$$X = \begin{bmatrix} x_{11} & \cdots & x_{1j} & \cdots & x_{1m} \\ \vdots & & \vdots & & \\ x_{n1} & \cdots & x_{nj} & \cdots & x_{nm} \end{bmatrix}$$

row = data point
col = attribute/feature

data set with n data points with m features each

feature space = R^n of X (# of ind cols)

<u>comparing data points</u>

Dissimilarity func - outputs: ↑ if dissimilar ↓ if similar

- distance function : all must always be true
$$\begin{cases} d(i,j) = 0 \text{ iff } i=j \\ d(i,j) = d(j,i) \\ d(i,j) \le d(i,k) + d(k,j) \end{cases}$$

- Minkowski Distance : $L_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{1/p}$ for d amount of x,y points
    - $p \ge 1$, but doesnt have to be a whole #
    - $p=1 \rightarrow$ Manhattan dist
    - $p=2 \rightarrow$ Euclidian dist
    - aka L_p Norm

- $d(A,B) = \|A-B\|$ (A, B: vecs), $d(0,X) = \|X\|$
- Norm = some reference to distance, not all dist funcs create a Norm

Similarity funcs : ↑ if similar, ↓ if dissimilar

- Jaccard Similarity : $JSim(x,y) = \frac{|x \cap y|}{|x \cup y|}$
    : $JDist(x,y) = 1 - JSim(x,y)$
    - *use if data points are mostly similar / small difference is significant

- Cosine Similarity : $s(x,y) = \cos(\theta)$ st θ = angle between x and y
    - $\cos = 1 \rightarrow$ proportional
    - $\cos = 0 \rightarrow$ orthogonal (perpindicular/right angle)
    - $\cos = -1 \rightarrow$ opposite
    - *use when direction is more important than magnitude

to get corresponding dissim func:
$d(x,y) = \frac{1}{s(x,y)}$ or $k - s(x,y)$ for some k

clustering can be ambiguous - no 1 correct answer (but can be wrong ones)

partional clustering - partition dataset into k partitions, such that
variance (cost function) is minimized

cost function - how "good" a clustering is: high = bad (expensive), low = good (cheap)

$$\sum_{i}^{k} \sum_{x \in C_i} d(x, \mu_i)^2$$

X = dataset = $\{x_1, \ldots, x_n\}$

k = centers = $\{\mu_1, \ldots, \mu_n\}$

d = distance

k = # clusters, n = # data points

needs to be a balance between cost and # clusters

- cant say lowest cost, bc cost = 0 when $k = n$
- $k = 1$, $k = n$ too easy
- if $X \in \mathbb{R}^n$ st $n > 2$, very difficult (cant visualize)

Lloyd's alg:

1. randomly pick k data points as centers
2. unassign all points
3. assign each point in dataset to closest center
4. compute new centers as means of each cluster
5. repeat 2-4 until done (centers don't change)