Week 11

Data Mining and Knowledge Discovery

Dr John Evans
j.evans8@herts.ac.uk

University of Hertfordshire UH

Plan for today

Bisecting K-means

Other problems

Cluster Evaluation

Recap

- We discussed the K-means clustering method
- We saw how, in Euclidean space, we want to minimise the total SSE.
- We saw how other proximity measures can be used for centroids, proximity and objective functions.
- We have seen problems with choosing initial centroids.
- We saw other issues with the K-means algorithm, such as:
  - Handling empty clusters.
  - Outliers.
  - Reducing the SSE with postprocessing.

Limitations of the K-means algorithm

- Only identifies spherical shaped clusters i.e cannot identify if the clusters are non-spherical, or of various size and density.
- Suffers from local minima.
- Has a problem when the data contains outliers.

Bisecting K-means

A useful extension of the basic K-means algorithm is the *bisecting K-means algorithm*.

- This is a hybrid of partitional and hierarchical clustering and can recognise clusters of any size or shape.
- It works well when k is large and, in general, performs better on entropy measurement than K-means.
- Only the data points from one cluster and two centroids are used in each bisecting stage of Bisecting K-means. Consequently, computation time is shortened.

Bisecting K-means

A useful extension of the basic K-means algorithm is the *bisecting K-means algorithm*.

- This is a hybrid of partitional and hierarchical clustering and can recognise clusters of any size or shape.
- It works well when k is large and, in general, performs better on entropy measurement than K-means.
- Only the data points from one cluster and two centroids are used in each bisecting stage of Bisecting K-means. Consequently, computation time is shortened.

Idea: Split the set of all points into two clusters, select one of these clusters to split, and so on, until K clusters have been produced.

Intuitive difference

- You attend a seminar. Upon entering the room, you take a seat without giving any thought to whether you can hear the speaker, see the board/slides etc. This is the K-means approach.

Intuitive difference

- You attend a seminar. Upon entering the room, you take a seat without giving any thought to whether you can hear the speaker, see the board/slides etc. This is the K-means approach.
- If instead you scan the room when you enter, and then choose a seat based on factors such as hearing the speaker, seeing the board/slides, then this is the bisecting approach.

The bisecting K-means algorithm

Input:	k: the number of clusters, D: a data set containing n objects
Output:	A set of k clusters
Method:	
Step 1	Initialize the list of clusters to accommodate the cluster consisting of all points;
Step 2	**repeat**
Step 2a	Remove a cluster from the list of clusters,
Step 2b	{Perform several 'trial' bisections of the chosen cluster};
Step 2c	**for** $i = 1$ to *number of trials* **do**
Step 2d	Bisect the selected cluster using basic K-means;
Step 2e	**end for**;
Step 2f	Select the two clusters from the bisection with the lowest total SSE;
Step 2g	Add these two clusters to the list of clusters;
Step 3	**until** The list of clusters contains k clusters.

How do we choose which cluster to split?

How do we choose which cluster to split?

- Choose the largest cluster at each step.
- Choose the cluster with the highest SSE.
- Choose based on some other criterion.

Basic example

- Suppose we use SSE and $k = 3$.

Basic example

- Suppose we use SSE and $k = 3$.
- First, all data points are put into a single cluster, C.
- Next, C is split into two clusters (using K means, $k = 2$): C_1 and C_2.
- We haven't got three clusters yet, so we need to split again.

Basic example

- Suppose we use SSE and $k = 3$.
- First, all data points are put into a single cluster, C.
- Next, C is split into two clusters (using K means, $k = 2$): C_1 and C_2.
- We haven't got three clusters yet, so we need to split again.
- We look at which has the higher SSE. Suppose C_1 has the higher SSE.
- In this case, we split C_1 into C_1' and C_1''.
- We now have three clusters. Of course, if we chose a different choice than SSE, we would end up with different clusters.

Feeding this back into K-means

- Since we are using the K-means algorithm 'locally' (i.e. to bisect individual clusters), the final set of clusters does not represent a clustering that is a local minimum with respect to the total SSE.
- Thus, we often refine the resulting clusters by using their cluster centroids as the initial centroids for the standard K-means algorithm.

Example

- Bisecting K-means is less susceptible to initialisation problems.
- To show this, we look at the Figure on the next slide.
- In this case, $k = 4$ and in Iteration 1, two pairs of clusters are found.
- In Iteration 2, we next split the right-hand pair of clusters, before the left-hand clusters are split in Iteration 3.
- This results in 4 clusters, as required.
- As bisecting K-means performs several trial bisections, and takes the one with the lowest SSE, this algorithm has less trouble with initialisation.

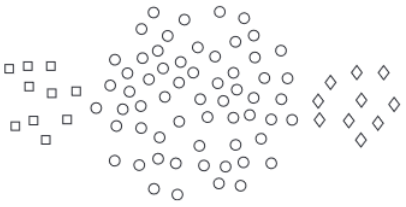
Example

(a) Iteration 1. (b) Iteration 2. (c) Iteration 3.

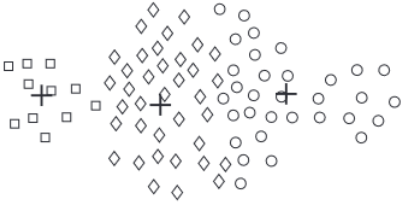
More problems to consider (Non-spherical clusters)

- K-means has trouble detecting ‘natural’ clusters when these are
  - non-spherical
  - when clusters have widely different sizes
  - when clusters have widely different densities.
- This is visualised on the next few slides.

Clusters of very different sizes.



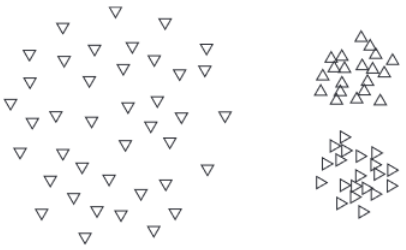
(a) Original points.



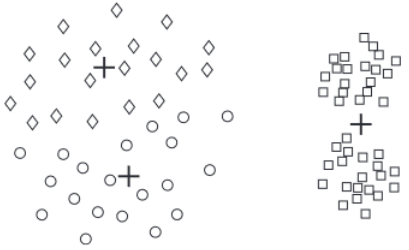
(b) Three K-means clusters.

K-means cannot find the three natural clusters because one of the clusters (the central one) is much larger than the other two. Thus, the larger cluster is broken, while one of the smaller clusters is combined with a portion of the larger cluster.

Clusters of different densities



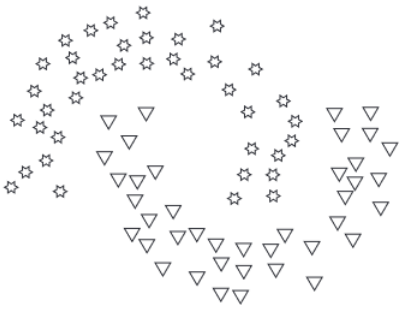
(a) Original points.



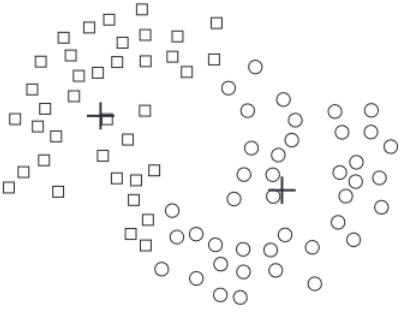
(b) Three K-means clusters.

This time, K-means fails because the two smaller clusters are much denser than the larger cluster.

Clusters that are not spherical



(a) Original points.



(b) Two K-means clusters.

Finally, K-means finds two clusters that mix portions of the two natural clusters because it is trying to find spherical clusters.

The reason for these difficulties

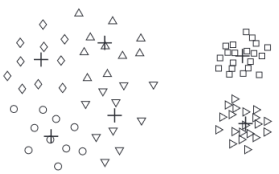
- The K-means objective function is a mismatch for the kinds of clusters we are trying to find.
- This is because it is minimised by globular clusters of equal size and density, or by clusters that are well-separated.

The reason for these difficulties

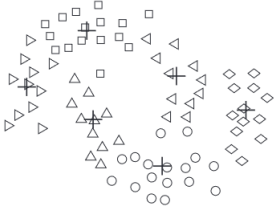
- The K-means objective function is a mismatch for the kinds of clusters we are trying to find.
- This is because it is minimised by globular clusters of equal size and density, or by clusters that are well-separated.
- To overcome these limitations, we must be willing to accept a clustering that breaks the natural clusters into a number of subclusters.
- This is shown on the next slide.
- This demonstrates what happens to our three data sets if instead we find six clusters instead of two or three. Each smaller cluster is pure in the sense that it contains only points from one of the natural clusters.

Example (Data in Euclidean space continued)

(a) Unequal sizes.



(b) Unequal densities.



(c) Non-spherical shapes.

Strengths and weaknesses of the K-means algorithm

- K-means is simple and can be used for a wide variety of data types.
- K-means is quite efficient, even though multiple runs are often performed.
- Some variants (such as bisecting K-means) are even more efficient, and are less susceptible to initialisation problems.

Strengths and weaknesses of the K-means algorithm

- K-means is simple and can be used for a wide variety of data types.
- K-means is quite efficient, even though multiple runs are often performed.
- Some variants (such as bisecting K-means) are even more efficient, and are less susceptible to initialisation problems.
- K-means is not suitable for all types of data. It cannot handle non-spherical clusters, clusters of different sizes or clusters of different densities. These limitations can often be overcome by finding pure subclusters if the number of clusters specified is large enough.

Strengths and weaknesses of the K-means algorithm

- K-means is simple and can be used for a wide variety of data types.
- K-means is quite efficient, even though multiple runs are often performed.
- Some variants (such as bisecting K-means) are even more efficient, and are less susceptible to initialisation problems.
- K-means is not suitable for all types of data. It cannot handle non-spherical clusters, clusters of different sizes or clusters of different densities. These limitations can often be overcome by finding pure subclusters if the number of clusters specified is large enough.
- K-means also has trouble clustering data that contains outliers. Outlier detection and removal can help significantly in such situations.

Strengths and weaknesses of the K-means algorithm

- K-means is simple and can be used for a wide variety of data types.
- K-means is quite efficient, even though multiple runs are often performed.
- Some variants (such as bisecting K-means) are even more efficient, and are less susceptible to initialisation problems.
- K-means is not suitable for all types of data. It cannot handle non-spherical clusters, clusters of different sizes or clusters of different densities. These limitations can often be overcome by finding pure subclusters if the number of clusters specified is large enough.
- K-means also has trouble clustering data that contains outliers. Outlier detection and removal can help significantly in such situations.
- K-means is restricted to data for which there is a notion of a centre (a centroid). A related technique known as K-medoid clustering does not have this restriction, but it is more expensive.

Evaluating supervised learning

- In classification (supervised), the evaluation of the resulting classification model is an integral part of the process. We saw well-accepted evaluation measures and procedures, such as accuracy and cross-validation, respectively.

Evaluating supervised learning

- In classification (supervised), the evaluation of the resulting classification model is an integral part of the process. We saw well-accepted evaluation measures and procedures, such as accuracy and cross-validation, respectively.
- However, by its very nature, cluster evaluation is not a well-developed or commonly used part of cluster analysis.
- After all, many times cluster analysis is conducted as part of exploratory data analysis.
- As such, evaluations seems to be an unnecessary complication.

Evaluating supervised learning

- In classification (supervised), the evaluation of the resulting classification model is an integral part of the process. We saw well-accepted evaluation measures and procedures, such as accuracy and cross-validation, respectively.
- However, by its very nature, cluster evaluation is not a well-developed or commonly used part of cluster analysis.
- After all, many times cluster analysis is conducted as part of exploratory data analysis.
- As such, evaluations seems to be an unnecessary complication.
- Added to this, there are problems arising from the different types of clusters.In some sense, each clustering algorithm defines its own type of cluster, and this may require its own evaluation measure.
- For example, K-means clusters might be evaluated in terms of the SSE, but for density-based clusters (which need not be spherical), SSE would not work well.

Cluster evaluation

- Nonetheless, cluster evaluation (traditionally known as *cluster validation*) can be important, and ought to be a key part of the cluster analysis.
- A key motivation here is that almost every clustering algorithm will find clusters in a data set, regardless of whether the data set actually has a natural cluster structure.

Other considerations to consider

- Unsupervised techniques, i.e. do not make use of any external information.
  1. Determining the *clustering tendency* of a set of data.
  2. Determining the correct number of clusters.
  3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.

Other considerations to consider

- Unsupervised techniques, i.e. do not make use of any external information.
  1. Determining the *clustering tendency* of a set of data.
  2. Determining the correct number of clusters.
  3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
- Supervised techniques, i.e. require external information.
  4. Comparing the results of a cluster analysis to externally known results, such as externally provided class labels.

Other considerations to consider

- Unsupervised techniques, i.e. do not make use of any external information.
  1. Determining the *clustering tendency* of a set of data.
  2. Determining the correct number of clusters.
  3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
- Supervised techniques, i.e. require external information.
  4. Comparing the results of a cluster analysis to externally known results, such as externally provided class labels.
- Can be either supervised or unsupervised.
  5. Comparing two sets of clusters to determine which is better.

We will consider: 1, 2 and 4.

Clustering tendency

- A naive approach to determining whether a given data set has clusters is to simply try and cluster it.
- However, most clustering algorithms will always find clusters, regardless of whether the data is highly clustered, uniformly distributed or entirely random.

Clustering tendency

- A naive approach to determining whether a given data set has clusters is to simply try and cluster it.
- However, most clustering algorithms will always find clusters, regardless of whether the data is highly clustered, uniformly distributed or entirely random.
- To overcome this, we have several options:
  - First, we could attempt to evaluate the resulting clusters in some way and then only claim a data set has clusters if at least some of the clusters are of good quality.
  - However, a problem immediately presents itself as clusters in the data can be of a different type than those sought by our clustering algorithms.

Clustering tendency

- A naive approach to determining whether a given data set has clusters is to simply try and cluster it.
- However, most clustering algorithms will always find clusters, regardless of whether the data is highly clustered, uniformly distributed or entirely random.
- To overcome this, we have several options:
  - First, we could attempt to evaluate the resulting clusters in some way and then only claim a data set has clusters if at least some of the clusters are of good quality.
  - However, a problem immediately presents itself as clusters in the data can be of a different type than those sought by our clustering algorithms.
  - To address this, we could use multiple algorithms and again evaluate the quality of the resulting clusters.
  - If the clusters are uniformly poor, then this may indeed indicate that there are no clusters in the data.

Clustering tendency

- Alternatively, we could attempt to evaluate clustering tendency *without* clustering.

Clustering tendency

- Alternatively, we could attempt to evaluate clustering tendency *without* clustering.
- The most common approach, especially for data in Euclidean space, has been to use statistical tests for spatial randomness.
- As might be expected, choosing the correct model, estimating the parameters and evaluating the statistical significance of the hypothesis that the data is non-random is a challenge.
- Nevertheless, several approaches have been developed, especially for data in low-dimensional Euclidean space.

Hopkins statistic

- The idea is to generate p points that are randomly distributed across the entire space (denote the collection of these by Y) and also sample p actual data points from our data set X (without replacement), which we label Z.
- Usually, $p << n$, where n is the number of data points in our data set X.

Hopkins statistic

- The idea is to generate p points that are randomly distributed across the entire space (denote the collection of these by Y) and also sample p actual data points from our data set X (without replacement), which we label Z.
- Usually, $p << n$, where n is the number of data points in our data set X.
- Then, for both sets of points, we find the distance to the nearest neighbour in our original data set. We label these as follows:
  - u_i, the distance of $y_i \in Y$ from its nearest neighbour in X;
  - w_i, the distance of $z_i \in Z$ from its nearest neighbour in X.

  Given this, the Hopkins Statistic H is given by,

$$H = \frac{\sum_{i=1}^{p} u_i}{\sum_{i=1}^{p} (u_i + w_i)}.$$

Hopkins statistic

- If the randomly generated points and the sample of data points have roughly the same nearest neighbour distances, then H will be near 0.5.
- If there are clusters, then H approaches 1 with a value of (or close to) 1 meaning the data is highly clustered.
- If the data is uniformly distributed, then H will tend to 0.

Of course, due to sampling, the results may vary with different executions. This means we usually repeat multiple times to get an accurate measure.

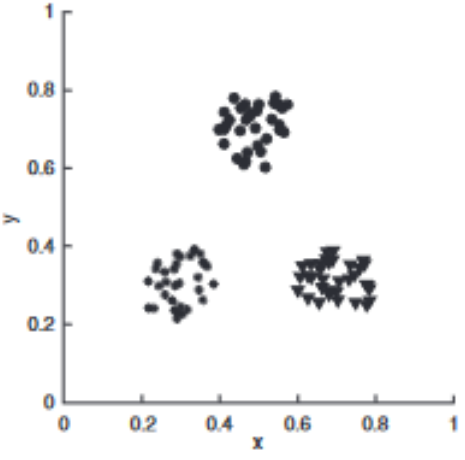
Example

- Consider the data set in the next slide. It contains 100 uniformly distributed points.
- For $p = 20$ and with 100 different trials, the average value of H was found to be 0.56 with a standard deviation of 0.03.
- For completeness, in the figure in two slides, this data set has been clustered using K-means, where $k = 3$.

Example (100 uniformly distributed points)

Example (Clustered with $k = 3$)

Example 2



Example 2

- The figure in the previous slide is obviously well separated into three clusters.
- In this case (for $p = 20$ as before, and where 100 trials were once again performed), the average value of H was 0.95 with a standard deviation of 0.006.

Determining the correct number of clusters

- There are various unsupervised cluster evaluation measures to approximately determine the ‘correct’ number of clusters.
- One such approach is to plot SSE against the number of clusters for a (bisecting) K-means clustering of the data set.

[1] This is often called the ‘elbow’.

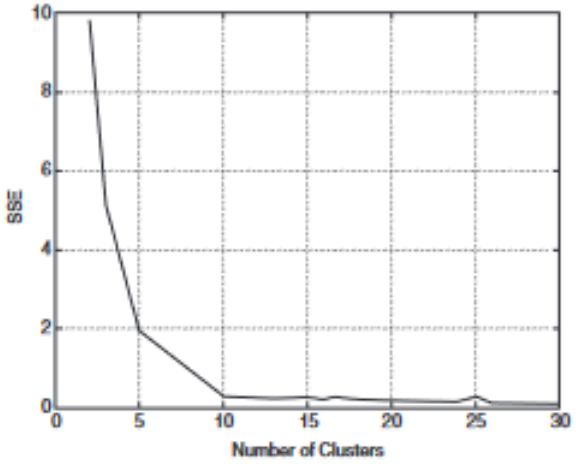
Determining the correct number of clusters

- There are various unsupervised cluster evaluation measures to approximately determine the ‘correct’ number of clusters.
- One such approach is to plot SSE against the number of clusters for a (bisecting) K-means clustering of the data set.
- For example, in the next slid, there is a figure which naturally has 10 clusters.
- We then show the aforementioned plot and notice that there is a distinct levelling-off[1] in the SSE when the number of clusters is equal to 10.

[1] This is often called the ‘elbow’.

Example

The ‘elbow’



Supervised measures of cluster validity

- When we have external information about the data, it is typically in the form of externally derived class labels for the data objects.
- In such cases, the usual procedure is to measure the degree of correspondence between the cluster labels and the class labels.

Supervised measures of cluster validity

- When we have external information about the data, it is typically in the form of externally derived class labels for the data objects.
- In such cases, the usual procedure is to measure the degree of correspondence between the cluster labels and the class labels.
- Why do this? After all, why perform a cluster analysis when we already have the class labels?
- Some motivations for this include comparing clustering techniques with the ‘truth’, or evaluating the extent to which a manual classification process can be automatically produced by cluster analysis.
- Another motivation might be to evaluate whether objects in the same cluster tend to have the same label for semi-supervised learning techniques.

Two sets of approaches

- The first set use measures from classification, such as entropy, to evaluate the extent to which a cluster contains objects of a single class.

Two sets of approaches

- The first set use measures from classification, such as entropy, to evaluate the extent to which a cluster contains objects of a single class.
- The second group of methods is related to the similarity measures for binary data, such as the Jaccard measure.
- These approaches measure the extent to which two objects that are in the same class are in the same cluster, and vice versa.

Two sets of approaches

- The first set use measures from classification, such as entropy, to evaluate the extent to which a cluster contains objects of a single class.
- The second group of methods is related to the similarity measures for binary data, such as the Jaccard measure.
- These approaches measure the extent to which two objects that are in the same class are in the same cluster, and vice versa.
- For convenience, we refer to these two types of measures as *classification-oriented* and *similarity-oriented*.

Classification-oriented

- We start with entropy and recall the definition, this time in the context of clustering.
- The degree to which each cluster consists of objects of a single class is represented by its entropy.

Classification-oriented

- We start with entropy and recall the definition, this time in the context of clustering.
- The degree to which each cluster consists of objects of a single class is represented by its entropy.
- To compute this, we first calculate the class distribution of the data. In other words, for a cluster i we compute p_{ij}, the probability that a member of a cluster i belongs to class j as $p_{ij} = \frac{m_{ij}}{m_i}$, where
  - $m_i =$ the number of objects in cluster i;
  - $m_{ij} =$ the number of objects of class j in cluster i.

Classification-oriented

- We start with entropy and recall the definition, this time in the context of clustering.
- The degree to which each cluster consists of objects of a single class is represented by its entropy.
- To compute this, we first calculate the class distribution of the data. In other words, for a cluster i we compute p_{ij}, the probability that a member of a cluster i belongs to class j as $p_{ij} = \frac{m_{ij}}{m_i}$, where
  - $m_i =$ the number of objects in cluster i;
  - $m_{ij} =$ the number of objects of class j in cluster i.
- Once computed, the entropy formula is then the same as usual:

$$e_i = -\sum_{j=1}^{L} p_{ij} \log_2(p_{ij}),$$

where $L =$ the number of classes.

Total entropy

- The total entropy for a set of clusters is calculated as the sum of entropies of each cluster weighted by the size of each cluster, i.e.

$$e = \sum_{i=1}^{k} \frac{m_i}{m} e_i,$$

where k is the number of clusters and m is the total number of data points.

Other performance measures

- Purity
  This measures the extent to which a cluster contains objects of a single class.

Other performance measures

- Purity
  This measures the extent to which a cluster contains objects of a single class.
  With notation as above, the purity of a cluster i is given by,

$$purity(i) = max_j(p_{ij}).$$

Other performance measures

- Purity
  This measures the extent to which a cluster contains objects of a single class.
  With notation as above, the purity of a cluster i is given by,

$$purity(i) = max_j(p_{ij}).$$

  The overall purity of a clustering is then given by,

$$\sum_{i=1}^{k} \frac{m_i}{m} purity(i).$$

Other performance measures

- Purity
  This measures the extent to which a cluster contains objects of a single class. With notation as above, the purity of a cluster i is given by,

$$purity(i) = max_j(p_{ij}).$$

  The overall purity of a clustering is then given by,

$$\sum_{i=1}^{k} \frac{m_i}{m} purity(i).$$

- Precision
  This is the fraction of a cluster that consists of objects of a specified class. The precision of cluster i with respect to class j is $precision(i,\, j) = p_{ij}$.

Example

- We use the k-means algorithm with the cosine similarity measure to cluster 3204 newspaper articles from the *Los Angeles Times*.
- These articles come from six different classes: Entertainment, Financial, Foreign, Metro, National, and Sports.
- We summarise the results of the k-means clustering where $k = 6$ in the table on the following slide.

The table

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports	Entropy	Purity
1	3	5	40	506	96	27	1.2270	0.7474
2	4	7	280	29	39	2	1.1472	0.7756
3	1	1	1	7	4	671	0.1813	0.9796
4	10	162	3	119	73	2	1.7487	0.4390
5	331	22	5	70	13	23	1.3976	0.7134
6	5	358	12	212	48	13	1.5523	0.5525
Total	354	555	341	943	273	738	1.1450	0.7203

Demonstrating a computation

- We consider the first cluster only.
- For this, $m_i = 677$. By labelling columns 2-7 by 1-6, we see that $m_{11} = 3$, $m_{12} = 5$, $m_{13} = 40$, $m_{14} = 506$, $m_{15} = 96$, $m_{16} = 27$.
- Thus, $p_{11} = 3/677$, $p_{12} = 5/677$, $p_{13} = 40/677$, $p_{14} = 506/677$, $p_{15} = 96/677$, $p1_{16} = 27/677$.

Demonstrating a computation

- We consider the first cluster only.
- For this, $m_i = 677$. By labelling columns 2-7 by 1-6, we see that $m_{11} = 3$, $m_{12} = 5$, $m_{13} = 40$, $m_{14} = 506$, $m_{15} = 96$, $m_{16} = 27$.
- Thus, $p_{11} = 3/677$, $p_{12} = 5/677$, $p_{13} = 40/677$, $p_{14} = 506/677$, $p_{15} = 96/677$, $p1_{16} = 27/677$.
- Putting this all together, we get the following calculations:

$$\begin{aligned} e_1 = & \quad -\sum_{j=1}^{6} p_{1j} \log_2(p_{1j}) & = 1.22699, \\ purity(1) = & \quad max_j(p_{1j}) & = 506/677. \end{aligned}$$

Comments on entropy/purity

- The ideal situation is one in which each cluster is pure (i.e. contains only documents from one class).
- In reality, this is unlikely to be the case. Nevertheless, many clusters are 'nearly' pure[2].
- e.g. In Cluster 3 is very good, containing almost entirely documents from the Sports section. This is reflected by the entropy being very low/purity being very high.
- To improve the entropy/purity of the others, we could increase the value of k.

[2] Perhaps 'reasonably' is a better word here.

Similarity-oriented measures

- All measures are based on the premise that any two objects that are in the same cluster should be in the same class, and vice versa.
- We can view this approach as involving the comparison of two matrices:

Similarity-oriented measures

- All measures are based on the premise that any two objects that are in the same cluster should be in the same class, and vice versa.
- We can view this approach as involving the comparison of two matrices:
  1. Ideal Cluster Similarity Matrix
     This has 1 in the ij^{th} position if two objects i and j are in the same cluster, and 0 otherwise;

Similarity-oriented measures

- All measures are based on the premise that any two objects that are in the same cluster should be in the same class, and vice versa.
- We can view this approach as involving the comparison of two matrices:
  1. Ideal Cluster Similarity Matrix
     This has 1 in the ij^{th} position if two objects i and j are in the same cluster, and 0 otherwise;
  2. Class Similarity Matrix
     This is defined with respect to class labels and has a 1 in the ij^{th} entry if two objects i and j belong to the same class, and 0 otherwise.

N.B. We can take the correlation of these two matrices as a measure of cluster validity. This measure is known as *Hubert's* Γ *Statistic.*

Example

Suppose the case where we have five data points P_1, P_2, P_3, P_4, P_5, two clusters $C_1 = \{P_1, P_2, P_4\}$ and $C_2 = \{P_4, P_5\}$, and two classes $L_1 = \{P_1, P_2\}$ and $L_2 = \{P_3, P_4, P_5\}$.

Example

Suppose the case where we have five data points $P_1,\ P_2,\ P_3,\ P_4,\ P_5$, two clusters $C_1 = \{P_1,\ P_2,\ P_4\}$ and $C_2 = \{P_4,\ P_5\}$, and two classes $L_1 = \{P_1,\ P_2\}$ and $L_2 = \{P_3,\ P_4,\ P_5\}$. We can then compute the *Ideal Cluster Similarity Matrix* and *Class Similarity Matrix* as follows:

$$ICSM = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \quad CSM = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Measures for binary similarity

- In general, we can use apply any of the measures for binary similarity to these matrices (we can convert the matrices into binary vectors by appending the rows).
- For clarity, we repeat the definitions of the four quantities used to define those similarity measures, modifying them slightly to reflect the current context.[3]

$f_{00} =$ *number of pairs of objects having a different class and a different cluster;*

$f_{01} =$ *number of pairs of objects having a different class and the same cluster;*

$f_{10} =$ *number of pairs of objects having the same class and a different cluster;*

$f_{11} =$ *number of pairs of objects having the same class and cluster.*

[3] Note that there are $\frac{m(m-1)}{2}$ pairs, where m is the number of objects.

Rand Coefficient and Jaccard Coefficient

- We consider the Simple Matching Coefficient (known as the Rand statistic in this context) and the Jaccard coefficient.
- We recall their definitions here:

$$\text{Rand Statistic} = \frac{f_{00}+f_{11}}{f_{00}+f_{01}+f_{10}+f_{11}},$$

$$\text{Jaccard Coefficient} = \frac{f_{11}}{f_{01}+f_{10}+f_{11}}.$$

Example

- Consider the matrices computed in the previous example.
- With the above notation, we have $f_{00} = 4,\ f_{01} = 2,\ f_{10} = 2,\ f_{11} = 2$.
- Thus, the Rand statistic is $(2 + 4)/10 = 0.6$ and the Jaccard coefficient is $2/(2 + 2 + 2) = 0.33$.

Summary

- We have discussed the Bisecting K-means algorithm.
- We have seen other problems with the K-means algorithm. These include the following:
  - Clusters of (very) different sizes.
  - Clusters of (very) different densities.
  - Clusters that are non-spherical.
- We have seen how to evaluate clusters, either using unsupervised techniques (such as clustering tendency, the number of clusters etc.), supervised techniques (such as comparing with externally known results), or a combination of both (such as by determining two sets of clusters).
- We have seen problems with choosing initial centroids.
- We have seen the Hopkins statistic, as well as how to use entropy in this context.
- We have seen the Rand Coefficient and Jaccard Coefficient.