CSE 151 Machine Learning
Instructor: Kamalika Chaudhuri
Projecting Data

How to find the subspace $S$ to project to?

One option: PCA subspace
Properties of Symmetric $A$

$$\max_{x \in \mathbb{R}^n} x^\top A x$$

Such that: $\|x\| = 1$

**Solution:** $x = x_1$ (eigenvector for the largest eigenvalue)

maximum value of $x^\top A x$ is the largest eigenvalue

**Question:** If $x$ is an eigenvector with eigenvalue $c$, what is $x^\top A x$?
Covariance Matrix

Suppose $x_1, \ldots, x_n$ are data points in $\mathbb{R}^d$

Covariance matrix of $x_1, \ldots, x_n$:

$$\Sigma = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^\top$$

where:

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$$
How to find a good subspace?

Data lies almost on a line. Find the direction of maximum variance.
Max Variance Direction

Suppose data in $\mathbb{R}^d$ is centered so that the mean is 0

Variance along a direction $u$ is:

$$
\frac{1}{n} \sum_{i=1}^{n} (x_i, u)^2 = \frac{1}{n} \sum_{i=1}^{n} (u^T x_i)^2 = \frac{1}{n} \sum_{i=1}^{n} u^T x_i x_i^T u = \frac{1}{n} u^T \left( \sum_{i} x_i x_i^T \right) u
$$

Recall that: $A = \sum_{i} x_i x_i^T$ is symmetric

Finding the max. variance direction is equivalent to finding the top eigenvector of $A$. Similar fact can be shown for the $k$-dimensional subspace of max variance.
Dimension Reduction Algorithm

Given data matrix $X_{n \times d}$

1. Compute mean $m$ of the data vectors

2. Compute covariance matrix:

$$
\Sigma = \frac{1}{n} \sum_{i=1}^{n} (x_i - m)(x_i - m)^\top
$$

3. Compute $U_{k \times d}$, the top $k$ eigenvectors of $\Sigma$

4. Project:

$$
P(X) = XU^\top
$$
Example 1

Eigenvalues: 5.49, 1.77
Example 2

Iris data set: 4 features, 3 labels

\[
\text{Cov} = \begin{bmatrix}
0.6811 & -0.0390 & 1.2652 & 0.5135 \\
-0.0390 & 0.1868 & -0.3196 & -0.1172 \\
1.2652 & -0.3196 & 3.0924 & 1.2877 \\
0.5135 & -0.1172 & 1.2877 & 0.5785
\end{bmatrix}
\]

Eigenvalues:
\[ [4.1967, 0.2406, 0.0780, 0.0235] \]

Projections to top 2 PCA subspace
Clustering

Group the data into a number of groups or clusters, based on inherent similarity structure in the data
k-means Algorithm

**Input:** Data $x_1,..,x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ cluster centers $c_1,..,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
k-means Algorithm

**Input:** Data $x_1,..,x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ cluster centers $c_1,..,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
k-means Algorithm

**Input:** Data $x_1,..,x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ cluster centers $c_1,..,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
**k-means Algorithm**

**Input:** Data $x_1, \ldots, x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ cluster centers $c_1, \ldots, c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
**k-means Algorithm**

**Input:** Data \(x_1, \ldots, x_n\), \(k = \#\text{clusters}\)

1. Randomly guess \(k\) cluster centers \(c_1, \ldots, c_k\)

2. Repeat until convergence:
   (a) Assign each \(x_i\) to its closest center. This forms \(k\) clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
**k-means Algorithm**

**Input:** Data $x_1,..,x_n$, $k$ = #clusters

1. Randomly guess $k$ cluster centers $c_1,..,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
**k-means Algorithm**

**Input:** Data $x_1,..,x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ cluster centers $c_1,..,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
**k-means Algorithm**

**Input:** Data $x_1,...,x_n$, $k = \#clusters$

1. Randomly guess $k$ cluster centers $c_1,...,c_k$

2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center. This forms $k$ clusters.
   (b) Update the centers of each cluster as the mean of the data points in the cluster.
k-means Algorithm - Example 2

Picture thanks to Michael Jordan
k-means Distortion Function

\[
J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2
\]

1. Data points \(x_1, \ldots, x_n\), Centers \(c_1, \ldots, c_k\)

2. \(c_{x_i}\) is the center assigned to \(x_i\)

Each iteration of the k-means algorithm decreases the distortion function. Why?
k-means Distortion Function

**Input:** Data $x_1, \ldots, x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ centers $c_1, \ldots, c_k$
2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center
   (b) Update cluster centers as the mean of data points in the cluster

**Distortion function:**

$$J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2$$
k-means Distortion Function

**Input:** Data $x_1, \ldots, x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ centers $c_1, \ldots, c_k$
2. Repeat until convergence:
   
   (a) Assign each $x_i$ to its closest center
   
   (b) Update cluster centers as the mean of data points in the cluster

**Distortion function:**

$$J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2$$

Given cluster centers $c_1, \ldots, c_k$, (a) decreases the value of each term
**k-means Distortion Function**

**Input:** Data $x_1,...,x_n$, $k = \#\text{clusters}$

1. Randomly guess $k$ centers $c_1,...,c_k$
2. Repeat until convergence:
   (a) Assign each $x_i$ to its closest center
   (b) Update cluster centers as the mean of data points in the cluster

**Distortion function:**

$$J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \|x_i - c_{x_i}\|^2$$

Given clusters $C_j$, cluster assignments for each $x_i$, $J$ can be written as:

$$J = \sum_{j=1}^{k} \sum_{i \in C_j} \|x_i - c_j\|^2 = \sum_{j=1}^{k} \left( \sum_{i \in C_j} \|x_i - \bar{c}_j\|^2 + |C_j| \cdot \|\bar{c}_j - c_j\|^2 \right)$$

where: $\bar{c}_j = \frac{1}{|C_j|} \sum_{i \in C_j} x_i$

Minimum when: $c_j = \bar{c}_j$
**k-means Distortion Function**

\[ J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2 \]

1. Data points \( x_1, \ldots, x_n \), Centers \( c_1, \ldots, c_k \)
2. \( c_{x_i} \) is the center assigned to \( x_i \)

Each iteration of k-means algorithm decreases \( J \)

Does k-means terminate in finite time?
k-means Distortion Function

\[ J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2 \]

1. Data points \( x_1, \ldots, x_n \), Centers \( c_1, \ldots, c_k \)
2. \( c_{x_i} \) is the center assigned to \( x_i \)

Each iteration of k-means algorithm decreases \( J \)

There are a finite number of ways to cluster \( n \) points into \( k \) groups. Since \( J \) keeps decreasing, it will never cycle, and eventually, it will run through all these ways.
k-means Distortion Function

\[ J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2 \]

1. Data points \( x_1, \ldots, x_n \), Centers \( c_1, \ldots, c_k \)
2. \( c_{x_i} \) is the center assigned to \( x_i \)

Each iteration of k-means algorithm decreases \( J \)

Does it always find an optimal clustering?
k-means Distortion Function

\[ J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \| x_i - c_{x_i} \|^2 \]

1. Data points \( x_1, \ldots, x_n \), Centers \( c_1, \ldots, c_k \)
2. \( c_{x_i} \) is the center assigned to \( x_i \)

Each iteration of k-means algorithm decreases \( J \)

Does it always find an optimal clustering?

No, it finds a **local optimum**. Finding the optimum of the distortion function \( J \) is NP Hard.
Example of Suboptimal k-means Solution
How to Improve Convergence?

1. Start with many different initial configurations, and run k-means multiple times. Pick the result that has minimum distortion.

2. Use **furthest-first heuristic** to pick the initial k centers:
   
   - $c_1$ = a randomly drawn data point
   - $c_2$ = data point $x_i$ for which $d(x_i, c_1)$ is maximum
   - $c_3$ = data point $x_i$ for which $\min(d(x_i, c_1), d(x_i, c_2))$ is maximum
How to Improve Convergence?

1. Start with many different initial configurations, and run k-means multiple times. Pick the result that has minimum distortion.

2. Use **furthest-first heuristic** to pick the initial k centers:
   - $c_1$ = a randomly drawn data point
   - $c_2$ = data point $x_i$ for which $d(x_i, c_1)$ is maximum
   - $c_3$ = data point $x_i$ for which $\min(d(x_i, c_1), d(x_i, c_2))$ is maximum
How to Improve Convergence?

1. Start with many different initial configurations, and run k-means multiple times. Pick the result that has minimum distortion

2. Use **furthest-first heuristic** to pick the initial k centers:
   - \( c_1 \) = a randomly drawn data point
   - \( c_2 \) = data point \( x_i \) for which \( d(x_i, c_1) \) is maximum
   - \( c_3 \) = data point \( x_i \) for which \( \min( d(x_i, c_1), d(x_i, c_2) ) \) is maximum
How to Improve Convergence?

1. Start with many different initial configurations, and run k-means multiple times. Pick the result that has minimum distortion.

2. Use **furthest-first heuristic** to pick the initial k centers:
   - \( c_1 \) = a randomly drawn data point
   - \( c_2 \) = data point \( x_i \) for which \( d(x_i, c_1) \) is maximum
   - \( c_3 \) = data point \( x_i \) for which \( \min( d(x_i, c_1), d(x_i, c_2) ) \) is maximum
How to determine k?

Use model selection criterion, such as AIC or BIC

If some labelled data is available, use labelled data to determine k

Use validation data to see if when the distortion function with respect to the validation starts increasing
When does k-means work well?

Good for k-means

Bad for k-means

Black dots: Centers that optimize the distortion function:

\[ J(x_1, \ldots, x_n, c_1, \ldots, c_k) = \sum_{i=1}^{n} \|x_i - c_{x_i}\|^2 \]
In Class Problem

Consider the following data points:

(1, 0), (2, 0), (3, 0), (1, 2), (2, 2), (3, 2)

Starting with the centers (1, 0) and (3, 2), run iterations of k-means on these data points till convergence.