Module Code: CS3DS19
Assignment Report Title: Major Coursework Assignment
Student Number: 26011251
Date when work completed: 22/03/21
Actual hours spent for the assignment: 25

Task 1 - Data Exploration and Clustering

K- Means Clustering and Evaluation Metrics

The first task requires the clustering of a multidimensional data set. A cluster can be described as 'a collection of data points aggregated together because of certain similarities' [1].

The K-means algorithm has been used to assign each item in the dataset a cluster grouping. In the algorithm, K number of centroids are randomly selected. For each data point, the distance to each centroid is measured. The centroid closest to the data point is the cluster it belongs to. This process is repeated with a new centroid until either: the assignment of data points to clusters does not change, or the maximum number of iterations has been reached.

The success of the K-means algorithm in clustering data can be measured by several different evaluation metrics:

- The classification accuracy shows how many correct predictions have been made against the total number of predictions. A high accuracy may be misleading however if the data set is imbalanced as the model may fail to correctly classify any members of the minority class.
- The entropy is the measure of uncertainty; a high entropy indicates great uncertainty in prediction, and a low entropy indicates great confidence in prediction.
- The Cohen's Kappa is amount of observed non-chance agreement divided by the possible amount of non-chance agreement. It is used to calculate how much of the data set could be predicted by chance alone [2].

Knime Workflow

The 'wine' dataset is obtained from chemical analysis of three different cultivars of wine, grown in the same region of Italy. Each data record contains the cultivar ID number, and the thirteen chemical constituents as numerical attributes.

In the Knime workflow, the wine.csv file is read and the class numbers are converted to strings (this is for later comparison with the predicted results). In the first branch, each class is assigned a colour for easier visual representation. The PCA node is then used to reduce the number of dimensions down from thirteen to two for representation on a scatter plot. Principal Component Analysis visualises data in such a way to emphasise variation and bring out strong patterns in a dataset [3]. The data columns are resorted and the two PCAs are shown on the scatter plot. Each data point on the plot is a different wine, coloured according to its cultivar.

In the second branch, the data is classified by the K-means algorithm. As there are three cultivars of wine, the value of K is set to three. The data is again coloured, reduced to two dimensions, and shown on a scatter plot. The data points are coloured by the cluster they have been assigned to. Three additional scatter plots are also created for each individual cluster, selected by the row filter node on three separate subbranches. On the final subbranch, the clusters are renamed to the class they represent. The classification accuracy is then calculated by the scorer node.

The final, short branch, uses a modified K-Means cluster node which is then fead into an entropy scorer.

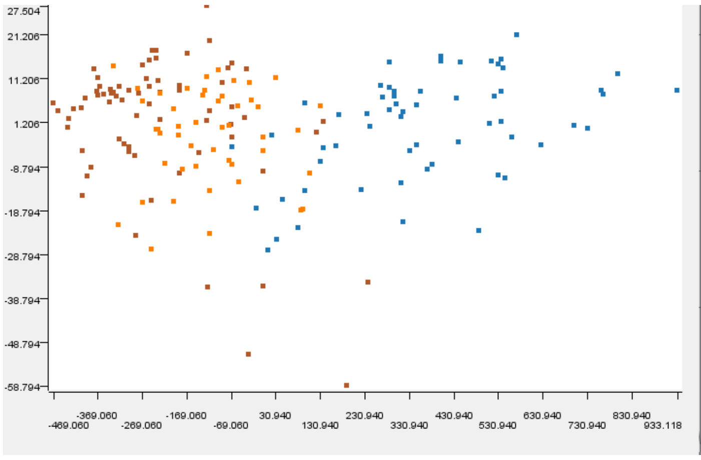
This workflow is reused with normalised results so the difference in classification performance can be compared. On the second workflow, min/max normalisation has been used to fit the thirteen chemical attributes between values of 0 and 1.

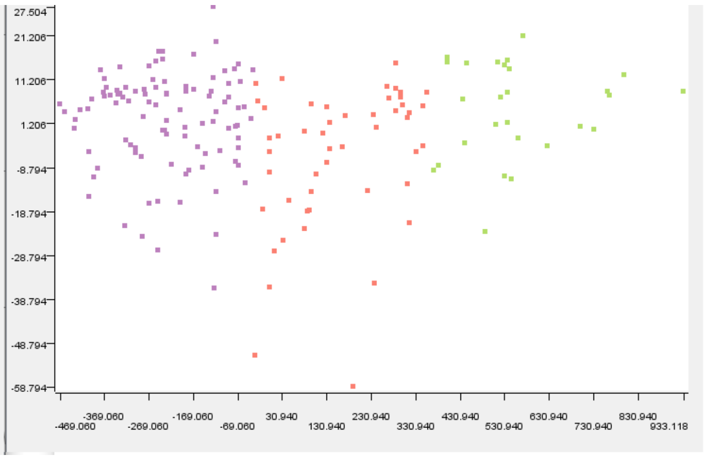
Results and Comparison

The first and second plots show the distribution of the classes and the three clusters generated by the K-means algorithm. Of the 178 data points, only 102 have been allocated a cluster corresponding to their true class. Towards the left side of Plot 1, wines of the second and third cultivar are intermixed with one another. When allocating clusters by distance to the nearest centroid, these data points were each assigned to the same cluster.

The intermixing of data points of the second and third cultivar indicate they are more compositionally similar to each other than to wines of the first cultivar. The right cluster (green) on the second plot only has wines from the first cultivar as they are distinctly different enough from wines of the second and third cultivars to be plotted separately.

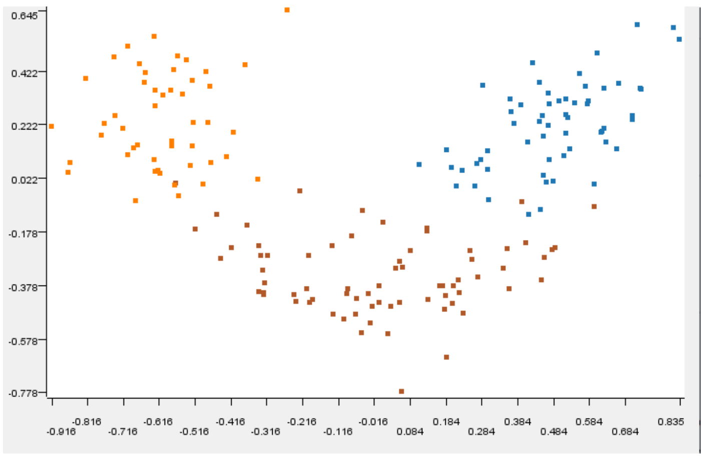
An accuracy of 57.3% does not necessarily indicate that the clustering is ineffective, in fact, it shows that wines of different cultivars are compositionally similar enough to be grouped together. This could be beneficial in cases where there is shortage of a certain cultivar, as an alternative can be found in a similar wine within the group.

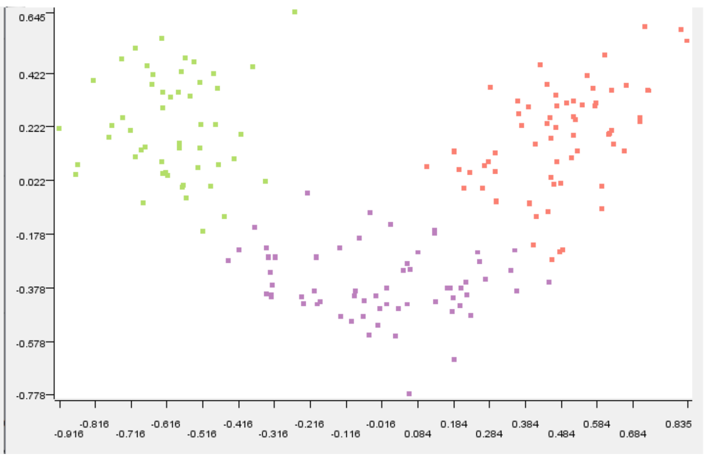




Plot 1 - PCA plot showing class distribution *Plot 2 - PCA plot after clustering*

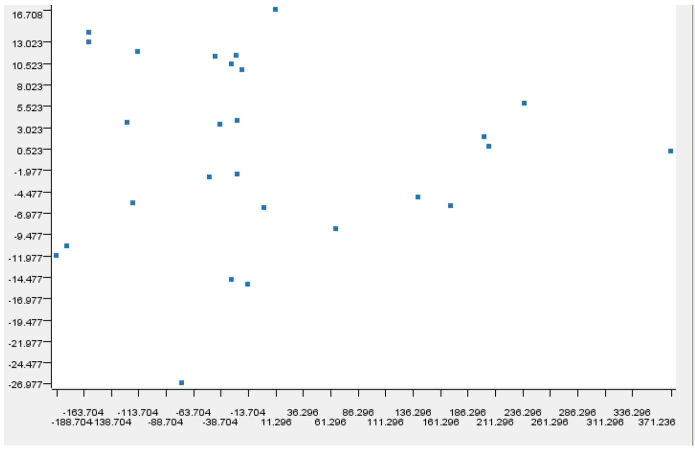
With 169 correctly classified results, and an accuracy of 94.4%, the plots of the normalised data show a much greater distinction between cultivars. Each class is clearly found within its own cluster, unlike without normalisation in which the second and third cultivar wines were found to intermix greatly within the left-most cluster. The normalised results show that the cultivars are compositionally distinct from each other; while an alternate wine may be sought from another in the cluster, it will most certainly be of the same cultivar, unlike what is shown in the first two plots.

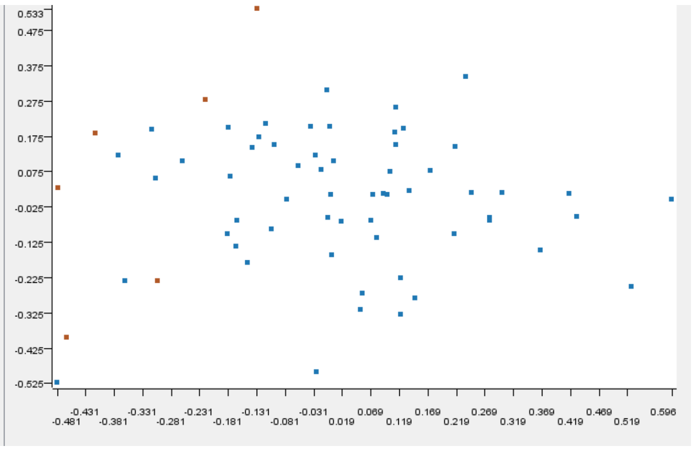




Plot 3 - PCA plot showing class distribution (normalised) *Plot 4 - PCA plot after clustering (normalised)*

The PCA plots for the first cultivar clearly show the effects of normalisation. Although the normalised cluster contains 6 data points from cultivar two, it contains all 59 data points of the first cultivar, compared to just 27 without normalisation. Furthermore, the data points in Plot 6 are more densely clustered, showing a greater similarity between the wines than found in Plot 5.

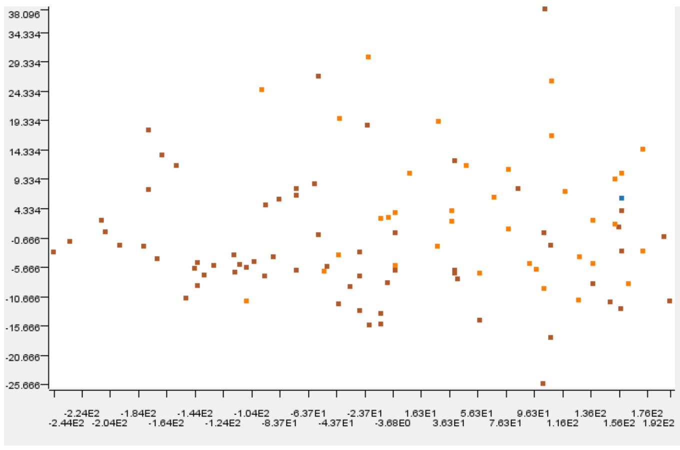


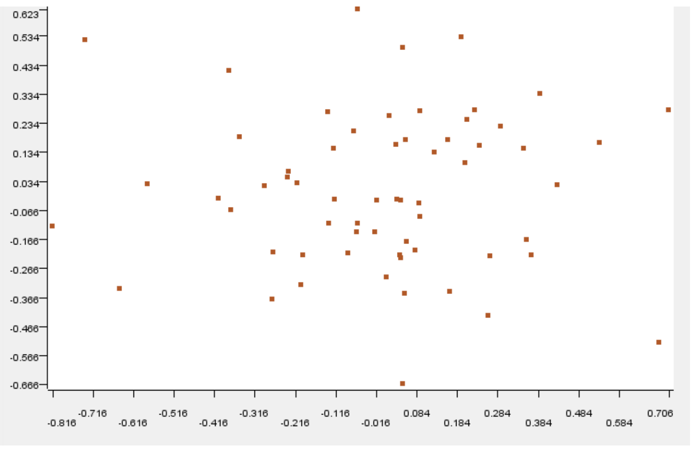


Plot 5 - PCA plot for Cultivar 1 cluster

Plot 6 - PCA plot for Cultivar 1 cluster (normalised)

The effects of normalisation are also clearly shown by the PCA plots for the second cultivar. The normalised cluster contains 62 of the 71 data points for cultivar two. While there are 64 in the cluster without normalisation, as well as 37 from the third cultivar and 1 from the first.





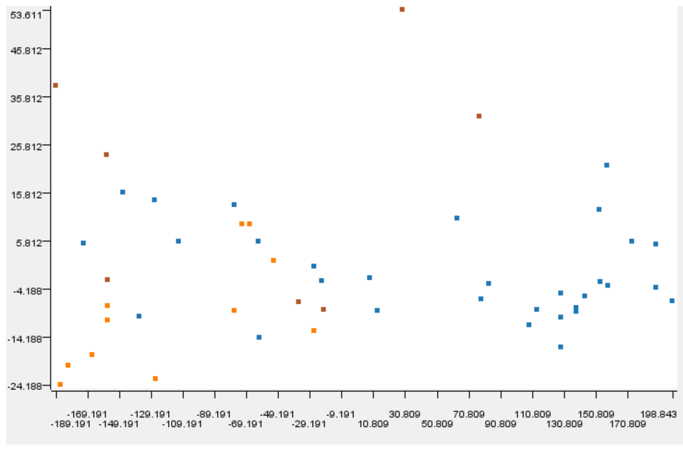
Plot 7 - PCA plot for Cultivar 2 cluster

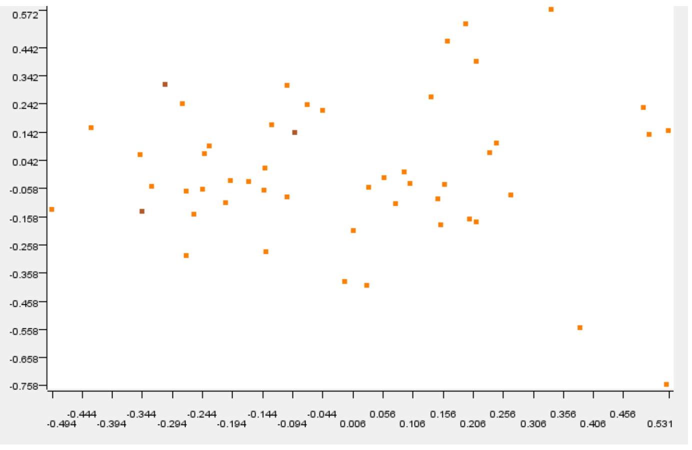
Plot 8 - PCA plot for Cultivar 2 cluster (normalised)

The plots for the third and final cultivar are much the same. All 48 of the third cultivar data points are found within the normalised cluster, compared to only 11 without normalisation of the data.

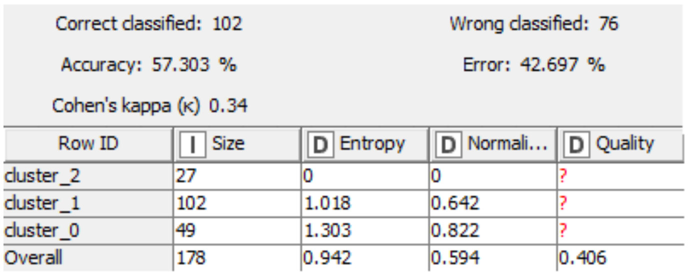
Plot 9 - PCA plot for Cultivar 3 cluster

Plot 10 - PCA plot for Cultivar 3 cluster (normalised)





Conclusion from Results

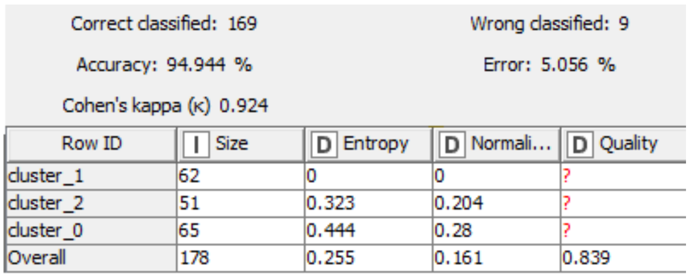

Correct classified: 102 Wrong classified: 76

Accuracy: 57.303 % Error: 42.697 %

Cohen's kappa (κ) 0.34

Row ID	Size	Entropy	Normali...	Quality
cluster_2	27	0	0	?
cluster_1	102	1.018	0.642	?
cluster_0	49	1.303	0.822	?
Overall	178	0.942	0.594	0.406

Results without normalisation


Correct classified: 169 Wrong classified: 9

Accuracy: 94.944 % Error: 5.056 %

Cohen's kappa (κ) 0.924

Row ID	Size	Entropy	Normali...	Quality
cluster_1	62	0	0	?
cluster_2	51	0.323	0.204	?
cluster_0	65	0.444	0.28	?
Overall	178	0.255	0.161	0.839

Results with normalisation

Overall, the K-Means algorithm produced a more accurate clustering with normalisation of the data than without. The normalised data has a normalised entropy value of 0.161, indicating low uncertainty in the results. On the other hand, the regular data has a normalised entropy value of 0.594, indicating moderate uncertainty in the predictions.

The normalised data produced a Cohen's Kappa of 0.924, indicating that the agreement between the true class values and the predicted clusters is not down to chance, but instead the strong accuracy of the classification. The Cohen's Kappa score for the regular data however is only 0.34, indicating only a fair level of agreement, with a moderate likelihood of predicting a correct result by chance .

Task 2 - Comparison of Classification Models

Description of the two Classification algorithms adopted

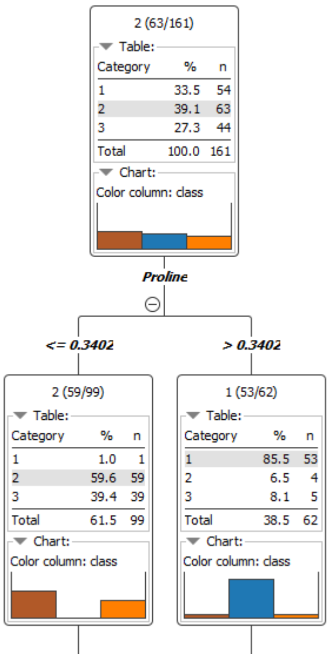
The first classification algorithm used is the Decision Tree learner. Decision Trees 'predict the class or value of the target variable by learning simple decision rules inferred from training data' [4]. The training of the Decision Tree model follows three key steps [5]:

1. The best attribute of the dataset is chosen as the root of the tree.
2. The training set is then split into subsets based upon their attribute values.
3. Repeat step 1 and 2 on each subset until leaf nodes are created in all branches of the tree.

With the 'wine.csv' dataset, the 'Proline' variable is chosen as the root attribute. The data points are then split up depending on whether they are less than or equal to 0.3402, or greater than 0.3403. This process is repeated for each of the other variables until all data points have been assigned a class.

An advantage of the decision tree algorithm is that easily interpreted, with clear conditions for classification. Decisions Trees also do not require scaling or pre-handling of missing data before constructing the model.

A disadvantage of Decision Trees is the potential for overfitting, with results stuck in local minimas. Furthermore, Decision Trees can struggle with prediction when using large and complex data sets, leadings to potential inaccuracies and long execution times [6].



2 (63/161)

Category	%	n
1	33.5	54
2	39.1	63
3	27.3	44
Total	100.0	161

2 (59/99)

Category	%	n
1	1.0	1
2	59.6	59
3	39.4	39
Total	61.5	99

1 (53/62)

Category	%	n
1	85.5	53
2	6.5	4
3	8.1	5
Total	38.5	62

Root and first branches of wine.csv decision tree

The second classification algorithm used is the Naive Bayes learner. The principle of the Naive Bayes classifier is that 'the presence of a particular feature in a class is unrelated to the presence of any other feature' [7]. Using the Bayes Theorem, the class with the highest posterior probability is the outcome of the prediction.

$$P(c|x) = \frac{P(x|c)\,P(c)}{P(x)}$$

P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes)
P(c) is the prior probability of class
P(x|c) is the likelihood which is the probability of predictor given class
P(x) is the prior probability of predictor

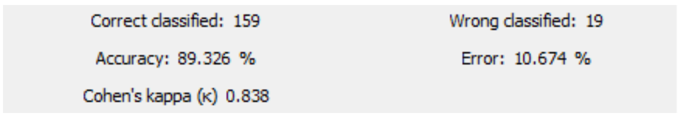
An advantage of the Naive Bayes classifier is that it is easy to implement given only the probability is to be calculated. Additionally, the classifier works well for data sets with a high dimensionality. If the independent assumption does not hold true however, the performance of the classifier will be very low [8].

Description of the 10-fold cross-validation method

In 10-fold cross-validation, the original data set is randomly partitioned into 10 equally sized subsamples. 1 subsample is used as the test data and the other 9 for training the classifier. After classification, the test subsample is switched out for one of the 9 training subsamples. This process is repeated until every set has been predicted on once, at which point the results are averaged.

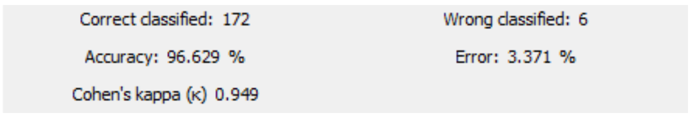
An advantage of K-fold cross-validation is that it works well on both small and large data sets. Moreover, all of the data is used in testing the model, giving a fair and well-rounded evaluation metric. Since the data set must be split up many times, and given the repetition of the process, the computing power can be high. Additionally, it may take a long time to process large data sets, making the discovery of optimal hyper parameters a potentially lengthy process [9].

Experimental Results



Correct classified: 159 Wrong classified: 19
Accuracy: 89.326 % Error: 10.674 %
Cohen's kappa (κ) 0.838

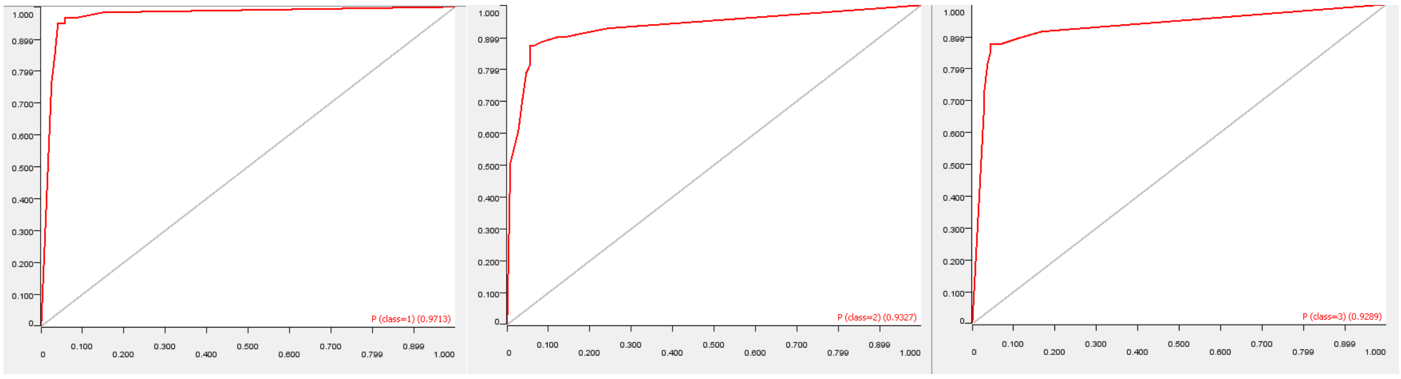
Decision tree classification accuracy



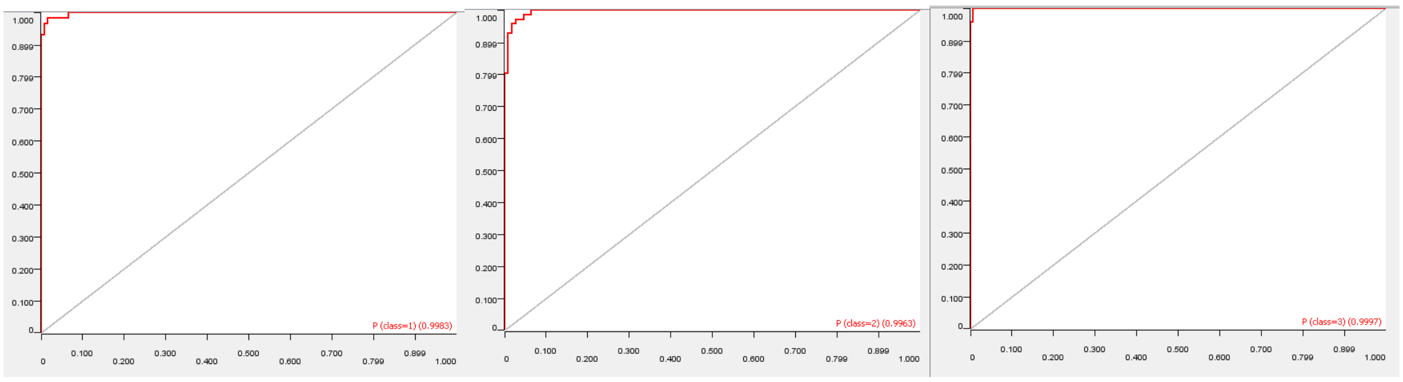
Correct classified: 172 Wrong classified: 6
Accuracy: 96.629 % Error: 3.371 %
Cohen's kappa (κ) 0.949

Naive Bayes classification accuracy

The Naive Bayes classifier was able to predict the class labels with greater accuracy than the Decision Tree classifier; Naives Bayes achieved an accuracy of 96.63% compared the Decision Tree classifier's 89.33%. Both achieved a high Cohen's Kappa, indicating little chance in correct prediction making.

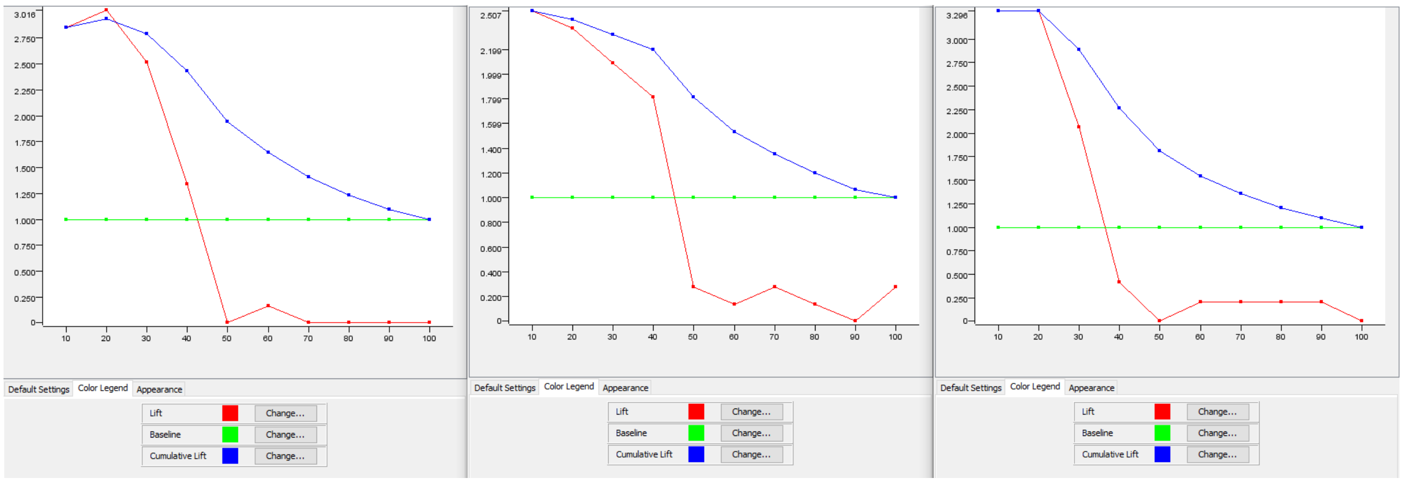


ROC curves for Decision Tree classifier

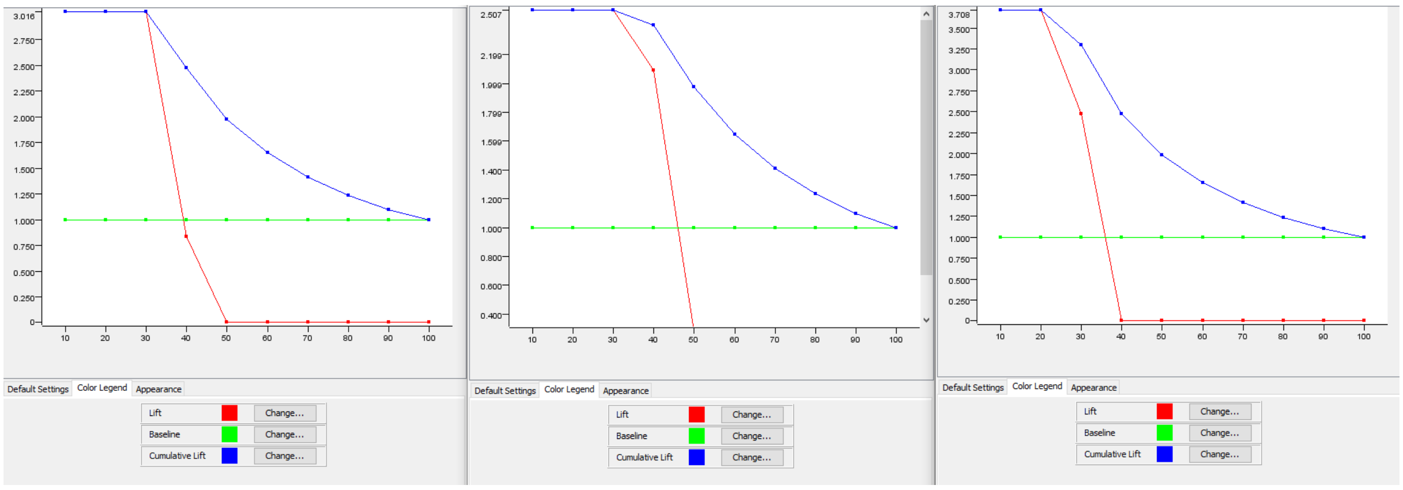


ROC curves for Naive Bayes classifier

The ROC curves show that the probability to correctly predict each class is high for both classifiers. A value close to 1 indicates a strong ability to predict each class. With a greater accuracy, the Naive Bayes classifier naturally has a greater ability to predict each class compared to the Decision Tree classifier.



Lift graph for Decision Tree classifier



Lift graph for Naive Bayes classifier

The lift graphs also reflect the Naive Bayes classifier’s greater performance.

Conclusion

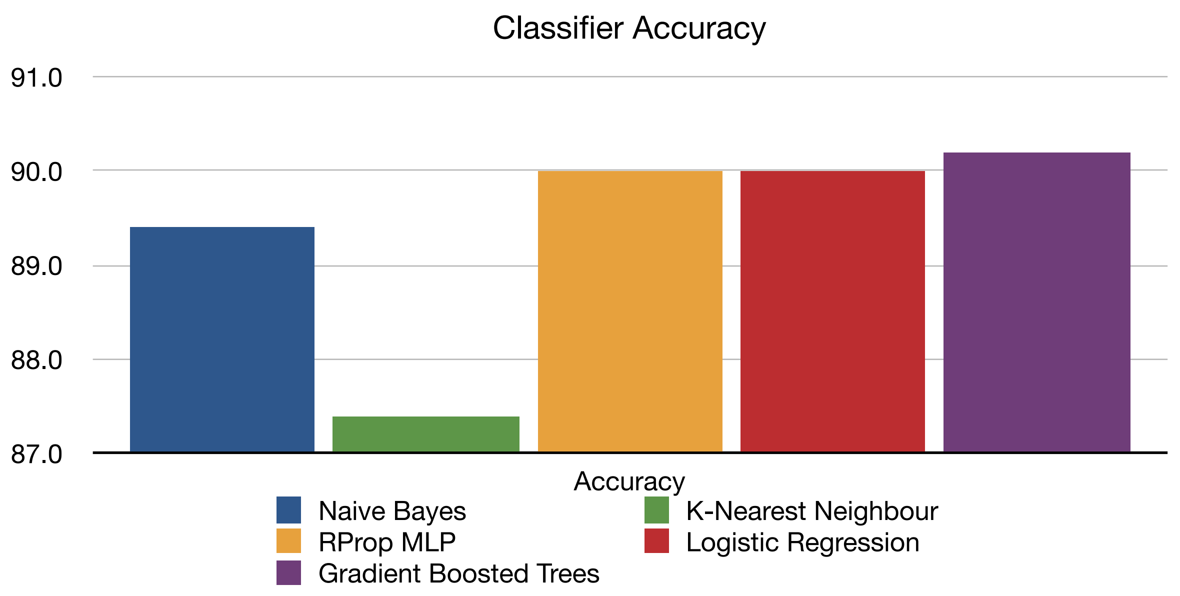
Though both classifiers predicted the class labels for the wine dataset with a high accuracy, the Naive Bayes classifier performed better than the Decision Tree classifier. The Naive Bayes classifier had a higher probability of successfully predicting each class, as shown by the ROC curves.

Task 3 - Data Science Challenge

Description of the data mining algorithm, the workflow, and the predicted performance indices.

To find out the most appropriate classifier to use, the training set was partitioned and used for training and testing with a variety of classifiers. As the training set had 100,000 rows, and the final test set 1000, the partitioning was done on a 99:1 ratio.

The Gradient Boosted Trees classifier was found to be the most accurate of the classifiers tested. The Gradient Boosted Decision Tree works using three main elements [10]:

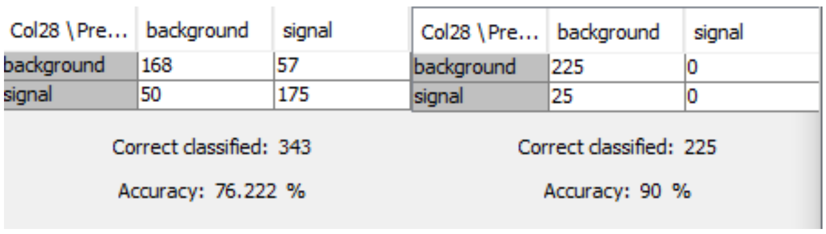


Classifier	Naive Bayes	K-Nearest Neighbour	RProp MLP	Logistic Regression	Gradient Boosted Trees
Accuracy	89.4	87.4	90.0	90.0	90.2

1. The loss function to be optimised
2. The weak learner to make predictions
3. The additive model to add weak learners to minimise the loss function

As an advanced model specifically designed for classification, it's no surprise the Gradient Boosted Decision Tree was the most successful classifier; an accuracy of 90.2% is misleading however. The training data contains significantly more entries for background particles than signal, as such, the classifier can predict all the particles as background and still achieve a high accuracy.

To amend this problem, the training data can be resampled using the SMOTE node to increase the number of signal particle entries. So the SMOTE node will resample the data within an acceptable amount of time, the data is reduced to 25% of its original size. Though the Gradient Boosted Tree classifier only achieved an accuracy of 76.22% with resampling, it correctly identified many signal particles, unlike without resampling in which all particles were labelled as background particles.



Col28 \ Pre...	background	signal
background	168	57
signal	50	175

Correct classified: 343

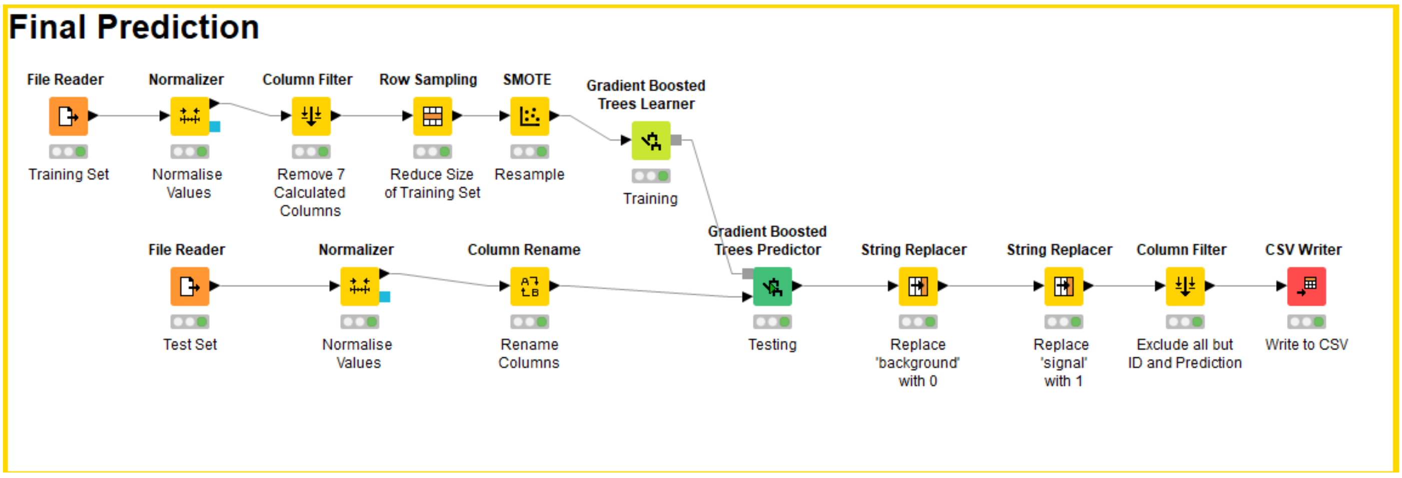
Accuracy: 76.222 %

Col28 \ Pre...	background	signal
background	225	0
signal	25	0

Correct classified: 225

Accuracy: 90 %

Classification accuracy with Smote resampling versus without



Final workflow to predict signal and background particles

The final workflow trains the Gradient Boosted Trees learner with the resampled training set. The predictor node then predicts the classes for the test set. The predicted class strings are then replaced with binary values, and are written to a csv file with the sample IDs.

Conclusion

Through comparison of classifier accuracy, an appropriate model was chosen for the final prediction. The data was resampled to account for the imbalance in background and signal particle entries within the data set.

Bibliography

Task 1

1. Dr.Michael J. Garbade Understanding K-Means Clustering in Machine Learning 12/09/18
2. Jeffrey Girard Explain Cohen's kappa in the simplest way 19/03/17
3. Victor Powell Principal Component Analysis Explained Visually

Task 2

4. Nagesh Singh Chauhan Decision Tree Algorithm, Explained
5. Rajul Saxena How Decision Tree Algorithm Works 30/01/17
6. Holy Python Decision Tree Pros and Cons
7. Sunil Ray 6 Easy Steps to Learn Naive Bayes Algorithm with codes in Python and R 11/09/17
8. Anuuz Soni Pros and Cons of Naive Bayes Classifier :- 04/07/20
9. Anber Arif Cross Validation in Machine Learning 03/12/20

Task 3

10. Jason Brownlee A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning 15/08/20