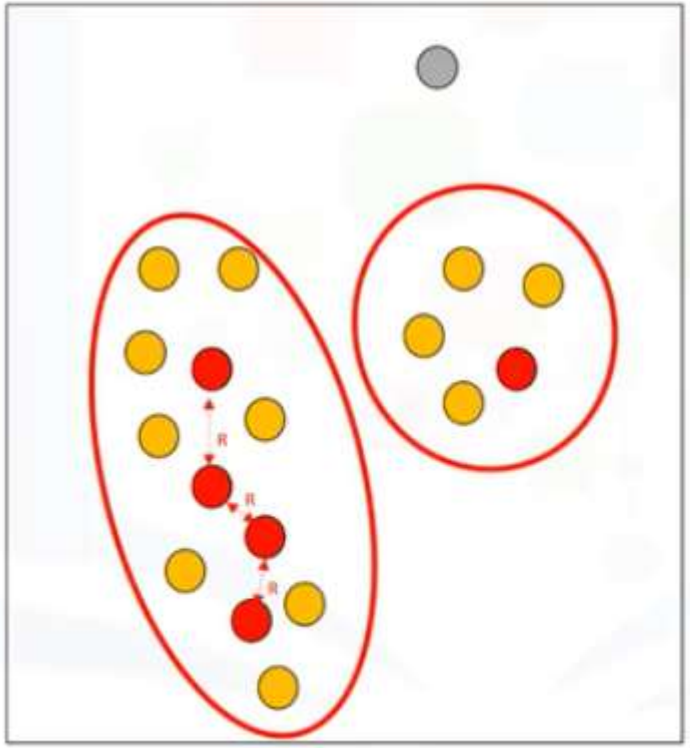
Outlier point if
  1- it is not a core point
  or
  2- NOT reachable from core point



Visit all points and label them as
Core
Border
Outlier/Noise

Connect all core points

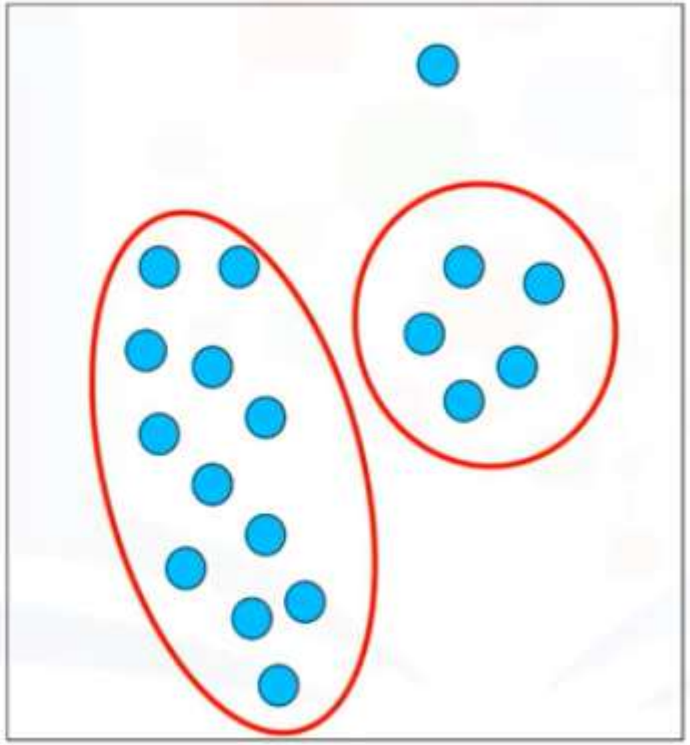




And all their borders to form a cluster

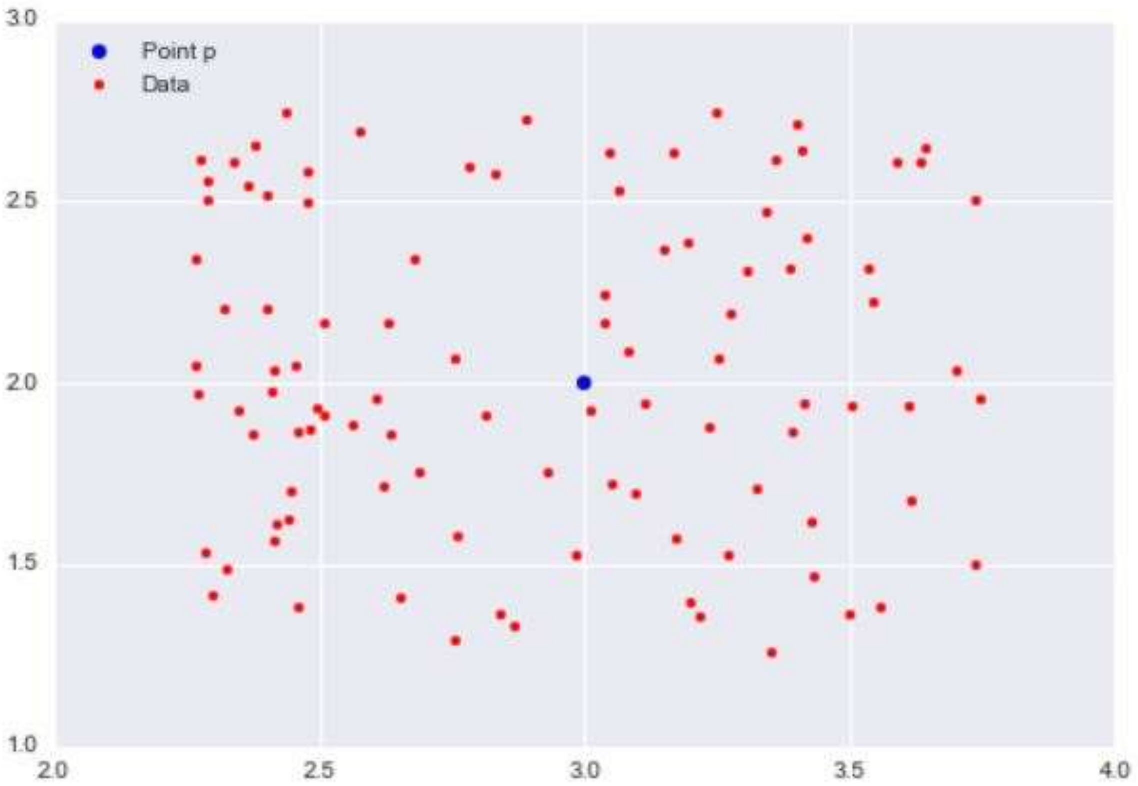
DB_Scan

Advantage of DBSCAN

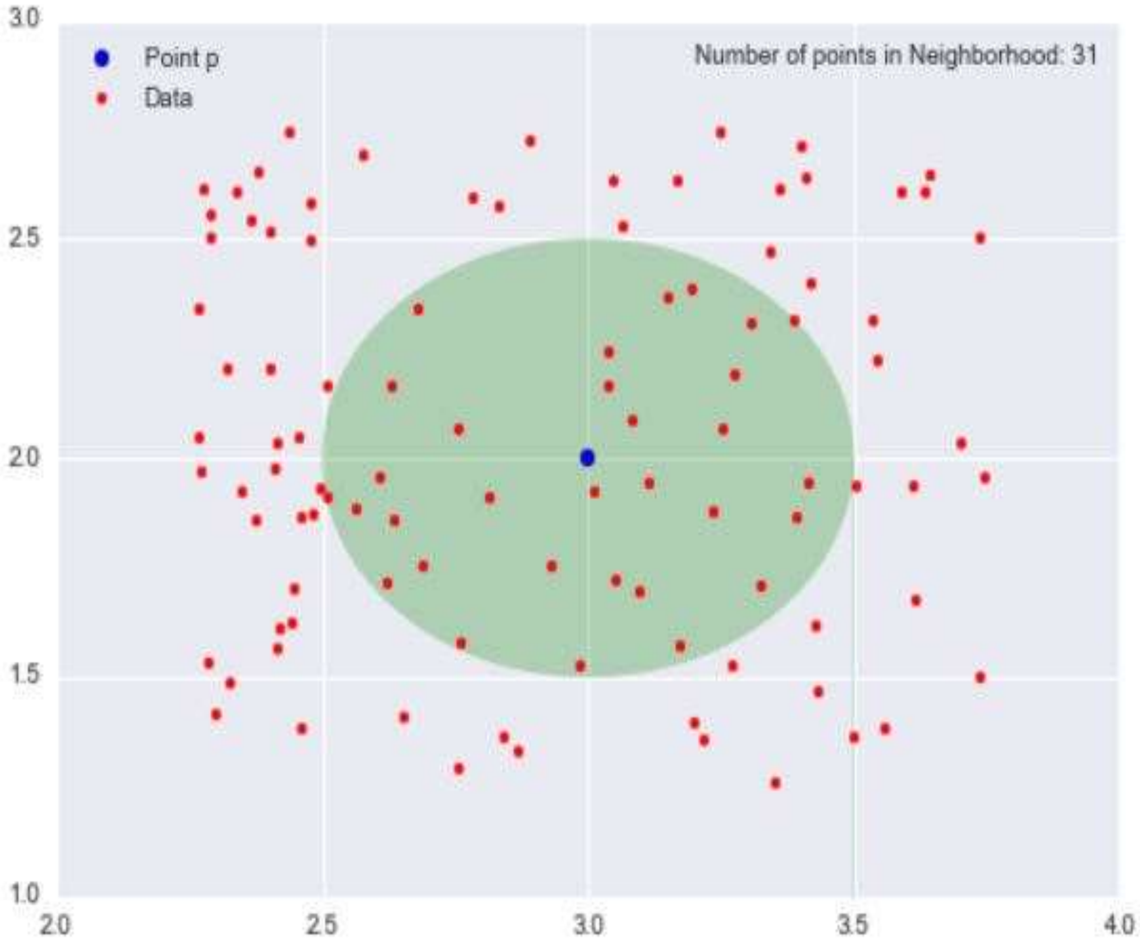


1. Arbitrarily shaped clusters
2. Robust to outliers
3. Does not require specification of the number of clusters

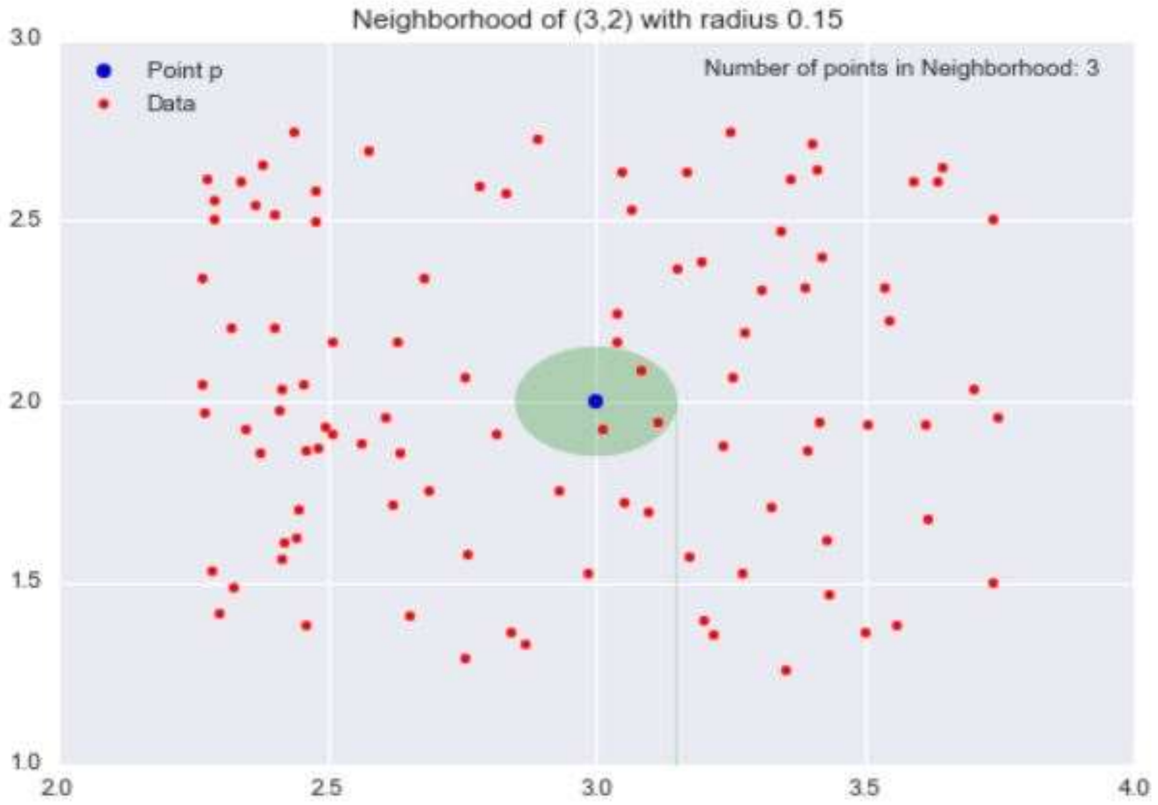
100 data points, select (3,2)



ε = 0.5➔ 31 points

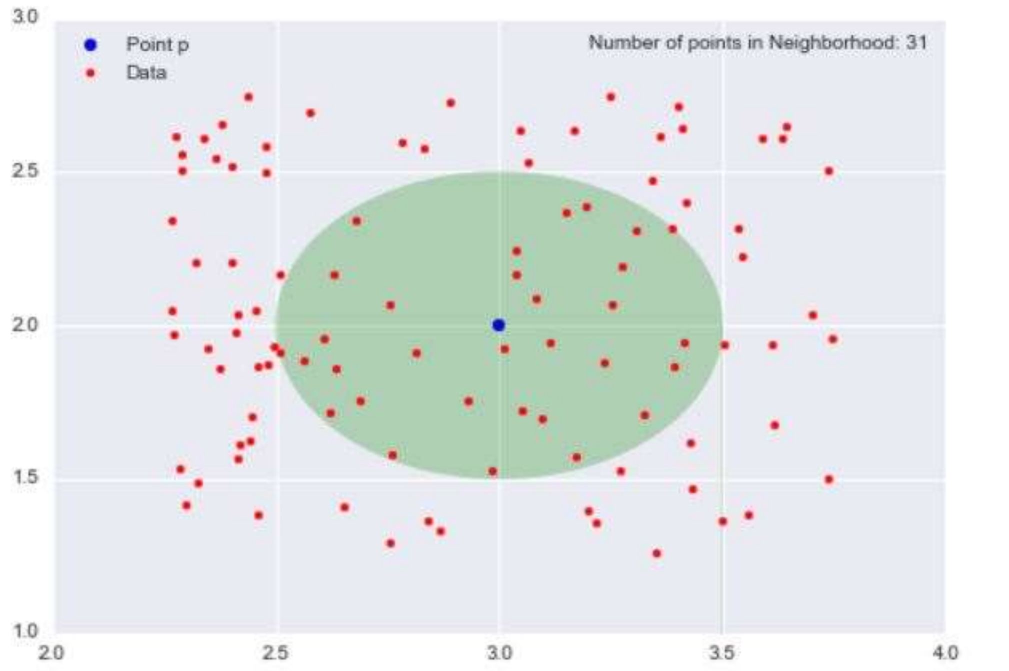


ε = 0.15➔ 3 points



Decreasing ε from 0.5 to 0.15 (a 70% reduction), the number of points is decreased in our neighborhood from 31 to 3 (a 90% reduction)

density = mass/volume

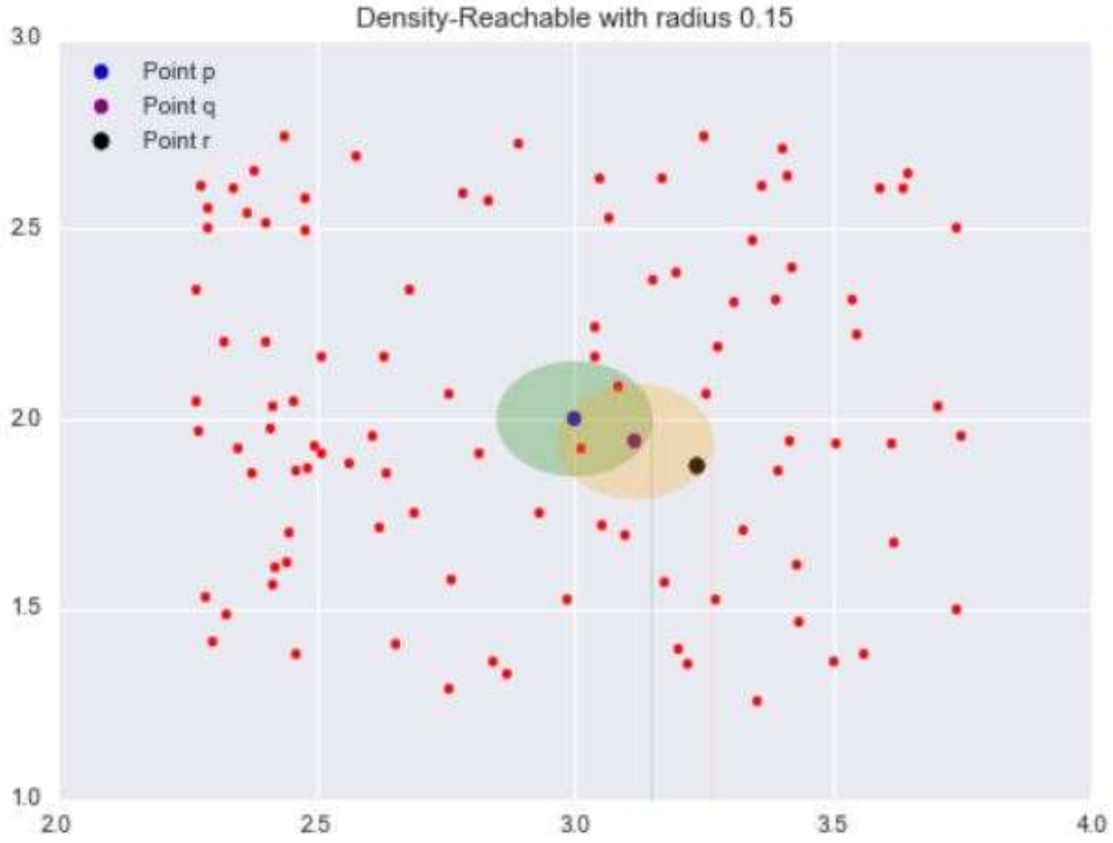


density = number of data points/Л0.5^2

local density approximation at * p = (3,2) is calculated as
density = mass/volume = 31/(π/4) = 124/π ~= 39.5

➔Cluster points who have similar local density approximations

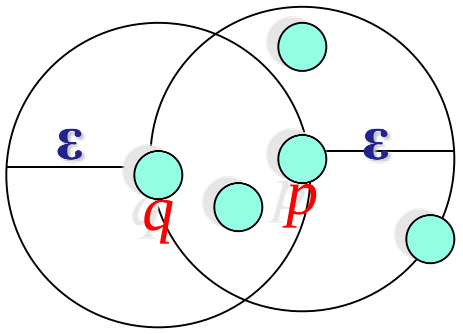
- Starting at a point p, then the point r is density-reachable (friends of friend) from the point p.
- (directly-reachable of a core point p are its “friends”)
- “friends of a friend of a friend … of a friend” are also density-reachable



By picking larger values of ε, more points become density-reachable, and by choosing smaller values of ε, fewer points become density-reachable.

Density-Reachability

- **Directly density-reachable**
  - **An object q is directly density-reachable from object p if p is a core object and q is in p’s ε-neighborhood.**

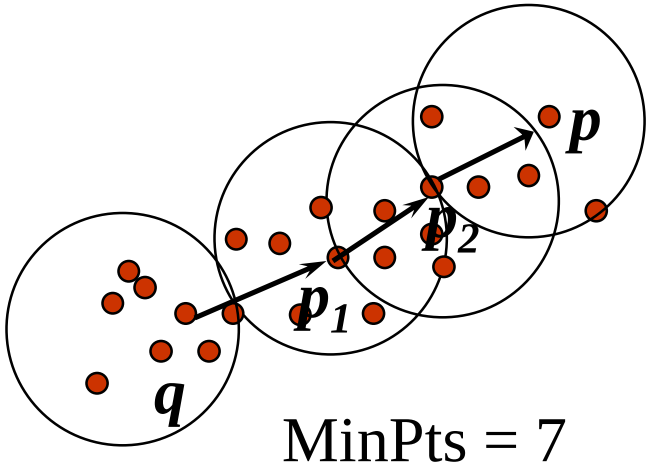


MinPts = 4

- **q is directly density-reachable from p**
- **p is not directly density- reachable from q?**
- **Density-reachability is asymmetric.**

Density-reachability

- Density-Reachable (directly and indirectly):
  - A point p is directly density-reachable from p2;
  - p2 is directly density-reachable from p1;
  - p1 is directly density-reachable from q;
  - p←p2←p1←q form a chain.

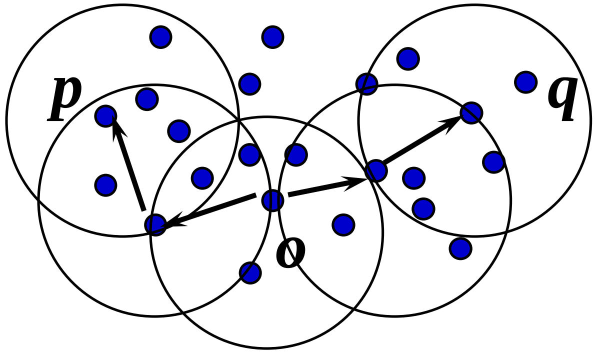


MinPts = 7

- p is (indirectly) density-reachable from q
- q is not density- reachable from p?

Density-Connectivity

- **Density-reachable is not symmetric**
  - **not good enough to describe clusters**
- **Density-Connected**
  - **A pair of points p and q are density-connected if they are commonly density-reachable from a point o.**



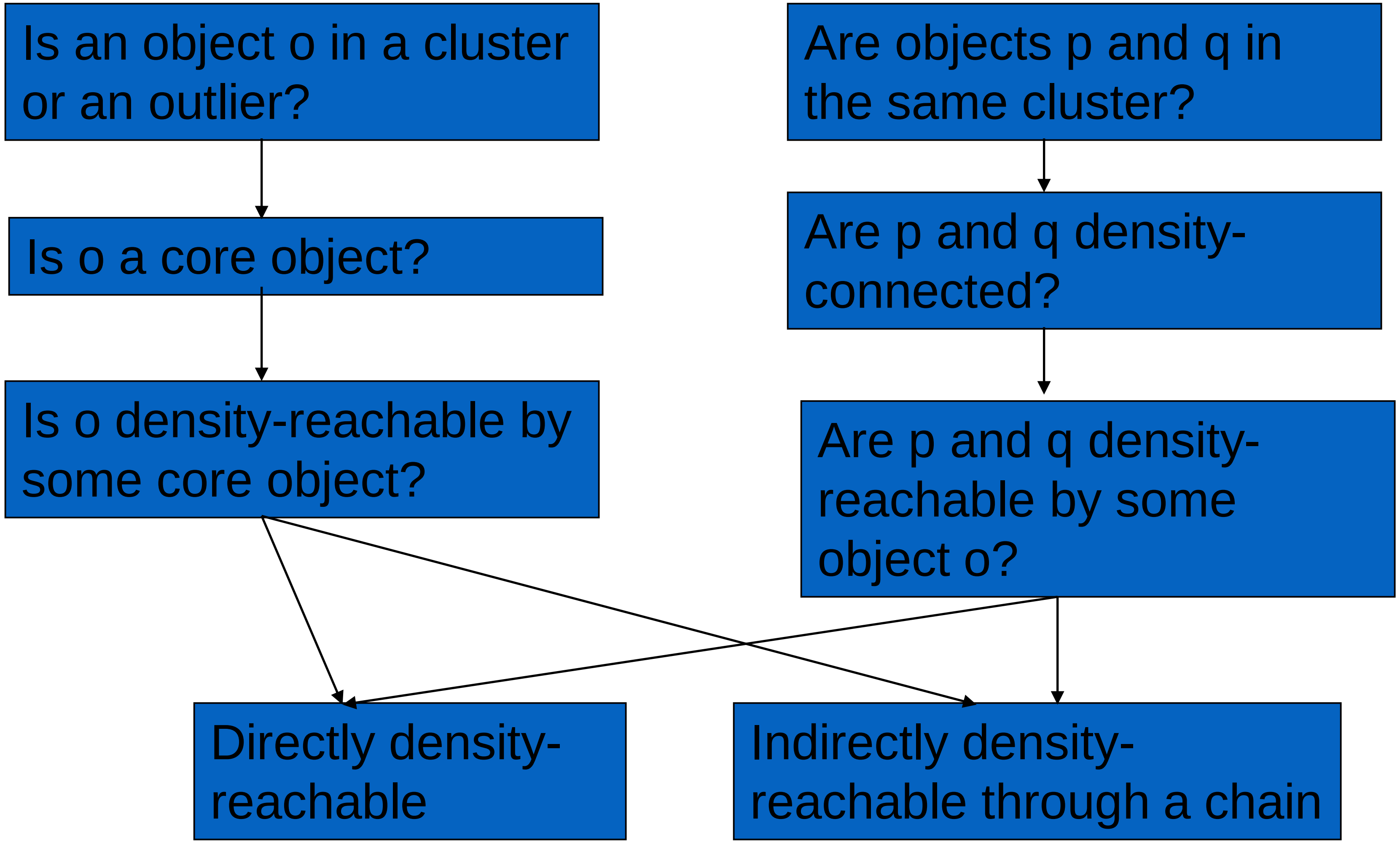
- **Density-connectivity is symmetric**

Formal Description of Cluster

- Given a data set D, parameter ε and threshold MinPts.
- A cluster C is a subset of objects satisfying two criteria:
  - *Connected:* $\forall$ p,q $\in$C: p and q are density-connected.
  - *Maximal:* $\forall$ p,q: if p $\in$C and q is <u>density-reachable from p</u>, then q $\in$C. (avoid redundancy)

P is a core object.

Review of Concepts



DBSCAN Algorithm

```
Input: The data set D

Parameter: ε, MinPts

For each object p in D
    if p is a core object and not processed then
        C = retrieve all objects density-reachable from p
        mark all objects in C as processed
        report C as a cluster
    else mark p as outlier
    end if

End For
```

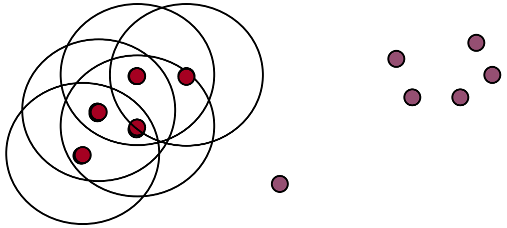
DBScan Algorithm

DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* wrt *Eps* and *MinPts*.
- If *p* is a core point, a cluster is formed.
- If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

DBSCAN Algorithm: Example

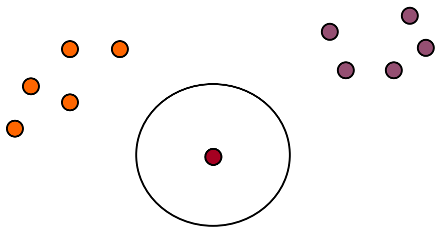
- Parameter
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

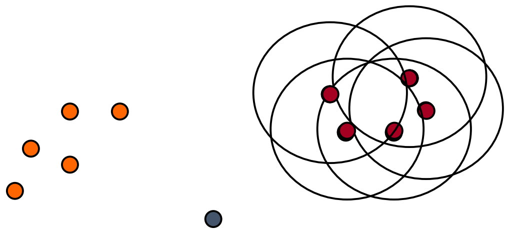
- Parameter
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

- Parameter
  - ε = 2 cm
  - *MinPts* = 3



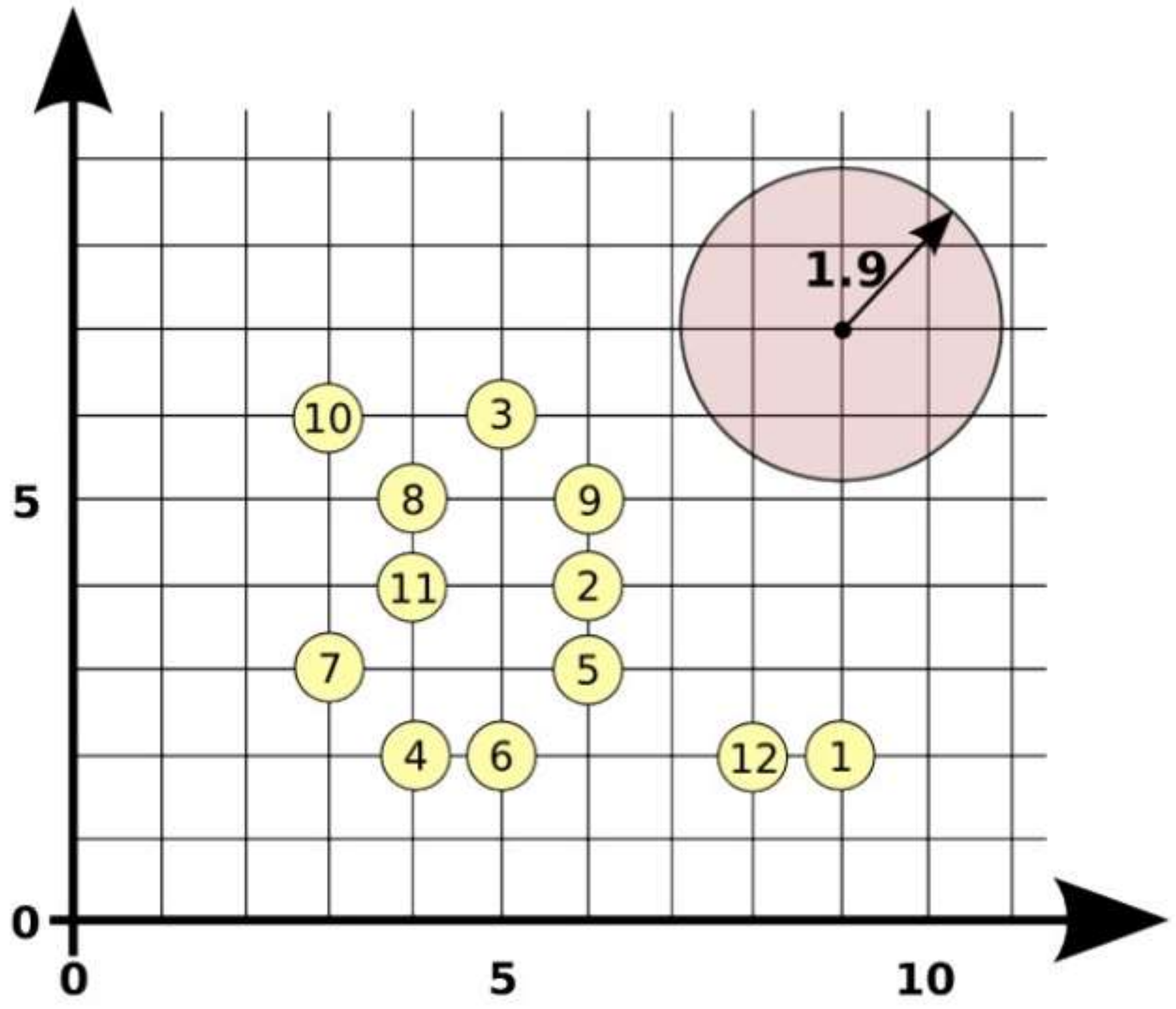
for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN eps=2 Minpts=2

If Epsilon is 2 and minpoint is 2, what are the clusters that DBScan would discover with the following 8 examples: A1=(2,10), A2=(2,5), A3=(8,4), A4=(5,8), A5=(7,5), A6=(6,4), A7=(1,2), A8=(4,9).

1. Create the distance matrix using Euclidean
2. Find Neighborhood of each points
3. Define core,border,outliers
4. Define clusters

DBSCAN eps=1.9 Minpts=4



```python
# Let's import all your dependencies first

from sklearn.cluster import DBSCAN
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
# Import .csv file and convert it to a DataFrame object
df = pd.read_csv("customers.csv");

print(df.head())
```

```
   Channel  Region  Fresh  Milk  Grocery  Frozen  Detergents_Paper  Delicassen
0        2       3  12669  9656     7561     214              2674        1338
1        2       3   7057  9810     9568    1762              3293        1776
2        2       3   6353  8808     7684    2405              3516        7844
3        1       3  13265  1196     4221    6404               507        1788
4        2       3  22615  5410     7198    3915              1777        5185
```

```python
#FRESH: annual spending (m.u.) on fresh products (Continuous);
#MILK: annual spending (m.u.) on milk products (Continuous);
#GROCERY: annual spending (m.u.)on grocery products (Continuous);
#FROZEN: annual spending (m.u.)on frozen products (Continuous)
#DETERGENTS_PAPER: annual spending (m.u.) on detergents and paper products (Continuous)
#DELICATESSEN: annual spending (m.u.)on and delicatessen products (Continuous);
#CHANNEL: customersâ€™ Channel - Horeca (Hotel/Restaurant/CafÃ©) or Retail channel (Nominal) REGION
```

```python
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 8 columns):
Channel             440 non-null int64
```

```
# there is no missing value in the dataset and all the data is integer in type
```

```
print(df.describe())
```

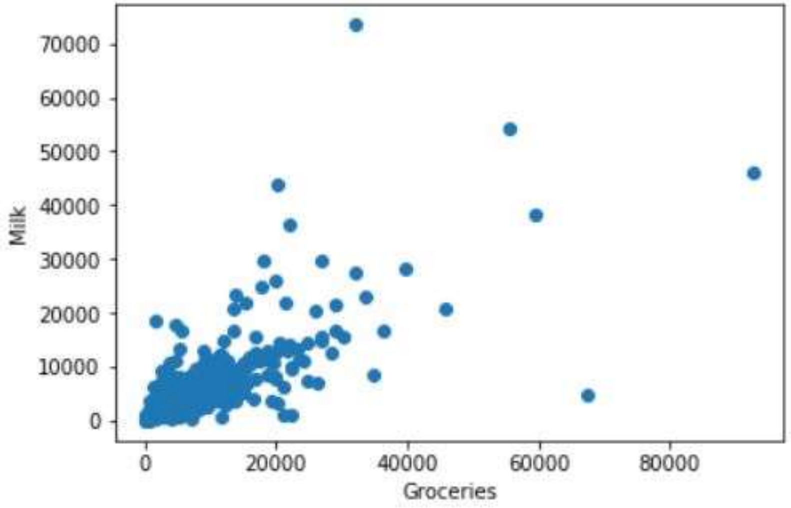
```
          Channel      Region          Fresh          Milk       Grocery  \
count  440.000000  440.000000     440.000000    440.000000    440.000000
mean     1.322727    2.543182   12000.297727   5796.265909   7951.277273
std      0.468052    0.774272   12647.328865   7380.377175   9503.162829
min      1.000000    1.000000       3.000000     55.000000      3.000000
25%      1.000000    2.000000    3127.750000   1533.000000   2153.000000
50%      1.000000    3.000000    8504.000000   3627.000000   4755.500000
75%      2.000000    3.000000   16933.750000   7190.250000  10655.750000
max      2.000000    3.000000  112151.000000  73498.000000  92780.000000

             Frozen  Detergents_Paper    Delicassen
count    440.000000        440.000000    440.000000
mean    3071.931818       2881.493182   1524.870455
std     4854.673333       4767.854448   2820.105937
min       25.000000          3.000000      3.000000
25%      742.250000        256.750000    408.250000
50%     1526.000000        816.500000    965.500000
75%     3554.250000       3922.000000   1820.250000
max    60869.000000      40827.000000  47943.000000
```

```
#most of the data in this dataset is continuous in nature except for two features: Channel and Region so we can drop them
df.drop(["Channel", "Region"], axis = 1, inplace = True)
```

```
# Let's get a view of the data after the drop

print(df.head())
```

```
   Fresh  Milk  Grocery  Frozen  Detergents_Paper  Delicassen
0  12669  9656     7561     214              2674        1338
1   7057  9810     9568    1762              3293        1776
2   6353  8808     7684    2405              3516        7844
```

```
# Let's plot the data now
x = df['Grocery']
y = df['Milk']
plt.scatter(x,y)
plt.xlabel("Groceries")
plt.ylabel("Milk")
plt.show()
```




```
#normalize each attribute by scaling it to 0 mean and unit variance.
```

```
df = df[["Grocery", "Milk"]]
df = df.as_matrix().astype("float32", copy = False)
```

```
stscaler = StandardScaler().fit(df)
df = stscaler.transform(df)
```

```
dbsc = DBSCAN(eps = .5, min_samples = 15).fit(df)
```

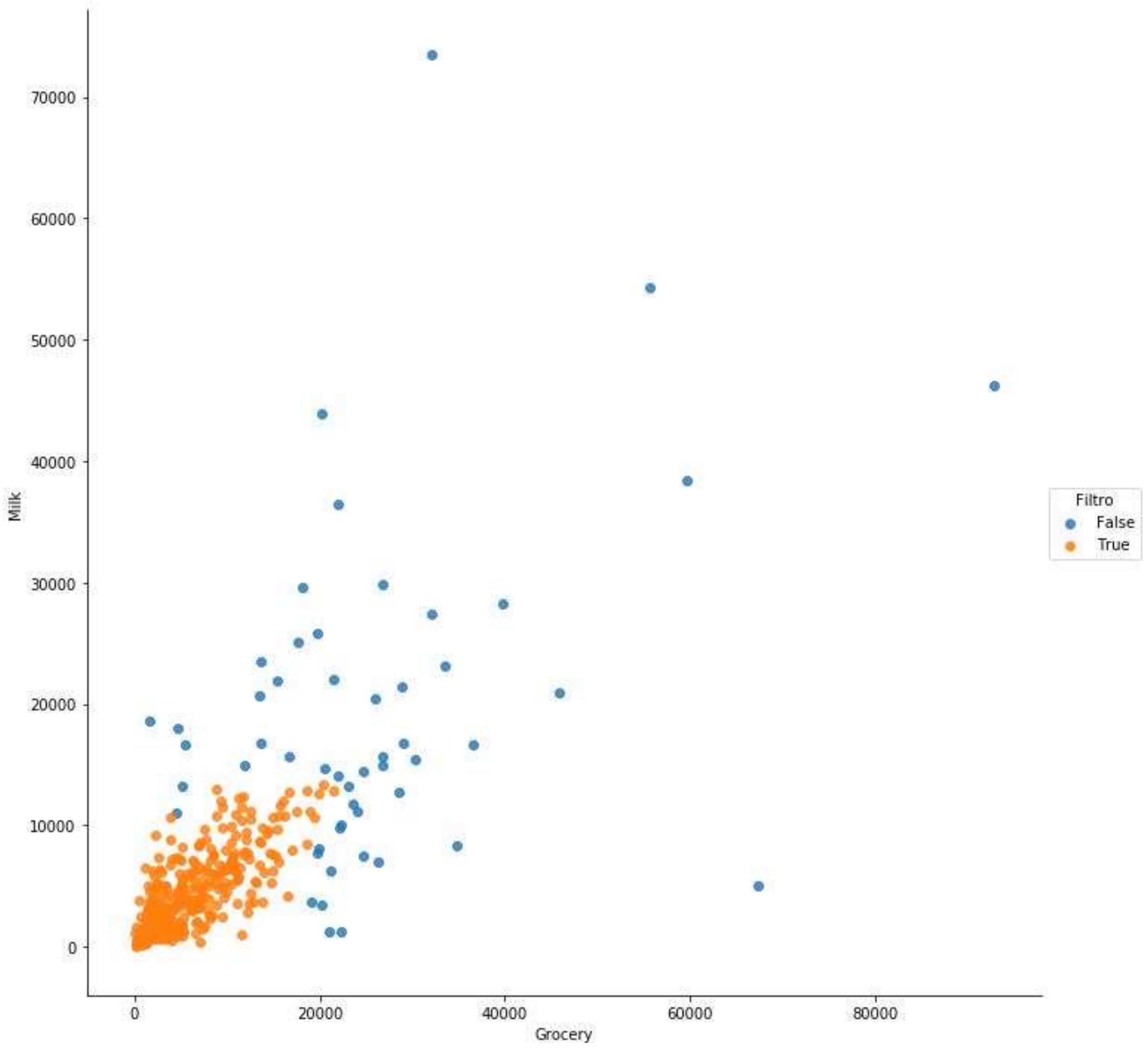
```
labels = dbsc.labels_
core_samples = np.zeros_like(labels, dtype = bool)
core_samples[dbsc.core_sample_indices_] = True
```

```
filtro=list(core_samples)

filtro
```

```
 False,
 True,
 True,
 True,
 True,
 True,
 True,
 True,
 True,
 True,
 False,
 True,
 True,
 True,
 True,
 True,
 True,
 False,
 False,
```

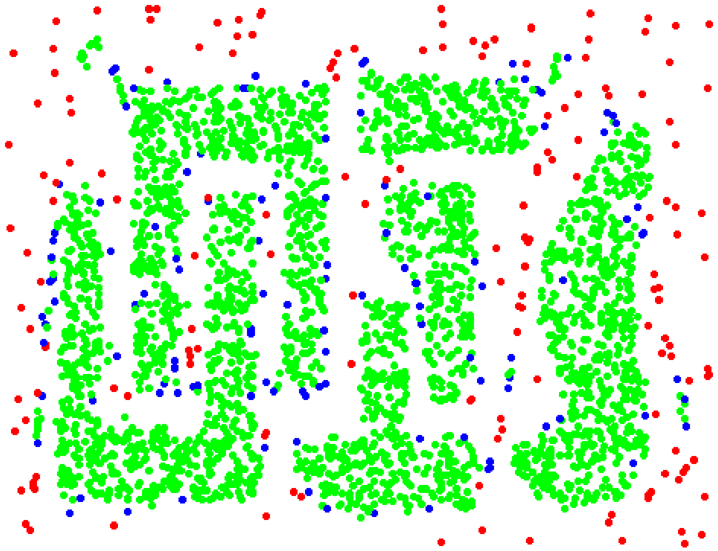
```
df0 = pd.read_csv("customers.csv");
df0["Filtro"]=filtro

sns.lmplot("Grocery","Milk",data=df0,fit_reg=False,hue="Filtro",size=10)
```

<seaborn.axisgrid.FacetGrid at 0x248ce343828>



Example

Original Points

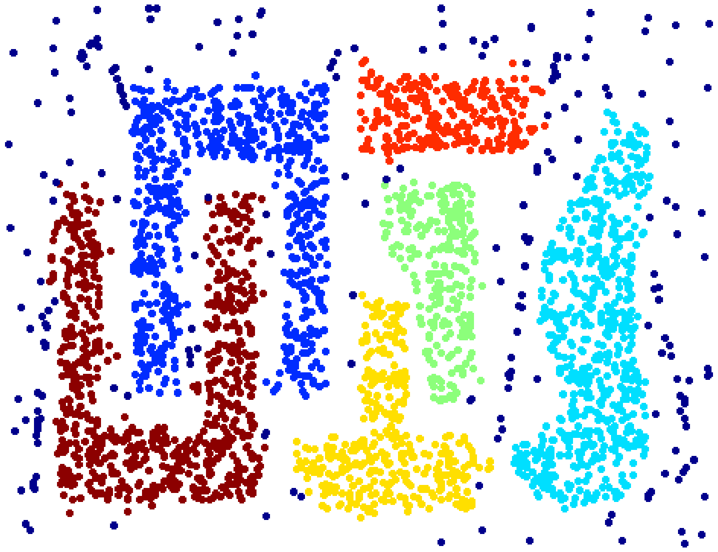


Point types: core, border and outliers

ε = 10, MinPts = 4

When DBSCAN Works Well

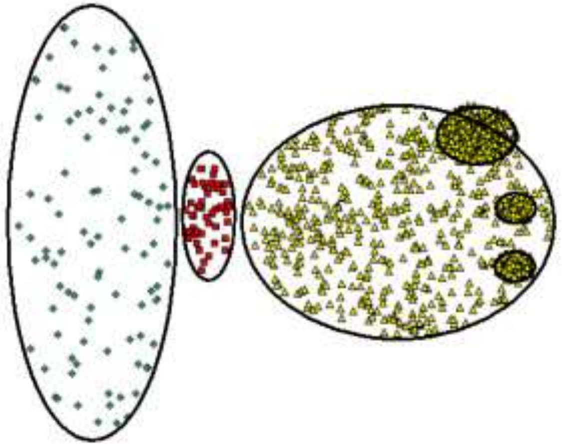
Original Points



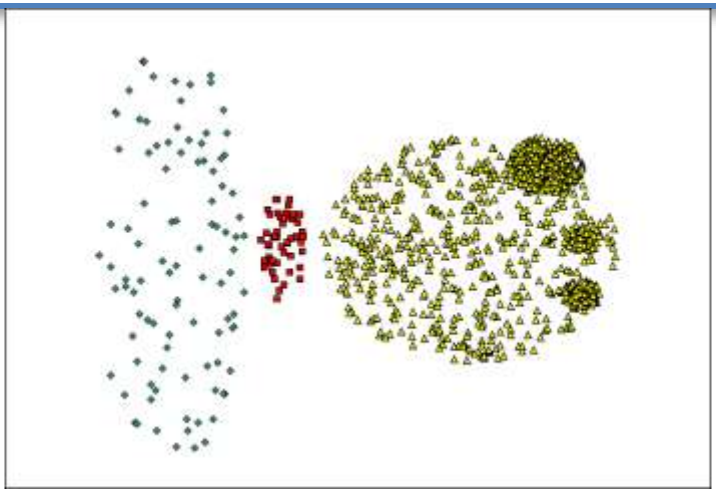
Clusters

- **Resistant to Noise**
- **Can handle clusters of different shapes and sizes**

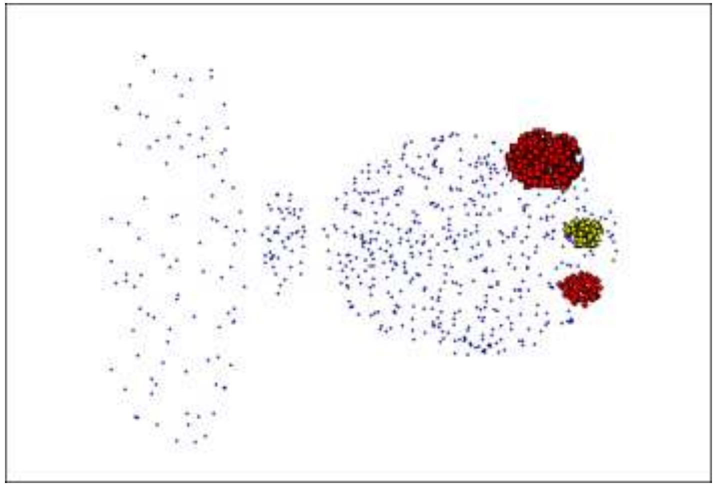
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

- **Cannot handle Varying densities**
- **sensitive to parameters**

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

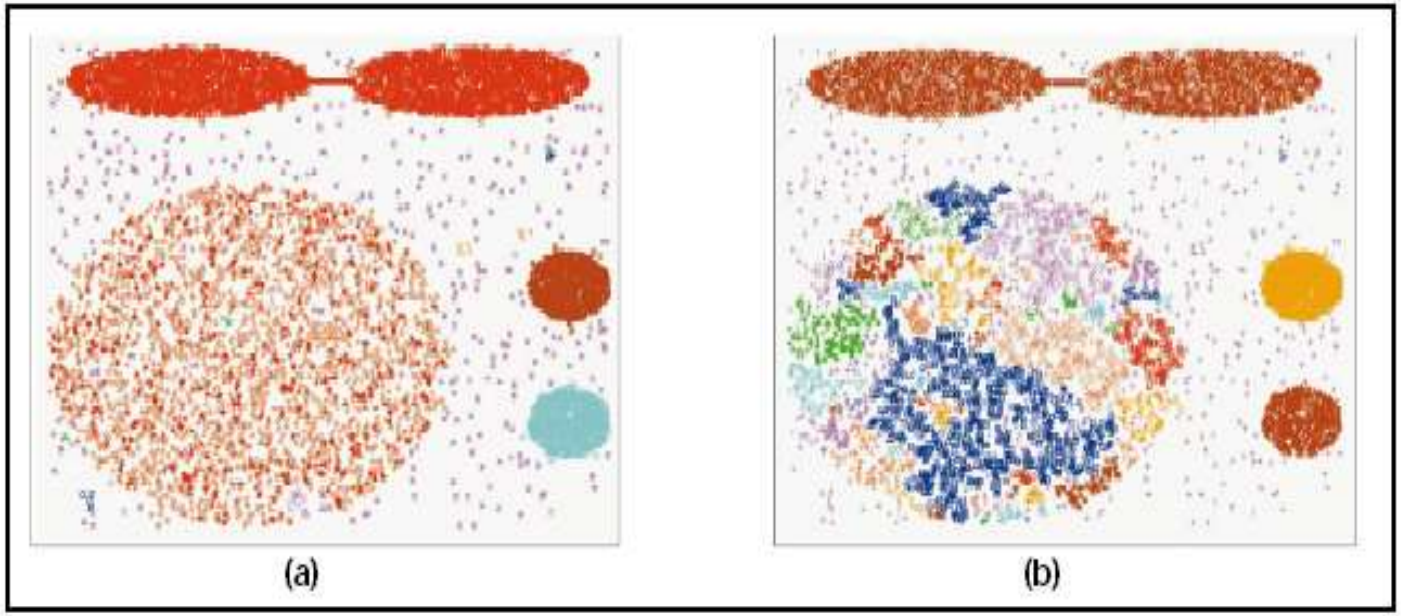
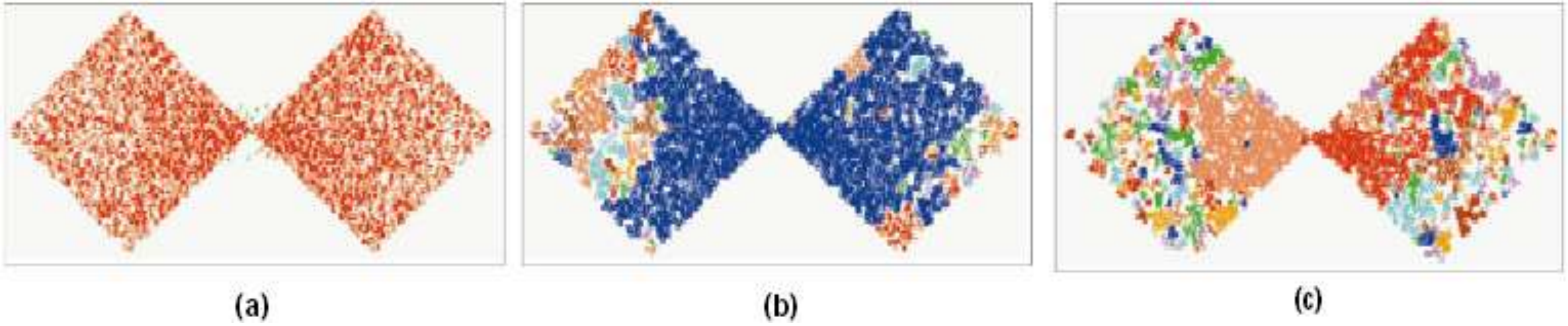
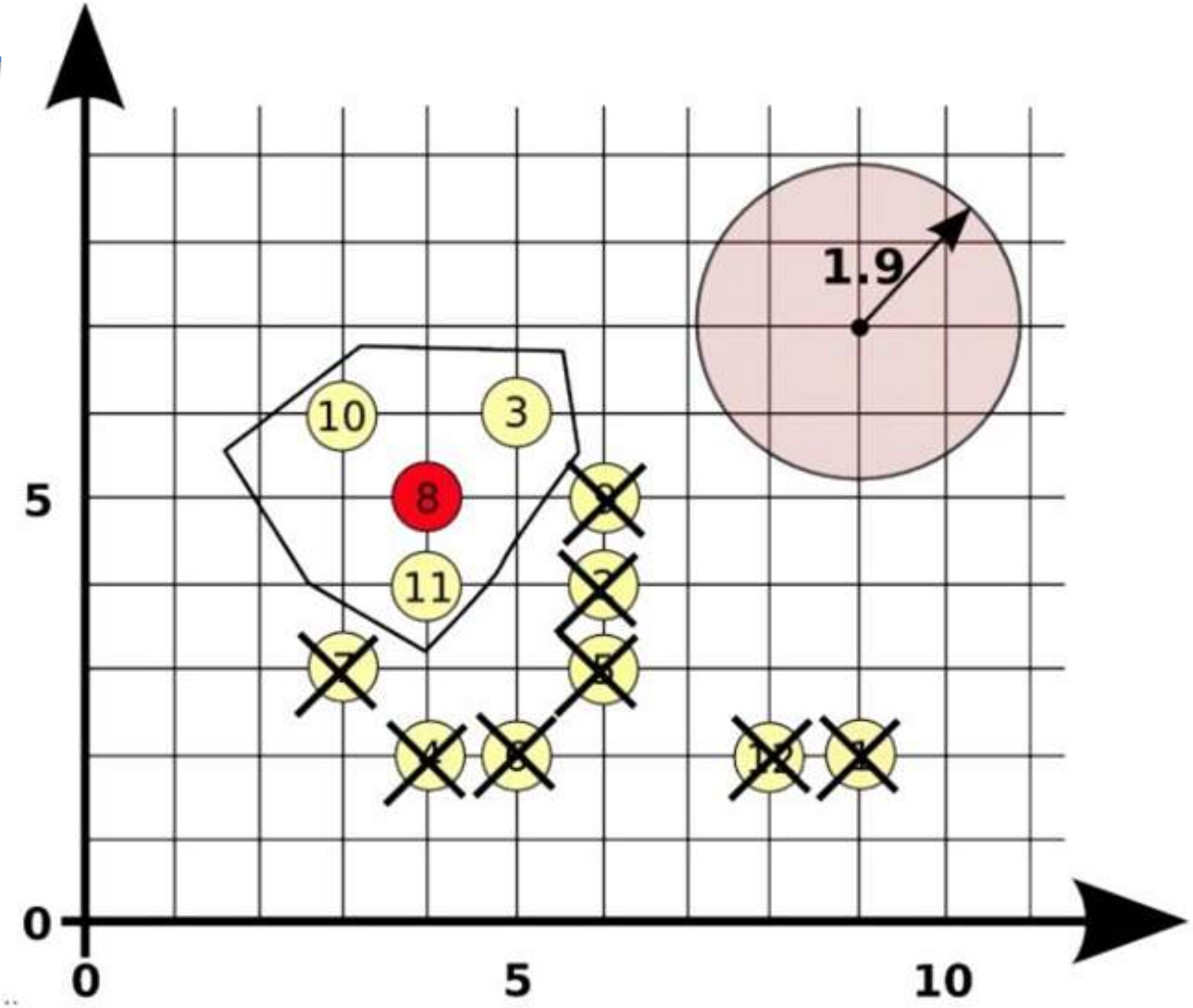


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.




1.9
10
3
8
11
5
0
0
5
10