Lecture Notes on Clustering – April 2022 – Data Science class

These lecture notes are an outline of key concepts in clustering – unsupervised learning to organise data points into groups or clusters related either be mean distance or by density.

At the end of the notes are exercises and tasks to complete for 2 study points

All of the clustering algorithms use "distance" so we need to be clear what is meant by that

https://towardsdatascience.com/3-distances-that-every-data-scientist-should-know-59d864e5030a

https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

These algorithms are for quantitative data – for categorical data you can use K-modes and with strings you can also use Hamming Distance on strings of the same length.

K-Means

K-Means is a *clustering algorithm*, that means, it can split a training set X` into :`K` clusters, or as Pena et al. puts it, "the fundamental data clustering problem may be defined as grouping similar objects together".

Pena, José M., Jose Antonio Lozano, and Pedro Larranaga. "An empirical comparison of four initialization methods for the k-means algorithm." Pattern recognition letters 20.10 (1999): 1027-1040.

Clustering (in unsupervised learning) is very similar to the problem of classification in supervised learning.

Pena et al. describes the algorithm in detail.
The data set is initially split into `K` clusters, and the *cluster centroids* are calculated for each cluster, as a feature vector with each feature set as the mean of that feature in the data set. Then, for each sample, the euclidean distance (distance function) to each cluster centroid is calculated, and the sample assigned to the centroid with the smallest error. This process repeats until convergence.

Pena et al. discusses various initializations of the algorithm, but end up concluding that the usual random initialization perform quite well.

The K-Means algorithm is widely used and is considered a very well performing algorithm.

There is a description in Ch 20 Clustering (pp 263 – 277) of the book and you can use the code to create your own clustering tool.

Use Cases for K Means

- Market Segmentation (marketing and advertising)
- Data Analysis
- Anomaly Detection (finding outliers)
- Dimensionality reduction (dimensions becomes number of clusters not all the features)
- Segment an image (reduce the total number of colours in the image – used in tracking applications to speed up searching…)
- Search Engines image searches

It is considered to be one of the fastest clustering algorithms.

Disadvantages

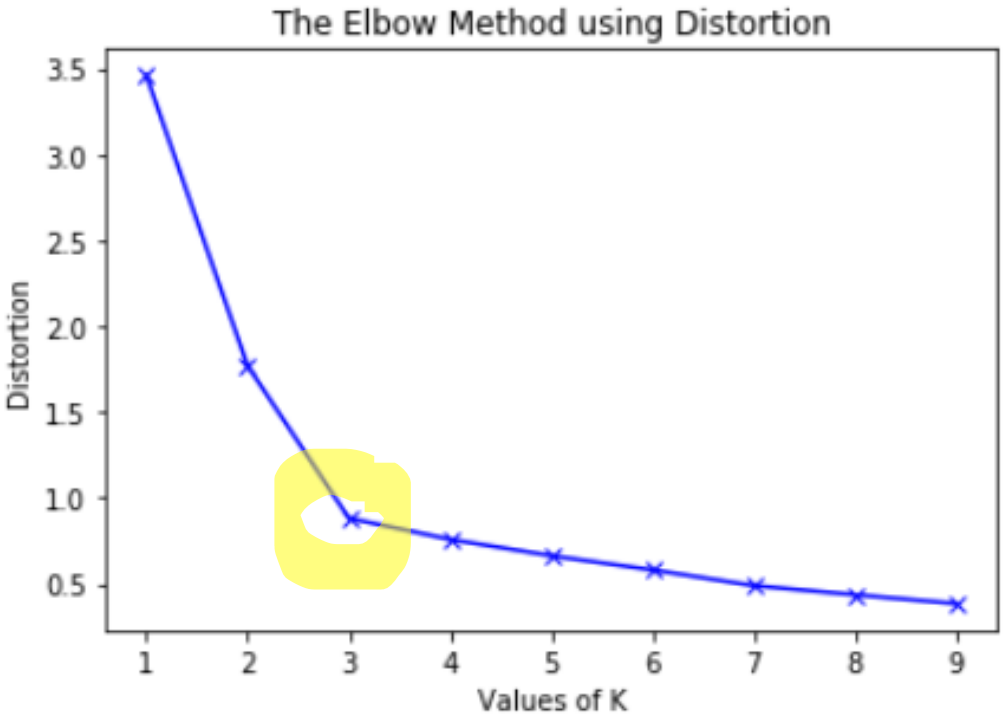
Need to run the algorithm a number of times to try and find an optimal solution as lots depends on initialisation value (the centroid or central values for the clusters from which distance is measured)

Doesn't handle odd shapes well (e.g. elliptical) and can create very odd clusters as a result.

Finding 'K'

The performance of a particular initialisation can be measured with 'inertia' – this is the value of the distance between a data poins and the nearest centroid (the mean squared error) – for any value of K the lowest inertia is the best model. However, as the number of clusters increases the inertia value falls (any data point is going to be nearer a centroid) so not always helpful.

The inertia values can be plotted against different values of K however and usually there is an 'elbow' where an increase in K does not lead to much decrease in inertia


The Elbow Method using Distortion
Distortion
3.5
3.0
2.5
2.0
1.5
1.0
0.5
1
2
3
4
5
6
7
8
9
Values of K

DBSCAN

Density Based Spatial Clustering of Applications with Noise

Rather than group things based on distance from a mean it groups densely packed items together

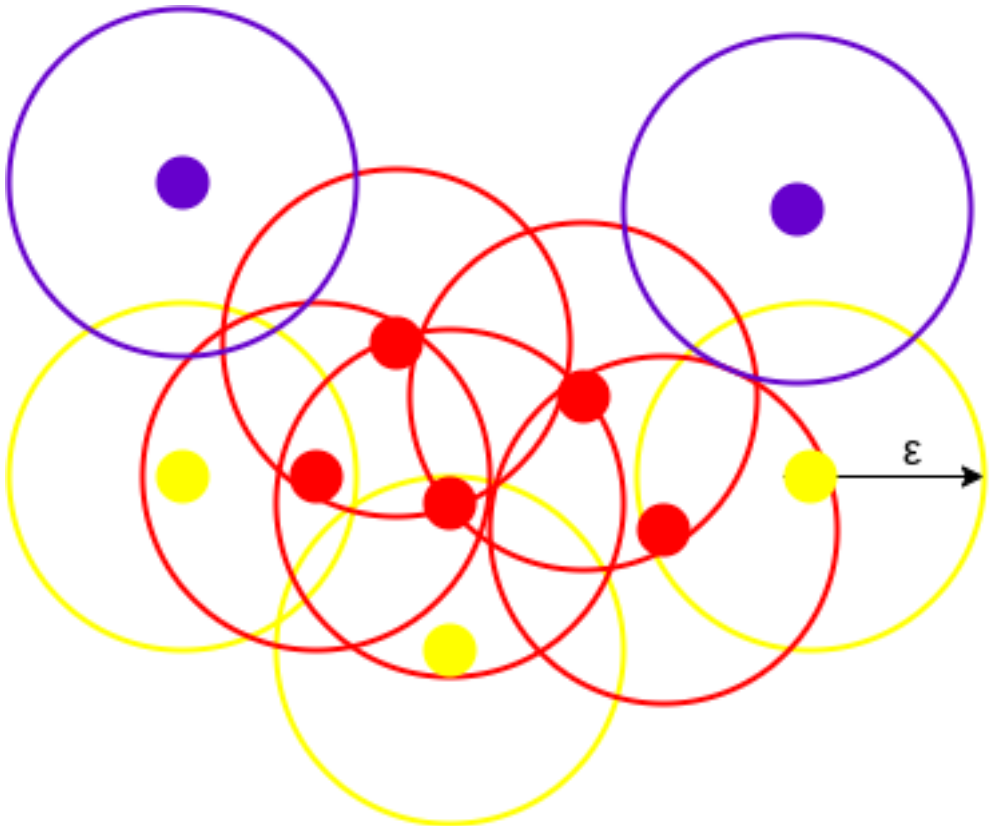
No need to specify the number of clusters beforehand

Two inputs
Epsilon – the readius of a circle around each datapoint to check density (thus putting all things inside the radius in the same dense cluster)

Minpoints – the minimum number of points required for something to count as a cluster (this is how it is good at excluding outliers or 'noise')

In higher dimensional data sets epsilon is a "hypersphere" but epsilon still measures the radius of the sphere

Core points



Red circles are core points – at least 3 data points are inside the epsilon of each one. Yellow are border points – yellow are 'border points' more than one data point in circle but less than minpoints. Purple are 'noise' as there is only itself inside epsilon

DBSCAN uses Euclidean distance – but depending on the library can use other distances

If 2 data points are in the same cluster they have CONNECTIVITY. If a data point can be reached either directly or indirectly in another cluster it has REACHABILITY.

Selecting Values for Epsilon and Minpoints

Minpoints – no point having it as 1 as each point would be its own cluster so 3 is usually seen as the minimum number. Ideally it will be at least the number of dimensions of data + 1

(as DBSCAN is used a lot with 2D data this means 3 is a very common starting point for minpoints).

Epsilon – as with K Means the value of epsilon can be taken from an elbow graph

Good introduction with an example in Python

https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/

Exercises – upload answers on Peergrade by 3 May 2022

1. Work through the book – build your own K Means cluster function and carry out the modelling done in the book.
2. Write a brief description of the use cases for K Means and DBSCAN algorithms. Hints on good use cases are in the notes – do further research as needed (e.g on DBSCAN used in recommendation engines
3. Thinking about these two algorithms provide a formal definition of clustering
4. Describe with examples of plotting two ways of selecting the value of K for K-means clustering
5. Which algorithm is good for large datasets and which one is good for regions of high density
6. What does 'high density mean' in DBSCAN
7. Describe with an example how the initialisation of K Means can affect the inertia value and the quality of the model.

8. Prepare a model using K Means for the Olivetti data set from scikit learn – this exercise comes from Gueron p275

   The Olivetti facial recognition data set contains 400 greyscale images of faces, each person was photographed 10 times. Load the dataset from Sklearn.datasets.fetch_olivetti_faces(). Split the dataset ensuring there are the same number of images of each person (stratified sampling). Cluster the images using K means and ensure you have a good number of clusters. Visualise the data.

   (code for this is available at https://github.com/ageron/handson-ml2)

Sources

Pena, José M., Jose Antonio Lozano, and Pedro Larranaga. "An empirical comparison of four initialization methods for the k-means algorithm." Pattern recognition letters 20.10 (1999): 1027-1040.

Hands On Machine Learning with Scikit Learn, Keras and Tensor Flow, 2nd Edition.
Gueron A, O'Reilly Boston – ISBN 978-1-492-03264-9