COMP 432 Machine Learning

Clustering

Computer Science & Software Engineering
Concordia University, Fall 2024

Course Evaluation

- Please, fill out the **course evaluation form**.
- This is the official course evaluation conducted by the Centre for Teaching and Learning (CTL).
- It is totally **anonymous**. Please, take 10 minutes to fill out this important form.

Instructions:

- Open the browser, go to "My CU Account" (https://hub.concordia.ca/students/account.html) and login
- On the left side menu, click on "Academic > Course Evaluation"
- Choose a course to evaluate and click on "Evaluate On-line"

Summary of the last episode....

We have seen many **supervised learning** techniques:

- Linear Models (linear regression, logistic regression, multiclass logistic regression)
- Neural Networks (MLPs, CNNs, RNNs)
- Support Vector Machines (linear SVMs, Non-linear SVMs)
- Decision Trees, Random Forest, Boosting
- K-nearest neighbor, Naive Bayes

Today we are going to start discussing **unsupervised learning:**

- Clustering
- Density Estimation

Unsupervised Learning

- In **supervised learning**, we learn from data with **labels**.

  ↳ Our dataset is composed of both input features and labels:

  D = {($\mathbf{x}_1$,y_1), ($\mathbf{x}_2$,y_2), …., ($\mathbf{x}_N$,y_N)}

  Supervised learning is **learning by "examples"**.

- In **unsupervised learning**, we try to learn from data **without labels**.

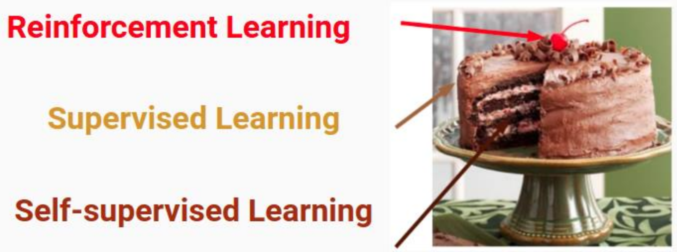
  ↳ Our dataset is composed of input features only:

  D = {$\mathbf{x}_1$,$\mathbf{x}_2$,…, $\mathbf{x}_N$}

  Supervised learning is **learning by "observation"**.

Unsupervised Learning

- There is a consensus on the importance of unsupervised learning in building **intelligent machines**:



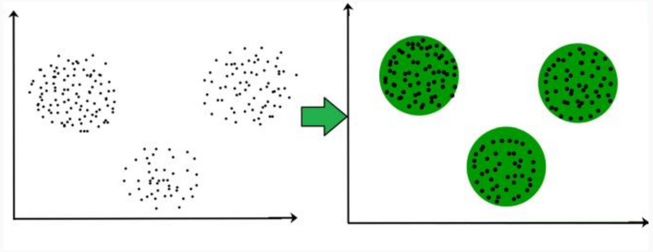
If intelligence is a cake, the bulk of the cake is **unsupervised learning**, the icing on the cake is **supervised learning**, and the cherry on the cake is **reinforcement learning (RL)**.

- We indeed learn a lot just by **observing** the world around us.

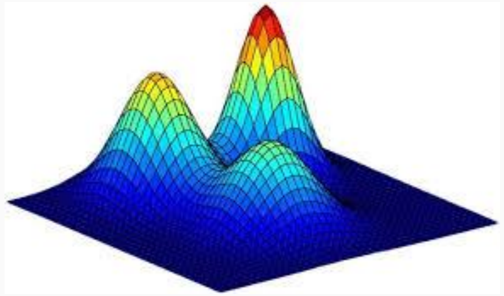
- Modern machine learning methods (e.g., deep neural network pre-trained with self-supervised learning) achieve state-of-the-art performance in many applications (e.g, computer vision, speech recognition, etc) if we **combine unsupervised and supervised learning** modalities.

Unsupervised Learning

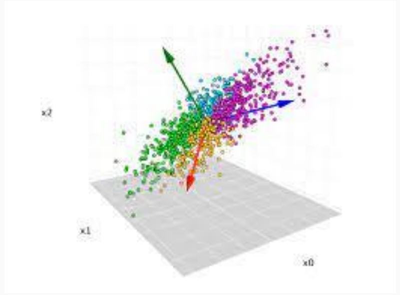
- There are different types of unsupervised learning.
- Some examples are:



Clustering



Probability Density Estimation



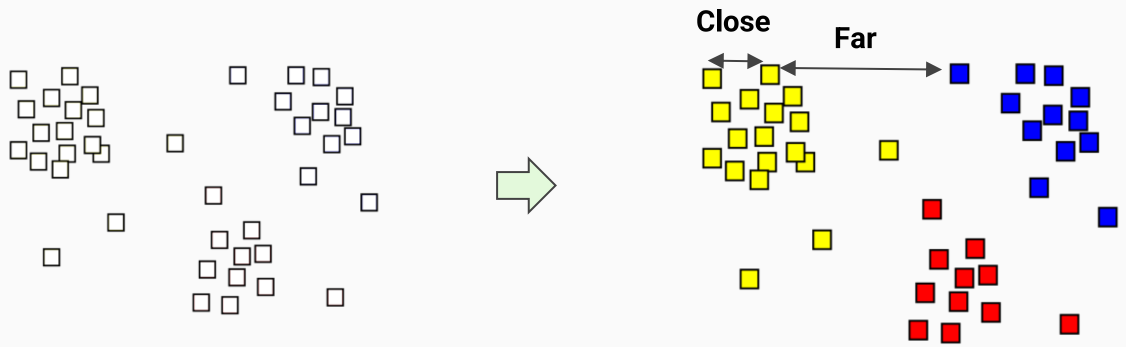
Feature transformation

- In the last part of this course, we will see unsupervised techniques belonging to these categories.

Clustering

- **Input**: an unlabeled data set **D** = {$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$} composed of D-dimensional features.

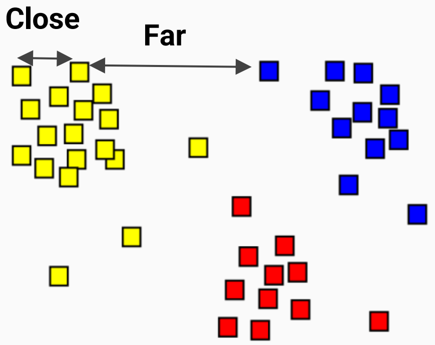
- **Goal**: group the data into "clusters" based on some measure of distance



We cannot assign a **semantic label** to each cluster (e.g, cat, dog, bird) but a least we can say *sample i* is similar to *sample j*.

A cluster is a group of "similar" data points that are close in the feature space according to a certain measure of **distance.**

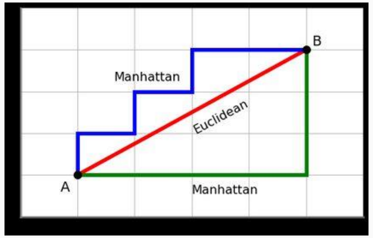
Clustering



- Different distance measures can be used. For instance:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2}$$ *Euclidean Distance*

$$d(\mathbf{x},\mathbf{y}) = ||\mathbf{x}-\mathbf{y}||^2 = \sum_{i=1}^{D}(x_i - y_i)^2$$ *Squared Euclidean Distance*



$$d(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{D}|(x_i - y_i)|$$

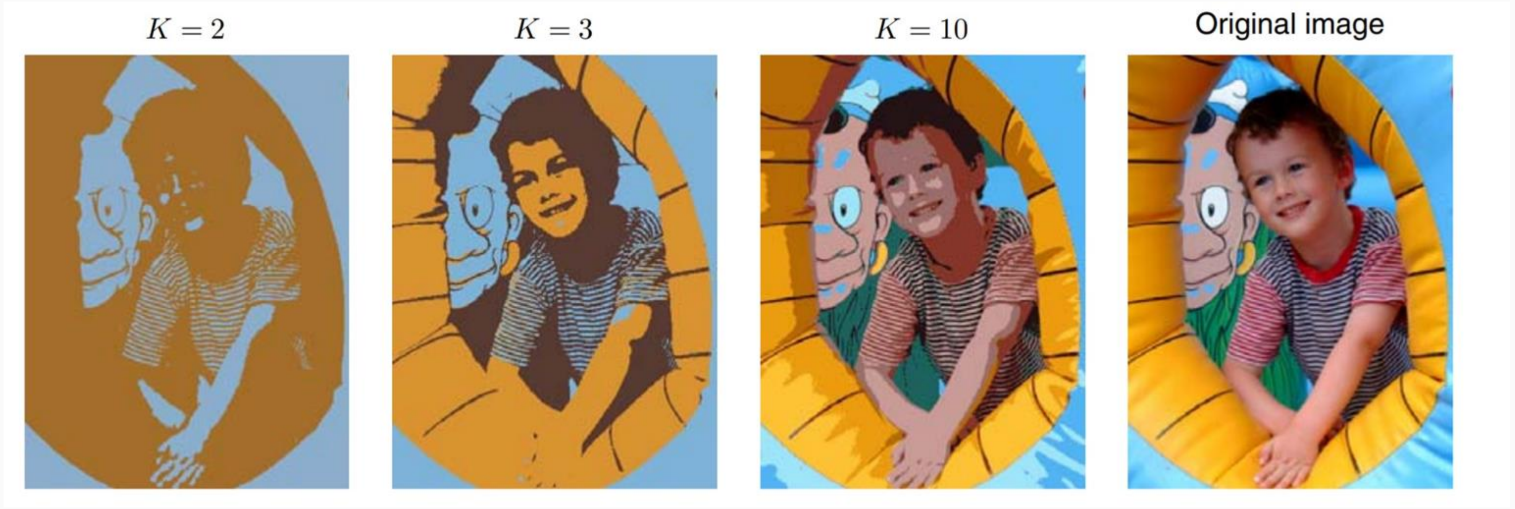
Manhattan Distance

$$d(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{D}(x_i - y_i)^{1/q}\right)^q$$

Minkowski Distance

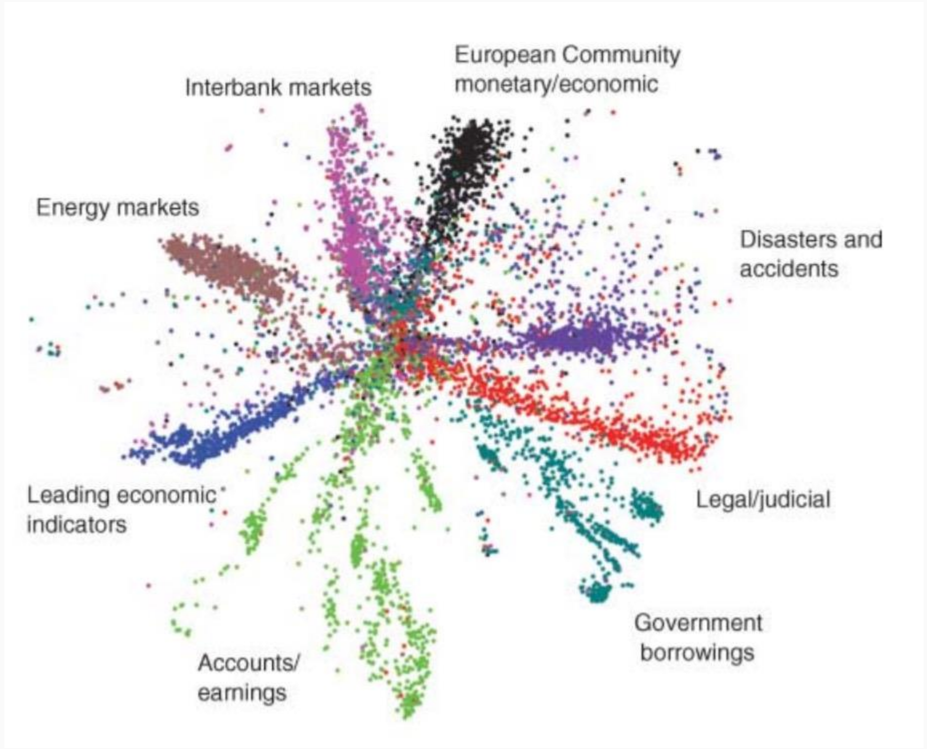
Clustering

Application example: **Quantization and compression**



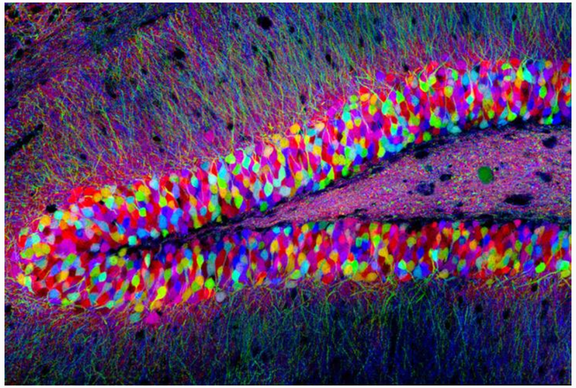
Clustering

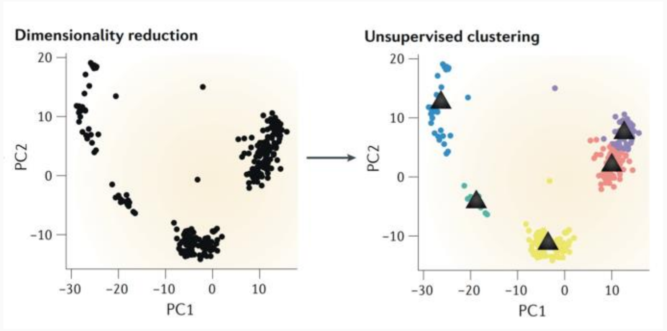
Application example: **Grouping documents or images**.



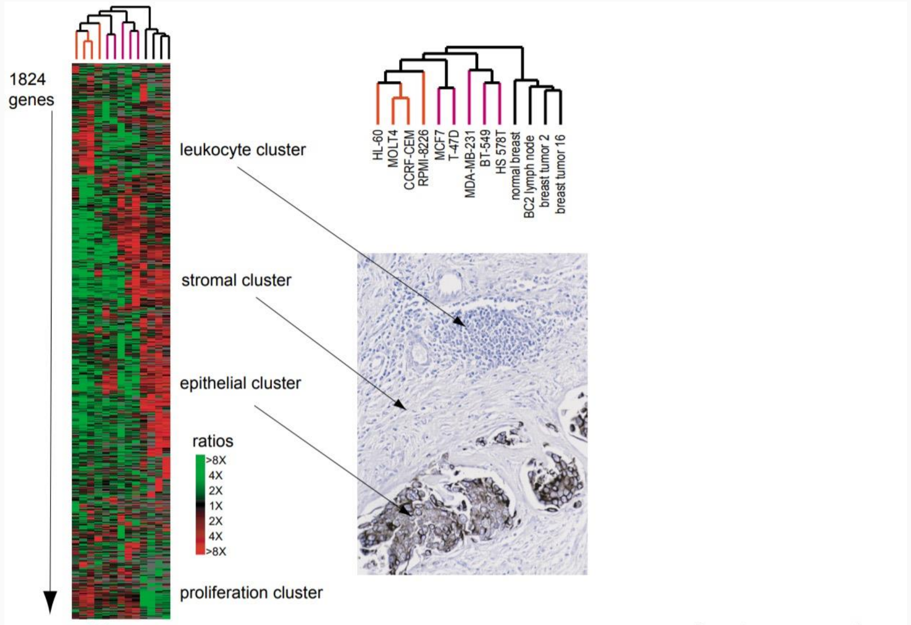
Clustering

Grouping Cells by Gene Activity





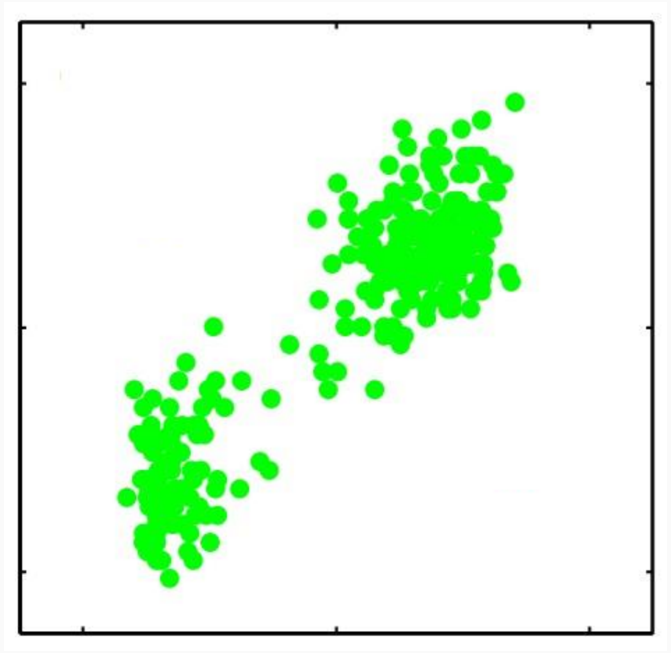
Cluster Cancer Types



Breast cancer specimen stained with anti-keratin antibodies, showing the complex mix of cell types characteristically found in breast tumours. The arrows highlight the different cellular components of this tissue specimen that were distinguished by the gene expression cluster analysis

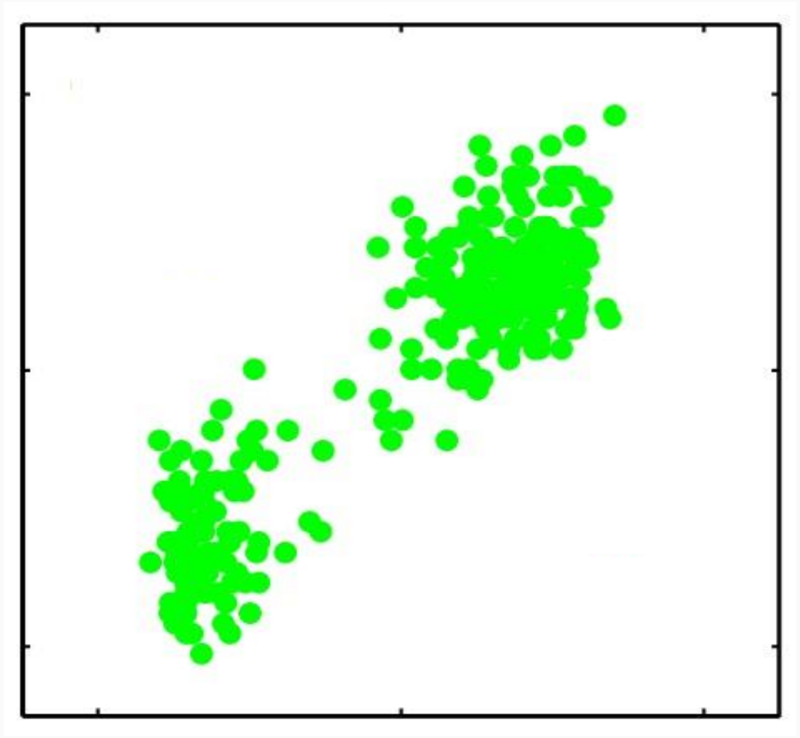
K-means clustering

- The goal of k-means is to progressively partition N observations into K clusters.
- Let’s see how this is done with an example:



- We want to cluster this dataset.
- First of all, we have to select the number of clusters K (which is a hyperparameter of the model).
- In this case, we select K=2.

K-means clustering



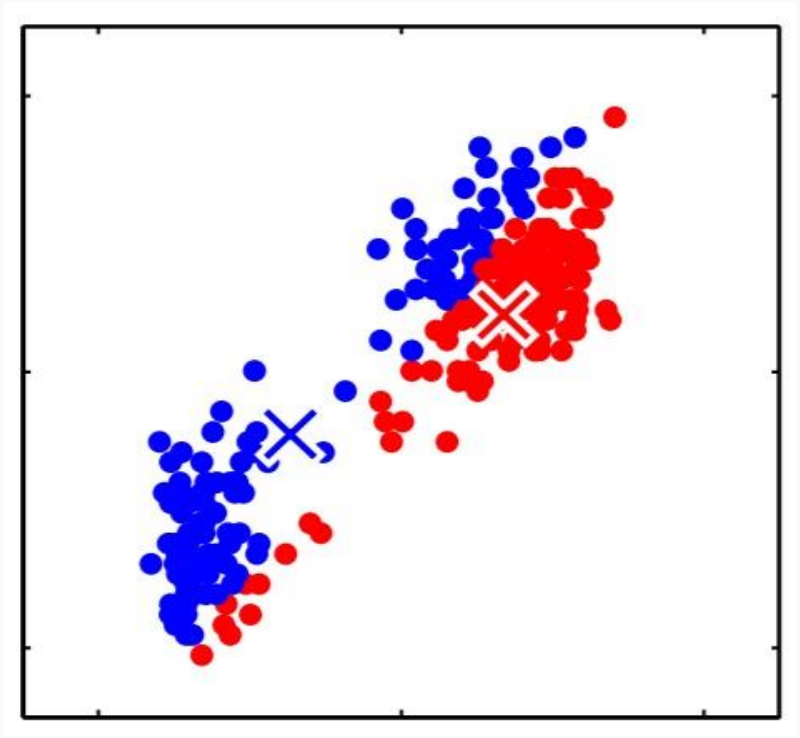
1. We sample K=2 random points in the input space.

   These points are called **centroids**.

   Each centroid represents a **different cluster**:

Centroid 1	➡	Class 1(Blue)
Centroid 2	➡	Class 1(Red)

K-means clustering



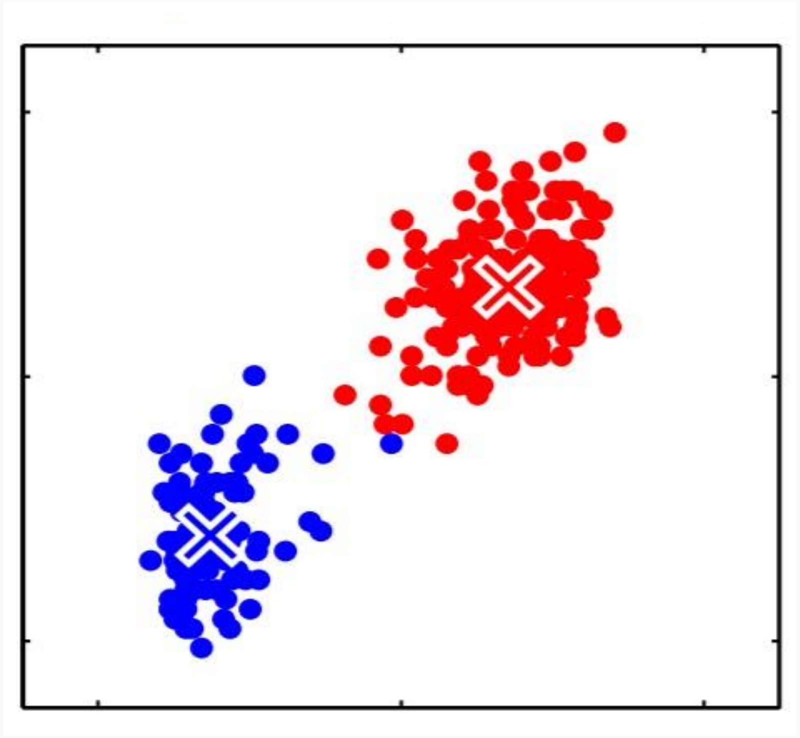
2. We assign each data point to the cluster of the closest centroid.

3. We **update** the centroids:

   We compute the **average** of all the points within each cluster.

   The new centroids are assigned to the average points.

K-means clustering



2. We assign each data point to the cluster of the closest centroid.

3. We **update** the centroids:

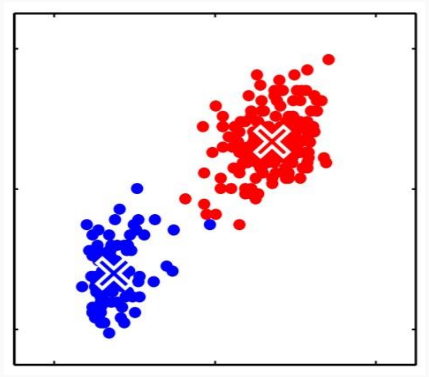
   We compute the **average** of all the points within each cluster.

   The new centroids are assigned to the average points.

4. Repeat 2 and 3 until centroids do not change anymore.

K-means clustering

- Mathematically, what we are trying to solve is the following **optimization problem**:



$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

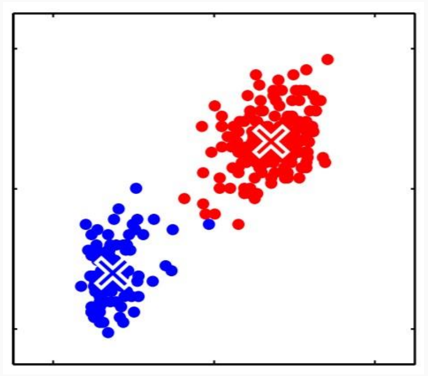
Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

$\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K]^T \longrightarrow$ Set of K centroids $\boldsymbol{\mu}_k \in \mathbb{R}^D \quad \boldsymbol{x}_i \in \mathbb{R}^D$

$\boldsymbol{r} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \dots, \boldsymbol{r}_N]^T \longrightarrow$ Cluster assignment variable $\boldsymbol{r}_i \in \mathbb{R}^K \longrightarrow$ This discrete variable r assigns each sample to a specific cluster.

K-means clustering

- Mathematically, what we are trying to solve is the following **optimization problem**:



$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- r_{ik} is a **discrete variable** that tells us which of the K clusters sample x_i is assigned to.

$$\begin{tikzcd} & r_{i1}=0 \\ \mathbf{x}_i \arrow[ur] \arrow[r] \arrow[dr] & r_{i2}=1 \\ & r_{i3}=0 \end{tikzcd}$$

The sample is assigned to class 2

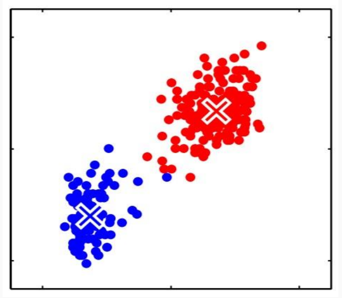
$$\begin{tikzcd} & r_{i1}=1 \\ \mathbf{x}_i \arrow[ur] \arrow[r] \arrow[dr] & r_{i2}=0 \\ & r_{i3}=0 \end{tikzcd}$$

The sample is assigned to class 1

r_{ik} is {0,1} and the sum over k must be one because, for each sample $\mathbf{x}_i$, we can select only one class.

K-means clustering

- Note that the term r_{ik} is needed to formulate the problem in a meaningful way.



$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r}, \boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- If we solve the problem **without the class assignment variable r**, we end up with a dummy solution!

$$\boldsymbol{\mu}^* = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2 \Rightarrow$$

Any solution is optimal if:

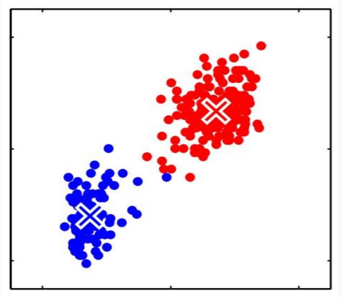
$$\underbrace{\frac{1}{K} \sum_{k=1}^{K} \boldsymbol{\mu}_k^*}_{\text{Average centroid}} = \underbrace{\frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i}_{\text{Average sample}} \longrightarrow$$

This is not meaningful

This is quadratic, we can compute a closed-form expression by setting the gradient to zero!

K-means clustering

- The distance term is needed as well!



$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- If we solve the problem without the distance term:

$$\mathbf{r}^* = \underset{\mathbf{r}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} \Rightarrow$$

Considering also the constraints, any random class assignment of the samples x_i is optimal

$\searrow$ This is not meaningful

K-means clustering

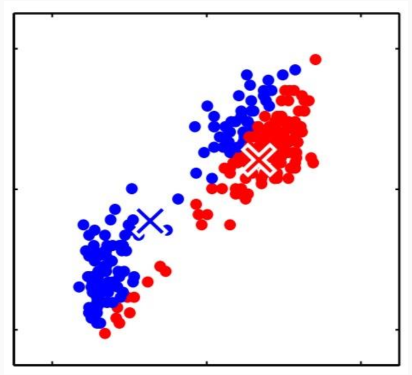
- To make our objective function meaningful, we need all terms!

$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

Compact clusters	$\Rightarrow$	Low objective
Spread clusters	$\Rightarrow$	High objective

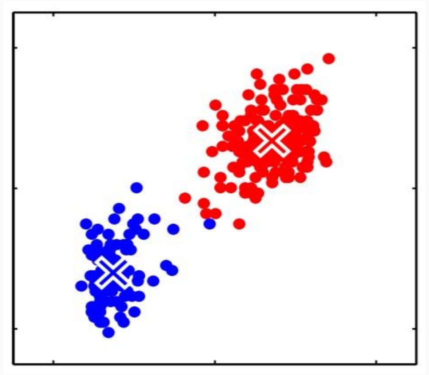
This is exactly what we want!



Bad solution

Why?

Because samples assigned to the same cluster are on average **quite distant**.



Good solution

Why?

Because samples assigned to the same cluster are on average **quite close**.

K-means clustering

How can we solve this problem?

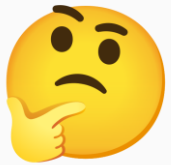
$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- One way would be to try all the possible combinations of **r** and **μ** and choose the one that minimizes our objective.

- *Is it feasible in this case?*

- There are too many combinations!



We need something clever. *Any idea?*

K-means clustering

- Let’s try to figure out the type of problems that we want to solve.

$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

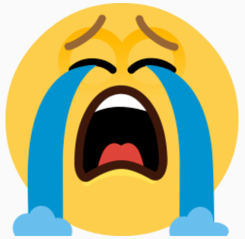
Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- *Is the objective function convex?*

The objective is not convex because of r_{ik}

It would have been convex wrt $\boldsymbol{\mu}_k$ without r_{ik}

- This means that there is *no direct (closed-form) solution* to this problem.
- Moreover, the objective might be **hard to optimize** (local minima, saddle points, etc)



K-means clustering

- Let’s try to figure out the type of problems that we want to solve.

$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

- *Is the objective at least differentiable?*

No, because r_{ik} is discrete!

- This means that we cannot use **gradient descent** as well!

- mmm...the problem is harder than we thought

- We have to accept the idea that we can only find an **approximate solution** to this problem (that might end up in a local minimum).

K-means clustering

$$\mathbf{r}^*, \boldsymbol{\mu}^* = \underset{\mathbf{r},\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

No fear! We can still derive an algorithm that gives us a reasonably good solution!

Any idea?

*We have two variables to optimize jointly (**r** and **μ**). What about if we optimize first over one variable and then over the other?*

This is called **Coordinate Descend.** Let’s assume we have two optimization variables a and b.

1. We **fix a** and we **optimize** the objective over **b** (at the beginning a is randomly initialized)
2. We fix **b** to the value found in 1 and we **optimize** the objective over **a**.
3. Repeat a and b until convergence.

K-means clustering

$$\mathbf{r}^*, \boldsymbol{\mu}^* = \operatorname*{argmin}_{\mathbf{r},\boldsymbol{\mu}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

No fear! We can still derive an algorithm that gives us a reasonably good solution!

Any idea?

For our problem, this means:

1. We fix the K centroids **μ** (at the beginning they are randomly initialized) and find the best value of **r**
2. We fix **r** to the value found in 1 ad we **optimize** the objective over **μ**.
3. Repeat 1 and 2 until convergence.

This algorithm does not converge in general to a global optimum. Note that the optimization variables are not independent!

But still…better than nothing….

K-means clustering

- With coordinate descent, we have to solve **two optimization problems** (one after the other) at each iteration.

- Let's take a look at the two **optimization problems.**

- **Problem 1**: We fix **μ** and we find the best **r**

$$\mathbf{r}^* = \underset{\mathbf{r}}{\text{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

$$r_{ik} = \begin{cases} 1 \text{ if } k = \underset{\mathbf{j}}{\text{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2 \\ 0 \text{ otherwise} \end{cases}$$

A direct solution exists if we fix **μ**

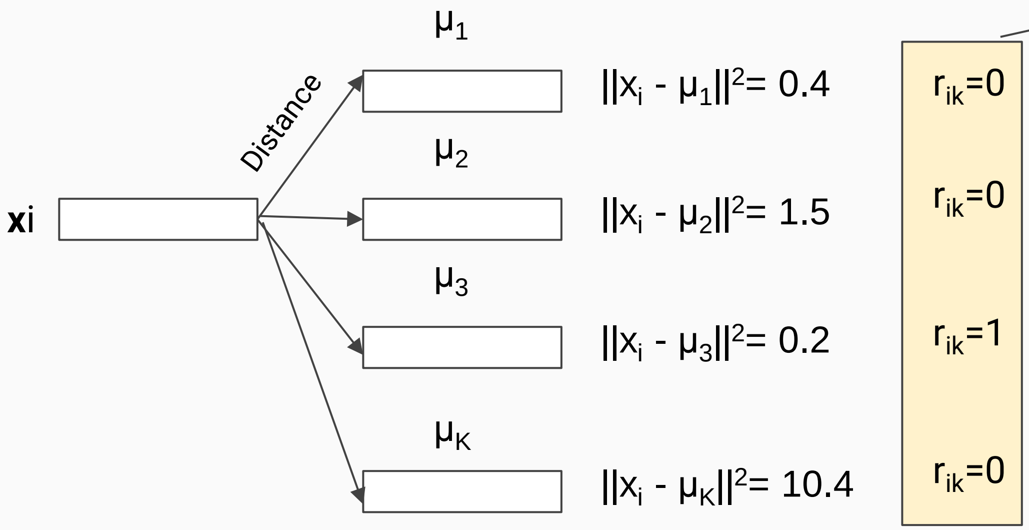
K-means clustering

Problem 1: We fix **μ** and we find the best **r**

$$\mathbf{r}^* = \underset{\mathbf{r}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

Subject to: $\sum_{k=1}^{K} r_{ik} = 1 \quad r_{ik} = \{0, 1\}$

$$r_{ik} = \begin{cases} 1 \text{ if } k = \underset{\mathbf{j}}{\operatorname{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2 \\ 0 \text{ otherwise} \end{cases}$$



$||x_i - \mu_1||^2 = 0.4$ — $r_{ik}=0$

$||x_i - \mu_2||^2 = 1.5$ — $r_{ik}=0$

$||x_i - \mu_3||^2 = 0.2$ — $r_{ik}=1$

$||x_i - \mu_K||^2 = 10.4$ — $r_{ik}=0$

This is the optimal setting for $\mathbf{r}_i$.

The objective function is 0.2 with the best setting. Any other choice of $\mathbf{r}_i$ that respects the constraints increases the objective.

$$k = 3 = \underset{\mathbf{j}}{\operatorname{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

K-means clustering

- Let’s now solve the other problem involved in the coordinate descent algorithm.

- **Problem 2**: We fix **r** and we find the best **μ**

$$\boxed{\boldsymbol{\mu}^* = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2}$$

$\longrightarrow$ If **r** is fixed, this function is **quadratic**!

We can compute a **closed-form solution** by setting the gradient over a generic $\boldsymbol{\mu}_k$ to zero

$$J = \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$$

$$= \sum_{i=1}^{N} r_{i1} ||\mathbf{x}_i - \boldsymbol{\mu}_1||^2 + \ldots + r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2 + \ldots + r_{iK} ||\mathbf{x}_i - \boldsymbol{\mu}_K||^2$$

(term expansion)

K-means clustering

$$J = \sum_{i=1}^{N} r_{i1}||\mathbf{x}_i - \boldsymbol{\mu}_1||^2 + \ldots + r_{ik}||\mathbf{x}_i - \boldsymbol{\mu}_k||^2 + \ldots + r_{iK}||\mathbf{x}_i - \boldsymbol{\mu}_K||^2$$

$$\nabla_{\boldsymbol{\mu}_k} J = \sum_{i=1}^{N} 2r_{ik}(\mathbf{x}_i - \boldsymbol{\mu}_k)(-1) = 0$$ (gradient computation)

$$\sum_{i=1}^{N} r_{ik}(\boldsymbol{\mu}_k - \mathbf{x}_i) = 0$$ (simplification)

$$\boxed{\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{N} r_{ik}}}$$

What is this?

This is the mean of the data points belonging to the same cluster!

K-means clustering

- We have derived the algorithm used at the beginning:

1. Hard assignment step ("**E step**"):

$$r_{ik} = \begin{cases} 1 \text{ if } k = \underset{\mathbf{j}}{\operatorname{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2 \\ 0 \text{ otherwise} \end{cases}$$

   for k={1,...,K}

2. Mean update step ("**M step**"):

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{N} r_{ik}} \quad \text{for k=\{1,...,K\}}$$

3. Stop when nothing changes.

$\rightarrow$ This type of coordinate descent is called the **Expectation-Maximization algorithm** (EM).

The EM algorithm is guaranteed to **improve the cost function** at each iteration.

When the cost function does not increase, the algorithm has reached a critical point (local/global minimum, saddle points, etc).

K-means clustering

- *How do we initialize the centroids?*
- The EM algorithm can get sucked in local minimum.
- The choice of initial centroids has a big impact on the chance of success (good clustering)

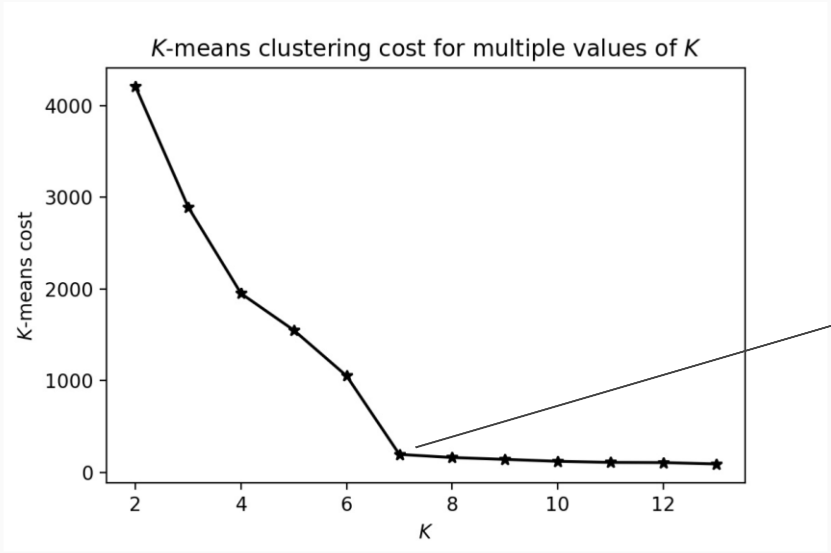
Random: set the initial centroids to be a random point or a random point selected from the dataset.

Any better idea?

K-means++: set initial centroids **μ** to be as far as possible with high probability (spread). The default for scikit-learn's K-Means class

K-means clustering

- K is the **number of clusters** to highlight.
- It is a **hyperparameter** of the model.
- *How can we find the best value for K?*



We can try multiple values of K, plot the objective (cost) of each clustering, and look for an **elbow inflection point**.

K=7 is probably a good number of clusters for this dataset.

K-means clustering

Advantages

- 🙂 Simple algorithm with easy Implementation.
- 🙂 Scalable to large datasets.

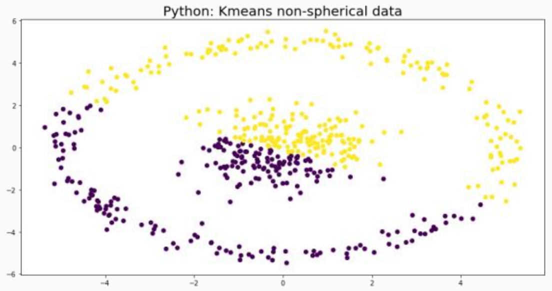
Disadvantages

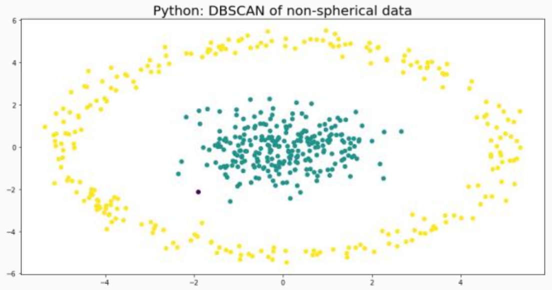
- ☹ The number of clusters K must be defined in advance.
- ☹ Sensitivity to **local minima**.
- ☹ Assumes **clusters** of **equal size**.
- ☹ Assumes **isotropic clusters** (similar variability in all directions).
- ☹ Assumes that the squared Euclidean distance is useful.

K-means clustering

- To improve the limitations of k-means, several alternatives have been proposed.
- One popular alternative is called **DBSCAN**.
- Based on **growing clusters** from the most **dense regions**.
- Handles arbitrary cluster topology, including **non-spherical clusters**.
- Can handle **outliers** better than K-means.

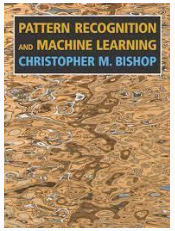
- Other alternatives are:
- **Hierarchical clustering** (where it outcome is a tree).
- **Fuzzy clustering** (where assign each sample to all the clusters according to a certain probability).





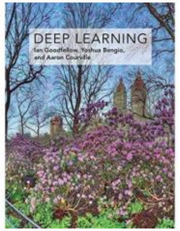
StatQuest

Additional Material



9.0.0 Mixture Models and EM

9.1.0 K-means Clustering
9.1.1 Image segmentation and compression

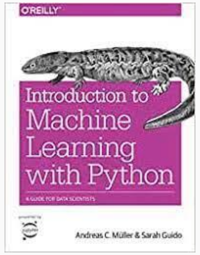


3.9: Common Probability Distributions

5.4: Estimators, Bias, and Variance

5.8.2: K-mean clustering

5.11.1: Curse of dimensionality


O'REILLY
Introduction to Machine Learning with Python
A GUIDE FOR DATA SCIENTISTS
Andreas C. Müller & Sarah Guido

3.5 Clustering