Review for the Final

Shuai Li

John Hopcroft Center, Shanghai Jiao Tong University

https://shuaili8.github.io

https://shuaili8.github.io/Teaching/VE445/index.html

Exam code

- Exam on Dec 6, 8:00-9:40 at Dong Xia Yuan 102 (lecture classroom)
- Finish the exam paper by yourself
- Allowed:
  - Calculator
- Not allowed:
  - Books, materials, cheat sheet, …
  - Phones, any smart device
- No entering after 8:25
- Early submission period: 8:30--9:25

Coverage of final

- Basics
- Supervised learning
  - Regression
  - SVM and Kernel methods
  - Decision Tree
- Deep learning
  - Neural Networks
  - Backpropagation
  - Convolutional Neural Network
  - Recurrent Neural Network
- Unsupervised learning
  - K-means, Agglomerative clustering, BFR, CURE
  - PCA, SVD, Autoencoder, Feature selection
  - EM, GMM
  - HMM
- Learning theory
  - Bias-variance decomposition

Exam contents

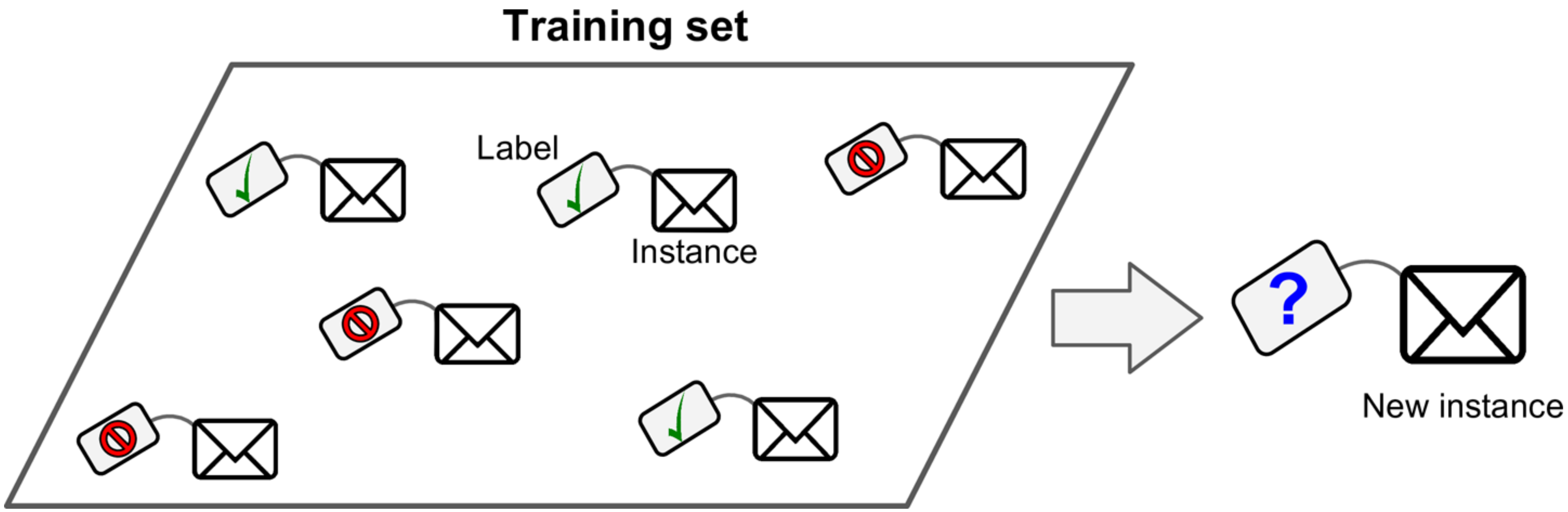
- 4 questions
  - 3 with 30 marks
    - Computation tasks
    - Concept understanding
  - 1 with 10 marks
    - Algorithm description

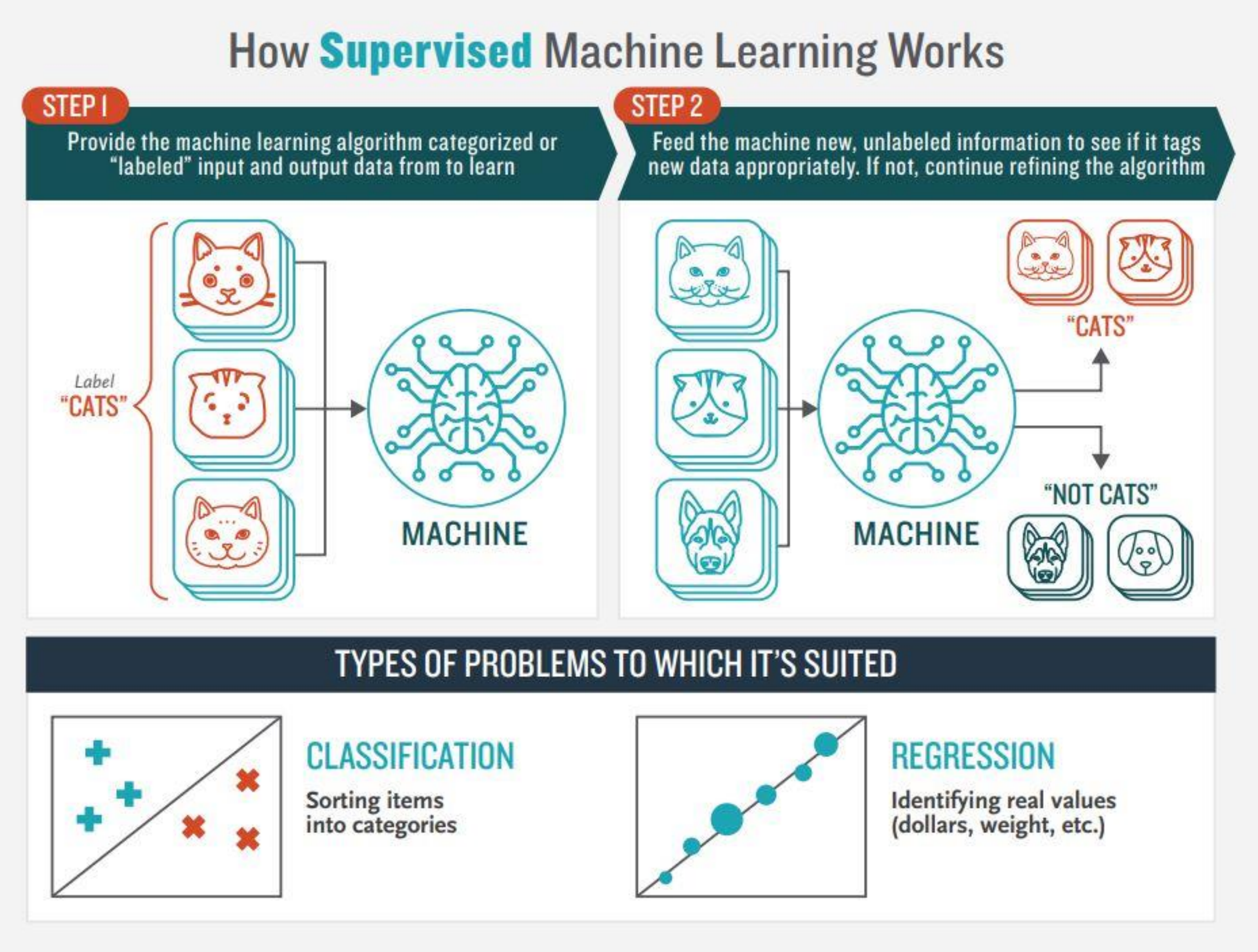
Supervised Learning

Machine Learning Categories

- Unsupervised learning
  - No labeled data
- Supervised learning
  - Use labeled data to predict on unseen points
- Semi-supervised learning
  - Use labeled data and unlabeled data to predict on unlabeled/unseen points
- Reinforcement learning
  - Sequential prediction and receiving feedbacks

Supervised learning example





How Supervised Machine Learning Works

STEP 1

Provide the machine learning algorithm categorized or "labeled" input and output data from to learn

STEP 2

Feed the machine new, unlabeled information to see if it tags new data appropriately. If not, continue refining the algorithm

TYPES OF PROBLEMS TO WHICH IT'S SUITED

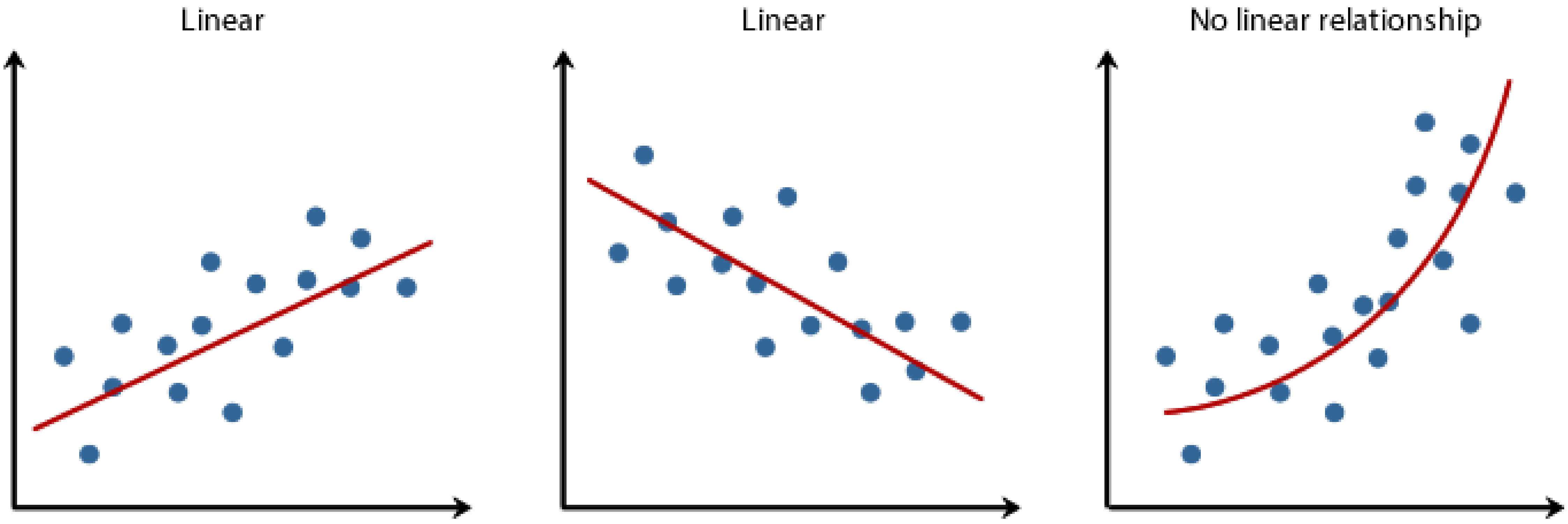
CLASSIFICATION

Sorting items into categories

REGRESSION

Identifying real values (dollars, weight, etc.)

Regression example



Copyright 2014. Laerd Statistics.

Model Evaluations

Classification -- Model evaluations

- Confusion Matrix
  - TP – True Positive ; FP – False Positive
  - FN – False Negative; TN – True Negative

	Predicted Class		
Actual Class		Class = Yes	Class = No
	Class = Yes	a (TP)	b (FN)
	Class = No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}$$

Classification -- Model evaluations

- Limitation of Accuracy
  - Consider a 2-class problem
    - Number of Class 0 examples = 9990
    - Number of Class 1 examples = 10
  - If a “stupid” model predicts everything to be class 0, accuracy is 9990/10000 = **99.9** %

- The accuracy is misleading because the model does not detect any example in class 1

Classification -- Model evaluations

- Cost-sensitive measures

		Predicted Class	
Actual Class		Class = Yes	Class = No
	Class = Yes	a (TP)	b (FN)
	Class = No	c (FP)	d (TN)

$$\text{Precision (p)} = \frac{TP}{TP + FP} = \frac{a}{a + c}$$

$$\text{Recall (r)} = \frac{TP}{TP + FN} = \frac{a}{a + b}$$

$$\text{F - measure (F)} = \frac{2rp}{r + p} = \frac{2a}{2a + b + c}$$

Harmonic mean of Precision and Recall
(Why not just average?)

Example

- Given 30 human photographs, a computer predicts 19 to be male, 11 to be female. Among the 19 male predictions, 3 predictions are not correct. Among the 11 female predictions, 1 prediction is not correct.

	Predicted Class		
Actual Class		Male	Female
	Male	a = TP = 16	b = FN = 1
	Female	c = FP = 3	d = TN = 10

Example

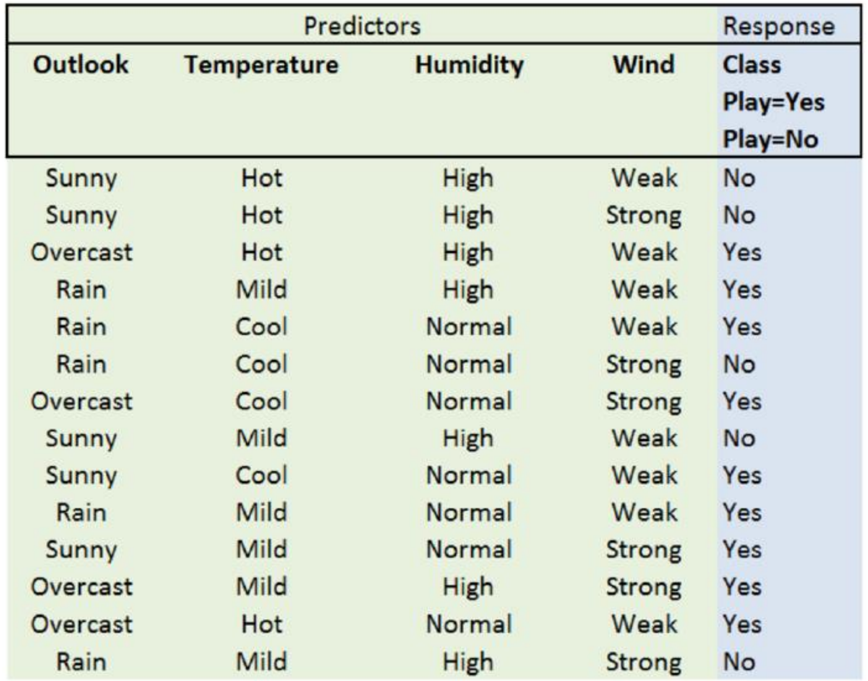
	Predicted Class		
Actual Class		Male	Female
	Male	a = TP = 16	b = FN = 1
	Female	c = FP = 3	d = TN = 10

- Accuracy = (16 + 10) / (16 + 3 + 1 + 10) = 0.867
- Precision = 16 / (16 + 3) = 0.842
- Recall = 16 / (16 + 1) = 0.941
- F-measure = 2 (0.842)(0.941) / (0.842 + 0.941)
  = 0.889

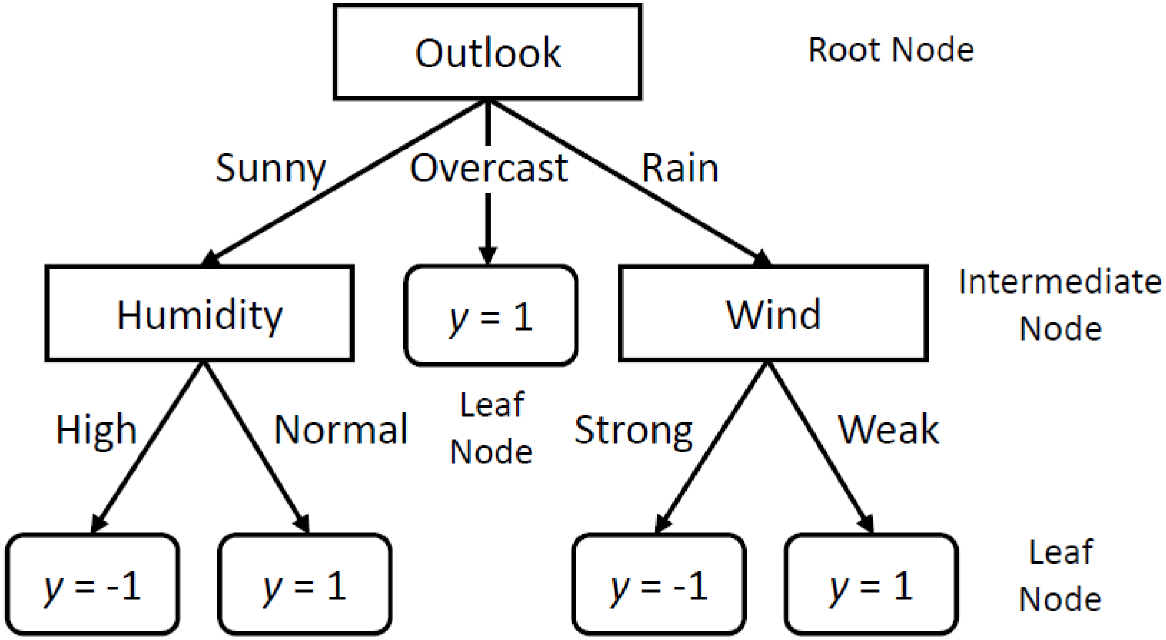
Decision Tree

Tree models

- Tree models
  - Intermediate node for splitting data
  - Leaf node for label prediction
- Discrete/categorical data example



Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class** **Play=Yes** **Play=No**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

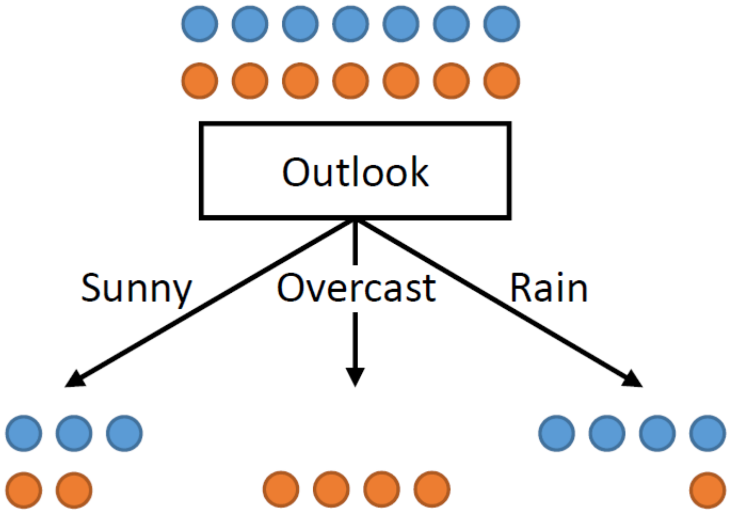


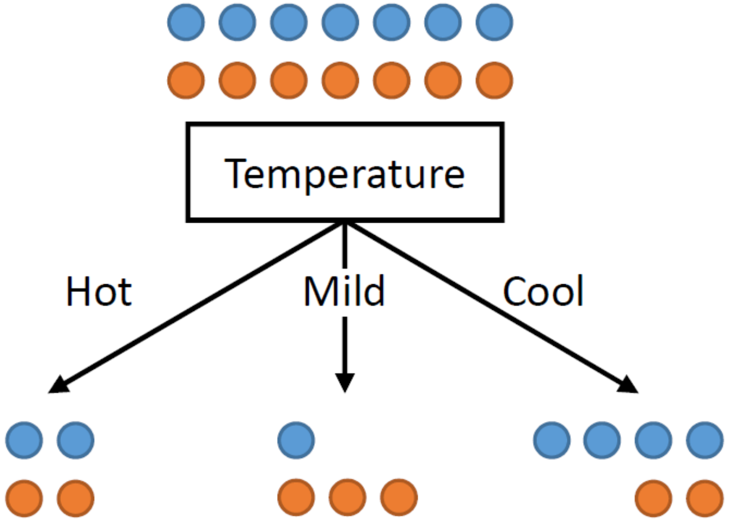
Tree models

- Tree models
  - Intermediate node for splitting data
  - Leaf node for label prediction
- Discrete/categorical data example
- Key questions for decision trees
  - How to select node splitting conditions?
  - How to make prediction?
  - How to decide the tree structure?

Node splitting

- Which node splitting condition to choose?





- Choose the features with higher classification capacity
  - Quantitatively, with higher information gain

Information Theory

Motivating example 1

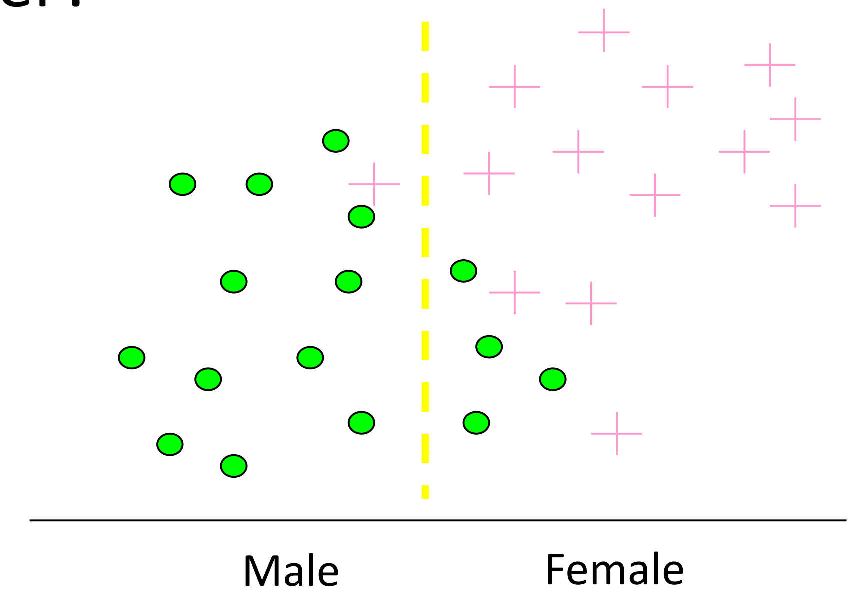
- Suppose you are a police officer and there was a robbery last night. There are several suspects and you want to find the criminal from them by asking some questions.

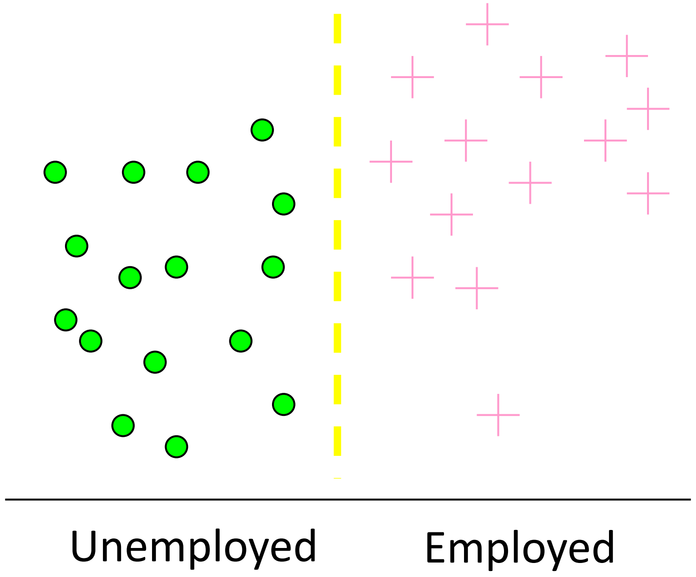
- You may ask: where are you last night?
- You are not likely to ask: what is your favorite food?

- Why there is a preference for the policeman? Because the first one can distinguish the guilty from the innocent. It is more informative.

Motivating example 2

- Suppose we have a dataset of two classes of people. Which split is better?





- We prefer the right split because there is no outliers and it is more certain.

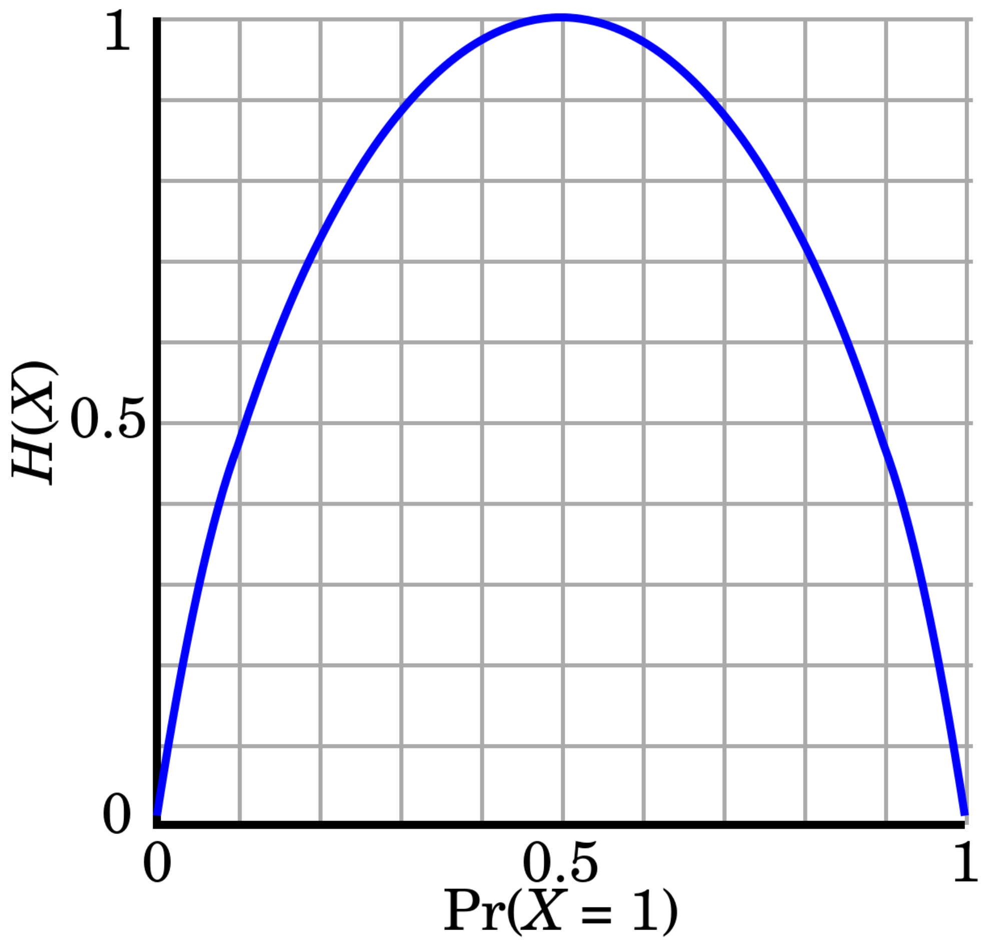
Entropy

- How to measure the level of informative (first example) and level of certainty (second example) in mathematics?
- Entropy (more specifically, Shannon entropy) is the expected value (average) of the information contained in each message
- Suppose X is a random variable with n discrete values
$$P(X = x_i) = p_i$$
then the entropy is
$$H(X) = -\sum_{i=1}^{n} p_i \log_2(p_i)$$
- Easy to verify $H(X) = -\sum_{i=1}^{n} p_i \log_2(p_i) \leq -\sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = \log n$

Illustration

- Entropy for binary distribution

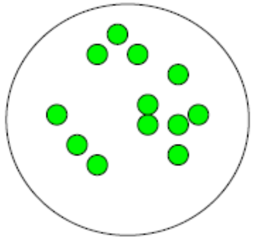
$$H(X) = -p_1 \log_2(p_1) - p_0 \log_2(p_0)$$



Entropy examples

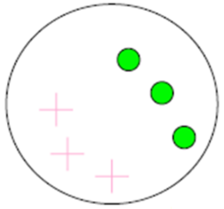
- What is the entropy of a group in which all examples belong to the same class?
  - Entropy = $-\,1\log_2 1\ =\ 0$

Minimum uncertainty



- What is the entropy of a group with 50% in either class?
  - Entropy = $-0.5\log_2 0.5 - \ 0.5\log_2 0.5\ = 1$

Maximum uncertainty



Conditional entropy

- Specific conditional entropy of X given $Y = y$

$$H(X|Y = y) = -\sum_{i=1}^{n} P(X = i|Y = y) \log P(X = i|Y = y)$$

- Conditional entropy of X given Y

$$H(X|Y) = \sum_{y} P(Y = y) H(X|Y = y)$$

- Information gain of X given Y

$$I(X, Y) = H(X) - H(X|Y)$$

Interpretation from coding perspective

- Usually entropy denotes the minimal average message length of the best coding (theoretically)

- When the probability distribution is composed of $\frac{1}{2^i}$, then the average length of Hoffman code is the entropy

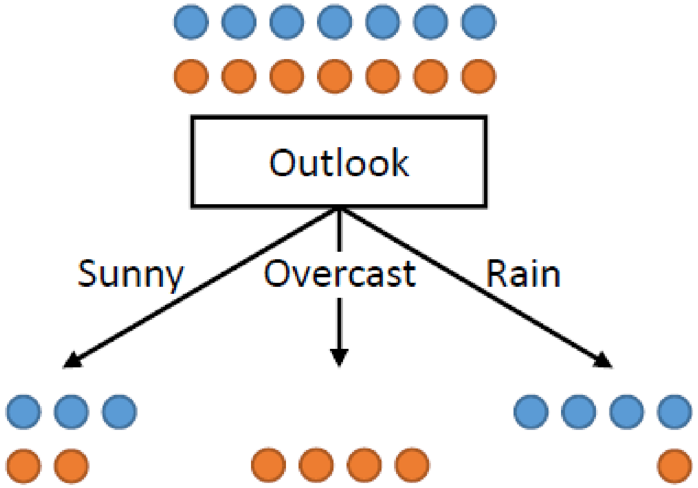
Decision Tree

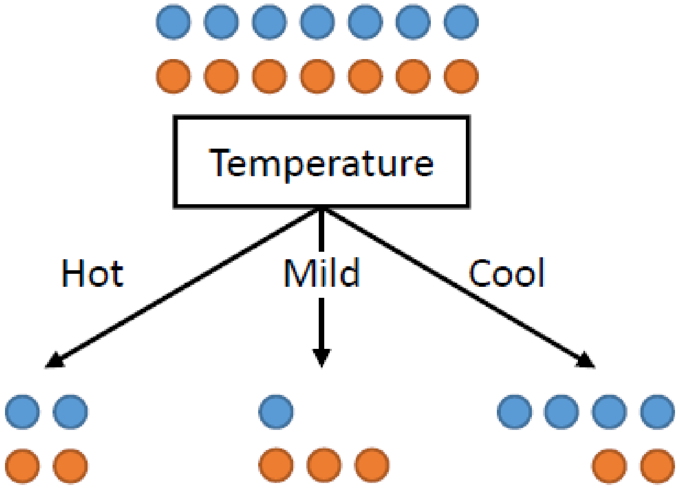
Node Splitting

- Information gain

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log P(X=i|Y=v)$$

$$H(X|Y) = \sum_{v\in \text{values}(Y)} P(Y=v)H(X|Y=v)$$





$$H(X|Y=S) = -\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5} = 0.9710$$

$$H(X|Y=O) = -\frac{4}{4}\log\frac{4}{4} = 0$$

$$H(X|Y=R) = -\frac{4}{5}\log\frac{4}{5} - \frac{1}{5}\log\frac{1}{5} = 0.7219$$

$$H(X|Y) = \frac{5}{14}\times 0.9710 + \frac{4}{14}\times 0 + \frac{5}{14}\times 0.7219 = 0.6046$$

$$I(X,Y) = H(X) - H(X|Y) = 1 - 0.6046 = 0.3954$$

$$H(X|Y=H) = -\frac{2}{4}\log\frac{2}{4} - \frac{2}{4}\log\frac{2}{4} = 1$$

$$H(X|Y=M) = -\frac{1}{4}\log\frac{1}{4} - \frac{3}{4}\log\frac{3}{4} = 0.8113$$

$$H(X|Y=C) = -\frac{4}{6}\log\frac{4}{6} - \frac{2}{6}\log\frac{2}{6} = 0.9183$$

$$H(X|Y) = \frac{4}{14}\times 1 + \frac{4}{14}\times 0.8113 + \frac{5}{14}\times 0.9183 = 0.9111$$

$$I(X,Y) = H(X) - H(X|Y) = 1 - 0.9111 = 0.0889$$

Node splitting

- We want to determine which attribute in a given set of training feature vectors is most useful for discriminating between the classes to be learned
- Information gain tells us how important a given attribute of the feature vectors is
  - Is used to decide the ordering of attributes in the nodes of a decision tree
- Information gain of X given Y

$$I(X,Y) = H(X) - H(X|Y)$$

Example

- Given a dataset of 8 students about whether they like the famous movie *Gladiator*, calculate the entropy in this dataset

Like
Yes
No
Yes
No
No
Yes
No
Yes

Example (cont.)

- Given a dataset of 8 students about whether they like the famous movie *Gladiator*, calculate the entropy in this dataset

Like
Yes
No
Yes
No
No
Yes
No
Yes

$$E(Like) = -\frac{4}{8}\log\left(\frac{4}{8}\right) - -\frac{4}{8}\log\left(\frac{4}{8}\right)=1$$

Example (cont.)

- Suppose we now also know the gender of these 8 students, what is the conditional entropy on gender?

Gender	Like
Male	Yes
Female	No
Male	Yes
Female	No
Female	No
Male	Yes
Male	No
Female	Yes

Example (cont.)

- Suppose we now also know the gender of these 8 students, what is the conditional entropy on gender?
- The labels are divided into two small dataset based on the gender

Gender	Like
Male	Yes
Female	No
Male	Yes
Female	No
Female	No
Male	Yes
Male	No
Female	Yes

Like (male)
Yes
Yes
Yes
No

P(Yes | male) = 0.75

Like(female)
No
No
No
Yes

P(Yes | female) = 0.25

Example (cont.)

- Suppose we now also know the gender of these 8 students, what is the conditional entropy on gender?
  - $\mathrm{P(Yes|male)} = 0.75$
  - $\mathrm{P(Yes|female)} = 0.25$
  - $\mathrm{H(Like|male)}$
    $= -\frac{1}{4} log\left(\frac{1}{4}\right) - \frac{3}{4} log\left(\frac{3}{4}\right)$
    $= -0.25 * -2 - 0.75 * -0.41 = 0.81$
  - $\mathrm{H(Like|female)}$
    $= -\frac{3}{4} log\left(\frac{3}{4}\right) - \frac{1}{4} log\left(\frac{1}{4}\right)$
    $= -0.75 * -0.41 - 0.25 * -2 = 0.81$

Gender	Like
Male	Yes
Female	No
Male	Yes
Female	No
Female	No
Male	Yes
Male	No
Female	Yes

Example (cont.)

- Suppose we now also know the gender of these 8 students, what is the conditional entropy on gender?
  - $\mathrm{E(Like|Gender)}$
    $= E(Like|male) * P(male)$
    $+E(Like|female) * P(female)$
    $= 0.5 * 0.81 + 0.5 * 0.81 = 0.81$
  - $\mathrm{I(Like, Gender) = E(Like) - E(Like|Gender)}$
    $= 1 - 0.81 = 0.19$

Gender	Like
Male	Yes
Female	No
Male	Yes
Female	No
Female	No
Male	Yes
Male	No
Female	Yes

Example (cont.)

- Suppose we now also know the major of these 8 students, what about the conditional entropy on major?

Major	Like
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Example (cont.)

- Suppose we now also know the major of these 8 students, what about the conditional entropy on major?
- Three datasets are created based on major

Major	Like
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Like (math)
Yes
No
No
Yes

P(Yes|math) = 0.5

Like(history)
No
No

P(Yes|history) = 0

Like(cs)
Yes
Yes

P(Yes|cs) = 1

Example (cont.)

- Suppose we now also know the major of these 8 students, what about the new Entropy
  - $\mathrm{H(Like|Math)} = -\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right)$=1
  - $\mathrm{H(Like|CS)} = -\frac{2}{2}\log\left(\frac{2}{2}\right) - \frac{0}{2}\log\left(\frac{0}{2}\right)$=0
  - $\mathrm{H(Like|history)} = -\frac{2}{2}\log\left(\frac{2}{2}\right) - \frac{0}{2}\log\left(\frac{0}{2}\right)$=0
  - $\mathrm{H(Like|Major)}$
    $$\begin{aligned} &= \mathrm{H(Like|math)} \times P(math) \\ &\quad +\mathrm{H(Like|History)} \times P(History) \\ &\quad +\mathrm{H(Like|cs)} \times P(cs) \\ &= 0.5 * 1 + 0.25 * 0 + 0.25 * 0 = 0.5 \end{aligned}$$
  - $\mathrm{I(Like, Major)} = \mathrm{E(Like)} - \mathrm{E(Like|Major)}$
    $= 1 - 0.5 = 0.5$

Major	Like
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Example (cont.)

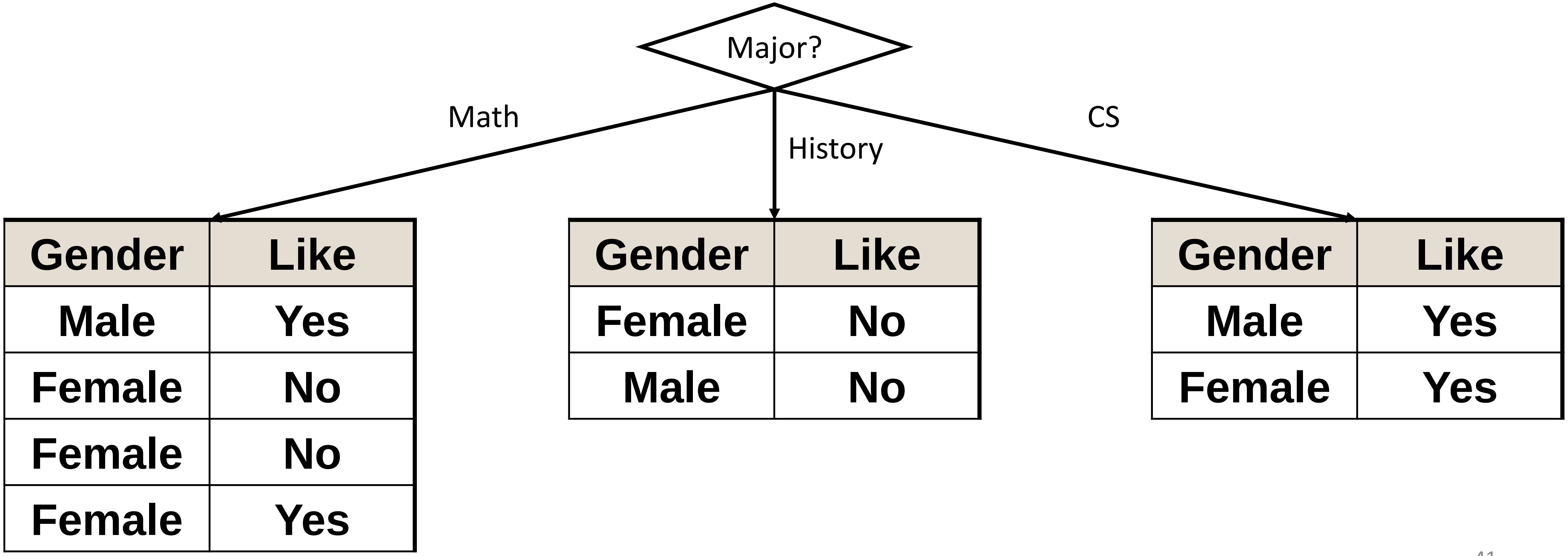
- Compare gender and major
- As we have computed
  - $\mathrm{I(Like, Gender) = E(Like) - E(Like|Gender) = 1 - 0.81 = 0.19}$
  - $\mathrm{I(Like, Major) = E(Like) - E(Like|Major) = 1 - 0.5 = 0.5}$

- Major is the better feature to predict the label "like"

Gender	Major	Like
Male	Math	Yes
Female	History	No
Male	CS	Yes
Female	Math	No
Female	Math	No
Male	CS	Yes
Male	History	No
Female	Math	Yes

Example (cont.)

- Major is used as the decision condition and it splits the dataset into three small one based on the answer



Major?

Math

History

CS

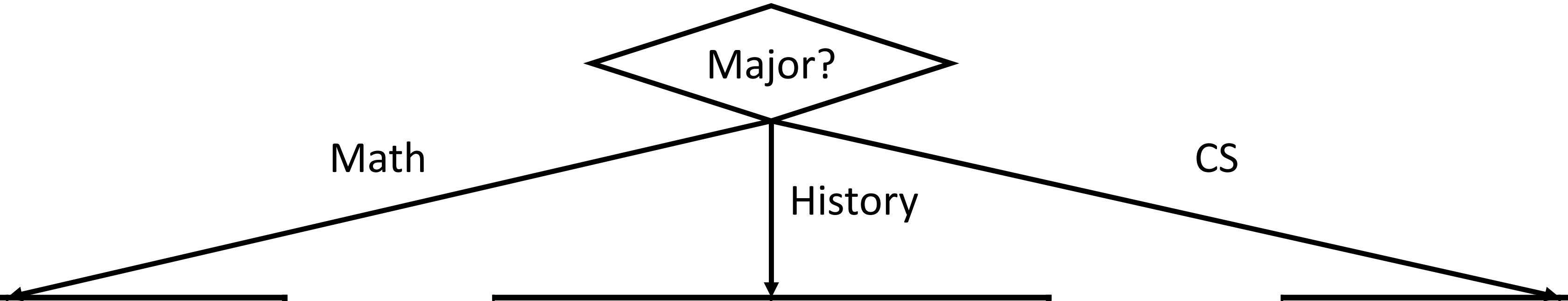
Gender	Like
Male	Yes
Female	No
Female	No
Female	Yes

Gender	Like
Female	No
Male	No

Gender	Like
Male	Yes
Female	Yes

Example (cont.)

- The history and CS subset contain only one label, so we only need to further expand the math subset

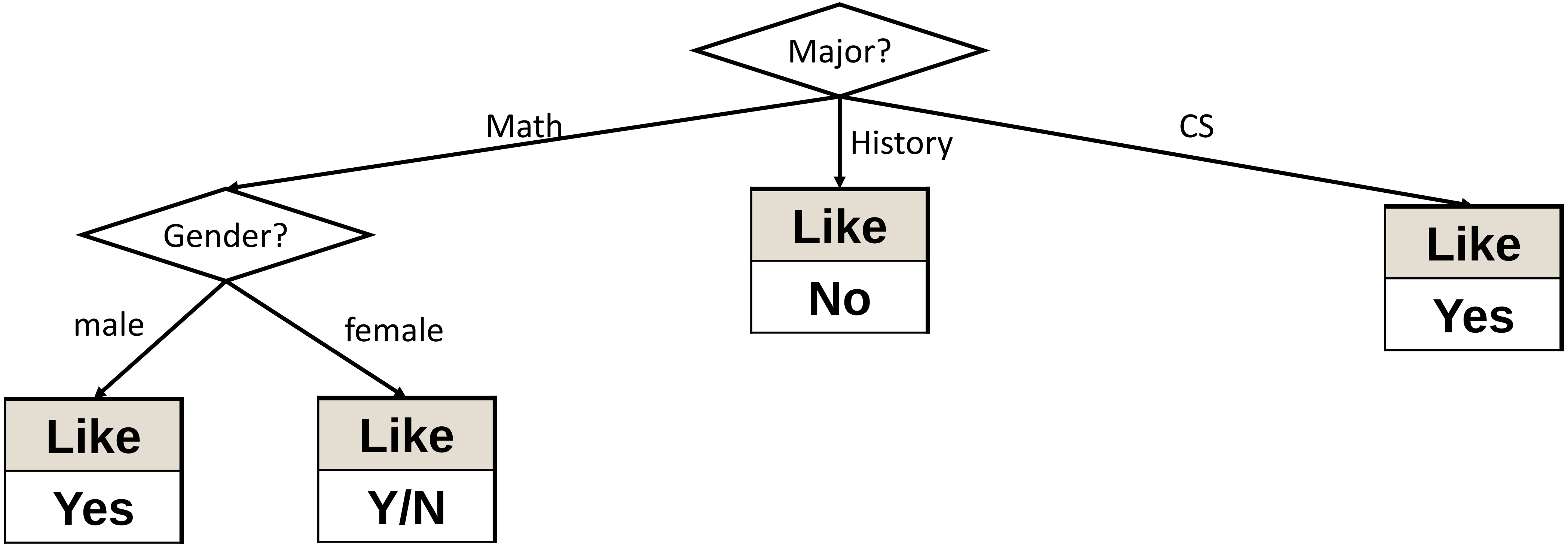


Gender	Like
Male	Yes
Female	No
Female	No
Female	Yes

Gender	Like
Female	No
Male	No

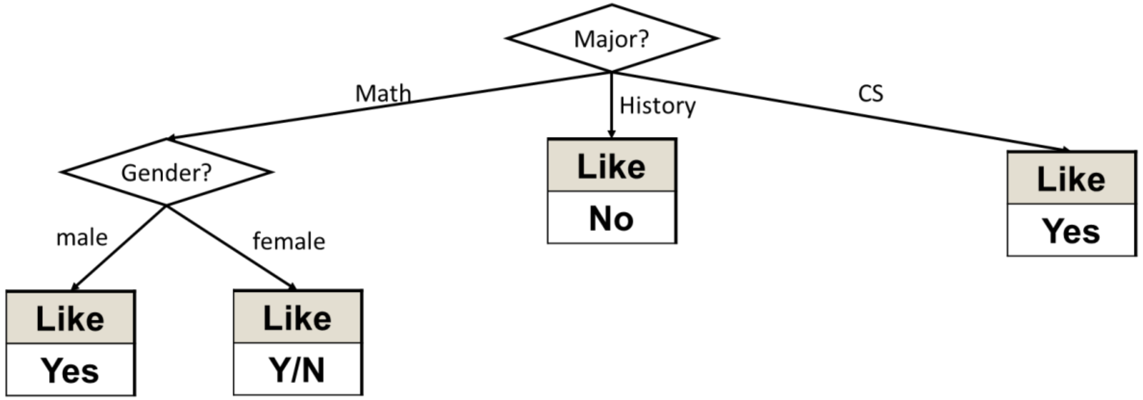
Gender	Like
Male	Yes
Female	Yes

Example (cont.)



Example (cont.)

- In the stage of testing, suppose there come a female students from the CS department, how can we predict whether she like the movie Gladiator?
  - Based on the major of CS, we will directly predict she like the movie.
  - What about a male student and a female student from math department?



Decision tree building: ID3 algorithm

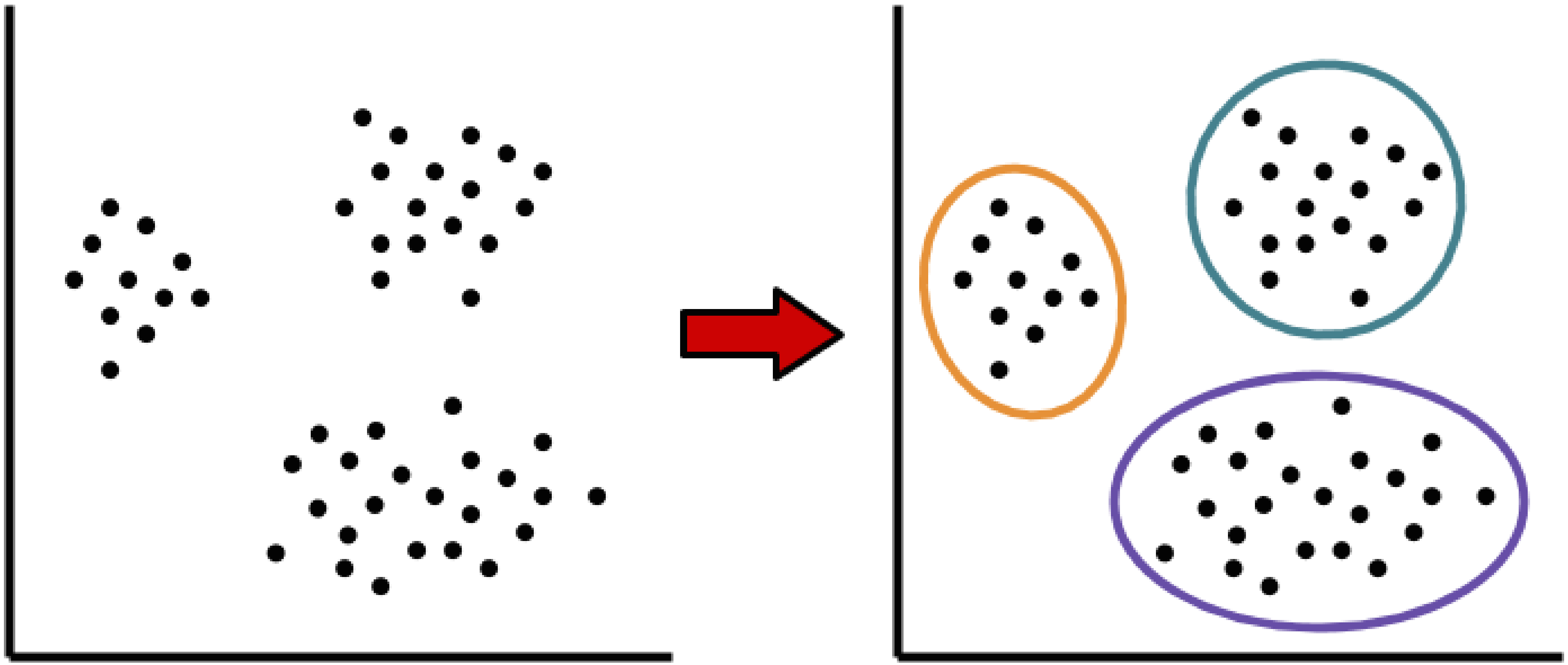
- Algorithm framework
  - Start from the root node with all data
  - For each node, calculate the information gain of all possible features
  - Choose the feature with the highest information gain
  - Split the data of the node according to the feature
  - Do the above recursively for each leaf node, until
    - There is no information gain for the leaf node
    - Or there is no feature to select
- Testing
  - Pass the example through the tree to the leaf node for a label

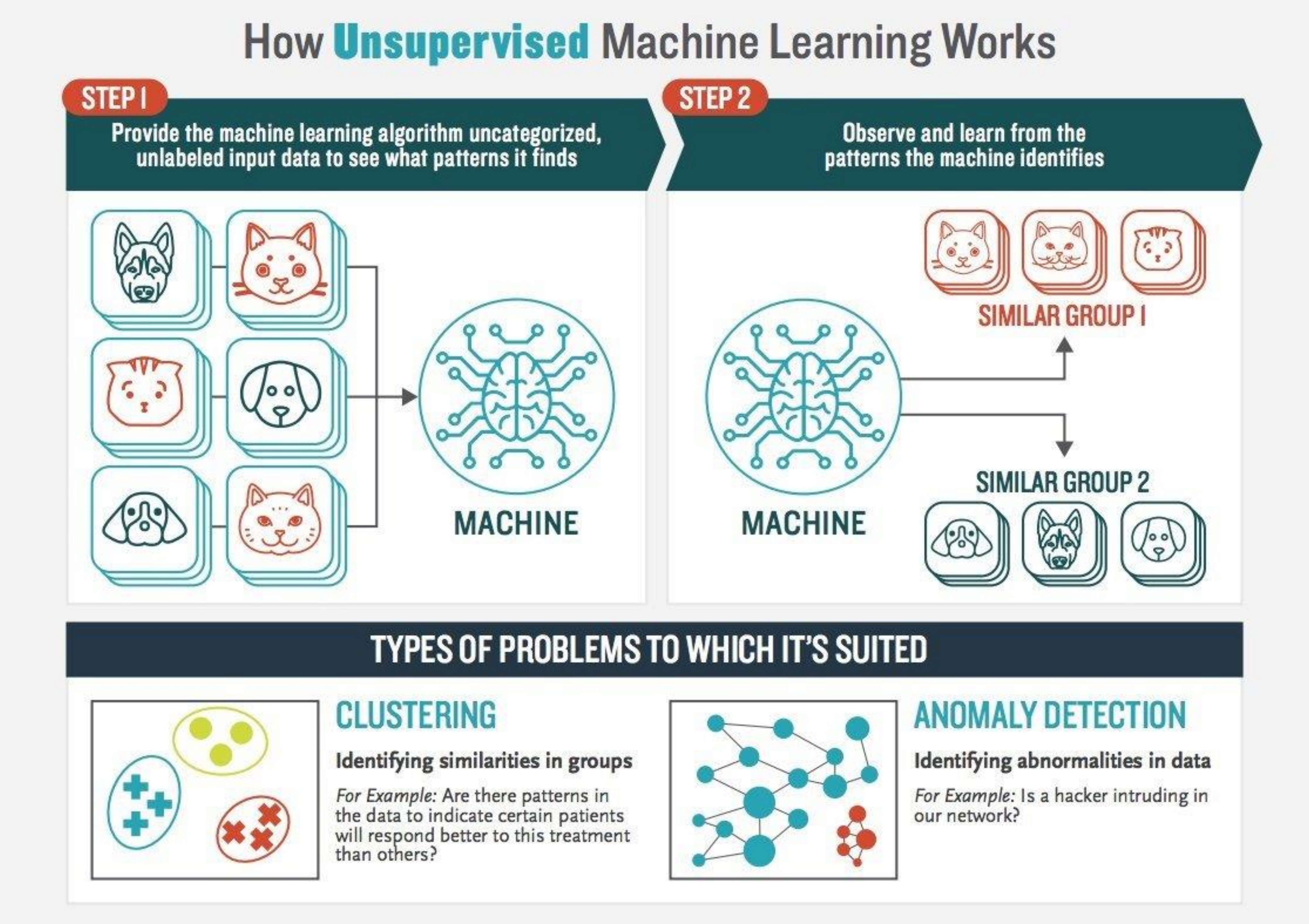
Unsupervised Learning

Machine learning categories

- Unsupervised learning
  - No labeled data
- Supervised learning
  - Use labeled data to predict on unseen points
- Semi-supervised learning
  - Use labeled data and unlabeled data to predict on unlabeled/unseen points
- Reinforcement learning
  - Sequential prediction and receiving feedbacks

Unsupervised learning example





How Unsupervised Machine Learning Works

STEP 1 Provide the machine learning algorithm uncategorized, unlabeled input data to see what patterns it finds

STEP 2 Observe and learn from the patterns the machine identifies

TYPES OF PROBLEMS TO WHICH IT'S SUITED

CLUSTERING

Identifying similarities in groups

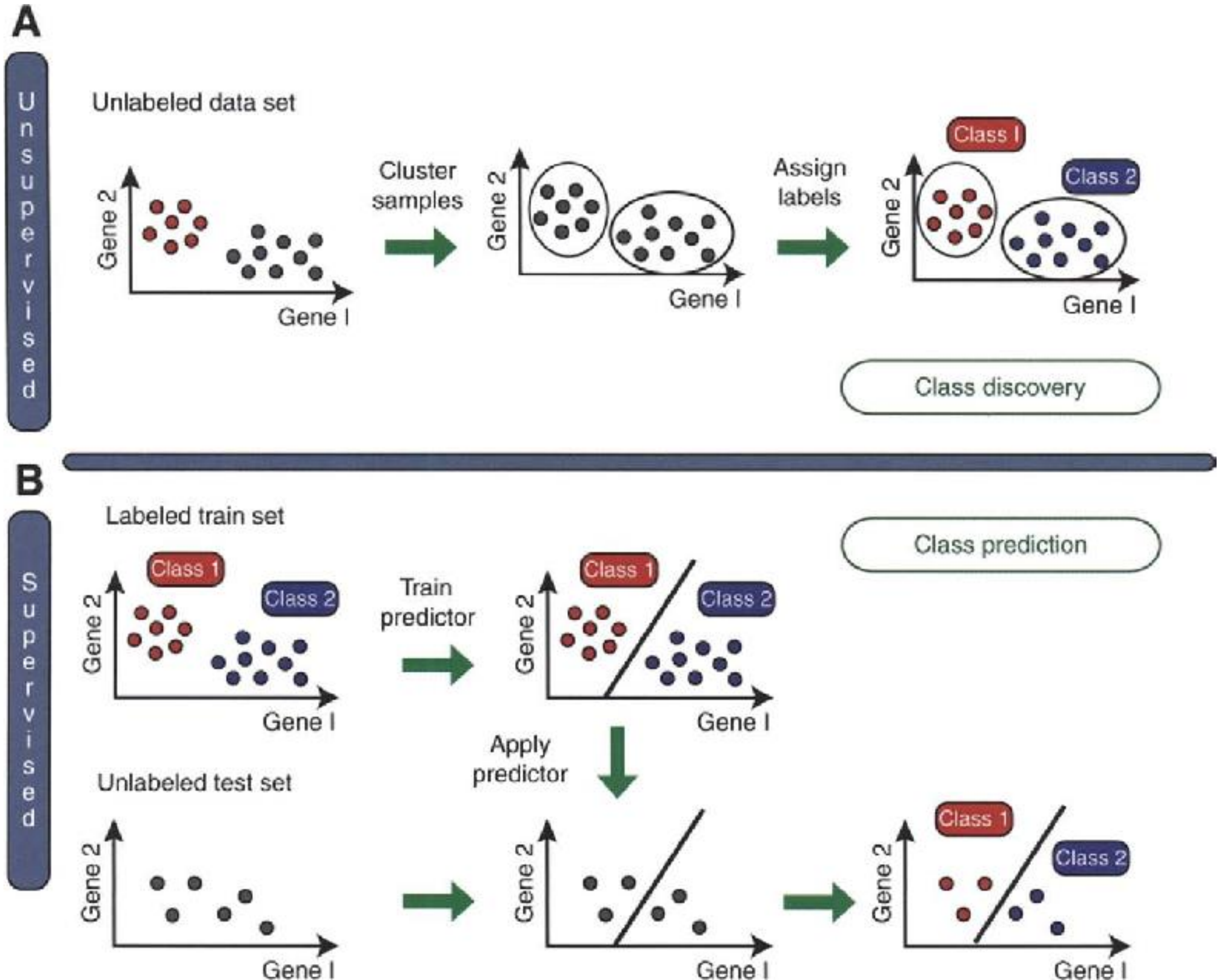
For Example: Are there patterns in the data to indicate certain patients will respond better to this treatment than others?

ANOMALY DETECTION

Identifying abnormalities in data

For Example: Is a hacker intruding in our network?

Supervised Learning	Unsupervised Learning
Input data is labelled	Input data is unlabeled
Uses training dataset	Uses just input dataset
Used for prediction	Used for analysis
Classification and regression	Clustering, density estimation and dimensionality reduction


A
Unsupervised
Unlabeled data set
Gene 2
Gene I
Cluster samples
Gene 2
Gene I
Assign labels
Class I
Class 2
Gene 2
Gene I
Class discovery
B
Supervised
Labeled train set
Class 1
Class 2
Gene 2
Gene I
Train predictor
Class 1
Class 2
Gene 2
Gene I
Class prediction
Apply predictor
Unlabeled test set
Gene 2
Gene I
Gene 2
Gene I
Class 1
Class 2
Gene 2
Gene I

Clustering

Clustering

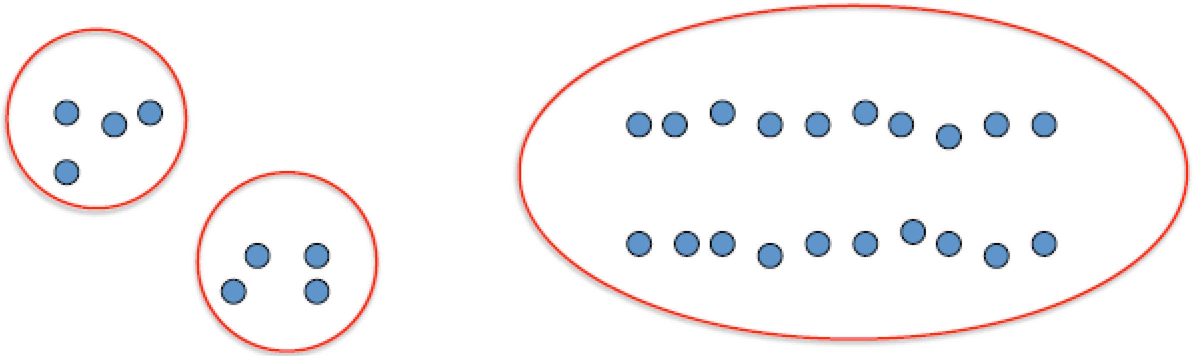
- Unsupervised learning
- Requires data, but no labels
- Detect patterns e.g. in
  - Group emails or search results
  - Customer shopping patterns
  - Regions of images
- Useful when don’t know what you’re looking for
- But: can get gibberish

Clustering (cont.)

- Goal: Automatically segment data into groups of similar points
- Question: When and why would we want to do this?
- Useful for:
    - Automatically organizing data
    - Understanding hidden structure in some data
    - Representing high-dimensional data in a low-dimensional space
- Examples: Cluster
    - customers according to purchase histories
    - genes according to expression profile
    - search results according to topic
    - Facebook users according to interests
    - a museum catalog according to image similarity

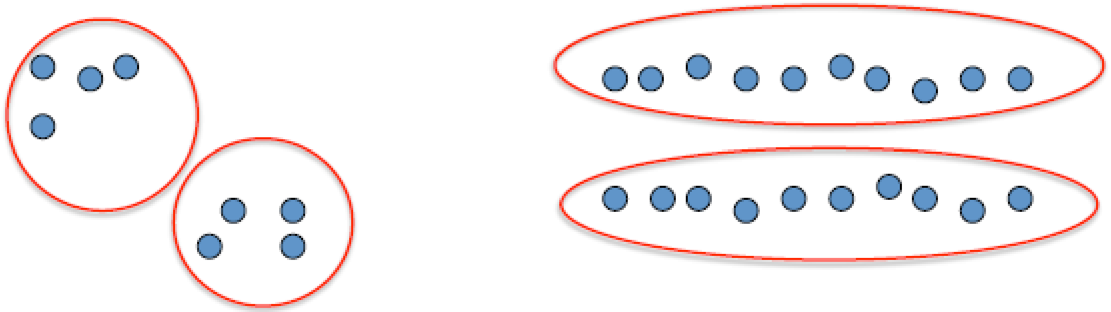
Intuition

- Basic idea: group together similar instances
- Example: 2D point patterns



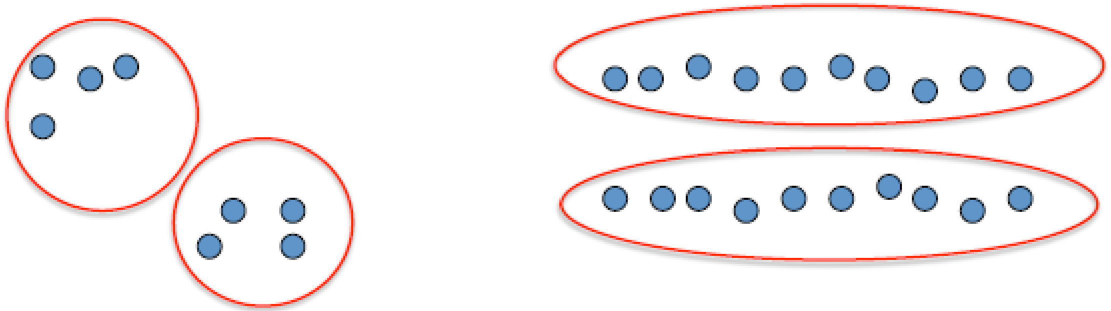
Intuition (cont.)

- Basic idea: group together similar instances
- Example: 2D point patterns



Intuition (cont.)

- Basic idea: group together similar instances
- Example: 2D point patterns



- What could “similar” mean?
- – One option: small Euclidean distance (squared)
- – Clustering results are crucially dependent on the measure of similarity (or distance) between “points” to be clustered

Set-up

- Given the data: $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

- Each data point x is d-dimensional:
$$x_i = (x_{i,1}, \ldots, x_{i,d})$$

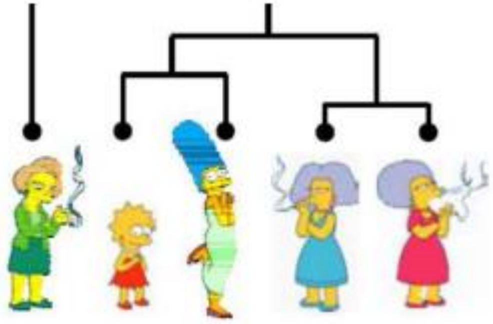
- Define a distance function between data:
$$d(\mathbf{x}_n, \mathbf{x}_m).$$

- Goal: segment the data into K groups

Clustering algorithms

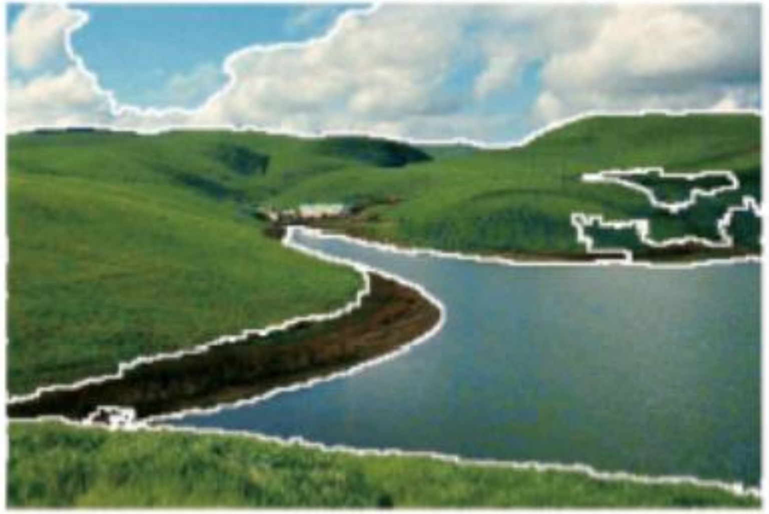
- Partition algorithms (flat clustering)
  - K-means
  - Mixture of Gaussian
  - Spectral Clustering

- Hierarchical algorithms
  - Bottom up-agglomerative
  - Top down-divisive



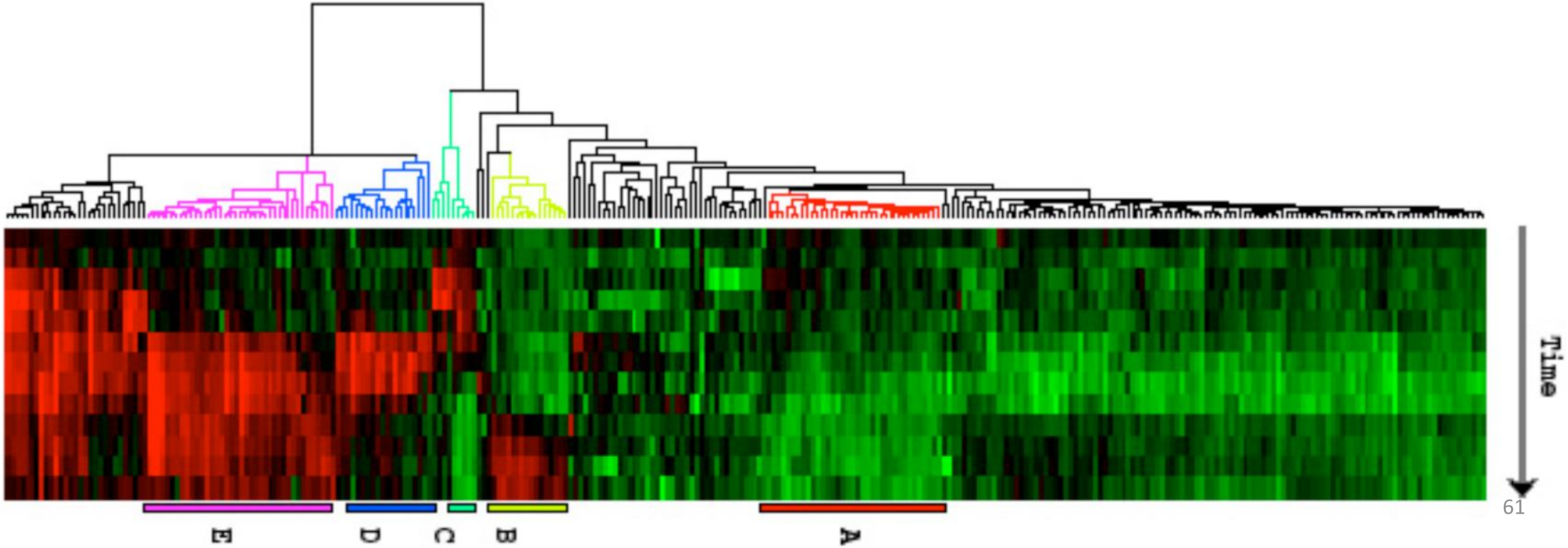
Example

- Image segmentation
- Goal: Break up the image into meaningful or perceptually similar regions



Example 2

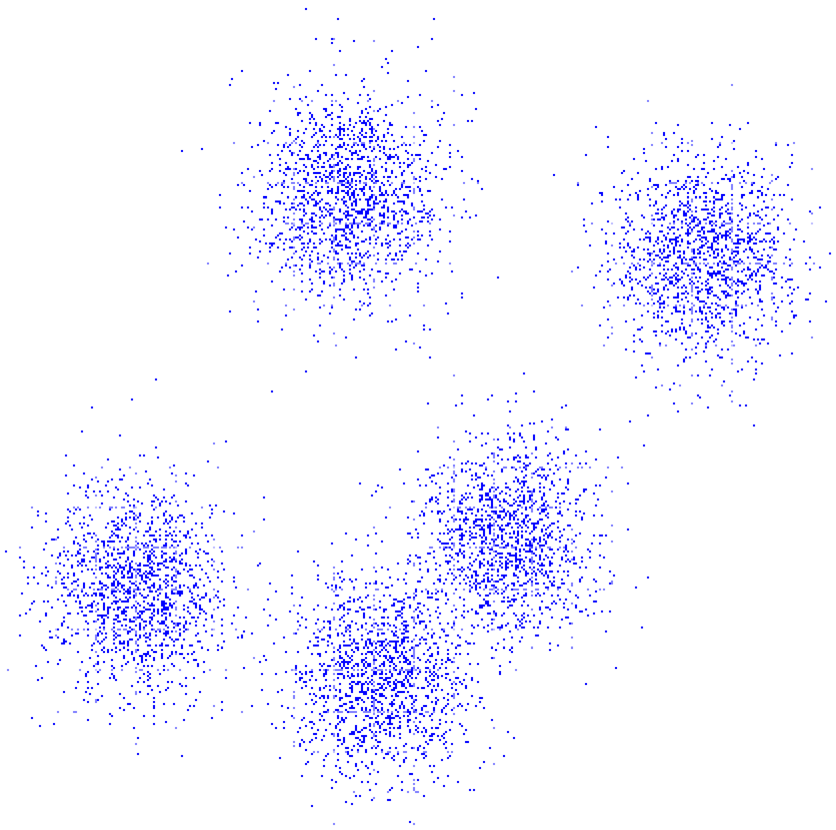
- Gene expression data clustering
  - Activity level of genes across time



K-Means

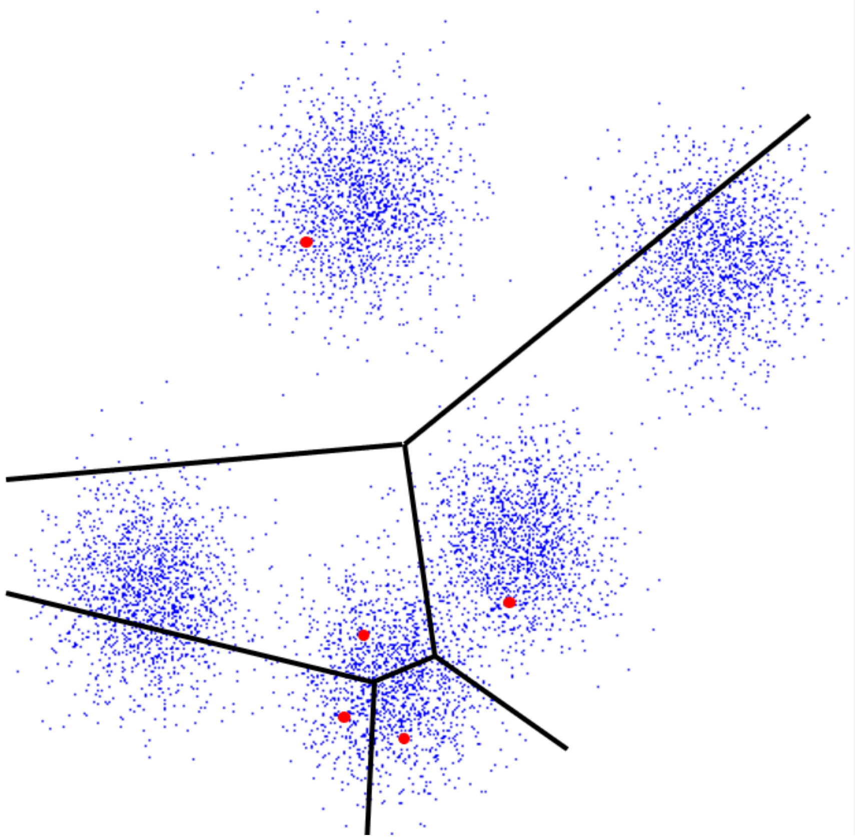
K-Means

- An iterative clustering algorithm
- Initialize: Pick K random points as cluster centers
- Alternate:
  - Assign data points to closest cluster center
  - Change the cluster center to the average of its assigned points
- Stop: when no points' assignments change



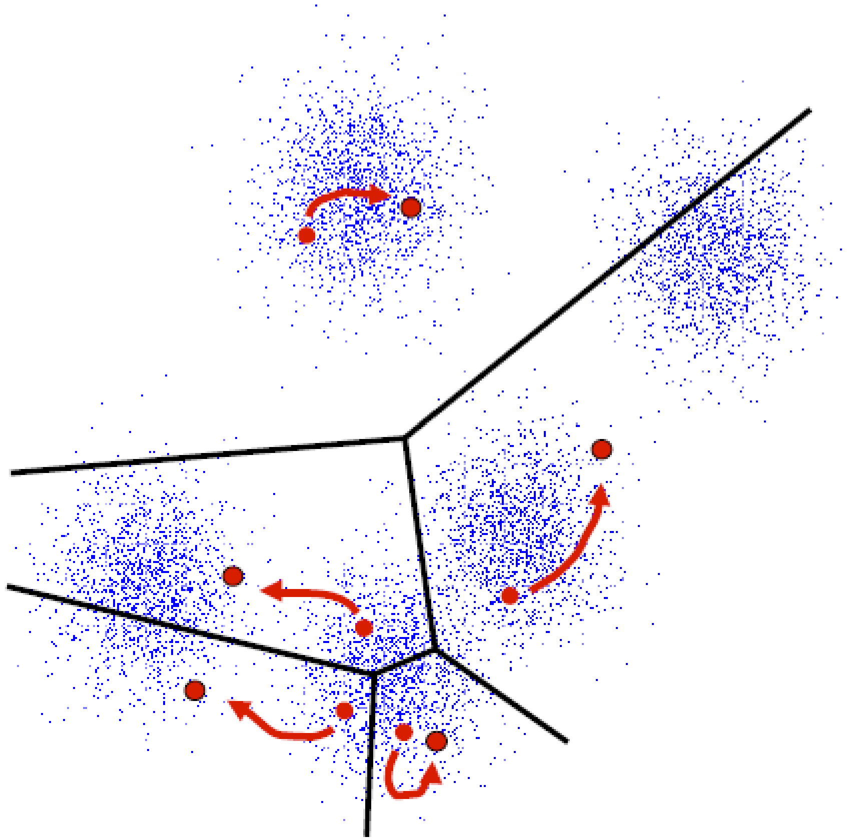
K-Means (cont.)

- An iterative clustering algorithm
- Initialize: Pick K random points as cluster centers
- Alternate:
  - Assign data points to closest cluster center
  - Change the cluster center to the average of its assigned points
- Stop: when no points' assignments change



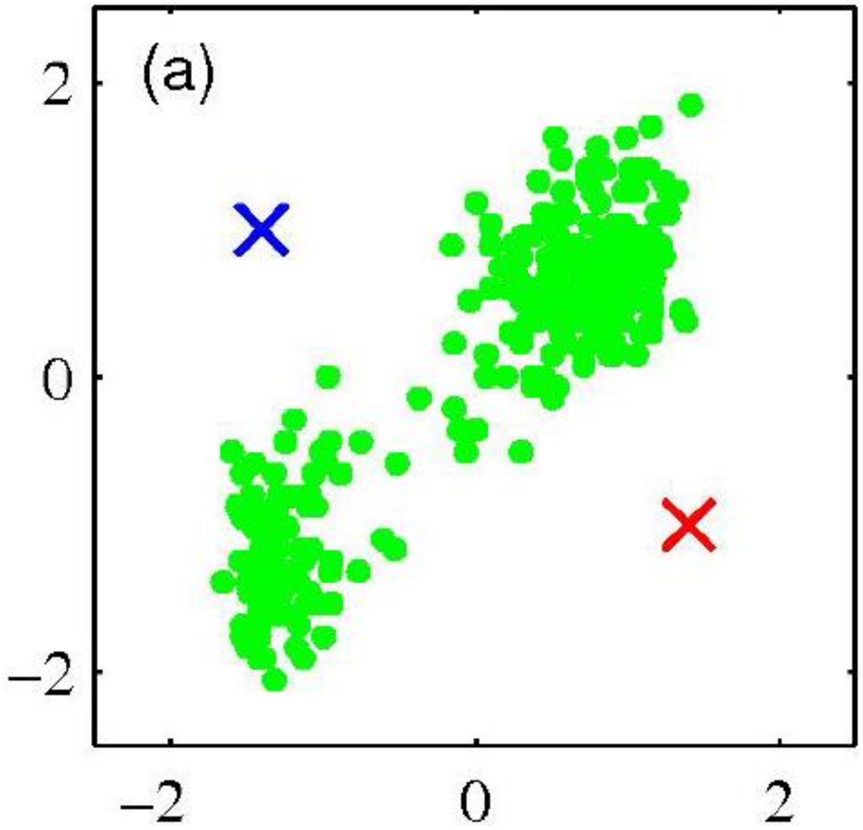
K-Means (cont.)

- An iterative clustering algorithm
- Initialize: Pick K random points as cluster centers
- Alternate:
  - Assign data points to closest cluster center
  - Change the cluster center to the average of its assigned points
- Stop: when no points' assignments change



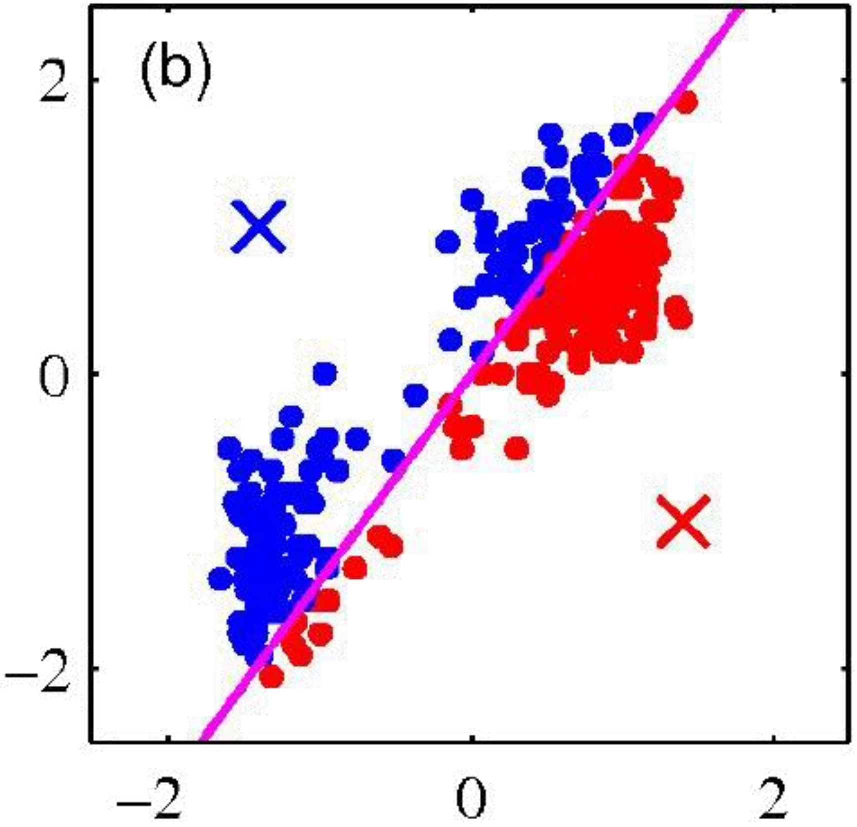
Example

- Pick K random points as cluster centers (means)
- Shown here for $K = 2$



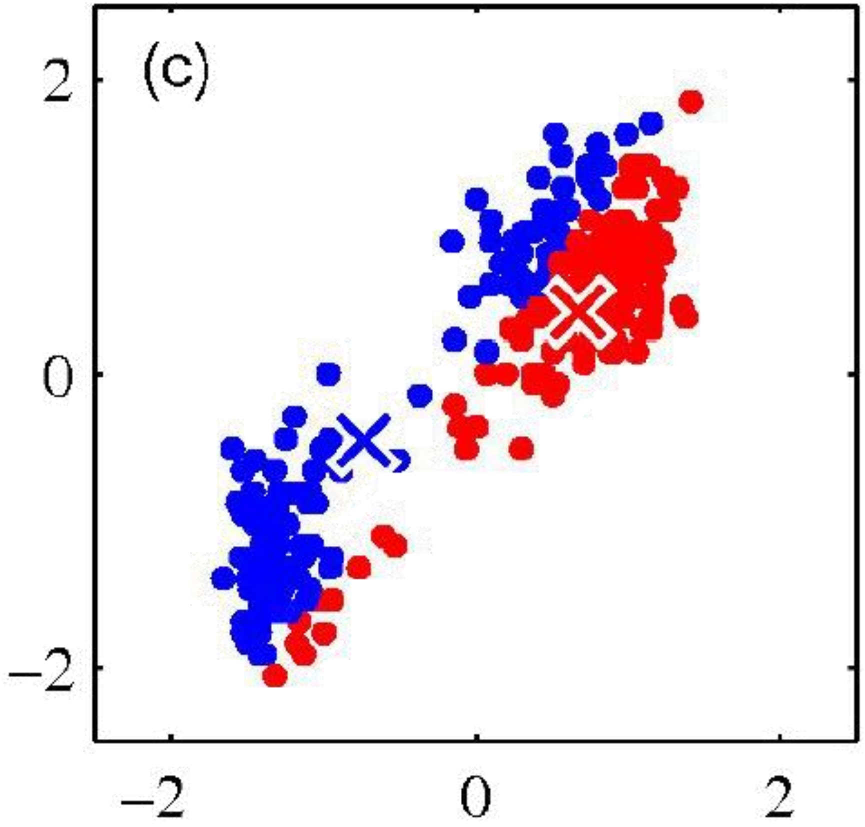
Example (cont.)

- Iterative step 1
- Assign data points to closest cluster center



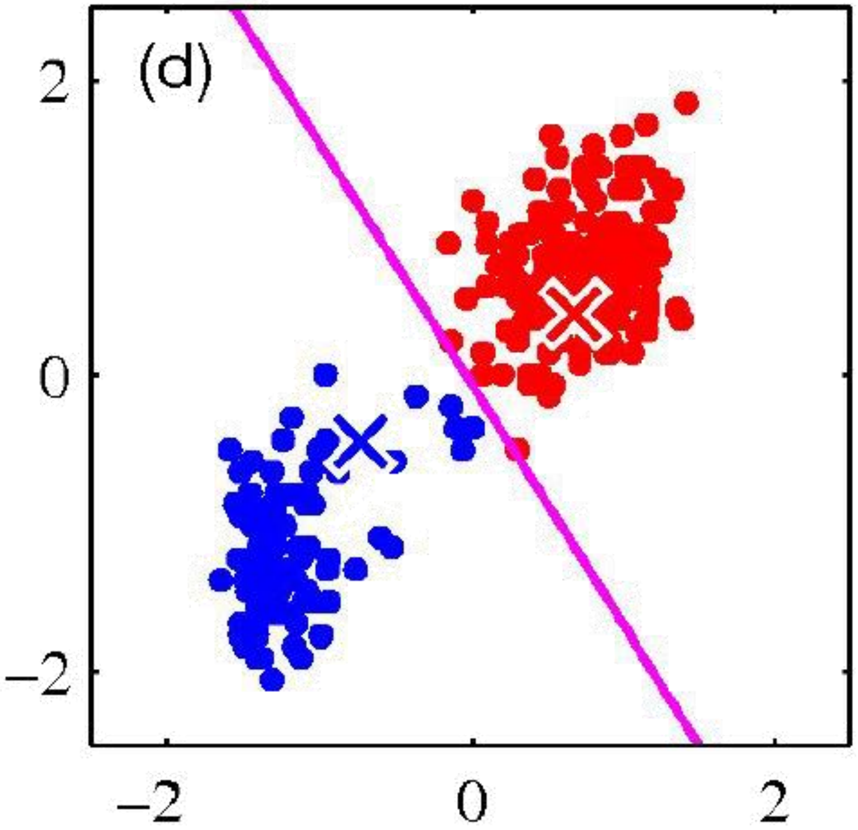
Example (cont.)

- Iterative step 2
- Change the cluster center to the average of the assigned points



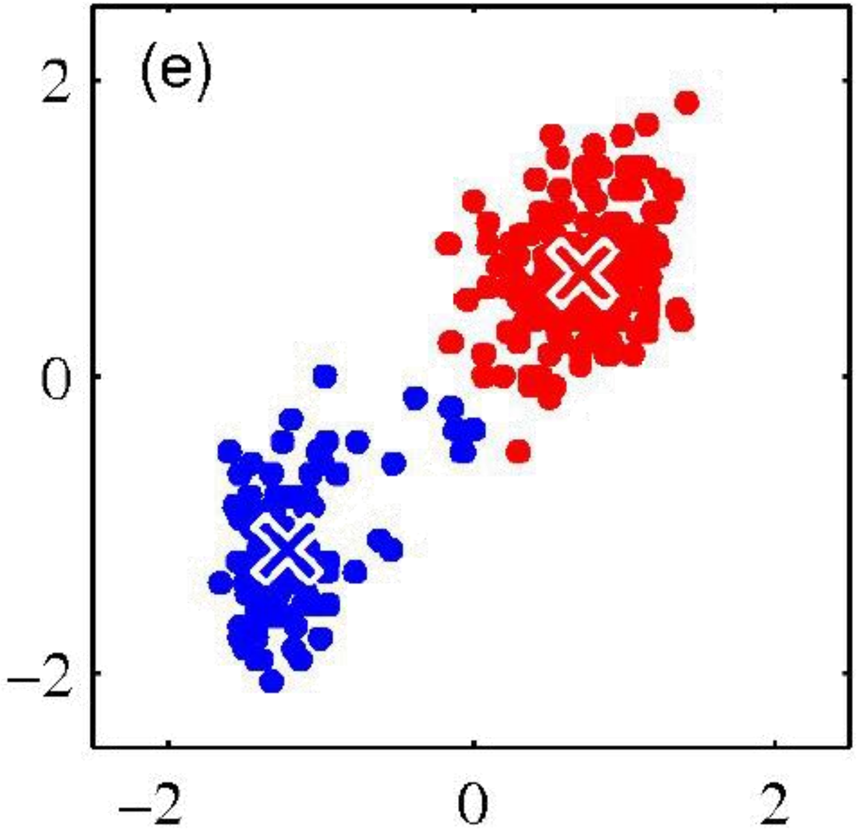
Example (cont.)

- Repeat until convergence
- Convergence means that the differences of the center positions in two continuous loops is smaller than a threshold



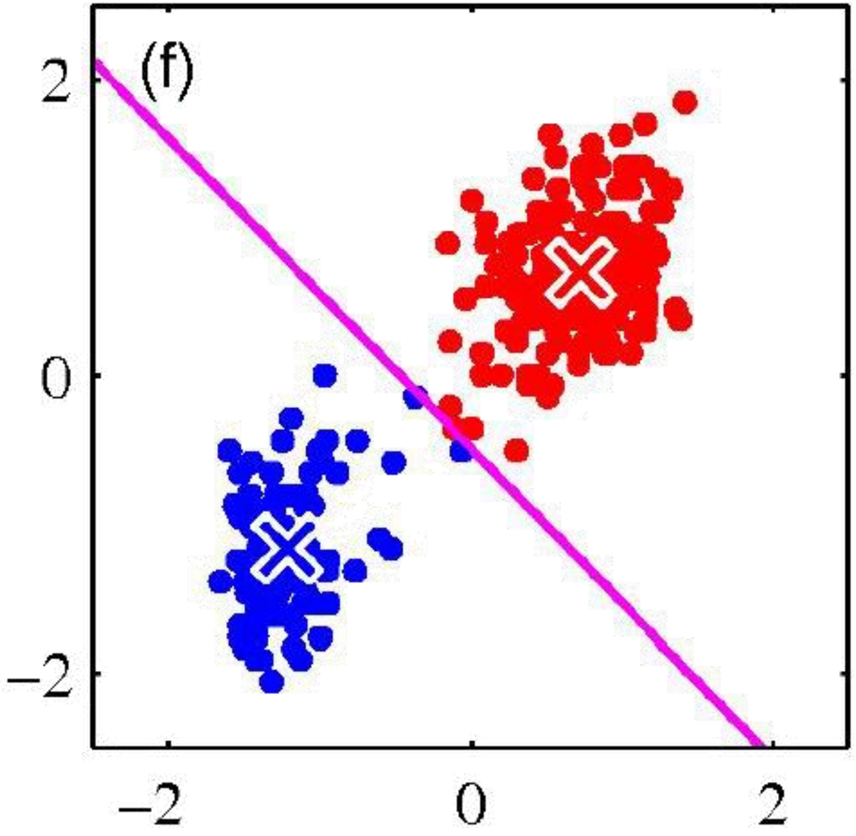
Example (cont.)

- Repeat until convergence
- Convergence means that the differences of the center positions in two continuous loops is smaller than a threshold



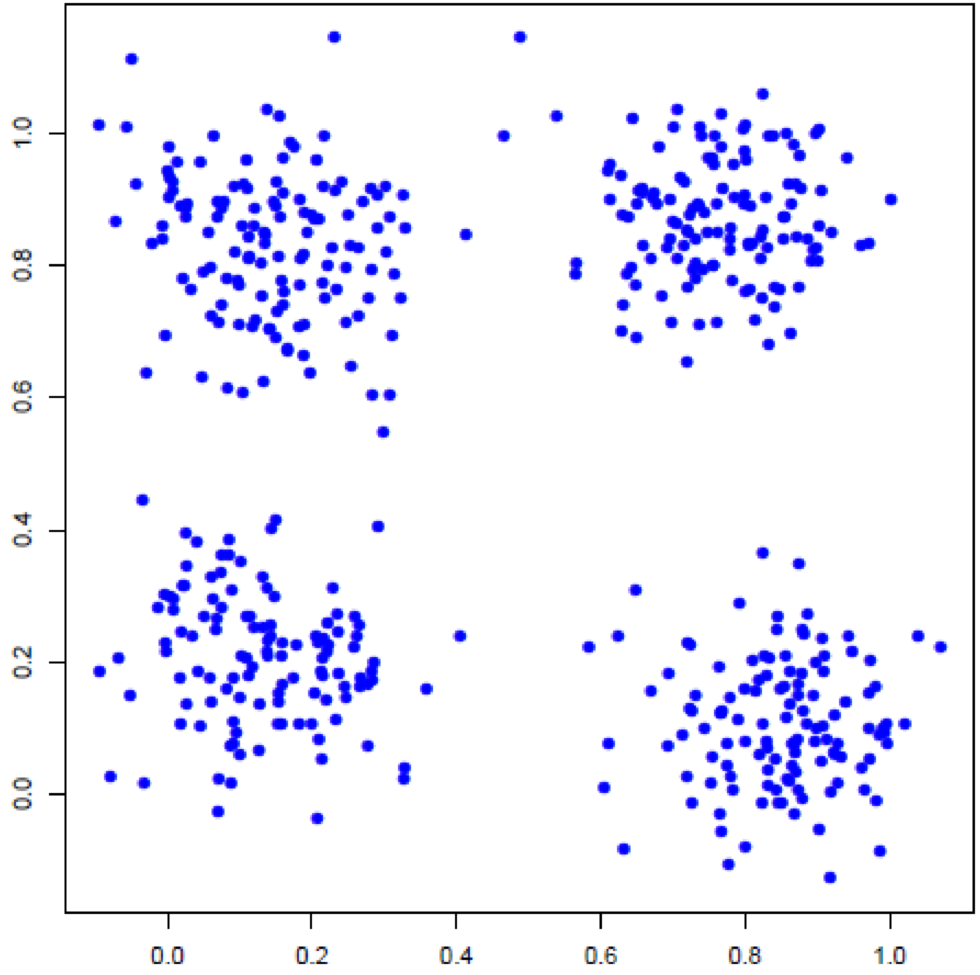
Example (cont.)

- Repeat until convergence
- Convergence means that the differences of the center positions in two continuous loops is smaller than a threshold



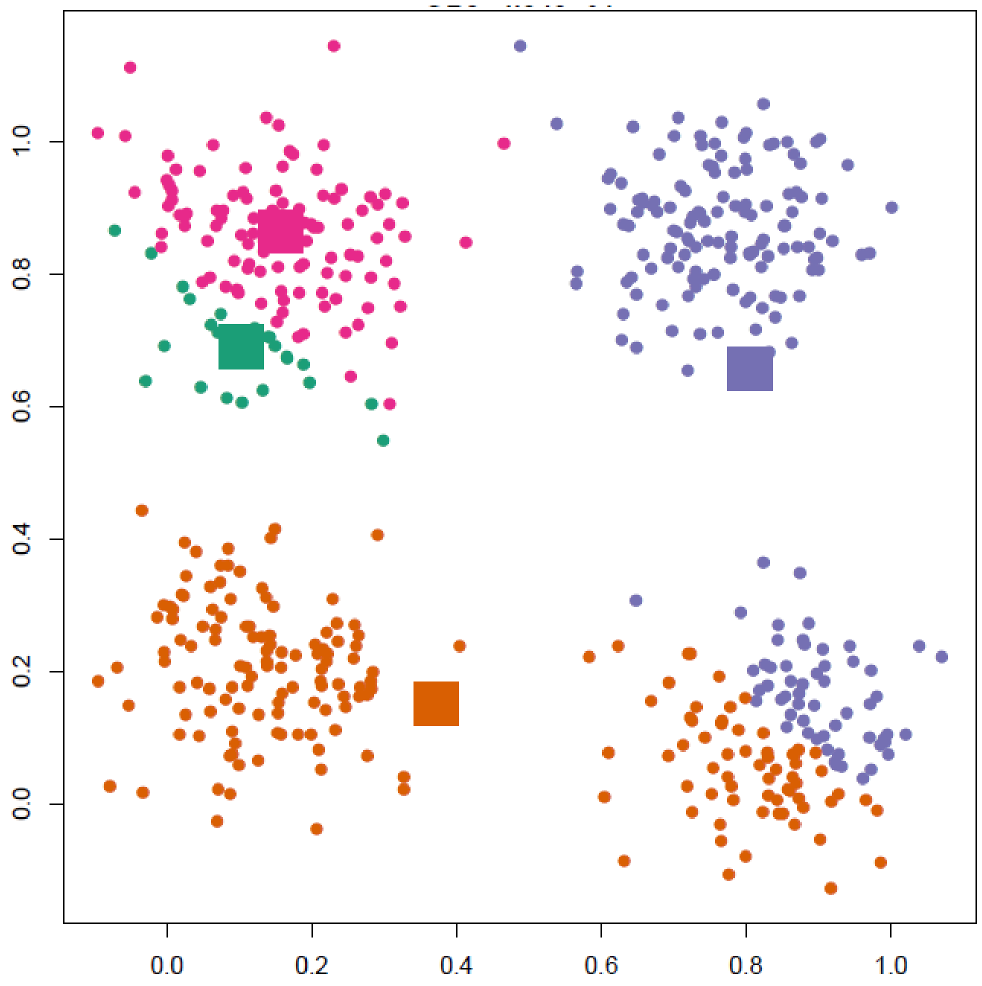
Example 2

- $K = 4$



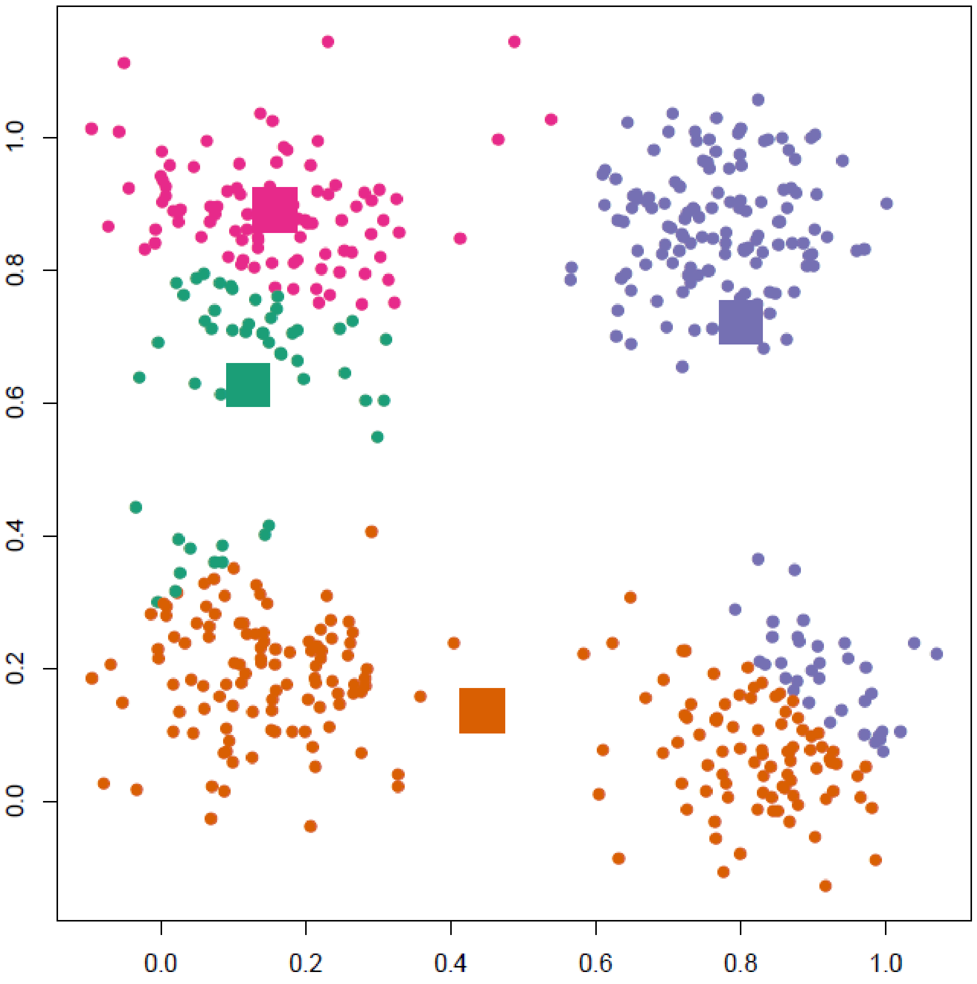
Example 2 (cont.)

- $K = 4$



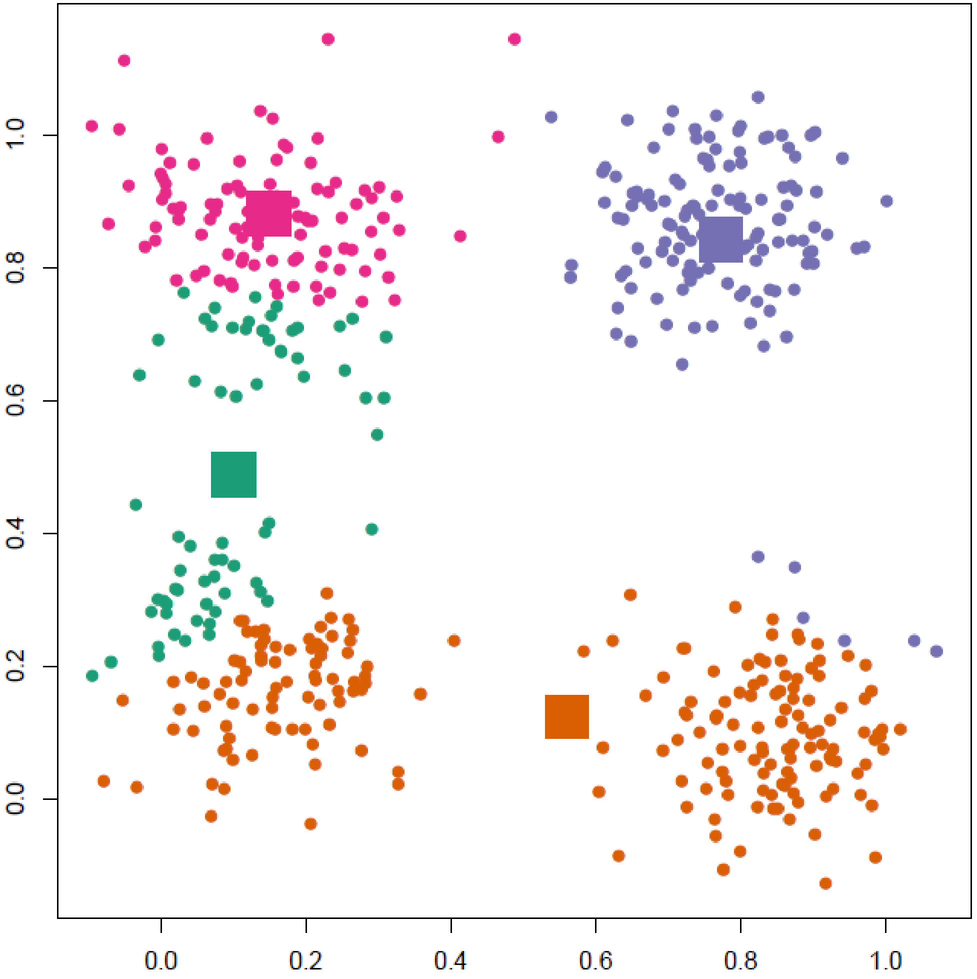
Example 2 (cont.)

- $K = 4$



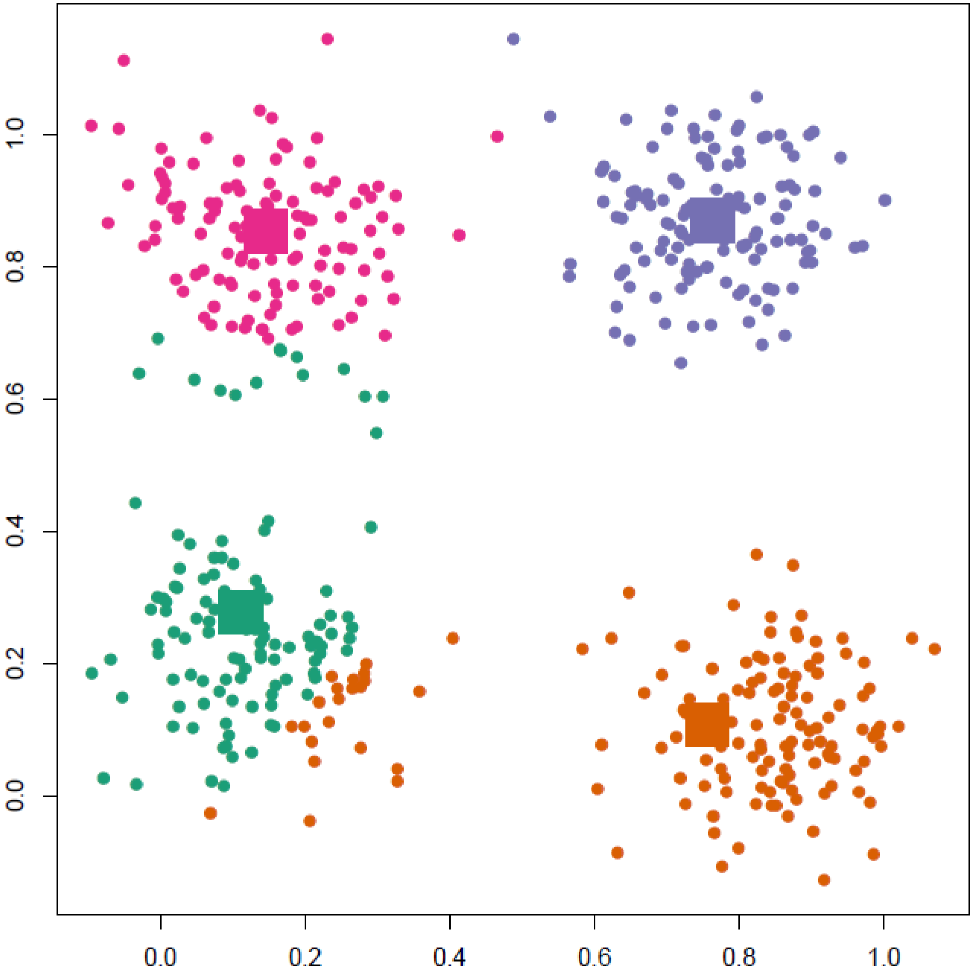
Example 2 (cont.)

- $K = 4$



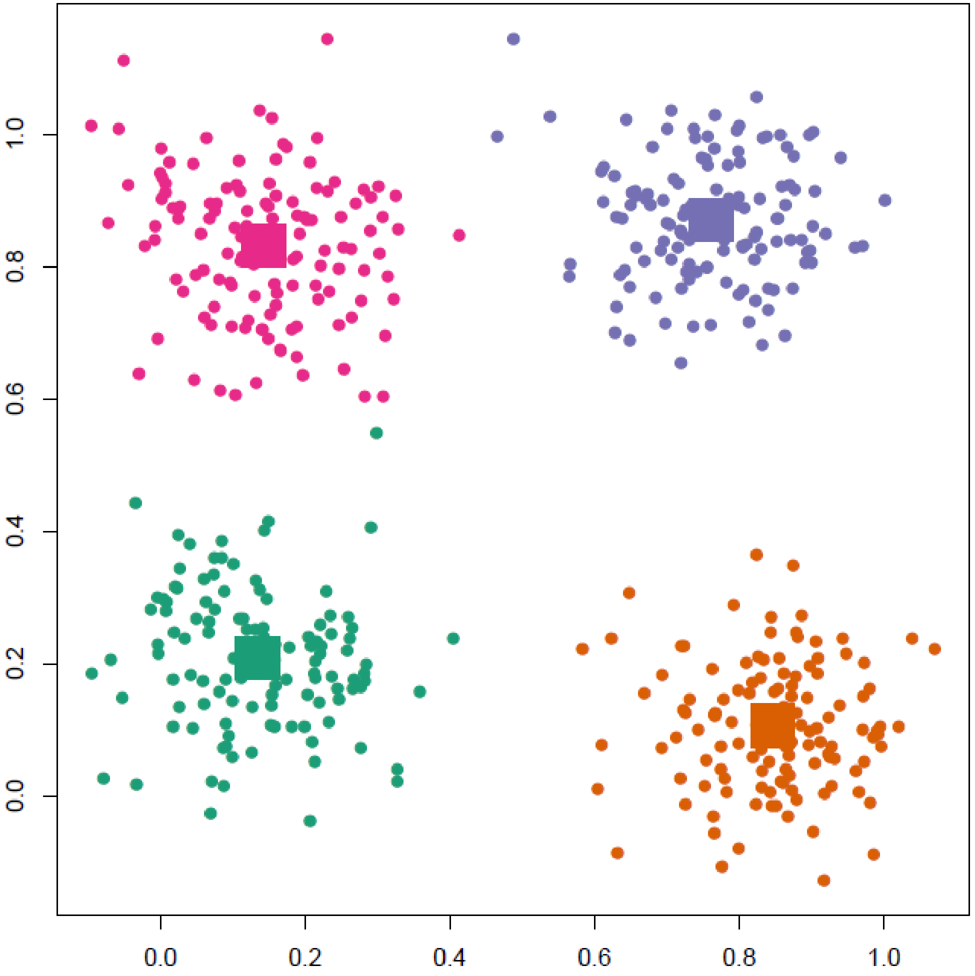
Example 2 (cont.)

- $K = 4$



Example 2 (cont.)

- $K = 4$



Remained Questions in K-Means

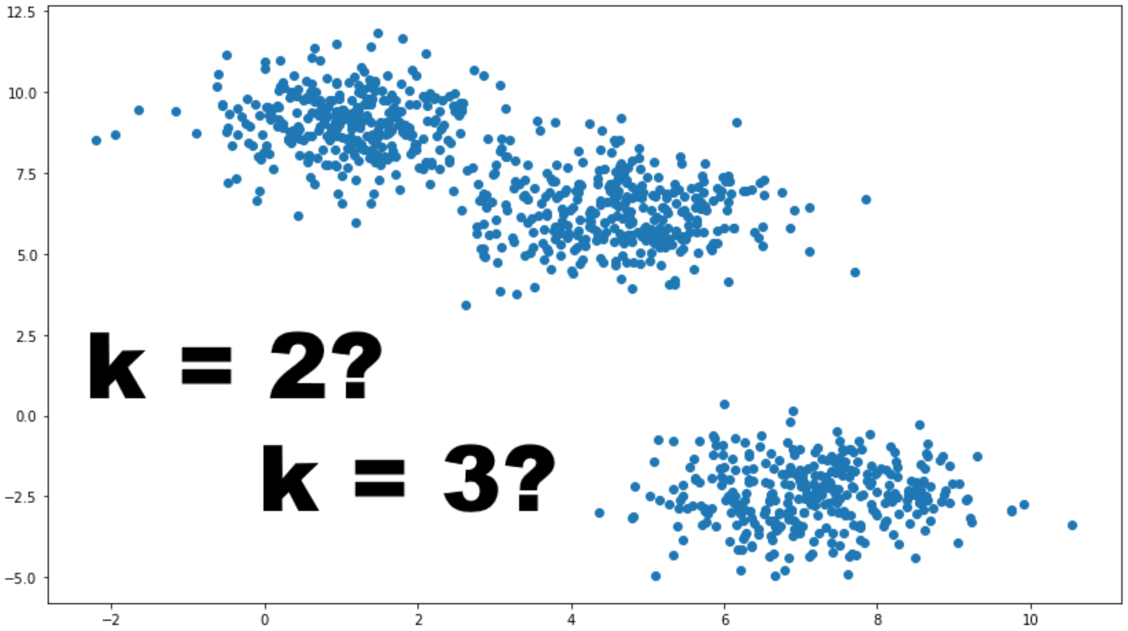
Remained questions in K-means

- Although the workflow of K-means is straight forward, there are some important questions that need to be discussed

- How to choose the hyper-parameter K?
- How to initialize?

How to choose K?

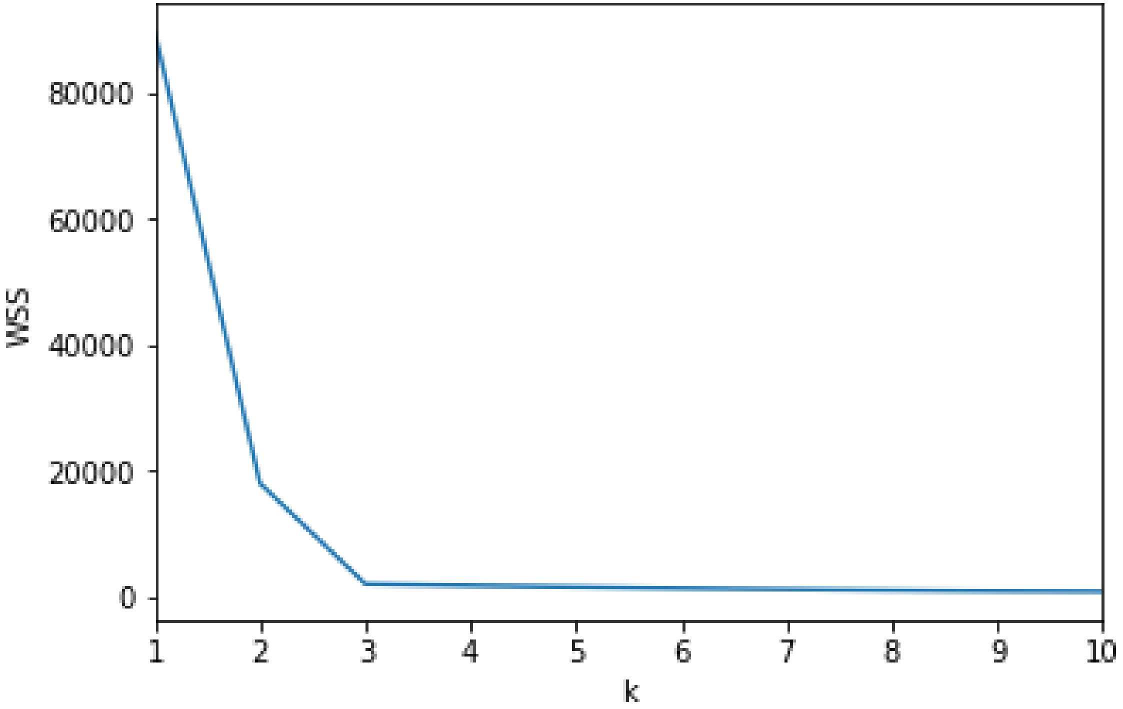
- K is the most important hyper-parameter in K-means which strongly affects its performance. In some situation, it’s not an easy task to find the proper K
- The solution includes:
  - The elbow method
  - The silhouette method



The elbow method

- Calculate the Within-Cluster-Sum of Squared Errors (WSS) for different values of K, and choose the K ***for which WSS stops dropping significantly***. In the plot of WSS-versus-k, this is visible as an elbow

- Example: $K = 3$



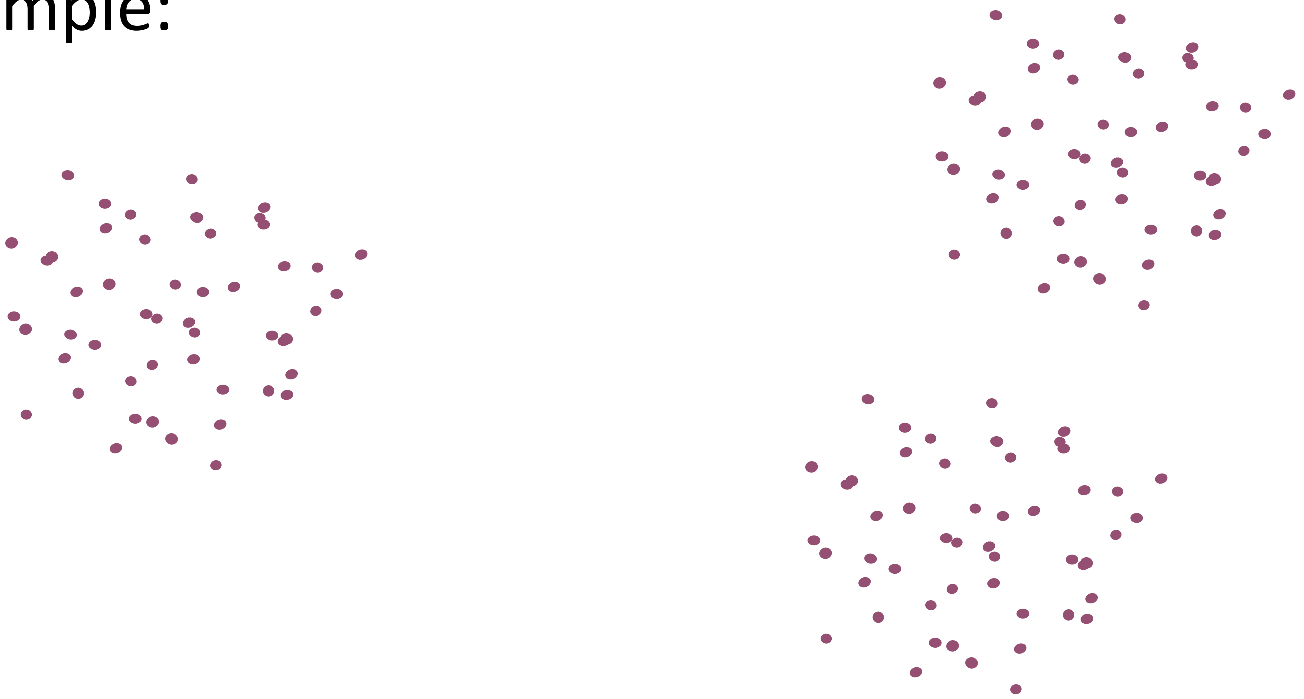
The silhouette method

- The problem of the elbow method is that in many situations the most suitable K cannot be unambiguously identified. So we need the silhouette method

- The silhouette value measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation). The range of the silhouette value is between +1 and -1. A high value is desirable and indicates that the point is placed in the correct cluster

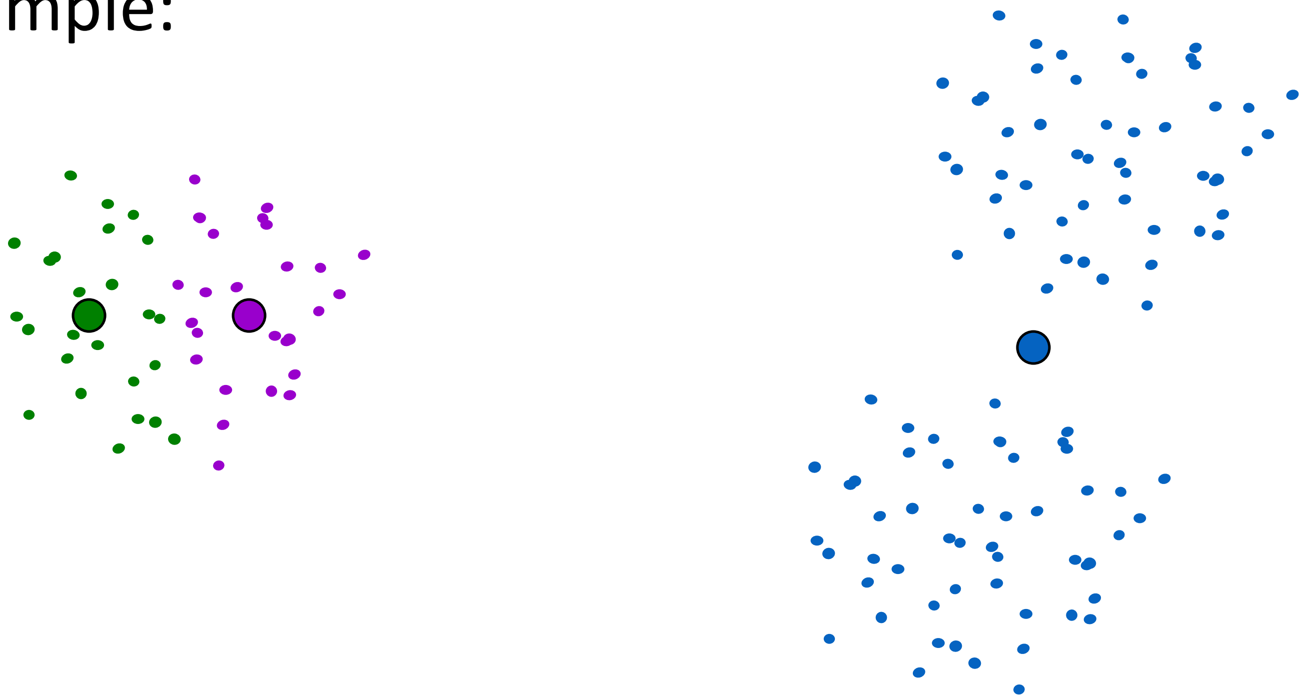
How to initialize center positions?

- The positions of the centers in the stage of initialization are also very important in K-means algorithms. In some situations it can produce totally different clustering results
- Example:



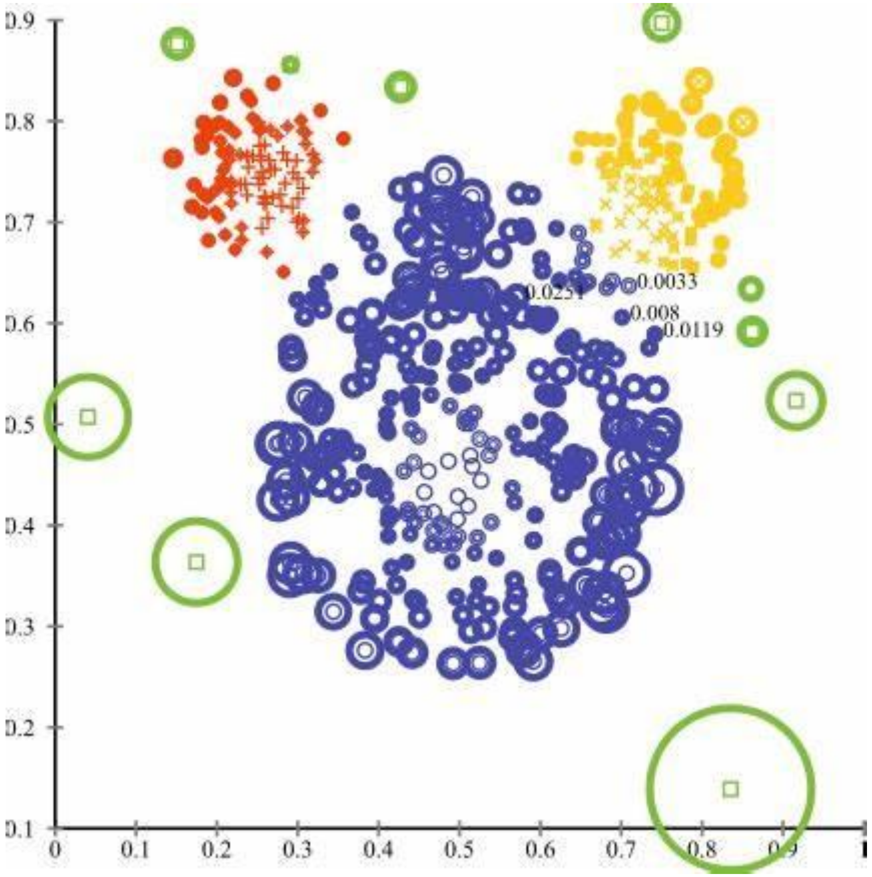
How to initialize center positions? (cont.)

- The positions of the centers in the stage of initialization are also very important in K-means algorithms. In some situations it can produce totally different clustering results
- Example:



A possible solution

- Pick one point at random, then $K - 1$ other points, each as far away as possible from the previous points
    - OK, as long as there are no *outliers* (points that are far from any reasonable cluster)

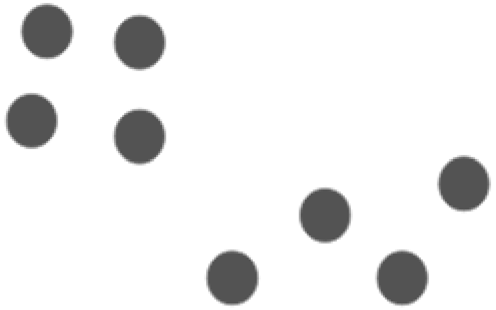


K-means++

1. The first centroid is chosen uniformly at random from the data points that we want to cluster. This is similar to what we do in K-Means, but instead of randomly picking all the centroids, we just pick one centroid here
2. Next, we compute the distance d_x is the nearest distance from data point x to the centroids that have already been chosen
3. Then, choose the new cluster center from the data points with the probability of x being proportional to d_x^2
4. We then repeat steps 2 and 3 until K clusters have been chosen

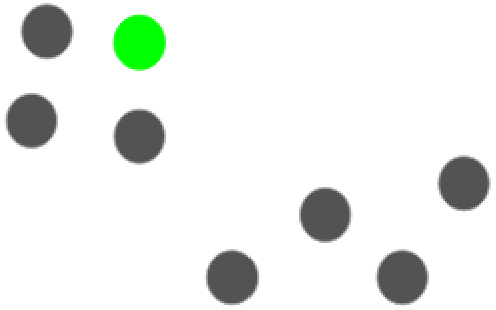
Example

- Suppose we have the following points and we want to make 3 clusters here:



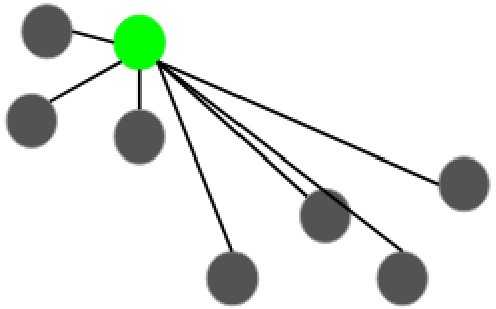
Example (cont.)

- First step is to randomly pick a data point as a cluster centroid:



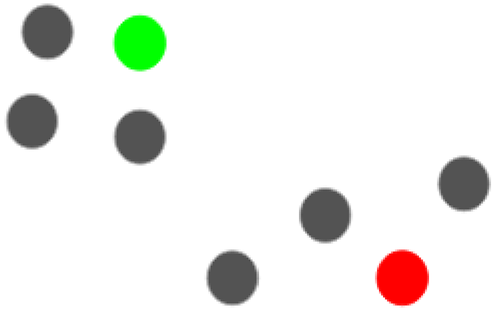
Example (cont.)

- Calculate the distance d_x of each data point with this centroid:



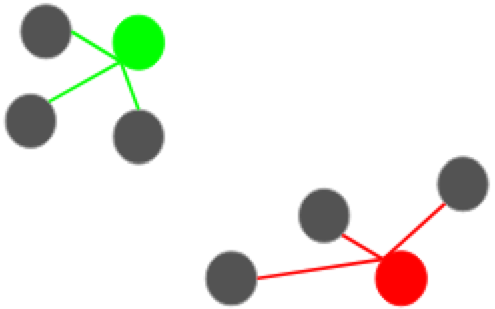
Example (cont.)

- The next centroid will be sampled with the probability proportional to d_x^2
- Say the sampled is the red one



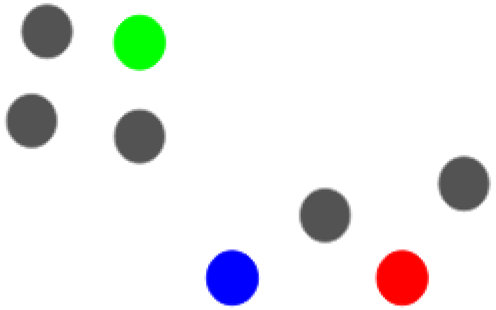
Example (cont.)

- To select the last centroid, compute d_x, which is the distance to its closest centroid



Example (cont.)

- Sample the one with the probability proportional to d_x^2
- Say, the blue one



Properties of K-Means

How to measure the performance

- K-means can be evaluated by the sum of distance from points to corresponding centers, or the WSS

number of clusters

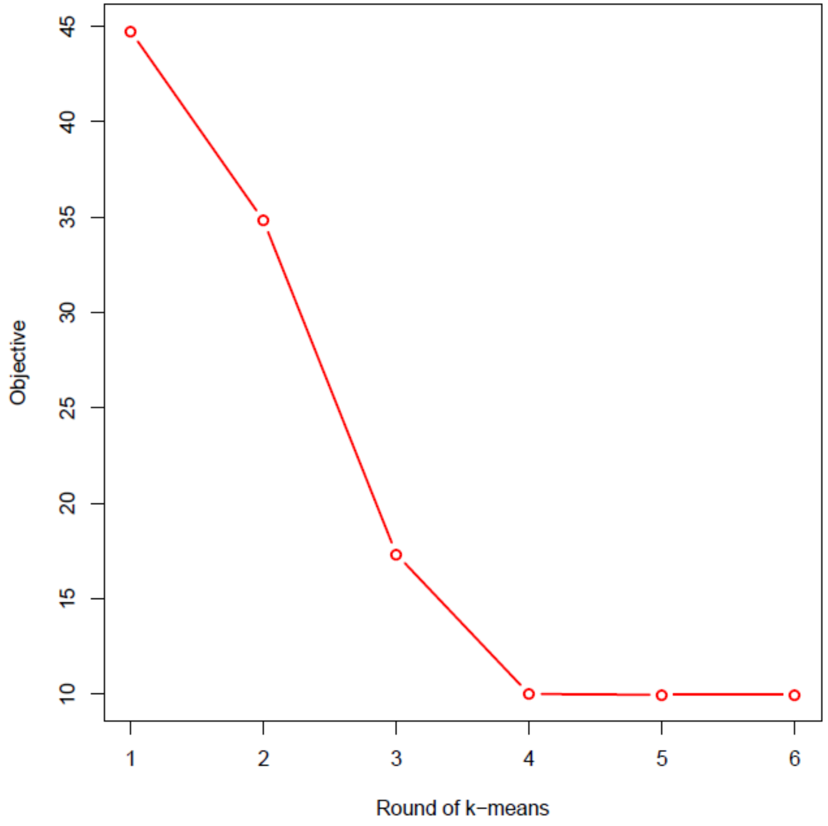
number of cases

centroid for cluster *j*

case *i*

$$\text{objective function} \leftarrow J = \sum_{j=1}^{k}\sum_{i=1}^{n}\underbrace{\left\|x_i^{(j)} - c_j\right\|^2}_{\text{Distance function}}$$

- The loss will approach zero when increase K



Properties of the K-means algorithm

- Guaranteed to converge in a finite number of iterations

- Running time per iteration:
  1. Assign data points to closest cluster center
     O(KN) time
  2. Change the cluster center to the average of its assigned points
     O(N)

Convergence of K-means

Objective

$$\min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

1. Fix μ, optimize *C:*

$$\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2 = \min_{c} \sum_{i}^{n} \left| x_i - \mu_{x_i} \right|^2$$

Step 1 of kmeans

2. Fix *C*, optimize μ:

$$\min_{\mu} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

– Take partial derivative of μ_i and set to zero, we have

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

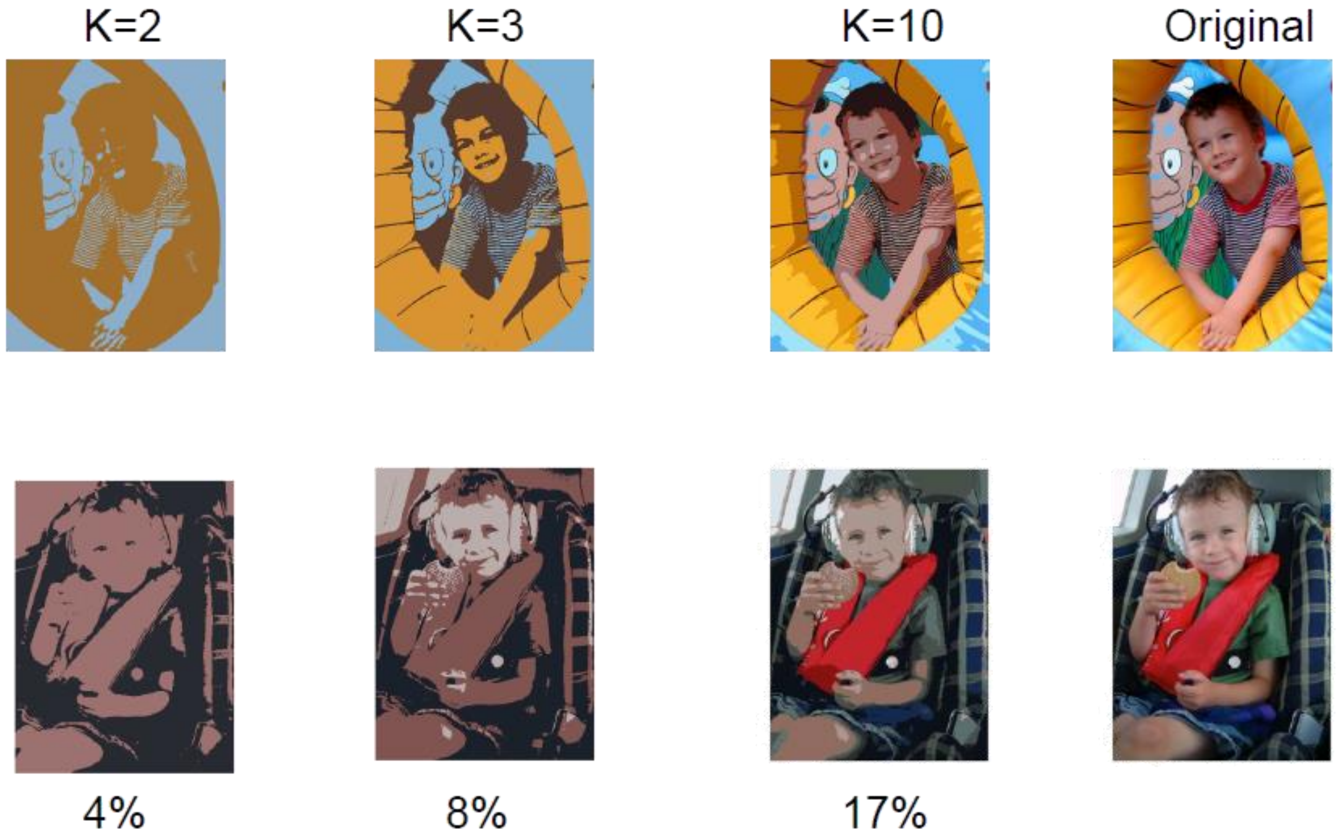
Step 2 of kmeans

Not guaranteed to converge to optimal

Kmeans takes an alternating optimization approach, each step is guaranteed to decrease the objective – thus guaranteed to converge

Application: Segmentation

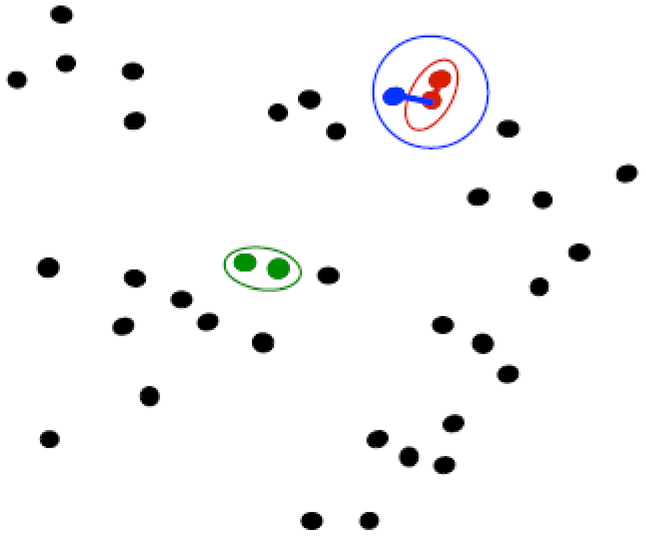
- Goal of segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance
- Cluster the colors



Agglomerative Clustering

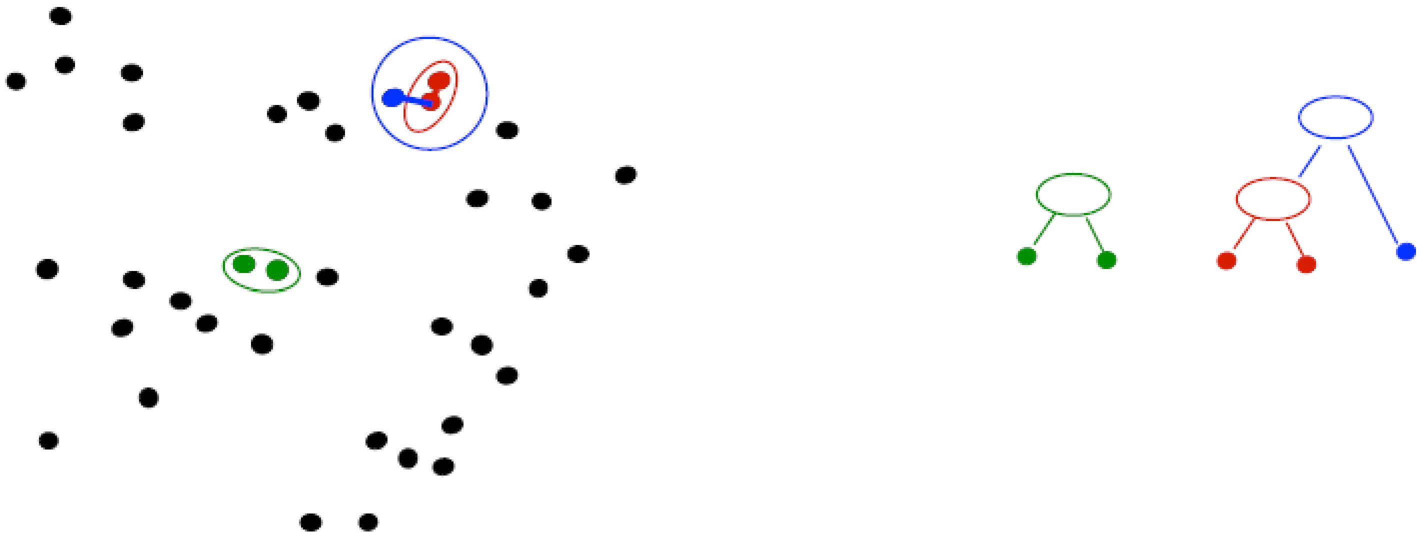
Agglomerative clustering

- Agglomerative clustering:
    - First merge very similar instances
    - Incrementally build larger clusters out of smaller clusters
- Algorithm:
    - Maintain a set of clusters
    - Initially, each instance in its own cluster
    - Repeat:
        - Pick the two closest clusters
        - Merge them into a new cluster
        - Stop when there's only one cluster left



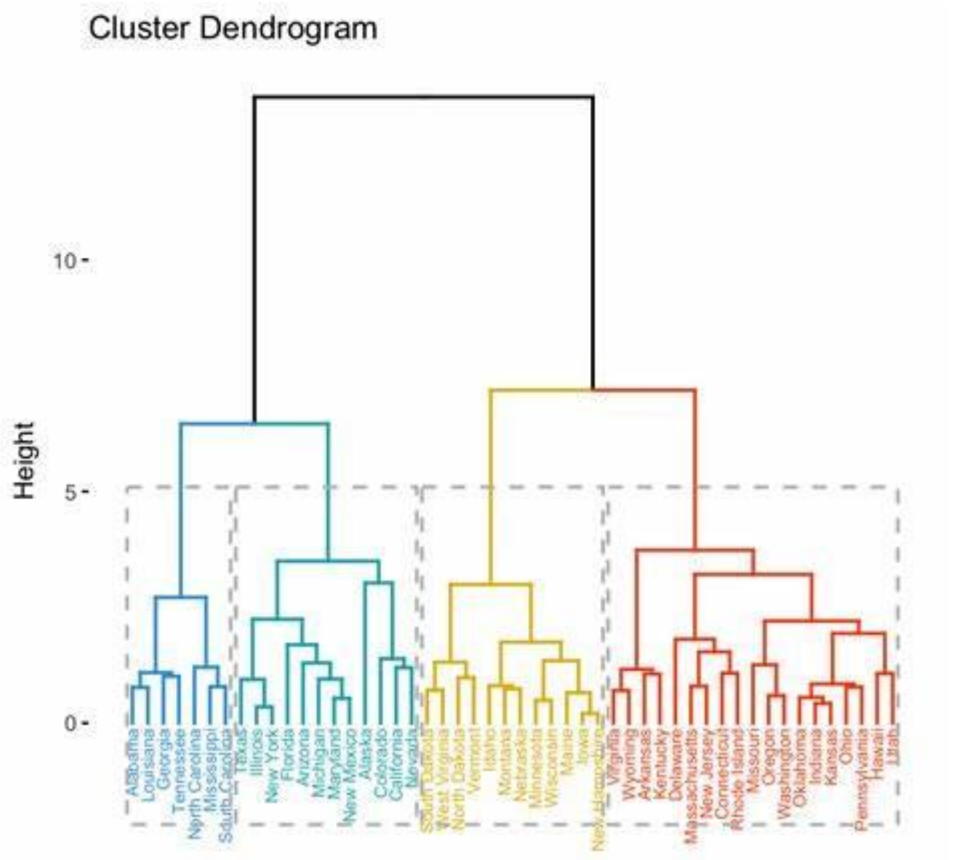
Agglomerative clustering

- Produces not one clustering, but a family of clusterings represented by a dendrogram



Example

- Different heights give different clustering

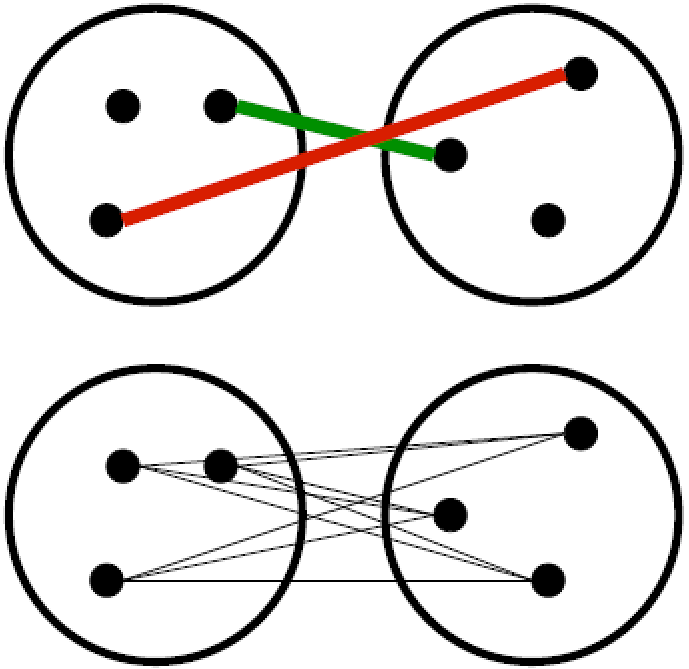


Closeness

- How should we define “closest” for clusters with multiple elements?

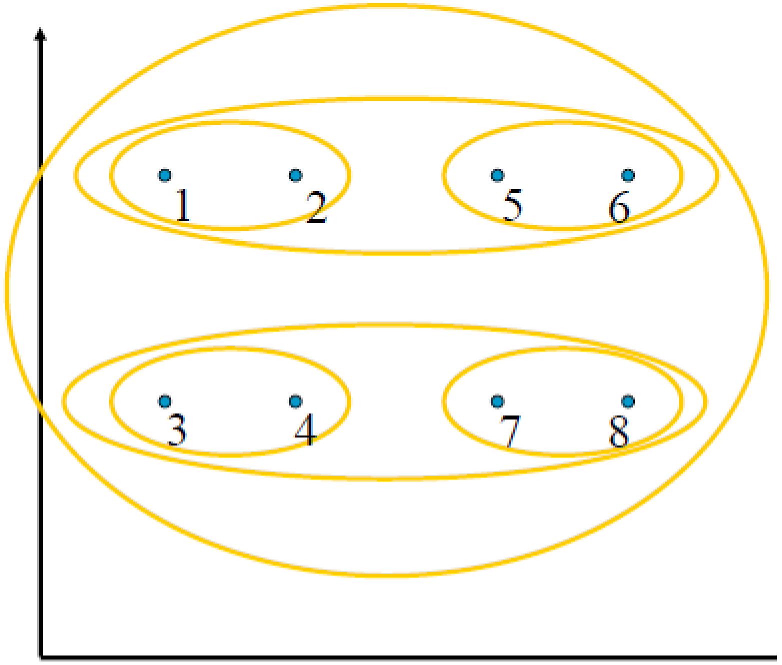
- Many options:
  - Closest pair (single-link clustering)
  - Farthest pair (complete-link clustering)
  - Average of all pairs

- Different choices create different clustering behaviors

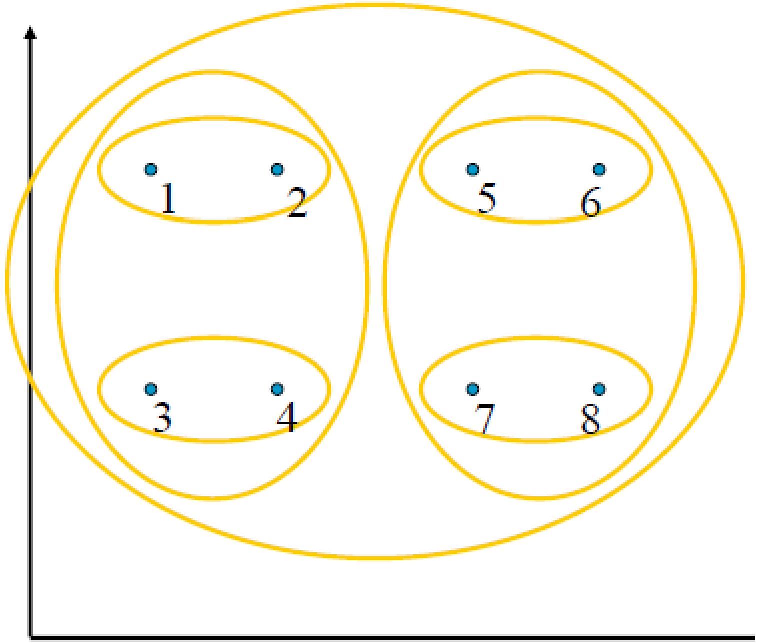


Closeness example

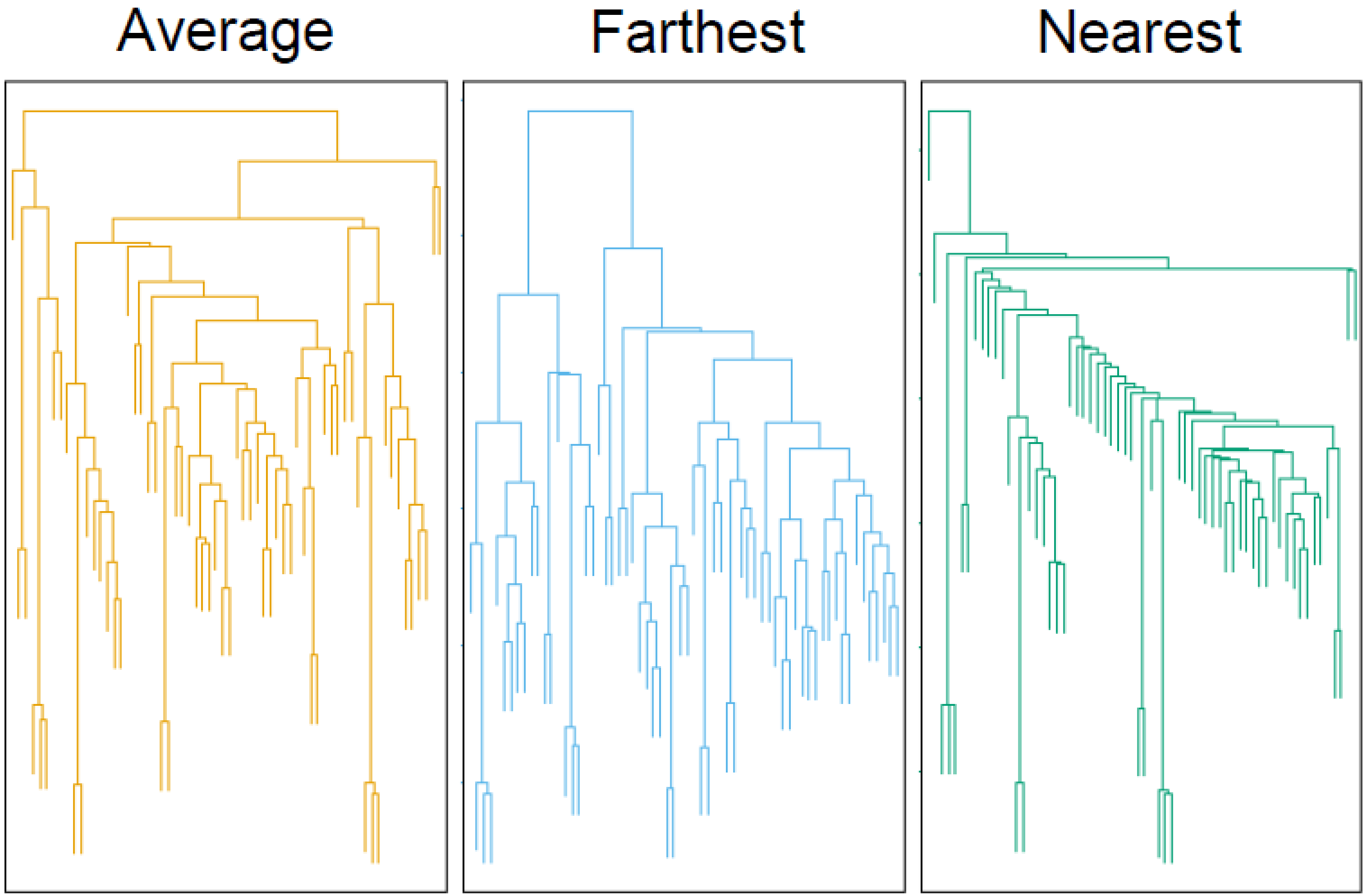
Closest pair
(single-link clustering)



Farthest pair
(complete-link clustering)

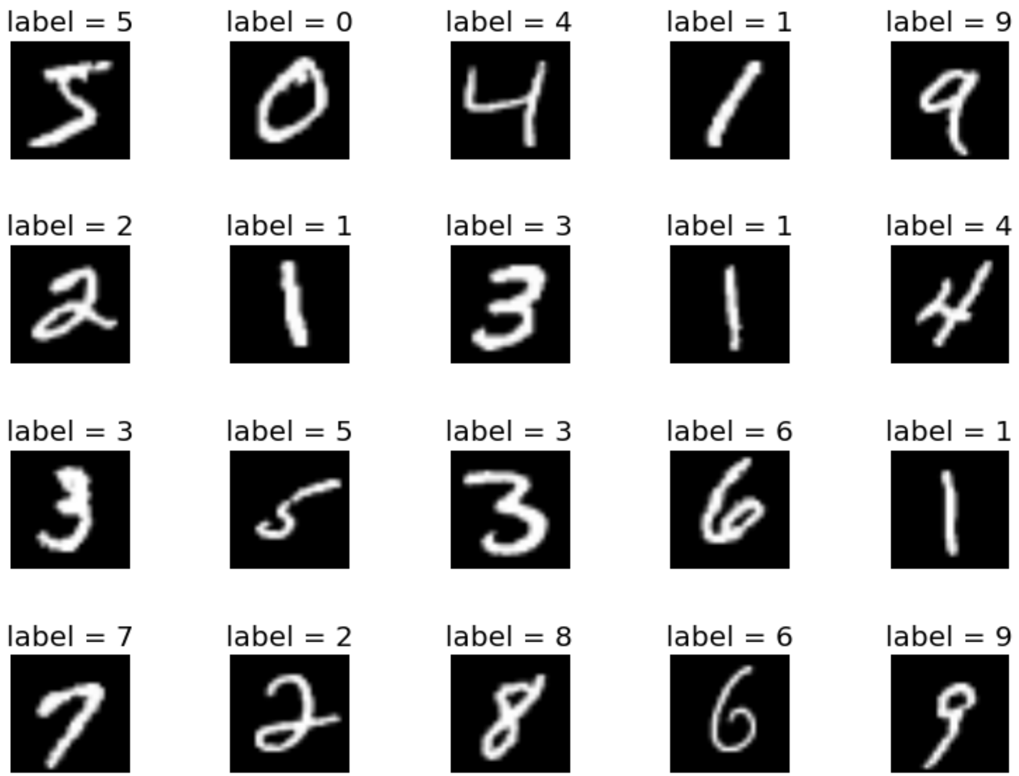


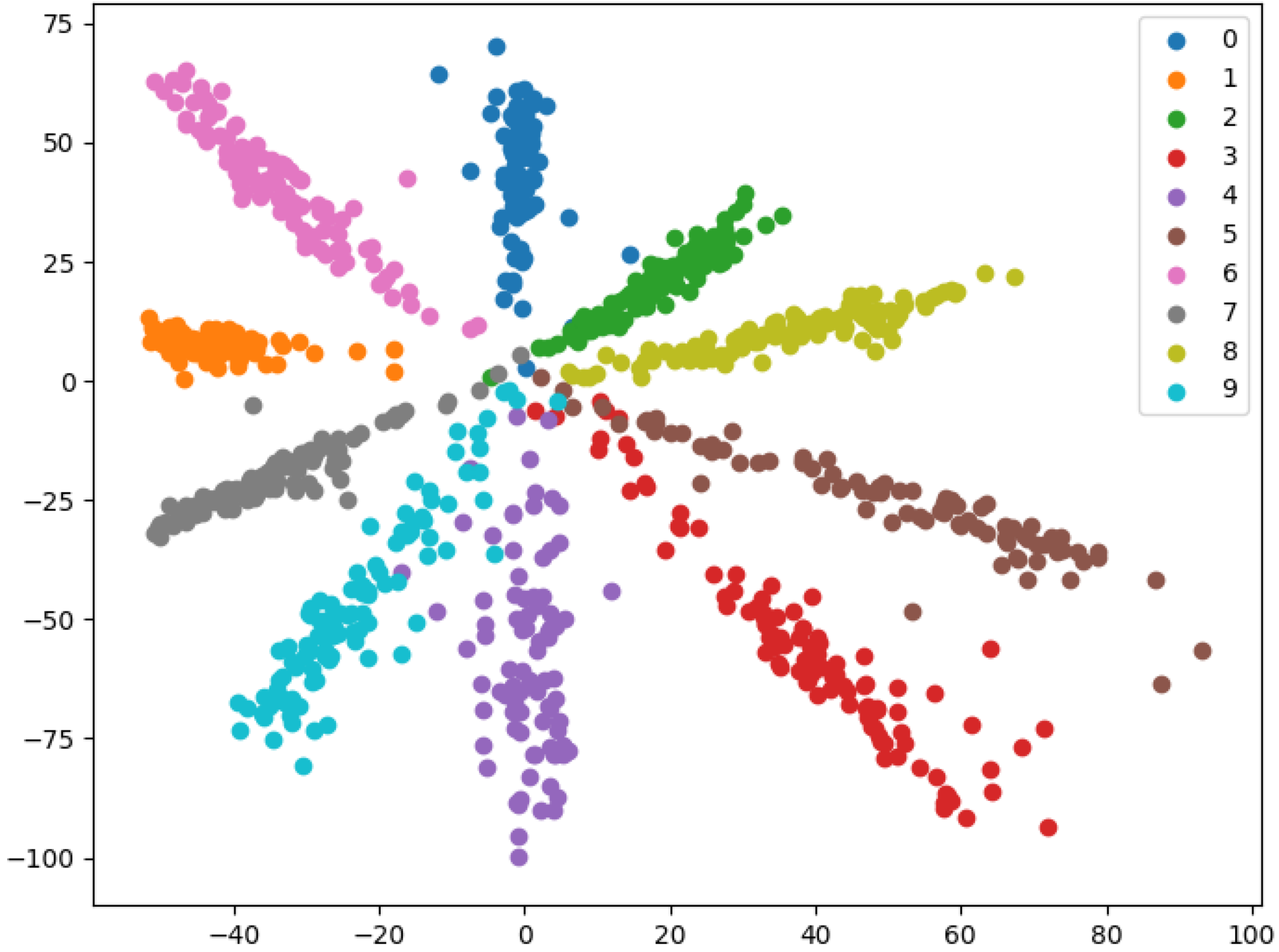
Closeness example 2



Dimensionality Reduction

Example – MNIST dataset





Principal Components Analysis

PCA

Principal components analysis (PCA)

- Principal components analysis (PCA) is a technique that can be used to simplify a dataset
- It is usually a linear transformation that chooses a new coordinate system for the data set such that
  - greatest variance by any projection of the dataset comes to lie on the first axis (then called the first principal component)
  - the second greatest variance on the second axis, and so on
- PCA can be used for reducing dimensionality by eliminating the later principal components

Example

- Consider the following 3D points

1	2	4	3	5	6
2	4	8	6	10	12
3	6	12	9	15	18

- If each component is stored in a byte, we need $18 = 3 \times 6$ bytes

Example (cont.)

- Looking closer, we can see that all the points are related geometrically
  - they are all in the same direction, scaled by a factor:

$$\begin{bmatrix}1\\2\\3\end{bmatrix} = 1 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}4\\8\\12\end{bmatrix} = 4 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}5\\10\\15\end{bmatrix} = 5 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

$$\begin{bmatrix}2\\4\\6\end{bmatrix} = 2 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}3\\6\\9\end{bmatrix} = 3 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}6\\12\\18\end{bmatrix} = 6 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

Example (cont.)

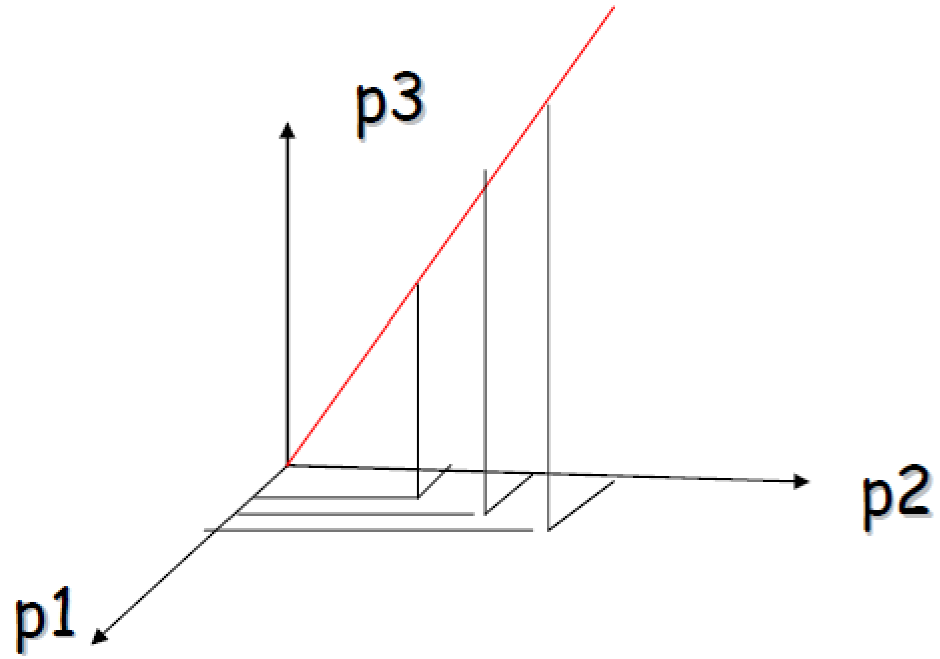
$$\begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} = 1 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} \qquad \begin{array}{|c|}\hline 4\\\hline 8\\\hline 12\\\hline\end{array} = 4 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} \qquad \begin{array}{|c|}\hline 5\\\hline 10\\\hline 15\\\hline\end{array} = 5 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array}$$

$$\begin{array}{|c|}\hline 2\\\hline 4\\\hline 6\\\hline\end{array} = 2 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} \qquad \begin{array}{|c|}\hline 3\\\hline 6\\\hline 9\\\hline\end{array} = 3 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} \qquad \begin{array}{|c|}\hline 6\\\hline 12\\\hline 18\\\hline\end{array} = 6 \times \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array}$$

- They can be stored using only 9 bytes (50% savings!):
    - Store one direction (3 bytes) + the multiplying constants (6 bytes)

Geometrical interpretation

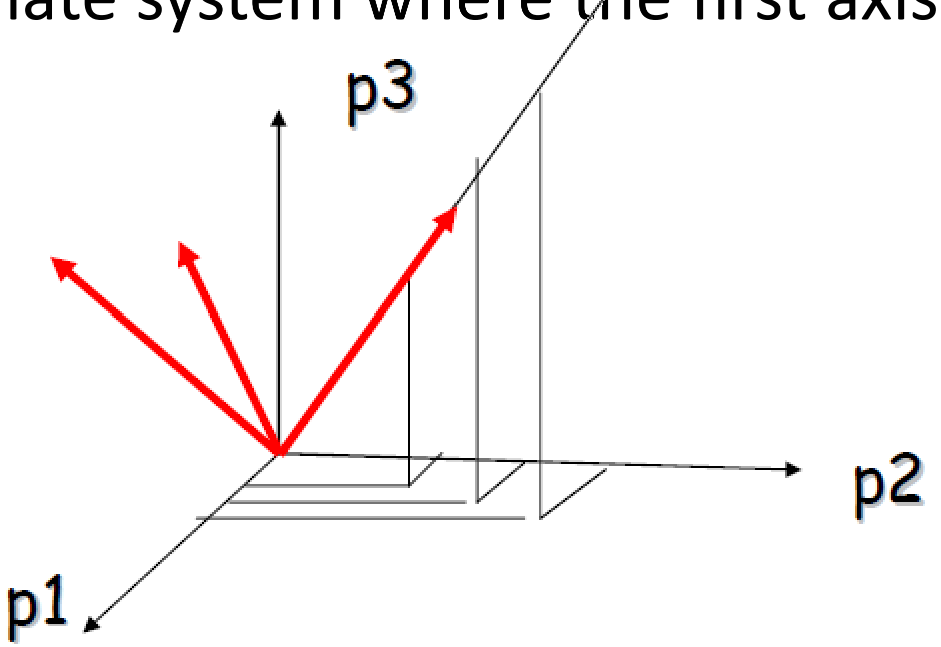
- View points in 3D space



- In this example, all the points happen to lie on one line
  - a 1D subspace of the original 3D space

Geometrical interpretation

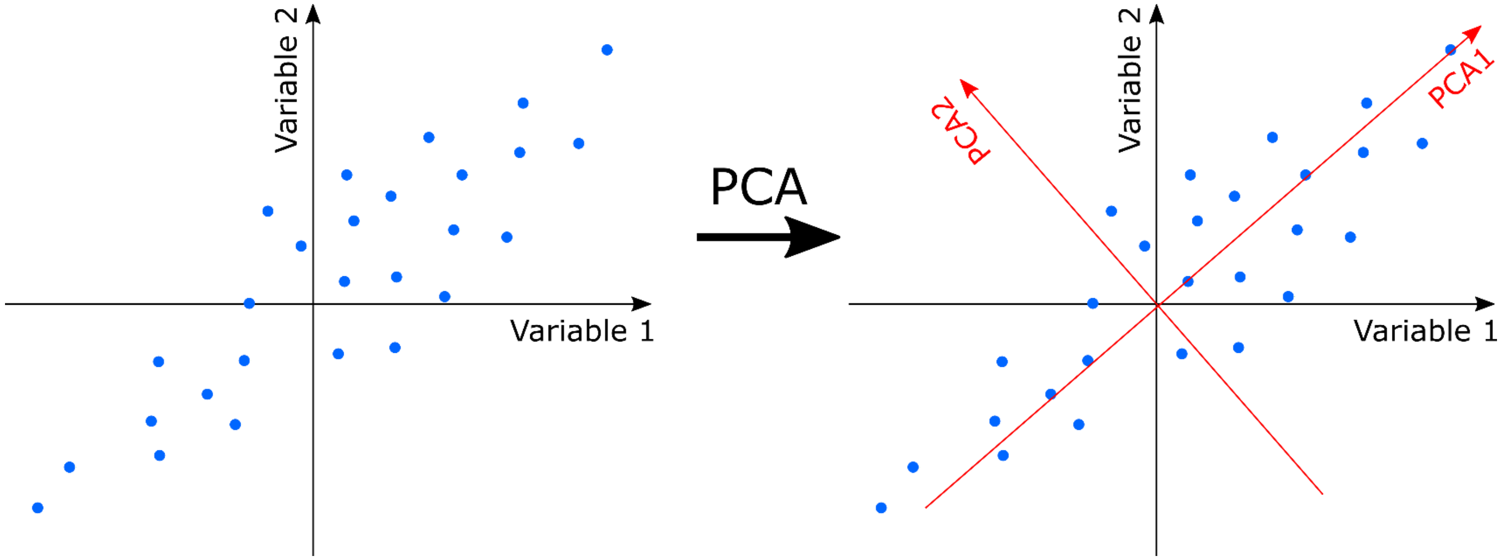
- Consider a new coordinate system where the first axis is along the direction of the line



- In the new coordinate system, every point has only one non-zero coordinate
  - we only need to store the direction of the line (a 3 bytes point) and the nonzero coordinates for each point (6 bytes)

Back to PCA

- Given a set of points, how can we know if they can be compressed similarly to the previous example?
  - We can look into the correlation between the points by the tool of PCA



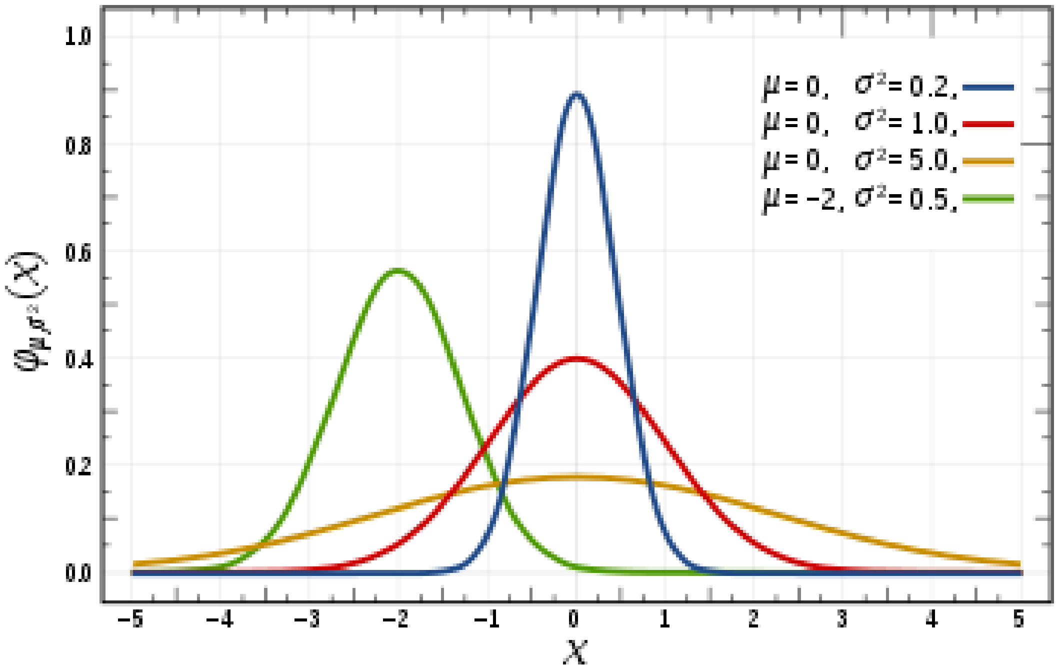
From example to theory

- In previous example, PCA rebuilds the coordination system for the data by selecting
  - the direction with largest variance as the first new base direction
  - the direction with the second largest variance as the second new base direction
  - and so on

- Then how can we find the direction with largest variance?
  - By the eigenvector for the covariance matrix of the data

Review – Variance

- Variance is the expectation of the squared deviation of a random variable from its mean
  - Informally, it measures how far a set of (random) numbers are spread out from their average value



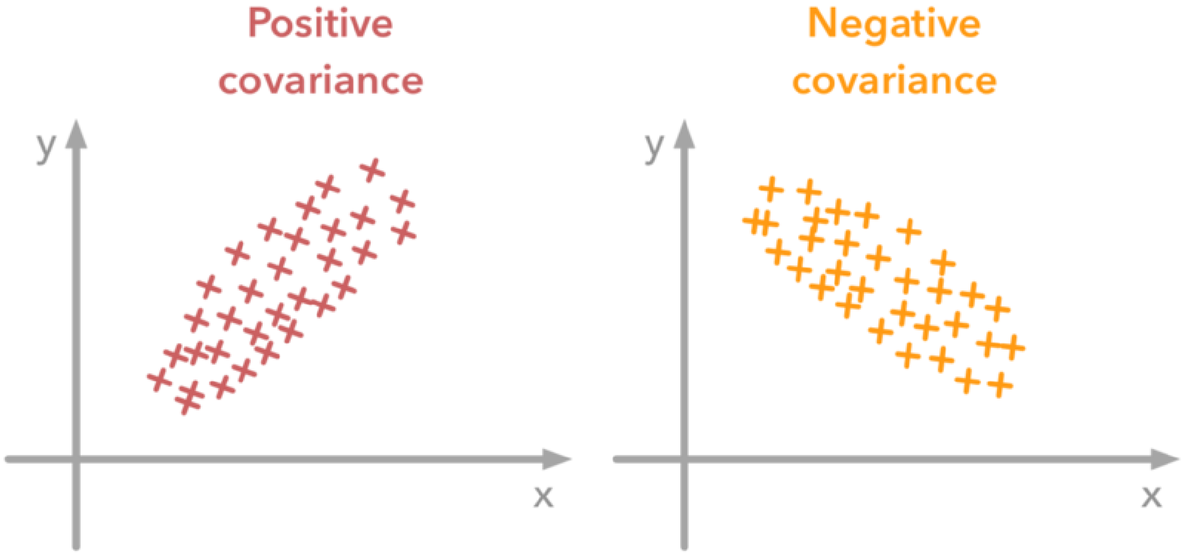
Review – Covariance

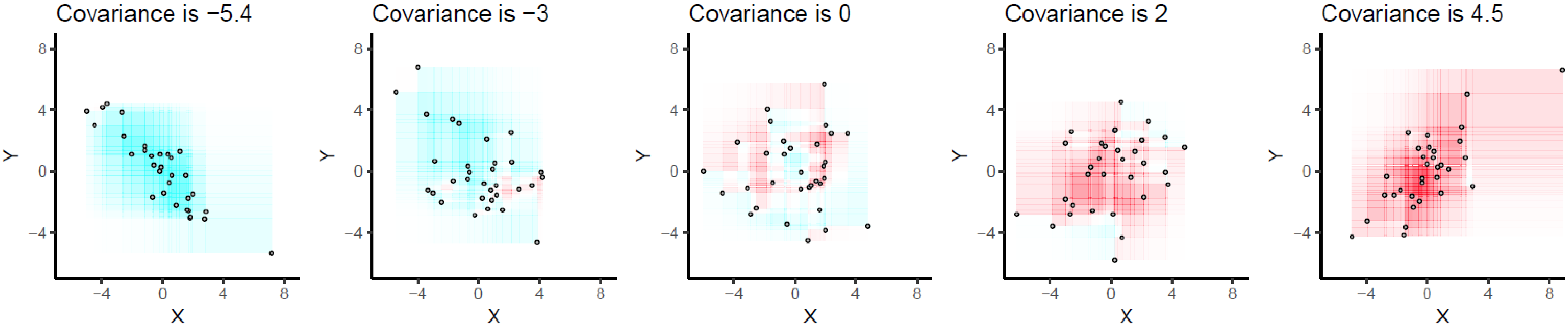
- Covariance is a measure of the joint variability of two random variables
  - If the greater values of one variable mainly correspond with the greater values of the other variable, and the same holds for the lesser values, (i.e., the variables tend to show similar behavior), the covariance is positive
    - E.g. as the number of hours studied increases, the marks in that subject increase
  - In the opposite case, when the greater values of one variable mainly correspond to the lesser values of the other, (i.e., the variables tend to show opposite behavior), the covariance is negative
  - The sign of the covariance therefore shows the tendency in the linear relationship between the variables
  - The magnitude of the covariance is not easy to interpret because it is not normalized and hence depends on the magnitudes of the variables. The normalized version of the covariance, the correlation coefficient, however, shows by its magnitude the strength of the linear relation

Review – Covariance (cont.)

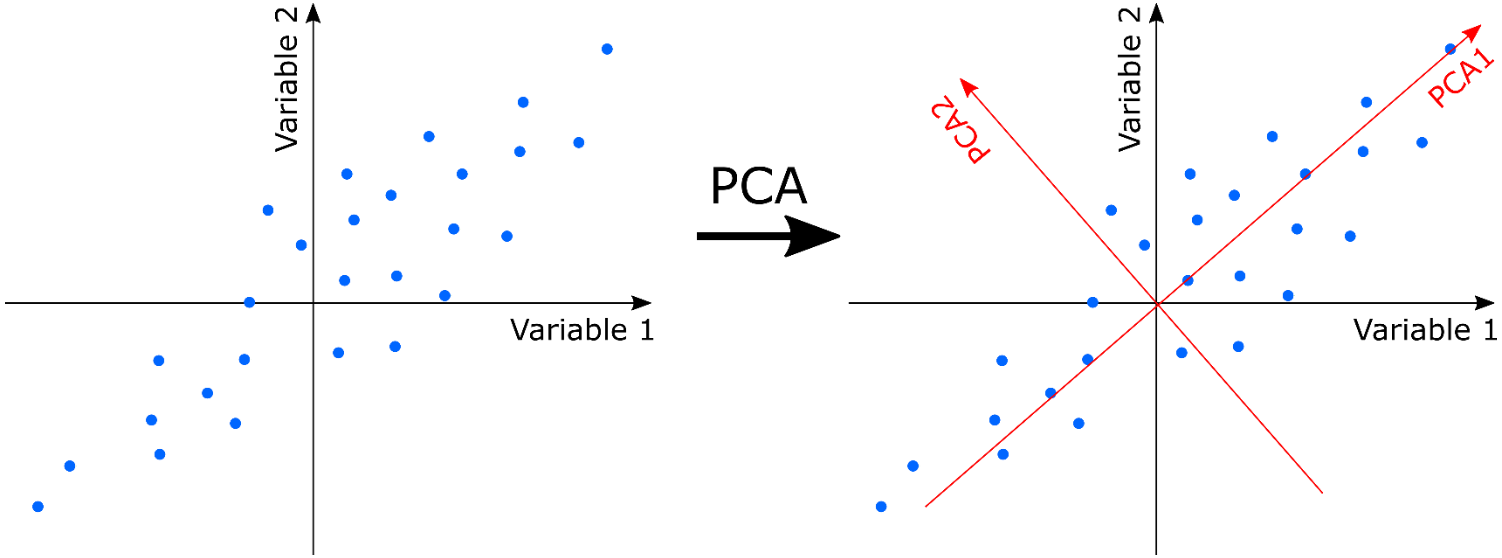
- Sample covariance

$$\text{covariance}(X, Y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N - 1}$$





PCA



- PCA tries to identify the subspace in which the data approximately lies in
- PCA uses an orthogonal transformation on the coordinate system to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components
    - The number of principal components is less than or equal to $\min\{d, N\}$

Covariance matrix

- Suppose there are 3 dimensions, denoted as X, Y, Z. The covariance matrix is

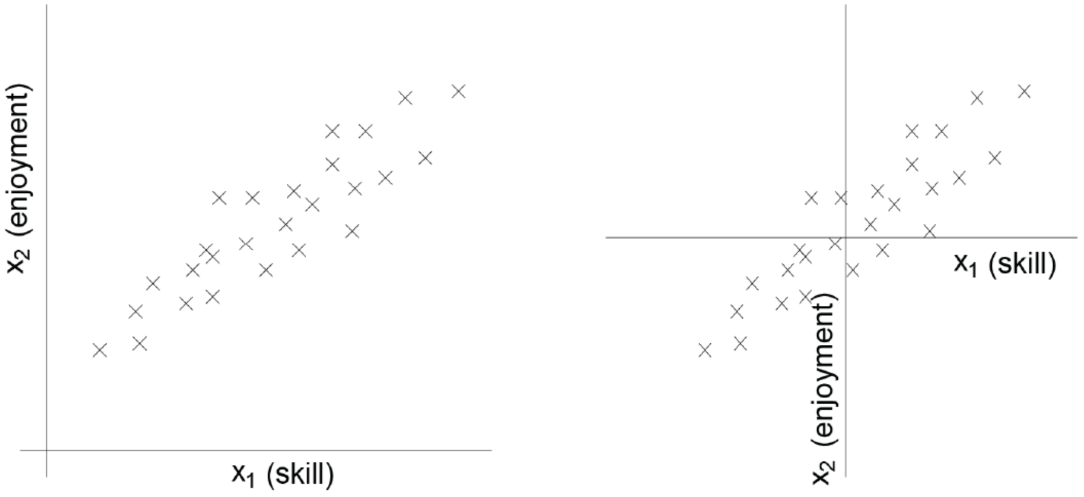
$$COV = \begin{bmatrix} COV(X,X) & COV(X,Y) & COV(X,Z) \\ COV(Y,X) & COV(Y,Y) & COV(Y,Z) \\ COV(Z,X) & COV(Z,Y) & COV(Z,Z) \end{bmatrix}$$

- Note the diagonal is the covariance of each dimension with respect to itself, which is just the variance of each random variable
- Also $COV(X,Y) = COV(Y,X)$
  - hence matrix is symmetric about the diagonal
- d-dimensional data will result in a $d \times d$ covariance matrix

Covariance in the covariance matrix

- Diagonal, or the variance, measures the deviation from the mean for data points in one dimension
- Covariance measures how one dimension random variable varies w.r.t. another, or if there is some linear relationship among them

Data processing



- Given the dataset $D = \left\{x^{(i)}\right\}_{i=1}^{N}$
- Let $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}$

$$X = \begin{bmatrix} \left(x^{(1)} - \bar{x}\right)^{\top} \\ \left(x^{(2)} - \bar{x}\right)^{\top} \\ \vdots \\ \left(x^{(N)} - \bar{x}\right)^{\top} \end{bmatrix} \in \mathbb{R}^{N \times d}$$

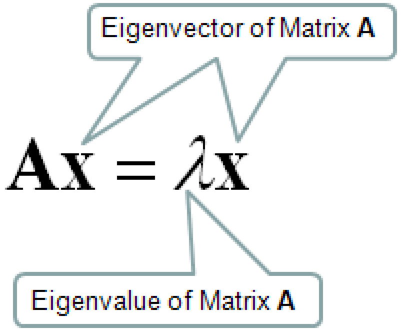
  - Move the center of the data set to 0

Data processing (cont.)

- $Q = X^\top X = [x^{(1)} - \bar{x} \quad x^{(2)} - \bar{x} \quad \dots \quad x^{(N)} - \bar{x}] \begin{bmatrix} \left(x^{(1)} - \bar{x}\right)^\top \\ \left(x^{(2)} - \bar{x}\right)^\top \\ \vdots \\ \left(x^{(N)} - \bar{x}\right)^\top \end{bmatrix}$
    - Q is square with d dimension
    - Q is symmetric
    - Q is the covariance matrix [aka scatter matrix]
    - Q can be very large (in vision, d is often the number of pixels in an image!)
        - For a 256×256 image, $d = 65536$!!
        - Don't want to explicitly compute Q

PCA

- By finding the eigenvalues and eigenvectors of the covariance matrix, we find that the eigenvectors with the largest eigenvalues correspond to the dimensions that have the strongest variation in the dataset
- This is the principal component
- Application:
  - face recognition, image compression
  - finding patterns in data of high dimension



$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

PCA theorem

- Theorem:
- Each $x^{(i)}$ can be written as: $x^{(i)} = \bar{x} + \sum_{j=1}^{d} g_{ij} e_j$
  where e_j are the d eigenvectors of Q with non-zero eigenvalues

- **Notes:**
  1. The eigenvectors $e_1 e_2 \cdots e_d$ span an **eigenspace**
  2. $e_1 e_2 \cdots e_d$ are $d \times 1$ orthonormal vectors (directions in d-Dimensional space)
  3. The scalars g_{ij} are the coordinates of $x^{(i)}$ in the space

$$g_{ij} = \langle x^{(i)} - \bar{x}, e_j \rangle$$

Using PCA to compress data

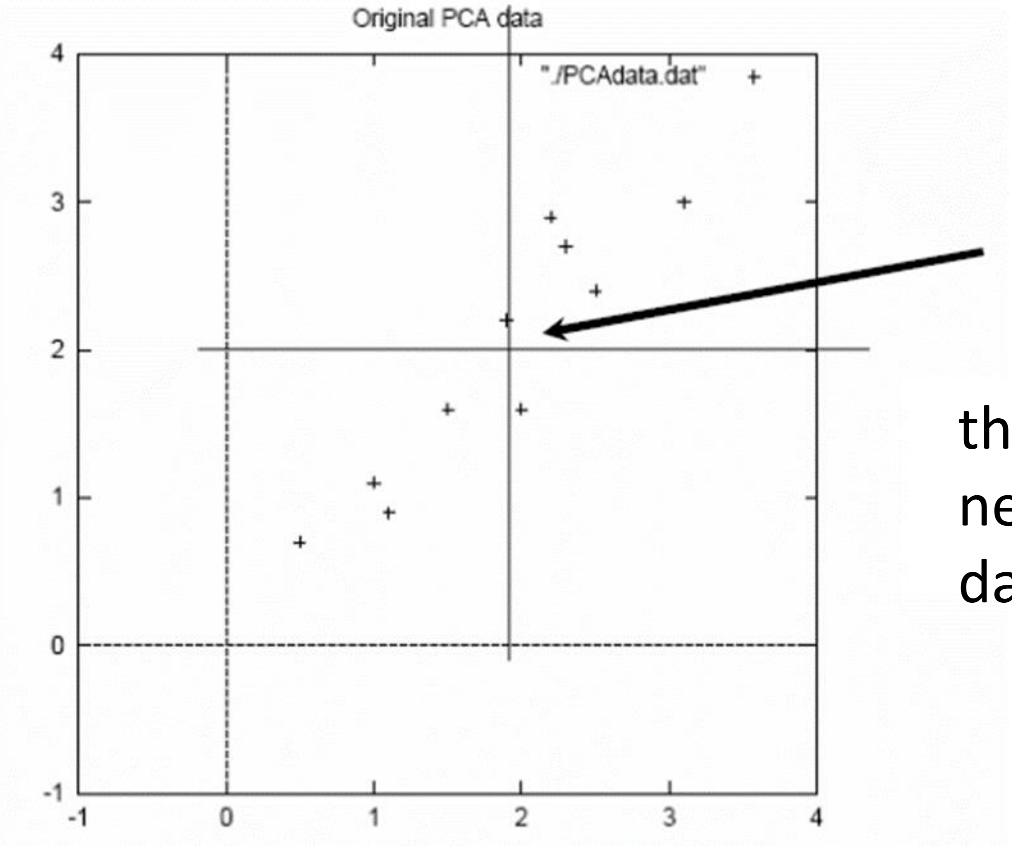
- Expressing x in terms of $e_1 e_2 \cdots e_d$ doesn’t change the size of the data
- However, if the points are highly correlated, many of the new coordinates of x will become zero or close to zero
- Sort the eigenvectors e_i according to their eigenvalue
$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$$
- Assume $\lambda_j \approx 0$ if $j > k$. Then

$$x^{(i)} \approx \bar{x} + \sum_{j=1}^{k} g_{ij} e_j$$

Example – STEP 1

DATA:

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9



mean

this becomes the new origin of the data from now on

http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf

Example – STEP 2

- Calculate the covariance matrix

$$\mathrm{Cov} = \begin{bmatrix} 0.616555556 & 0.615444444 \\ 0.615444444 & 0.716555556 \end{bmatrix}$$

- since $\mathrm{cov}(X, Y)$ is positive, it is expect that x and y increase together

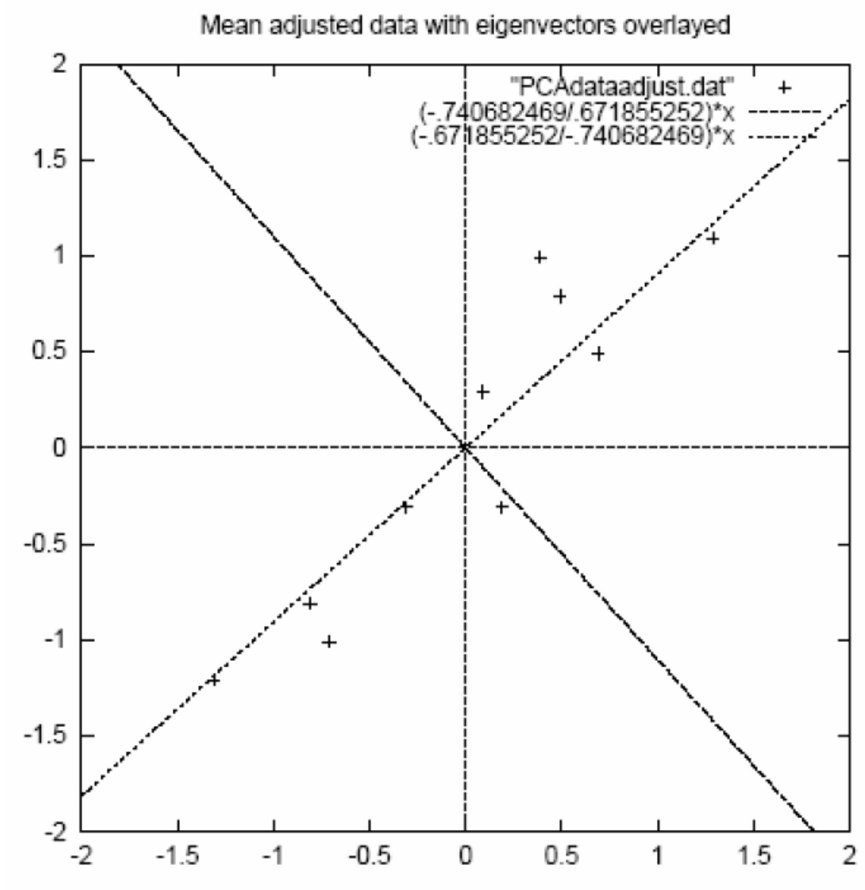
Example – STEP 3

- Calculate the eigenvectors and eigenvalues of the covariance matrix

- $\text{eigenvalues} = \begin{bmatrix} 0.0490833989 \\ 1.28402771 \end{bmatrix}$

- $\text{eigenvectors} = \begin{bmatrix} -0.735178656 & -0.677873399 \\ 0.677873399 & -0.735178656 \end{bmatrix}$

Example – STEP 3 (cont.)



- Eigenvectors are plotted as diagonal dotted lines on the plot
- Note they are perpendicular to each other
- Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit
- The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount

Example – STEP 4

- Feature vector $= [e_1 \quad e_2 \quad \cdots \quad e_d]$

- We can either form a feature vector with both of the eigenvectors:

$$\begin{bmatrix} -0.735178656 & -0.677873399 \\ 0.677873399 & -0.735178656 \end{bmatrix}$$

- or, we can choose to delete the smaller, less significant component:

$$\begin{bmatrix} -0.677873399 \\ -0.735178656 \end{bmatrix}$$

Example – STEP 5

$$\text{FinalData}_{N\times d} = \begin{bmatrix} g\left(x^{(1)}\right)^{\mathsf{T}} \\ \vdots \\ g\left(x^{(N)}\right)^{\mathsf{T}} \end{bmatrix}_{N\times d} \begin{bmatrix} e_1^{\mathsf{T}} \\ \vdots \\ e_d^{\mathsf{T}} \end{bmatrix}_{d\times d}$$

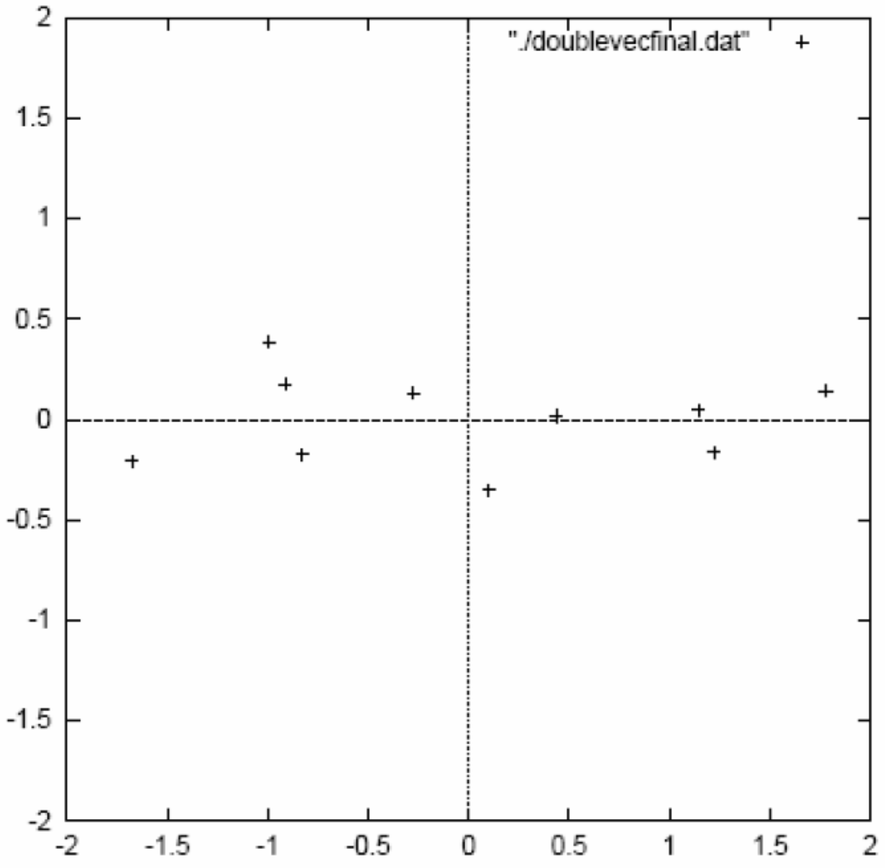
- Deriving new data coordinates

  FinalData = RowZeroMeanData x RowFeatureVector

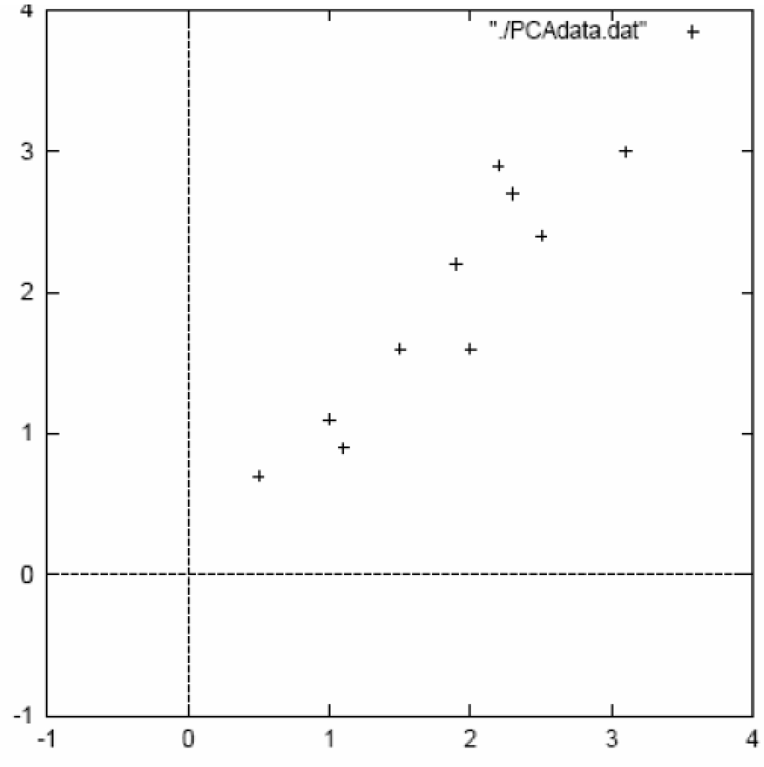
- RowZeroMeanData is the mean-adjusted data, i.e. the data items are in each row, with each column representing a separate dimension
- RowFeatureVector is the matrix with the eigenvectors in the columns transposed so that the eigenvectors are now in the rows, with the most significant eigenvector at the top
- Note: We rotate the coordinate axes so high-variance axis comes first

Example – STEP 5 (cont.)

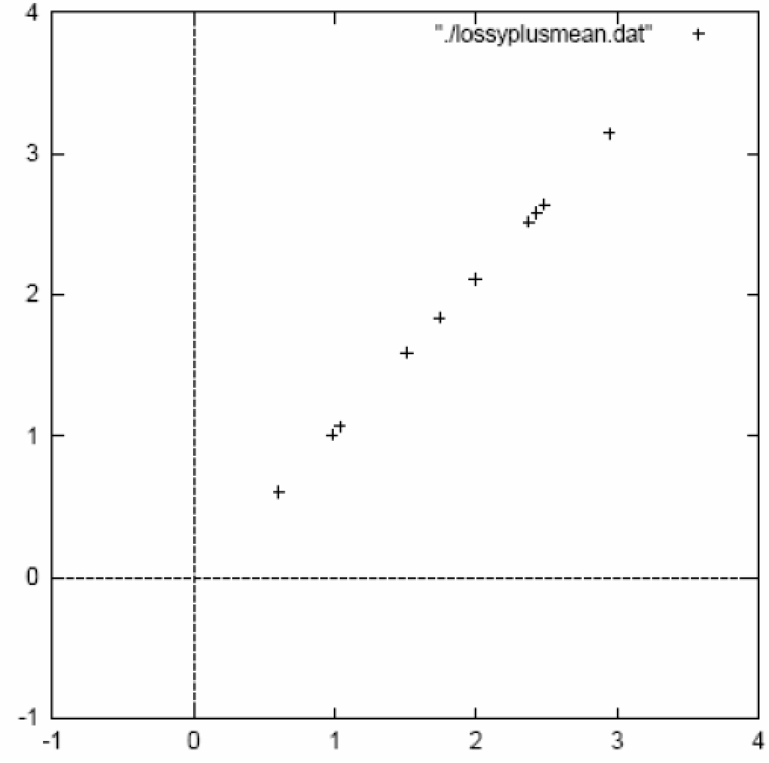
- The plot of the PCA results using both the two eigenvector



Example – Final approximation



2D point cloud



Approximation using one eigenvector basis

Example – Final approximation

$$\text{FinalData}_{N\times d} = \begin{bmatrix} g\left(x^{(1)}\right)^{\top} \\ \vdots \\ g\left(x^{(N)}\right)^{\top} \end{bmatrix}_{N\times d} \begin{bmatrix} e_1^{\top} \\ \vdots \\ e_d^{\top} \end{bmatrix}_{d\times d}$$

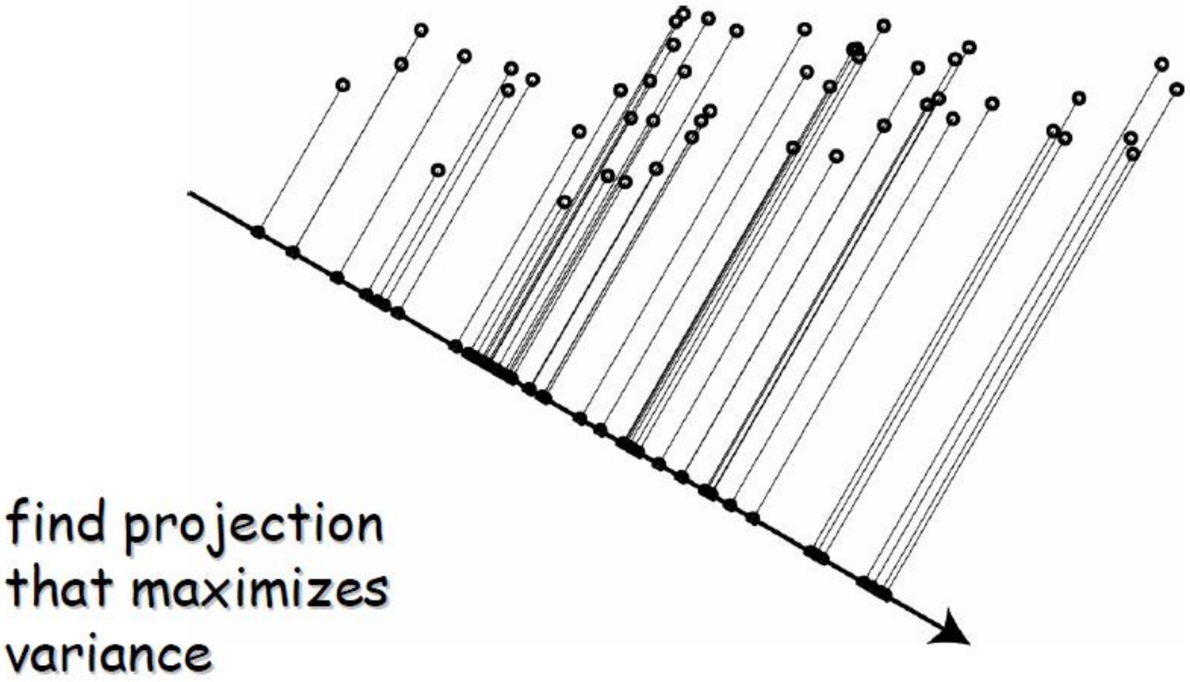
$$\approx \begin{bmatrix} g\left(x^{(1)}\right)_1 & \cdots & g\left(x^{(1)}\right)_k & \cancel{\cdots} & \cancel{g\left(x^{(1)}\right)_d} \\ & & \vdots & & \\ g\left(x^{(N)}\right)_1 & \cdots & g\left(x^{(N)}\right)_k & \cancel{\cdots} & \cancel{g\left(x^{(N)}\right)_d} \end{bmatrix}_{N\times d} \begin{bmatrix} e_1^{\top} \\ \vdots \\ e_k^{\top} \\ \vdots \\ e_d^{\top} \end{bmatrix}_{d\times d}$$

Zero!!

Revisit the eigenvectors in PCA

- It is critical to notice that the *direction of maximum variance* in the input space happens to be same as the *principal eigenvector of the covariance matrix*

- Why?



Revisit the eigenvectors in PCA (cont.)

- The projection of each point x to a direction u (with $\|u\| = 1$) is

$$x^\top u$$

- The variance of the projection is

$$\sum_{i=1}^{N} \left(\left(x^{(i)} - \bar{x} \right)^\top u \right)^2 = u^\top Q u$$

  which is maximized when u is the eigenvector with the largest eigenvalue

- $Q = \sum_{j=1}^{d} \lambda_j e_j e_j{}^\top = E \Lambda E^\top$ with $\Lambda = \begin{bmatrix} \lambda_1 & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \lambda_d \end{bmatrix}$

Review – Total/Explained variance

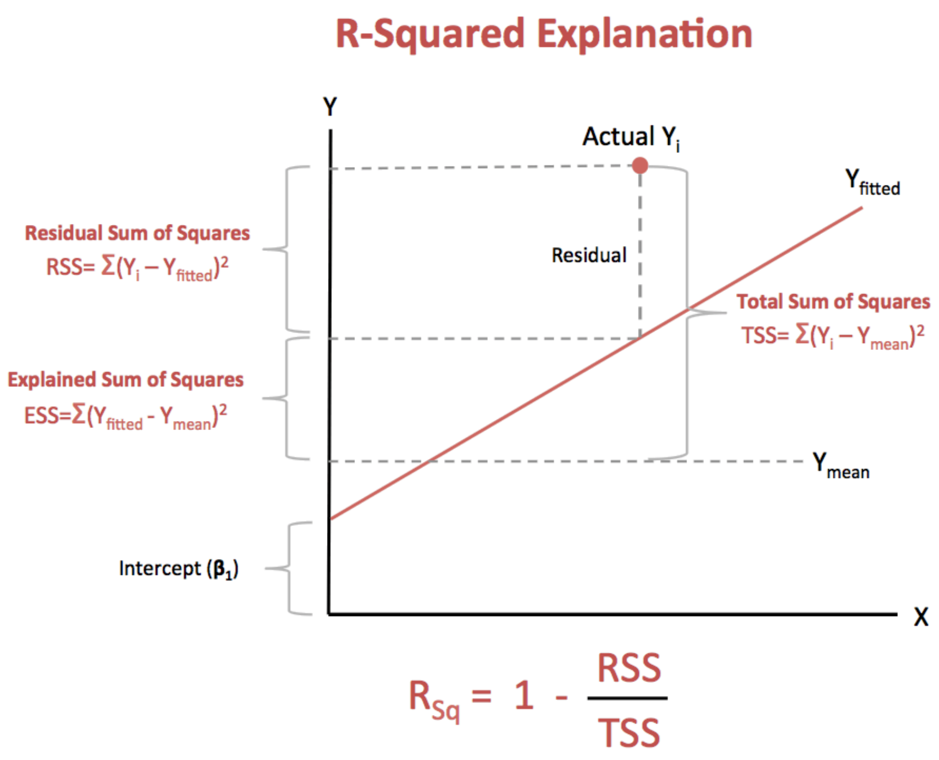
- $R^2 = \frac{explained\ variance}{total\ variance}$

- Total variance: $SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2$

- Explained variance: $SS_{\text{reg}} = \sum_i (f_i - \bar{y})^2$

- Or, it can be computed as:

$R^2 \equiv 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$ where $SS_{\text{res}} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2$



Total variance and PCA

- Note that $I = e_1 e_1^\top + \cdots + e_d e_d^\top$
- Total variance is
- $\sum_{i=1}^{N} \left(x^{(i)} - \bar{x}\right)^\top \left(x^{(i)} - \bar{x}\right)$
- $= \sum_{i=1}^{N} \left(x^{(i)} - \bar{x}\right)^\top \left(e_1 e_1^\top + \cdots + e_d e_d^\top\right)\left(x^{(i)} - \bar{x}\right)$
- $= \sum_{j=1}^{d} e_j^\top Q \, e_j = \lambda_1 + \cdots + \lambda_d$

Total variance and PCA (cont.)

- Approximation of each $x^{(i)} - \bar{x} \approx \sum_{j=1}^{k} g_{ij} e_j =: \tilde{x}^{(i)} - \bar{x}$
- Then the explained variance is
- $\sum_{i=1}^{N} \left(\tilde{x}^{(i)} - \bar{x}\right)^{\top} \left(\tilde{x}^{(i)} - \bar{x}\right)$
- $= \sum_{i=1}^{N} \left(\tilde{x}^{(i)} - \bar{x}\right)^{\top} \left(e_1 e_1^{\top} + \cdots + e_d e_d^{\top}\right)\left(\tilde{x}^{(i)} - \bar{x}\right)$
- $= \sum_{j=1}^{d} e_j^{\top} \tilde{Q} e_j = \lambda_1 + \cdots + \lambda_k$
- where $\tilde{Q} = \sum_{i=1}^{N} \left(\tilde{x}^{(i)} - \bar{x}\right) \left(\tilde{x}^{(i)} - \bar{x}\right)^{\top} = E \tilde{\Lambda} E^{\top}$ with
- $\tilde{\Lambda} = \begin{bmatrix} \lambda_1 & & & & \\ & \ddots & & & \\ & & \lambda_k & & \\ & & & 0 & \\ & & & & 0 \end{bmatrix}$

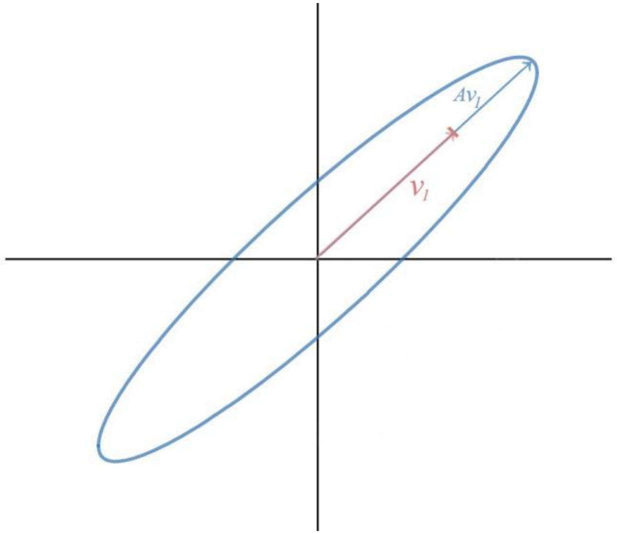
Eigenvalues and Eigenvectors

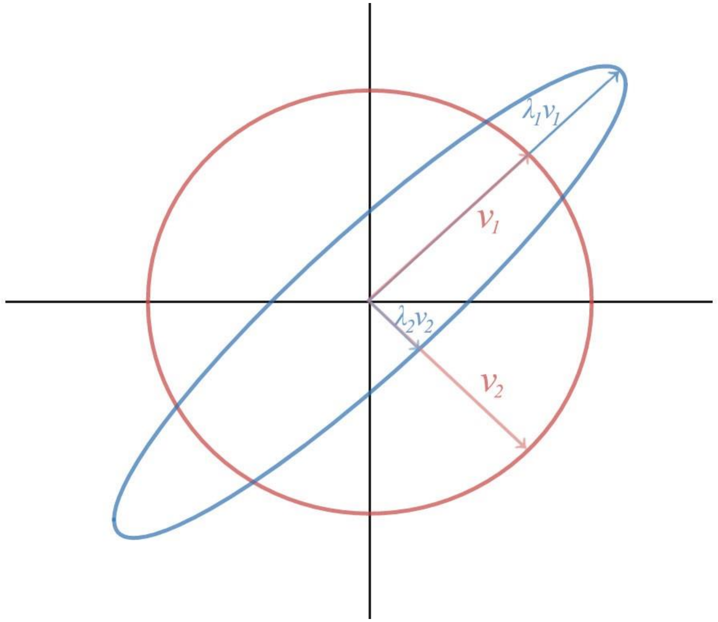
Properties

- Scalar λ and vector v are eigenvalues and eigenvectors of A

$$Av = \lambda v$$

- Visually, Av lies along the same line as the eigenvector of v





Example

$$\underset{A}{\begin{bmatrix}1 & -3 & 3\\ 3 & -5 & 3\\ 6 & -6 & 4\end{bmatrix}}\begin{bmatrix}1/2\\ 1/2\\ 1\end{bmatrix} = \underset{\textit{eigenvalue}}{4}\;\underset{\textit{eigenvector}}{\begin{bmatrix}1/2\\ 1/2\\ 1\end{bmatrix}}$$

$$\begin{bmatrix}1 & -3 & 3\\ 3 & -5 & 3\\ 6 & -6 & 4\end{bmatrix}\begin{bmatrix}1\\ 1\\ 0\end{bmatrix} = -2\begin{bmatrix}1\\ 1\\ 0\end{bmatrix}$$

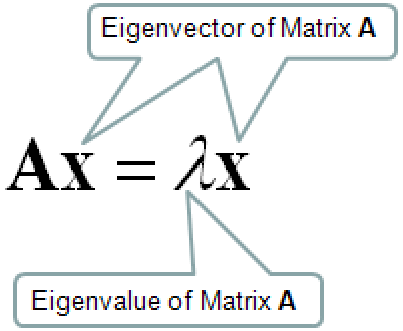
$$\begin{bmatrix}1 & -3 & 3\\ 3 & -5 & 3\\ 6 & -6 & 4\end{bmatrix}\begin{bmatrix}-1\\ 0\\ 1\end{bmatrix} = -2\begin{bmatrix}-1\\ 0\\ 1\end{bmatrix}$$

How to solve eigenvalues and eigenvectors?

- If $(A - \lambda I)x = 0$ has a nonzero solution for λ, then $A - \lambda I$ is not invertible. Then the determinant of $A - \lambda I$ must be zero

- λ is an eigenvalue of A if and only if $A - \lambda I$ is singular:

$$\det(A - \lambda I) = 0$$



Eigenvector of Matrix **A**
$\mathbf{Ax} = \lambda\mathbf{x}$
Eigenvalue of Matrix **A**

Example

- Find the eigenvalues and eigenvectors of $A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$

- Need to solve $\det(A - \lambda I) = 0$
- $A - \lambda I = \begin{bmatrix} 1 - \lambda & 2 \\ 2 & 4 - \lambda \end{bmatrix}$
- $\det(A - \lambda I) = (1 - \lambda)(4 - \lambda) - 2 \times 2 = \lambda^2 - 5\lambda$
- $\det(A - \lambda I) = 0$ would imply $\lambda = 0$ and $\lambda = 5$

Example (cont.)

- Then solve $(A - \lambda I)x = 0$ to get the eigenvectors for each of the eigenvalues
- $Ax = 0$ has a nonzero solution of $\begin{bmatrix} 2 \\ -1 \end{bmatrix}$, or $\begin{bmatrix} 2/\sqrt{5} \\ -1/\sqrt{5} \end{bmatrix}$
- $(A - 5I)x = 0$ has a nonzero solution of $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$, or $\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$
- Note: Eigenvectors for different eigenvalues are orthogonal

Singular Value Decomposition

SVD

SVD

- Singular Value Decomposition (SVD) is a factorization method of matrix. It states that any $m \times n$ matrix A can be written as the product of 3 matrices:

$$A = U\,S\,V^T$$

- Where:
  - U is $m \times m$ and its columns are orthonormal eigenvectors of $AA^\top$
  - V is $n \times n$ and its columns are orthonormal eigenvectors of $A^\top A$
  - S is $m \times n$ is a diagonal matrix with r elements equal to the root of the positive eigenvalues of $AA^\top$ or $A^\top A$ (both matrices have the same positive eigenvalues anyway)

In full matrix form

$$A_{m\times n} = U_{m\times m} S_{m\times n} V^{T}_{n\times n}$$

$$\underset{m\times n}{\overset{\mathrm{A}}{\begin{pmatrix} x_{11} & x_{12} & & x_{1n} \\ & & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{m\times m}{\overset{\mathrm{U}}{\begin{pmatrix} u_{11} & & u_{m1} \\ & \ddots & \\ u_{1m} & & u_{mm} \end{pmatrix}}} \underset{m\times n}{\overset{\mathrm{S}}{\begin{pmatrix} \sigma_1 & & & 0 \\ & \ddots & & \\ & & \sigma_r & \\ & & & \ddots \\ 0 & & & 0 \end{pmatrix}}} \underset{n\times n}{\overset{\mathrm{V^T}}{\begin{pmatrix} v_{11} & & v_{1n} \\ & \ddots & \\ v_{n1} & & v_{nn} \end{pmatrix}}}$$

Example

- Let’s assume：

$$A = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$$

- We can have:

$$AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix} \qquad A^T A \quad = \begin{pmatrix} 13 & 12 & 2 \\ 12 & 13 & -2 \\ 2 & -2 & 8 \end{pmatrix}$$

Example (cont.)

- Compute U and V respectively:

$$AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix}$$

eigenvalues: $\lambda_1 = 25$, $\lambda_2 = 9$

eigenvectors

$$u_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \quad u_2 = \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$$

$$A^TA = \begin{pmatrix} 13 & 12 & 2 \\ 12 & 13 & -2 \\ 2 & -2 & 8 \end{pmatrix}$$

eigenvalues: $\lambda_1 = 25$, $\lambda_2 = 9$, $\lambda_3 = 0$

eigenvectors

$$v_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{pmatrix} \quad v_2 = \begin{pmatrix} 1/\sqrt{18} \\ -1/\sqrt{18} \\ 4/\sqrt{18} \end{pmatrix} \quad v_3 = \begin{pmatrix} 2/3 \\ -2/3 \\ -1/3 \end{pmatrix}$$

Example (cont.)

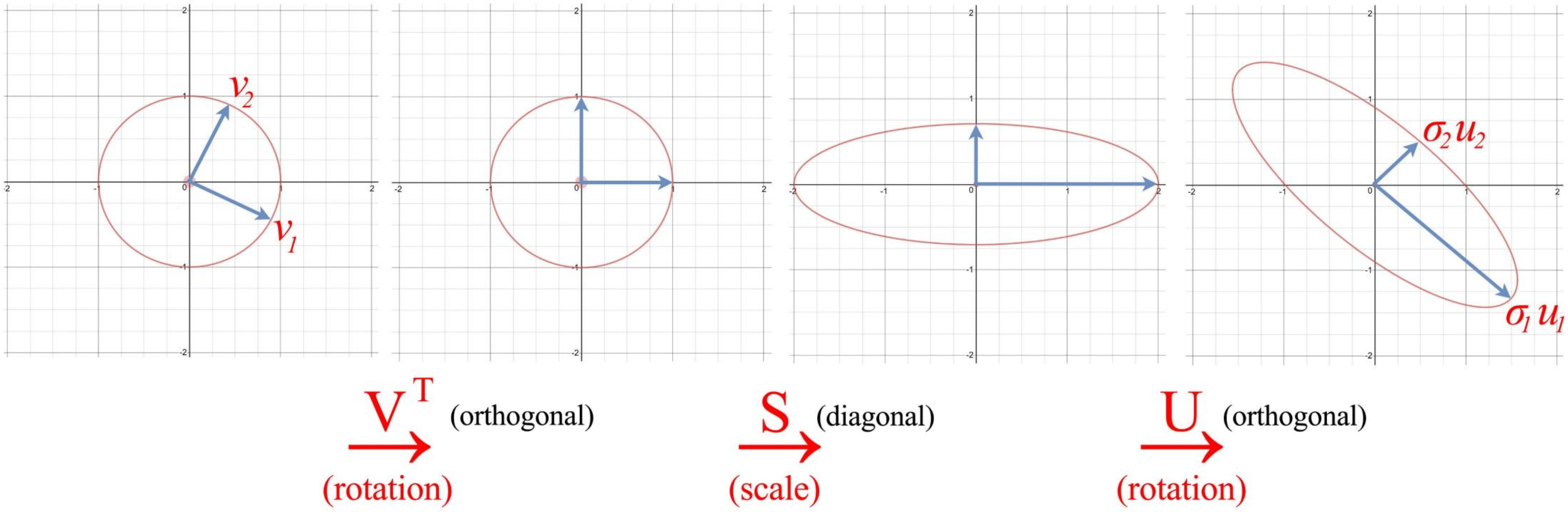
- Finally, we have:

$$A = USV^T = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}$$

Visualization

- The eigenvector v_j of V is transformed into $Av_j = \sigma_j u_j$

$$Ax = USV^T x$$



$\xrightarrow{V^T}$ (orthogonal) (rotation)

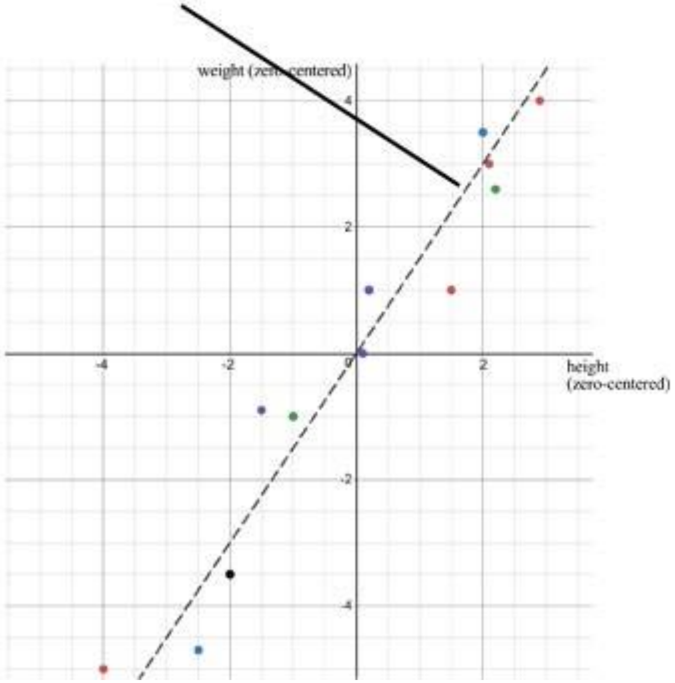
$\xrightarrow{S}$ (diagonal) (scale)

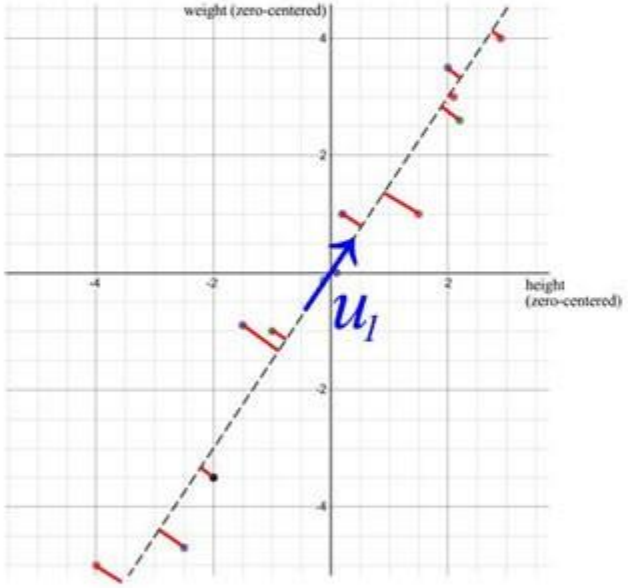
$\xrightarrow{U}$ (orthogonal) (rotation)

Insight

$$\underbrace{A}_{m \times n} = U\,S\,V^T = \overbrace{\sigma_1 \underbrace{u_1}_{m \times 1} \underbrace{v_1^T}_{1 \times n}}^{\text{more significant}} + \dots + \overbrace{\underbrace{\sigma_r u_r v_r^T}_{m \times n}}^{\text{less significant}}$$

$$S = \sigma_1 \underline{u_1} v_1^T + \cancel{\sigma_2 u_2 v_2^T}$$





SVD and PCA

$$(S^\top S)_{d\times d} = \begin{bmatrix} \sigma_1^2 & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \sigma_d^2 \end{bmatrix}$$

- $X = US_{N\times d}V^\top$
- $Q = X^\top X = V(S^\top)_{d\times N}U^\top \cdot US_{N\times d}V^\top = V(S^\top S)_{d\times d}V^\top$

- This explains why singular value σ_j is the square root of the eigenvalue λ_j of Q
- $\lambda_j = \sigma_j^2$
- Also V contains eigenvectors of $X^\top X$
- Similarly, $XX^\top = US_{N\times d}V^\top \cdot V(S^\top)_{d\times N}U^\top = U(SS^\top)_{N\times N}U^\top$

$$(SS^\top)_{N\times N} = \begin{bmatrix} \sigma_1^2 & \cdots & & & \\ \vdots & \ddots & & \vdots & \\ & & \sigma_d^2 & & \\ & \cdots & & \ddots & \\ & & & & 0 \end{bmatrix}$$

Application

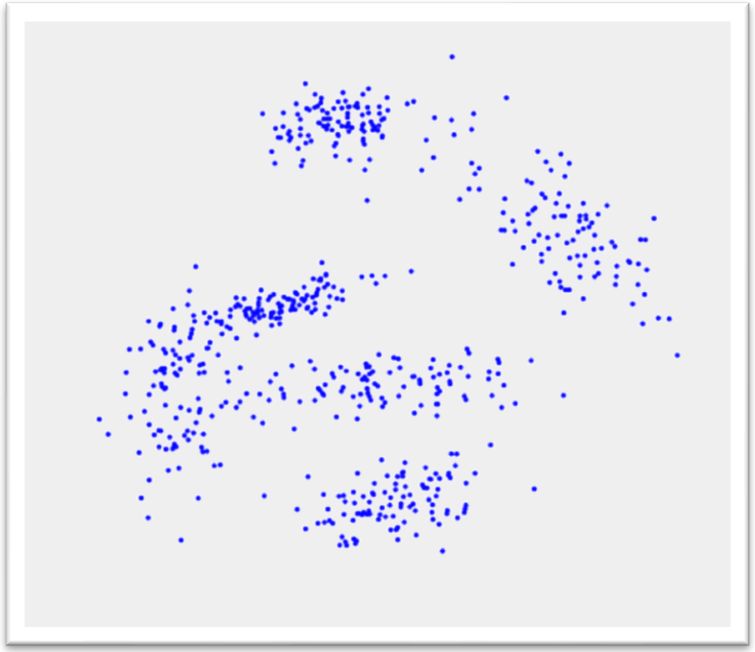
- Matrix factorization in recommendation system

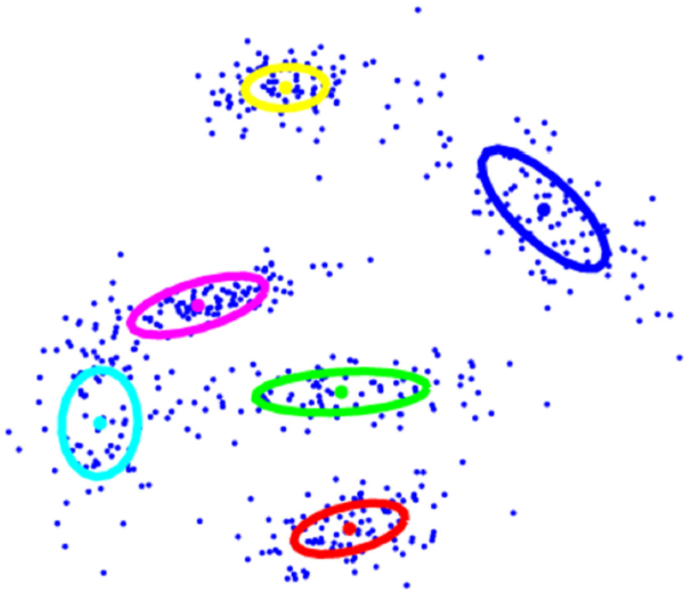
- Suppose in the database of Taobao, there are m users and n items, and an $m \times n$ binary matrix A
  - Each entry indicates whether a user has bought an item or not
  - As each user only buys very few items among all items, the matrix is very sparse
- As the manager of Taobao, how can you predict the likelihood that a user will buy a given item?
- SVD could help

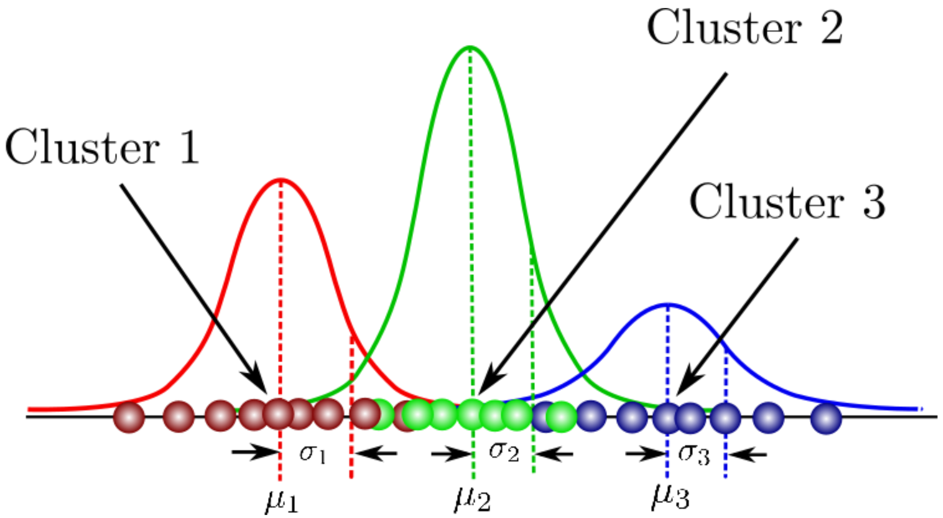
Gaussian Mixture Models

Gaussian Mixture Models

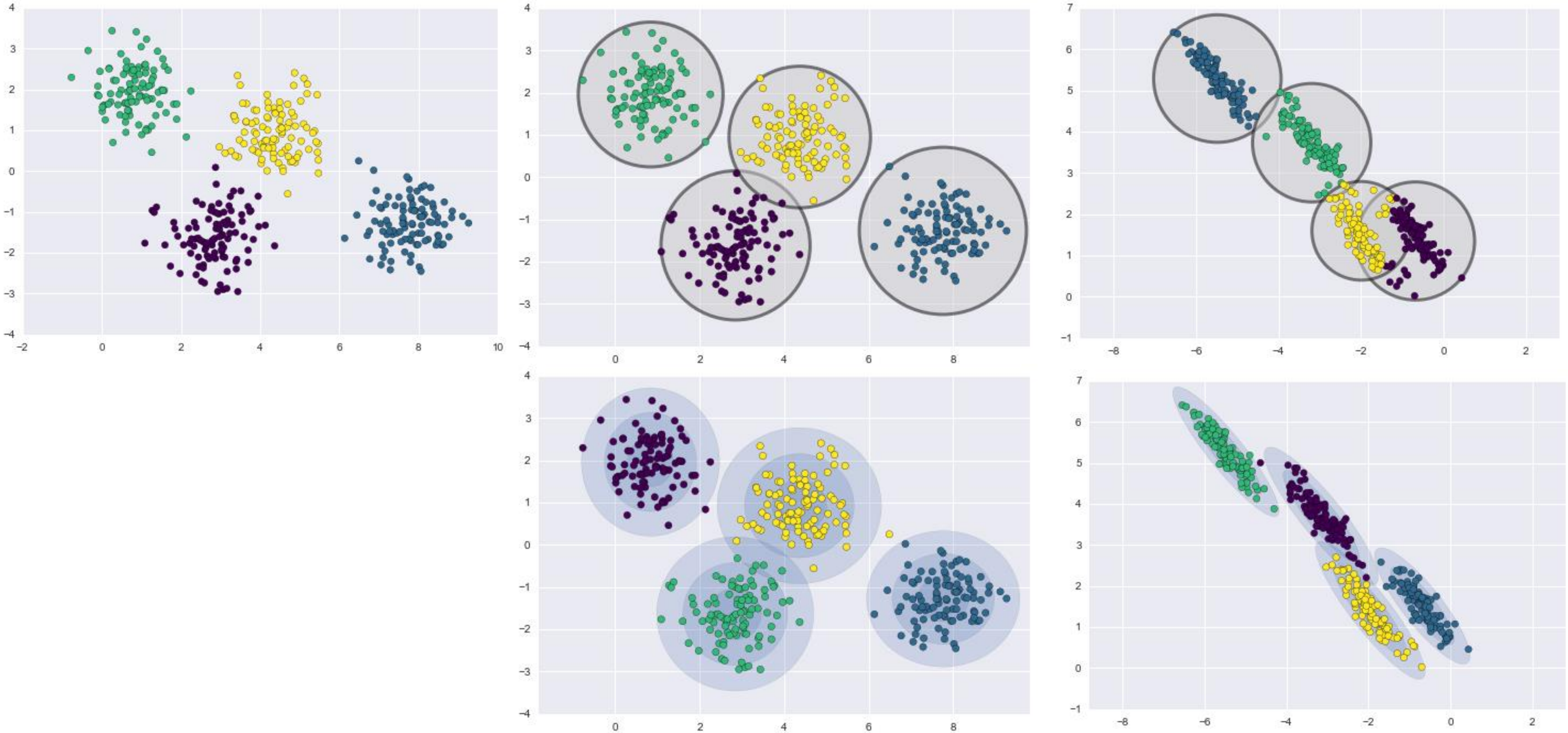
- Is a clustering algorithms
- Difference with K-means
  - K-means outputs the label of a sample
  - GMM outputs the probability that a sample belongs to a certain class
  - GMM can also be used to generate new samples!







K-means vs GMM



High-dimensional Gaussian distribution

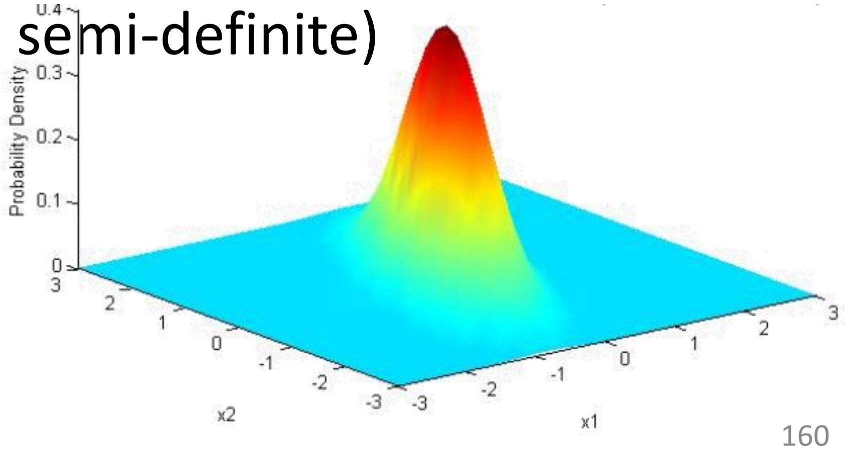
$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- The probability density of Gaussian distribution on $x = (x_1, \dots, x_d)^\top$ is

$$\mathcal{N}(x|\mu,\Sigma) = \frac{\exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)}{\sqrt{(2\pi)^d|\Sigma|}}$$

  - where μ is the mean vector
  - Σ is the symmetric covariance matrix (positive semi-definite)
- E.g. the Gaussian distribution with

$$\mu = (0,0)^T \qquad \Sigma = \begin{pmatrix} 0.25 & 0.30 \\ 0.30 & 1.00 \end{pmatrix}$$



Mixture of Gaussian

- The probability given in a mixture of *K* Gaussians is:

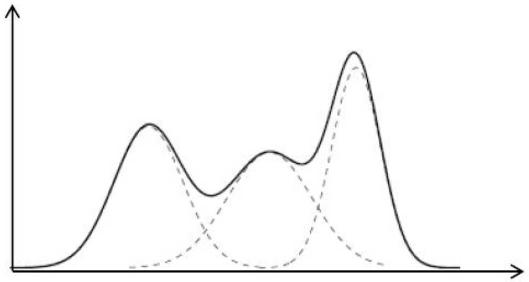
$$p(x) = \sum_{j=1}^{K} w_j \cdot \mathcal{N}(x|\mu_j, \Sigma_j)$$

where w_j is the prior probability of the j-th Gaussian

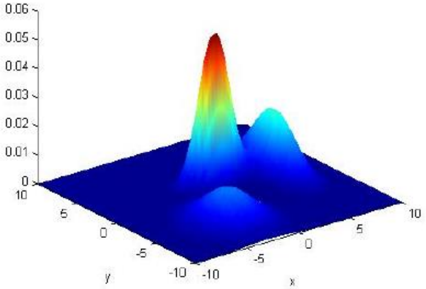
$$\sum_{j=1}^{K} w_j = 1 \quad \text{and} \quad 0 \le w_j \le 1$$

- Example

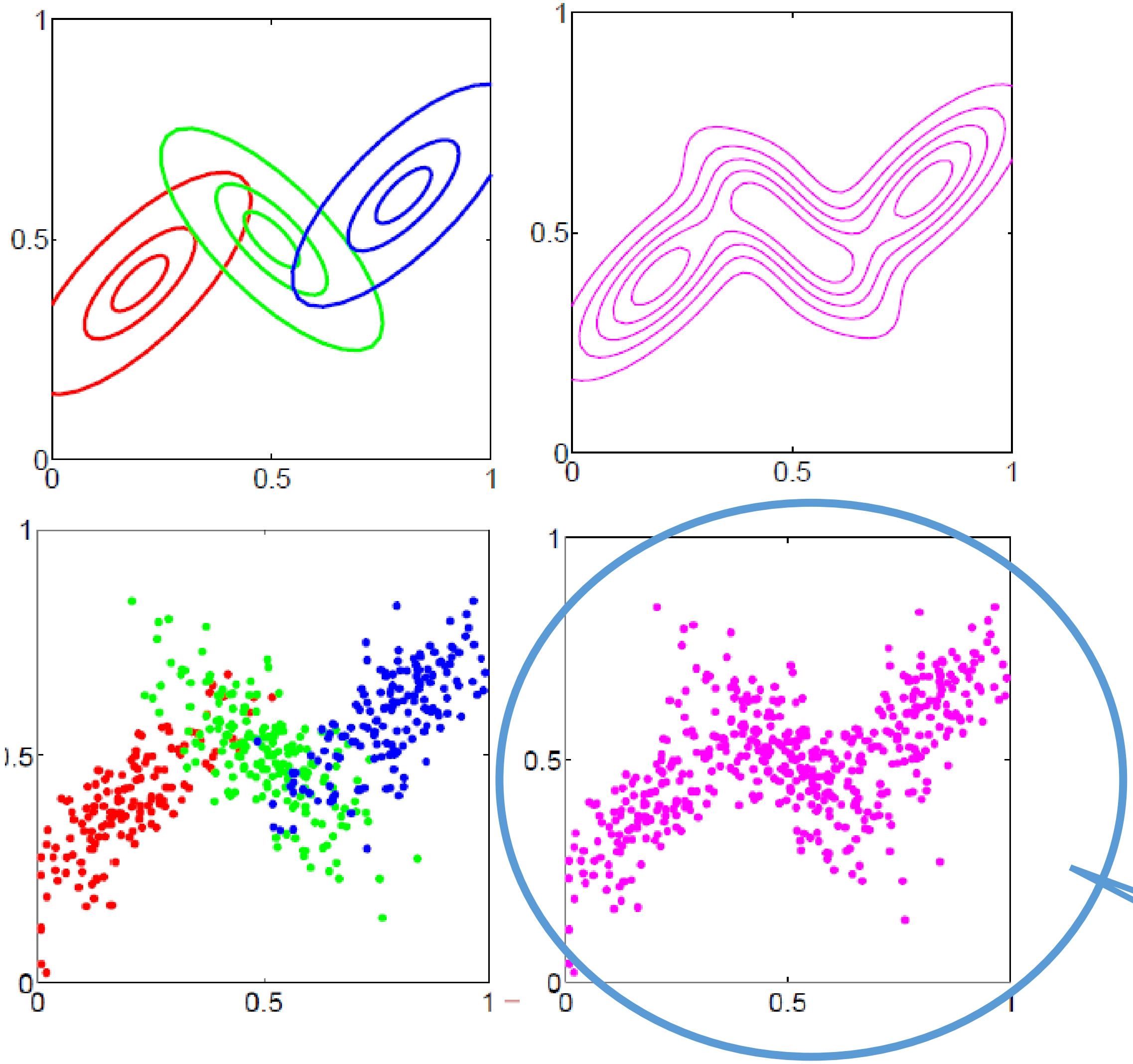
d=1:

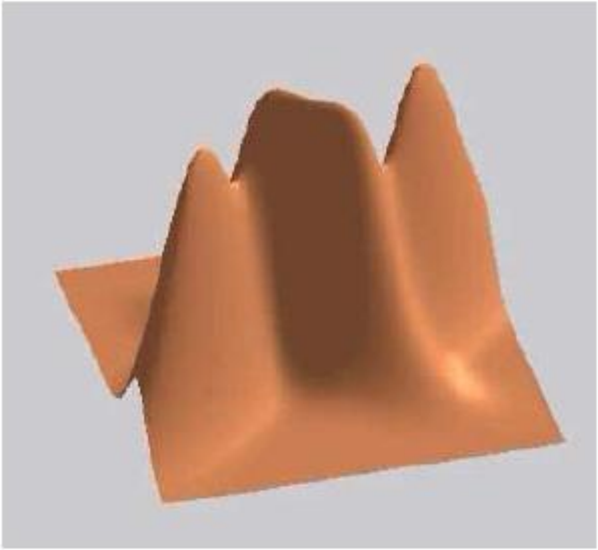


d=2:



Examples





Observation

Data generation

- Let the parameter set $\theta = \{w_j, \mu_j, \Sigma_j : j\}$, then the probability density of mixture Gaussian can be written as

$$p(x|\theta) = \sum_{j=1}^{K} w_j \cdot \mathcal{N}(x|\mu_j, \Sigma_j)$$

- Equivalent to generate data points in two steps
  - Select which component j the data point belongs to according to the categorical (or multinoulli) distribution of $(w_1, \dots, w_K)$
  - Generate the data point according to the probability of j-th component

Learning task

- Given a dataset $X = \left\{x^{(1)}, x^{(2)}, \cdots, x^{(N)}\right\}$ to train the GMM model

- Find the best θ that maximizes the probability $\mathbb{P}(X|\theta)$
- Maximal likelihood estimator (MLE)

$$\boxed{\theta^{*} = \arg\max_{\theta} p(X \mid \theta) = \arg\max_{\theta} \prod_{i=1}^{N} p(x_i \mid \theta)}$$

Introduce latent variable

- For data points $x^{(i)}, i = 1, \dots, N$, let's write the probability as

$$\mathbb{P}\left(x^{(i)} \middle| \theta\right) = \sum_{j=1}^{K} w_j \cdot \mathcal{N}(x^{(i)} | \mu_j, \Sigma_j)$$

  where $\sum_{j=1}^{K} w_j = 1$

- Introduce latent variable
  - $z^{(i)}$ is the Gaussian cluster ID indicates which Gaussian $x^{(i)}$ comes from
  - $\mathbb{P}\left(z^{(i)} = j\right) = w_j$ "Prior"
  - $\mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \theta\right) = \mathcal{N}(x^{(i)} | \mu_j, \Sigma_j)$
  - $\mathbb{P}\left(x^{(i)} \middle| \theta\right) = \sum_{j=1}^{K} \mathbb{P}\left(z^{(i)} = j\right) \cdot \mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \theta\right)$

Maximal likelihood

- Let $l(\theta) = \sum_{i=1}^{N} \log \mathbb{P}(x^{(i)}; \theta) = \sum_{i=1}^{N} \log \sum_j \mathbb{P}(x^{(i)}, z^{(i)} = j; \theta)$ be the log-likelihood
- We want to solve

$$\operatorname{argmax} l(\theta) = \operatorname{argmax} \sum_{i=1}^{N} \log \sum_{j=1}^{K} \mathbb{P}\left(z^{(i)} = j\right) \cdot \mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \theta\right)$$

$$= \operatorname{argmax} \sum_{i=1}^{N} \log \sum_{j=1}^{K} w_j \cdot \mathcal{N}(x^{(i)} | \mu_j, \Sigma_j)$$

- No closed solution by solving

$$\frac{\partial l(X|\theta)}{\partial w} = 0, \qquad \frac{\partial l(X|\theta)}{\partial \mu} = 0, \qquad \frac{\partial l(X|\theta)}{\partial \Sigma} = 0$$

Likelihood maximization

- If we know $z^{(i)}$ for all i, the problem becomes

$$\operatorname{argmax} l(\theta) = \operatorname{argmax} \sum_{i=1}^{N} \log \mathbb{P}\left(x^{(i)} \middle| \theta\right)$$

$$= \operatorname{argmax} \sum_{i=1}^{N} \log \mathbb{P}\left(x^{(i)}, z^{(i)} \middle| \theta\right)$$

$$= \operatorname{argmax} \sum_{i=1}^{N} \log \mathbb{P}\left(x^{(i)} \middle| \theta, z^{(i)}\right) + \log \mathbb{P}\left(z^{(i)} \middle| \theta\right)$$

$$= \operatorname{argmax} \sum_{i=1}^{N} \log \mathcal{N}\left(x^{(i)} \middle| \mu_{z^{(i)}}, \Sigma_{z^{(i)}}\right) + \log w_{z^{(i)}}$$

The solution is

- $w_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left\{z^{(i)} = j\right\}$
- $\mu_j = \frac{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\} x^{(i)}}{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\}}$
- $\Sigma_j = \frac{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\}\left(x^{(i)}-\mu_j\right)\left(x^{(i)}-\mu_j\right)^{\top}}{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\}}$

Average over each cluster

Likelihood maximization (cont.)

- Given the parameter $\theta = \{w_j, \mu_j, \Sigma_j : j\}$, the posterior distribution of each latent variable $z^{(i)}$ can be inferred
- $\mathbb{P}\left(z^{(i)} = j \middle| x^{(i)}; \theta\right) = \dfrac{\mathbb{P}\left(x^{(i)}, z^{(i)} = j \middle| \theta\right)}{\mathbb{P}\left(x^{(i)} \middle| \theta\right)}$
- $= \dfrac{\mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \mu_j, \Sigma_j\right)\mathbb{P}\left(z^{(i)} = j \middle| w\right)}{\sum_{j'=1}^{K} \mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j'; \mu_{j'}, \Sigma_{j'}\right)\mathbb{P}\left(z^{(i)} = j' \middle| w\right)}$

- Or $w_j^{(i)} = \mathbb{P}\left(z^{(i)} = j \middle| x^{(i)}; \theta\right) \propto \mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \mu_j, \Sigma_j\right)\mathbb{P}\left(z^{(i)} = j \middle| w\right)$

Likelihood maximization (cont.)

$w_j^{(i)} = \mathbb{P}(z^{(i)} = j | x^{(i)}; \theta)$

- For every possible values of $z^{(i)}$'s

The solution is

- $w_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\}$
- $\mu_j = \frac{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\} x^{(i)}}{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\}}$
- $\Sigma_j = \frac{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\} (x^{(i)}-\mu_j)(x^{(i)}-\mu_j)^{\top}}{\sum_{i=1}^{N} \mathbf{1}\{z^{(i)}=j\}}$

Which is equivalent to

- $w_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\}$
- $\left(\sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\}\right) \mu_j = \sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\} x^{(i)}$
- $\left(\sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\}\right) \Sigma_j = \sum_{i=1}^{N} \mathbf{1}\{z^{(i)} = j\} \left(x^{(i)} - \mu_j\right)\left(x^{(i)} - \mu_j\right)^{\top}$

Take expectation

- Take the expectation on probability of $z^{(i)}$

Take expectation on two sides

- $w_j = \frac{1}{N} \sum_{i=1}^{N} \mathbb{P}(z^{(i)} = j)$
- $\left(\sum_{i=1}^{N} \mathbb{P}(z^{(i)} = j)\right) \mu_j = \sum_{i=1}^{N} \mathbb{P}(z^{(i)} = j) x^{(i)}$
- $\left(\sum_{i=1}^{N} \mathbb{P}(z^{(i)} = j)\right) \Sigma_j = \sum_{i=1}^{N} \mathbb{P}(z^{(i)} = j) \left(x^{(i)} - \mu_j\right)\left(x^{(i)} - \mu_j\right)^{\top}$

The solution is

- $w_j = \frac{1}{N} \sum_{i=1}^{N} w_j^{(i)}$
- $\mu_j = \frac{\sum_{i=1}^{N} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{N} w_j^{(i)}}$
- $\Sigma_j = \frac{\sum_{i=1}^{N} w_j^{(i)} (x^{(i)}-\mu_j)(x^{(i)}-\mu_j)^{\top}}{\sum_{i=1}^{N} w_j^{(i)}}$

Likelihood maximization (cont.)

- If we know the distribution of $z^{(i)}$'s, then it is equivalent to maximize

$$l\left(\mathbb{P}_{z^{(i)}}, \theta\right) = \sum_{i=1}^{N} \sum_{j=1}^{K} \mathbb{P}\left(z^{(i)} = j\right) \log \frac{\mathbb{P}\left(x^{(i)}, z^{(i)} = j | \theta\right)}{\mathbb{P}\left(z^{(i)} = j\right)}$$

- Compared with previous if we know the values of $z^{(i)}$

$$\operatorname{argmax} \sum_{i=1}^{N} \log \mathbb{P}\left(x^{(i)}, z^{(i)} \middle| \theta\right)$$

  - Note that it is hard to solve if we directly maximize

$$l(\theta) = \sum_{i=1}^{N} \log \mathbb{P}\left(x^{(i)} \middle| \theta\right) = \sum_{i=1}^{N} \log \sum_{j=1}^{K} \mathbb{P}\left(z^{(i)} = j | \theta\right) \mathbb{P}\left(x^{(i)} \middle| z^{(i)} = j; \theta\right)$$

is a lower bound of

Expectation maximization methods

- E-step:
  - Infer the posterior distribution of the latent variables given the model parameters
- M-step:
  - Tune parameters to maximize the data likelihood given the latent variable distribution
- EM methods
  - Iteratively execute E-step and M-step until convergence

EM for GMM

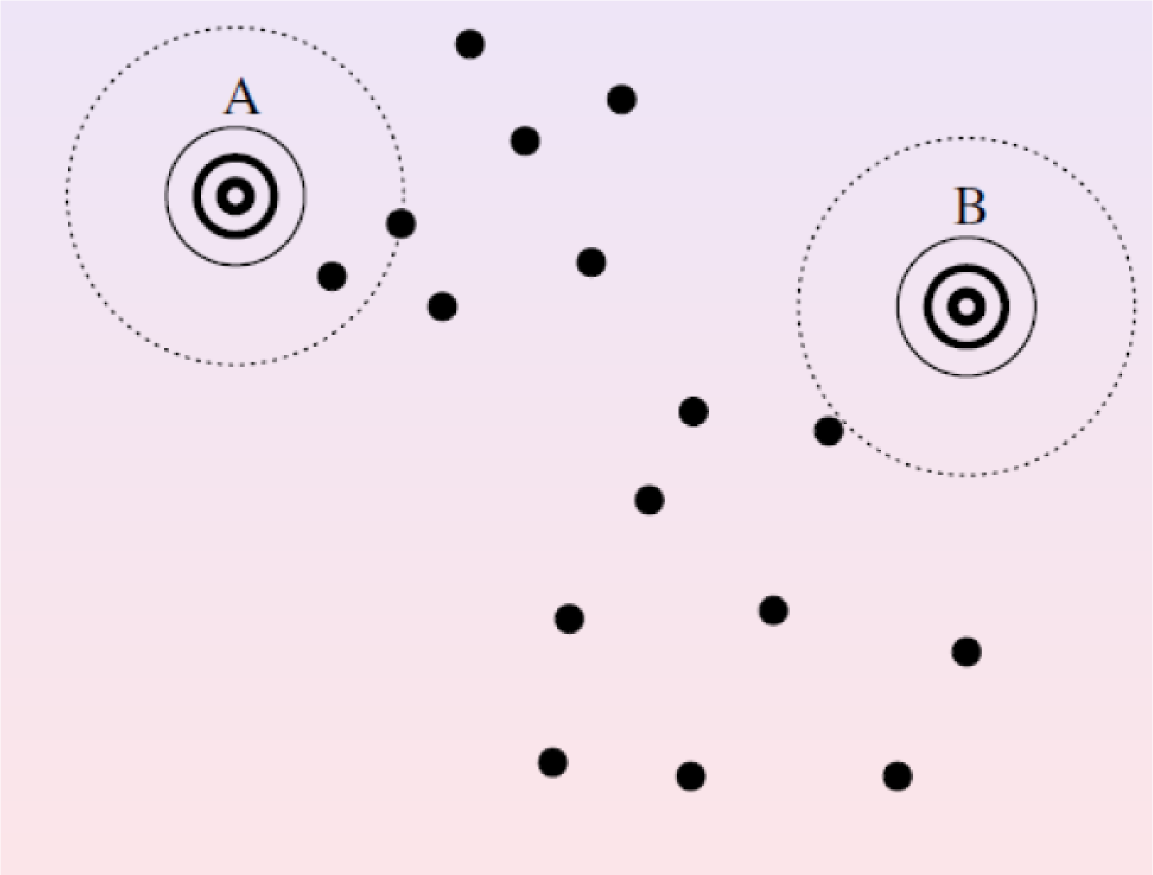
- Repeat until convergence:{
  (E-step) For each i, j, set
  $$w_j^{(i)} = \mathbb{P}\left(z^{(i)} = j \middle| x^{(i)}, w, \mu, \Sigma\right)$$
  (M-step) Update the parameters
  $$w_j = \frac{1}{N}\sum_{i=1}^{N} w_j^{(i)}, \qquad \mu_j = \frac{\sum_{i=1}^{N} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{N} w_j^{(i)}}$$
  $$\Sigma_j = \frac{\sum_{i=1}^{N} w_j^{(i)} \left(x^{(i)} - \mu_j\right)\left(x^{(i)} - \mu_j\right)^{\top}}{\sum_{i=1}^{N} w_j^{(i)}}$$

  }

Example

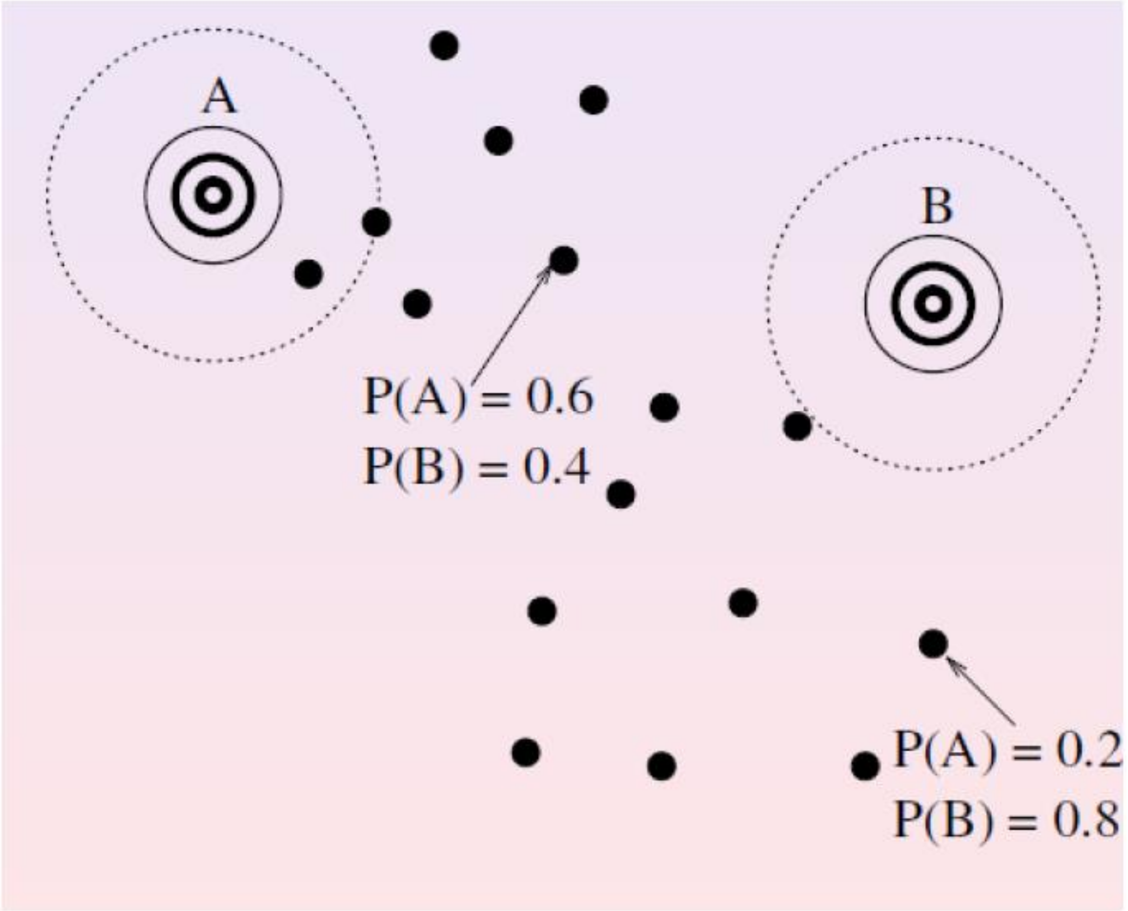
$$p(x|\theta) = \sum_{j=1}^{K} w_j \cdot \mathcal{N}(x|\mu_j, \Sigma_j)$$

- Hidden variable: for each point, which Gaussian generates it?



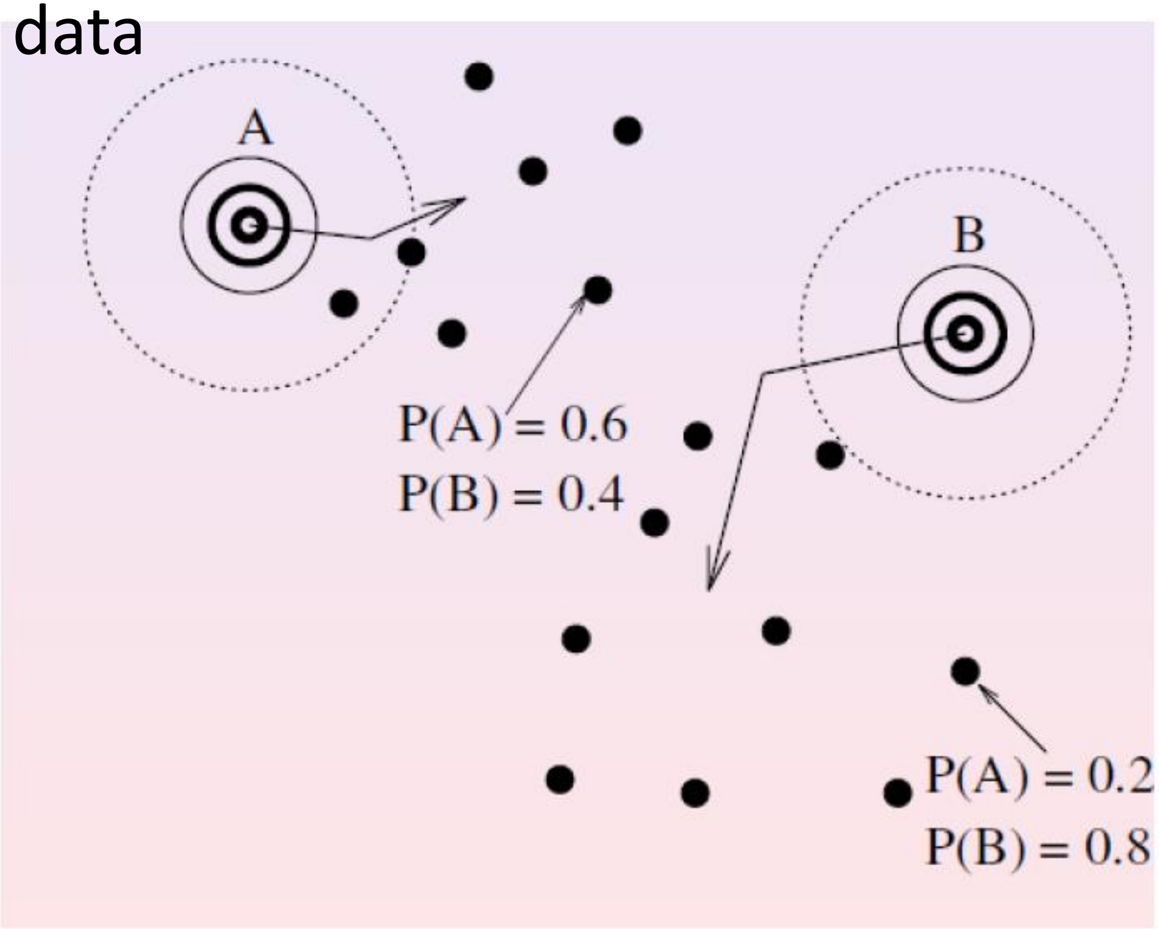
Example (cont.)

- E-step: for each point, estimate the probability that each Gaussian component generated it

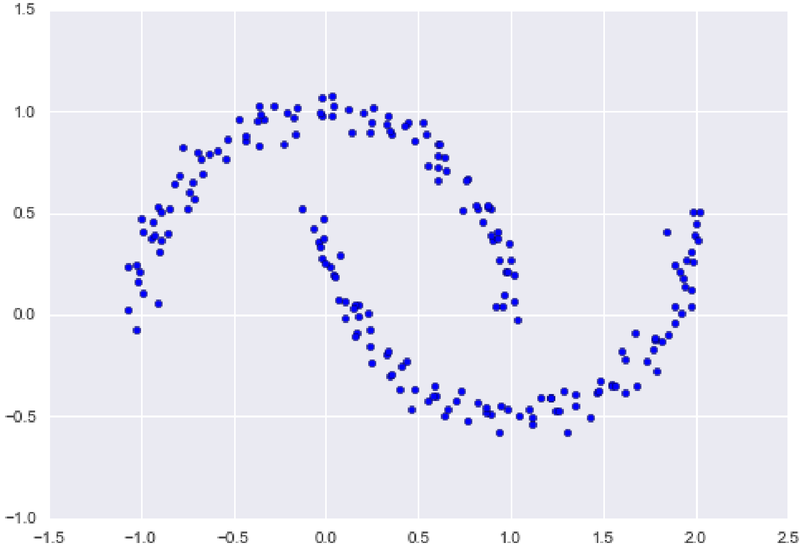


Example (cont.)

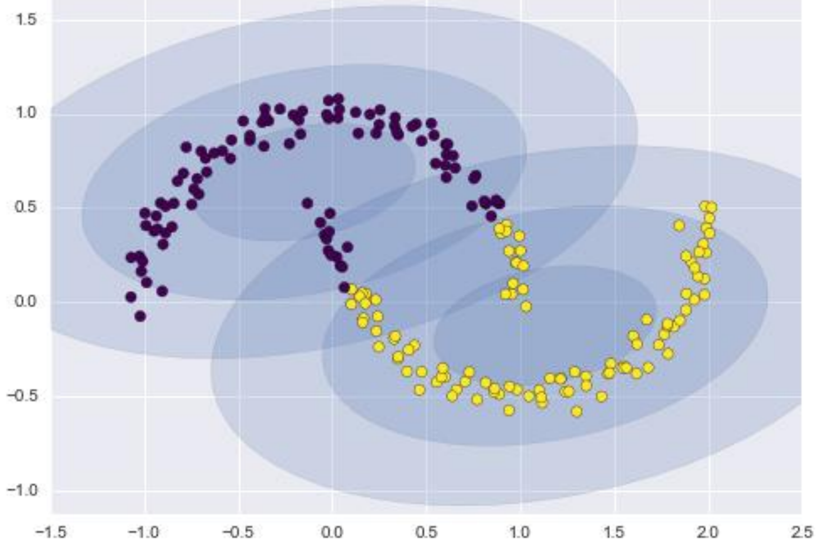
- M-Step: modify the parameters according to the hidden variable to maximize the likelihood of the data



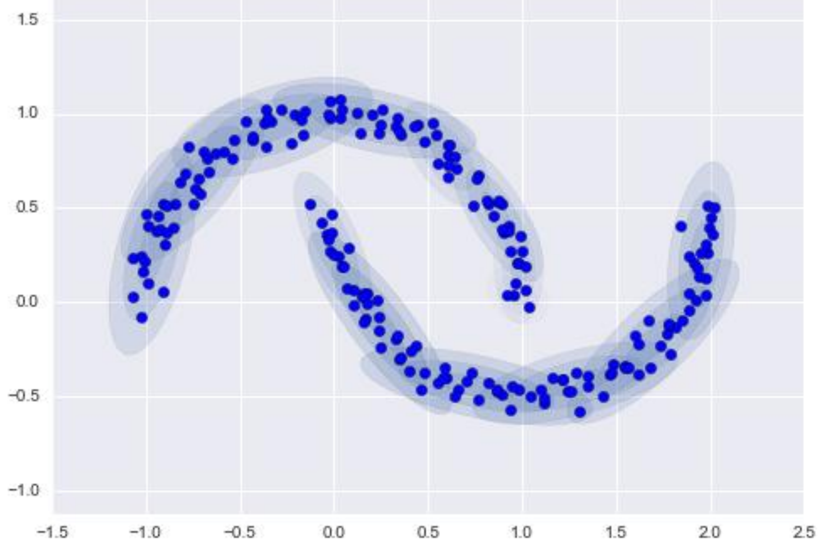
Example



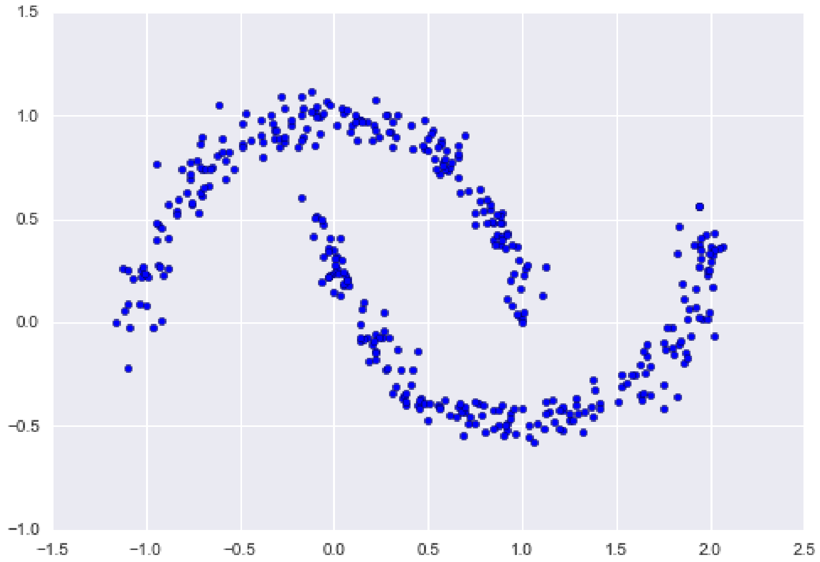
$K = 2$



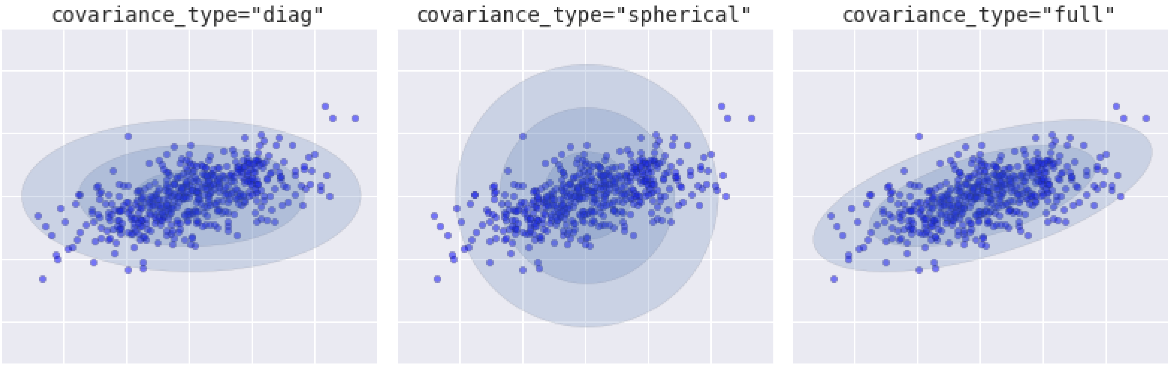
$K = 16$



400 new data points generated from the 16-GMM

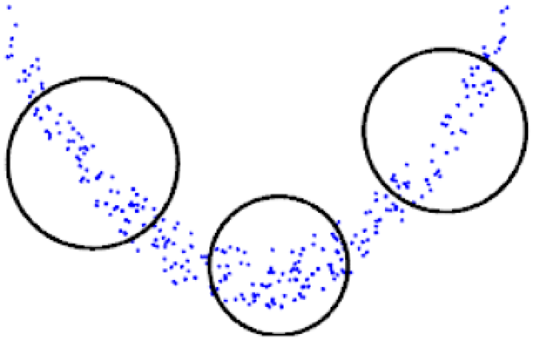


Remaining issues (cont.)



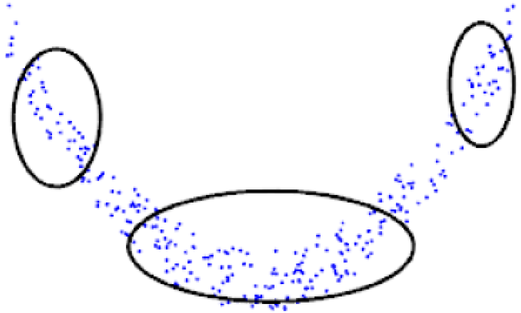
- Simplification of the covariance matrices

Case 1: Spherical covariance matrix $\Sigma_j = diag(\sigma_j^2, \sigma_j^2, ..., \sigma_j^2) = \sigma_j^2 I$



-Less precise.
-Very efficient to compute.

Case 2: Diagonal covariance matrix $\Sigma_j = diag(\sigma_{j1}^2, \sigma_{j2}^2, ..., \sigma_{jd}^2)$



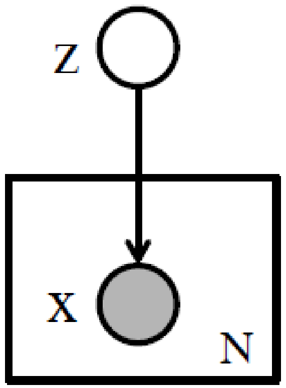
-More precise.
-Efficient to compute.

Expectation and Maximization

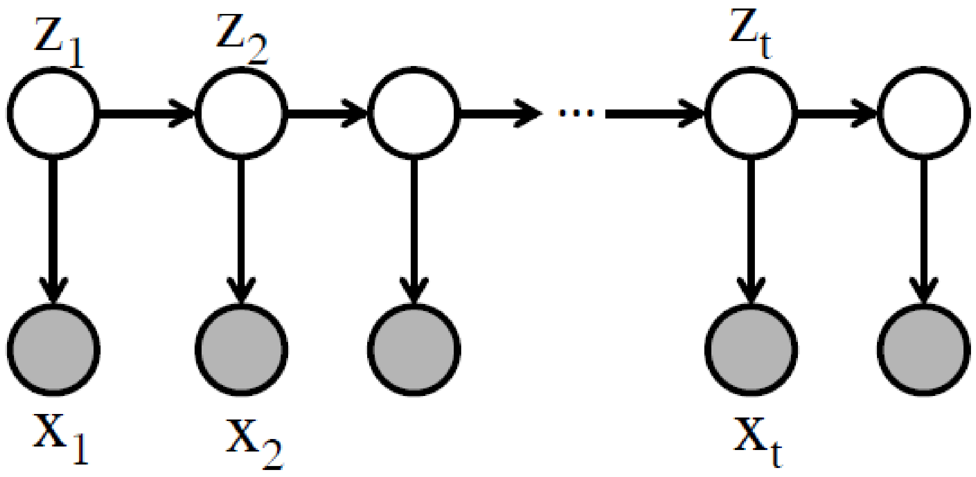
Background: Latent variable models

- Some of the variables in the model are not observed
- Examples: mixture model, Hidden Markov Models, LDA, etc

Mixture Model



Hidden Markov Model



Background: Marginal likelihood

- Joint model $\mathbb{P}(x, z|\theta)$, θ is the model parameter
- With z unobserved, we marginalize out z and use the marginal log-likelihood for learning

$$\mathbb{P}(x|\theta) = \sum_{z} \mathbb{P}(x, z|\theta)$$

- Example: mixture model

$$\mathbb{P}(x|\theta) = \sum_{k} \mathbb{P}(x|z = k, \theta_k)\mathbb{P}(z = k|\theta_k) = \sum_{k} \pi_k \mathbb{P}(x|z = k, \theta_k)$$

where π_k is the mixing proportions

Examples

- Mixture of Bernoulli

$$p(\mathbf{x}|\mathbf{z}=k,\theta_k) = p(\mathbf{x}|\mu_k) = \prod_i \mu_k^{x_i}(1-\mu_k)^{1-x_i}$$

- Mixture of Gaussians

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}=k,\theta_k) &= p(\mathbf{x}|\mu_k,\Sigma_k) \\ &= \frac{1}{|2\pi\Sigma_k|^{\frac{D}{2}}}\exp\left(-\frac{1}{2}(\mathbf{x}-\mu_k)^\top\Sigma_k^{-1}(\mathbf{x}-\mu_k)\right) \end{aligned}$$

- Hidden Markov Model

$$p(\mathbf{Z}) = p(\mathbf{z}_1)\prod_t p(\mathbf{z}_t|\mathbf{z}_{t-1})$$

$$p(\mathbf{X}|\mathbf{Z}) = \prod_t p(\mathbf{x}_t|\mathbf{z}_t)$$

Learning the hidden variable Z

- If all z observed, the likelihood factorizes, and learning is relatively easy

$$\ell(\mathbf{x}, \mathbf{z}|\theta) = \log p(\mathbf{x}, \mathbf{z}|\theta) = \log p(\mathbf{z}|\theta) + \log p(\mathbf{x}|\mathbf{z}, \theta)$$

- If z not observed, we have to handle a sum inside log
  - Idea 1: ignore this problem and simply take derivative and follow the gradient
  - Idea 2: use the current θ to estimate z, fill them in and do fully-observed learning

- Is there a better way?

General EM methods

- Repeat until convergence{
  (E-step) For each data point i, set
  $$q_i(j) = \mathbb{P}\left(z^{(i)} = j | x^{(i)}; \theta\right)$$
  (M-step) Update the parameters
  $$\text{argmax}_\theta \; l(q_i, \theta) = \sum_{i=1}^{N} \sum_{j} q_i(j) \log \frac{\mathbb{P}(x^{(i)}, z^{(i)} = j; \theta)}{q_i(j)}$$

}

Convergence of EM

- Denote $\theta^{(t)}$ and $\theta^{(t+1)}$ as the parameters of two successive iterations of EM, we want to prove that

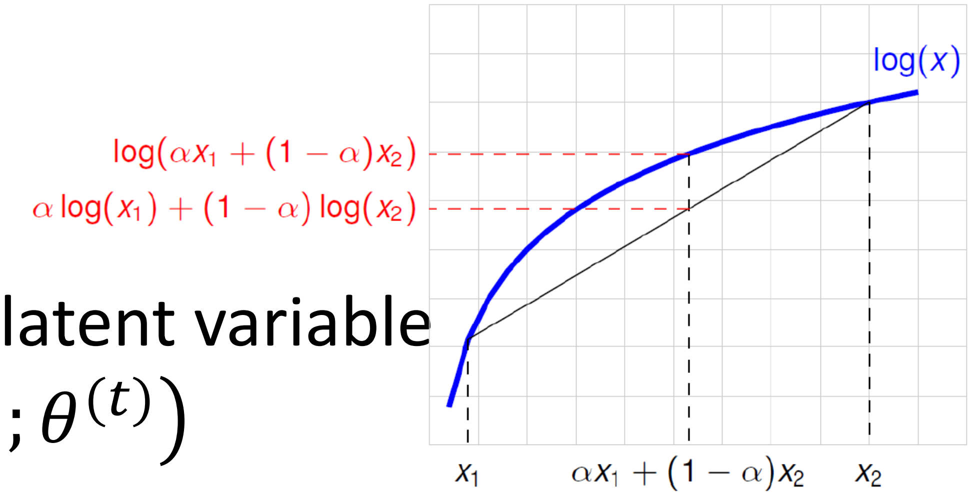
$$l\left(\theta^{(t)}\right) \leq l\left(\theta^{(t+1)}\right)$$

where note that

$$l(\theta) = \sum_{i=1}^{N} \log \mathbb{P}(x^{(i)}; \theta) = \sum_{i=1}^{N} \log \sum_{j} \mathbb{P}(x^{(i)}, z^{(i)} = j; \theta)$$

- This shows EM always monotonically improves the log-likelihood, thus ensures EM will at least converge to a local optimum

Proof



- Start from $\theta^{(t)}$, we choose the posterior of latent variable

$$q_i^{(t)}(j) = p\left(z^{(i)} = j | x^{(i)}; \theta^{(t)}\right)$$

- By Jensen's inequality on the log function

- $l\left(q, \theta^{(t)}\right) = \sum_{i=1}^{N} \sum_j q_i(j) \log \frac{p(x^{(i)}, z^{(i)} = j; \theta^{(t)})}{q_i(j)}$

$$\leq \sum_{i=1}^{j} \log \sum_j q_i(j) \cdot \frac{p\left(x^{(i)}, z^{(i)} = j; \theta^{(t)}\right)}{q_i(j)}$$

$$= \sum_{i=1}^{j} \log \sum_j p\left(x^{(i)}, z^{(i)} = j; \theta^{(t)}\right) = l\left(\theta^{(t)}\right) \Rightarrow \boxed{l\left(\theta^{(t)}\right) = l\left(q_i^{(t)}, \theta^{(t)}\right) \geq l\left(q, \theta^{(t)}\right)}$$

- When $q_i^{(t)}(j) = p\left(z^{(i)} = j | x^{(i)}; \theta^{(t)}\right)$, the above inequality is equality

Proof (cont.)

- Then the parameter $\theta^{(t+1)}$ is obtained by maximizing

$$l\left(q_i^{(t)}, \theta\right) = \sum_{i=1}^{N} \sum_{j} q_i^{(t)} \log \frac{p(x^{(i)}, z^{(i)} = j; \theta)}{q_i^{(t)}}$$

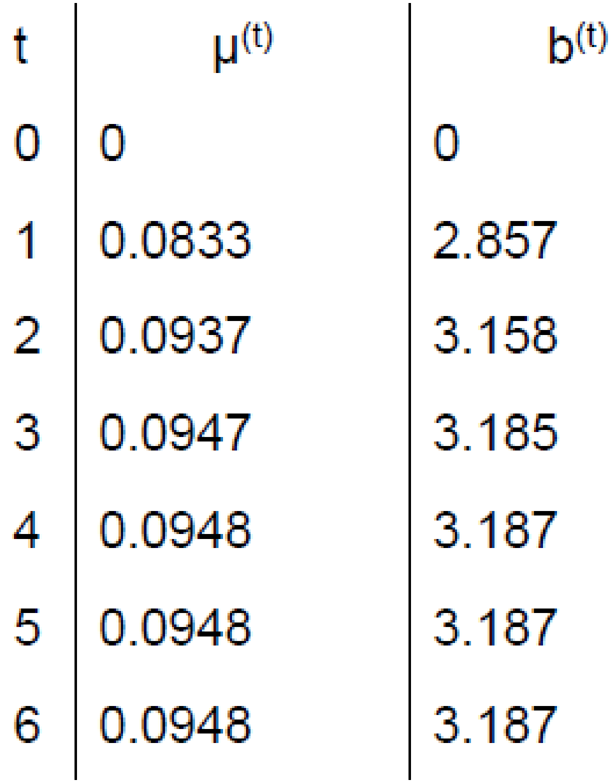
- Thus

$$l(\theta^{(t+1)}) \geq l\left(q_i^{(t)}, \theta^{(t+1)}\right) \geq l\left(q_i^{(t)}, \theta^{(t)}\right) = l(\theta^{(t)})$$

- Since the likelihood is at most 1, EM will converge (to a local optimum)

Example

- Follow previous example, suppose $h = 20, c = 10, d = 10, \mu^{(0)} = 0$
- Then
- Convergence is generally linear: error decreases by a constant factor each time step



t	$\mu^{(t)}$	$b^{(t)}$
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

K-means and EM

- After initialize the position of the centers, K-means interactively execute the following two operations:
    - Assign the label of the points based on their distances to the centers
        - Specific posterior distribution of latent variable
        - E-step
    - Update the positions of the centers based on the labeling results
        - Given the labels, optimize the model parameters
        - M-step