Machine Learning by ambedkar@IISc

- Unsupervised Learning
- Dimensionality Reduction
- K-means Clustering

Agenda

What is Unsupervised Learning

Principle Component Analysis and Dimensionality Detection

Clustering

What is Unsupervised Learning

Unsupervised Learning

- **Input:** A set of unlabeled examples, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}$
- **Objective:** Find *patterns* in observed data
- **Challenge:** Since there is no ground-truth or labels it is very difficult to evaluate the algorithms.

Unsupervised Learning

- **Examples:**
  - *Clustering* - Grouping observed data into unlabeled *clusters*
    - identifying social circles, summarizing observed data etc.
  - *Dimensionality Reduction* - Finding a low-dimensional representation of the data
    - visualization, compression, structure analysis etc.
  - *Anomaly Detection* - Spotting outliers in the data
    - detecting fraudulent transactions, data cleaning etc.[1]
  - *Density Estimation* - Finding the underlying probability distribution from which $\mathcal{D}$ has been sampled.

[1] The discovery of Higgs Boson relied on one such algorithm

Principle Component Analysis and Dimensionality Detection

Dimensionality Reduction

- **Input:** A dataset $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$ where each $\mathbf{x}_n \in \mathbb{R}^d$
- **Objective:** Find a low-dimensional representation of each point $\tilde{\mathbf{x}}_n \in \mathbb{R}^k$ where $k < d$
- **In other words:** Find a k-dimensional coordinate system and represent all the points in this coordinate system
  - Need to find orthonormal vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K$ which form the basis of the new coordinate system
  - Need to way to represent the original points in this new coordinate system
- **Main Question:** How to choose the low dimensional space and embed the points in it?

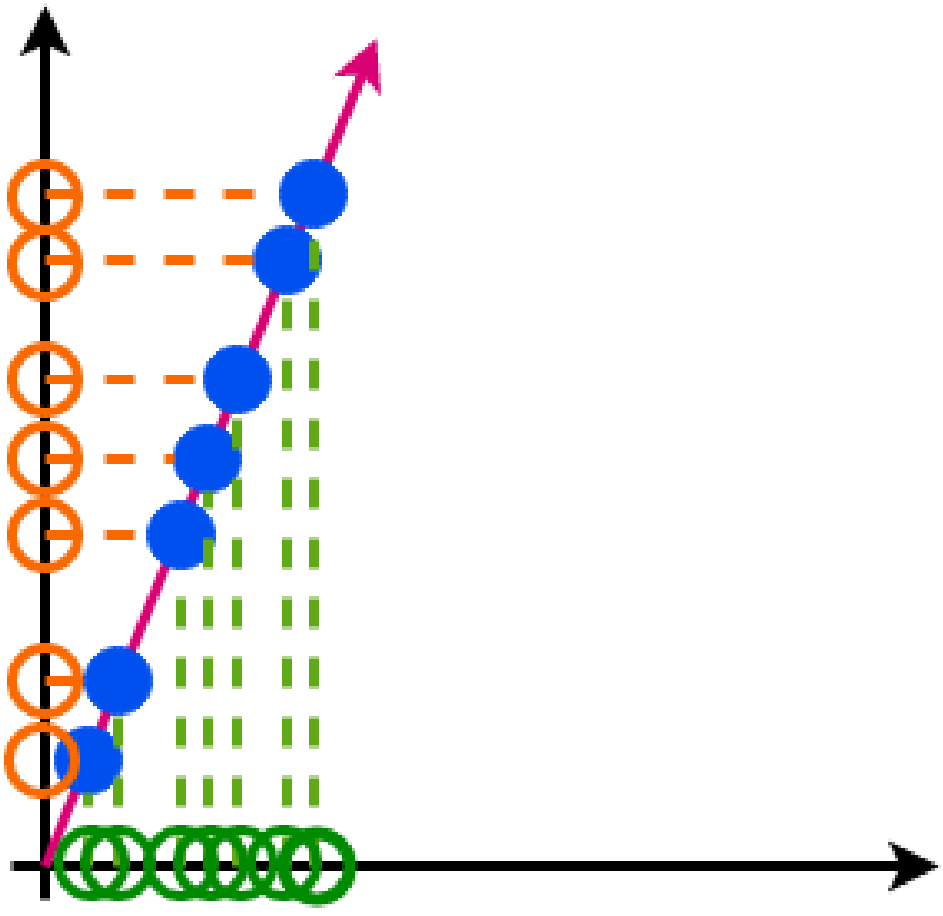
Dimensionality Reduction - Applications

- **Visualization:** Find a 2 or 3 dimensional representation of data such that the essence of data is not lost
  - Visualizing financial profile of individuals in two dimensions to identify patterns
- **Compression:** Embed the points in a lower dimensional space such that various topological properties are preserved to optimize storage
  - Minimizing the number of colours needed to represent an image. Efficient encoding schemes can then be used for compression
- **Feature Selection:** Remove redundant or less informative features
  - Identifying and eliminating functionally related or highly correlated features like density, mass and volume

Dimensionality Reduction - Toy Example

- Given 7 points in two dimensions. Need 14 numbers to store x and y coordinates of all points
- **Idea 1:** Discard y coordinate of all points (green points). Only 7 numbers needed now. Lot of information lost.
- **Idea 2:** Discard x coordinate of all points (orange points). Only 7 numbers needed now. Better than green points.
- **Idea 3:** Save the slope of pink line and the x (or y) coordinate of each point. Need to store 8 numbers. No information lost.

Dimensionality Reduction - Toy Example



Dimensionality Reduction - Toy Example - Findings

- Simply discarding coordinates is not a good idea
- Not all ways of dimensionality reduction are equally good
- Need to quantify the amount of information lost while performing dimensionality reduction
- Real data is not as neat as the toy example, need a way to deal with noise

- **Revised Objective:** To find a k dimensional subspace of $\mathbb{R}^d$ and linearly project data onto this subspace while minimizing the "loss of information"
  - Non-linear dimensionality reduction methods exist but are beyond the current scope
  - We will consider Principle Component Analysis (PCA)

Dimensionality Reduction - Principle Component Analysis

- Let $\mathbf{u} \in \mathbb{R}^d$ be a direction along which we want to project data
- Thus, $\tilde{\mathbf{x}}_n = (\mathbf{x}_n{}^\intercal \mathbf{u})\mathbf{u}$. Note that one only needs to store $\mathbf{x}_n{}^\intercal \mathbf{u}$ for each n
- PCA uses variance in projected data as a measure of information
  - Information content is assumed to be proportional to variance of projected data
  - Need to retain maximum information, thus, need to find $\mathbf{u}$ such that variance of projected data is maximized

$$\mathbf{u}^* = \arg \max_{\mathbf{u}:||\mathbf{u}||=1} \mathrm{Var}(\{\tilde{\mathbf{x}}_n\}_{n=1}^N)$$

Dimensionality Reduction - PCA (contd. . .)

$$\mathrm{Var}(\{\tilde{\mathbf{x}}_n\}_{n=1}^N) = \frac{1}{N}\sum_{n=1}^{N}\left(\tilde{\mathbf{x}}_n^2 - \mathrm{E}[\tilde{\mathbf{x}}_n]^2\right)$$

- Assume WLOG that $\mathrm{E}[\mathbf{x}_n] = \mathbf{0}$, thus $\mathrm{E}[\tilde{\mathbf{x}}_n] = \mathrm{E}[\mathbf{x}_n]^\intercal \mathbf{u} = 0$
- Also, $\tilde{\mathbf{x}}_n^2 = \mathbf{u}^\intercal \mathbf{x}_n \mathbf{x}_n^\intercal \mathbf{u}$, thus we get:

$$\sum_{n=1}^{N}\left(\tilde{\mathbf{x}}_n^2 - \mathrm{E}[\tilde{\mathbf{x}}_n]^2\right) = \mathbf{u}^\intercal\left(\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^\intercal\right)\mathbf{u}$$

- Note that $\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^\intercal$ is the covariance matrix $\mathbf{X}$ of observed data since $\mathrm{E}[\mathbf{x}_n] = \mathbf{0}$. Thus:

$$\mathrm{Var}(\{\tilde{\mathbf{x}}_n\}_{n=1}^N) = \frac{1}{N}\mathbf{u}^\intercal\mathbf{X}\mathbf{u}$$

Dimensionality Reduction - PCA (contd. . .)

- The constant can be dropped for the purpose of optimization. Hence the optimization problem becomes:

$$\mathbf{u}^* = \arg \max_{\mathbf{u}:||\mathbf{u}||=1} \mathbf{u}^\intercal \mathbf{X} \mathbf{u}$$

- This is a constrained optimization problem, the Lagrangian is given by:

$$\mathcal{L}(\mathbf{u}, \mu) = \mathbf{u}^\intercal \mathbf{X} \mathbf{u} + \mu(\mathbf{u}^\intercal \mathbf{u} - 1)$$

$$\nabla_{\mathbf{u}} \mathcal{L} = 0 \Rightarrow 2\mathbf{X}\mathbf{u} + 2\mu\mathbf{u} = 0$$

$$\Rightarrow \mathbf{X}\mathbf{u} = -\mu\mathbf{u}$$

- Thus, the optimal $\mathbf{u}$ must be an eigenvector of $\mathbf{X}$. Since we want to maximize $\mathbf{u}^\intercal \mathbf{X} \mathbf{u}$, $\mathbf{u}$ must be the eigenvector corresponding to largest eigenvalue. Hence:

$\mathbf{u}^*$ = eigenvector of $\mathbf{X}$ corresponding to largest eigenvalue

Dimensionality Reduction - PCA (contd. . .)

- Usually $k > 1$ thus we want to find $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k$ and not just $\mathbf{u}^*$
- Setting $\mathbf{u}_1 = \mathbf{u}^*$, one can find $\mathbf{u}_2$ as follows:

$$\mathbf{u}_2 = \arg\max_{\mathbf{u}:||\mathbf{u}||=1, \mathbf{u}^\intercal \mathbf{u}_1 = 0} \mathbf{u}^\intercal \mathbf{X} \mathbf{u}$$

- $\mathbf{u}^\intercal \mathbf{u}_1 = 0$ is needed to avoid correlations in projected data
- One can show that $\mathbf{u}_2$ is the eigenvector of $\mathbf{X}$ corresponding to second largest eigenvalue
- Similarly $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k$ are the eigenvectors of $\mathbf{X}$ corresponding to k largest eigenvalues. Also:

$$\tilde{\mathbf{x}}_n = \mathbf{U}^\intercal \mathbf{x}_n$$

where, $\mathbf{U} \in \mathbb{R}^{d \times k}$ is a matrix containing $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k$ in its columns

Dimensionality Reduction - PCA (contd. . .)

Algorithm 1 Principle Component Analysis

Input: Dataset $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$ and number of dimensions k

Output: Low dimensional vectors $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_n\}_{n=1}^N$

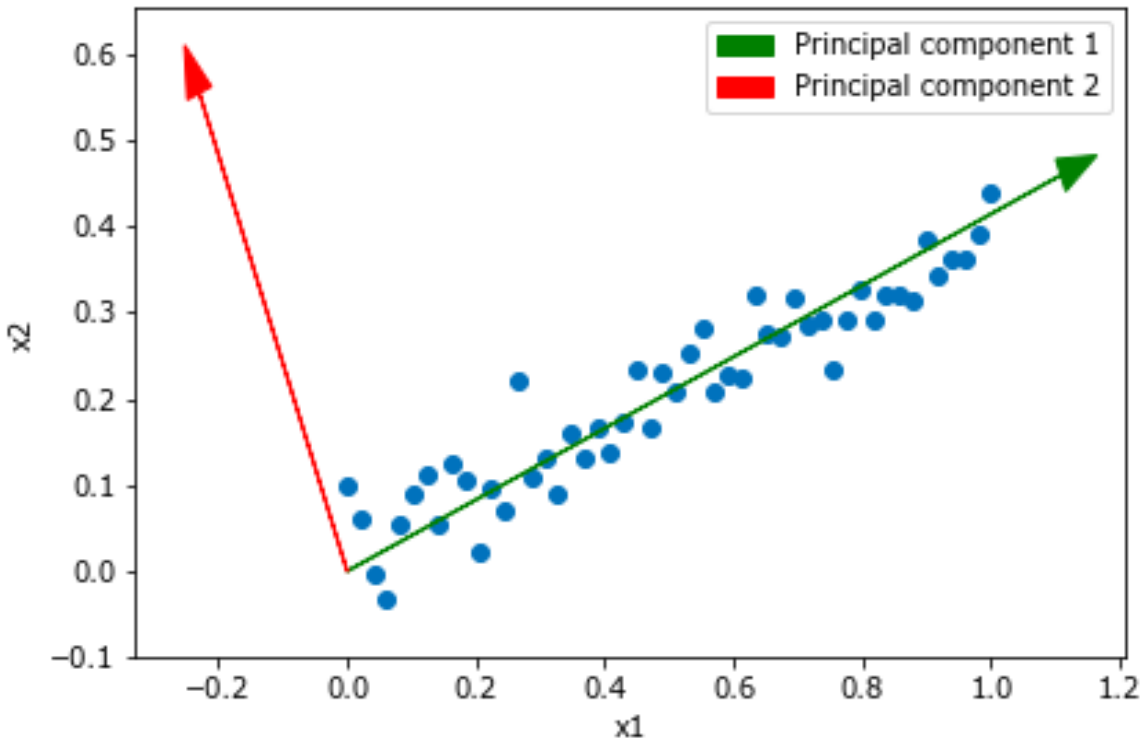
Normalize the data so that it is zero mean

Compute $\mathbf{X} = \sum_{n=1}^N \mathbf{x}_n \mathbf{x}_n^\intercal$

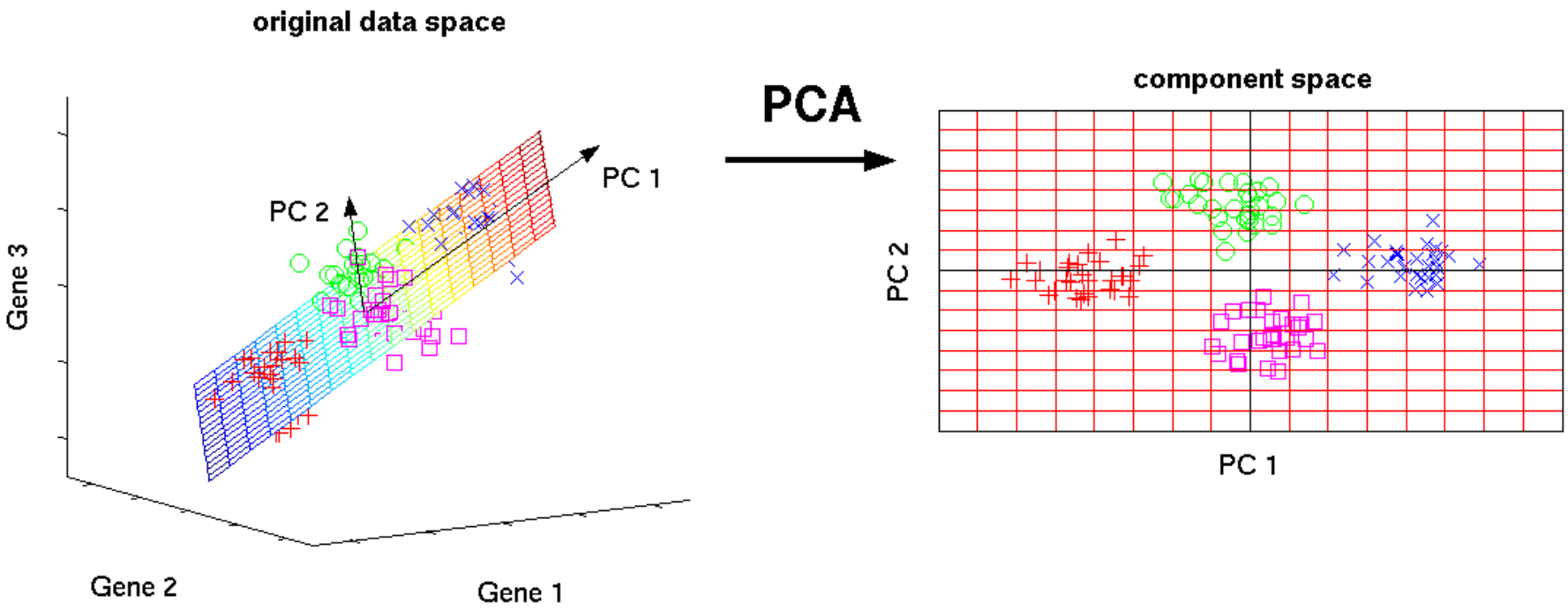
Find $\mathbf{U} \in \mathbb{R}^{d \times k}$ containing top k eigenvectors of $\mathbf{X}$ as columns

Compute $\tilde{\mathbf{x}}_n \in \mathbb{R}^k$ such that $\tilde{\mathbf{x}}_n = \mathbf{U}^\intercal \mathbf{x}_n$, for all $n = 1, \ldots, N$

PCA Example


Principal component 1
Principal component 2
x1
x2
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

PCA Example


original data space
PC 1
PC 2
Gene 3
Gene 2
Gene 1
PCA
component space
PC 2
PC 1

Clustering

Clustering

- **Input:** Data points $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$, a similarity/distance function $d(.,\ .)$ defined on elements of $\mathcal{D}$ and the number of clusters K
- **Objective:** *Partition* the given N points into K subsets $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$ such that:
  - $\mathbf{C}_k \subset \mathcal{D}$, $\mathbf{C}_j \neq \Phi$ for all $k = 1, \ldots, K$
  - $\mathbf{C}_i \cap \mathbf{C}_j = \Phi$ for all $i, j = 1, \ldots, k$, $i \neq j$
  - $\cup_{k=1}^K \mathbf{C}_j = \mathcal{D}$
  - Points in the same cluster are more similar than points across clusters (w.r.t. $d(.,\ .)$)
- Variants that allow fractional membership of points to clusters or overlapping clusters exist but we will assume that each point belongs to exactly one cluster

Clustering (contd. . .)

	$\mathbf{x}^{(i)}$	$d(.,\ .)$	**Clusters**
Eye Gaze Tracker	(x, y) coordinate on screen where user is looking	Euclidean distance	Hot-spots on screen
Social Media	A binary vector indicating friends of person i	$\frac{1}{\#\text{common friends}}$	Friendship groups
Documents	Bag of words representation of document i	$\frac{1}{\#\text{common words}}$	Topics
Biology	Genes	Task dependent	Gene expression patterns

Table 1: Some examples related to clustering

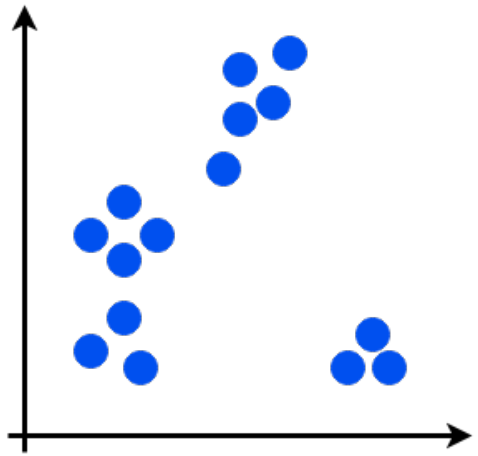
Clustering - Approaches

- **Agglomerative** (bottom-up) vs **Divisive** (top-down)
- **Monothetic** (considers features sequentially) vs **Polythetic** (considers features all at once)
- **Hard** (single cluster membership) vs **Fuzzy** (mixed memberships allowed)
- **Hierarchical** (creates hierarchy) vs **Partitional** (disjoint, unordered clusters)

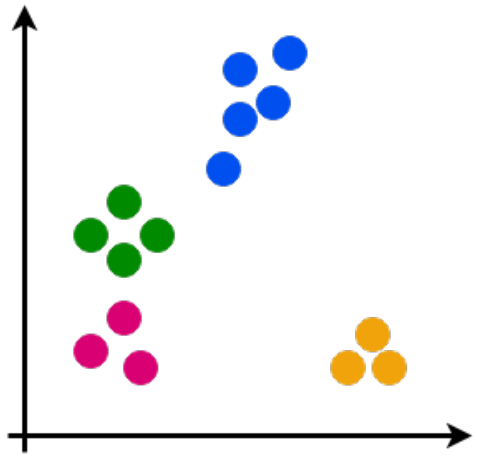
Any clustering algorithm can be classified based on this scheme

Example: We will see that k-Means is a polythetic, hard and partitional clustering algorithm

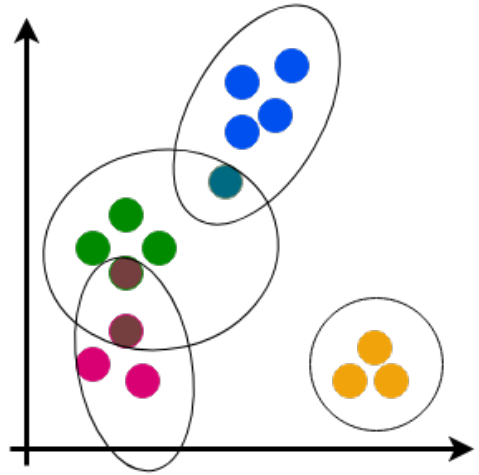
Clustering - Toy Example



Data



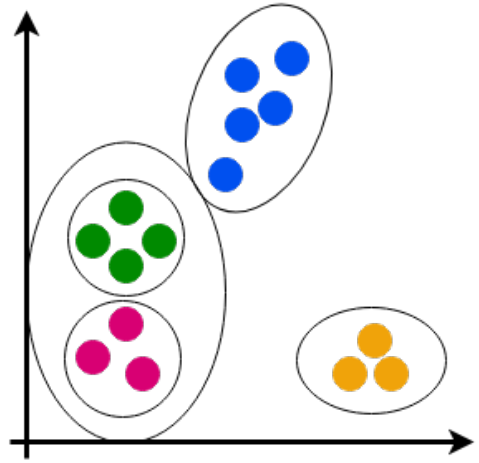
Hard Clustering



Fuzzy Clustering

Clustering is an exploratory data analysis problem.

There is no single "correct" solution.



Hierarchical Clustering

Clustering - Popular Algorithms

- k-Means and k-Medoids
- Spectral clustering
- Expectation Maximization for Gaussian Mixture Models
- Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
- etc.

Clustering - k-Means

- Let $\mathcal{C}$ denote the set of all possible cluster assignment for the given dataset $\mathcal{D}$
- $\mathbf{c} \in \mathcal{C}$ is such that $\mathbf{c} \in \{1, \ldots, k\}^m$, where $\mathbf{c}_i = j$ iff $\mathbf{x}^{(i)} \in \mathbf{C}_j$. Recall that:
  - k is the number of clusters
  - m is the number of data points
  - $\mathbf{C}_j$ is the j^{th} cluster
- Ideally one would like to solve the following problem:

$$\mathbf{c}^* = \arg\min_{\mathbf{c} \in \mathcal{C}} \sum_{j=1}^{k} \sum_{i_1, i_2 = 1}^{m} \mathbf{1}\{\mathbf{c}_{i_1} = j, \mathbf{c}_{i_2} = j\} ||\mathbf{x}^{(i_1)} - \mathbf{x}^{(i_2)}||^2,$$

  i.e. minimize the distance between points in same cluster
- This optimization is NP hard so k-Means clustering solves a relaxed version of this problem

Clustering - k-Means (contd. . .)

Algorithm 2 k-Means Clustering Algorithm

Input: Dataset $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{m}$ and number of clusters k

Output: Cluster assignment vector $\mathbf{c} \in \{1, \ldots k\}^m$, cluster centers $\mu_1, \ldots, \mu_k$

Initialize $\mu_1, \ldots, \mu_k$ by randomly choosing k distinct points from $\mathcal{D}$

repeat

Set $\mathbf{c}_i = \arg\min_j ||\mathbf{x}^{(i)} - \mu_j||^2$ for $i = 1, 2, \ldots, m$

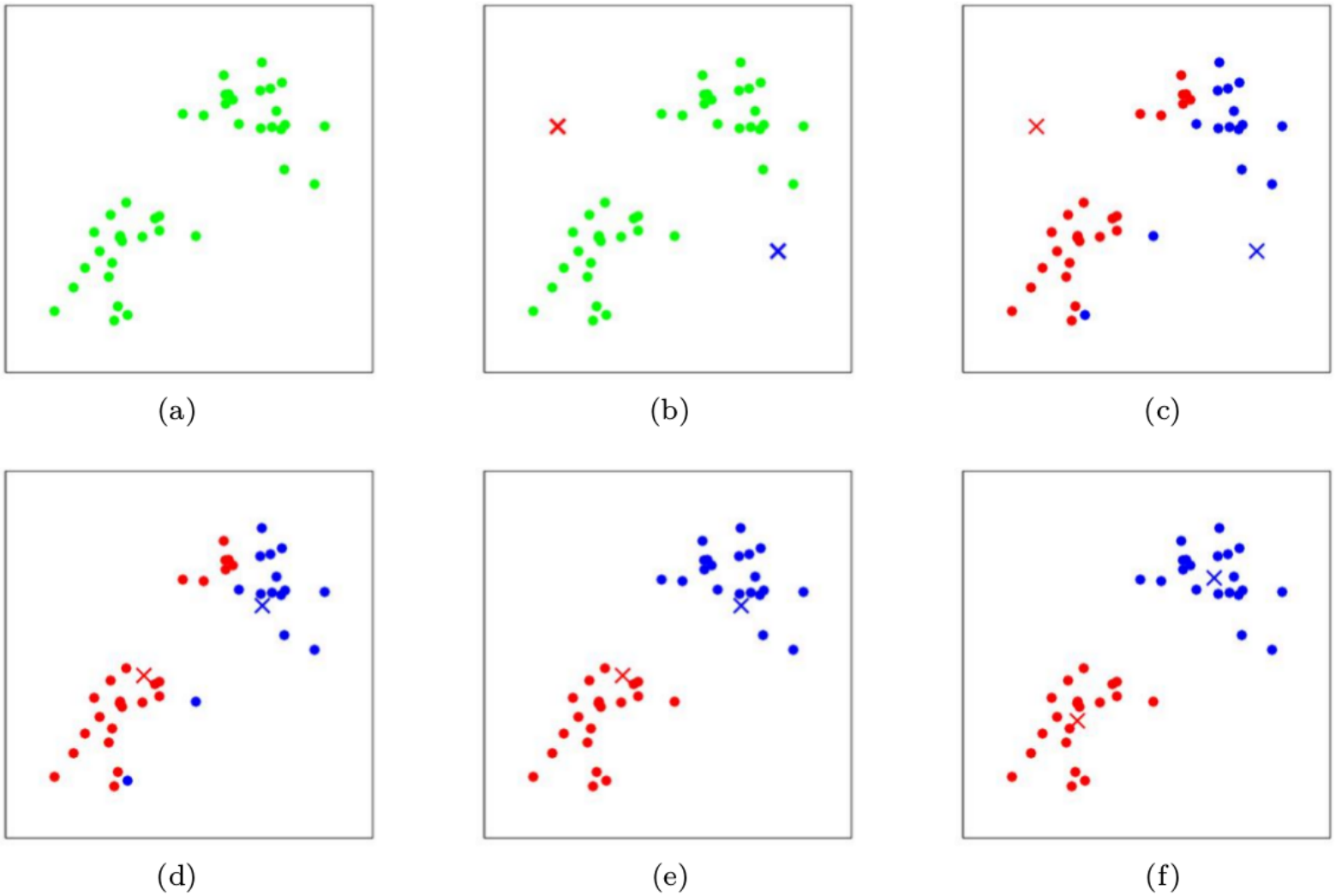
Set $\mu_j = \frac{1}{|\{i:\mathbf{c}_i=j\}|} \sum_{i=1}^{m} \mathbf{1}\{\mathbf{c}_i = j\}\mathbf{x}^{(i)}$ for all $j = 1, 2, \ldots, k$

until convergence

Breaks the optimization problem into two parts

- Optimization over memberships $\mathbf{c}$ keeping $\mu_1, \ldots, \mu_k$ fixed
- Optimization over cluster centers $\mu_1, \ldots, \mu_k$ keeping $\mathbf{c}$ fixed

Clustering - k-Means (contd. . .)



k-Means on a toy dataset[2]

[2] Image Source: Andrew Ng, CS-229 Lecture Notes

Clustering - k-Means (contd. . .)

Limitations of k-Means:

- Not suitable for non-spherical clusters because of the use of Euclidean distance
  - Transform data appropriately before performing k-Means (as we will see later for spectral clustering) or use kernel k-Means
- Not robust to outliers because of the use of arithmetic mean
  - Remove outliers before clustering
- Susceptible to sub-optimal solutions
  - Run the algorithm multiple times with random initializations