Overview

- Unsupervised Learning
- Clustering
  - K-means/mediods
  - Mixture of Gaussians/Expectation Maximization (EM)
  - Agglomerative Clustering

Supervised vs. Unsupervised Learning

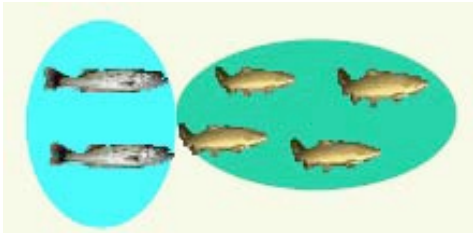
- In general our data comes in two flavors:
  1. Supervised
  2. Unsupervised
- With *supervised* data we have not only the observations, $\{X_i\}_{i=1}^N$ but also associated labels, $\{Y_i\}_{i=1}^N$.
  - Together these form our dataset: $\{X_i, Y_i\}_{i=1}^N$
  - We will later use this type of data to do
    - Regression
    - Classification
- With *unsupervised* data we only have the observations, $\{X_i\}_{i=1}^N$

Why Unsupervised Learning?

- It's harder ☹
  - How do we know if results are meaningful since there no answers (labels) to test against?
- We need it though
  - Labeling large datasets is very costly
  - May have no idea what/how many classes there are (data mining)
  - May want to use clustering to gain some insight into the structure of the data before designing a classifier

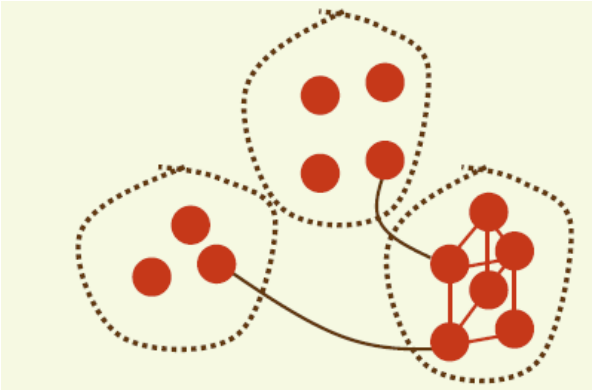
Unsupervised Learning

- The most common type of unsupervised learning is *clustering* where we attempt to learn underlying patters from data

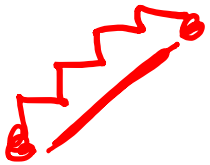


What is Clustering?

- Clustering: The process of grouping a set of objects into classes of similar objects
  - Items within cluster should be similar
  - Observations from different clusters should be dissimilar
- This is the most common form of *unsupervised learning*



Issues for Clustering



- Need a notion of similarity/distance
  - Similarity: Gaussian, Cosine
  - Distance: L2 (Euclidian), L1 (Manhattan)
  - See Blackboard for these
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid "trivial" clusters – too small or too large
    - Use some statistics?
- How to evaluate cluster quality?
  - It's unsupervised after all….

Hard vs Soft Clustering

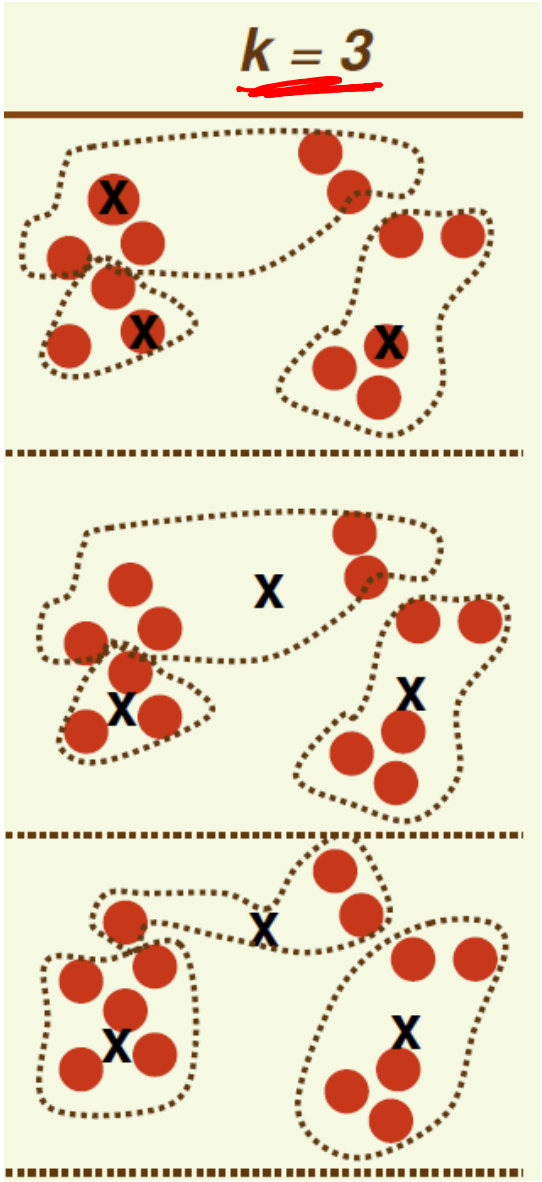
- Hard clustering: Each observation belongs to exactly one cluster
  - More common and easier to do
- Soft clustering: An observation can belong to more than one cluster
  - And/or can belong to clusters with some probability

k-Means/Mediods/Modes

- http://user.ceng.metu.edu.tr/~akifakkus/courses/ceng574/k-means/

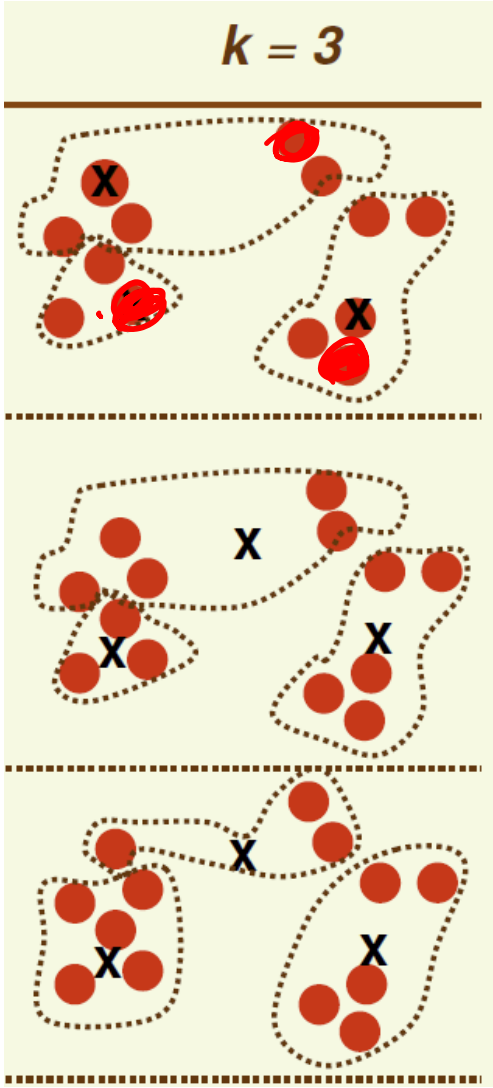
k-means/medoids

- Assumes observations are real-valued vectors
- Each observation is associated with a single *reference vector* (i.e cluster center).
  - This is typically the reference vector that the observation is closest to or most similar to.
  - For simplicity/intuition, we'll compare observations using Euclidean distance.
    - Therefore we assign an observation to the reference vector it is closest to.



k-means/mediods

- At each iteration
  1. Each observation is reassigned to the reference vector closest to it
  2. The reference vectors are updated based on the observations that are associated with it
- This continues until some termination criteria is met
- A few design considerations:
  - Where do the reference vectors start?
  - What are some termination criteria?



Initial Reference Vector Choice

- Results can vary based on initial reference vector choice.
- Some choices can result in poor convergence rate, or convergence to sub-optimal clustering
- Some ideas might be:
  - Select good initial reference vectors using a heuristic (e.g. instances least similar to any existing mean)
  - Try out multiple starting points
  - Initialize with the results of another method

Termination Conditions

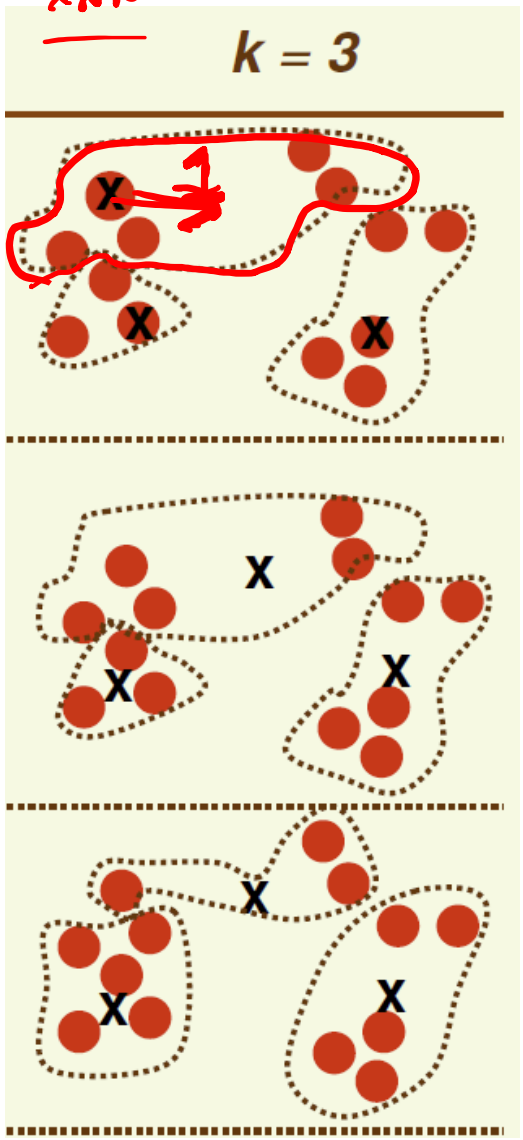
- Several possibilities
  1. A fixed number of iterations
  2. Between iterations
     1. Cluster assignments don't change
     2. Reference vectors don't change (by much)

k-means

- Ok, so how to we update the reference vectors?
- One idea is to set the reference vector to be the *mean* of the observations associated with it.
- Given cluster C_i (set of observations associated with reference vector i) of size $|C_i|$, we compute the center of gravity or mean of points in that cluster as:

$$a_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

- This approach is called *k-means*

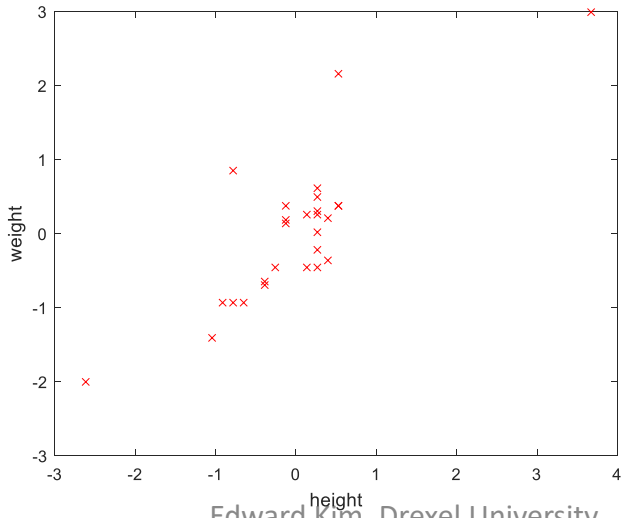


k-Means Pseudocode

- (Somehow) select k vectors $\{a_1, a_2, \ldots, a_k\}$ as the initial reference vectors (cluster centers).
- Until clustering converges or other stopping condition:
  - For each observation x
    - Assign x to the cluster C_i such that
      $$i = \underset{i}{\text{argmin}}\big(dist(x, a_i)\big)$$
  - For each cluster C_i compute the new reference vector as the mean of those associated with it:
    - Update $a_i = \frac{1}{|C_i|}\sum_{x \in C_i} x$

k-Means Example

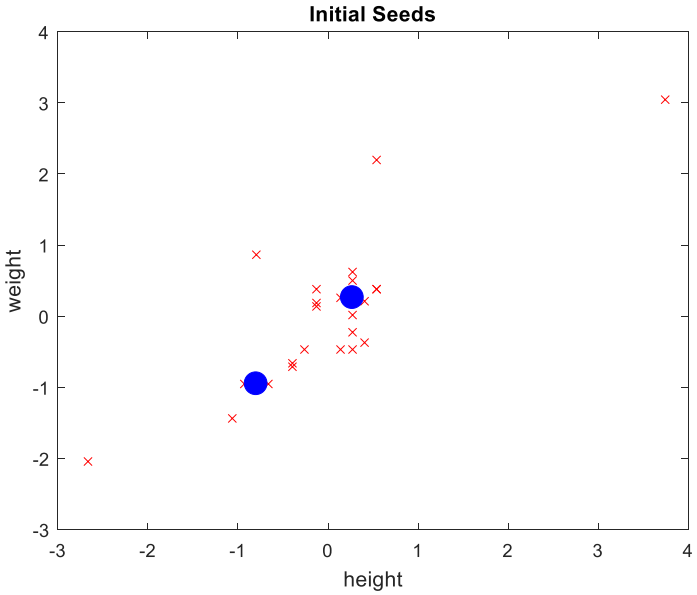
- We want to figure out t-shirt sizes based on people's height and weight
- Standardize features
  - $\mu_{height} = 67.963, \sigma_{height} = 7.633$
  - $\mu_{weight} = 159.3704, \sigma_{weight} = 42.0560$



height (inches)	weight (pounds)
61	120
65	130
72	250
69	120
62	195
62	120
60	100
70	140
70	160
65	132
48	75
72	175
67	167
69	140
96	285
70	172
70	185
71	168
70	180
69	170
70	150
70	170
71	144
66	140
67	175
67	165
72	175

k-Means Example

- Let's assume we want 2 shirt sizes
  - $k = 2$
- Initial reference vectors

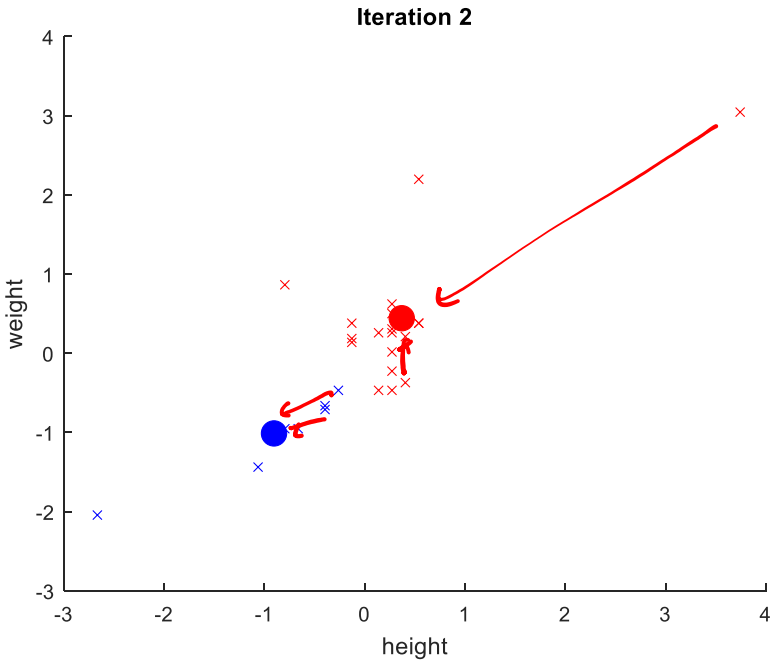


Example

- After 2 iterations, the vectors stop moving and we get the following assignments
- And the reference vectors are at:
  - a_1=[0.3703, 0.4229]
  - a_2=[-0.8794, -1.0045]
- So we consider our final model

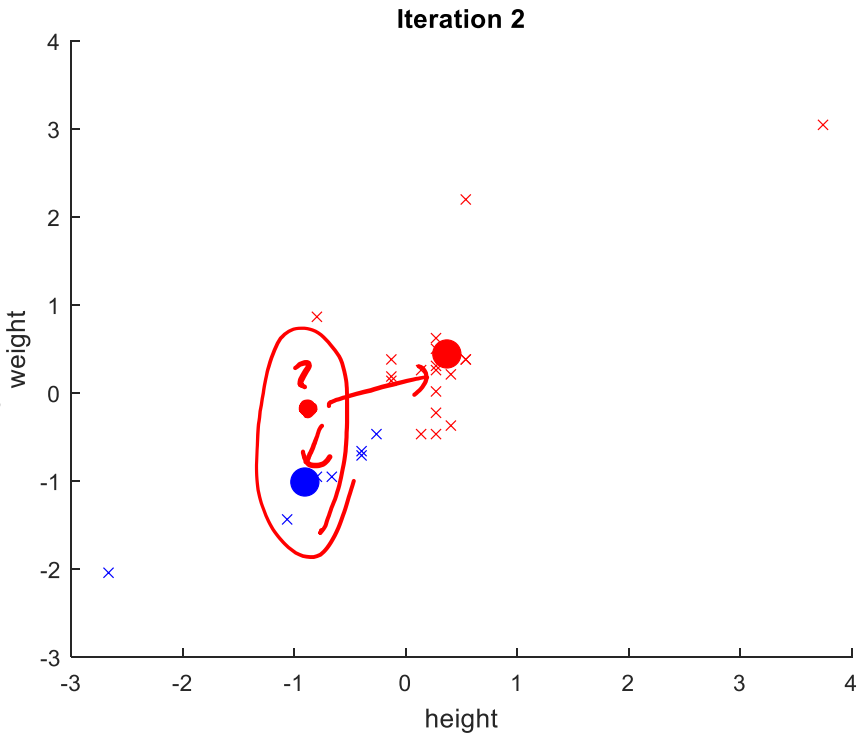
$$\{a_1, a_2\}$$

- If we want to know what those are in our original space we can "undo" standardization:
  - $a_1 \rightarrow (a_1).* \left[\sigma_{height}, \sigma_{weight}\right] + \left[\mu_{height}, \mu_{weight}\right]$
  - $a_1 \rightarrow [70.7895, 177.1579]$
  - $a_2 \rightarrow [61.25, 117.1250]$



Example

- Given a new person with measurements x, we can assign them a shirt size based on the model
  1. Standardize x using the original $\mu_{height}, \mu_{weight}, \sigma_{height}, , \sigma_{weight}$
  2. Compute distance to each reference vector in this standardized space.
  3. Choose the cluster it's closest to.



Derivation of K-Means

- Why is the mean of a cluster, μ_i, the best reference vector a_i for k-means?
- K-Means is based on using the sum of the square of the distances to measure the "goodness" of our solution:

$$J(a_i) = \sum_{x \in C_i} (x - a_i)^2$$

- This is referred to as the ***least squared error.***
- Since x and a_i are actually *vectors*, this equation should be written as:

$$J(a_i) = \sum_{x \in C_i} (x - a_i)(x - a_i)^T$$

Derivation of K-Means

- We want to find a value for the reference vector a_i that minimizes this distance.
- So we'll take the derivative of $J(a_i)$ with respect to a_i

$$\frac{dJ(a_i)}{da_i} = \frac{d}{da_i}\left(\sum_{x \in C_i}(x - a_i)(x - a_i)^T\right) = \sum_{x \in C_i} -2(x - a_i)$$

- Now let's set this equal to zero:

$$\sum_{x \in C_i} -2(x - a_i) = 0$$

- We can re-write this as

$$\sum_{x \in C_i} x = \sum_{x \in C_i} a_i$$

Derivation of K-Means

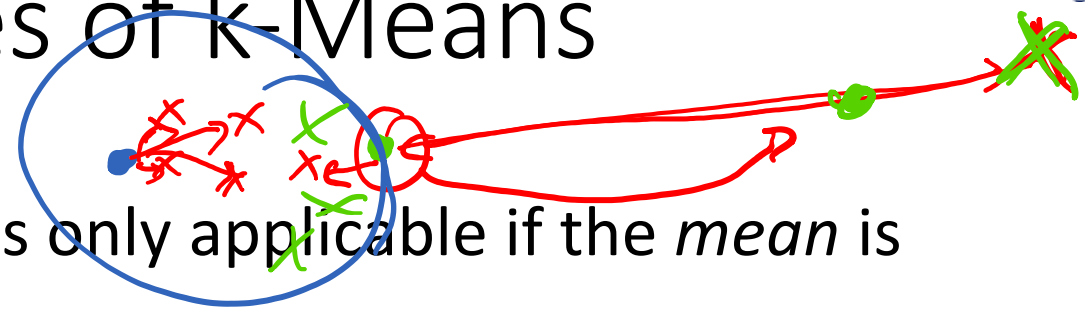
$$\sum_{x \in C_i} x = \sum_{x \in C_i} a_i$$

- Let $|C_i|$ be the number of members in cluster C_i
- Then $\sum_{x \in C_i} a_i = |C_i| a_i$
- Therefor (via substitution):

$$a_i = \left(\frac{1}{|C_i|} \sum_{x \in C_i} x \right)$$

- Which is the definition of the mean of the cluster, $\mu(C_i)$
- So if $a_i = \mu(C_i)$ then we have minimized the least squared error

Weaknesses of k-Means



- The algorithms is only applicable if the *mean* is defined
  - For **categorical** data use k-mode where the centroid is represented by most frequent values
- The user needs to specify k
- The algorithm is sensitive to outliers
  - Outliers are data points that are very far away from other data points
  - Outliers could be errors in the data recording or some special data points with very different values
  - One solution is to use **k-medoids** (which uses the L1 distance and chooses the median of each feature)

Expectation Maximization

Via (Gaussian) Mixture Models

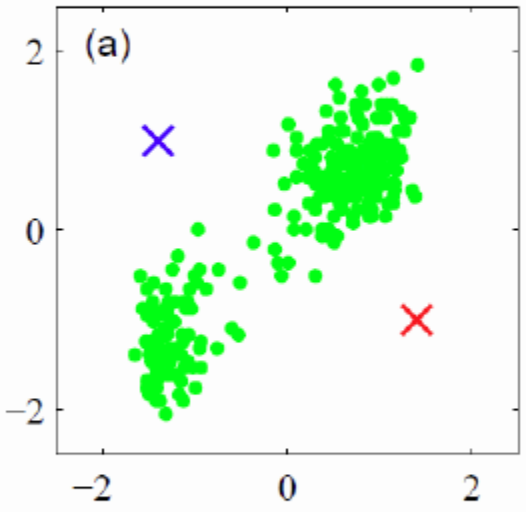
Expectation Maximization

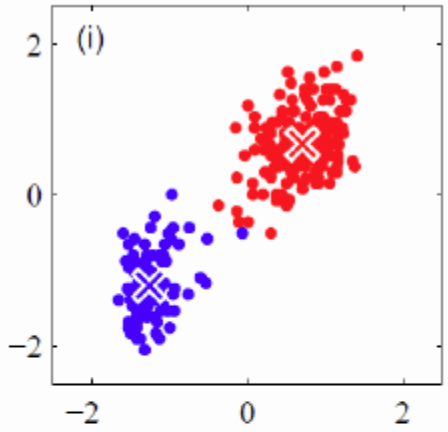
- k-means (medians, etc..) is an instance of an *expectation-maximization* (EM) algorithm.
- These types of algorithms try to find a solution by alternating between two steps:
  1. *Expectation* – Here's we predict out outcome.
     - For k-means, this is assigning each instance to a cluster.
  2. *Maximization* – Here we update our model to maximize/minimize something.
     - For k-means this is moving the reference vector to the mean of the cluster in order to minimize the distance.
     - Conversely we sometimes maximize the log-likelihood

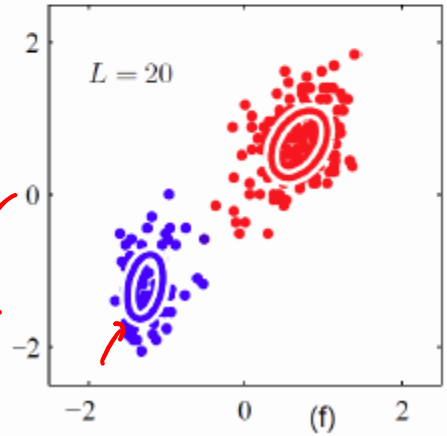
Gaussian Mixture Models

- Another example of expectation-maximization (EM) is discovering parameters for Gaussians
  - Our final "model" is set of Gaussians (Gaussian Mixture Model, or GMM)
- Here rather than identifying clusters by "nearest" centroids, we instead fit a set of k Gaussians to the data
  - This involves updating (M-Step) both the mean and variance of each Gaussian.

K-Means vs GMM





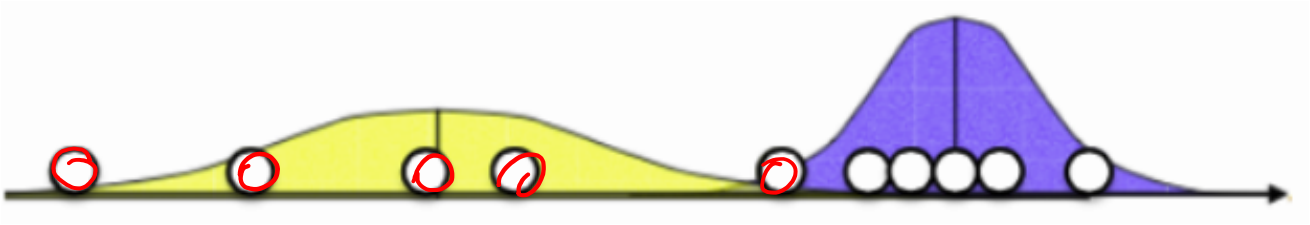


Hard clustering: a data point is assigned a cluster.

Soft clustering: a data point is explained by a mix of multiple Gaussians probabilistically.

Mixture Models in 1D

- Given:
  - Observations $\{X_i\}_{i=1}^{N}$
  - We decided that we want to model the data using $\mathrm{k} = 2$ Gaussians.
- Output
  - Our goal is to find the parameters of the two Gaussians, $\mathcal{N}_1(\mu_1, \sigma_1)$ and $\mathcal{N}_2(\mu_2, \sigma_2)$, pertaining to the two clusters, C_1 and C_2, respectively, that maximize the log likelihood of the data (or minimize some distance)



1D EM for GMM

- Similar to k-means…
- EM for GMM
  1. Start with two random starting vectors μ_1 and μ_2 and compute sample standard deviations σ_1 and σ_2 to initialize Gaussians $\mathcal{N}_1(\mu_1, \sigma_1)$ and $\mathcal{N}_2(\mu_2, \sigma_2)$
  2. Initialize probabilities of each Gaussian. We will need this for the E-Step computations. Typically this will be done uniformly unless there is prior information: $P(\mathcal{N}_i) = \frac{1}{k}$ where k is the number of Gaussians.
  3. (E-Step): For each observation compute the likelihood (expectation) of each Gaussian given the observation.
     - That is, compute $P(\mathcal{N}_1|x)$ and $P(\mathcal{N}_2|x)$
     - We'll see how to do this in the next slide
  4. (M-Step): Adjust $\mathcal{N}_1(\mu_1, \sigma_1)$ and $\mathcal{N}_2(\mu_2, \sigma_2)$, $P(\mathcal{N}_1)$ and $P(\mathcal{N}_2)$ to better fit the estimates (maximize the likelihood).
     - We'll also see how to do this in the next slide as well
  5. Iterate (repeat 3-4) until convergence

Mixture Models in 1D

Computing the expectation $P(\mathcal{N}_i|x)$

- Given $P(x|\mathcal{N}_i)$ and $P(\mathcal{N}_i)$ we can compute $P(\mathcal{N}_i|x)$ using Bayes' Rule!

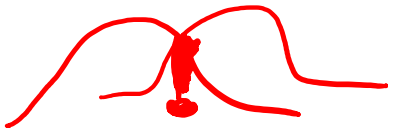
$$P(\mathcal{N}_i|x) = \frac{P(\mathcal{N}_i)P(x|\mathcal{N}_i)}{\sum_{j=1}^{k} P(\mathcal{N}_j)P(x|\mathcal{N}_j)}$$

- Since the denominator is independent of the Gaussian $\mathcal{N}_i$ we can first compute

$$P(\mathcal{N}_i|x) \propto P(\mathcal{N}_i)P(x|\mathcal{N}_i)$$

- $P(\mathcal{N}_i)$ has some current value (to be updated in the M-Step)
- Assuming a Gaussian distribution, $P(x|\mathcal{N}_i)$ is proportional to the probability density function (pdf) as

$$P(x|\mathcal{N}) \propto p(x|\mathcal{N}(\mu,\sigma)) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



- Finally to get $P(\mathcal{N}_i|x)$
  - Let $M = \sum_{j=1}^{k} P(\mathcal{N}_j)P(x|\mathcal{N}_j)$
- Then

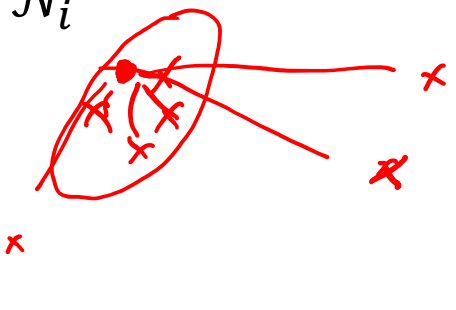
$$P(\mathcal{N}_i|x) = \frac{P(\mathcal{N}_i)P(x|\mathcal{N}_i)}{M}$$

Mixture Models in 1D

To Maximization!

- Just like with k-means, in order to find the new centers (part of the maximization step) we want to compute the average vector of those observations associated with the cluster (from the expectation step).
- Let $P(\mathcal{N}_i|X_j)$ be the probability that observation j belongs to cluster i, which we compute in the expectation step
  - For k-means $P(\mathcal{N}_i|X_i) \in \{0,1\}$
- For convenience, let $\beta_i = \sum_{n=1}^{N} P(\mathcal{N}_i|X_n)$, the total sum of the probably of observations belonging to cluster $\mathcal{N}_i$
- We could re-write $u_i = \frac{1}{|\mathcal{N}_i|} \sum_{x \in \mathcal{N}_i} x$ then as

$$u_i = \frac{\sum_{n=1}^{N} P(\mathcal{N}_i|X_n) X_n}{\beta_i}$$



Mixture Models in 1D

To Maximization!

- In GMM from our expectation step so our update/maximization rule for the mean is:

$$\mu_i = \frac{\sum_{n=1}^{N}(P(\mathcal{N}_i|X_n)X_n)}{\beta_i}$$

- Likewise our update rule for the standard deviation is similar:

$$\sigma_i^2 = \frac{\sum_{n=1}^{N}(P(\mathcal{N}_i|X_n)(X_n-\mu_i)^2)}{\beta_i}$$

- Finally we need to update $P(\mathcal{N}_i)$ which is just the average expectation for this class:

$$P(\mathcal{N}_i) = \frac{\beta_i}{N}$$

total # of samples

1D EM for GMM

Putting this all together…

1. Start with two random starting vectors μ_1 and μ_2 and compute sample standard deviations σ_1 and σ_2 to initialize Gaussians $\mathcal{N}_1(\mu_1, \sigma_1)$ and $\mathcal{N}_2(\mu_2, \sigma_2)$
2. Initialize probabilities of each Gaussian. Typically this will be done uniformly unless there is prior information:

$$P(\mathcal{N}_i) = \frac{1}{k}$$

3. (E-Step): For each observation compute the probability of each Gaussian given the observation

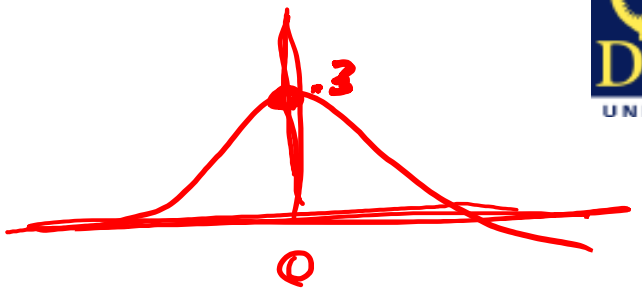
$$P(\mathcal{N}_1|x) \text{ and } P(\mathcal{N}_2|x)$$

4. (M-Step):
   a. Using the expectations computed in the E-Step, update the Gaussian Parameter μ_i, σ_i
   b. Update the priors $P(\mathcal{N}_i)$
5. Iterate (repeat 3-4) until convergence

Example: 1D GMM

- Datapoints (D=1)
  - $X = [.78, .72, .66, .51, .86, .83, .53, .32, .79, .97]^T$
- Let's assume two Gaussians ($k = 2$)
- Pick two samples as initial means
  - μ_1=.78, μ_2=.51
- Use the standard deviation of the data relative to the means chosen above for the initial standard deviation
  - $\sigma_1 = \sqrt{\frac{\sum_{i=1}^{N}(x_i - u_1)^2}{N}}$=0.2025., σ_2=0.2628
- Assuming uniform distribution of classes (initially)
  - $P(\mathcal{N}_1) = P(\mathcal{N}_2) = 0.5$

Example: 1D GMM



2.000001

1.99959

2

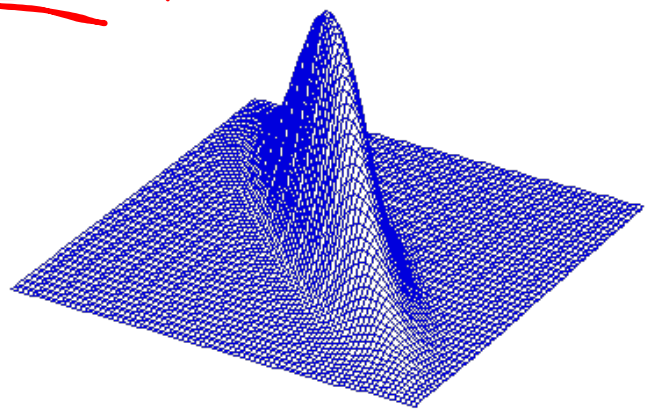
- Start the EM process!
- (E-Step)
  - For data point 0.78
    - $p(0.78|\mathcal{N}_1) = \frac{1}{\sigma_1\sqrt{2\pi}} e^{-(0.78-\mu_1)^2/(2\sigma_1^2)}$
    - $p(0.78|\mathcal{N}_1) = 1.97$
    - $p(0.78|\mathcal{N}_2) = 0.8955$
    - $P(\mathcal{N}_1|0.78) \propto p(0.78|\mathcal{N}_1)P(\mathcal{N}_1) = 0.985$
    - $P(\mathcal{N}_2|0.78) \propto p(0.78|\mathcal{N}_2)P(\mathcal{N}_2) = 0.4478$
    - Normalize so $P(\mathcal{N}_1|0.78) + P(\mathcal{N}_2|0.78) = 1$
      - $P(\mathcal{N}_1|0.78) = 0.6875$
      - $P(\mathcal{N}_2|0.78) = 0.3125$
  - Etc.. for all other points

1D GMM

- (M-Step)
  - $\beta_1 = \sum_{n=1}^{N} P(\mathcal{N}_1 | X_n)$
  - $\mu_1 = \frac{\sum_{n=1}^{N} (P(\mathcal{N}_1 \mid X_n) X_n)}{\beta_1}$
  - $\sigma_1^2 = \frac{\sum_{n=1}^{N} (P(\mathcal{N}_1 \mid X_n)(X_n - \mu_1)^2}{\beta_1}$
  - $P(\mathcal{N}_1) = \frac{\beta_1}{N}$
- And continue until convergence…

Mixture Models in D>1

- If we have more than 1 dimensional data (as is usually the case) then we must replace the standard deviation σ_i with the covariance matrix Σ_i
- So now for each model we need:
  - Prior $P(\mathcal{N}_i)$
  - Mean **vector** μ_i of length $1 \times D$
  - **Covariance matrix** Σ_i of size $D \times D$

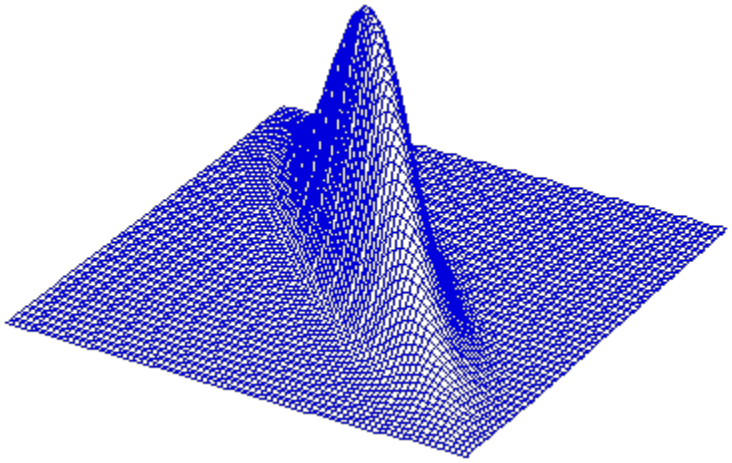


Mixture Models in D>1

- Then for the estimate $p(x|\mathcal{N})$ is

$$p(x|\mathcal{N}_i) = \frac{1}{\sqrt{(2\pi)^D |\Sigma_i|}} \exp\left(-\frac{(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)}{2}\right)$$

  - Where $|\Sigma_i|$ is the determinant of Σ_i



Mixture Models in D>1

- Maximization
  - Update mean for feature j of Gaussian i (using updated prior)

$$\mu_{i,j} = \frac{\sum_{n=1}^{N} P(\mathcal{N}_i|X_n)X_{n,j}}{\beta_i}$$

  - Update the covariance between features j and t of Gaussian i:

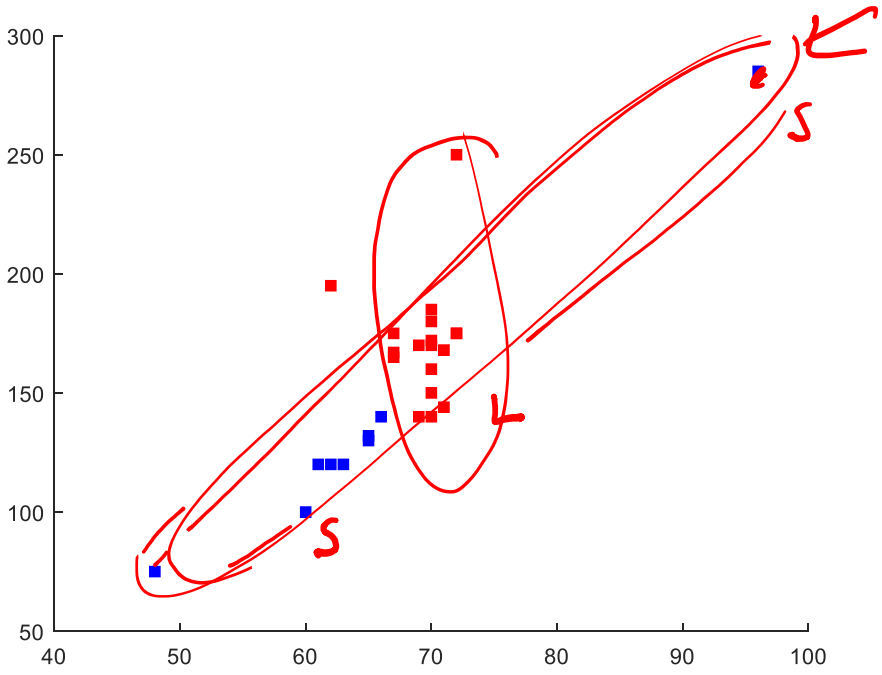
$$(\Sigma_i)_{j,t} = \frac{\sum_{n=1}^{N} P(\mathcal{N}_i|X_n)(X_{n,j} - \mu_{i,j})(X_{n,t} - \mu_{i,t})}{\beta_i}$$

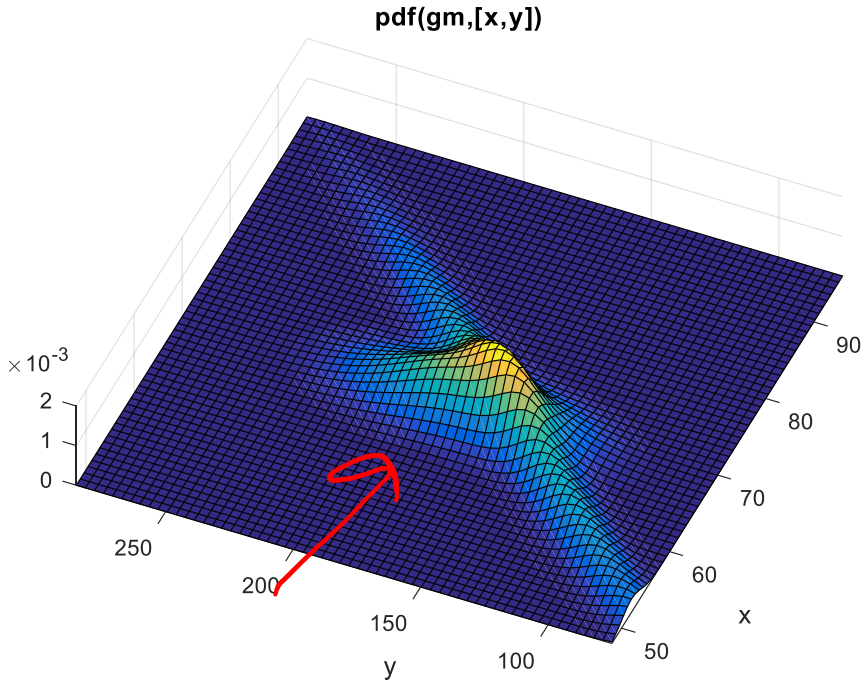
  - Update prior for Gaussian i:

$$p(\mathcal{N}_i) = \frac{\beta_i}{N}$$

Multivariate GMM

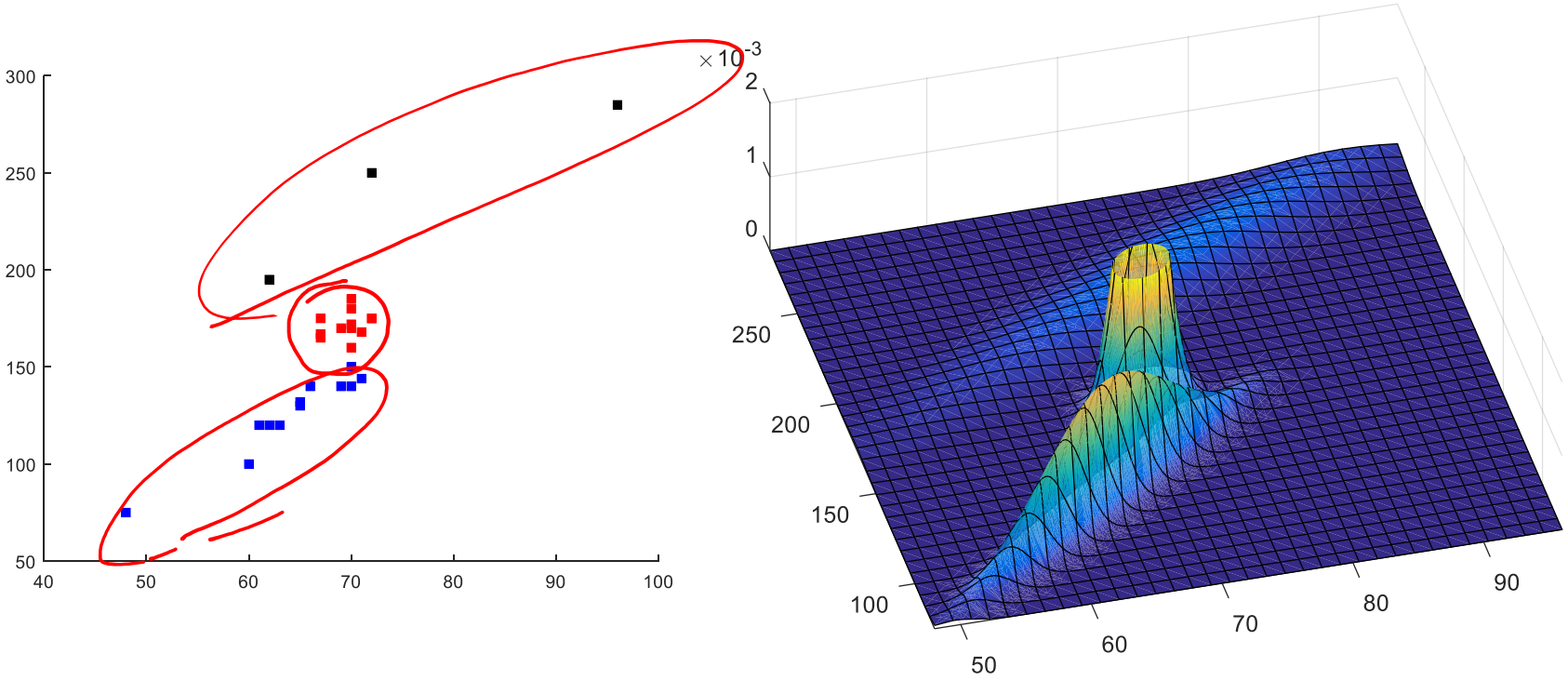
- Example: From the 2D t-shirt data K=2



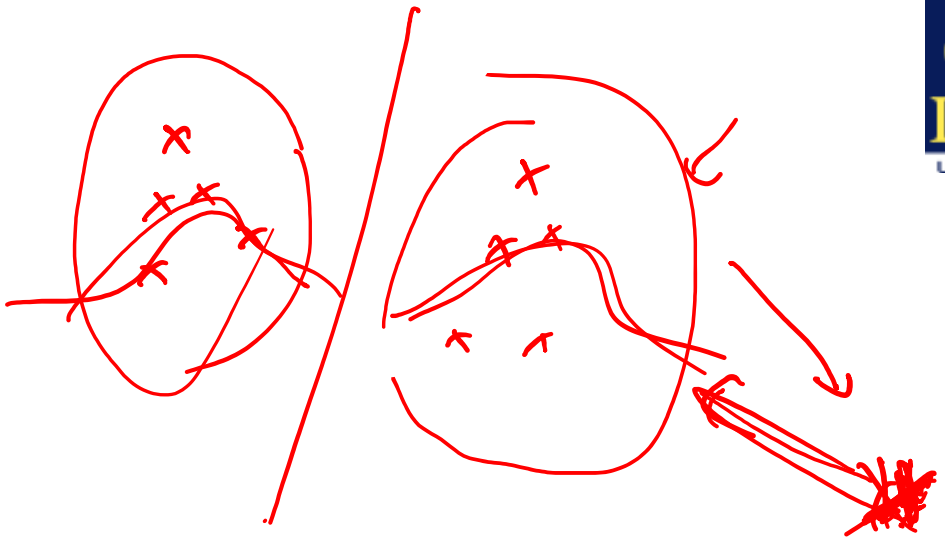


Multivariate GMM

- With K=3 clusters ….



EM Summary

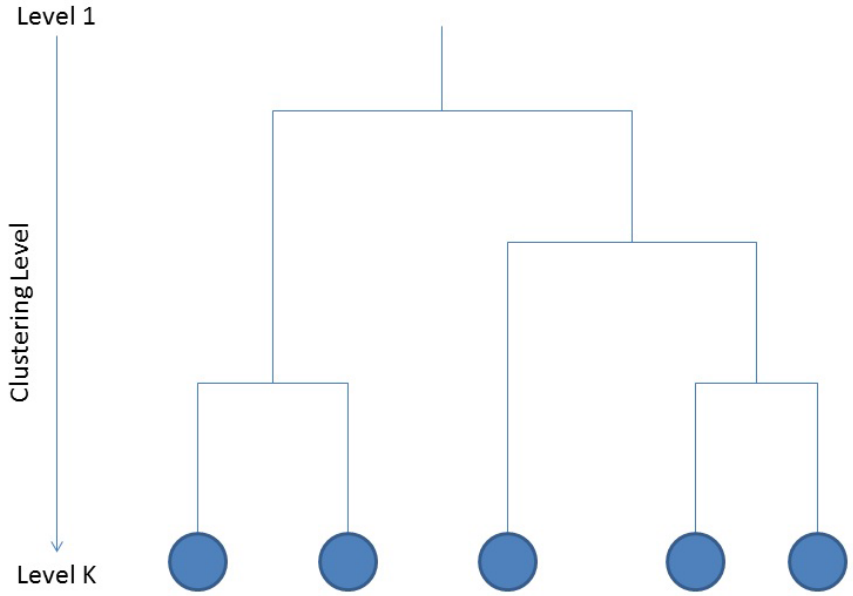


- Mixture Models
  - Advantages
    - If the assumed data distribution is correct it works well
  - Disadvantages
    - If the assumed data distribution is wrong, results can be quite bad
      - In particular type of distribution and number of components
- Expectation-Maximization is a general algorithm for parameter estimation
  - Like gradient descent
  - Remember k-means is actually just a version of EM

Hierarchical Clustering

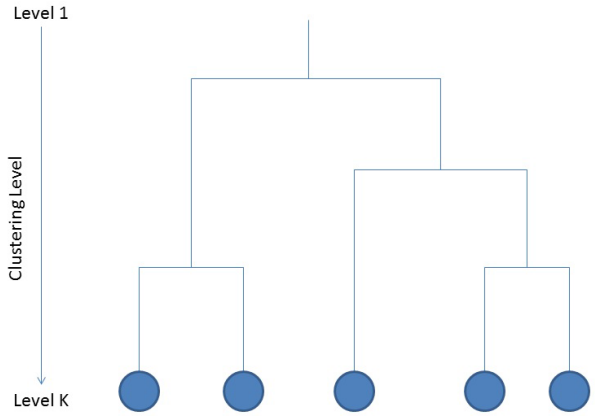
Hierarchical Agglomerative Clustering (HAC)

- With hierarchical agglomerative clustering we're building a clustering binary tree
- This can be done either bottom up, or top down
- At each iteration we have a new set of clusters



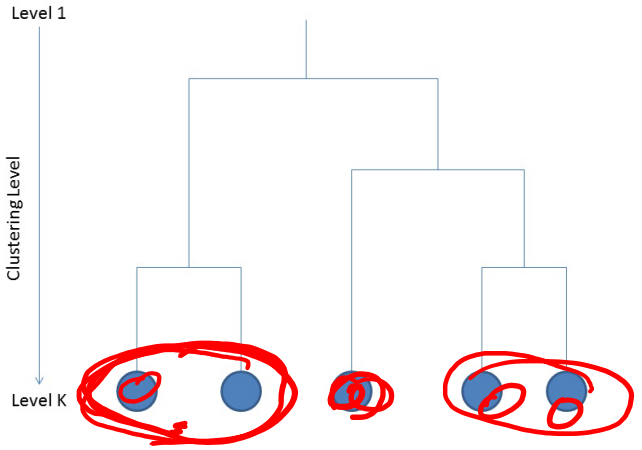
Hierarchical Agglomerative Clustering (HAC)

- Top-Down Approach
  - At first everything is part of a single cluster.
  - Then split this into two clusters based on some criterial
  - Now choose one of these two clusters, and split it into two
    - Now we have three total clusters
  - Etc.. until each cluster only has one observation in it (called a singleton)



Hierarchical Agglomerative Clustering (HAC)

- Bottom-Up Approach
  - At first everything is its own cluster
    - If there's N observations, then there's N clusters
  - Choose two of these clusters to merge
    - Now there's $N - 1$ clusters
  - From these $N - 1$ clusters, choose two to merge
    - Now there's $N - 2$ clusters
  - Etc.. until there is only one cluster
- Let's just look at the bottom-up approach.



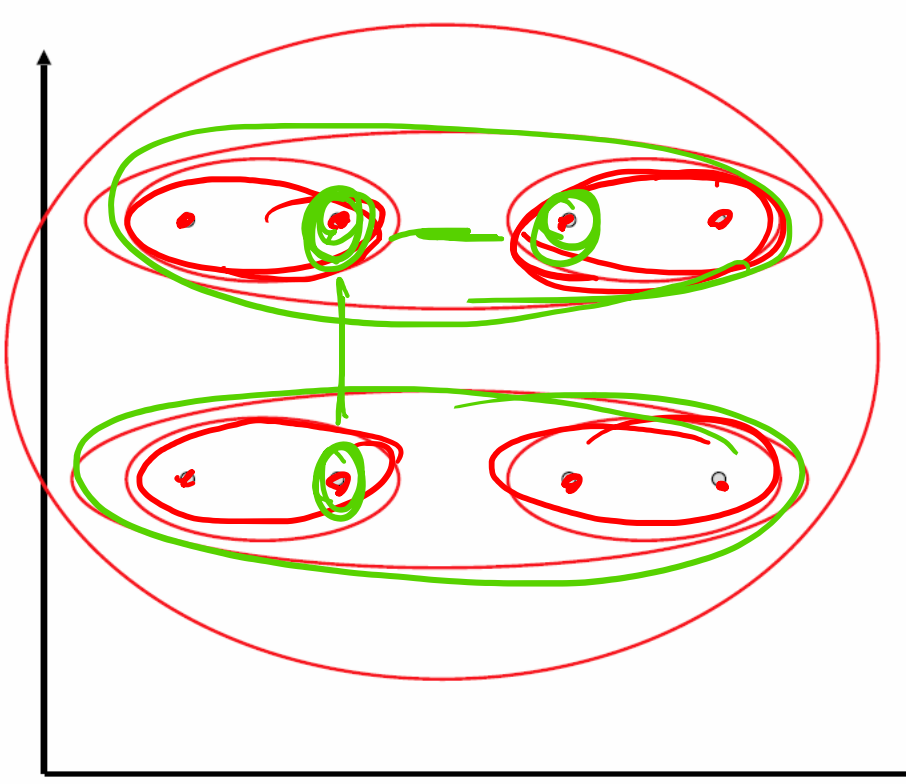
Closest Pair of Clusters

- When deciding which to clusters to merge we typically need to determine which two clusters are closest.
- There's many variants to defining closest pair of clusters
  - Single link – Similarity of the most similar
  - Complete link – Similarity of the furthest points
  - Average link – Average pair-wise similarity between clusters

Single Link Example

Single link – Similarity of the most similar

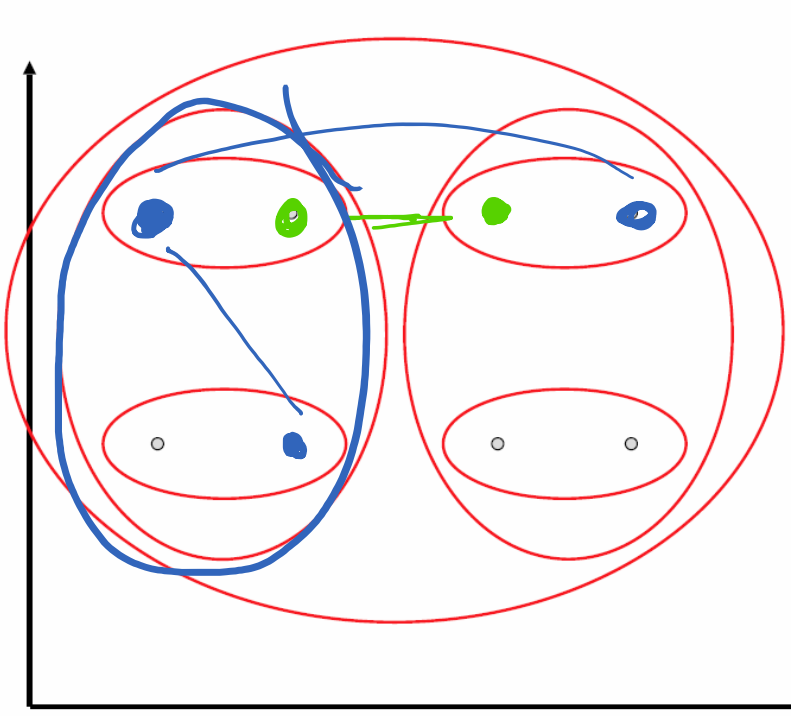
$$sim(C_i, C_j) = \max_{x \in C_i, y \in C_j} sim(x, y)$$



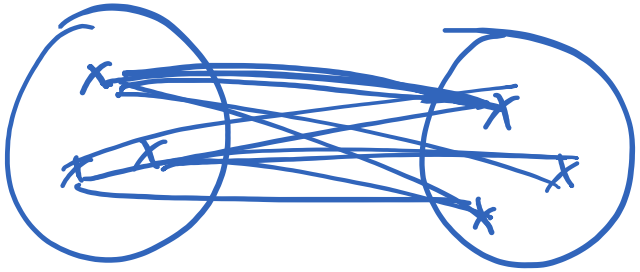
Complete Link HAC

Complete link – Similarity of the furthest points

$$sim(C_i, C_j) = \min_{x \in C_i, y \in C_j} sim(x, y)$$



Average HAC



- Average Link
  - Compromise between single and complete link
  - Average over all pairs *between* the two original clusters

$$sim(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{y \in C_j} sim(x, y)$$

HAC Pseudocode

- Let $C^{(k)}$ be the set of clusters at clustering level k and $C_i^{(k)}$ to be the i^{th} cluster in cluster-set $C^{(k)}$
- Initialize cluster level, $k = N$ and cluster-set $C_i^{(k)} = \{x_i\}$ for $i = 1, \ldots N$
- While $k \geq 1$
  - Find closest clusters in $C^{(k)}$, $C_a^{(k)}$, $C_b^{(k)}$ according to some metric.
  - Create new cluster set $C^{(k-1)} = C^{(k)}$
  - Remove from $C^{(k-1)}$, $C_a^{(k-1)}$ and $C_b^{(k-1)}$ and add $C_a^{(k)} \cup C_b^{(k)}$
  - $k \rightarrow k - 1$

Choosing The Clustering Level

- Ok so we have a HAC tree.
- What can we use it for?
- If we know how many clusters we want (if the problem dictates k), then we just choose the level of the tree that has that many clusters.
- What if we don't know it a-priori?

Clustering Quality

- We need to provide some way to measure the quality of a given clustering
- A good clustering will produce clusters in which
  - The intra-class (that is, intra-cluster) similarity is high
  - The inter-class similarity is low

Intra-Cluster Distance

- For a given cluster i and a chosen distance/similarity function d, the follow computes the average pairwise intra-cluster distance G_i

$$G_i = \frac{\sum_{x,y \in C_i} d(x,y)}{(2|C_i|)}$$

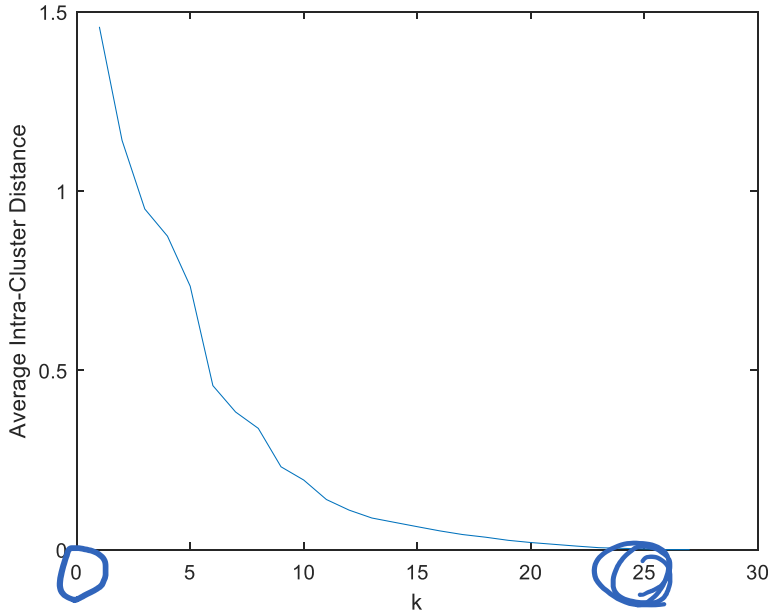
- The weighed (by cluster size) average intra-cluster distance for clustering level j is then:

$$W_j = \frac{\sum_{i=1}^{j} |C_i| G_i}{N}$$

- One idea might be to look at how the weighed average intra-cluster distance varies as a function of the number of clusters.

Graph Based Approaches

- Below is a graph showing the weighted average intra-cluster distance
- There are 30 observations
  - Clustering level 1 has everything in one cluster, and thus a large weighted intra-cluster distance
  - Cluster level 27 has everything as its own cluster, and thus a weighed average intra-cluster distance of zero.



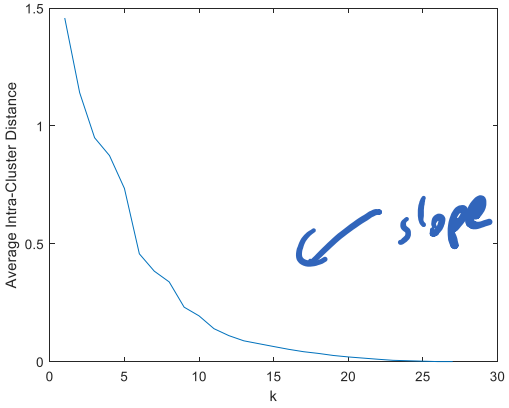
Graph Based Approaches

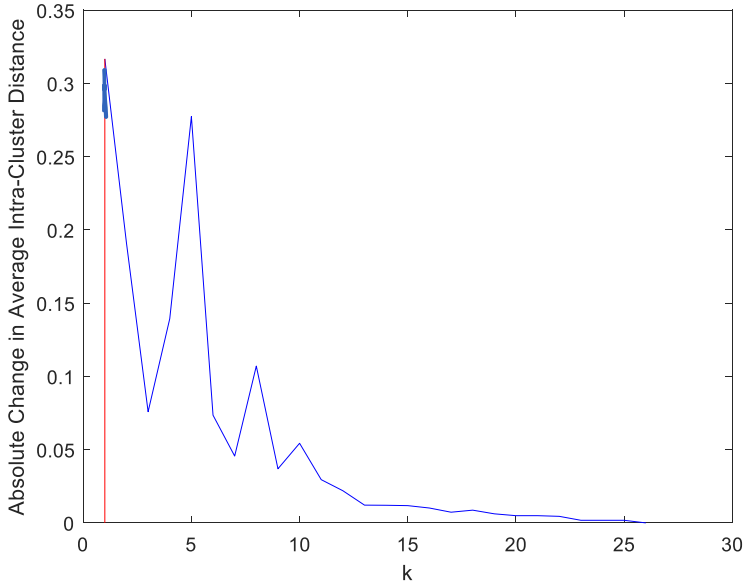
- Obviously just choosing the minimum of this isn't all that useful
  - It will always choose $k = N$
- How about the slope?
- The slope at location W_j is:

$$W_j' = \frac{(W_{j+1} - W_{j-1})}{2}$$

- Maybe select the place where there's the steepest absolute slope?

$$k = \underset{j}{\operatorname{argmax}}(|W_j'|)$$

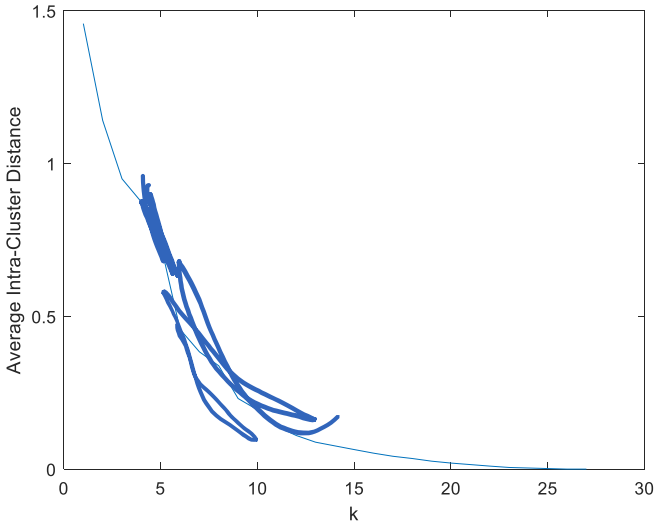


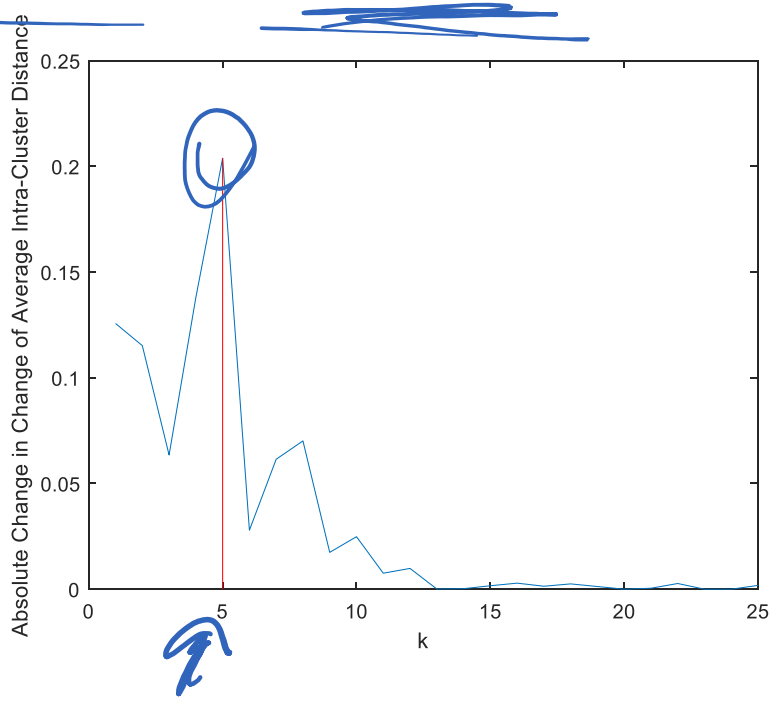


Graph Based Approaches

- How about the place where there's the greatest *change in slope*
  - Maximize the curvature
  - Second derivative!

$$W_j'' = \frac{(W'_{j+1} - W'_{j-1})}{2} = \frac{\left(\frac{(W_{j+2} - W_j)}{2} - \frac{(W_j - W_{j-2})}{2}\right)}{2} = \frac{(W_{j+2} - 2W_j + W_{j-2})}{4}$$





External Criteria for Clustering Quality

- Of course in the end we probably want to figure out how our algorithm is behaving.
- Maybe we can ask some people “after the fact” to do the clustering task and compare theirs to ours.
- Still unsupervised since the labels didn’t influence how we clustered. We just used it for evaluation

External Evaluation of Cluster Quality

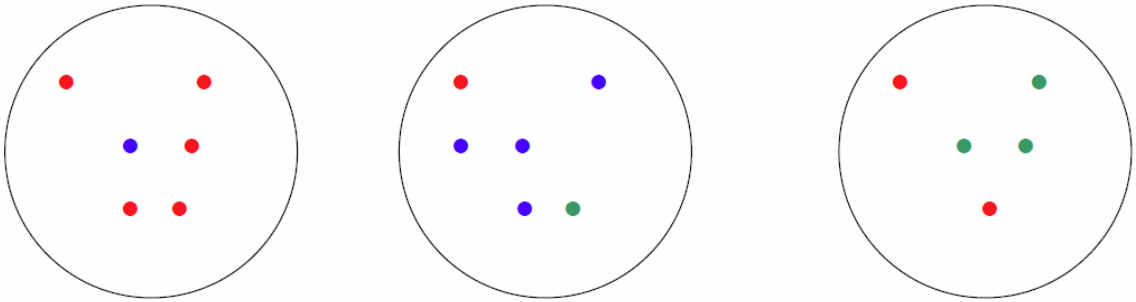
- Simple measure: **purity**
  - Let N_{ij} be the number of instances of (supervised) label *j* within cluster C_i
  - The purity of cluster C_i is then defined as

$$Purity(C_i) = \frac{1}{|C_i|} \max_j N_{ij}$$

  - Then we can define the average purity of this clustering as

$$Purity = \frac{1}{N} \sum_{i=1}^{k} |C_i| Purity(C_i)$$

Purity Example



Cluster I Cluster II Cluster III

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

$$\text{Total Purity} = \frac{1}{17}\left(6 * \frac{5}{6} + 6 * \frac{4}{6} + 5 * \frac{3}{5}\right) = \frac{12}{17} \approx 70\%$$

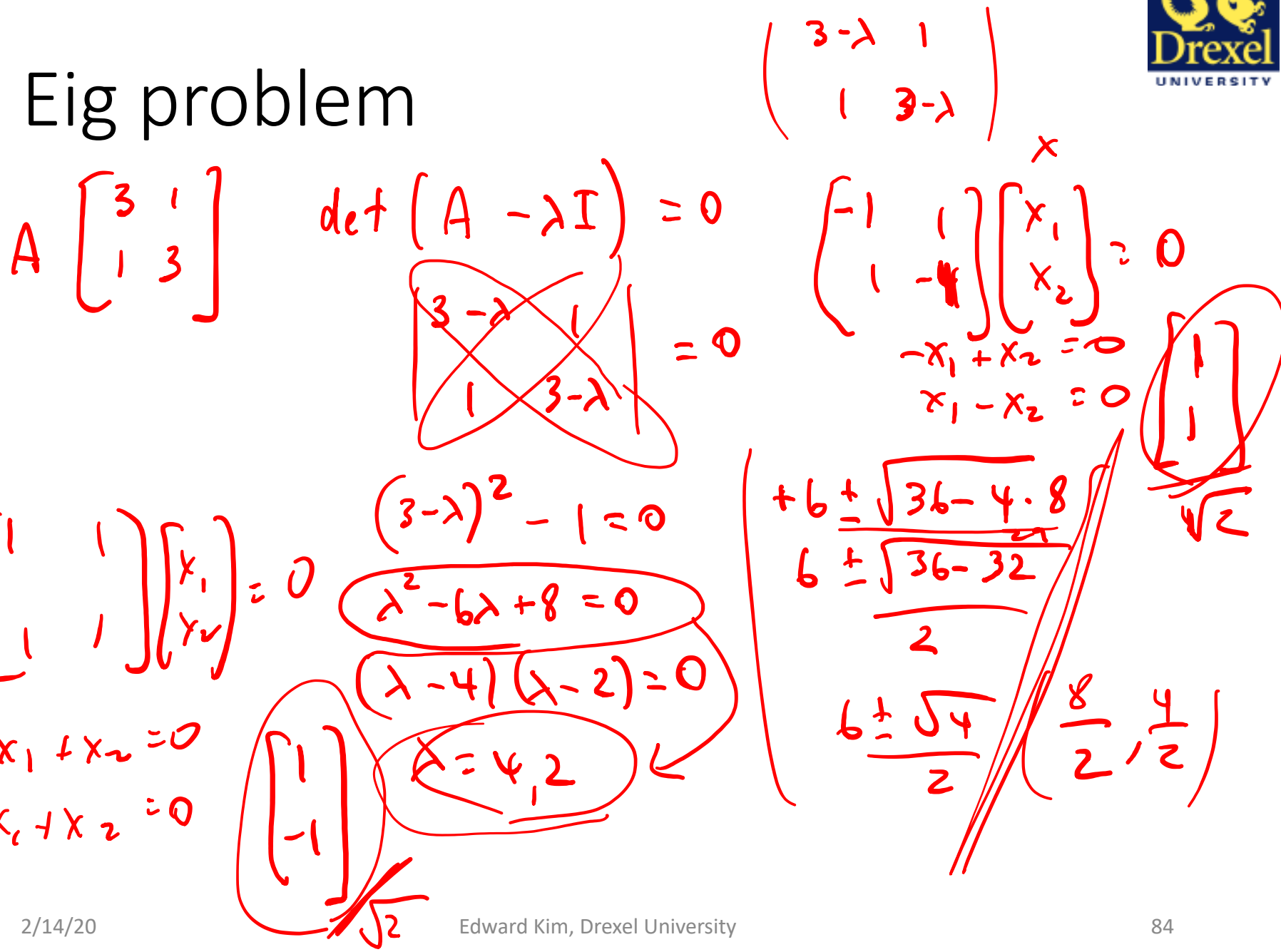
External Evaluation of Cluster Quality

- Purity is biased because having $k = N$ clusters maximizes purity
  - But it can be useful in compare methods with the same clustering level
- Other measurements include
  - Entropy of a cluster using the supervised labels.
  - Mutual information between supervised labels and clusters
  - Rand Index

Resources

- http://www.umass.edu/landeco/teaching/multivariate/schedule/cluster1.pdf

Eig problem


A
det(A − λI) = 0
= 0
(3−λ)² − 1 = 0
λ² − 6λ + 8 = 0
(λ − 4)(λ − 2) = 0
λ = 4, 2
x₁ + x₂ = 0
x₁ + x₂ = 0
√2
−x₁ + x₂ = 0
x₁ − x₂ = 0
√2
6 ± √(36 − 4·8) / 2
6 ± √(36 − 32) / 2
6 ± √4 / 2
8/2, 4/2

Theory 1

1. 1a

$$X = \begin{bmatrix} -2 \\ -5 \\ -3 \\ 0 \\ -6 \\ -2 \\ 1 \\ 5 \\ -1 \\ 3 \end{bmatrix} \quad ; y = \begin{bmatrix} 1 \\ -4 \\ 1 \\ 3 \\ 11 \\ 5 \\ 0 \\ -1 \\ -3 \\ 1 \end{bmatrix}$$

$$X_{stand} = \begin{bmatrix} 1 & -0.294 \\ 1 & -1.177 \\ 1 & -0.588 \\ 1 & 0.294 \\ 1 & 0.588 \\ 1 & 1.765 \\ 1 & 0 \\ 1 & 1.177 \end{bmatrix}$$

$$\theta = (X^T X)^{-1}(X^T y)$$

$$\theta = \begin{bmatrix} 1.4 \\ -1.504 \end{bmatrix}$$

$$\theta_0 = 1.4$$

$$\theta_1 = -1.504$$

$$g(x) = (x-1)^4$$

$$\frac{x}{dx} g(x) = 4(x-1)^3$$

$$\text{Gradient} = 4(x-1)^3$$

$$g'(x) = 4(x-1)^4$$

$$4(x-1)^4 = 0$$

$$4(1-1)^4 = 0$$

$$4(0)^4 = 0$$

$$0 = 0$$

Global minima $= 1$, which is the point at which $g'(x) = 0$

Theory 2

Y	x_1	x_2	Count
+	T	T	3
+	T	F	4
+	F	T	4
+	F	F	1
-	T	T	0
-	T	F	1
-	F	T	3
-	F	F	5

$$H(P_{(v_1)}, P_{v_2}) = \sum_{i=1}^{2} = -(P_i)log_2(P_i)$$
$$H(\tfrac{4}{7}, \tfrac{3}{7}) = -\tfrac{4}{7}log_2(\tfrac{4}{7}) + \tfrac{-3}{7}log_2(\tfrac{3}{7})$$
$$= \tfrac{-4}{7}(-0.8073) + \tfrac{-3}{7}(-1.2223)$$
$$= 0.4613 + 0.5238$$
$$= 0.985$$

$$E(H(A)) = \sum_{i=1}^{K} \frac{p_i+n_i}{p+n} H(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i})$$

$$H(x_1) = \frac{8}{21}(H(\frac{7}{8}, \frac{1}{8})) + \frac{13}{21}(H(\frac{5}{13}, \frac{8}{13}))$$

$$= \frac{8}{21}(\frac{-7}{8}log_2(\frac{7}{8}) + \frac{-1}{8}log_2(\frac{1}{8})) + \frac{13}{21}(\frac{-5}{13}log_2(\frac{5}{13}) + \frac{-8}{13}log_2(\frac{8}{13}))$$

$$= \frac{8}{21}(0.544) + \frac{13}{21}(0.961)$$

$$= 0.802$$

$$H(x_2) = \frac{10}{21}(H(\frac{7}{10}, \frac{3}{10})) + \frac{11}{21}(H(\frac{5}{11}, \frac{6}{11}))$$

1

$$= \frac{10}{21}(\frac{-7}{10}log_2(\frac{7}{10}) + \frac{-3}{10}log_2(\frac{3}{10})) + \frac{11}{21}(\frac{-5}{11}log_2(\frac{5}{11}) + \frac{-6}{11}log_2(\frac{6}{11}))$$

$$= \frac{10}{21}(0.8813) + \frac{11}{21}(0.9940)$$

$$= 0.940$$

$$IG(x_1) = entropy - H(x_1)$$

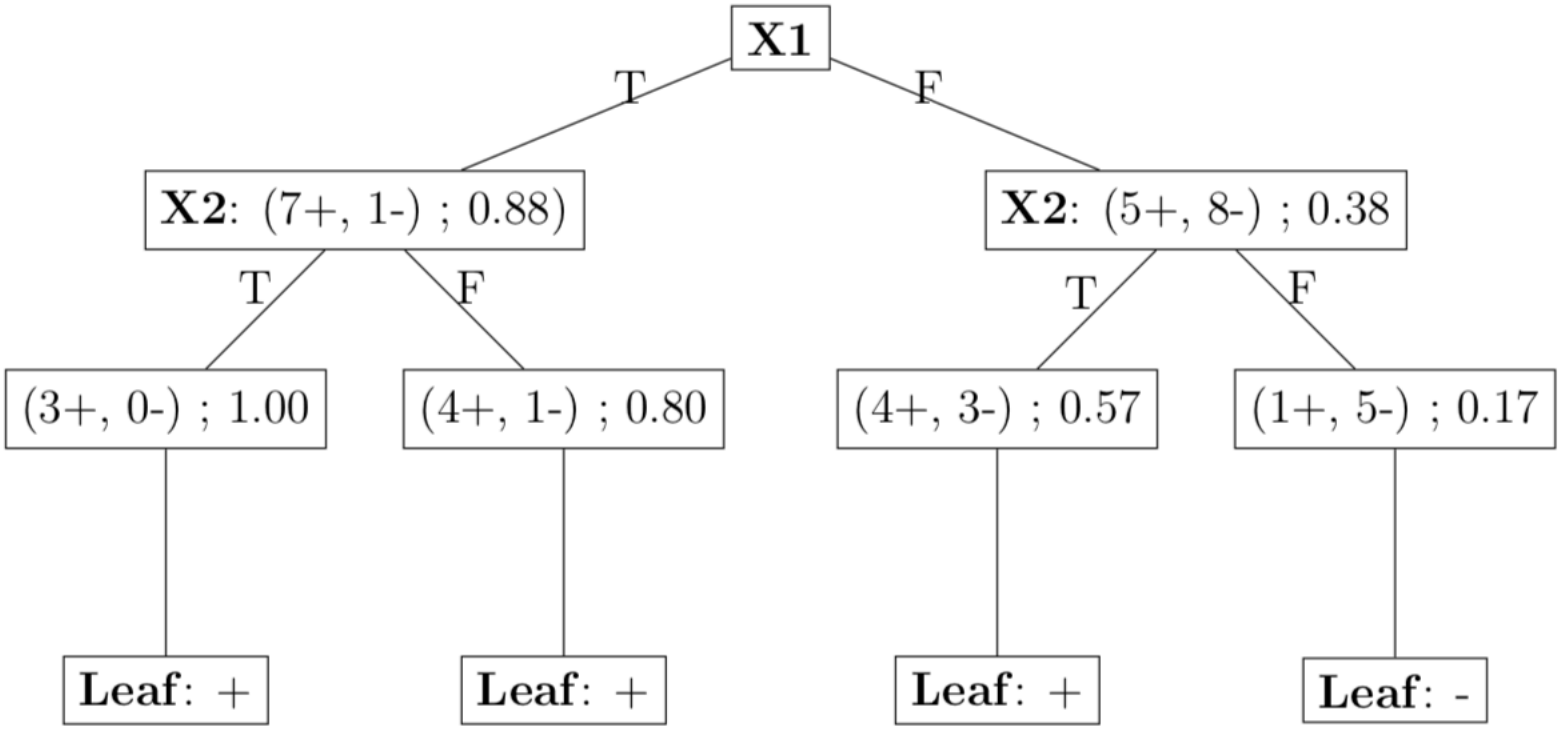
$$= 0.985 - 0.802$$

$$= 0.183$$

$$IG(x_2) = entropy - H(x_2)$$

$$= 0.985 - 0.940$$

$$= 0.045$$


X1
T
F
X2: (7+, 1-) ; 0.88)
X2: (5+, 8-) ; 0.38
T
F
T
F
(3+, 0-) ; 1.00
(4+, 1-) ; 0.80
(4+, 3-) ; 0.57
(1+, 5-) ; 0.17
Leaf: +
Leaf: +
Leaf: +
Leaf: -