BROWN
Computer Science

CS1951A: Data Science

Lecture 15: Unsupervised learning

Lorenzo De Stefani
Spring 2022

Today

- Supervised vs. Unsupervised Learning
- Clustering with K-Means
- The Expectation-Maximization paradigm
  - Soft K-Means
  - EM for weighting weak experts in mail spam detection

Supervised vs. Unsupervised Learning

- Supervised: Traning data has explicit labels
  - Sentiment analysis—review text -> star ratings
  - Image tagging—image -> caption
- Unsupervised: Traning data does not have explicit labels
  - Clustering—find groups similar customers
  - Dimensionality Reduction—find features that differentiate individuals

Many variations

- Semi Supervised—Combining large amounts of unlabelled with smaller amounts of labelled (pretraining)

- Weakly/Distantly Supervised—using noisy labels or partial labels (bootstrapping, automatically-labeled data)

- Reinforcement Learning—label on the result of a sequence of actions, but not on each action (games, robotics)

- …

Supervised vs. Unsupervised Learning

- Supervised: Traning data has explicit labels
  - Sentiment analysis—review text -> star ratings
  - Image tagging—image -> caption
- Unsupervised: Traning data does not have explicit labels
  - Clustering—find groups similar customers
  - Dimensionality Reduction—find features that differentiate individuals

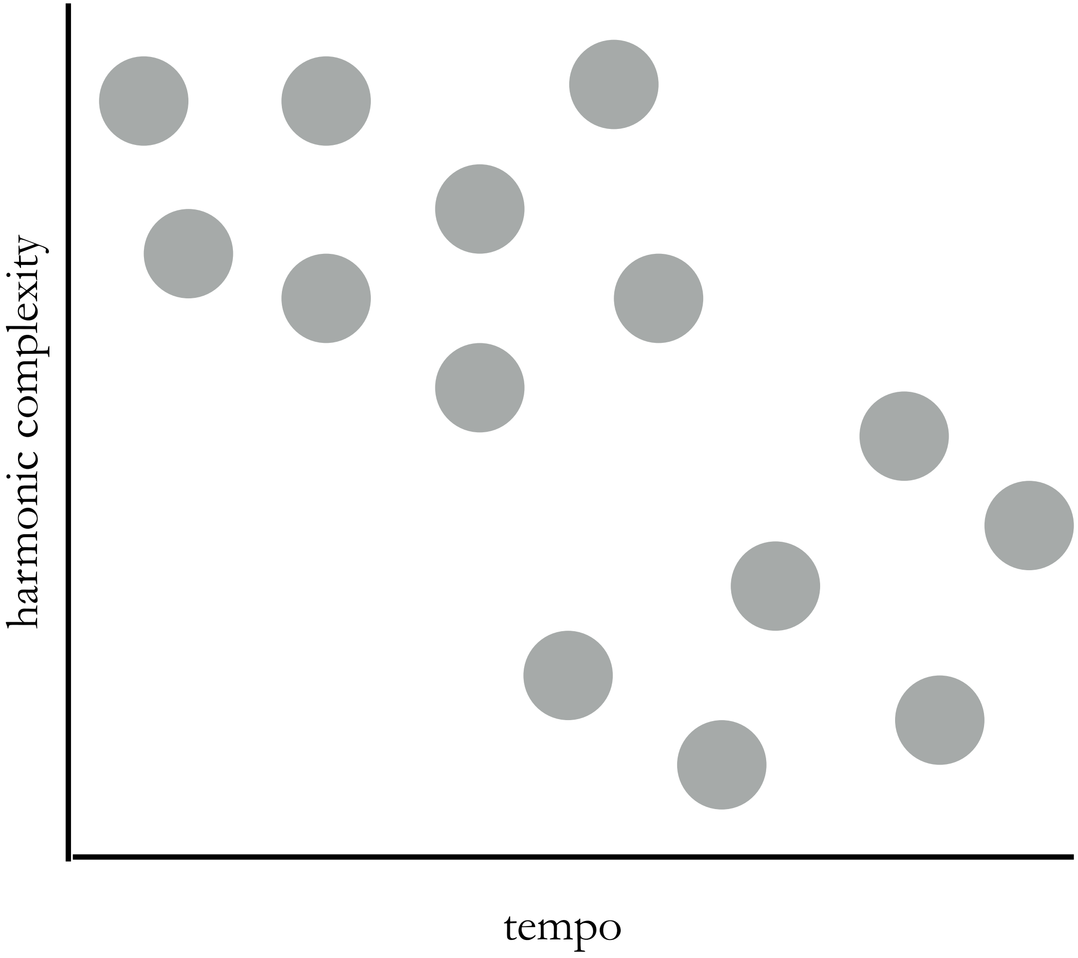
Unsupervised Learning

- "Finding structure in data" (vs. predicting labels)
  - Learning the distribution of the independent variables
- In data science, this is typically for "exploratory analysis".
- Or for preprocessing/featurizing
- In ML, right now, used extensively for "pretraining"
  - (e.g. autoencoding, dimensionality reduction, language modeling*)

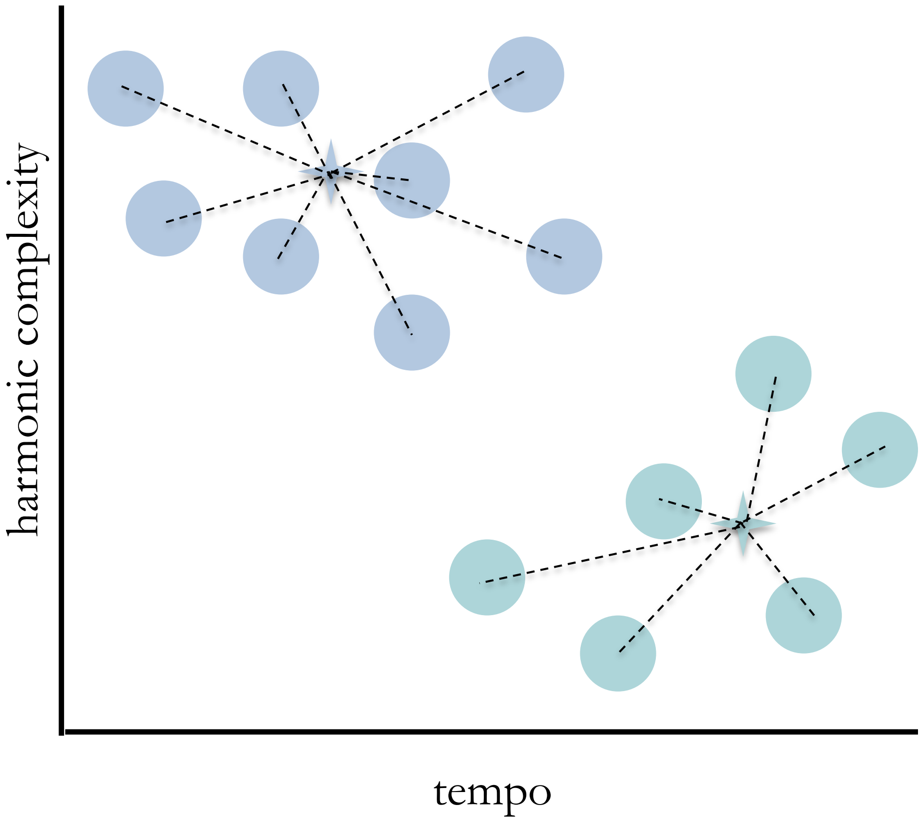
Clustering Scenario

- Find groups of customers with similar tastes
  - Find topics within a set of news articles
  - Find genres within a library of music
  - Extrapolating: make predictions about your new setting based on behavior of similar settings
- Think of it as partition/assign costumers to groups in which they fit well

Clustering

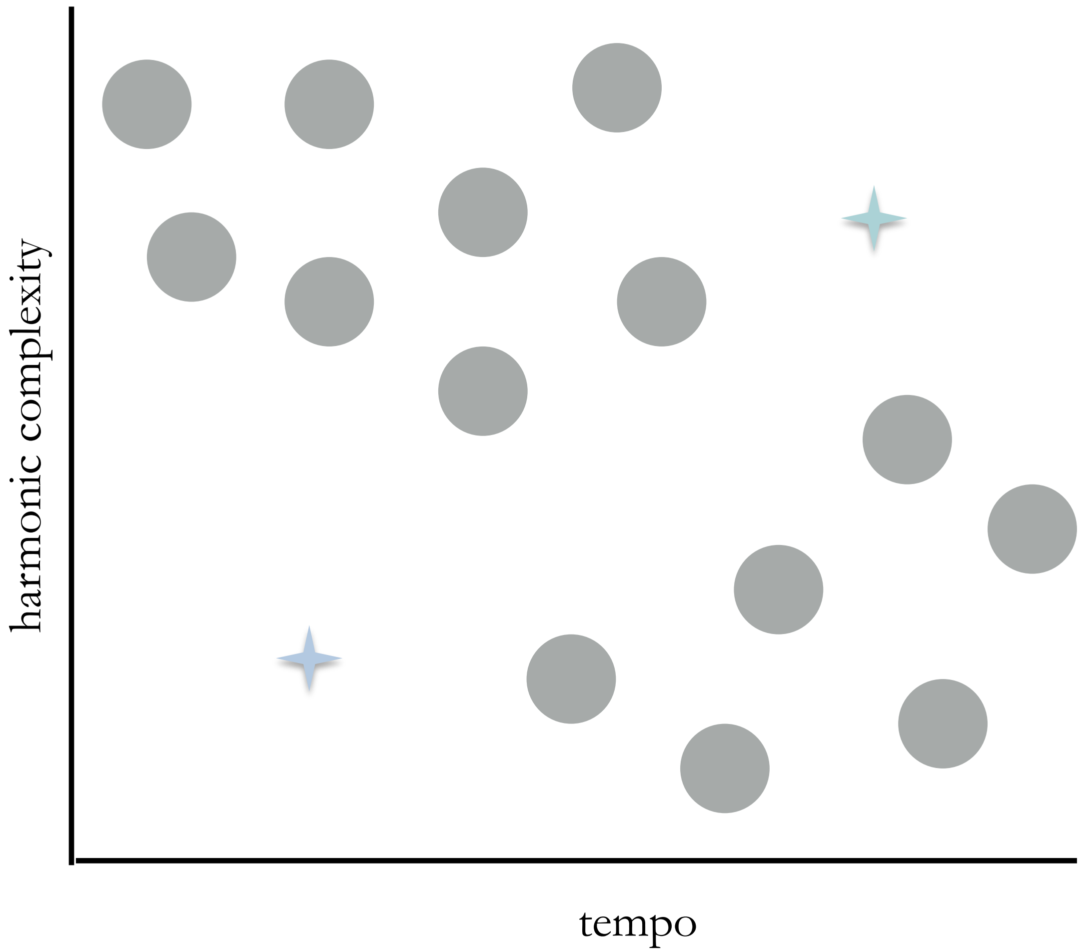


Clustering



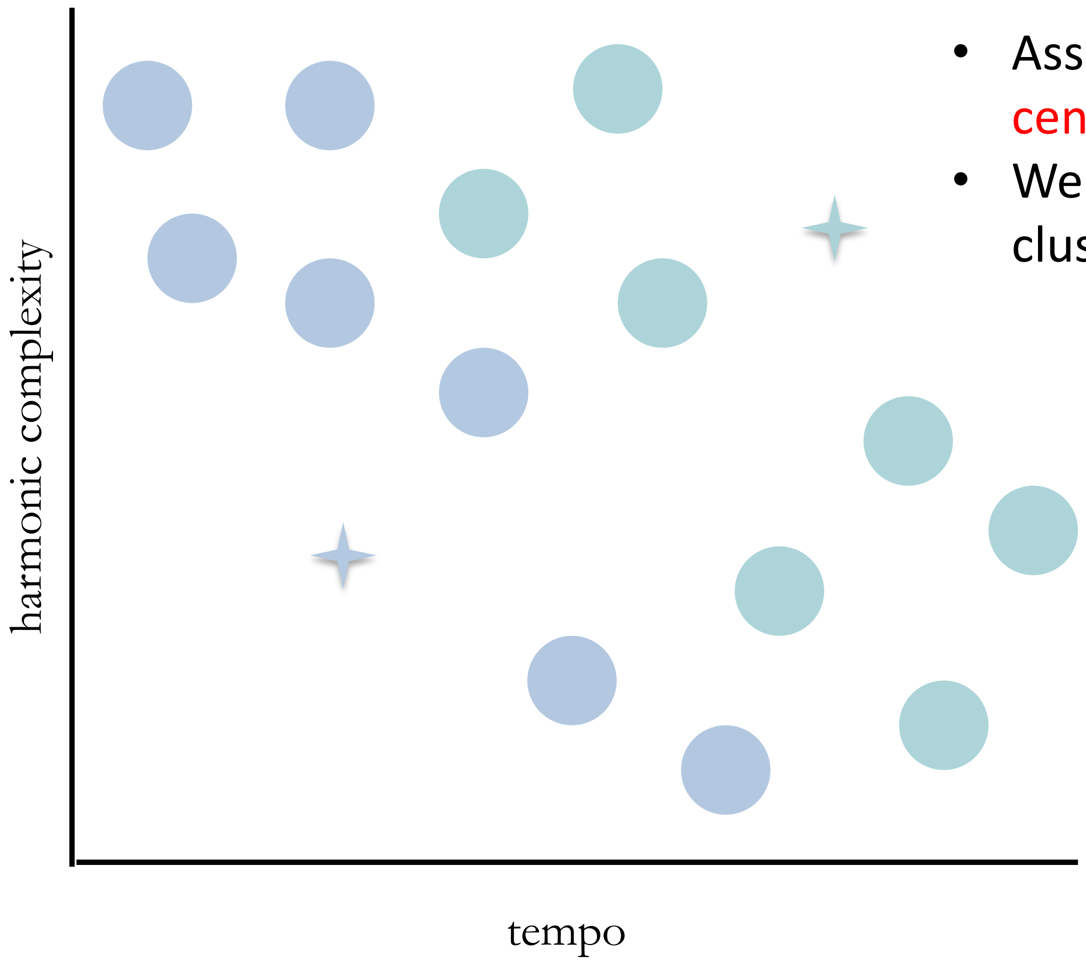
- Each cluster has a center:
  - Points are assigned to the cluster whose center is the closest
  - Ties are broken arbitrarily or by default
- To assign the clusters correctly we essentially need to find good centers

K Means



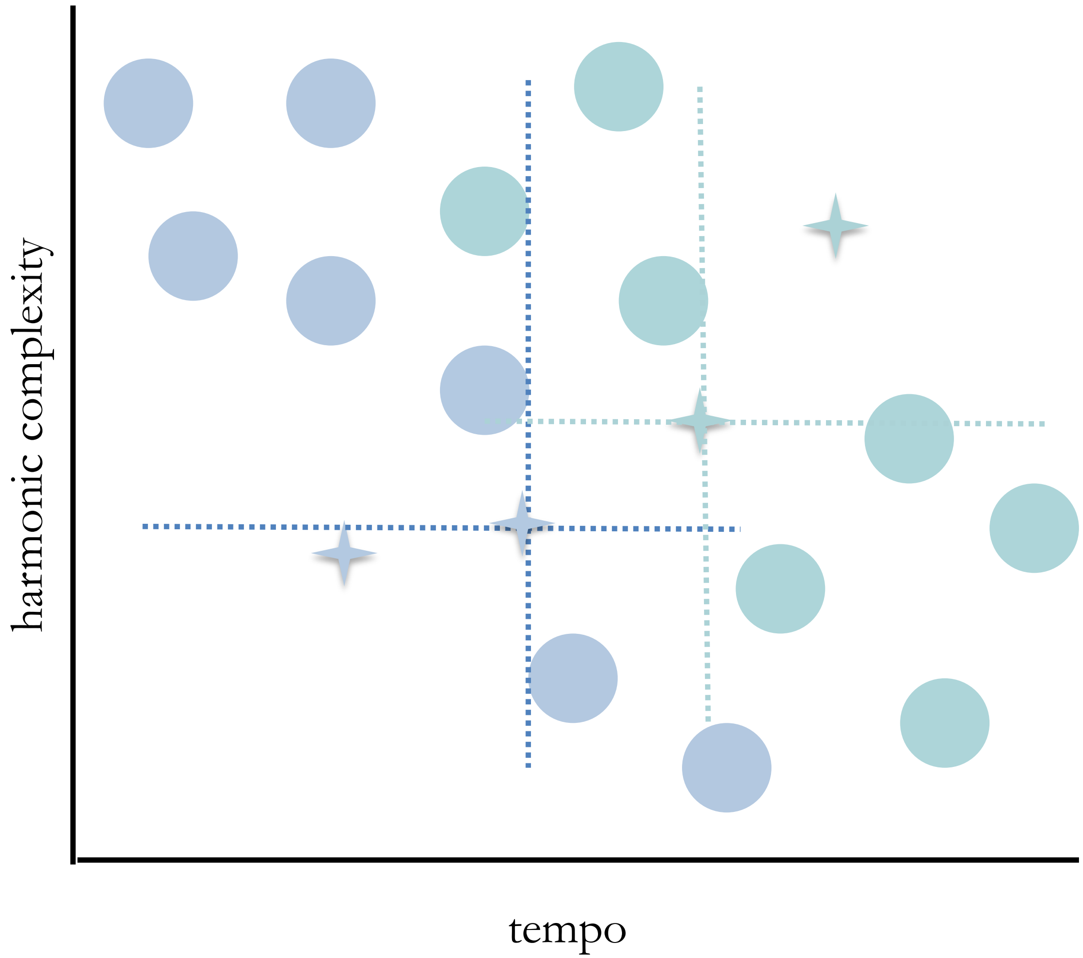
We start by randomly guessing the initial placement of the centers

K Means



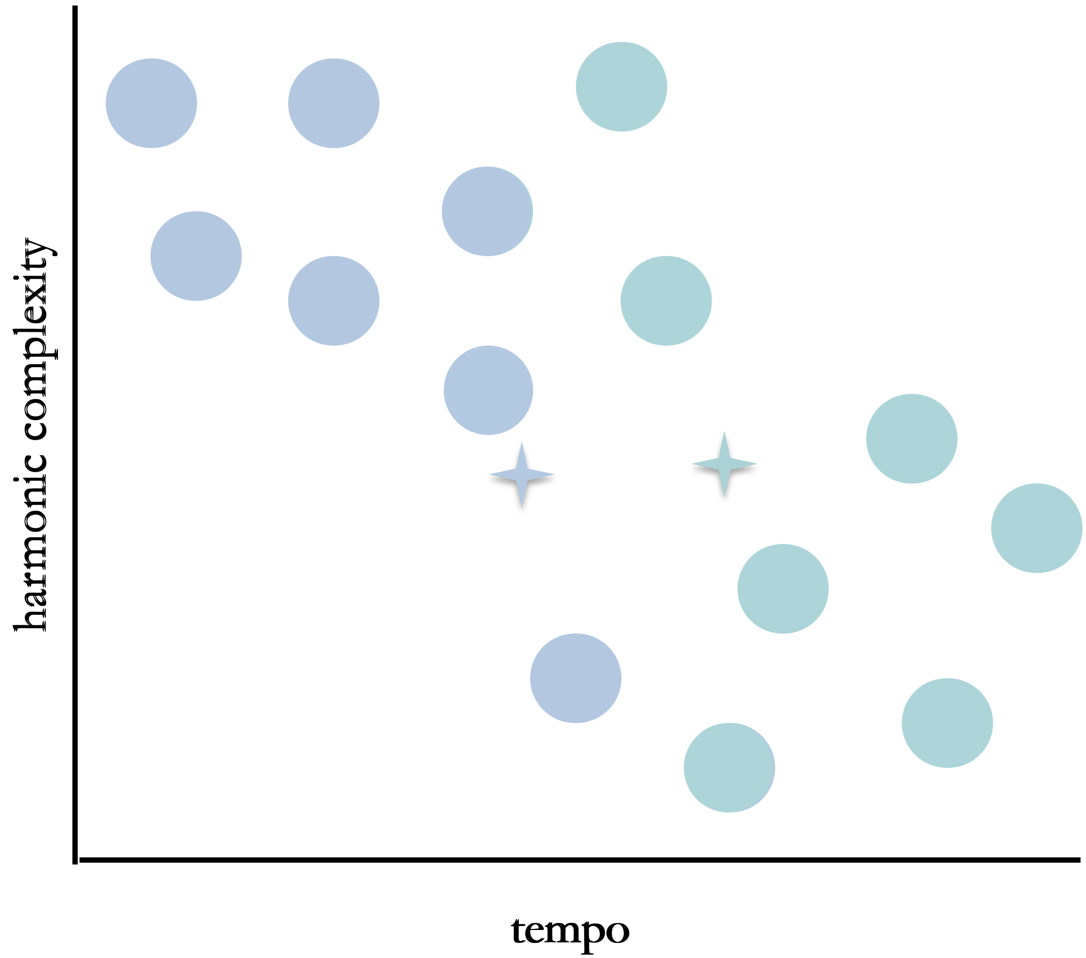
- Assign each point to the closest center
- We are assigning the point to the cluster of the center

K Means



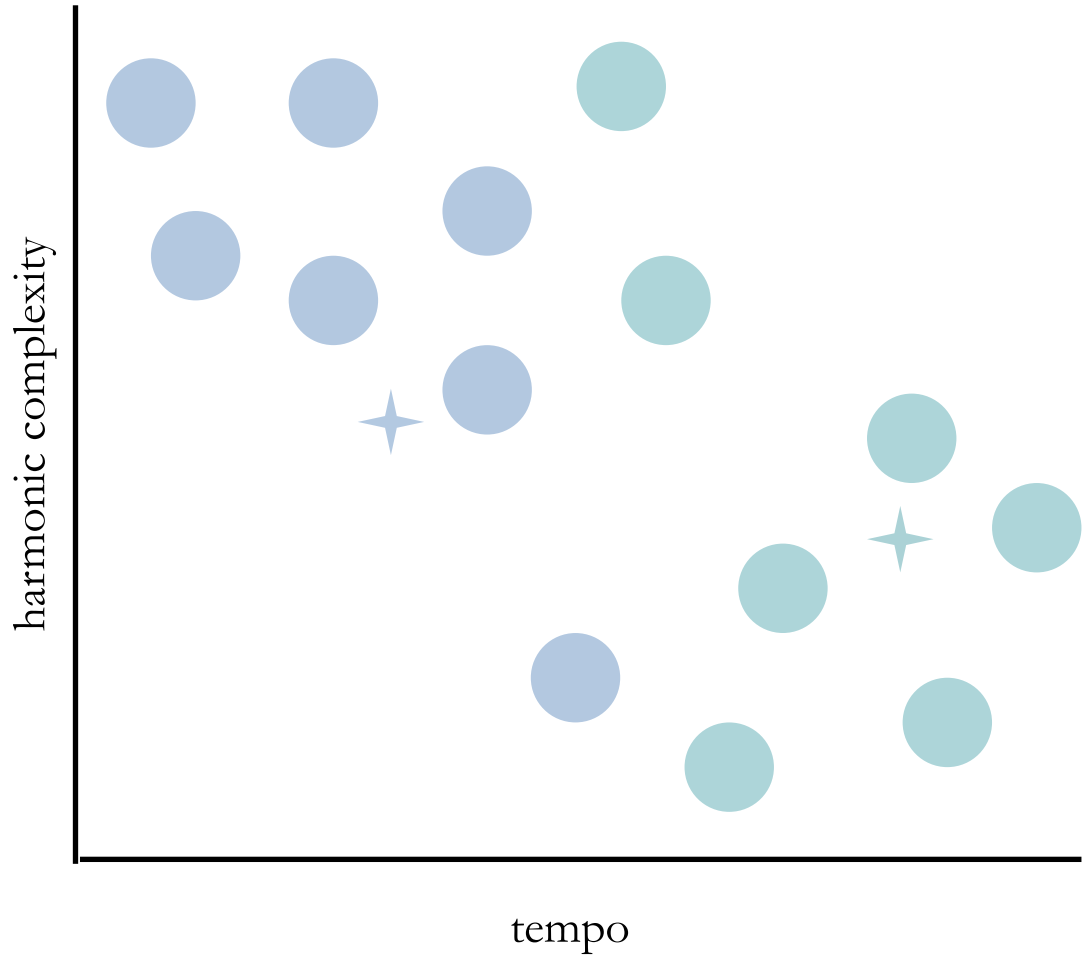
For each cluster compute a new center by averaging the coordinates of the points currently allocated to the center

K Means



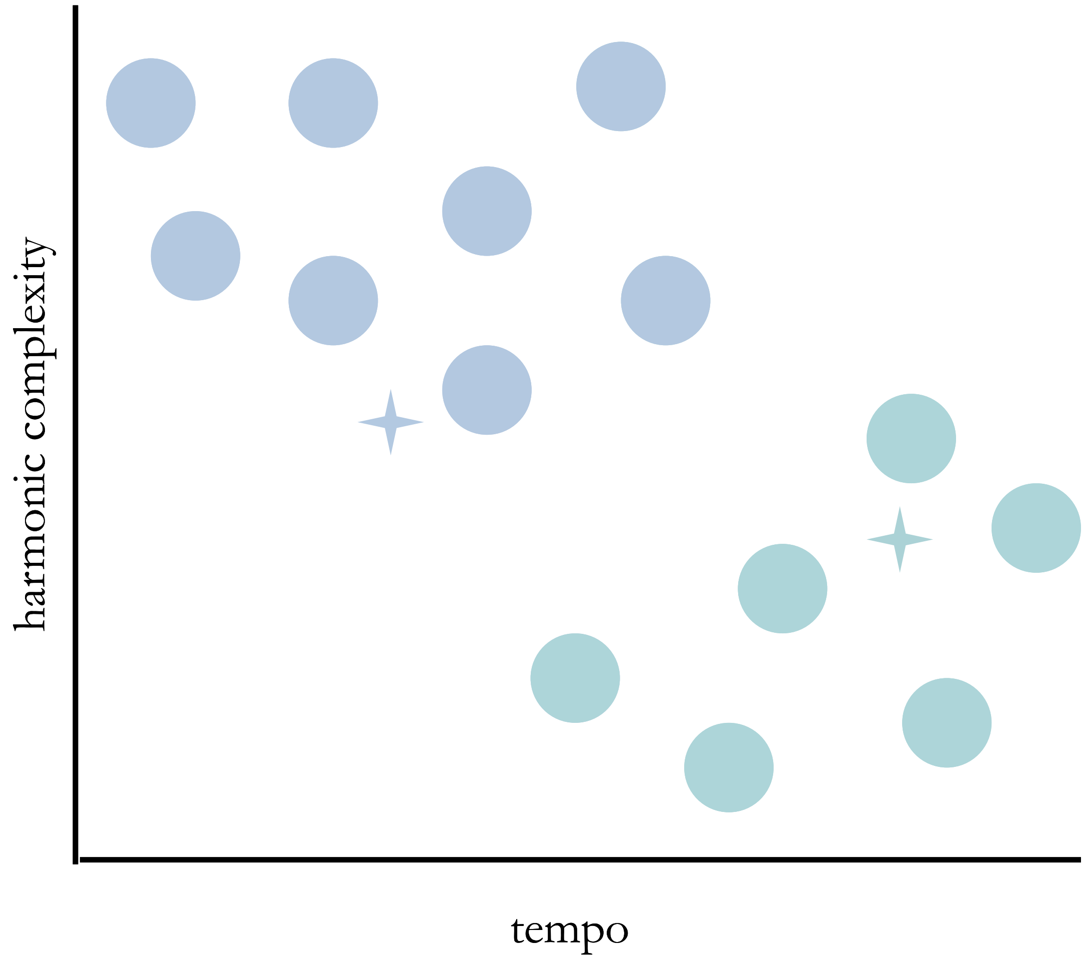
Assign each point to closest center among the new ones

K Means



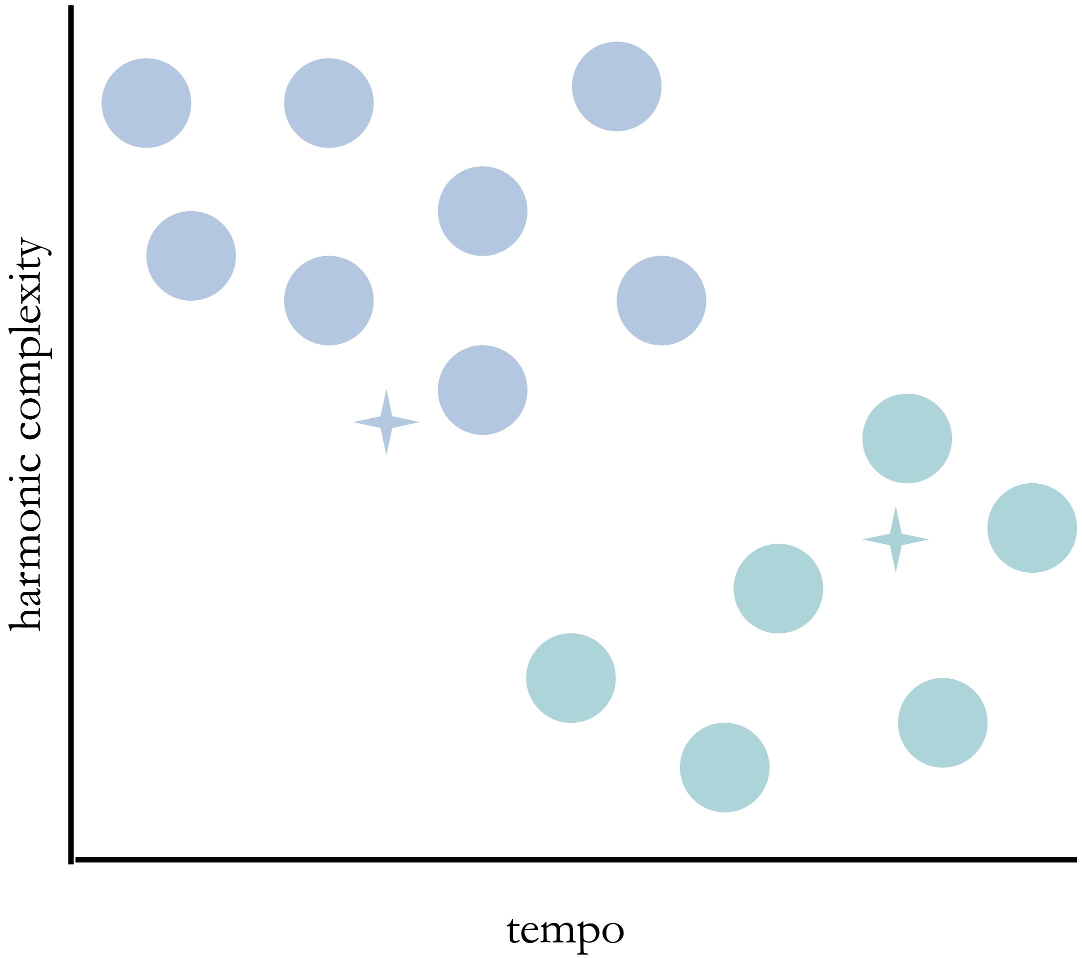
Compute new centers again by averaging the coordinates

K Means



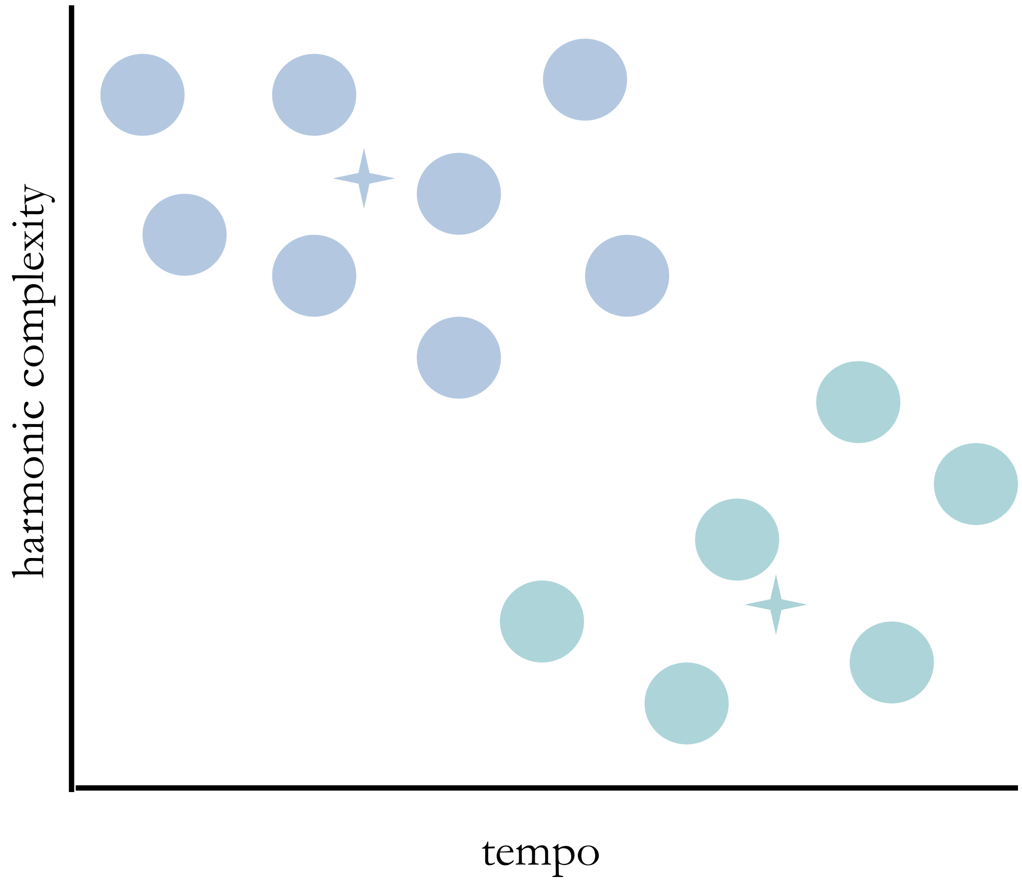
Re-assign each point to the cluster with the closest center

K Means



Re-assign each point to the cluster with the closest center

K Means



- Re-assign each point to the cluster with the closest center
- When the centers and the cluster assignments do not change anymore, we have convergence

K Means

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

"Hyperparameters" (i.e. not model parameters)

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

How many clusters we want to find

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

When to stop iterating: maximum number of iterations or minimum threshold of improvement

- Useful to prevent situations when convergence would take too long

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

Initialization: Randomly guess what the means are

- Purely random can lead to cold start
- If we have prior/domain knowledge we can use it for a better initial guess

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

Repeat/iterate until your hyperparameters say to stop

- Convergence will trigger mindiff criterion

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

Assign each point to its closest center

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

K Means

Recompute the center to be the mean of the points assigned to each cluster

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

Question time !

What is the "loss" that we are trying to minimize here?

(a) Number of clusters
(b) Distance of points to their respective clusters
(c) Distance between clusters
(d) Probability of observed data

Question time !

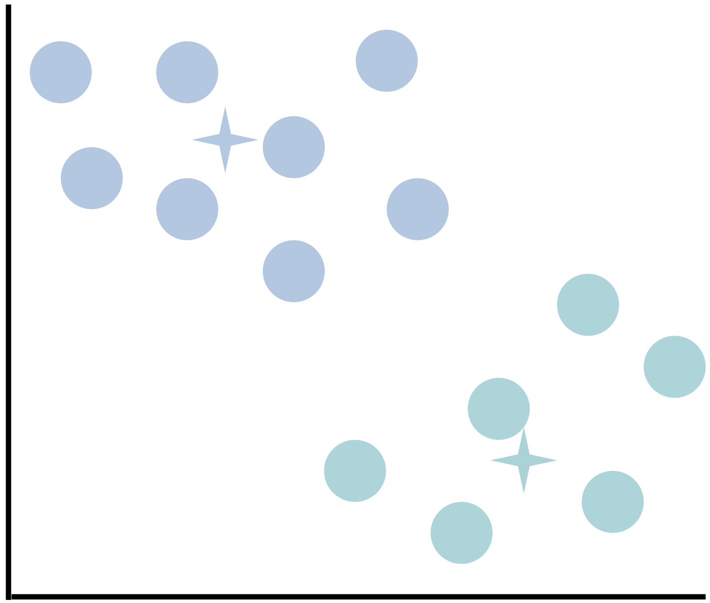
Is this a good objective?

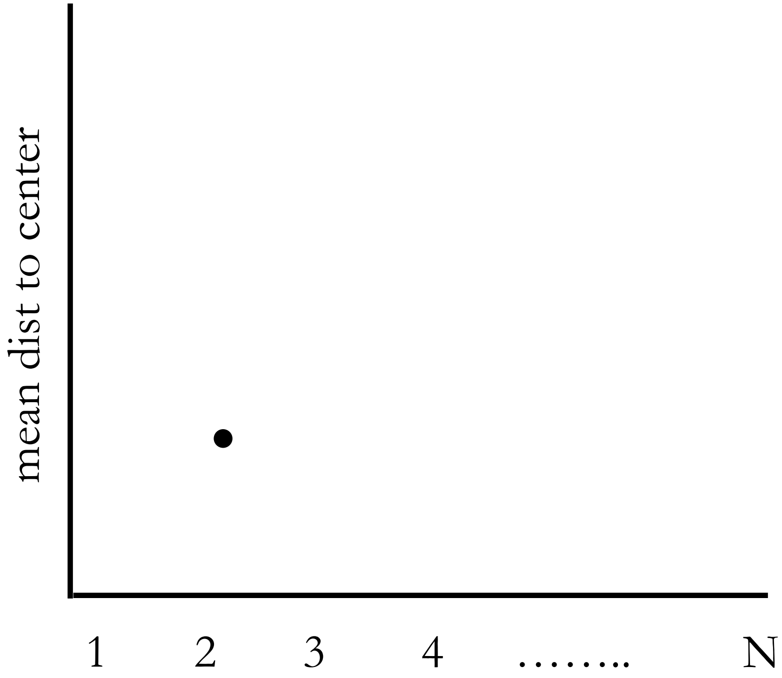
(a) Yes
(b) No
(c) Depends on the application

Computing the distances

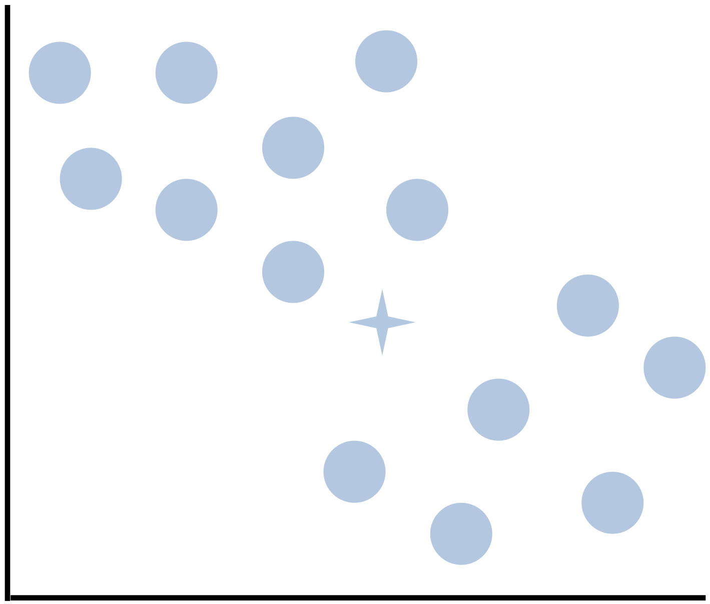
- Different scales/ranges in the features being represented may skew our evaluations
  - Imagine data points with features (age, income)
  - If we simply compute the Euclidian distance, we will skew heavily towards income
- Two possible solutions:
  - Normalize the ranges of the dimension to a fixed set range
  - Opportunely weight different features according to their range
  - Weights can also be used to emphasize/deemphasize the impact/importance of some of the features

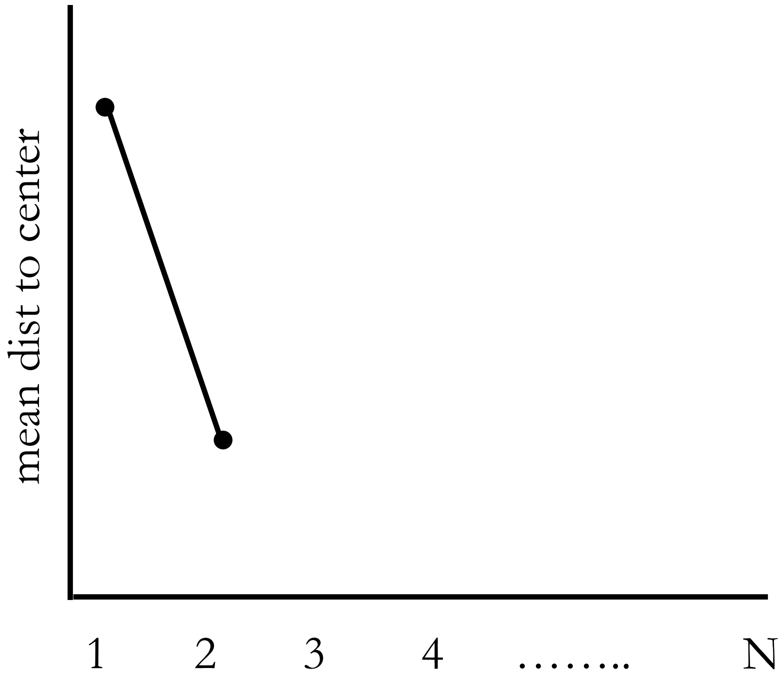
How many clusters?



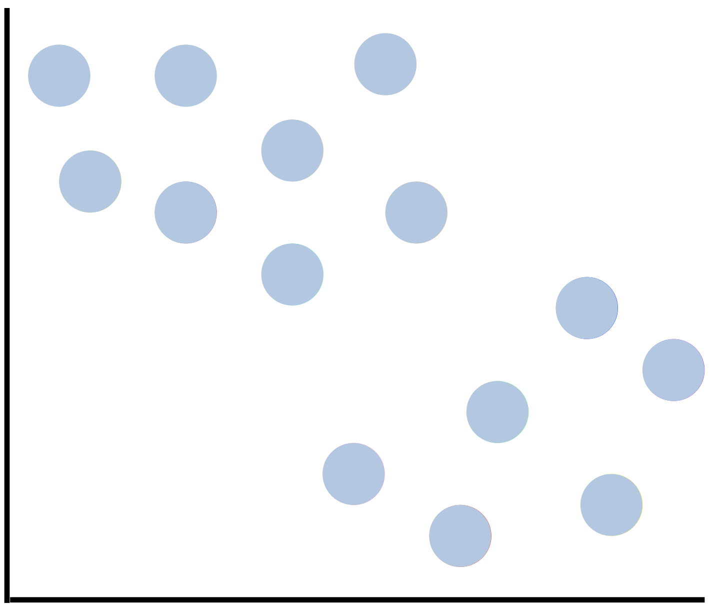


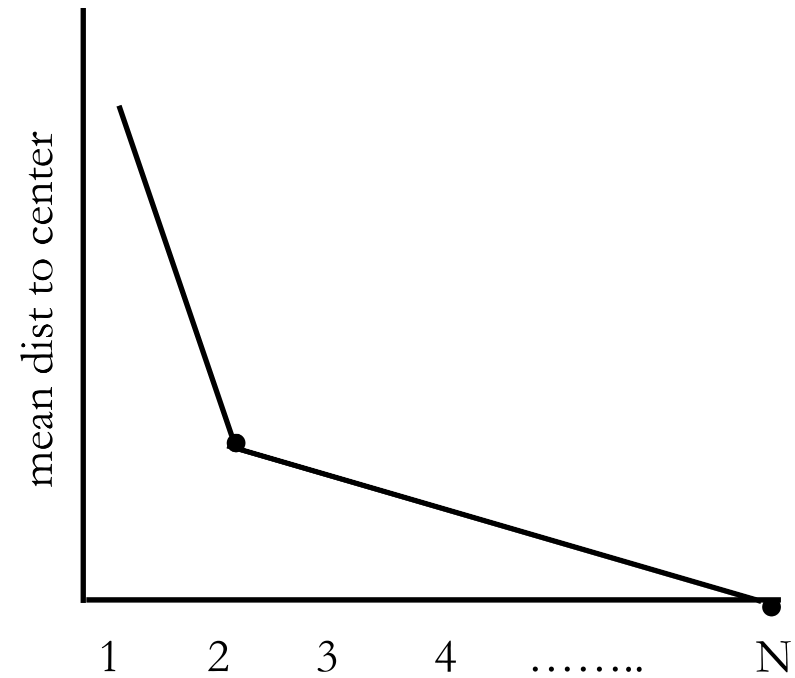
How many clusters?





How many clusters?

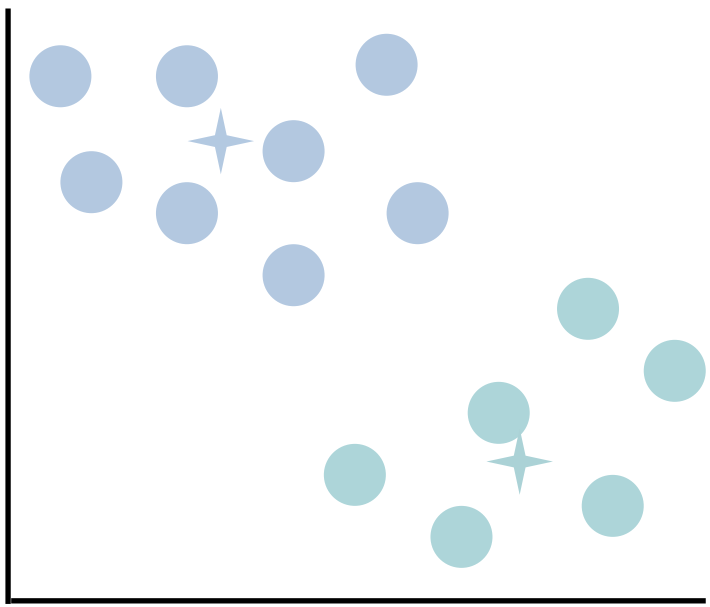


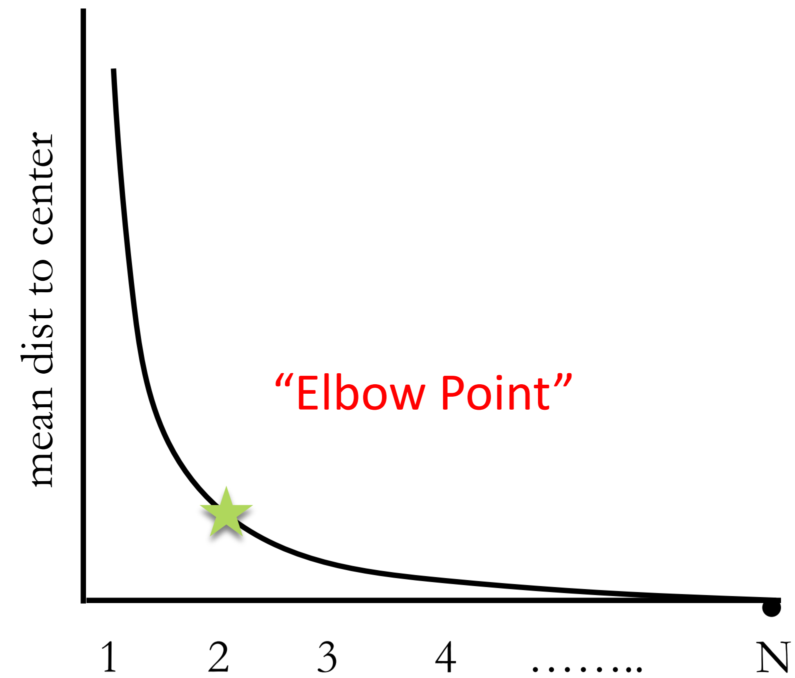


In the extreme: as many clusters as points

- Each points serves as the center of its own cluster
- Each points has zero distance from the center of its cluster
- Is this really a good idea?

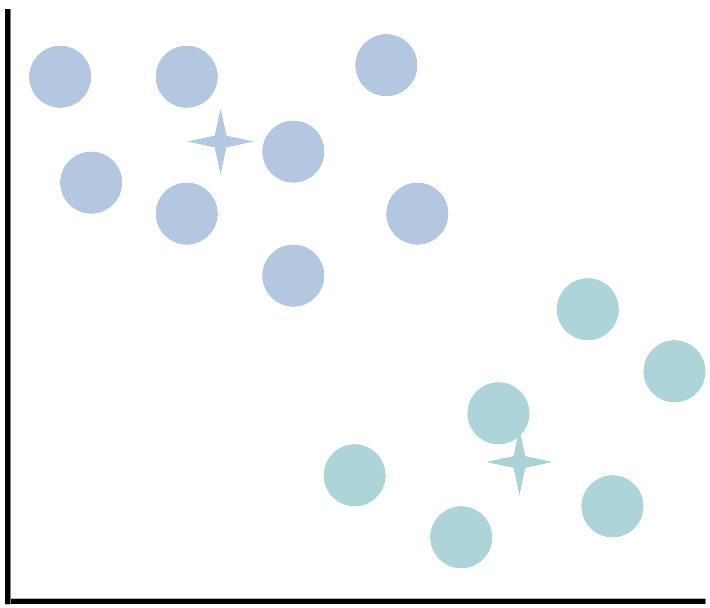
How many clusters?

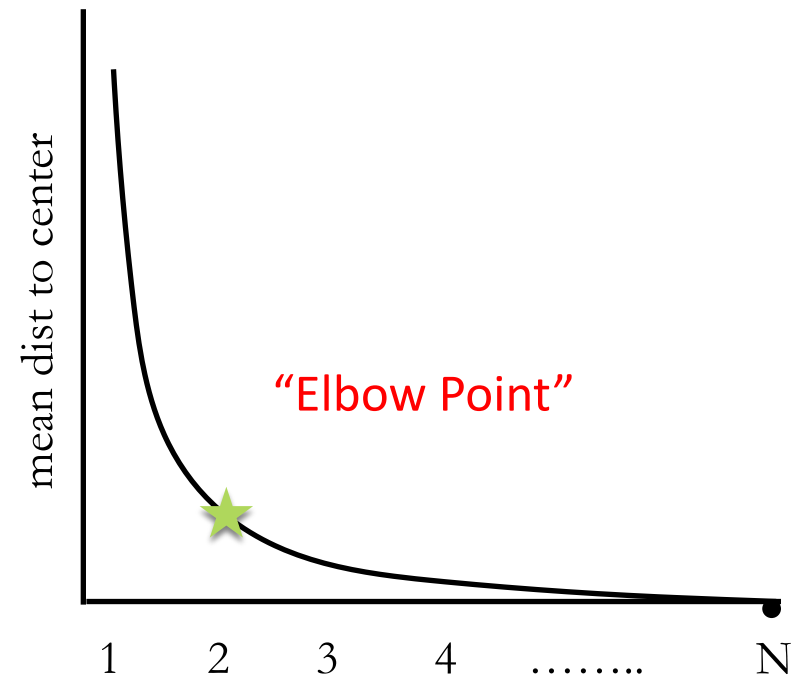




- Elbow point - # of clusters such that it is possible to assign points with a "small enough" mean distance from the centers
- Represents an accuracy/complexity trade-off
- More Clusters → higher accuracy and complexity
- Less Clusters → lower accuracy and complexity

How many clusters?





Techniques to fix the number of clusters:

- Intuition
- Domain Expertise
- Iteration

Silhouette technique

- A method to evaluate the quality of your clustering based on the consistency/similarity of the points in each cluster
  - Points $a, b, \ldots, i, j, \ldots$
  - For each point i, let C_i be its assigned cluster

- For each point compute

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

Average distance from points in its own cluster

- $a(i) = 0$ if $|C_i| = 1$

$$b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

Average distance from points in closest other cluster

- $b(i) = \infty$ if there is no other cluster

Silhouette technique

The silhouette score of each point is:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- $-1 \leq s(i) \leq 1$
- The closest $s(i)$ is to 1 the better it fits in its assigned cluster
- For $s(i)$ to be close to 1, we must have $b(i) >> a(i)$
- If $s(i)$ is close to -1, it would be more appropriate if i was clustered in its closest neighboring cluster.
- The mean of the $s(i)$'s over all points of a cluster is a measure of how tightly grouped all the points in the cluster are.

Silhouette technique

- The mean of the $s(i)$'s over all data of the entire dataset, called the silhouette coefficient is a measure of how appropriately the data have been clustered.
- If there are too many or too few clusters used in the clustering algorithm (e.g. k-means), some of the clusters will have silhouettes close to zero
- We considering multiple possible values k for the number of clusters we can pick

$$k^* = \max_{\{k_1, k_2, \dots\}} \frac{1}{n} \sum s(i)$$

Expectation Maximization (EM)

- A very general framework for non-parametric learning algorithms
- Two main phases repeated iteratively:
  - Expectation: given the current belief of the correct properties of the data distribution/parameters of the model, estimate the likelihood of the observed data
  - Maximization: update the parameters of believed distribution/model to maximize the probability

Expectation Maximization (EM)

Cold start: randomly pick a starting point

- Values of parameters of the model
- Initial naïve belief on the model/distribution

```
randomly initialize params
while not converged:
    data = estimate_likelihood(params)
    params = maximize_likelihood(data)
```

Expectation Maximization (EM)

Estimation Step: estimate the likelihood of data under current parameter setting

```
randomly initialize params
while not converged:
    data = estimate_likelihood(params)
    params = maximize_likelihood(data)
```

Expectation Maximization (EM)

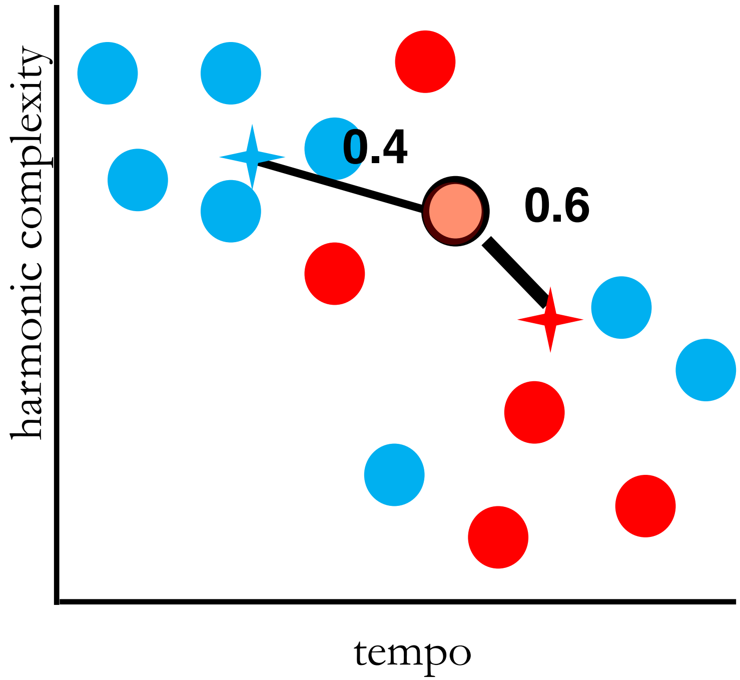
Maximization Step: adjust the the parameters to maximize the expectation of the data

```
randomly initialize params
while not converged:
    data = estimate_likelihood(params)
    params = maximize_likelihood(data)
```

EM and clustering

We can modify K-means clustering to include properties of the EM design (Soft K-Means)

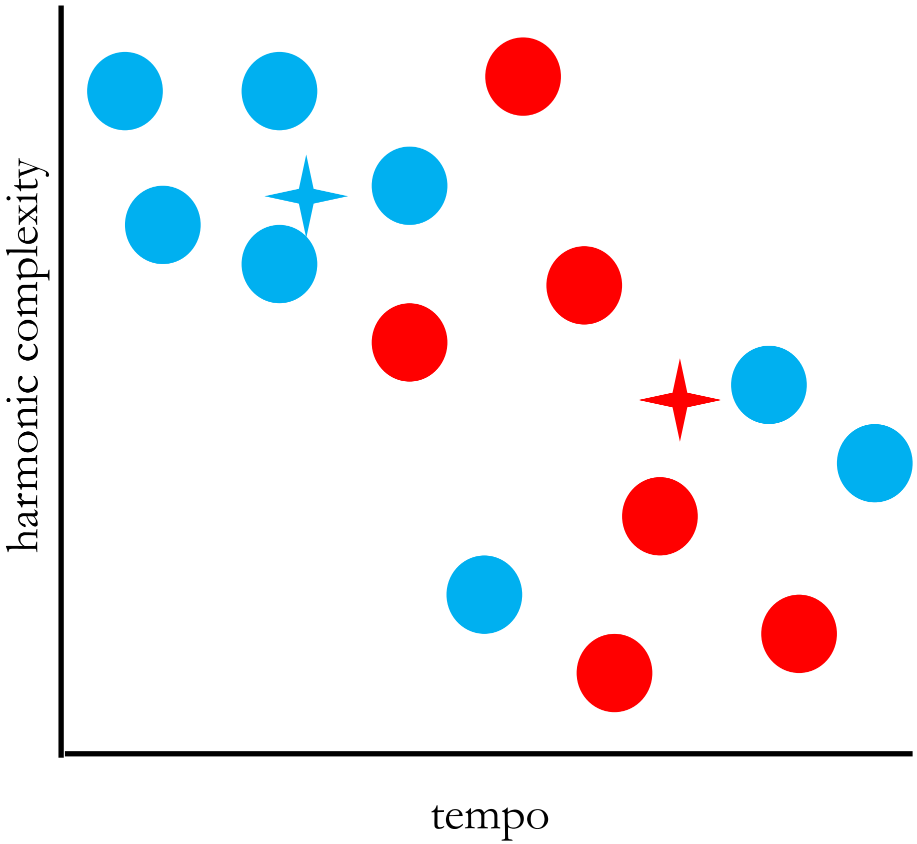
- Start with a random guess of the centers



Expectation (E) step:

- We associate to each point a discrete distribution of belonging in each cluster
  - For example, a point belong in a cluster with probability inversely proportional to its distance to the centers
- We assign points to clusters randomly according to such probabilities

EM and clustering



Maximization (M) step:

- Compute the new centers of the clusters as the means of the points currently assigned to each of them
- These would be the centers that would maximize the probability of the observed outcome

Iterate E and M step until convergence is reached or stopping conditions are met

EM K Means

The critical difference will be the way we update the assignments (the E-steps)

```
define parameters: K, max_iter, min_diff

iter = 0
change = inf
means = [random() for _ in range(K)]
while iter < max_iter and change > min_diff:
    update_assignments()
    compute_new_means()
    change = max_i(dist(new_mean_i, old_mean_i))
    iter += 1
```

EM for confidence in user reviews

Goal: Find "true" labels despite noisy annotations from workers…

	worker1	worker2	worker3	worker4	worker5
email1					
email2					
email3					
email4					
email5					

EM for confidence in user reviews

	worker1	worker2	worker3	worker4	worker5
[illegible]	[illegible]	[illegible]	[illegible]	not	spam
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	spam
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	spam	[illegible]	[illegible]	[illegible]	[illegible]
email5	spam	not	[illegible]	[illegible]	[illegible]

Easy! If you tell me how much to trust each worker, I can trivially compute labels

Sure, just tell me the labels and I can easily figure out which workers to trust.

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

	spam	not
email1	?	?
email2	?	?
email3	?	?
email4	?	?
email5	?	?

w1	spam	not
is spam	?	?
is not	?	?

w2	spam	not
is spam	?	?
is not	?	?

w3	spam	not
is spam	?	?
is not	?	?

w4	spam	not
is spam	?	?
is not	?	?

w5	spam	not
is spam	?	?
is not	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

P(email1 is spam)

	spam	not
email1	?	?
email2	?	?
email3	?	?
email4	?	?
email5	?	?

w1	spam	not
is spam	?	?
is not	?	?

w2	spam	not
is spam	?	?
is not	?	?

w3	spam	not
is spam	?	?
is not	?	?

w4	spam	not
is spam	?	?
is not	?	?

w5	spam	not
is spam	?	?
is not	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

P(w1 says spam | not spam)

	spam	not
email1	?	?
email2	?	?
email3	?	?
email4	?	?
email5	?	?

w1	spam	not
is spam	?	?
is not	?	?

w2	spam	not
is spam	?	?
is not	?	?

w3	spam	not
is spam	?	?
is not	?	?

w4	spam	not
is spam	?	?
is not	?	?

w5	spam	not
is spam	?	?
is not	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam		
email3	not	spam	not		
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

Initialization:
All workers make perfect decisions!

w1	spam	not
is spam	1	0
is not	0	1

w2	spam	not
is spam	1	0
is not	0	1

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	1	0
is not	0	1

	spam	not
email1	?	?
email2	?	?
email3	?	?
email4	?	?
email5	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0	1

w2	spam	not
is spam	1	0
is not	0	1

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	1	0
is not	0	1

Compute probability of labels by averaging over distribution of votes

	spam	not
email1	?	?
email2	?	?
email3	?	?
email4	?	?
email5	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0	1

w2	spam	not
is spam	1	0
is not	0	1

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	1	0
is not	0	1

Compute probability of labels by averaging over distribution of votes

	spam	not
email1	1+0+0+0+1 /5	0+1+1+1+0 /5
email2	?	?
email3	?	?
email4	?	?
email5	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0	1

w2	spam	not
is spam	1	0
is not	0	1

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	1	0
is not	0	1

Compute probability of labels by averaging over distribution of votes

	spam	not
email1	0.4	0.6
email2	1	0
email3	0.4	0.6
email4	0.8	0.2
email5	0.4	0.6

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0	1

w2	spam	not
is spam	1	0
is not	0	1

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	1	0
is not	0	1

Compute probability of labels by averaging over distribution of votes

	spam	not
email1	0.4	**0.6**
email2	?	?
email3	?	?
email4	?	?
email5	?	?

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

Assume the correct labels are those identified by the majority and update the decision matrices

	spam	not
email1	0.4	**0.6**
email2	**1**	0
email3	0.4	**0.6**
email4	**0.8**	0.2
email5	0.4	**0.6**

w1	spam	not
is spam		
is not		

w2	spam	not
is spam		
is not		

w3	spam	not
is spam		
is not		

w4	spam	not
is spam		
is not		

w5	spam	not
is spam		
is not		

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	?	?
is not		

w2	spam	not
spam		

Assume the correct labels are those identified by the majority and update the decision matrices

	spam	not
1	0.4	0.6
2	1	0
3	0.4	0.6
email4	0.8	0.2
email5		0.6

Clicker Question!

(a)0.4, 0.6

(b)0.6, 0.4

(c)0.8, 0.2

(d)1.0, 0.0

not		

w5	spam	not
spam		
not		

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

	spam	not
email1	0.4	0.6
email2	**1**	0
email3	0.4	0.6
email4	**0.8**	0.2
email5		0.6

w1	spam	**not**
is spam	1	0
is not		

w2	spam	not
spam	1	0
not	0	1

w3	spam	not
spam	1	0
not	0	1

w4	spam	not
spam	1	0
not	0	1

w5	spam	not
spam	1	0
not	0	1

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

	spam	not
email1	0.4	**0.6**
email2	1	0
email3	0.4	**0.6**
email4	0.8	0.2
email5		**0.6**

w1	**spa m**	not
is spam	1	0
is not	0.67	0.33

w2	spam	not
spam	1	0
not	0	1

w3	spam	not
spam	1	0
not	0	1

w4	spam	not
spam	1	0
not	0	1

w5	spam	not
spam	1	0
not	0	1

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

	spam	not
email1	0.4	0.6
email2	1	0
email3	0.4	0.6
email4	0.8	0.2
email5		0.6

w1	spam	not
is spam	1	0
is not	0.67	0.33

w2	spam	not
is spam	1	0
is not	0.33	0.67

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	0.5	0.5
is not	1	0

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

Re-assign labels using weighted majority vote

- Each worker is weighted according to their probability of being correct

	spam	not
email1	1.5	4.34
email2		
email3		
email4		
email5		

w1	spam	not
is spam	1	0
is not	**0.67**	0.33

w2	spam	not
is spam	1	0
is not	0.33	**0.67**

w3	spam	not
is spam	1	0
is not	0	**1**

w4	spam	not
is spam	1	0
is not	0	**1**

w5	spam	not
is spam	0.5	0.5
is not	**1**	0

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0.67	0.33

w2	spam	not
is spam	1	0
is not	0.33	0.67

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	0.5	0.5
is not	1	0

	spam	not
email1	0.26	0.74
email2		
email3		
email4		
email5		

Normalize!

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0.67	0.33

w2	spam	not
is spam	1	0
is not	0.33	0.67

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	0.5	0.5
is not	1	0

	spam	not
email1	0.26	0.74
email2	0.69	0.31
email3	0.29	0.71
email4	0.82	0.18
email5	0.26	0.74

Compute for all emails

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	
is not		

w2	spam	not
is spam	1	0
is not	0.33	0.67

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	0.5	0.5
is not	1	0

	spam	not
email1	0.26	**0.74**
email2	**0.69**	0.31
email3	0.29	**0.71**
email4	**0.82**	0.18
email5	0.26	**0.74**

Iterate the process until convergence

	w1	w2	w3	w4	w5
email1	spam	not	not	not	spam
email2	spam	spam	spam	spam	spam
email3	not	spam	not	not	spam
email4	spam	spam	spam	spam	not
email5	spam	not	not	not	spam

w1	spam	not
is spam	1	0
is not	0.67	0.33

w2	spam	not
is spam	1	0
is not	0.33	0.67

w3	spam	not
is spam	1	0
is not	0	1

w4	spam	not
is spam	1	0
is not	0	1

w5	spam	not
is spam	0.5	0.5
is not	1	0

In this example we have convergence in 1 round

	spam	not
email1	0.26	0.74
email2	0.69	0.31
email3	0.29	0.71
email4	0.82	0.18
email5	0.26	0.74