COMP3003 Machine Learning

**Exercise 1: Unsupervised machine learning techniques**

*Reece Davies [10572794]*

**Plymouth University**

**2020**

# Table of Contents

Contents

[Table of Contents 2](#_Toc58497912)

[Introduction 3](#_Toc58497913)

[Literature Review 3](#_Toc58497914)

[Define Unsupervised Machine Learning 3](#_Toc58497915)

[Real world example 1: Email spam filter 3](#_Toc58497916)

[Real world example 2: Anomaly detection for fraudulent activity 4](#_Toc58497917)

[Implementation of Kmeans in MATLAB 4](#_Toc58497918)

[Generating the dataset 4](#_Toc58497919)

[Concatenating the training datasets into a single dataset 5](#_Toc58497920)

[Implement Kmeans clustering 6](#_Toc58497921)

[References 11](#_Toc58497922)

# Introduction

This document will analyse and explain how unsupervised machine learning can be implemented in real world scenarios, specifically with use of the technique ‘Kmeans clustering’. Examples will be given of specific problems and why unsupervised machine learning techniques would prove to be beneficial in creating a solution. Furthermore, Kmeans will be implemented in the software ‘MATLAB’ to demonstrate the process of unsupervised machine learning with a designated dataset; code, diagrams, and results will be used in correlation to this.

# Literature Review

## Define Unsupervised Machine Learning

Unsupervised machine learning algorithms analyse the patterns from a dataset without being given any indication of what the values have been labelled as, or what outcome should be derived. The algorithm will independently detect patterns in the training dataset and use it as a reference for identifying similar patterns in the testing dataset. Furthermore, the primary objective of using unsupervised learning over supervised learning, is to gain a better understanding of the overall structure of the data. By making use of these algorithms, it allows the user to conduct tasks that supervised learning would not be able to achieve. However, unsupervised learning can be somewhat unpredictable when compared to other learning techniques as there is no correct answer to support the testing dataset.

Unsupervised learning can be categorised into to two main categories: clustering, and association. Clustering is used to discover the inherent groups of the data by identifying recognisable patterns that define each class. Some methods of clustering are as follows: ‘hierarchical clustering’, ‘k-means’, ‘mixture models’, ‘DBSCAN’, and ‘OPTICS algorithm’. Association on the other hand, is used for discovering key relations between separate objects in a large dataset by pinpointing items that frequently occur together. This document will focus on the application of Kmeans clustering: a centroid based algorithm that groups the dataset into separate clusters by calculating the distance assigned to a designated point in each cluster, known as a centroid. Each cluster will be associated with a specific centroid, where the quantity of centroids is defined by ‘K’. A centroid of a particular cluster is calculated with the cluster’s mean value. All data points will be assigned to a specific cluster, which is computed based on the set distance between the data point and all available centroids. The centroid will again recompute each cluster because of the different values that come into the group; this step is again repeated until there are no changes to the output.

Example of Kmeans

## Real world example 1: Email spam filter

Kmeans clustering has proven to be an effective tool in identifying spam emails; businesses implement Kmeans algorithms in the user’s mailbox, where it flags whether a recently received email is considered spam. The algorithm analyses the contents of the different sections of the email – header, body, and sender – which are grouped together into separate clusters. These groups are then classified as ‘spam’ or ‘genuine’. Each email is categorised by implementing the method ‘vector determination’. The primary goal of using machine learning in a spam filter is to improve the reliability of identifying spam, and therefore means it will be less likely to flag a genuine email as ‘spam’, specifically those coming from an automated newsletter system.

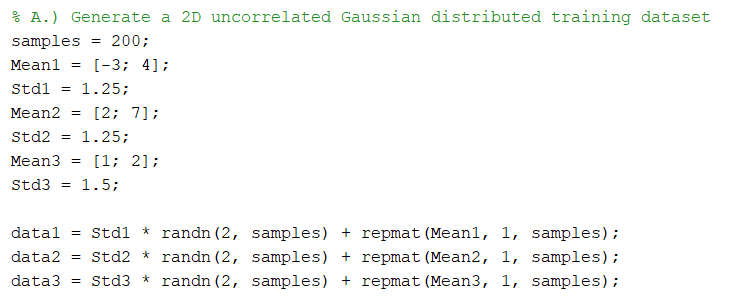
## Real world example 2: Anomaly detection for fraudulent activity

Due to the rapid growth in electronic ecommerce services, there has been an immense increase of fraudulent activity, specifically in the banking industry. Fraud is the term used to describe the act of criminal deception committed specifically to secure illegitimate personal gain. In this case, fraud applies to unauthorised activity taken place in electronic payment systems. As a result of the ease of use and time efficiency of online banking, customers prefer to use an online payment method as opposed to face-to-face transactions. This of course means fraudulent transactions have also increased and therefore techniques need to be implemented to detect such crimes.

By making use of Kmeans clustering algorithms, it is possible to analyse customers’ activity in search for anomalies that may potentially be acts of fraud. To specify, an anomaly is an element with properties that differ from the majority of observed data. The Kmeans algorithm classifies ‘n’ number of data points into the pre-specified ‘k’ number of clusters, where it will be grouped based on the transactions’ shared similarities. The groups would most likely be dependent on the risk or probability of a particular transaction being associated with an act of fraud; for instance: K is defined as 3 and each transaction is categorised as ‘low risk’, ‘medium risk’, or ‘high risk’. Furthermore, Kmeans can also be applied to detect other types of fraud or anomaly based searches, such as insurance claims, data mining for healthcare fraud, network traffic anomalies, etc.

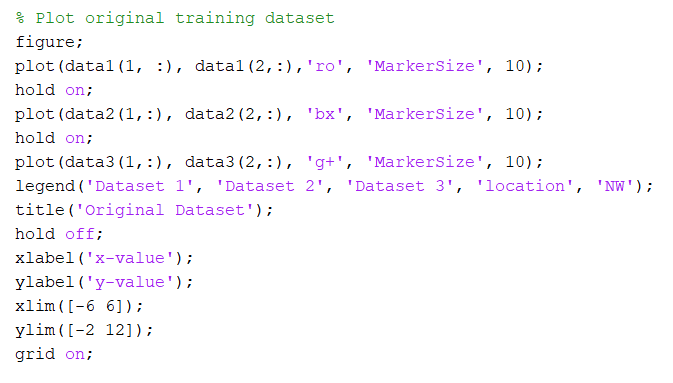
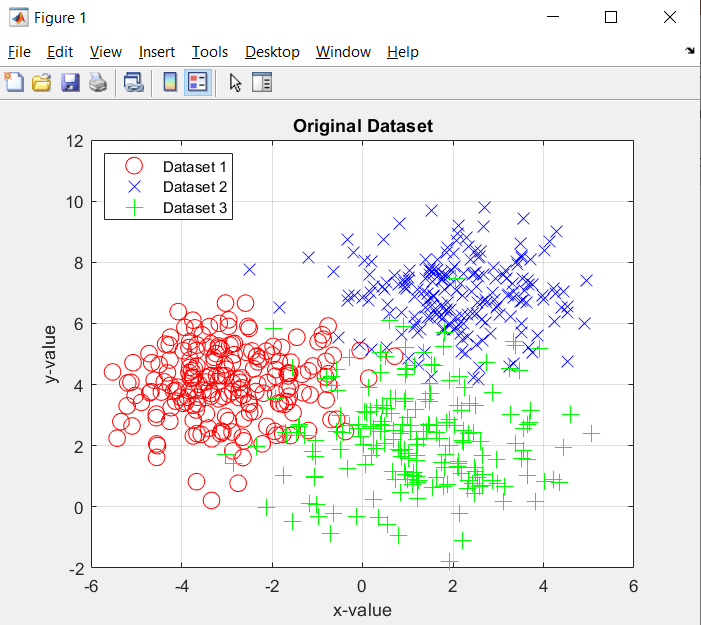
# Implementation of Kmeans in MATLAB

## Generating the dataset

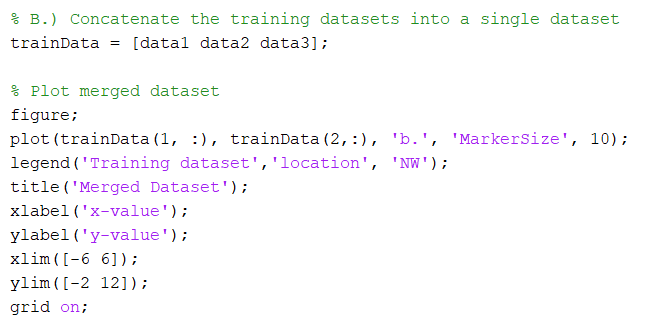
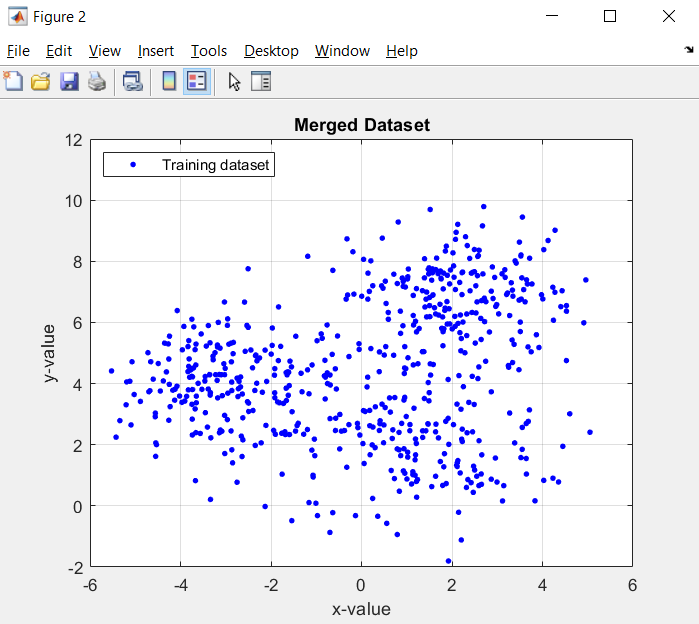
The specification demanded that the dataset was a 2D uncorrelated Gaussian distributed training dataset with a total of six hundred individual points for three separate clusters. The three clusters have been declared as “**data1**”, “**data2**”, and “**data3**” alongside with a designated mean and standard deviation variable. These variables have been structured in such a manner to ensure that the clusters remain separate from one another but contain certain datapoints that overlap to a neighbouring cluster. To specify – the mean controls the centre of the cluster, whilst the standard deviation is responsible for how dispersed the data is in relation to the mean’s value. The larger the standard deviation, the more dispersed the datapoints will be from the mean; if the standard deviation is a smaller value however, the datapoints will be more concentrated.

Each cluster is generated by multiplying the standard deviation with a 2-by-200 matrix of normally distributed random variables, which is made by using the MATLAB “**randn**” function and adding the result to a 1-by-200 array that contains duplicate values of the specified mean, that is generated with the use of the MATLAB “**repmat**” function.

Subsequently, this dataset was plotted on a two-dimension plane using the MATLAB “**plot**” function, as stated in the assignment brief. The three clusters have been given separate colours and shapes to help the user distinguish one cluster from another; “**data1**” is red ellipses, “**data2**” is blue crosses, and “**data3**” is green pluses.

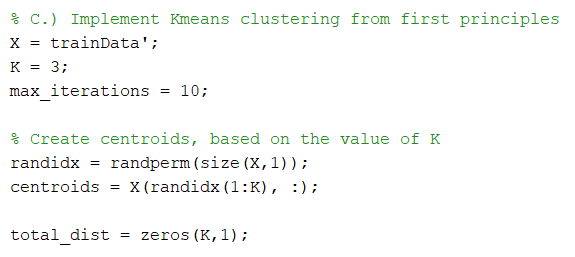
 

## Concatenating the training datasets into a single dataset

 The three separate clusters needed to be concatenated into a single merged dataset, as the Kmeans algorithm will need to be trained with one dataset of n number of points. This was simply created by joining the three clusters into a single matrix which was declared as “**trainData**”. As a result, “**trainData**” becomes a matrix consisting of two rows and six hundred columns, with the first two-hundred columns consisting of the values from the first cluster, the second two-hundred from the second cluster, and the third two-hundred from the third cluster, respectively.

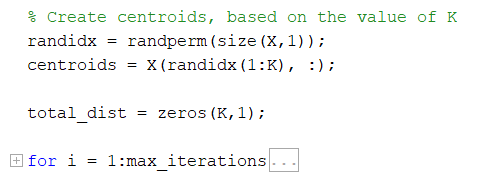
This new dataset was then plotted onto a two dimensional plane, similarly to when plotting the three clusters. Of course, the identity of each cluster is no longer shown on the graph as separate colours because this figure only consists of one dataset.

## Implement Kmeans clustering

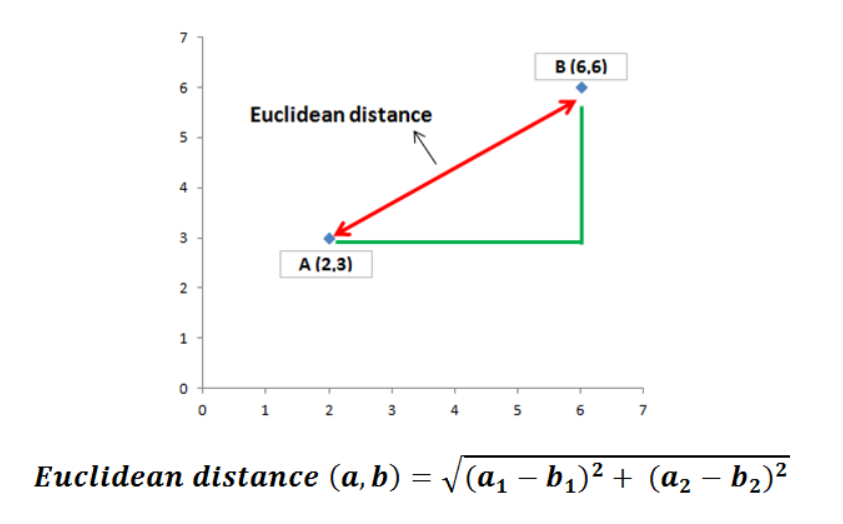


Once the three clusters were integrated into a single dataset, it needed to be implemented into the Kmeans algorithm. Firstly, the “**trainData**” matrix was required to be transposed in order for the algorithm to function correctly; this was done by creating a new variable “**X**” that equals to the transpose of “**trainData**”. The dataset needs to be transposed because the Kmeans algorithm takes a matrix where each column corresponds to the observed sample. Furthermore, transposing a matrix in MATLAB can be accomplished by accompanying the matrix with a single quotation mark (**‘**). To specify – transposing a matrix interchanges the row and column index for each element, and thus the end result is a matrix where the rows and columns have been replaced with one another. The variable “**K**” will be number of clusters that the dataset must be categorised into, which has been declared as 3 in this scenario. Another variable, “**max\_iterations**”, is to specify the number of times that the centroid of a particular cluster is calculated, with each iteration creating a more accurate outcome.

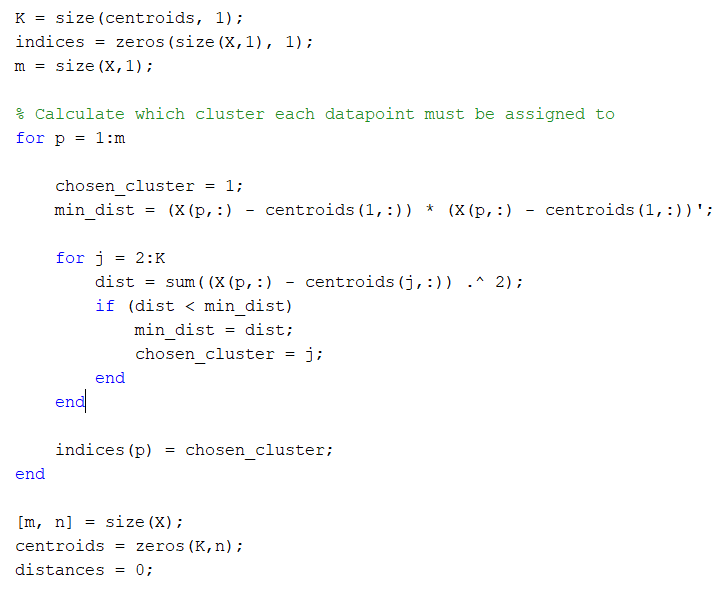
Example of transposing a matrix

Once K had been declared, the centroids for each cluster need to be calculated; firstly, the variable “**randidx**” was created with the use of the MATLAB “**randperm**” function, taking the number of rows of the training dataset “**X**” as its argument. As a result, “**randidx**” becomes a single row vector with each column consisting of a random permutation of integers from 1 to 600 without any repeating elements. The centroids were declared within the “**centroids**” variable, which becomes a 3x2 matrix. A ‘for’ loop is declared which is executed ten times, as stated with the use of the “**max\_iterations**” variable; all centroid calculations are performed from within this loop.

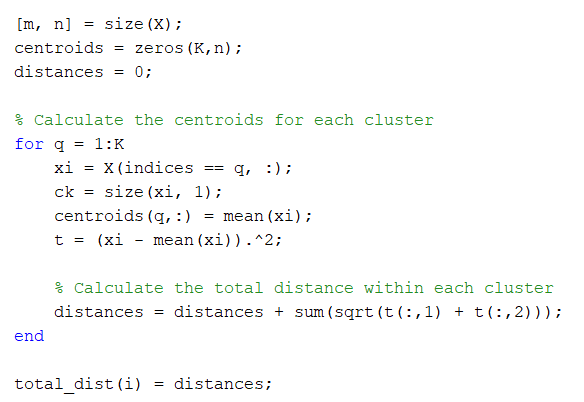
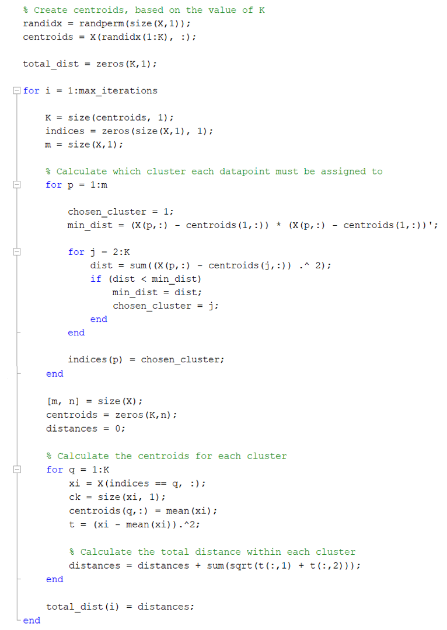
Kmeans functions is an iterative method based on the “Expectation Maximisation algorithm” and once the number of cluster has been determined, the algorithm must calculate the distance between all the datapoints and each centroid. Firstly, the algorithm calculates which cluster each datapoint must be assigned to. A new ‘for’ loop is created and is executed for every datapoint in the training dataset, therefore it will be run six hundred times, as stated with the variable “**m**”. The minimum distance is calculated between the chosen datapoint and a specific centroid, which is done with the use of the “Euclidian Distance” formula. The variable “**min\_dist**” is used to represent this value.



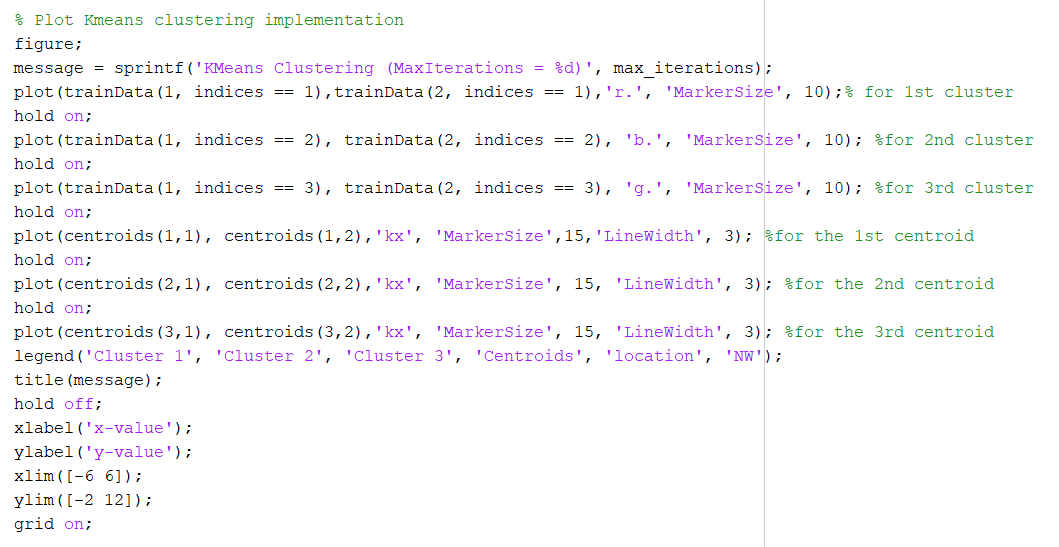
A new ‘for’ loop is declared and is executed for the remaining centroids. The distance between a datapoint and the given centroid is then calculated from within this loop and assigned to the “**dist**” variable. If this distance is smaller than the minimum distance previously calculatd, the minimum distance variable is then assigned to this new value, as the datapoint is closer to this specific centroid. Subsequently, the datapoint is assigned to the cluster that is dependent on the chosen centroid; this is represented in the “**chosen\_cluster**” variable.

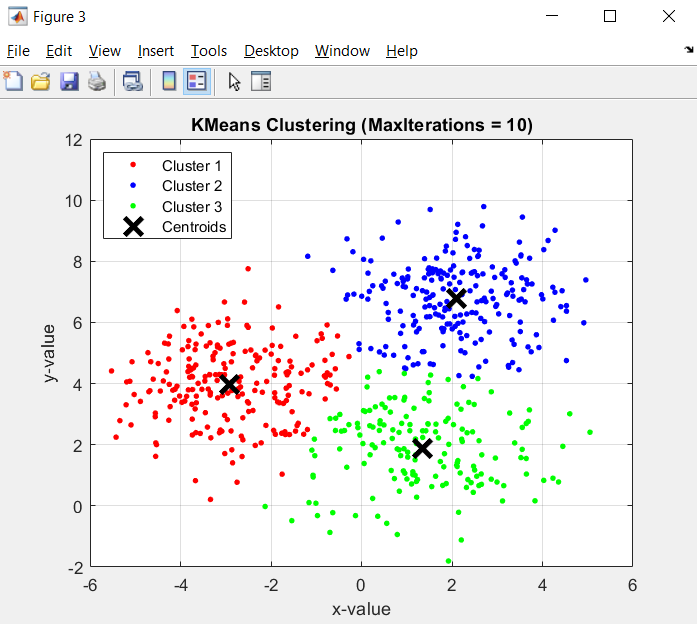


Once the different clusters have been declared, the algorithm will calculate the centroids for each cluster. With each iteration, the chosen centroid becomes more accurate to the centre of the cluster. Of course, the initial centroid may potentially be miscalculated and thus based on the outer edge of the cluster, which in turn would hinder the classification process; this is why the centroids are repeatedly calculated for more accurate results. This done by setting the centroid to the mean value of all the datapoints within a designated cluster. Furthermore, the total distance is calculated if the user intended to create a plot for the learning rate of this particular calculation.

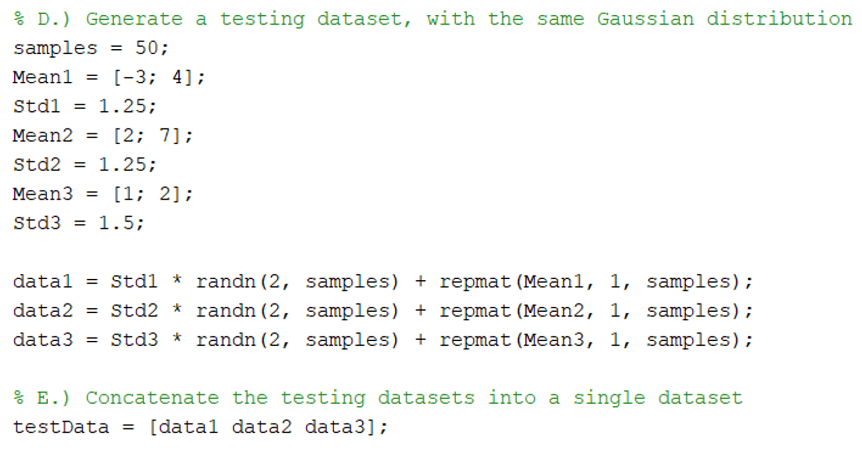


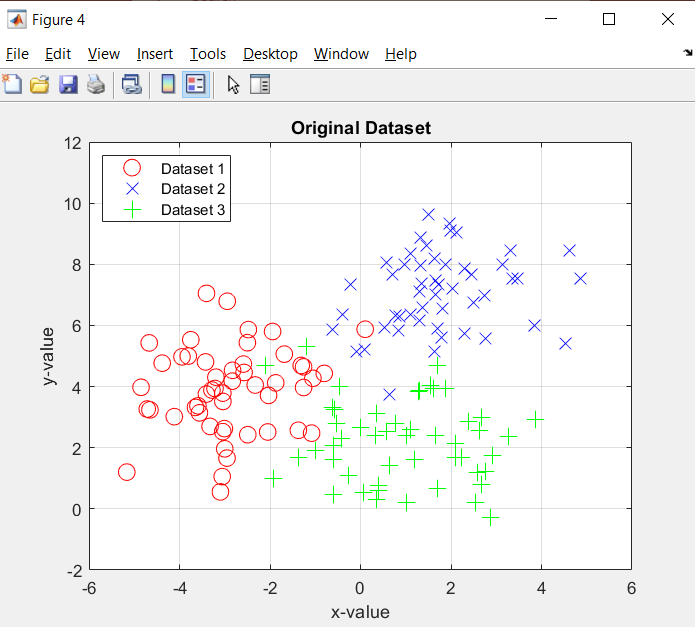
To demonstrate the final results of the Kmeans algorithm, it has been plotted onto a two dimensional plan with each cluster being a separate colour, alongside with the centroids being marked as black ‘**X**’s.

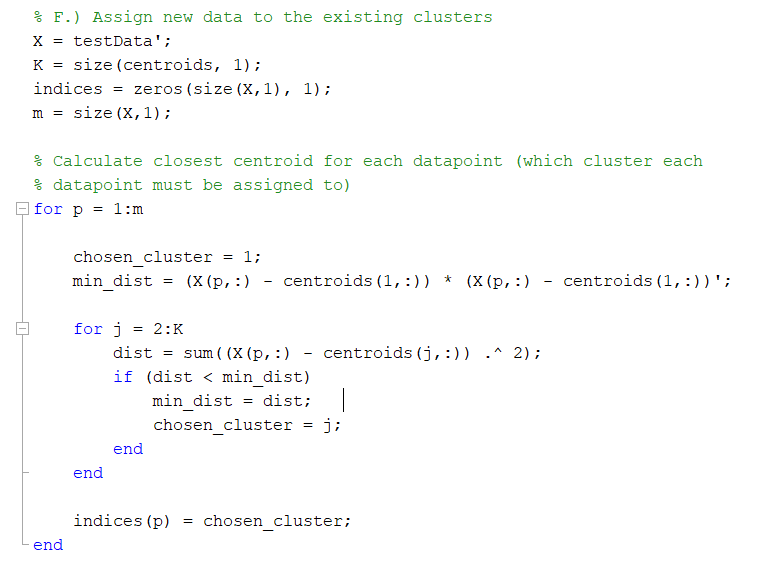


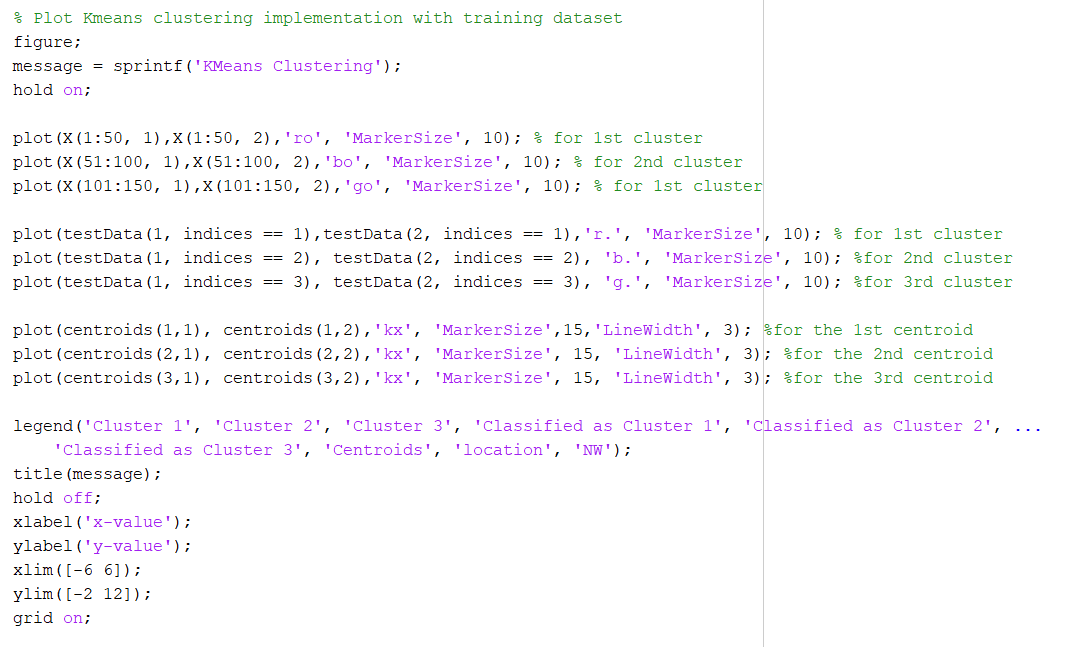


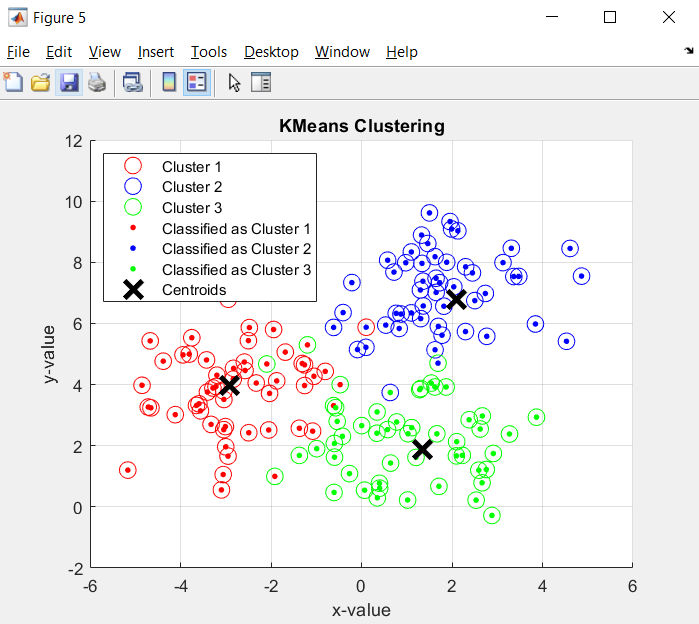
## Applying the Kmeans algorithm to a testing dataset

Once the Kmeans algorithm has calculated the clusters with the use of a training dataset, additional datasets can be applied to categorise the points provided with said data. To demonstrate this, a new dataset is generated that has the same gaussian distribution as a training dataset, however only consisting of three clusters of 50 values. Furthermore, the three data matrices are concatenated into a single merged dataset, which will be applied to the Kmeans algorithm. This is done in the same manner as when the training dataset was generated.



To test this new data, the algorithm needed to loop through all datapoints in the provided dataset and calculate the distance between said datapoint and all centroids; the datapoint is then assigned to a specific cluster, based on which centroid is closest. The “**indices**” variable is used to represent which cluster it has been assigned to. The test results are then plotted on a new figure showing what the datapoints have been categorised as, alongside with its original cluster.





# Conclusion

By analysing the results represented in “**figure 5**”, it is clear that some datapoints have been misclassified, with a total of seven datapoints being wrongly assigned to a neighbouring cluster. This is due to its value being significantly different from the centroid calculated for its original cluster. In addition, one major issue of the Kmeans algorithm is that when the data consists of many outliers, the calculated centroids can be inaccurate in comparison to the true centre of the cluster. This becomes particularly problematic when the data from neighbouring clusters have very similar values which therefore causes many overlaps in the different clusters. Furthermore, when the provided data consists of varying sizes or density, the algorithm finds it difficult to categorise such data.

Another issue that Kmeans contains, is that the user has to specify the value K manually, which can become especially troublesome when K is required to be a high value. This is because Kmeans is dependent on its initial values; when K is low, this problem can be mitigated by recalculating the centroids and clusters n number of times. However, as K increases, more advanced versions of Kmeans are required to properly calculate the centroids of each cluster. Nevertheless, the Kmeans algorithm is simple, easy to implement, and can be scaled to categorise large datasets.

# References

Unsupervised learning: <https://www.datarobot.com/wiki/unsupervised-machine-learning/>

Unsupervised learning: <https://www.guru99.com/unsupervised-machine-learning.html>

Kmeans (a lot of information): <https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/>

Spam filter (brief intro): <https://datafloq.com/read/7-innovative-uses-of-clustering-algorithms/6224>

Spam filter: <https://ijcert.org/ems/ijcert_papers/V3I1210.pdf>

Anomaly detection (contains important image): <https://www.datatechnotes.com/2020/05/introduction-to-anomaly-detection-methods.html>

Credit card fraud (primary source): <http://ijcsit.com/docs/Volume%206/vol6issue02/ijcsit20150602177.pdf>

Credit card fraud (secondary source): <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.680.1195&rep=rep1&type=pdf>

Kmeans: <https://uk.mathworks.com/help/stats/kmeans.html>

Kmeans without toolbox: <https://uk.mathworks.com/matlabcentral/fileexchange/74216-k-means-clustering-without-using-built-in-function>

Kmeans process explained: <https://towardsdatascience.com/k-means-clustering-explained-4528df86a120#:~:text=K%2Dmeans%20clustering%20aims%20to,methods%20to%20measure%20the%20distance>.

Centroid calculations: <https://www.datasciencecentral.com/profiles/blogs/steps-to-calculate-centroids-in-cluster-using-k-means-clustering>

Kmeans drawbacks: <https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages>