3/26/2024: K-Means Clustering

Machine Learning: Algorithms that learn from data

Supervised Learning	Unsupervised Learning

Supervised Learning

Training dataset:

$D = \{(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})\}$

$x^{(i)}$: input to model

$y^{(i)}$: output (model predicts this)

Goal: Learn mapping from x to y

Unsupervised Learning

Training dataset only contain x's

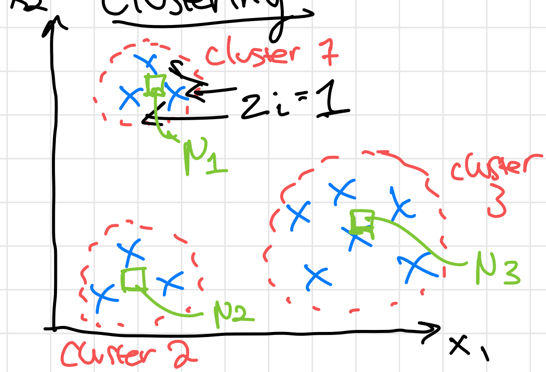
$D = \{x^{(1)}, \dots, x^{(n)}\}$

No "correct output"

Goal: Learn what structure is present in data

1. Group/subpopulation/cluster structure
2. Low-dimensional structure
3. Similarity/Relationships between words (word2vec)

Clustering



Input: Dataset $= \{x^{(1)}, \dots, x^{(n)}\}$ and # of clusters K

Output: An assignment $z_1, \dots, z_n$ where each $z_i \in \{1, 2, \dots, K\}$ denotes the cluster assigned to example $x^{(i)}$

K-means Clustering Algorithm

Idea #1: Write down a loss function to define "badness" of assignment $z_1, \dots, z_n$

Idea #2: Add more parameters to help define loss func.
Learn the "centroid" of each cluster

$\mu_1, \ldots, \mu_K$ where

$\mu_j \in \mathbb{R}^d$ is "center of mass" of cluster j

Payoff: Loss of assignment & choice of centroids is how far each $x^{(i)}$ is from its assigned centroid

Loss:

$$L(z_{1:n}, \mu_{1:K}) = \sum_{i=1}^{n} \|x^{(i)} - \underbrace{\mu_{z_i}}_{\text{Centroid of Cluster assigned to } x^{(i)}}\|^2$$

$= [z_1, \ldots, z_n]$

Cluster ID assigned to $x^{(i)}$

"Reconstruction Error":
If we only knew assignments $z_{1:n}$ & means $\mu_{1:K}$,

how far are we from reconstructing $\{x^{(1)}, \ldots, x^{(n)}\}$?

Goal: Minimize $L(z_{1:n}, \mu_{1:K})$ with respect to $z_{1:n}, \mu_{1:K}$

Can't do gradient descent because z_i's are discrete ie only be one cluster or another, no "in-between"

Solution: Alternating Minimization

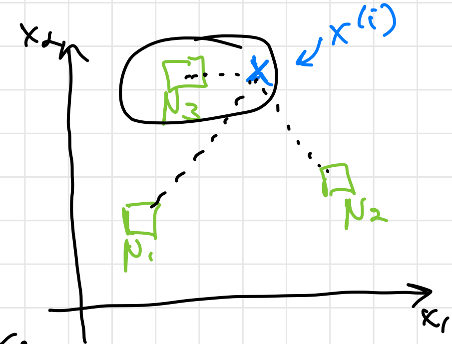
① Start with random choice of $\mu_1, \ldots, \mu_K$

Alternate until $\boxed{\text{Convergence}}$ ← when $z_{1:n}$'s & $\mu_{1:k}$'s stop changing.

(2) Choose $z_{1:n}$ to minimize L given current $\mu_{1:k}$

(3) Choose $\mu_{1:k}$ to minimize L given current $z_{1:k}$

$\boxed{\text{Step 1}}$ Choose each μ_j to be random distinct $x^{(i)}$



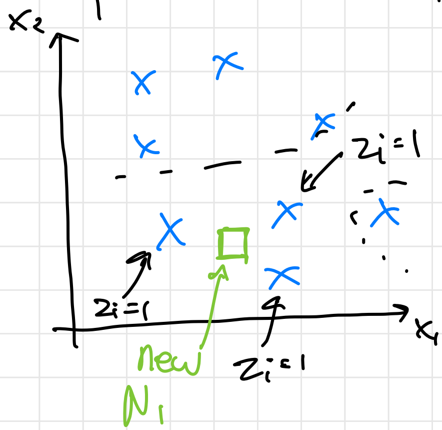
$\boxed{\text{Step 2}}$ Minimizing L w.r.t. $z_{1:n}$

For each $i = 1, \ldots, n$

set $z_i = \underset{j=1,\ldots,k}{\operatorname{argmin}} \|x^{(i)} - \mu_j\|^2$

$\boxed{\text{Step 3}}$ Minimizing L w.r.t. $\mu_{1:k}$

Intuitively: μ_j should be mean of all points where $z_i = j$



$$\text{minimize} \sum_{i=1}^{n} \|x^{(i)} - \mu_{z_i}\|^2$$

$$= \sum_{j=1}^{k} \sum_{i: z_i = j} \|x^{(i)} - \mu_j\|^2$$

For particular index j, minimize w.r.t. μ_j

$$\Leftrightarrow \text{minimize} \sum_{i: z_i = j} \|x^{(i)} - \mu_j\|^2$$

Take gradient & set to 0

$$\nabla_{\mu_j} \sum_{i: z_i = j} \|x^{(i)} - \mu_j\|^2 = \sum_{i: z_i = j} \cancel{2} (x^{(i)} - \mu_j) \cdot \cancel{(-1)} = 0$$

$$\sum_{i : z_i = j} x^{(i)} = \underbrace{\left| \{ i : z_i = j \} \right|}_{\text{\# of points assigned to cluster } j} \cdot \mu_j$$

$$\mu_j = \frac{1}{\left| \{ i : z_i = j \} \right|} \cdot \sum_{i : z_i = j} x^{(i)} = \text{mean of all } x^{(i)} \text{ where } z_i = j$$

<u>No guarantee of finding optimal solution</u>

- Algorithm runs until it finds local optimum
- Random initialization affects final result

<u>How to choose K?</u> This is a hyperparameter

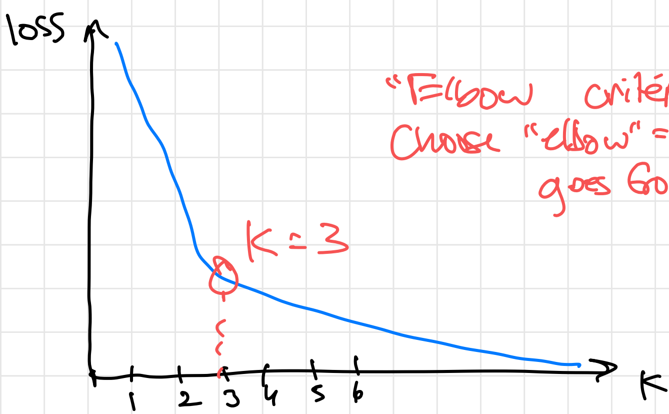
Wrong way: Choose K to minimize loss on dev set

Why not? Larger K always makes loss lower

K=1: large loss

K=2: lower loss

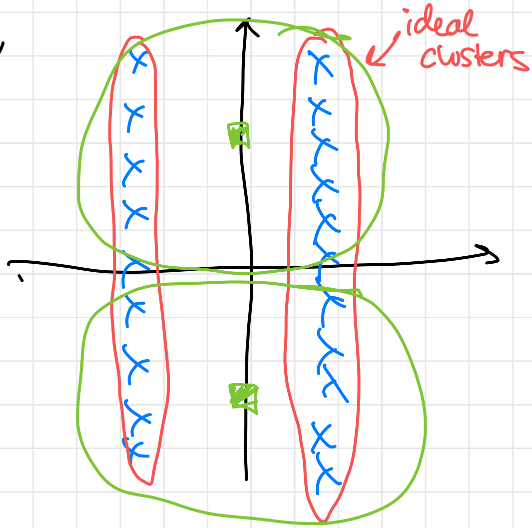
K=5: loss goes down



"Elbow criterion":

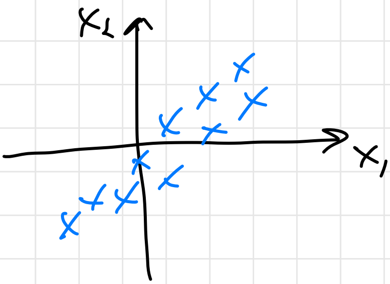
Choose "elbow" = point where curve goes from steep descent to shallow descent

K-means uses Euclidean distance, so it assumes that clusters are spherical

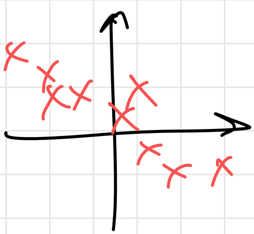


Goal: new algorithm that can learn <u>location</u> and <u>shape</u> of clusters

= covariance



x_1 & x_2 are positively correlated $\Leftrightarrow$ positive covariance



negative correlation $\Leftrightarrow$ negative covariance