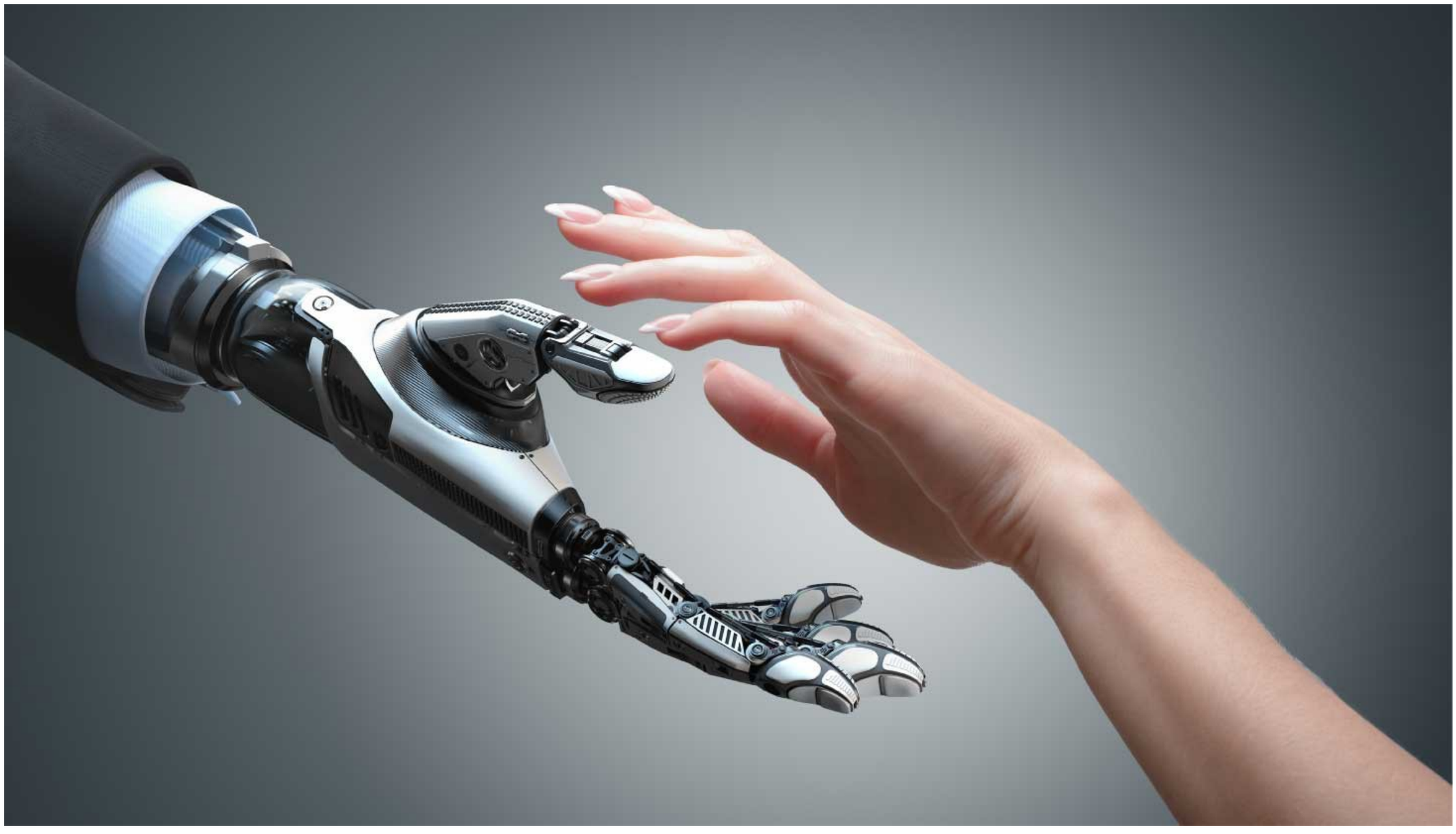
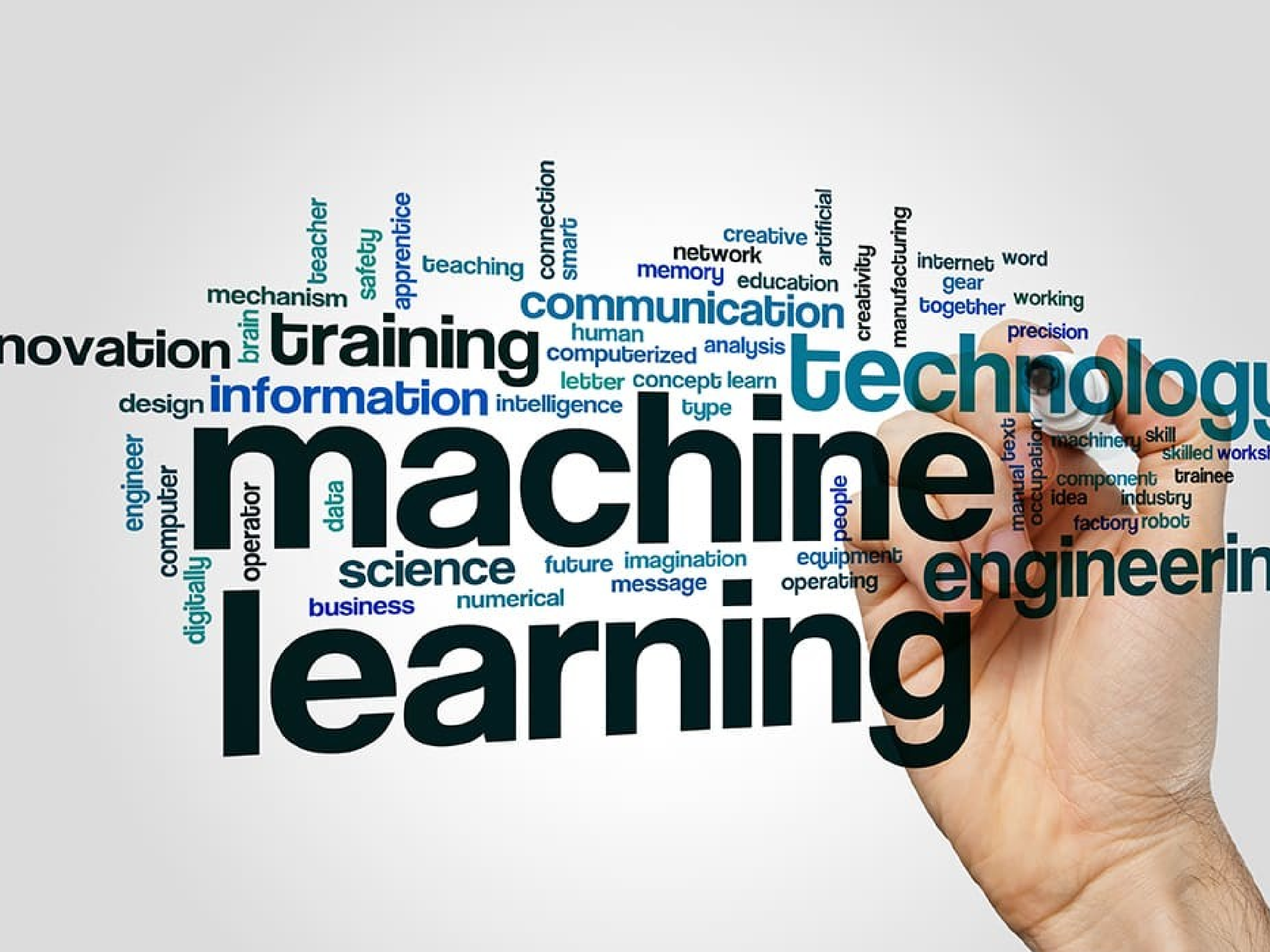
Data Analysis
using
Machine Learning



Various slides were derived from a talk by David Taylor

Myroslav Kavatsyuk, CMST


machine
learning
engineering
technology
training
communication
information
innovation
mechanism
brain
teacher
safety
apprentice
teaching
connection
smart
creative
network
memory
education
artificial
creativity
manufacturing
internet
word
gear
together
working
precision
human
computerized
analysis
letter
concept
learn
design
intelligence
type
machinery
skill
skilled
component
trainee
idea
industry
factory
robot
engineer
computer
digitally
operator
data
science
future
imagination
message
numerical
business
equipment
operating
people

What is machine learning (ML)?

"It is the use of <u>algorithms</u> to create <u>knowledge</u> from <u>data</u>."

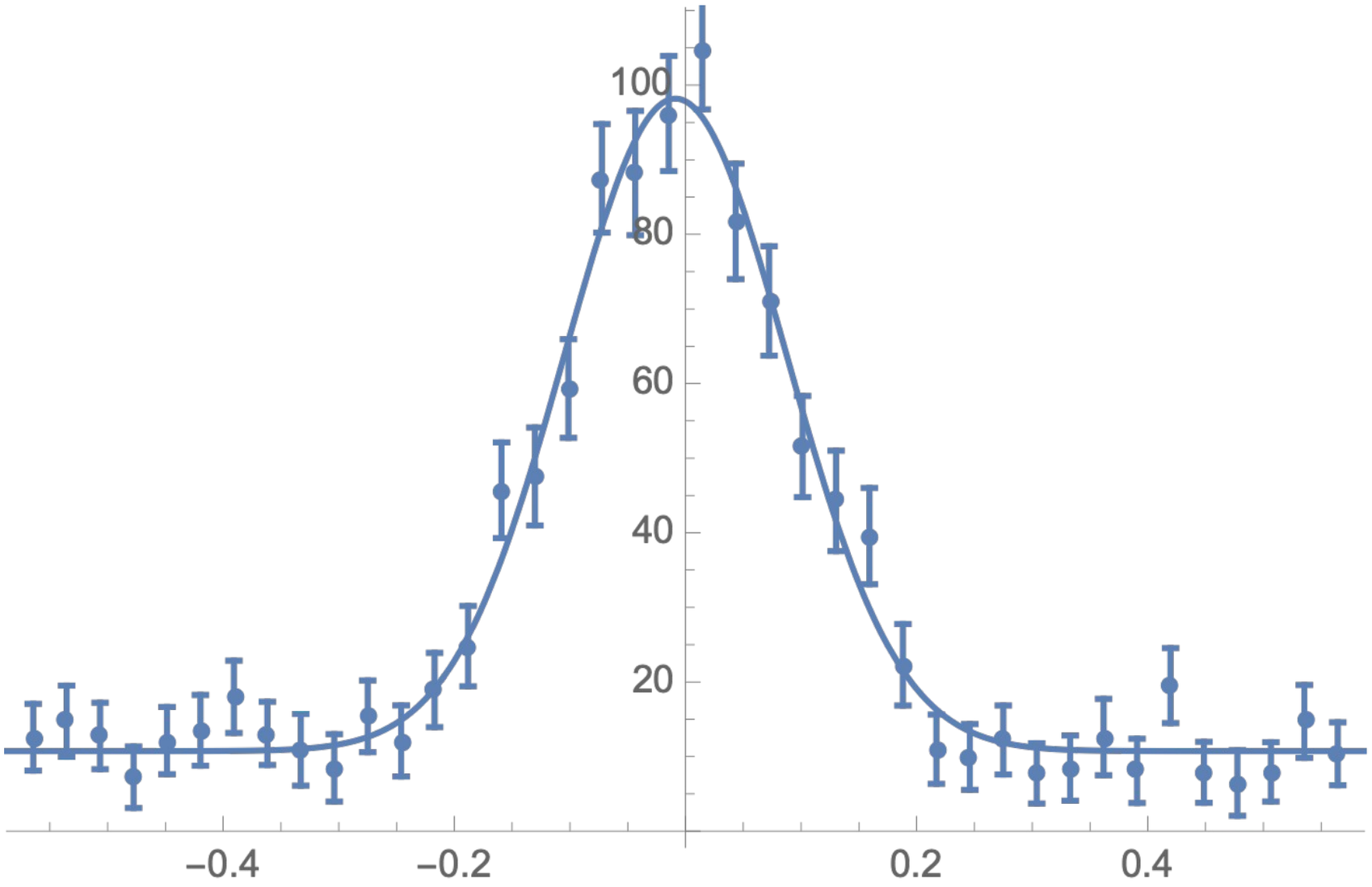
A set of instructions/rules,
e.g.
If A, then do B.
Then if C do D otherwise E,
etc.

Predict, model, classify,
e.g. getting smarter!

Any type of format
of a "measurement"
you can think of!

What is machine learning (ML)?

Q: How many have used ML methods in their study?



Should be everyone! —> *Regression*

Examples of ML

Autocorrect

Google page ranking

Netflix suggestions

Credit card fraud detection

Stock trading systems

Climate modeling and weather forecasting

Facial recognition

Self-driving cars

Nomenclature

… is a true disaster:

Random Forest

SOM

Clusters

Logistic Regression

Bias

(un)supervised

Gradient descent

Feature

Silhouette

Begging

Perceptron

MLP

K-Means

Boosting

Kernel Density

DBSCAN

KDE

kNN

KMedoids

BDT

Support Vector Machine

Learning Vector Quantisation

Vocabulary dates back from the 50’s

Upcoming lectures

- Focus on “basic” concepts of ML
- No hardcore mathematics, sorry ;(
- Unsupervised and supervised methods
- Review a few of the *many* algorithms
- Illustration by examples
- How to evaluate the performance of ML

- *Assignment*: data analysis of detector response to photons and neutrons

Lets take a “simple” example

The “fruit” dataset

“features”

“instances”

	color_id	color_name	elongatedness	weight	sweetness	acidity
169	4	orange	0.08	144	3.58	1290
170	5	red	0.11	182	3.58	1295
171	4	orange	0.11	144	3.59	1035
172	4	orange	0.09	143	3.63	1015
173	6	yellow	0.47	123	3.64	380
174	6	yellow	0.56	126	3.69	465
175	5	red	0.11	189	3.71	780
176	4	orange	0.19	144	3.82	845
177	5	red	0.09	191	3.92	1065
178	2	brown	0.15	152	4.00	1035

The “fruit” dataset

	color_id	color_name	elongatedness
169	4	orange	0.08
170	5	red	0.11
171	4	orange	0.11
172	4	orange	0.09
173	6	yellow	0.47
174	6	yellow	0.56
175	5	red	0.11
176	4	orange	0.19
177	5	red	0.09
178	2	brown	0.15

Color: a categorical variable, assigned an arbitrary numeric ID from 1-6.

I introduced "noise" by simulating color-blindedness in the observations,
e.g. green to red

The “fruit” dataset

	color_id	color_name	elongatedness	we		
169	4	orange	0.08	14		
170	5	red	0.11	18		
171	4	orange	0.11	14		
172	4	orange	0.09	14		
173	6	yellow	0.47	123	3.64	380
174	6	yellow	0.56	126	3.69	465
175	5	red	0.11	189	3.71	780
176	4	orange	0.19	144	3.82	845
177	5	red	0.09	191	3.92	1065
178	2	brown	0.15	152	4.00	1035

Elongatedness:

0 = circular,

1=twice as long as wide,

infinite = line

The “fruit” dataset

Weight, sweetness and acidity are numerical measurements; we have "lost" the units, but that's okay, since we'll standardize everything.

				edness	weight	sweetness	acidity
					144	3.58	1290
					182	3.58	1295
					144	3.59	1035
					143	3.63	1015
					123	3.64	380
					126	3.69	465
					189	3.71	780
176	4	orange	0.19		144	3.82	845
177	5	red	0.09		191	3.92	1065
178	2	brown	0.15		152	4.00	1035

How to manipulate associated tabular data?

There are **various** ways, but my advice:

Mathematica: **Dataset**

Python: **pandas (DataFrame)**

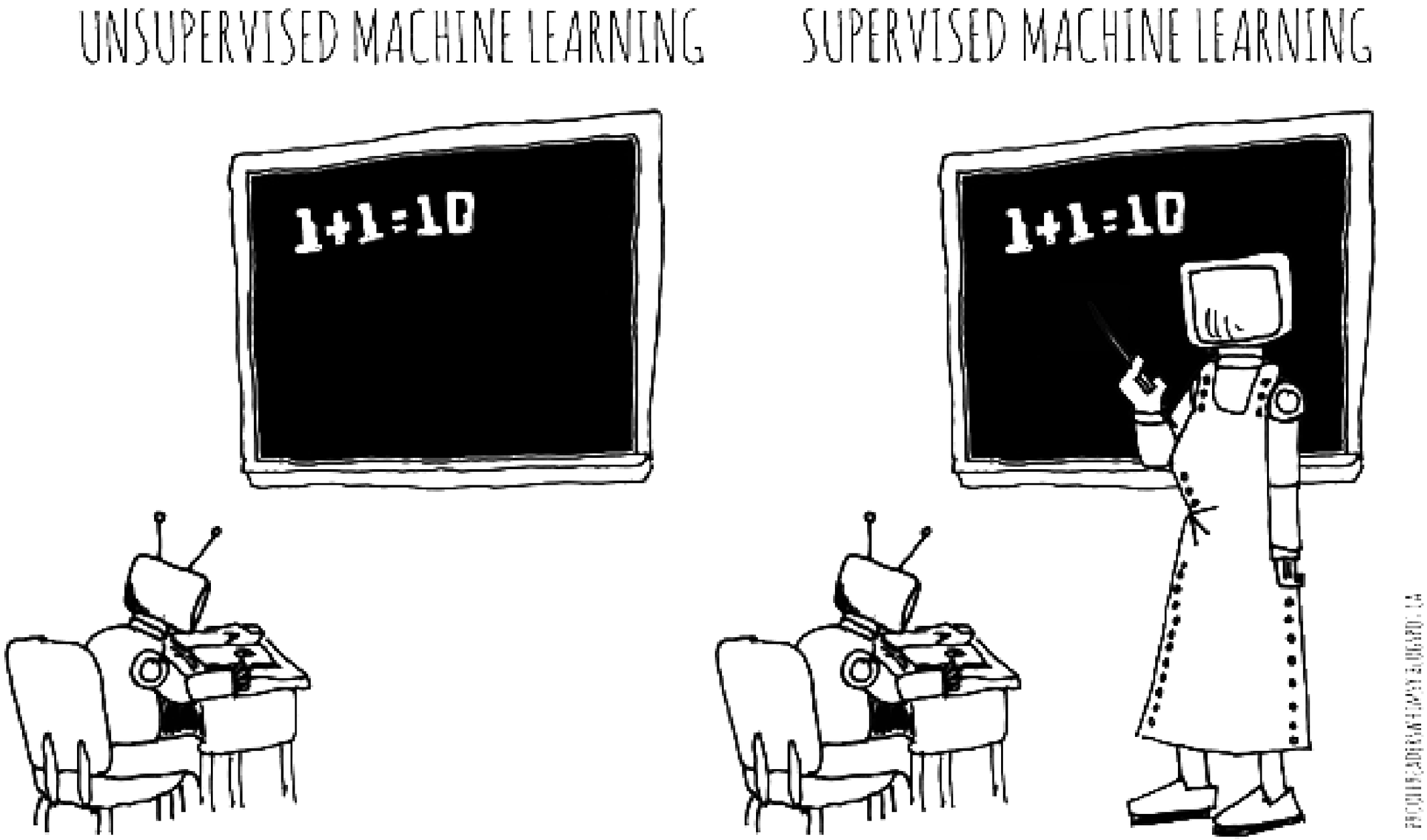
Importing data from a csv-file

$ cat *fruit.csv*

fruitid,fruitname,colorid,colorname,elongatedness,weight,sweetness,acidity
1,orange,2,brown,0.14,152,3.56,1095
1,orange,2,brown,0.09,167,3.33,1080
1,orange,2,brown,0.15,152,4,1035
1,orange,2,brown,0.03,155,3,680
1,orange,2,brown,0.08,147,2.69,1020
1,orange,2,brown,0.14,147,3.55,1045
1,orange,2,brown,0.14,152,3.33,985
1,orange,2,brown,0.18,159,2.88,1515
1,orange,3,green,0.08,152,3.17,1185
1,orange,4,orange,0.18,174,3.03,1120
1,orange,4,orange,0.12,162,3.1,1260
1,orange,4,orange,0.09,186,3.2,830
1,orange,4,orange,0.18,176,3.38,795
1,orange,4,orange,0.15,162,2.87,1285
1,orange,4,orange,0.14,173,2.57,1130
… etc …

Let’s import our fruit data!

Unsupervised versus supervised



Unsupervised = *exploratory* ML

Supervised = *trained* ML

Unsupervised = *exploratory* ML

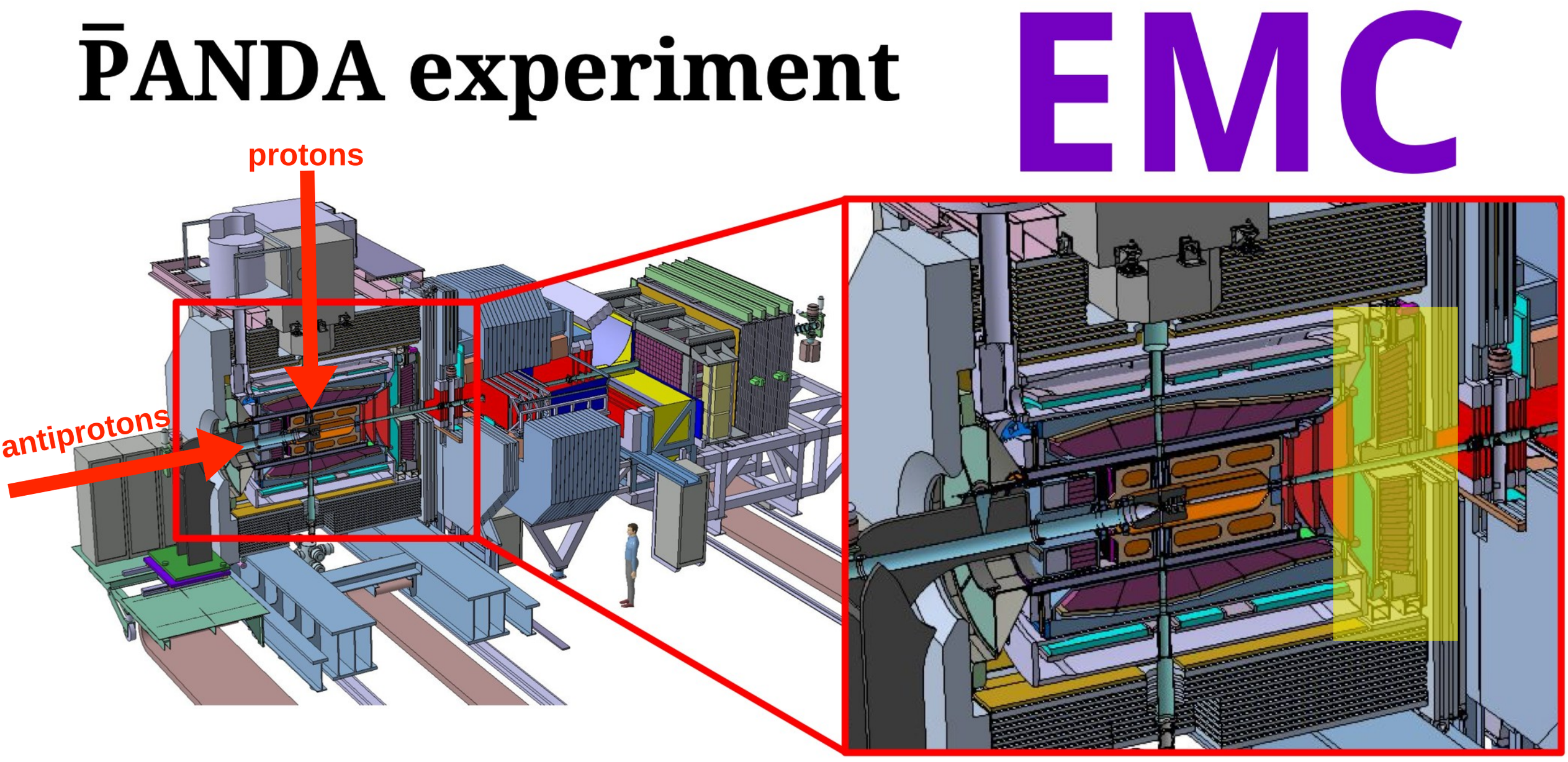
Unsupervised learning is also known as "clustering".

We try to find "clusters" in which members of the cluster have more in common with each other than instances that are not in the cluster.

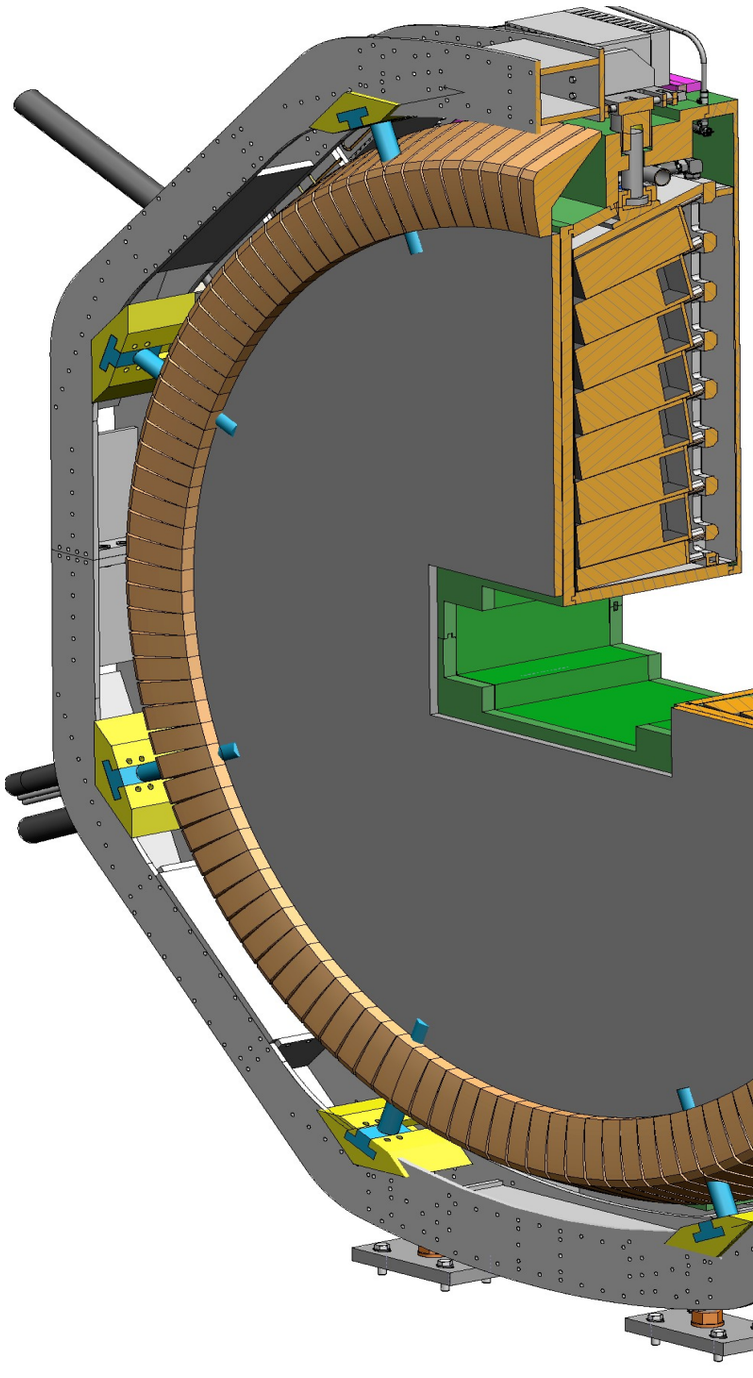
Unsupervised = *exploratory* ML

Python: ***sklearn.cluster.<xyz>***
Mathematica: ***FindClusters[...]***

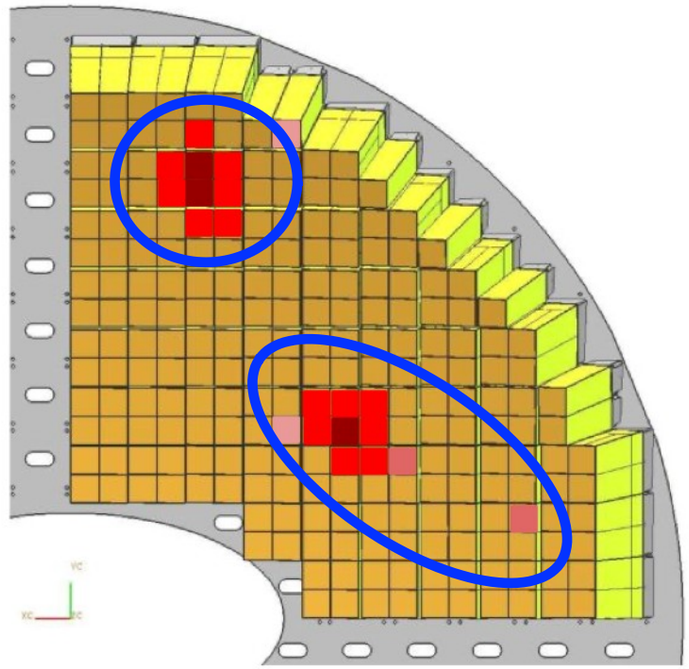
Clustering - example from my own research…

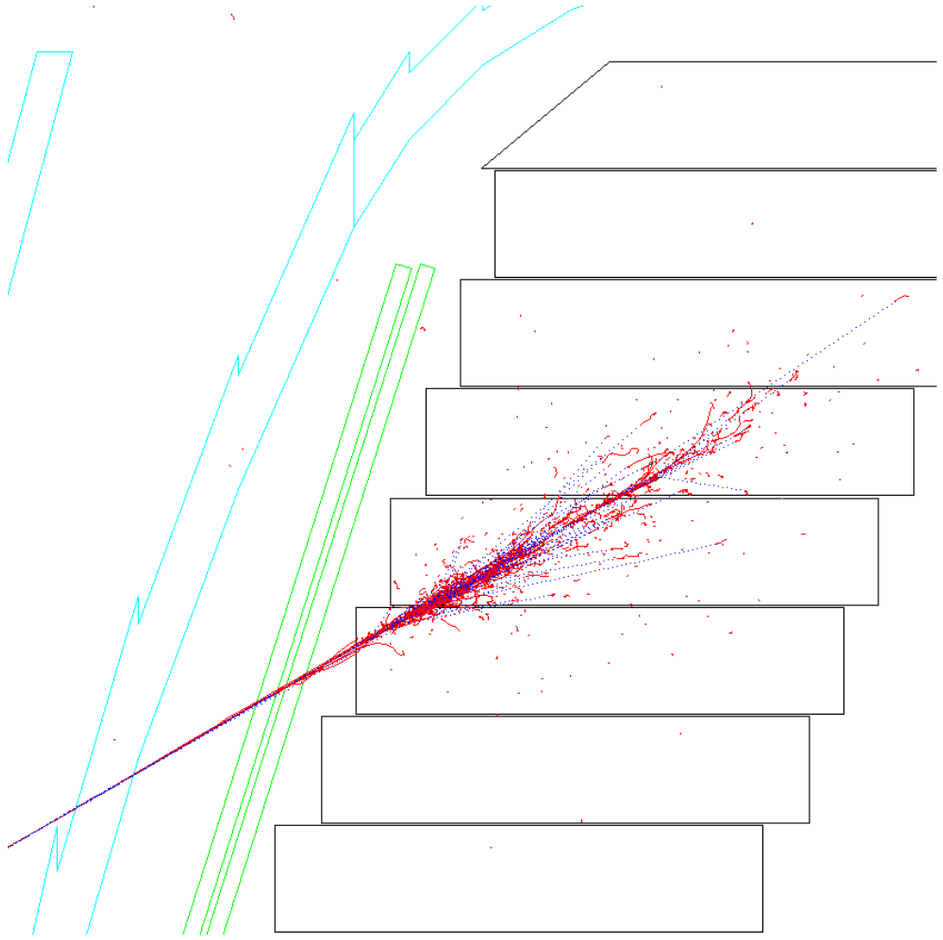


Clustering - example from my own research…

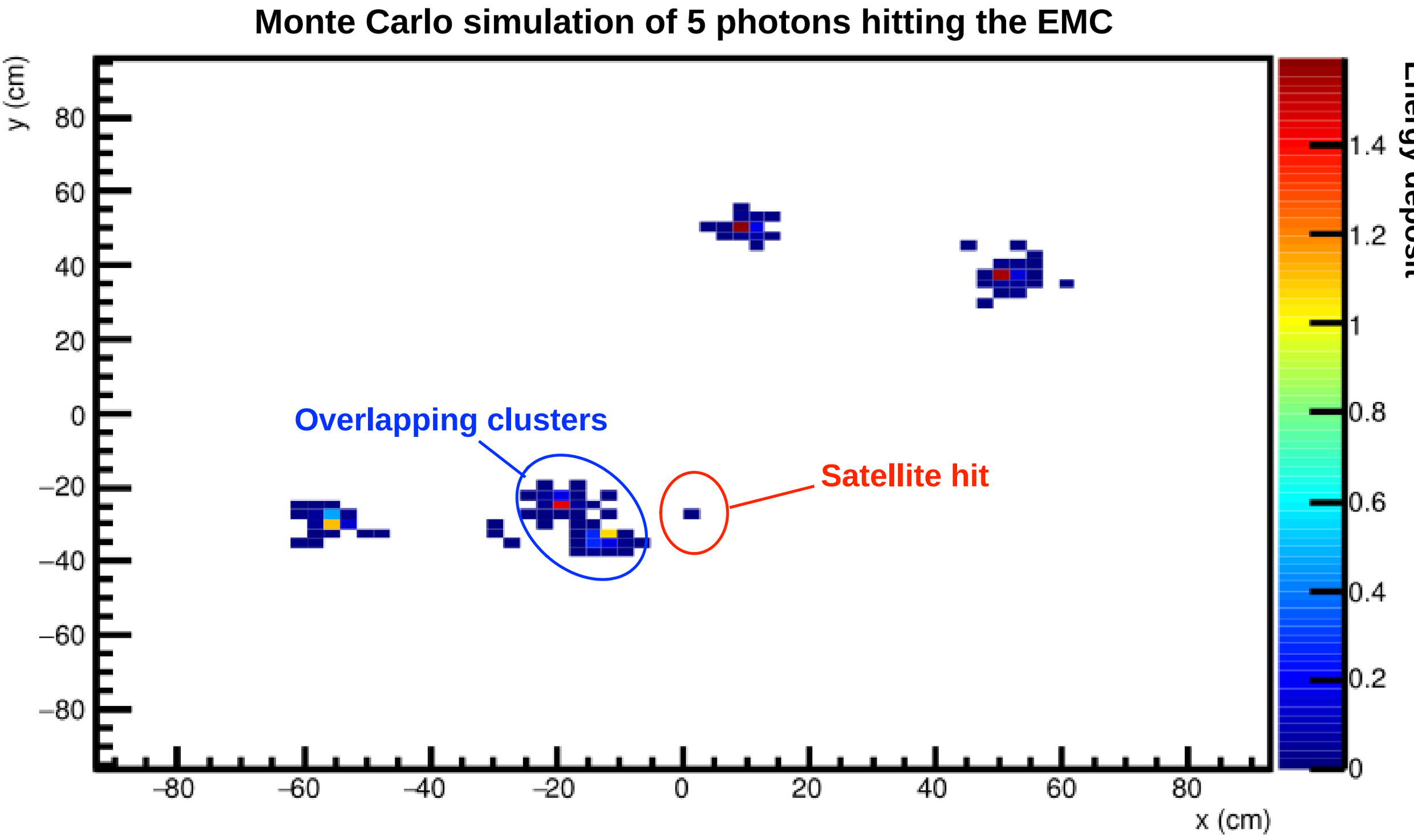


Electromagnetic Calorimeter
“Forward Endcap”





Clustering - example from my own research…

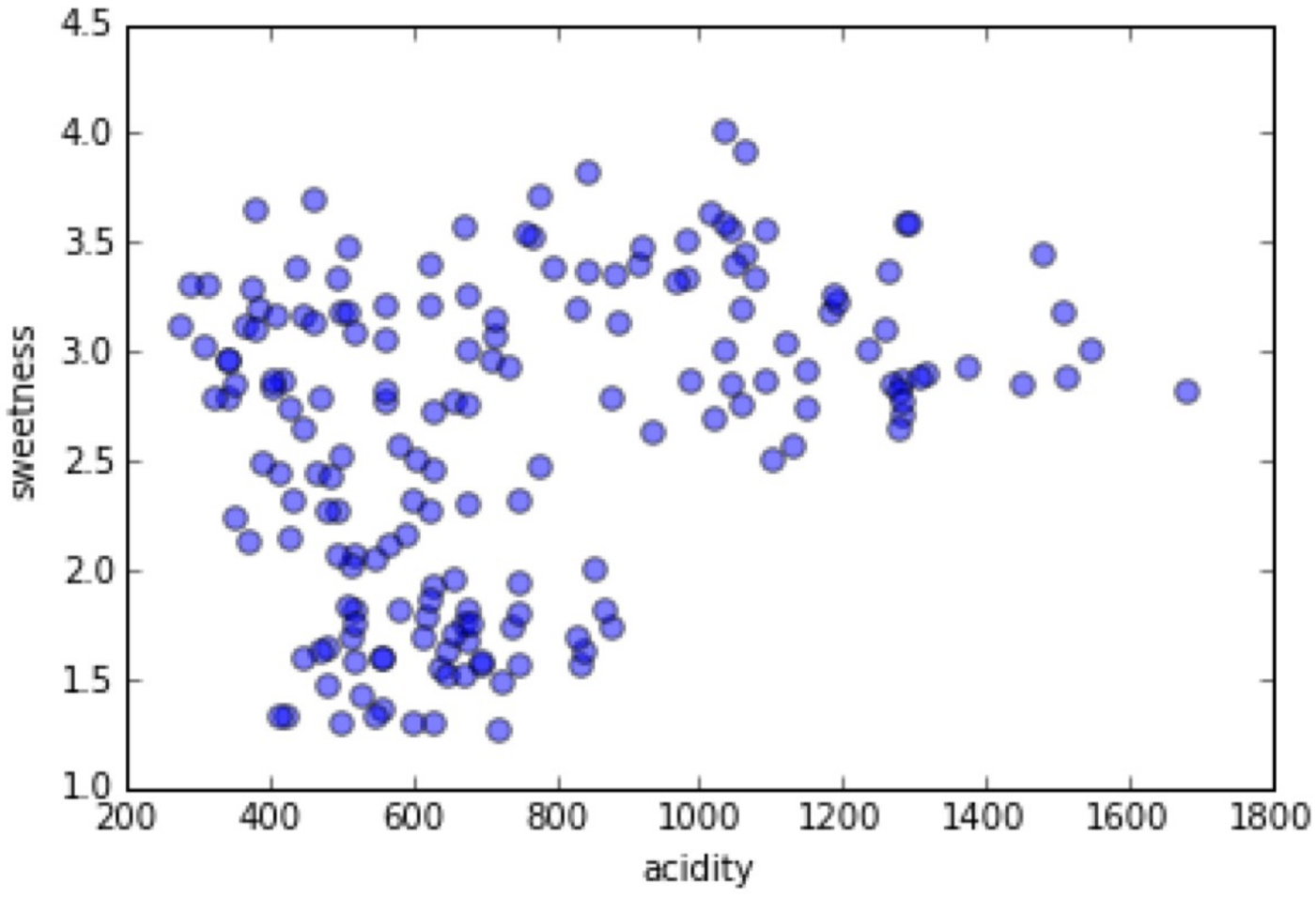


Let's do some

Unsupervised Machine Learning

of our fruit dataset

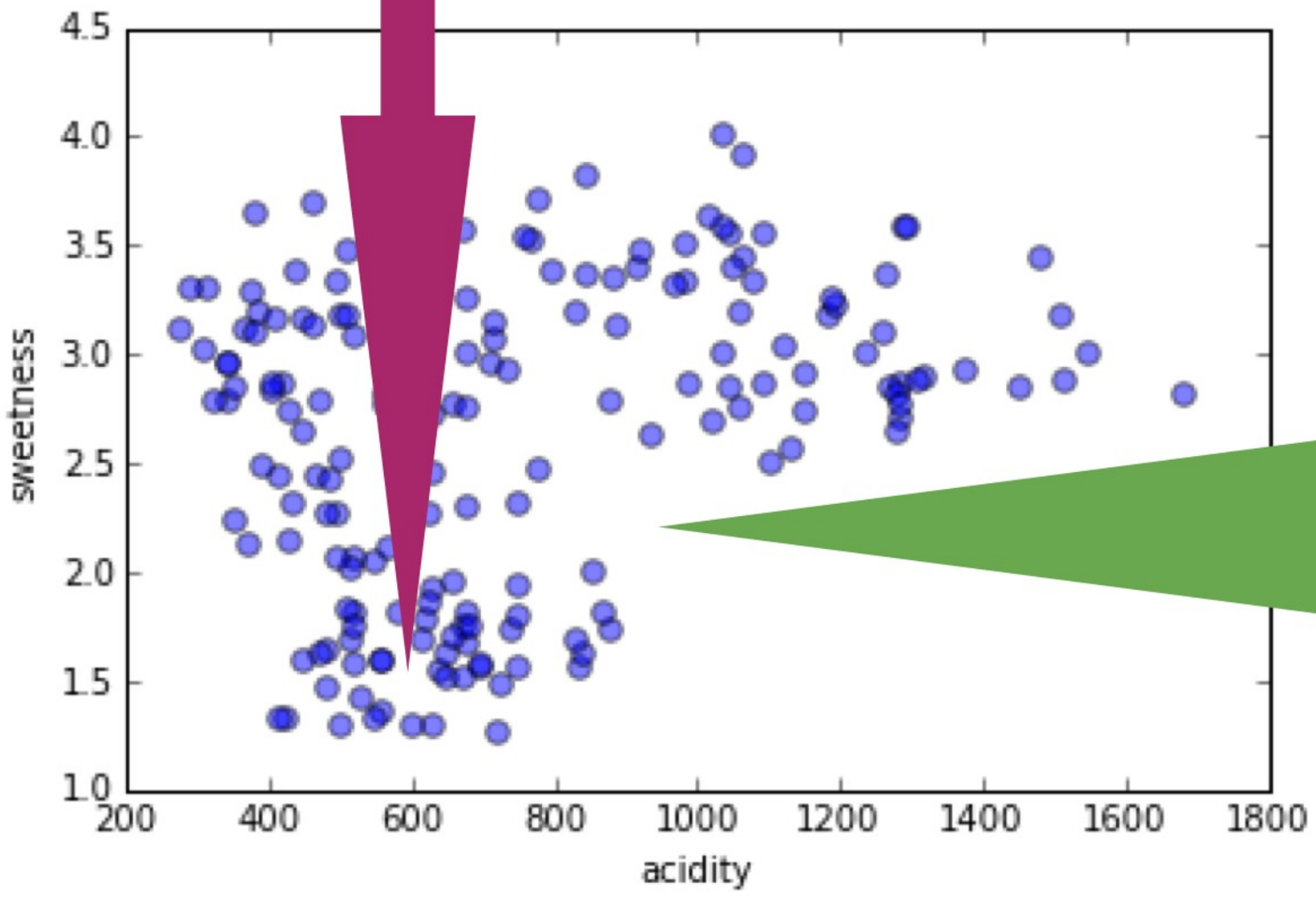
We will use only two of the five features of our dataset ("sweetness" and "acidity"), because it's a lot easier to visualize only two dimensions of numeric features, but clustering can be done on any number of dimensions with any kind of features.



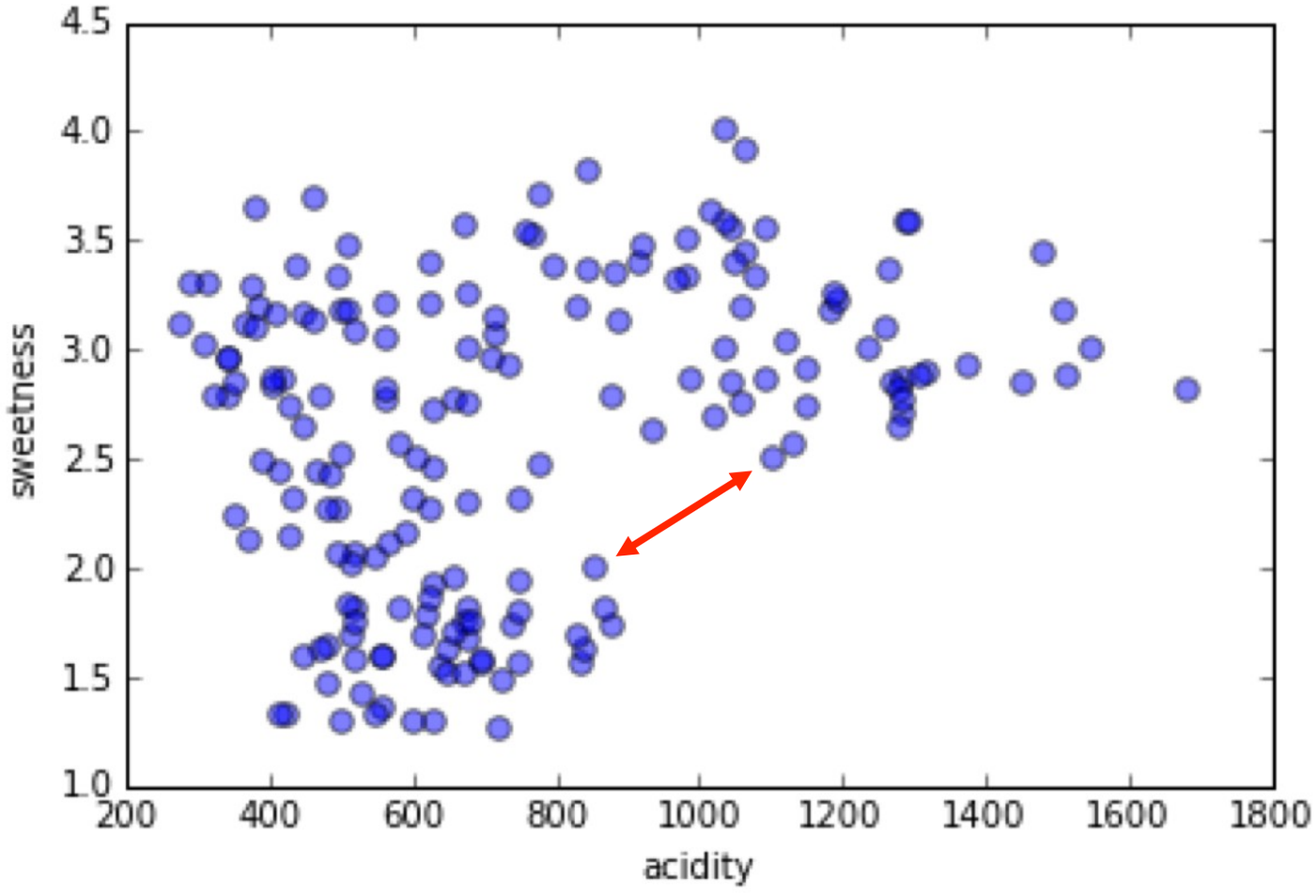
How many clusters do you see?

Another way to think of a cluster regions of **high density** separated by regions of **low density**.

By defining what we mean by high and low density, we can have a "natural", i.e. intrinsic to the dataset, determination of how many clusters there are.

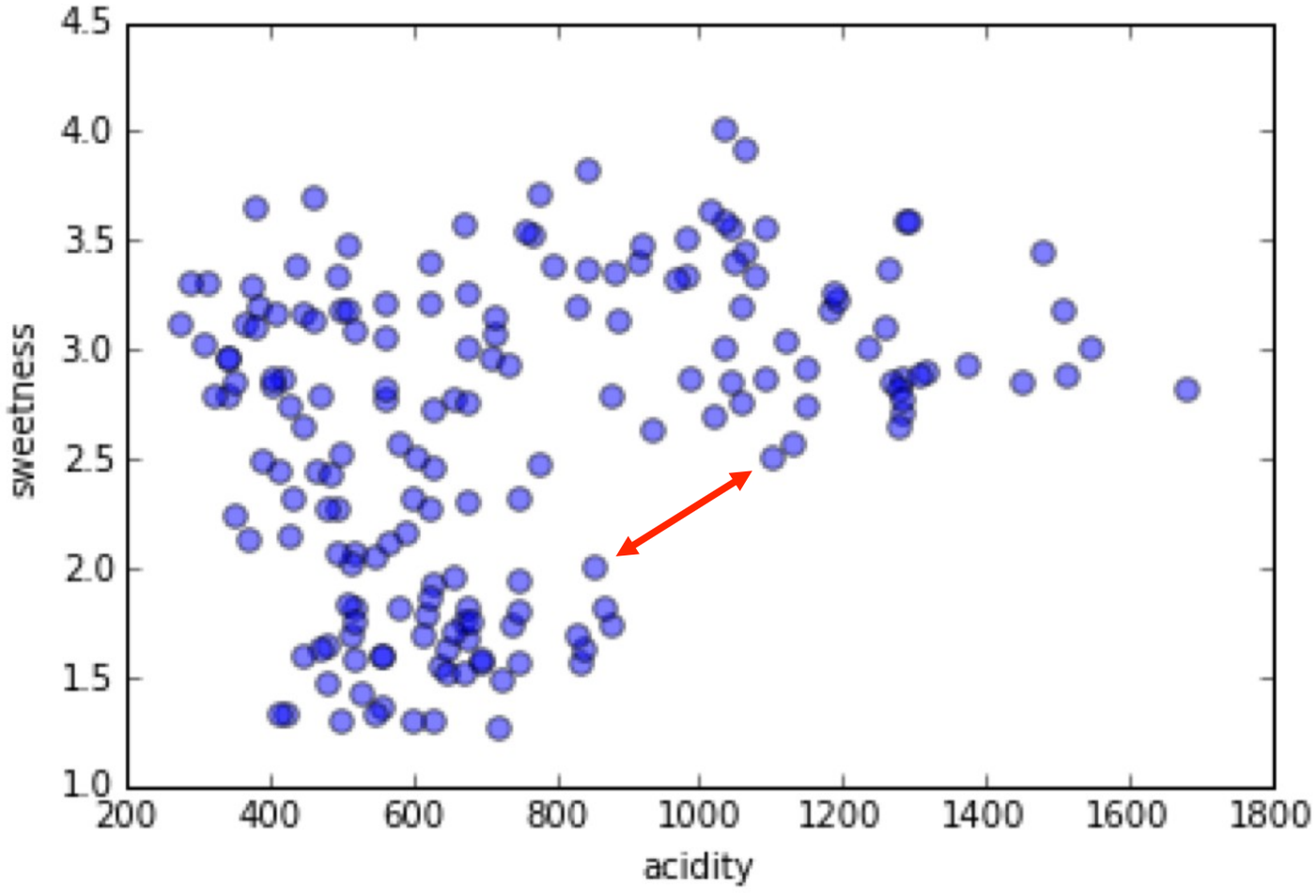


We need a distance function. (Sometimes we call it a 'similarity function'; similarity is the opposite of distance, but they measure the same thing.)



A common distance function for numeric data is Euclidian distance.

We need a distance function. (Sometimes we call it a 'similarity function'; similarity is the opposite of distance, but they measure the same thing.)



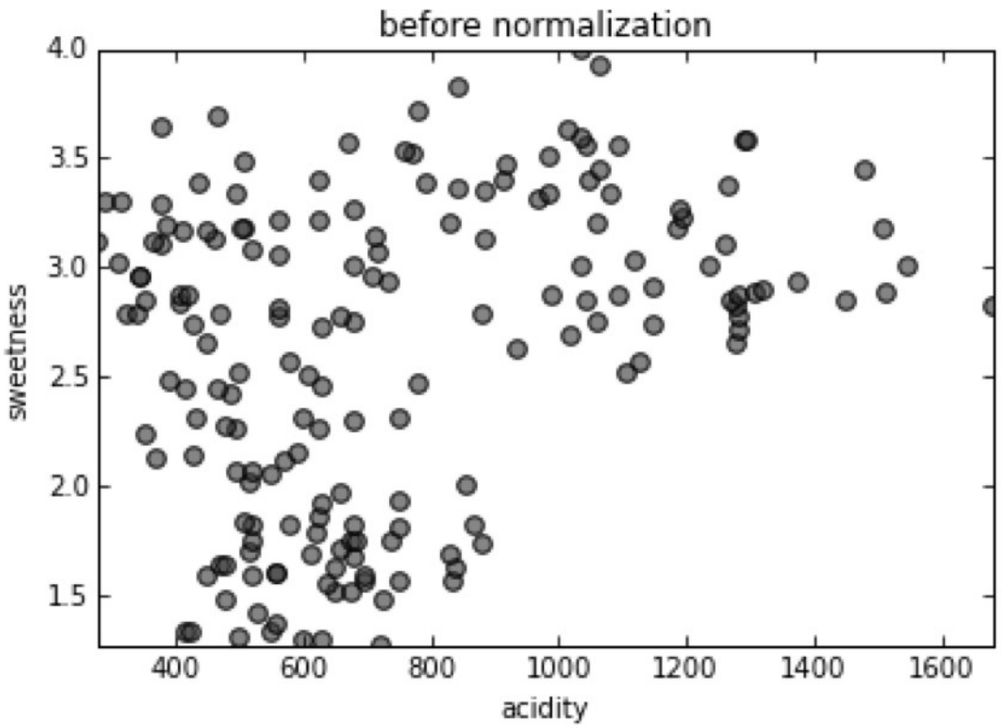
A common distance function for numeric data is Euclidian distance.

But the data must be normalized! Otherwise, in this case, acidity would have a much higher effect on the distance function than sweetness just because of its (arbitrary) units.

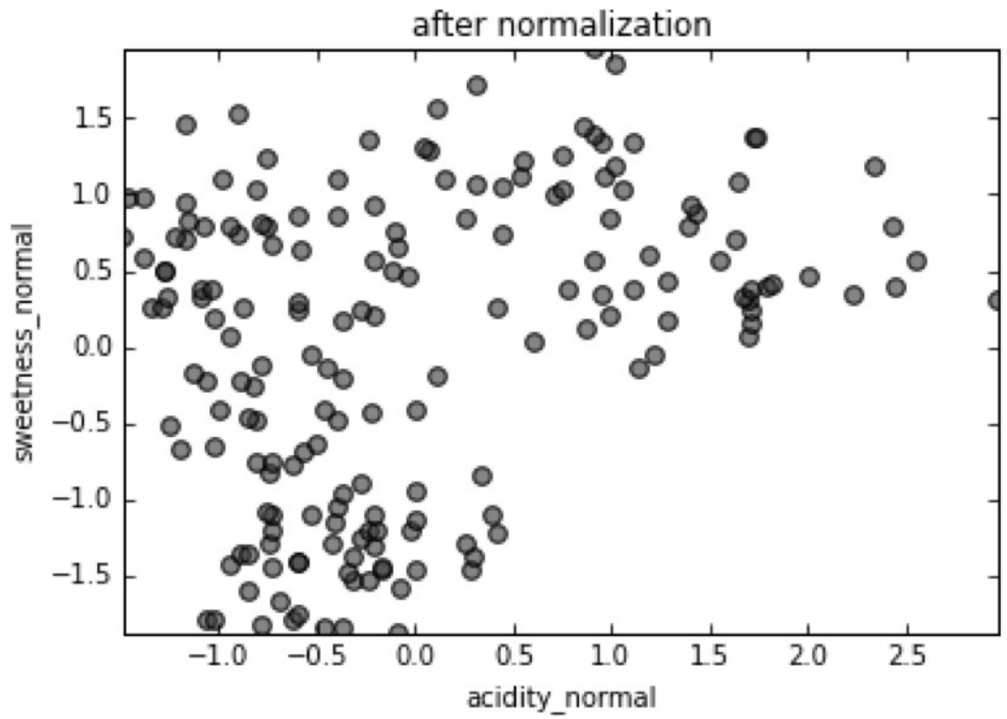
Standardization

To make features comparable, they must be normalized so that mean = 0 and standard deviation = 1.

(Every data point, subtract mean, divide by std. dev.)



Without normalization, acidity would determine statistics more than sweetness simply because its units are measured in the thousands.



Normalized data is important for some parametric algorithms

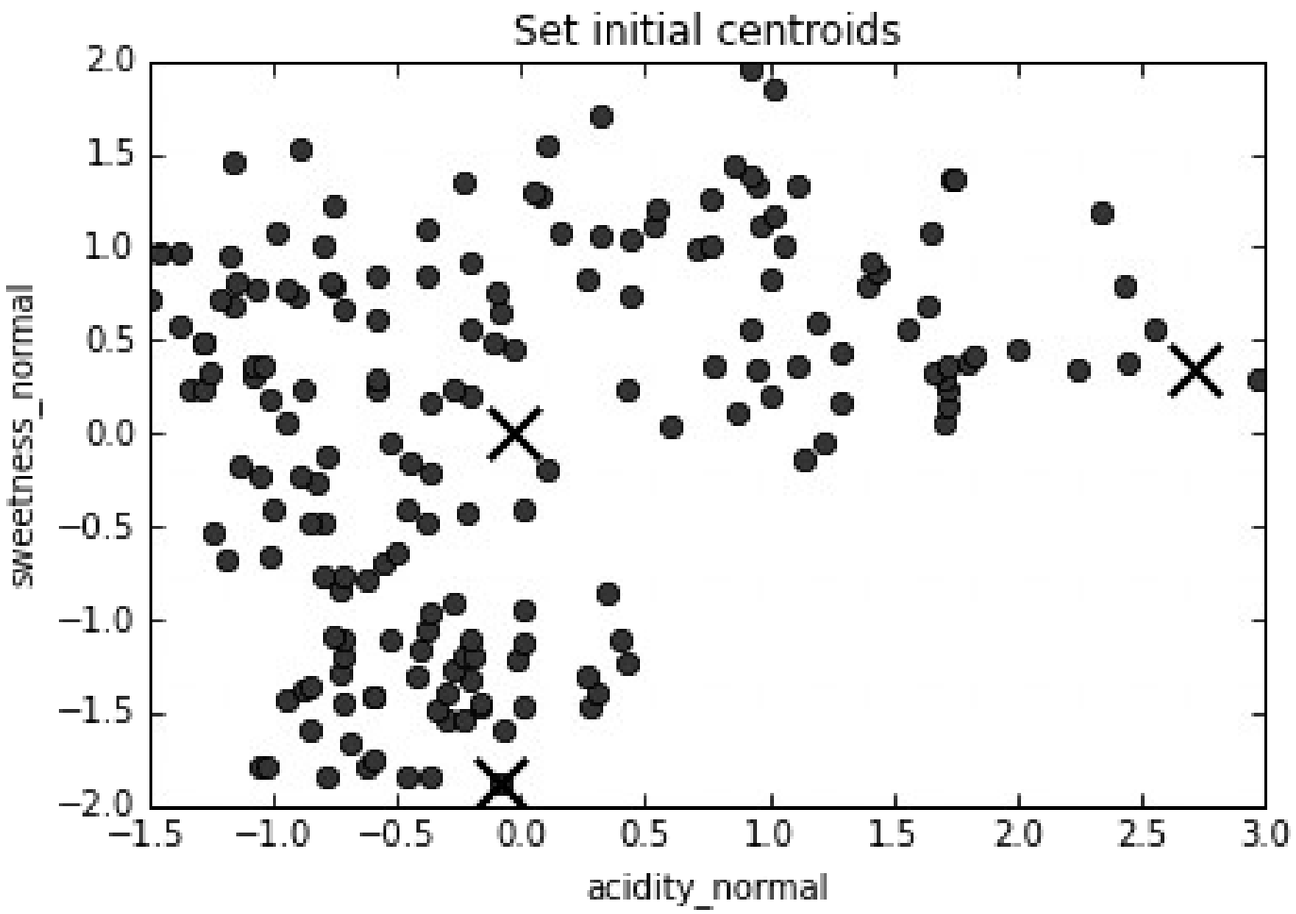
The K-Means algorithm

K-Means is the "go-to" algorithm for unsupervised machine learning, generally what everyone uses unless there's a reason to use something different.

It is robust and flexible and it scales well: if you have a lot of instances, random subsampling usually gives comparable results.

Its main drawback is you need to know how many clusters there are beforehand.

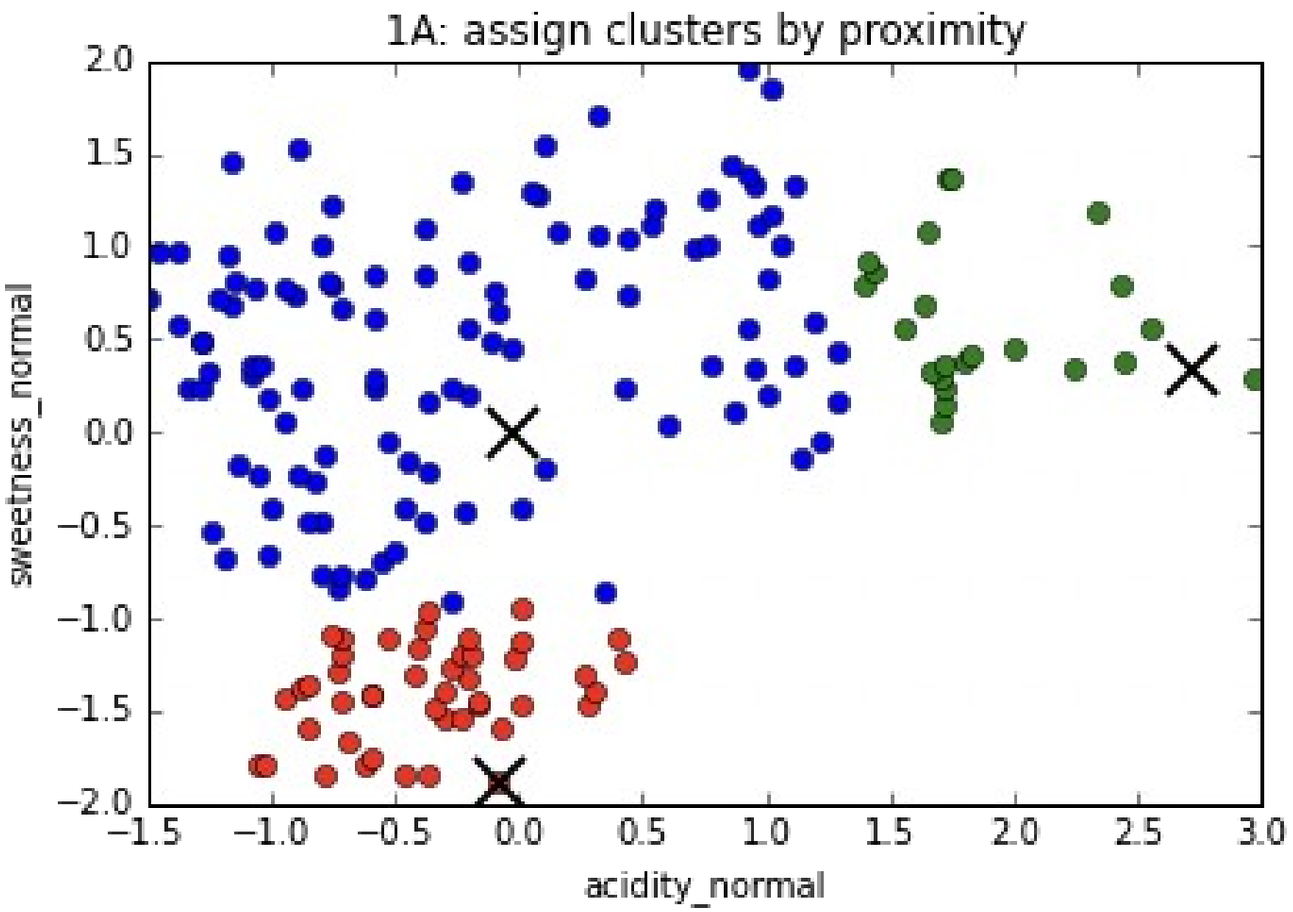
K-Means in action



1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

K-Means in action

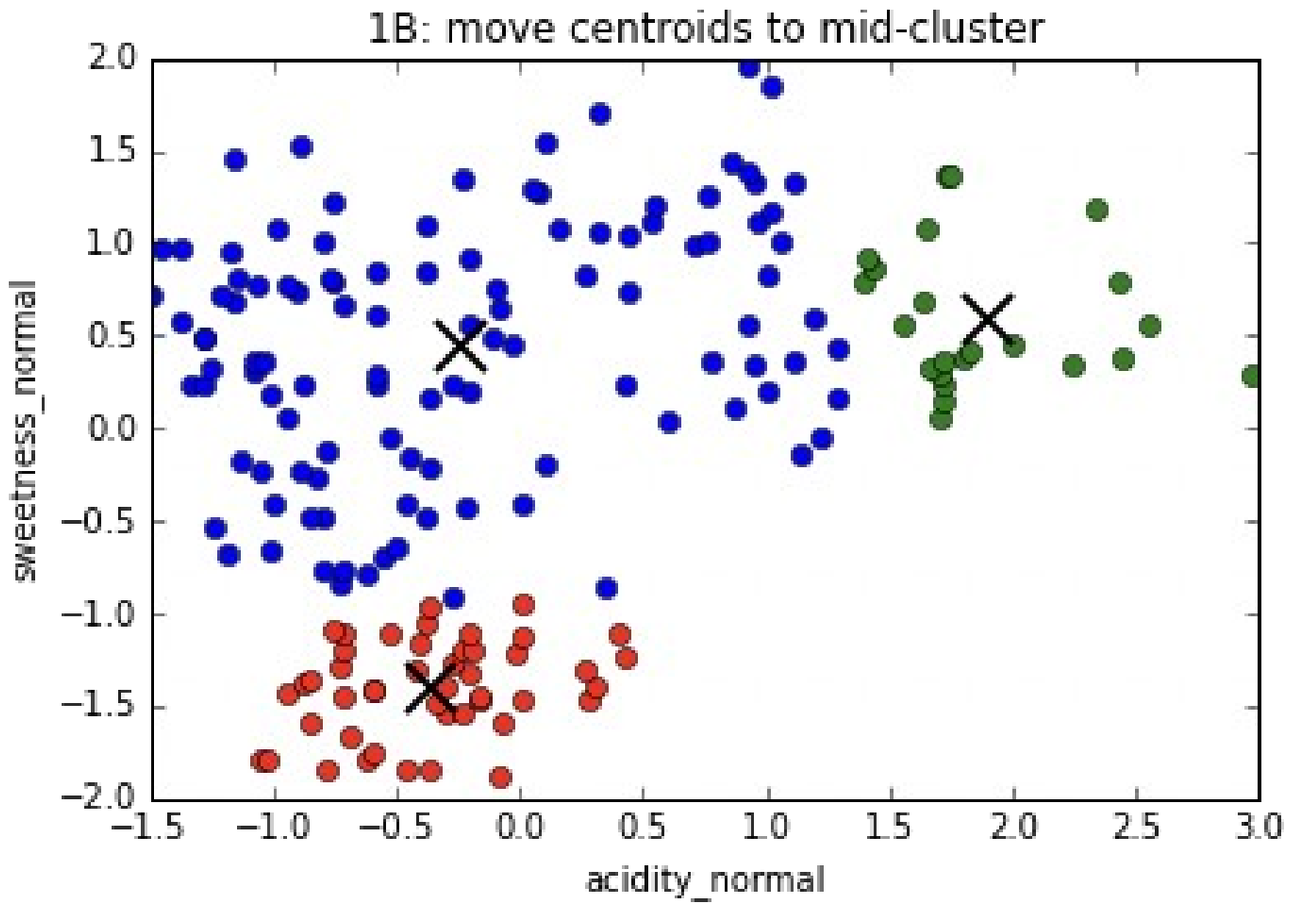


1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

K-Means in action



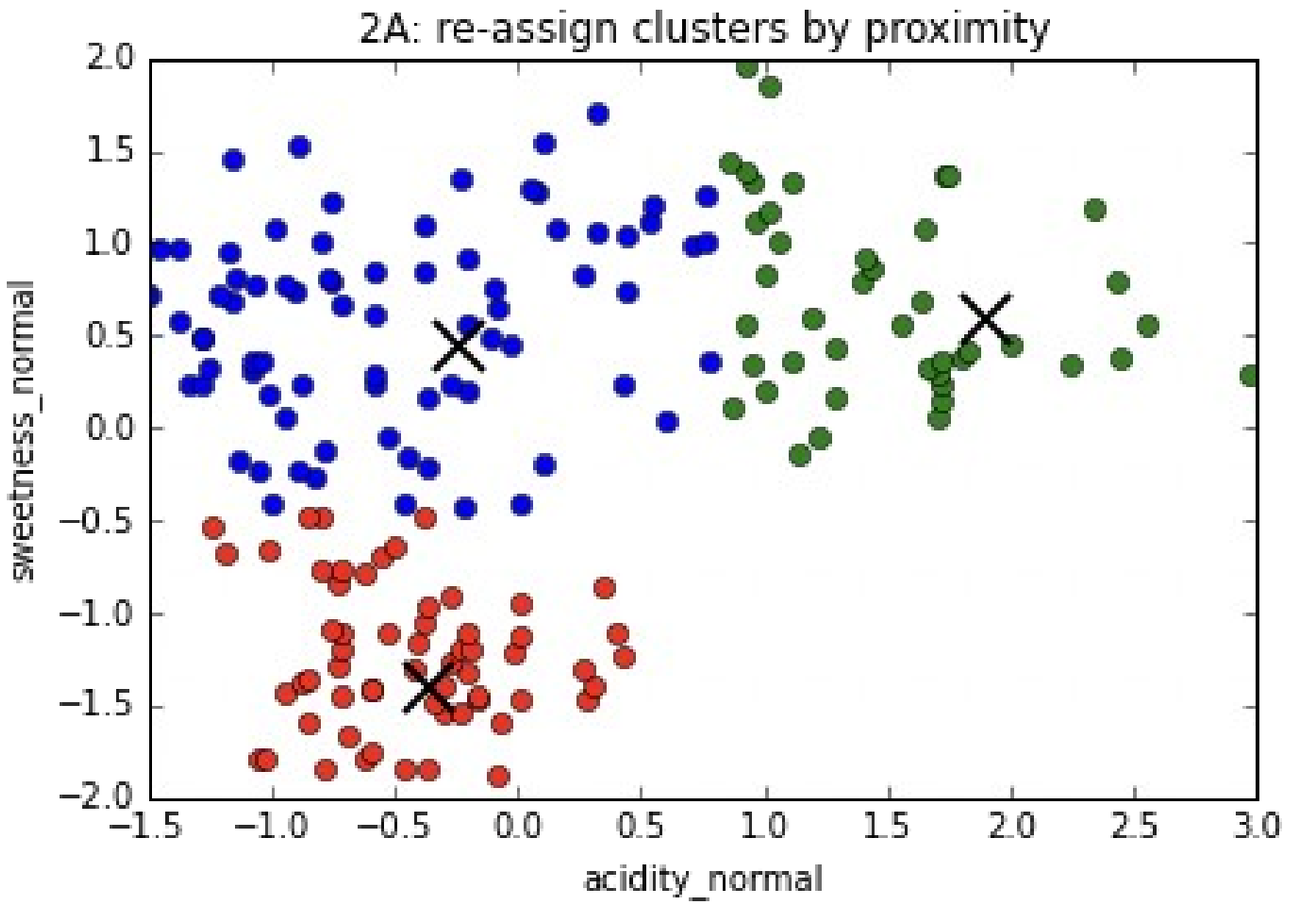
1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

4. Move centroids to geometrical center of points assigned to them

K-Means in action



1. Choose number of clusters, k (in this case, 3)

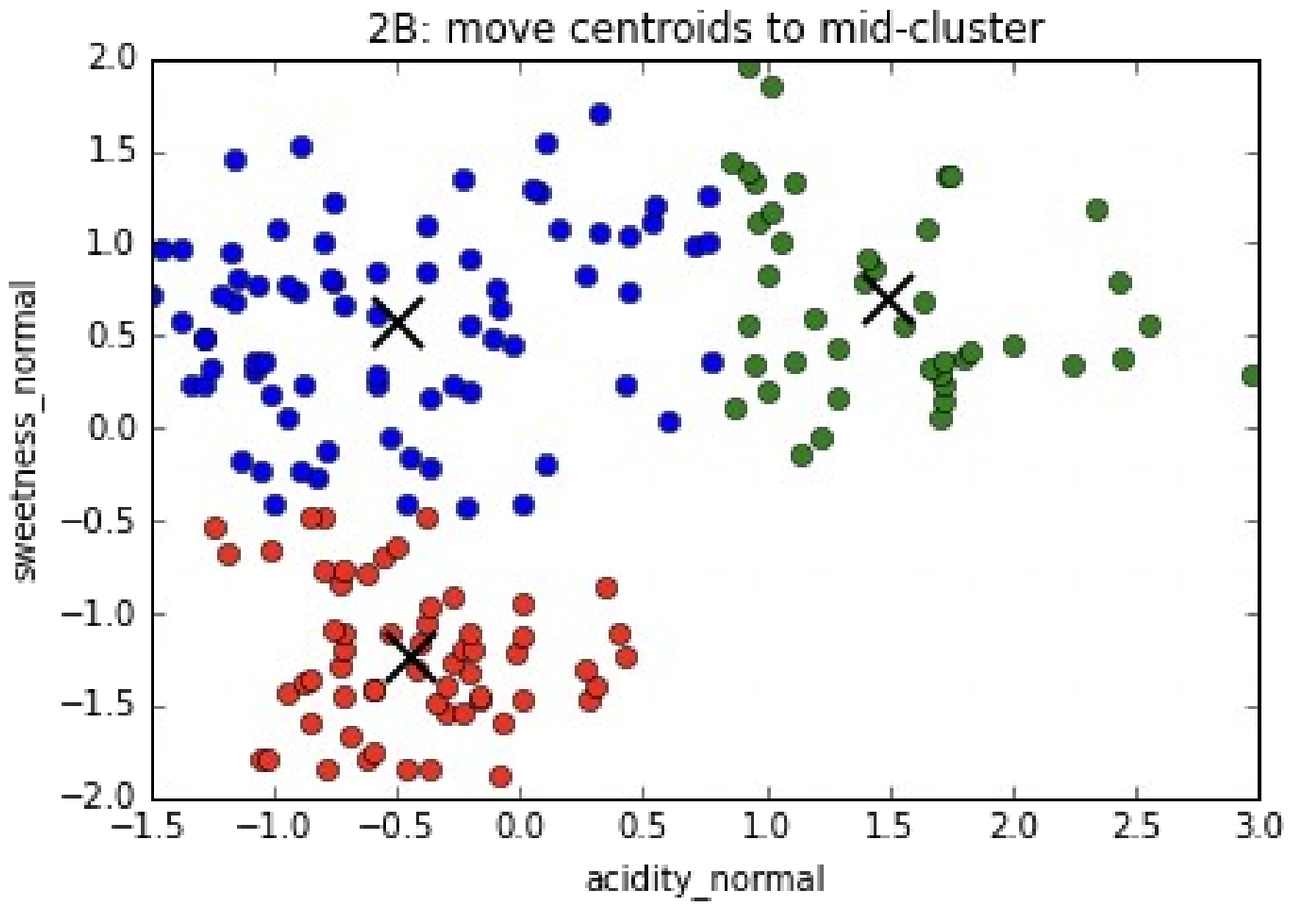
2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

4. Move centroids to geometrical center of points assigned to them

5. Repeat #3

K-Means in action



1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

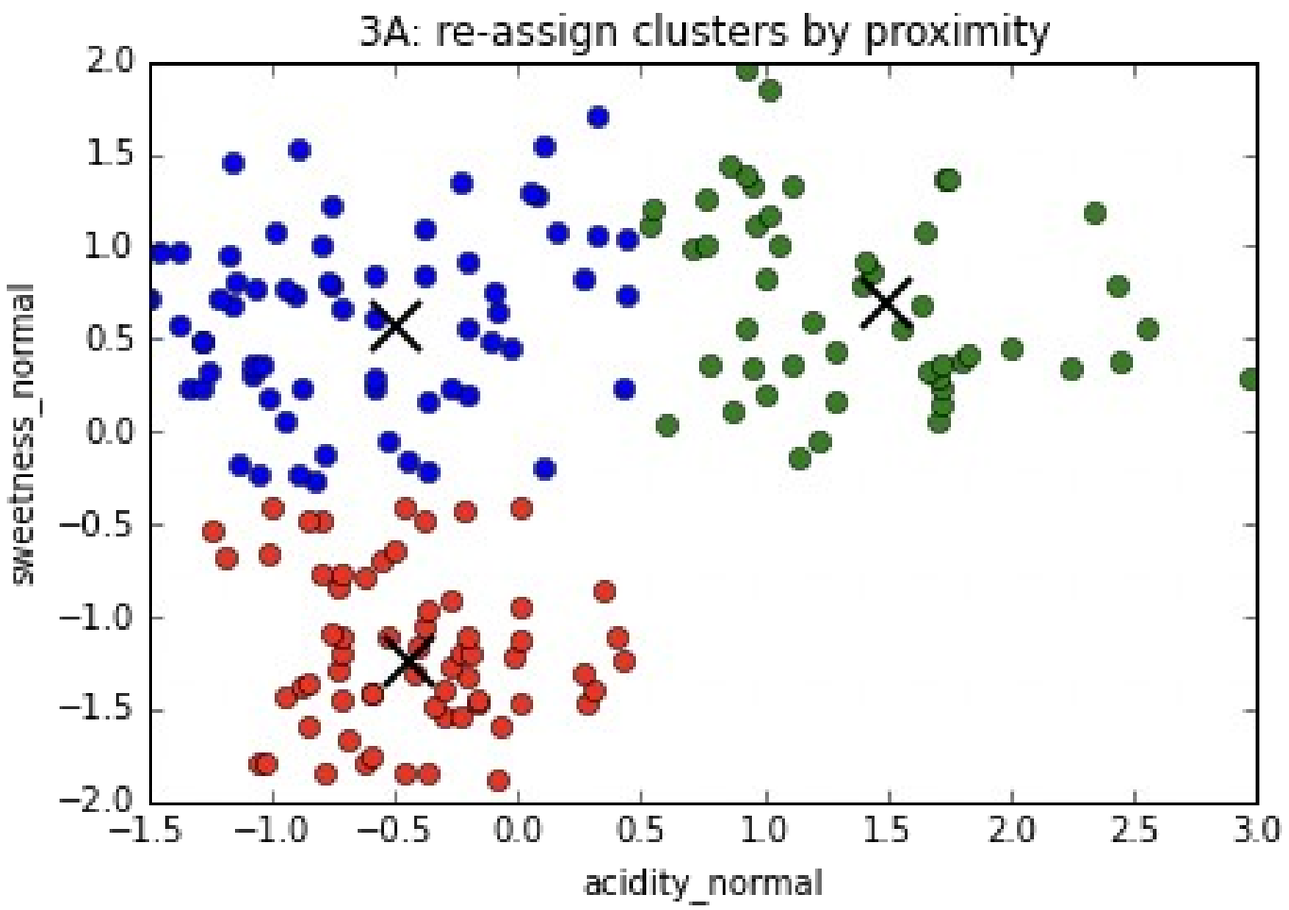
3. Assign every data point to a cluster based on proximity to centroid

4. Move centroids to geometrical center of points assigned to them

5. Repeat #3

6. Repeat #4

K-Means in action



1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

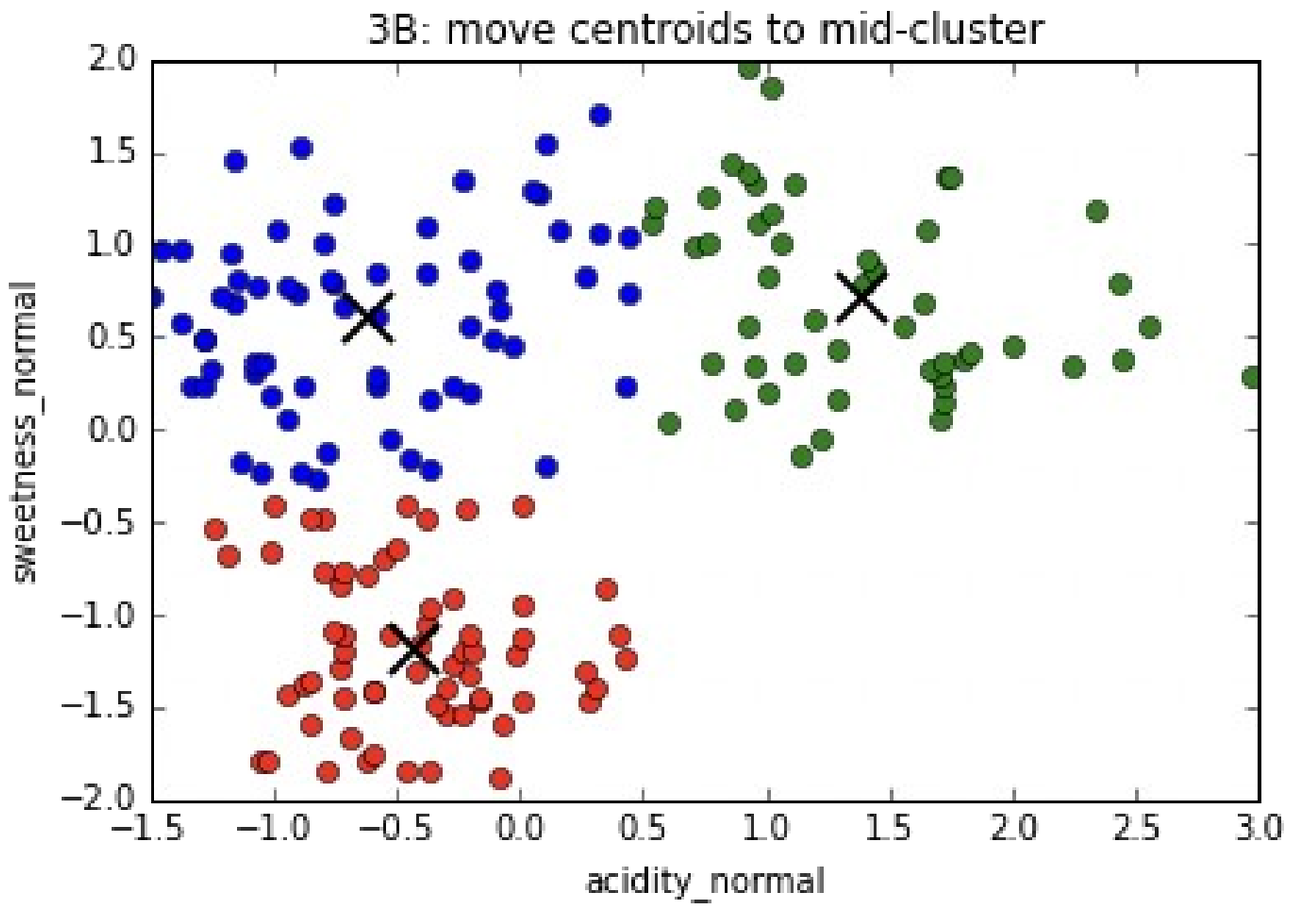
4. Move centroids to geometrical center of points assigned to them

5. Repeat #3

6. Repeat #4

Keep repeating ...

K-Means in action



1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

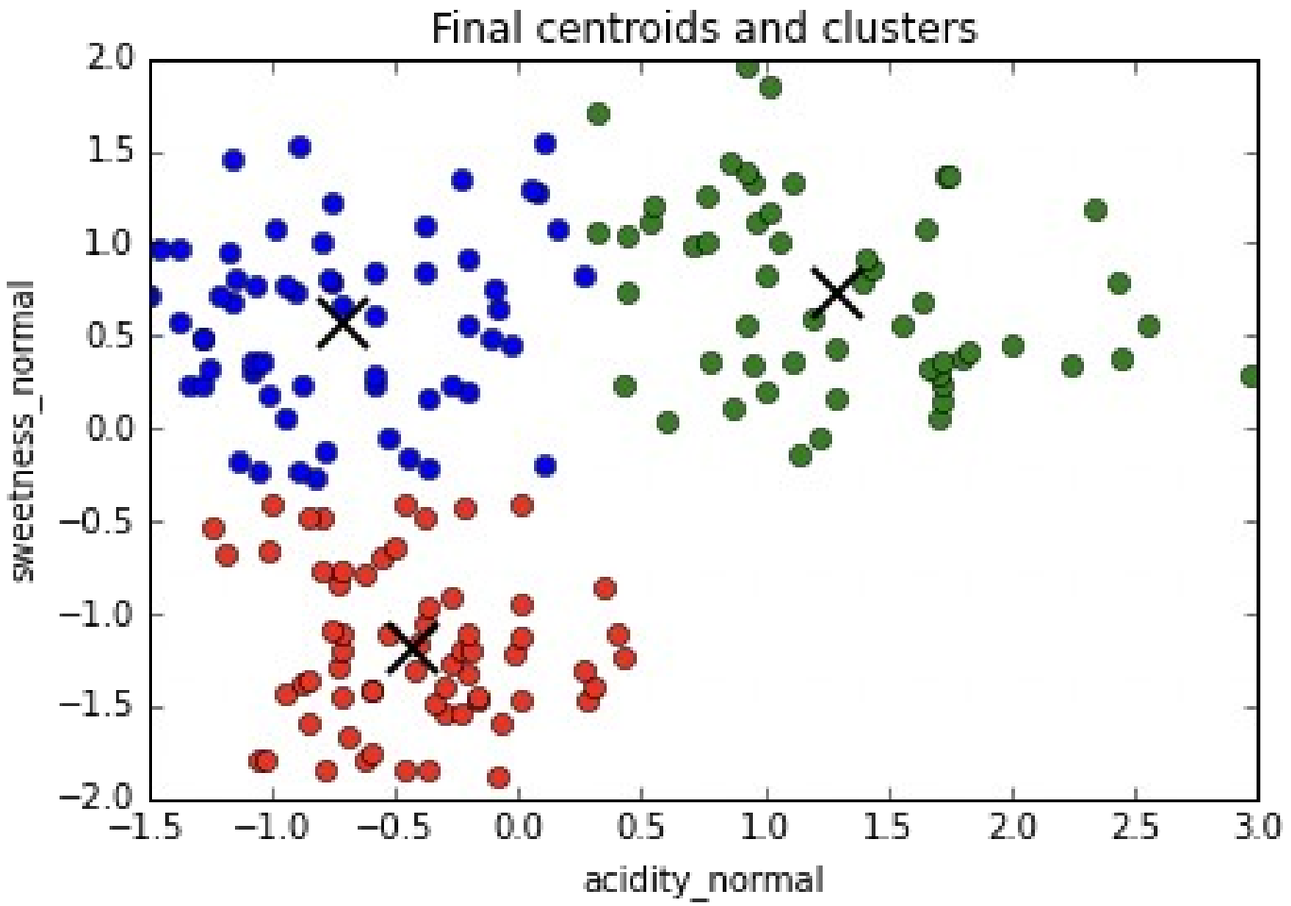
4. Move centroids to geometrical center of points assigned to them

5. Repeat #3

6. Repeat #4

Keep repeating ...

K-Means in action



1. Choose number of clusters, k (in this case, 3)

2. Assign (usually randomly) starting position for centroid of each cluster

3. Assign every data point to a cluster based on proximity to centroid

4. Move centroids to geometrical center of points assigned to them

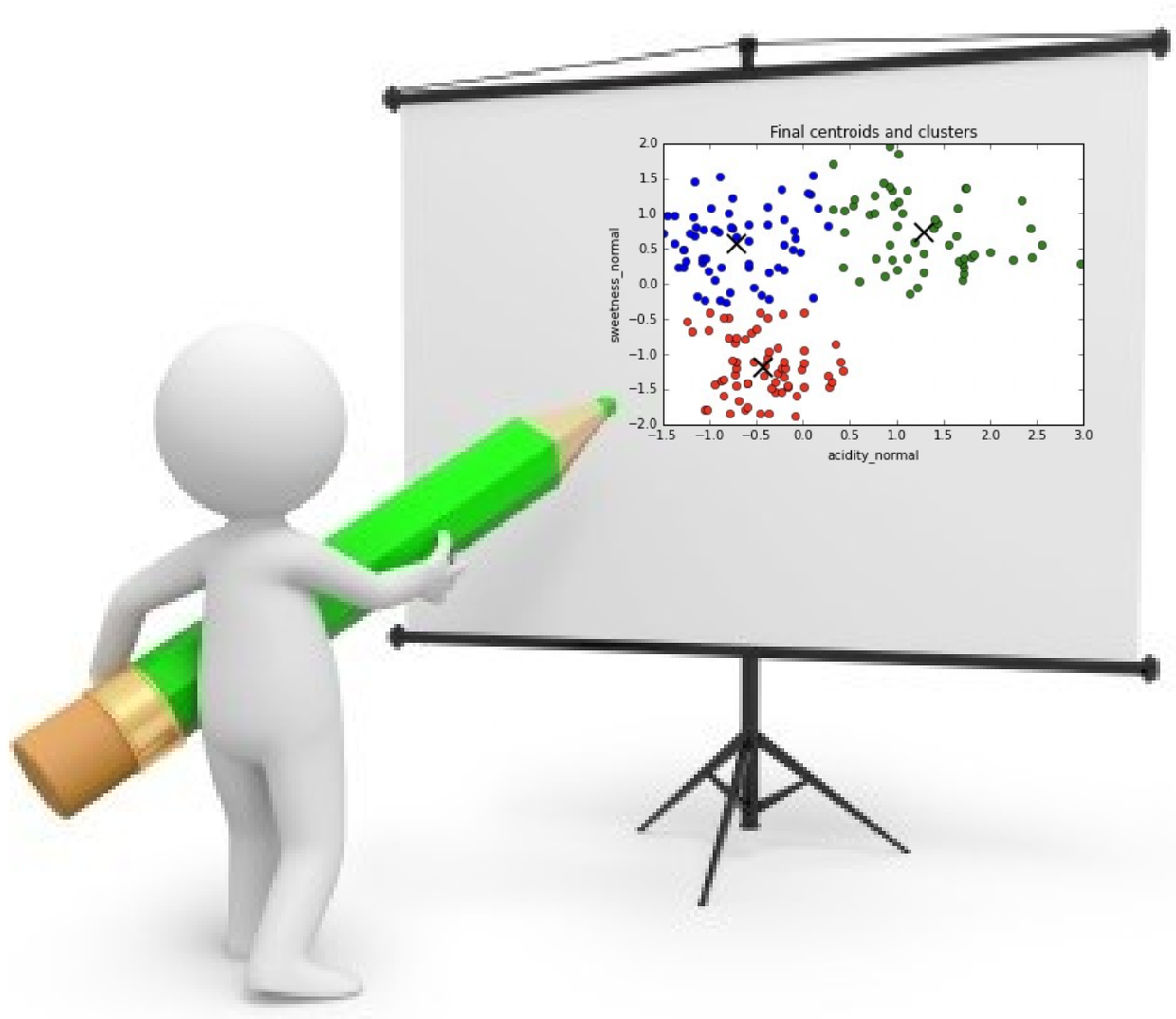
5. Repeat #3

6. Repeat #4

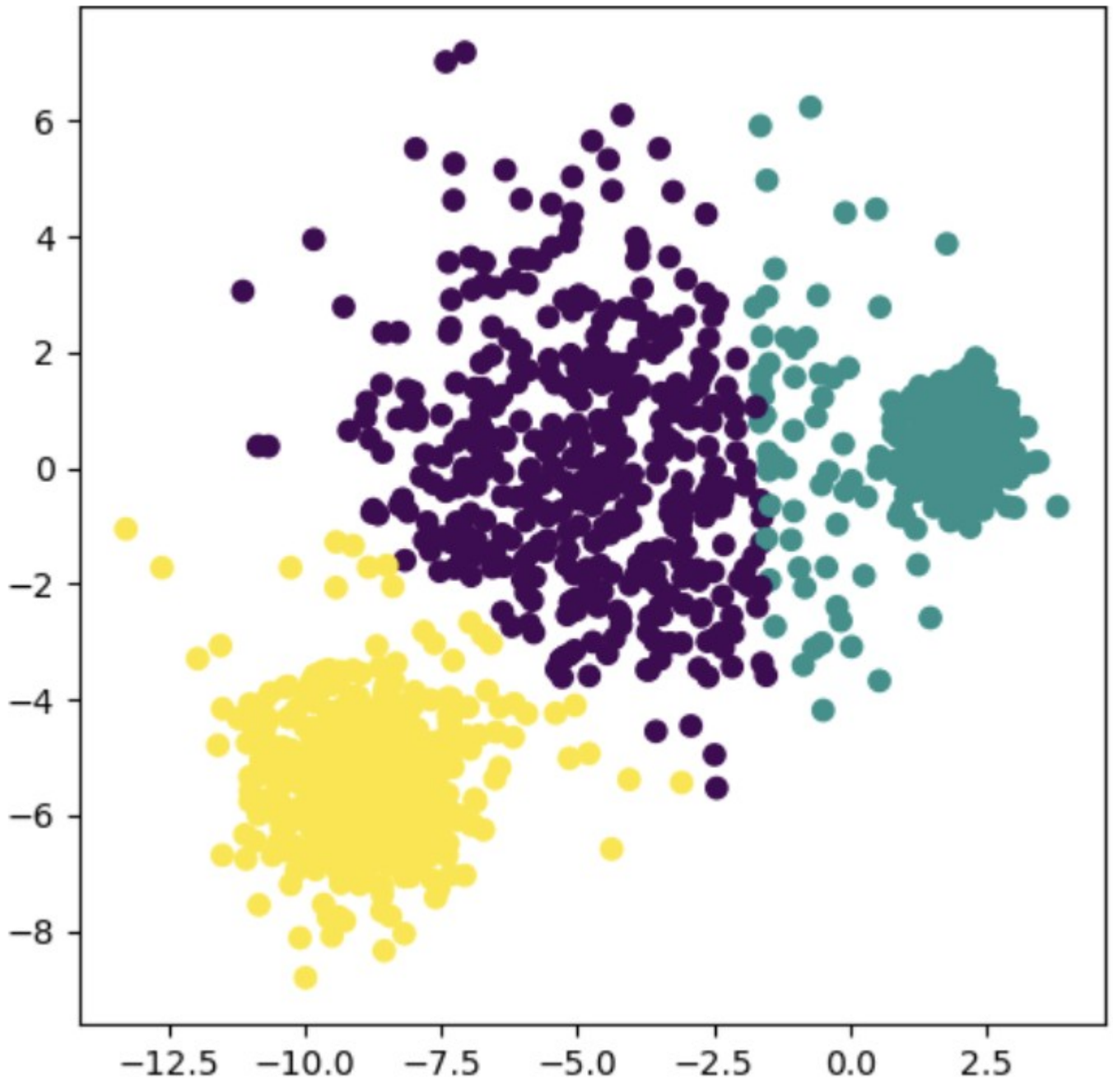
Keep repeating ...

... until convergence, i.e. clusters centers do not move from one repetition to the next

Demo ...


Final centroids and clusters
sweetness_normal
acidity_normal
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0

K-Means holy grail?

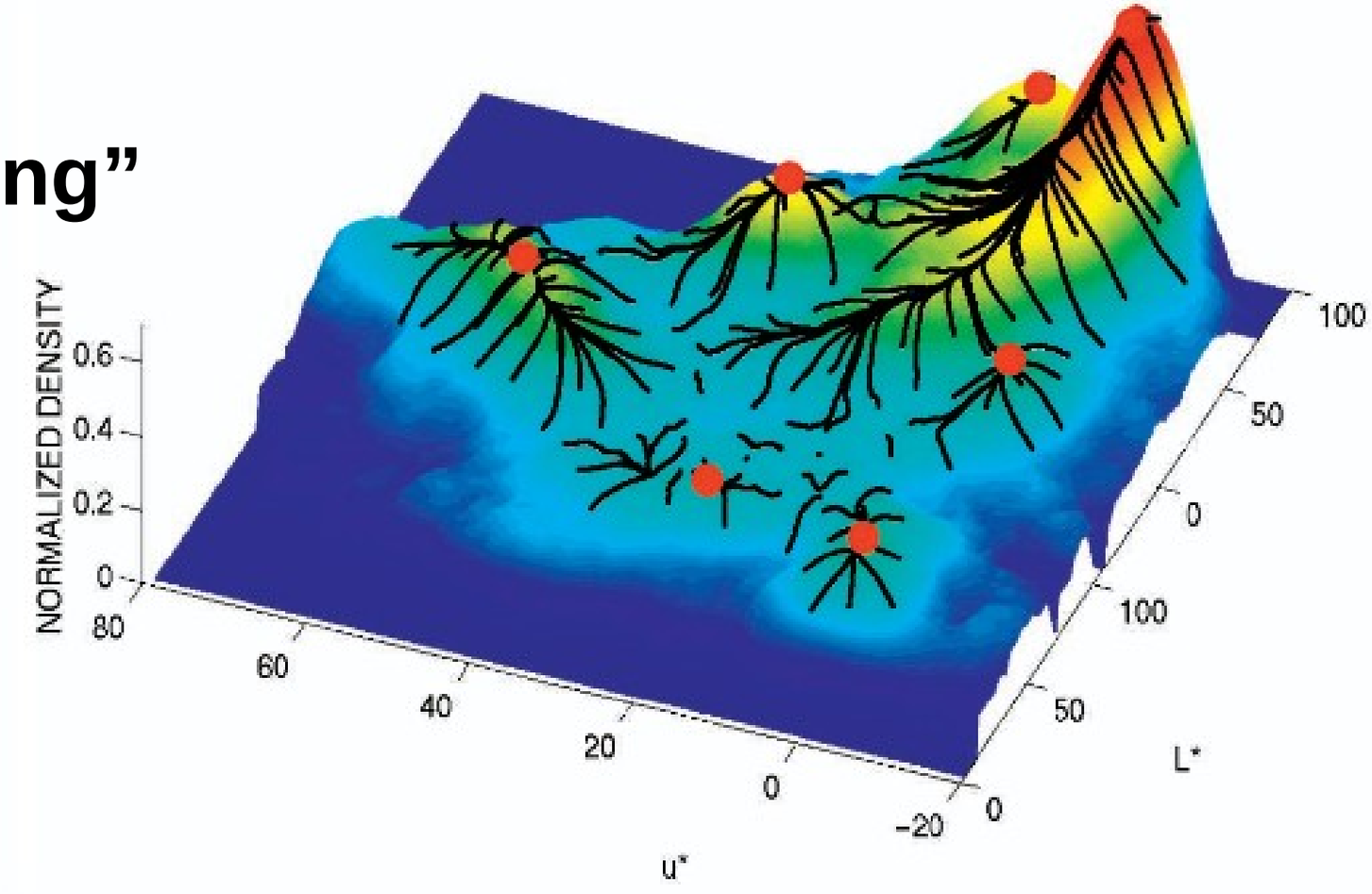


Nope: “use the *right* tool for the *right* job”

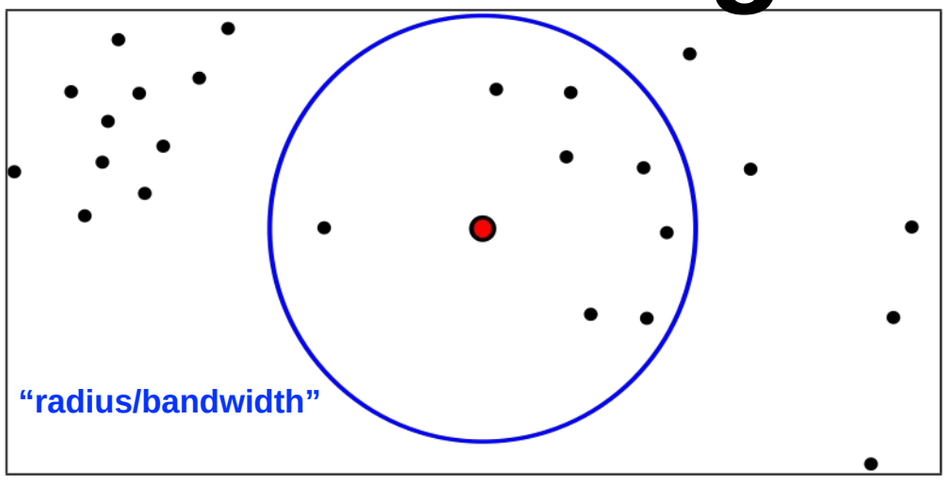
MeanShift

Locating maxima (*modes*) in density

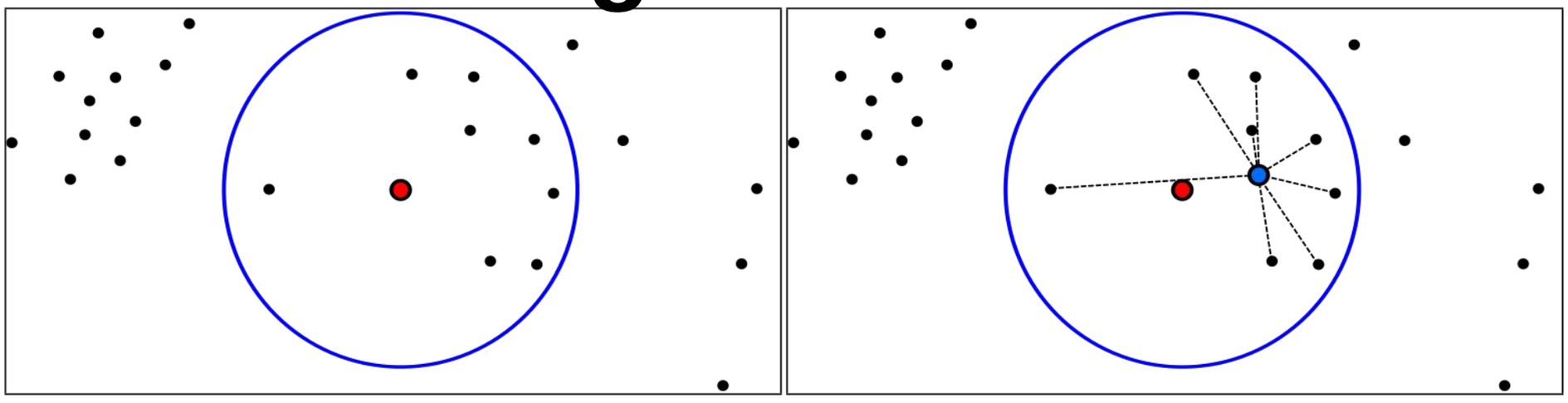
“Hill-climbing”



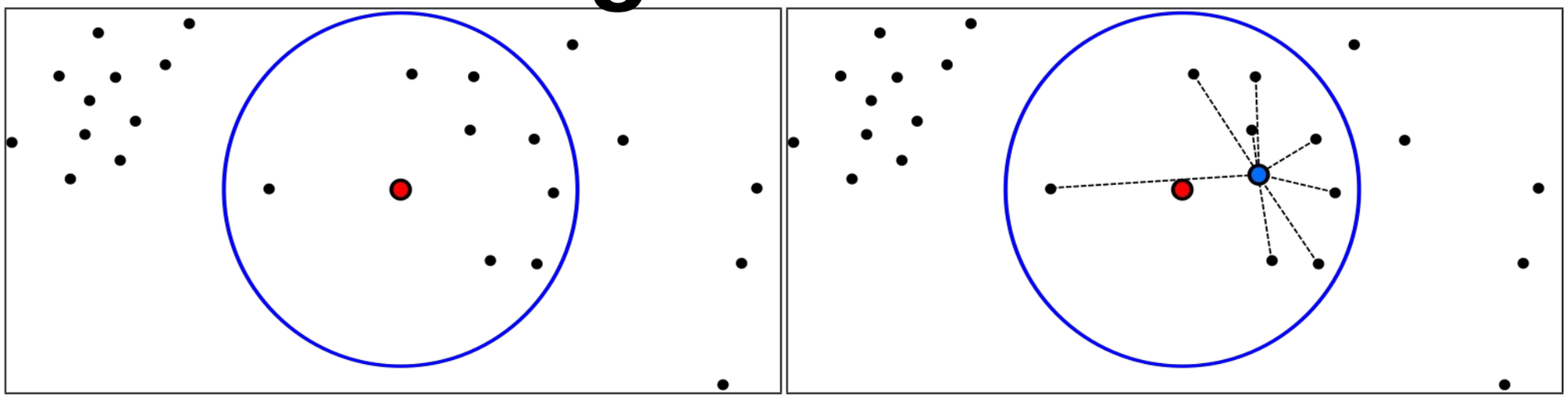
“Hill-climbing” MeanShift Method



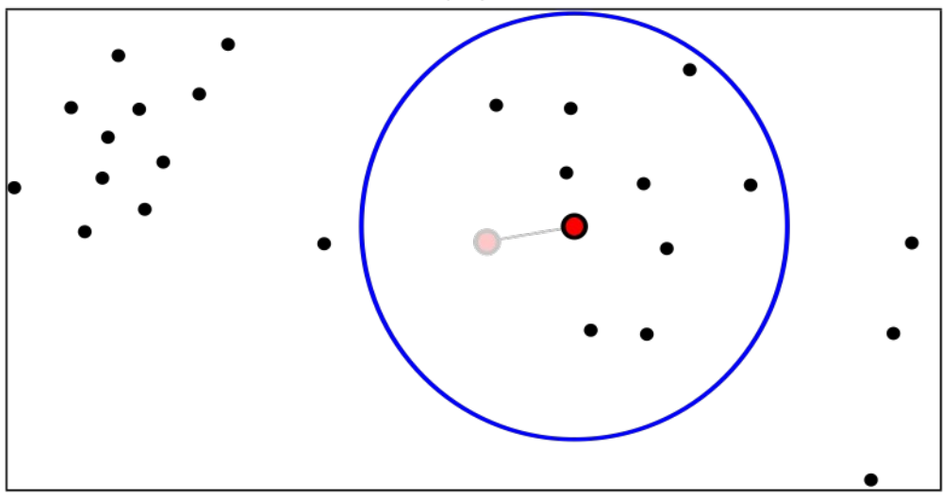
“Hill-climbing” MeanShift Method



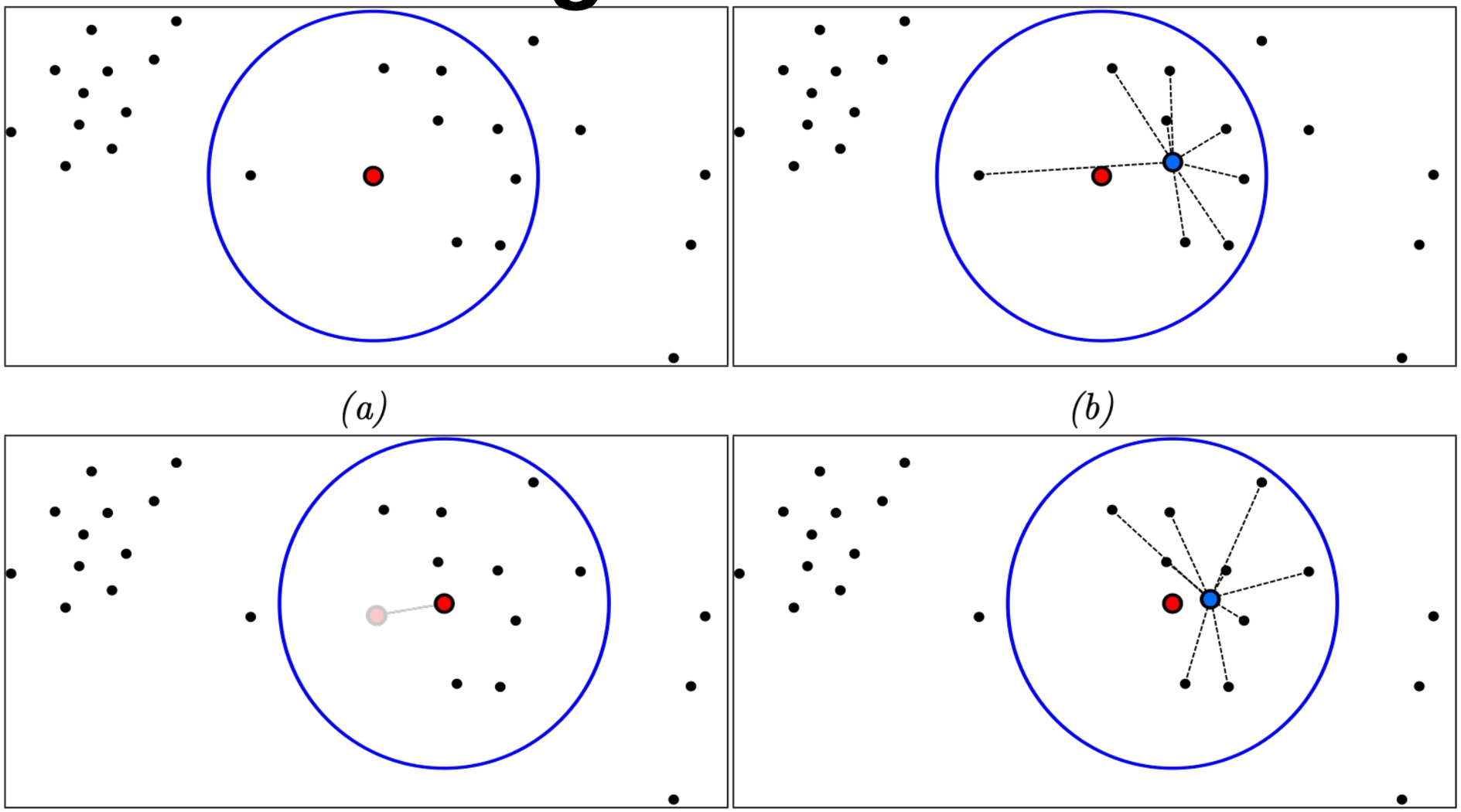
“Hill-climbing” MeanShift Method



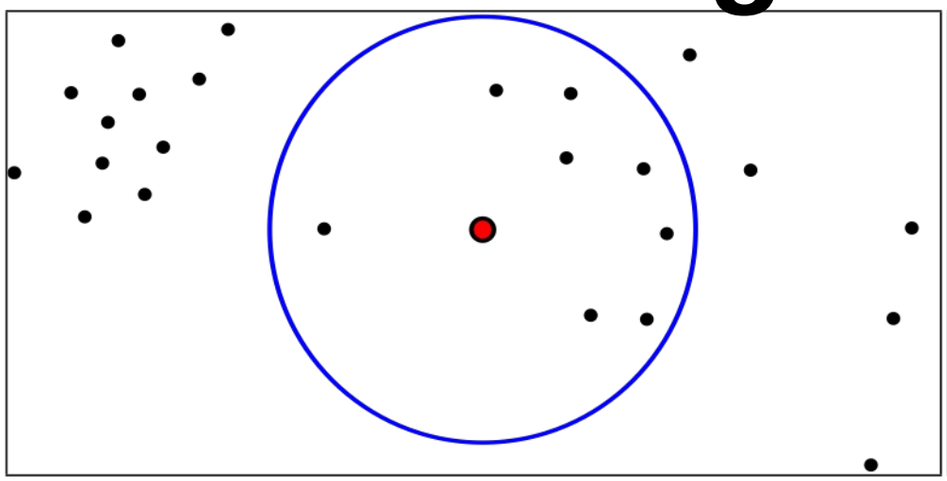
(a)



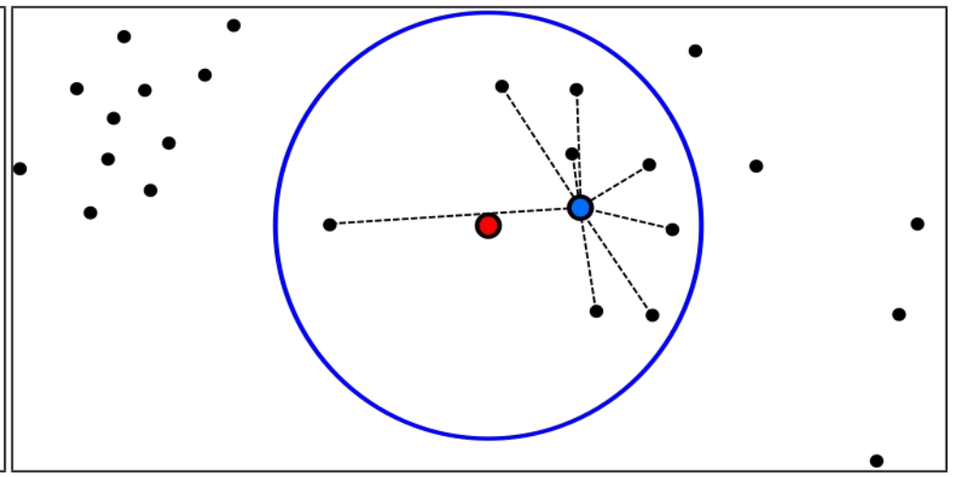
“Hill-climbing” MeanShift Method



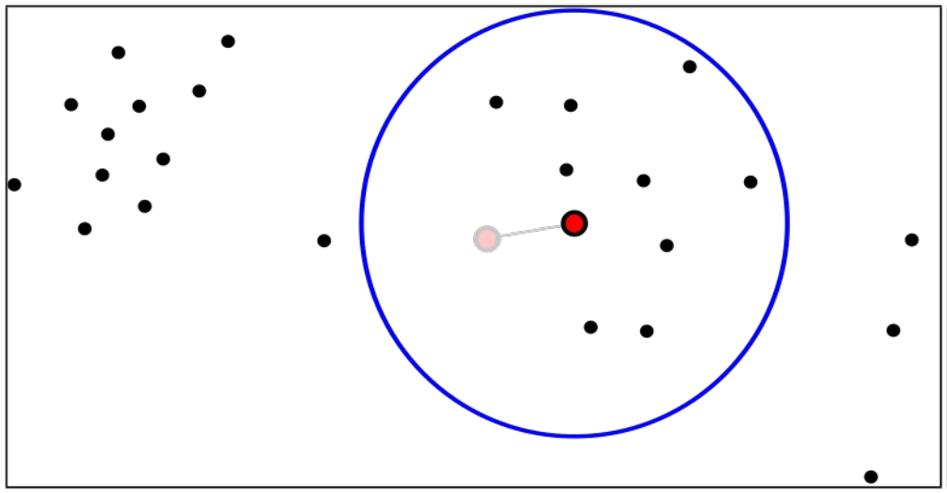
“Hill-climbing” MeanShift Method



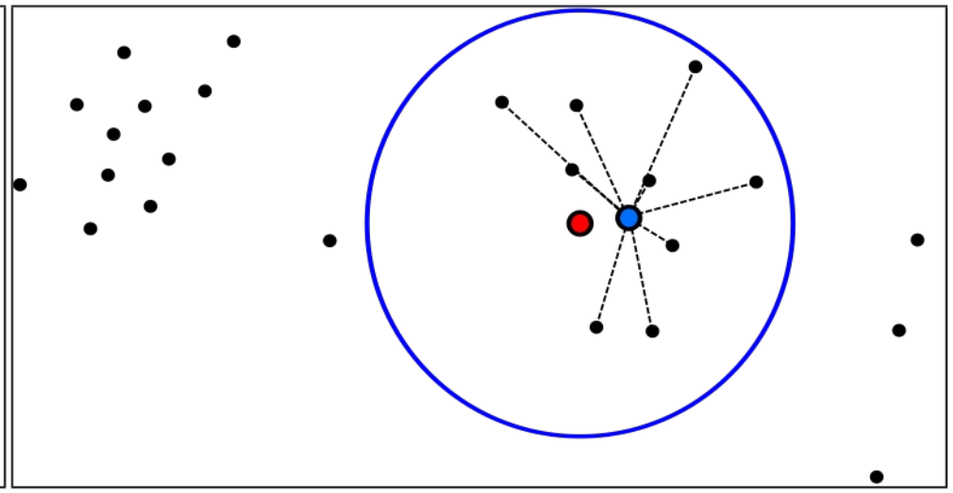
(a)

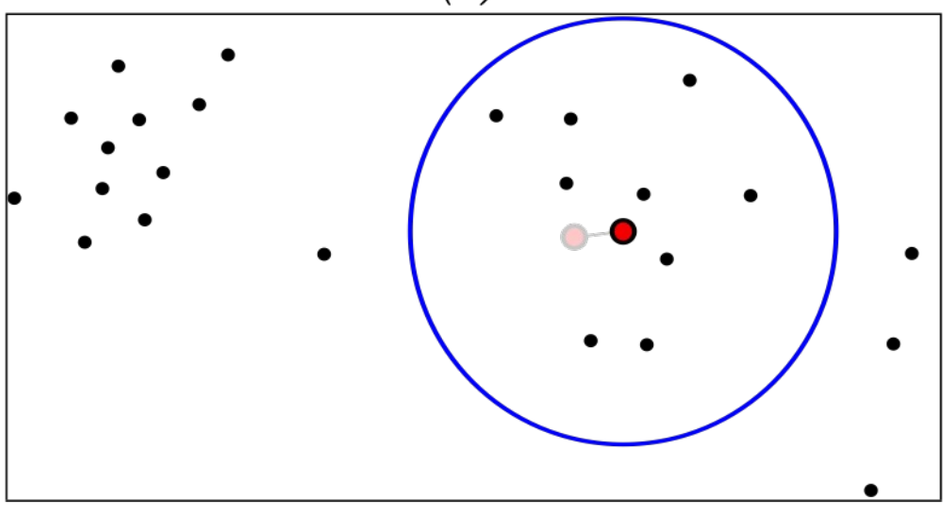


(b)



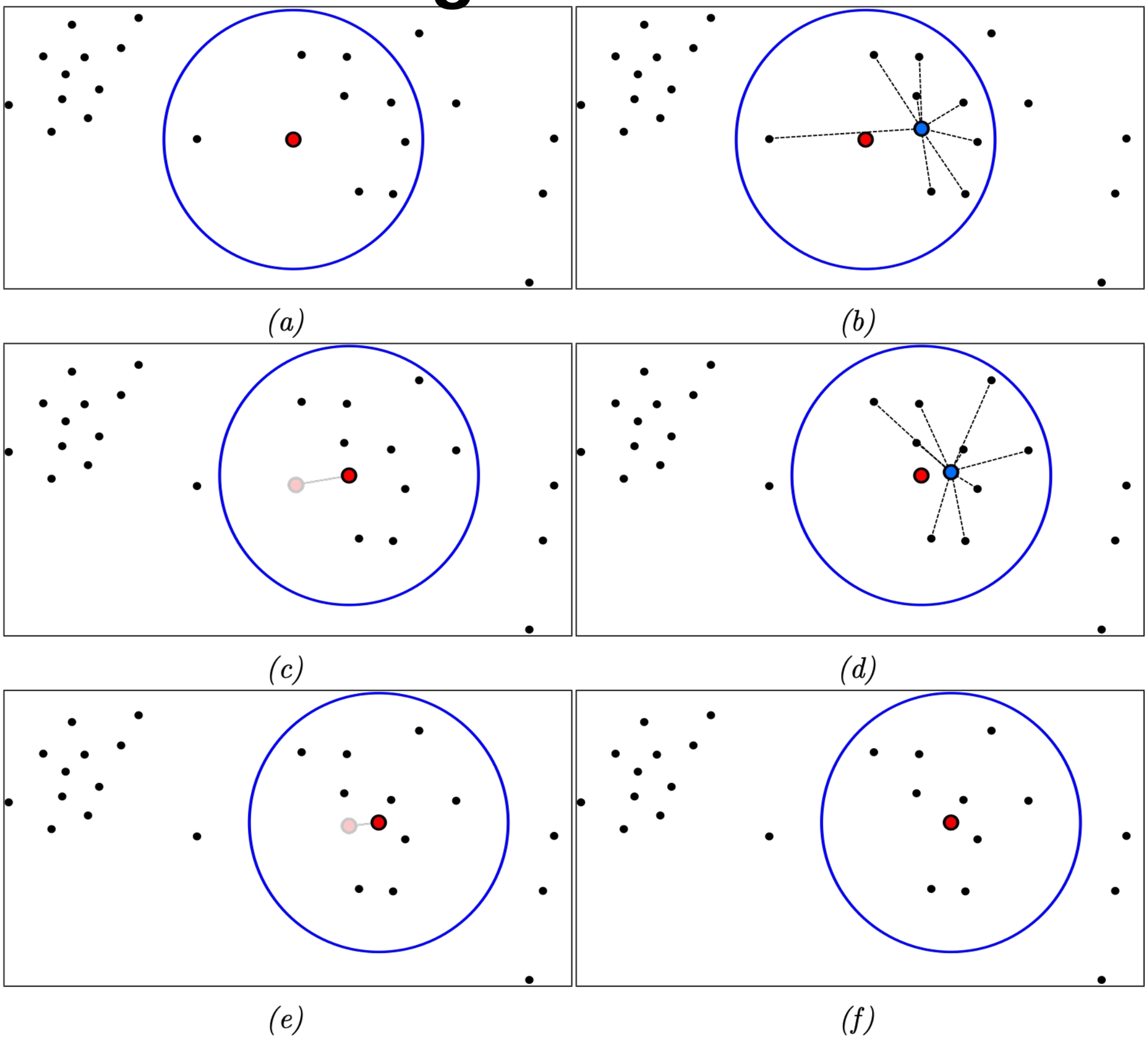
(c)



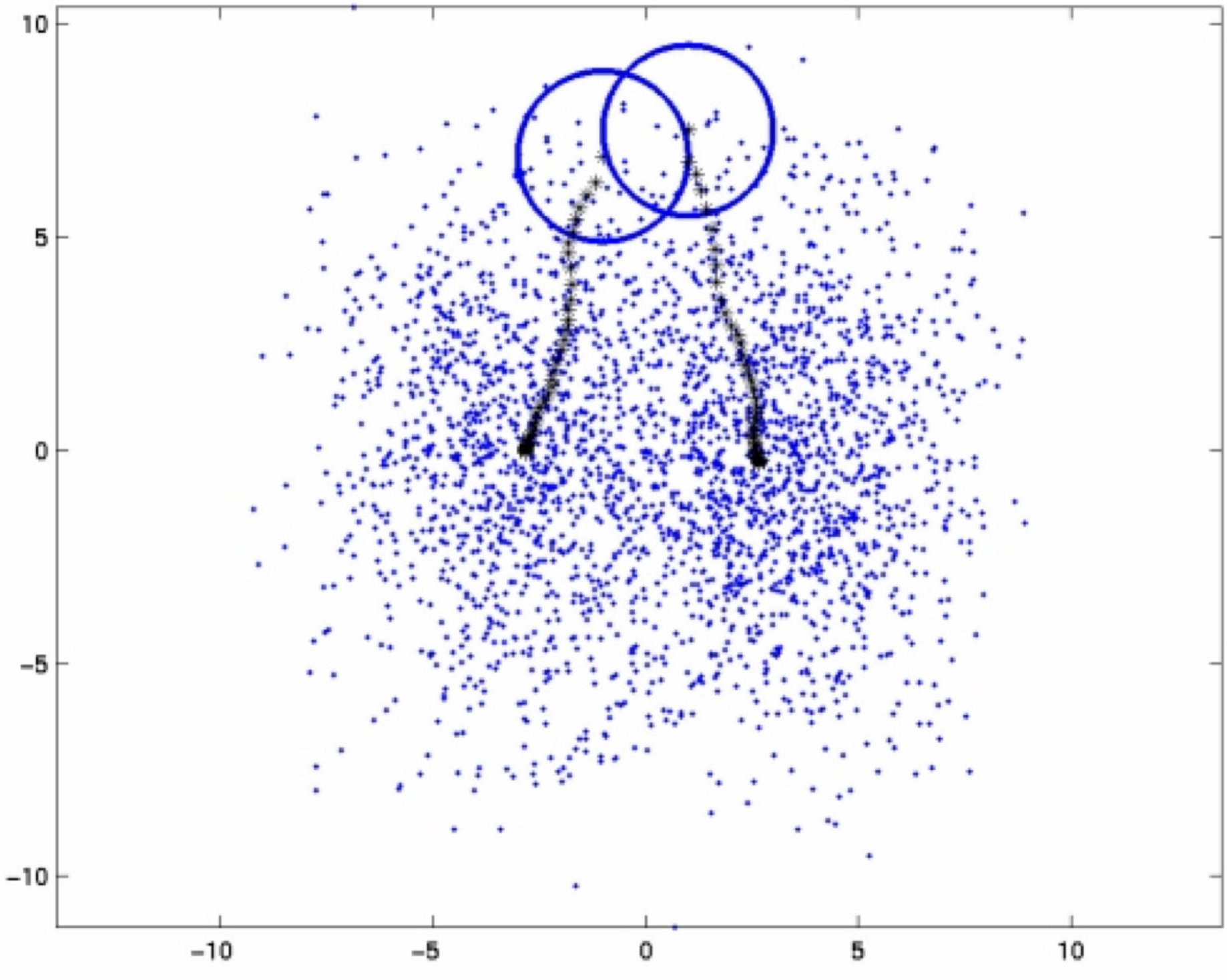


(e)

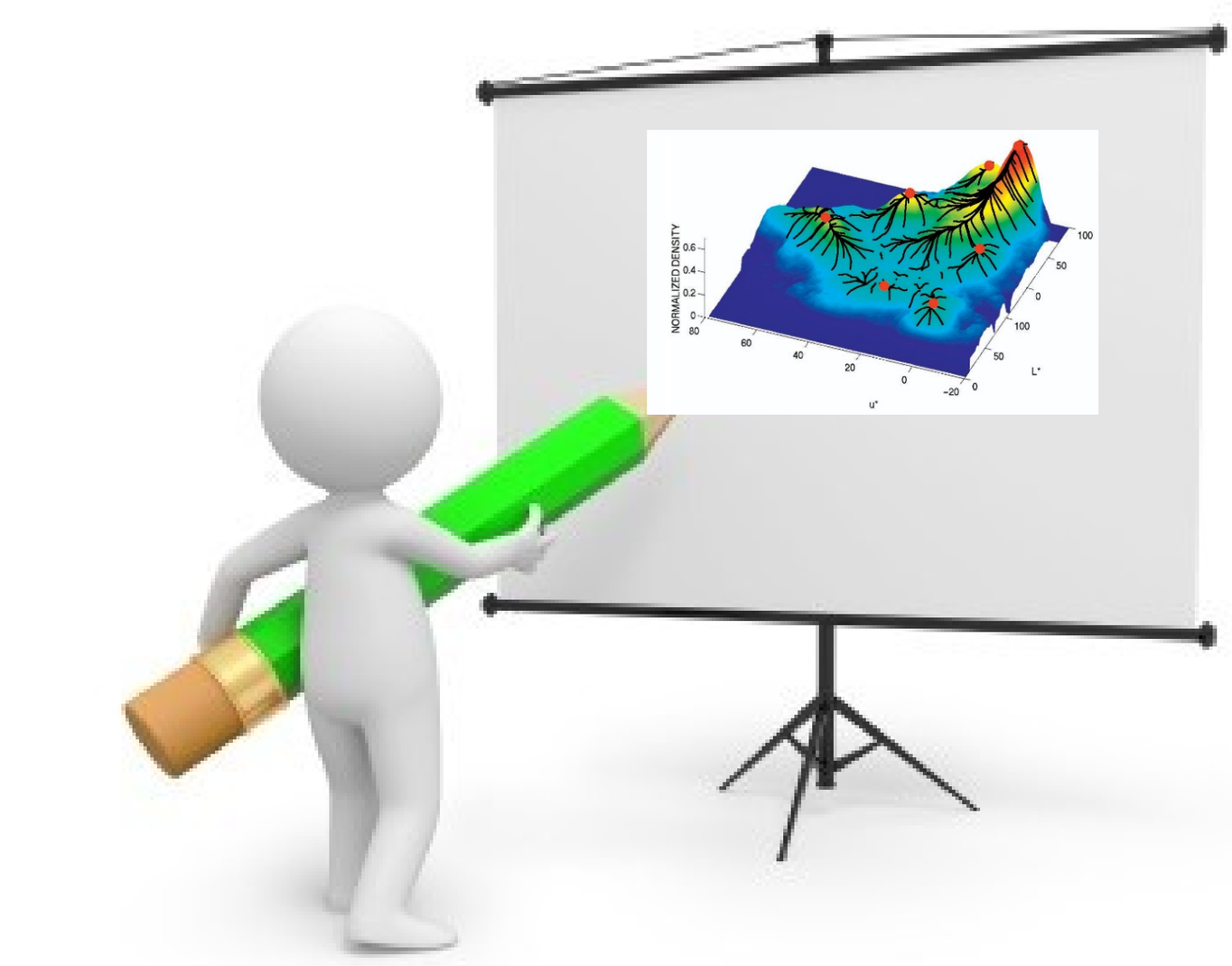
“Hill-climbing” MeanShift Method



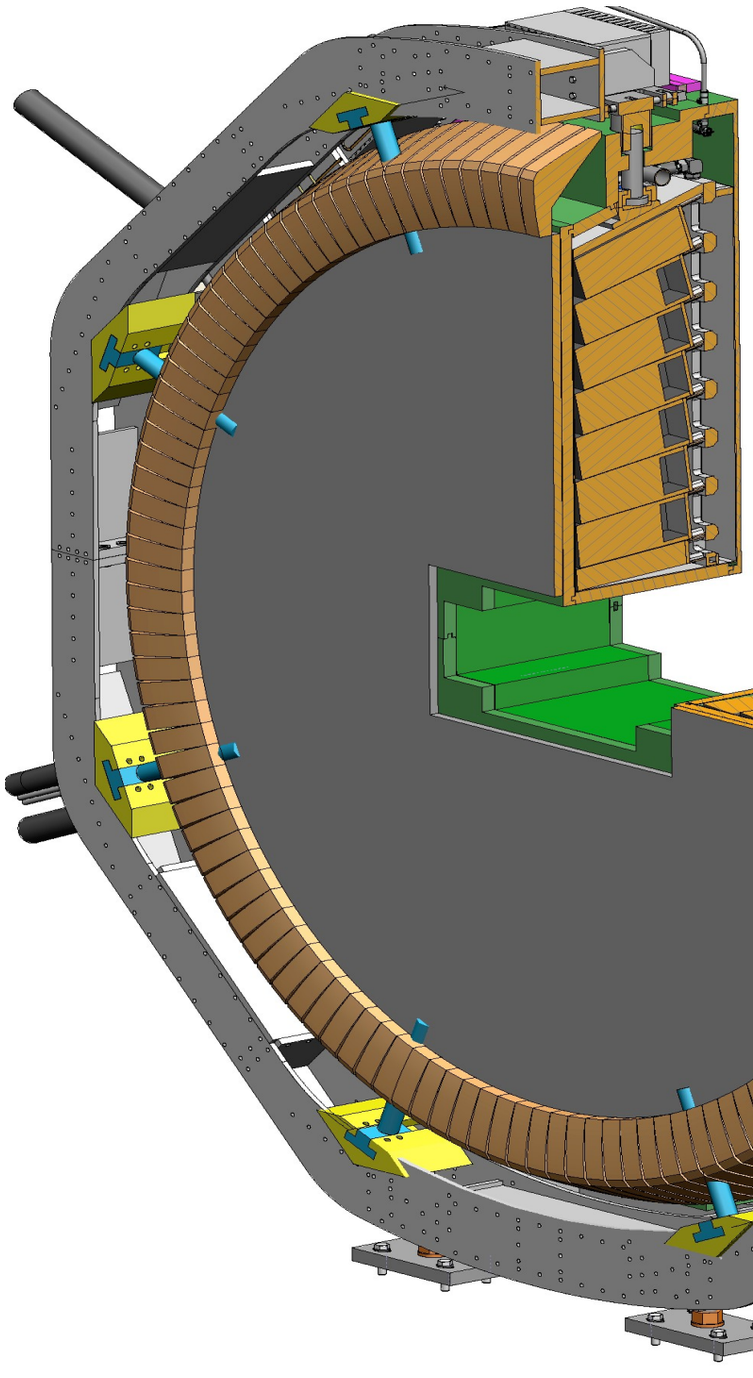
“Hill-climbing” MeanShift Method



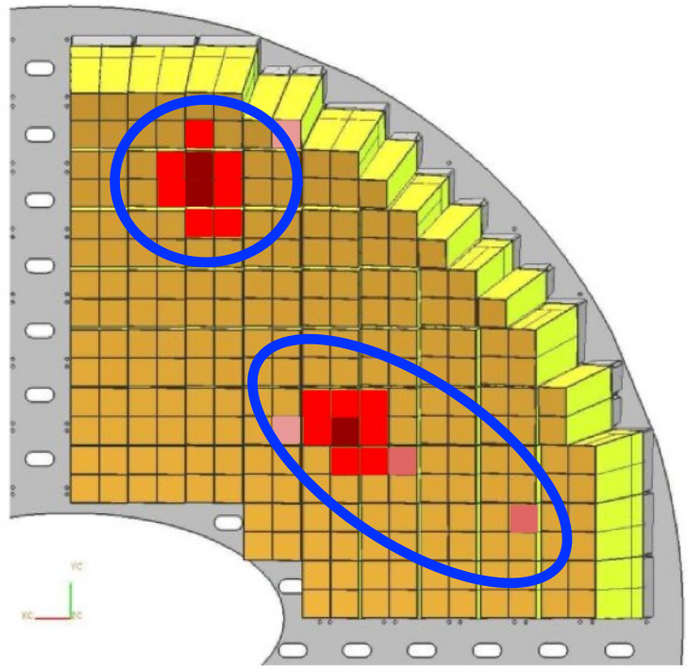
Demo…

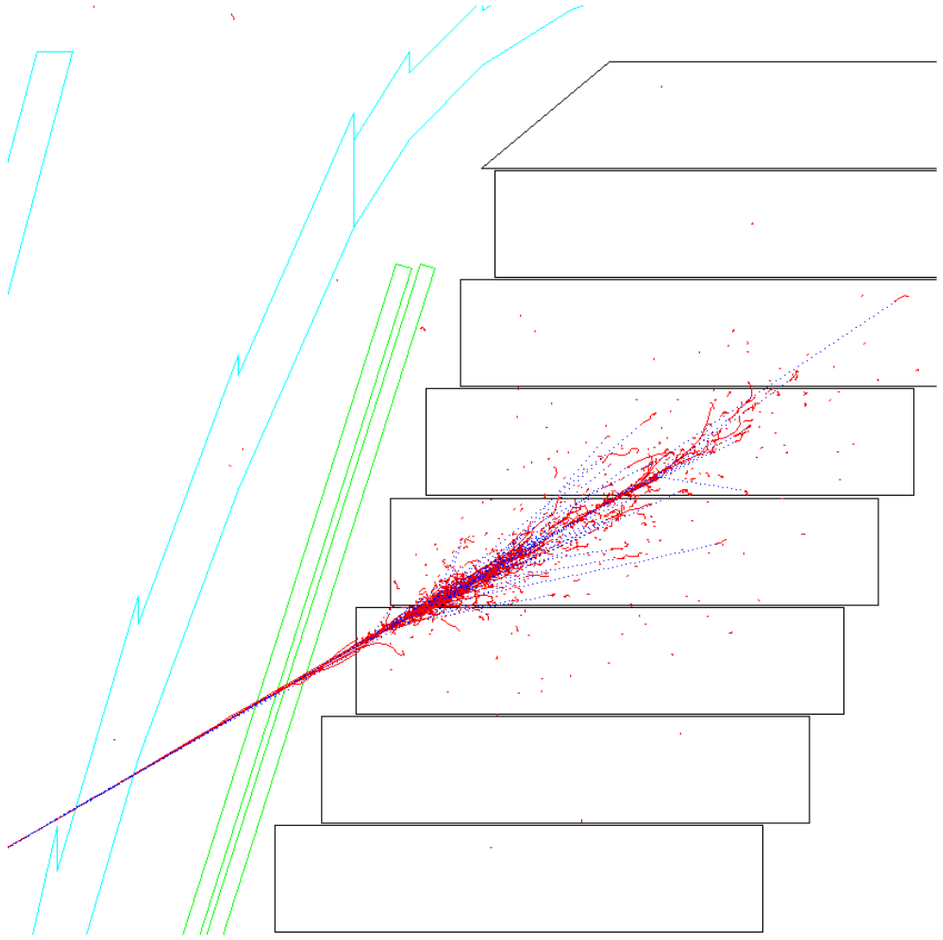


Clustering - example from my own research…

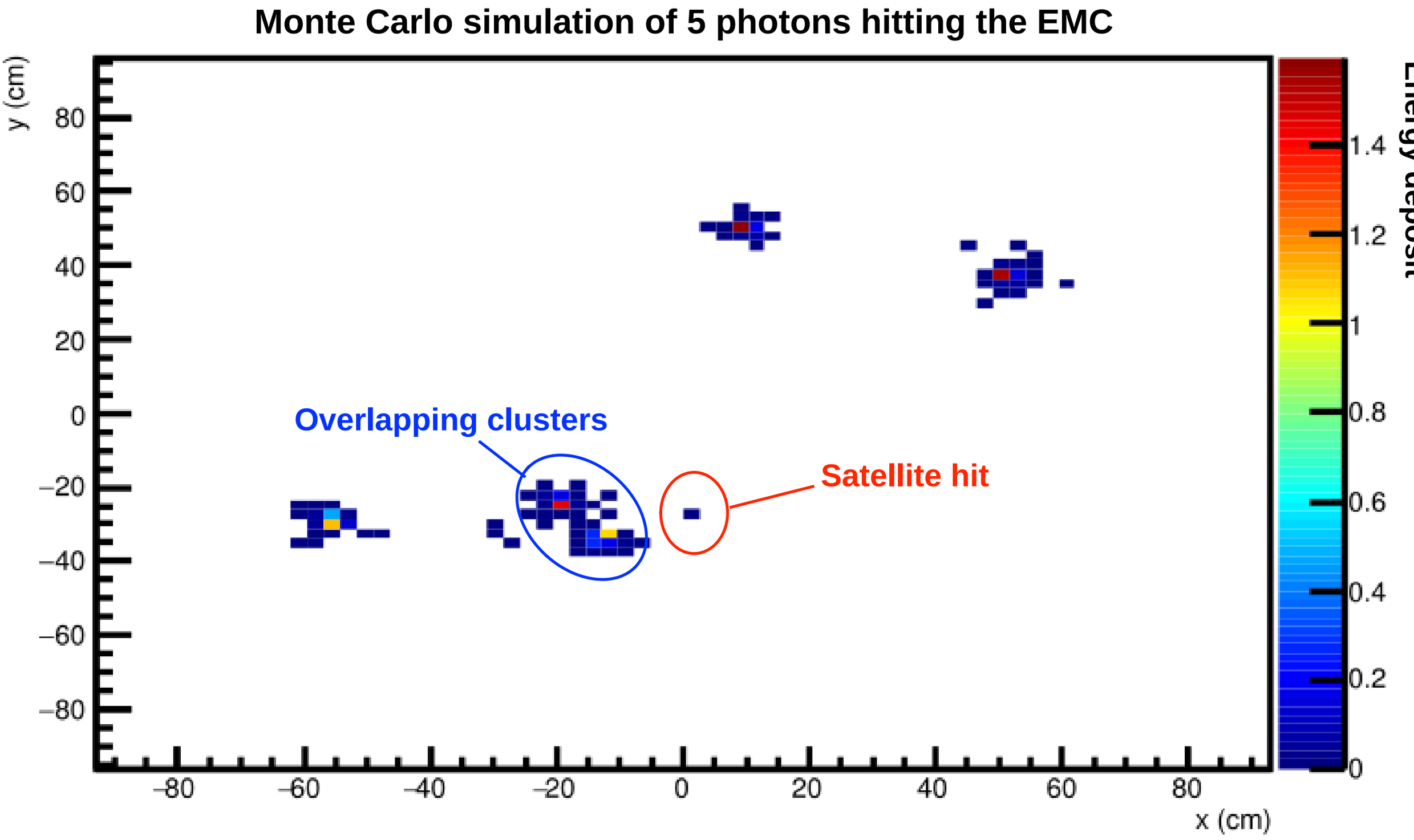


Electromagnetic Calorimeter
“Forward Endcap”

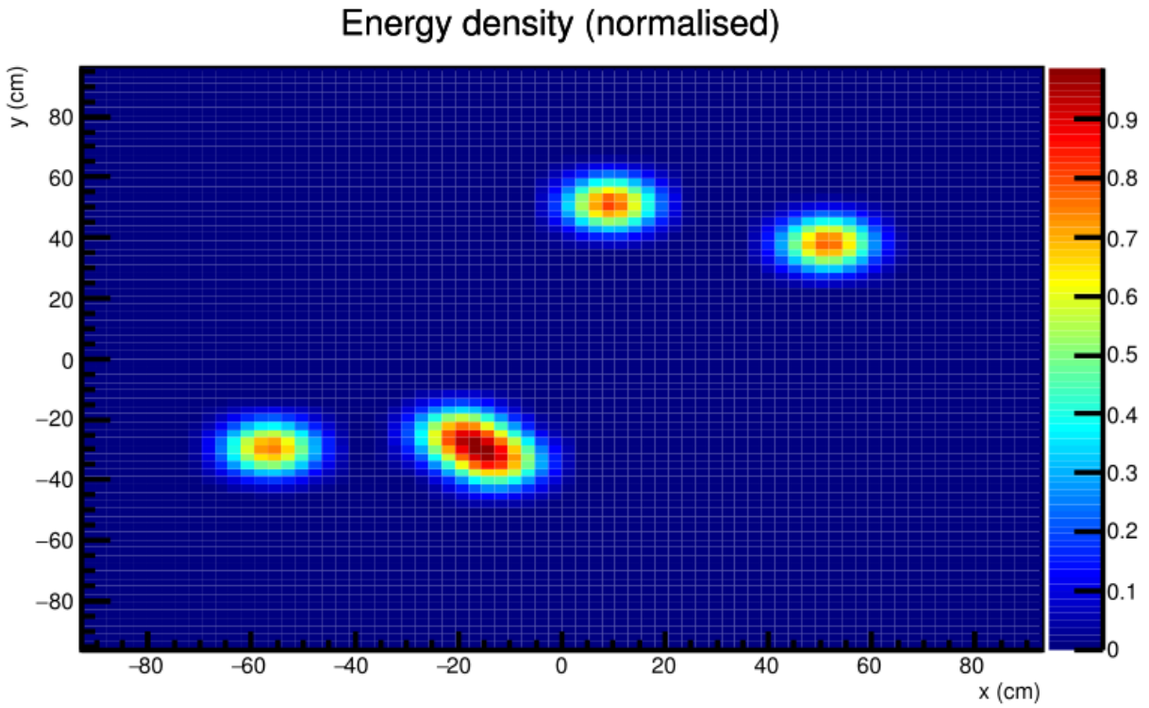


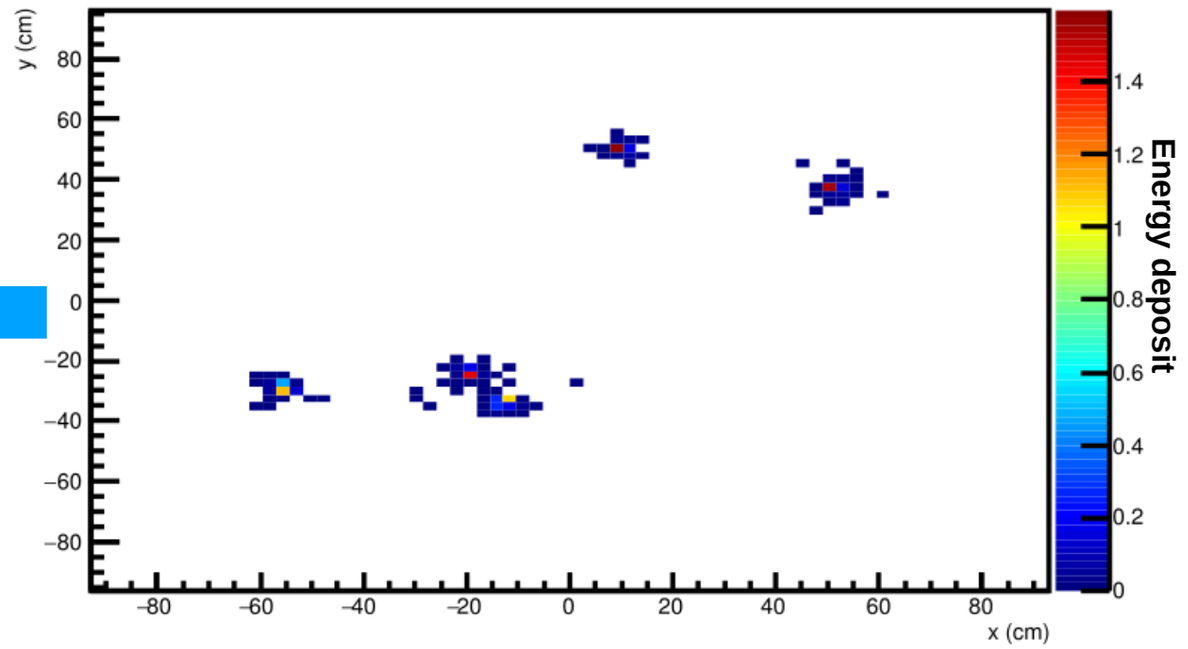


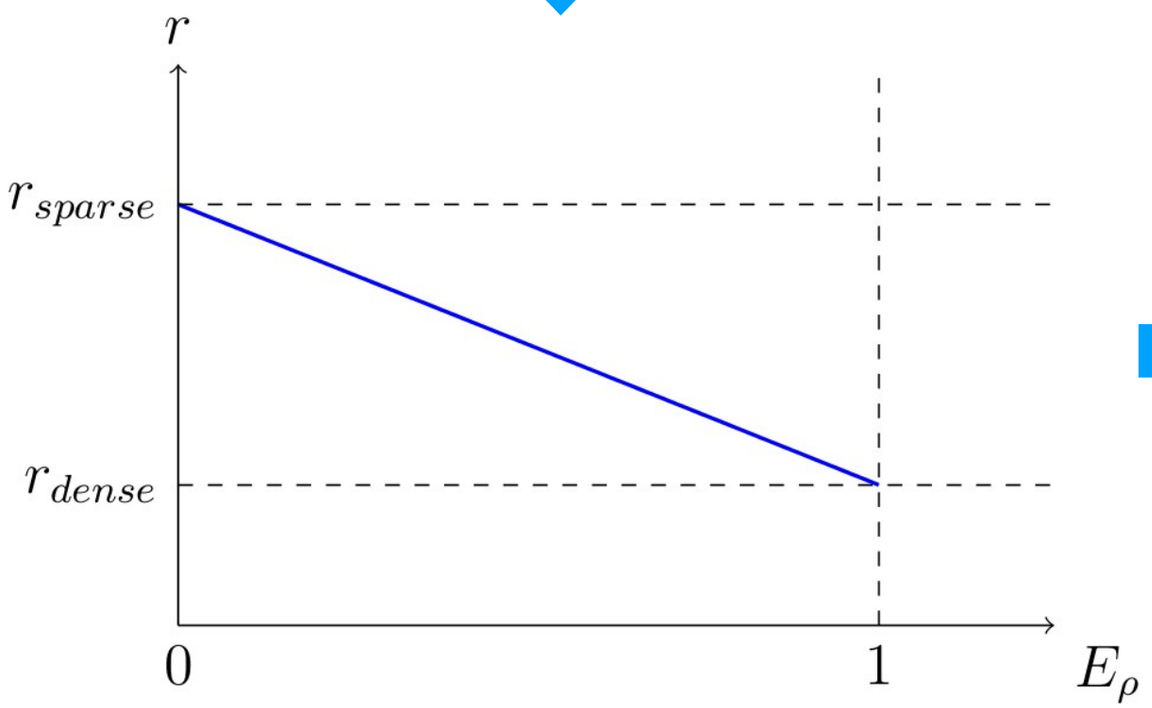
Clustering - example from my own research…



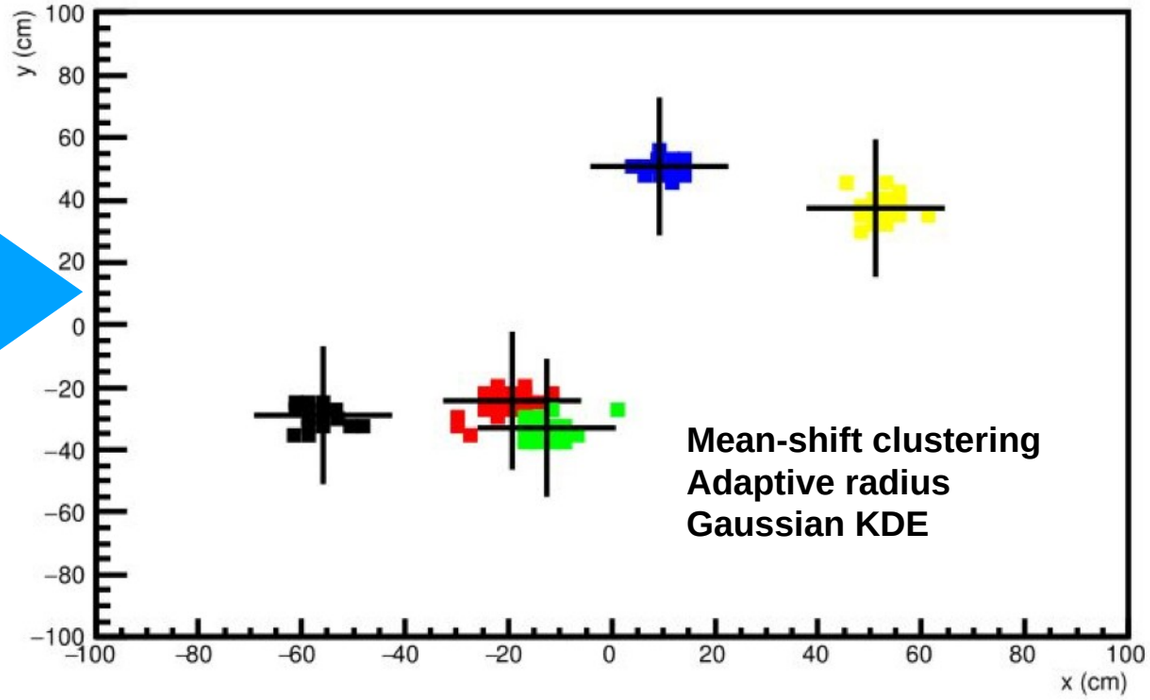
Clustering - example from my own research…







Han Kruiger, Michael Wilkinson, Mohammad Babai, JM

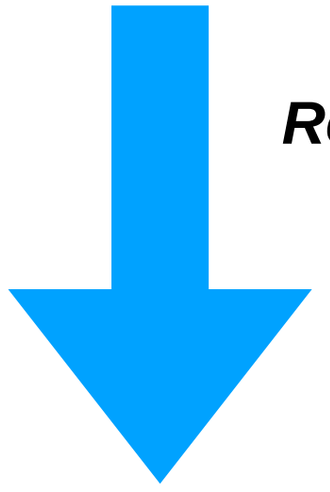


How to determine the “best” value for

K (KMeans)?

bandwidth (MeanShift)?

… (whatever cluster algorithm)?

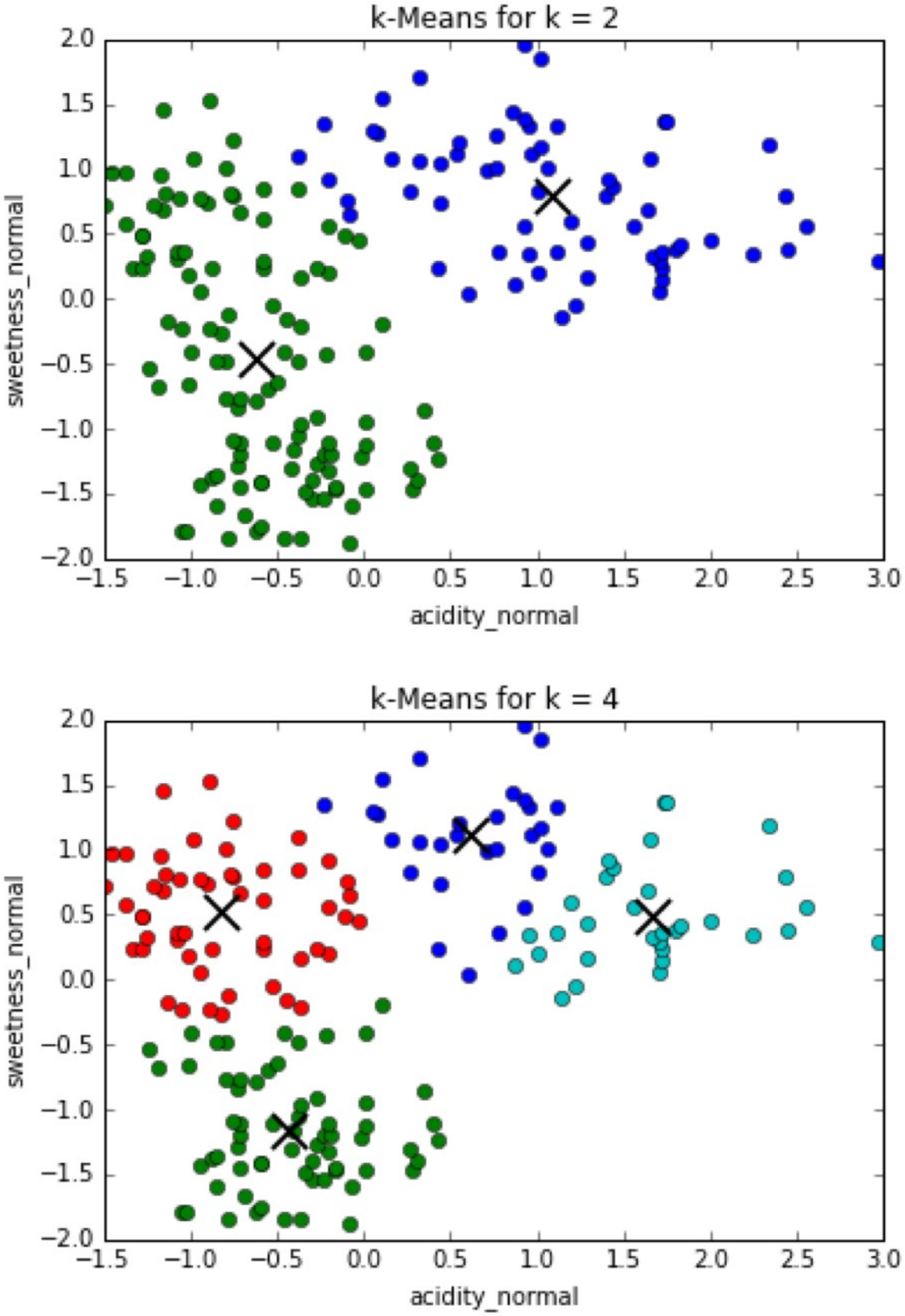


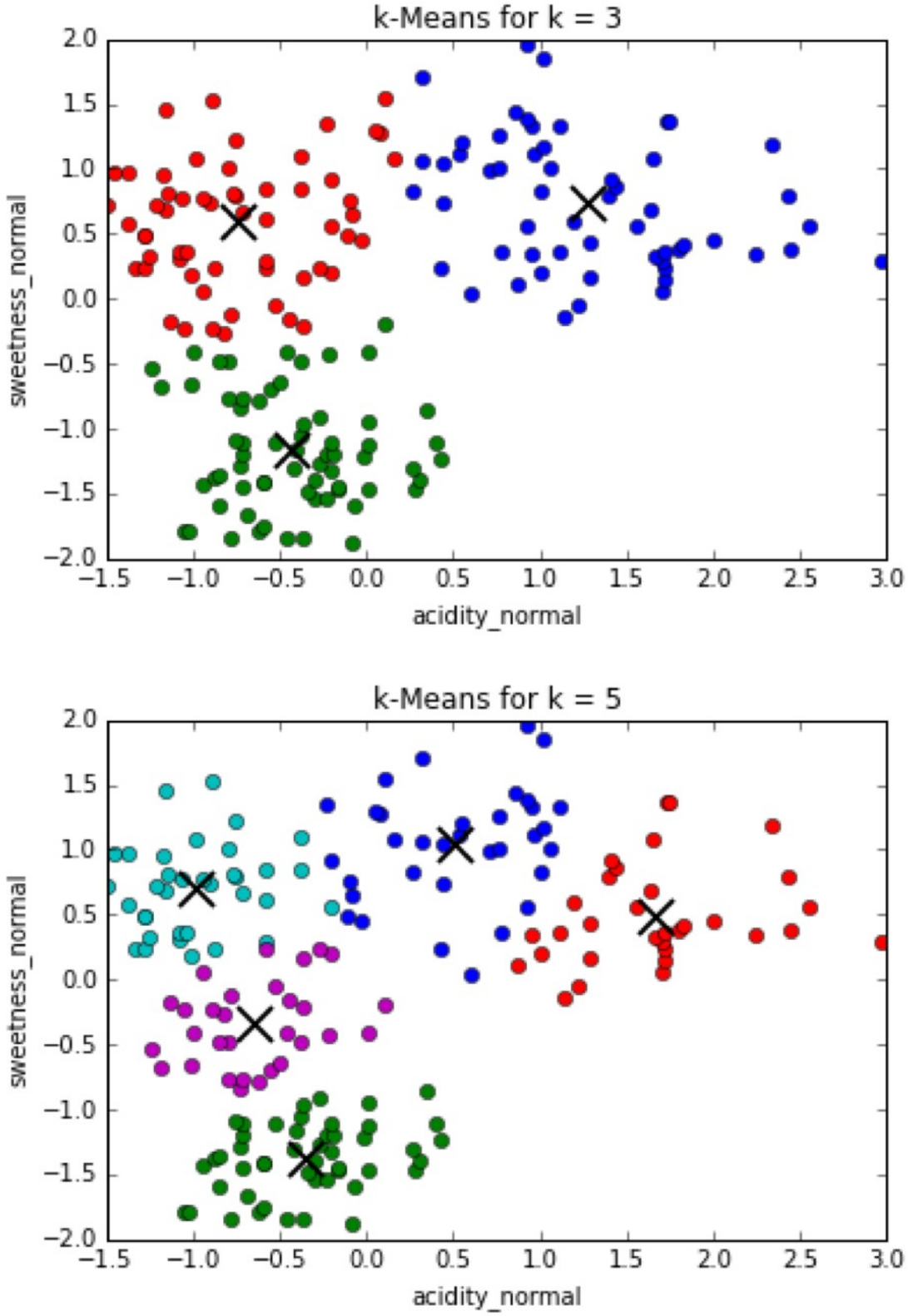
***Remember*: there is no free lunch in the world of ML!**

Silhouette coefficient

(Peter Rousseeuw, 1987)

Comparing different values of k





How to determine k, bandwidth, …?

Lowest average distance of *i* to data of closest <u>other</u> cluster (neighboring cluster)

Average distance of datum *i* with all data of <u>same</u> cluster

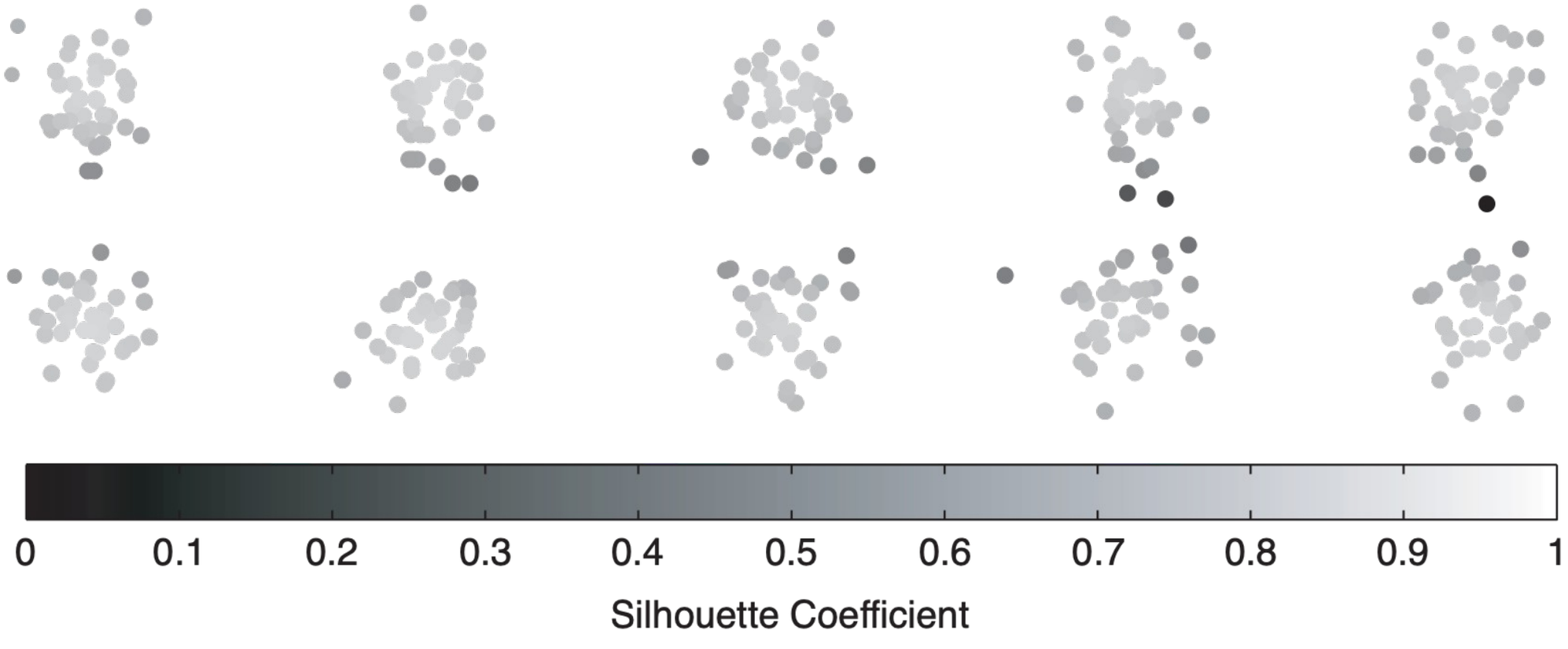
Datum *i*

Silhouette

$$s(i) = \frac{b(i) - a(i)}{b(i)}$$

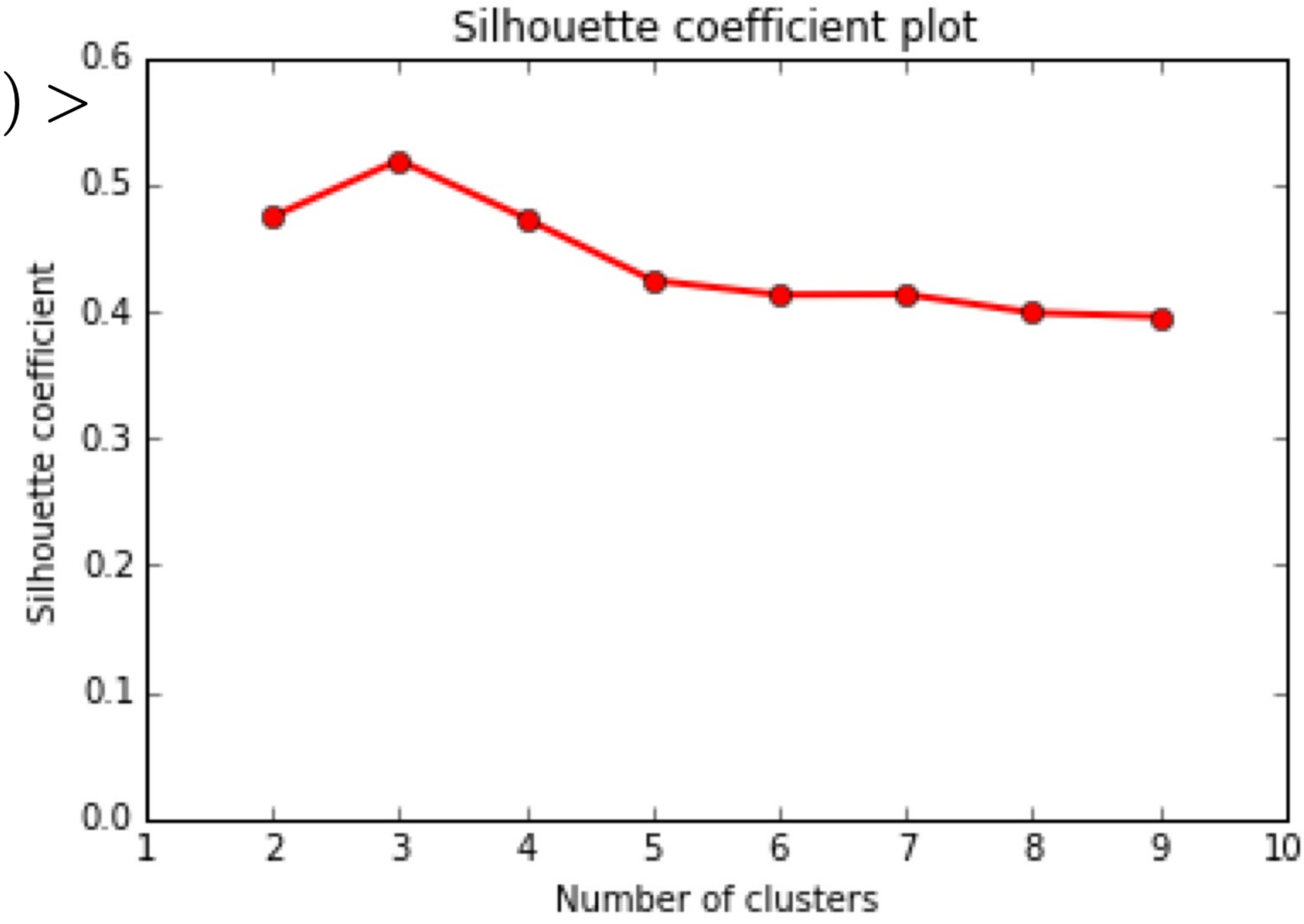
$$0 \leq s(i) \leq 1$$

Normalisation



How to determine “best” value for K?

$< s(i) >$



Clustering algorithms

Essence: *Flat* versus *hierarchical* clustering

- **Centroid-based:**
  KMeans, …
- **Density-based:**
  MeanShift, …
- **Graph-based:**
  Spectral, …