CS 412

APRIL 16TH – UNSUPERVISED LEARNING

Administrivia

HW4 is out

- Ensemble methods
  - Can specify hyperparameters for the `base_estimator`
- Testing and concentration bounds

The rest of the class

- Unsupervised Learning (HW5)
- Deep learning ←
- Project: meetings this week and next, sign up online
  - bb collab rooms set up
  - 20 minute meetings, 5-10 minute presentation + Q&A

$P(|X - E[X]| > \epsilon) < \delta$

DecisionTree( )

max leaf nodes

$\epsilon = \sqrt{\frac{1}{2m} \log \frac{2}{\delta}}$

$E_{test} \pm \epsilon$

99% confidence

→ $0.01 = \delta$

20% final exam

30% project

How to find a good size proejct

- Find something that is relevant to your research area
- Find a dataset that is interesting to you. ML gets done in all disciplines, so find something creative
- Start looking through large ML conferences to find some research that is interesting
  - Just read the abstracts, maybe the conclusions
- I'm not expecting new ML research in this project, it is 100% okay to take a project someone else has done and make a small tweak to it
  - E.g. Use a GAN model from a medical paper and apply it to traffic data
- If you are having trouble finding an interesting topic/dataset, let me know. When we meet as groups in a ~~couple~~ weeks, I want some progress to have been made.

1.) Book - anything we haven't covered so far / game

2)

Some data sources

- UIC
  - This is a research university, don't hesitate to **politely** reach out to faculty members about using data that they may have collected
- UCI
  - A good repository of clean (although possibly a little boring) data.
  - A good place to start if you want to try one approach on multiple data sets
- Others
  - Kaggle.com
  - mldata.org
  - US Census Bureau
  - MSR GPS data
  - Other local companies (again, be polite and recognize that lots of data is proprietary and may be hard to get)

Vary quality

Project Expectations

C Expectations (<20/30)
- Some new dataset
- Some model not covered in the HW
- Minimal discussion or analysis

B Expectations (25/30)
- Multiple data sets / multiple applied models
- Thorough analysis, discussion and comparison

A Expectations (30/30)
- At least one dataset with unique properties (time-series, missing data, few data points)
- Literature review of at least one recent and relevant ML article

A+ Expectations (35+/30)
- Multiple data sets
- Multiple research based applications
- Thorough discussion and analysis of the results

Project Expectations

All projects must have

- At least one new data set
- Some implementation portion
- Multiple models compared against each other
- Comparison against models from the course
- Statistical analysis
- Some portion of material outside of the class and a literature review of those portions
- Figures and tables
- **Citations!** It's okay to use other peoples research and other people's code but you **MUST** indicate where you found your work

I want to give you as much freedom as possible to find a project that's interesting to you

- It's okay to have some sections that are much larger than others. For example,
  - Review 5-10 relevant papers in a thorough lit review and have a small implementation component
  - Choose a simple new model to implement and implement it across several data sets
  - Opt for a deep dive of a new ML approach on a single data set

Project Deliverables

Project proposal (5 points)
- Already submitted

Progress report (10 points)
- Our meeting as a team on a day decided from the proposal
- This should not be a brainstorming session
- Your best opportunity to get help from me during the actual project

Final Report (30 points)
- Our meeting as a team on a day decided from the proposal
- This should not be a brainstorming session
- Your best opportunity to get help from me during the actual project

Individual team evaluations (5 points)
- Due on May 10th, with the project
- **All members must fill out the evaluations independently**

Sign up on
posted on quizz

Supervised learning vs. unsupervised learning

Supervised learning: discover patterns in the data that relate data attributes with a target (class) attribute.

- These patterns are then utilized to predict the values of the target attribute in future data instances.

labels are provided

Unsupervised learning: The data have no target attribute.

- We want to explore the data to find some intrinsic structures in them.

Digits: cluster 10 clusters to assume those are 10 digits

Clustering

Clustering is a technique for finding similarity groups in data, called clusters. I.e.,

- it groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.

Clustering is often called an unsupervised learning task as no class values denoting an a priori grouping of the data instances are given, which is the case in supervised learning.

Due to historical reasons, clustering is often considered synonymous with unsupervised learning.

- In fact, association rule mining is also unsupervised

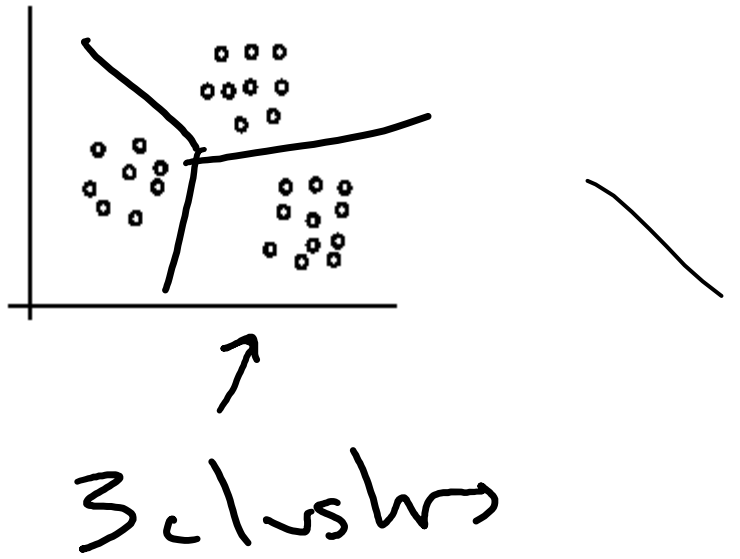
This chapter focuses on clustering.

An illustration

The data set has three natural groups of data points, i.e., 3 natural clusters.

How can we determine what these clusters are without labels?

- What makes this more difficult? How many clusters can there be?
-



What is clustering for?

Let us see some real-life examples

Example 1: groups people of similar sizes together to make “small”, “medium” and “large” T-Shirts.

- Tailor-made for each person: too expensive
- One-size-fits-all: does not fit all.

How many sizes?

Example 2: In marketing, segment customers according to their similarities

- To do targeted marketing.

How many clusters

What is clustering for? (cont…)

Example 3: Given a collection of text documents, we want to organize them according to their content similarities,

- To produce a topic hierarchy

In fact, clustering is one of the most utilized data mining techniques.

- It has a long history, and used in almost every field, e.g., medicine, psychology, botany, sociology, biology, archeology, marketing, insurance, libraries, etc.
- In recent years, due to the rapid increase of online documents, text clustering becomes important.

Aspects of clustering

A clustering algorithm

- Partitional clustering
- Hierarchical clustering
- …

A distance (similarity, or dissimilarity) function

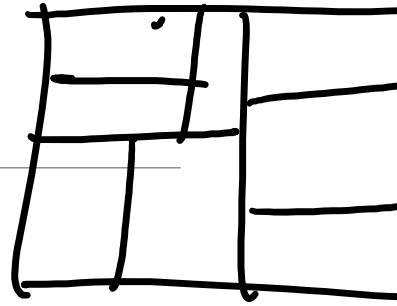
Clustering quality

- Inter-clusters distance ⇒ maximized
- Intra-clusters distance ⇒ minimized

The quality of a clustering result depends on the algorithm, the distance function, and the application.

K-means clustering

partition the space



K-means is a partitional clustering algorithm

Let the set of data points (or instances) D be

{x1, x2, …, xn},

- where xi = (xi1, xi2, …, xir) is a vector in a real-valued space $X \subseteq R^r$, and r is the number of attributes (dimensions) in the data.

The k-means algorithm partitions the given data into k clusters.

- Each cluster has a cluster center, called centroid.
- k is specified by the user (or found with experimentation)

For Digits: 20 clusters

K-means algorithm

Given k, the k-means algorithm works as follows:

- Randomly choose k data points (seeds) to be the initial centroids, cluster centers
- Assign each data point to the closest centroid
- Re-compute the centroids using the current cluster memberships.
- If a convergence criterion is not met, go to 2).

K-means algorithm – (cont …)

Algorithm *k*-means(*k*, *D*)

1 Choose *k* data points as the initial centroids (cluster centers)
2 **repeat**
3 **for** each data point $\mathbf{x} \in D$ do
4 compute the distance from **x** to each centroid;
5 assign **x** to the closest centroid // a centroid represents a cluster
6 **endfor**
7 re-compute the centroids using the current cluster memberships
8 **until** the stopping criterion is met

partition

Strengths of k-means

Strengths:

- Simple: easy to understand and to implement
- Efficient: Time complexity: O(tkn),
  - where n is the number of data points,
  - k is the number of clusters, and
  - t is the number of iterations. ← assumes it does converge
- Since both k and t are small. k-means is considered a linear algorithm.

K-means is the most popular clustering algorithm.

Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity. (In fact, the optimal k-cover is provably NP-hard to find)

← squared distance from all points to center

the assignment of points that minimizes cluster size is NP-hard

Weaknesses of k-means

The algorithm is only applicable if the mean is defined.

- For categorical data, k-mode - the centroid is represented by most frequent values.

The user needs to specify k.

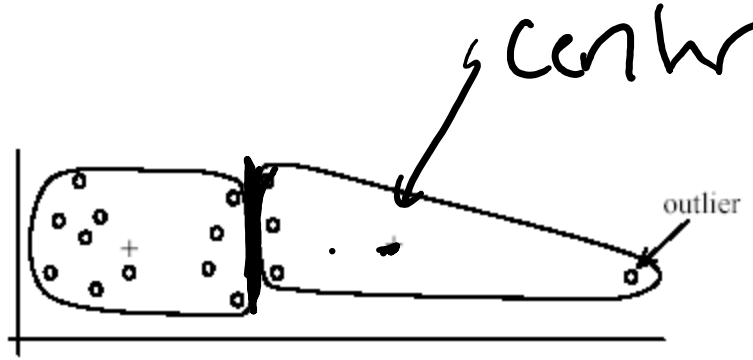
The algorithm is sensitive to outliers

- Outliers are data points that are very far away from other data points.
- Outliers could be errors in the data recording or some special data points with very different values.

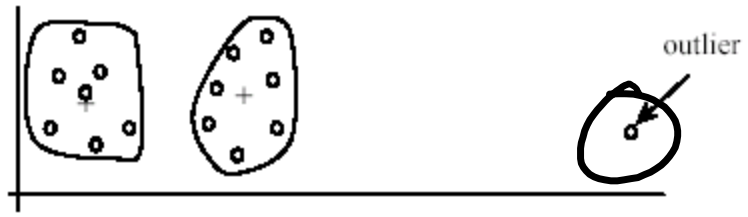
Weaknesses of k-means: Problems with outliers

Sensitive to shit

Seeds



(A): Undesirable clusters



(B): Ideal clusters

center

can produce poor clusters

make k = 3

Weaknesses of k-means: To deal with outliers

One method is to remove some data points in the clustering process that are much further away from the centroids than other data points.

- To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.

Another method is to perform random sampling. Since in sampling we only choose a small subset of the data points, the chance of selecting an outlier is very small.

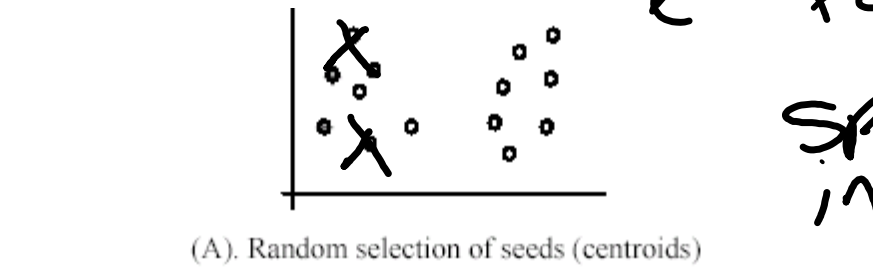
- Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

A k-means model is uniquely described by the k-centers

Weaknesses of k-means (cont …)

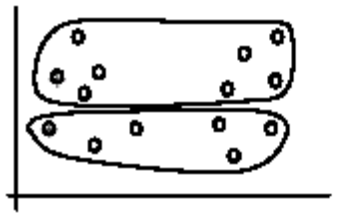
The algorithm is sensitive to initial seeds.

Rerun the random code multiple times.

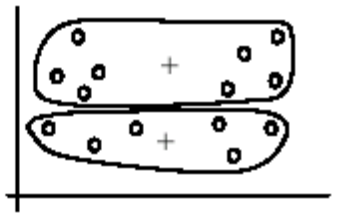


(A). Random selection of seeds (centroids)

easy to spot in 2D



(B). Iteration 1



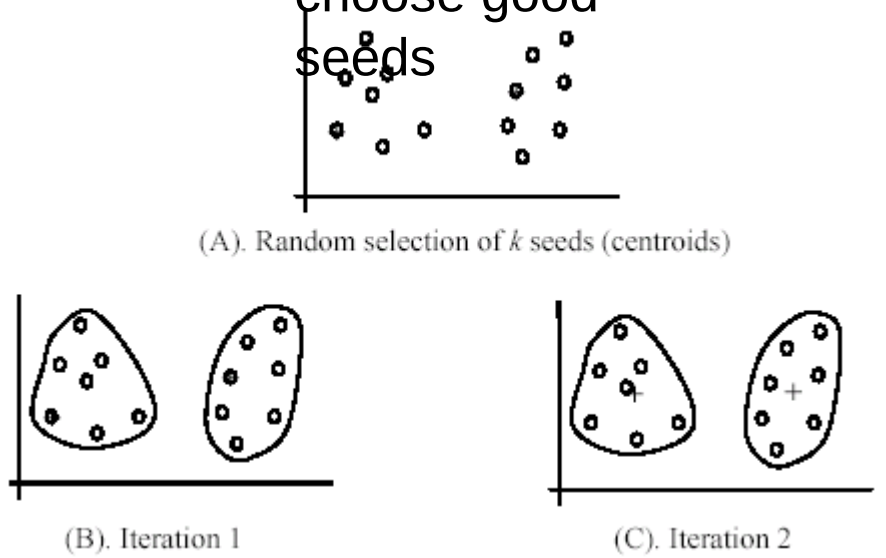
(C). Iteration 2

bad cluster

Weaknesses of k-means (cont …)

If we use different seeds: good results

There are some methods to help choose good seeds



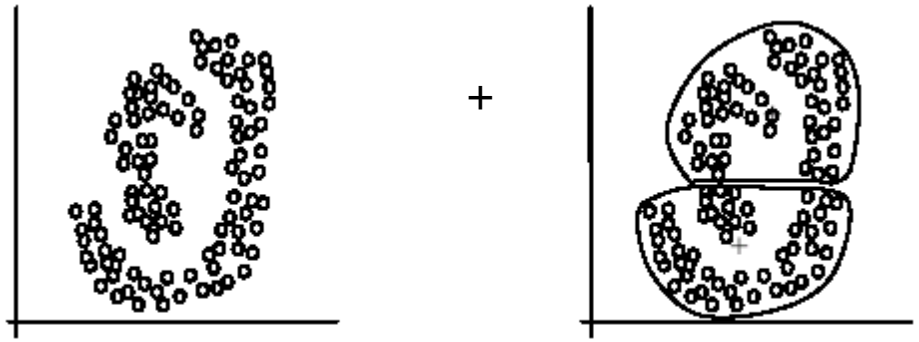
(A). Random selection of k seeds (centroids)

(B). Iteration 1

(C). Iteration 2

Weaknesses of k-means (cont …)

The k-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



(A): Two natural clusters (B): *k*-means clusters

K-means summary

Despite weaknesses, k-means is still the most popular algorithm due to its simplicity, efficiency and

- other clustering algorithms have their own lists of weaknesses.

No clear evidence that any other clustering algorithm performs better in general

- although they may be more suitable for some specific types of data or applications.

Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

Common ways to represent clusters

Use the centroid of each cluster to represent the cluster.

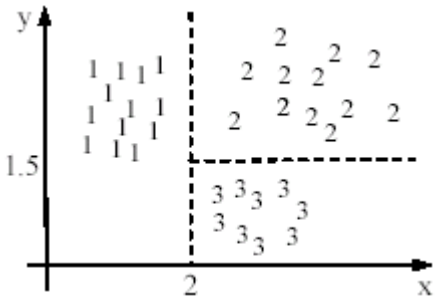
- compute the radius and
- standard deviation of the cluster to determine its spread in each dimension

- The centroid representation alone works well if the clusters are of the hyper-spherical shape.
- If clusters are elongated or are of other shapes, centroids are not sufficient

Using classification model

All the data points in a cluster are regarded to have the same class label, e.g., the cluster ID.

- run a supervised learning algorithm on the data to find a classification model.



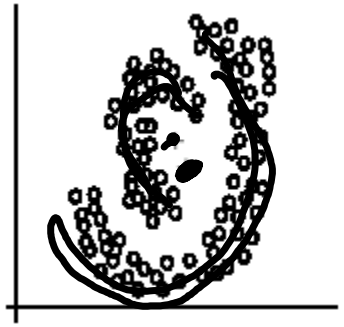
$x \leq 2 \rightarrow$ cluster 1
$x > 2, y > 1.5 \rightarrow$ cluster 2
$x > 2, y \leq 1.5 \rightarrow$ cluster 3

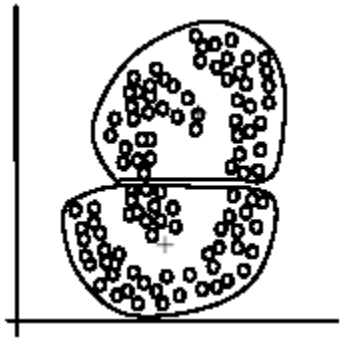
Clusters of arbitrary shapes

Hyper-elliptical and hyper-spherical clusters are usually easy to represent, using their centroid together with spreads.

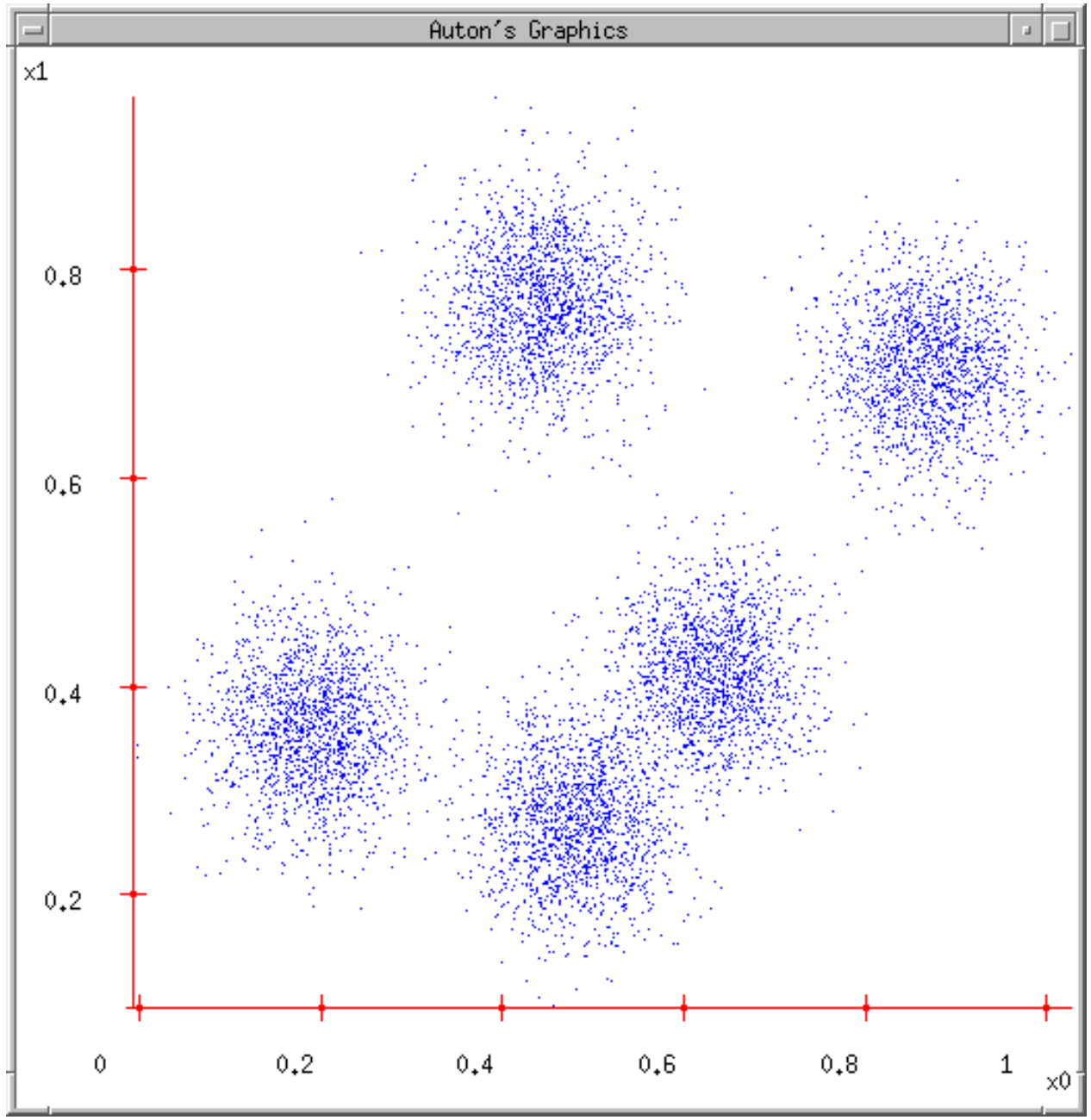
Irregular shape clusters are hard to represent. They may not be useful in some applications.

- Using centroids are not suitable (upper figure) in general
- K-means clusters may be more useful (lower figure), e.g., for making 2 size T-shirts.



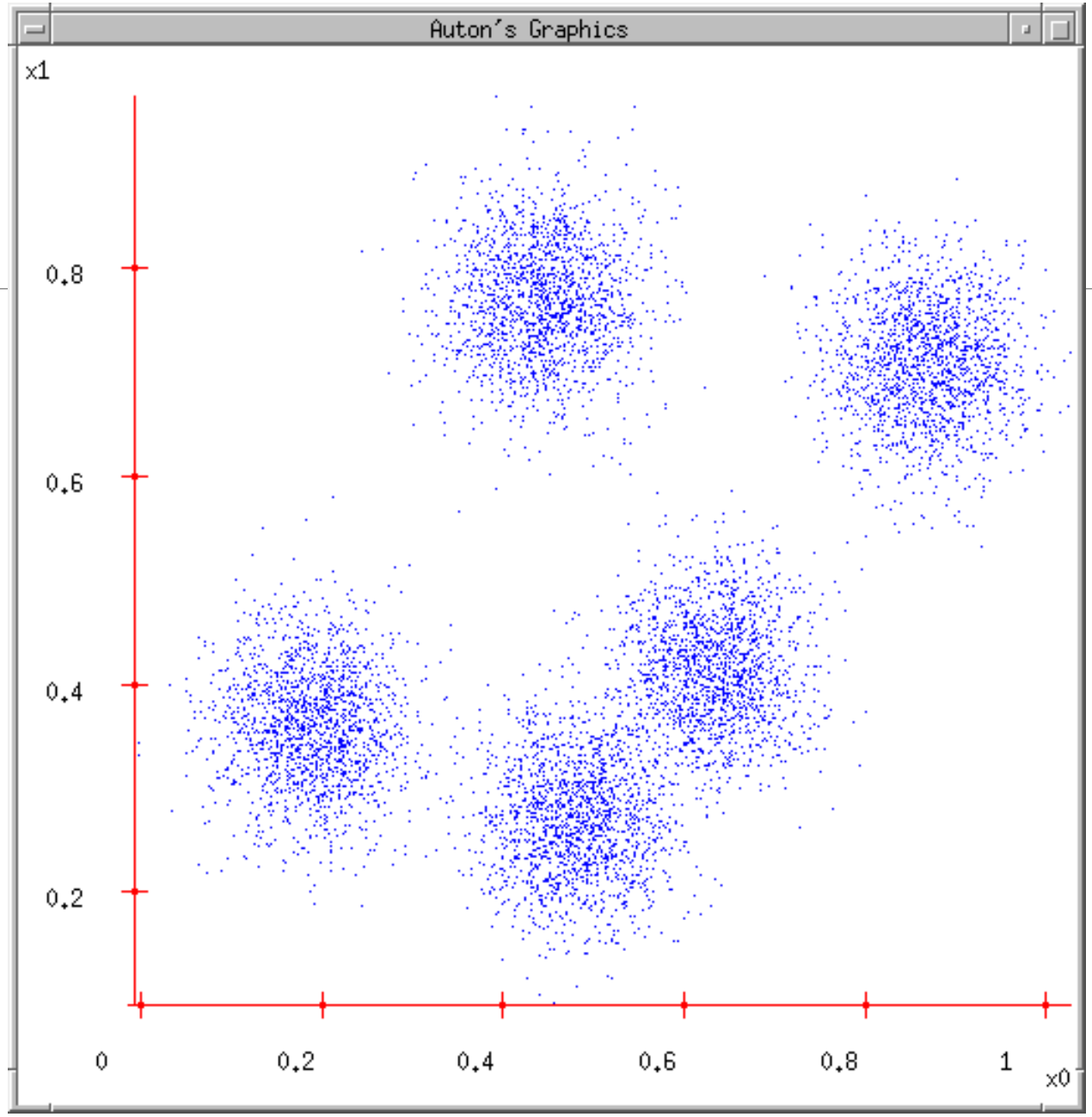


Kernel transformation
— watch out!


Auton's Graphics
x1
0.8
0.6
0.4
0.2
0
0.2
0.4
0.6
0.8
1
x0

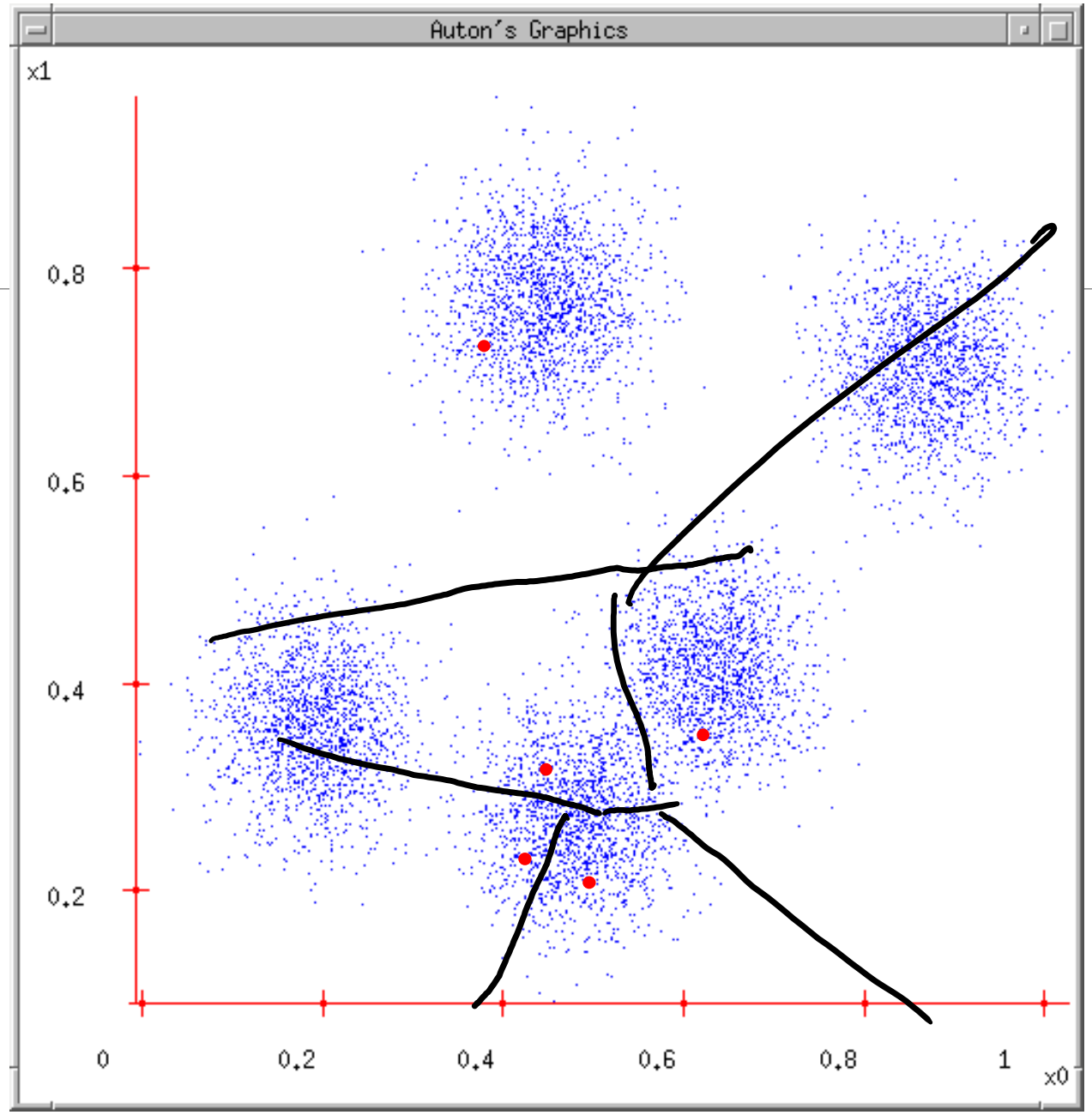
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



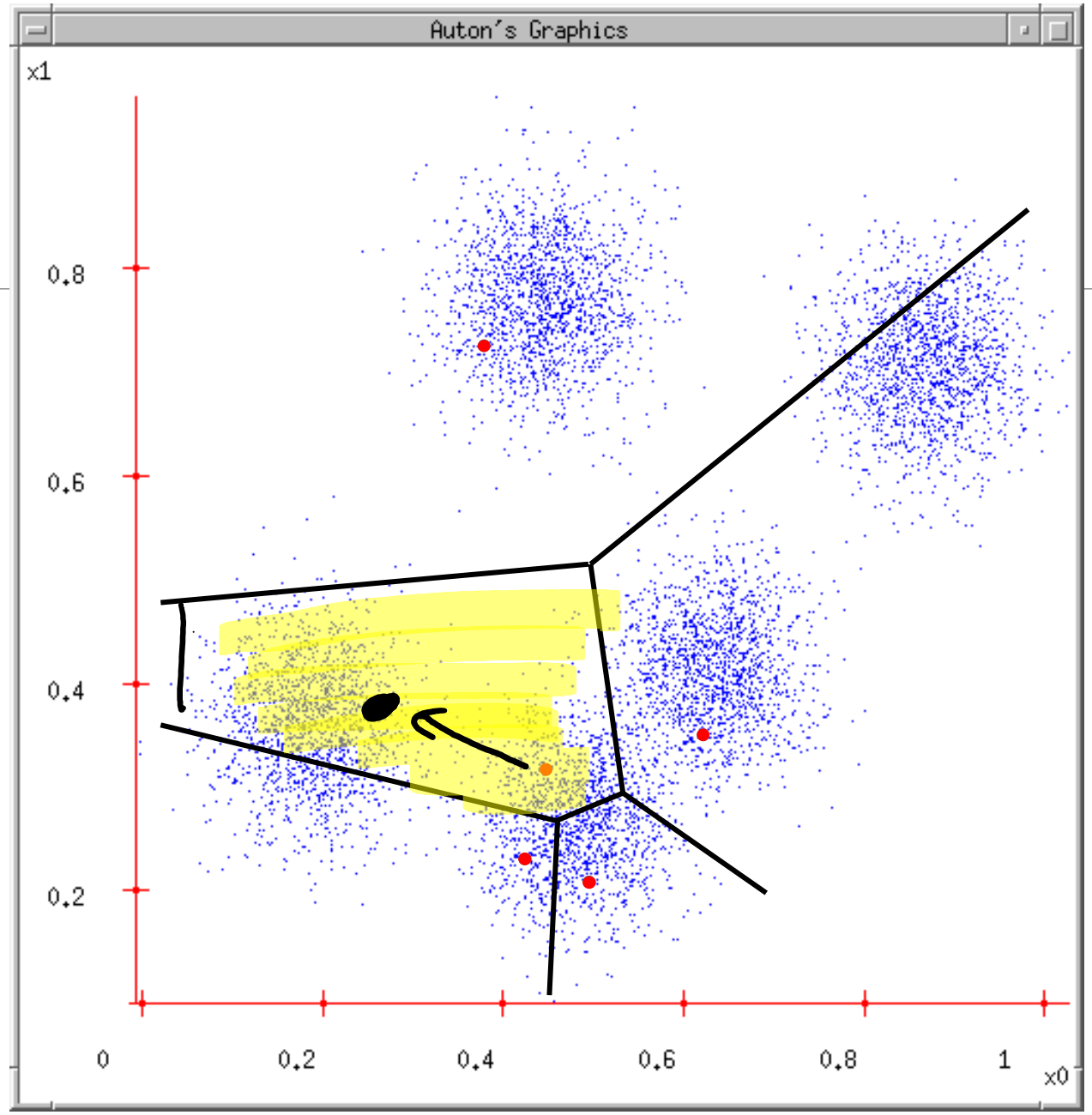
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



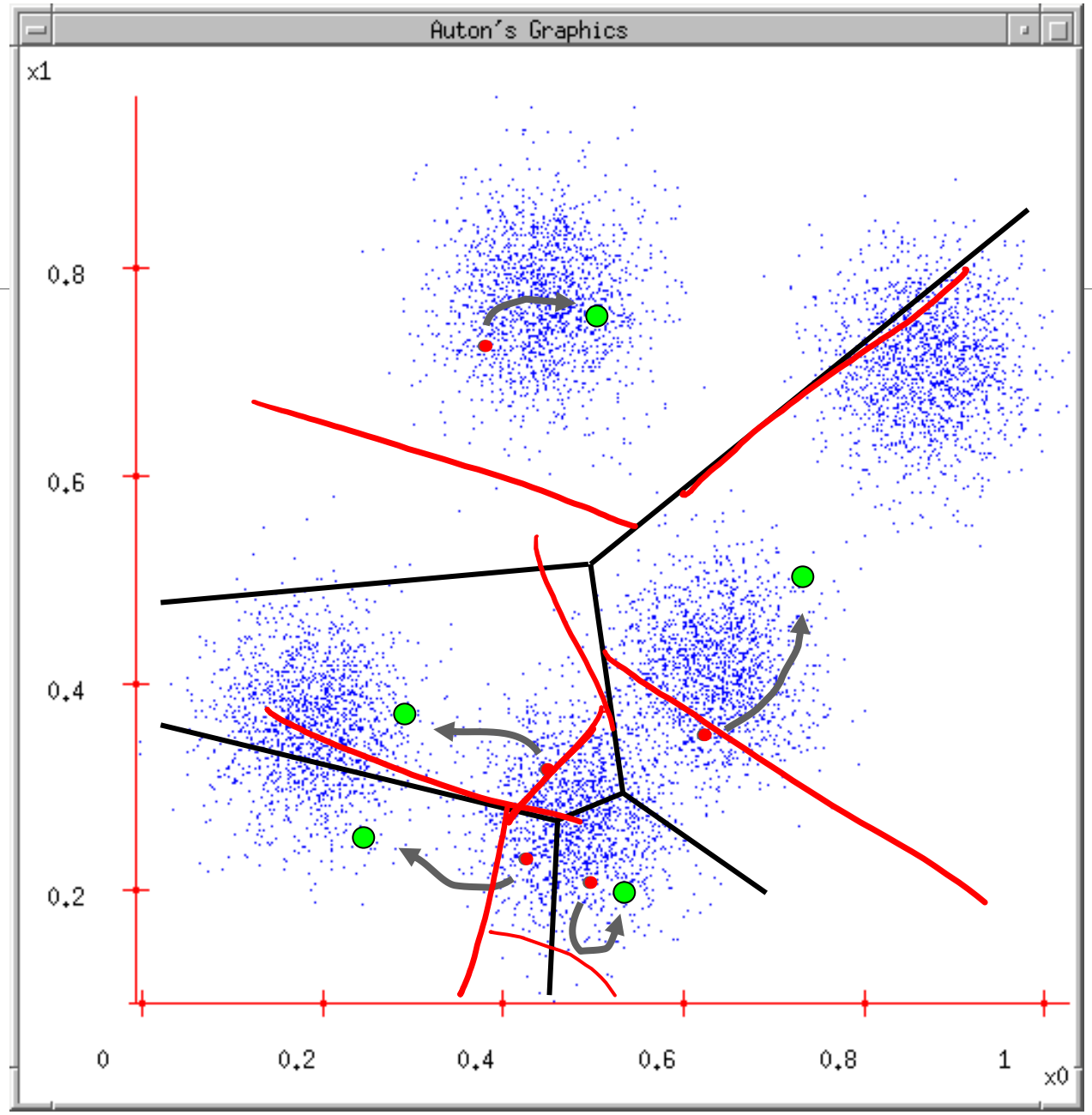
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



k-means Clustering

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

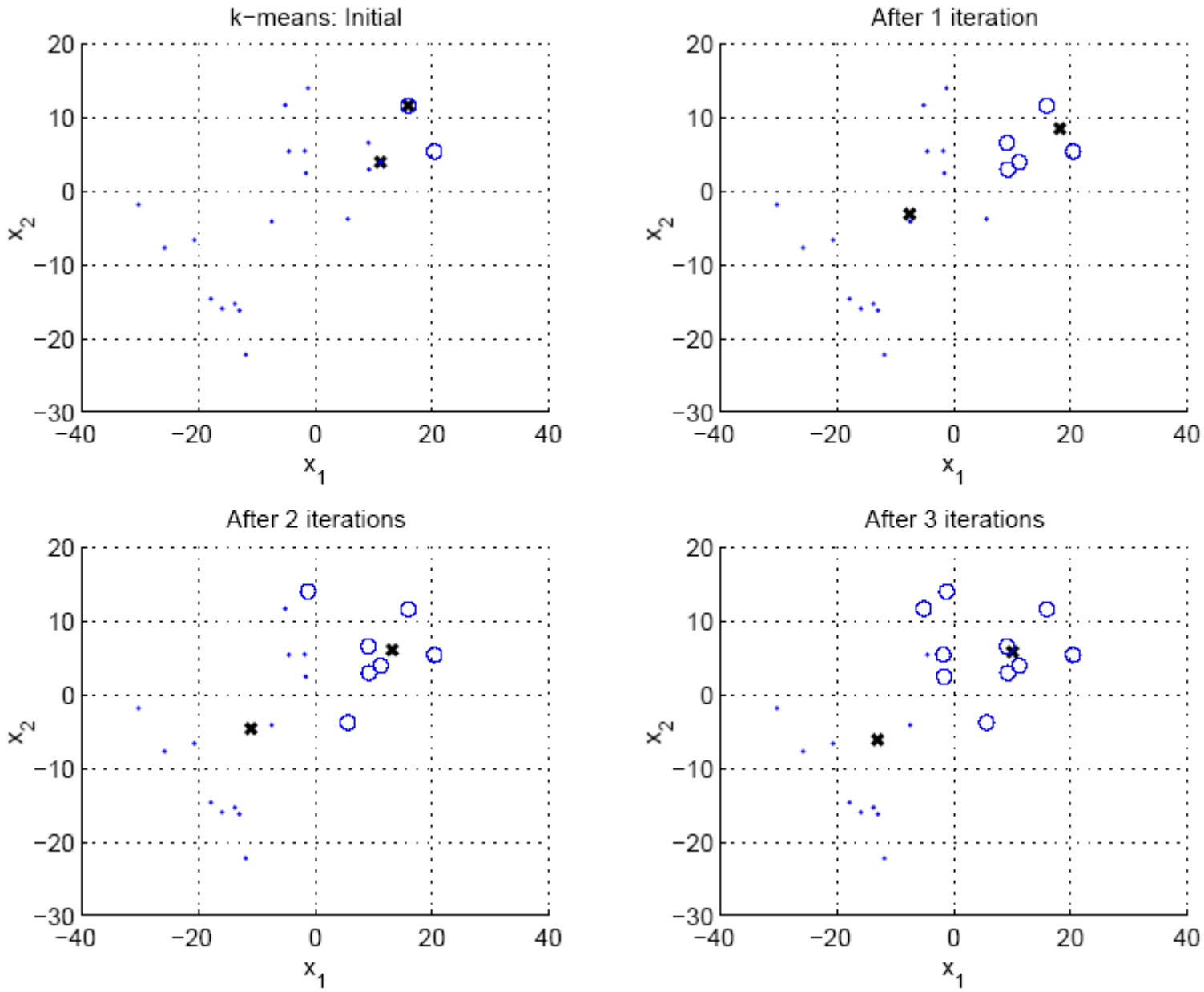
For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

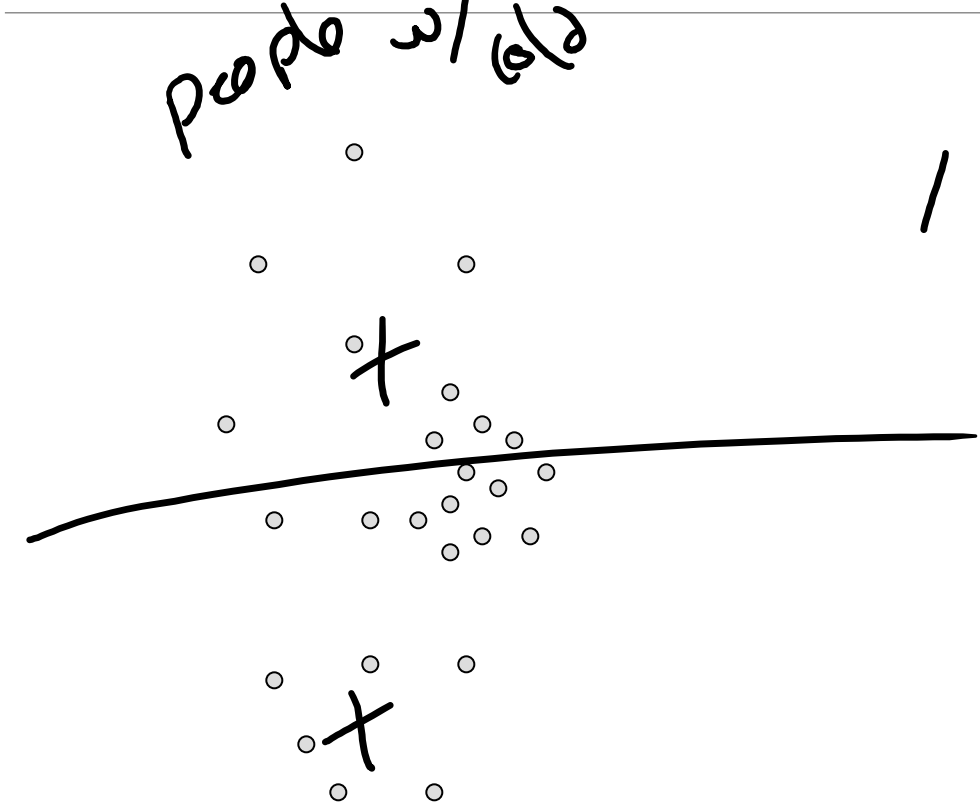
For all $\boldsymbol{m}_i, i = 1, \ldots, k$

$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge


k−means: Initial
After 1 iteration
After 2 iterations
After 3 iterations
x1
x2
20
10
0
−10
−20
−30
−40
−20
0
20
40

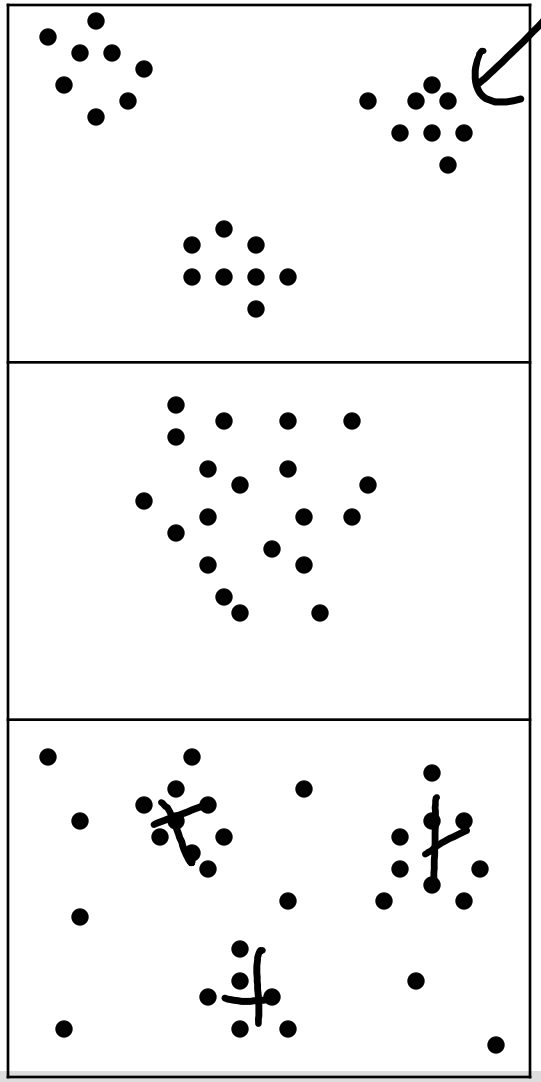
Bad cases for k-means



- Overlapping clusters
- Some clusters may be "wider" than others

almost impossible to solve w/ k means since its partition

has $x_1 > 10$



Sometimes easy

Sometimes impossible

and sometimes in between

Choosing k

Ideally, k is defined by the application, e.g., color quantization

Plot data (Projection to low dimension) and check for clusters

Incremental (leader-cluster) algorithm: Add one at a time until "elbow" (reconstruction error/log likelihood/intergroup distances)

Manually check for meaning

low likelihood

metric to consider the quality of the cluster

After Clustering

Clustering methods find similarities between instances and use it to group instances

Allows knowledge extraction through

- number of clusters,
- prior probabilities,
- cluster parameters, i.e., center, range of features (demographic/transaction).

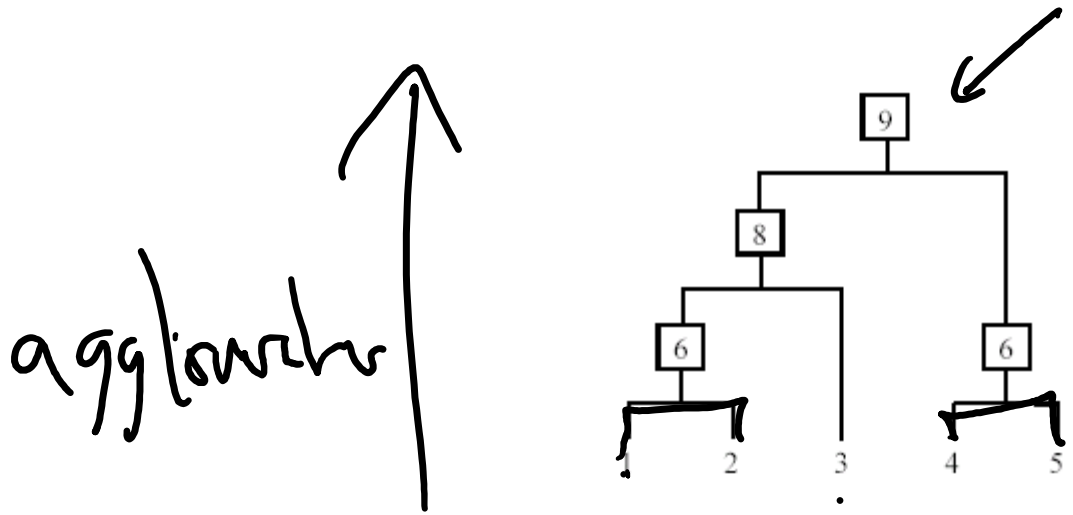
Other clustering methods

- Agglomerative
- Divisive

Hierarchical clustering

Hierarchical Clustering

Produce a nested sequence of clusters, a tree,
also called Dendrogram.



Types of hierarchical clustering

Agglomerative (bottom up) clustering: It builds the dendrogram (tree) from the bottom level, and

- merges the most similar (or nearest) pair of clusters
- stops when all the data points are merged into a single cluster (i.e., the root cluster).

Divisive (top down) clustering: It starts with all data points in one cluster, the root.

- Splits the root into a set of child clusters. Each child cluster is recursively divided further
- stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

difficult because it is harder to quantify the quality of the clusters

lots more options at each step

Agglomerative clustering

Very common clustering approach.

At the beginning, each data point forms a cluster (also called a node).

Merge nodes/clusters that have the least distance.

Go on merging

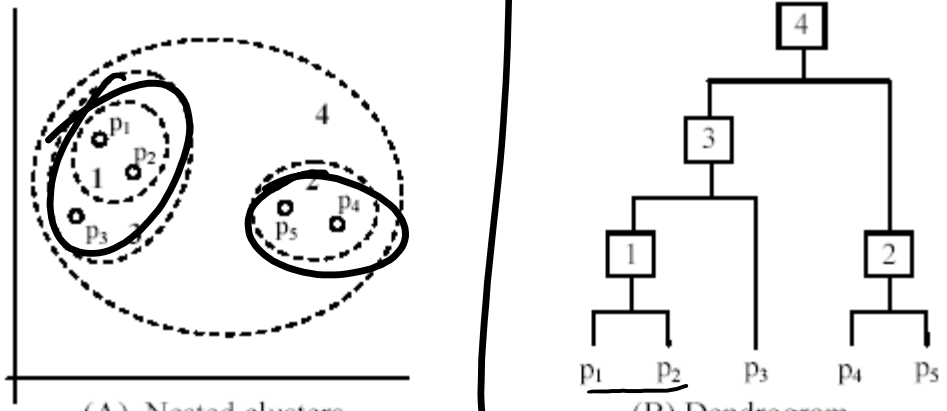
Eventually all nodes belong to one cluster

Agglomerative clustering algorithm

Algorithm Agglomerative(D)

1 Make each data point in the data set D a cluster,
2 Compute all pair-wise distances of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in D$;
2 **repeat**
3 find two clusters that are nearest to each other;
4 merge the two clusters form a new cluster c;
5 compute the distance from c to all other clusters;
12 **until** there is only one cluster left

An example: working of the algorithm



(A). Nested clusters

(B) Dendrogram

#s correspond to the order which the clusters were created

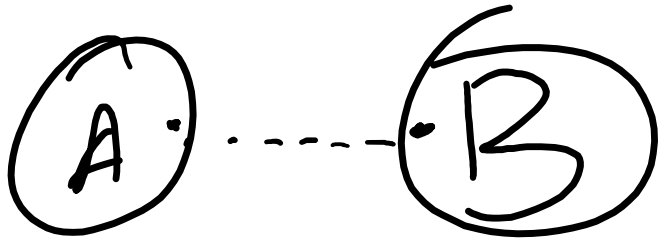
Measuring the distance of two clusters

A few ways to measure distances of two clusters.

Results in different variations of the algorithm.

- Single link
- Complete link
- Average link
- Centroids
- …

Single link

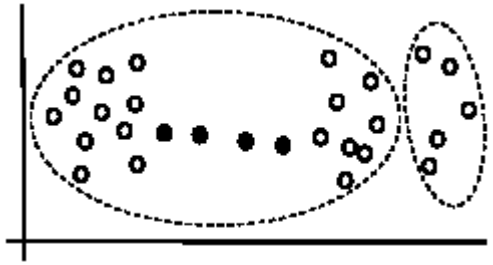


distance = distance of two closest points

The distance between two clusters is the distance between two closest data points in the two clusters, one data point from each cluster.

It can find arbitrarily shaped clusters, but

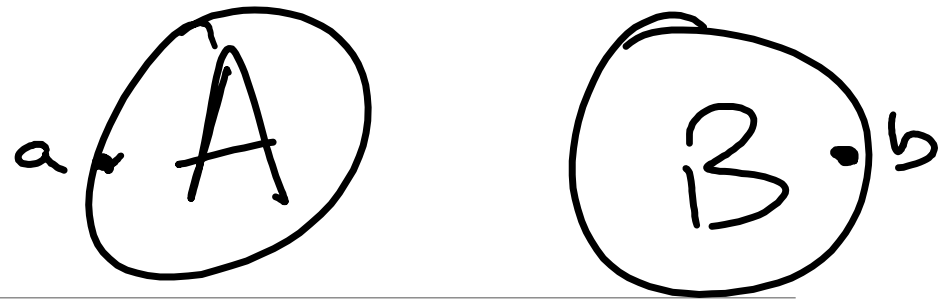
- It may cause the undesirable "chain effect" by noisy points



Two natural clusters are split into two

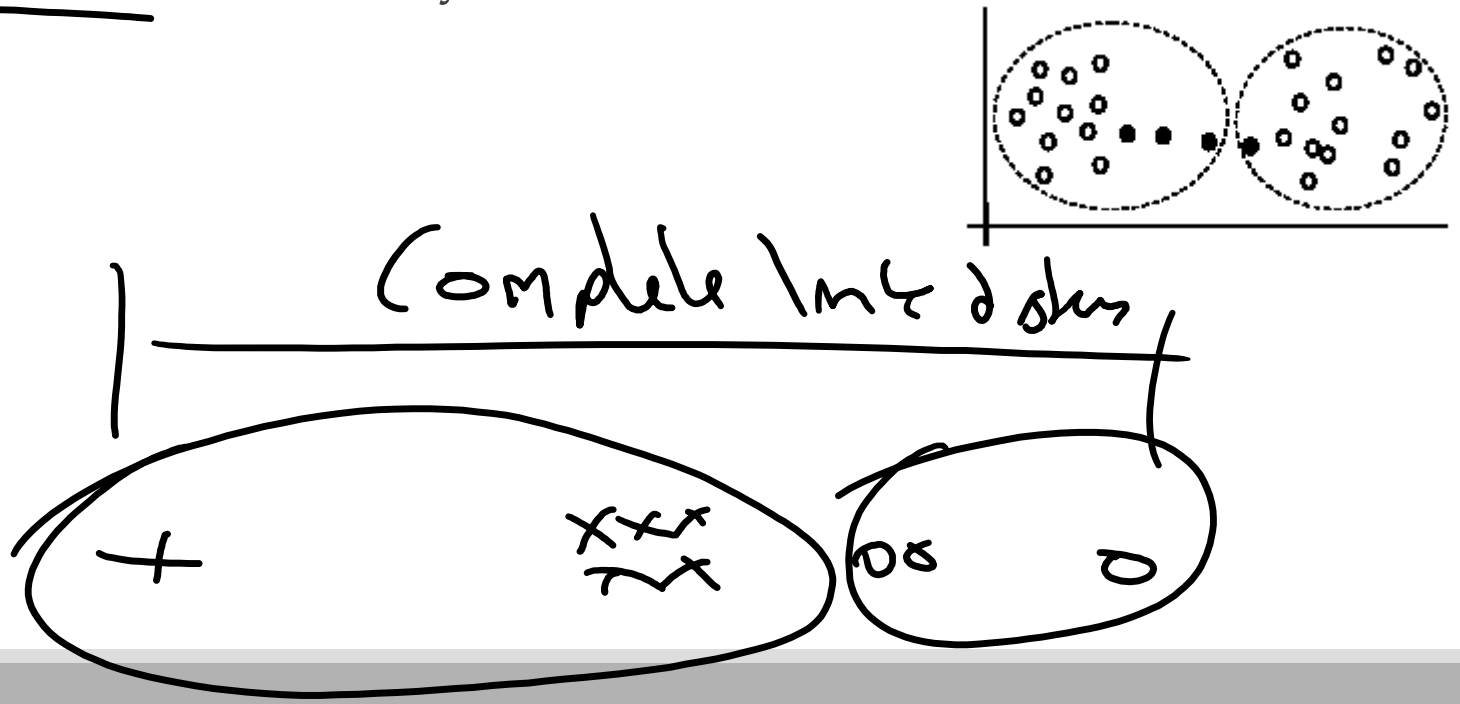
undesirably

Complete link

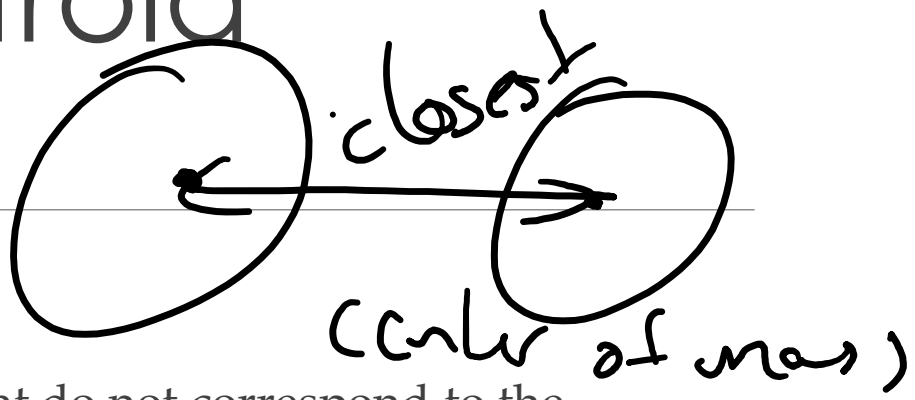


The distance between two clusters is the distance of two furthest data points in the two clusters.

It is sensitive to outliers because they are far away



Average link and centroid methods



Average link: A compromise between

- the sensitivity of complete-link clustering to outliers and
- the tendency of single-link clustering to form long chains that do not correspond to the intuitive notion of clusters as compact, spherical objects.
- In this method, the distance between two clusters is the average distance of all pair-wise distances between the data points in two clusters.

Centroid method: In this method, the distance between two clusters is the distance between their centroids

Need to define centroids

Road map

Basic concepts

K-means algorithm

Representation of clusters

Hierarchical clustering

Distance functions

Data standardization

Handling mixed attributes

Which clustering algorithm to use?

Cluster evaluation

Discovering holes and data regions

Summary

Distance functions

Key to clustering. “similarity” and “dissimilarity” can also commonly used terms.

There are numerous distance functions for

- Different types of data
  - Numeric data
  - Nominal data
- Different specific applications

Euclidean distance and Manhattan distance

For real numbers

If h = 2, it is the Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{ir} - x_{jr})^2}$$

If h = 1, it is the Manhattan distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{ir} - x_{jr}|$$

Weighted Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + \ldots + w_r(x_{ir} - x_{jr})^2}$$

Squared distance and Chebychev distance

Squared Euclidean distance: to place progressively greater weight on data points that are further apart.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ir} - x_{jr})^2$$

Chebychev distance: one wants to define two data points as "different" if they are different on any one of the attributes.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \max(|x_{i1} - x_{j1}|, |x_{i2} - x_{j2}|, ..., |x_{ir} - x_{jr}|)$$

Data standardization

In the Euclidean space, standardization of attributes is recommended so that all attributes can have equal impact on the computation of distances.

Consider the following pair of data points

- xi: (0.1, 20) and xj: (0.9, 720).

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(0.9-0.1)^2 + (720-20)^2} = 700.000457,$$

The distance is almost completely dominated by (720-20) = 700.

Standardize attributes: to force the attributes to have a common value range

Interval-scaled attributes

Their values are real numbers following a linear scale.

- The difference in Age between 10 and 20 is the same as that between 40 and 50.
- The key idea is that intervals keep the same importance through out the scale

Two main approaches to standardize interval scaled attributes, range and z-score. f is an attribute

$$range(x_{if}) = \frac{x_{if} - \min(f)}{\max(f) - \min(f)},$$

Interval-scaled attributes (cont …)

Z-score: transforms the attribute values so that they have a mean of zero and a mean absolute deviation of 1. The mean absolute deviation of attribute f, denoted by sf, is computed as follows

$$s_f = \frac{1}{n}\left(|x_{1f} - m_f| + |x_{2f} - m_f| + \ldots + |x_{nf} - m_f|\right),$$

$$m_f = \frac{1}{n}\left(x_{1f} + x_{2f} + \ldots + x_{nf}\right),$$

Z-score: $$z(x_{if}) = \frac{x_{if} - m_f}{s_f}.$$

Ratio-scaled attributes

income
5,000 → 10,000
100,000 → 105,000

Numeric attributes, but unlike interval-scaled attributes, their scales are exponential,

For example, the total amount of microorganisms that evolve in a time t is approximately given by

AeBt,

- where A and B are some positive constants.

Do log transform:

- Then treat it as an interval-scaled attribuete $\log(x_{if})$

if a doubly of a featur always
Causes the same chng

Categorical

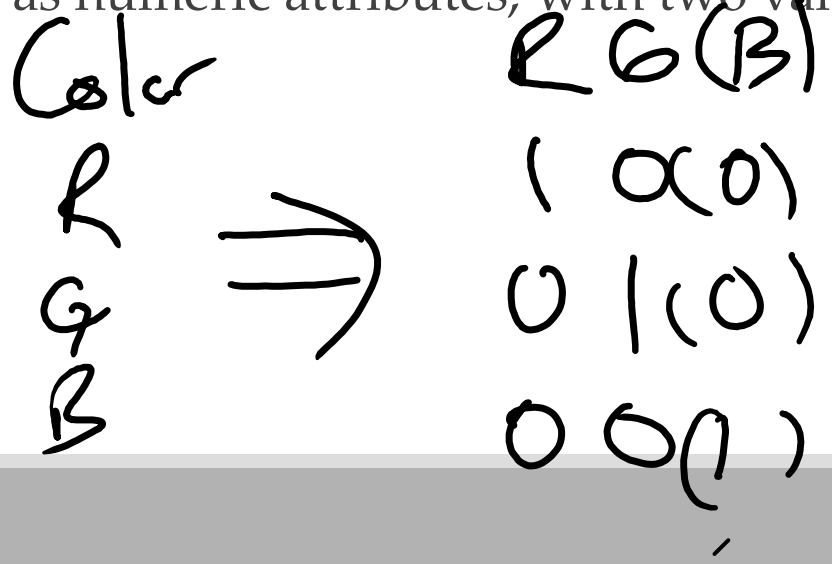
Nominal attributes

Sometime, we need to transform nominal attributes to numeric attributes.

Transform nominal attributes to binary attributes.

- The number of values of a nominal attribute is v.
- Create v binary attributes to represent them.
- If a data instance for the nominal attribute takes a particular value, the value of its binary attribute is set to 1, otherwise it is set to 0.

The resulting binary attributes can be used as numeric attributes, with two values, 0 and 1.



Nominal attributes: an example

Nominal attribute fruit: has three values,

- Apple, Orange, and Pear

We create three binary attributes called, Apple, Orange, and Pear in the new data.

If a particular data instance in the original data has Apple as the value for fruit,

- then in the transformed data, we set the value of the attribute Apple to 1, and
- the values of attributes Orange and Pear to 0

Ordinal attributes

Likert scale:
Strongly disagree
⋮
Strongly agree

Ordinal attribute: an ordinal attribute is like a nominal attribute, but its values have a numerical ordering. E.g., 1 2 3

- Age attribute with values: Young, MiddleAge and Old. They are ordered.
- Common approach to standardization: treat is as an interval-scaled attribute.

What does this suppose?
The semantic distance between Young and MA is the same as MA and Old.

Mixed attributes

Our distance functions given are for data with all numeric attributes, or all nominal attributes, etc.

Practical data has different types:

- Any subset of the 6 types of attributes,
  - interval-scaled,
  - symmetric binary,
  - asymmetric binary,
  - ratio-scaled,
  - ordinal and
  - nominal

Convert to a single type

One common way of dealing with mixed attributes is to

- Decide the dominant attribute type, and
- Convert the other types to this type.

E.g, if most attributes in a data set are interval-scaled,

- we convert ordinal attributes and ratio-scaled attributes to interval-scaled attributes.
- It is also appropriate to treat symmetric binary attributes as interval-scaled attributes.

So all data pomly or categorical except for afew numrical ones

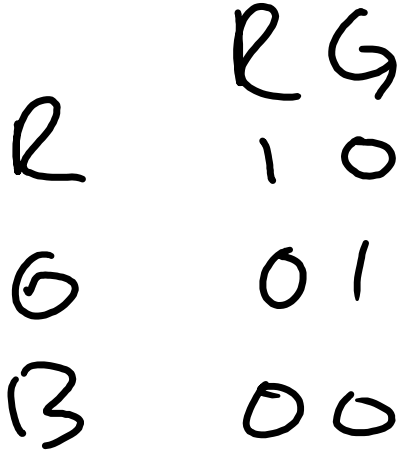
Numerical → categorical $X \Rightarrow X_1 < 0.5, X_1 \geq 0.5$

Convert to a single type (cont …)

It does not make much sense to convert a nominal attribute or an asymmetric binary attribute to an interval-scaled attribute,

- but it is still frequently done in practice by assigning some numbers to them according to some hidden ordering, e.g., prices of the fruits

Alternatively, a nominal attribute can be converted to a set of (symmetric) binary attributes, which are then treated as numeric attributes.



How to choose a clustering algorithm

No Error

Clustering research has a long history. A vast collection of algorithms are available.

- We only introduced several main algorithms.

Choosing the "best" algorithm is a challenge.

- Every algorithm has limitations and works well with certain data distributions.
- It is very hard, if not impossible, to know what distribution the application data follow. The data may not fully follow any "ideal" structure or distribution required by the algorithms.
- One also needs to decide how to standardize the data, to choose a suitable distance function and to select other parameter values.

Choose a clustering algorithm (cont …)

Due to these complexities, the common practice is to

- run several algorithms using different distance functions and parameter settings, and
- then carefully analyze and compare the results.

The interpretation of the results must be based on insight into the meaning of the original data together with knowledge of the algorithms used.

Clustering is highly application dependent and to certain extent subjective (personal preferences).