Applied Data Mining

Rockhurst University

Clustering

Section 1

Introduction

Cluster analysis

- Our first unsupervised learning method: clustering. Unsupervised learning is ‘unsupervised’ because we do not have a target (outcome variable).
- Clustering is meant to be used for “knowledge discovery” instead of “prediction.” The basis of clustering is what sociologists call “homophily” — or birds of a feather flock together.
- The goal of clustering is to find groups, or clusters, in a data set. We want to partition our dataset so that observations within each group are similar to each other while observations in different groups are different from each other.

What is clustering?

Cluster: A collection of data objects

- Similar (or related) to one another within the same group
- Dissimilar (or unrelated) to the objects in other groups

Cluster analysis (or clustering, data segmentation, ?)

- Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes (i.e., learning by observations vs. learning by examples: supervised)

Let's define groups to see which observations are "like" each other.

How many clusters can/should we find?

Section 2

K-based clustering

Forming clusters

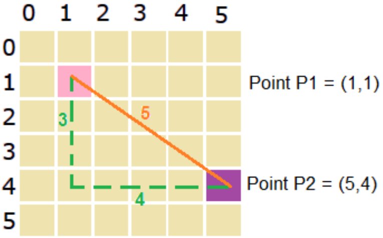
1. Choose number of clusters k
2. Partition the data set into k clusters so that the sum of squared distances is minimized between the data points (p) and some center point $[c(i)]$ in each cluster. $E = \sum_{i=1}^{k} \sum_{p \in C_i} (d(p, c_i))^2$ where $d(p, c_i)$ is the distance between the point and the center.

How do we determine the center? The most common methods are:

- **k-means**: Each cluster is represented by the center of the cluster
- **k-medoids or PAM (Partition around medoids)**: Each cluster is represented by one of the objects in the cluster

Measuring distance

Don't forget the most common measures are Euclidean and Manhattan distances.



Euclidean distance = $\sqrt{(5-1)^2 + (4-1)^2} = 5$

Manhattan distance = $|5-1| + |4-1| = 7$ [1]

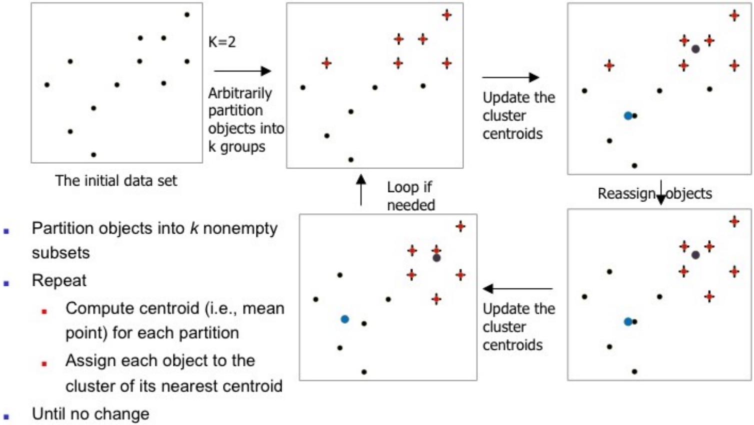
[1] Source: https://prismoskills.appspot.com/lessons/2D_and_3D_Puzzles/Chapter_05_-_Distance_between_points.jsp

Algorithm details

K-means in R uses the Hartigan & Wong algorithm:

- Choose centroids at random.
- Each data point is assigned to the nearest centroid.
- Update the value of the centroid as the mean of the data points.
- If the centroid has been updated in the previous step, then for each data point:
    - Calculate the within-cluster sum of squares if the data point is in its current cluster (let's call this cluster 1).
    - Calculate the within-cluster sum of squares if the data point is in a different cluster (let's call this cluster 2)
    - If the sum of squares of cluster 2 ¡ sum of squares of cluster 1, then move the data point from cluster 1 to cluster 2
- Repeat until no case can change cluster.

Process pictures


The initial data set
K=2
Arbitrarily partition objects into k groups
Update the cluster centroids
Reassign objects
Update the cluster centroids
Loop if needed

- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

[2]

Clustering in action: ▸ Example 1 ▸ Example 2

[2] Source:http://hanj.cs.illinois.edu/bk1/

Evaluating clusters

Okay so we created some clusters. Are they "good"? We look at:

- **Sum of squares** : measure of how tightly packed the data points are.
- **Cohesive**: high intra-class similarity — low within cluster sum of squares.
- **Distinctive**: low inter-class similarity — high between cluster sum of squares.

K-means tries to minimize within cluster sum of squares and maximize between cluster sum of squares.

Limitations of Kmeans

The k-means algorithm is sensitive to outliers!

Like the arithmetic mean, an object with an extremely large value may substantially distort the distribution of the data.

Example: If most of our observations can be represented by 2,2,2,3,3,3,4,4,4 — the mean is 3. If we add 23 to the list, then the mean is 5 which is larger than any of the original observations.

Is there a solution that isn't sensitive to outliers? **Median**!

For our simple example, the median for the original list is 3. When we add the large outlier (23), the median is still 3.

Kmediods

Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster

Kmediods finds the most central point for the cluster.

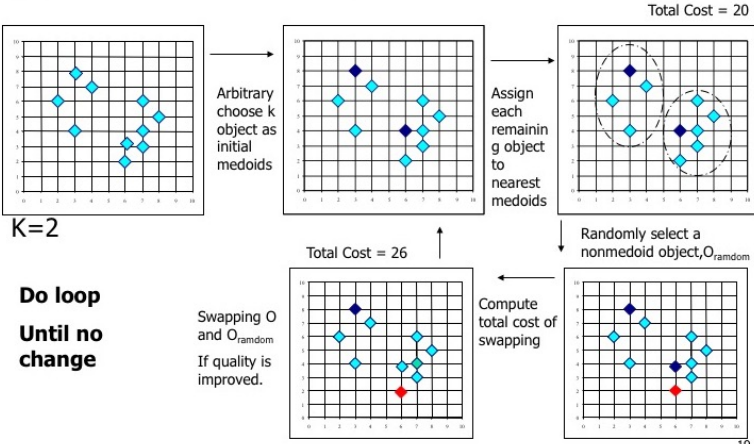
Popular algorithm: PAM (Partitioning Around Medoids, Kaufmann & Rousseeuw 1987)

- Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
- PAM works effectively for small data sets, but does not scale well for large data sets (due to the computational complexity)

Efficiency improvement on PAM

- CLARA (Kaufmann & Rousseeuw, 1990): PAM on samples
- CLARANS (Ng & Han, 1994): Randomized re-sampling

Process pictures



Total Cost = 20

K=2

Arbitrary choose k object as initial medoids

Assign each remainin g object to nearest medoids

Randomly select a nonmedoid object,O_{ramdom}

Total Cost = 26

Do loop

Until no change

Swapping O and O_{ramdom}

If quality is improved.

Compute total cost of swapping

[3]

[3] Source:http://hanj.cs.illinois.edu/bk1/

Choosing K

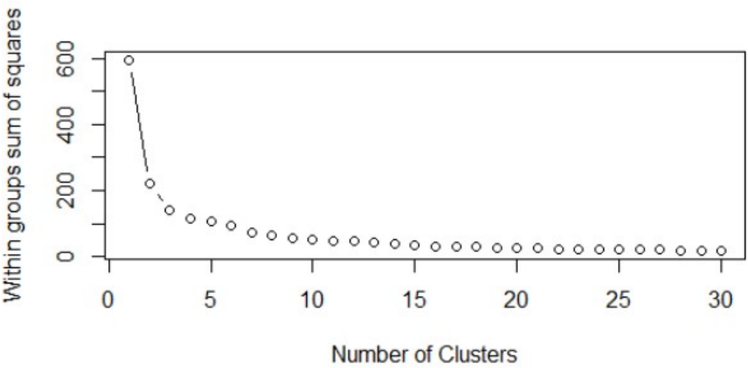
Another instance where we have to choose K. How do we do that?

- Look at cluster plots
- Compare the within and between sum of squares. We want cohesive clusters (low within SS) that are distinct (high between SS).

  If $Total = Within + Between$ then we want this ratio to be high:

  $\frac{between\ sum\ of\ squares}{total\ sum\ of\ squares}$
- Elbow method: more formal comparish that graphs the within sum of squares agains the number of clusters.

Another elbow chart



[4]

[4] https://www.dezyre.com/data-science-in-r-programming-tutorial/k-means-clustering-techniques-tutorial

Section 3

Heirarchical clustering

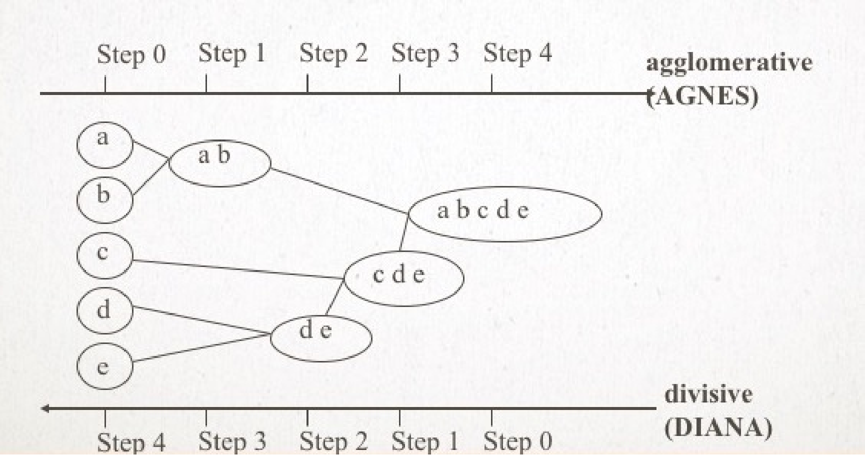
Alternative approach: don't choose K first

Instead of randomly picking *k* points and building groups around that, why don't we try something similar to a tree:

- The leaves are the individual data points.
- The root is the entire dataset (one big cluster).
- All the nodes in between are "clusters of clusters," so to speak.
- The tree in hierarchical clustering is called a **dendrogram**.
- There are two ways to grow the tree.
  - Agglomerative Nesting (AGNES) is a bottom-up approach.
  - Divisive Analysis (DIANA) is a top-down approach.

  The bottom up approach is more commonly used than the top down approach.

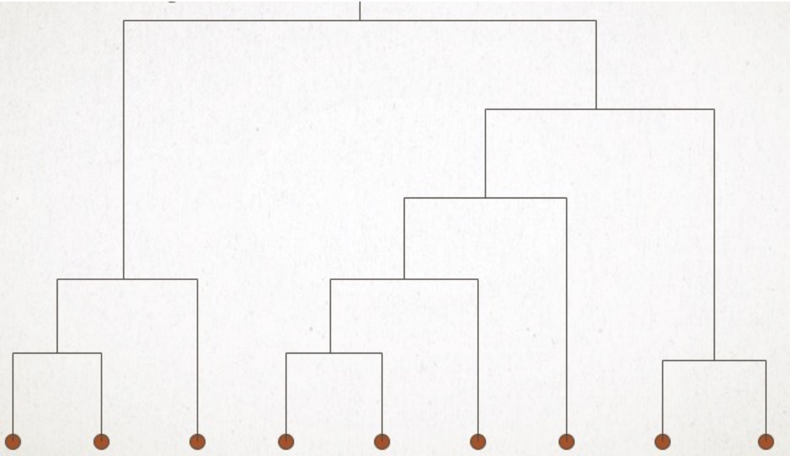
In pictures


Step 0
Step 1
Step 2
Step 3
Step 4
agglomerative
(AGNES)
a
b
c
d
e
a b
a b c d e
c d e
d e
divisive
(DIANA)
Step 4
Step 3
Step 2
Step 1
Step 0

Technical details

- Uses distance matrix as clustering criteria.
- Does not require the number of clusters k as an input, but needs a termination condition
- Decompose data objects into a several levels of nested partitioning (tree of clusters), graphically depicted as a dendrogram
- A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster

In pictures



Measuring distance

In kmeans and kmedoids, we measure distance between centroid (or medoid) and a data point. In hierarchical clustering, we measure distance between groups of data points (i.e. clusters).

If we are going to look at distance between clusters, we need to decide HOW to do this because a cluster has multiple data points.

- **Single linkage**: nearest distance between two records in two clusters. This method groups together records that are farther apart from each other in the early stages.
- **Complete linkage**: farthest distance between two records in two clusters. This method groups together records that are closer together in the early stages.
- **Average linkage**: Each pairwise distance is calculated, and the average of all such distances are calculated.
- **Centroid linkage**: Distance between group means is calculated.
- **Ward's method**: Maximize R-square when grouping records.

Hierarchical (Agglomerative) Clustering (AGNES)

This is the bottom up approach:

- Start with each record as its own cluster.
- The two closest records are merged into one cluster.
- At every step, the two clusters with the smallest distance are merged. This translates to mean that either a single record can be added to an existing cluster or two existing clusters are combined. (Schmueli et al 2018, p. 369).

Unlike k-means, hiearchical clustering only passes through the data set once.

Combining approaches

It is common to use hierarchical clustering to help choose k.

- Look at the dendrogram. Do you see distinct cluster groupings? Use your business knowledge.
- Rule of thumb: Look for the "step" (i.e. node level) that has the largest increase in height. We'll look at R code to do this.
- Pseudo t^2: Compare the calculated value against the critical value. Pick the smallest number of clusters where the calculated value ¡ critical value.

Section 4

Categorical data

Beyond Kmeans

Recall k-means uses the concept of mean (or average) when calculating centroids. What's the mean for a categorial variable? What's the median?

Neither of these concepts really fits well which is why they aren't really appropriate for categorical data. How "wrong" is it?

It depends: in some instances, it works just fine. That said, the interpretation of the centroids can be difficult. And as a practical consideration, some R packages don't accept factor variables.

One Hot Coding

When faced with categorical variables and software methods that don't accept them, we automatically recode them into dummy variables. This is referred to as "One Hot" coding (Ralambondrainy (1995)) . We've done this with "Gender" and other categorical variable where we would code them as:

Name	Yes	No
Female	1	0
Male	1	0

This is essentially what the **dummies** package in R does. Remember we used this with our "Drew" data example to create all binary variables.

Problems?

There are several drawbacks with this approach:

- Recoding into dummy variables mean you are increasing the size of the data set, and, consequently, the computational costs.
- The cluster centroid (i.e. mean) does not have a practical interpretation. You will get a mean value between 0 and 1, whcih variables.
- Euclidean distance does not make sense when you only have values of 0 and 1.

Even with these issues, this is a common approach in data mining.

Alternative approach #1

Instead of using Euclidean distance, you can use **Gower's similarity measure** and pair it with k-medoids. Gower's measure requires that all variables must be scaled to a [0,1] range.

Gower's measure is a "weighted average of the distances computed for each variable" (Shmueli et al. 2018, p. 366).

Here's the technical calculations of Gower's measure:

$$s_{ij} = \frac{\sum_{m=1} w_{ijm} s_{ijm}}{\sum_{m=1} w_{ijm}}$$

where s_{ijm} is the similarity between records i and j on measurement m and w_{ijm} is a binary weight given to the corresponding distance.

Gower's details

- For binary measurements, $s_{ijm} = 1$ if $x_{im} = x_{jm} = 1$ and 0 otherwise. $w_{ijm} = 1$ unless $x_{im} = x_{jm} = 0$.

  For example, if x_i and x_j are both cases where Female has Yes=1 then $s_{ijm} = 1$ and $w_{ijm} = 1$

- For nonbinary categorical measurements, $s_{ijm} = 1$ if both records are in the same category, and otherwise $s_{ijm} = 0$. $w_{ijm} = 1$ unless $x_{im} = x_{jm} = 0$.

  For example, with a 3 class factor variable (Red,Blue,Green), if x_i and x_j both have the same class (Blue), then $s_{ijm} = 1$ and $w_{ijm} = 1$

To calculate Gower's measure, you have to create a customized dissimilarity matrix and then apply one of the clustering algorithms.

Alternative approach #2

The **k-modes algorithm** (Huang, 1997b) extends the k-means paradigm to cluster categorical data by using

- a simple matching dissimilarity measure for categorical objects (Kaufman and Rousseeuw, 1990),
- modes instead of means for clusters, and
- a frequency-based method to update modes in the k-means fashion clustering process to minimise the clustering cost function.

How does it really work?

Simple dissimilarity measure

Kmeans can use, and hierarchical clusters require, a **dissimilarity matrix** as input. The dissimilarity concept is related to distance: if points are far away from each other, they are not similar. Kmodes changes that definition to consider similarity to be based on *mismatches*:

- Let X and Y be two categorical objects described by m categorical attributes.
- The dissimilarity measure between X and Y can be defined by the total mismatches of the corresponding attribute categories of the two objects.
- The smaller the number of mismatches is, the more similar the two objects.
- This measure is often referred to as *simple matching* (Kaufman and Rousseeuw, 1990).

Similar process

Now that we have a “distance” measure, we need to form clusters using it. The process is very similar to Kmediods:

- Select k initial modes; one for each cluster.
- Assign an object to the cluster whose mode is nearest to it. Update the mode of the cluster after each assignment.
- After all objects have been assigned, retest the dissimiliarity of objects against the current modes. If an object is found such that its nearest mode belongs to another cluster rather than its current one, reassign object to that cluster and update the modes of both clusters.
- Repeat step abote until no object has changed clusters after a full cycle test of the entire data set (Huang 1998, p. 290).

Notice that you still must test out multiple k's to find the best one when working with k-modes algorithm.

Section 5

Mixed data

Clustering with mixed data types

We have approaches for continuous data and approaches for categorical data, but what if you have both in your dataset?

You can use Gower's Measure with K-Medoids with the appropriate preprocessing: Convert all the categorical variables into [0,1] range and min-max normalize all the continuous variables into [0,1] range.

- Pro: we know how to do this and the packages are well developed.
- Con: computationally (and time) intensive and k-mediods doesn't scale up for large data sets.

k-prototype

Huang (1997a; 1997b; 1998) proposed an extension of the k-modes algorithm that is suitable for clustering continuous and categorical variables. k-prototype is not computationally costly and can be scaled up to large data sets.

Assume that we have two mixed-type records, X, and Y. Each record has multiple attributes (or variables), both numeric and categorical.

The dissimilarity between two mixed-type objects is described as the sum of two components:

$dissimilarity(X, Y) = E + \lambda M$

Where E is the squared Euclidean distance measure on the numeric attributes (i.e. k-means) and M is the matching dissimilarity measure on the categorical attributes (i.e. k-modes) λ is a weight value.

Choosing λ

Now we have another choice to make: λ

Huang suggests:

- Use the average standard deviation of numeric attributes as the default λ.
- Smaller λ favors numeric attributes
- Larger λ favors categorical attributes.

This is really new: the R package 'clustMixType' was released in October 2017.