## *K-means algorithm*

The **K-Means algorithm** was utilised to gain deeper insights into the dataset that may not be accessible through supervised machine learning models.

### **Data Cleansing**

All selected features for the K-Means test required encoding before analysis. Once the encoding process was completed, six features out of a total of 38 were chosen for the experiments.

### **Exploration & Visualisation**

Based on the **silhouette score**, the clustering appeared to be effective. The optimal number of clusters was 3, more than 3 reduced the **silhouette score**. When visualised, the data formed **three distinct groups**, each with clear boundaries and well-defined centroids.

To further analyse the clusters, the outputs were exported to **excel** for examination.

### **Variance Analysis**

The **variance** for each feature was examined to determine its impact on the performance of the model. Features with the **highest variance** contributed the most to the effectiveness of the clustering process.

### **Summary**

This type of unsupervised model would be ideal if attempting to cluster customers based on features like purchase amount and frequency of purchases.

## *Parameters tuning* for *the different algorithms*

Attempted to improve the performance of the different models by fine-tuning the parameters of each one.

**Linear Discriminant Analysis**

* + Increase the number of splits – this is recommended for improving the performance of all the models. Not viable in this instance, there is a class with only 2 members so I can’t have more than 2 splits.
  + Standardise the features – prevents features with a large numerical range from overshadowing features with a small numerical range. Experiments with this did not deliver a material change
  + LDA Solver – tried both ‘lsqr’ & ‘eigen’ – it appears the sample size is large enough that this did not make a material difference.

### **Decision Tree Classifier**

* + Increase the number of splits – this is recommended for improving the performance of all the models. Not viable in this instance, there is a class with only 2 members so I can’t have more than 2 splits.
  + RandomForestClassifier – increases accuracy by increasing the weight of the most important features (in this case the top 5 features). Experiments with this did not deliver a material change
  + GridSearch to find the best combination of parameters provided for the model – no combination of decision tree parameters made a material difference to the accuracy of the model. The performance of the RandomForestClassifier is measurably better.

**XGBoost**

* + Attempted a Randomized Search instead of Grid Search to find the best combination of parameters provided for the XGBClassifier. There was a marginal improvement with one combination.
  + Standard Scaler – subtracts the mean from each feature normalising the data to improve model performance. Experiments with this did not deliver a material change.

**Logistic Regression**

* + Regularisation – used a low value to improve generalization. Experiments with this did not deliver a material change.
  + Solver – tried different solvers to see which had the most impact on the performance of the model.
    - ‘liblinear’ – Delivered a measurable improvement to the performance of the model.
    - ‘newton-cg’ – There was some improvement but the ‘liblinear’ performed better.
  + Penalty - only the ‘l2’ worked with the Logistic Regression Model used for this experiment. Experiments with this did not deliver a material change.
  + Standard Scaler – subtracts the mean from each feature normalising the data to improve model performance. Experiments with this did not deliver a material change.

**GaussianNB**

* + Attempted a combination of, ‘var\_smoothing-le-1’, and a Standard Scaler. There was negligible improvement in the performance of this model.
  + Tried the GaussianNB version of the RandomForestClassifier – ‘feature\_selection’. There was negligible improvement in the performance of this model.

**Summary**

The most noticeable improvement was with solvers on the Logistic Regression Model. Accuracy improved by 2% when the ‘liblinear’ value was used with the solver parameter.

The XGBClassifier was highly accurate before attempting to fine-tune the parameters. Randomised Search identified a combination of parameter values that improved performance of this model by 0.1%.

Fine-tuning parameters can improve performance; how much, depends on the model and the dataset.