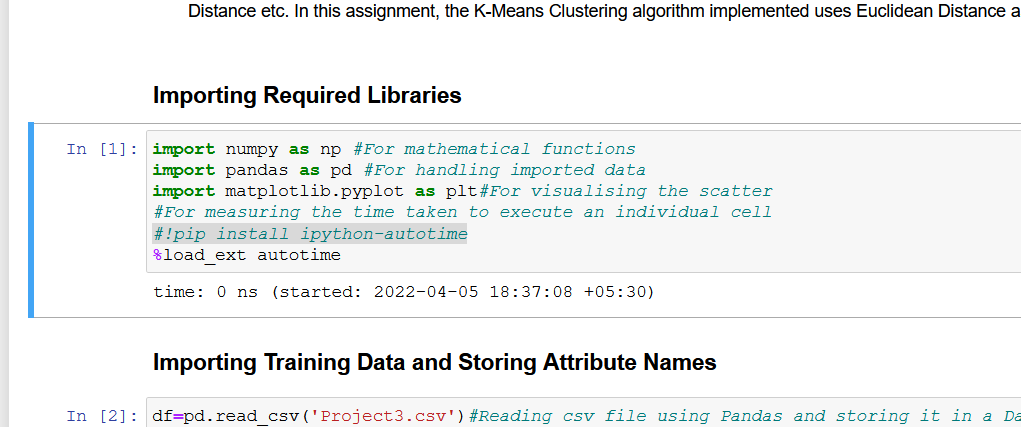
Machine Learning

Project 3: K-Means Clustering

**Instructions to Use:**

1. The file is in .ipynb format so please use Anaconda-Jupyter notebook or Google Colab to run it.
2. Store the .csv data file in the same directory with the name “Project3.csv”.
3. Run each block of code sequentially to avoid errors.
4. If execution is done in Google Colab, refresh the file directory after execution to view the output files (if not visible immediately).
5. If the execution is done on Google Colab or if autotime is throwing an error on your local PC, then please remove the # symbol present at the start of the highlighted line (in the image below). This will install the necessary files for autotime to work. The autotime function is used to display the execution time of each cell.



**Approach and Analyses:**

In this assignment, I have developed the K-Means Clustering Algorithm for segmenting the target market of a supermarket based on a customer database consisting of 2000 customers. I have done the experiment in two sections. The first section involves a very simple analysis and segmentation of the market based on only the numerical attributes namely the ‘Age’ and the ‘Income’. The second section involves One-Hot-Encoding of the categorical data followed by clustering based on all the attributes. In either case, however, the ‘ID’ column is dropped since it does not provide any relevant information and is used merely for serializing the data. In order to stick to the file naming convention, the first output file is named ‘19EC30055\_P3.out’ while the second one is named ‘19EC30055\_P3\_OHE.out’ implying that ‘19EC30055\_P3.out’ is the main output file.

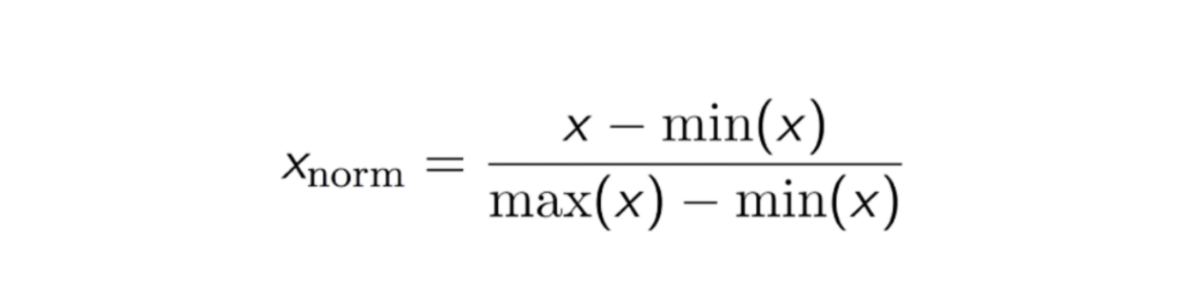
**K-Means Clustering Algorithm**

* The K-Means Clustering algorithm is one of the simplest and most popular unsupervised learning algorithms.
* It is a part of a class of unsupervised learning algoithms called Clustering algorithms. Clustering involves grouping data points into different sets based on their degree of similarity. It is majorly of two types - Heirarchical Clustering and Partitioning Clustering. Heirarchical clustering can be further subdivided into Agglomerative and Divisive Clustering while Partition Clustering can be divided into K-Means Clustering and Fuzzy C-Means Clustering.
* K-Means performs the division of objects into clusters that share similarities and are dissimilar to the objects belonging to another cluster.
* The term K refers to the number of clusters in which the algorithm is required to divide the data. It is usually defined by the data analyst. The optimal K for a given dataset can be obtained by running the algorithm for a range of K's and choosing that K which is not too large (to avoid high computation times) but also sufficiently minimises the scatter within a cluster (intra-cluster distance).
* The similarity between data points is quantified using a distance metric. This distance metric can be one of several metrics - Manhattan Distance, Euclidean Distance etc. In this assignment, the K-Means algorithm uses Euclidean Distance as the metric.

**Normalisation**

Normalisation is a scaling technique in Machine Learning applied during pre-processing to change the values of the columns in the dataset to use a common scale. Since the KMeans Clustering algorithm is a distance-based algorithm, it is very prone to errors due to different ranges of features. Hence normalization is necessary so that all the attributes have values in the range 0-1 and no attribute dominates over the other because of mismatched ranges. In this dataset, the Income attribute has a maximum value of 309364 and a minimum of 35832 while the Age attribute has a minimum value of 18 and a maximum value of 76. Thus, Income has a range of 273532 and Age has a range of 58. Without normalization, the distances between data points will be dominated by the Income attribute and any information that the Age attribute may provide will be lost.

Normalisation is done using the following formula:



That is, every attribute of the column is reduced by the minimum value in that column and then divided by the range of that column. This way all values in the column will take real values between 0 and 1 and no single column will dominate Euclidean distance merely on the basis of a larger range.

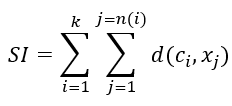
**K-Means Implementation:**

The detailed step-by-step implementation, explanation along with well-commented code is provided in the Jupyter notebook. The explanation here will therefore be brief and to the point.

The K-Means is implemented as a class which contains several member functions. The predict function is the main driver which uses other functions to perform the segmentation. The random function of the numpy.random class is used to randomly choose k indices from the 2000 entries. These k points are set as the centroids initially. Next, the Euclidean distance between each of the other data points and these centroids is found one-by-one and the data point is assigned to the cluster corresponding to that centroid with which it has the highest proximity (smallest Euclidean distance). The mean of these clustered points is then found and is set as the new centroid. This is done for each of the k clusters. This process is repeated until convergence – when the new centroids obtained are the same as those in the previous iteration. A maximum iteration limit of 200 was set but convergence often occurred in much lesser iterations.

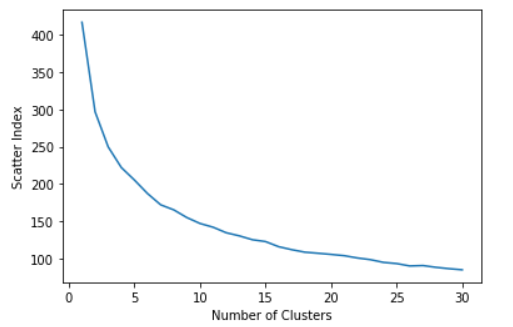
**Choosing Optimal K – Elbow Method:**

A fundamental step in an unsupervised learning algorithm such as K-Means is to determine the number of clusters into which the data is to be segment. For the K-Means algorithm this is done using the Elbow Method. In this method, some scatter index is used to quantify the quality of clustering. The scatter index I have used is the sum of intra-cluster distances over all clusters – the Euclidean distance between all the points within a cluster and the cluster centroid are calculated, added together and the sums thus obtained for all clusters are again summed together, that is,



Where k is the number of clusters, n(i) is the number of data points in the ith cluster, ci is the centroid of the ith cluster, xj is the jth data point within the cluster and d(g,h) is a function that returns the Euclidean distance between g and h.

While the scatter index will surely reduce with an increase in k, a very high k would require a lot of computation time and therefore it is necessary to strike a trade-off between the number of iterations and the scatter index. In the elbow method, the graph of scatter index is plotted against the number of iterations upto some chosen maximum value of k and then the value of k at the “elbow”/point of inflection of the graph is chosen as the optimal number of clusters. The plot obtained in the first section is shown below:



We can see that the elbow point was obtained when the number of clusters was around 9.

**Problem with Categorical Attributes:**

The K-Means algorithm cannot work when the data contains both numerical and categorical attributes as distances between two data points cannot be naturally defined for the categorical attributes. In such a scenario, one of two alternatives can be chosen. The first one is to simply drop the categorical attributes and use only the numerical attributes. Another alternative can be to One-Hot-Encode the data (or use other methods of converting the categorical data to numerical data) and then run the K-Means algorithm.

**One-Hot Encoding**

For categorical attributes without any ordinal relationship, the integer encoding of categories is replaced with new binary attributes. The number of binary attributes depends on the number of categories in the original attribute. These binary attributes are also called dummy attributes and are a useful means of representing non-numerical values. For example, 'Settlement Size' is an attribute with values 0, 1 and 2 representing small, mid-sized and big cities respectively. These can be replaced by three binary attributes namely, 'Small city', 'Mid-sized city' and 'Big city' respectively. If the 'Settlement Size' for a particular data entry was initially 2 (big city), the three new attributes will have the values 0, 0 and 1 respectively. Similar encoding is done for the rest of the categorical attributes.

**Drawbacks of One-Hot-Encoding**

* The analysis with OHE gave a very jagged scatter index plot with no well-defined elbow even after running the algorithm for 50 clusters. This is because OHE has led to the introduction of several new features leading to what is called the “curse of dimensionality” – very high dimensional data can cause parallelism and multicollinearity.
* For analysis, 27 was chosen (very conservatively) as a good enough cluster number but looking at the practicality side of things creating 27 market segments would also not be feasible.

**Conclusion:**

* The K-Means Clustering algorithm is a very useful tool for segmenting unlabelled data into clusters with similar data points and is therefore useful for market segmentation, insurance fraud detection, identifying crime-prone areas, etc.
* It is very useful when the data consists of purely numerical data. The first analysis consisting of only the “Age” and “Income” attributes demanded the need for 9 (though 7 or 8 could also be chosen) market segments which was fairly reasonable.
* In the presence of mixed data – numerical and categorical, the K-Means algorithm may provide acceptable results only when the number of categorical attributes as well as the number of possible categories per attribute is small unlike the given dataset.