Unsupervised Learning: k-means

CS771: Introduction to Machine Learning

Nisheeth

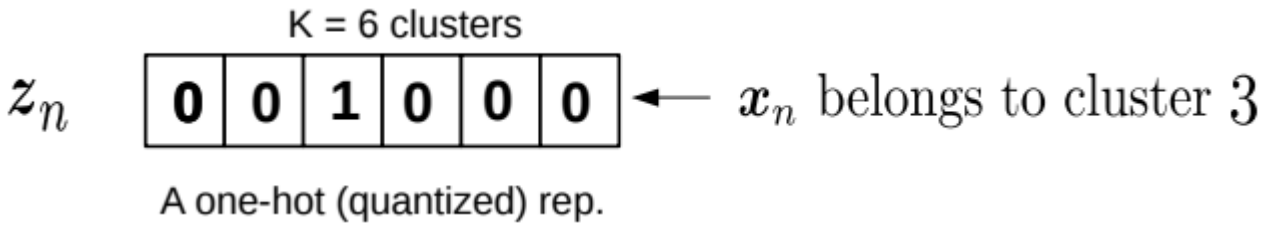
Unsupervised Learning

- It's about learning interesting/useful structures in the data (unsupervisedly!)
- There is no supervision (no labels/responses), only inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$
- 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 unsup. learning algos can also be seen as learning a new representation of data
  - For example, hard clustering can be used to get a one-hot representation

Already looked at some basic approaches(e.g., estimating a Gaussian given observed data)

Each point belongs deterministically to a single cluster

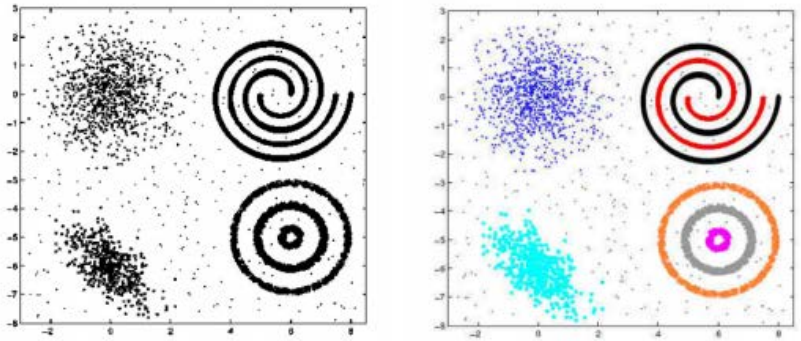
In contrast, there is also soft/probabilistic clustering in which $\boldsymbol{z}_n$ will be be probability vector that sums to 1 (will see later)



Clustering

In some cases, we may not know the right number of clusters in the data and may want to learn that (technique exists for doing this but beyond the scope)

- Given: N unlabeled examples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$; desired no. of 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 b/w inputs, since no labels are given
- Without labels, similarity can be hard to define

- Thus using the right distance/similarity is very important in clustering
- In some sense, related to asking: "Clustering based on what"?

Picture courtesy: http://www.guy-sports.com/humor/videos/powerpoint presentation dogs.htm

Clustering: Some Examples

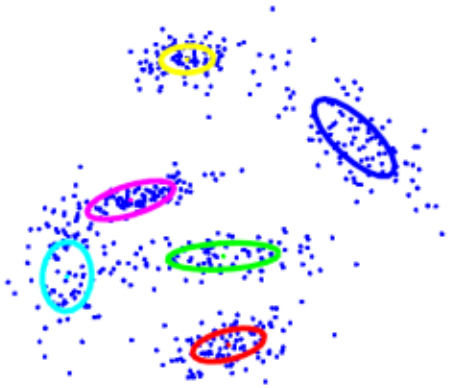
- Document/Image/Webpage Clustering
- Image Segmentation (clustering pixels)

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

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

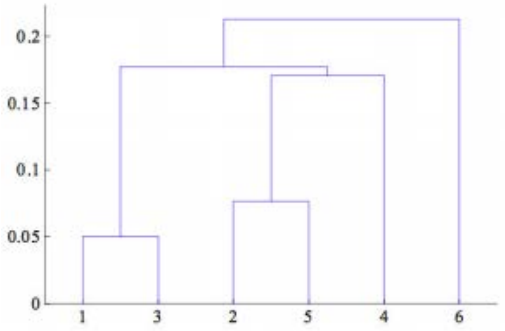
Types of Clustering

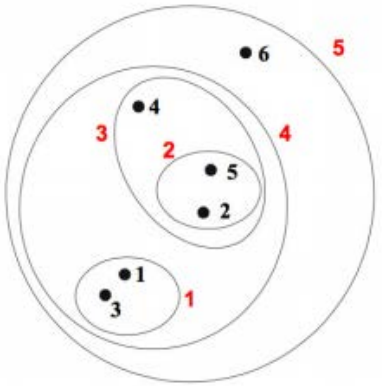
- Flat or Partitional clustering
  - Partitions are independent of each other



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

In hierarchical clustering, we can look at a clustering for any given number of cluster by "cutting the dendrogram at an appropriate level (so K does not have to be specified)





Hierarchical clustering gives us a clustering at multiple levels of granularity

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

- Input: N unlabeled examples $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$; $\boldsymbol{x}_n \in \mathbb{R}^D$; desired no. of partitions K
- Desired Output: Cluster assignments of these N examples and K cluster means
- Initialize: The K cluster means denoted by $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K$; with 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 input $\boldsymbol{x}_n$ to its closest cluster center (based on the smallest Eucl. distance)

$\mathcal{C}_k$: Set of examples assigned to cluster k with center $\boldsymbol{\mu}_k$

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

  - Update the cluster means

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

  - Repeat until not converged

Some ways to declare convergence if between two successive iterations:

- Cluster means don't change
- Cluster assignments don't change
- Clustering "loss" doesn't change by much

K-means algorithm: Summarized Another Way

- Notation: $z_n \in \{1,2,\ldots,K\}$ or $\boldsymbol{z}_n$ is a K-dim one-hot vector
  - ($z_{nk} = 1$ and $z_n = k$ mean the same)

K-means algo can also be seen as doing a compression by "quantization": Representing each of the N inputs by one of the $K < N$ means

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

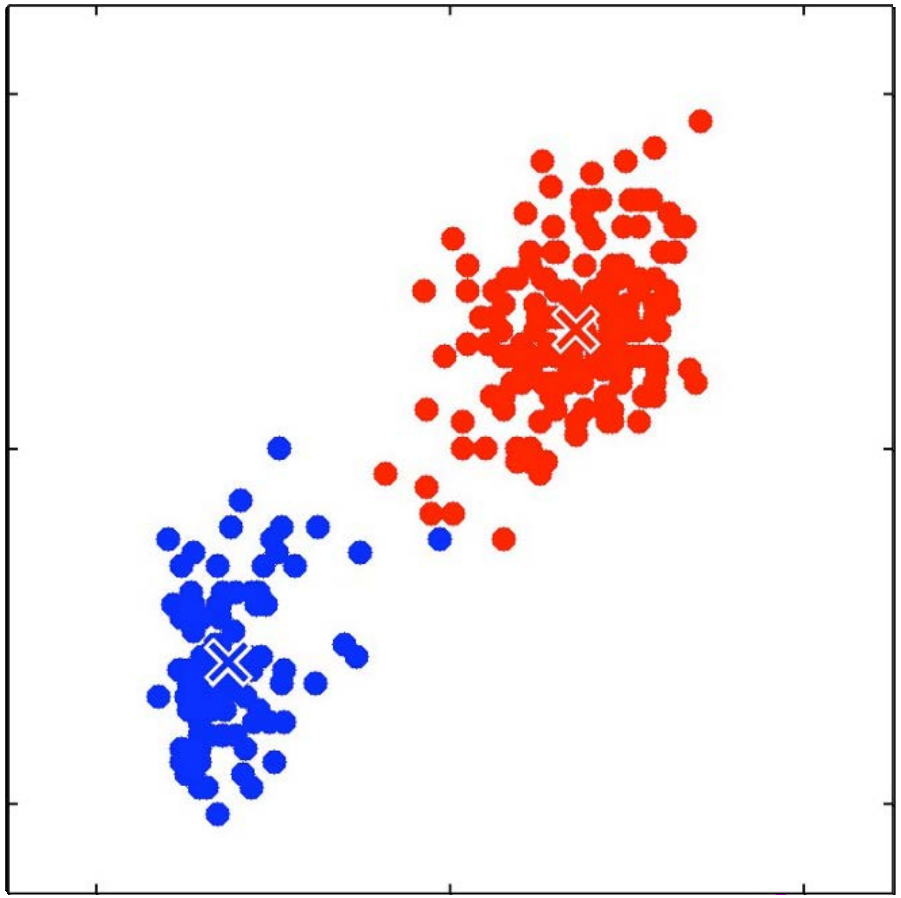
$$\mu_k = \text{mean}(\mathcal{C}_k), \quad k = 1, \ldots, K$$

4. Go to step 2 if not yet converged

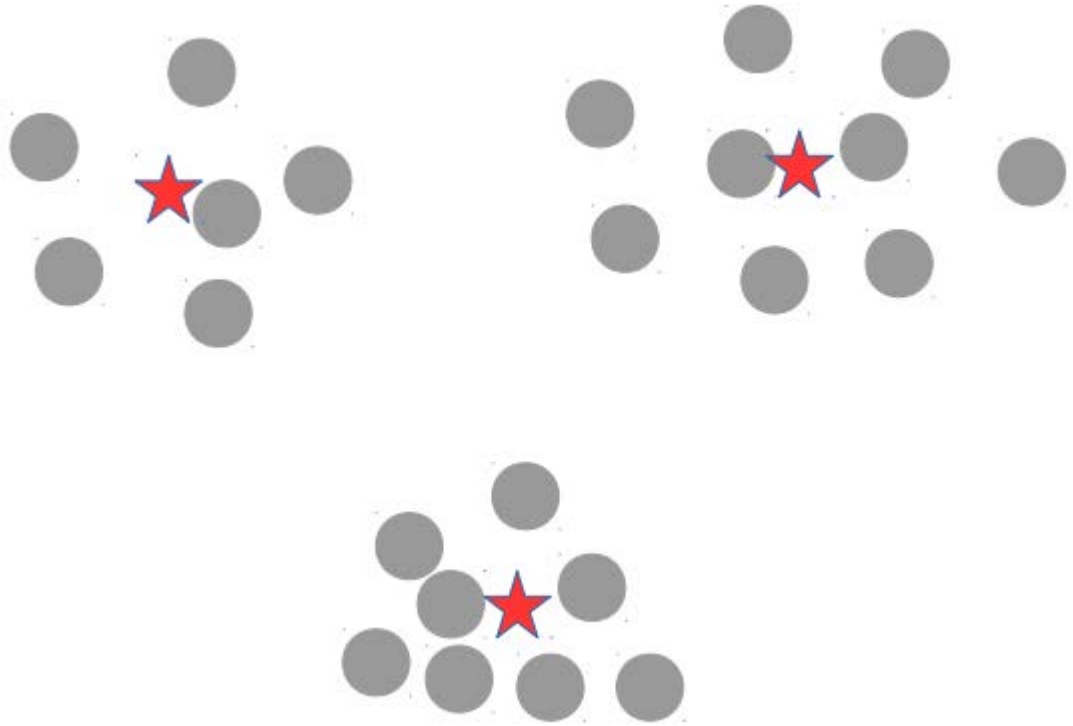
Can be fixed by modeling each cluster by a probability distribution, such as Gaussian (e.g., Gaussian Mixture Model; will see later)

This basic K-means models each cluster by a single mean μ_k. Ignores size/shape of clusters

K-means: An Illustration



K-means = LwP with Unlabeled Data



- Guess the means
- Predict the labels (cluster ids)
- Recompute the means using these predicted labels
- Repeat until not converged

The *K*-means Algorithm: Some Comments

- One of the most popular clustering algorithms
- Very widely used, guaranteed to converge (to a local minima; will see a proof)
- Can also be used as a sub-routine in graph clustering (in the Spectral Clustering algorithm): Inputs are given as an $N \times N$ adjacency matrix A ($A_{nm} = 0/1$)
  - Perform a spectral decomposition of the graph Laplacian of A to get F (an $N \times K$ matrix)
  - Run K-means using rows of the F matrix as the inputs
- Has some shortcomings but can be improved upon, e.g.,
  - Can be kernelized (using kernels or using kernel-based landmarks/random features)
  - More flexible cluster sizes/shapes via probabilistic models (e.g., every cluster is a Gaussian)
  - Soft-clustering (fractional/probabilistic memberships): z_n is not one-hot but a probability vector
  - Overlapping clustering – an input can belong to multiple clusters: z_n is a binary vector
  - .. even deep learning based K-means (use a deep network to extract features and run K-means on those)
- Let's look at K-means in some more detail now..

What Loss Function is *K*-means Optimizing?

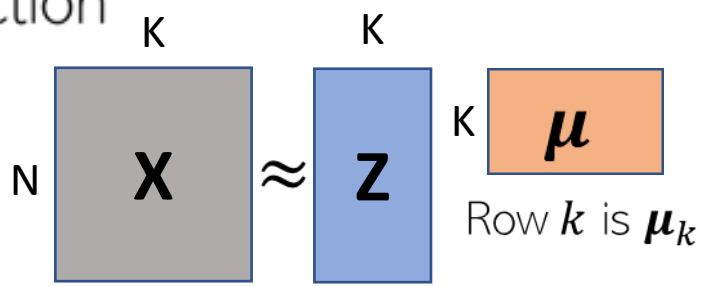
$\boldsymbol{z}_n = [z_{n1}, z_{n2}, \ldots, z_{nK}]$ denotes a length K one-hot encoding of $\boldsymbol{x}_n$

- Let $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K$ be the K cluster centroids/means
- Let $z_{nk} \in \{0, 1\}$ be s.t. $z_{nk} = 1$ if $\boldsymbol{x}_n$ belongs to cluster $\boldsymbol{k}$, and 0 otherwise
- Define the distortion or "loss" for the cluster assignment of $\boldsymbol{x}_n$

$$\ell(\boldsymbol{\mu}, \boldsymbol{x}_n, \boldsymbol{z}_n) = \sum_{k=1}^{K} z_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

- Total distortion over all points defines the K-means "loss function"

$$L(\boldsymbol{\mu}, \mathbf{X}, \mathbf{Z}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} ||\boldsymbol{x}_n - \mu_k||^2 = \underbrace{||\mathbf{X} - \mathbf{Z}\boldsymbol{\mu}||_F^2}_{\text{matrix factorization view}}$$



- The K-means problem is to minimize this objective w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{Z}$
  - Alternating optimization on this loss would give the K-means (Lloyd's) algorithm we saw earlier!

K-means Loss: Several Forms, Same Meaning!

- Notation: $\boldsymbol{X}$ is $N \times D$
- $\boldsymbol{Z}$ is $N \times K$ (each row is a one-hot z_n or equivalently $z_n \in \{1,2,\ldots,K\}$)
- $\boldsymbol{\mu}$ is K × D (each row is a μ_k)

Replacing the ℓ_2 (Euclidean) squared by ℓ_1 distances givesn the K-medoids algorithm (more robust to outliers)

Distortion on assigning $\boldsymbol{x}_n$ to cluster z_n

$$\mathcal{L}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu}) = \sum_{n=1}^{N} ||\boldsymbol{x}_n - \mu_{z_n}||^2$$

$$\mathcal{L}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu}) = \sum_{k=1}^{K} \underbrace{\sum_{n:z_n=k} ||\boldsymbol{x}_n - \mu_k||^2}_{\text{within cluster variance}}$$

$$\mathcal{L}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}||\boldsymbol{x}_n - \mu_k||^2$$

$$\mathcal{L}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu}) = \underbrace{||\mathbf{X} - \mathbf{Z}\boldsymbol{\mu}||_F^2}_{\text{as matrix factorization}}$$

$$\{\hat{\mathbf{Z}},\hat{\boldsymbol{\mu}}\} = \arg\min_{\mathbf{Z},\boldsymbol{\mu}} \mathcal{L}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu})$$

Total "distortion" or reconstruction error

- Note: Most unsup. learning algos try to minimize a distortion or recon error

Optimizing the K-means Loss Function

- So the K-means problem is

$$\{\hat{\mathbf{Z}}, \hat{\boldsymbol{\mu}}\} = \arg\min_{\mathbf{Z},\boldsymbol{\mu}} \mathcal{L}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu}) = \arg\min_{\mathbf{Z},\boldsymbol{\mu}} \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}||\boldsymbol{x}_n - \mu_k||^2$$

- Can't optimize it jointly for $\boldsymbol{Z}$ and $\boldsymbol{\mu}$. Let's try alternating optimization for $\boldsymbol{Z}$ and $\boldsymbol{\mu}$

Alternating Optimization for K-means Problem

1. Fix $\boldsymbol{\mu}$ as $\hat{\boldsymbol{\mu}}$ and find the optimal $\mathbf{Z}$ as

$$\hat{\mathbf{Z}} = \arg\min_{\mathbf{Z}} \mathcal{L}(\mathbf{X}, \mathbf{Z}, \hat{\boldsymbol{\mu}}) \quad \text{(still not easy - next slide)}$$

2. Fix $\mathbf{Z}$ as $\hat{\mathbf{Z}}$ and find the optimal $\boldsymbol{\mu}$ as

$$\hat{\boldsymbol{\mu}} = \arg\min_{\boldsymbol{\mu}} \mathcal{L}(\mathbf{X}, \hat{\mathbf{Z}}, \boldsymbol{\mu})$$

3. Go to step 1 if not yet converged

Solving for Z

- Solving for $\boldsymbol{Z}$ with $\boldsymbol{\mu}$ fixed at $\hat{\boldsymbol{\mu}}$

$$\hat{\mathbf{Z}} = \arg\min_{\mathbf{Z}} \mathcal{L}(\mathbf{X}, \mathbf{Z}, \hat{\mu}) = \arg\min_{\mathbf{Z}} \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} ||\boldsymbol{x}_n - \hat{\mu}_k||^2$$

- Still not easy. $\boldsymbol{Z}$ is discrete and above is an NP-hard problem
  - Combinatorial optimization: K^N possibilities for $\boldsymbol{Z}$ ($N \times K$ matrix with one-hot rows)

- Greedy approach: Optimize $\boldsymbol{Z}$ one row ($\boldsymbol{z_n}$) at a time keeping all others $\boldsymbol{z_n}$'s (and the cluster means $\mu_1, \mu_2, \ldots, \mu_K$) fixed

$$\hat{\boldsymbol{z}}_n = \arg\min_{\boldsymbol{z}_n} \sum_{k=1}^{K} z_{nk} ||\boldsymbol{x}_n - \hat{\mu}_k||^2 = \arg\min_{\boldsymbol{z}_n} ||\boldsymbol{x}_n - \hat{\mu}_{z_n}||^2$$

- Easy to see that this is minimized by assigning $\boldsymbol{x_n}$ to the closest mean
  - This is exactly what the K-means (Lloyd's) algo does!

Solving for $\boldsymbol{\mu}$

- Solving for $\boldsymbol{\mu}$ with $\boldsymbol{Z}$ fixed at $\widehat{\boldsymbol{Z}}$

$$\hat{\boldsymbol{\mu}} = \arg\min_{\boldsymbol{\mu}} \mathcal{L}(\mathbf{X}, \hat{\mathbf{Z}}, \boldsymbol{\mu}) = \arg\min_{\boldsymbol{\mu}} \sum_{k=1}^{K} \sum_{n:\hat{z}_n=k} ||\boldsymbol{x}_n - \mu_k||^2$$

- Not difficult to solve (each $\boldsymbol{\mu_k}$ is a real-valued vector, can optimize easily)

$$\hat{\mu}_k = \arg\min_{\mu_k} \sum_{n:\hat{z}_n=k} ||\boldsymbol{x}_n - \mu_k||^2$$

- Note that each $\boldsymbol{\mu_k}$ can be optimized for independently

- (Verify) This is minimized by setting $\widehat{\mu_k}$ to be mean of points currently in cluster $\boldsymbol{k}$
  - This is exactly what the K-means (Lloyd's) algo does!

Convergence of *K*-means algorithm

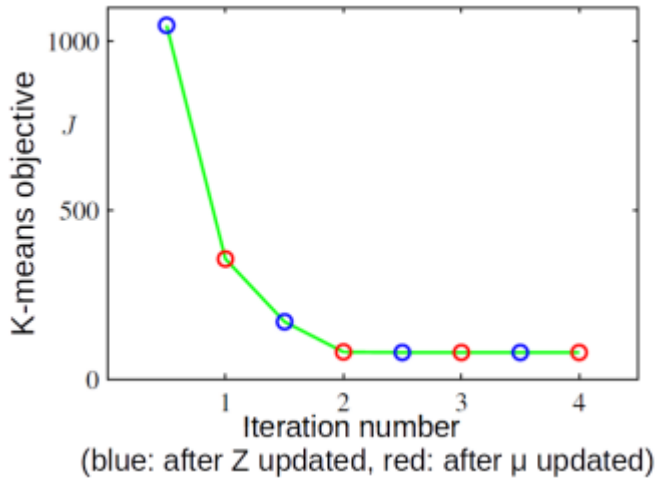
- Each step (updating $\boldsymbol{Z}$ or $\boldsymbol{\mu}$) can never increase the K-means loss
- When we update $\boldsymbol{Z}$ from $\boldsymbol{Z}^{(t-1)}$ to $\boldsymbol{Z}^{(t)}$

$$\mathcal{L}(\mathbf{X}, \mathbf{Z}^{(t)}, \boldsymbol{\mu}^{(t-1)}) \leq \mathcal{L}(\mathbf{X}, \mathbf{Z}^{(t-1)}, \boldsymbol{\mu}^{(t-1)}) \quad \text{because} \quad \mathbf{Z}^{(t)} = \arg\min_{\mathbf{Z}} \mathcal{L}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu}^{(t-1)})$$

- When we update $\boldsymbol{\mu}$ from $\boldsymbol{\mu}^{(t-1)}$ to $\boldsymbol{\mu}^{(t)}$

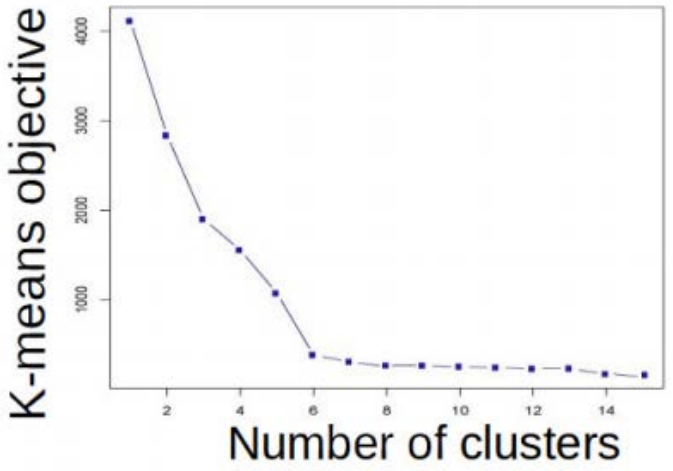
$$\mathcal{L}(\mathbf{X}, \mathbf{Z}^{(t)}, \boldsymbol{\mu}^{(t)}) \leq \mathcal{L}(\mathbf{X}, \mathbf{Z}^{(t)}, \boldsymbol{\mu}^{(t-1)}) \quad \text{because} \quad \boldsymbol{\mu}^{(t)} = \arg\min_{\boldsymbol{\mu}} \mathcal{L}(\mathbf{X}, \mathbf{Z}^{(t)}, \boldsymbol{\mu})$$

- Thus the K-means algorithm monotonically decreases the objective



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"



K=6 is the elbow point

- Can also information criterion such as AIC (Akaike Information Criterion)

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

and choose K which gives the smallest AIC (small loss + large K values penalized)

- More advanced approaches, such as nonparametric Bayesian methods (Dirichlet Process mixture models also used, not within K-means but with other clustering algos)

Coming up next

- Improvements to K-means
  - Soft-assignments
  - Handling complex cluster shapes (basic K-means assumes spherical clusters)
- Evaluating clustering algorithms (how to evaluate without true labels)
- Probabilistic approaches to clustering