Unsupervised Learning

- Roughly speaking, it is about learning interesting structures in the data (unsupervisedly!)
- There is no supervision (no labels/responses), only inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$
- Some examples of unsupervised learning
  - Clustering: Grouping similar inputs together (and dissimilar ones far apart)
  - Dimensionality Reduction: Reducing the data dimensionality
  - Estimating the probability density of data (which distribution "generated" the data)
- Most unsupervised learning algos can also be seen as learning a new representation of data
  - Typically a compressed representation, e.g., clustering can be used to get a one-hot representation

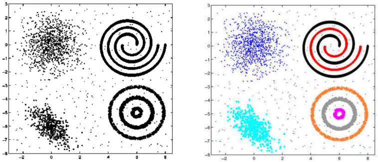
K = 6 clusters

$$z_n \quad \boxed{0}\boxed{0}\boxed{1}\boxed{0}\boxed{0}\boxed{0} \longleftarrow x_n \text{ belongs to cluster } 3$$

A one-hot (quantized) rep.

Clustering

- Given: N unlabeled examples $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$; no. of desired partitions K
- Goal: Group the examples into K "homogeneous" partitions



(a) Input data (b) Desired clustering

Picture courtesy: "Data Clustering: 50 Years Beyond K-Means", A.K. Jain (2008)

- Loosely speaking, it is classification without ground truth labels
- A good clustering is one that achieves:
  - High within-cluster similarity
  - Low inter-cluster similarity

Similarity can be Subjective

- Clustering only looks at similarities, no labels are given
- Without labels, similarity can be hard to define

- Thus using the right distance/similarity is very important in clustering
- Also important to define/ask: "Clustering based on what"?

Picture courtesy: `http://www.guy-sports.com/humor/videos/powerpoint_presentation_dogs.htm`

Clustering: Some Examples

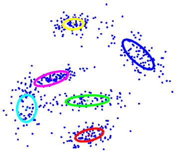
- Document/Image/Webpage Clustering
- Image Segmentation (clustering pixels)

- Clustering web-search results
- Clustering (people) nodes in (social) networks/graphs
- .. and many more..

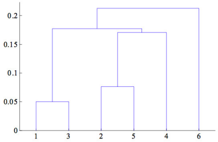
Picture courtesy: `http://people.cs.uchicago.edu/~pff/segment/`

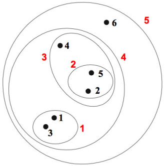
Types of Clustering

1. **Flat or Partitional clustering**
   - Partitions are independent of each other



2. **Hierarchical clustering**
   - Partitions can be visualized using a tree structure (a dendrogram)





   - Possible to view partitions at different levels of granularities by "cutting" the tree at some level

Flat Clustering: K-means algorithm (Lloyd, 1957)

- **Input:** N examples $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$; $\boldsymbol{x}_n \in \mathbb{R}^D$; the number of partitions K
- **Desired Output:** Cluster assignments of these N examples and K cluster means $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K$
- **Initialize:** K cluster means $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K$, each $\boldsymbol{\mu}_k \in \mathbb{R}^D$
  - Usually initialized randomly, but good initialization is crucial; many smarter initialization heuristics exist (e.g., K-means++, Arthur & Vassilvitskii, 2007)
- **Iterate:**
  - (Re)-Assign each example $\boldsymbol{x}_n$ to its closest cluster center (based on the smallest Euclidean distance)

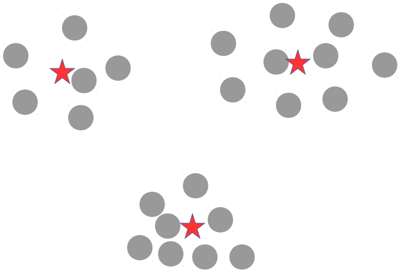
$$\boxed{\mathcal{C}_k = \{n : \quad k = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2\}}$$

    ($\mathcal{C}_k$ is the set of examples assigned to cluster k with center $\boldsymbol{\mu}_k$)
  - Update the cluster means

$$\boxed{\mu_k = \text{mean}(\mathcal{C}_k) = \frac{1}{|\mathcal{C}_k|} \sum_{n \in \mathcal{C}_k} \boldsymbol{x}_n}$$

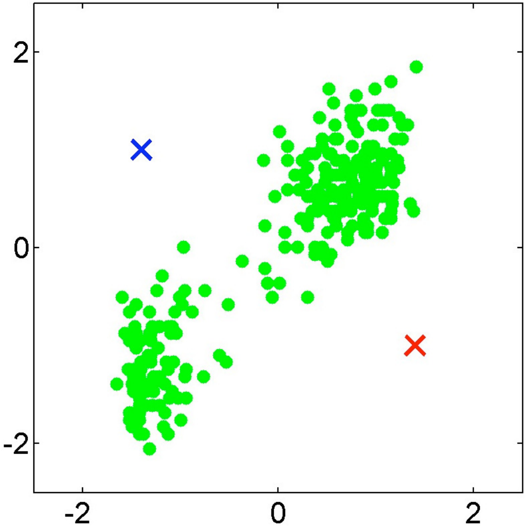
  - Repeat while not converged
- Stop when cluster means or the "loss" does not change by much

K-means = Prototype Classification (with unknown labels)

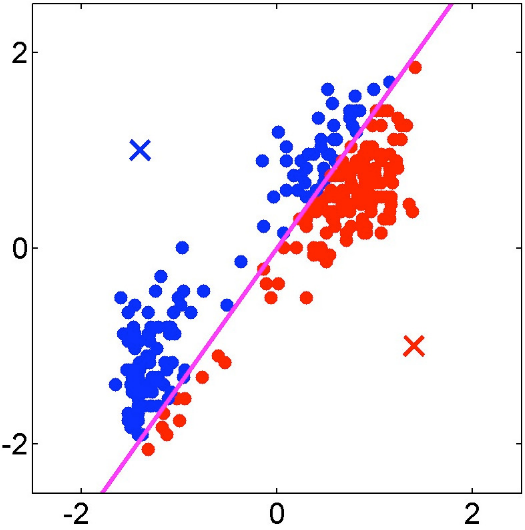


- **Guess the means**
- **Predict the labels**
- **Recompute the means**
- **Repeat**

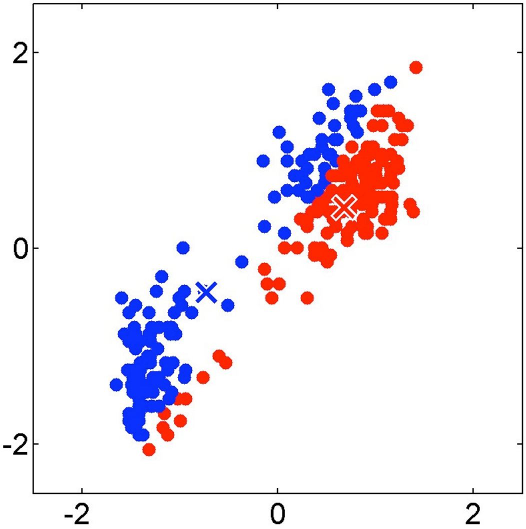
K-means: Initialization (assume $K = 2$)



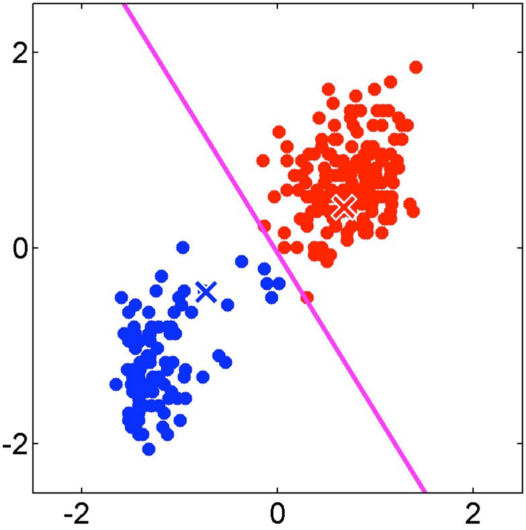
K-means iteration 1: Assigning points



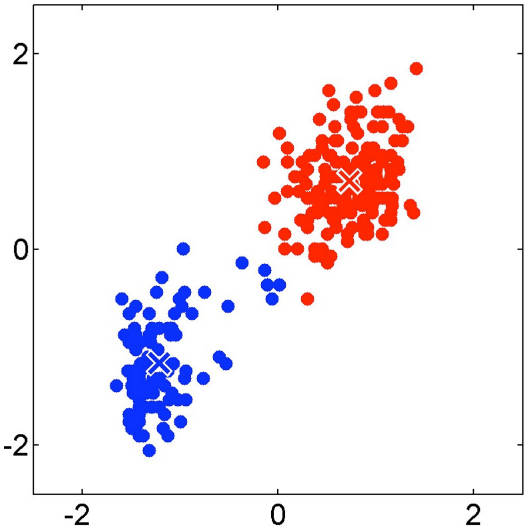
K-means iteration 1: Recomputing the centers



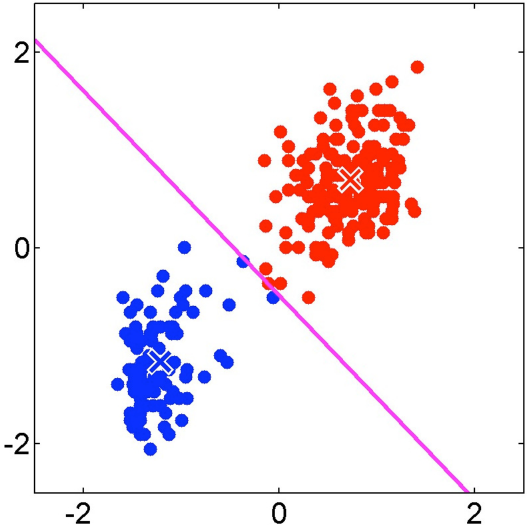
K-means iteration 2: Assigning points



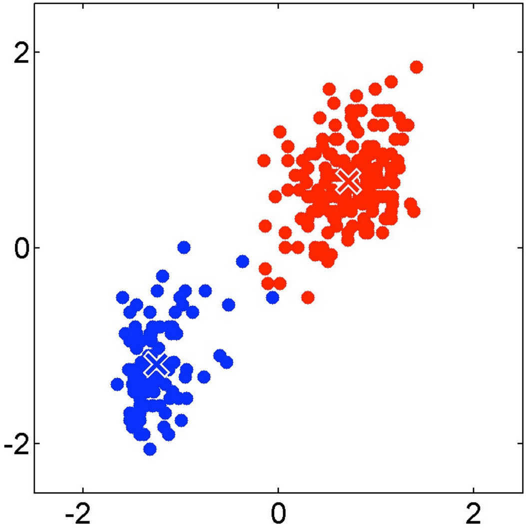
K-means iteration 2: Recomputing the centers



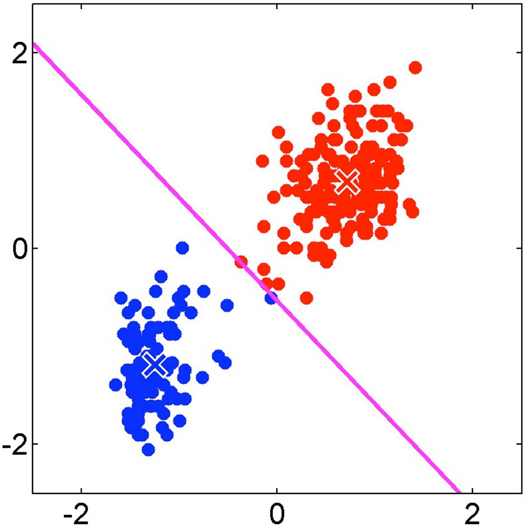
K-means iteration 3: Assigning points



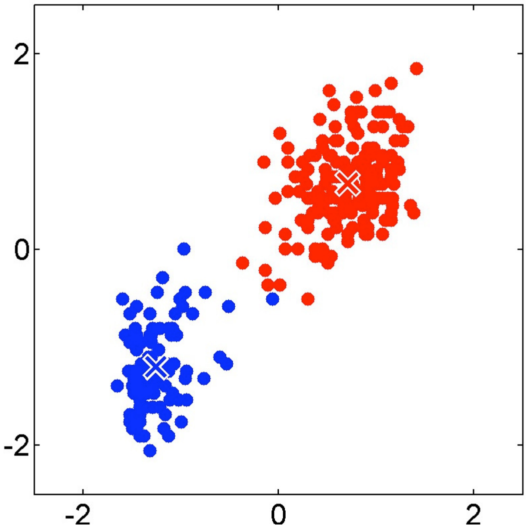
K-means iteration 3: Recomputing the centers



K-means iteration 4: Assigning points



K-means iteration 4: Recomputing the centers



Recap: K-means Algorithm

- Goal: Assign N inputs $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, with each $\boldsymbol{x}_n \in \mathbb{R}^D$, to K clusters (flat partitioning)
- Notation: $z_n \in \{1, \ldots, K\}$ or $\boldsymbol{z}_n$ is a K-dim one-hot vector($z_{nk} = 1$ and $z_n = k$ mean the same)

K-means Algorithm

1. Initialize K cluster means $\mu_1, \ldots, \mu_K$
2. For $n = 1, \ldots, N$, assign each point $\boldsymbol{x}_n$ to the closest cluster

$$z_n = \arg\min_{k \in \{1,\ldots,K\}} ||\boldsymbol{x}_n - \mu_k||^2$$

3. Suppose $\mathcal{C}_k = \{\boldsymbol{x}_n : z_n = k\}$. Re-compute the means

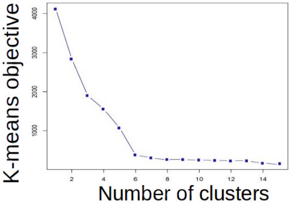
$$\mu_k = \text{mean}(\mathcal{C}_k), \quad k = 1, \ldots, K$$

4. Go to step 2 if not yet converged

- Note: The basic K-means models each cluster only by a mean μ_k. Ignores size/shape of clusters

K-means: Choosing K

- One way to select K for the K-means algorithm is to try different values of K, plot the K-means objective versus K, and look at the "elbow-point"



- For the above plot, $K = 6$ is the elbow point
- Can also information criterion such as AIC (Akaike Information Criterion)

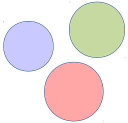
$$AIC = 2\mathcal{L}(\hat{\boldsymbol{\mu}}, \mathbf{X}, \hat{\mathbf{Z}}) + KD$$

.. and choose the K that has the smallest AIC (discourages large K)

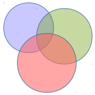
- Several other approaches when using probabilistic models for clustering, e.g., comparing marginal likelihood $p(\mathbf{X}|K)$, using nonparametric Bayesian models, etc.

K-means: Hard vs Soft Assignments

- Makes hard assignments of points to clusters
  - A point either completely belongs to a cluster or doesn't belong at all
  - No notion of a soft assignment (i.e., probability of being assigned to each cluster: say $K = 3$ and for some point $\boldsymbol{x}_n$, $p_1 = 0.7, p_2 = 0.2, p_3 = 0.1$)



Hard-assignment okay



Hard-assignment tricky

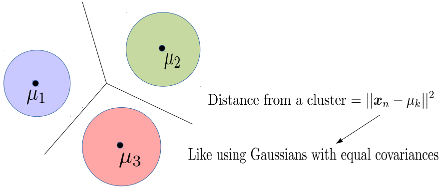
- A heuristic to get soft assignments: Transform distances from clusters into probabilities

$$\gamma_{nk} = \frac{\exp(-||\boldsymbol{x}_n - \mu_k||^2)}{\sum_{\ell=1}^{K} \exp(-||\boldsymbol{x}_n - \mu_\ell||^2)} \quad \text{(prob. that } \boldsymbol{x}_n \text{ belongs to cluster } k\text{)}$$

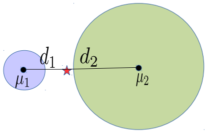
- These heuristics are used in "fuzzy" or "soft" K-means algorithms
- Soft K-means μ_k updates are slightly different: $\mu_k = \frac{\sum_{n=1}^{N} \gamma_{nk} \boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_{nk}}$ (all points used, but fractionally)

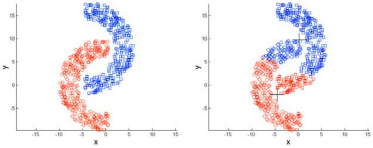
K-means: Decision Boundaries and Cluster Sizes/Shapes

- K-mean assumes that the decision boundary between any two clusters is linear
- Reason: The K-means loss function implies assumes equal-sized, spherical clusters



- Assumes clusters to be roughly equi-populated, and convex-shaped. Otherwise, may do badly





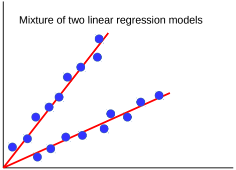
- Kernel K-means can help address some of these issues. Probabilistic models is another option

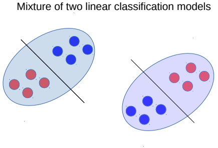
Clustering vs Classification

- Any clustering model typically learns two type of quantities
  - Parameters Θ of the clustering model (e.g., cluster means $\boldsymbol{\mu} = \{\mu_1, \ldots, \mu_K\}$ in K-means)
  - Cluster assignments $\mathbf{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$ for the points
- If the cluster assignments $\mathbf{Z}$ are known, learning the parameters Θ is just like learning the parameters of a classification model (typically generative classification) using labeled data
- Therefore it helps to think of clustering <u>as</u> (generative) classification with unknown labels
- This equivalence is very important and makes it possible to solve clustering problems
- Therefore many clustering problems are typically solved in the following fashion
  1. Initialize Θ somehow
  2. Predict $\mathbf{Z}$ given current estimate of Θ
  3. Use the predicted $\mathbf{Z}$ to improve the estimate of Θ (like learning a generative classification model)
  4. Go to step 2 if not converged yet

Clustering can help supervised learning, too

- Often "difficult" supervised learning problems can be seen as mixture of simpler models
- Example: Nonlinear regression or nonlinear classification as mixture of linear models





- An alternative to kernel methods and deep learning :-)
- Don't know which point belongs to which linear model $\Rightarrow$ Clustering problem
- Can therefore solve such problems as follows
  1. Initialize each linear model somehow (maybe randomly)
  2. Cluster the data by assigning each point to its "closest" linear model
  3. (Re-)Learn a linear model for each cluster's data. Go to step 2 if not converged.
- Often called Mixture of Experts models. Will look at these more formally after mid-sem