Unsupervised Learning: Clustering with DBSCAN

Mat Kallada

STAT 2450 - Introduction to Data Mining

Supervised Data Mining: Predicting a column called the label

The domain of data mining focused on **prediction**:

- **Predict** stock prices
- **Predict** animal species
- **Predict** court case outcomes
- **Predict** [YOU NAME IT!]

Supervised Data Mining emphasizes prediction of a column.

Supervised Data Mining: The art of prediction

Constructs a function to predict a label given an **unlabeled feature vector**

Your Collected Data

Feature Vector	Label
<1,4>	5
<5,1>	6
...	...

Supervised Data Mining: The art of prediction

Constructs a function to predict a label given an **unlabeled feature vector**

Your Collected Data

Feature Vector	Label
<1,4>	5
<5,1>	6
	...

Construct f(x) using a data mining technique

Predicted Label = f(any vector)

f denotes the relationship between features and labels

Supervised Data Mining

- Neural Networks
- Decision Trees
- K-Nearest Neighbours
- Support Vector Machines

Can attempt to approximate the underlying function

Formulated by either:

- Classification (categorical predictions)
- Regression (numerical predictions)

Unsupervised Data Mining

Another Domain of Data Mining

Methods that **do not predict a label column**

Only working with feature vectors

No label here!

Feature Vector
<1,4>
<5,1>

Clustering and Dimensionality Reduction are typically **unsupervised**

Unsupervised Data Mining

Unsupervised learning helps us find **interesting information** about our dataset solely looking at the features alone.

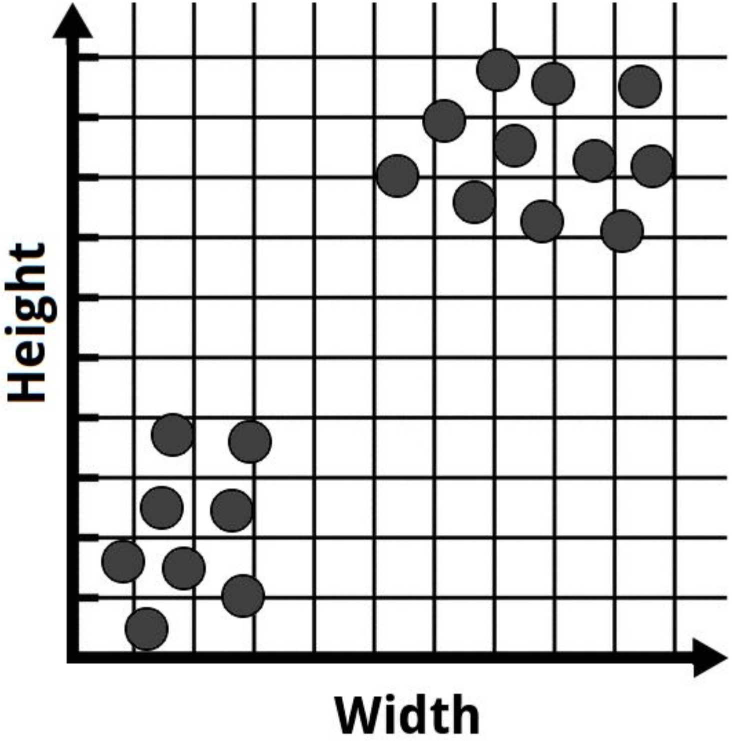
What is the task of “Clustering”

What is the task of “Clustering”

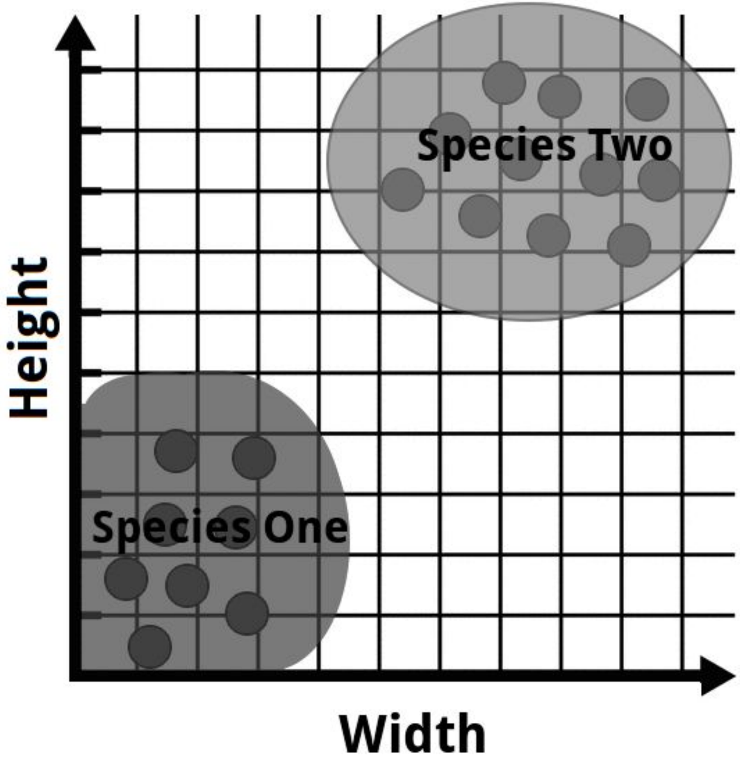
Assigning groups to a set of feature vectors

For example, finding the different personality groups that follow you on Twitter

What is the task of “Clustering”



What is the task of “Clustering”



What is the task of "Clustering"

By eyeballing that data, you would probably say that the **blob of observations on the left** is one animal species, and the **blob on the right is a different species**.

Clustering can be thought of assigning "classification labels" to unlabeled data.

What are some algorithms to perform clustering?

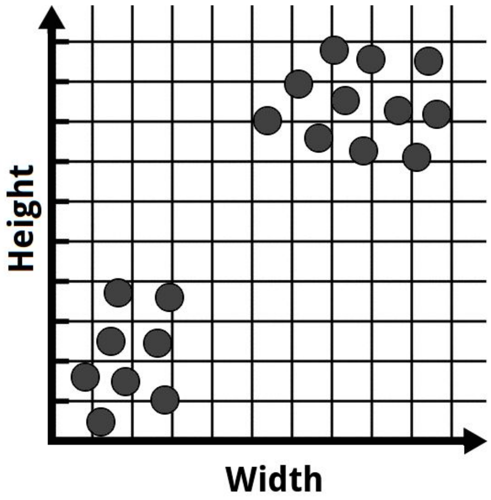
Two popular ones are:

- DBSCAN Clustering
- K-means Clustering

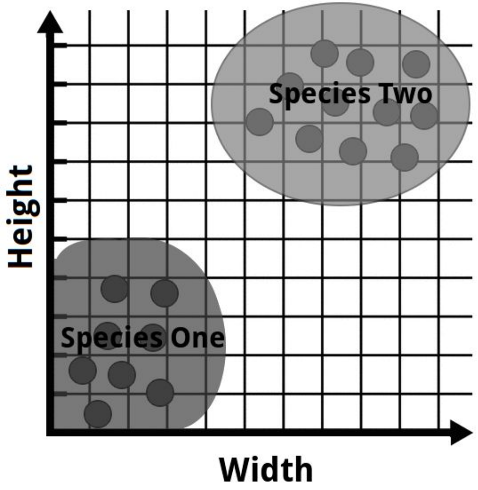
We will look at **both of these** in this course!

Let’s think about this...

What was your thought process here?



How did you do this?



DBSCAN: Density-based Clustering

You made the guess by looking at **how close these data points** were to one another.

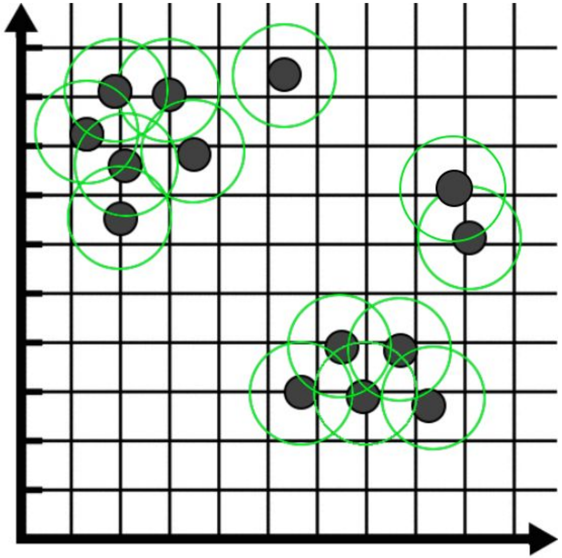
DBSCAN: Density-based Clustering

Looking at the **density** (or closeness) of our observations is a common way to discover clusters in a dataset.

In this lecture, we will be looking at a density-based clustering technique called DBSCAN (an acronym for “Density-based spatial clustering of applications with noise”).

DBSCAN: The first step of the algorithm

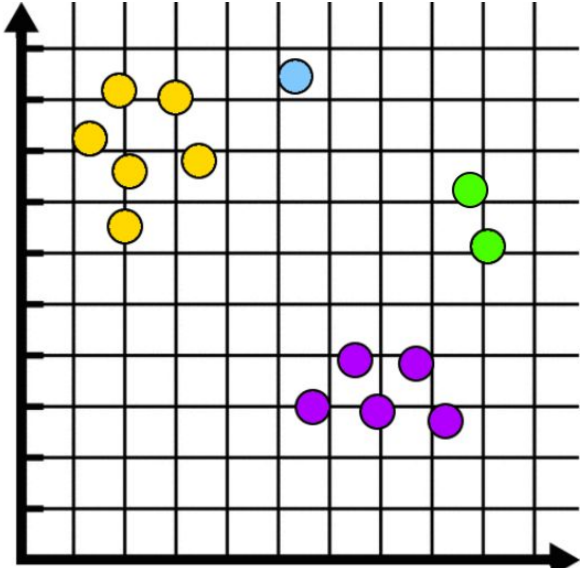
Step 1: DBSCAN starts by identifying the neighboring observations of each observation **within some radius (a hyperparameter)**.



Identifying radius of *some length* for each vector

DBSCAN: The <u>second step</u> of the algorithm

<u>**Step 2:**</u> Any data point that is <u>**within the data point of radius**</u> of another data point are in the **same cluster**



Observations that are within another observation radius are assigned to the same cluster.

<u>Four clusters</u> were found in total.

NOTE: DBSCAN is sort-of like KNN

In the sense that we need to find all **neighbours** of a given data point

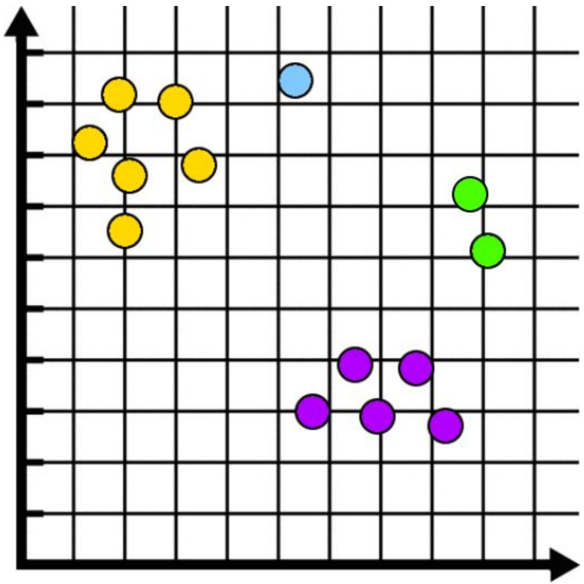
We need to find the **neighbourhood** of each point and this is computationally intensive

DBSCAN: The second step of the algorithm

There is something **pretty odd** with the clusters on the right.

There are three observations which are noise and we end up creating two clusters entirely from these bad observations themselves.

What should we do to avoid this?



Blue cluster and green cluster are formed from noise!!

DBSCAN: The <u>second step</u> - avoiding noise

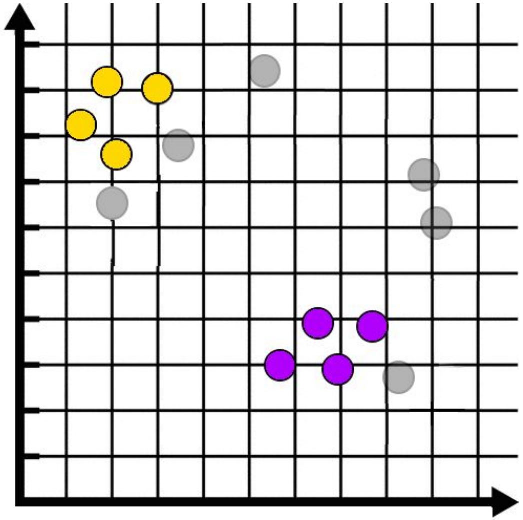
How about we only consider points that contain a **<u>minimum number of samples</u>** within their radii?

The other data points can be neglected and be considered as noise.

Let me go ahead and set this value to two; meaning that there must be at least 2 observations within the radius of a data point to be accounted-for.

The faded data points are considered noise and are not associated with any clusters.

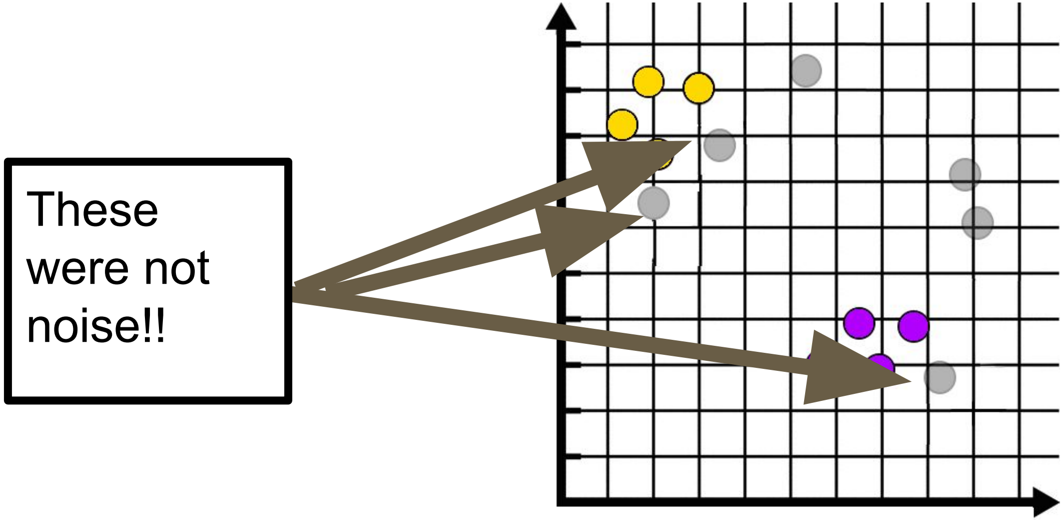
DBSCAN: The second step - avoiding noise



These **colored points** have at least 2 data points within their neighbourhood.

Grey points do not have 2 data points in their raidus and are called **noise.**

DBSCAN: The second step - avoiding noise



Wait, this also doesn't seem so right

Although, the **noisy data points are now removed**, some of the data points on the edges of the clusters were also falsely chosen as noise.

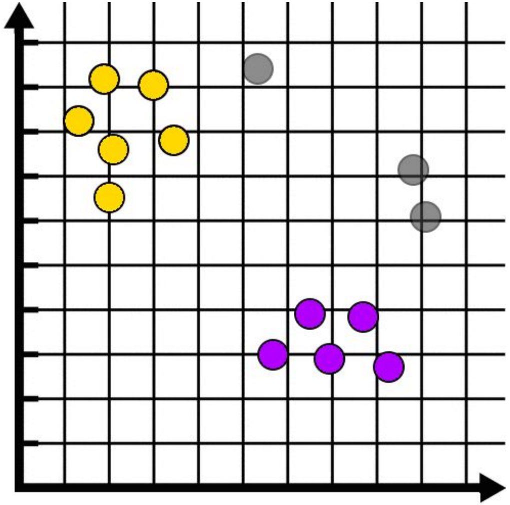
DBSCAN: The second step - avoiding noise

We need a way to fix this and retain these **edge data points** -

We should consider noise to be: observations which don't satisfy the minimum neighbour requirement **and** are not within the radius of a **core observation** (observations that do satisfy the requirement).

DBSCAN: The <u>second step</u> - avoiding noise

As shown below, if we try this then we'll knock out only the noisy observations.



DBSCAN: The Clustering Method in a Nutshell

That's it! We have two steps here.

Step 1: Compute the neighbourhood of all data points

Step 2: Group the data points that have at least some specific number of points in their cluster.

Edge points should be taken into consideration in Step 2

DBSCAN: The Hyperparameters

There are two hyperparameters we need to specify here:

Radius, *How far apart should observations be, to be considered in the same cluster?*

Minimum # of Samples, *How many observations should be in the radius of a data point?*

DBSCAN: The Hyperparameters

How do we pick these hyperparameters?

DBSCAN: The Hyperparameters

We don’t have **any training score**!

This is **unsupervised**!

Therefore, we can’t grid search.

DBSCAN: The Hyperparameters

We have to manually choose these parameters using **domain-specific knowledge** related to the problem at hand

Interpret whether the resulting clusters makes logical sense.

In other words, pick them and hope for the BEST!!!

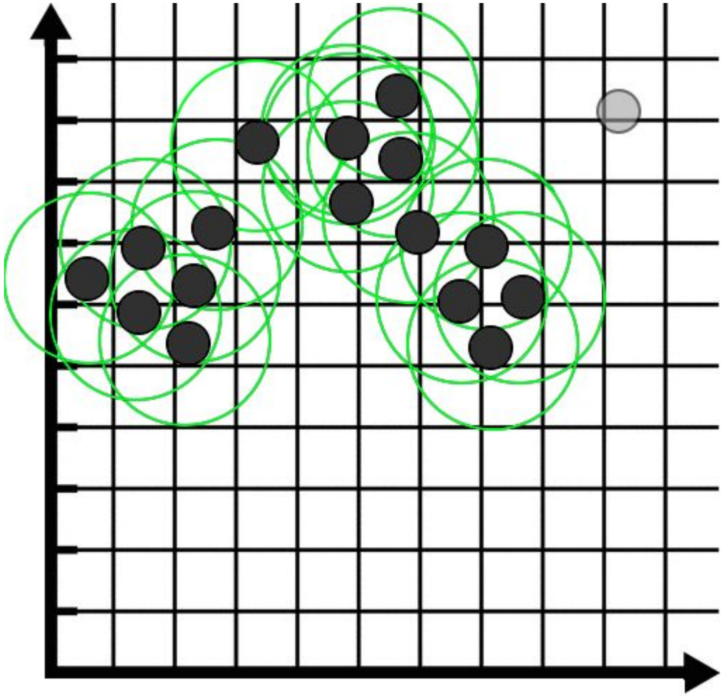
Choosing the hyperparameters is still an active topic in the literature. There are a ton of “rules of thumb”, but nothing concrete.

DBSCAN: Incorrect Radius

Radius is set way too high on right

DBSCAN thinks that there is only one cluster!

What would happen if its too low?

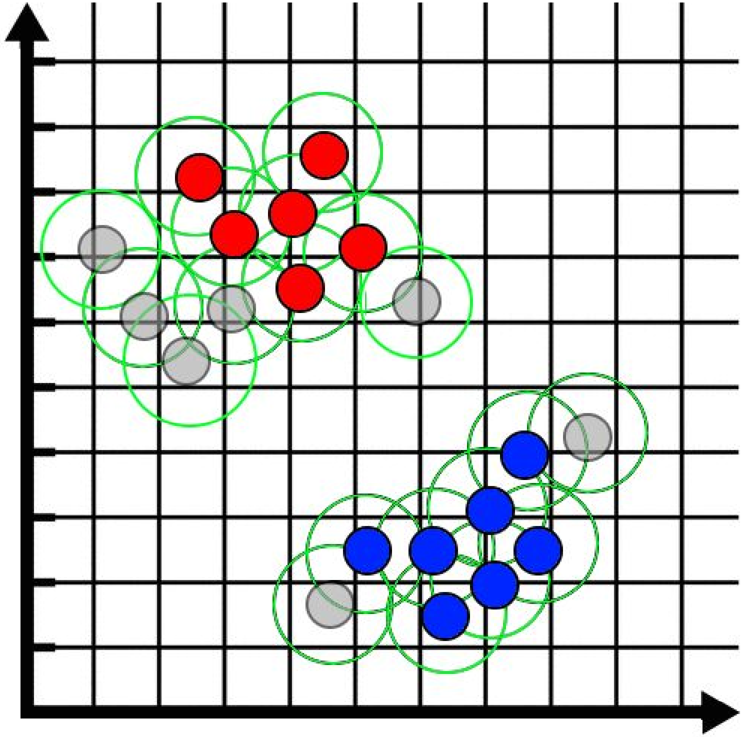


DBSCAN: Incorrect “Minimum Sample” Parameter

Minimum Sample is set way too high on right

DBSCAN thinks good observations are noise.

What would happen if its too low?



DBSCAN: Hyperparameters

Experiment with these two parameters and look into the observations of the resulting clusters and determine whether they are logically similar.

In the case of clustering measurements (heights in cm, widths in cm) of petting zoo animals, I would go for a **radius** value of around 10, since I'm not expecting too much variation between cluster samples

For the **minimum number of samples**, I'd pick 4-5 just in case the zoo has a decent number of deformed animals - if there was a lot of deformed animals in the zoo, I would opt for a higher value for this parameter.

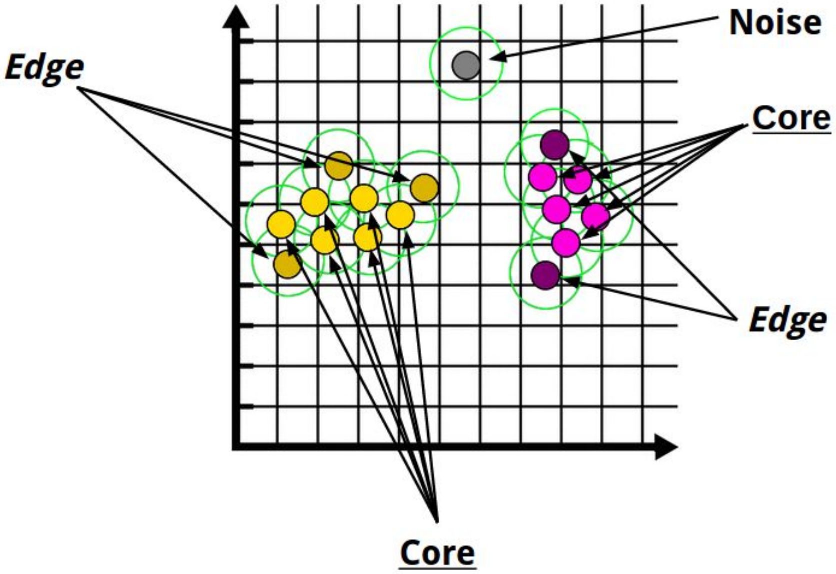
DBSCAN: Types of points in DBCAN

Let's make a note that we've encountered three types of data points in the context of DBSCAN.

Observation Type	Description
Core	*Data points lying within the cluster itself*: data points which satisfy the minimum samples requirement
Edge	*Data points lying outside the cluster*: data points that are within the radius of a core point yet do not satisfy the minimum samples requirement.
Noise	*Data points that are bad training observations*: data points that do not contain the minimum number of samples nor are within the radius of a core point.

DBSCAN: Types of points in DBCAN

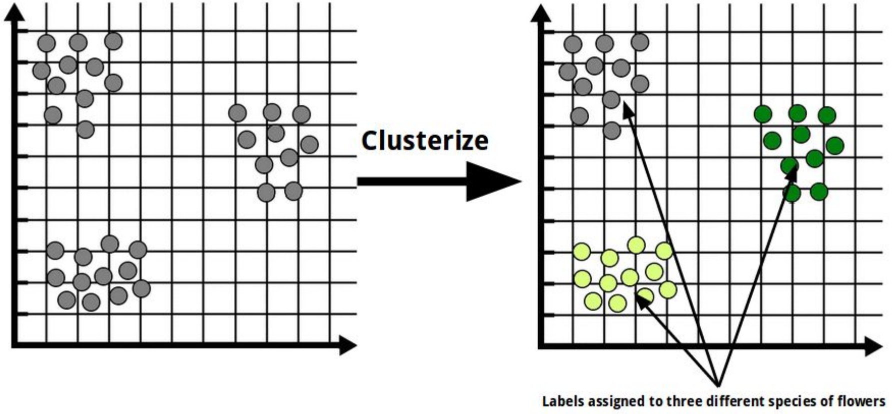
Below is a graphic indicating the different observation types on a sample dataset clustered with DBSCAN.



DBSCAN: A clustering approach!

Clustering is great for understanding the organization of a dataset.

For example, you could discover the different types of customers based on loyalty characteristics, hence getting a better idea how to serve them better.



Why can’t we evaluate the performance of clustering?

Why can’t we evaluate the performance of clustering?

We don’t actually know the answer here.

That is, we don’t know the “right” cluster associations

It is **unsupervised**, there is no “answer” or “label” column

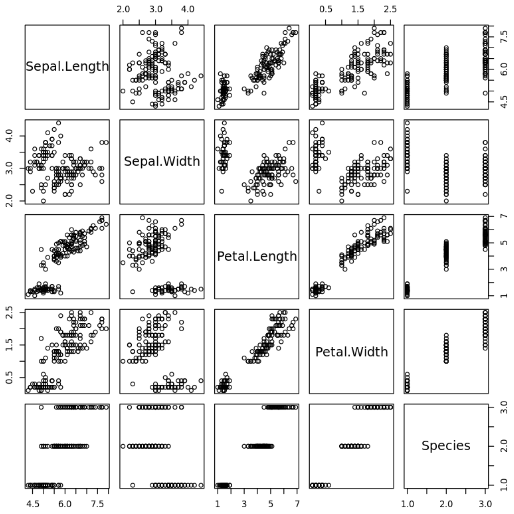
DBSCAN in R

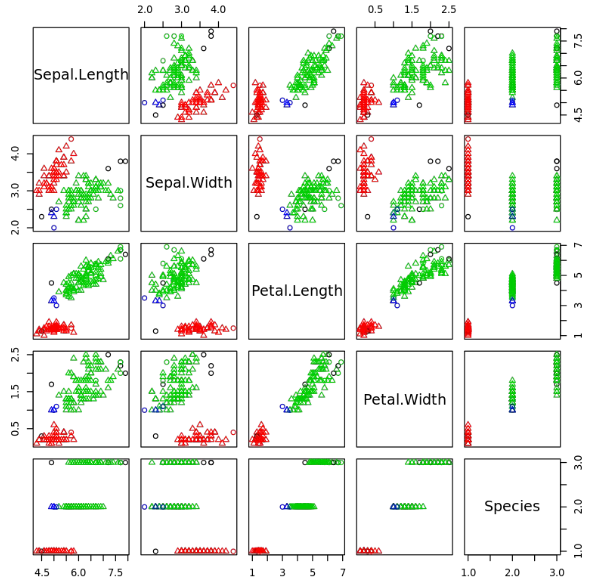
It’s time to put DBSCAN clustering into play with R’s **fpc package**.

We will try applying DBSCAN towards the iris flower dataset.

We’d expect to discover clusters which each represent a certain type of flower.

DBSCAN in R





DataJoy Link: https://www.getdatajoy.com/examples/56ddb58f697743550fc1c303

DBSCAN is affected by the “Curse of Dimensionality”

Data mining methods sometimes don’t work properly when with **high-dimensional** data

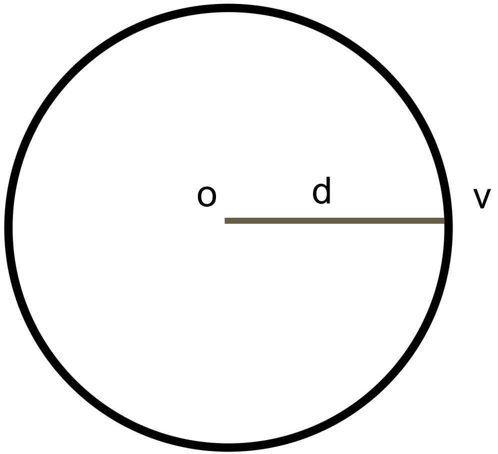
That is, datasets with a large feature space

Your cluster results sometimes may not make sense.

Any data mining technique that uses distance is subject to the curse of dimensionality

DBSCAN is affected by the "Curse of Dimensionality"

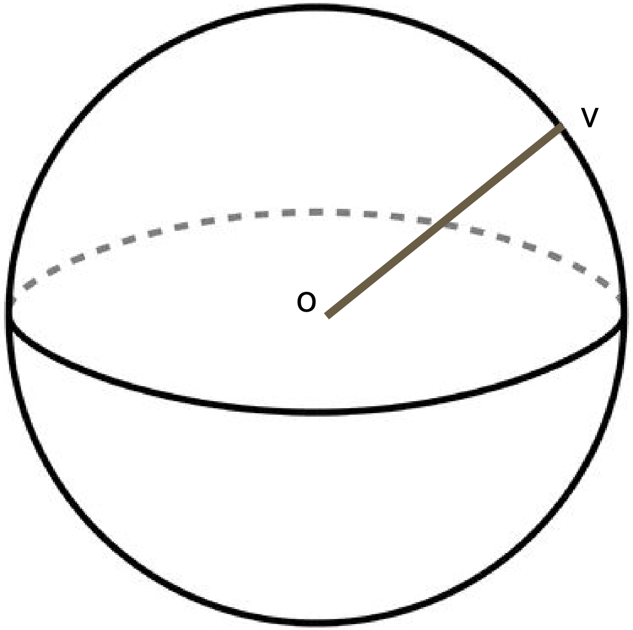
Euclidean Distance becomes meaningless in higher-dimensions



Say d(o, v) = 5

DBSCAN is affected by the “Curse of Dimensionality”

Euclidean Distance becomes meaningless in higher-dimensions

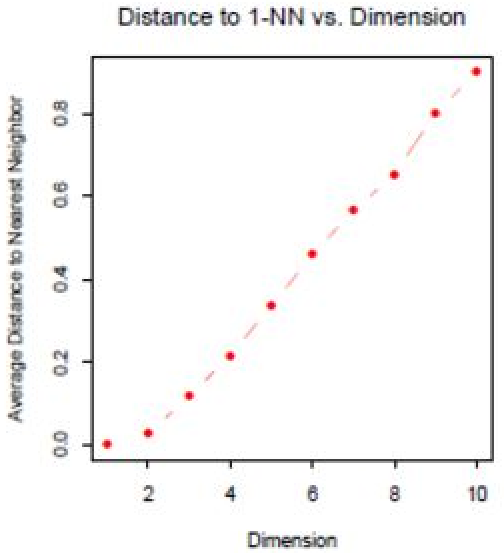


Now d(o, v) = 59

DBSCAN is affected by the “Curse of Dimensionality”

Euclidean distance between two points becomes larger as we add more dimensions

Comparing whether one observation is “similar” on the basis of euclidean distance becomes troublesome.



Next Class

- We will talk about **Principal Components Analysis**
    - A strategy for reducing dimensionality of data points
- **Assignment 4** will be released shortly