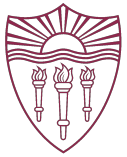
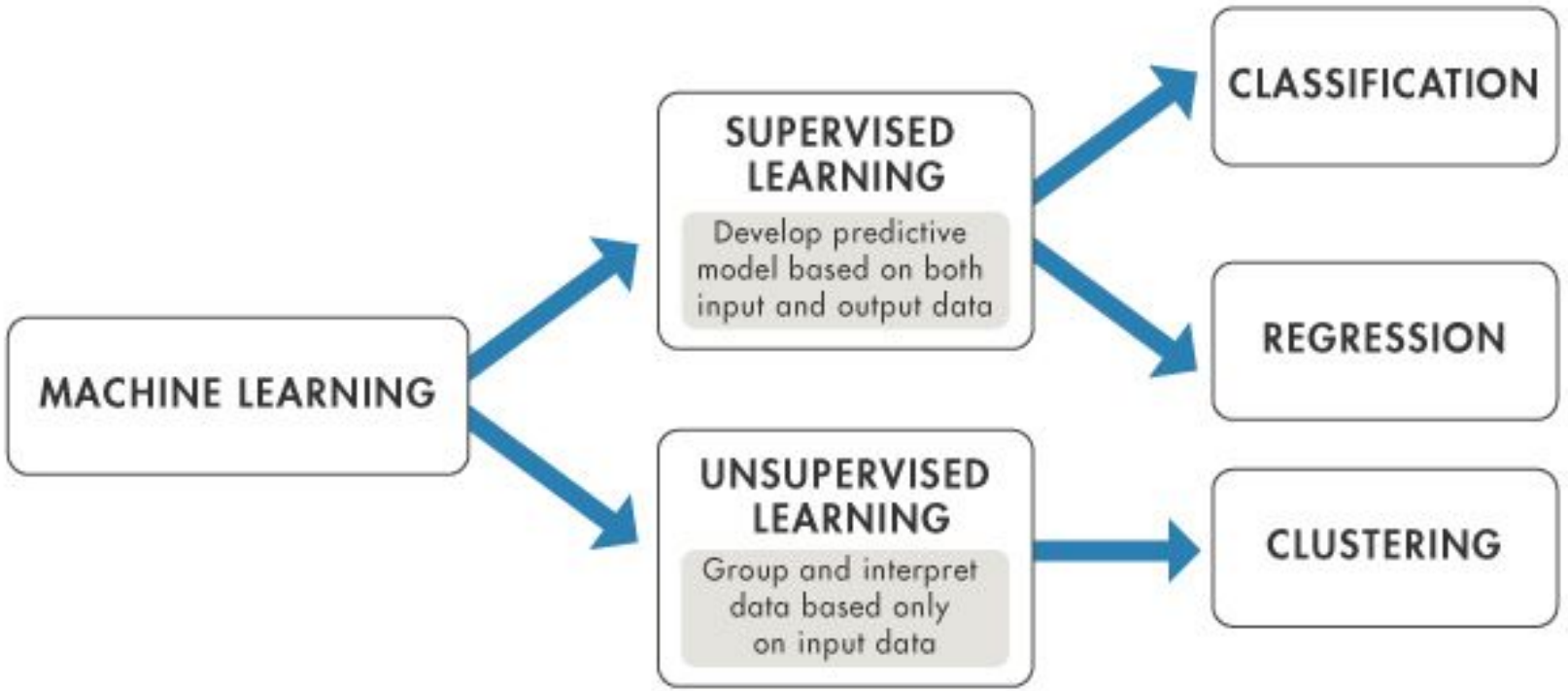


UNSUPERVISED LEARNING

Kristina Lerman
USC Information Sciences Institute
Spring 2025

Supervised vs unsupervised learning

Based on sample/training data and their given class labels or categories, is it possible to train a model that generalizes over unseen data to decide what class the sample belongs to?

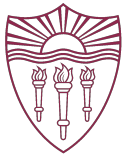


Output: **label**
- Binary
- Categorical

Output: **response/outcome**
- Binary
- Real

Output: **no output**
Input: Binary, Nominal, Real

Source: DeepAI.org

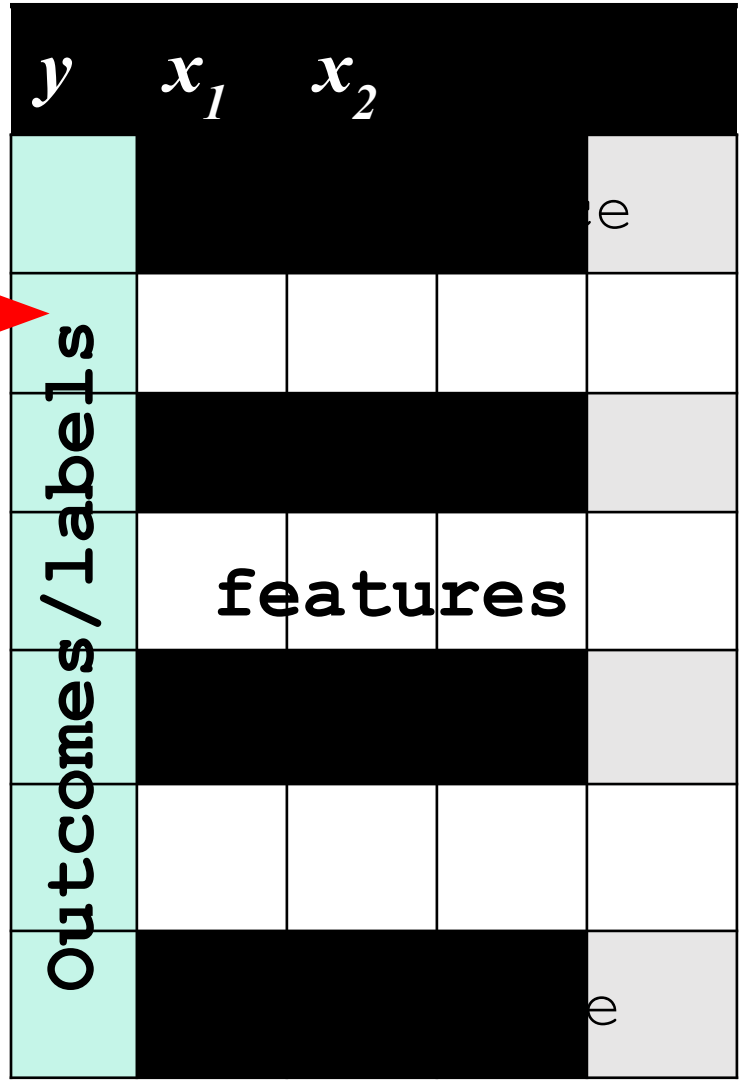


WHY UNSUPERVISED LEARNING?

Why unsupervised learning?

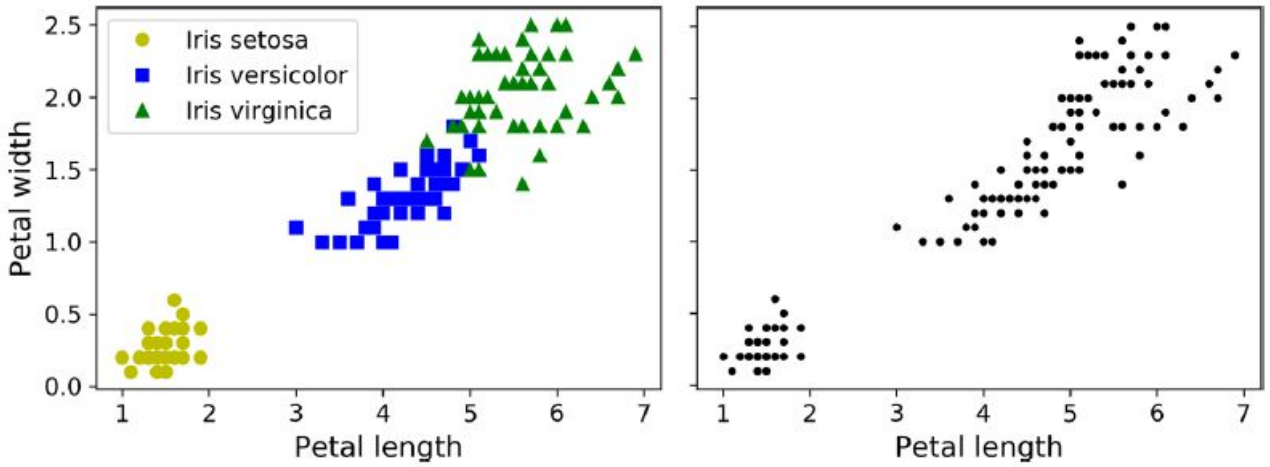
- Learning from unlabeled data
- Usually, no ground truth data, labels or outcomes are given

- "If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake." – Yann LeCun



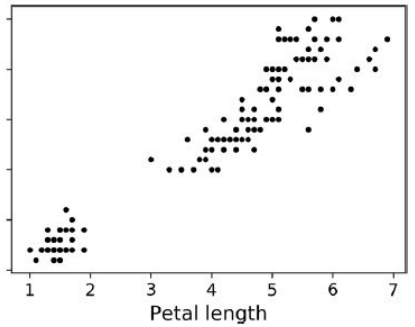
What is unsupervised learning?

- **Supervised learning**: e.g, Classification
- **Unsupervised learning**:
  - Just as in classification, each data point is assigned to a group/label
  - But groups/labels are not known
  - But we got data! (sometimes a lot!)
- And what can we find in the data?



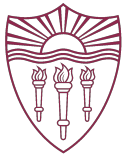
What is clustering?

- Want to find a ‘natural’ grouping between data instances
  - We want to find ‘similar’ instances and treat them in the same way
- No universal definition of what a good cluster is. Different algorithms capture different kinds of clusters.
  - Data points near a centroid
  - Dense regions
  - Well-separated regions



- Data instances within a cluster are closer to each other than to data points in different clusters
  - **High intra-cluster similarity**
  - **Low inter-cluster similarity**

Issues to consider

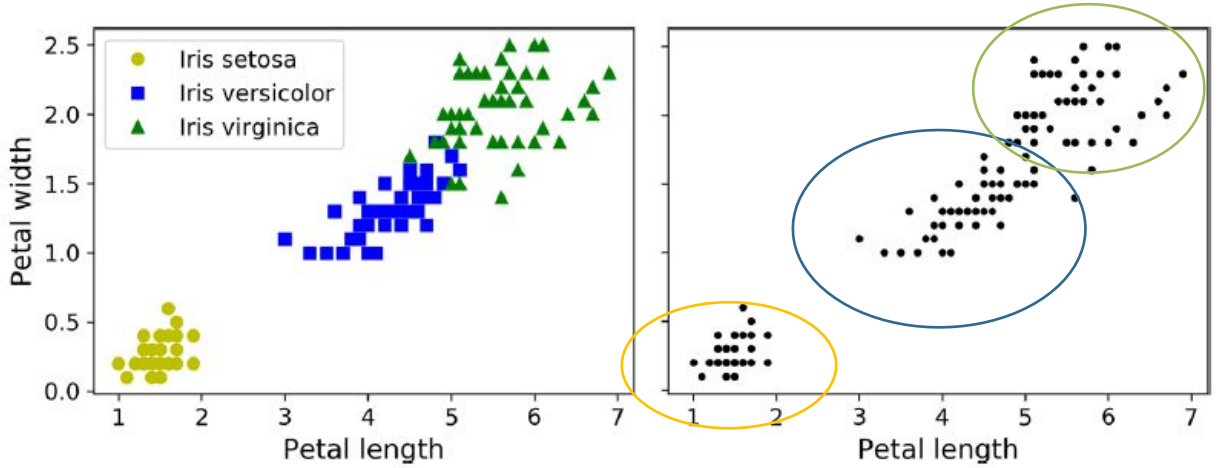


- Nature of clusters
  - Set of clusters or hierarchy of clusters?
  - Crisp or fuzzy?
  - Distant compact sets of items vs dense sets separated by low-density regions
  - How many clusters are there?
- Nature of items
  - Numeric? Categoric? High-dimensional?
  - If multi-dimensional, are values of a similar order of magnitude, or do some dominate?
  - How to measure similarity?
- Are the clusters valid?
  - External validity: e.g., prior knowledge of the problem
  - Internal validity: do the clusters fit the data well?

What does it mean for items to be **similar**?

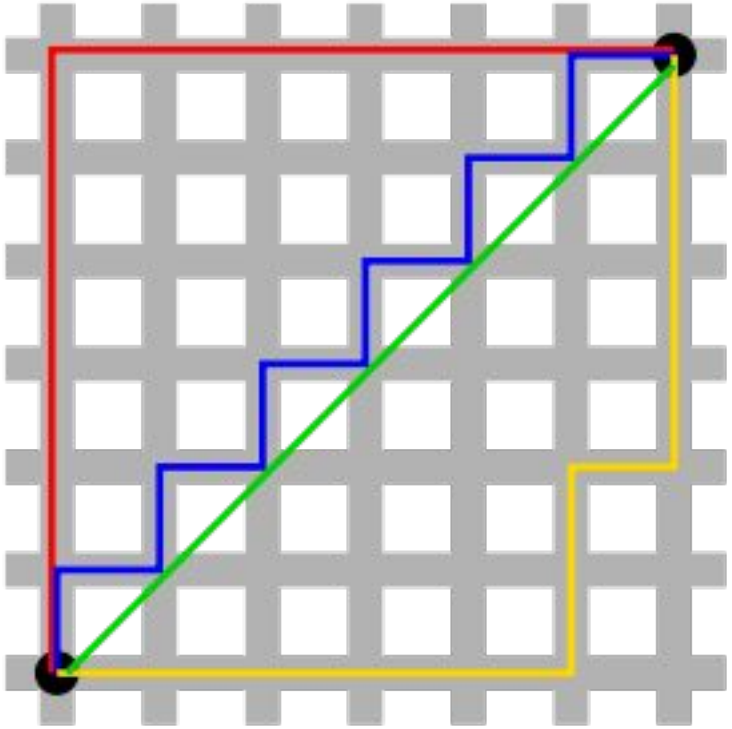
- How to define similarity is one of the most important questions in unsupervised learning.
- Similarity often measured using **distance measures** (e.g. Euclidean distance, or Mahalanobis Distance)

$$||x - y|| = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$$



Numeric Distance Metrics

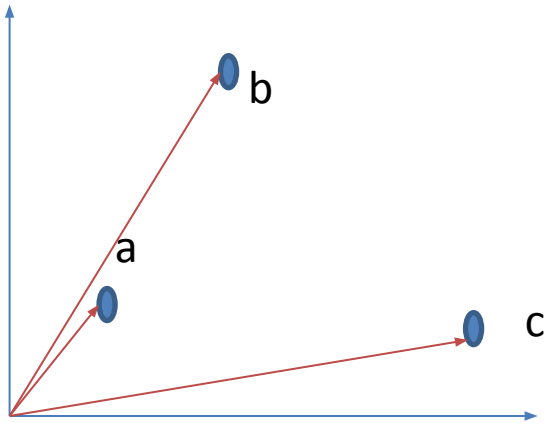
Names	Formula
Euclidean distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
squared Euclidean distance	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan distance	$\|a-b\|_1 = \sum_i \|a_i - b_i\|$
maximum distance	$\|a-b\|_\infty = \max_i \|a_i - b_i\|$
Mahalanobis distance	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ where S is the covariance matrix
cosine similarity	$\frac{a \cdot b}{\|a\|\|b\|}$



Cosine Distance

- Similarity = Cosine of angle btw vectors: A & B
  - Numerator is the **dot product** of vectors A and B
  - Denominator is the product of the **Euclidean distance** of each vector from the origin

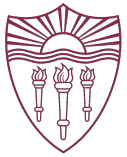
    (length of the vector)



- distance = 1- Cosine(A, B)

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Symbolic Distance Metrics

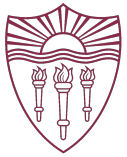


- **Hamming distance** between two symbolic strings of equal length is the number of positions at which the corresponding systems are different.
  - Measures the minimum number of substitutions required to change one string into the other
- **Levenshtein (edit) distance** is a metric for measuring the amount of difference between two sequences.
  - Is defined as the minimum number of edits needed to transform one string into the other.

1001001
1000100
HD = 3

LD(BIOLOGY, BIOLOGIA) = 2
BIOLOG**Y** -> BIOLOG**I** **(1 substitution)**
BIOLO G I -> BIOLOG**IA** **(1 insertion)**

Similarity measure must be a metric



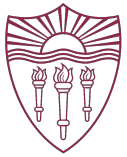
- You can choose any distance measure as similarity metric
- Distance metrics satisfy the following properties (axioms): Given a distance measure d(.,.)

Self-similarity: $d(a, a) = d(b, b), \quad \forall a, b \in X$

Minimality: $d(a, a) < d(a, b), \quad \forall a, b \in X, a \neq b$

Symmetry: $d(a, b) = d(b, a), \quad \forall a, b \in X$

Triangle inequality: $d(a, b) + d(b, c) >= d(a, c), \quad \forall a, b, c \in X,$



DATA STANDARDIZATION

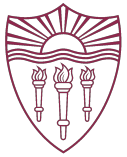
Data Standardization

- In the Euclidean space, standardization of features is recommended so that all attributes can have equal impact on the computation of distances.
- Consider the following pair of data points
  - $\mathbf{x}_i$: (0.1, 20) and $\mathbf{x}_j$: (0.9, 720).

$$dist(\mathbf{xi}, \mathbf{xj}) = \sqrt{(0.9 - 0.1)^2 + (720 - 20)^2} = 700.00,$$

- The distance is almost completely dominated by (720-20) = 700.
- Standardize attributes: force features to have a common range

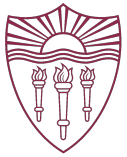
Interval-scaled attributes



- 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 throughout 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



- 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 s_f, is computed as follows

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

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

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

Ratio-scaled attributes

- Numeric attributes, but unlike interval-scaled features, their scales are exponential,
- For example, the total amount of microorganisms that evolve in a time *t* is approximately given by

  Ae^{Bt},

  where *A* and *B* are some positive constants.

•Do log transform:

– Then treat it as an interval-scaled feature

$$\log(x_{if})$$

Nominal features

- Sometimes, 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: one hot encoding

- Nominal attribute *food*: has three values,
  - Apple, Chicken, and Broccoli
  - We create three binary attributes in the new data: Apple, Chicken, and Broccoli
  - If a particular data instance in the original data has Apple as the value for *food*,
    - then in the transformed data, we set the value of the attribute Apple to 1, and the values of attributes Chicken and Broccoli to 0

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

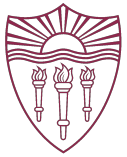
→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Ordinal attributes

- Ordinal attribute: an ordinal attribute is like a nominal attribute, but its values have a numerical ordering.
  - Age attribute with values: Young, MiddleAge and Old. They are ordered.
  - Common approach to standardization: treat is as an interval-scaled attribute.



HIERARCHICAL CLUSTERING

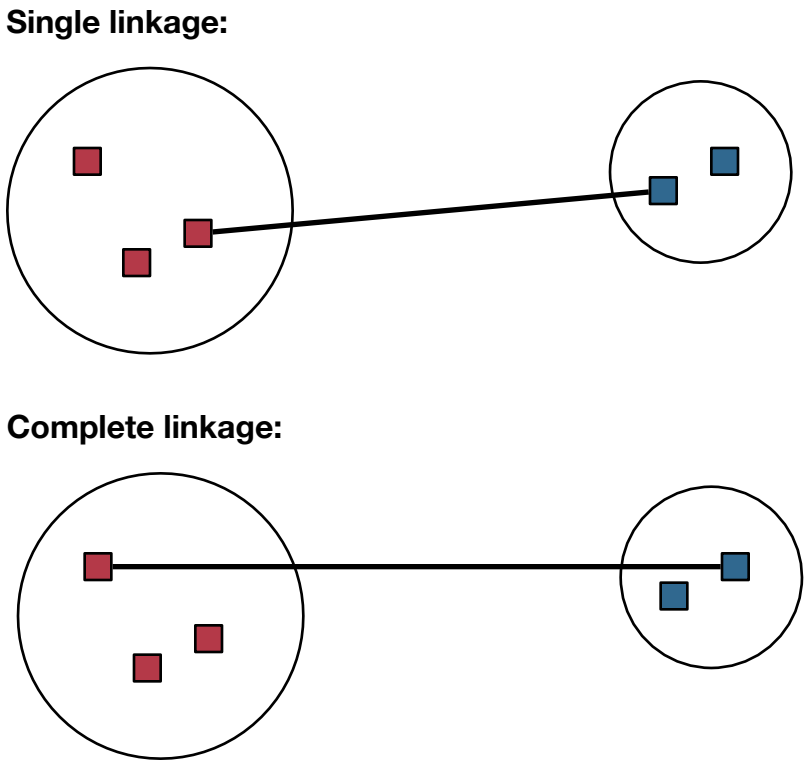
Hierarchical Clustering

- Two basic types of hierarchical clustering
  - Agglomerative clustering (bottom to top)
  - Divisive clustering (top to bottom)
- Based on a pre-defined distance measure and linkage criterion we split (divisive) and merge (agglomerative) clusters depending on the type
- Based on greedy search, therefore very slow and not scalable

- **Distance measures and linkage criteria have a big influence on the outcome of clusters!**

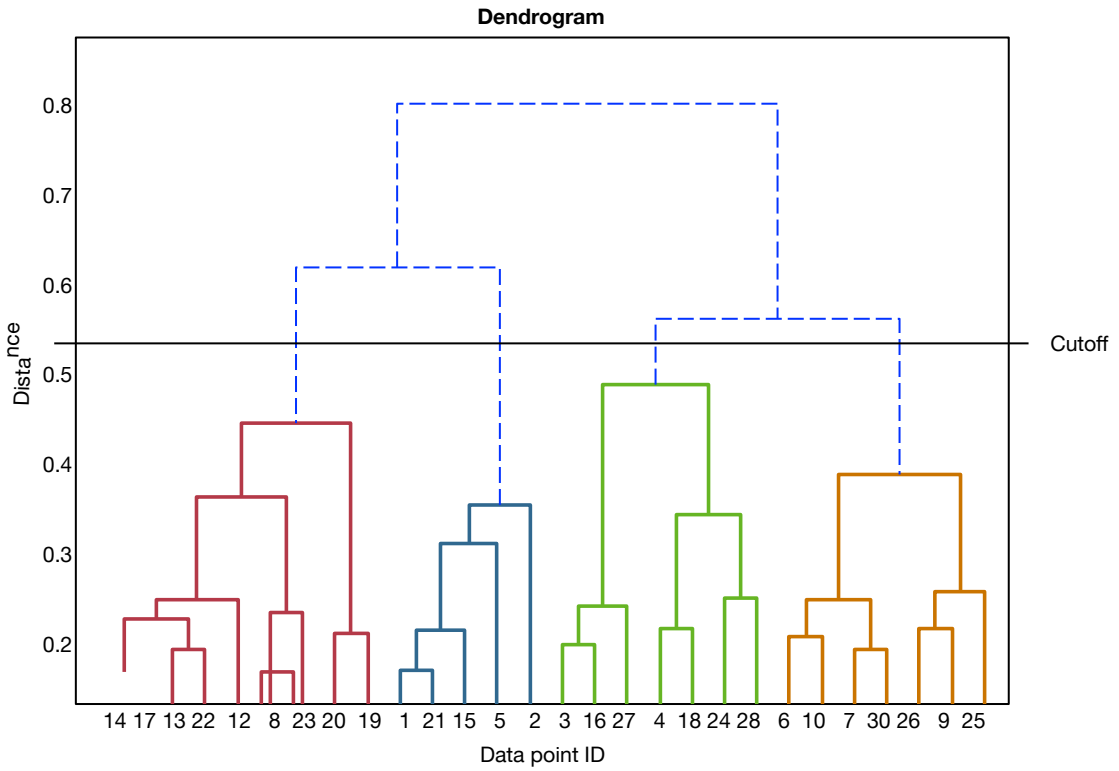
Hierarchical Clustering (2)

- Linkage types:
  - Single linkage (single minimal distance)
  - Complete linkage (maximal minimal distance)
  - Average linkage (minimal average distance)
- Different problems can occur. (e.g. chaining effects)



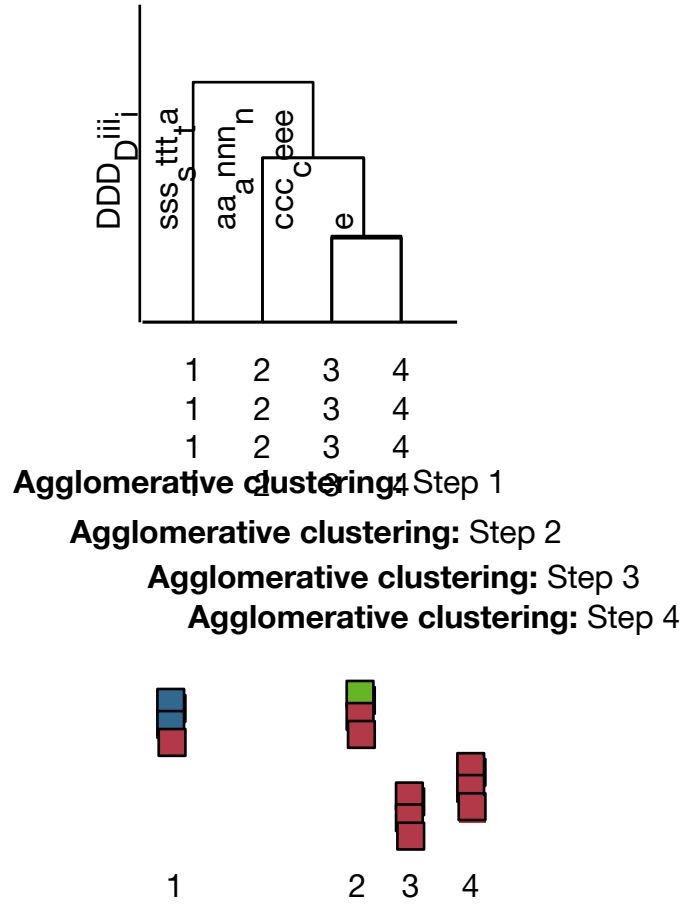
Dendrogram

- Tree diagrams
- Dendrograms are used to visualize distances and clusters
- Useful for analyzing hierarchical clusters with different cut-offs

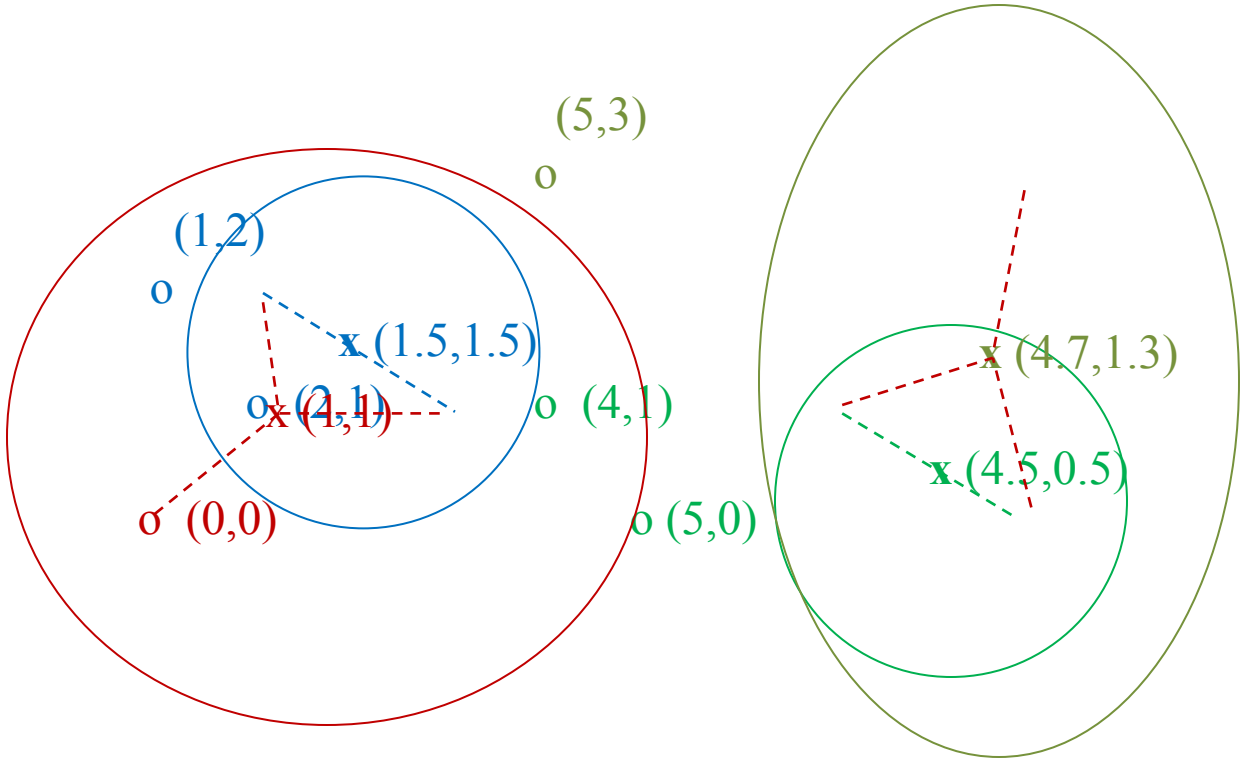


Agglomerative Clustering

- Algorithm:
  1. Start: each data point belongs to a separate cluster
  2. Step: merge closest clusters based on linkage and distance
  3. End: all data points belong to the same cluster



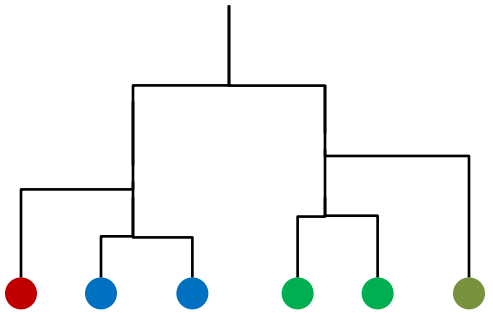
Example: Hierarchical clustering



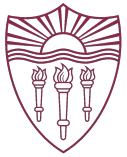
Data:

o … data point

x … centroid



Dendrogram



COMPETITIVE CLUSTERING

Competitive Clustering

- Base strategy of defining clusters with different variants
  - K-means clustering
  - Self-organizing maps
  - Affinity propagation, …
- Again, chosen distance measure is the key
- Based on the "winner takes it all" principle
- After enough training steps it is assumed that winners (i.e. cluster centroids) are good representations of data within their cluster.

$$j^* = \arg\min_j ||x - c_j||$$

k-Means Clustering

- Find *k* reference vectors (prototypes/codewords) which best represent data
- Reference vectors, $\boldsymbol{m}_j$, $j = 1,\ldots,k$
- Use nearest (most similar) reference:

$$\left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\|$$

- Reconstruction error

$$E\left(\{\mathbf{m}_i\}_{i=1}^k \middle| \mathrm{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - \mathbf{m}_i\right\|$$

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

k-means Clustering

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

Repeat

Assign to centroid

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

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

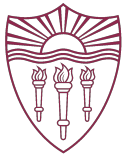
Update centroid

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

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

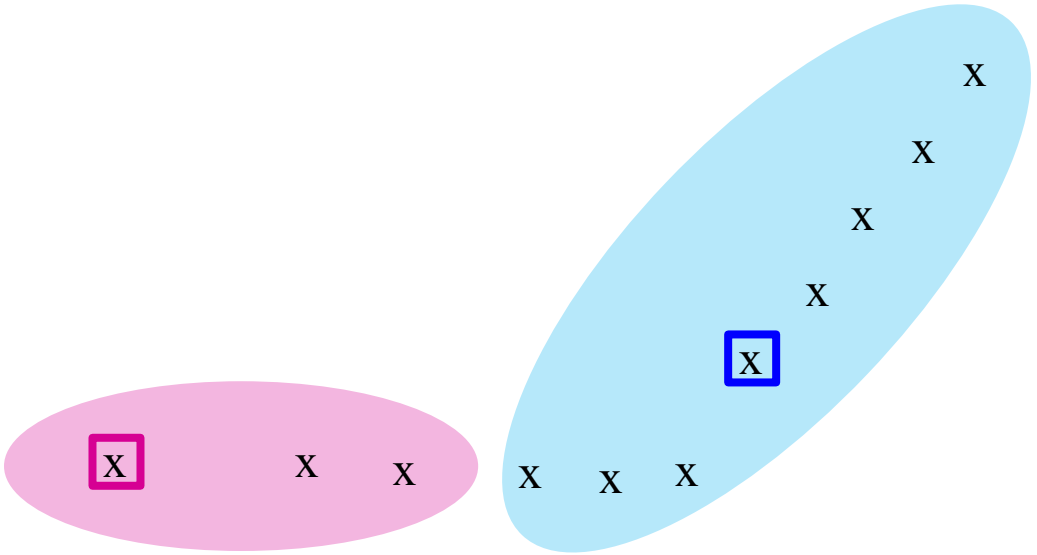
Until $\boldsymbol{m}_i$ converge

Populating Clusters



- **1)** For each point, **place it in the cluster whose current centroid it is nearest**
- **2) After all points are assigned, update the locations of centroids of the *k* clusters**
- **3) Reassign all points to their closest centroid**
  - Sometimes moves points between clusters
- **Repeat 2 and 3 until convergence**
  - **Convergence:** Points don't move between clusters and centroids stabilize.

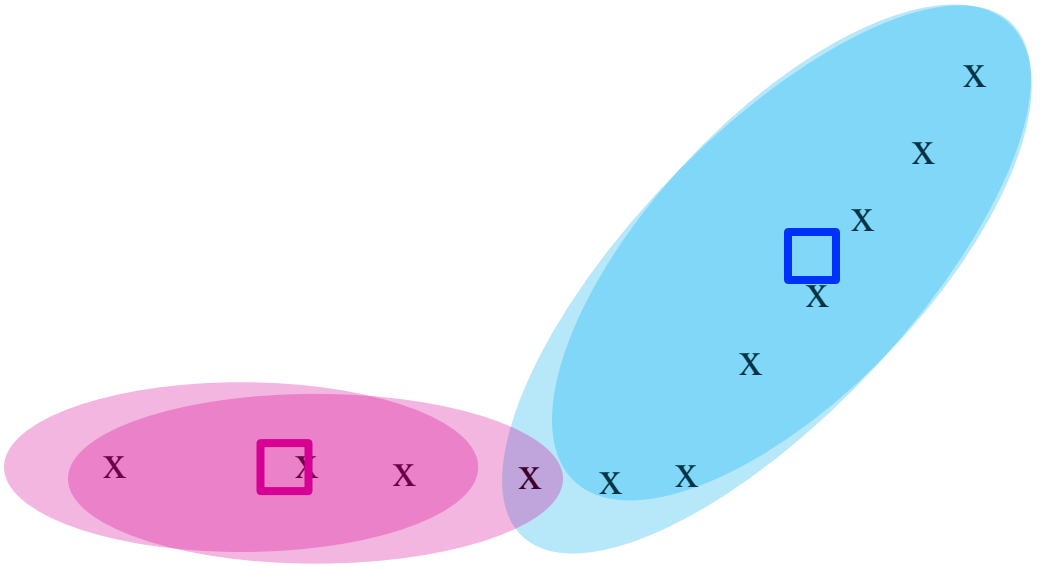
Example: Assigning Clusters



x … data point
□ … centroid

Clusters after round 1

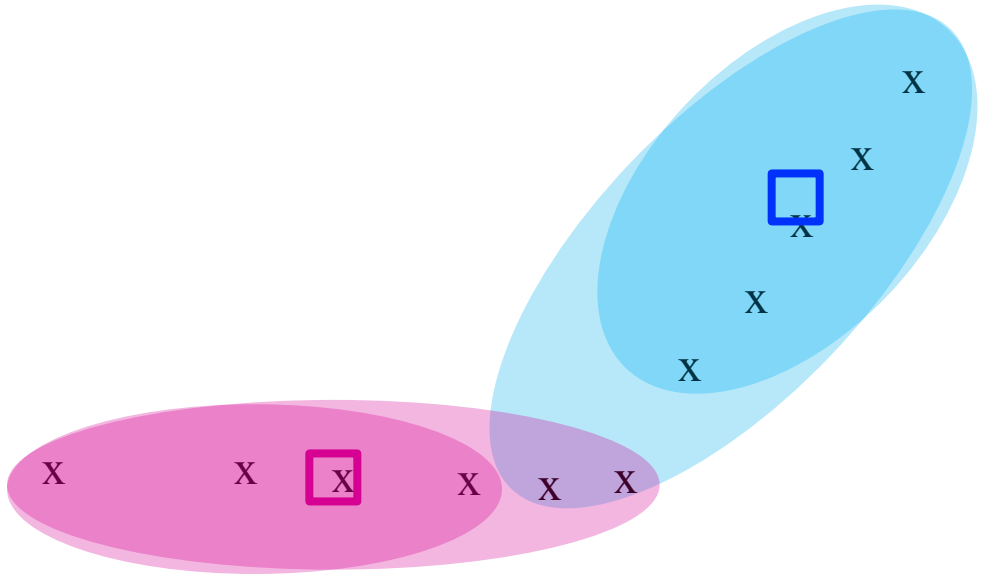
Example: Assigning Clusters



x … data point
□ … centroid

Clusters after round 2

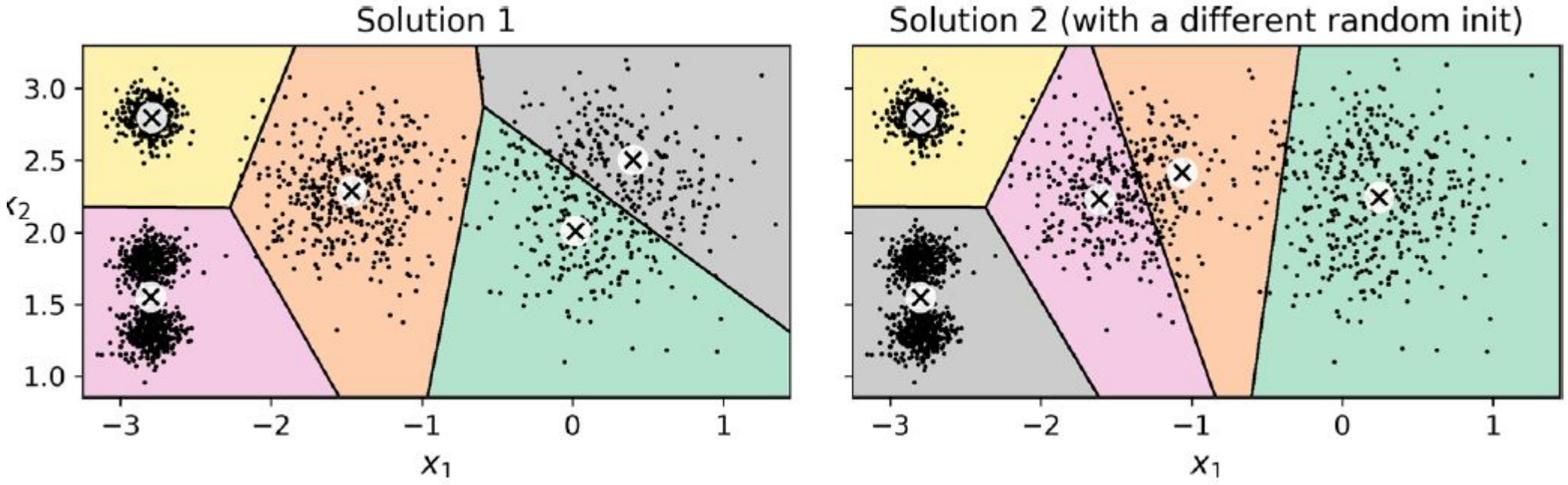
Example: Assigning Clusters



x … data point
□ … centroid

Clusters at the end

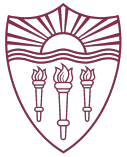
Watch out for local minima!



Convergence

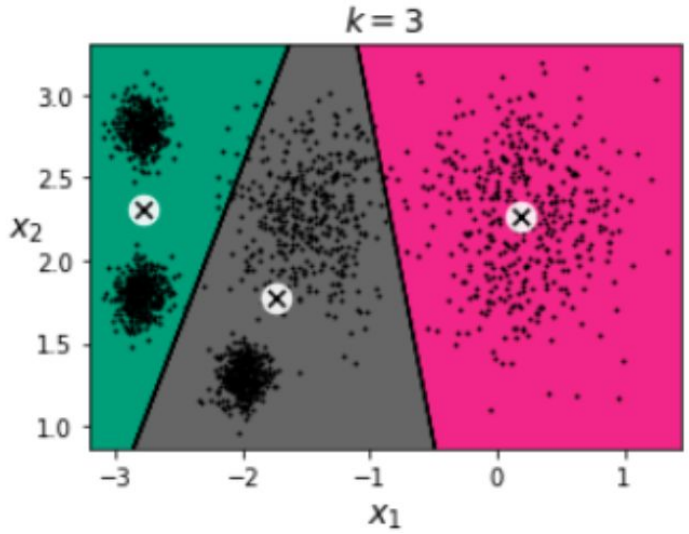
- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Data partition is unchanged.
  - Centroid positions don't change convergence
- Does k-means always converge? i.e., reach a state in which clusters don't change.
  - Yes. It does, as it minimizes the overall distortion towards a local optimum.
  - How to find a better minimum?
    - Solution: e.g. run it multiple times
- K-means is a special case of a general procedure known as the Expectation Maximization (EM) algorithm.

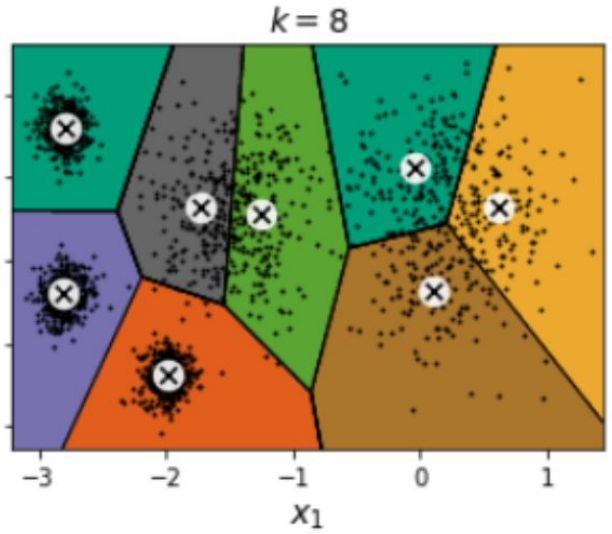
Computational complexity



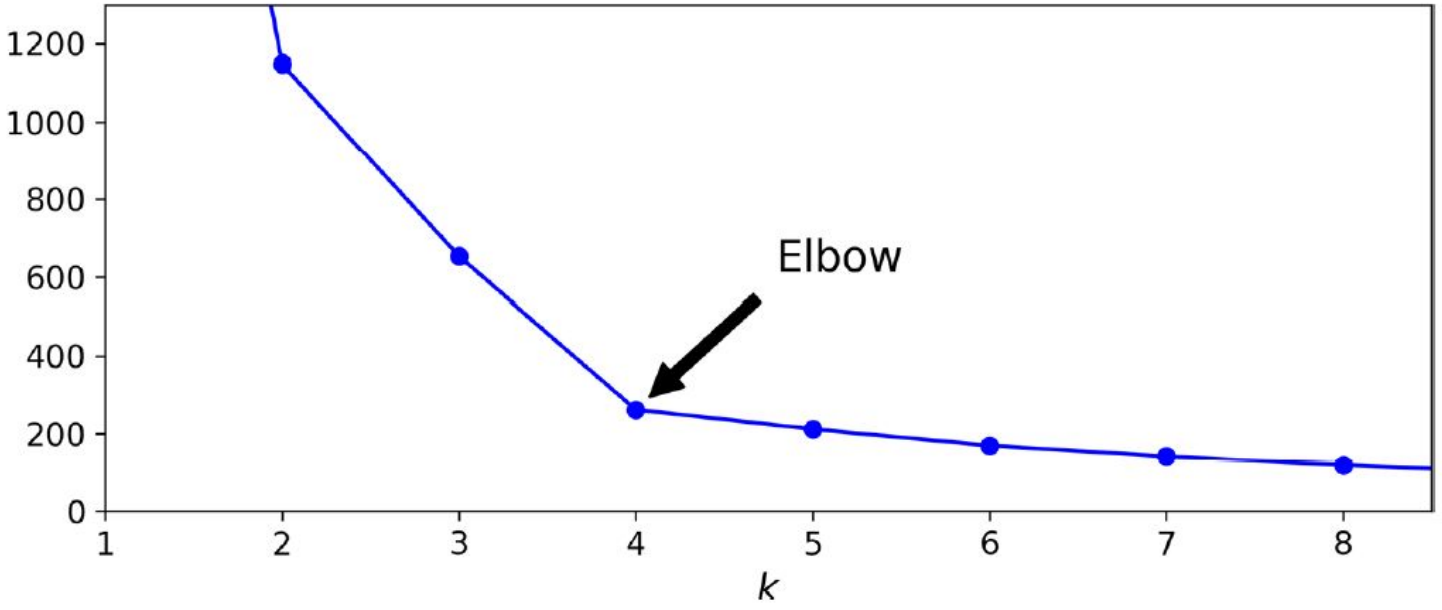
- The computational complexity of the algorithm is generally linear with the number of instances *N*, the number of clusters *K*, and the number of dimensions *m*.
- However, this is only true when the data has a clustering structure. If it does not, then the complexity can increase exponentially with the number of instances.
- This is rare, and K-Means is generally one of the fastest clustering algorithms.

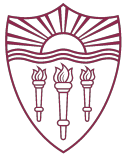
How many clusters?





Mean squared distance

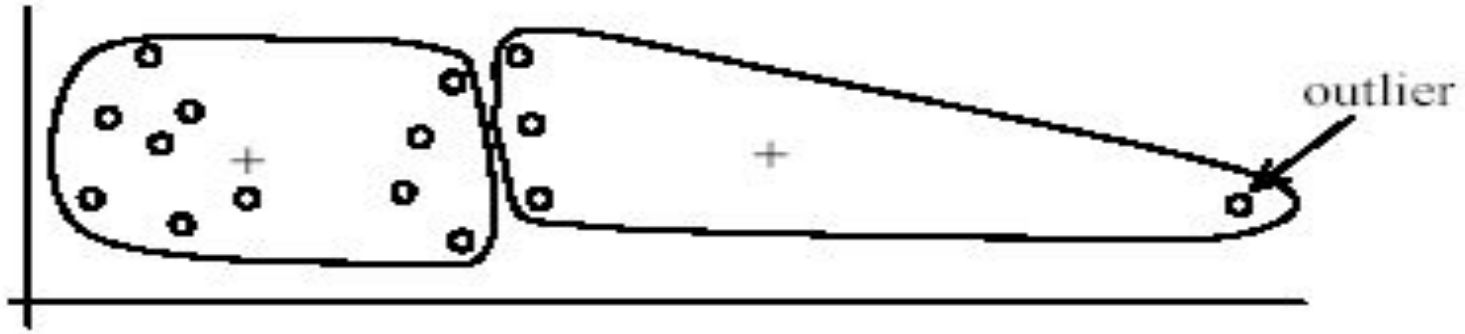




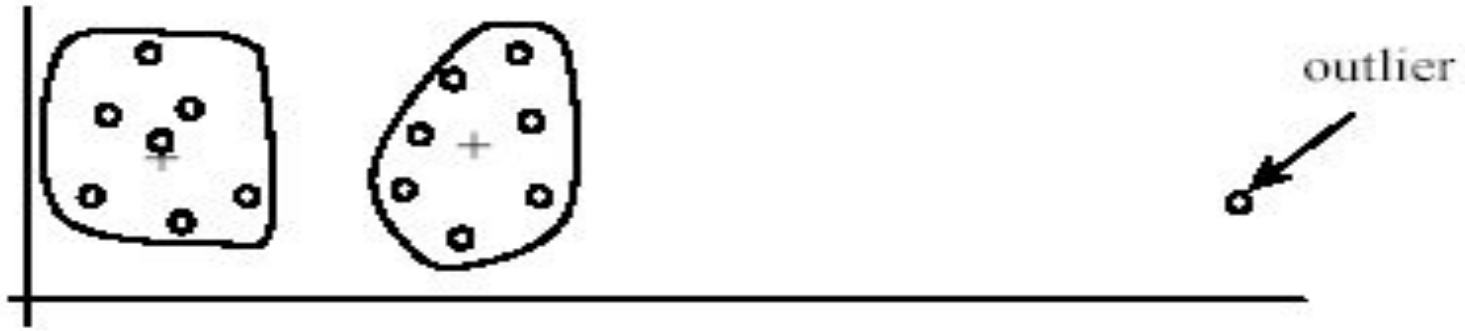
Weaknesses of K-Means

- 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

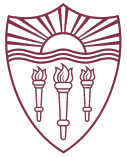


(A): Undesirable clusters



(B): Ideal clusters

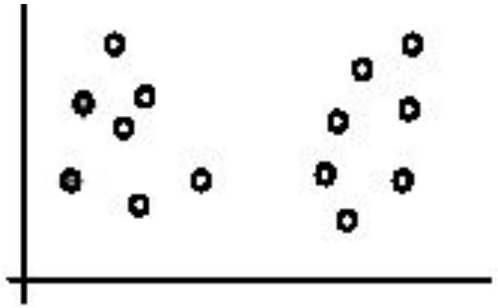
To deal with outliers



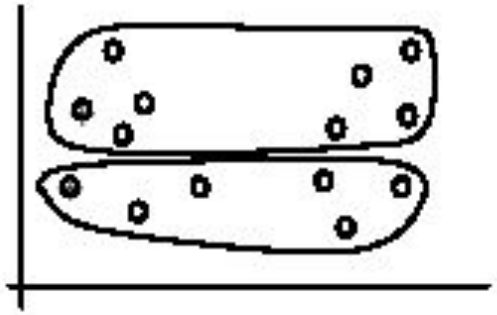
- One method is to remove some data points in the clustering process that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Another method is to perform random sampling
  - 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

Weaknesses of k-means

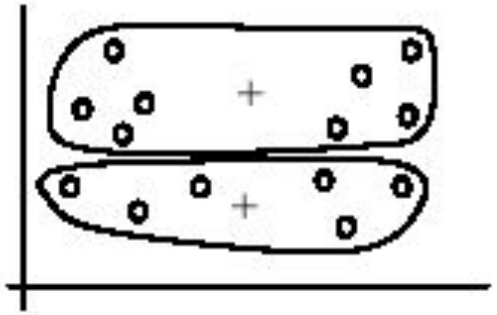
- The algorithm is sensitive to initial seeds



(A). Random selection of seeds (centroids)



(B). Iteration 1

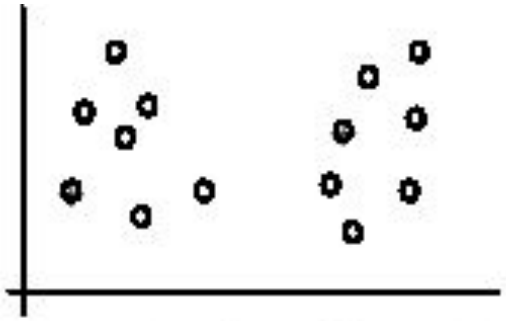


(C). Iteration 2

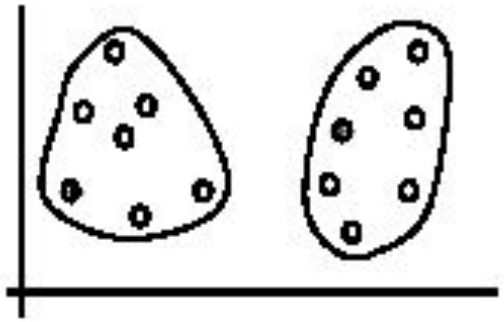
Weaknesses of k-means (cont …)

- If we use different seeds: good results

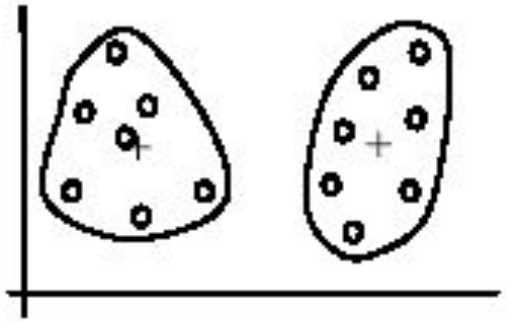
There are some methods to help choose good seeds



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



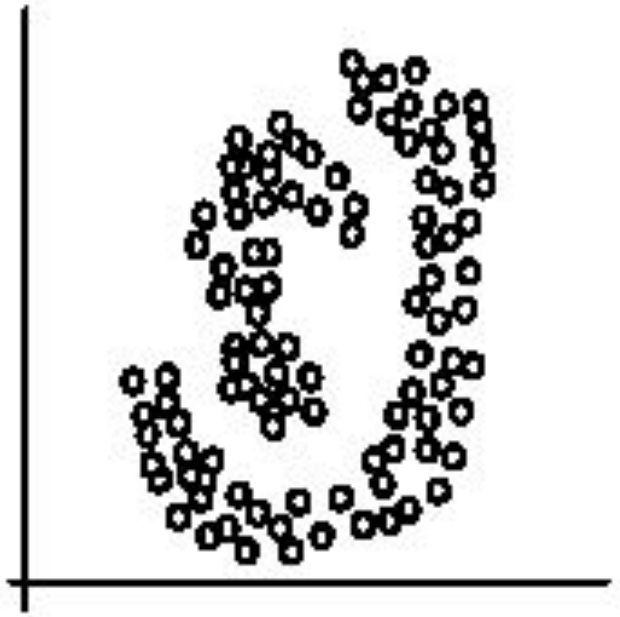
(B). Iteration 1



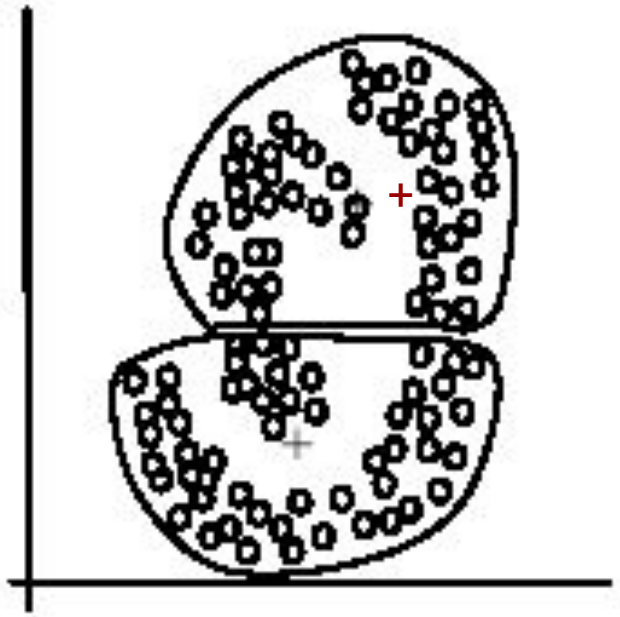
(C). Iteration 2

Weaknesses of k-means (cont …)

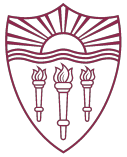
The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres)



(A): Two natural clusters

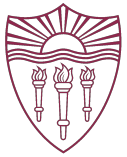


(B): *k*-means clusters



CLUSTER EVALUATION

Cluster Evaluation: a hard problem

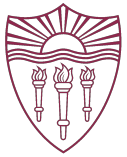


- The quality of a clustering is very hard to evaluate because we do not know the correct clusters

•Some methods are used:

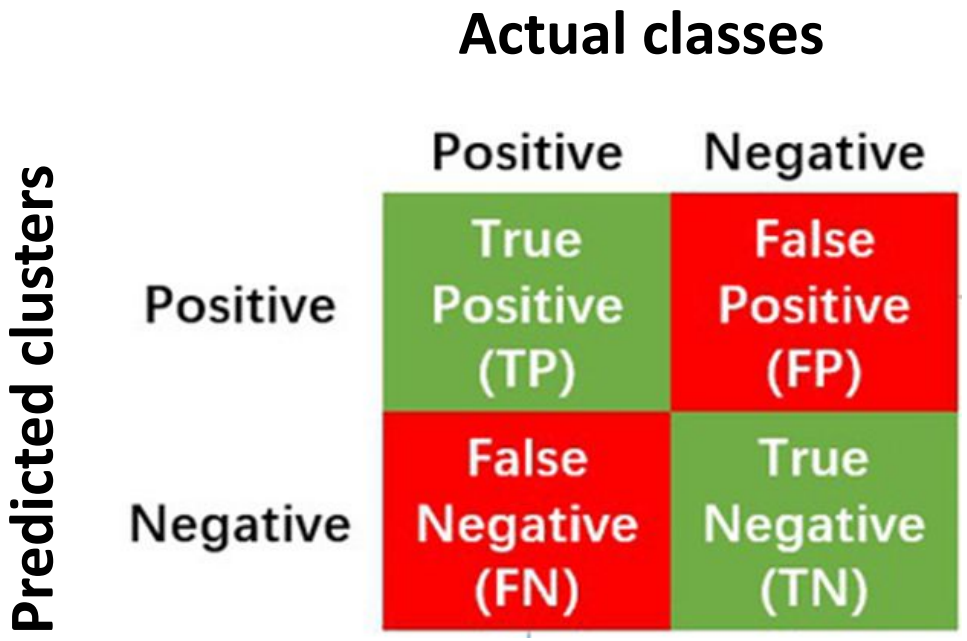
  - User inspection
  - Study centroids, and spreads
  - Entropy, Purity
  - Silhouette score

Cluster evaluation: ground truth



- We use some labeled data (for classification)
- Assumption: Each class is a cluster.
  - After clustering, a confusion matrix is constructed. From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
    - Let the classes in the data D be $C = (c_1, c_2, \ldots, c_k)$. The clustering method produces k clusters, which divides D into k disjoint subsets, $D_1, D_2, \ldots, D_k$.

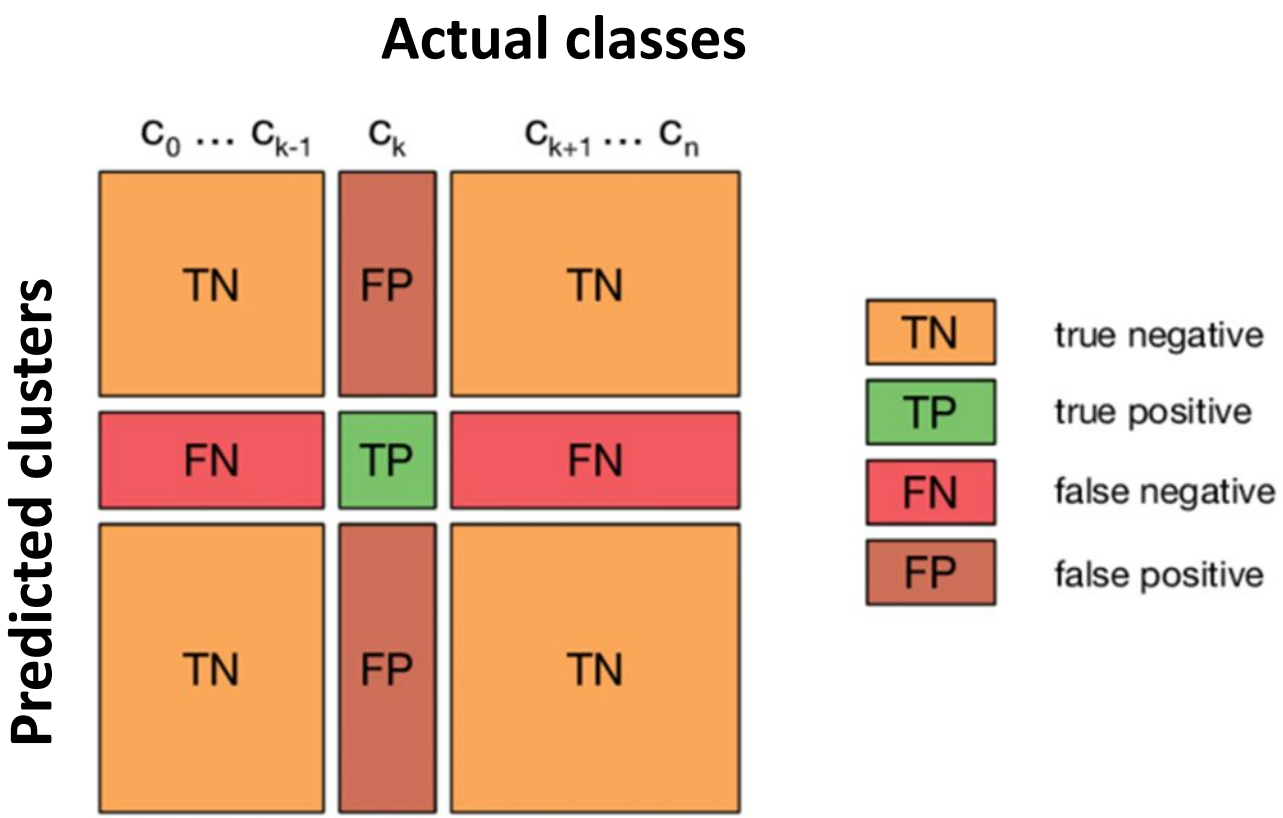
Confusion matrix



- Precision: PR = TP/(TP+FP)
- Recall: Re = TP/(TP+FN)
- Accuracy = (TP+TN)/(TP+FP+TN+FN)

- F1=2*Pr*Re/(Pr+Re)

Multi-class Confusion Matrix



- **Micro F1** calculates TP, TN, … metrics globally by aggregating the contributions of all classes, and calculates precision and recall using global totals
- Micro F1 is more suitable when class imbalance is not significant or when each individual instance is equally important.
- **Macro F1** calculates the F1 score for each class independently and then averages them. Each class contributes equally, regardless of its size.
- more suitable when dealing with imbalanced data, as it gives equal weight to all classes.

Example

Class	True Positives	False Positives	False Negatives
Class A	50	10	5
Class B	30	20	10
Class C	10	5	25

Micro F1

- Total TP = 50+30+10=90
- Total FP = 10+20+5=35
- Total FN = 5+10+25=40
- Precision=90/(90+35)
- Recall=90/(90+40)
- Micro F1=2*Precision*Recall/(Precision+Recall)

Macro F1

- Calculate Precision, Recall, and F1 for each class
- Average the F1 scores across classes

Evaluation measures: Entropy

Entropy: For each cluster, we can measure its entropy as follows:

$$entropy(D_i) = -\sum_{j=1}^{k} \Pr_i(c_j) \log_2 \Pr_i(c_j), \quad (29)$$

where $\Pr_i(c_j)$ is the proportion of class c_j data points in cluster i or D_i. The total entropy of the whole clustering (which considers all clusters) is

$$entropy_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times entropy(D_i) \quad (30)$$

Evaluation measures: purity

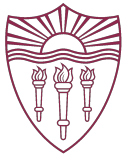
Purity: This again measures the extent that a cluster contains only one class of data. The purity of each cluster is computed with

$$purity(D_i) = \max_j(\Pr_i(c_j)) \tag{31}$$

The total purity of the whole clustering (considering all clusters) is

$$purity_{total}(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times purity(D_i) \tag{32}$$

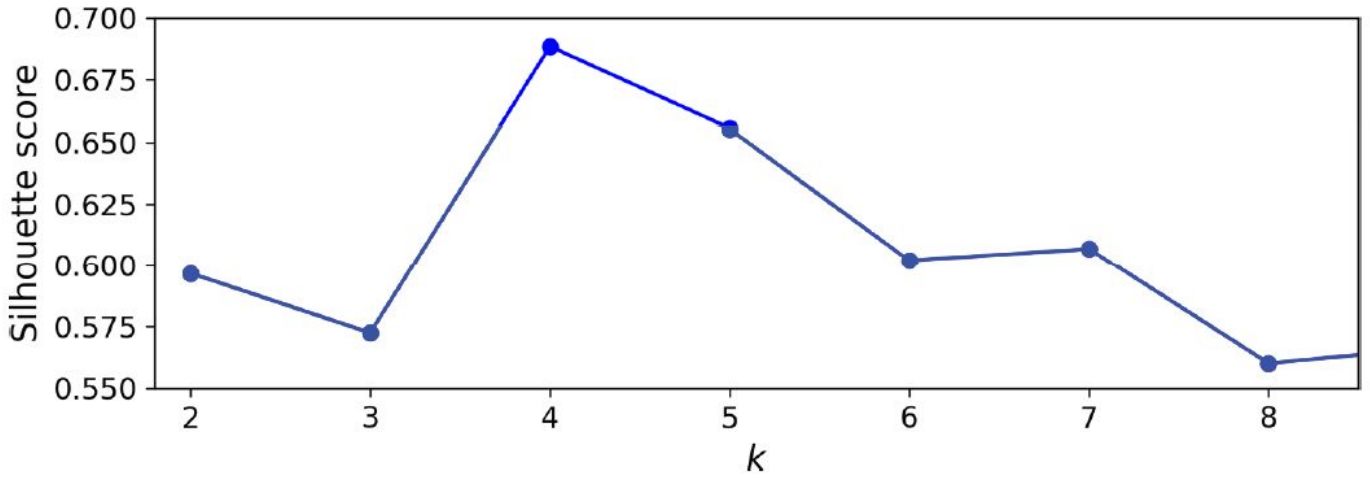
Evaluation based on internal information



- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how similar to each other the data points in the same cluster are
  - i.e., how near they are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.

Silhouette score

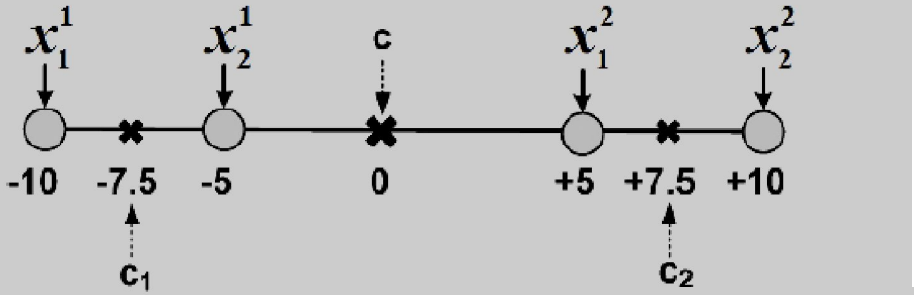
- Silhouette score measures how similar an instance is to its own cluster (cohesion) compared to other clusters (separation) and ranges from −1 to +1.
- If most instances have a high value, then the clustering configuration is appropriate. If many points have negative values, then there are too many or too few clusters.



Cohesiveness

- Cohesiveness
  - In statistical terms, this is equivalent to having a small standard deviation, i.e., being close to the mean value.
  - In clustering, this translates to being close to the centroid of the cluster

$$cohesiveness = \sum_{i=1}^{k} \sum_{j=1}^{n(i)} dist(x_j^i, c_i)^2$$



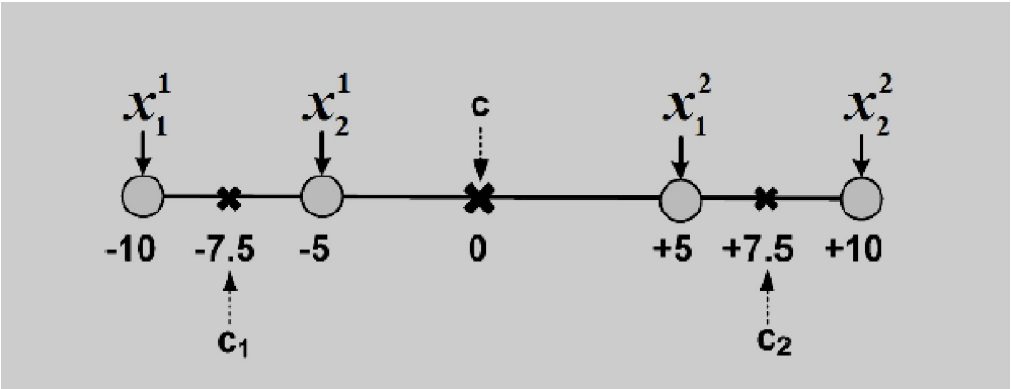
$$cohesiveness = |-10-(-7.5)|^2+|-5-(-7.5)|^2+|5-7.5)|^2+|10-7.5|^2 = 25. \quad (5.59)$$

Separateness

- Separateness
  - We are also interested in clusterings with clusters that are well separated from one another.
  - In statistics, separateness can be measured by standard deviation.
  - Standard deviation is maximized when instances are far from the mean.
  - In clustering terms, this is equivalent to cluster centroids being far from the mean of the entire dataset

$$separateness = \sum_{i=1}^{k} dist(c, c_i)^2$$

Separateness Example



$$separateness = |-7.5-0|^2 + |7.5-0|^2 = 112.5.$$

- In general we are interested in clusters that are both cohesive and separate -> Silhouette index

Silhouette Index

- For any instance x that is a member of cluster C

- Compute the within-cluster average distance

$$a(x) = \frac{1}{|C| - 1} \sum_{y \in C, y \neq x} ||x - y||^2.$$

- Compute the average distance between x and instances in cluster G that is closest to x in terms of the average distance between x and members of G

$$b(x) = \min_{G \neq C} \frac{1}{|G|} \sum_{y \in G} ||x - y||^2.$$

Silhouette Index

- Clearly we are interested in clusterings where a(x)<b(x)

$$s(x) = \frac{b(x) - a(x)}{\max(b(x), a(x))},$$

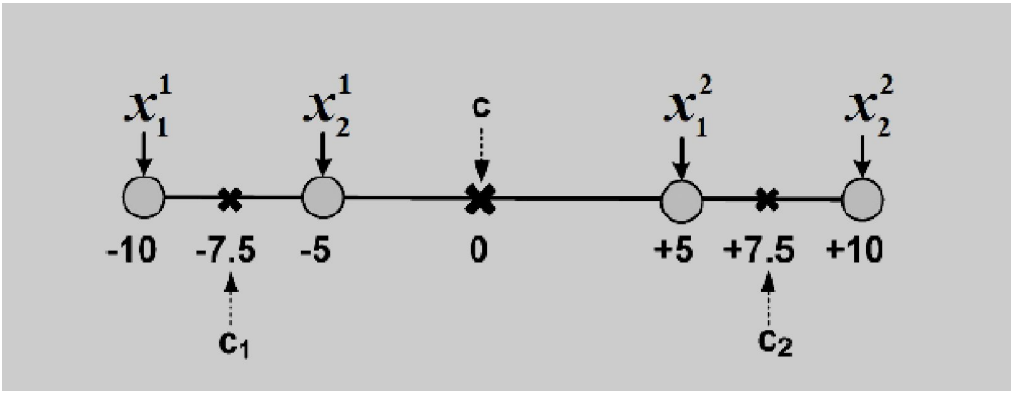
$$silhouette = \frac{1}{n} \sum_{x} s(x).$$

- Silhouette can take values between [-1,1]
- The best case happens when for all x,
  - a(x)=0, b(x)>a(x)

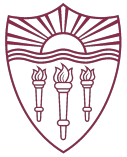
Silhouette Index - Example

$$a(x_1^1) = |-10-(-5)|^2 = 25$$
$$b(x_1^1) = \frac{1}{2}(|-10-5|^2 + |-10-10|^2) = 312.5$$

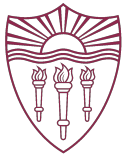
$$s(x_1^1) = \frac{312.5-25}{312.5} = 0.92$$
$$a(x_2^1) = |-5-(-10)|^2 = 25$$
$$b(x_2^1) = \frac{1}{2}(|-5-5|^2 + |-5-10|^2) = 162.5$$
$$s(x_2^1) = \frac{162.5-25}{162.5} = 0.84$$
$$a(x_1^2) = |5-10|^2 = 25$$
$$b(x_1^2) = \frac{1}{2}(|5-(-10)|^2 + |5-(-5)|^2) = 162.5$$
$$s(x_1^2) = \frac{162.5-25}{162.5} = 0.84$$
$$a(x_2^2) = |10-5|^2 = 25$$
$$b(x_2^2) = \frac{1}{2}(|10-(-5)|^2 + |10-(-10)|^2) = 312.5$$
$$s(x_2^2) = \frac{312.5-25}{312.5} = 0.92.$$



Other clustering methods

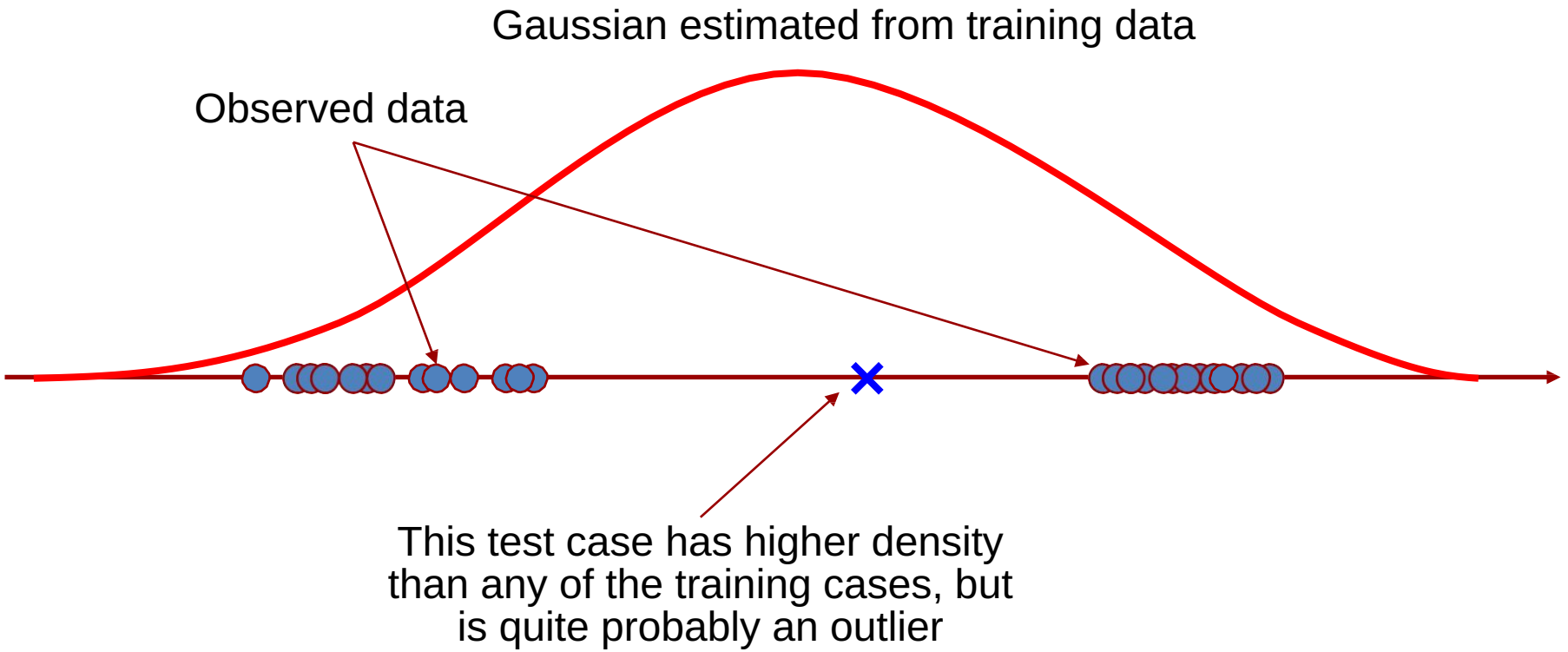


- DBSCAN
  - Clusters as continuous regions of high density. Good for spatial clustering
  - Can identify any number of clusters of any shape (with 2 hyperparameters)
  - Robust to outliers
- Affinity propagation
  - Every data point votes for another data point as its representative
  - After convergence, each representative and its voters form a cluster
  - Computationally expensive
- Spectral clustering
  - Creates a low-dimensional embedding from the similarity matrix
  - Can capture complex cluster structures
  - Efficient approximate methods

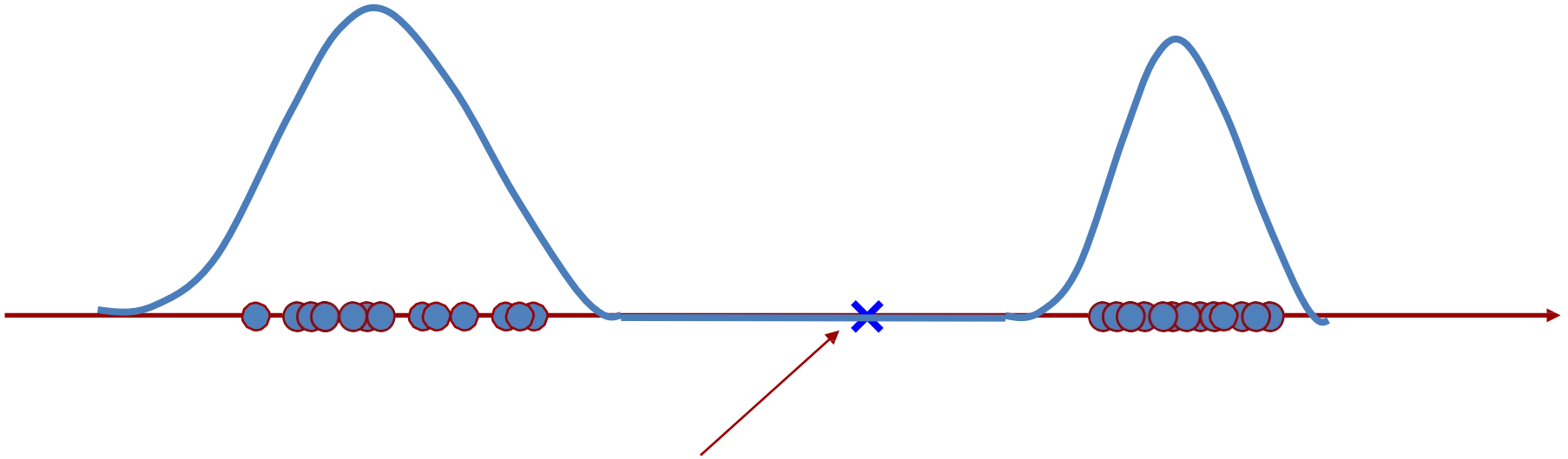


GAUSSIAN MIXTURE MODELS

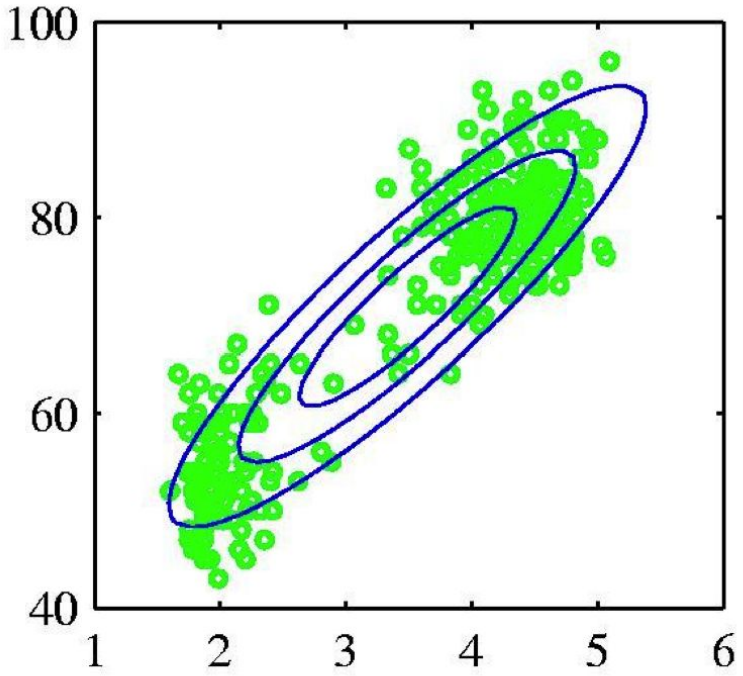
Illustration



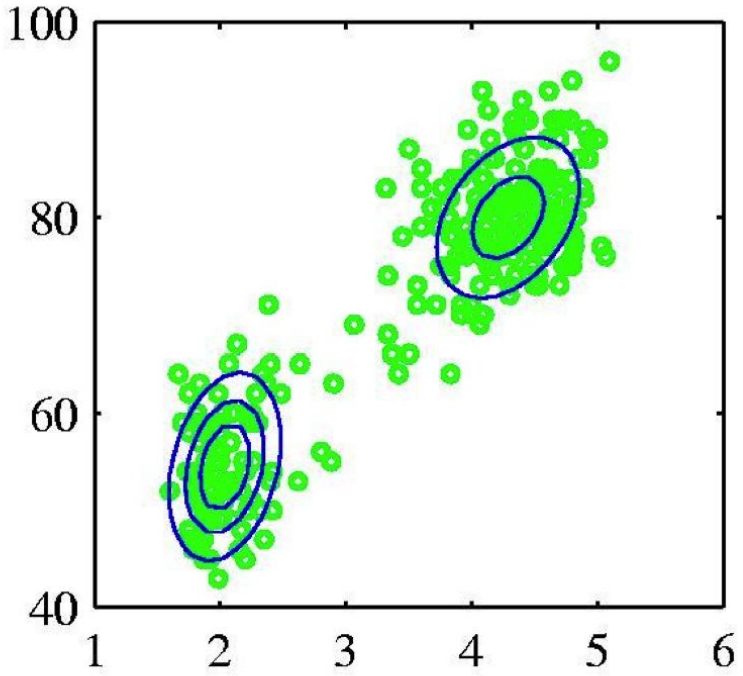
Illustration



Mixture of Gaussians



Single Gaussian



Mixture of two Gaussians

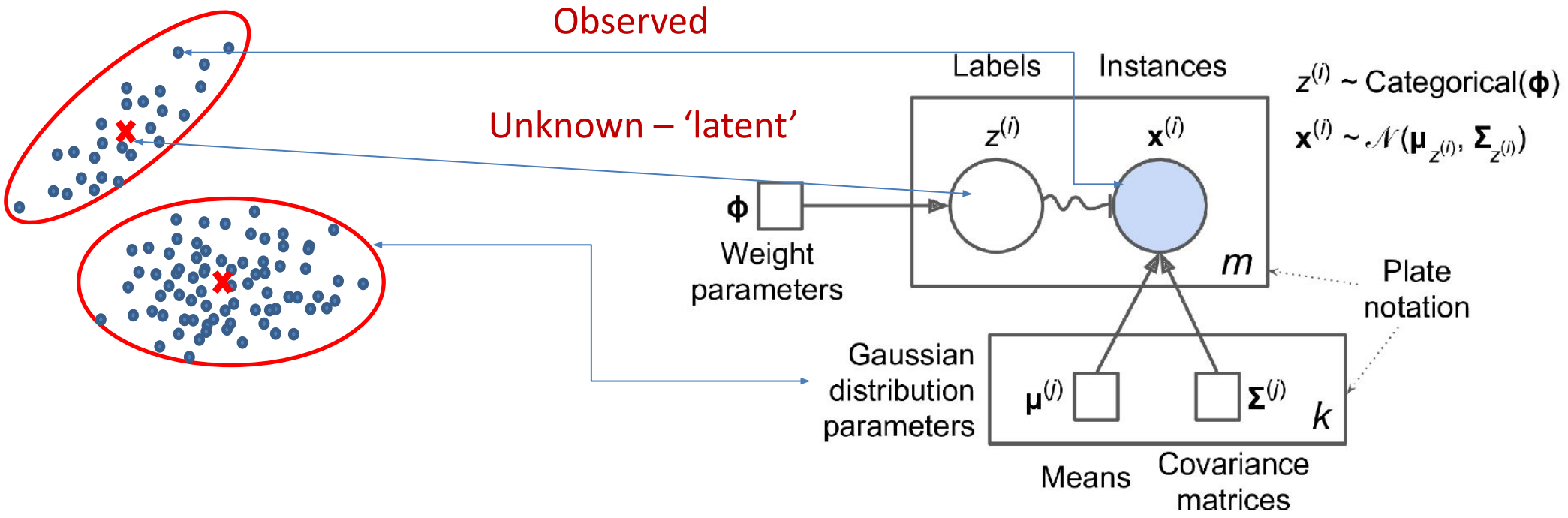
- Any data can be represented by a mixture of Gaussian clusters
- Each cluster can have a different ellipsoidal shape, size, density, and orientation

Mixture Densities

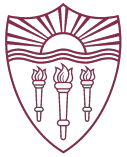
- Mixture Density: $p(\boldsymbol{x}) = \sum_{i=1}^{k} p(\boldsymbol{x}|\mathcal{G}_i)P(\mathcal{G}_i)$
- Component Density: $p(\boldsymbol{x}|\mathcal{G}_i)$
- Mixture Proportion: $P(\mathcal{G}_i)$

- **Learning:** estimating *component densities* and *proportions*.
- k is a hyper parameter.

What are we learning?



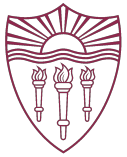
Gaussian mixture models



- **Expectation-Maximization algorithm** (EM algorithm): Iterative algorithm that finds the model parameters for which the observed data is most likely
  - Parameters: mean and covariance
    - **E-step:** Estimate labels/responsibilities for data given the parameters.
    - **M-step:** Update parameters given the estimated labels of E-step.
  - Repeat until convergence.
- K-means is a special case of EM algorithm
- **Issues:** Can end up in local minima, number of mixtures again is to be chosen and validated

Summary

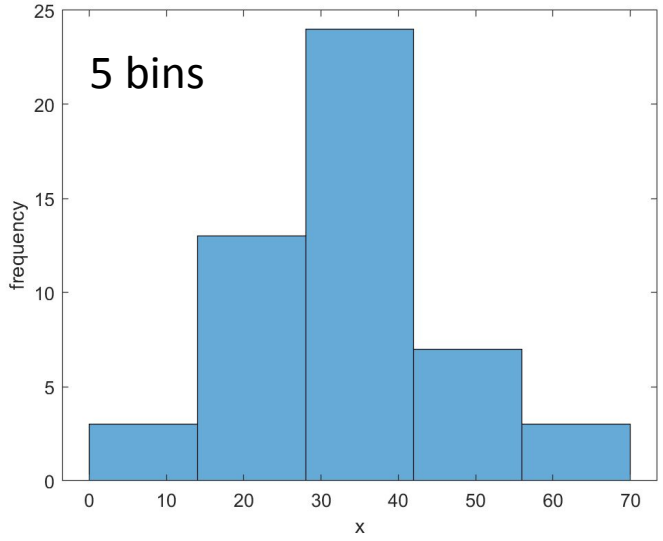
- Why learning from unlabeled data is important?
- Clustering algorithms
  - Hierarchical clustering
  - K-means
  - Gaussian mixture models
  - Expectation Maximization
- Non-parametric methods
  - Probability density estimation
- Dimensionality reduction/data embedding

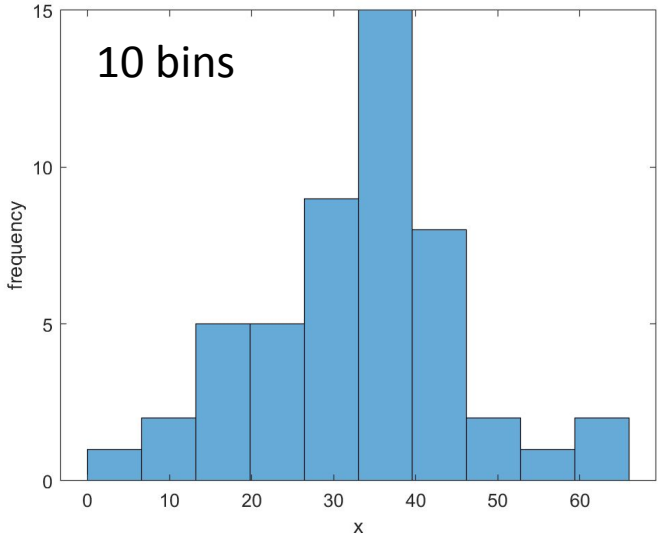


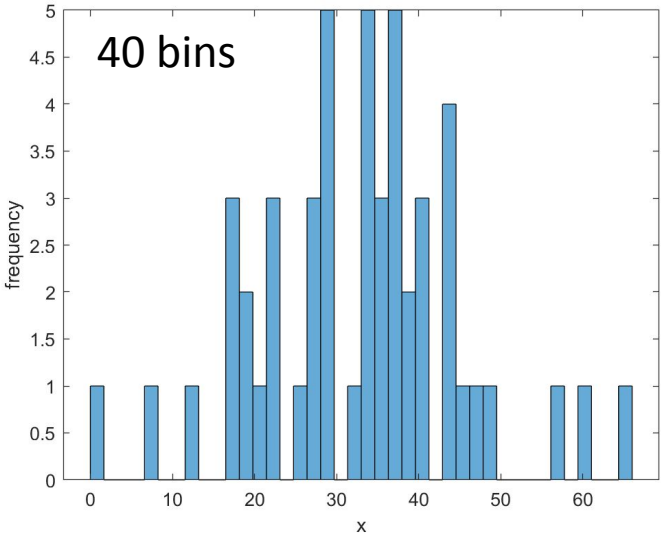
NONPARAMETRIC ESTIMATION

Histogram

- Histogram is the most popular way to visualize data
- Bin size changes the shape of the histogram
- For normal data, use equal size bins

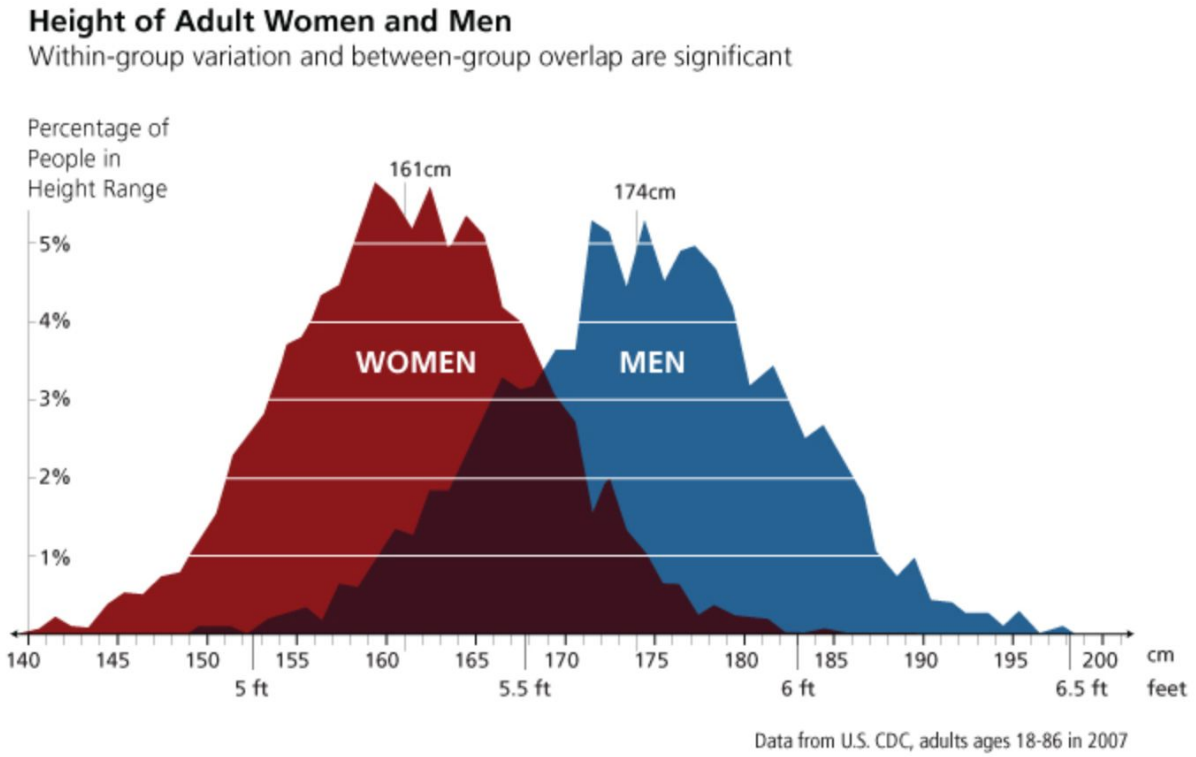




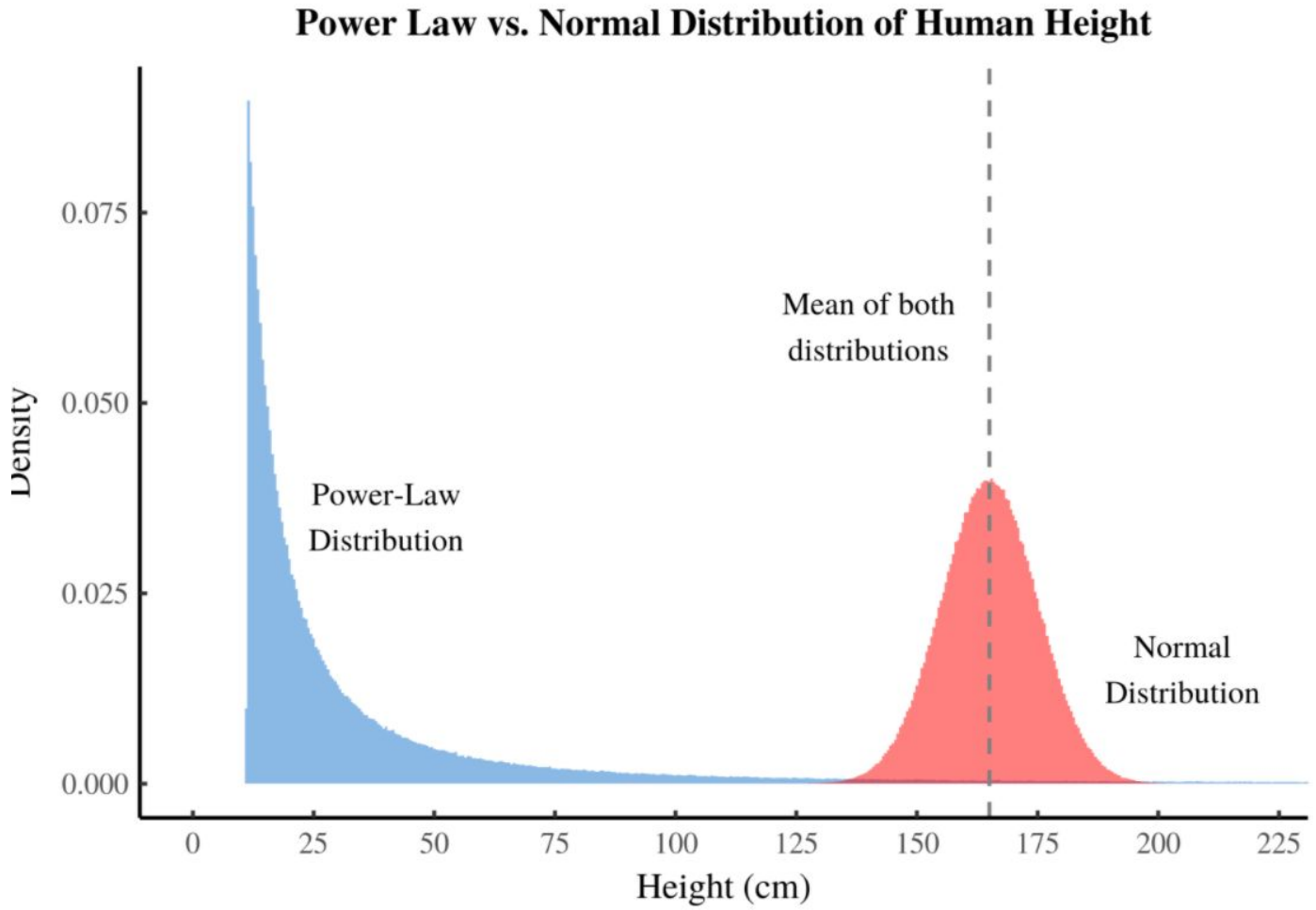


Probability density

- Probability density function *f* shows you where the data is: $P(a \leq X \leq b) = \int_a^b f(x)d_x$
- Note: y-value does not give the probability of observing data point x
- Instead, think of areas; Area under the pdf has to be 100%
- Must be estimated from data: parametric vs nonparametric



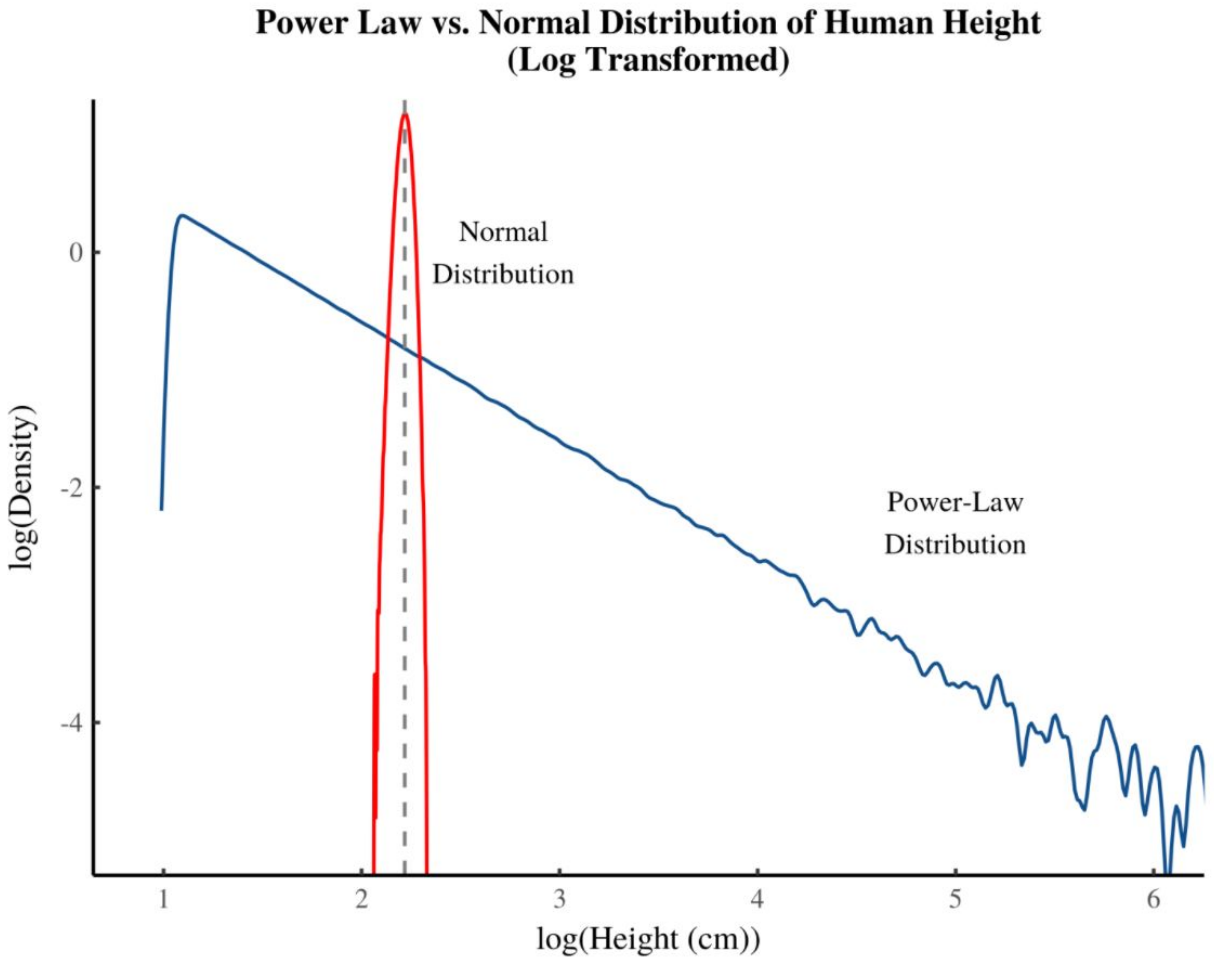
Normal vs skewed data



Equal size bins have sparse data in the tail; use log bins

Source: https://economicsfromthetopdown.com/2019/04/25/visualizing-power-law-distributions/

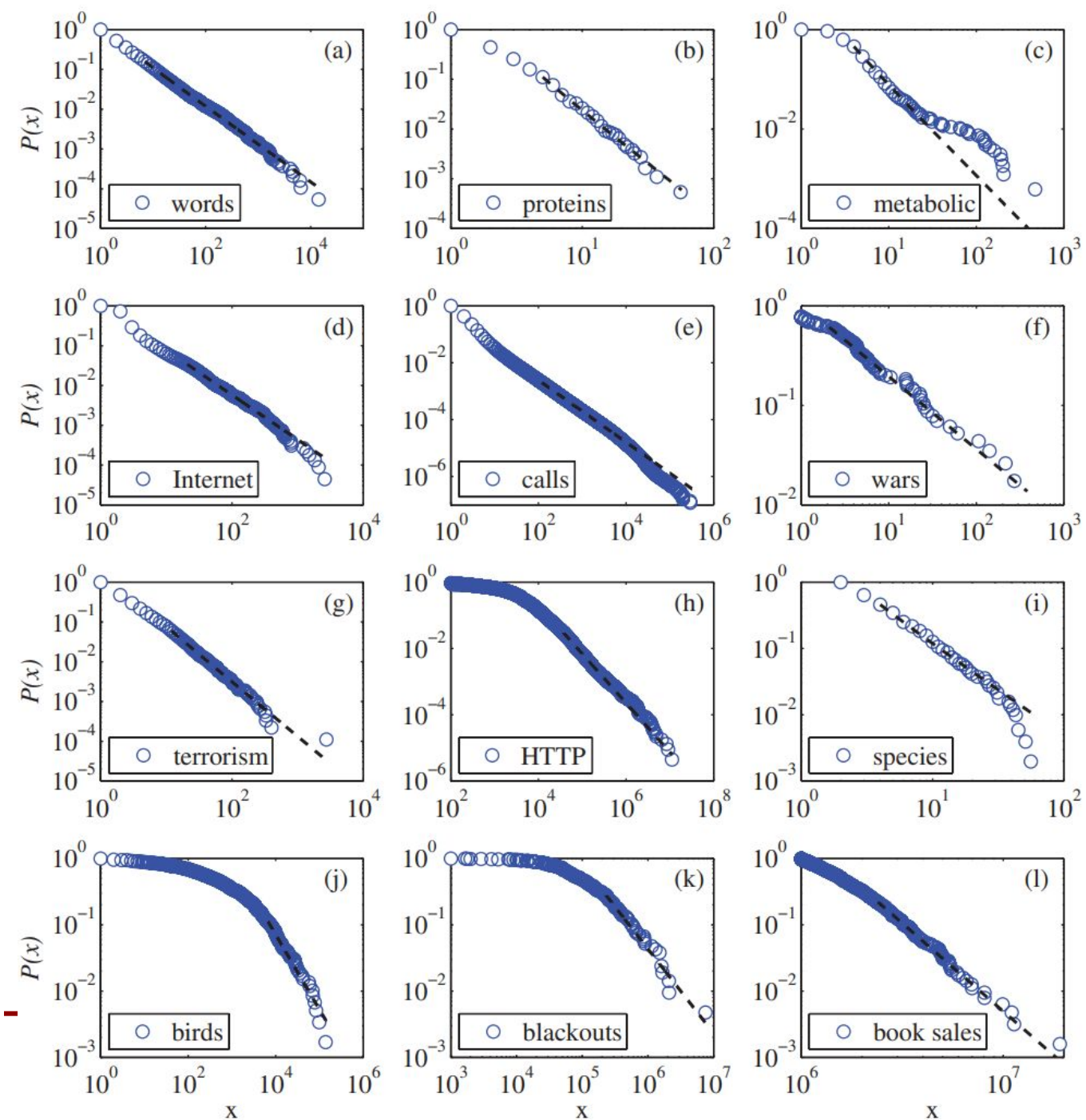
Normal vs skewed data



Source: https://economicsfromthetopdown.com/2019/04/25/visualizing-power-law-distributions/

Lots of data is highly skewed

(a) Frequency of unique words in Moby Dick by Herman Melville.
(b) Degree distribution of protein interaction network.
(c) and metabolic network of E. coli.
(d) Degree distribution of autonomous systems on the Internet.
(e) Number of long-distance phone calls received by AT&T customers.
(f) Number of deaths in wars from 1816–1980 measured.
(g) Deaths due to terrorist attacks worldwide 1968 -2006.
(h) Number of bytes by HTTP (web) requests.
(i) The number of species per genus of mammals during the late Quaternary period.
(j) Frequency of sightings of bird species in the US.
(k) Number of customers affected by blackouts in the US.
(l) The sales volume of bestselling books in the US.



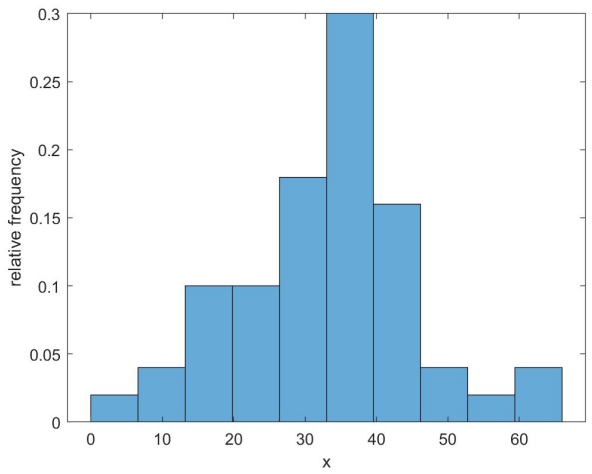
Clauset, A., Shalizi, C. R., & Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, *51*(4), 661-703.

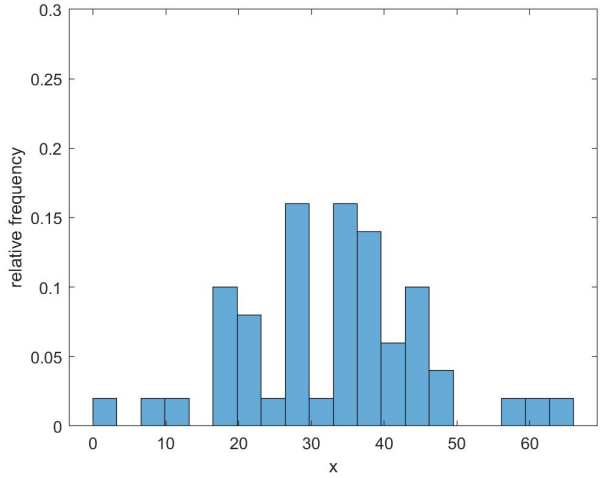
Density Estimation

- Given the training set $\boldsymbol{X}=\{x^t\}_t$ drawn iid from $p(x)$
- Divide data into bins of size h
- Histogram:

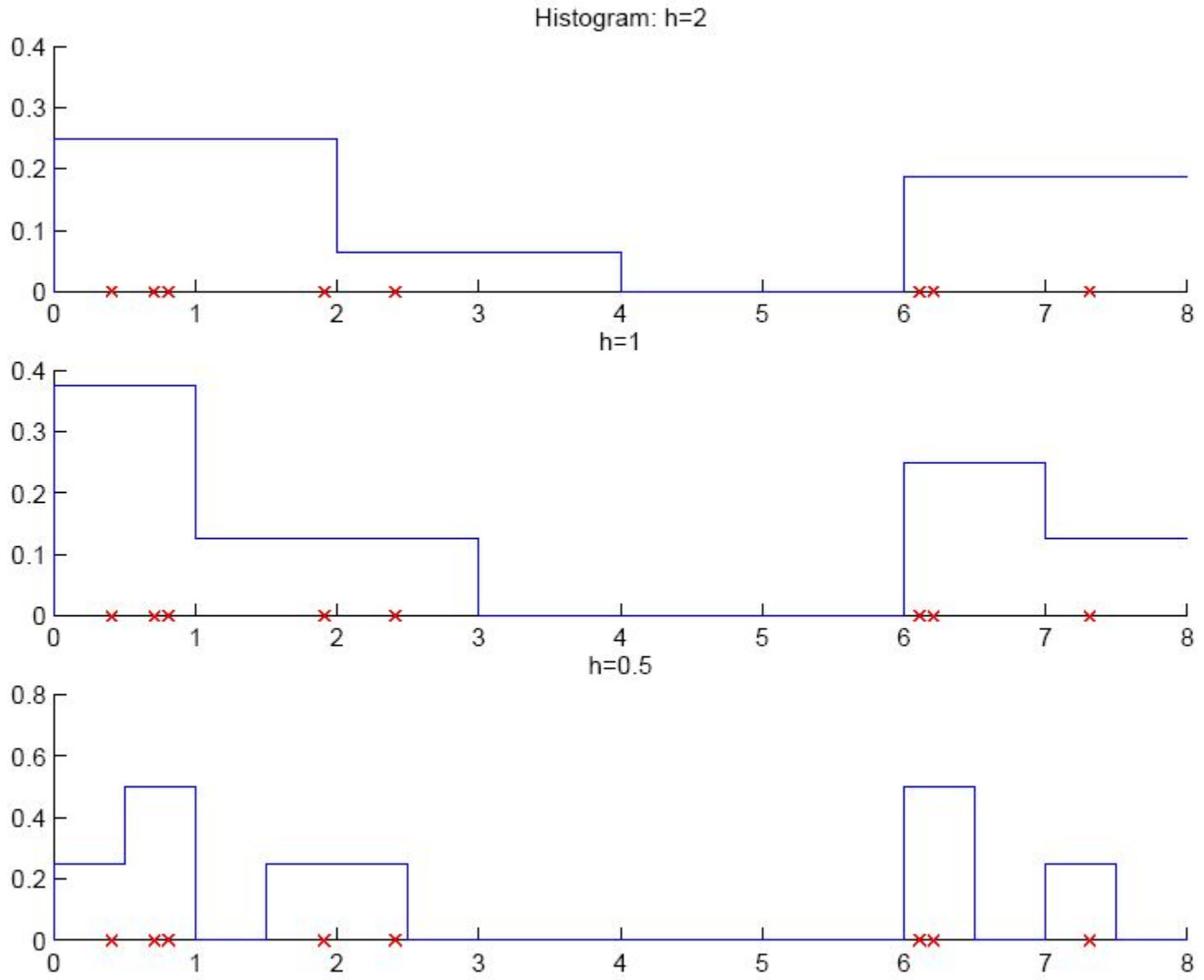
$$\hat{p}(x)=\frac{\#\left\{x^t \text{ in the same bin as } x\right\}}{Nh}$$

- Area equals 100%



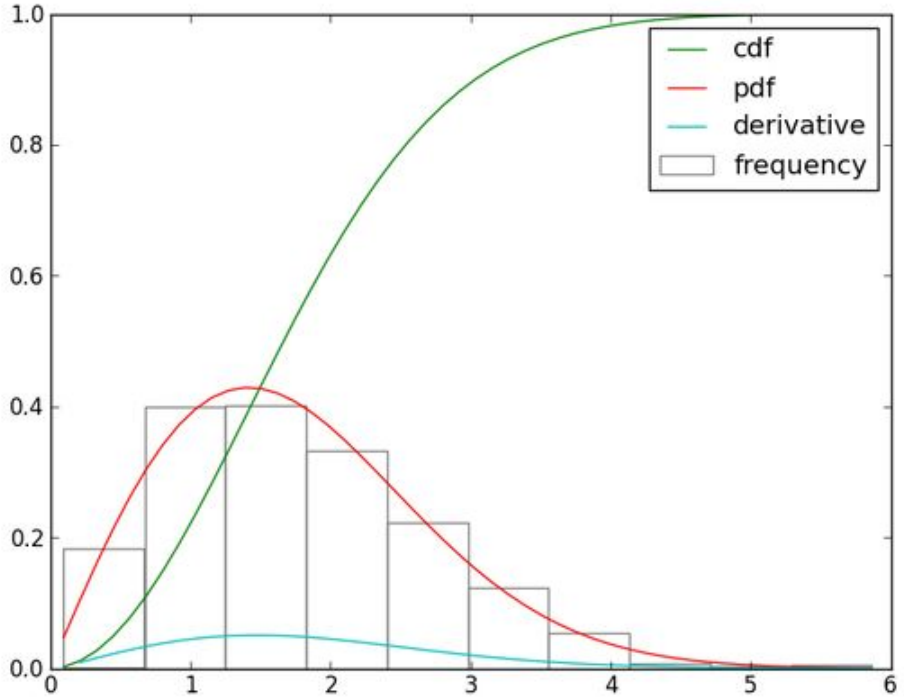


Estimator often not smooth


Histogram: h=2
h=1
h=0.5

Histograms, PDF, CDF, CCDF

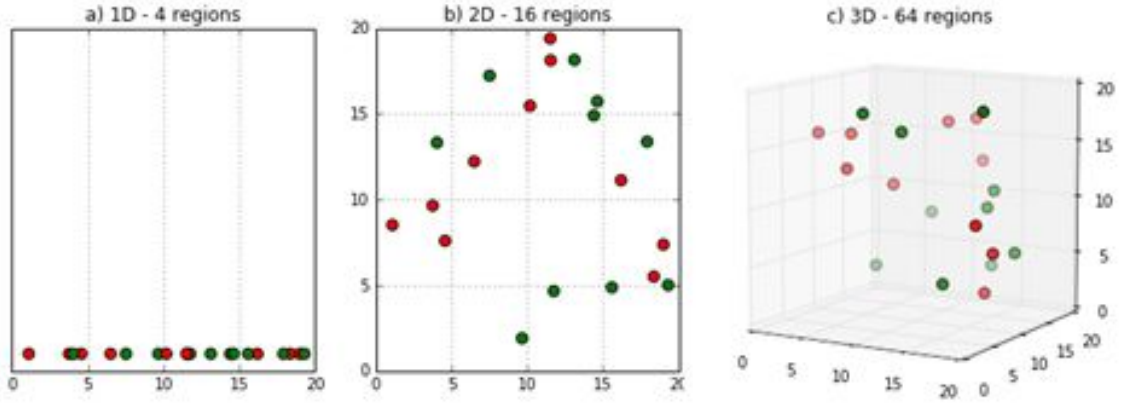
- **Histogram**: visualizes the frequency distribution of data values
  - How many data points have value within that range?
- **PDF** (Probability density function): Continuous representation of the likelihood of a random variable takes a particular range of values.
  - How likely is it for data to have that value?
- **CDF** (Cumulative Distribution Function): represents the probability that a random variable takes on a value less than or equal to a given value
  - How likely is it for data to have values at least as x?
- **CCDF** (Complementary Cumulative Distribution Function): CCDF = 1 - CDF
  - How likely is it for data to have values more than x?



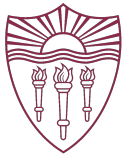
Example: Distribution of dice roll

Multivariate Data

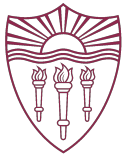
- Watch out for using non-parametric estimates with high-dimensional data: curse of dimensionality
- Example: dim(X)=8, and we use a histogram with 10 bins per dimension
  - 10^8 bins, most will be empty
  - Estimates will be mostly zero
- Choose bin size carefully



Outlier Detection



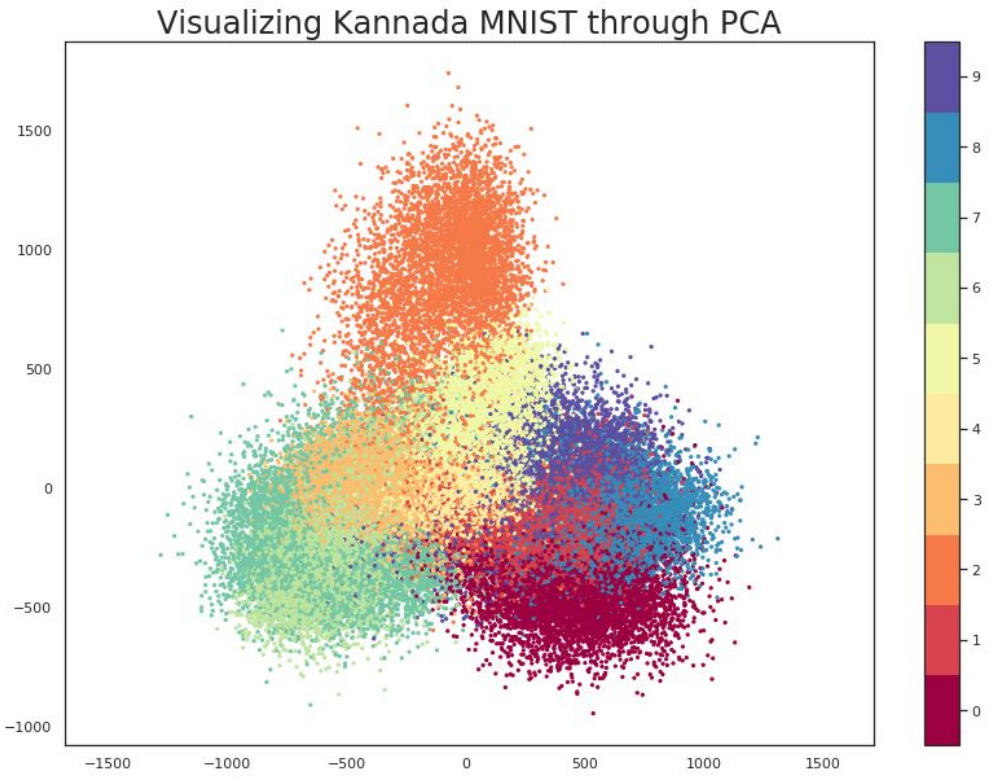
- Find outlier/novelty points
- Not a two-class problem because outliers are very few, of many types, and seldom labeled
- Instead, one-class classification problem: Find instances that have low probability
- In nonparametric case: Find instances far away from other instances



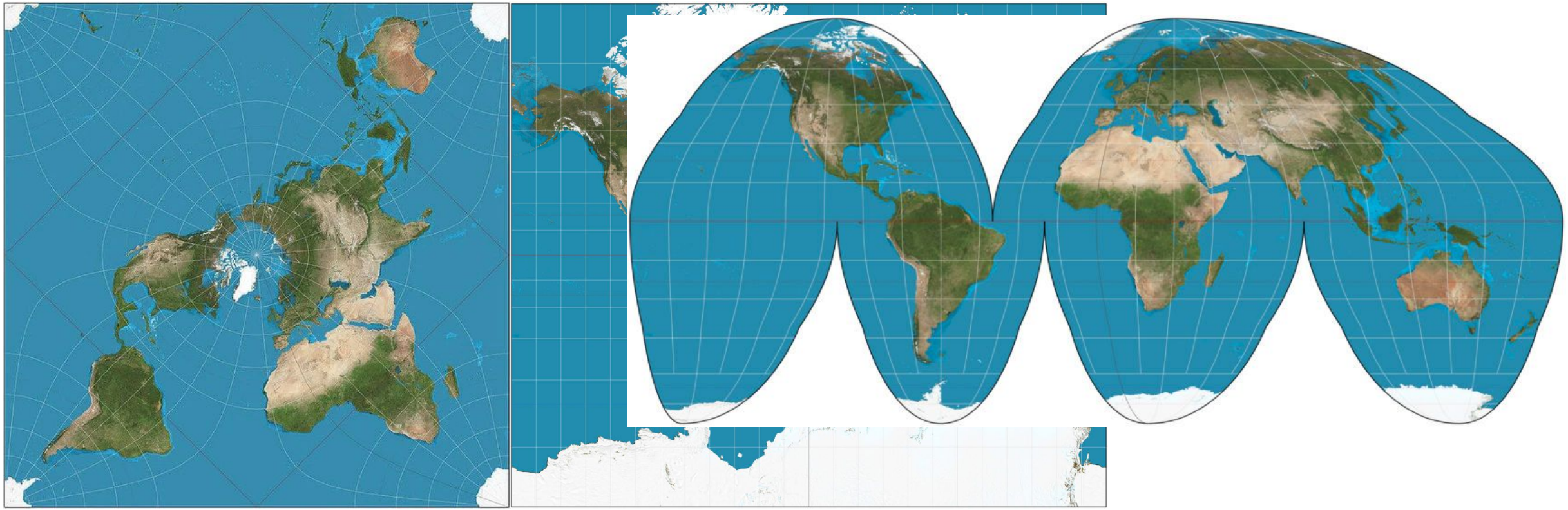
VISUALIZING HIGH DIMENSIONAL DATA USING DIMENSIONALITY REDUCTION TECHNIQUES

Visualizing high dimensional data

- No visualization is available for high dimensional data (only partially; lose a lot of information)
- Methods to allow us to visualize/analyze data from higher dimensions in lower dimensions without losing a good part of the information
- Dimensionality reduction



Pay attention to distortions



Dimensionality reduction

- Given dataset *R* with *m* rows(observations) and *n* columns(features).
- Create a new dataset with *m* rows and *n'* columns that best summarizes the original data
- *n'*<<*n*, where *n* is a large number (*n*>>3)

- Two common solutions
  - PCA (Principal Component Analysis)
  - t-SNE (t-distributed Stochastic Neighbor Embedding)
    - Visualizing Data using t-SNE (original paper)

PCA (Principal Component Analysis)

Consider the original data set with m rows and n columns as a matrix R (columns of R should be standardized)

The goal is to find two **low rank** P (m rows, n' columns) and Q (n' rows, n columns)

such that R' = PQ best approximate R.
minimize $||R - R'||_F$

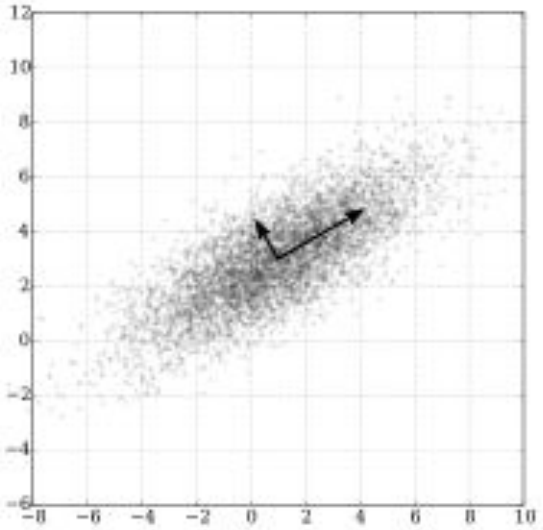
Then P is the low dimensional version of R. Rows of Q are the n' features extracted, represented in original n dimensional space.

$$\|\mathbf{A}\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

PCA (Principal Component Analysis)

How do we calculate P and Q?

Intuition -- The n data points distributed along certain axes in the m dimensional space.



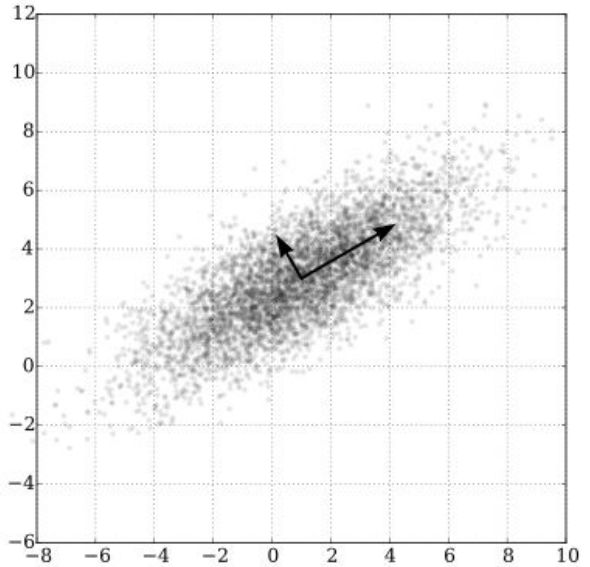
The axes are given by the eigenvectors of covariance matrix

$C = R^T R$.

Find only first n' eigenvectors with largest absolute eigenvalues. Project data to those axes.

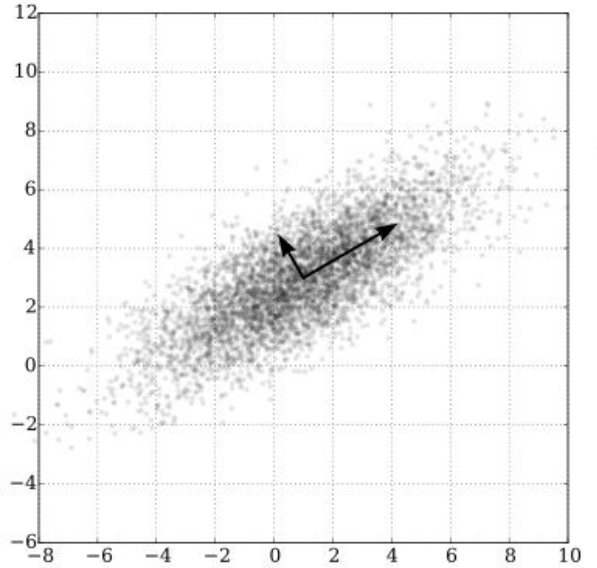
Principal Component Analysis - PCA

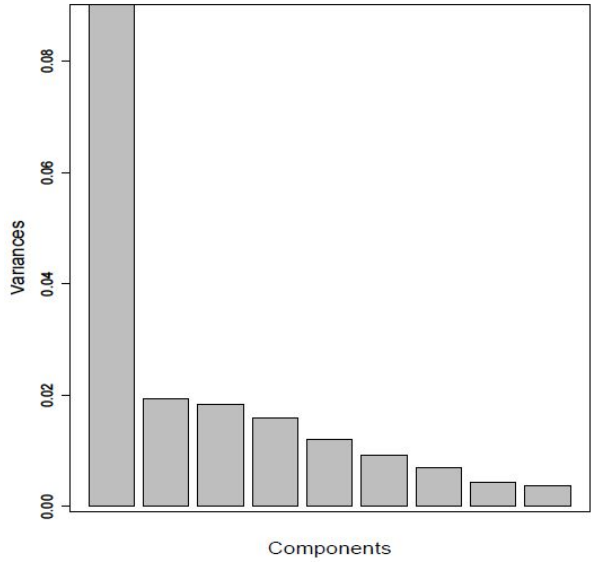
- Unsupervised method that identifies the internal structure of high-dimensional data that best explains its variance
- Embeds the data in a new lower-dimensional space, such that
  - The *first component* is that (normalized) linear combination of the variables with the largest variances.
  - The *second principal component* has largest variance, subject to being uncorrelated with the first….etc.
- Thus, with many correlated features, we replace them with a small set of principal components that capture their joint variation.



PCA (Principal Component Analysis)

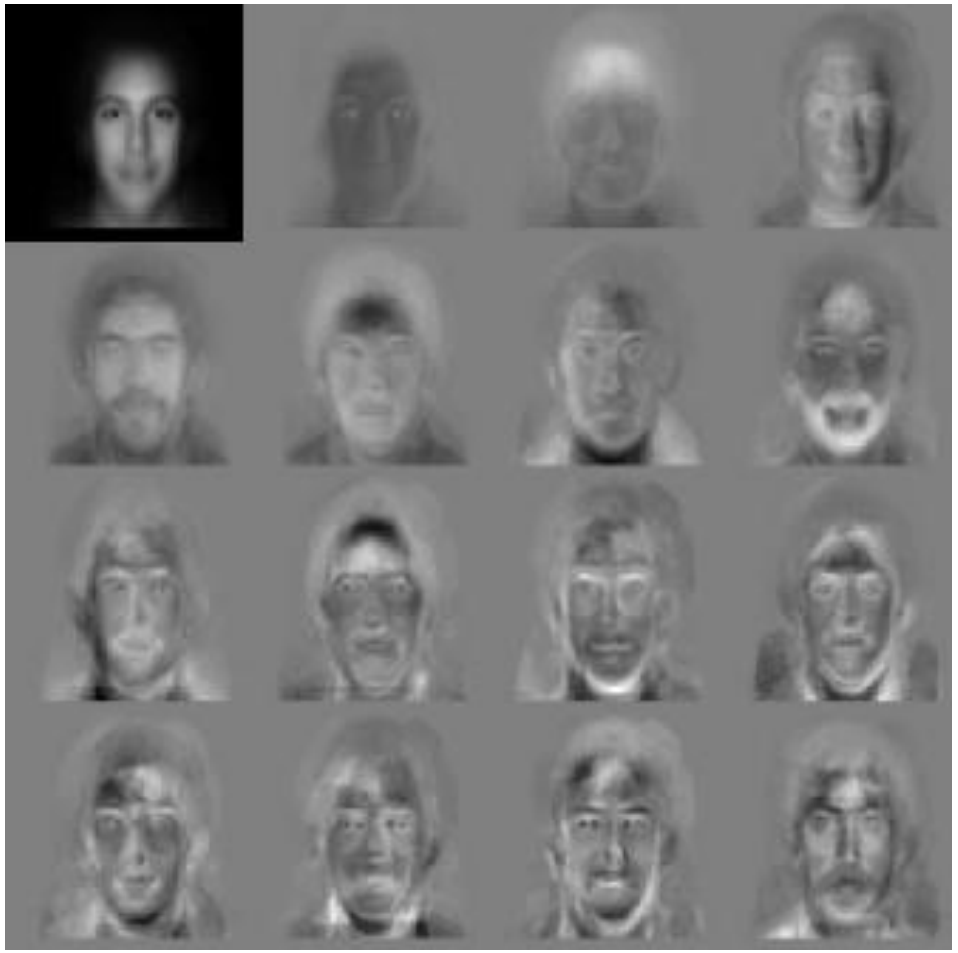
- Eigenvectors are orthogonal towards each other and have length one
- The first couple of eigenvectors explain most of the variance observed in the data
- Low eigenvalues indicate little loss of information if omitted



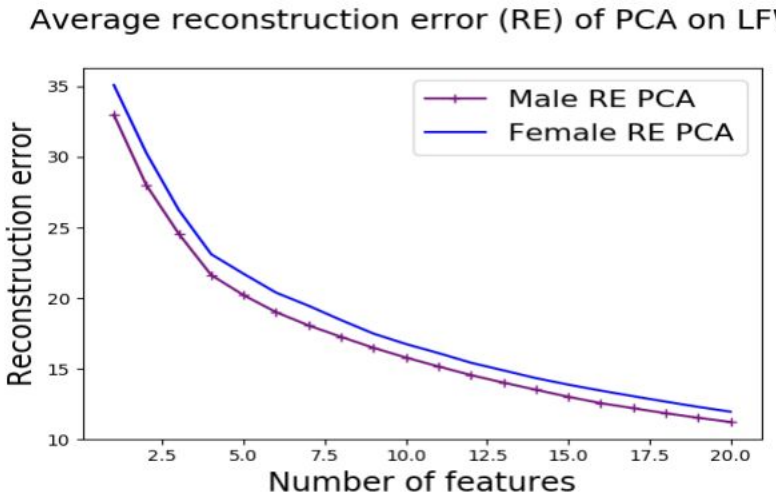


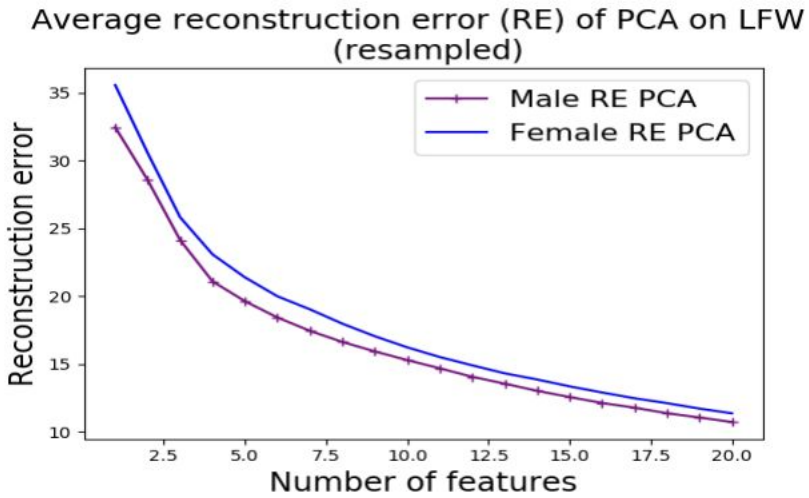
Principal Component Analysis

- Use cases for PCA:
  - Data compression
  - Feature selection and feature reduction
  - Data visualization
  - Eigenfaces are the principal components of a large set of “faces”
    - Shape
    - Hair
    - Beard
    - Glasses
    - …



Fair PCA





$$error(Y, Z) = \|Y - Z\|_F^2$$

Samadi, Samira, et al. "The price of fair PCA: one extra dimension." *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 2018.

Fair PCA

- Ensure reconstruction loss for two groups, *A* and *B,* is similar.

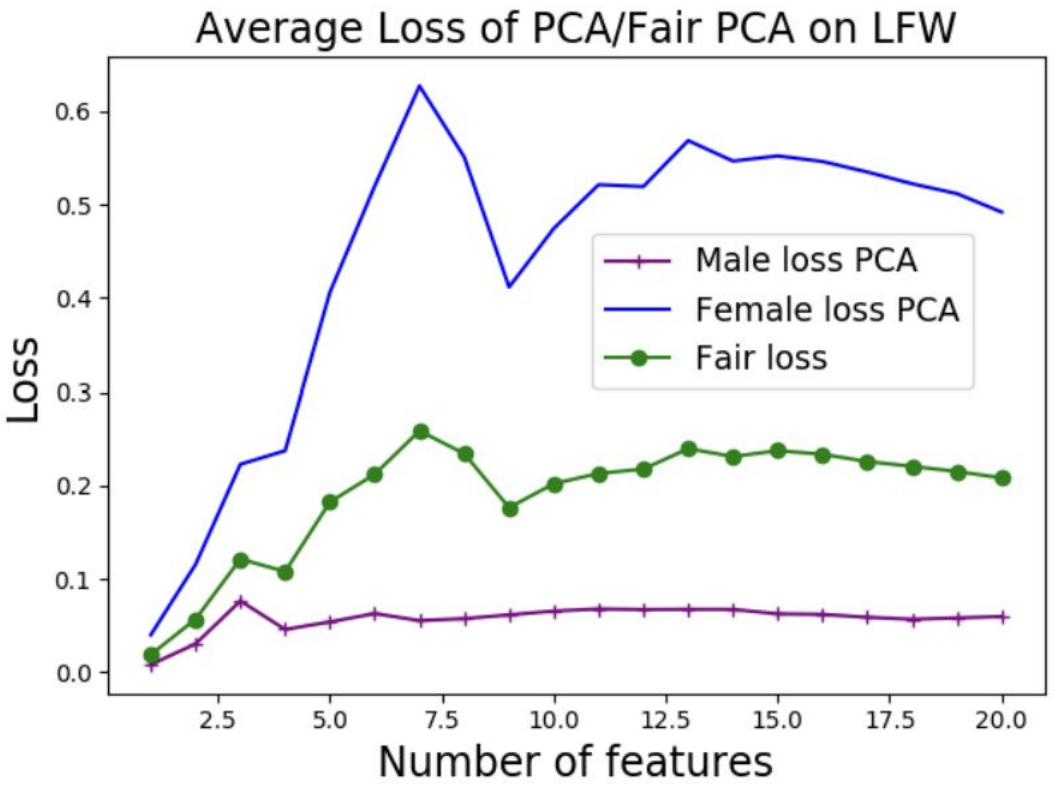
$$loss(Y, Z) := \|Y - Z\|_F^2 - \|Y - \widehat{Y}\|_F^2.$$

Definition 4.4 (Fair PCA). *Given m data points in $\mathbb{R}^n$ with subgroups A and B, we define the problem of finding a fair PCA projection into d-dimensions as optimizing*

$$\min_{U \in \mathbb{R}^{m \times n},\ \mathsf{rank}(U) \leq d} \max \left\{ \frac{1}{|A|} loss(A, U_A), \frac{1}{|B|} loss(B, U_B) \right\}, \tag{1}$$

where U_A and U_B are matrices with rows corresponding to rows of U for groups A and B respectively.

Fair PCA



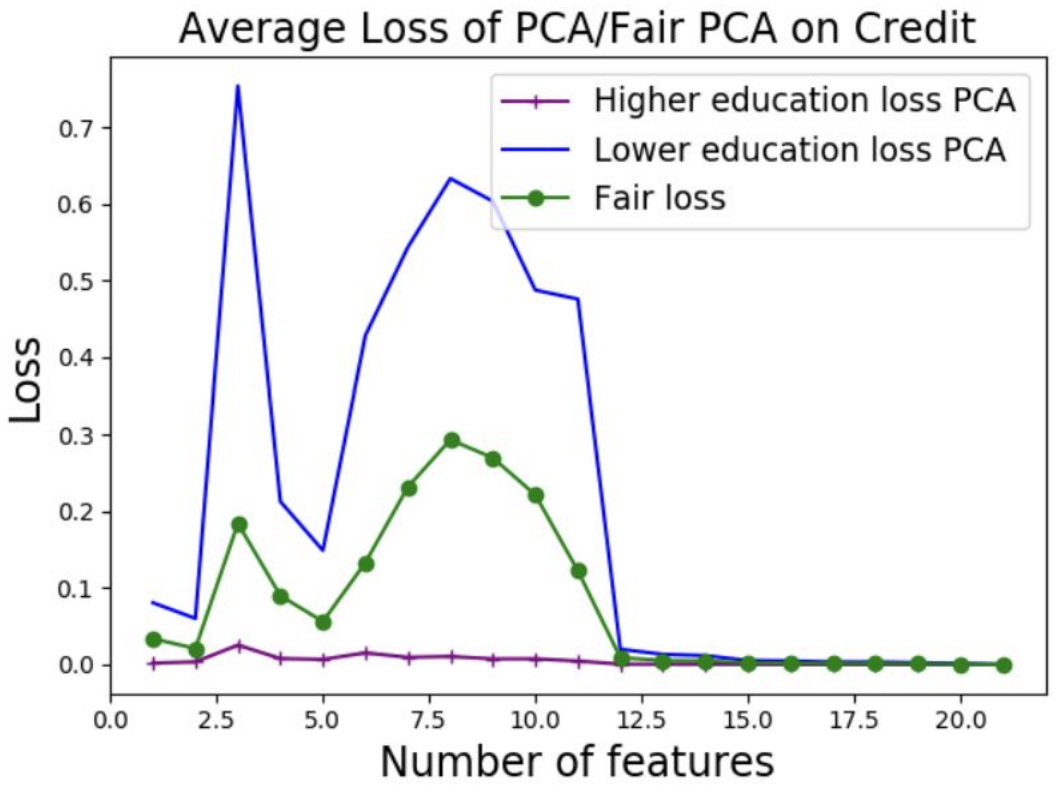


Figure 4: Loss of PCA/Fair PCA on LFW and the Default Credit data set.

t-distributed Stochastic Neighbor Embedding (t-SNE)

- Nonlinear dimensionality reduction technique for embedding high-dimensional data in a low-dimensional space (two or three dimensions)
- Used for visualizing high-dimensional data
- Models each object as a 2- or 3-dimensional point such that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

t-SNE (t-distributed Stochastic Neighbor Embedding)

For original data, define similarity measures

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

Where $$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||^2 / 2\sigma_i^2)}$$

Define the similarity measures of transformed data (y) as

$$q_{ij} = \frac{(1+||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_k \sum_{l \neq k} (1+||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

t-SNE (t-distributed Stochastic Neighbor Embedding)

And set $p_{ii} = 0 \quad q_{ii} = 0$

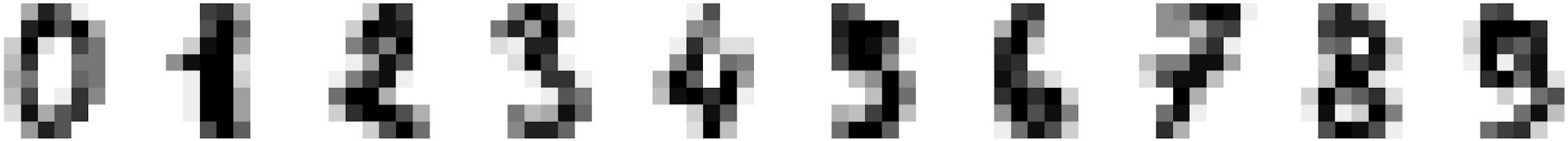
Minimize $\mathrm{KL}(P \parallel Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$

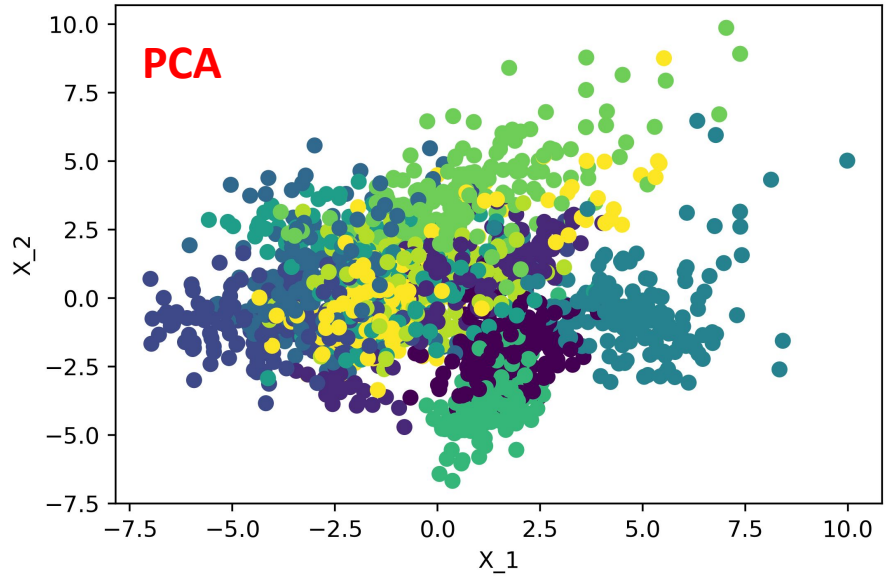
Using gradient descent.

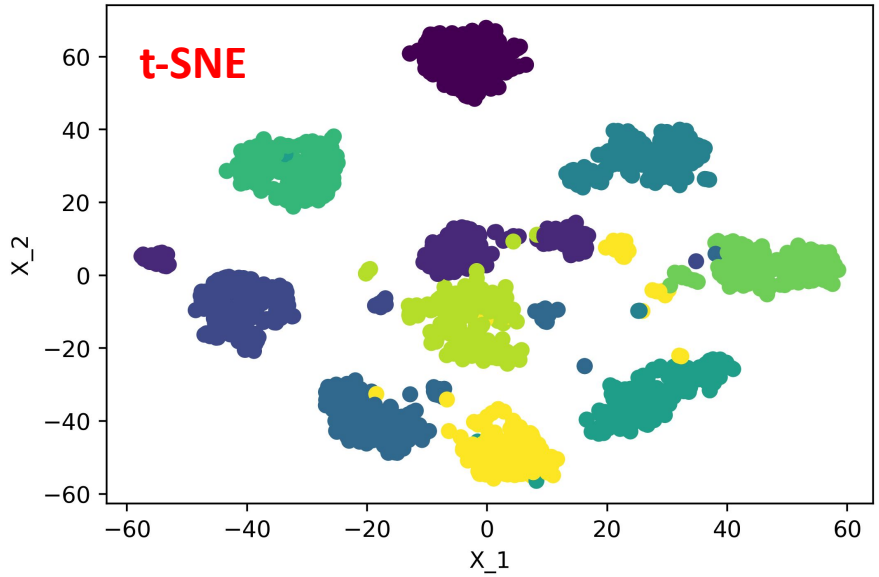
Intuition: Similar data observations are located closer in the transformed space.

Example -- Handwritten digits images

Original data (8*8 images, 64 dimensions)

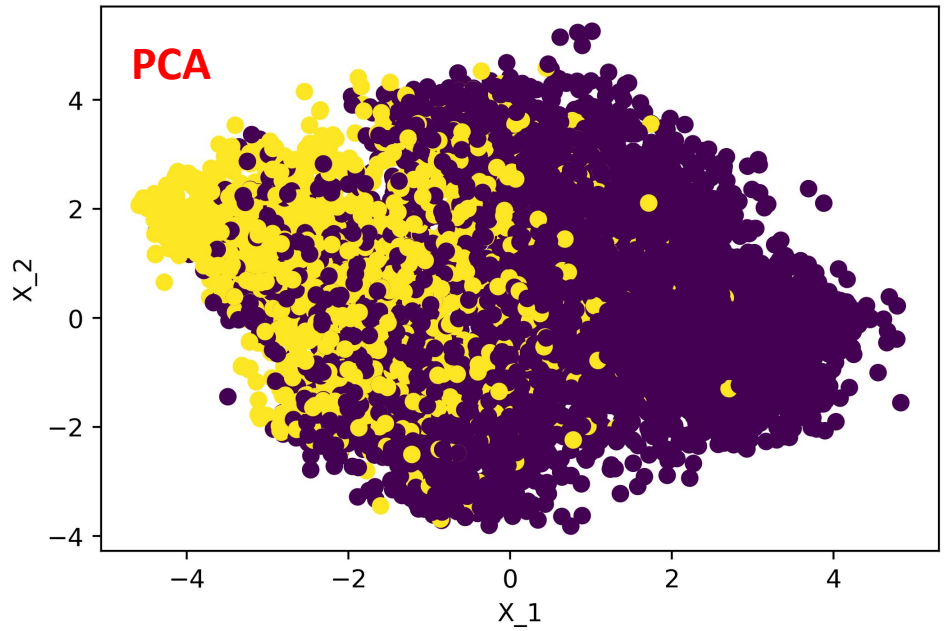


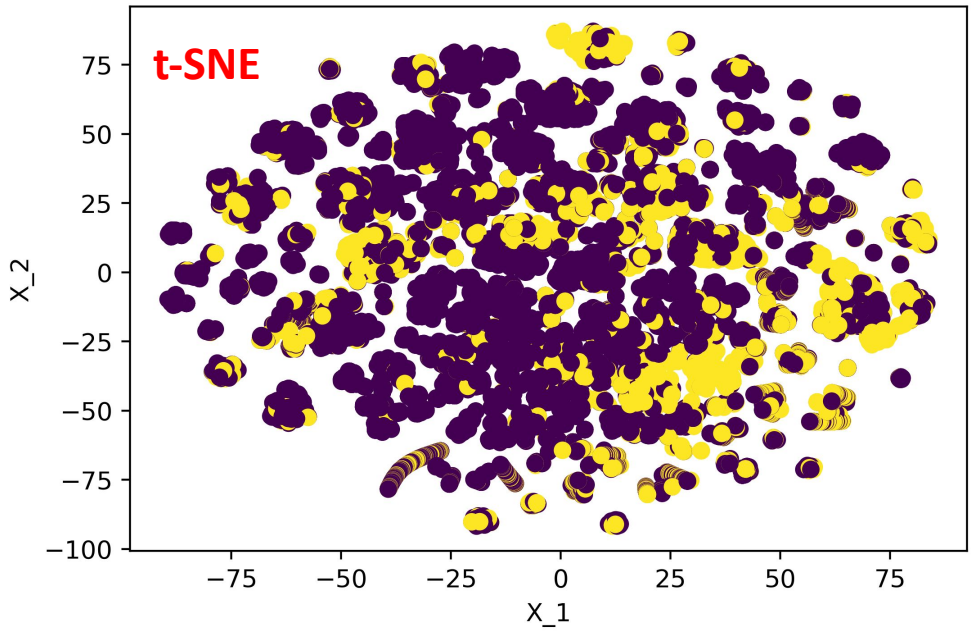




Example -- Adult Dataset (Tabular)

Adult dataset contains features of individuals. The outcome is whether the person makes $50K annually. (shown as yellow)





Summary

PCA

- Linear
- Global structure
- Affected by outliers
- Deterministic
- Fitting data to an ellipsoid

t-SNE

- Nonlinear
- Local structure
- Not sensitive to outliers
- Stochastic
- Based on data similarity
- Usually the best method for image data