CS2109S: Introduction to AI and Machine Learning

Lecture 11: Unsupervised Learning

11 November 2023

Announcement

- We’ll release the midterm grades soon (later today or tomorrow)
- The mockup final will also be released soon
  - We’re trying to integrate it on a new platform for better experience

Recap

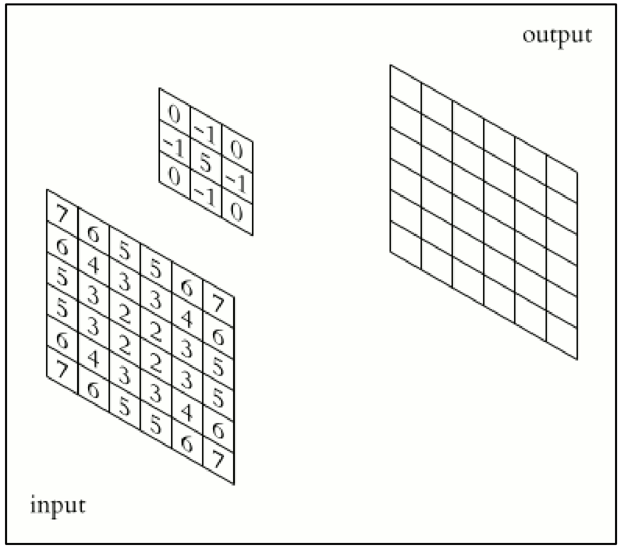
- Deep Neural Networks – neural networks with >3 layers
- Convolution Neural Networks
  - Motivation: handling **spatial structure**, translation **invariant**
  - Convolution (multiply-sum), Pooling (downsampling) Layer, and Common Architectures
  - Applications: image recognition, image segmentation, object detection
- Recurrent Neural Networks
  - Motivation: handling **sequential** data
  - Recurrent Neural Networks and Variants:
    - **neural networks with loop** $y_t, h_t = RNN(x_t, h_{t-1})$
  - Applications: machine translation, summarization, etc
- Attention, Transformers, GPT, and ChatGPT
  - Attention: **focus on things that matters**. **Massive** neural networks; train with **billions of data**.
- Issues with Deep Learning: overfitting, gradient vanishing/exploding

Image credit: https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37

ERRATA

Convolution: 2D

$$f_{conv}(\boldsymbol{X}) = \boldsymbol{W} * \boldsymbol{X}$$



0	1	1	0	0
0	1	1	0	0
1	0	0	1	1
0	1	1	0	1
1	1	1	1	1

Image Input
$\boldsymbol{X}$

$*$

0	-1	0
-1	5	-1
0	-1	0

Kernel / Filter
$\boldsymbol{W}$

$=$

3	3	-2
-3	-3	4
3	3	-4

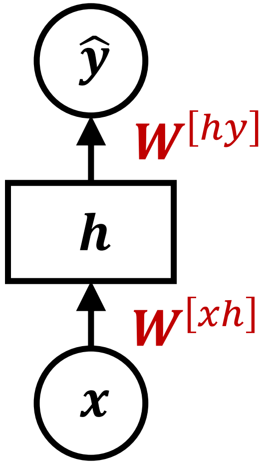
Feature Map

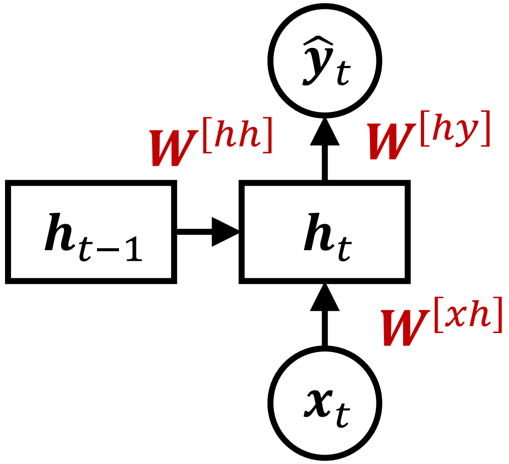
What if we want to detect other features (e.g., mouth)?

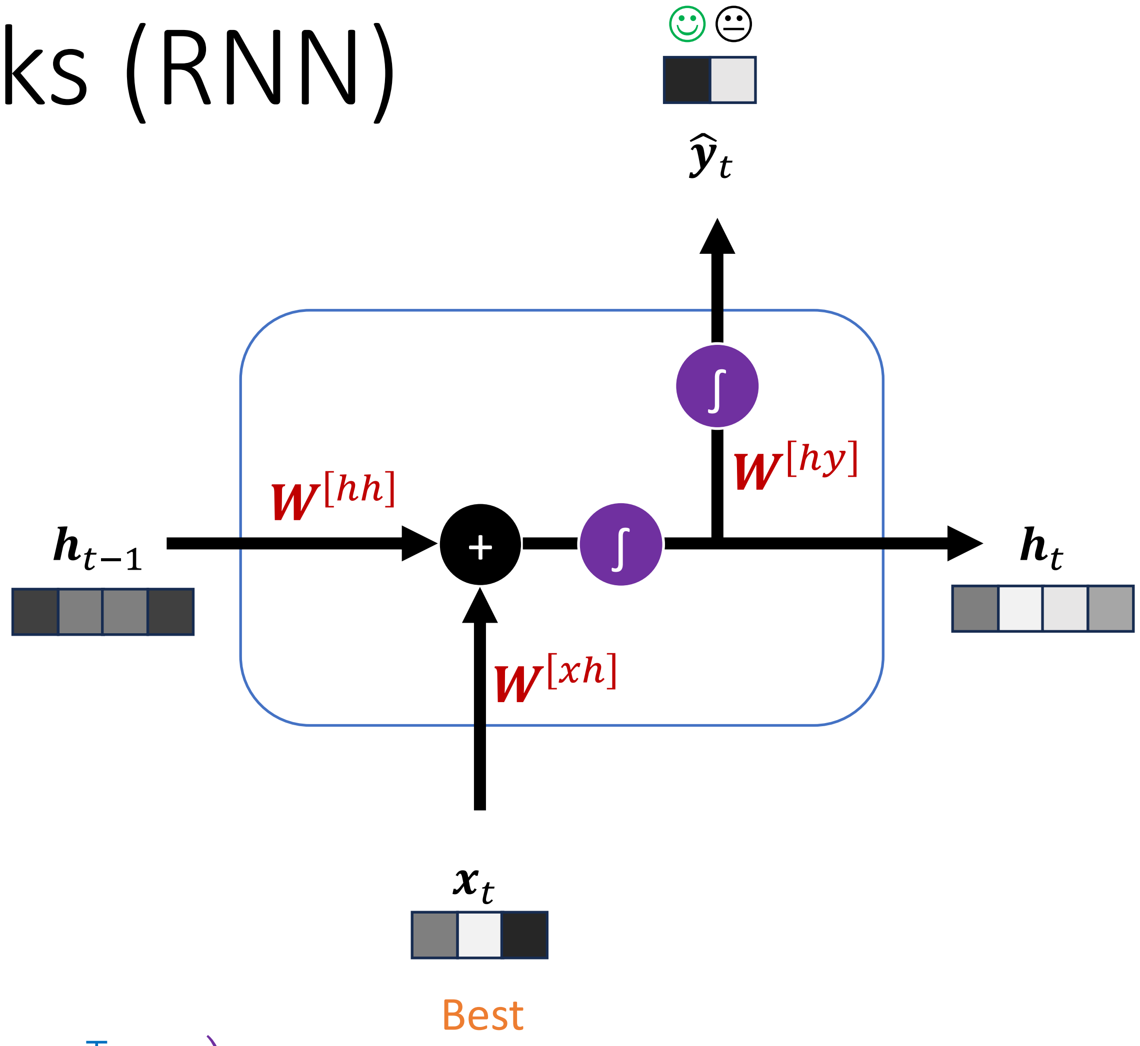
Multiply the sliding input window with kernel then **sum**

RECAP

Recurrent Neural Networks (RNN)







Feed-forward networks

$$\boldsymbol{y} = g^{[y]}\left(\left(\boldsymbol{W}^{[hy]}\right)^{\top}\boldsymbol{h}\right)$$

$$\boldsymbol{h} = g^{[h]}\left(\left(\boldsymbol{W}^{[xh]}\right)^{\top}\boldsymbol{x}\right)$$

Recurrent Neural Networks

$$\boldsymbol{y}_t = g^{[y]}\left(\left(\boldsymbol{W}^{[hy]}\right)^{\top}\boldsymbol{h}_t\right)$$

$$\boldsymbol{h}_t = g^{[h]}\left(\left(\boldsymbol{W}^{[xh]}\right)^{\top}\boldsymbol{x}_t + \left(\boldsymbol{W}^{[hh]}\right)^{\top}\boldsymbol{h}_{t-1}\right)$$

Outline

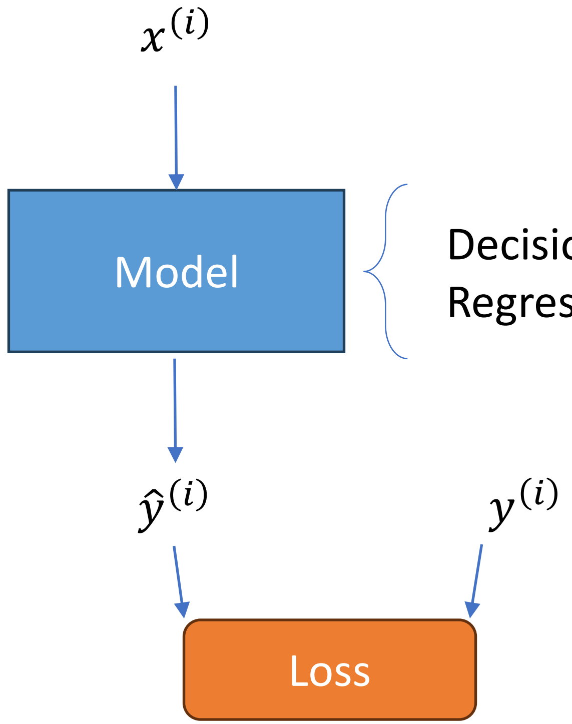
- Unsupervised Learning
- K-means clustering
  - Algorithm
  - Measuring the goodness of clusters
  - Picking the number of clusters
  - Variants
- Hierarchical clustering
  - Algorithm
  - Dendograms
  - Distance Metrics
  - Applications
- Dimensionality Reduction
  - Singular Value Decomposition (SVD)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

Outline

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

Supervised Learning

Given a set of m input-output pairs (training samples) $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, learn to make a prediction.

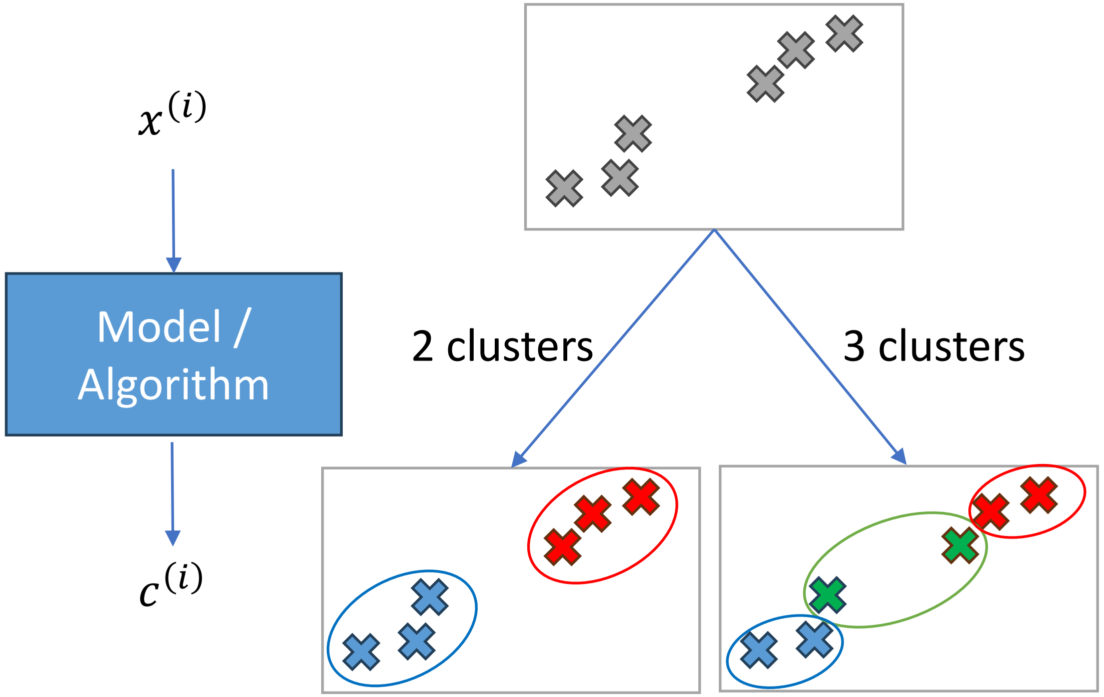


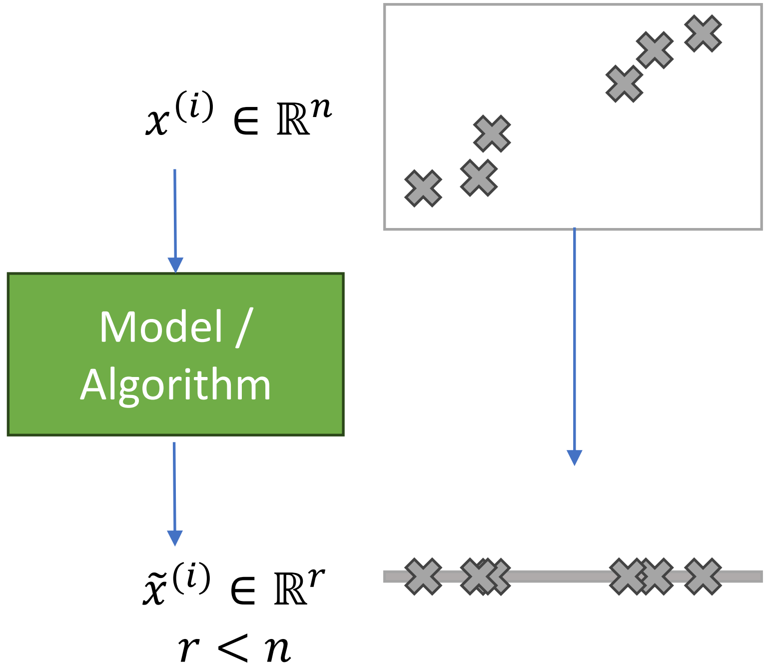
Decision Trees , Linear Regression, Logistic Regression, SVM, Neural Networks

Unsupervised Learning

Given a set of m data points $\{x^{(1)}, \ldots, x^{(m)}\}$, learn the pattern in the data.

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

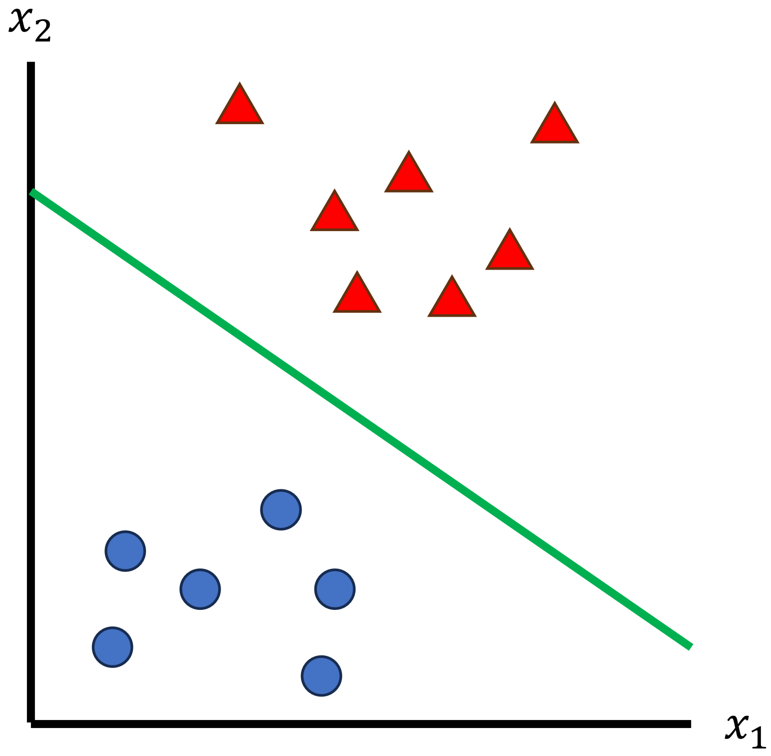




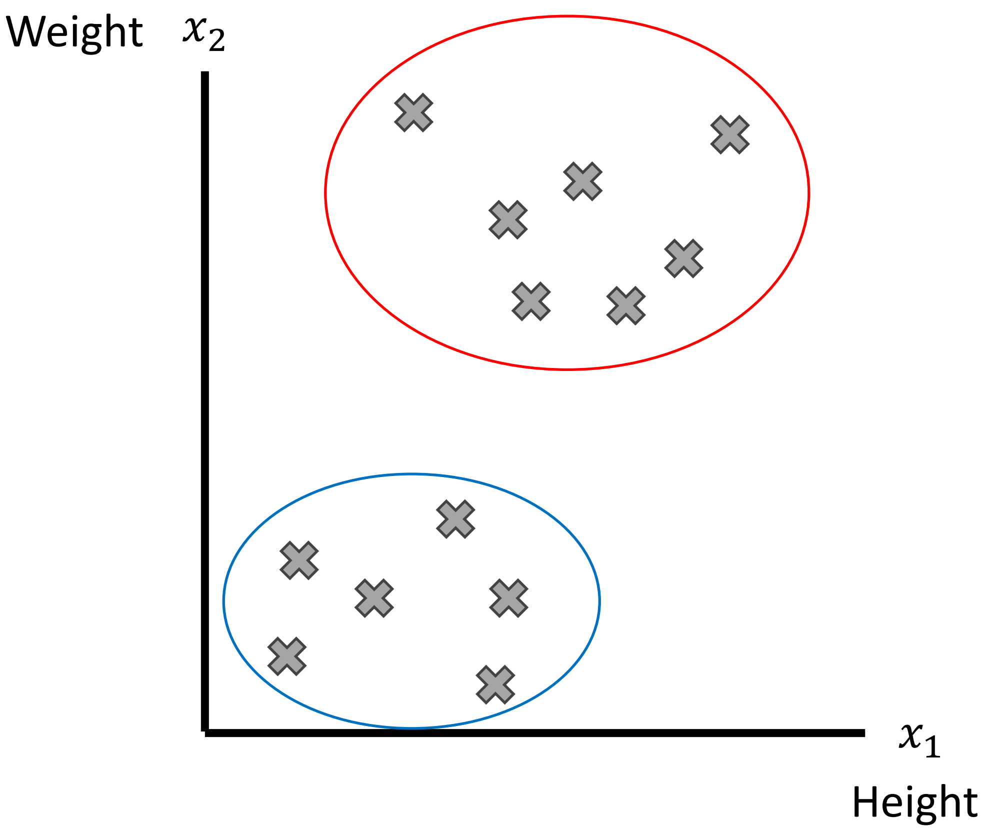
Outline

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

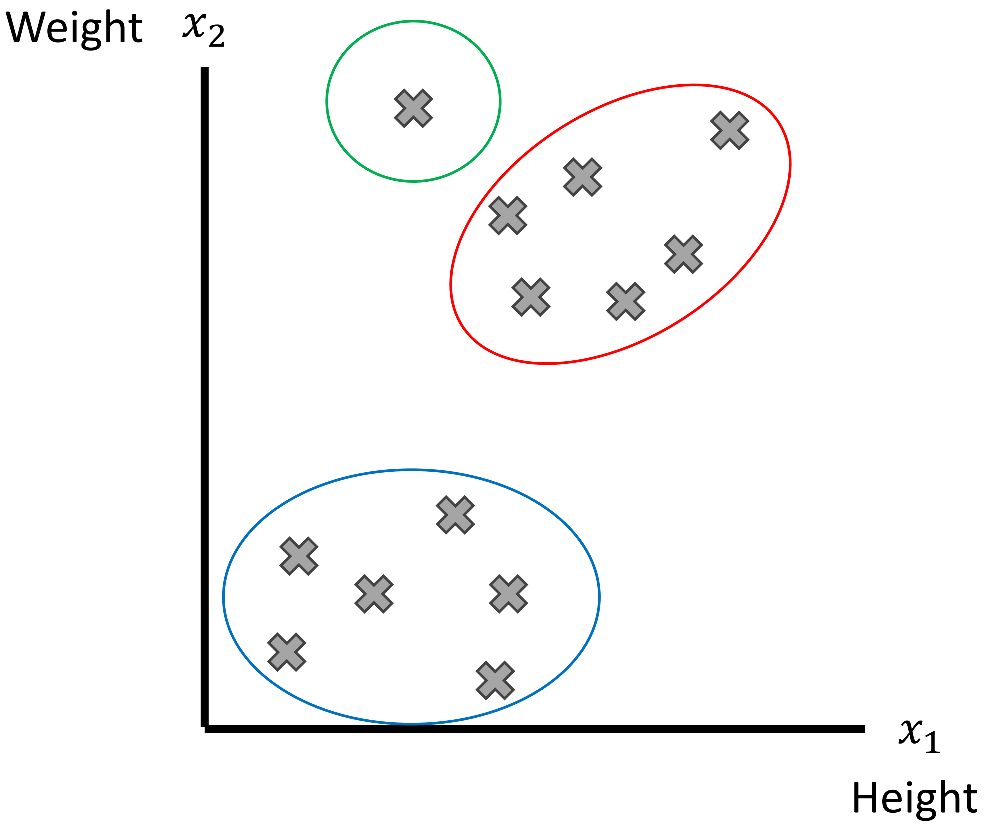
Classification



Clustering

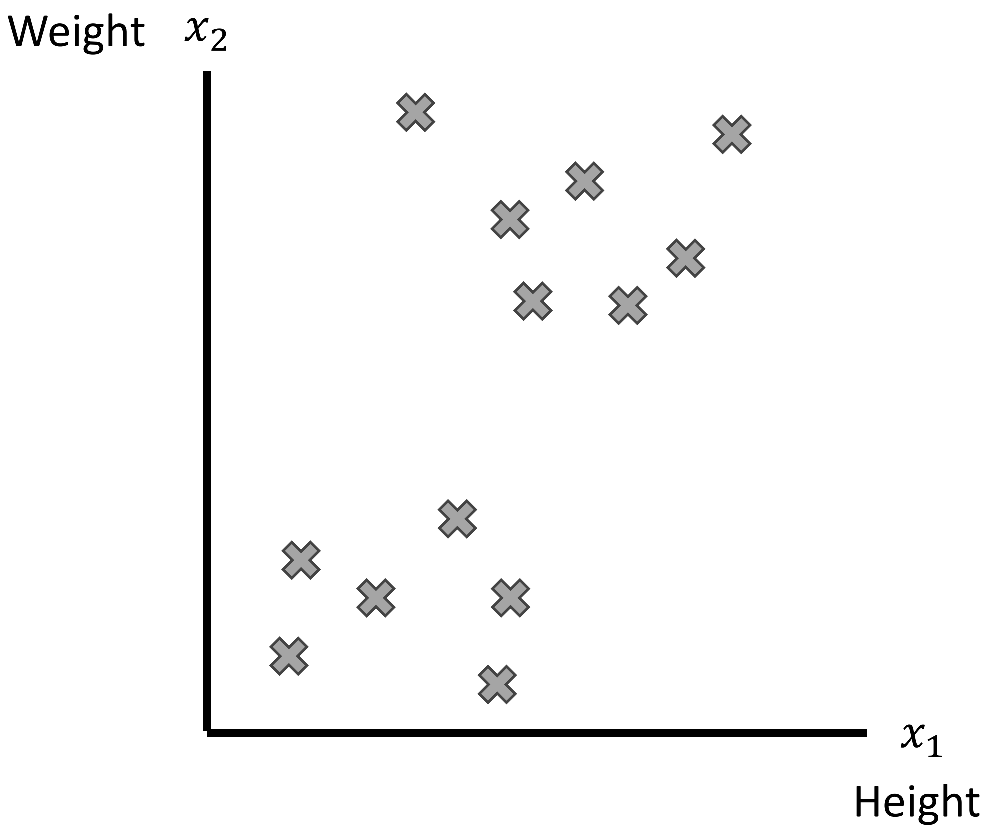


Clustering



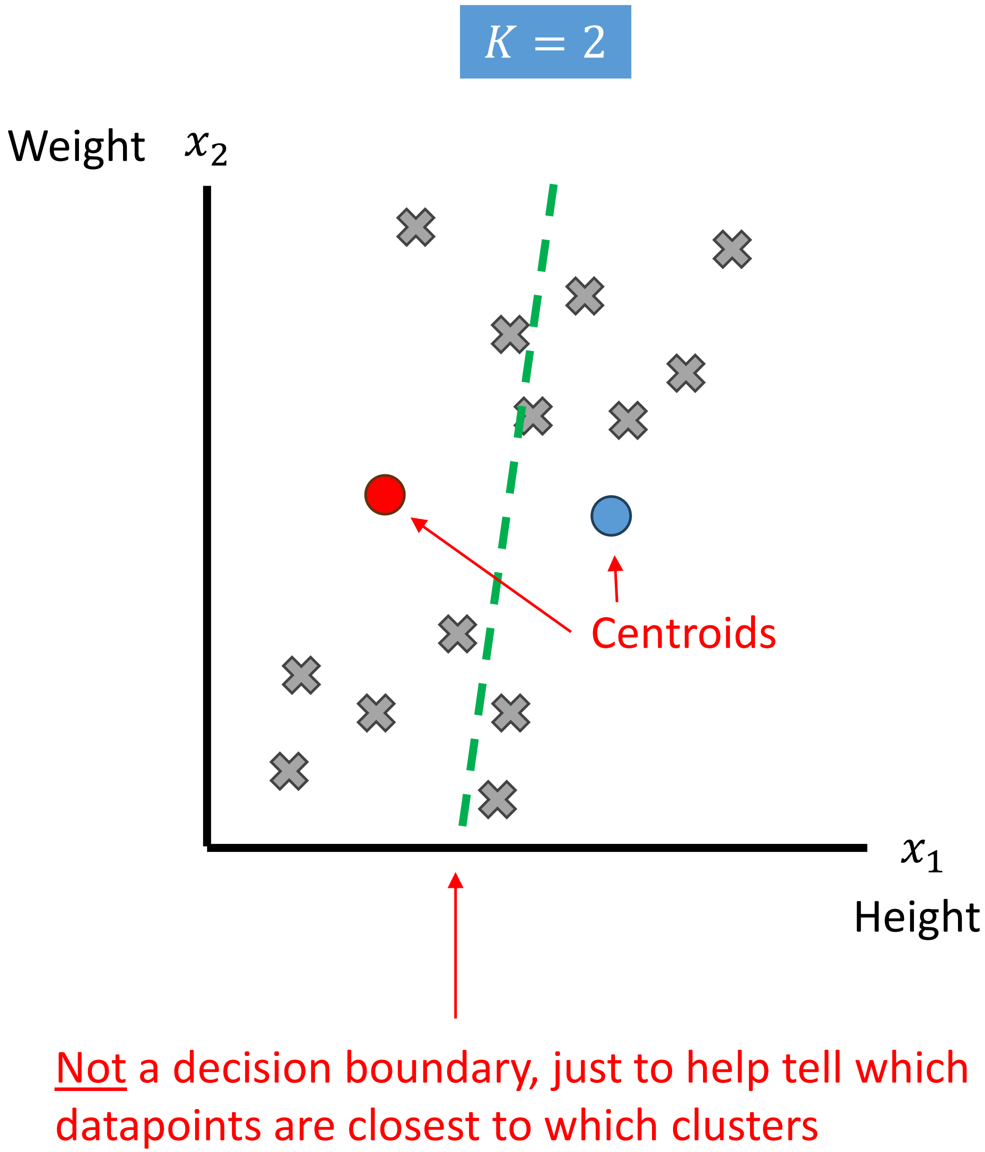
K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, m$:
    - $c^{(i)} \leftarrow$ 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



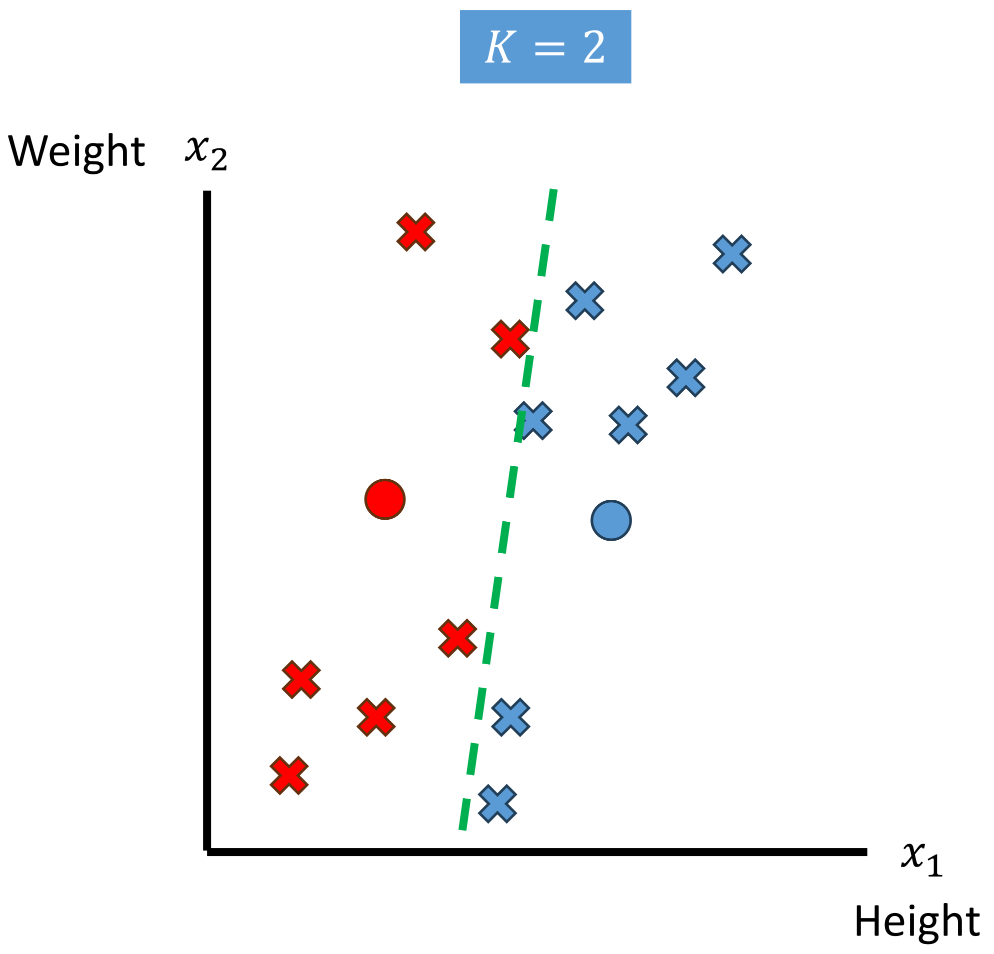
K-Means

- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, m$:
    - $c^{(i)} \leftarrow$ 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



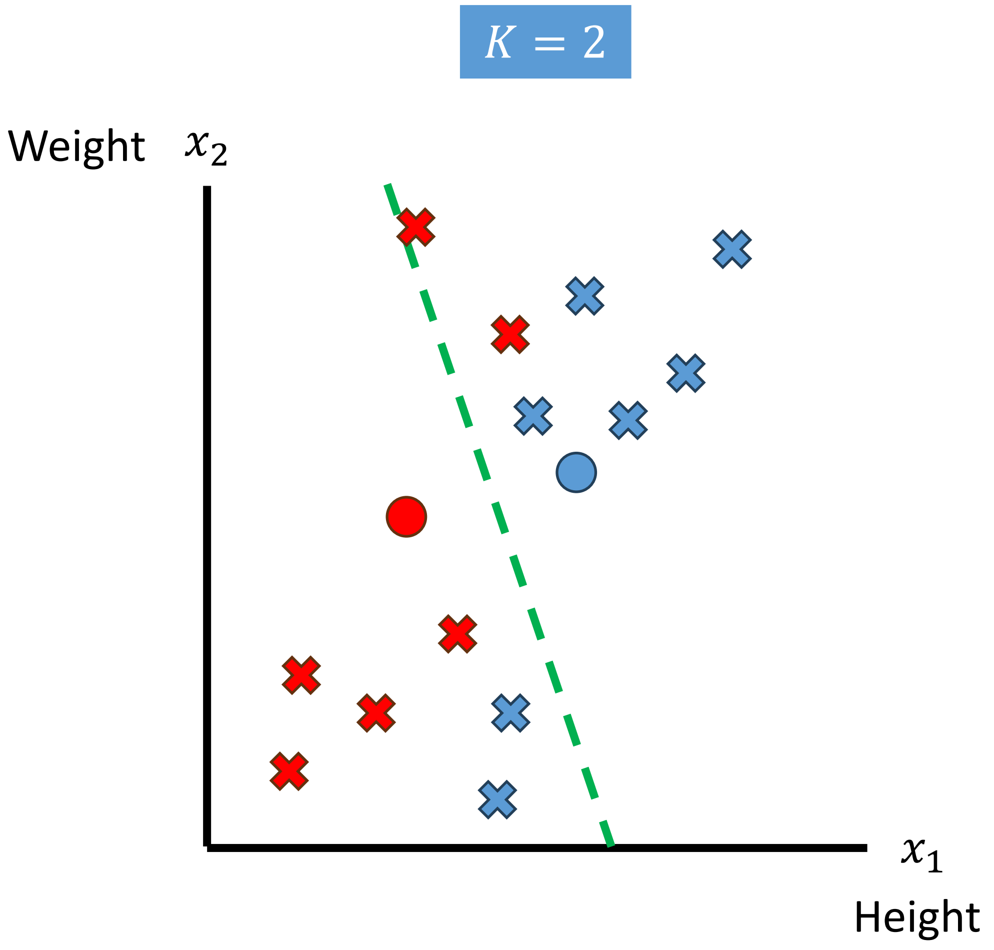
K-Means

- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, m$:
    - $c^{(i)} \leftarrow$ 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



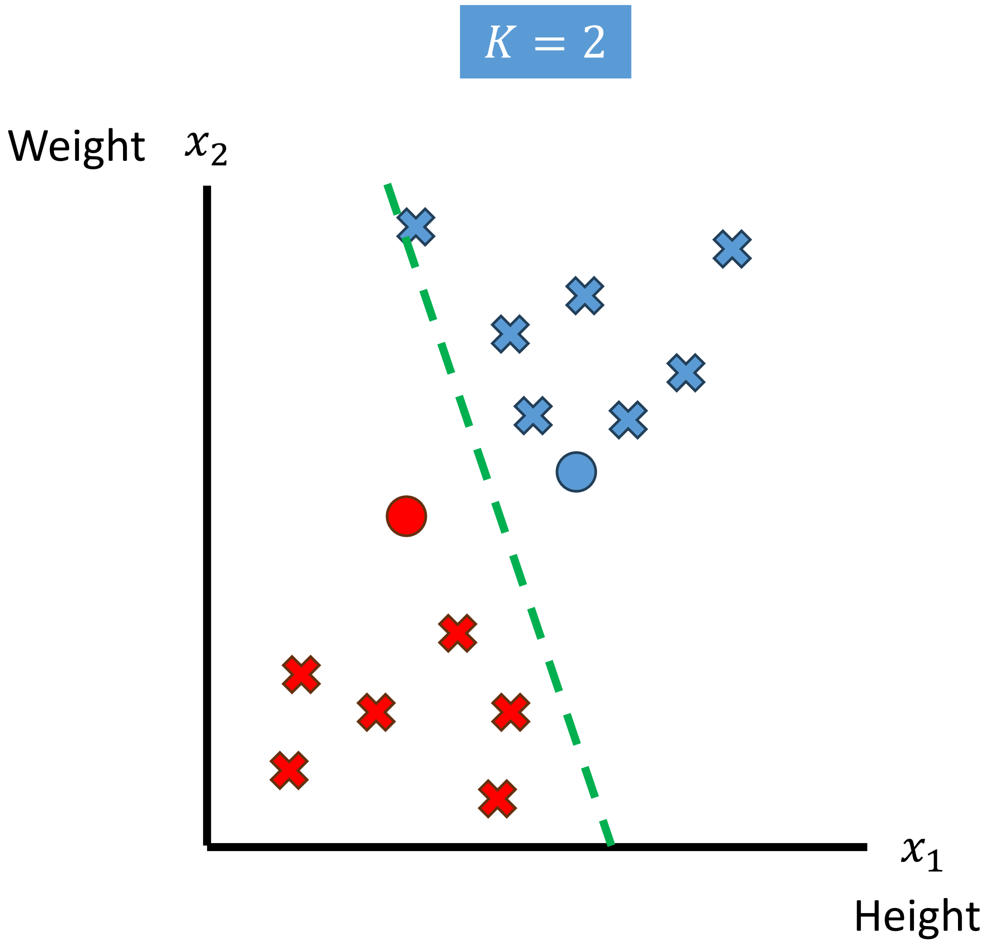
K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, m$:
    - $c^{(i)} \leftarrow$ 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



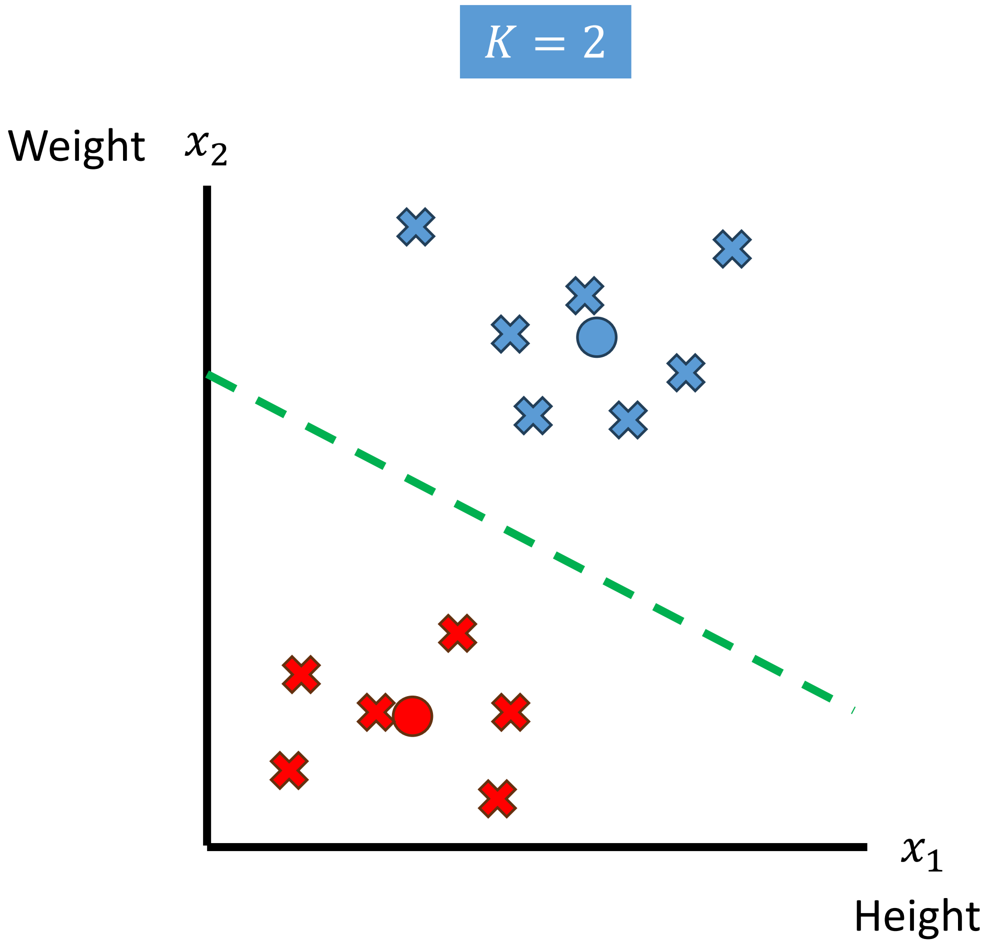
K-Means

- Randomly initialize K centroids: $\mu_1, \dots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \dots, m$:
    - $c^{(i)} \leftarrow$ 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



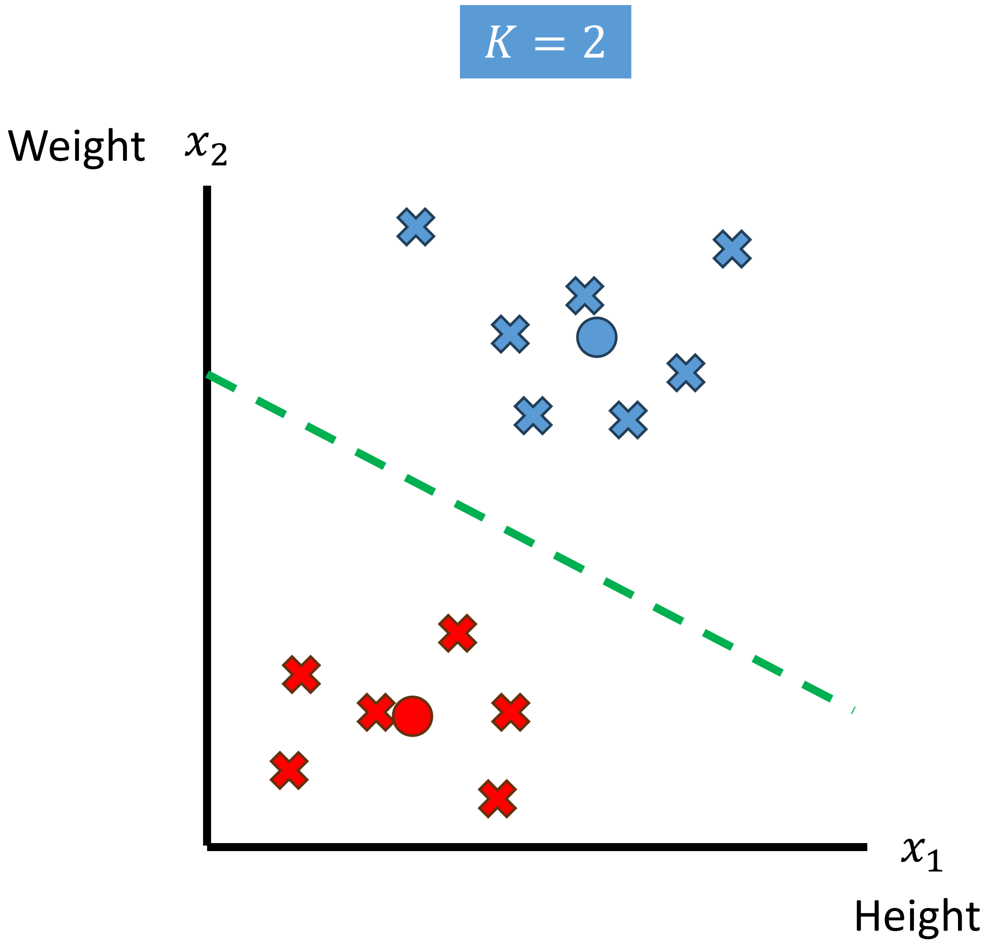
K-Means

- Randomly initialize K centroids: $\mu_1, \ldots, \mu_K$
- Repeat until convergence:
  - For $i = 1, \ldots, m$:
    - $c^{(i)} \leftarrow$ 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 $i = 1, \ldots, m$:
    - $c^{(i)} \leftarrow$ 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

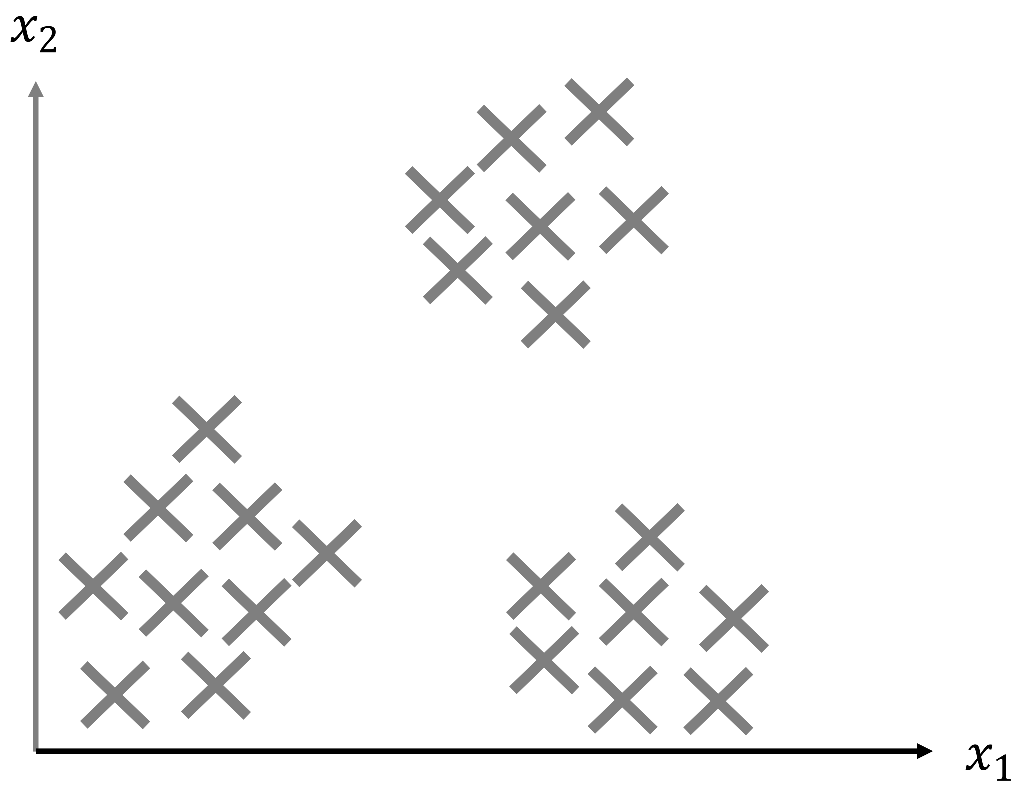


No more change: **converged!**

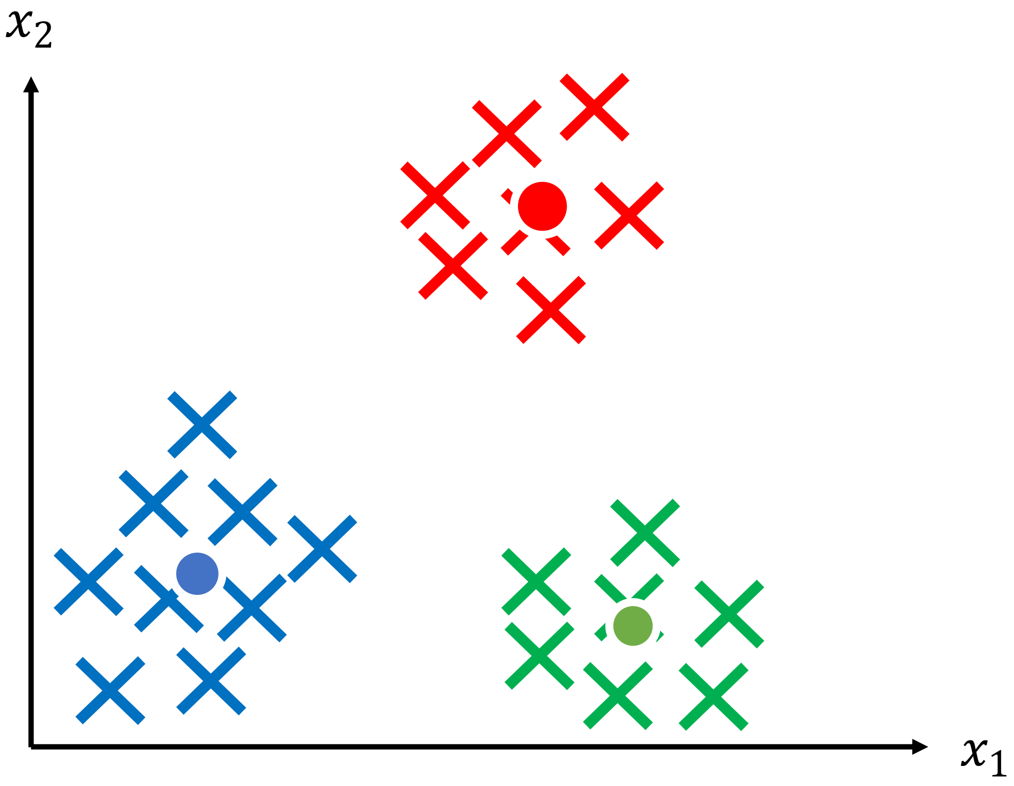
K-Means: Convergence

Tutorial 10!

K-Means: Local Optima

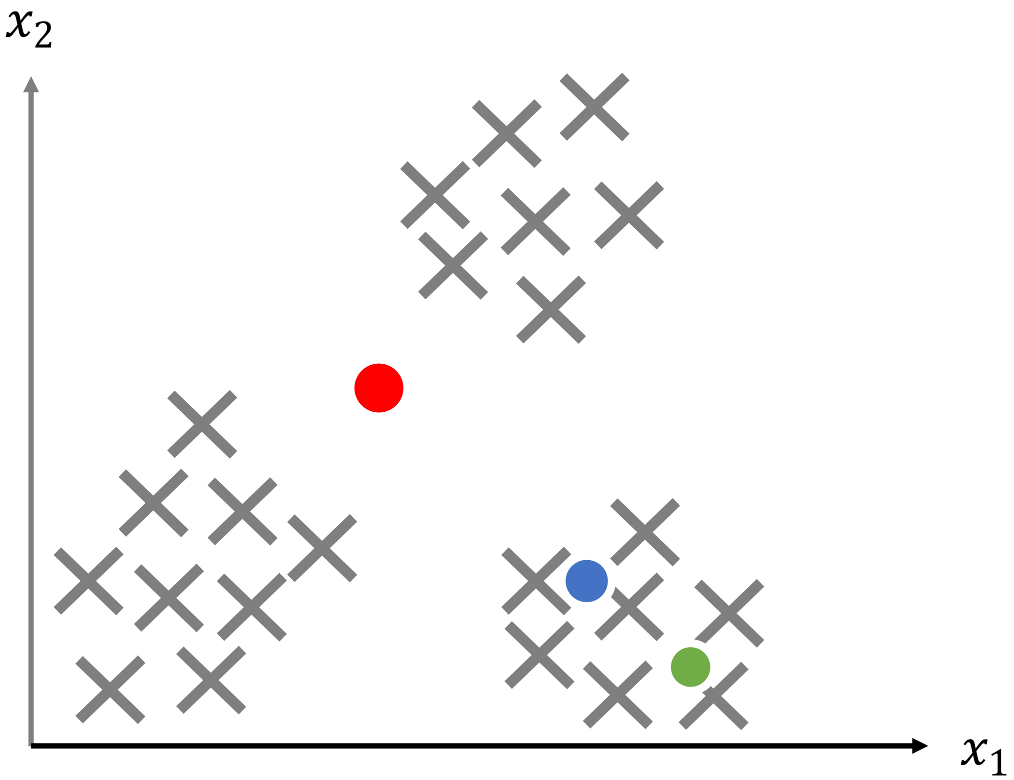


K-Means: Local Optima

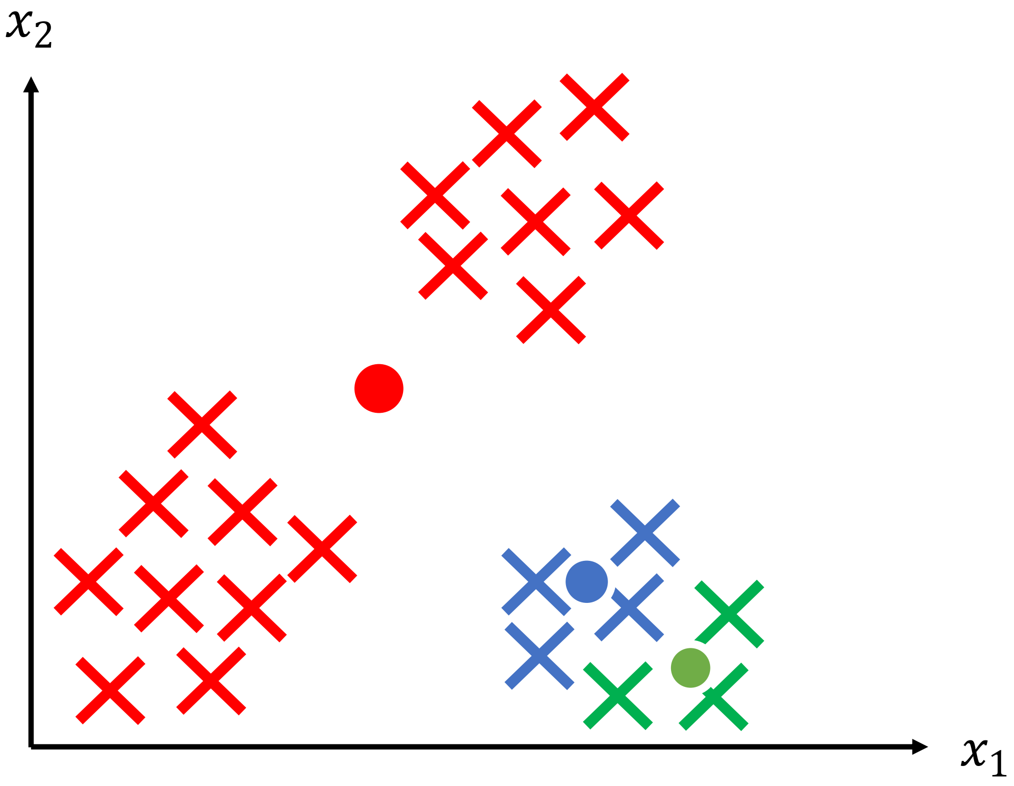


Expectations

K-Means: Local Optima



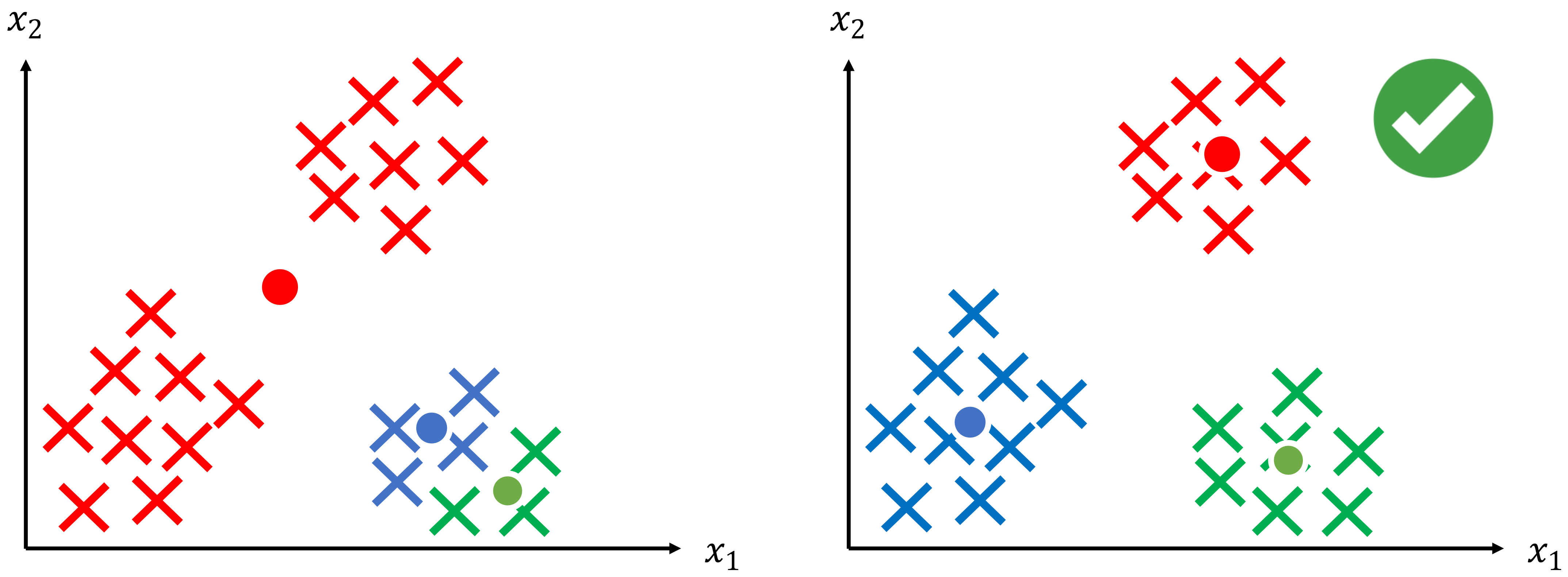
K-Means: Local Optima



(Possible) Reality

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

K-Means: Measuring the Goodness

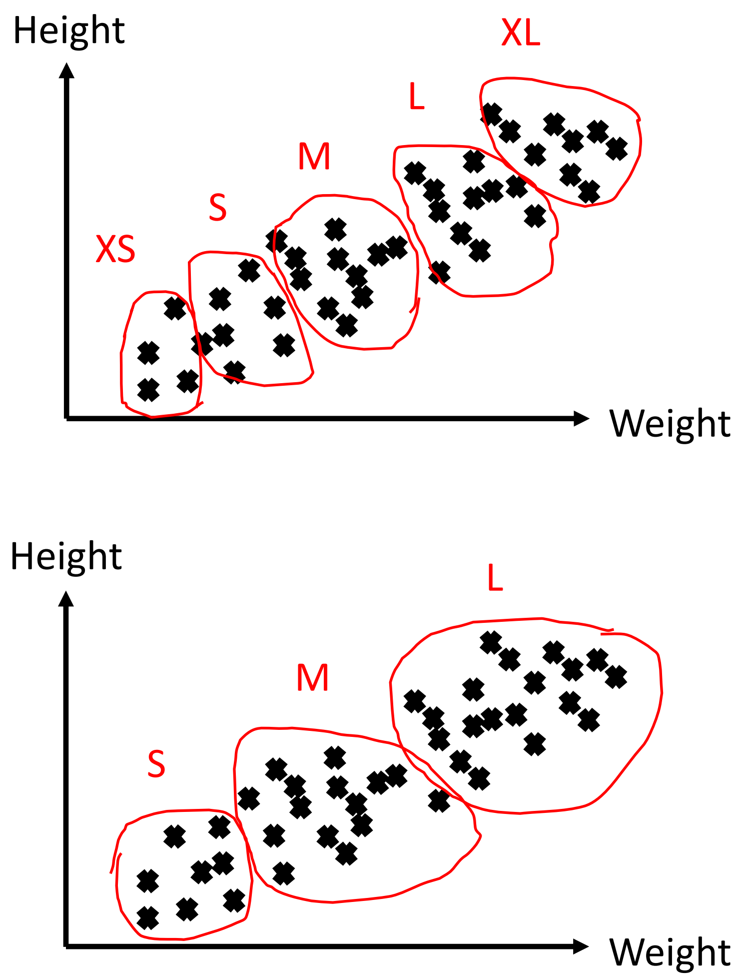


$$J\left(c^{(1)}, c^{(2)}, \cdots, c^{(m)}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \cdots, \boldsymbol{\mu}_K\right) = \frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{c^{(i)}}\right\|^2$$

K-Means: Picking the Number of Clusters

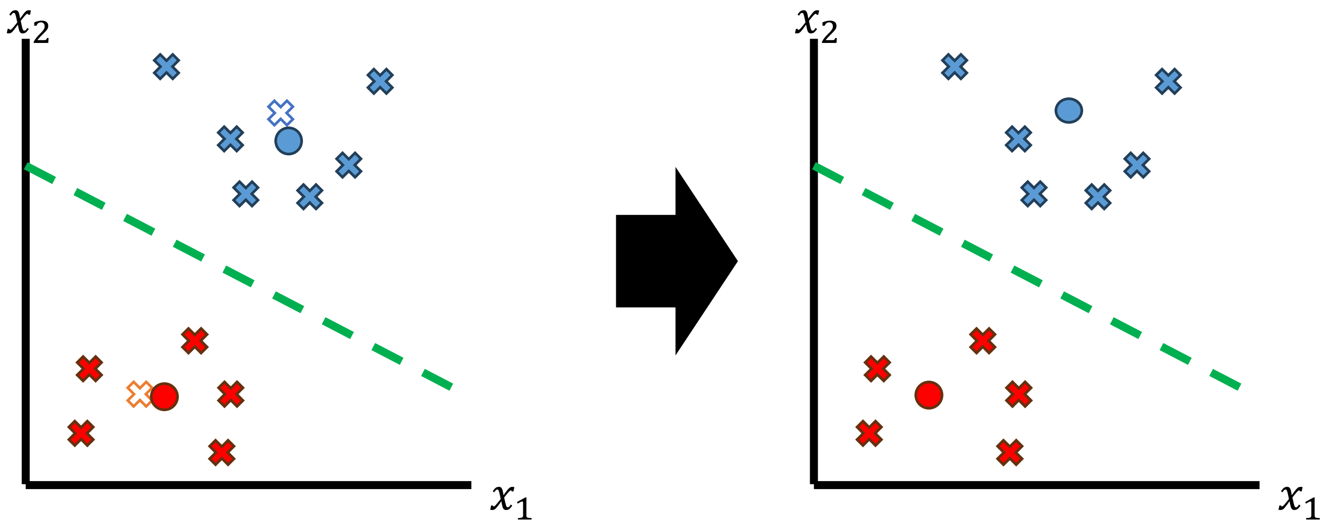
Elbow Method

Business Needs



K-Means: Variants

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



Outline

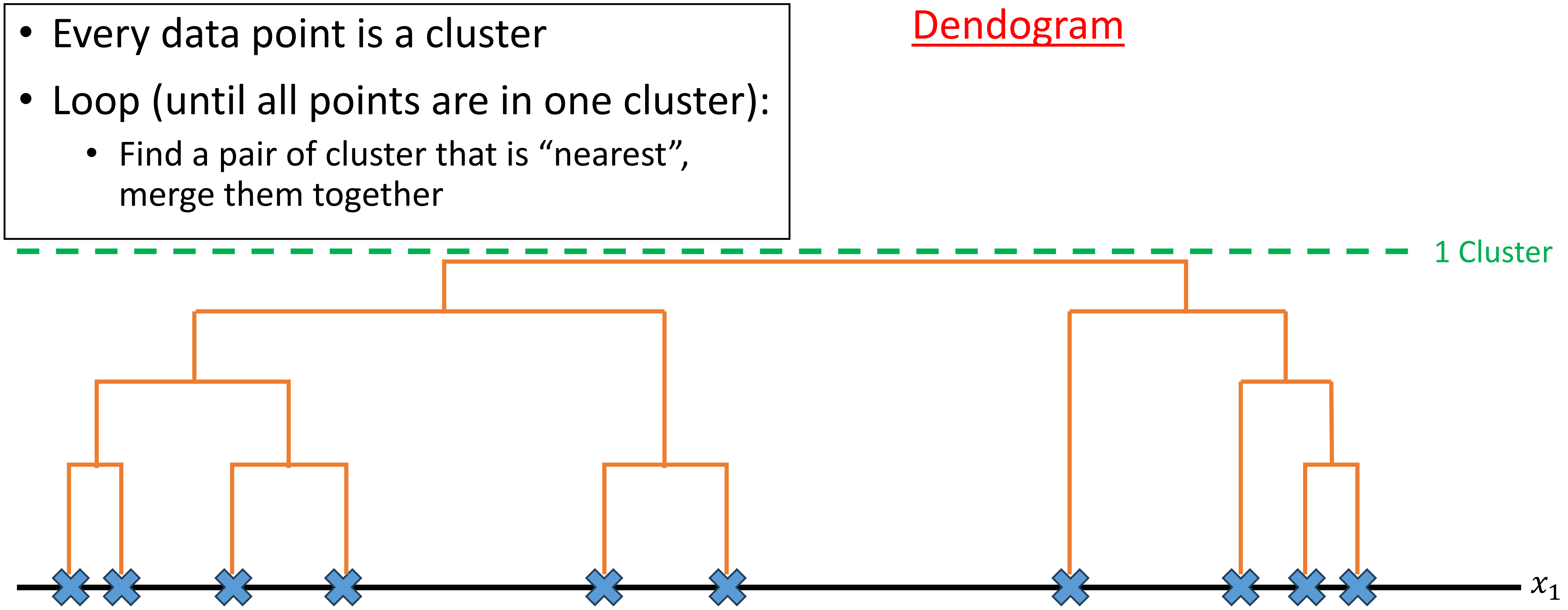
- Unsupervised Learning
- K-means clustering
  - Algorithm
  - Measuring the goodness of clusters
  - Picking the number of clusters
  - Variants
- **Hierarchical clustering**
  - Algorithm
  - Dendograms
  - Distance Metrics
  - Applications
- Dimensionality Reduction
  - Singular Value Decomposition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

Hierarchical Clustering

- Every data point is a cluster
- Loop (until all points are in one cluster):
  - Find a pair of cluster that is “nearest”, merge them together

Hierarchical Clustering: 1D

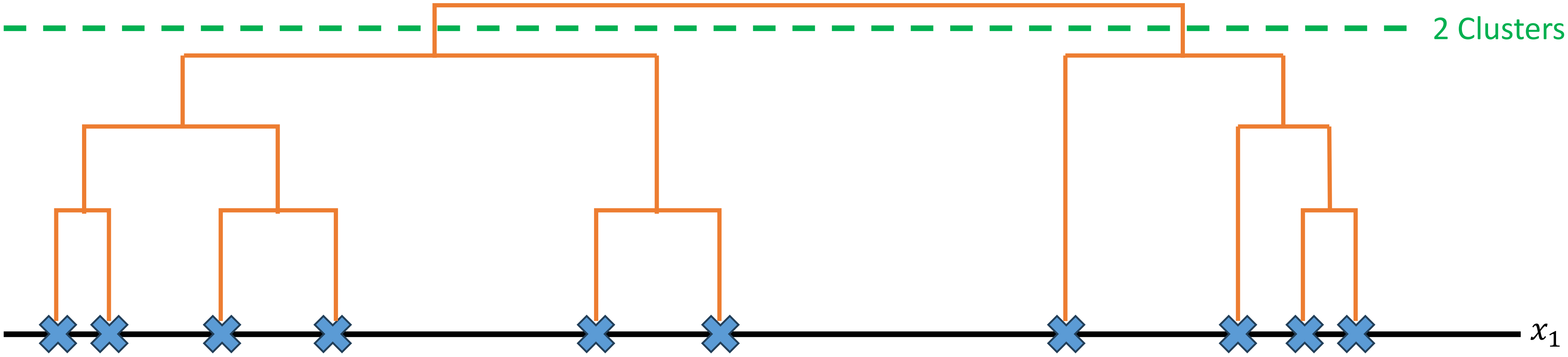
- Every data point is a cluster
- Loop (until all points are in one cluster):
  - Find a pair of cluster that is “nearest”, merge them together



Hierarchical Clustering: 1D

- Every data point is a cluster
- Loop (until all points are in one cluster):
  - Find a pair of cluster that is "nearest", merge them together

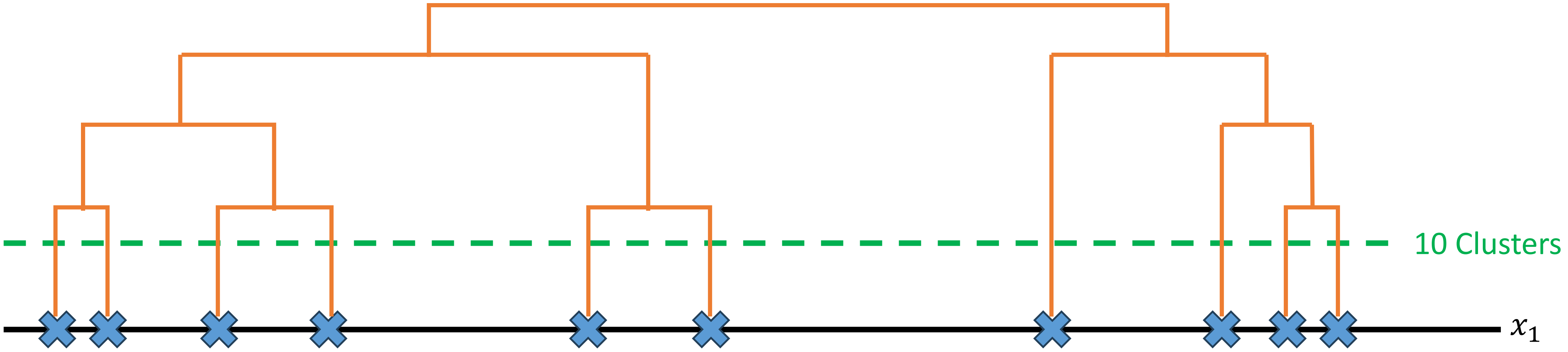
Dendogram



Hierarchical Clustering: 1D

- Every data point is a cluster
- Loop (until all points are in one cluster):
  - Find a pair of cluster that is “nearest”, merge them together

<u>Dendogram</u>

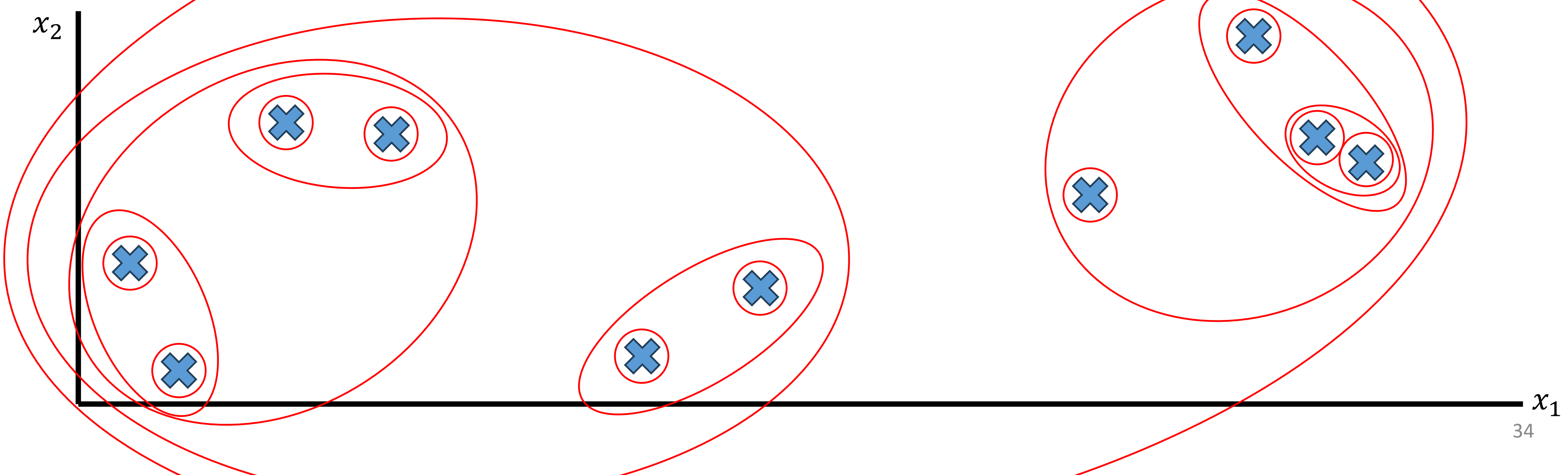


Hierarchical Clustering: 2D

How do we compute the distance between clusters?

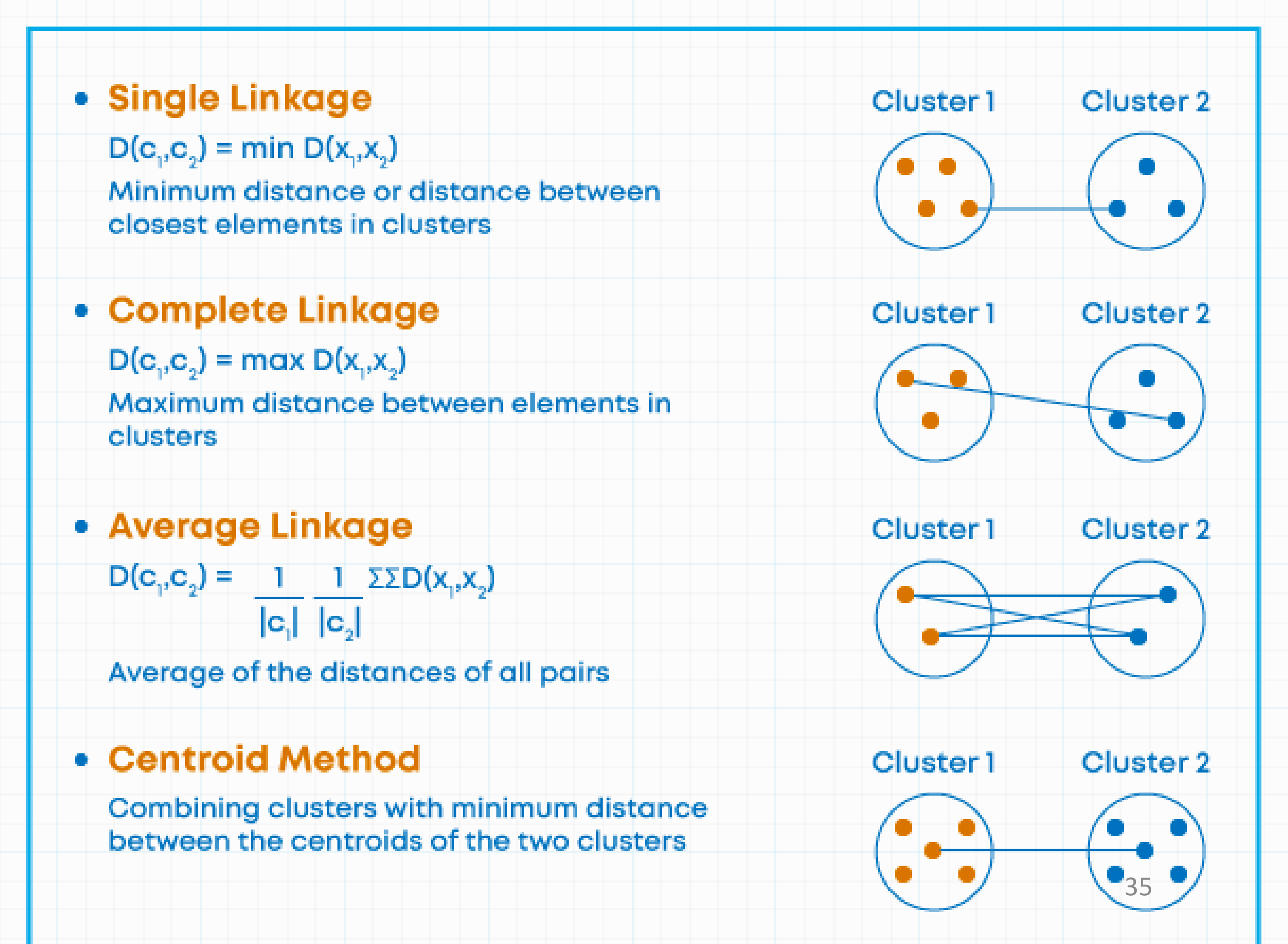
<u>Dendogram will be similar to 1D</u>

- Every data point is a cluster
- Loop (until all points are in one cluster):
  - Find a pair of cluster that is "nearest", merge them together



Hierarchical Clustering: Notes

- Many options to compute the distance between clusters
  - See image on the right
  - Euclidean distance
  - Manhattan distance
- High space and time complexity: impractical for large datasets



Hierarchical Clustering: Applications

- **Customer Segmentation:** Utilizing hierarchical clustering enables the segmentation of customers according to their purchasing behavior, preferences, or demographic data.
- **Gene Expression Analysis:** The application of hierarchical clustering can aid in the analysis of gene expression data, revealing patterns or clusters of genes exhibiting similar expression profiles.
- **Recommender Systems:** Hierarchical clustering serves as a valuable tool in constructing recommender systems, grouping similar users or items based on their preferences or behavior.
- **Social Network Analysis:** The implementation of hierarchical clustering is beneficial for analyzing social networks, uncovering communities or groups of individuals sharing similar social connections or interests.

Outline

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

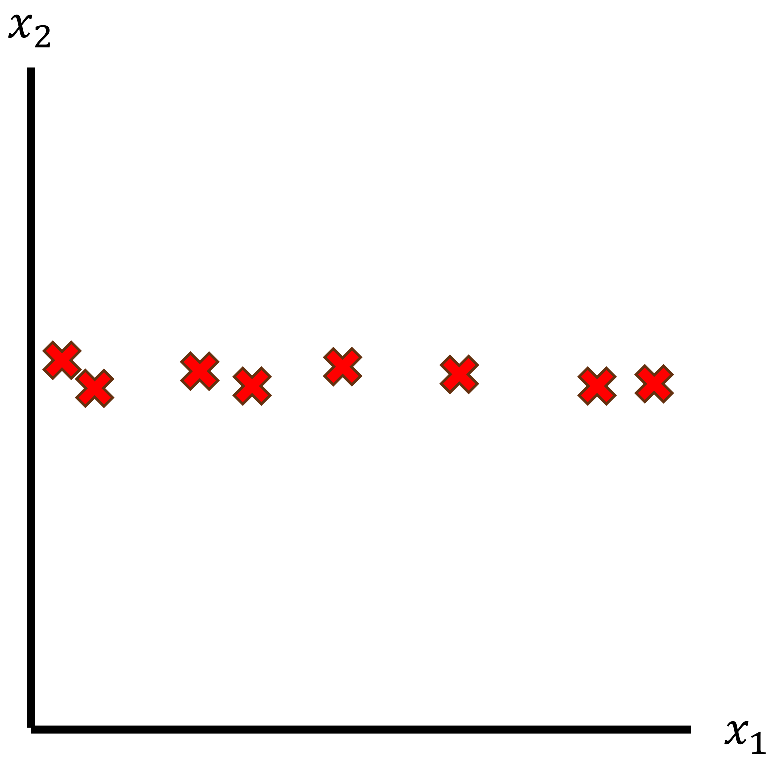
Dimensionality Reduction

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

- HD images have 1280×720=921600 features

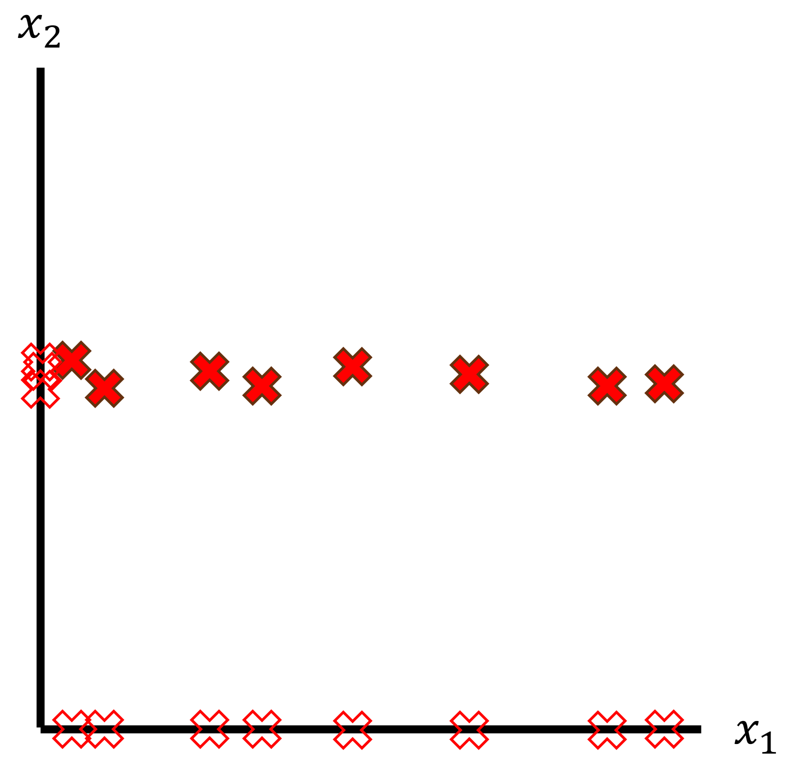
Curse of dimensionality: number of samples to learn a hypothesis class increase <u>exponentially</u> with the number of features

Dimensionality Reduction: Example 1



Dimensionality Reduction: Example 1

Reduction



Identify the “most important” components (dimensions)

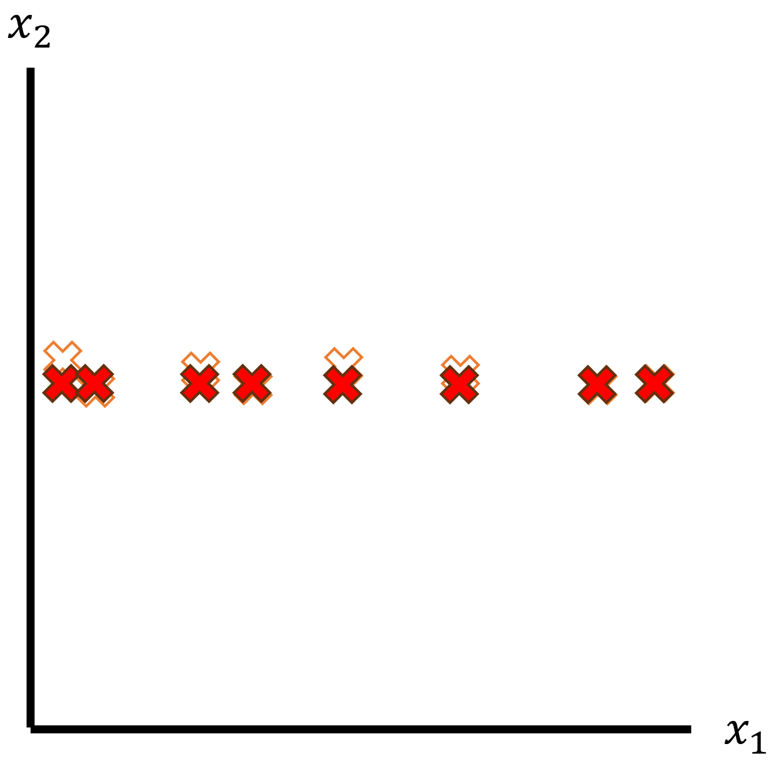
Dimensionality Reduction: Example 1

Reduction

Remove the "non-important" components (dimensions)
(Projecting the data from 2D to 1D)

Dimensionality Reduction: Example 1

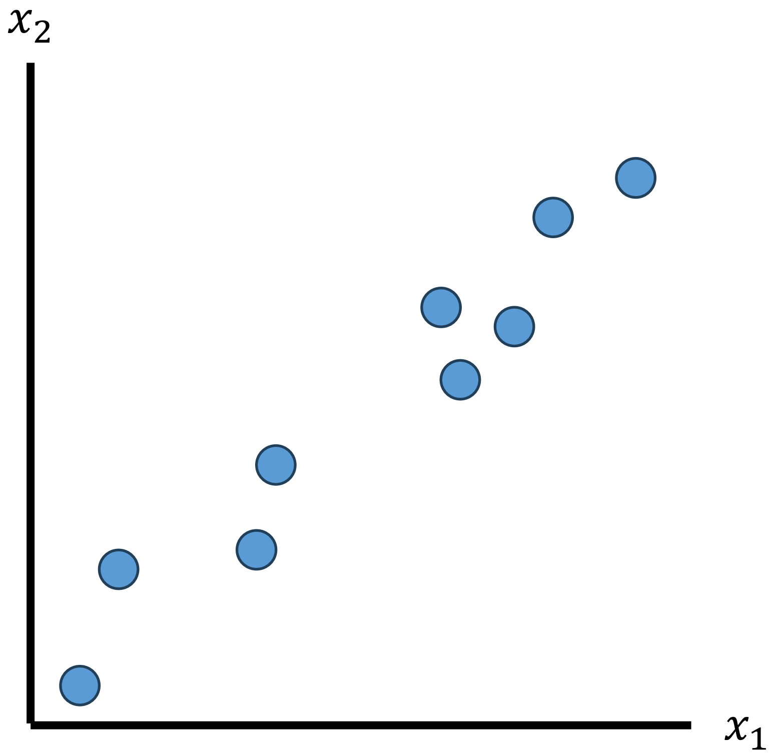
Reconstruction



Add the "non-important" components (dimensions) back

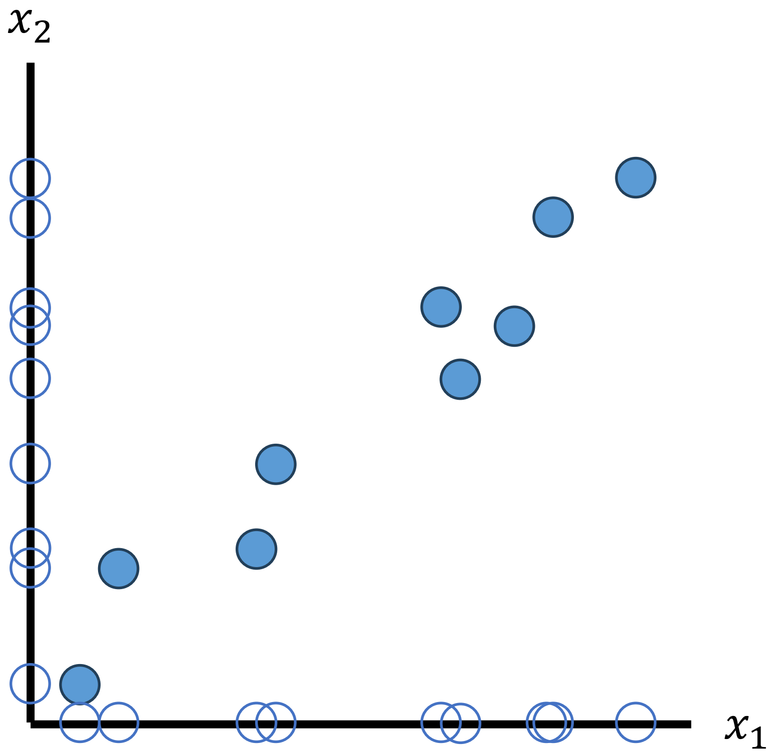
(Projecting the data from 1D to 2D)

Dimensionality Reduction: Example 2



Dimensionality Reduction: Example 2

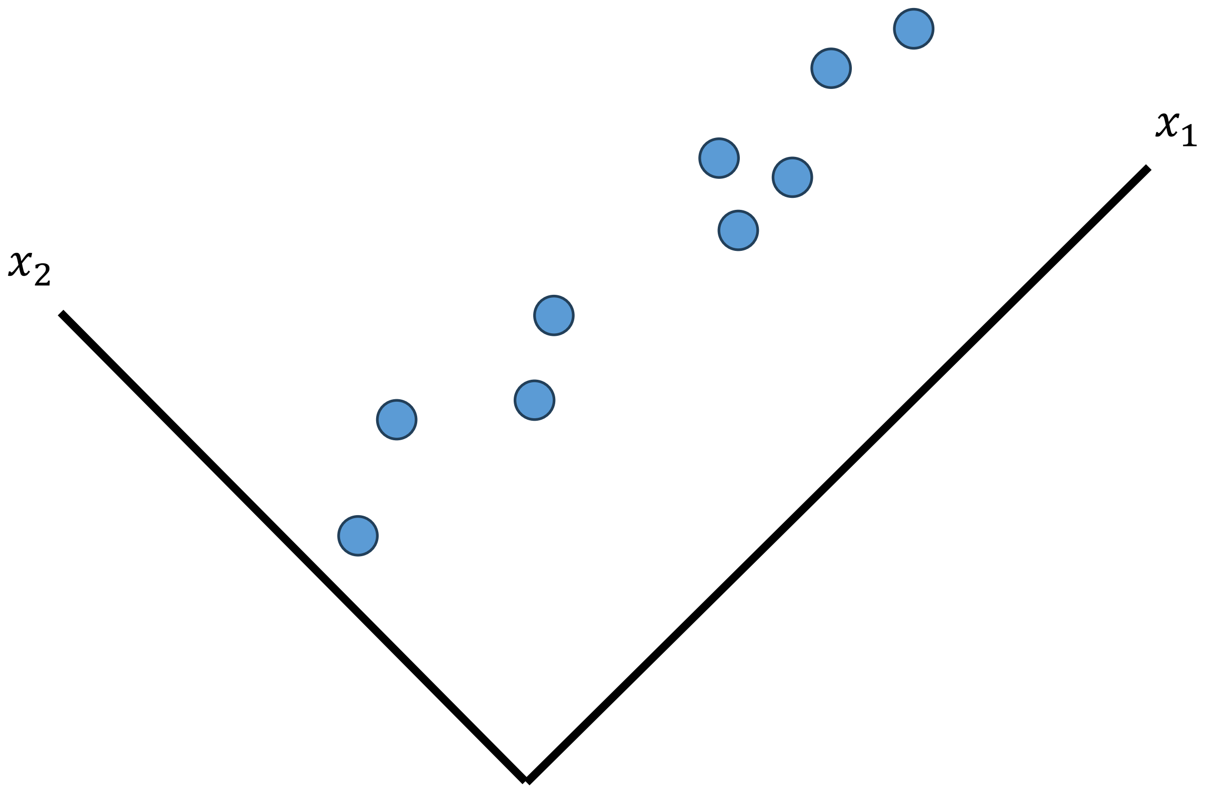
Reduction



Both components (dimensions) are equally important!

Dimensionality Reduction: Example 2

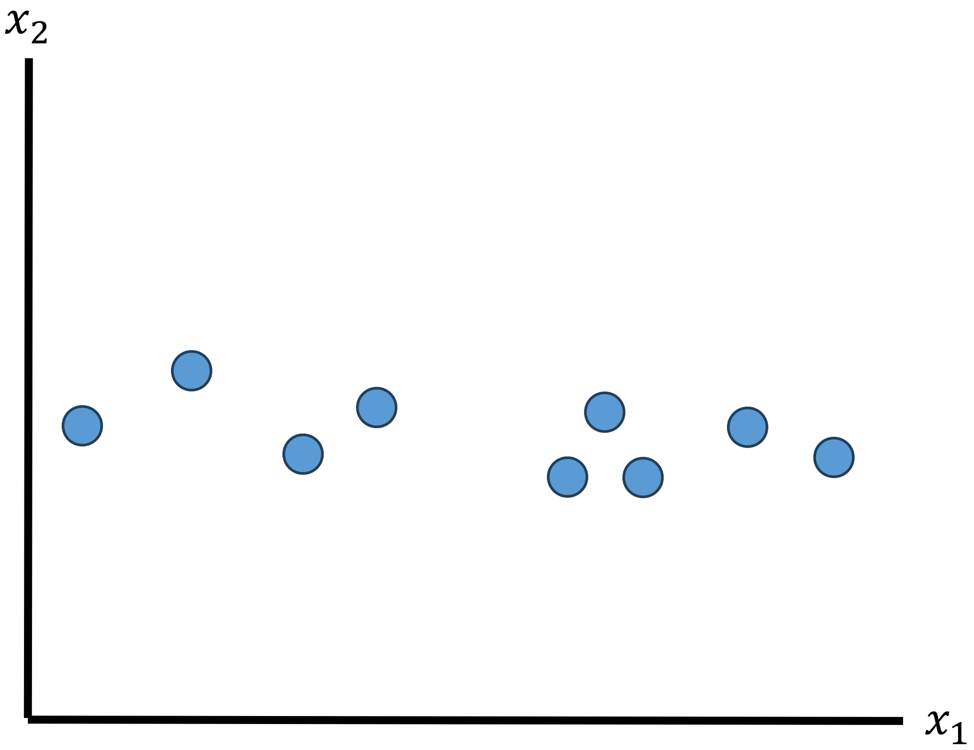
Reduction



Change the basis of the vector to remove dependence between components (dimensions)

Dimensionality Reduction: Example 2

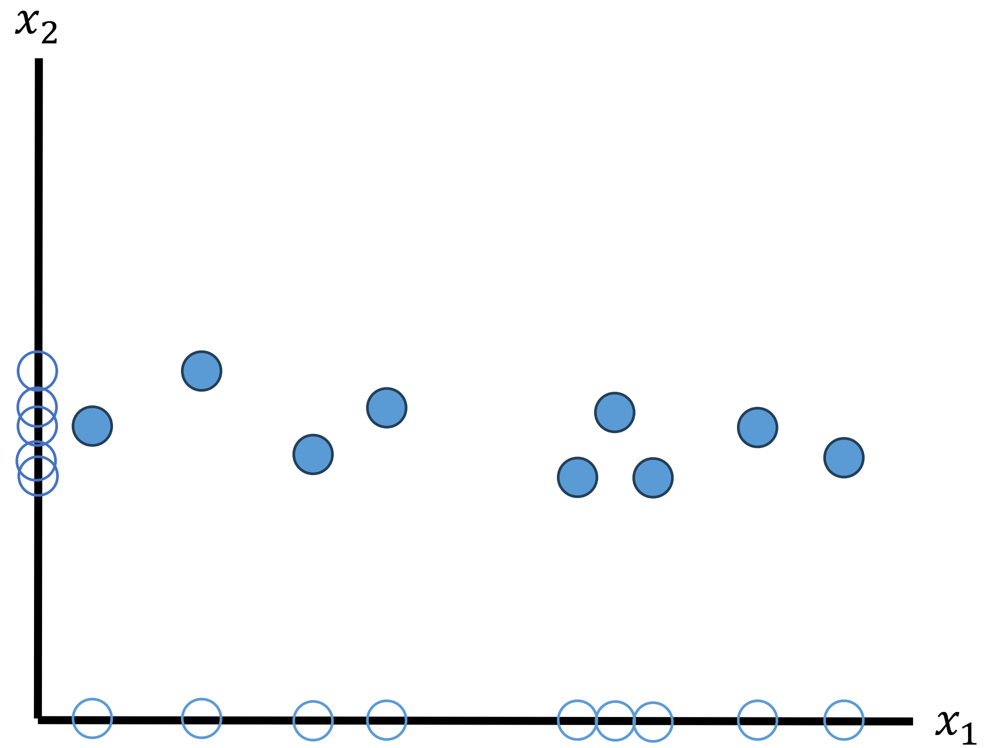
Reduction



Change the basis of the vector to remove dependence between components (dimensions)

Dimensionality Reduction: Example 2

Reduction



Identify the “most important” components (dimensions)

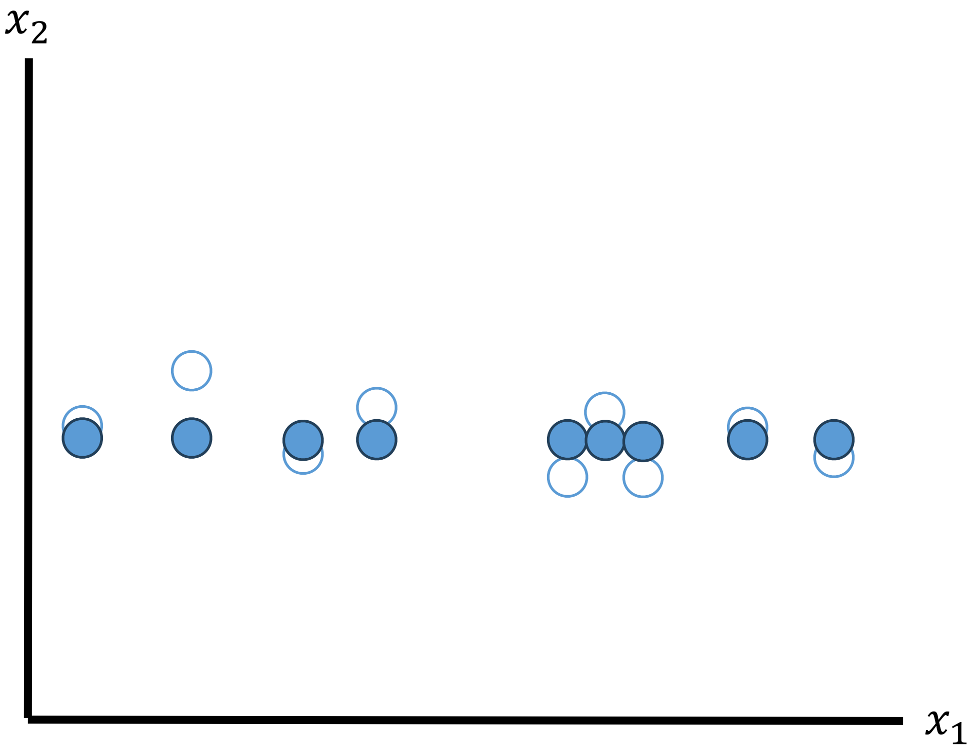
Dimensionality Reduction: Example 2

Reduction

Remove the "non-important" components (dimensions)
(Projecting the data from 2D to 1D)

Dimensionality Reduction: Example 2

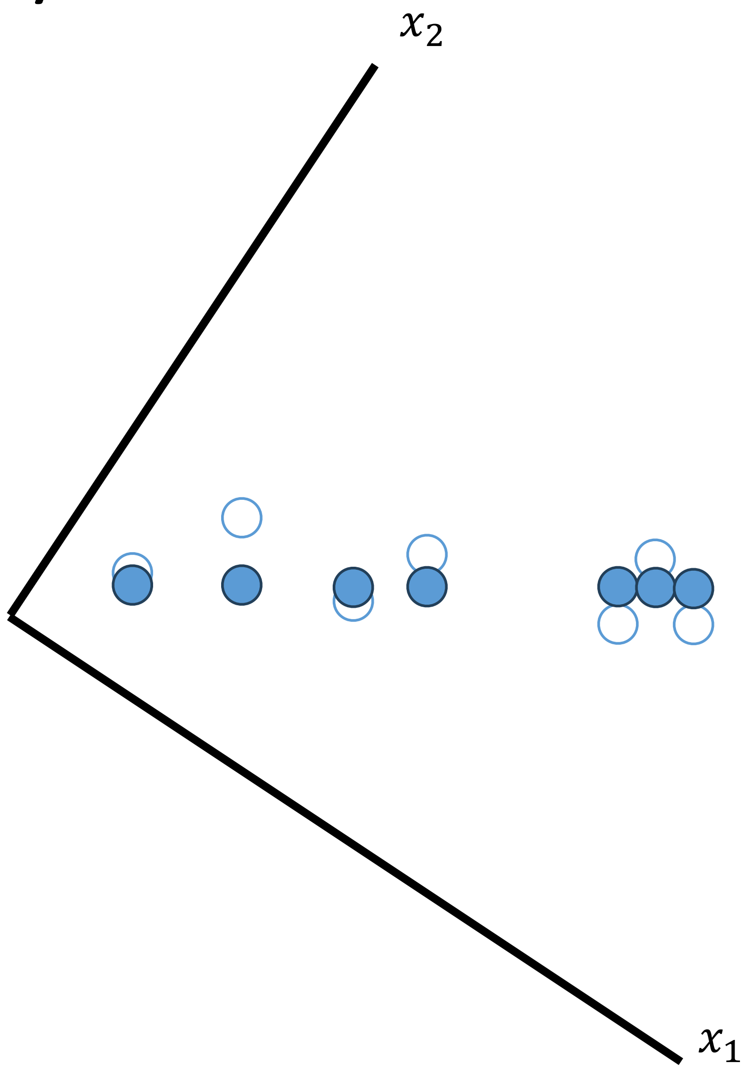
Reconstruction



Add the "non-important" components (dimensions) back
(Projecting the data from 1D to 2D)

Dimensionality Reduction: Example 2

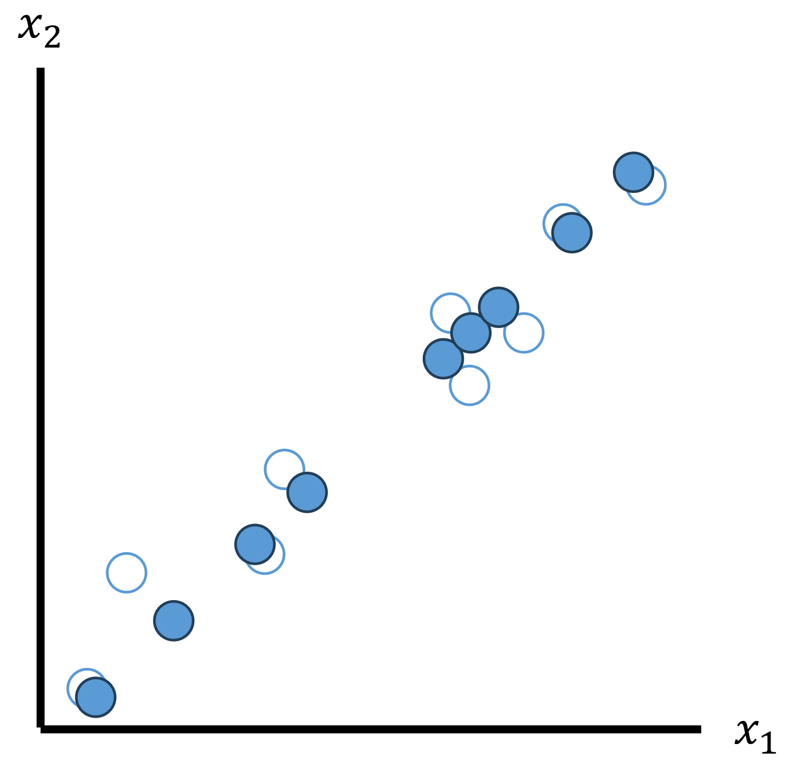
Reconstruction



Restore the basis

Dimensionality Reduction: Example 2

Reconstruction



Intuition behind:

- Singular Value Decomposition (SVD)
- Principal Component Analysis (PCA)

Restore the basis

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

Singular Value Decomposition (SVD)

Elements are ordered by importance

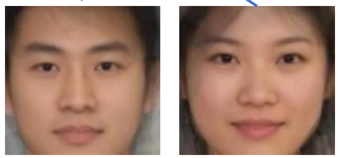
	New basis (left-singular vectors)	Basis Importance (singular values)	Combiner (right-singular vectors)
$n \times m$	$n \times n$	$n \times m$	$m \times m$

$$\boldsymbol{X} = \begin{bmatrix} | & | & & | \\ \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_m \\ | & | & & | \end{bmatrix} = \boldsymbol{U\Sigma V^T} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_1 & \boldsymbol{u}_2 & \cdots & \boldsymbol{u}_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_m \\ & \mathbf{0} & & \end{bmatrix} \begin{bmatrix} | & | & & | \\ \boldsymbol{v}_1 & \boldsymbol{v}_2 & \cdots & \boldsymbol{v}_m \\ | & | & & | \end{bmatrix}^{\boldsymbol{T}}$$

decreasing

$\boldsymbol{x}_1$ $\boldsymbol{x}_2$ $\boldsymbol{x}_m$ $\in \mathbb{R}^n$

Assume: $n \gg m$



"Face templates"

$$\begin{bmatrix} - & \boldsymbol{v}_1^T & - \\ - & \boldsymbol{v}_2^T & - \\ & \cdots & \\ - & \boldsymbol{v}_m^T & - \end{bmatrix}$$

"Mixtures" of "face templates" to make each individual face in X

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

Singular Value Decomposition (SVD)

Elements are ordered by importance

New basis (left-singular vectors) — $n \times r$

Basis Importance (singular values) — $r \times r$

Combiner (right-singular vectors) — $r \times m$

$n \times m$

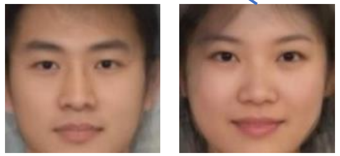
$$X = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_m \\ | & | & & | \end{bmatrix} \approx \tilde{U}\tilde{\Sigma}\tilde{V}^T = \cancel{= U\Sigma V^T} = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \cdots & u_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_m \\ & \mathbf{0} & & \end{bmatrix} \begin{bmatrix} | & | & & | \\ v_1 & v_2 & \cdots & v_m \\ | & | & & | \end{bmatrix}^T$$

decreasing

$$\begin{bmatrix} - & v_1^T & - \\ - & v_2^T & - \\ & \vdots & \\ - & v_m^T & - \end{bmatrix}$$

x_1 x_2 x_m $\in \mathbb{R}^n$

Assume: $n \gg m$



"Face templates"

"Mixtures" of "face templates" to make each individual face in X

Dimensionality reduction:
Remove some of the "non-useful" components/basis (e.g., face templates)

U unitary, $UU^T = U^TU = I$:
$UU^TX = IX = X$, $\tilde{U}\tilde{U}^TX \approx IX = X$

Reduction: $\tilde{U}^TX = Z \in \mathbb{R}^{r\times m}$
Reconstruction: $\tilde{U}Z = \tilde{X} \approx X$

Singular Value Decomposition (SVD)

- **Algorithm:**
  - Outside of the scope of this course
  - If you are really 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

Principal Component Analysis (PCA)

Statistical interpretation of SVD. Trying to capture components that maximize the *statistical variations* of the data.

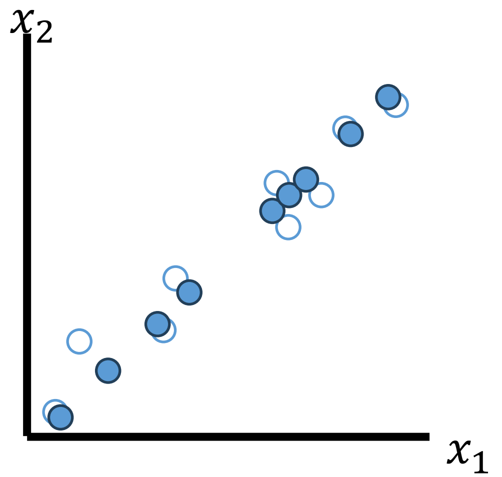
Steps:

1. Compute mean over samples: $\bar{\boldsymbol{x}} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x_i}$
2. Compute mean-centered data: $\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x_i} - \bar{\boldsymbol{x}}$
3. Create the covariance matrix of the mean-centered data: $Cov(\widehat{\boldsymbol{X}}) = \widehat{\boldsymbol{X}}^T \widehat{\boldsymbol{X}}$
4. Compute SVD for $Cov(\widehat{\boldsymbol{X}})$ to get the U matrix (new basis)

Reduction: $\widetilde{\boldsymbol{U}}^{\boldsymbol{T}} \boldsymbol{X} = \boldsymbol{Z} \in \mathbb{R}^{\boldsymbol{r \times m}}$
Reconstruction: $\widetilde{\boldsymbol{U}} \boldsymbol{Z} = \widetilde{\boldsymbol{X}} \approx \boldsymbol{X}$

PCA: Picking the Number of Components

- Average squared projection: $\frac{1}{\mathrm{m}}\sum_{i=1}^{m}\|\boldsymbol{x}_i - \widetilde{\boldsymbol{x}}_i\|^2$
- Total variation: $\frac{1}{m}\sum_{i=1}^{m}\|\boldsymbol{x}_i\|^2$
- Choose minimum k, such that



$$\frac{\frac{1}{\mathrm{m}}\sum_{i=1}^{m}\|\boldsymbol{x}_i - \widetilde{\boldsymbol{x}}_i\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|\boldsymbol{x}_i\|^2} \le \delta$$ (Usually 0.01)

"Retain at least 99% of variance in the data"

$$= 1 - \frac{\sum_{i=1}^{k}\boldsymbol{\sigma_i^2}}{\sum_{i=1}^{n}\boldsymbol{\sigma_i^2}}$$

$$\boldsymbol{X} = \boldsymbol{U\Sigma V^T}$$
$$\boldsymbol{X^T X} = \left(\boldsymbol{U\Sigma V^T}\right)^{\boldsymbol{T}}\boldsymbol{U\Sigma V^T}$$
$$= \boldsymbol{V\Sigma^T U^T U\Sigma V^T}$$
$$= \boldsymbol{V\Sigma^T \Sigma V^T}$$

Σ

$$= \boldsymbol{V}\begin{bmatrix} \sigma_1^2 & & \\ & \sigma_2^2 & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix}\boldsymbol{V^T}$$

U

Where to get this?

Equivalent to $\frac{\sum_{i=1}^{k}\boldsymbol{\sigma_i^2}}{\sum_{i=1}^{n}\boldsymbol{\sigma_i^2}} \ge 1 - \delta$ (Usually 0.99)

PCA: Applications

- Compression: save space, speed up learning
- Visualization (k=2 or 3)

Summary

- Unsupervised Learning: learn the pattern of the data without label
- K-means clustering
  - Algorithm: find **centroids** based on the data points, **assign each point to the closest centroid**
  - Measuring the goodness of clusters: distance of each point to their centroid
  - Picking the number of clusters: **elbow method**, business needs
  - Variants: **K-medoids**, etc
- Hierarchical clustering
  - Algorithm: **each point is a cluster**, connect a pair of "closest" clusters, repeat
  - Dendograms: a way to **visualize** hierarchical clusters
  - Distance Metrics: min, max, average, etc
  - Applications
- Dimensionality Reduction: finding **new basis** that best captures the data
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA): **statistical interpretation** of SVD

Coming Up Next Week

- **AI & Ethics**
- **Recap of the entire materials**
- **Details on the final assessment**

To Do

- **Lecture Training 11**
  - +100 Free EXP
  - +50 Early bird bonus
- **Problem Set 7 is due <u>today</u>!**
- **Problem Set 8 is out!**