CS2109S: Introduction to AI and Machine Learning

Lecture 8:
Unsupervised Learning

18 March 2025

Last time

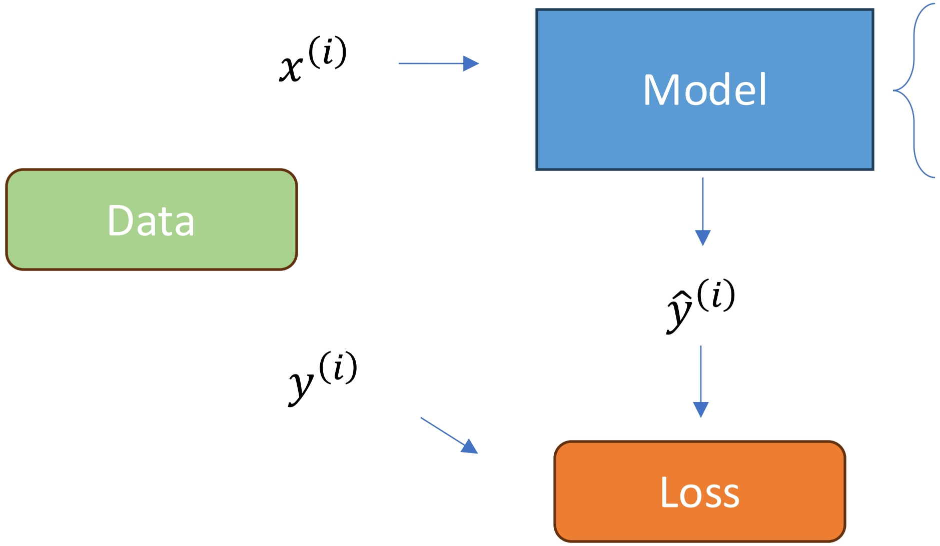
- Regularization
- Kernel method
- Support Vector Machine

Outline

- Unsupervised Learning
- K-means clustering
  - Algorithm
  - Measuring the goodness of clusters
  - Picking the number of clusters
  - Variants
- Dimensionality Reduction
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA)

Supervised Learning

Given a set of N input-output pairs (training samples) $\{(x^{(1)}, y^{(1)}), \dots, (x^{(N)}, y^{(N)})\}$, learn a model that makes predictions.



Decision Trees , Linear Regression, Logistic Regression, SVM, Neural Networks (later)

MSE, MAE, BCE

Unsupervised Learning - Motivation

- We often have data where we don’t have the ground truth outputs.
  - In some cases, obtaining the ground truth can be very expensive.
  - Examples:
    - Topic modelling: documents/images/videos, where we do not know the category like health, finance, ...
    - Preventative medicine: health data, where we do not know disease/no disease.
    - …
- Human: much of our own learning happens by experiencing data with no ground truths.
- Examples:
  - Clustering, dimensionality reduction, image segmentation, GPT (contains unsupervised learning), …

Unsupervised Learning

Given a set of N data points $\{x^{(1)}, \dots, x^{(N)}\}$, learn patterns in the data.

Types of unsupervised learning:

- **Clustering**: identify clusters in the data
- **Dimensionality reduction**: find a lower-dimensional representation of the data
- …

Clustering

Clustering is a type of unsupervised machine learning technique used to **group similar data points into clusters or groups**.

The goal is to organize a set of objects in such a way that objects within the same group (cluster) are more similar to each other than to those in other groups.

In clustering, the number of clusters or groups is not predefined by the data.

Common applications:

- Data segmentation
- Anomaly detection

Dimensionality Reduction

Dimensionality reduction is a technique to **reduce the number of features (variables or dimensions)** in a dataset while retaining as much of the relevant information as possible.

Common applications:

- Visualizations
- Feature extractions / transformations

Outline

- Unsupervised Learning
- **K-means clustering**
  - Algorithm
  - Measuring the goodness of clusters
  - Picking the number of clusters
  - Variants
- Dimensionality Reduction
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA)

Background: Distance

Suppose that we have two d-dimensional vectors, u and v.

The straight-line **Euclidean distance** is defined as $\|u - v\| = \sqrt{\sum_{i=1}^{d}(u_i - v_i)^2}$

Machine learning models that rely only on distances between data points are called distance-based models.

(Do you recall similarity functions from kernel methods?)

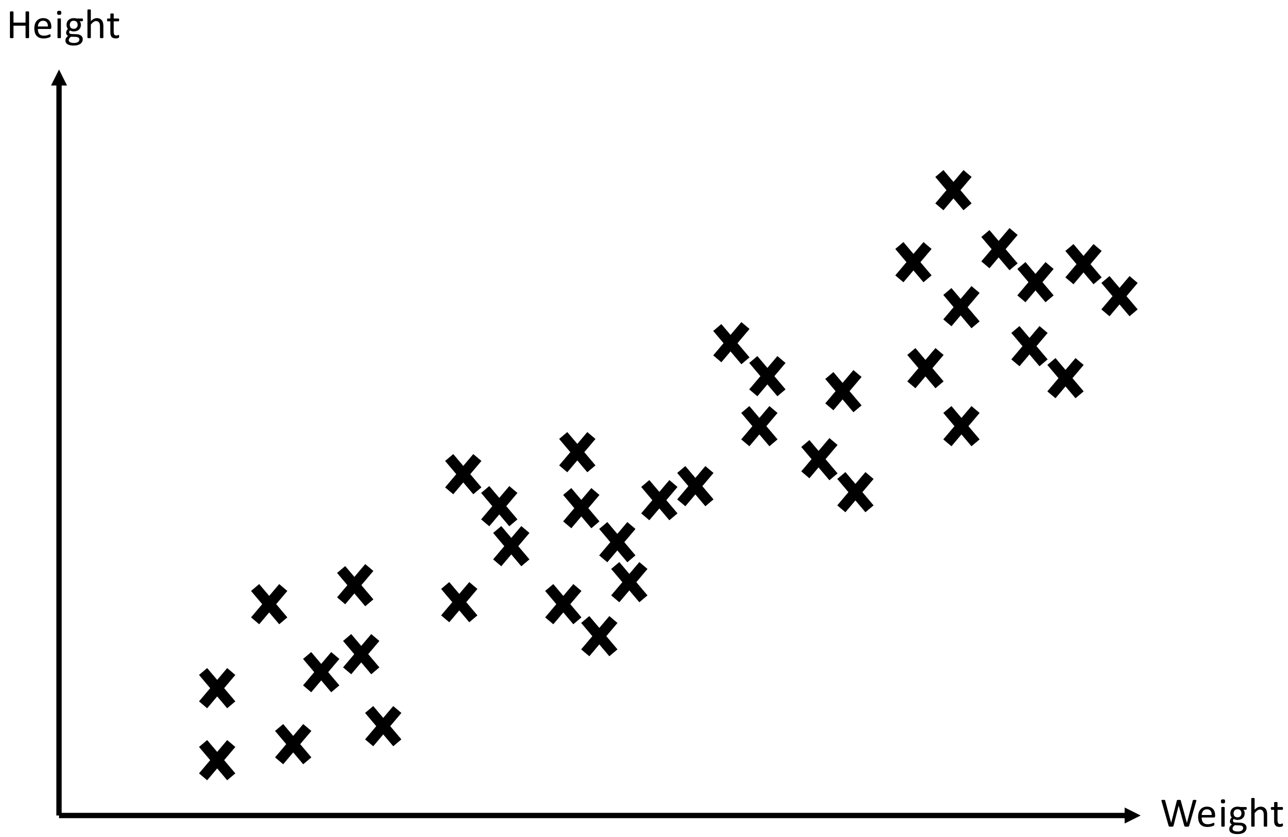
Example: Clothing Company

Suppose that

- You have data about your customers' **width** and **height**
- You want to mass-produce clothes of **K distinct sizes** such that the fit for everyone is pretty good (you only have K machines!)
- What would you do?

Example: Clothing Company

Goal: Mass-produce clothes of **K distinct sizes** such that the **fit for everyone is pretty good**.



Group people together!

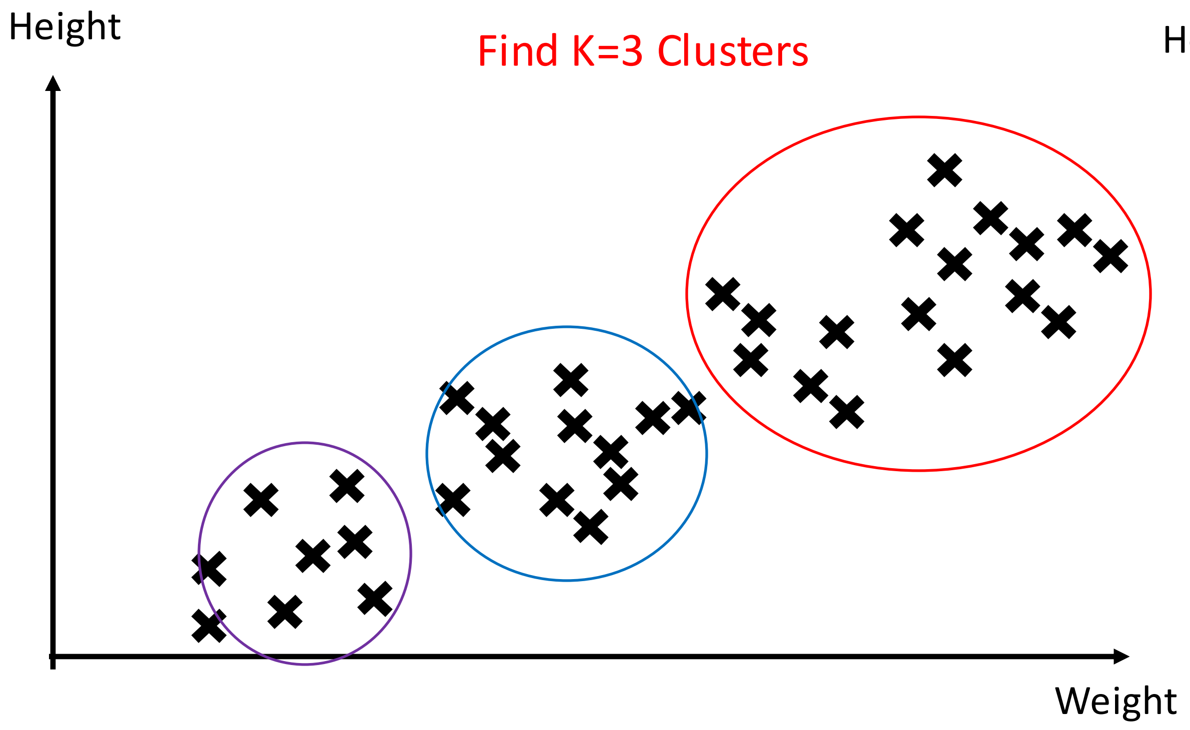
How many groups?

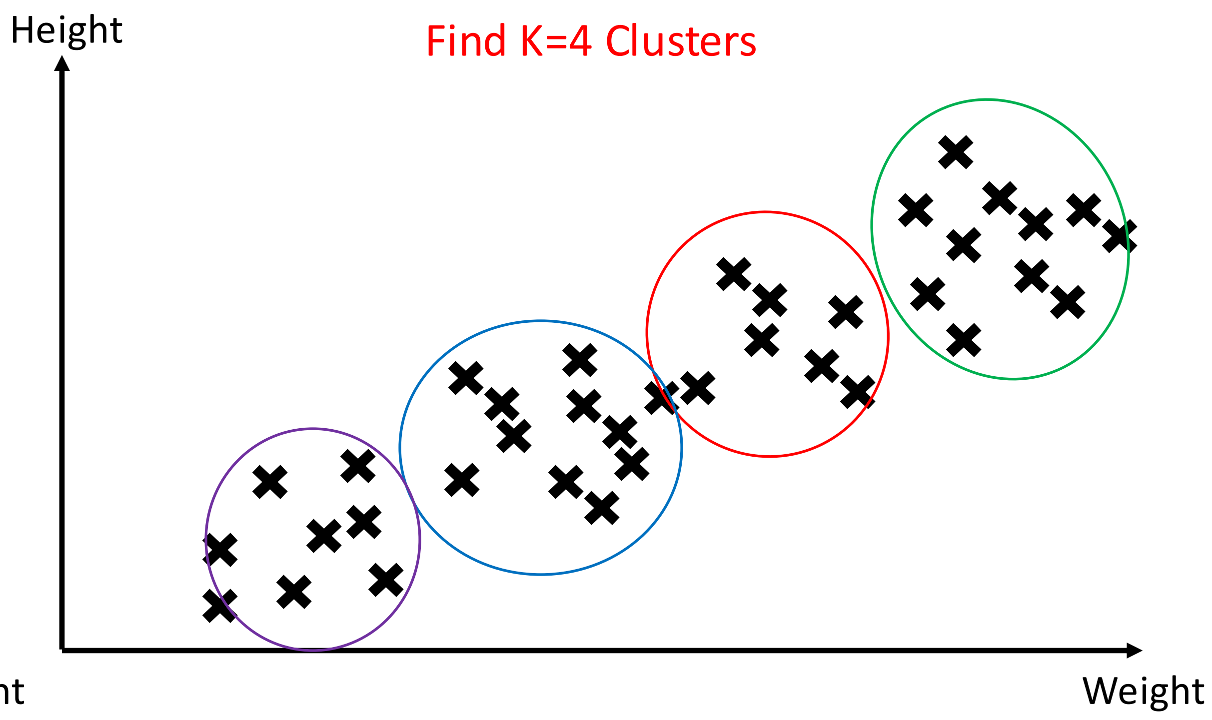
Up to us

This is called **clustering**.

Example: Clothing Company

Goal: Mass-produce clothes of **K distinct sizes** such that the **fit for everyone is pretty good**.





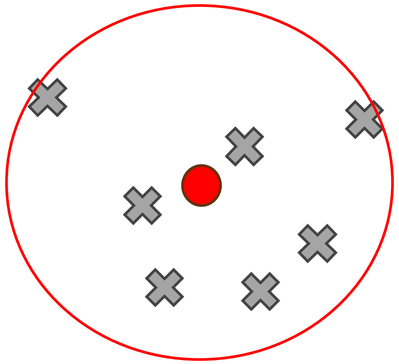
Centroid

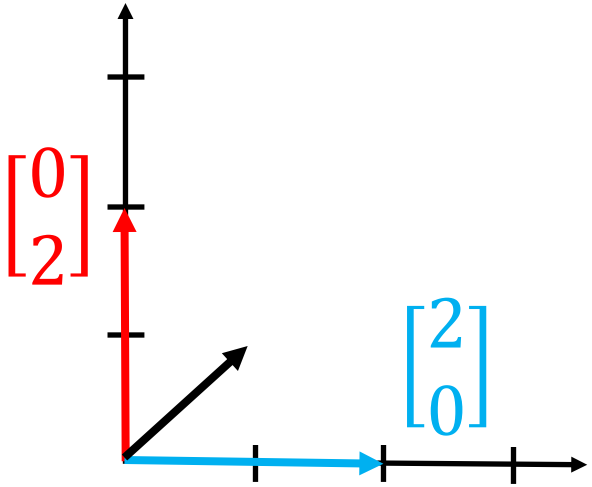
A **centroid** is the average of a set of points $x^{(i)}$ in a cluster.

For cluster j with N_j data points:

$$\mu_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x^{(i)}$$

A centroid defines a cluster.



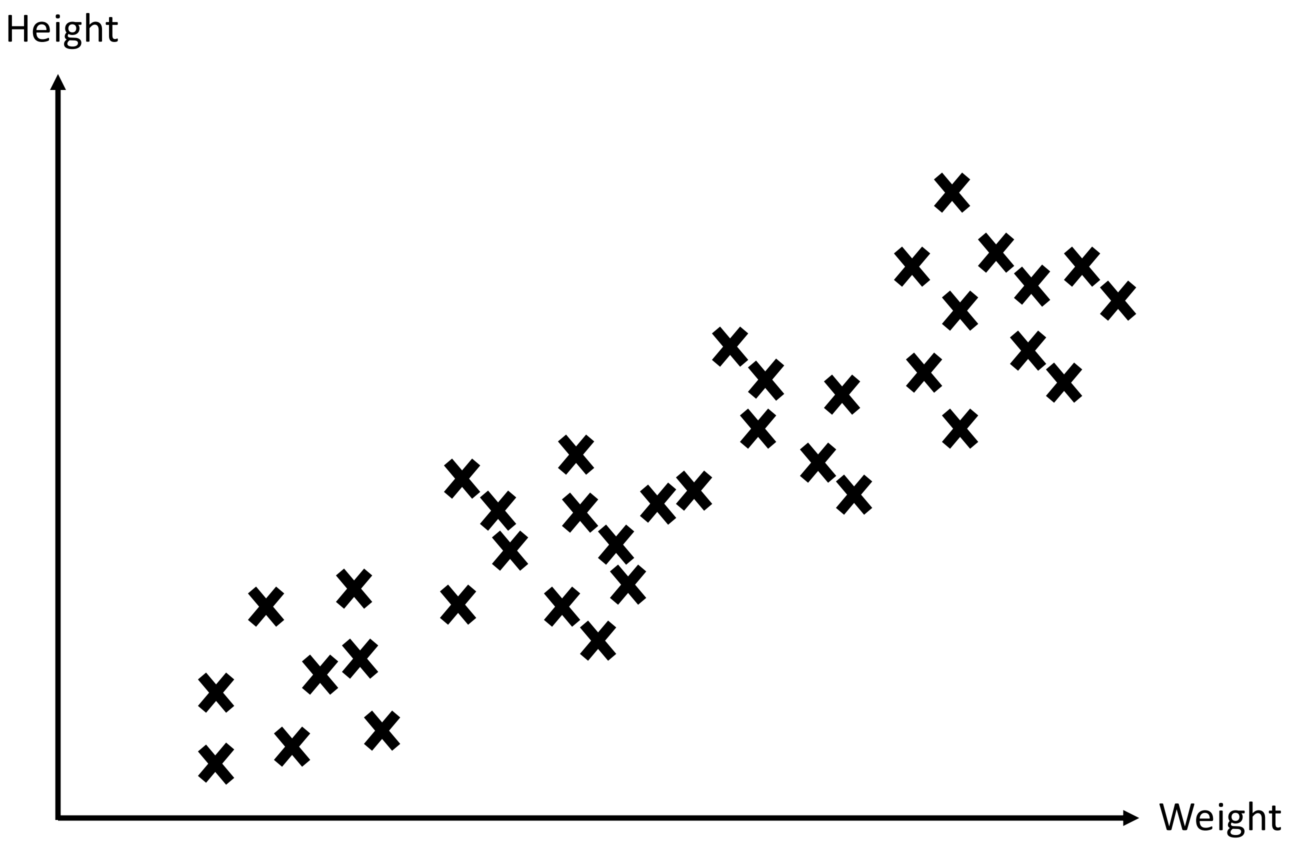


$$\frac{1}{2}\left(\begin{bmatrix} 2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 2 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Procedurally

How to Group Data?

Find K=3 Clusters



Let's start with the **centroids**
(which define the clusters)

Where should we put them (initially)?
We don't know the "correct" grouping.
➢ Randomly put the centroids.

What are the groups?
How should we assign the points to the group?
➢ Assign points to the nearest centroids.

Are the centroids "centroids"?
i.e., the center of the groups
➢ No! Need to recompute.

Are all points correctly grouped?
i.e., assigned to the nearest centroids
➢ No! need to reassign.

Assignments

Given N data points and K clusters. Define the variables

$$c^{(1)}, c^{(2)}, \cdots, c^{(N)} \in \{1, \dots, K\},$$

which represent the assignments to a cluster for each data point. These variables are updated in the K-Means algorithm together with the centroids.

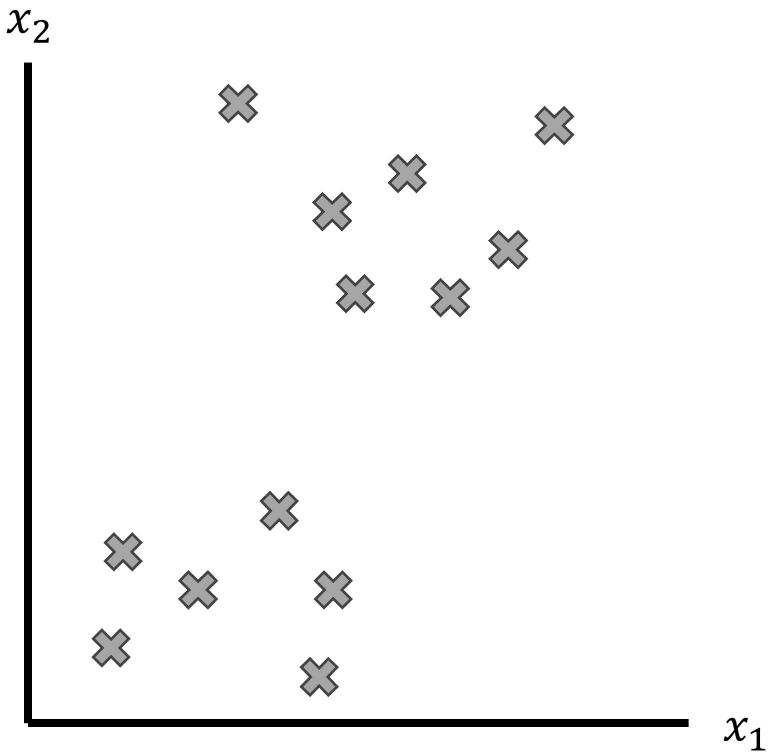
K-Means

- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \dots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \dots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k

K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \ldots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \ldots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k

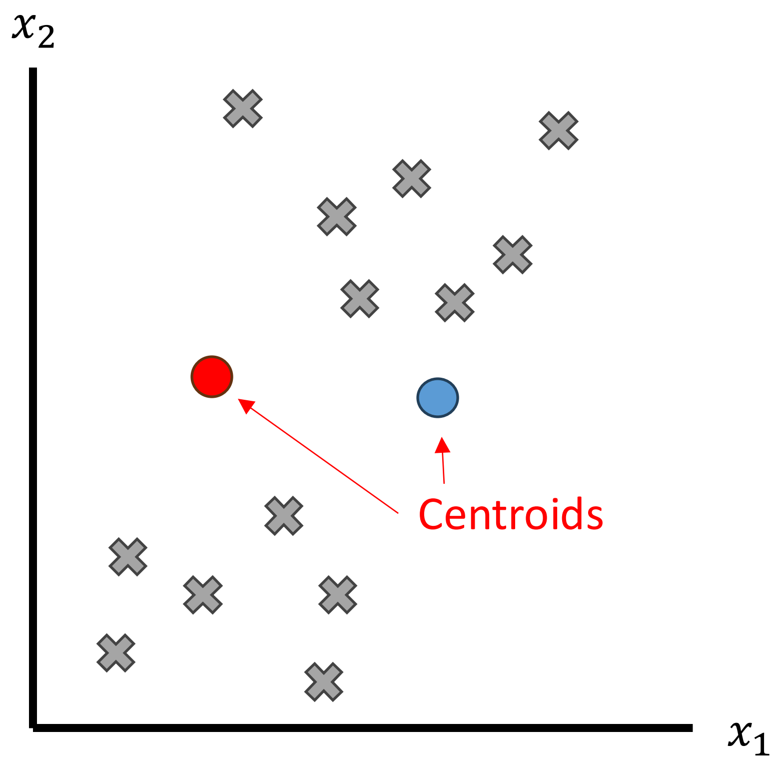
$K = 2$



K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \ldots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \ldots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k

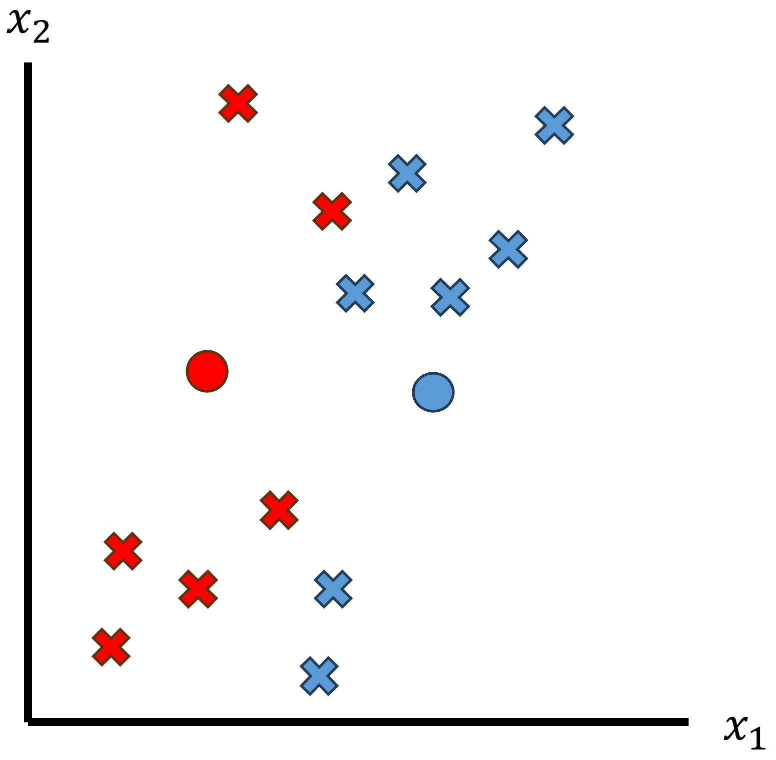
$K = 2$



K-Means

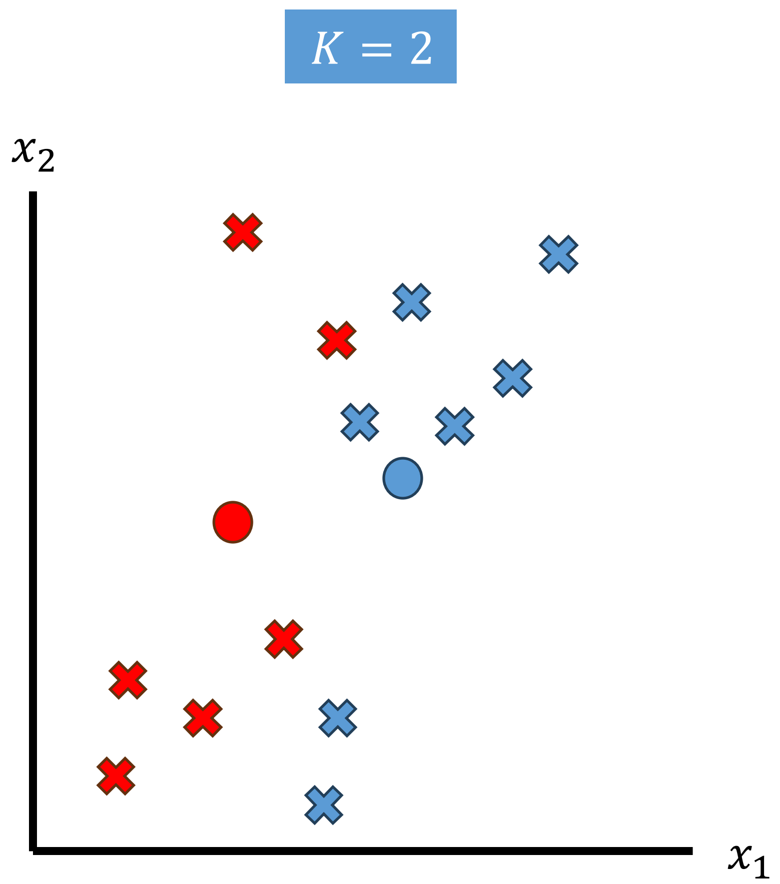
- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \dots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \dots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k

$K = 2$



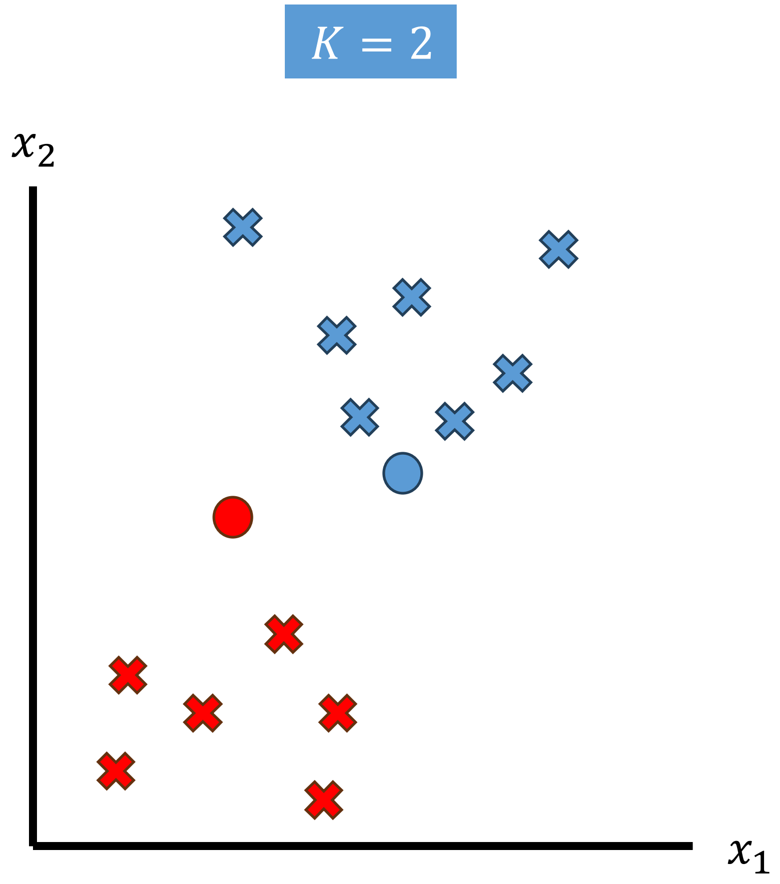
K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \ldots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \ldots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k



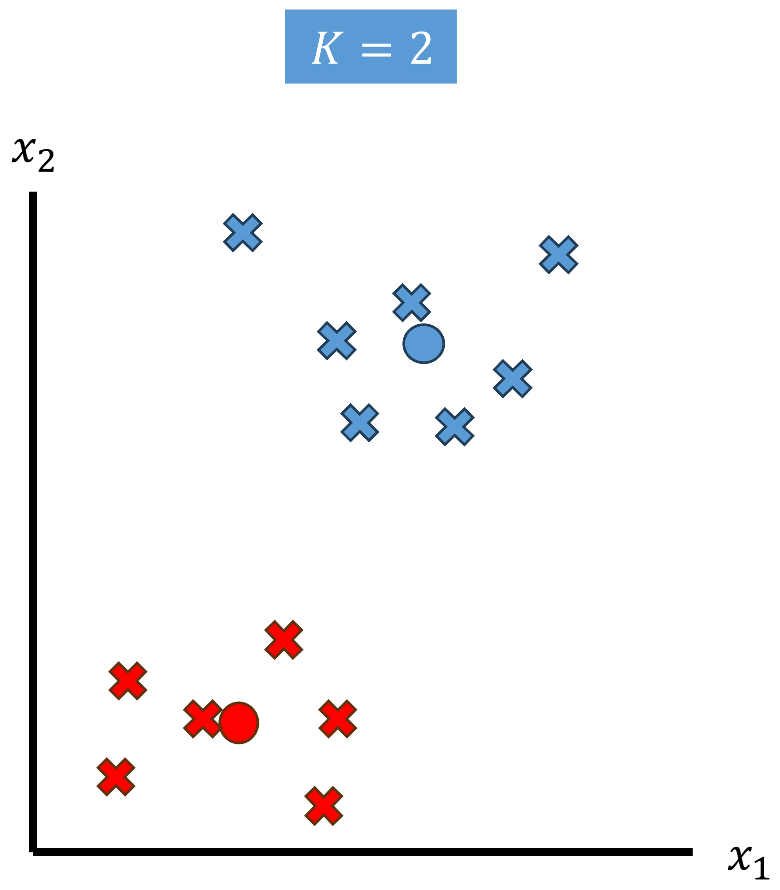
K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \ldots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \ldots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k



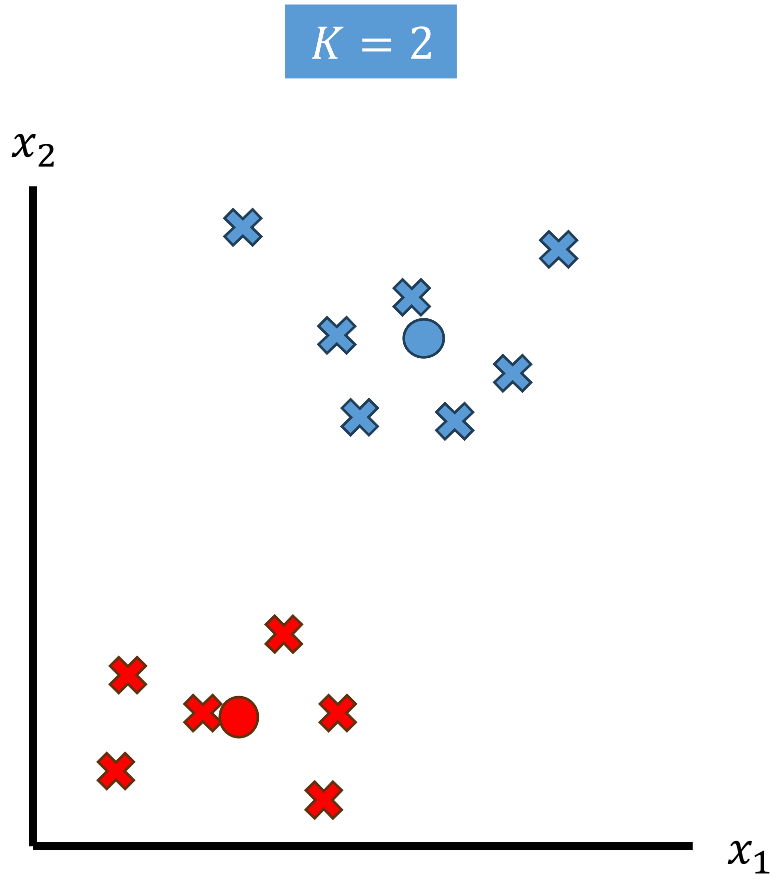
K-Means

- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \dots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \dots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k



K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, N$:
    - $c^{(i)} \leftarrow$ index of cluster centroid $(\mu_1, \ldots, \mu_K)$ closest to $x^{(i)}$
  - For $k = 1, \ldots, K$:
    - $\mu_k \leftarrow$ centroid of data points $x^{(i)}$ assigned to cluster k

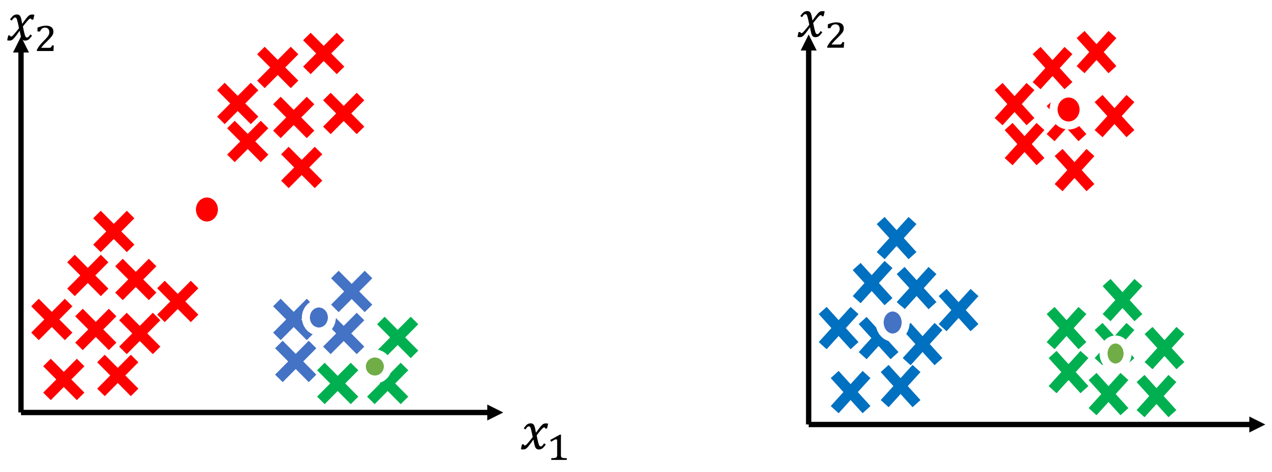


No more change: **converged!**

K-Means: Measuring Quality of Clusters

We can measure the distortion of the clusters: the average square distance of each sample to its assigned centroid.

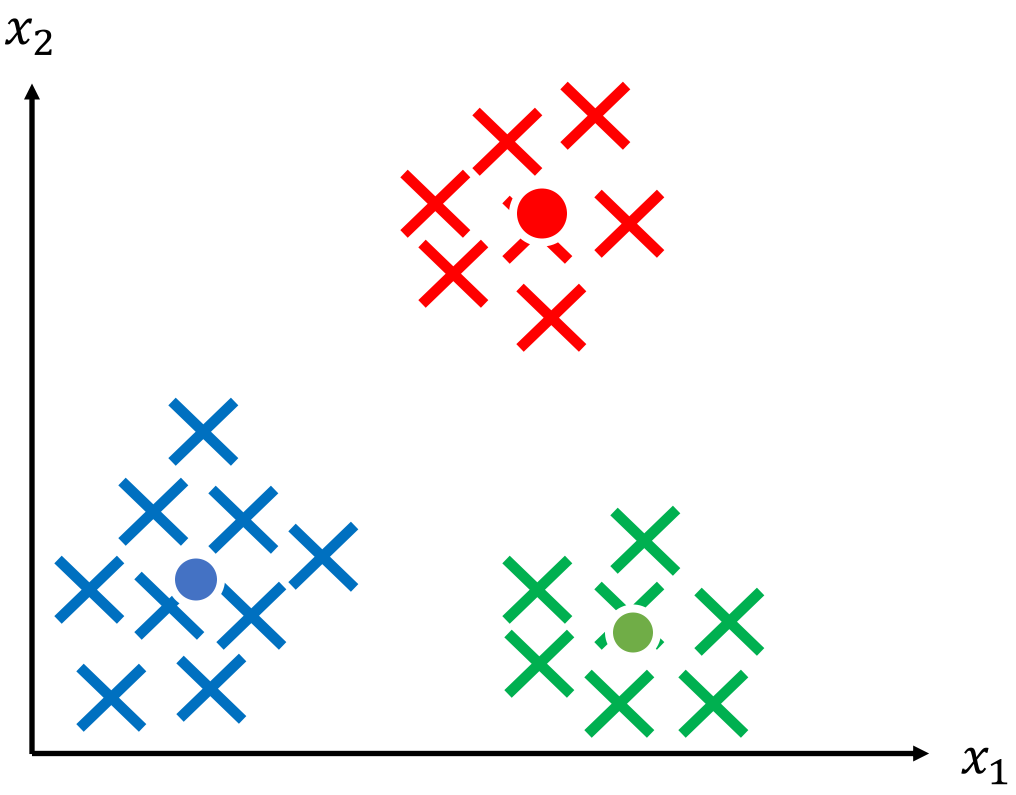
$$J\left(c^{(1)}, c^{(2)}, \cdots, c^{(N)}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K\right) = \frac{1}{N}\sum_{i=1}^{N}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{c^{(i)}}\right\|^2$$



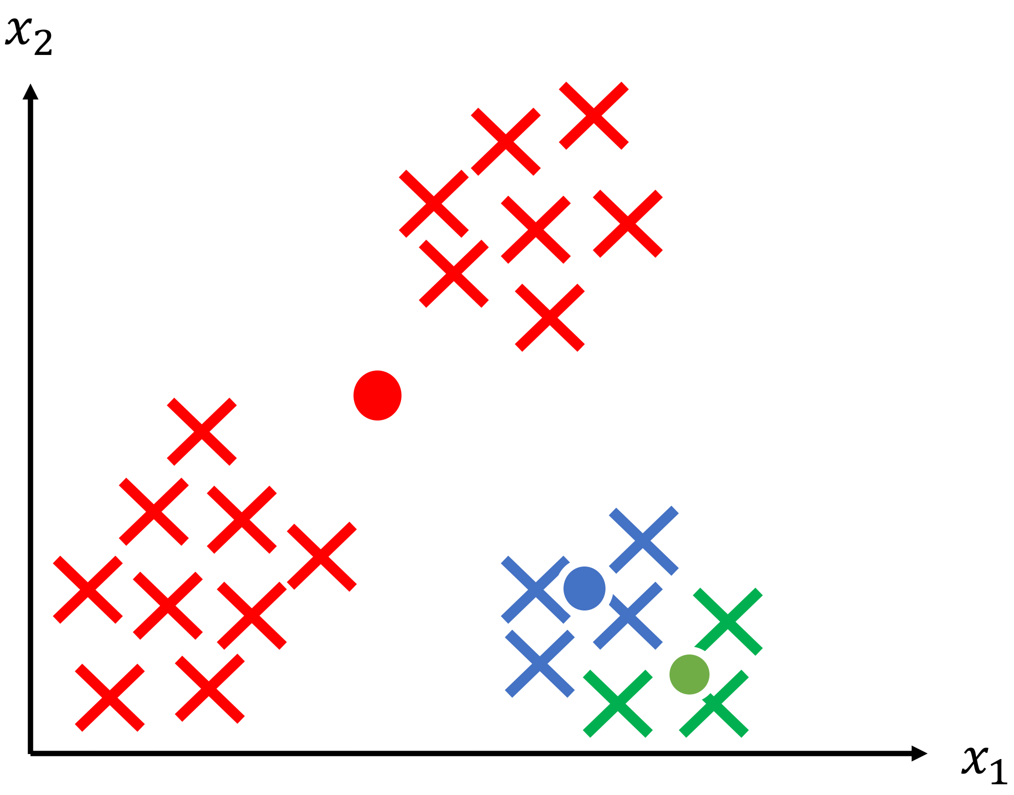
Converged

Converged, but lower distortion

K-Means: Local Optima



Ideal outcome



Possible outcome

Outcome is a stable configuration (the centroids will not move)

K-Means: Convergence

Theorem (Informal):

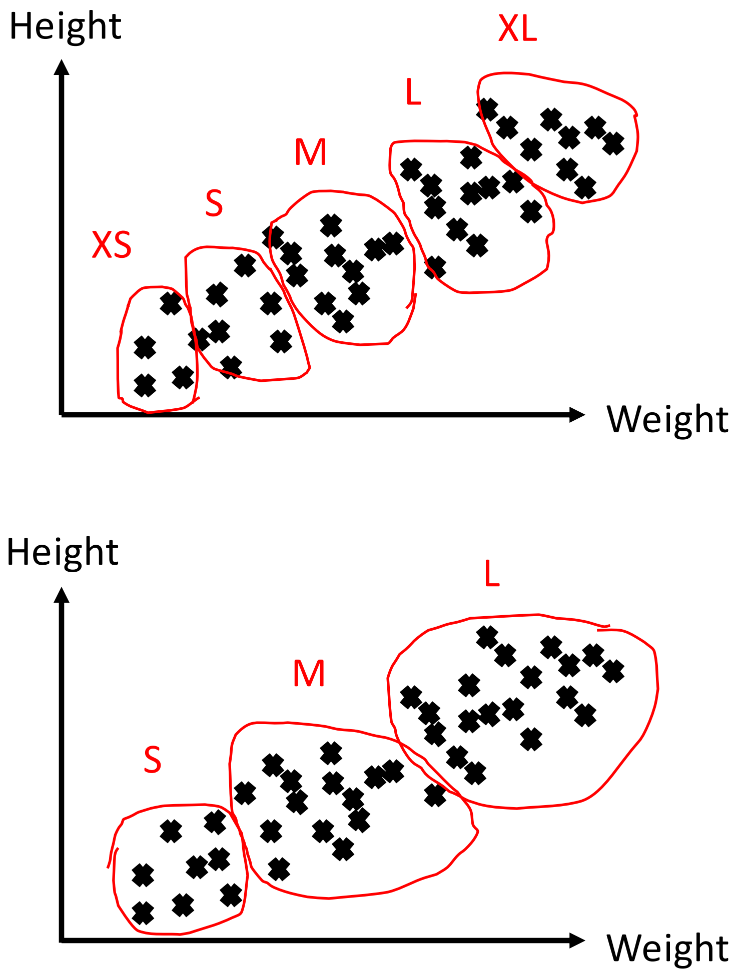
Each step in the K-Means algorithm never increases distortion.

More on convergence will be discussed in **Tutorial 7.**

K-Means: Picking the Number of Clusters

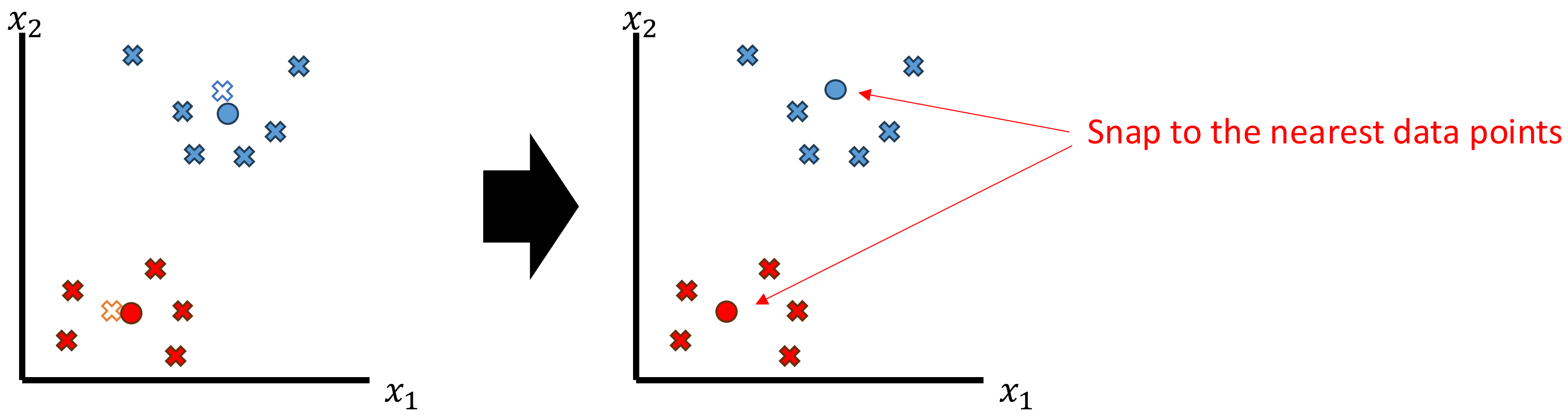
Elbow Method
Disclaimer: some data may not have elbow, or have multiple elbows.

Application dependent



K-Means: Variants

- Pick K initial centroids randomly from the points in the data
- K-Medoids: pick the data points that are closest to the centroids, and use them as the centroids.
  - "Snap" the centroids to the nearest data points



Clustering vs Classification

Aspect	Classification	Clustering
Type of Feedback	Supervised Learning	Unsupervised Learning
Input	Input-output pairs $\{(x^{(i)}, y^{(i)})\}$	Input only $\{x^{(i)}\}$
Output	A model: $h(x) \rightarrow \hat{y}$	Clustering/grouping Example: $x^{(1)}$ is assigned to group 5
Methods	Hyperplane-based (e.g., SVM), probability-based (e.g., logistic regression)	Distance-based
Number of • Classes (for classification) • Clusters (for clustering)	Defined by the dataset	Up to us*

*) the ones covered in this course

Poll Everywhere

What is an incorrect statement about K-means?

A: The K parameter is chosen by the user.

B: The centroids are always located at the given data points.

C: K-means minimizes the distortion.

D: Multiple restarts of K-means give different solutions.

Poll Everywhere

What is an incorrect statement about K-means?

A: The K parameter is chosen by the user.

B: The centroids are always located at the given data points.

C: K-means minimizes the distortion.

D: Multiple restarts of K-means give different solutions.

Outline

- Unsupervised Learning
- K-means clustering
  - Algorithm
  - Measuring the goodness of clusters
  - Picking the number of clusters
  - Variants
- Dimensionality Reduction
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA)

High-dimensional Features

Many machine learning problems have data with **high-dimensional features**.

For example:

- HD images have 1280×720=921,600 features

Is there any problem with this?

Curse of Dimensionality

Theorem (Sample Complexity, Informal):

Number of samples N needed to learn a hypothesis class increases exponentially with the number of features d:

$$N = O(2^d)$$

This is often called the curse of dimensionality.

High-dimensional Features

Many machine learning problems have data with **high-dimensional features**.

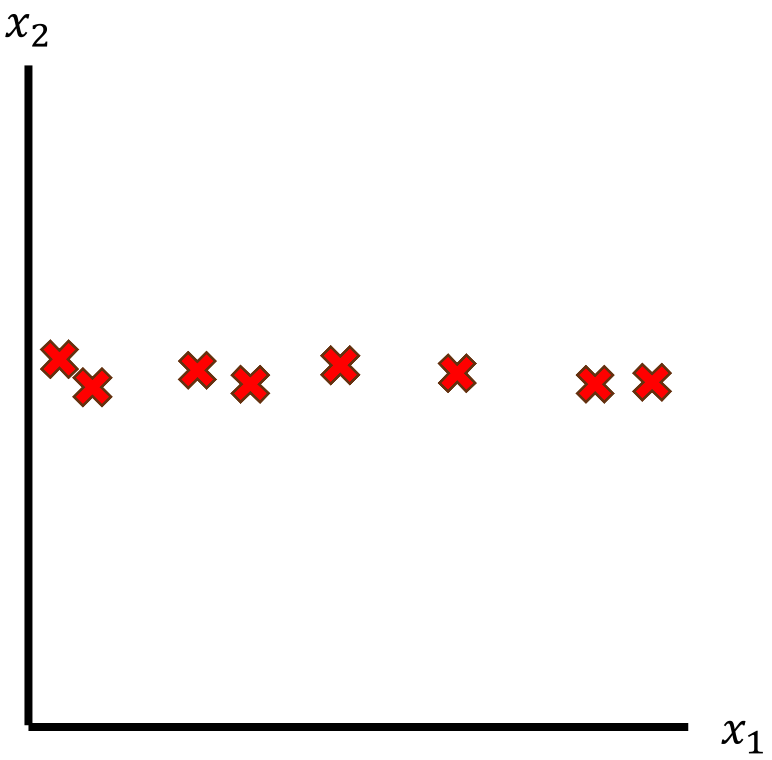
For example:

- HD images have $1280 \times 720 = 921{,}600$ features

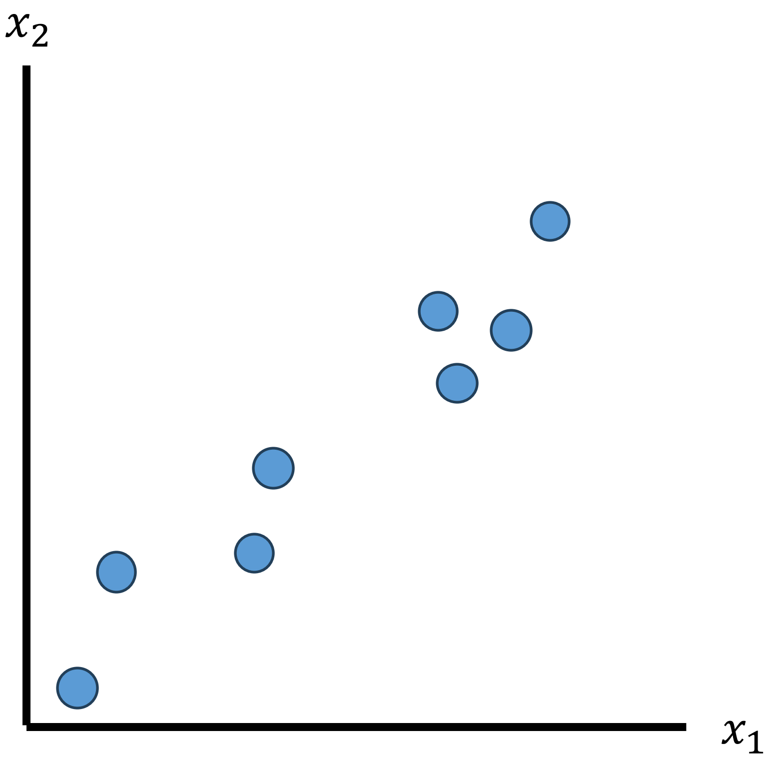
Machine learning model that takes in this high-dimensional input may suffer from the curse of dimensionality.

How to reduce the number of features?

Key Idea: Remove Redundant Features

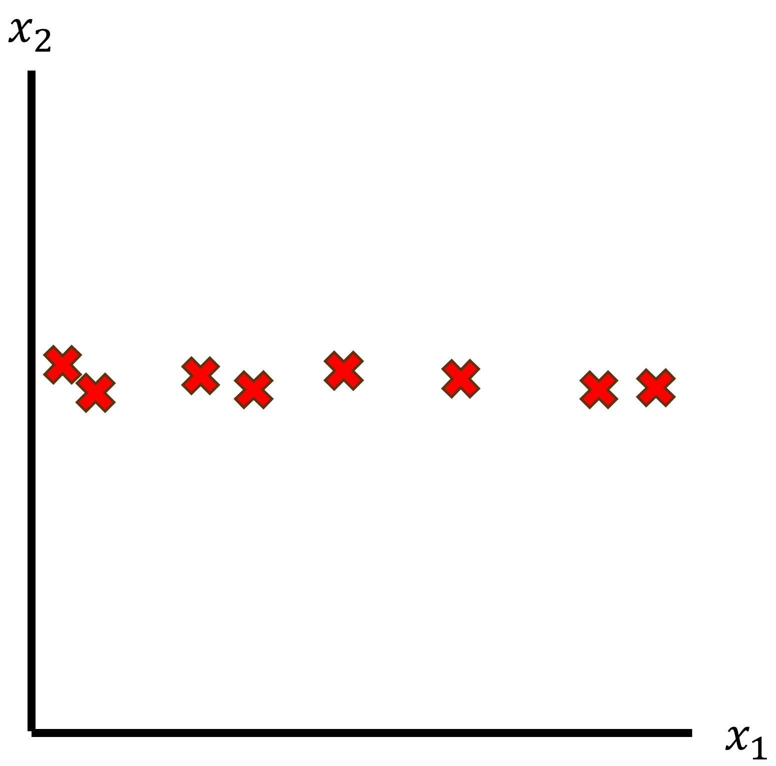


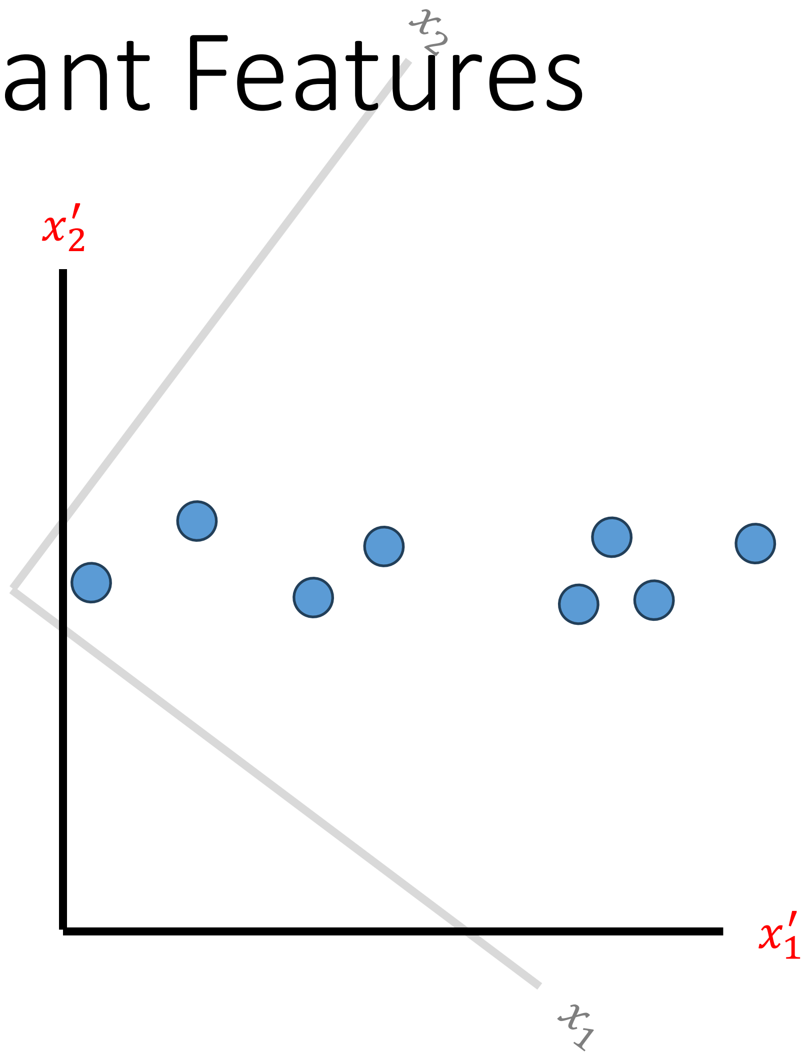
Feature x_1 is important
Feature x_2 is redundant



One of the features can be
removed if we change the basis

Key Idea: Remove Redundant Features





Feature x_1 is important
Feature x_2 is redundant

Feature x_1' is important
Feature x_2' is redundant

Recall: Data Matrix

$$
\begin{array}{ccc}
 & \text{Features} & \\
X = & \begin{bmatrix} 1 & x_1^{(1)} & & x_d^{(1)} \\ 1 & x_1^{(2)} & \cdots & x_d^{(2)} \\ 1 & \vdots & & \vdots \\ 1 & x_1^{(N)} & & x_d^{(N)} \end{bmatrix} & \text{Data points}
\end{array}
$$

This matrix is everything we have in the unsupervised setting.

Recall: Data Matrix (Transposed)

Data points

$$X^{T} = \begin{bmatrix} 1 & 1 & & 1 \\ x_1^{(1)} & x_1^{(2)} & & x_1^{(N)} \\ \vdots & \vdots & \cdots & \vdots \\ x_d^{(1)} & x_d^{(2)} & & x_d^{(N)} \end{bmatrix} \quad \text{Features}$$

We work with this matrix in the following slides.

Identifying Important Features in Data Matrix

How do we identify important features in the following data matrix?

$$X^T = \begin{bmatrix} 1 & 1 & & 1 \\ x_1^{(1)} & x_1^{(2)} & & x_1^{(N)} \\ \vdots & \vdots & \cdots & \vdots \\ x_d^{(1)} & x_d^{(2)} & & x_d^{(N)} \end{bmatrix}$$

Use tools from linear algebra to find **better** and **more compact basis**.

Singular Value Decomposition (SVD)

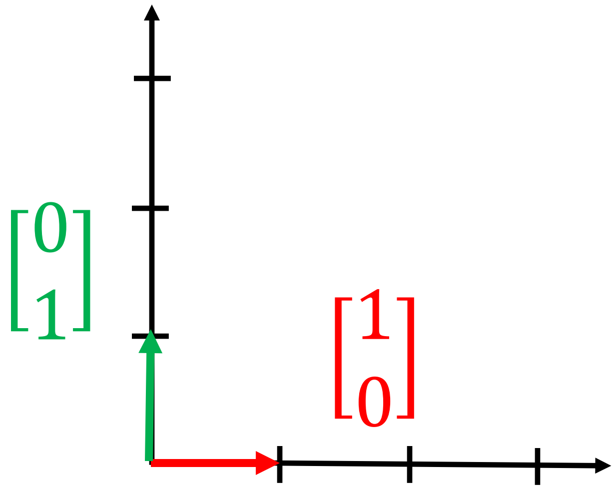
Given a matrix A, we can decompose it into 3 matrices: U, Σ, and V^T.

$$\boldsymbol{A = U\Sigma V^T}$$

Background: Orthonormal Basis

Orthonormality: Two vectors a and b are called orthonormal if they are:

- Normalized: $||a||^2 = a^T a = 1$ and $||b||^2 = 1$, and
- Orthogonal: $a^T b = 0$.



Orthonormal basis: A set of vectors such that every vector can be expressed as a unique linear combination of the vectors in the set.

SVD: Existence

Theorem: Without loss of generality, let $d > N$. For any $d \times N$ real-valued matrix $\boldsymbol{A}$, there exists a factorization $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\mathbf{V}^{\boldsymbol{T}}$ called SVD, such that:

- $\boldsymbol{U}$ is $d \times d$ and has d orthonormal columns — left singular vectors
- $\boldsymbol{\Sigma}$ is $\mathrm{d} \times N$ and is a diagonal matrix with $\sigma_j \geq 0$ — singular values
  - Ordering: $\sigma_1 \geq \sigma_2 \geq \ \ldots \geq \sigma_N \geq 0$
- $\mathbf{V}$ is $N \times N$ and has N orthonormal columns and rows — right singular vectors

Why "singular"? Possibly from integral theory, however it's unclear:
On the Early History of the Singular Value Decomposition (G. W. Stewart)

SVD: Schematic for X^T

$\boldsymbol{A = U\Sigma V^T}$

$$X^T = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(N)} \\ | & | & & | \end{bmatrix}$$

$$= U\Sigma V^T$$

$$= \underbrace{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(d)} \\ | & | & & | \end{bmatrix}}_{d \times d} \underbrace{\begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix}}_{d \times N} \underbrace{\begin{bmatrix} | & | & & | \\ v^{(1)} & v^{(2)} & \cdots & v^{(N)} \\ | & | & & | \end{bmatrix}^T}_{N \times N}$$

SVD: Interpretation

$\boldsymbol{X^T = U\Sigma V^T}$

$$X^T = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(N)} \\ | & | & & | \end{bmatrix}$$

$$= U\Sigma V^T$$

$$= \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(d)} \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} | & | & & | \\ v^{(1)} & v^{(2)} & \cdots & v^{(N)} \\ | & | & & | \end{bmatrix}^T$$

New (Orthonormal) Basis — Importance — Linear combination coefficients

SVD: Example

$$X^T = U\Sigma V^T$$

$$X^T = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} \\ x_2^{(1)} & x_2^{(2)} \end{bmatrix} = \begin{bmatrix} | & | \\ u^{(1)} & u^{(2)} \\ | & | \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} v_1^{(1)} & v_1^{(2)} \\ v_2^{(1)} & v_2^{(2)} \end{bmatrix}^T$$

$$= \begin{bmatrix} | & | \\ \sigma_1 u^{(1)} & \sigma_2 u^{(2)} \\ | & | \end{bmatrix} \begin{bmatrix} v_1^{(1)} & v_2^{(1)} \\ v_1^{(2)} & v_2^{(2)} \end{bmatrix}$$

$$= \begin{bmatrix} | & | \\ v_1^{(1)} \sigma_1 u^{(1)} + v_1^{(2)} \sigma_2\, u^{(2)} & v_2^{(1)} \sigma_1 u^{(1)} + v_2^{(2)} \sigma_2\, u^{(2)} \\ | & | \end{bmatrix}$$

Data points $x^{(i)}$ are linear combinations of basis vectors $u^{(1)}$ and $u^{(2)}$

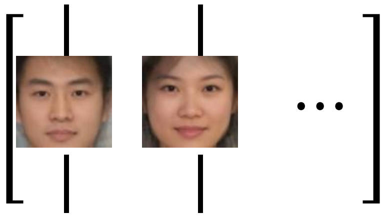
SVD: Another Interpretation

$$\boldsymbol{X^T = U\Sigma V^T}$$

$$X^T =$$

$$= U\Sigma V^T$$

$$=$$



$$\begin{bmatrix} 45 & 0 & \dots & 0 \\ 0 & 10 & \dots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \dots & 0.01 \\ 0 & 0 & \dots & 0 \end{bmatrix} \begin{array}{cccc} \text{Face 1} & \text{Face 2} & & \\ \downarrow & \downarrow & & \\ \multicolumn{4}{c}{\begin{bmatrix} 0.9 & -0.5 & \cdots & \cdots \\ -0.2 & 0.8 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ -0.1 & 0.4 & \cdots & \cdots \end{bmatrix}} \end{array}$$

Templates | Template Importance | Combination coefficients

Image source:
https://www.kinfacew.com/datasets.html
https://faceresearch.org

Dimensionality Reduction via SVD

$\boldsymbol{X^T} = \boldsymbol{U\Sigma V^T}$

Key idea: The N singular values tell us about the importance of the new basis vectors. Remove less important basis vectors.

How? Recall that singular values are ordered: $\sigma_1 \geq \sigma_2 \geq \ \dots \geq \sigma_N > 0$

We can set all singular values except the first r to 0.

$$\begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$r = 2$

SVD: Effect of Reduction

$$\boldsymbol{X^T = U\Sigma V^T}$$

$$X^T = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(N)} \\ | & | & & | \end{bmatrix}$$

$$= U\Sigma V^T$$

$$= \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(d)} \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} | & | & & | \\ v^{(1)} & v^{(2)} & \cdots & v^{(N)} \\ | & | & & | \end{bmatrix}^T$$

New (Orthonormal) Basis — Importance — Linear combination coefficients

SVD: Effect of Reduction

$\boldsymbol{X^T = U\Sigma V^T}$

Recall: Data points $x^{(i)}$ are linear combinations of basis vectors $u^{(j)}, j \in \{1, \ldots, d\}$

$$X^T = \begin{bmatrix} | & | & | & | \\ v_1^{(1)}\sigma_1 u^{(1)} + v_1^{(2)}\sigma_2 u^{(2)} + \cdots & v_2^{(1)}\sigma_1 u^{(1)} + v_2^{(2)}\sigma_2 u^{(2)} + \cdots & \ldots & \ldots \\ | & | & | & | \end{bmatrix}$$

After dimensionality reduction with $r = 2$, data points $x^{(i)}$ are linear combinations of $u^{(j)}, j \in \{1,2\}$

$$\tilde{X}^T = \begin{bmatrix} | & | & | & | \\ v_1^{(1)}\sigma_1 u^{(1)} + v_1^{(2)}\sigma_2 u^{(2)} & v_2^{(1)}\sigma_1 u^{(1)} + v_2^{(2)}\sigma_2 u^{(2)} & \ldots & \ldots \\ | & | & | & | \end{bmatrix}$$

SVD: Dimensionality Reduction

$r < N < d$

Recap: Dimensionality reduction starts at the singular value matrix:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix} \qquad \tilde{\Sigma} := \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

The truncated matrix $\tilde{\Sigma}$ implies a smaller basis matrix:

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(d)} \\ | & | & & | \end{bmatrix} \qquad \tilde{U} := \begin{bmatrix} | & | \\ u^{(1)} & u^{(2)} \\ | & | \end{bmatrix} \in d \times r$$

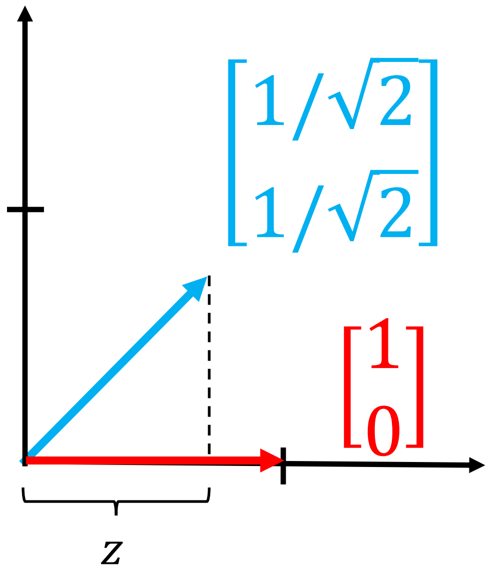
Can we use $\tilde{U}$ to compress vectors?

Background: Dot Product and Projection

Let u, v be two vectors.

$$u^T v = \sum_{j=1}^{d} u_j v_j = z||v||$$

The dot product gives the length of the projection of u onto v times the norm of v.



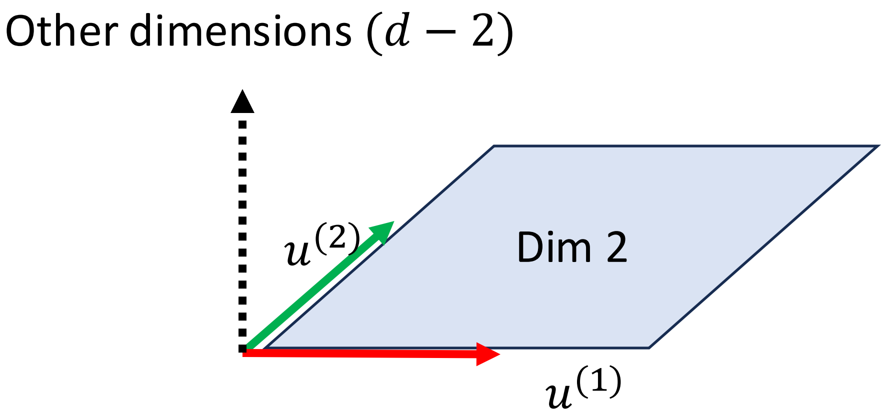
$$z = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}$$

Projection using New Basis

The matrix $\widetilde{U}$ defines 2-dimensional subspace of the feature space. (In general, r-dimensional)

$$\widetilde{U} = \begin{bmatrix} | & | \\ u^{(1)} & u^{(2)} \\ | & | \end{bmatrix}$$

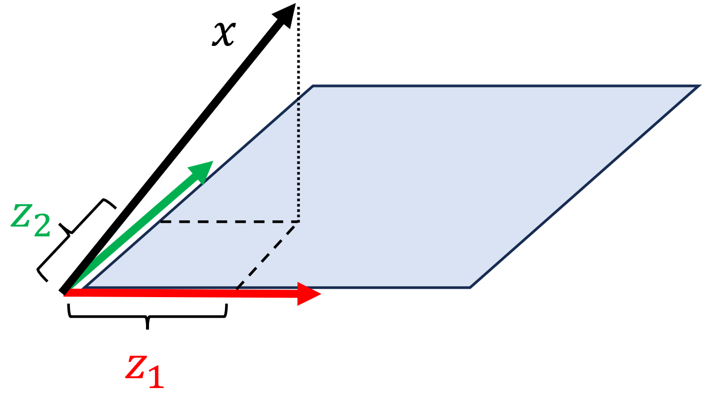
d-dimensional vector



For any feature vector x, multiplying with $\widetilde{U}$ obtains the linear combination coefficients in the 2-dimensional subspace:

$$\widetilde{U}^T = \begin{bmatrix} - & u^{(1)^T} & - \\ - & u^{(2)^T} & - \end{bmatrix} \qquad x = \begin{bmatrix} 1 \\ \dots \\ x_d \end{bmatrix}$$

$$\widetilde{U}^T x = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$



SVD: Compression

$r < N < d$

We can use the matrix $\widetilde{U}^T$ to compress the data matrix X^T:

$$Z := \widetilde{U}^T X^T$$

Here, Z is $r \times N$ matrix.

Intuition: we project the data points $x^{(i)}$ to the new (lower-dimensional) basis $\widetilde{U}$.
Z contains all the projection coefficients.

Can we de-compress (reconstruct) the original data?

SVD: Reconstruction

$r < N < d$

Theorem (informal): For a fixed r, it holds that that $\tilde{U}\tilde{U}^T \approx \mathbb{I}$, with approximation error dependent on r.

Thus, we can reconstruct the data in the original feature dimension d:

$$\tilde{X}^T := \tilde{U}Z$$

$$\tilde{X}^T = \tilde{U}\tilde{U}^T X^T \approx X^T$$

How much information is retained?

Background: Expected Value

Given a discrete random variable $X \in \{1,2,\ldots,M\}$ and a probability $p(X)$ for the outcomes of the random variable.

The expected value is defined as $\mathbb{E}_p[X] := \sum_{x=1}^{M} p(x)x$

Example: Uniform 6-sided dice.

$$\mathbb{E}_p[X] = \sum_{x=1}^{6} p(x)x$$
$$= \frac{1}{6}(1+2+3+4+5+6) = \frac{21}{6} = 3.5$$

Background: Variance

Given a random variable X and a probability $p(X)$ for the outcomes of the random variable.

The variance is defined as $\mathbb{V}_p[X] := \mathbb{E}_p\left[\left(X - \mathbb{E}_p[X]\right)^2\right]$.

Example: Uniform 6-sided dice.

$$\mathbb{V}_p[X] = \sum_{x=1}^{6} p(x)(x - 3.5)^2$$

$$= \frac{1}{6}(2.5^2 + 1.5^2 + 0.5^2 + 0.5^2 + 1.5^2 + 2.5^2) = \frac{35}{12} \approx 2.92$$

Mean-centred Data

Assume we work with mean-centred data.

1. Compute mean feature vector over samples: $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x^{(i)}$

2. Compute mean-centred data points: $\hat{x}^{(i)} = x^{(i)} - \bar{x}$ for all i.

3. Use mean-centred data matrix $\hat{X}^T \leftarrow \left[\hat{x}^{(1)}, \dots, \hat{x}^{(N)}\right]$.

SVD and Variances

$$\hat{X}^T = \overset{d \times d}{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(d)} \\ | & | & & | \end{bmatrix}} \overset{d \times N}{\begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_N \\ 0 & 0 & \dots & 0 \end{bmatrix}} \overset{N \times N}{\begin{bmatrix} | & | & & | \\ v^{(1)} & v^{(2)} & \cdots & v^{(N)} \\ | & | & & | \end{bmatrix}}^T$$

Theorem (informal): Given a mean-centred data matrix $\hat{X}^T$, the values $\frac{\sigma_j^2}{N-1}$ are variances of the data in the basis defined by the vectors $u^{(j)}$.

Compressing Data while Retaining Variance

Task: “Retain at least 99% of variance in the data”

Solution: Choose minimum r, such that $\frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{i=1}^{N} \sigma_i^2} \geq 0.99.$

Theorem: Using this r, the original data points $\hat{x}^{(i)}$ (mean-centred) and reconstructed data points $\tilde{x}^{(i)}$ are close as follows:

$$\frac{\sum_{i=1}^{N} \left\| \hat{x}^{(i)} - \tilde{x}^{(i)} \right\|^2}{\sum_{i=1}^{N} \| \hat{x}^{(i)} \|^2} \leq 0.01$$

Principal Component Analysis (PCA)

Statistics application of SVD. Capture components that maximize the *statistical variations* of the data. Same idea, but uses sample covariance matrix as an input (will be discussed in tutorial).

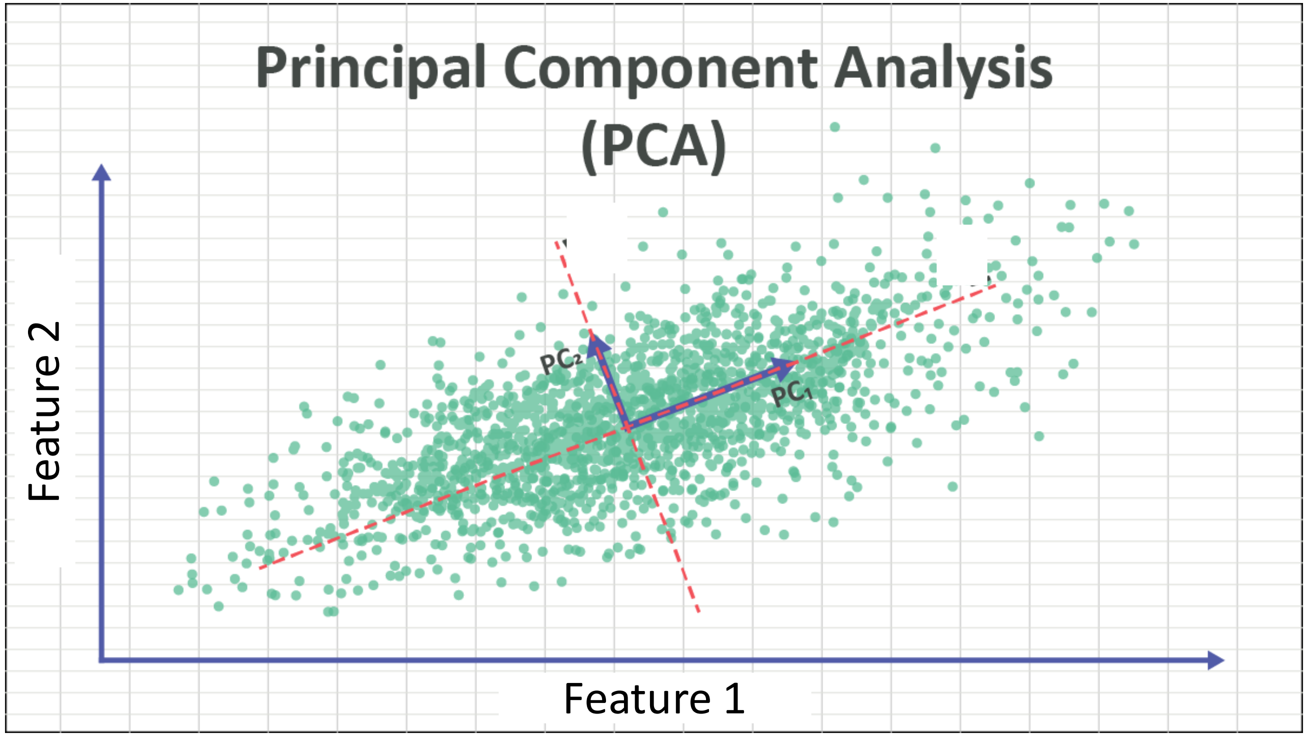


Image credit: numxl

Singular Value Decomposition (SVD)

Algorithm:

- Outside of the scope of this course
- If you are curious:
  - https://web.stanford.edu/class/cme335/spr11/lecture6.pdf

Implementations:

- Numpy: numpy.linalg.svd
  - https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html
- Matlab: svd
  - https://www.mathworks.com/help/matlab/ref/double.svd.html

Summary

- Unsupervised Learning: learn patterns of the data without labels
- K-means clustering
  - Algorithm: find **centroids** based on the data points, **assign each point to the closest centroid**
  - Measuring the quality of clusters: distance of each point to their centroid
  - Picking the number of clusters: **elbow method**, business needs
  - Variants: **K-medoids**, etc
- Dimensionality Reduction: finding **new basis** that best captures the data
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA): **statistical application** of SVD

Coming Up Next Week

- Neural networks
- Dr Conghui will take over starting from next week!

To Do

- **Lecture Training 8**
  - +250 Free EXP
  - +100 Early bird bonus
- **Problem Set 4**
- **Mini Project**
  - ~3 weeks left!