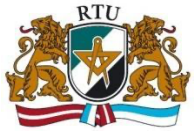

RIGA TECHNICAL UNIVERSITY

Faculty of Science and Information Technology Faculty

Institute of Applied Computer Systems

Fundamentals of Artificial Intelligence

Practical Assignment 2

Applying Methods of Machine Learning

Download Link for the Assignment Project

https://github.com/EdipCM/EdipCM_AI_PW2

Dataset sources: https://www.kaggle.com/datasets/dongeorge/seed-from-uci based on the dataset from https://archive.ics.uci.edu/ml/datasets/seeds#

Edip Can Mucek

201ADB014

Part 1 – Pre-processing/Exploring the Data

To complete this part of the assignment, you will need to take the following steps:

1. Selecting and describing the dataset based on the information given in the repository/database where the dataset was located.
2. If the dataset you have acquired from the repository is not in a format that is easy to work with (like a comma- separated-values, or .csv, file), convert it into the needed format. Your dataset file should consist of an $n \times d$ table, where d is the number of dimensions of the data and n is the number of data objects. Your columns should be arranged in the following way: data object ID, the class label of the data object, and then the collected feature values.
3. If the values of any feature are textual values (e.g. yes/no, positive/neutral/negative, etc.), they must be transformed into numerical values.
4. If some data objects are missing values of features, it is necessary to find a way to obtain them by studying additional sources of information.
5. Representing your training dataset visually and statistically:
   a) you must create at least two 2- or 3-dimensional scatter plots illustrating the separability of classes in your dataset based on different features; the student should avoid using the data object ID as a variable in the scatterplot;
   b) you must create at least 2 histograms showing the separation of classes for the features of interest;
   c) you must show 2 distributions for the features of interest;
   d) you must calculate statistics on your data (at least the central tendency and the dispersion of the feature values).

Include the following information in the report:

- description of the dataset (providing references to the sources of information used):
  - title, source, author and/or owner of the dataset;
  - description of the problem domain of the dataset;
  - licensing regarding the dataset (if any);
  - a way how the dataset was collected;
- description of the content of the dataset (providing references to the sources of information used):
  - the number of data objects in the dataset;
  - the number of classes in the dataset, the meaning of each class and the way of representing classes (explanation of the labels assigned to classes); if the data set provides several possible data classifications, then the report must clearly identify which classification is considered in the assignment;
  - the number of data objects belonging to each class;

    - the number and meaning of features in the dataset, as well as their value types and ranges (this information should be presented in a table consisting of the feature representation, its meaning, valuetype and range of values available in the dataset);
    - a snippet of the structure of your datafile in which the columns of your datafile and class labels are shown together with some data objects;
- conclusions coming from the analysis of scatter plots, histograms and distributions (from Step 5 in Part I) about the separability of your classes (remember to include your graphs in the report). Try to answer the following questions:
    - Whether classes in your dataset are balanced, or is one class (several classes) prevailing? It is determinedby how many data objects belong to each class.
    - Does the visual representation of the data allow the structure of the data to be seen? It is a question of whether data objects belonging to different classes are clearly separable.
    - How many data groupings can be identified by studying the visual representation of the data? It is a question of whether there are any separable groupings of data if the data objects of different classes merge
    - Are the identified data groupings close to each other or far from each other?
- conclusions coming from the analysis of statistical calculations (central tendency and dispersion).

Part II – Unsupervised Learning

For this part of the assignment, you will be running unsupervised clustering on your dataset. Part I gave you an understanding of what features and classes you have and how well you can separate data objects into classes. This part of the assignment aims to look at the data in an unsupervised fashion to see if the assumptions about class structure hold.

To complete this part of the assignment, you will need to take the following steps:

1. Apply two methods of unsupervised learning considered in class: (1) Hierarchical clustering and (2) K-Means.
2. Perform at least 3 experiments with Hierarchical clustering, freely changing the values of hyperparameters, and analysing the operation of the algorithm;
3. Perform experiments with the K-means algorithm using at least five different k values, calculate the Silhouette Score, and analyse the performance of the algorithm.

Include the following information in the report:

- Description of the hyperparameters available in the Orange tool and their meaning for each algorithm.
- Description of the experiments performed, clearly indicating the hyperparameter values used, and conclusions about the operation of each algorithm.

- Based on the analysis of the operation of both algorithms, conclusions made about whether the classes in the dataset are well or poorly separable.

Part III – Supervised Learning

For this part of the assignment, you will be running at least 3 classification algorithms on the data you collected and analysed in Part I and Part II of this assignment. One of the algorithms which you are obliged to use is artificial neural networks (ANN). Two other algorithms you can choose on your own.

To complete this part of the assignment, you will need to take the following steps:

1. Choose at least two supervised learning methods that are suitable for classification task. You can use the methodsconsidered in class and any other of the algorithms available in the Orange tool for classification task.
2. Divide your dataset in training and test sets.
3. For each algorithm, perform at least 3 experiments using the training dataset, changing the values of the algorithm hyperparameters and analysing the algorithm performance metrics.
4. For each algorithm, choose the trained model that provides the best algorithm performance.
5. Apply the trained model of each algorithm to the test dataset.
6. Evaluate and compare the performance of the trained models.

Include the following information in the report:

- Short description (1/3 of A4) of the essence of the supervised learning algorithms you have used and motivationfor choosing two of them (excluding the artificial neural network).
- Description of the hyperparameters available in the Orange tool and their meaning for each algorithm.

- Information on test and training datasets:
    - the total number of data objects added to the test and training datasets (by number and %);
    - information on how many data objects from each class are included in your training and test sets (by number and %);
- Using a table, represent the hyperparameter values used in the experiments for each algorithm.
- Conclusions on the performance of the models in the performed experiments, clearly identifying the model thatwill be used for testing.
- Test results of trained models and comparison and interpretation of their performance.

Introduction

This report is intended to outline the process of applying methods of machine learning using a chosen dataset. The title of the dataset used in the making of this report is 'Seeds Dataset'. The dataset can be found on UC Irvine Machine Learning Repository as well as Kaggle Datasets with the links below;

https://archive.ics.uci.edu/ml/datasets/seeds#

https://www.kaggle.com/datasets/dongeorge/seed-from-uci

The dataset was downloaded from Kaggle.

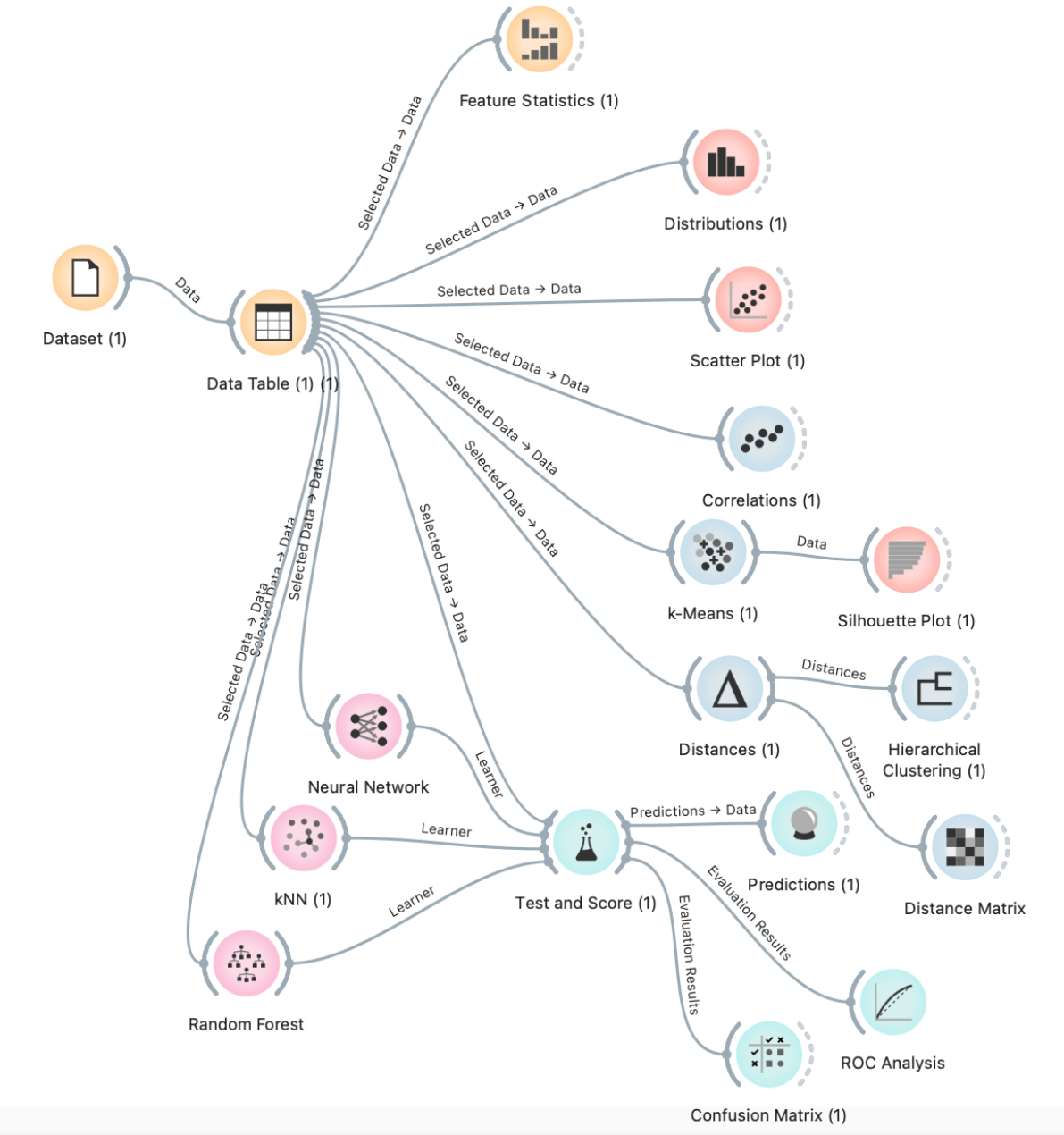

Figure 1: The complete workflow on Orange Tool

Part 1 – Pre-processing/Exploring the Data

The dataset file used in this report was retrieved as a comma-separated values format file on 11.05.2022 from the Kaggle Datasets. The dataset is originally created by M. Charytanowicz and J. Niewczas of the John Paul II Catholic University of Lublin in Poland. According to its creators, the dataset was collected using a soft x-ray technique and grains package and was donated to the UC Irvine Machine Learning Repository on 29.09.2012. The dataset is a collection of geometrical measurements of three different types of wheat kernels randomly picked. The dataset has three classes that determine if a wheat kernel belongs to the class Kama, Rosa or Canadian, based on 7 attributes. In addition to having 7 attributes, the dataset contains 210 data objects, 70 in each class. Below, an example of the first 30 elements of the dataset can be observed before any modifications were made to the dataset for ease of processing.[1]

A	P	C	LK	WK	A_Coef	LKG	target
15.26	14.84	0.871	5.763	3.312	2.221	5.22	0
14.88	14.57	0.8811	5.554	3.333	1.018	4.956	0
14.29	14.09	0.905	5.291	3.337	2.699	4.825	0
13.84	13.94	0.8955	5.324	3.379	2.259	4.805	0
16.14	14.99	0.9034	5.658	3.562	1.355	5.175	0
14.38	14.21	0.8951	5.386	3.312	2.462	4.956	0
14.69	14.49	0.8799	5.563	3.259	3.586	5.219	0
14.11	14.1	0.8911	5.42	3.302	2.7	5	0
16.63	15.46	0.8747	6.053	3.465	2.04	5.877	0
16.44	15.25	0.888	5.884	3.505	1.969	5.533	0
15.26	14.85	0.8696	5.714	3.242	4.543	5.314	0
14.03	14.16	0.8796	5.438	3.201	1.717	5.001	0
13.89	14.02	0.888	5.439	3.199	3.986	4.738	0
13.78	14.06	0.8759	5.479	3.156	3.136	4.872	0
13.74	14.05	0.8744	5.482	3.114	2.932	4.825	0
14.59	14.28	0.8993	5.351	3.333	4.185	4.781	0
13.99	13.83	0.9183	5.119	3.383	5.234	4.781	0
15.69	14.75	0.9058	5.527	3.514	1.599	5.046	0
14.7	14.21	0.9153	5.205	3.466	1.767	4.649	0
12.72	13.57	0.8686	5.226	3.049	4.102	4.914	0
14.16	14.4	0.8584	5.658	3.129	3.072	5.176	0
14.11	14.26	0.8722	5.52	3.168	2.688	5.219	0
15.88	14.9	0.8988	5.618	3.507	0.7651	5.091	0
12.08	13.23	0.8664	5.099	2.936	1.415	4.961	0
15.01	14.76	0.8657	5.789	3.245	1.791	5.001	0
16.19	15.16	0.8849	5.833	3.421	0.903	5.307	0
13.02	13.76	0.8641	5.395	3.026	3.373	4.825	0
12.74	13.67	0.8564	5.395	2.956	2.504	4.869	0
14.11	14.18	0.882	5.541	3.221	2.754	5.038	0

Figure 2: First 30 elements found in the dataset prior to processing

[1] M. Charytanowicz, J. Niewczas, 'seeds Data Set', UCI Machine Learning Repository, 29.09.2012, https://archive.ics.uci.edu/ml/datasets/seeds

As it can be observed in Figure 1, the attributes and the classes can be found. However, the dataset requires re-ordering as well as the addition of ID numbers for each data object of wheat example.

ID	target	A	P	C	LK	WK	A_Coef	LKG
1	0	15.26	14.84	0.871	5.763	3.312	2.221	5.22
2	0	14.88	14.57	0.881	5.554	3.333	1.018	4.956
3	0	14.29	14.09	0.905	5.291	3.337	2.699	4.825
4	0	13.84	13.94	0.895	5.324	3.379	2.259	4.805
5	0	16.14	14.99	0.903	5.658	3.562	1.355	5.175
6	0	14.38	14.21	0.895	5.386	3.312	2.462	4.956
7	0	14.69	14.49	0.879	5.563	3.259	3.586	5.219
8	0	14.11	14.1	0.891	5.42	3.302	2.7	5
9	0	16.63	15.46	0.874	6.053	3.465	2.04	5.877
10	0	16.44	15.25	0.888	5.884	3.505	1.969	5.533
11	0	15.26	14.85	0.869	5.714	3.242	4.543	5.314
12	0	14.03	14.16	0.879	5.438	3.201	1.717	5.001
13	0	13.89	14.02	0.888	5.439	3.199	3.986	4.738
14	0	13.78	14.06	0.875	5.479	3.156	3.136	4.872
15	0	13.74	14.05	0.874	5.482	3.114	2.932	4.825
16	0	14.59	14.28	0.899	5.351	3.333	4.185	4.781
17	0	13.99	13.83	0.918	5.119	3.383	5.234	4.781
18	0	15.69	14.75	0.905	5.527	3.514	1.599	5.046
19	0	14.7	14.21	0.915	5.205	3.466	1.767	4.649
20	0	12.72	13.57	0.868	5.226	3.049	4.102	4.914
21	0	14.16	14.4	0.858	5.658	3.129	3.072	5.176
22	0	14.11	14.26	0.872	5.52	3.168	2.688	5.219
23	0	15.88	14.9	0.898	5.618	3.507	0.7651	5.091
24	0	12.08	13.23	0.866	5.099	2.936	1.415	4.961
25	0	15.01	14.76	0.865	5.789	3.245	1.791	5.001
26	0	16.19	15.16	0.884	5.833	3.421	0.903	5.307
27	0	13.02	13.76	0.864	5.395	3.026	3.373	4.825
28	0	12.74	13.67	0.856	5.395	2.956	2.504	4.869
29	0	14.11	14.18	0.882	5.541	3.221	2.754	5.038
30	0	13.45	14.02	0.860	5.516	3.065	3.531	5.097

Figure 3: First 30 elements of the dataset after modifications

As it can be seen above, Figure 2 shows the first 30 data objects of the dataset in an ordered manner that will be used for the implementation of machine learning methods. The processed dataset includes unique ID numbers for each data object. Therefore, the number of attributes in the dataset is 8 in addition to the target classes. Furthermore, the target column which represents the classes is moved up to the second column. As the dataset only includes numerical values, there were not any conversions necessary. In the target class, the values ‘0, 1, 2’ refer to Kama, Rosa and Canadian respectively. The dataset is complete as it does not have any missing values for any of the attributes. Therefore, the ‘Impute’ function will not be needed. The final size of the dataset used is 210x9 including the target.

The table below depicts the attributes of the dataset, their definition, value type and the range of attribute values.

Feature Representation	Meaning	Value Type	Lower Range	Upper Range
ID	Unique ID number of each data object	Numerical	1	210
Target	Wheat classes	Numerical	0	2
A	Area of wheat kernel	Numerical	10.59	21.18
P	Perimeter of wheat kernel	Numerical	12.41	17.25
C	Compactness of wheat kernel	Numerical	0.8081	0.9183
LK	Length of wheat kernel	Numerical	4.899	6.675
WK	Width of wheat kernel	Numerical	2.63	4.033
A_Coef	Asymmetric coefficient	Numerical	0.7651	8.456
LKG	Length of kernel groove	Numerical	4.519	6.55

Table 1: Feature properties and values

The table below shows the statistics of central tendency and dispersion of the attributes used in the dataset.

Attribute Name	Central Tendency	Dispersion
A	14.8475	0.1955
P	14.5593	0.0895
C	0.871	0.0271
LK	5.6285	0.0785
WK	3.2586	0.1156
A_Coef	3.7002	0.4054
LKG	5.4081	0.0907

Table 2: Attribute property values

The table above shows that attributes are closely correlated, and the most data points lie in the vicinity of other attributes. The dispersion shows that the standard deviation of the attributes is low for the most part and that the data of the attributes do not tend to be far apart from one another.

Scatter Plots

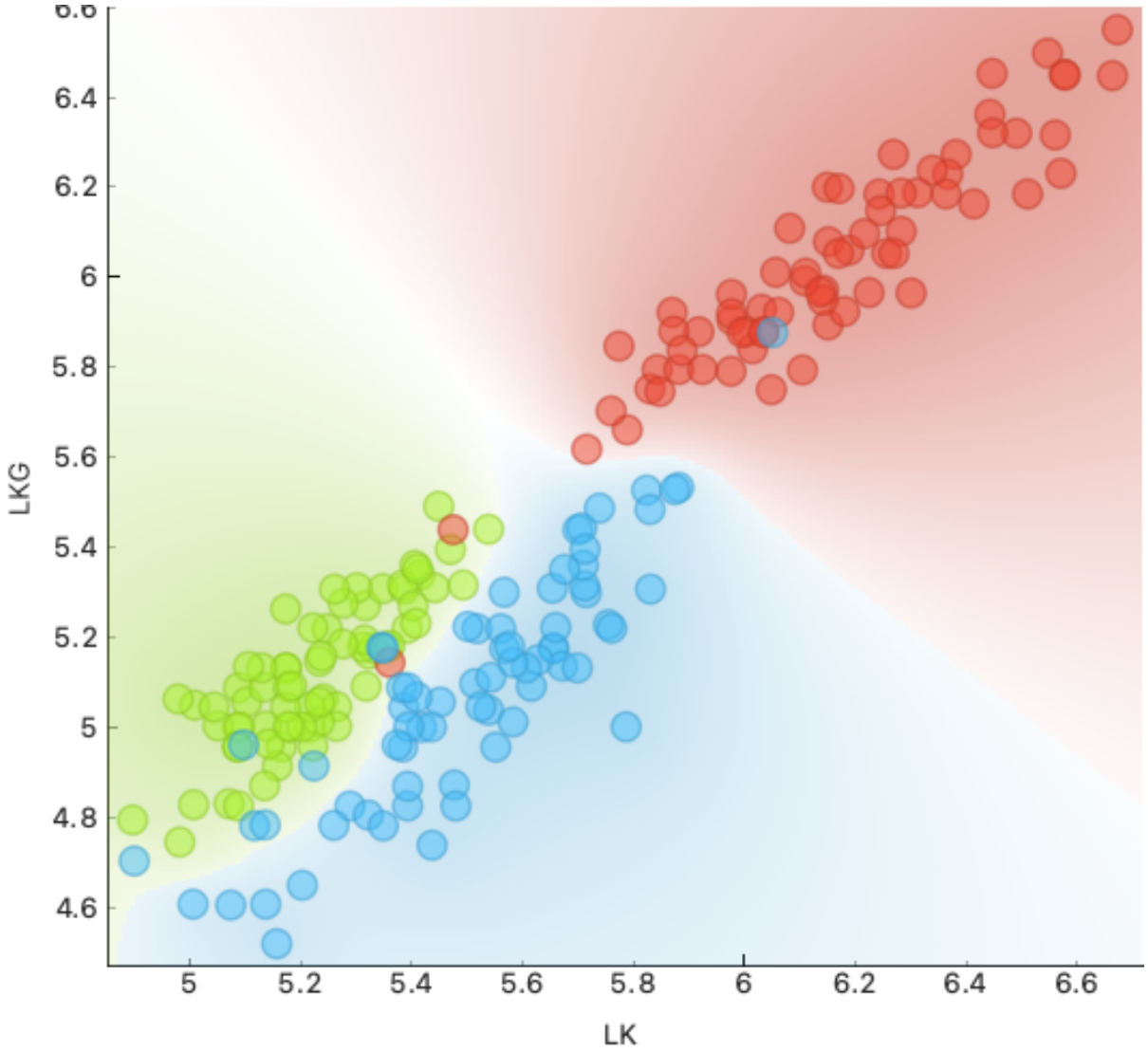

Figure 4: Scatter plot of length of wheat kernel and length of kernel groove

The figure above shows the relationship between the length of kernel and the length of kernel groove as each colour represents a different class. In the graph, the colour blue represents Kama wheat, red represents Rosa wheat and green represents Canadian wheat. These two attributes provide a high level of separability of data objects. Even though there are some outliers and close values in between classes, a distinction between the classes can be observed to a great extent. This is more evident in Rosa seed compared to the other two classes as they have a few overlapping data objects. The length of wheat kernel and length of kernel groove prove to be important attributes in classification.

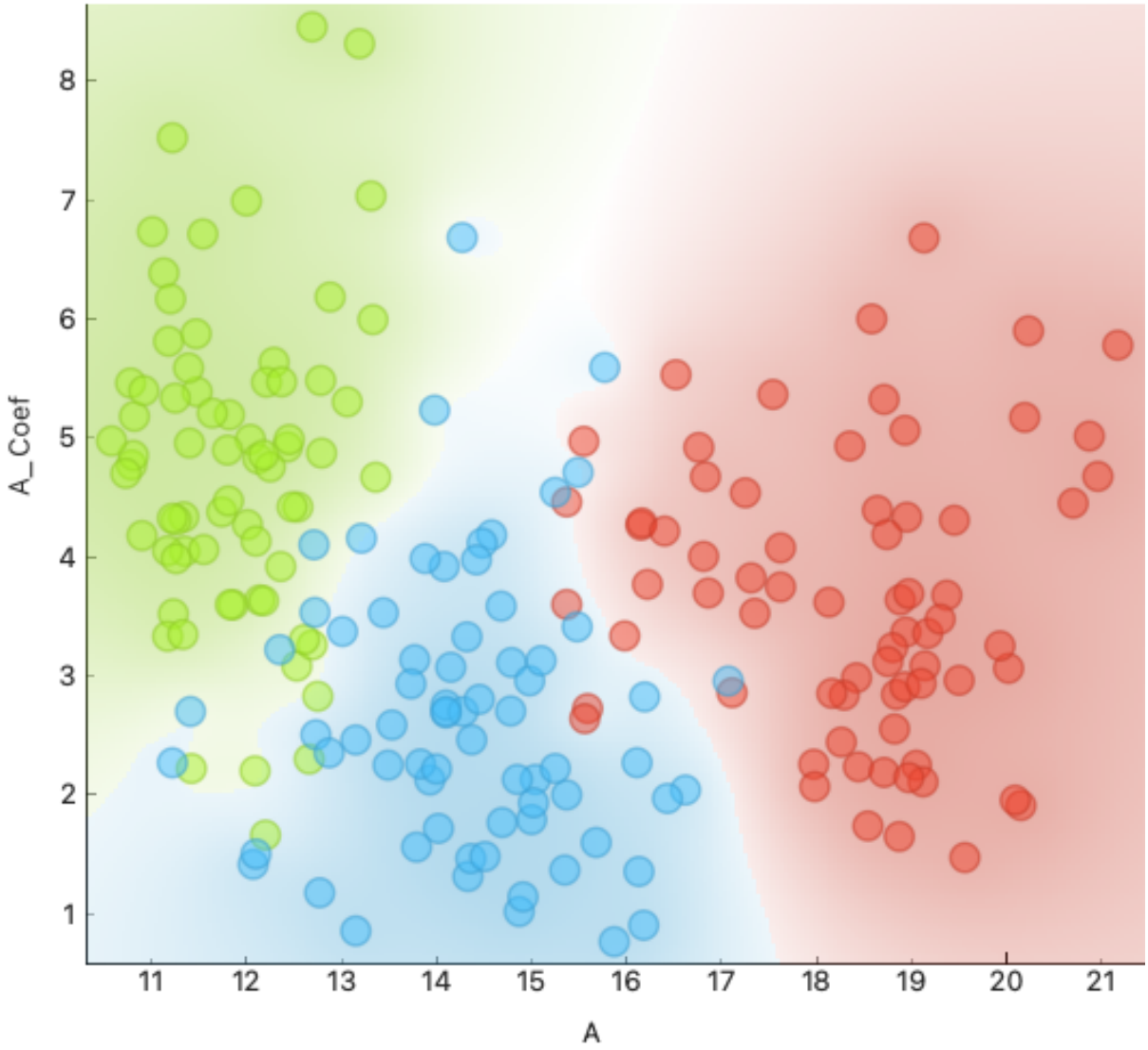

Figure 5: Scatter plot of area and asymmetric coefficient

The figure above further supports the observations made in the first scatter plot. As it was in the previous plot, there are overlapping areas, however there is a clear distinction between the classes. The scatter plot of area and asymmetric coefficient show high levels of separability of the data objects and are useful for classification.

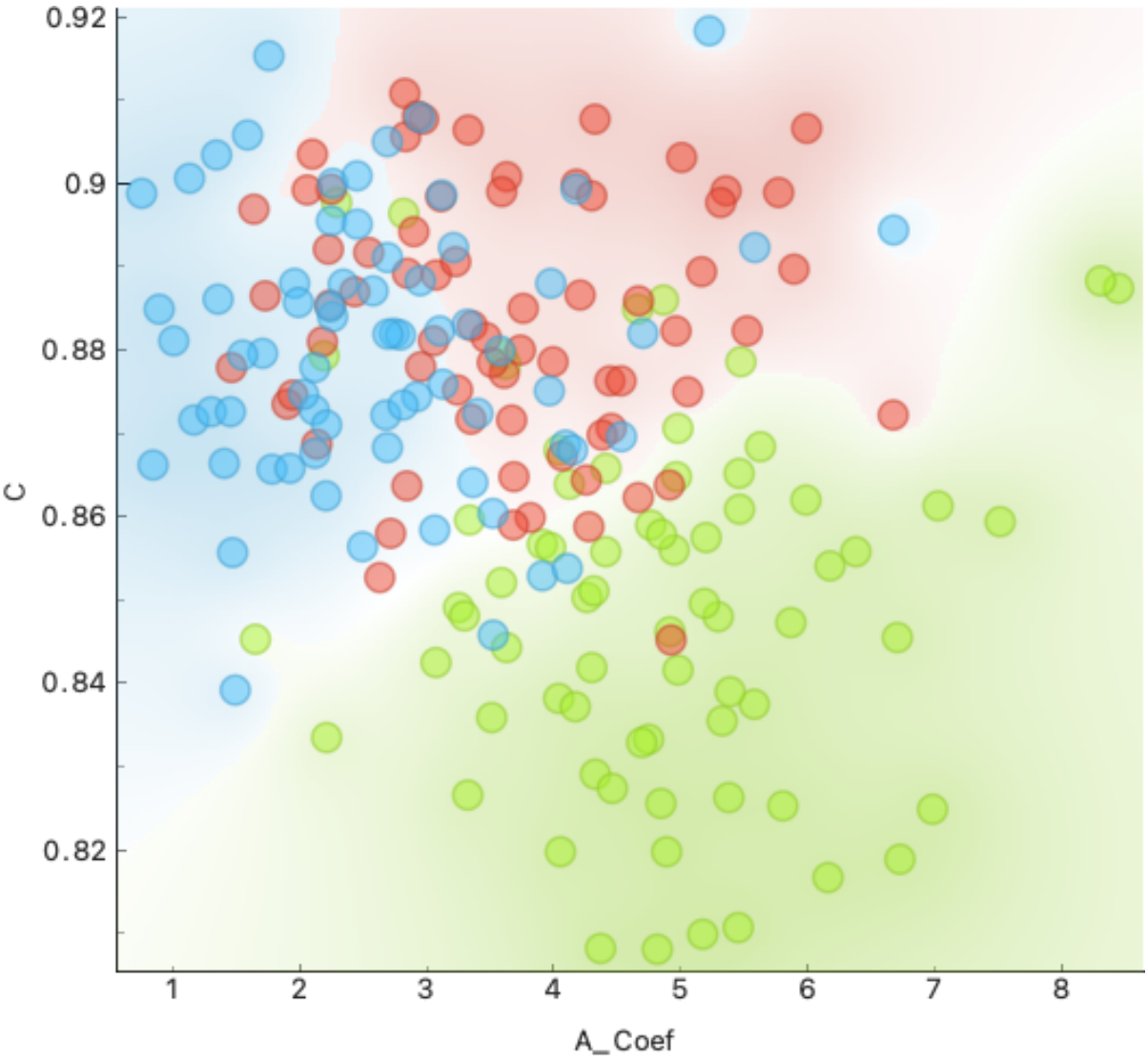

Figure 6: Scatter plot of asymmetric coefficient and compactness

In Figure 5, the scatter plot for asymmetric coefficient and compactness can be observed. Unlike the previous scatter plots, the classes are not clearly represented. Taking into consideration the relationships of compactness with most of the other attributes, it can be excluded from the classification.

Histograms

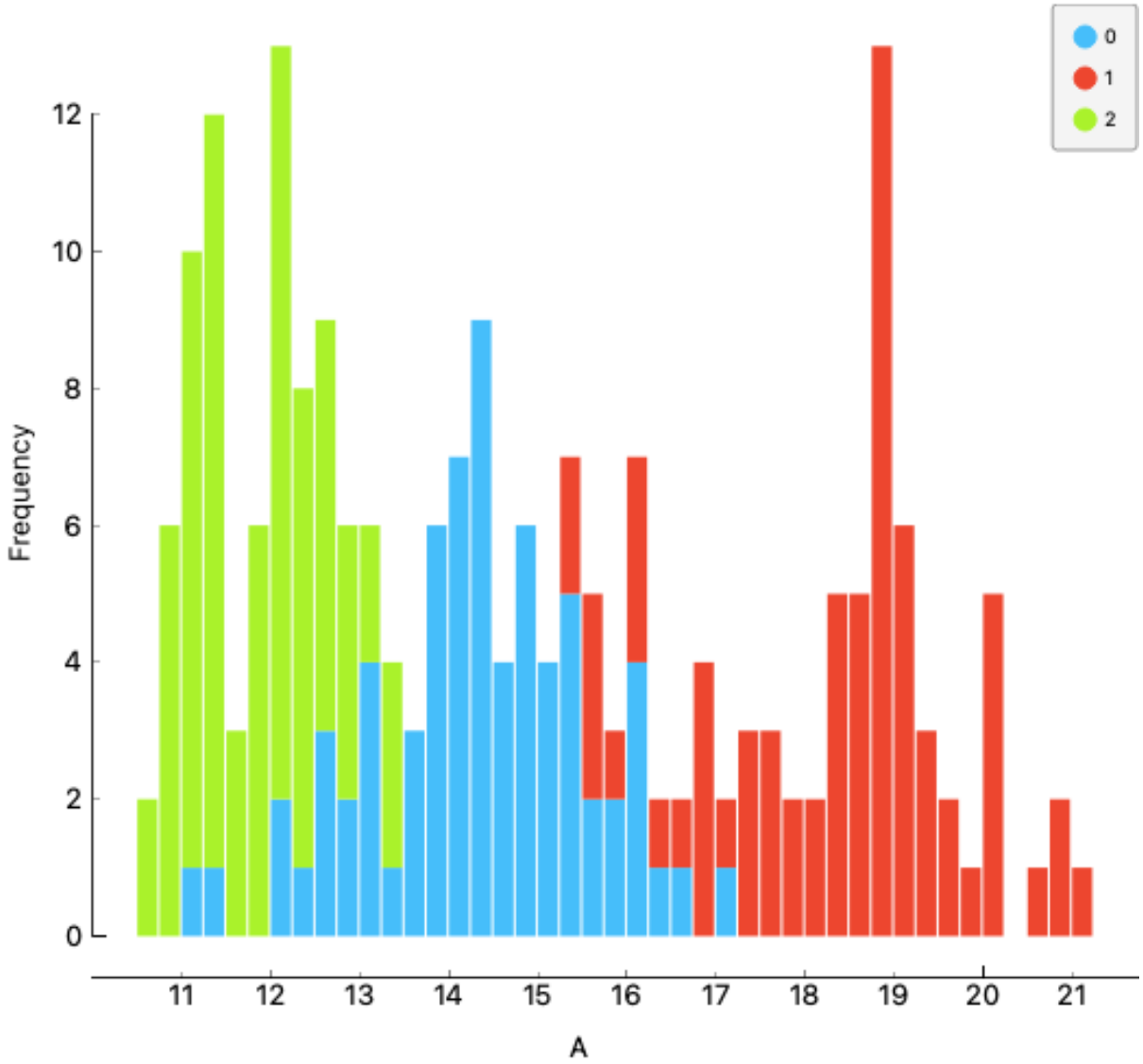

Figure 7: Histogram of area, split columns by the target

The histogram above represents the area of the wheat kernels. It can be observed that there are several overlapping elements that could belong to another class. However, the separability of classes remains high as a whole.

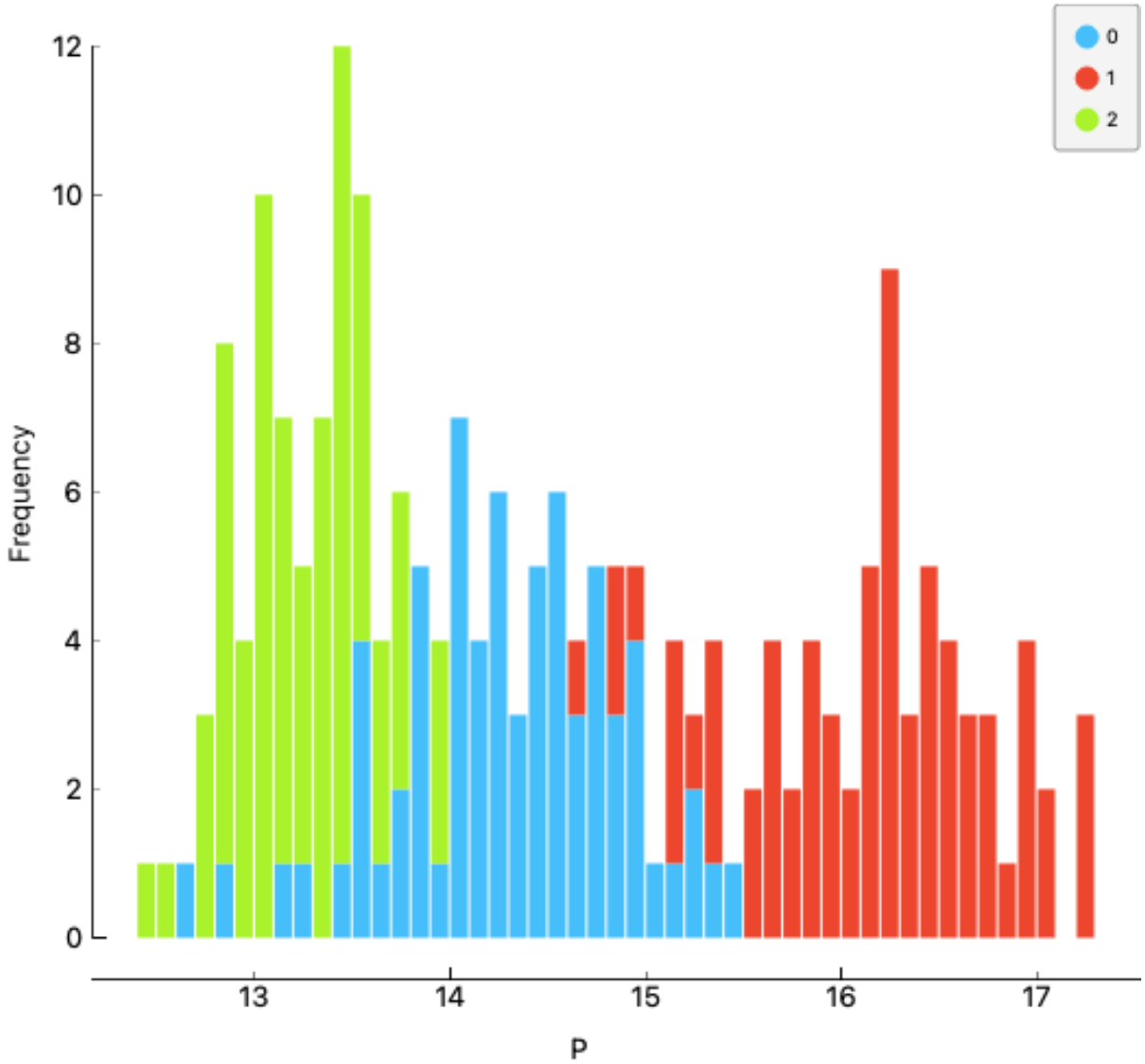

Figure 8: Histogram of perimeter, split columns by the target

The figure above, depicting the perimeter of the wheat kernels is similar to that of the histogram for the area in Figure 6. For example, leftmost data belonging to class 0, represented in blue, could be a part of class 2, represented in green. Just as several class 1 elements having attributes similar to class 0.

Distributions

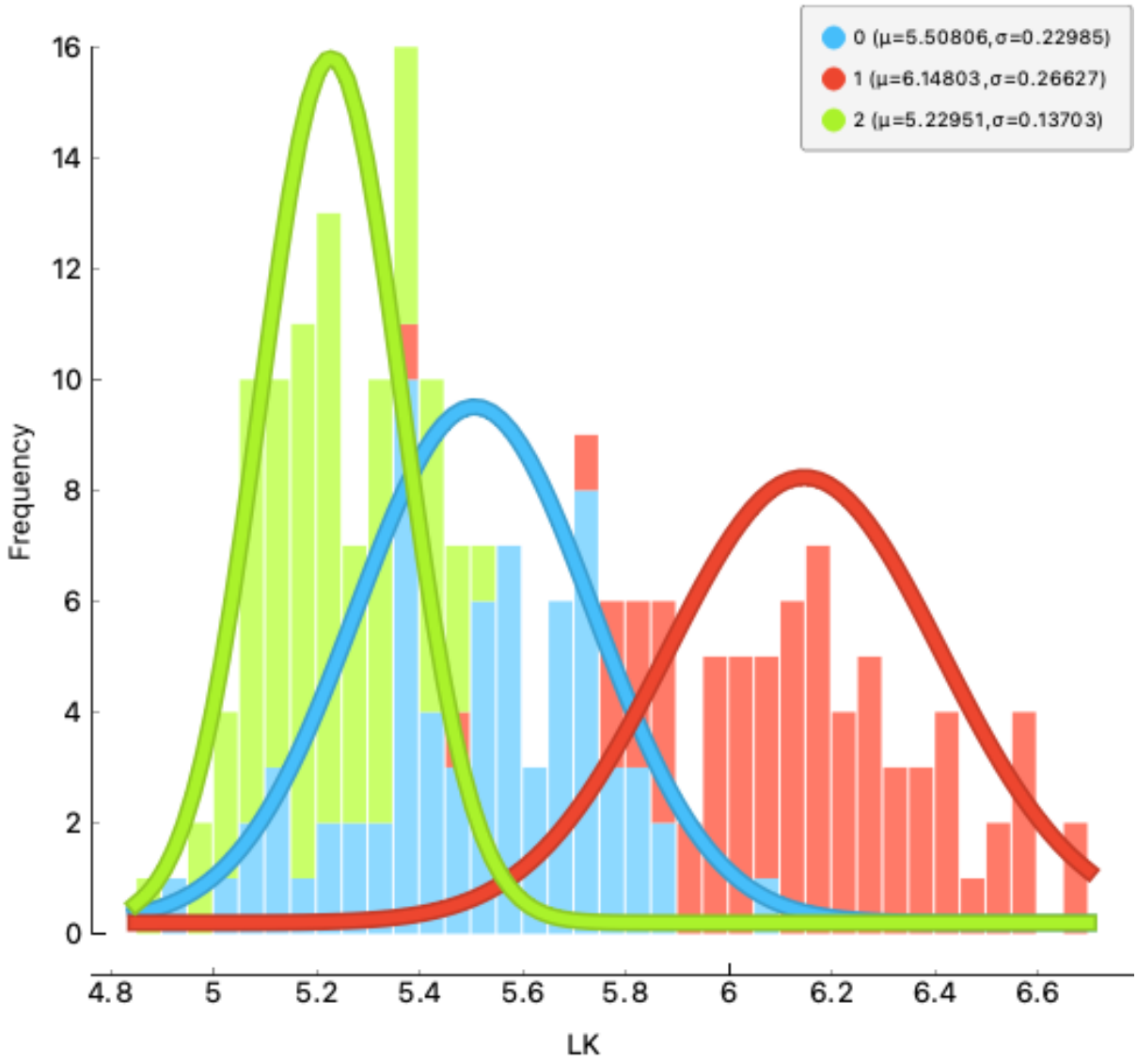

Figure 9: Normal distribution of kernel length

The figure shows the graphs that are close to normal distribution and is suitable for the machine learning algorithms.

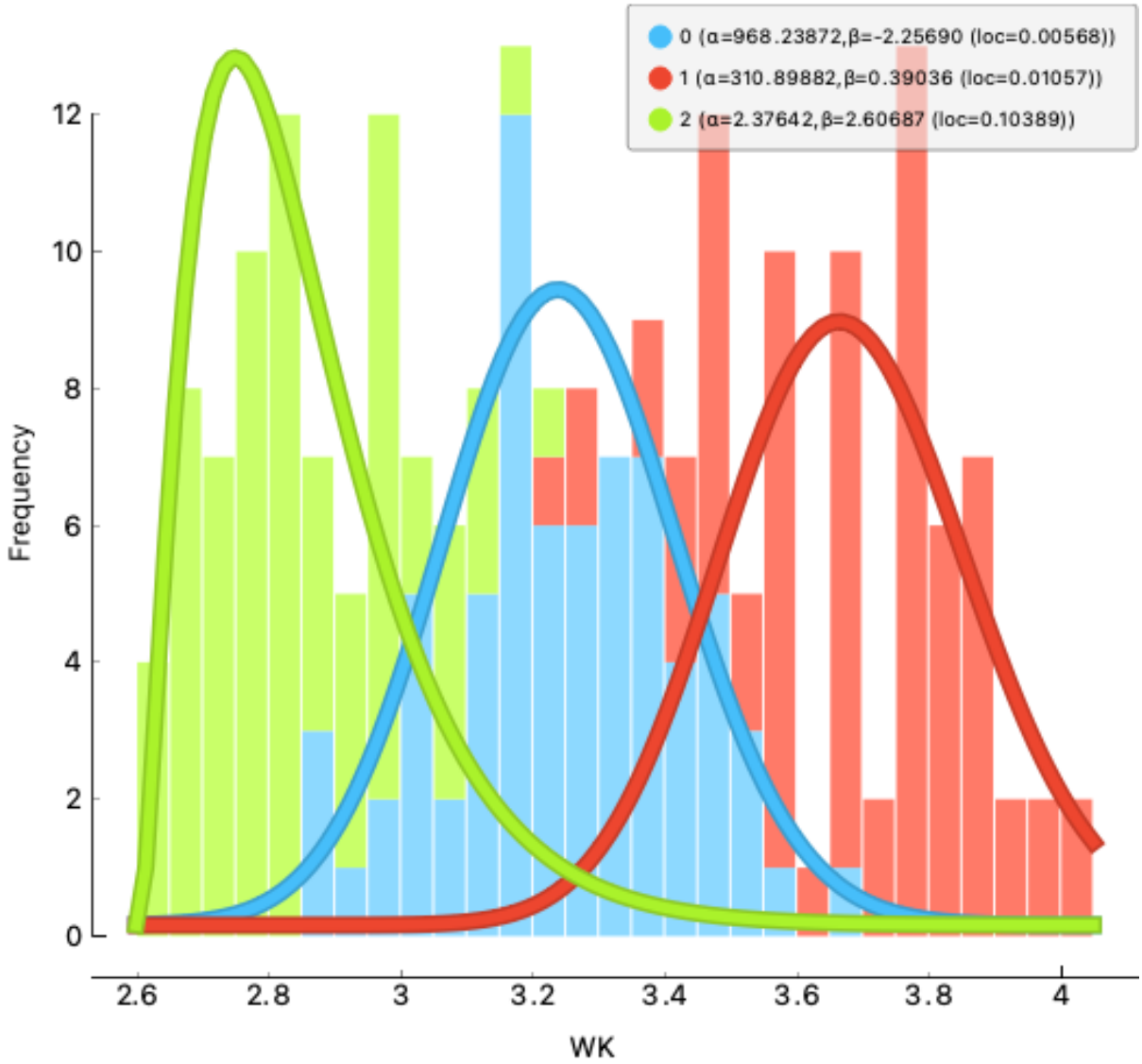

Figure 10: Gamma distribution of kernel width

In the figure above, Gamma distribution of the kernel width can be observed with shape and rate parameters. The distribution is only partially right-skewed as Gamma distribution graphs tend to be. The classes are clearly separable based on the graph of the distribution.

Conclusions

The dataset presents a perfect balance as the three classes have an equal amount of data objects. The visual data gathered from the scatter plots, histograms and the distributions show that the data objects of different classes are easily distinguishable to a great extent. Even though, there are a few objects that are outliers, the visual data depicts high level of separability. These findings are also supported by the statistical data from Table 2 as the data objects of the same class tend to be near each other.

Part 2 – Unsupervised Learning

In Unsupervised Machine Learning, Hierarchical clustering and K-means algorithm will be applied to the dataset.

K-Means

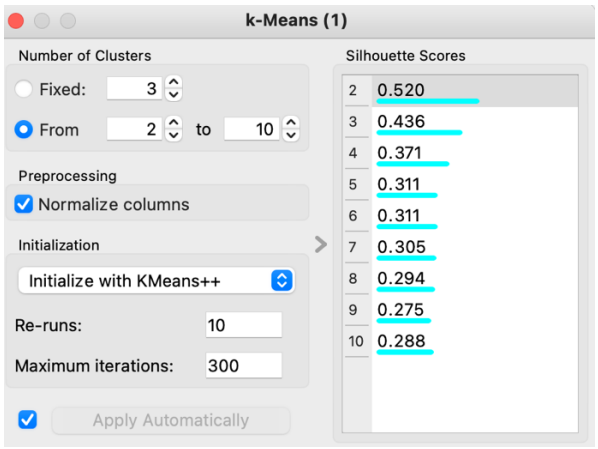

Figure 11: K-means with Kmeans++

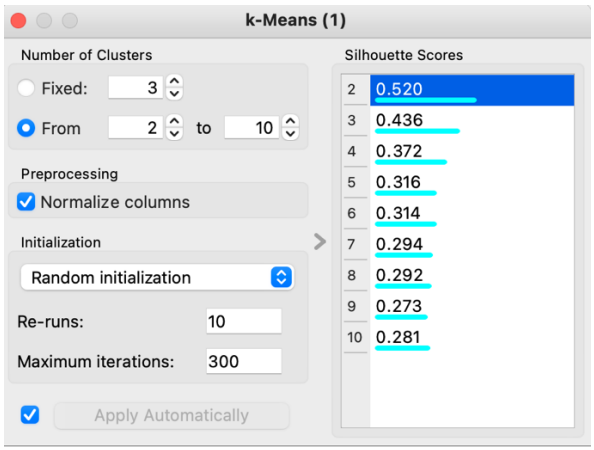

Figure 12: K-means with random initialization

The two figures above show the K-means algorithm and the silhouette scores related to the dataset. Both figures incorporate the same parameters except for one variable. They both have normalized columns as they show the silhouette scores from 2 to 10 clusters with 10 re-runs and 300 maximum iterations. However, the first figure on the left is initialized with KMeans++ while the figure on the right uses random initialization with random centroids as opposed to the data specific centroids chosen by KMeans++. As the list goes on to the second half of the 10 clusters, a slight difference can be observed between KMeans++ initialization and Random initialization. It was given that the target of the dataset has 3 classes. Nevertheless, both approaches suggest that having 2 clusters would suit the dataset better as the score for having 2 clusters is closer to the value of 1 on the Silhouette score section. This means that within the 3 clusters that exist in the dataset, there are a certain number of data objects that could belong to another cluster. This finding is also supported by the Silhouette Plot below.

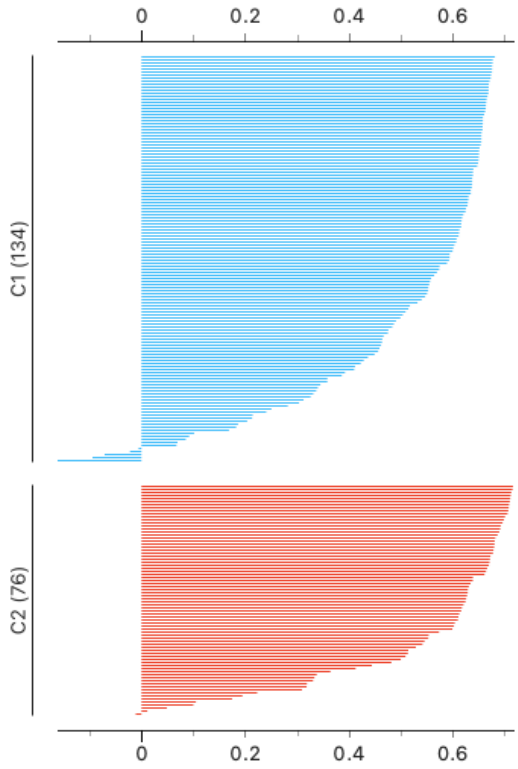

Figure 13: Silhouette Plot labelled for cluster

Figure 12 shows that, had there been 2 clusters instead of 3 as is the case in the dataset, there would have been 134 data objects belonging to cluster 1 and 76 objects belonging to cluster 2 as K-means algorithm predicted.

Hierarchical Clusters

In this bottom-up approach, the first experiment carried out on the dataset will have a normalized Euclidean distance metric on the rows of the dataset. The figure below depicts the clusters formed using the dataset. The linkage is complete and the annotations are set to the 'target'. Under the pruning section, the maximum depth is set to 13 and the top value for N is 3.

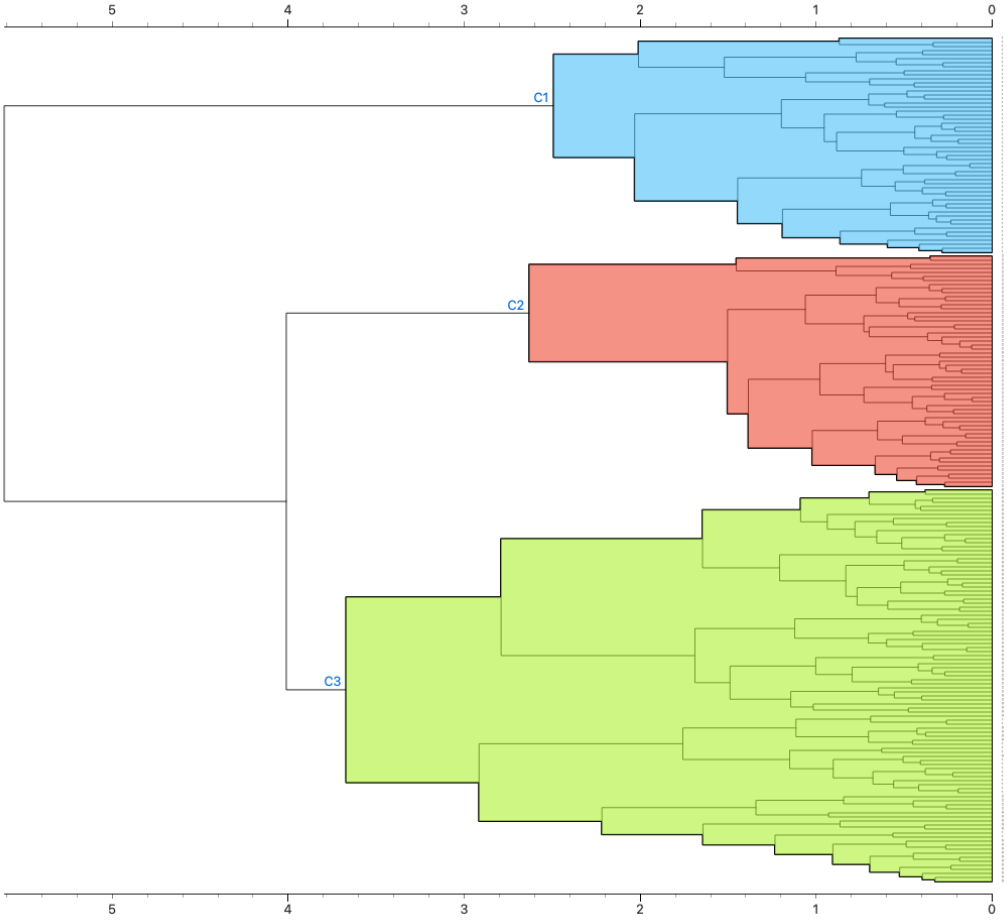

Figure 14: Hierarchical Cluster Dendrogram

The figure above shows the dendrogram of the dataset based on the given parameters. The figure is not as clear at first levels as per the data object size, however, at the first level all 210 data objects start as 210 clusters. The maximum depth of 13 allows each cluster to merge until the top N value of 3 is reached where the 3 clusters can be observed. The class separability is visible, however the amount of data objects in each cluster is changed from the balanced 70 data objects for each class. While cluster 1 and cluster 2 are closer in size, cluster 3 has additional data objects.

For the second experiment using hierarchical clusters, the parameters have been altered. The linkage defining the distance is set to average, which means that the graph will consider the average distance in between data objects or clusters. The annotations are set to ‘none’. The pruning has been changed to the maximum depth value of 5. The selection is once again has the value of 3 as top N. The below figure presents the dendrogram of the dataset based on these given parameters.

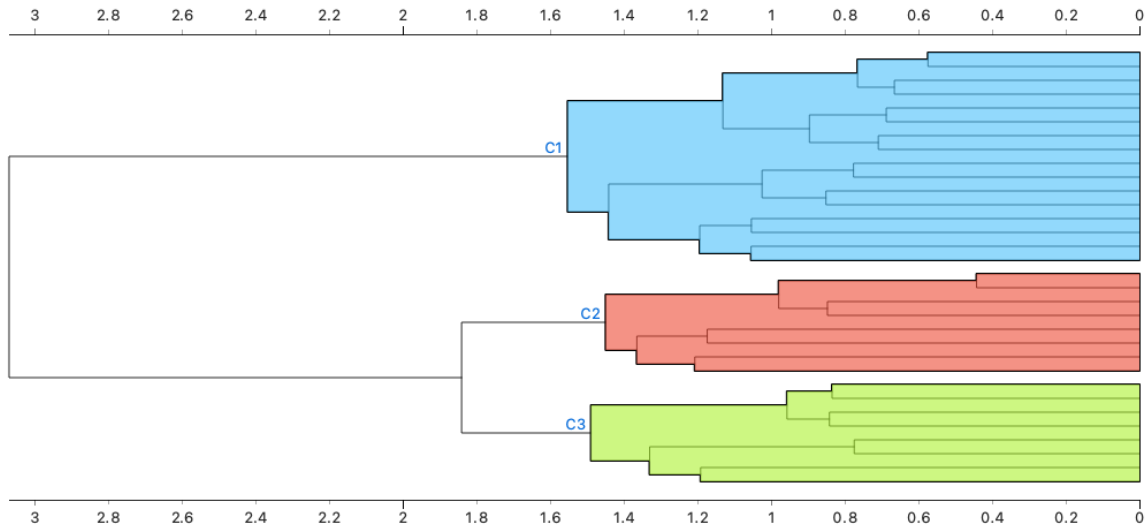

Figure 15: Hierarchical Cluster Dendrogram 2

The figure clearly presents the separability of the dataset as 3 clusters that are the target can be easily distinguished. The maximum depth value of 5 limits the amount of data objects as less clusters are included in the dendrogram. These clusters merge only up to 5 times with the most similar value to their own clusters.

Lastly, the effect of Manhattan distance and weighted distance will be checked to see if and how the clusters change. The below figure is created with Manhattan distance. The hyperparameters chosen are the weighted distance, maximum depth is 13 and the top N value is shown at 3 clusters.

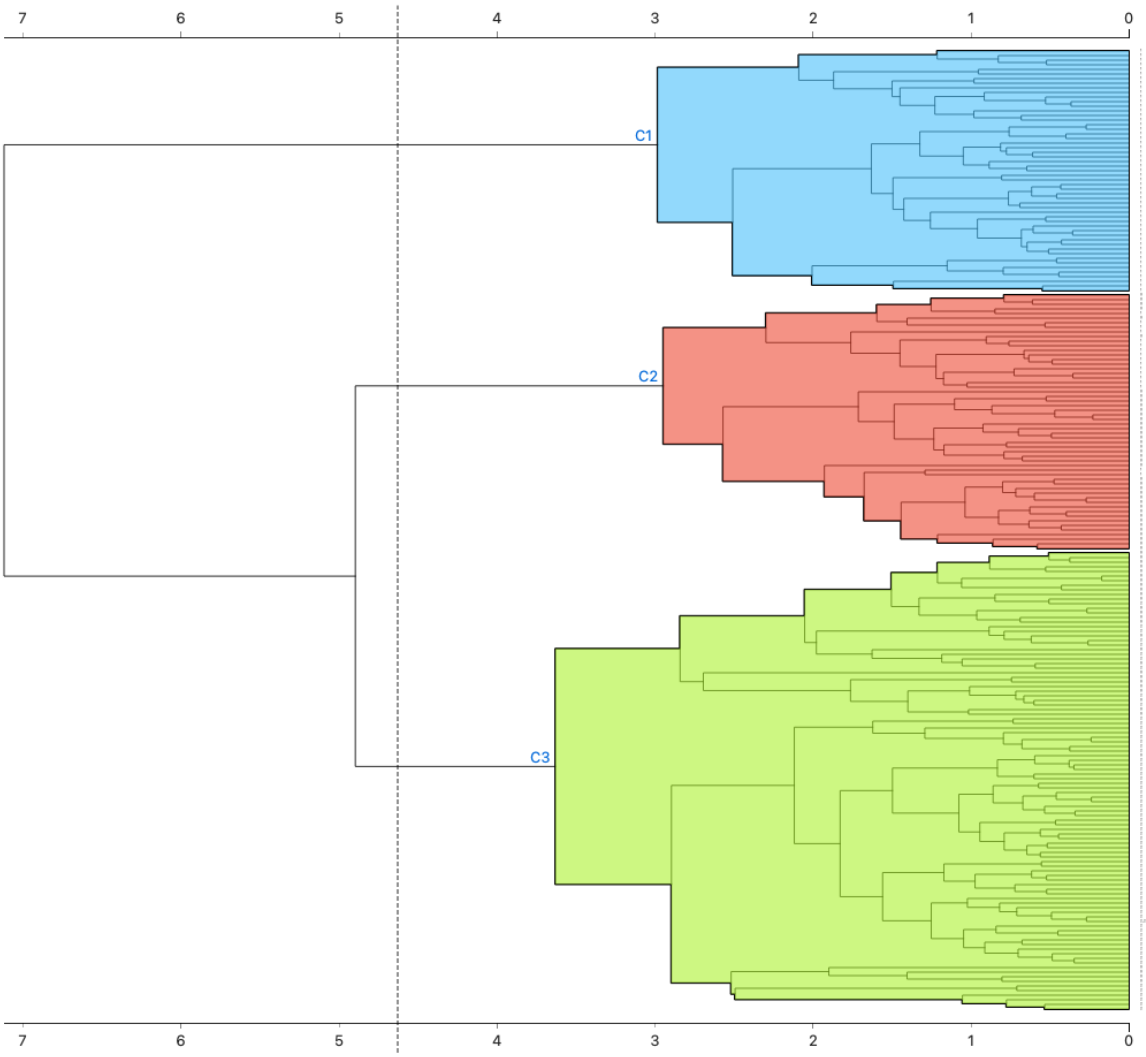

Figure 16: Hierarchical Cluster Dendrogram 3

Once again, the figure shows high separability among the clusters. It was observed that compared to the first dendrogram which had the same maximum depth value as this one, instead of all 210 data objects as first clusters, there were 208 clusters at first step since the switch to Manhattan distance was made. Although the clusters exhibit the high separability of the dataset, the clusters are not perfectly balanced. The clusters 1 and 2 share common size while cluster 3 is bigger. The figure shows that the clusters would be more balanced with 4 clusters at the top N value at 4.

Overall, each figure based on different hyperparameters present dendrograms where the high level of separability of the dataset is prevalent. Based on the different distance metrics and linkage types, the data objects can be manipulated into different clusters. Through these experiments, the divergent dendrograms allowed for presenting the data in different ways and helped in the comprehension of the characteristics of the sets.

Part 3 – Supervised Learning

This part of the assignment will be focused on Supervised Learning with three different classification algorithms. In addition to the Artificial Neural Networks, Random Forest and kNN will be used in classification of the dataset.

First of all, since our dataset is divided into three classes, binary classification systems cannot be used. In addition, the purpose of this supervised machine learning is to experiment with different hyperparameters. Due to this reason, algorithms such as Naïve-Bayes was overlooked due to the lack of hyperparameter changes.

The first algorithm picked, Random Forest, is a widely used algorithm that is suitable for classification tasks. Random Forest creates multiple decision trees based on the dataset. The outcome of these decision trees produces a vote on the classification. The majority vote decides the class the data object should belong to. When working with a large sample size, Random Forest can take significantly higher amounts of time to complete its classification task. However, the dataset used in this assignment allows Random Forest algorithm to remain efficient.

The second algorithm, that will be used in this Supervised Learning, is k-Nearest Neighbour (kNN) algorithm. This algorithm is a simple classification algorithm that uses similarity of data objects to sort them into classes. The similarity of the data objects here refers to the distance of the data objects. Data objects that are in the same class are likely to be in the same proximity. kNN algorithm takes a data object and looks at the k number of neighbours the data object. Based on the feature similarity, kNN classifies the data objects into the target classes. It was also chosen as the metric of distance, number of neighbours considered, and the weight are hyperparameters that can be changed to experiment with.

For each algorithm, the training data size will be at 70% which is 147 data objects. The sampling will be done at random with training and testing being done 10 times.

Neural Network

Neural Network has the hyperparameters shown in the table below.

Hyperparameter	Meaning
Neurons in hidden layers	Allows to set the number of neurons in hidden layers
Activation	Selects the activation function in neurons
Solver	Optimization algorithm
Regularization	Learning rate
Maximal number of iterations	Sets the maximum level of iterations that will be used by the algorithm

Table 3: Neural Network Hyperparameters

For the first experiment using Neural Network, the following hyperparameters will be used.

Neurons in hidden layers = 3
Activation = ReLu
Solver = Adam
Regularization = 0
Maximal number of iterations = 200

Model	AUC	CA	F1	Precision	Recall
Neural Network	0.684	0.329	0.175	0.231	0.329

Figure 17: Neural Networks Test and Score

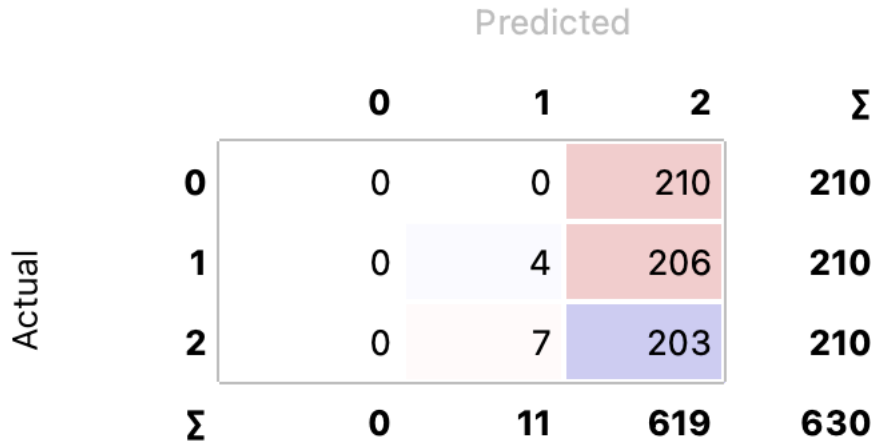

Figure 18: Neural Networks Confusion Matrix

As the results above show, Neural Networks algorithm with the set hyperparameters does not provide high level of precision in classification. One of the main reasons of this is that there is only a single hidden layer with only 3 neurons and the learning rate at 0.

For the second experiment using Neural Network, the following hyperparameters will be used.

Neurons in hidden layers = 3,5
Activation = Logistic
Solver = Adam
Regularization = 0.02
Maximal number of iterations = 200

Model	AUC	CA	F1	Precision	Recall
Neural Network	0.825	0.671	0.547	0.787	0.671

Figure 19: Neural Networks Test and Score 2

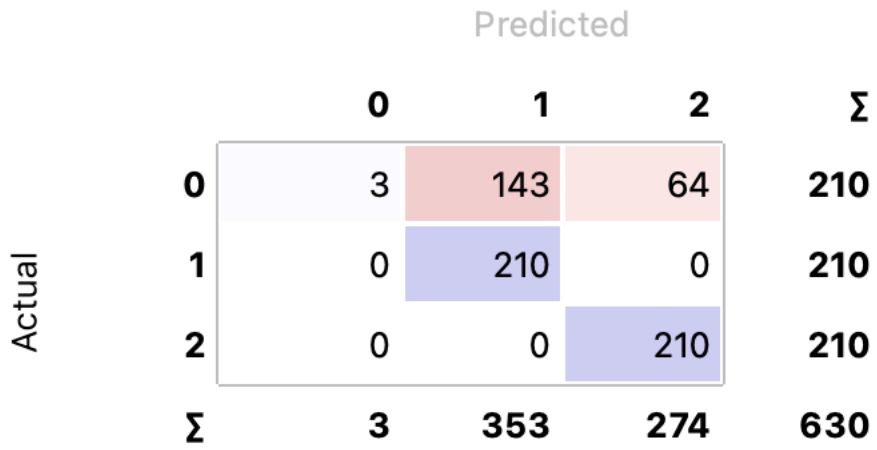

Figure 20: Neural Networks Confusion Matrix 2

Based on the results above, changes made to the hyperparameters provided a high level of improvement to all results. The algorithm managed to correctly classify all data objects in the classes 1 and 2. However, it was very poor with classifying data objects belonging to class 0. The main reason for the improvement may be the second hidden layer added with 5 new neurons.

For the third experiment using Neural Network, the following hyperparameters will be used.

Neurons in hidden layers = 3,5,10
Activation = Identity
Solver = SGD
Regularization = 0.1
Maximal number of iterations = 200

Model	AUC	CA	F1	Precision	Recall
Neural Network	0.971	0.863	0.862	0.862	0.863

Figure 21: Neural Networks Test and Score 3

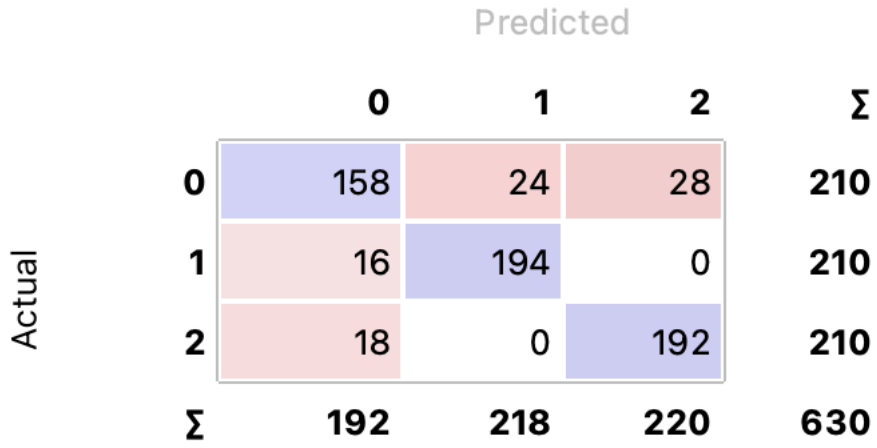

Figure 22: Neural Networks Confusion Matrix 3

The above results for the third experiment show the highest level of precision out of the three experiments. The three hidden layers with more neurons and the higher learning rate allows for a usable model that has high precision.

Based on the collective results from the three experiments with different hyperparameters, it can be observed that the higher amounts of hidden layers and neurons as well as the learning rate affect the performance greatly. As these hyperparameter values are increased, so does the level of performance output by the algorithm. Therefore, the model from the third experiment will be used.

Random Forest

Random Forest has the hyperparameters shown in the table below.

Hyperparameter	Meaning
Number of trees	Allows to set the number of decision trees
Number of attributes considered in each split	Allows to set the number of attributes that will be considered (cannot exceed the attribute level of the dataset)
Balance class distribution	Equalizes the distribution of the classes in the dataset
Limit depth of individual trees	Sets the maximum depth of each decision tree
Do not split subsets smaller than	Sets the minimum amount of data objects a subset can have without being split

Table 4: Random Forest Hyperparameters

For the first experiment using Random Forest, the following hyperparameters will be used.

Number of trees = 3
Number of attributes considered in each split = 3
Balance class distribution = Yes
Limit depth of individual trees = 2
Do not split subsets smaller than = 3

Model	AUC	CA	F1	Precision	Recall
Random Forest	0.949	0.863	0.865	0.868	0.863

Figure 23: Random Forest Test and Score

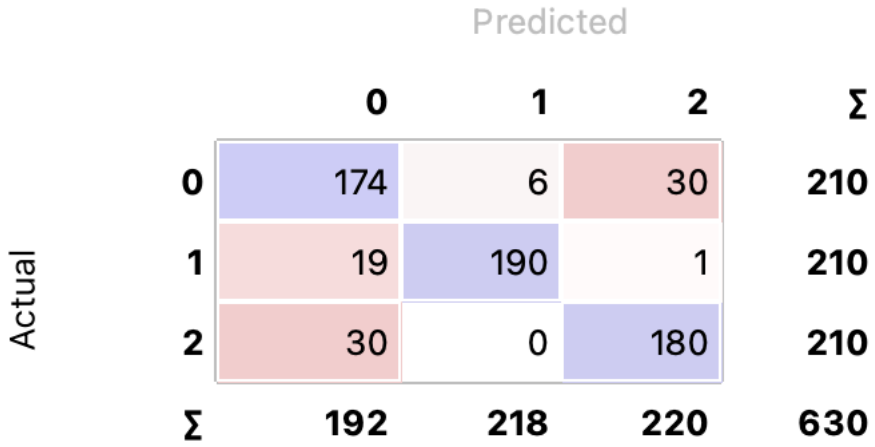

Figure 24: Random Forest Confusion Matrix

The results above show promising values as a first experiment. There is a high level of precision and area under the curve value. The only possible problem is the warning received from the Orange Tool itself that weighting by class may decrease the performance.

For the second experiment using Random Forest, the following hyperparameters will be used.

Number of trees = 7
Number of attributes considered in each split = 5
Balance class distribution = No
Limit depth of individual trees = 5
Do not split subsets smaller than = 5

Model	AUC	CA	F1	Precision	Recall
Random Forest	0.973	0.921	0.921	0.921	0.921

Figure 25: Random Forest Test and Score 2

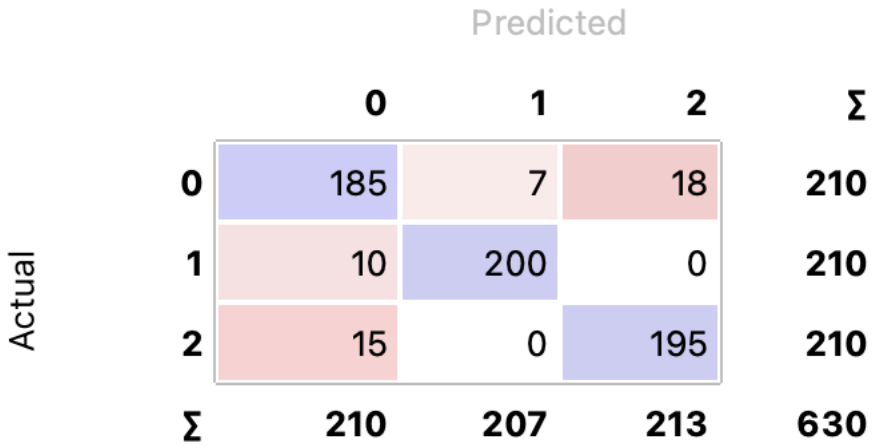

Figure 26: Random Forest Confusion Matrix 2

The results above for this experiment show higher performance compared to the results from the first experiment. One of the reasons for this may be the balance class distribution being turned off because of the warning of Orange Tool. The algorithm having a higher depth limit allowed it to function better and classify with higher accuracy.

For the third experiment using Random Forest, the following hyperparameters will be used.

Number of trees = 10
Number of attributes considered in each split = 6
Balance class distribution = No
Limit depth of individual trees = 10
Do not split subsets smaller than = 5

Model	AUC	CA	F1	Precision	Recall
Random Forest	0.973	0.911	0.911	0.911	0.911

Figure 27: Random Forest Test and Score 3

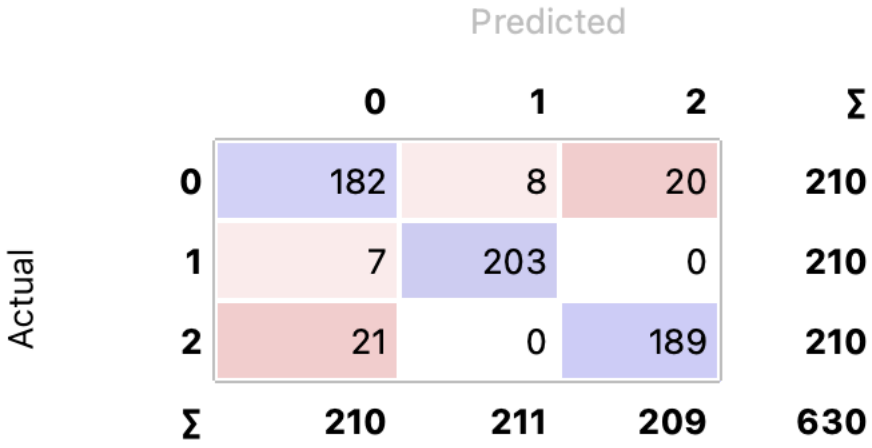

Figure 28: Random Forest Confusion Matrix 3

The third experiment presents similar results to the second experiment. The maximum number of attributes considered may have been a hinderance. The AUC levels of experiments two and three are identical, however the third model shows slightly lower performance in all other categories.

Therefore, the model from the second experiment will be picked.

kNN

kNN algorithm has the hyperparameters shown in the table below.

Hyperparameter	Meaning
Number of neighbours	Allows to set the number of neighbours that will be considered for each data object
Metric	Selects the metric for distance measurement
Weight	Decides if closer neighbours' votes carry more weight as opposed the farther ones in making a decision for classification

Table 5: kNN Hyperparameters

For the first experiment using kNN, the following hyperparameters will be used.

Number of neighbours = 1
Metric = Euclidean
Weight = Uniform

Model	AUC	CA	F1	Precision	Recall
kNN	0.920	0.894	0.893	0.894	0.894

Figure 29: kNN Test and Score

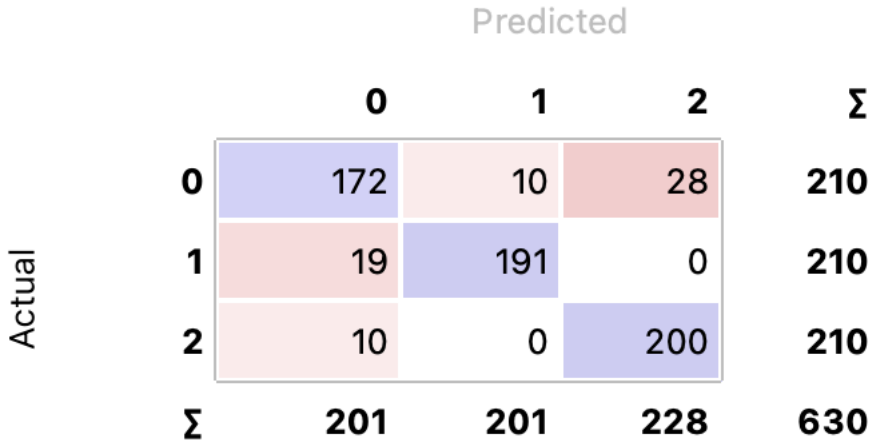

Figure 30: kNN Confusion Matrix

The results above show that the kNN algorithm is well-performing in the given parameters. The number of neighbours being 1 shows that the data objects are separated well, and the data objects of a certain class are close to the other data objects of the same class.

For the second experiment using kNN, the following hyperparameters will be used.

Number of neighbours = 4
Metric = Manhattan
Weight = Distance

Model	AUC	CA	F1	Precision	Recall
kNN	0.972	0.902	0.902	0.903	0.902

Figure 31: kNN Test and Score 2

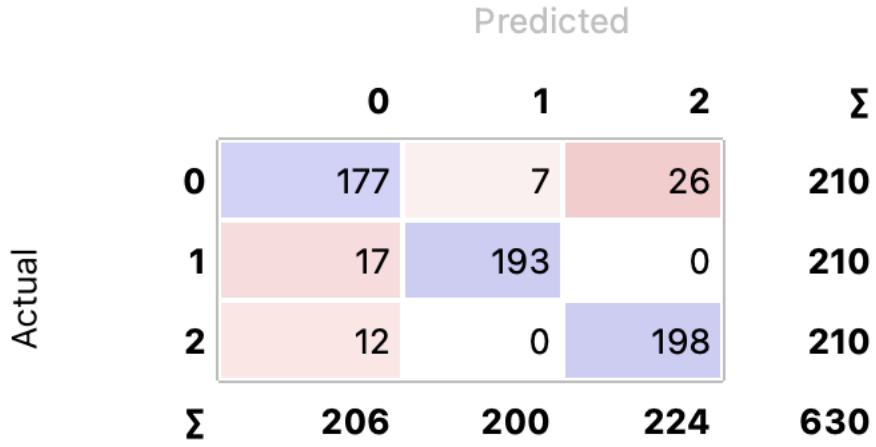

Figure 32: kNN Confusion Matrix 2

The results above show that using a higher number of neighbours and using the weight distance produces higher performance in the kNN algorithm for the given dataset.

For the third experiment using kNN, the following hyperparameters will be used.

Number of neighbours = 7
Metric = Euclidean
Weight = Distance

Model	AUC	CA	F1	Precision	Recall
kNN	0.983	0.881	0.881	0.882	0.881

Figure 33: kNN Test and Score 3

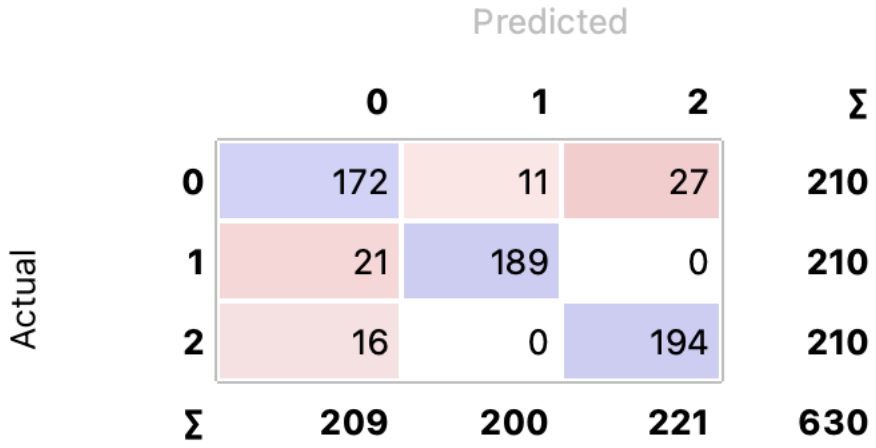

Figure 34: kNN Confusion Matrix 3

The results above show a very high level of AUC. However, the precision/recall of the algorithm is low, and many data objects are put in the wrong classes.

Therefore, the second model will be used.

Comparison of the Algorithms

The models used for each algorithm are below.

Neural Network

Neurons in hidden layers = 3,5,10
Activation = Identity
Solver = SGD
Regularization = 0.1
Maximal number of iterations = 200

Random Forest

Number of trees = 7
Number of attributes considered in each split = 5
Balance class distribution = No
Limit depth of individual trees = 5
Do not split subsets smaller than = 5

kNN

Number of neighbours = 4
Metric = Manhattan
Weight = Distance

Model	AUC	CA	F1	Precision	Recall
Random Forest	0.971	0.911	0.910	0.911	0.911
kNN	0.972	0.902	0.902	0.903	0.902
Neural Network	0.971	0.863	0.862	0.862	0.863

Figure 35: Test and Score for all three algorithms used

The results show that all three algorithms have near identical AUC. However, in other important performance parameters such as precision and recall, Neural Network falls behind of Random Forest and kNN algorithms. Based on the findings, while kNN algorithm produces classification with high accuracy, Random Forest appears to be the highest performance algorithm.

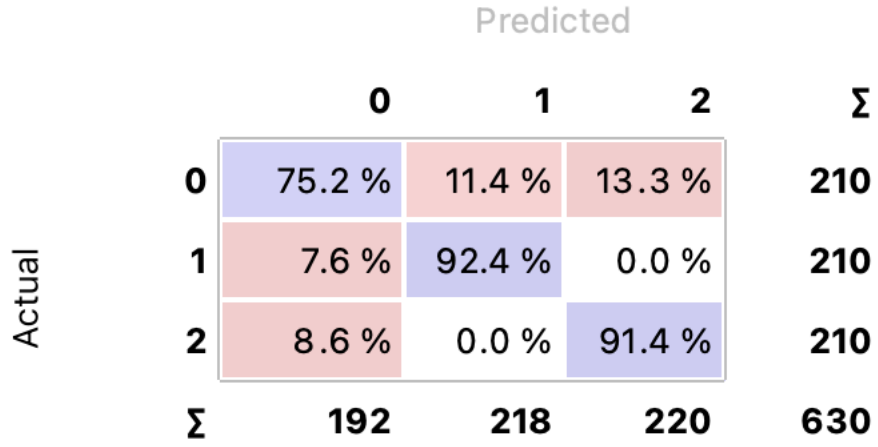

Figure 36: Neural Network Confusion Matrix Proportion of Actual Values

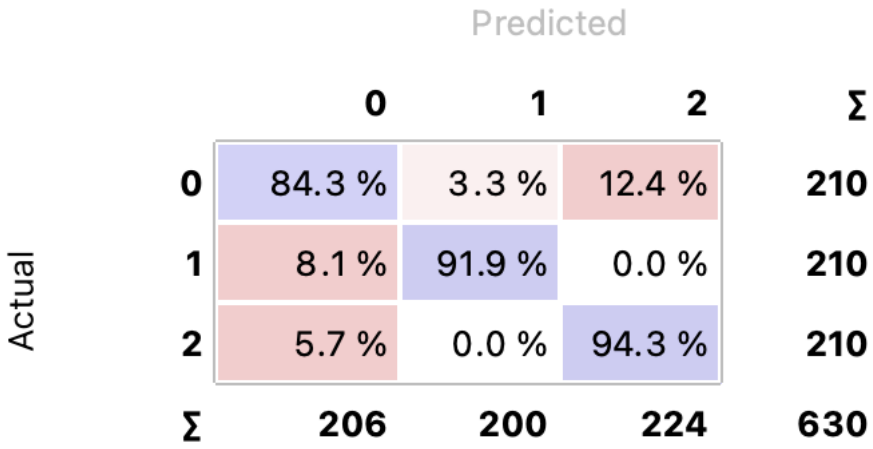

Figure 37: kNN Confusion Matrix Proportion of Actual Values

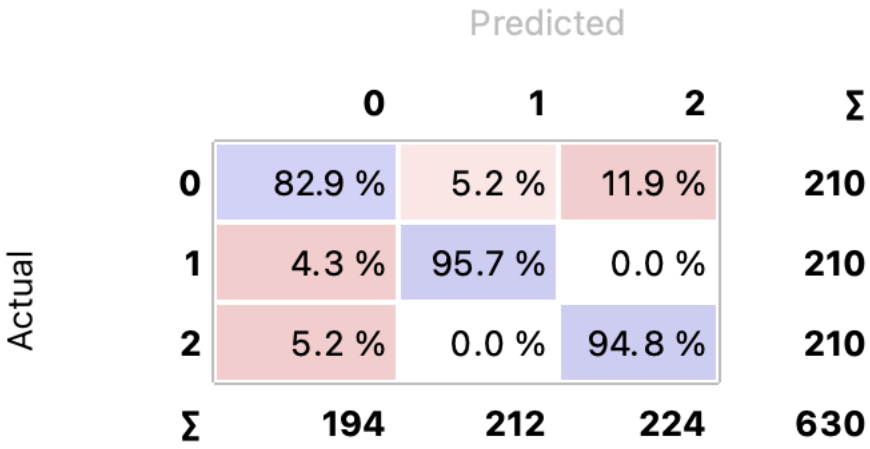

Figure 38: Random Forest Confusion Matrix Proportion of Actual Values

In agreement with the Test and Score table in Figure 35, the above figures show that Random Forest had the highest percentage of data objects classified correctly.

References and Sources

Orange Tool - https://orangedatamining.com/

Kaggle - https://www.kaggle.com/datasets/dongeorge/seed-from-uci

UCI - https://archive.ics.uci.edu/ml/datasets/seeds#