A logo for college computing

Description automatically generated

**Assessment Cover Page**

|  |  |
| --- | --- |
| *Student Full Name: Cristhian Elson Pereira Macedo* |  |
| *Student Number: 2024104* |  |
| *Module Title: Machine Learning for Business* |  |
| *Assessment Title: CA1* |  |
| *Assessment Due Date: 27th October 2024 23:59* |  |
| *Date of Submission: 27th October 2024* |  |

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.



**Dry Bean Analyses**

**Cristhian Elson Pereira Macedo**

**2024104**

Higher Diploma in Science in Data Analytics for Business

Machine Learning for Business (MLBus)

Lecturer: Dr . Muhammad Iqbal

CCT College Dublin

Dublin, Ireland

2024



**SUMMARY**

[1. Introduction 4](#_Toc180964290)

[2. Project Objectives 4](#_Toc180964291)

[3. Data Understanding 4](#_Toc180964292)

[4. DATA PREPARATION 5](#_Toc180964293)

[5. MACHINE LEARNING - UNSUPERVISED LEARNING 6](#_Toc180964294)

[6. UNSUPERVISED LEARNING CONCLUSIONS 8](#_Toc180964295)

[7. MACHINE LEARNING – TIME SERIES FIRST STEPS 9](#_Toc180964296)

[8. MACHINE LEARNING – TIME SERIES ANALYSIS 10](#_Toc180964297)

[9. DATA DICTIONARY 13](#_Toc180964298)

[References 14](#_Toc180964299)

# **Introduction**

This project aims to complete the first Assessment Task for the module Machine Learning for Business from the course Higher Diploma in Science in Data Analytics for Business by CCT College Dublin, this is a document to describe the project itself, whose name is “CristhianMacedo\_MLBusinessHDip\_CA1.ipynb” and should use this document for more clarification. This Machine Learning project uses the programming language Python, the environment of Anaconda Navigator with Jupyter Notebook.

The area that has been chosen to be covered is "Agriculture, Fisheries and Forestry and Food", with a data set about Dry beans containing 13,611 images of grains of seven different dry beans from the catalogue of the UC Irvine Machine Learning Repository and for Time Series task has been chosen the (GC=F) database about the Gold during 5 years updated on Thu 24 Oct 2024 16:02 Dublin, Ireland time zone.

# **Project Objectives**

After a search of a few data sets about Food, it was found that data of Beans that showed interesting to be analysed, the Dry beans database was chosen specifically for the use of clustering using unsupervised Machine Learning, such as K-Means Clustering, Hierarchical Clustering, K-Medoids Clustering, DBSCAN, OPTICS Clustering and Fuzzy C-means Clustering, were all of them it will be carried out to answer the questions below:

* What are the similarities or dissimilarities between the clusters?
* Which clustering solution showed the best performance?
* Which clustering algorithm is easy to use?

The second step of this assessment task that will be carried out is about Time Series. The data that has been chosen is about Gold during the last five years, from 2019-10-24 to 2024-10-24, updated on Thu 24 Oct 2024 16:02 Dublin, Ireland time zone, and it will answer the question:

* How to analyse and work with Time Series?
* What would be a prediction of the value of Gold for the future?

# **Data Understanding**

Starting the Project importing Pandas library and a warning method to suppress warnings, loading the data frame whose name is “df”, checking the head of it, shape where it returns 13611 rows (observations) with 17 columns (features) within names are: Area, Perimeter, MajorAxisLength, MinorAxisLength, AspectRation, Eccentricity, ConvexArea, EquivDiameter, Extent, Solidity, roundness, Compactness, ShapeFactor1, ShapeFactor2, ShapeFactor3, ShapeFactor4 and Class.

Next checking the info of the data, a quick view in “describe” method to check some statistic information, analysing null values, where in this case there are none to be dealt, it was analysed duplicates rows and returned 68 observations duplicates, tried to check them in a head table, but where difficult to be sure if they were, in fact, duplicates observations, checked the data type of features and understand the unique values of the Class feature.

# **DATA PREPARATION**

After completing the data understanding phase, it was started an exploratory data analysis step even though the data set was already ready to use, where were checked the distribution of each numerical feature using a "distplot" and "kdeplot" returning their respective distribution and density, also how skewed these features are to the left or right, or in case of a normal distribution, and points of mean and median in the distribution also with their Standard Deviation value, created also a “hist plot” and “boxplot” to visualise in another perspective.

For the categorical feature, in this case, only one called “class”, it was used a “countplot” to visualise their distribution in a bar plot, next was plotted a bar plot from “missingno” library to visualise the missing values in the features, in this case, the bars are completed, once there are no missing values.

Next, it was encoded the class feature using the method “LabelEncoder”, and opted to use Label Encoder once "Label encoding is suitable for categorical features with only two distinct categories. In this technique, each category is assigned a unique integer label. Categories are assigned integer values starting from 0." (Baruah, 2023). Pros: Works well for features with two categories and Cons: Machine learning algorithms may misinterpret the integer labels as having mathematical significance (Baruah, 2023).

Created a copy of "df" data frame in a new variable called "df\_outliers" to test and check outliers inside the features, created a few visualisations of the distribution of the outliers for each numerical feature using a "distplot", "violinplot" and "stripplot" also how skewed these features are, dealt with outliers using IQR method, getting the shape of before and after with 3017 observations of difference and their respective distribution in a boxplot, but left them once could be no errors, but just a piece of usual information without frequency.

Also, created an example of feature scaling testing Min Max Scaler, Standard Scaler, Normalisation and Robust Scaler, and their respective results in box plots, opted to save the results of RobustScaler in case of need in a new data frame called "df\_scaled" where this method is more robust with outliers.

# **MACHINE LEARNING - UNSUPERVISED LEARNING**

After a few analyses of the data set, tests will be carried out using K-Means Clustering, Hierarchical Clustering, K-Medoids Clustering, DBSCAN, OPTICS Clustering and Fuzzy C-means Clustering to check which clustering solution performs the best.

For all clustering tests is going to be used the values of the features or the base feature of Area, Perimeter, MajorAxisLength, MinorAxisLength, AspectRation, Eccentricity, ConvexArea, EquivDiameter, Extent, Solidity, roundness, Compactness, ShapeFactor1, ShapeFactor2, ShapeFactor3 and ShapeFactor4.

Also created two variables called "feature1\_index and feature2\_index" with constant numbers to use easily in the graphics and so on, when necessary. For the tests, it set a constant number of 2 to "feature1\_index" and 3 to "feature2\_index", to analyse columns 2 “MajorAxisLength” and 3 “MinorAxisLength” of the data frame.

**K-Means Clustering**, split the data in the “X” variable, getting just the values of the features, next created a cluster to train, test, split and predict the values, tested first with four clusters and plotted in a “Scatter Plot”, the algorithm worked pretty well, to create four clusters and show the centroids, next checked it with the elbow method, and seem that should be fewer clusters than 4, so created a new elbow method that shows in a red circle the exact number of clusters that should be used.

Using the Silhouette score, getting a total of 0.54 with four clusters and 0.66 with three clusters, plotted the results in the graphic with three clusters and also the result worked well next it was created a few functions were used to plot the tests, but this time, using Silhouette score and Davies-Bouldin index, difference from elbow test where the number of clusters should be three, the Silhouette and Davies-Bouldin returned as result the number of two clusters only, plotted a dendrogram to visualise it, next created a few of functions to show the tests side by side and the resulted graphic with two clusters, and the result seem interesting too.

**Hierarchical Clustering**, split the data in the “X” variable, getting just the values of the features next created a cluster to train, test, split and predict the values, tested first with four clusters and plotted in a “Scatter Plot”.

The algorithm worked pretty well, similar as before using K-Means, using the Silhouette score, getting a total of 0.56 with four clusters and 0.61 with three clusters, plotted the results in the graphic with three clusters and also the result worked well, next using a few functions to show side by side the tests and the resulted graphic with two clusters where using Silhouette score and Davies-Bouldin index returned the number of clusters that should be used as two, and also the result seems interesting next plotted a dendrogram with the optimal number of clusters as two.

**K-Medoids Clustering**, different of others clusters split the data in the “X” variable of all features and “y” to the target class feature, also getting just the values of the features next created a cluster to train, test, split and predict the values, tested first with four clusters and plotted in a “Scatter Plot”.

The prediction of the algorithm worked pretty well, before it was considering seven clusters and as predicted returned four in total, using the Silhouette score, getting a total of 0.41 with four clusters and 0.44 with three clusters, plotted the results in the graphic with three clusters and also the result worked well, similar to the **K-Means** and **Hierarchical Clustering**.

Next using a few functions to show side by side the tests and the resulting graphic with two clusters where using the **Silhouette score** returned two clusters and the **Davies-Bouldin** index returned five clusters that should be used. The result also seems interesting, next plotted a dendrogram with the optimal number of clusters as five in total and plotted a “Scatter Plot” to check it in five clusters with their respective centroids.

**DBSCAN**, split the data in the “X” variable, getting at this time the feature information not as before just the values of the feature using **K-Means Clustering, Hierarchical Clustering** and **K-Medoids Clustering**, next created a cluster to train, test, split and predict the values, tested first with a value of 30 eps and 1 as min samples, and plotted in a “Scatter Plot”. The result as far to be good as the previous clustering methods, the algorithm worked pretty well, next it was used “NearestNeighbors” method to try to improve the results of DBSCAN.

After creating a cluster to train, test, split and predict the values, it created a graphic Plotting the K-distance Graph with their respective distances, next it gets a mean of 0.739, next created again a cluster to train, test, split and predict the values, but now with a value of 0.74 eps and 1 as min samples, and the results seem similar as before, not good.

Also tested with different values such as, eps 1, 2, 3, 5 10, 20, etc., and min samples 1, 2, 3, 4, 5, 6 and so on., but unfortunately none of these tests performed well.

After tests with Silhouette score and Davies-Bouldin index, got as results 0.18 Silhouette and 2.59 Davies Bouldin, next created a classifier cluster: "dbscan" applying "DBSCAN" with the parameters: "eps = 0.18, min\_samples = 1" in a new feature called "DBSCAN\_silhouette\_labels" and plotted it, also created a new test with the parameters: "eps = 2.59, min\_samples = 1". in a new feature called "DBSCAN\_davies\_bouldin\_labels" and plotted it, and none of these tests performed well.

**OPTICS Clustering**, split the data in the “X” variable, getting at this time the feature information similar in **DBSCAN**, not as before just the values of the feature using **K-Means Clustering, Hierarchical Clustering** and **K-Medoids Clustering,** next created a cluster to train, test, split and predict the values, tested first with the parameters: “min\_samples = 3, xi = 0.02, min\_cluster\_size = 0.03” and plotted in a “Scatter Plot”. The algorithm worked a little well, better than **DBSCAN**, next created a test with the parameters “eps equal 0.2 and 0.3” and plotted it in a graphic with the Reachability Plot, OPTICS Clustering, DBSCAN clustering with eps = 0.2 and DBSCAN Clustering with eps = 0.3 to comparison.

**Fuzzy C-Means Clustering**, split the data in the “X” variable, getting just the values of the features similar to **K-Means Clustering, Hierarchical Clustering** and **K-Medoids Clustering**, next created a cluster to train, test, split and predict the values, tested first with four clusters and plotted in a “Scatter Plot”, for comparison the actual data points and the predicted.

The algorithm worked pretty well, similar the previous methods using K-Means, Hierarchical Clustering and K-Medoids Clustering, using the Silhouette score, getting a total of 0.56 with four clusters and 0.66 with three clusters, plotted the results in the graphic with three clusters and also the result worked well, with their respective centroids.

Next using a few functions to show side by side the tests and the resulted graphic with three clusters where using Silhouette score and Davies-Bouldin index returned the number of clusters that should be used as three, and also the result seems interesting next plotted a dendrogram with the optimal number of clusters as two.

Next create a comparison with 3 clusters between "K-Means Clustering", "Hierarchical Clustering", "K-Medoids Clustering", "DBSCAN", "OPTICS Clustering" and "Fuzzy C-means Clustering", created the clusters “kmeans, hierarchicalClustering, kMedoids, dbscan, optics and fuzzy” and the predictions “y\_kmeans, y\_hc, y\_kmc, y\_dbscan, y\_optics and fcm\_labels” using the features MajorAxisLength, MinorAxisLength

And plotted the results in graphics side by side, also calculated the scores of Silhouette Scores and Davies Bouldin Score for each cluster.

Nex created a comparison similar to before but this time with the features “Area and Perimeter” and with “ShapeFactor1 and ShapeFactor2” features.

# **UNSUPERVISED LEARNING CONCLUSIONS**

1. What are the similarities or dissimilarities between the clusters?

The clusters that have a parameter of “n\_clusters” number of clusters seem works better than the other ones, it is the case of K-Means Clustering, Hierarchical Clustering, K-Medoids Clustering and Fuzzy C-Means Clustering, all of them use the values of the features to work, that is not the case of DBSCAN and OPTICS Clustering that does not have a specific parameter for number of clusters, gets the all feature as information and works differently.

The clustering methods K-Means Clustering, Hierarchical Clustering, K-Medoids Clustering and Fuzzy C-Means Clustering work similarly and have better results.

1. Which clustering solution showed the best performance?

Comparison of the results of the Elbow method, the Silhouette score and the Davies-Bouldin index:

* K-Means Clustering: 4 clusters work well, Elbow method 3 clusters work well, Silhouette 2 clusters and Davies-Bouldin 2 clusters.
* Hierarchical Clustering: 4 clusters work well, 3 clusters work well, Silhouette 2 clusters and Davies-Bouldin 2 clusters.
* K-Medoids Clustering: 5 as predicted seems ok, 3 clusters work well, Silhouette 2 clusters work well and Davies-Bouldin 5 clusters work better than 2 clusters.
* DBSCAN: Difficult to find the best parameter and to work decently.
* OPTICS Clustering: 2 Clusters almost get a good result, also complicate to find the best parameters to use in it.
* Fuzzy C-Means Clustering: 4 clusters work well, 3 clusters work well, Silhouette 3 clusters work well and Davies-Bouldin 3.

Checking the results and the graphics, it seems that the K-Means Clustering, Hierarchical Clustering and Fuzzy C-Means Clustering showed the best performance.

1. Which clustering algorithm is easy to use?

After analyses, it would recommend the use of K-Means Clustering or Fuzzy C-Means Clustering, but Hierarchical Clustering and K-Medoids Clustering are good choices as well and easy to deal it.

Therefore, for segmentation, it would consider to use of K-Means Clustering or Fuzzy C-Means Clustering for simplicity.

# **MACHINE LEARNING – TIME SERIES FIRST STEPS**

Before working in Time Series, it was necessary to study a little more about how Time Series works and how to perform it, first, it is important to analyse the data set, to understand and continue working properly with it, been essential to understand a few properties or components, such as, "Trend, Seasonality, Noise or Irregularity and Cyclicity" and types of series "forecast or residuals".

* "Trend" - A long-term increase (upward) or decrease (downward).
* "Seasonality" - When patterns or fixed frequency, seasonal factors and variances, can be derived from an autocorrelation plot if it has a sinusoidal shape.
* "Noise, Variation, Anomalies or Irregularity" - Random noises.
* "Cyclicity" - Rises and falls that repeat frequently in a long period.

Next, check if the data is "Stationarity" or not, this is important before applying any model to get forecasting accurately; the series must be stationarity, and this normally is checked when the statistical properties do not change over time, should have constant "mean", "variance or standard deviation" and "covariance".

To check, whether the time series is stationary or not, we can use the "Rolling Statistics" visually or a statistical "Dickey-Fuller" test.

* Rolling Statistics: Plot a moving average or moving standard deviation to see visually if it varies with time.
* Dickey-Fuller or ADCF Test
  + H0 The Null hypothesis - if then p > 0, and the process is not stationary, checking Test Statistics and some critical values for confidence levels.
  + H1 Hypothesis - Otherwise, p = 0, or less than the critical values, the null hypothesis is rejected, and the process is considered to be stationary.

After understand initial instructions, it were imported the time series data frame named as “gold”, checked the head of it, null values, the info of it, data types, converted the “Date” feature to Data Time, next turned the “Date” feature as index and sorted the dates, create a filter to check one month of information, from 2024-01-01 to 2024-01-31, and seems that the data is daily in business days only.

Next created a copy of it in the variable called “gold\_outliers” and plotted a box plot of the “Close” feature, the feature that is going to be used in the time series analyses, next used the IQR technique to remove outliers, and check the box plot once again, next plotter a line plot to compare the before and after removing outliers.

# **MACHINE LEARNING – TIME SERIES ANALYSIS**

After a few tests to understand the data frame it where started once again to import the time series data frame and keep just the important information to be easy to use, test and plot in graphics, it started the analysis by creating a new Data Frame named "df" after loading the data set "GC=F\_historical\_Gold.csv", converting the column "Date" to DateTime, next setting it as feature index in a new data frame called "gold" and drop the other columns to make easy to plot the graphics where the x-axis it will be the "Date" feature and the y-axis the "Close" feature, and getting a "Head" of it.

Next, left a command comment to remove outliers in case of need, at this time, it will follow without removing it, next plot a “Line Plot” to check the Closing values of Gold for the last 5 years, next plotting a decompose method of seasonal to check “Trend, Seasonality and Residuals”.

Until this moment the code it was not working properly returning an error: “ValueError: You must specify a period or x must be a pandas object with a PeriodIndex or a DatetimeIndex with a freq not set to None”, after seach in google found a resolution, the problem occurs because the dates there are no associated frequencies like daily, weekly, monthly, and so on, but we can resolve it by adding a parameter of time called "period", learned after reading a forum question on stack overflow "decompose() for time series: ValueError: You must specify a period or x must be a pandas object with a DatetimeIndex with a freq not set to None" (and\_and, 2020).

Next it were create a function to plot the Time Series Plotline, Autocorrelation (ACF) and Partial Autocorrelation (PACF). At this moment is possible to assume:

* Trend: analysing the graphics, it is possible to see visually that there exists a trend, a long-term increase in the data.
* Seasonality: analysing the graphics, it is possible to see visually that there exists no Seasonality with cyclical patterns of fixed frequency.
* Noise, Variation, Anomalies or Irregularity, There exists.
* Cyclicity: There are rises and falls that occur but do not repeat frequently over a long period.

Next, it will be whether it is necessary to change the time series data to stationarity, using the method "rolling" to observe the moving average, considering 12 consecutive data points or months at a time, giving a yearly level for rolling "mean, var and std", it will be use the Dickey-Fuller test, first plotting rolling statistics with "Mean and Standard Deviation". It is possible to visualise that the rolling mean has a trend over time even though the rolling standard deviation is basically constant with time, then we deduce that it is necessary to make a time series stationary, keeping the mean and standard deviation invariant or constant with time.

Perform the Augmented Dickey-Fuller test to confirm if the time series is not stationary, after Stationarity Rolling Statistics and Stationarity Dickey-Fuller analysis, it was noted that the time series is not stationary, once "For a Time series to be stationary, its ADCF test should have:"

* "p-value to be low (according to the null hypothesis)".
* "The critical values at 1%, 5%, 10% confidence intervals should be as close as possible to the Test Statistics." (Arindam Chatterjee, 2018)

In the results, the p-value is higher than 0, in this case, 0.97, and the critical values "-3.43, -2.86, -2.56" are far from the Test Statistic value 0.31, at the moment, we can consider that the time series is not stationary, so we fail to reject the H0 null hypothesis.

Next, testing a few data transformations to achieve a stationary data series, first Log Scale, applying this method to get rid of time-varying variance, in this case, using a log transformation to punish larger values more than smaller values, before 1400 to 2800 and now 7.3 to 7.9, plotter a Time-Varying Variance using a log transformation and next a Time-Varying Variance using a log transformation with movingAverage in red.

Analysing the previous graphic, "we see that even though rolling mean is not stationary, it is still better than the previous case, where no transformation were applied to series." (Arindam Chatterjee, 2018). Also, it is noted that both case has a trend component, so it is possible to remove the trend component of both scenarios.

Next, subtracting the rolling mean from the original series and plotting it and detrending after taking the log to compare, next doing the same, but now, dropping the null values and creating a function to plot the results, first, subtract this rolling mean from the original series, drop null values.

Next, creates a function called "test\_stationarity" to Plotting rolling statistics with "Mean and Standard Deviation" and Perform the Augmented Dickey-Fuller test, the example below learned on the "Kaggle" website, "Time Series For beginners with ARIMA" (Arindam Chatterjee, 2018).

Next it were tried to turn the time series stationary using, Log Transformation, Decay Transformation, Time Shift Transformation but even though it tried the three different transformations, log, exp decay and time shift, the results were not very good, it will keep the log results once is possible to revert to the original scale during forecasting in case needed, next plotted the ACF and PACF to check the presence of seasonality and the q (MA) parameter number and PACF to identify the p (AR) parameter number.

Analysing the Autocorrelation Function (ACF) graphic, the curve touches the y = 0.0 line at x = 1.25. Thus, from theory, Q = 2 From the Partial Autocorrelation Function (PACF) graphic, the curve touches the y = 0.0 line at x = 1.5. Thus, from theory, P = 5. Once checked and confirmed that the time series data is non-stationary, instead of the "ARMA" model, it will be tested the "ARIMA" model for non-stationary cases.

Tried this time with the method “diff” and dropping the first observation as null, and checking line plots and ACF and PACF graphics with the Dickey-Fuller test, finally, the time series get stationary, next applied ARIMA and the results seemed really well, next tried with SARIMA with predictions of the values (p, d, q) getting the best model and plotted it.

After analyses with Time Series, applying ARIMA and SARIMA, the objective of "How to analyse and work with Time Series" was completed, also for the question "What would be a prediction of the value of Gold for the future?" the results seem to be very favourable for growth, there will be some drops, but the forecast is that it will continue to rise over time.

All the work was done based on studies in the classroom, reviewing examples, work and materials from classes at home, and also some research on the internet and the work found on the "Kaggle" website served as a great help in learning and being able to continue with the Time Series theme, article: "Time Series For beginners with ARIMA" (Arindam Chatterjee, 2018).

In addition to the assessment task, a free AI Writing Assistance Grammarly (Grammarly, 2009) is being used to help with English grammar while typing to check spelling. The project is available on GitHub class and the complete URL is available on References (Macedo, 2024).

# **DATA DICTIONARY**

1. Area (A): The area of a bean zone and the number of pixels within its boundaries.

2. Perimeter (P): Bean circumference is defined as the length of its border.

3. Major axis length (L): The distance between the ends of the longest line that can be drawn from a bean.

4. Minor axis length (l): The longest line that can be drawn from the bean while standing perpendicular to the main axis.

5. Aspect ratio (K): Defines the relationship between L and l.

6. Eccentricity (Ec): Eccentricity of the ellipse having the same moments as the region.

7. Convex area (C): Number of pixels in the smallest convex polygon that can contain the area of a bean seed.

8. Equivalent diameter (Ed): The diameter of a circle having the same area as a bean seed area.

9. Extent (Ex): The ratio of the pixels in the bounding box to the bean area.

10. Solidity (S): Also known as convexity. The ratio of the pixels in the convex shell to those found in beans.

11. Roundness (R): Calculated with the following formula: (4piA)/(P^2)

12. Compactness (CO): Measures the roundness of an object: Ed/L

13. ShapeFactor1 (SF1)

14. ShapeFactor2 (SF2)

15. ShapeFactor3 (SF3)

16. ShapeFactor4 (SF4)

17. Class (Seker, Barbunya, Bombay, Cali, Dermosan, Horoz and Sira)

\*\*Class Labels\*\*

Seker, Barbunya, Bombay, Cali, Dermosan, Horoz, and Sira

# **References**

and\_and (2020). decompose() for time series: ValueError: You must specify a period or x must be a pandas object with a DatetimeIndex with a freq not set to None. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/60017052/decompose-for-time-series-valueerror-you-must-specify-a-period-or-x-must-be [Accessed 2024].

Arindam Chatterjee (2018). Time Series For beginners with ARIMA. [online] kaggle.com. Available at: https://www.kaggle.com/code/freespirit08/time-series-for-beginners-with-arima [Accessed 25 Oct. 2024].

Baruah, I.D. (2023). All you need to know about encoding techniques! [online] ANOLYTICS. Available at: https://medium.com/anolytics/all-you-need-to-know-about-encoding-techniques-b3a0af68338b [Accessed 22 Oct. 2024].

Grammarly (2009). Grammarly. [online] Grammarly.com. Available at: https://app.grammarly.com/ [Accessed 2024].

Macedo, C. (2024). Machine Learning for Business-ca1 Ft CristhianMacedo2024104. [online] GitHub. Available at: https://github.com/CCT-College-Dublin/ca1-50-CristhianMacedo2024104 [Accessed 2024].

scikit-learn developers (2023). sklearn.cluster.AgglomerativeClustering. [online] scikit-learn. Available at: https://scikit-learn.org/1.3/modules/generated/sklearn.cluster.AgglomerativeClustering.html [Accessed 2024].

scikit-learn developers (2024). sklearn.cluster.AgglomerativeClustering. [online] scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html [Accessed 2024].

The Pandas Development Team (2024). pandas.to\_datetime — pandas 1.3.4 documentation. [online] pandas.pydata.org. Available at: https://pandas.pydata.org/docs/reference/api/pandas.to\_datetime.html [Accessed 24 Oct. 2024].

UC Irvine Machine Learning Repository (2020). UCI Machine Learning Repository. [online] archive.ics.uci.edu. Available at: https://archive.ics.uci.edu/dataset/602/dry+bean+dataset [Accessed 2024].

yahoo finance (2024). Gold Aug 24 (GC=F) Stock Price, News, Quote & History - Yahoo Finance. [online] Yahoo.com. Available at: https://finance.yahoo.com/quote/GC%3DF/ [Accessed 2024].