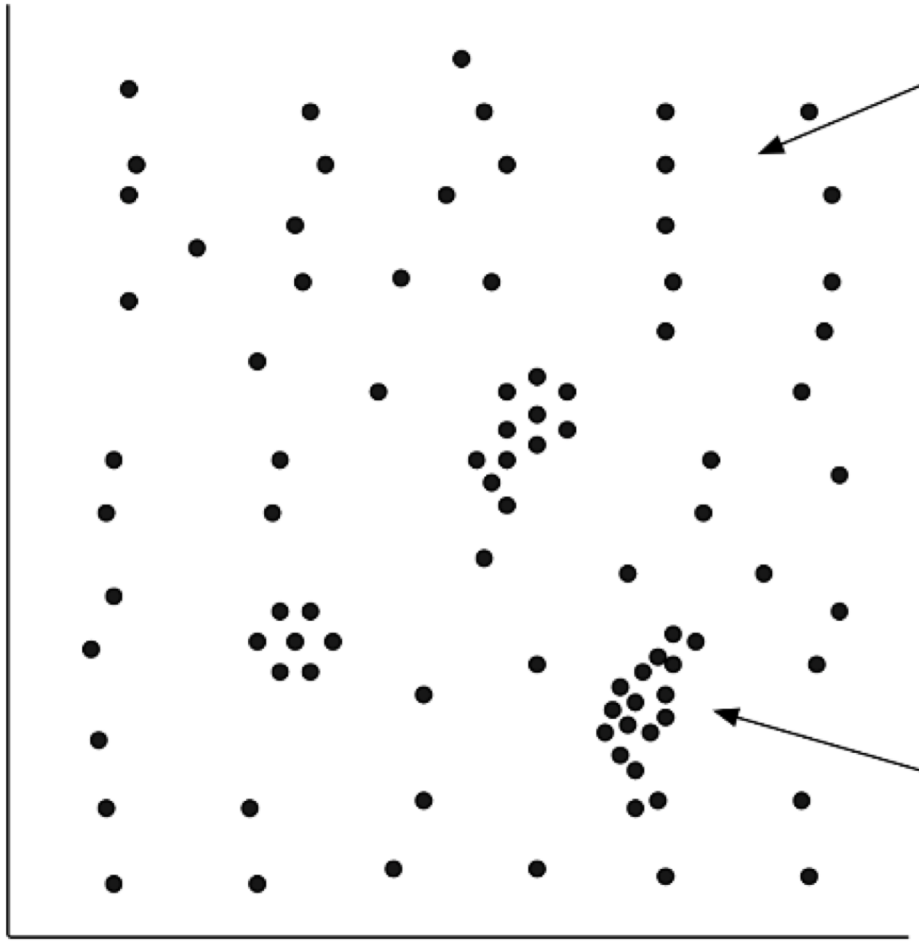
Density Based Clustering

Data Point Density



Area of low density

In density-based clustering, the **density** of observations determines the clusters.

How do we measure density?

Area of high density

DBSCAN Algorithm – Parameters

DBSCAN uses 2 parameters:

- a **distance** parameter to create ε-neighbourhoods, and
- the **minimum number of points** in an ε-neighbourhood required to include the n'hood in the cluster being constructed (including the centre)

3 distinct types of points:

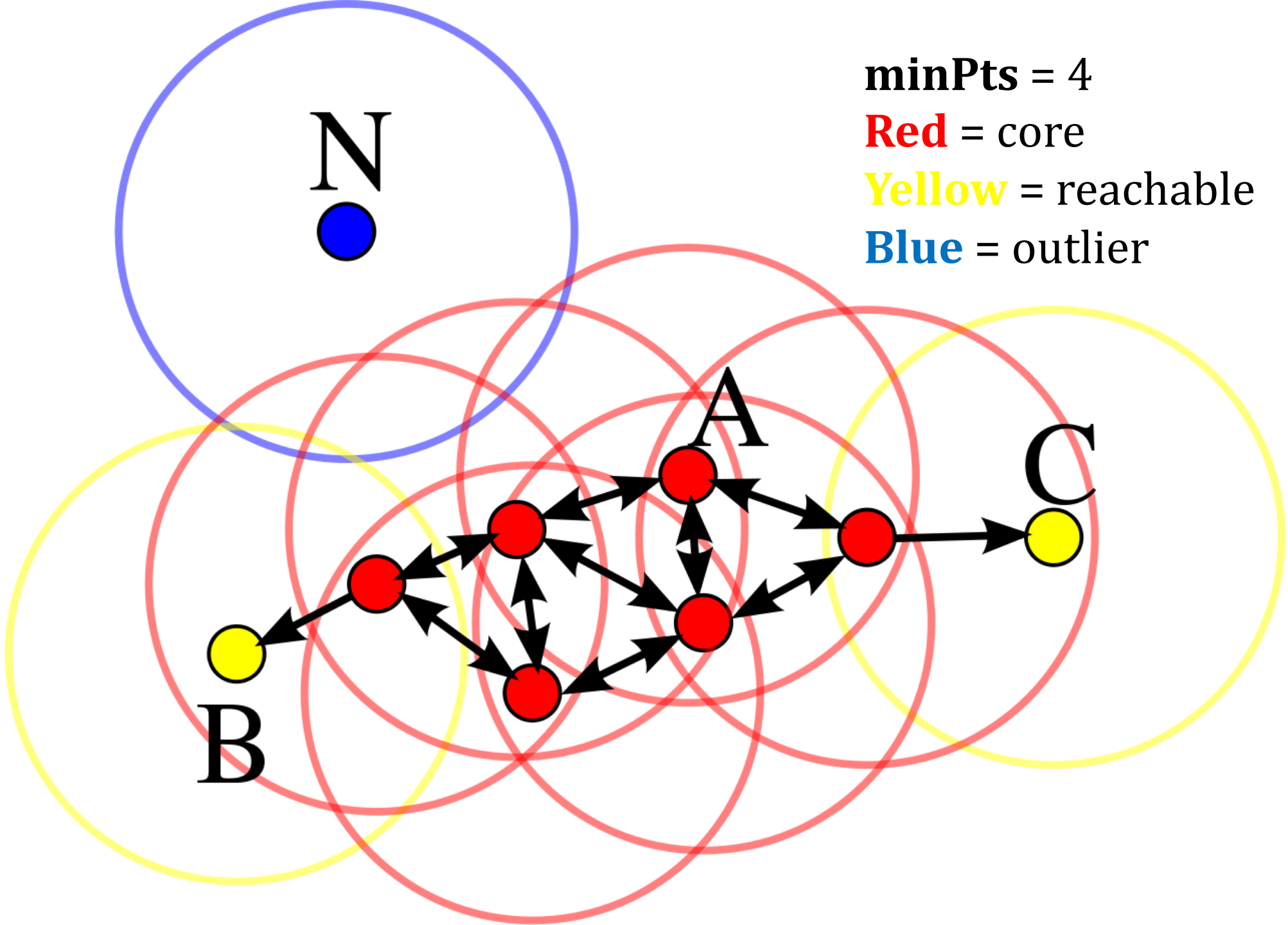
- **outliers**: out of reach of every other point
- **non-core** (**reachable**): within reach of some number of points below the min. threshold
- **core**: within reach of at least the minimum number of other points

DBSCAN Algorithm – Parameters

Reachability is not a symmetric relation: **no point is reachable from a non-core point** (a non-core point may be reachable, but nothing can be reached from it).

Two points *p* and *q* are **density-connected** if there is a point *o* such that both *p* and *q* are reachable from *o* (but density-connectedness *is* symmetric).

All points within a cluster are mutually density-connected. If a point is density-reachable from any point of the cluster, **it is part of the cluster as well**.


minPts = 4
Red = core
Yellow = reachable
Blue = outlier
N
A
B
C

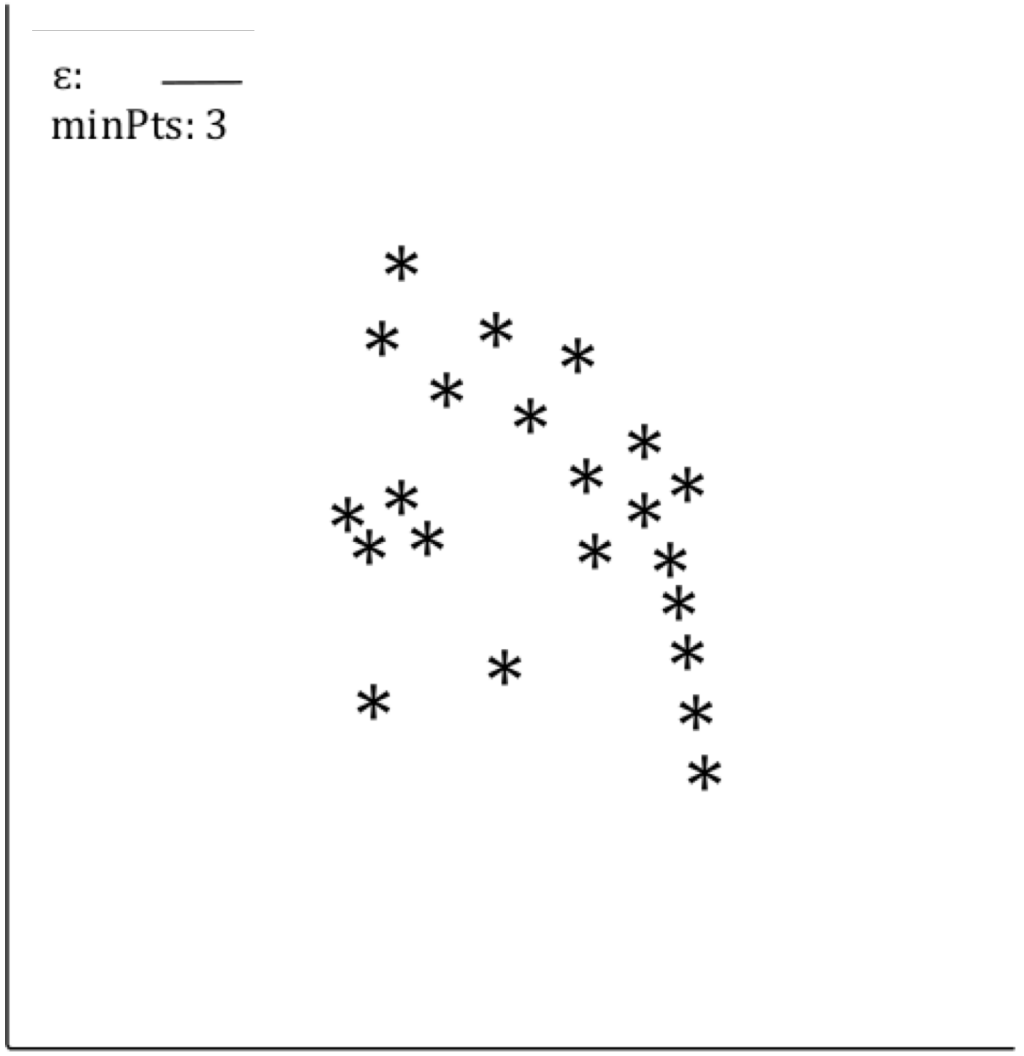
DBSCAN Algorithm

Given $\varepsilon > 0$ and minPts (as well as a distance metric d):

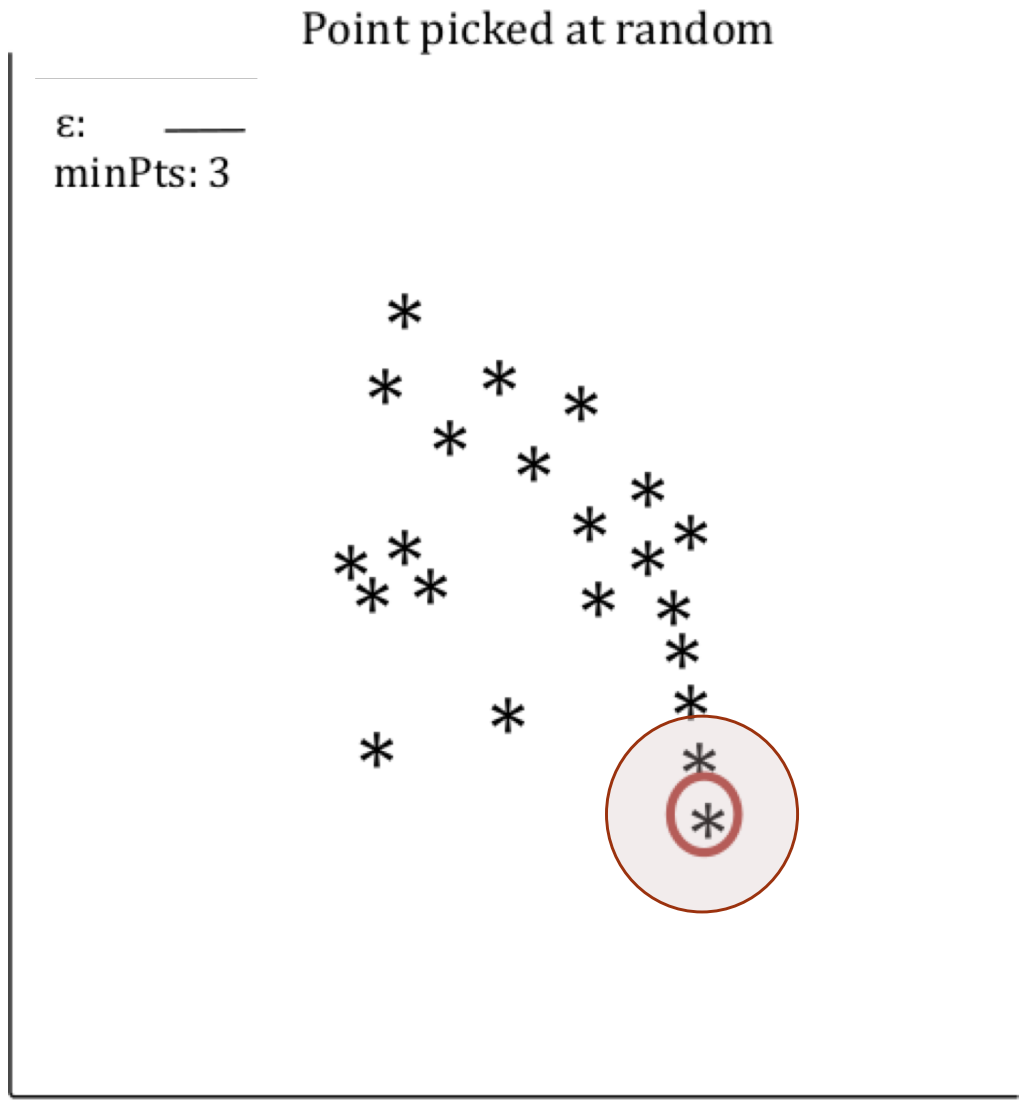
1. Find the ε-neighbours of every point, and identify the core points with more than minPts neighbours (including the core point).
2. Find the connected components of **core** points on the neighbor graph, ignoring all non-core points.
3. Assign each non-core point to a nearby cluster if the cluster is an ε neighbor, otherwise assign it to noise.

That's really all there is to it…

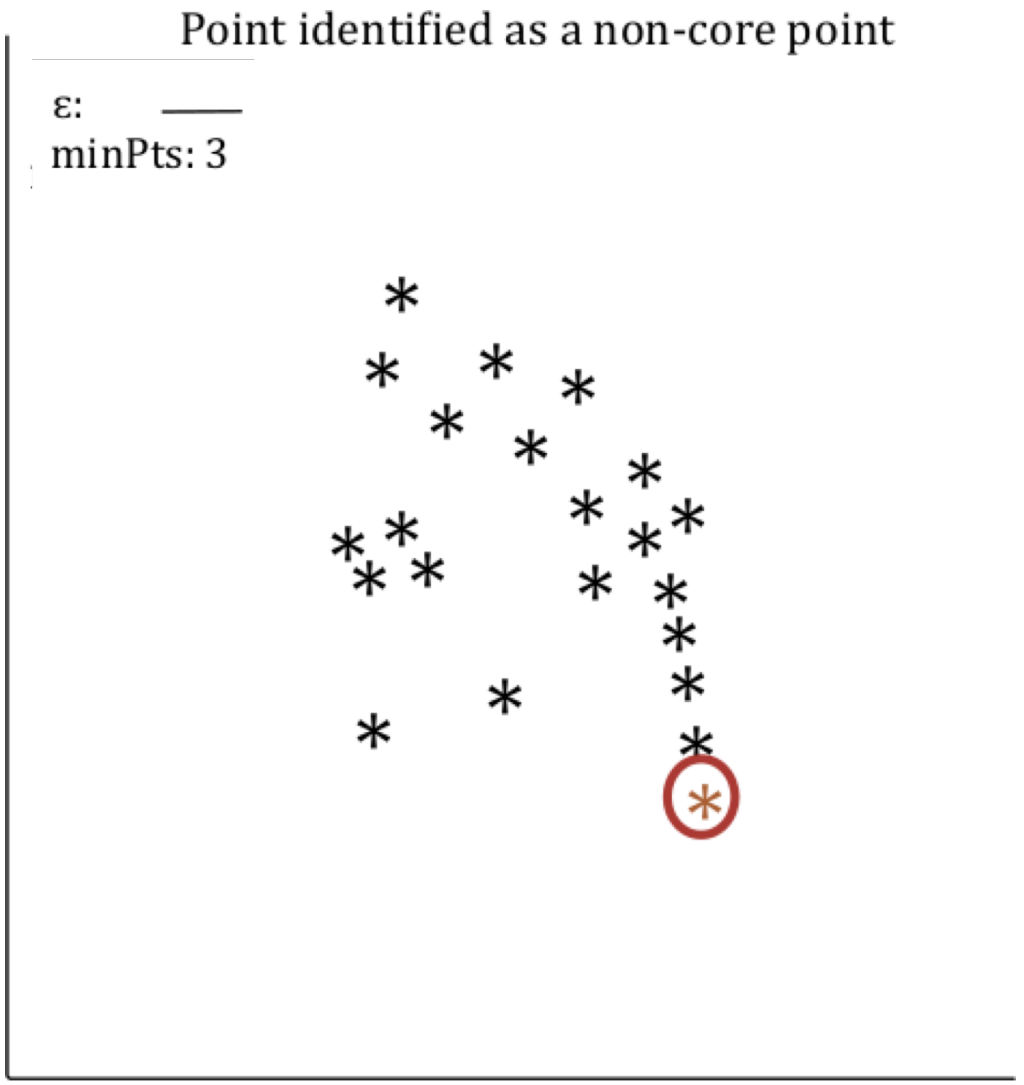
DBSCAN Example – Artificial Dataset



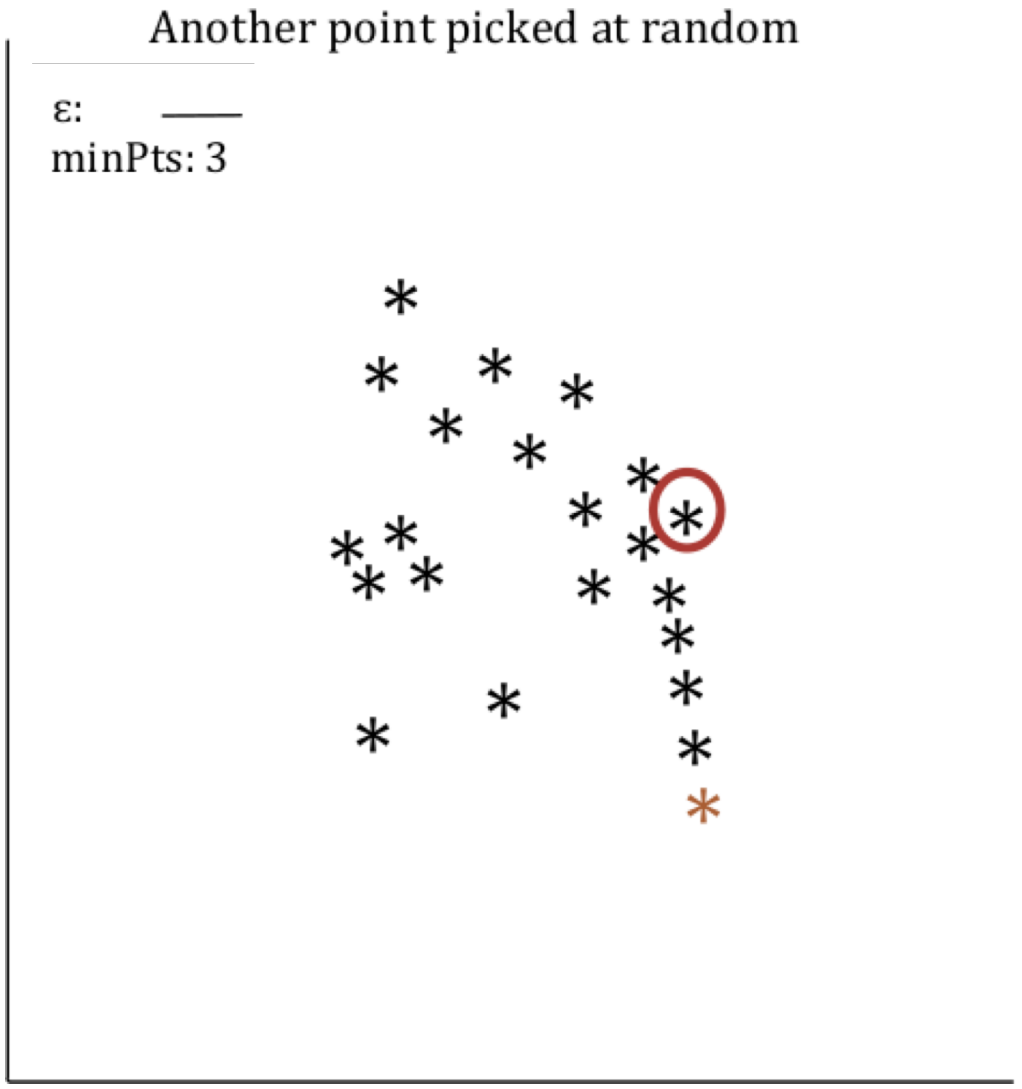
DBSCAN Example – Artificial Dataset



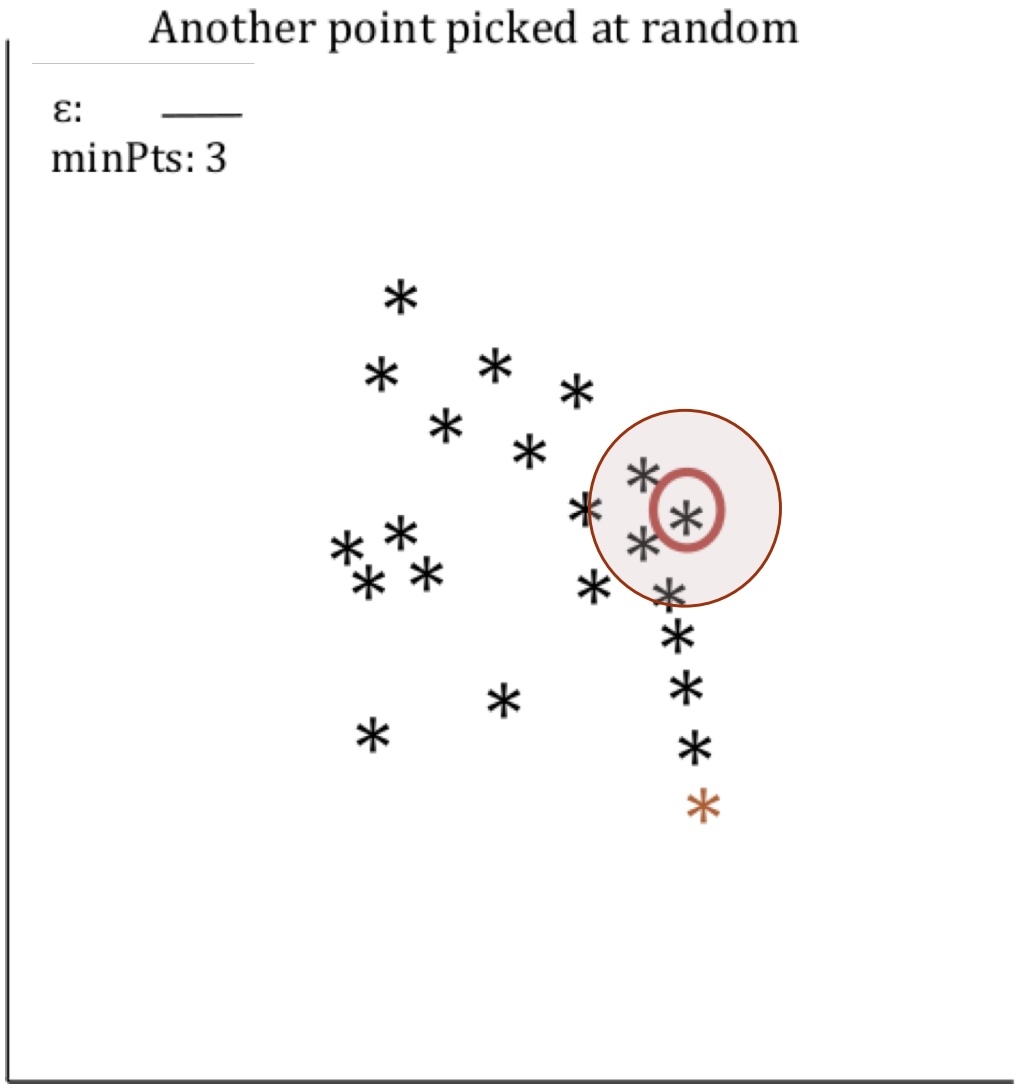
DBSCAN Example – Artificial Dataset



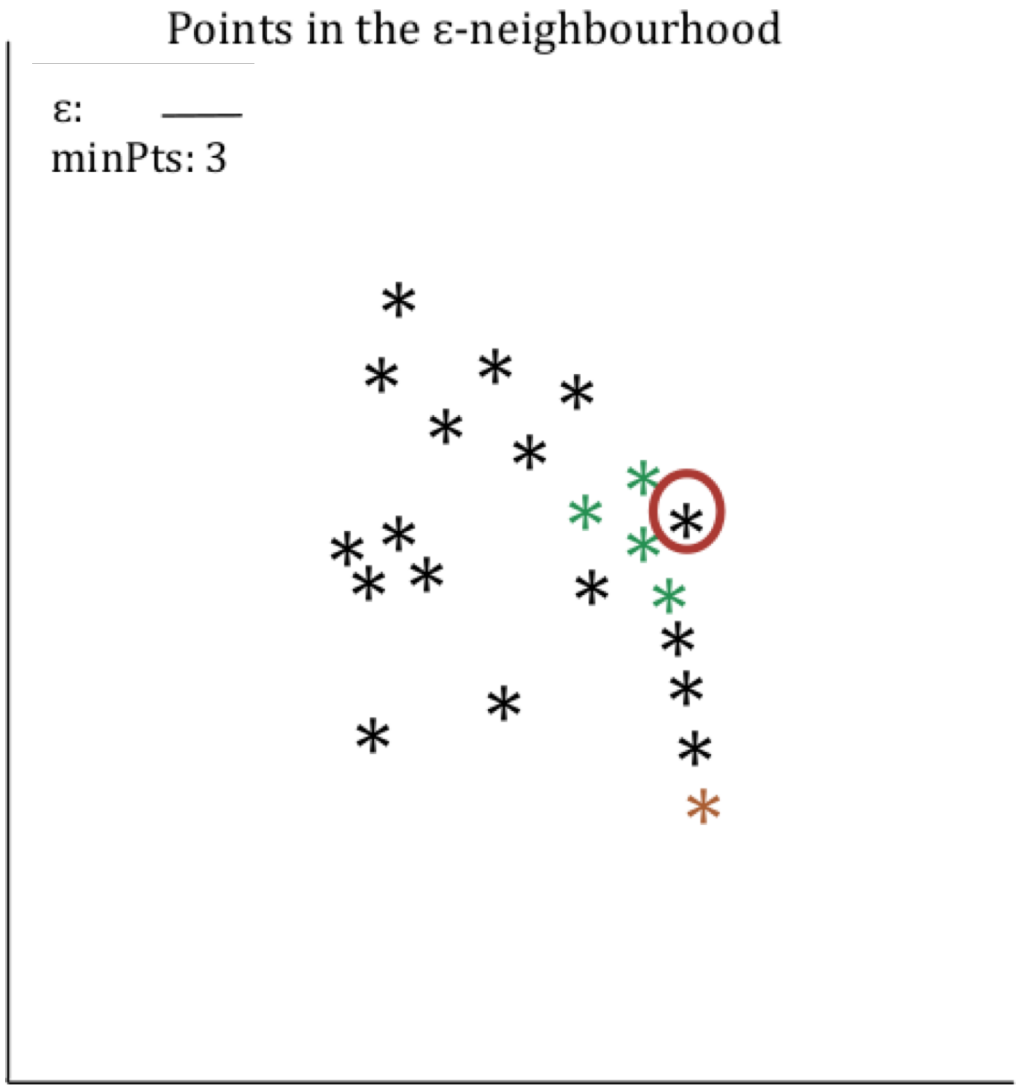
DBSCAN Example – Artificial Dataset



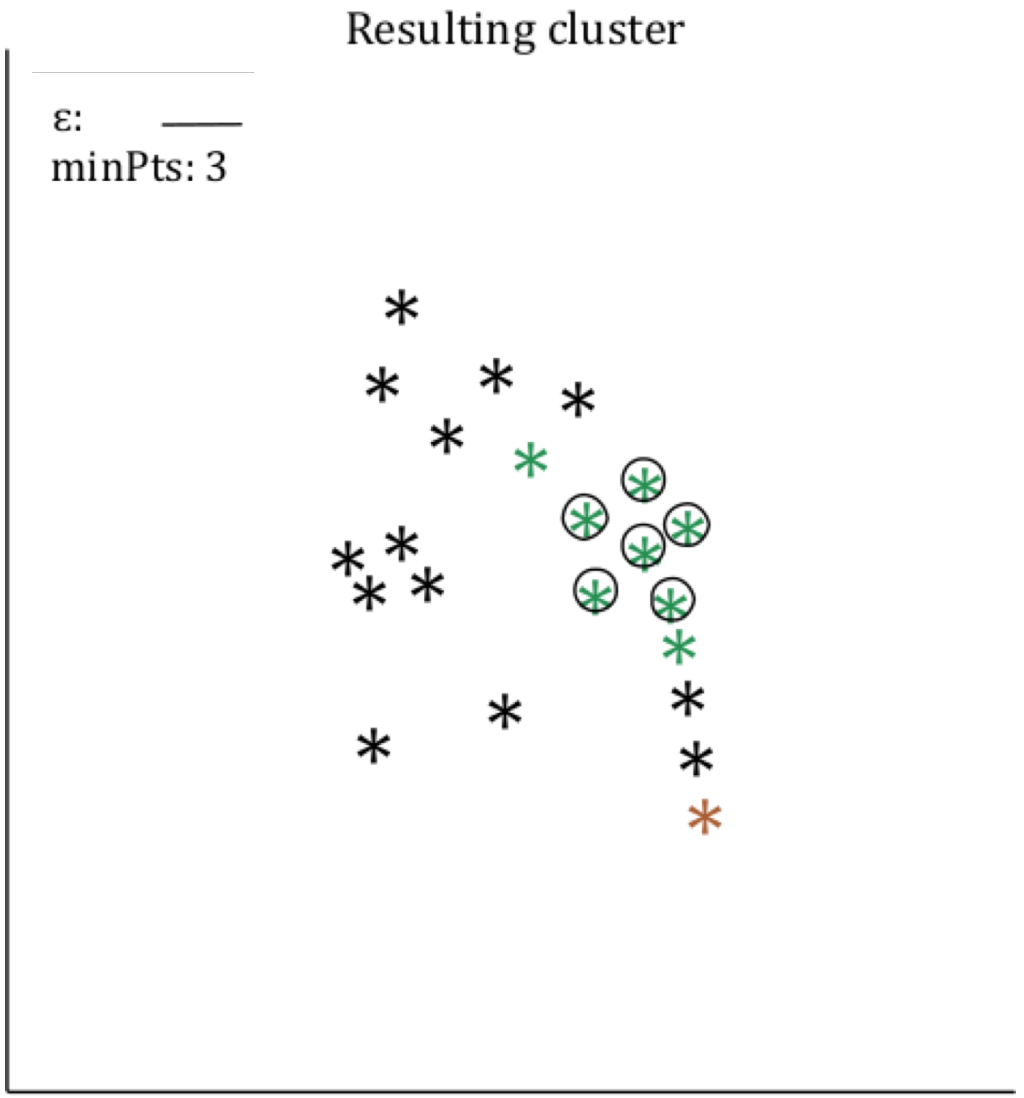
DBSCAN Example – Artificial Dataset



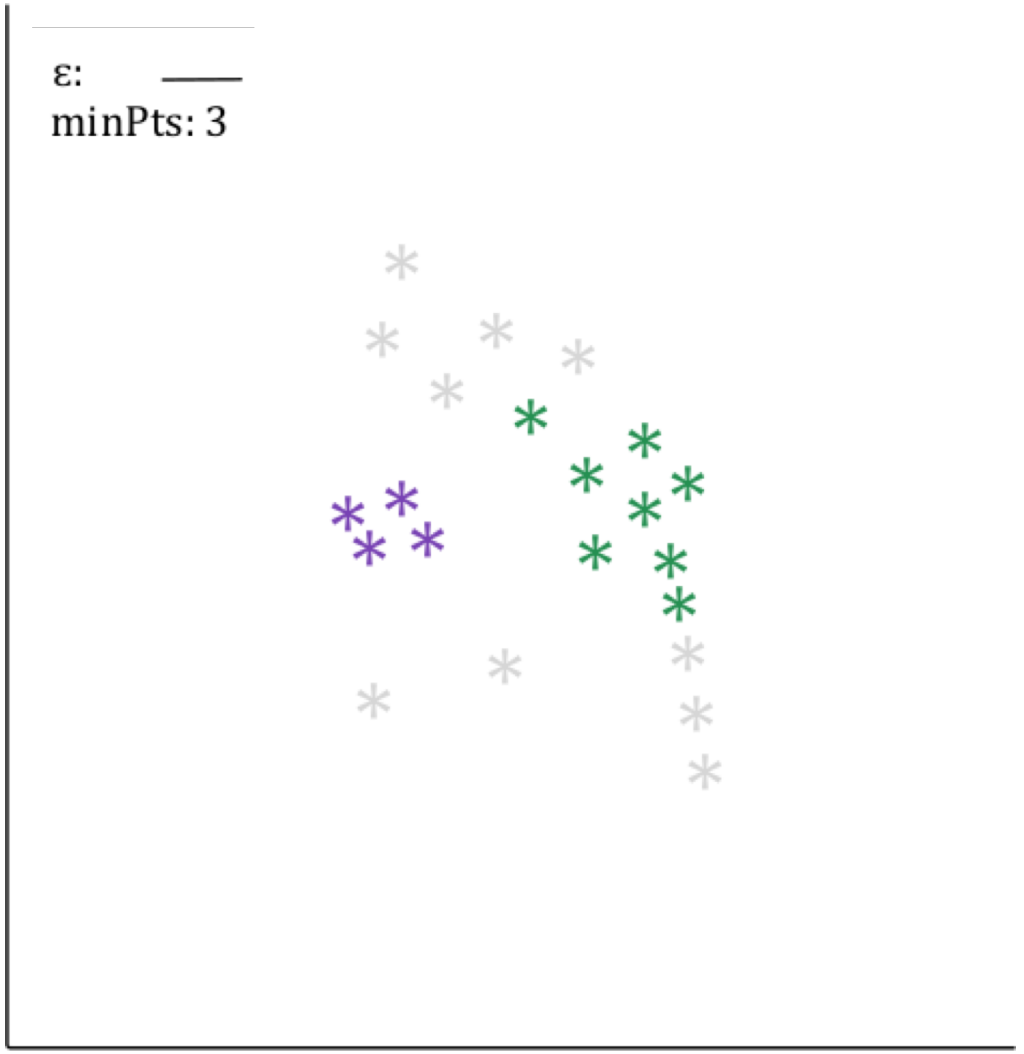
DBSCAN Example – Artificial Dataset

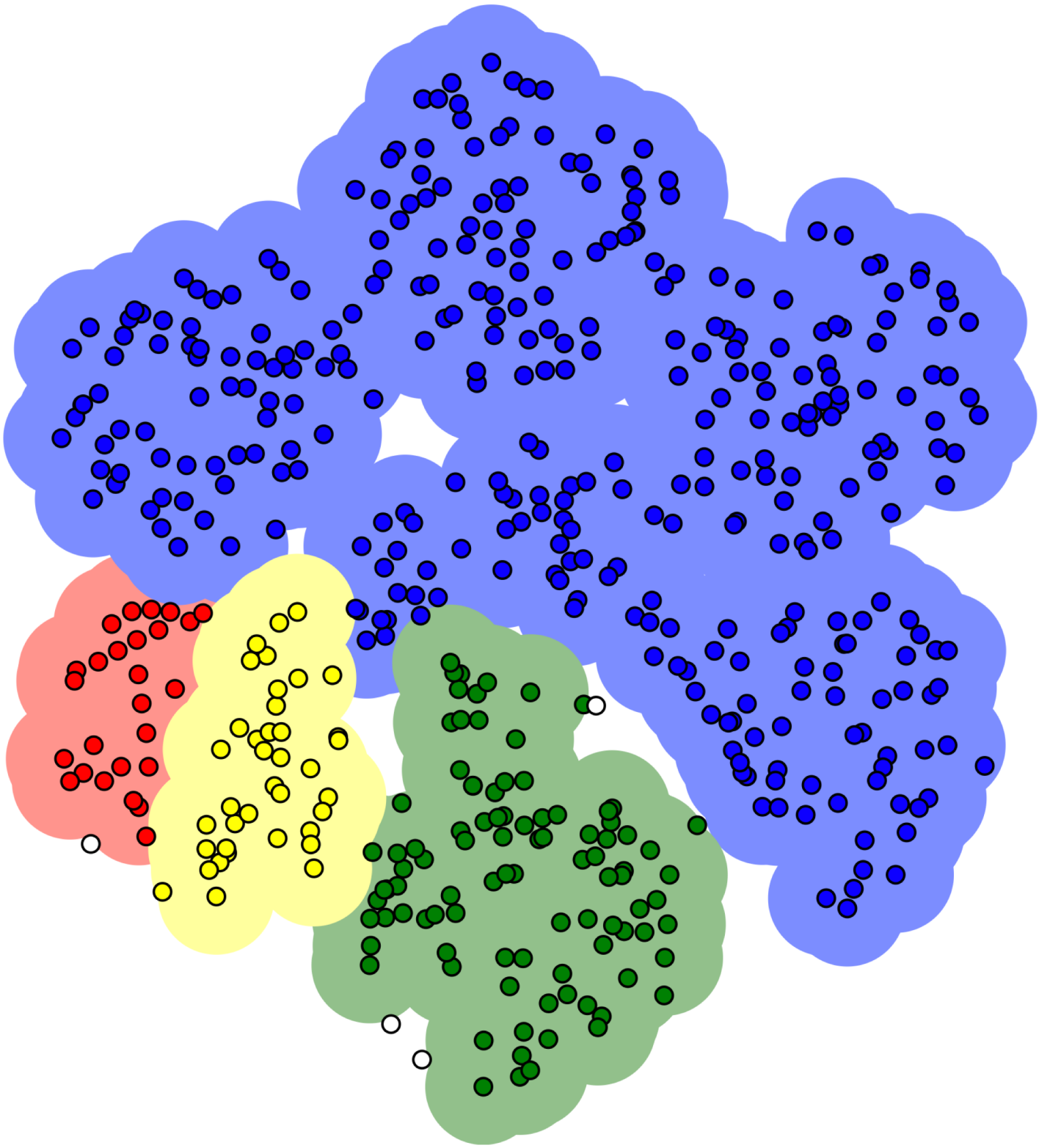


DBSCAN Example – Artificial Dataset

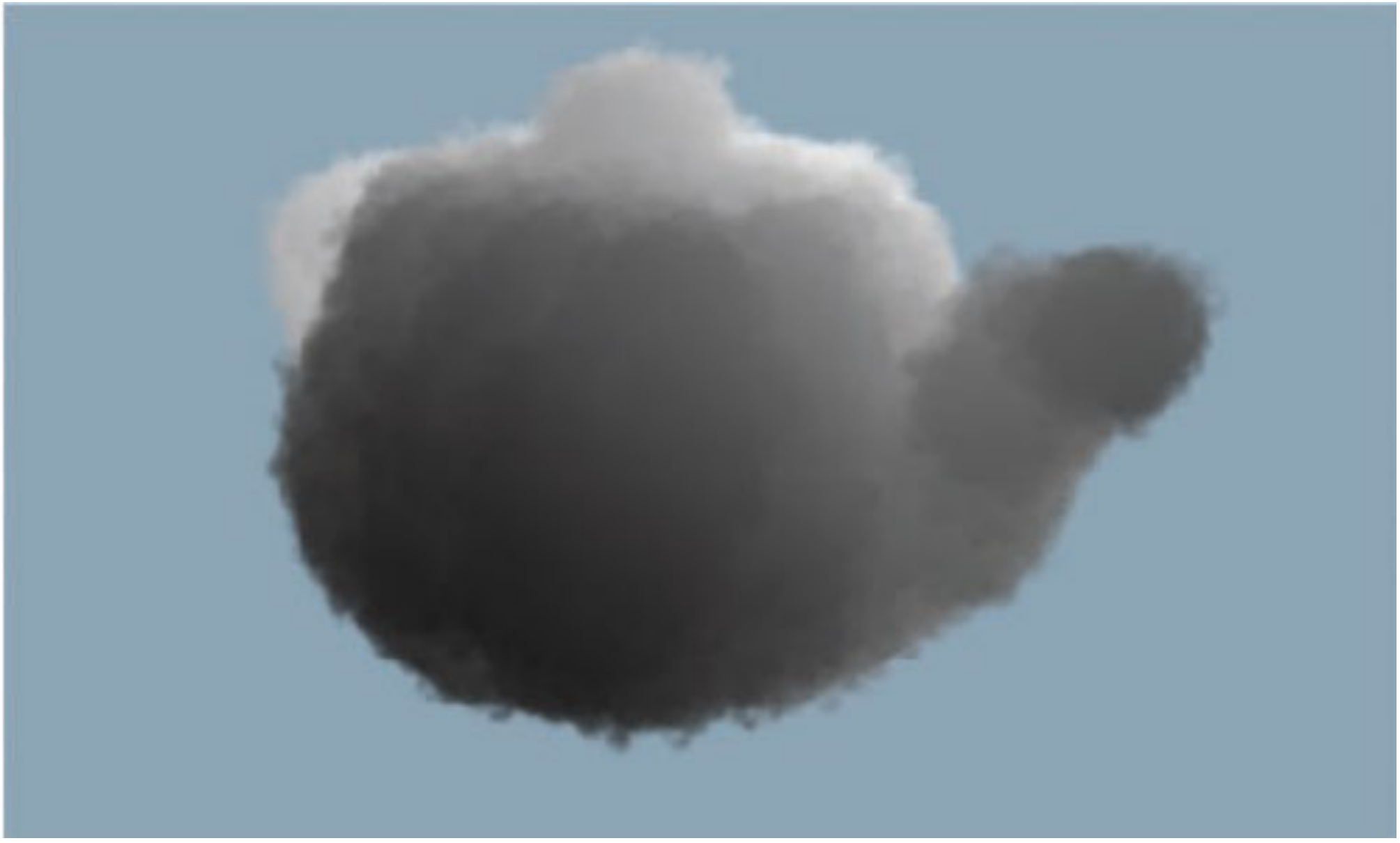


DBSCAN Example – Artificial Dataset





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

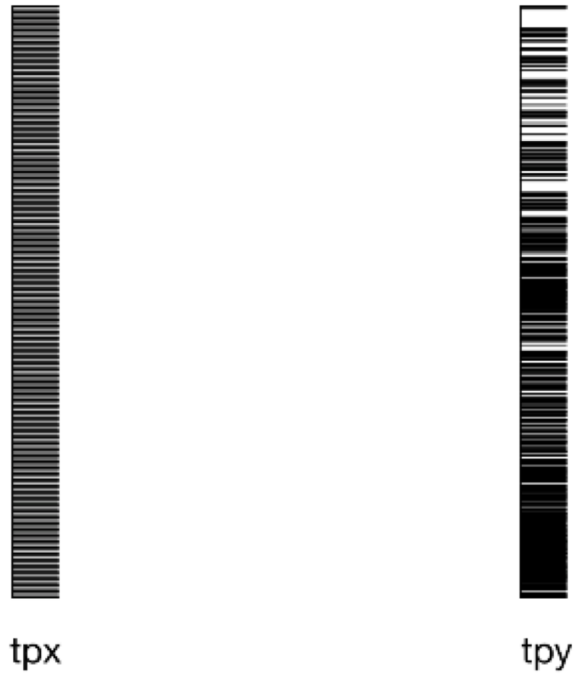


[Adapted from https://library.creativecow.net/articles/ussing_jonas/clouds_3dmax.php]

DBSCAN Clustering Challenge

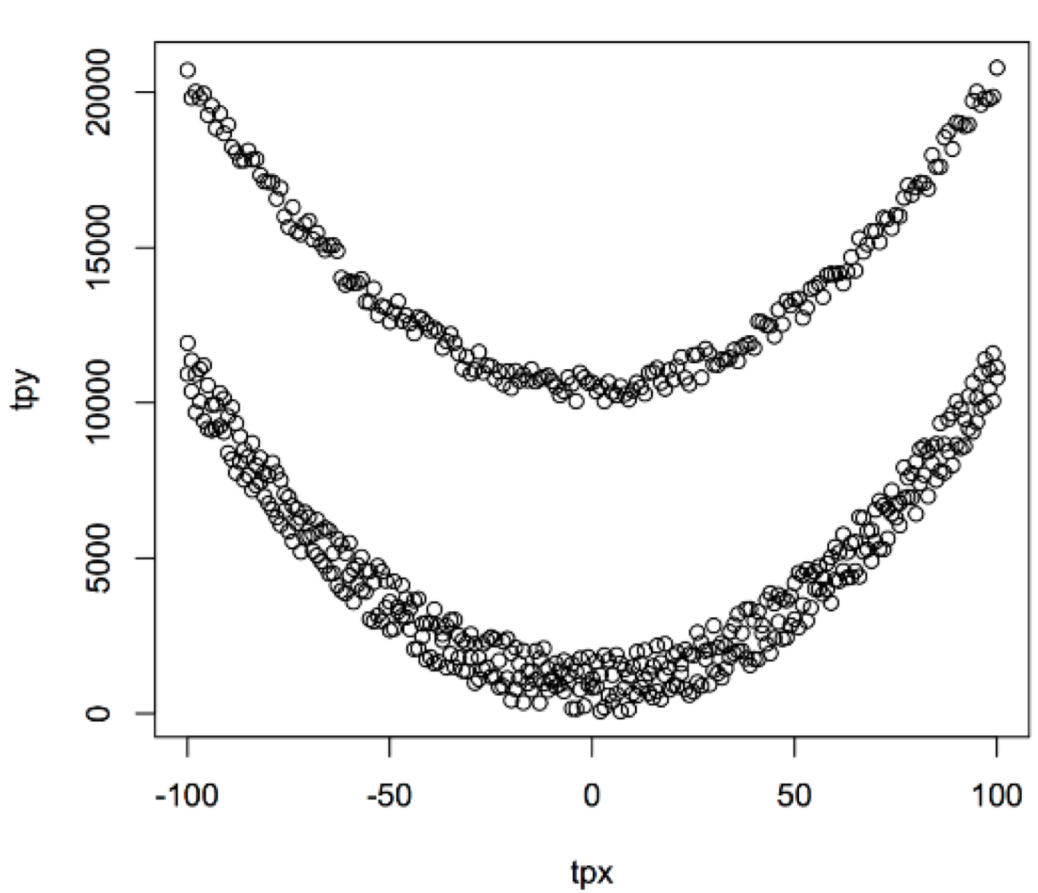
	tpx	tpy
[1,]	-100	10928.249
[2,]	-99	10376.446
[3,]	-98	9696.948
[4,]	-97	10049.223
[5,]	-96	9420.883
[6,]	-95	9171.636
[7,]	-94	9118.230
[8,]	-93	9166.522
[9,]	-92	9251.633
[10,]	-91	9059.243
[11,]	-90	8390.208
[12,]	-89	8186.269
[13,]	-88	7749.231
[14,]	-87	8092.249
[15,]	-86	7518.351
[16,]	-85	7674.044
[17,]	-84	7194.916
[18,]	-83	7340.763
[19,]	-82	7456.145
[20,]	-81	6990.375

First 20 of 603 data points from an artificially-constructed dataset.



1 dimensional plot of points from each column.

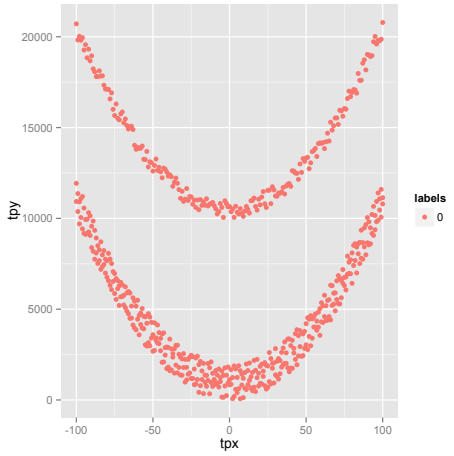
DBSCAN Clustering Challenge



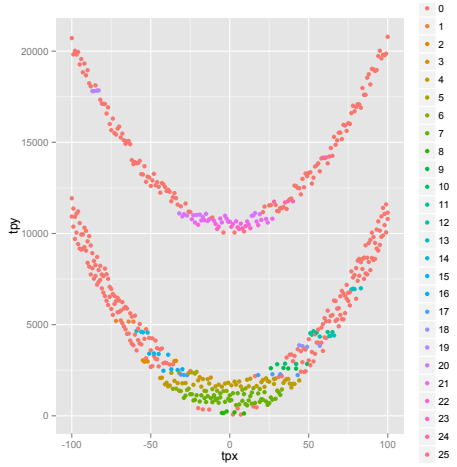
This looks like something DBSCAN should be able to handle…

… and better than *k*-means, too.

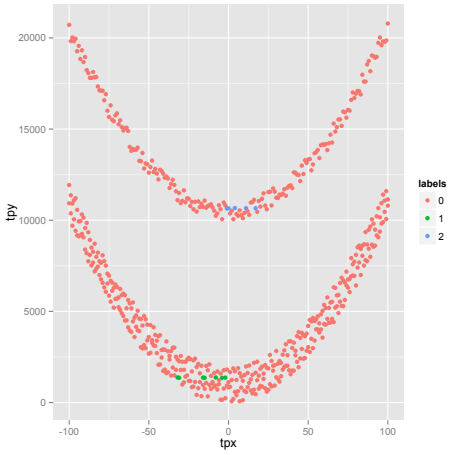
DBSCAN Clustering Challenge



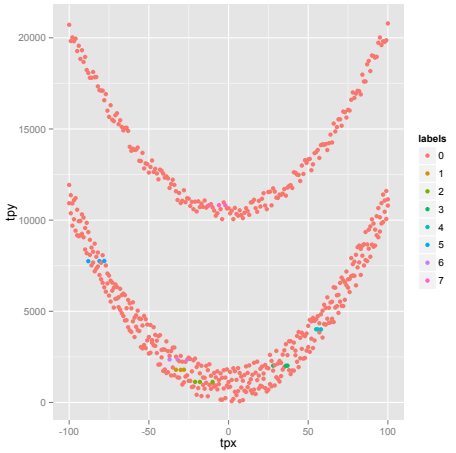
eps = 0.5 , minpts = 10 ,
0 clusters, 603 noise points



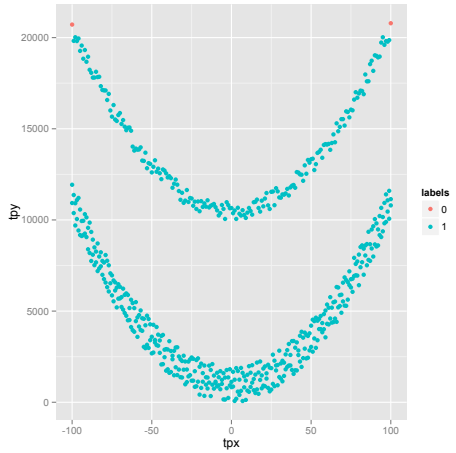
eps = 50, minpts = 4,
25 clusters, 357 noise points



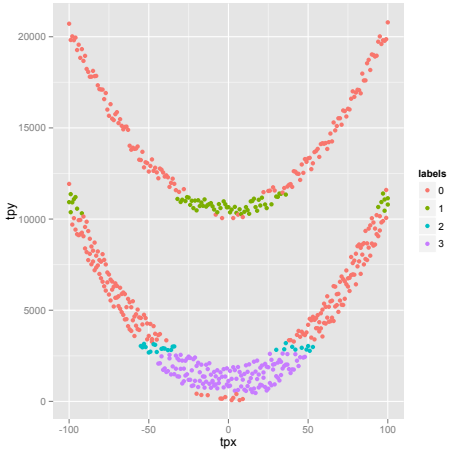
eps = 20, minpts = 5 ,
2 clusters, 591 noise points



eps = 10, minpts = 3,
7 clusters, 582 noise points



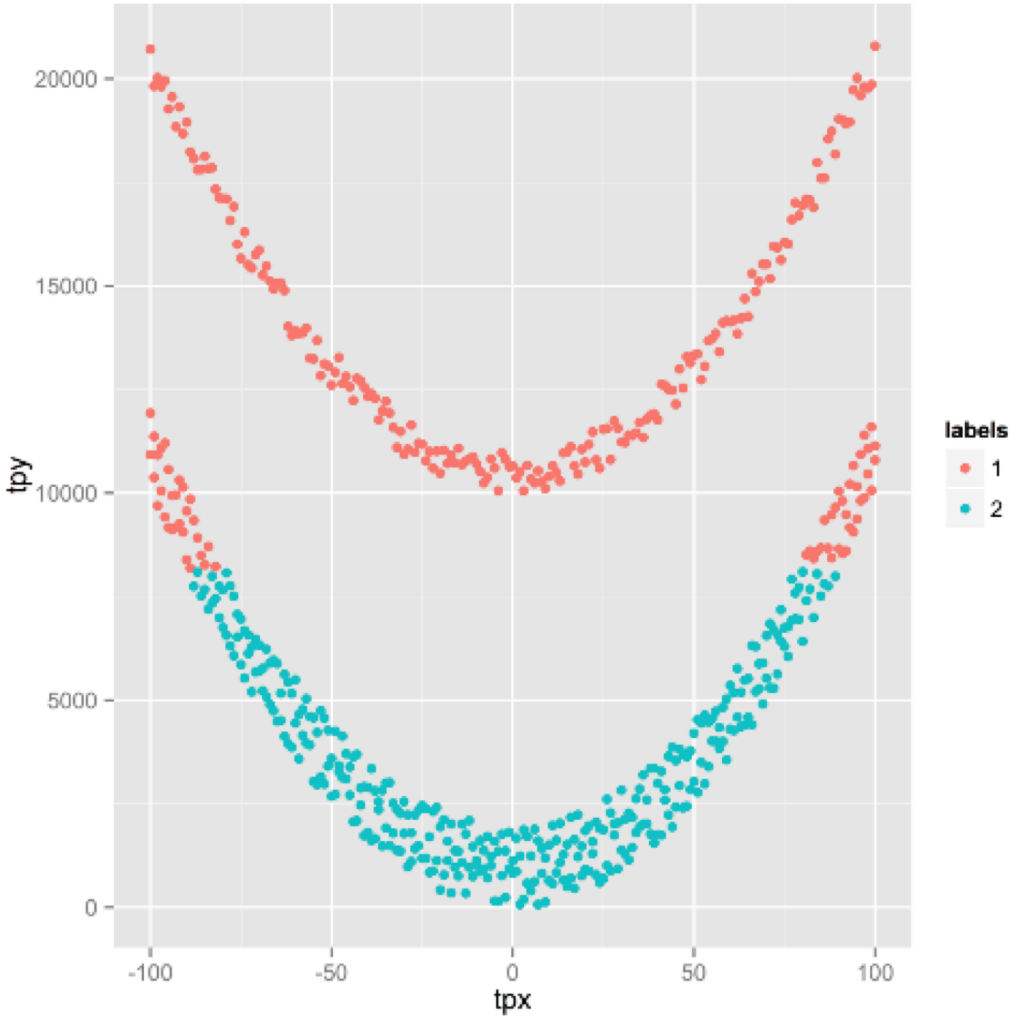
eps = 500, minpts = 5 ,
1 cluster, 2 noise points



eps = 200, minpts = 20 ,
3 clusters, 357 noise points

But it turns out that DBSCAN isn't working out so well for this dataset…

DBSCAN Clustering Challenge

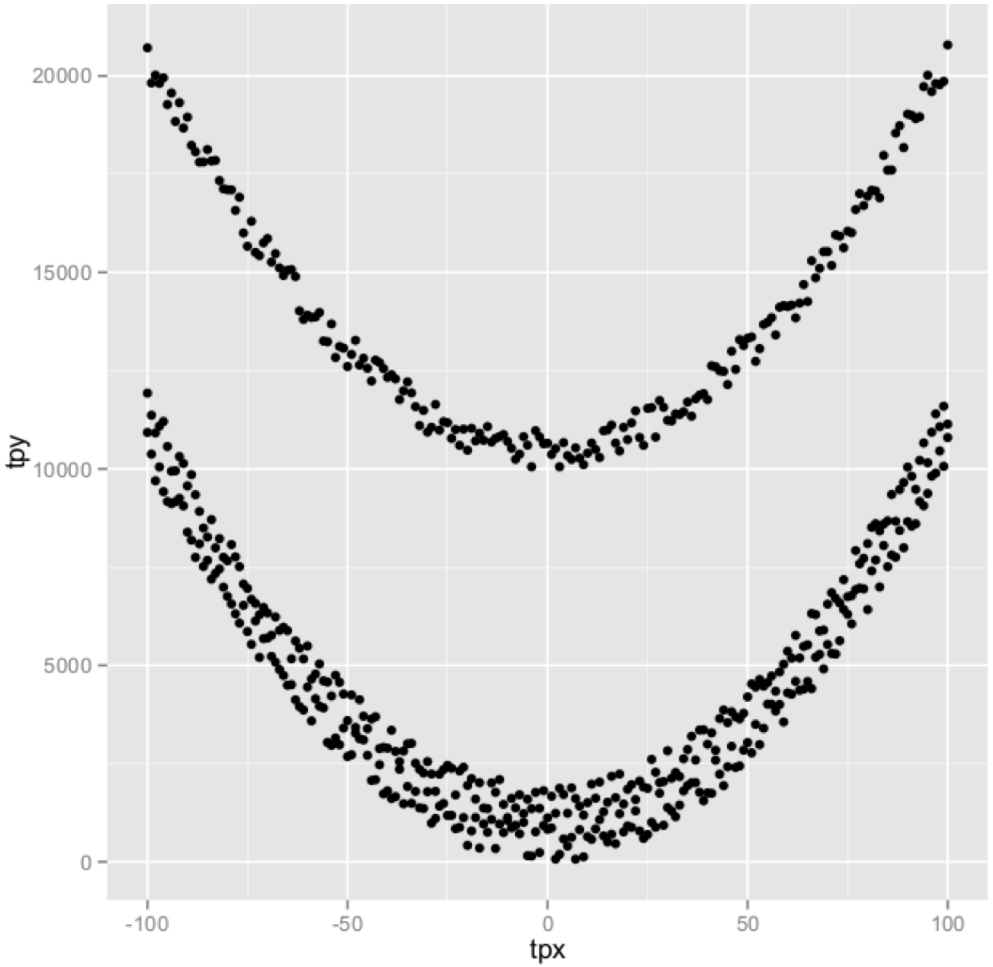


k-means appears to be doing a better job.

But is it really detecting the clusters more accurately, or just taking advantage of the separation between the two clusters?

Is there a way to get DBSCAN to work?

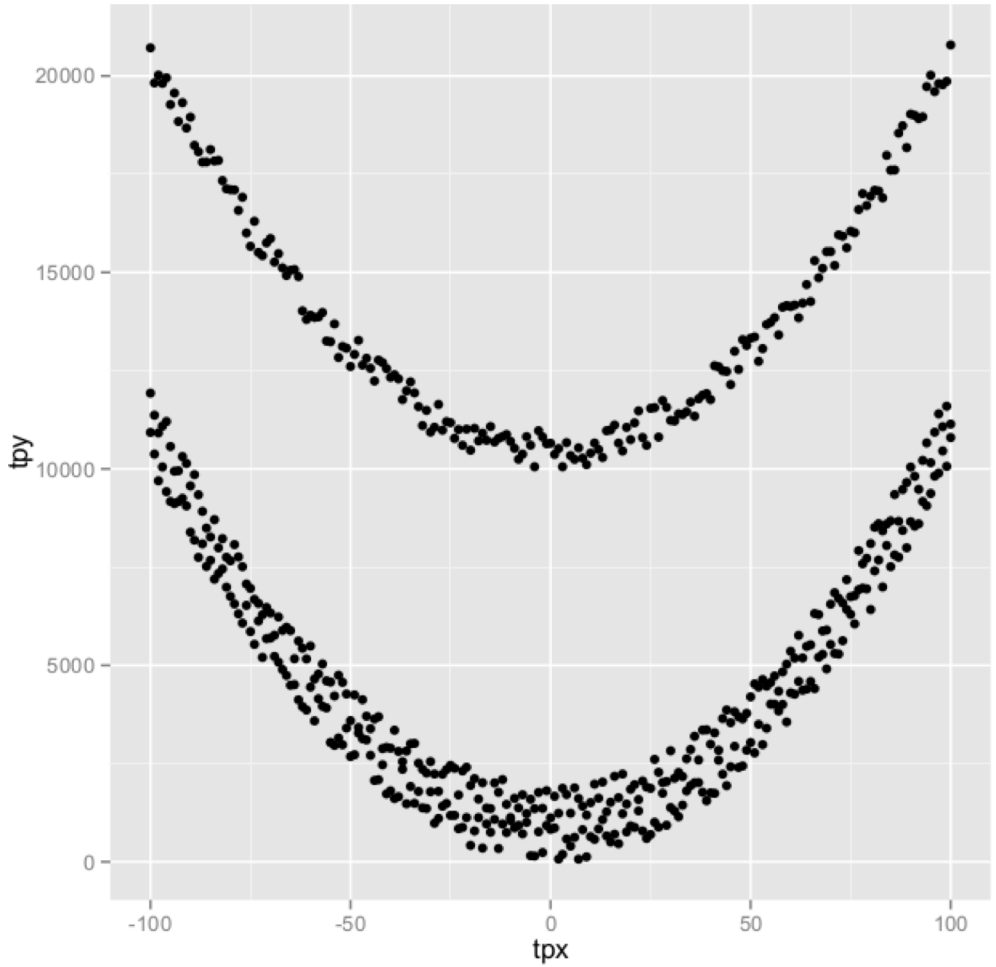
DBSCAN Clustering Challenge

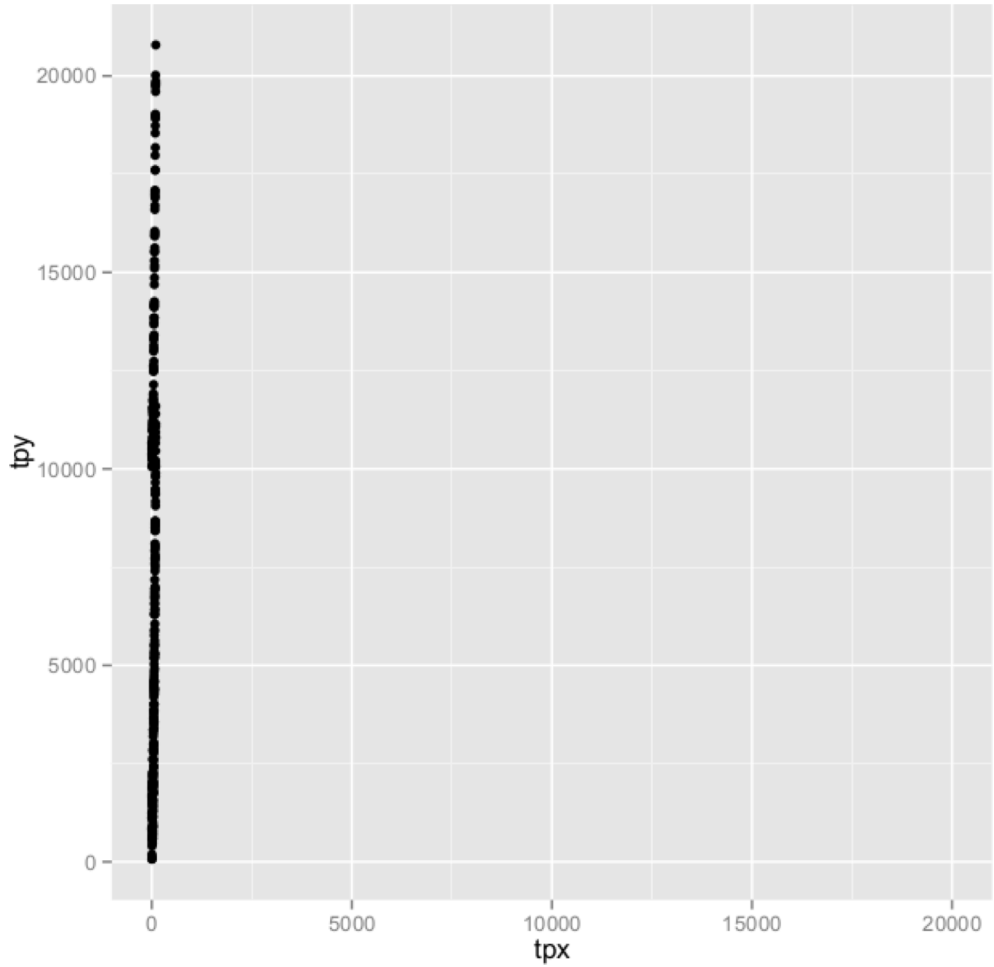


Take a closer look at the axes on this plot…

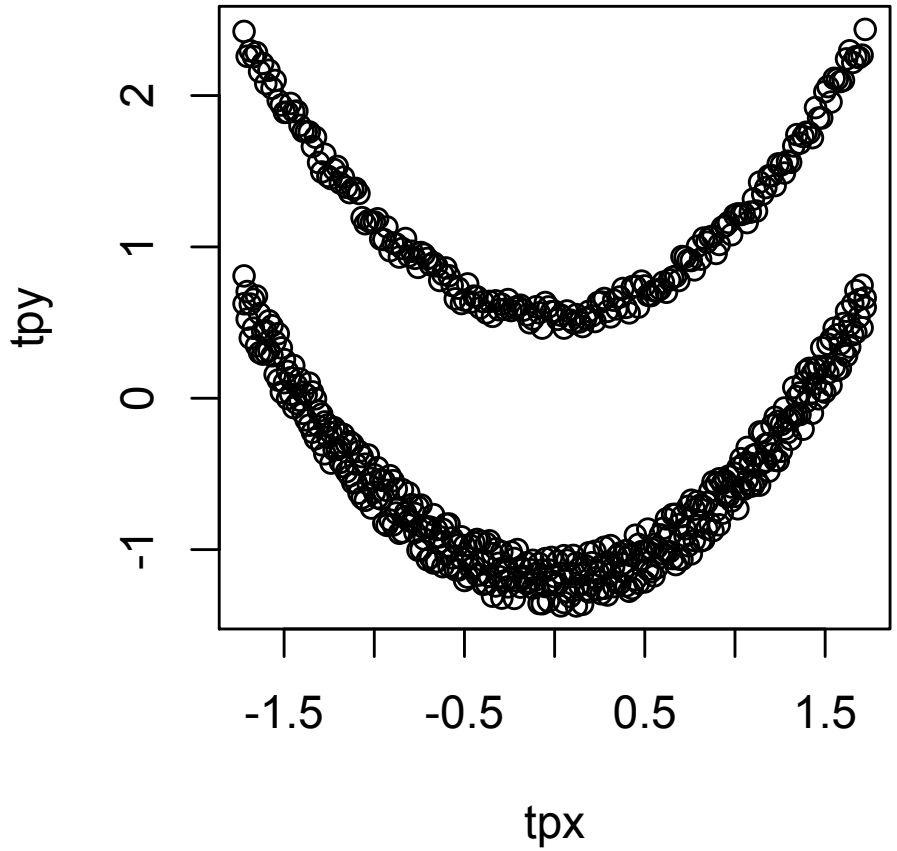
DBSCAN Clustering Challenge

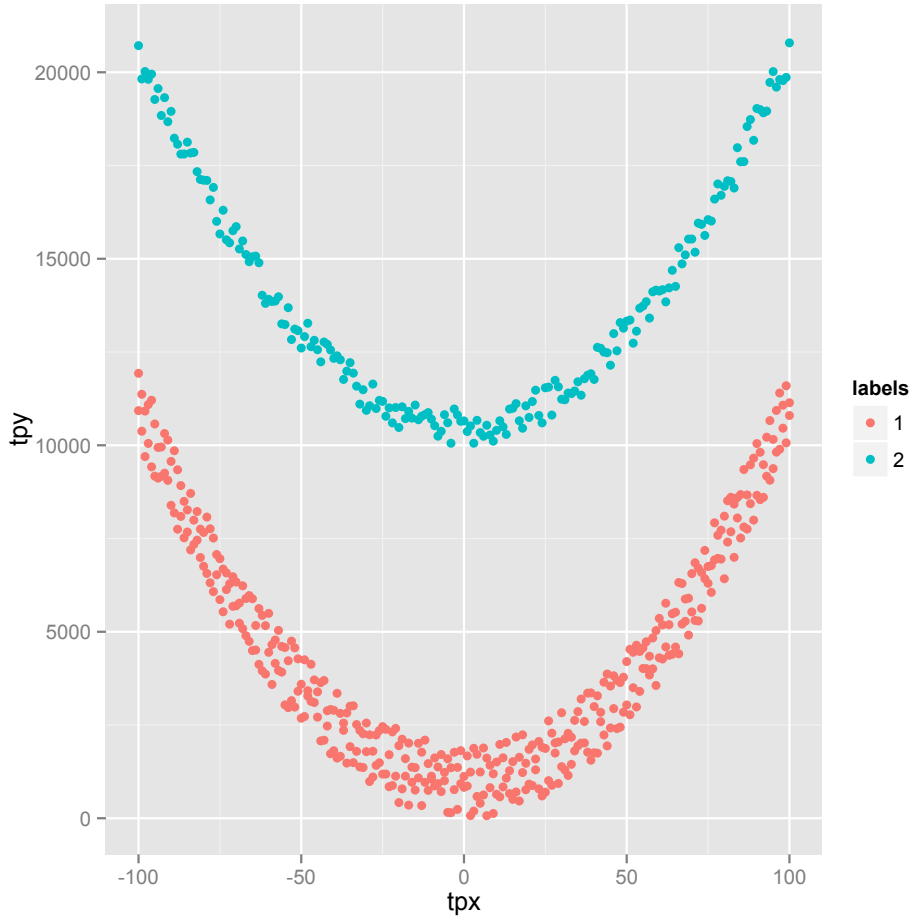
Re-plotted, with the axis adjusted to match the axis:





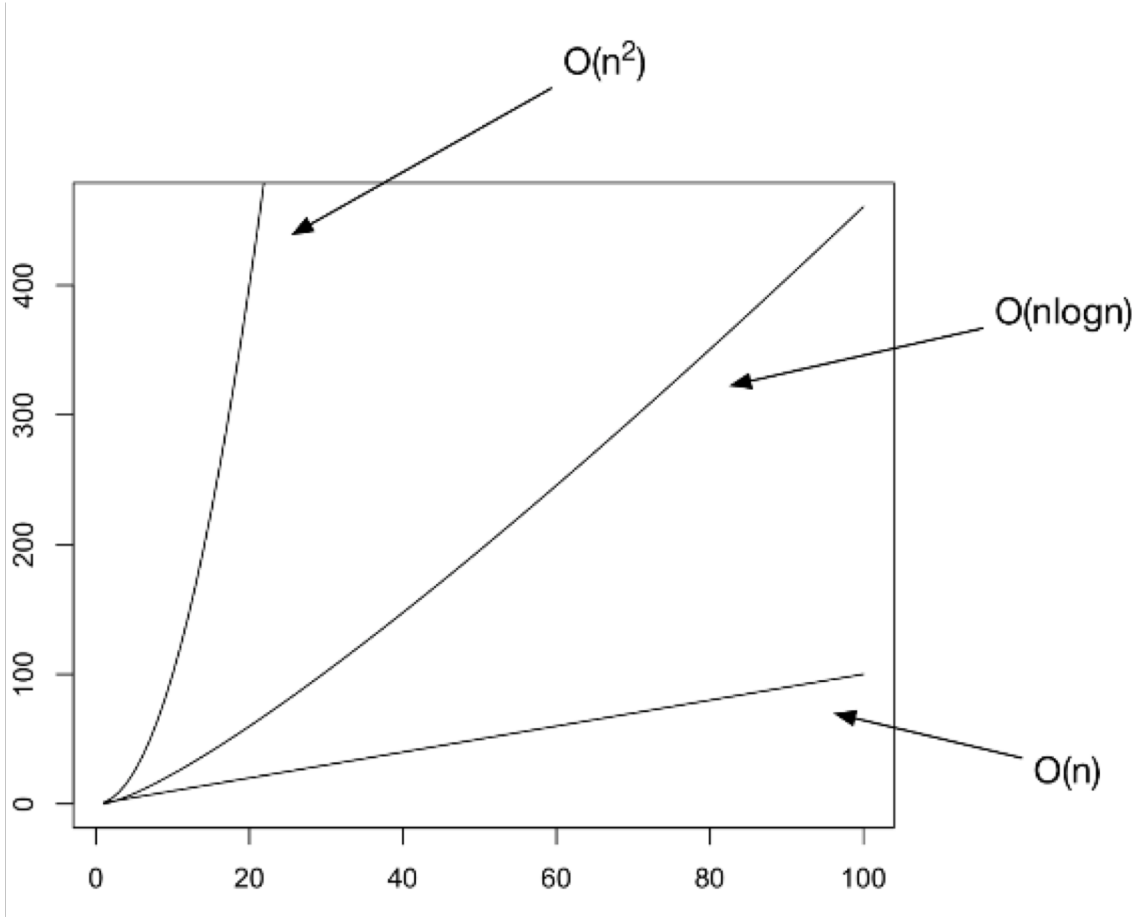
DBSCAN Clustering Challenge





When data is scaled, position and length of vectors are adjusted to normalize the distribution of the data. This pleases DBSCAN!

Comparing Algorithmic Complexity



DBSCAN can handle globular clusters and non-globular clusters — why isn't it being used all the time?

DBSCAN is $O(n \log n)$ in the best case scenario, whereas k-means is $O(nk)$ (more or less)

When the number of observations increases, DBSCAN is less efficient than k-means .

DBSCAN Advantages

No need to specify the number of clusters.

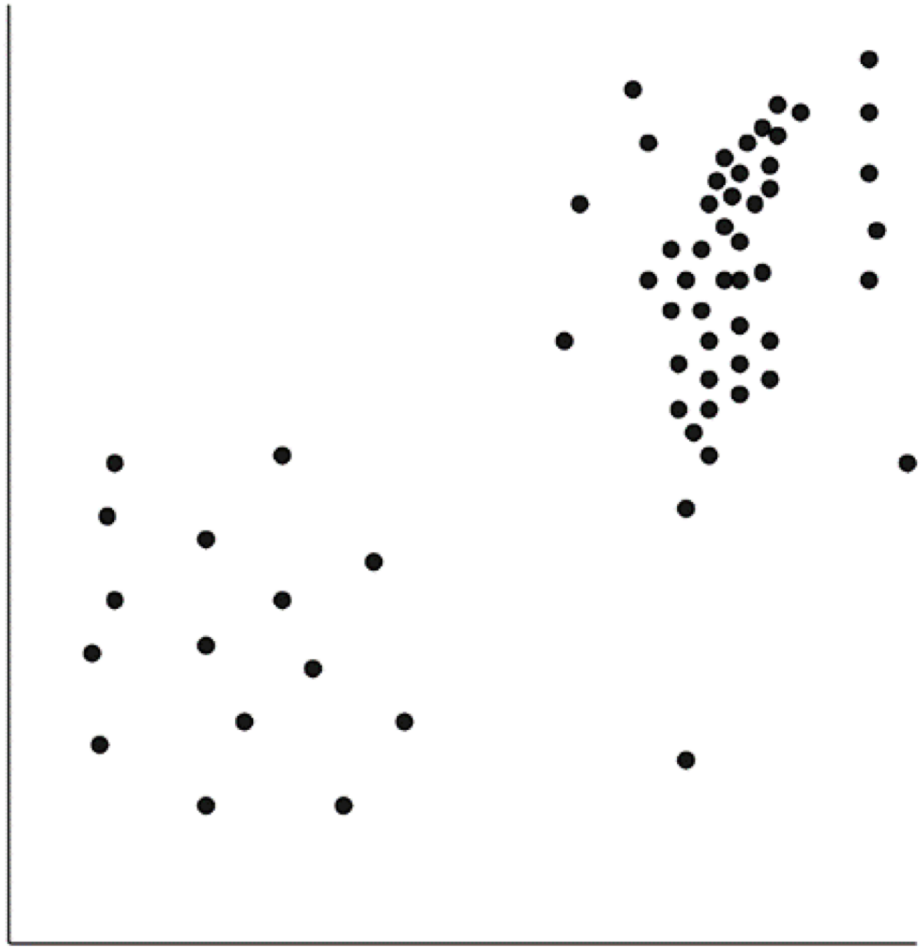
Can find arbitrarily shaped clusters.

Can recognize “noisy” points.

Robust to outliers.

Requires only two parameters (minPts and ε) which can be set by domain experts if the data is well understood.

DBSCAN Limitations



DBSCAN's clustering **kryptonite**: datasets where cluster density is not consistent across clusters.

Hard to set parameters that consistently capture clusters while identifying outliers.

Not entirely deterministic: border points that are reachable from more than one cluster can be part of either cluster, depending on the order.

Parameters Estimation

minPts:

- minPts $\geq$ # features + 1
- larger values are better for noisy data sets
- minPts $\geq 2 \times dim$ for large datasets or sets with duplicates

ε:

- if too small, a large prop. of observations is not clustered
- if too high, majority of observations are in the same cluster
- in general, small values are preferable

Distance function:

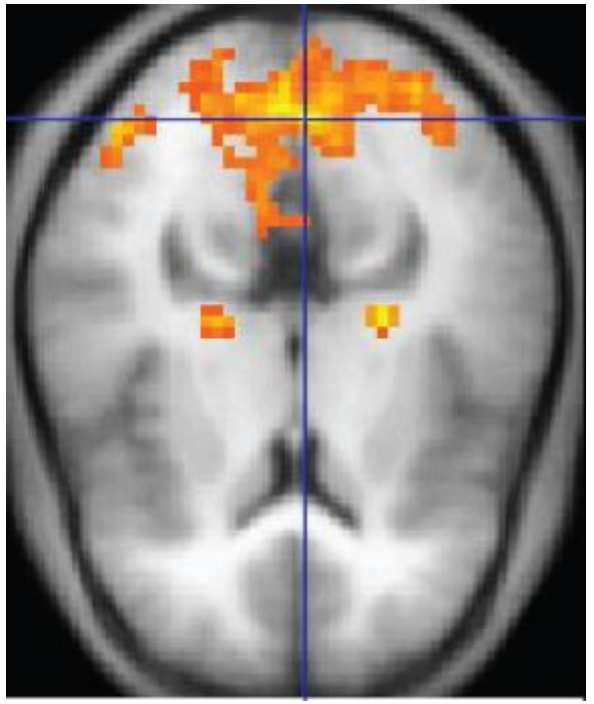
- has a major impact on the results
- should be selected before ε is chosen

DBSCAN Examples
Detecting Alzheimer's Disease

Mild cognitive impairments (MCI) are a known to be a risk for factor for development of Alzheimer's Disease

MCI are accompanied by changes in brain structure

But which changes indicate that people will go on to develop Alzheimer's?



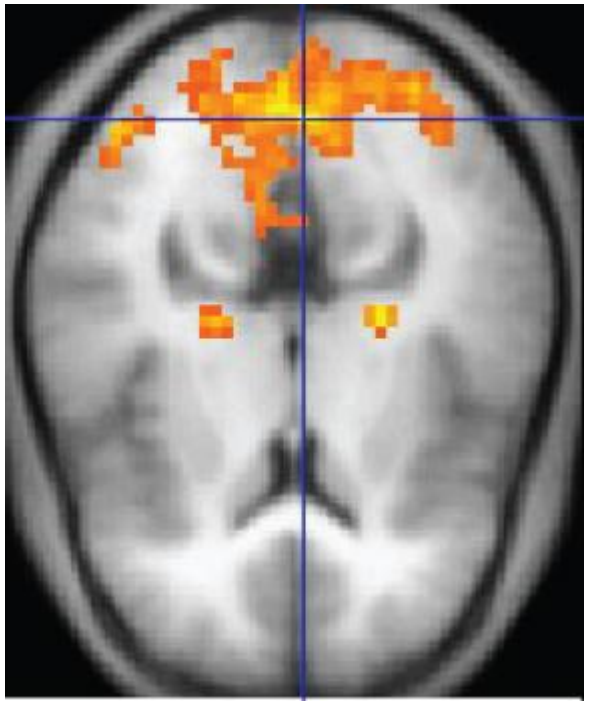
FMRI highlighting some areas of the pre-frontal cortex.

Automated detection of brain atrophy patterns based on MRI for the prediction of Alzheimer's disease, by C. Plant, S.J. Teipel, A. Oswald, C. Böhm, T. Meindl, J. Mourao-Miranda, A.W. Bokde, H. Hampel, M. Ewers.

DBSCAN Examples
Detecting Alzheimer's Disease

A number of different data science techniques applied to MRI data:

- Support Vector Machines
- Bayesian Statistics
- Voting Feature Intervals
- Feature Extraction
- DBSCAN



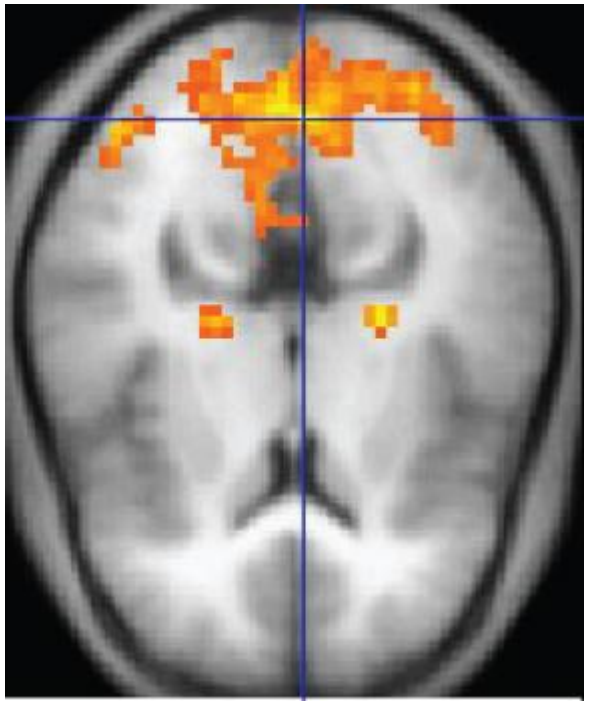
FMRI highlighting some areas of the pre-frontal cortex.

Automated detection of brain atrophy patterns based on MRI for the prediction of Alzheimer's disease, by C. Plant, S.J. Teipel, A. Oswald, C. Böhm, T. Meindl, J. Mourao-Miranda, A.W. Bokde, H. Hampel, M. Ewers.

DBSCAN Examples
Detecting Alzheimer's Disease

DBSCAN is used once voxels that provide high information about the classification of the image are identified using entropy based measures

DBSCAN then groups pixels with similar spatial and information levels to determine which parts of the brain are the most important for the diagnosis



FMRI highlighting some areas of the pre-frontal cortex.

Automated detection of brain atrophy patterns based on MRI for the prediction of Alzheimer's disease, by C. Plant, S.J. Teipel, A. Oswald, C. Böhm, T. Meindl, J. Mourao-Miranda, A.W. Bokde, H. Hampel, M. Ewers.

DBSCAN Examples

Some More Examples

A novel approach for predicting the length of hospital stay with DBSCAN and supervised classification algorithms

Simulation of DNA damage clustering after proton irradiation using an adapted DBSCAN algorithm

Where traffic meets DNA: mobility mining using biological sequence analysis revisited

Individual Movements and Geographical Data Mining. Clustering Algorithms for Highlighting Hotspots in Personal Navigation Routes