K-Means I

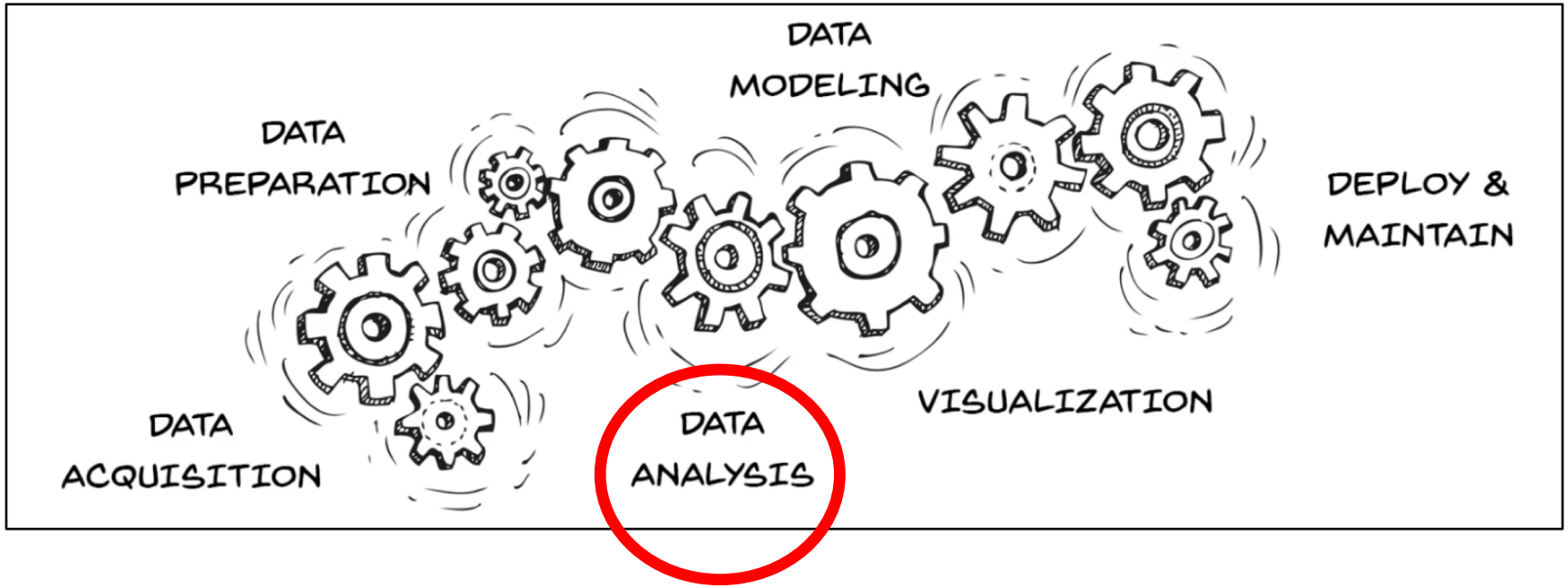
M. R. Hasan

CSCE 411/811

Data Modeling for Systems Development

Course Goal

- We will study **two broad topics**.
  - Data Analysis
  - Data Modeling



We will study **two broad topics**.
- Data Analysis
- Data Modeling

Topics

- Who are our customers? (clustering)
- Is this fraud? (Anomaly Detection)
- Do you want fries with that? (Association-Rule Mining)
- Chur or No Churn, that is the question (Classification)
- How much will it cost? (Regression)

- Data Analysis
  - Exploratory Data Analysis
  - Clustering (K-Means, Gaussian Mixture Model, DBSCAN, etc.)
  - Anomaly/Outlier Detection (GMM, DBSCAN, Local Outlier Factor, Isolation Forest, Minimum Covariance Determinant, etc.)
  - Association-rule Mining (Apriori)
  - Predictive Analytics (Classification & Regression)
  - Complexity of Data Analysis; Possible solutions: Algorithmic & Hardware

Python will be used for data analysis topics.

Readings

- Bishop: 9.1, 9.1.1
- Murphy: 11.4.2.5, 11.4.2.6, 11.4.2.7
- Geron: 9

What We Will Cover

- Unsupervised Learning
- Clustering
- K-Means Algorithm
- Expectation-Maximization (EM) Algorithm
- Theoretical Analysis

Clustering: Big Picture

Clustering

- Clustering techniques fall under the broad category of Unsupervised Learning.
- Unsupervised learning is a type of machine learning algorithm.

Unsupervised Learning

- In unsupervised learning problems we only have the **features of the data (X),** but **no labels (Y)**!
- Hence, the learning from data is **based only on X**.
- Y is not present in the training data.
- It's a learning technique "*without teacher*".

- *"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

There is a **huge potential** in unsupervised learning that we have only barely started to sink our teeth into.

Unsupervised Learning

- Most Common Unsupervised learning tasks:
  - Association Rule Mining
  - Dimensionality Reduction
  - Clustering
  - Anomaly detection
  - Density estimation

Unsupervised Learning

- Clustering
- The goal is to **group similar instances together** into clusters.
- Clustering is a great tool for data analysis, customer segmentation, recommender systems, search engines, image segmentation, semi-supervised learning, dimensionality reduction, and more.

Unsupervised Learning

- Anomaly detection
- The objective is to learn **what "normal" data looks like**, and then use that to detect abnormal instances, such as **defective items on a production line** or a *new trend in a time series*.

Unsupervised Learning

- Density estimation
- It assumes that data is generated by a random process.
- Then, it **estimates the probability density function** (PDF) of the random process that generated the dataset.
- Density estimation is commonly used for **anomaly detection**: instances located in very low-density regions are likely to be anomalies.
- It is also useful for **data analysis** and **visualization**.

Unsupervised Learning

- We will discuss the **clustering problem**.
- The clustering problems is known by **various names**:
  - Vector Quantization
  - Latent Variable Models
  - Hidden Variable Models
  - Mixture Models

Clustering problem

- We will discuss the following **clustering algorithms**.
  - K-Means
  - Gaussian Mixture Model (GMM)
  - DBSCAN

We begin with the **k-Means Clustering algorithm**

K-Means Clustering Algorithm

- It was proposed by **Stuart Lloyd at Bell Labs** in **1957** as a technique for pulse-code modulation.
- But it was only published outside of the company in 1982.
- In **1965 Edward W. Forgy** had published virtually the same algorithm.
- So K-Means is sometimes referred to as **Lloyd–Forgy**.

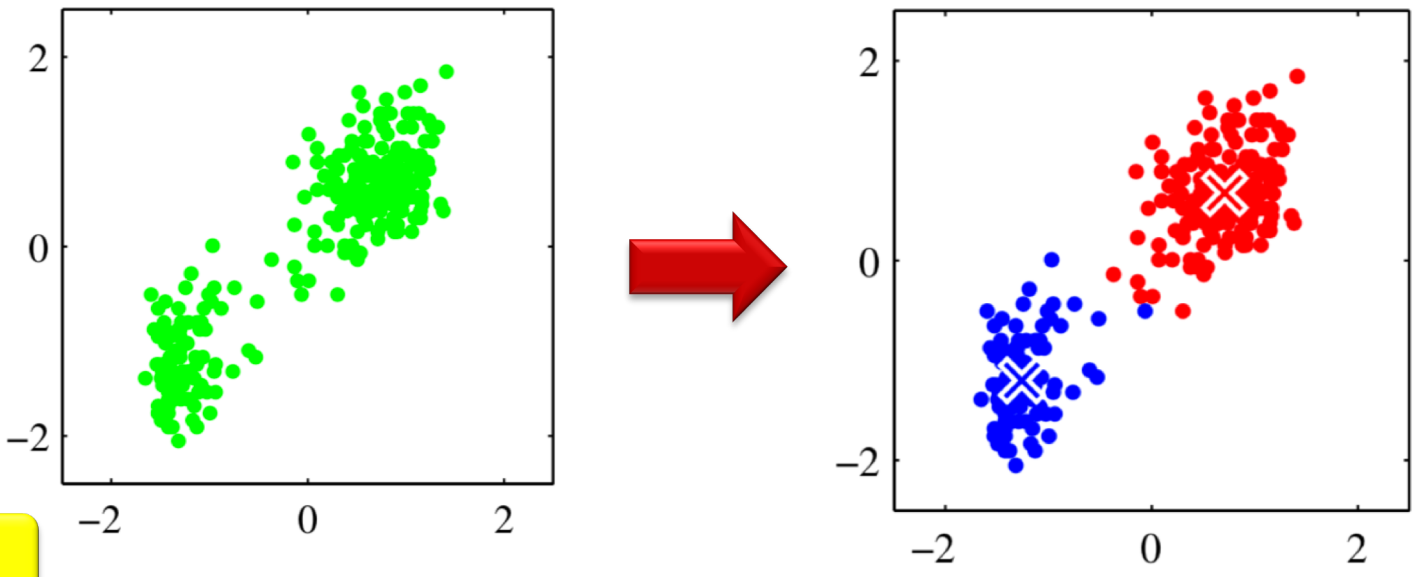
K-Means Clustering Algorithm: Roadmap

- First we will **define the problem.**
- Then, we will provide a **procedural description** of the algorithm.
- Finally, we will provide a **formal description and analysis** of the algorithm.

K-Means Algorithm: Problem Statement

Problem Statement

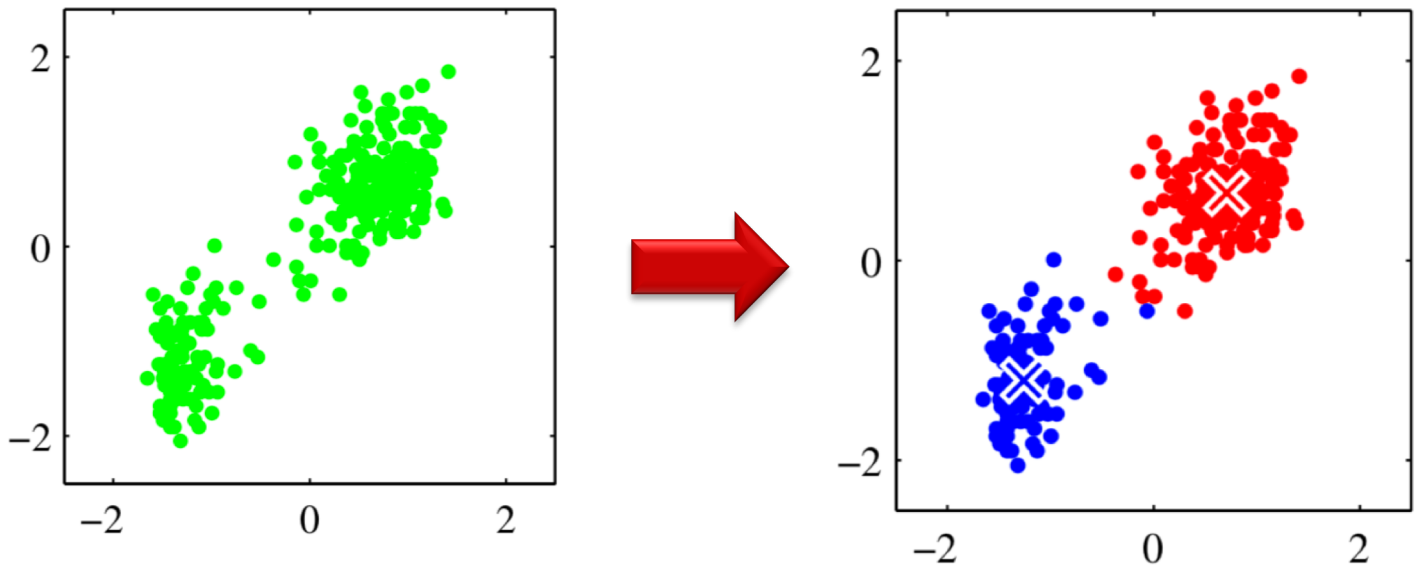
- Suppose we have a data set $\{x_1, \ldots, x_N\}$ consisting of N observations of a random **d-dimensional Euclidean variable x**.
- Our goal is to **partition the data set** into some number **k of clusters**.



d = 2

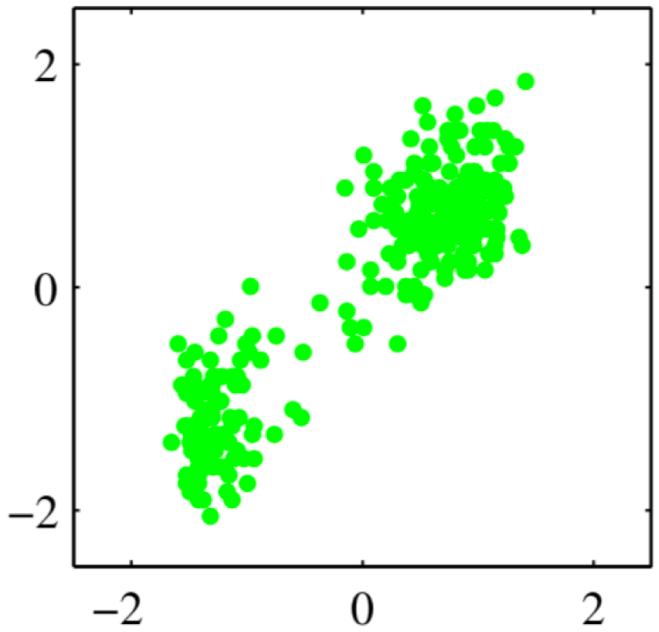
Problem Statement

- Assume for the moment that the **value of k is given**.
- **Intuitively**, we might think of a cluster as comprising a group of data points whose **inter-point distances are small** compared with the distances to points outside of the cluster.



Problem Statement

- We aim to solve the **following problem**:
- How to **identify groups, or clusters, of data points** in a multidimensional space?



First we will provide a **procedural description** of the k-Means clustering algorithm

K-Means Algorithm: Procedural Description

K-Means Clustering

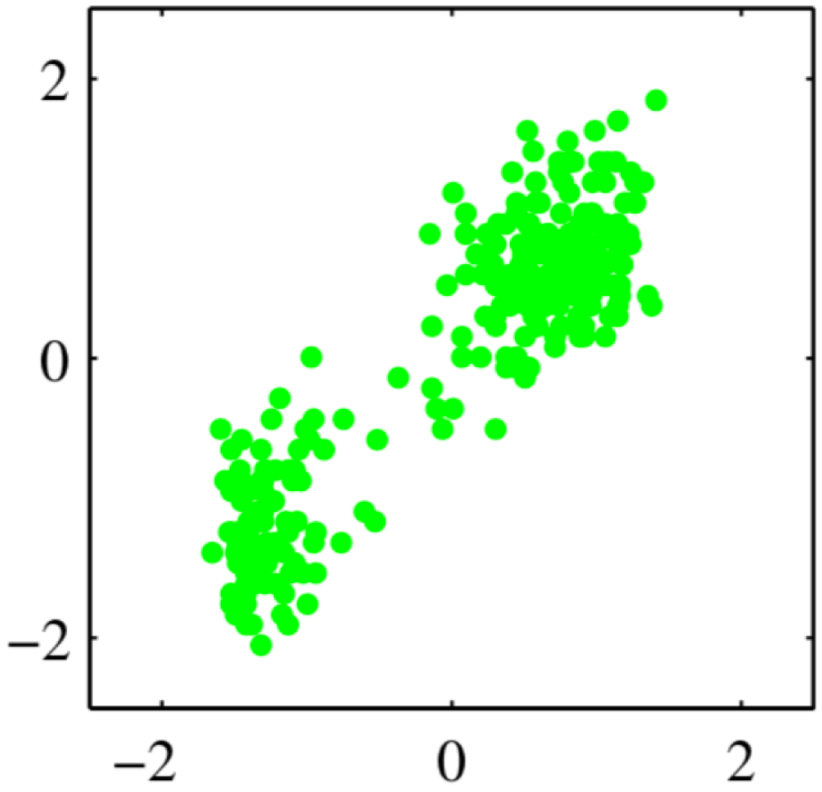
1) Set the number of clusters we want to have (e.g. k = 2)

2) Randomly guess k cluster center locations

3) Each data point finds out which center it's closest to

4) Each center finds the centroid of the points it owns and jumps there

5) Repeat until terminated



K-Means Clustering

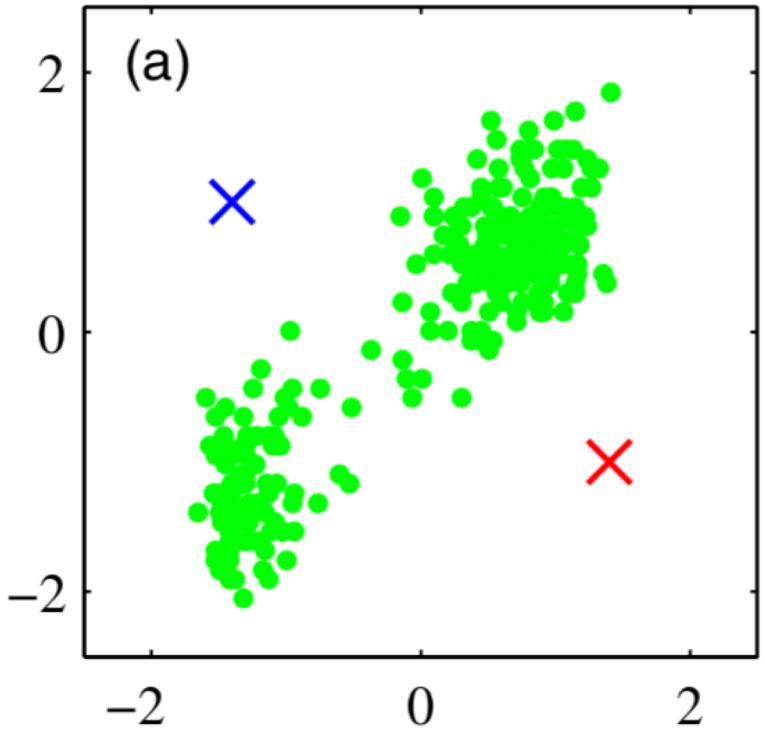
1) Set the number of clusters we want to have (e.g. k = 2)

2) Randomly guess k cluster center locations

3) Each data point finds out which center it's closest to

4) Each center finds the centroid of the points it owns and jumps there

5) Repeat until terminated



K-Means Clustering

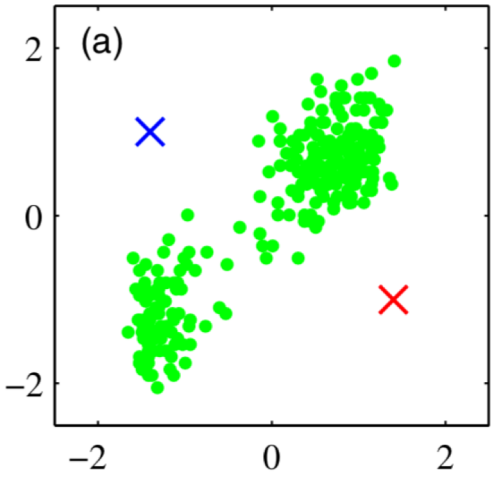
1) Set the number of clusters we want to have (e.g. k = 2)

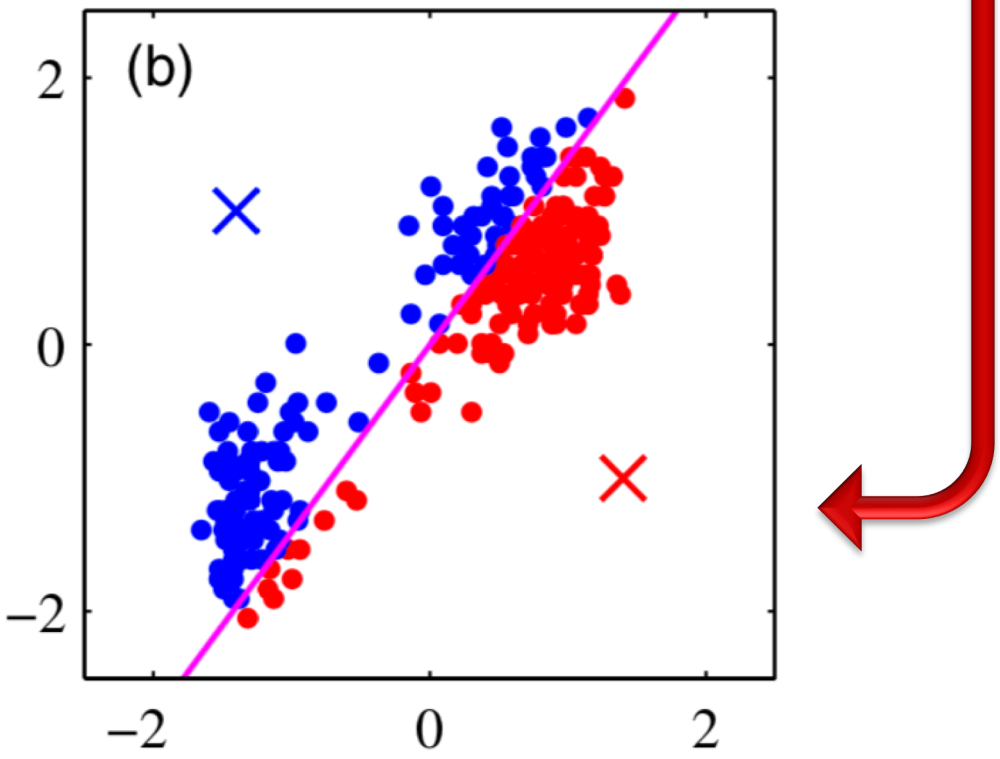
2) Randomly guess k cluster center locations

3) Each data point finds out which center it's closest to

4) Each center finds the centroid of the points it owns and jumps there

5) Repeat until terminated





K-Means Clustering

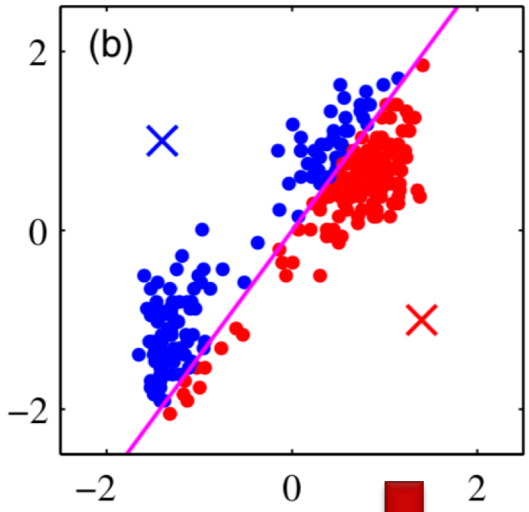
1) Set the number of clusters we want to have (e.g. k = 2)

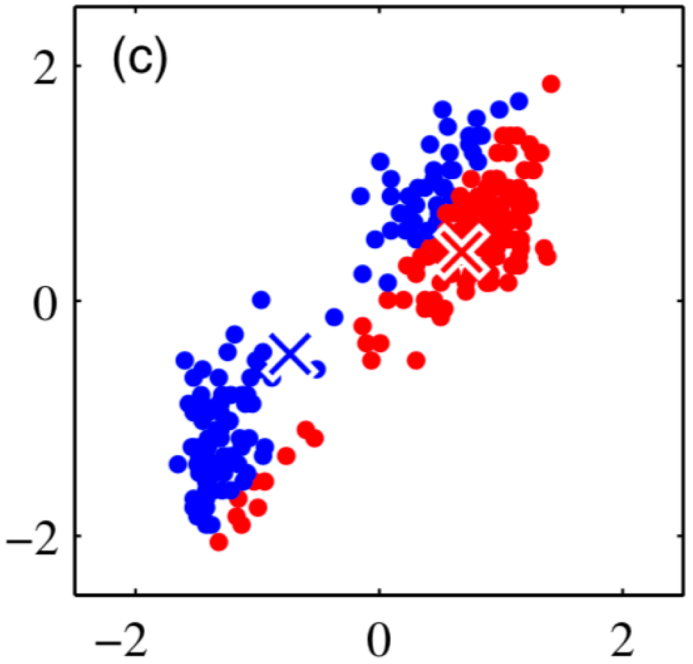
2) Randomly guess k cluster center locations

3) Each data point finds out which center it's closest to

4) Each center finds the centroid of the points it owns and jumps there

5) Repeat until terminated





K-Means Clustering

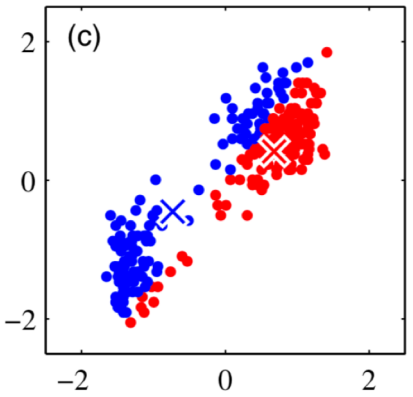
1) Set the number of clusters we want to have (e.g. k = 2)

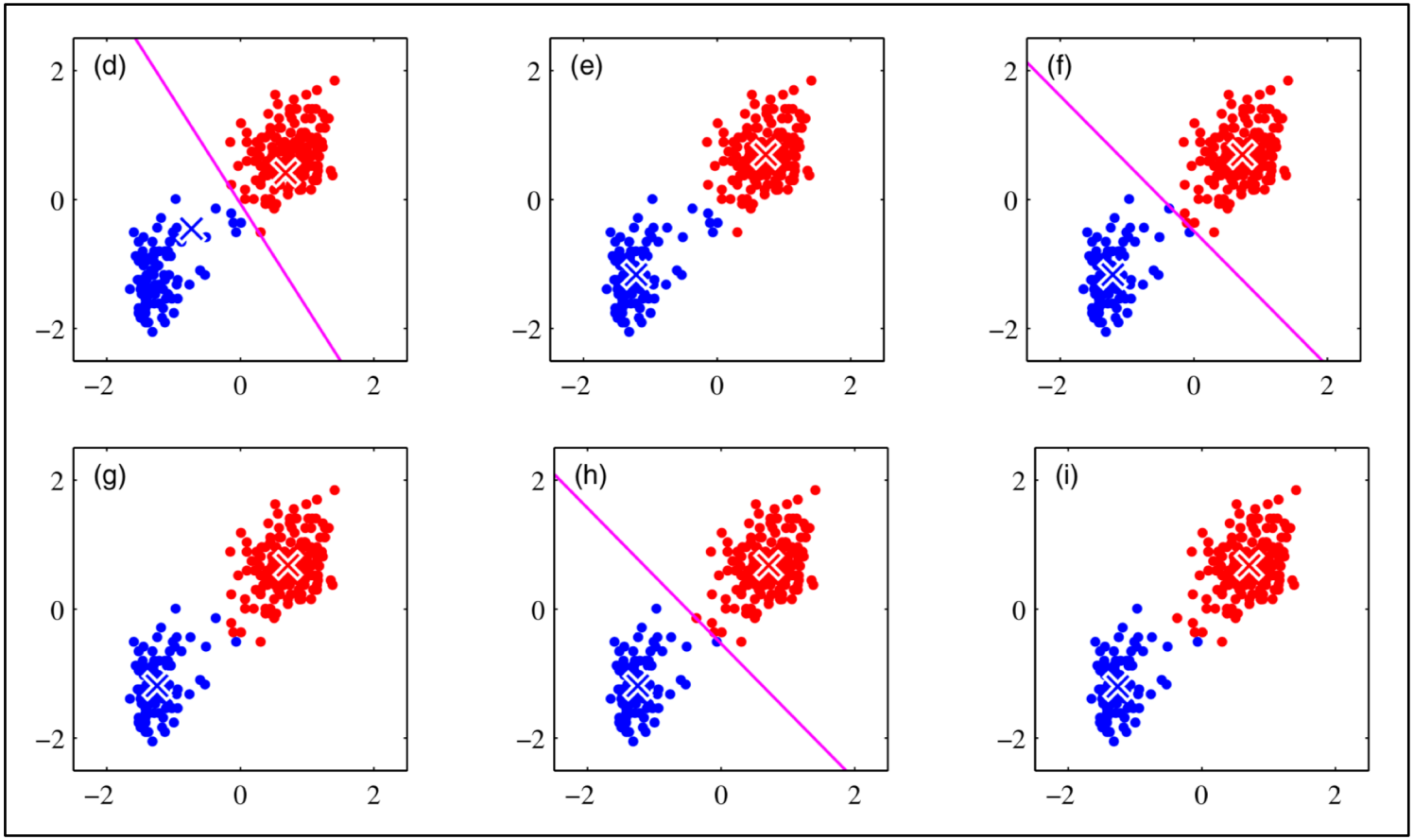
2) Randomly guess k cluster center locations

3) Each data point finds out which center it's closest to.

4) Each center finds the centroid of the points it owns and jumps there

5) Repeat until terminated



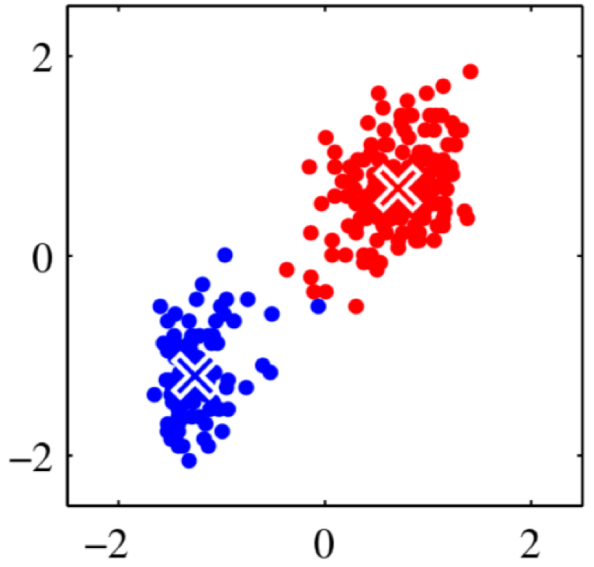


K-Means: Problem Formulation

Problem Formalization

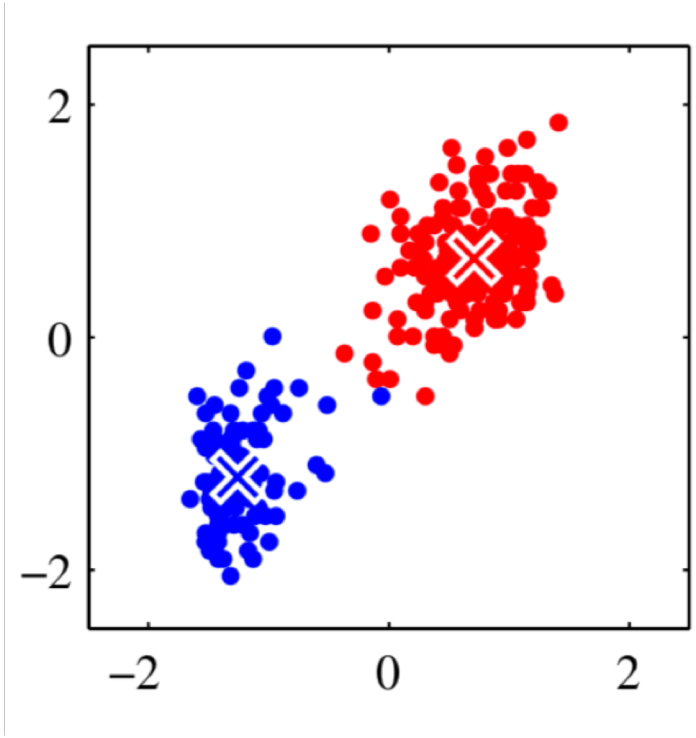
- Let's **formalize the problem**.
- We represent the **centers of the clusters** by a set of d-dimensional vectors μ_j, where $j = 1, \ldots, k \in \mathbb{R}^d$

Here μ_j is a **prototype** associated with the ***j***th cluster.



Problem Formalization

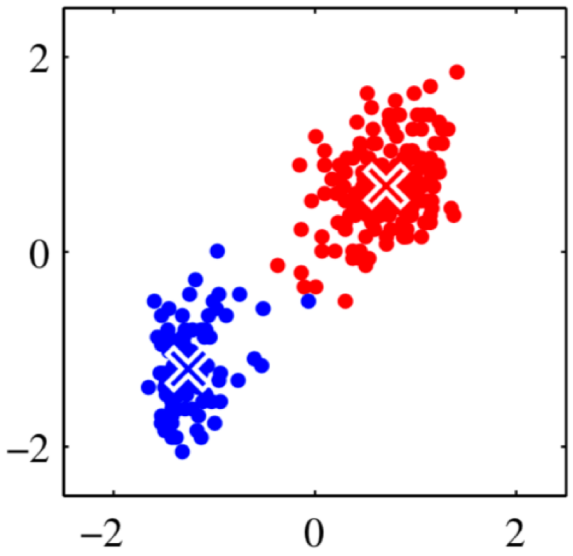
- To represent **cluster assignment IDs** we use a set of N-dimensional vector $\mathbf{C_i}$, where i = 1, 2, .., N.
- The C_i takes a value from the set {1, 2, …, k}



Problem Formalization

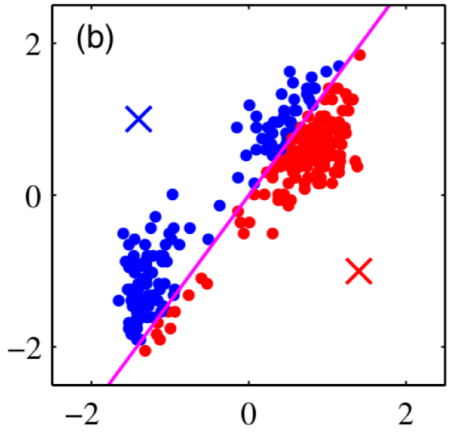
- Cluster assignments of **each data point x_i** is done by a set of **binary indicator variables** $a_{ij} \in \{0, 1\}$
- Where $j = 1, 2, 3, \ldots, k$ describing which of k clusters the data point x_i is assigned to.
- So, if data point **x_i is assigned to cluster j**, then $a_{ij} = 1$
- And $a_{ij} = 0$, $i \neq j$.

This is known as **1-of-k coding scheme**.



Problem Formalization

k = 2

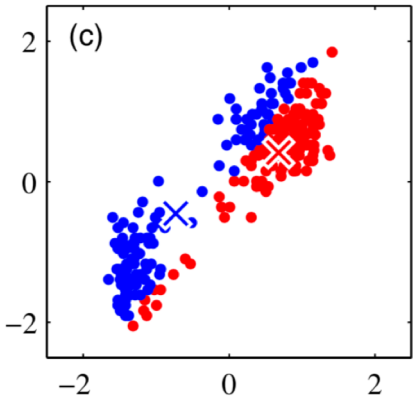


- Randomly initialize k centers.
- **Assignment step**:
  - Assign each data point to closest cluster center.

$$C_i \leftarrow \min_j \left\| \overrightarrow{x_i} - \overrightarrow{\mu_j} \right\|^2$$

- **Recenter Step**:
  - μ_j becomes **centroid/mean** of $\overrightarrow{x_i}$ assigned to j.

$$\mu_j \leftarrow \frac{\sum_{if\ a_{ij}=1} \overrightarrow{x_i}}{\sum_{i=1}^{N} a_{ij}}$$



K-Means Algorithm: Formalization

K-Means Algorithm

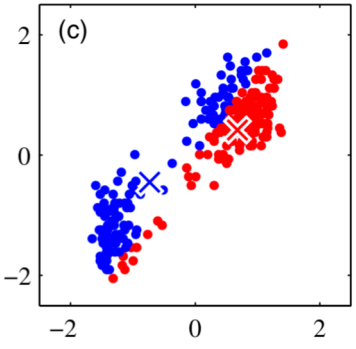
- Let's **formalize** the k-means algorithm.
- We need to understand:
  - What this algorithm is really doing?
  - Does it converge?

We present the k-Means algorithm as an **optimization algorithm**.

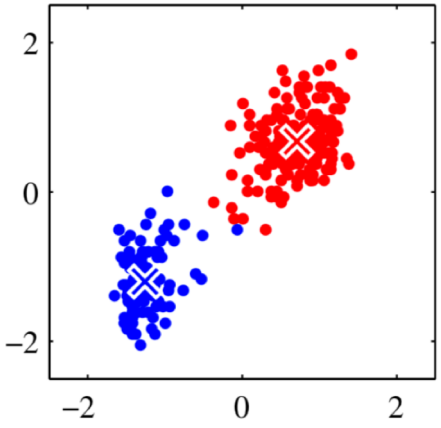
But what does it optimize?

Let's define the **optimization function**.

K-Means Algorithm



- The **objective function J** is defined as follows.
- It represents the **sum of the squares of the distances** of **each data point to its assigned vector μ_j** (cluster center).



$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \overrightarrow{x_i} - \overrightarrow{\mu_j} \parallel^2$$

- No. of data points
- No. of clusters
- Data i
- Centroid for cluster j

K-Means Algorithm

- Our goal is to **find values** for the $\{a_{ij}\}$ and the $\{\mu_j\}$ so as to **minimize J**.
- $J \triangleq \min_{\vec{\mu}_1,..,\vec{\mu}_k} \min_{\vec{a}_1,..,\vec{a}_N} F(\vec{\mu}, \vec{a}) = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$$

No. of data points

No. of clusters

Data i

Centroid for cluster j

J is known as the **distortion measure**

We "distort" the data by **replacing them with their cluster centers**.

K-Means Algorithm

- The **optimization is very hard**: $J \triangleq \min_{\vec{\mu}_1,..,\vec{\mu}_k} \min_{\vec{a}_1,..,\vec{a}_N} F(\vec{\mu}, \vec{a})$
- Because it's an optimization on the **mixture of continuous ($\vec{\mu}$) and discrete ($\vec{a}$) variables**.

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$$

We use a **coordinate descent algorithm**

Coordinate Descent Algorithm

- **Goal**: $\min_a \min_b F(a, b)$
- **Coordinate Descent**:
  - Fix a, minimize b
  - Fix b, minimize a
- Repeat

$$J \triangleq \min_{\vec{\mu}_1,..,\vec{\mu}_k} \min_{\vec{a}_1,..,\vec{a}_N} F(\vec{\mu}, \vec{a})$$

Coordinate descent algorithm **converges** to a local optimum if **F is bounded**

We will use **coordinate descent** algorithm to optimize the k-Means **cost function J**

K-Means Algorithm

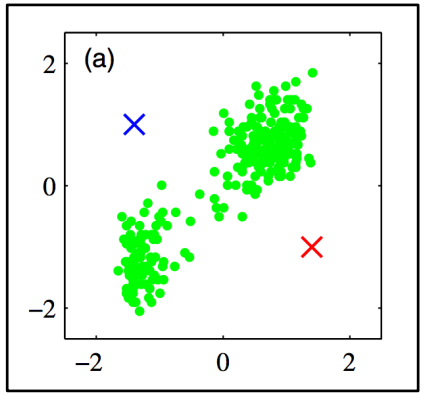
- As mentioned, the coordinate descent is an **iterative procedure**.
- Each iteration involves **two successive steps** corresponding to successive optimizations with respect to the a_{ij} and the μ_j.

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \overrightarrow{x_i} - \overrightarrow{\mu_j} \parallel^2$$

The two successive steps are represented by **Expectation-Maximization (EM)** algorithm.

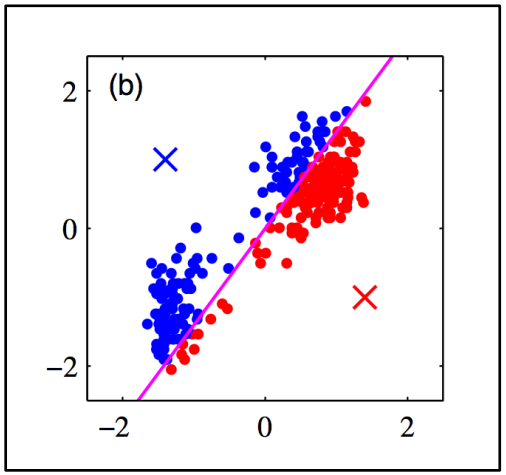
K-Means Algorithm

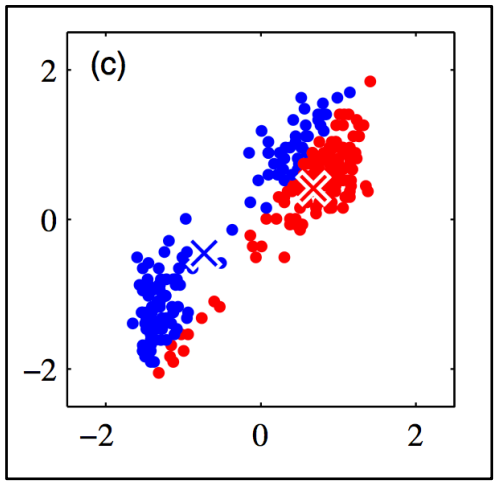
$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$$



- First we choose some **initial values** for the μ_j.

Then in the first phase we **minimize J with respect to the a_{ij}, keeping the μ_j fixed**. (Expectation step or "E" step)



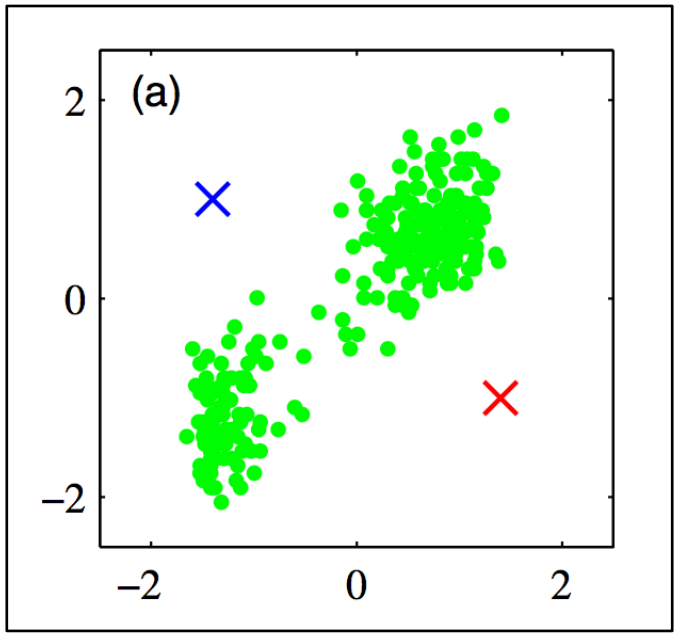


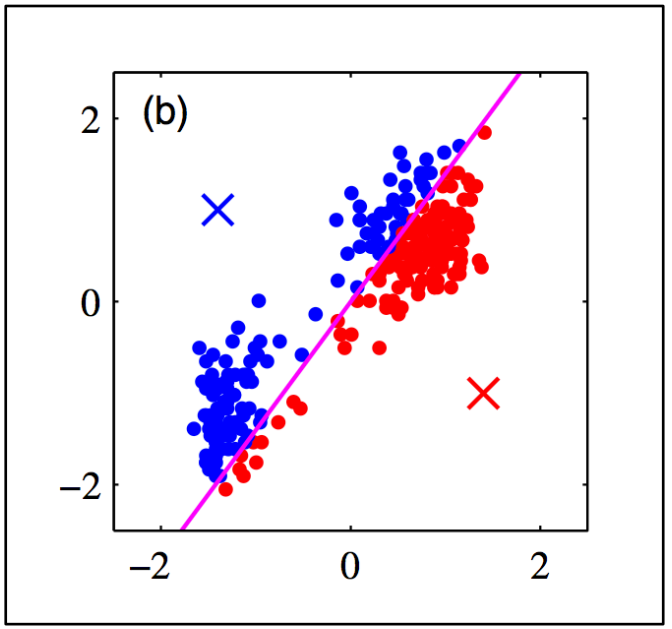
In the second phase we minimize **J with respect to the μ_j, keeping a_{ij} fixed**. (Maximization step or "M" step)

This **two-stage optimization** is then repeated until convergence.

“E” Step

- (a) The **initial choices for centers** μ_1 and μ_2 are shown by the red and blue crosses, respectively.
- (b) In the **initial E step**, each data point is assigned either to the red cluster or to the blue cluster, according to **which cluster center is nearer**.

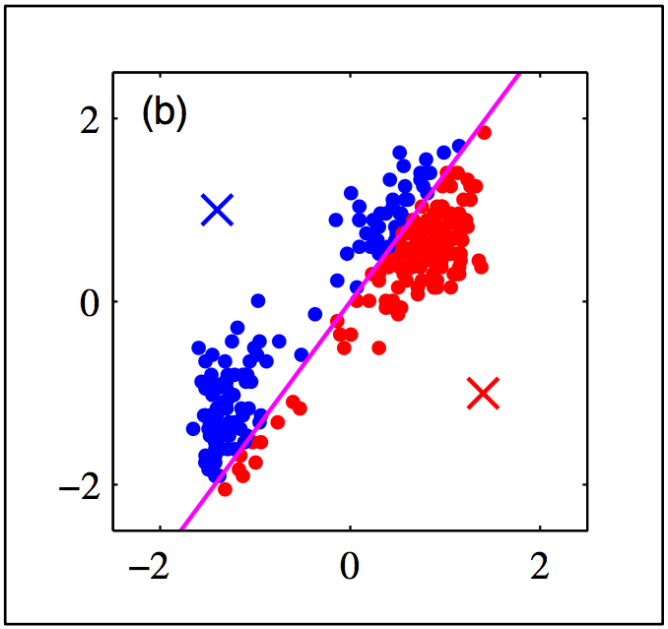




"E" Step

- "E" Step:
- The terms involving different *n* are **independent**.
- So we can **optimize for each *n* separately** by choosing a_{ij} to be 1 for **whichever value of k gives the minimum value** of $||x_i - \mu_j||^2$.

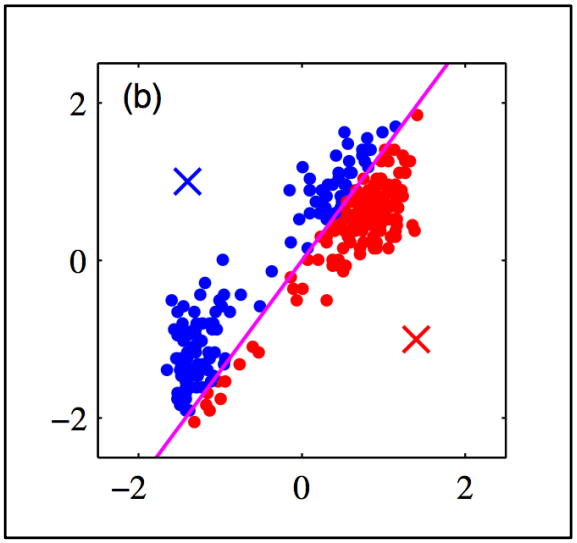
$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \| \vec{x_i} - \vec{\mu_j} \|^2$$



"E" Step

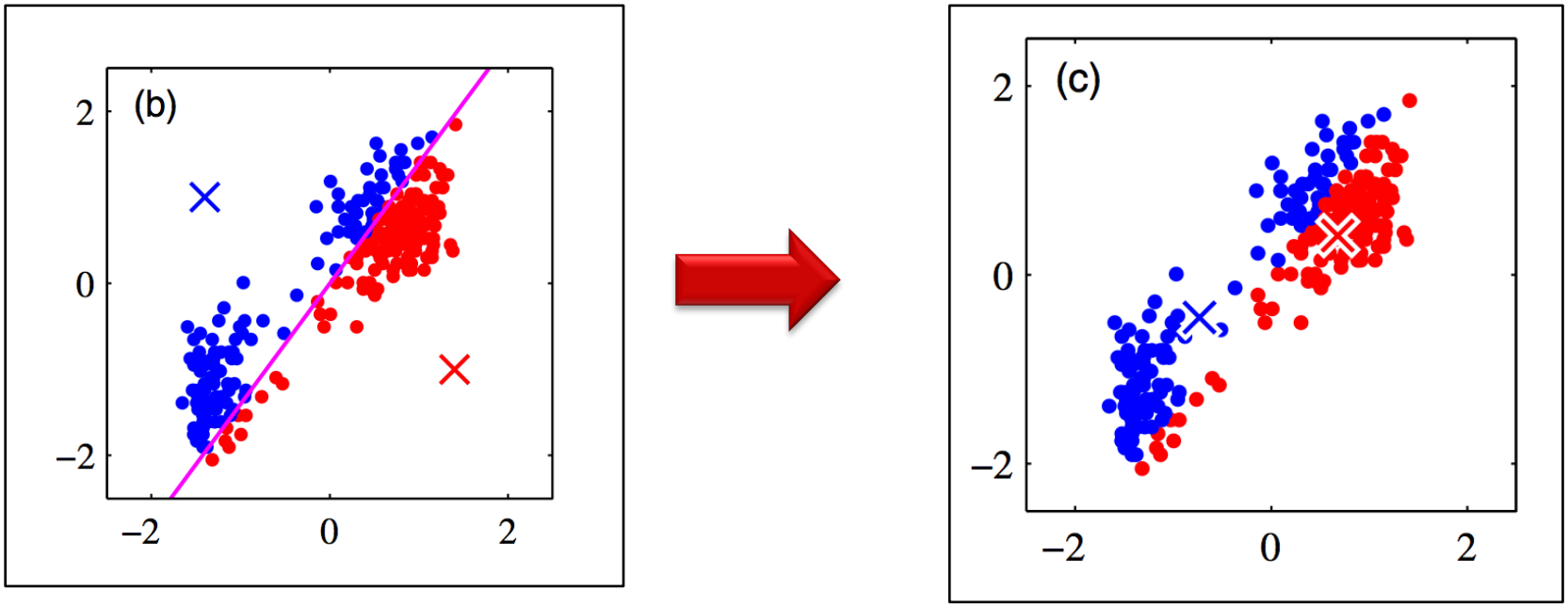
- In other words, we simply assign the ***i***th data point to the **closest cluster center**.
- More formally, this can be expressed as:

$$a_{ij} = \begin{cases} 1 & if\ j = \ argmin_l \ \| \overrightarrow{x_i} - \overrightarrow{\mu_j} \|^2 \\ 0 & otherwise \end{cases}$$



"M" Step

- "M" Step:
- Now consider the **optimization of the μ_j** with the a_{ij} held fixed.
- Each **cluster center is re-computed** to be the **mean of the points** assigned to the corresponding cluster.



"M" Step

- "M" Step: the **optimization of the μ_j** with the a_{ij} held fixed.
- The objective function J is a **quadratic function of μ_j**.
- Hence, it can be minimized by setting its **derivative with respect to μ_j to zero** giving:

$$J = \sum_{i=1}^{N}\sum_{j=1}^{k} a_{ij} \| \vec{x_i} - \vec{\mu_j} \|^2$$

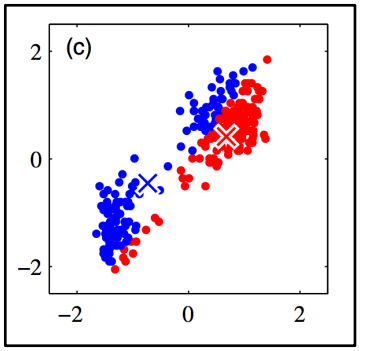
Derivative of J wrt μ

$$2\sum_{i=1}^{N} a_{ij}(x_i - \mu_j) = 0$$

We can easily **solve for μ_j** to give

$$\mu_j = \frac{\sum_i a_{ij} x_i}{\sum_i a_{ij}}$$

"M" Step



- The **denominator** in this expression is **equal to the number of points assigned to cluster j**.
- So this result has a simple interpretation.
- Set μ_j equal to the **mean of all of the data points x_i assigned to cluster j**.
- For this reason, the procedure is known as the **k-means algorithm**.

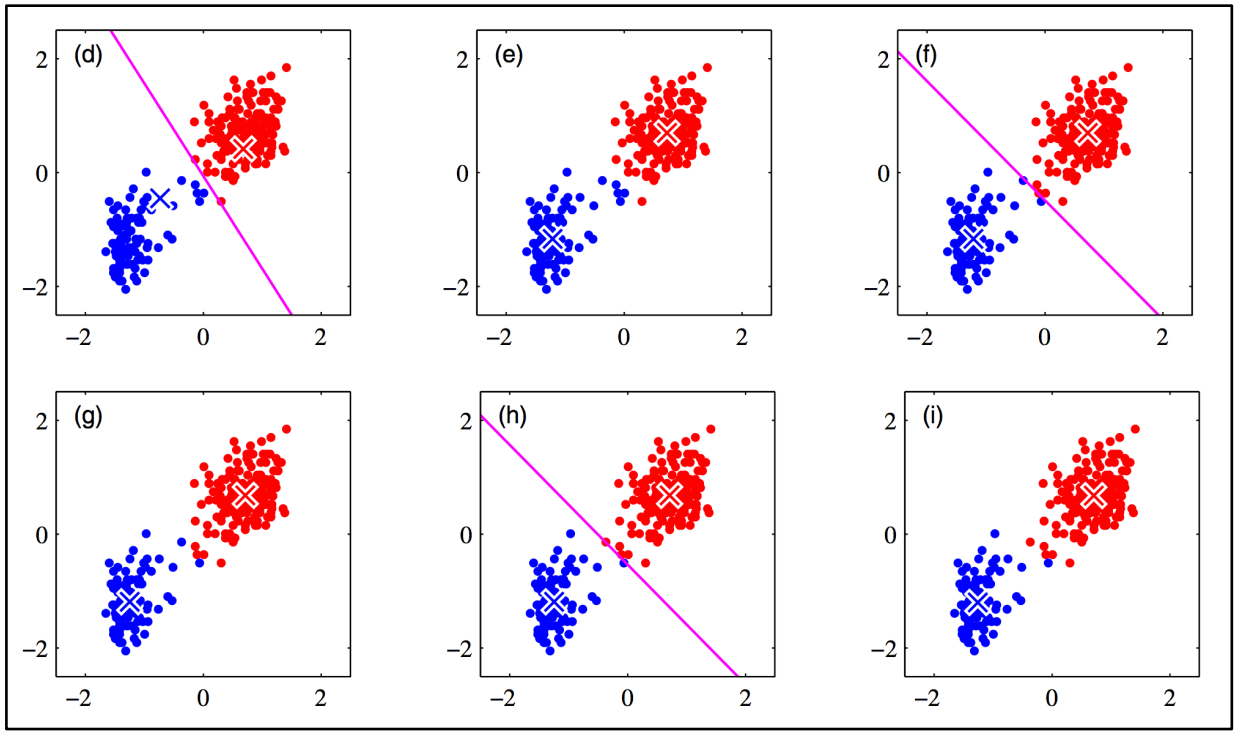
$$\mu_j = \frac{\sum_i a_{ij} x_i}{\sum_i a_{ij}}$$

Mean of all data points x_i assigned to cluster j

No. of data points assigned to cluster j

K-Means Algorithm

- The two phases of **re-assigning data points to clusters** and **re-computing the cluster means** are repeated in turn until there is no further change in the assignments.
- Or until some *maximum number of iterations* is exceeded.



K-Means Algorithm: Convergence Issue

K-Means

- The k-means algorithm **must converge** after a finite number of iterations.
- Why?
- Since both the E-step and the M-step minimize the distortion measure, the algorithm will **never change from a particular assignment of data points to prototypes**, unless the new assignment has a *lower value* for J.

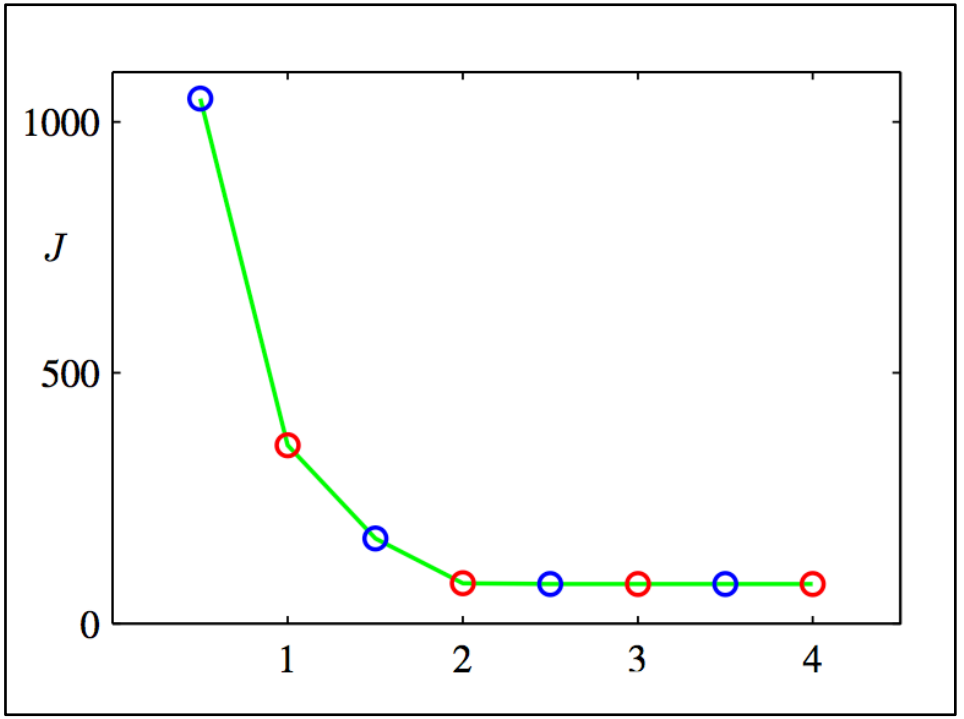
$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$$

K-Means

- In the following example J is represented after each E step (blue points) and M step (red points) of the k-means algorithm.
- The algorithm has **converged after the 3rd M step**.

The **final EM cycle produces no changes** in either the assignments or the prototype vectors.

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \| \vec{x_i} - \vec{\mu_j} \|^2$$



K-Means

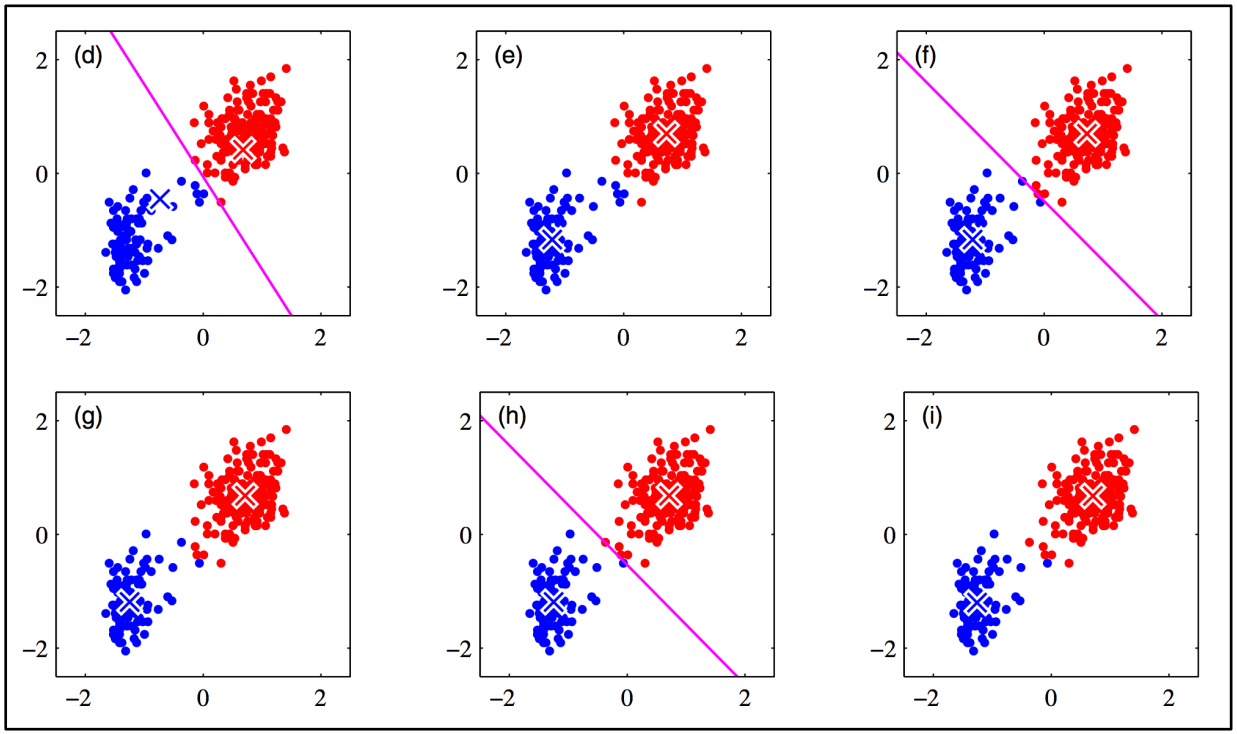
- There is a **finite number of possible assignments**, each with a corresponding unique minimum of J w.r.t. the prototypes, $\{\mu_j\}$.
- Thus, the k-means algorithm will **converge after a finite number of steps**, when no re-assignment of data points to prototypes will result in a decrease of J.
- When no-reassignment takes place, there also **will not be any change in $\{\mu_j\}$**.

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} a_{ij} \parallel \vec{x_i} - \vec{\mu_j} \parallel^2$$

K-Means Algorithm

- Thus, because each phase reduces the value of the objective function J, **convergence of the algorithm is assured**.

However, it may **converge to a local** rather than global minimum of J.



K-Means Algorithm: Practical Issues

K-Means: Practical Issues

- There are **three practical issues** related to the convergence of the k-Means algorithm.
  - Varity of Distance Measures
  - Data Standardization
  - Convergence to the Global Minimum