COSC2670: Assignment 2

Using Various Data Modeling Techniques to Investigate the Multi Variable Dependence of Wine Quality

Monty Sefer, s3784357

School of Science, RMIT University, Melbourne, VIC 3000, Australia

October 2023

1 Data Preparation and Regression Analysis

1.1 Data Preparation

The data set given features a number of variables such as sugar content, alcohol percentage and pH relating to the quality of a given wine graded from one to ten. In addition, there are also a a number of errors present within the data set including: null values, white space and string cells mixed into numerical columns. Any data point with the such errors was dropped from the set and analysis.

Following weak model performance, a decision was made to address outliers for all variables other than quality. This was done using the inter quartile range (IQR) method with a threshold of 1.5. That is any value greater than $Q3 + 1.5 \times IQR$ or less than $Q1 - 1.5 \times IQR$ being omitted. This threshold is easily adjustable within the code.

From this cleaned data a random sample of 600 values is taken according to an adjustable seed. This random sample is saved by the code as 'A2RandomSample.csv' and recalled as the data frame 'wine'. All subsequent tasks within the code use this data frame and thus the csv file. It is however possible to run any task independently from another provided the package cell and this task, code section 1.1 are ran first.

1.2 Linear Regression

The relationship between wine density and alcohol percentage has been investigated using a linear regression model. I wrote multiple coded methods to achieve this, the superior of which trains on 75% of the data and is tested against the remaining 25%. That is to say this model predicts the value of alcohol percentage for any given density according to a linear relationship. Figure 1 illustrates the data split by showing testing data in blue and training data in red.

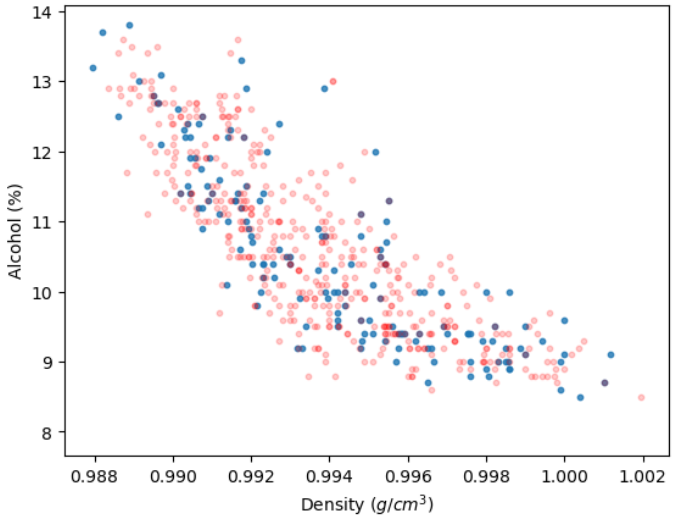

Figure 1: Plot of alcohol percentage as a function of density. Visually demonstrates sample split by presenting testing data in blue and training in red. Generated with seed = 12345.

Figure 2 shows the fit of the model to the testing data. The equation of this fit is given by $y = -356x + 364$, where y is the predicted alcohol percentage and x is the density in grams per cm^3. The R^2 value of .689 indicates substantial linearity in the relationship between alcohol and density. Given that alcohol is less dense than water, it is logically consistent that as the density increases percentage of alcohol in the wine should decrease. This is exactly what is observed and modelled in the data reinforcing the accuracy of both the data and linear regression modeling method.

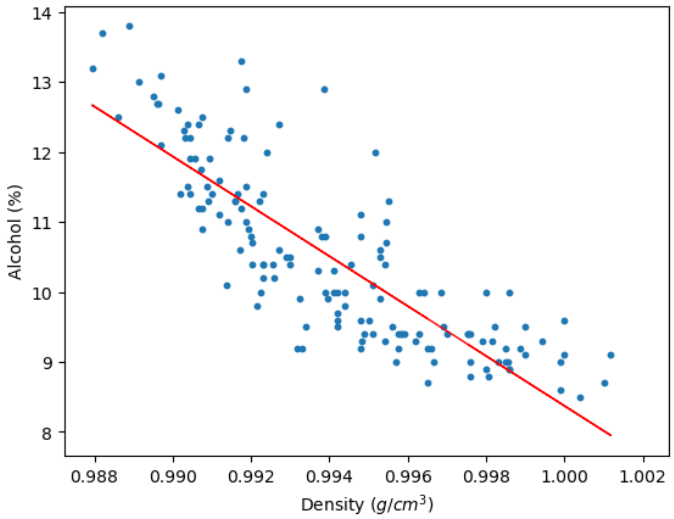

Figure 2: Plot of alcohol percentage as a function of density. Testing data is fitted in blue with linear regression model in red. The equation of the trend line is given by $y = -356x + 364$ with a coefficient of determination, $R^2 = 0.689$. Generated with seed = 12345.

1.3 Box Plot Comparison

The side by side boxplot in figure 3 is used for a rudimentary analysis of the relationship between alcohol percentage and wine quality. It is generally observed that the quality level of wines increase with alcohol percentage on average. This trend is not observed in quality levels 3 or 4, however the former has far too few data points for any analysis. Figure 3 itself is not grounds for any meaningful analysis or strong conclusions to be made. On its own and out of context this plot could mislead one to conclude moonshine or high proof spirits to be the highest quality wines. This speaks for the need of multi variable modelling techniques as wine quality is dependant on far more than alcohol percentage alone. At best this figure gives insight into the range of alcohol percentages present in each quality grade of wine.

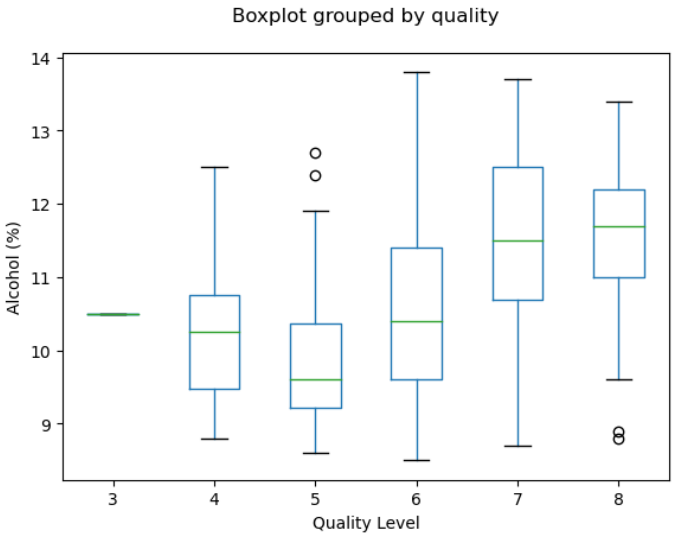

Figure 3: Side by side box plot of the alcohol percentage grouped by the quality level

2 K-Nearest Neighbour Model

2.1 Training and Evaluation

A k-NN algorithm had been applied to the wine data set. The parameters utilized are mostly default with exception of the k value which has been set to $k = 21$. This is in keeping with the general rule of $k = \sqrt{N}$, in this case $k = \sqrt{(600 \times .85)} = 21.21 \approx 21$. In order to evaluate the performance of this model a confusion matrix heat map as well as accuracy, precision and f-score have been recorded and analysed.

The metrics presented in table 1 generally indicate that the k-NN model is quite weak and unreliable. Accuracy isn't a good evaluation metric due to the imbalance in classes for the data. Precision and by extension f-score are better due to our models likelihood to assign lots of true positive and false positive. Either way all of these values indicate the model is weak. Generally, half or all predicted qualities are incorrect.

Table 1: Accuracy, precision and f-score metrics for k-NN model evaluation

Accuracy	**Precision**	**F-Score**
52.0%	53.5%	48.8%

Perhaps the best measure for model performance is the confusion matrix's presented in figure 4. The normalized matrix clearly shows the models tendency to predict a quality of 6 regardless of the true value or input variables. I believe this to be due to the subjective nature of 'quality'. It may be that there is not a strong relationship between the data set variables and the subjective

nature of variable, quality. This paired with 6 having the most data points may be causing the model to preferentially predict a quality 6 even where incorrect.

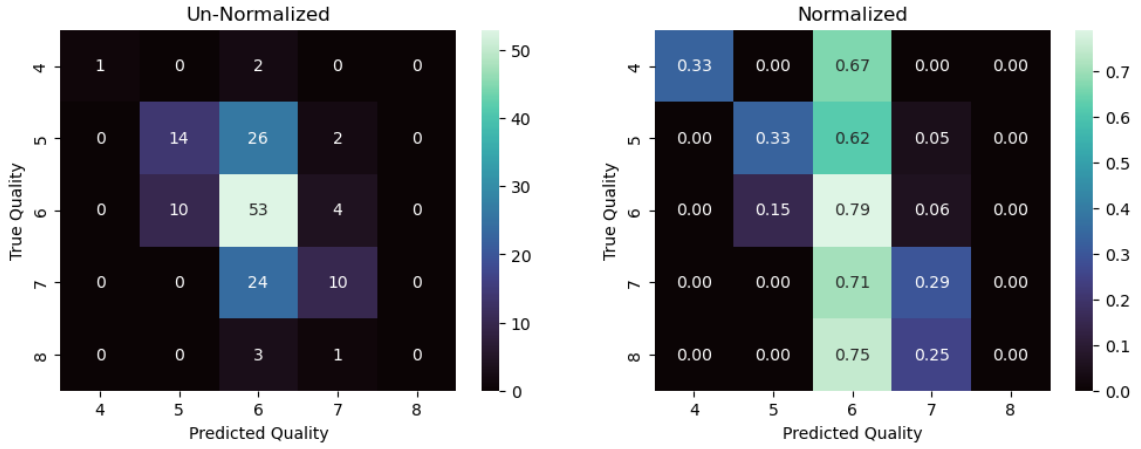

Figure 4: Normalized and un-normalized confusion matrix of true and predicted wine qualities. Generated with seed = 12345.

2.2 Optimizing k-value

I wrote a number of methods for determining the best value for the number of neighbors the model should check before classification (k-value). The best of which models k-NN for a range of k values and sample seeds, averaging and plotting the accuracy, precision and f-score for each k-value. Only taking one measurement for each k value as in code section 2.2.3 had incredible volatility and no meaningful conclusion could be drawn about k as the conclusion would be wildly different every time the code was run. Taking the average of a many runs addresses this and is achieved in both code sections 2.2.2 and 2.2.1, with 2.2.1 being more optimized and faster.

The results of running code section 2.2.1 can be observed in figure 5. The f-score is the best metric for evaluating the k-NN model of the three present. It can be seen that as k increases f-score rapidly decreases. This is likely due to the data under fitting and thus performing very poorly on test data at high k values. When k is equal to one the f-score is the highest. What is occurring here is over fitting, the model is performing well on test data but would not be strong when applied to new data. Thus, the value of $k = 10$ was used to avoid both under fitting and over fitting while ensuring a high f-score.

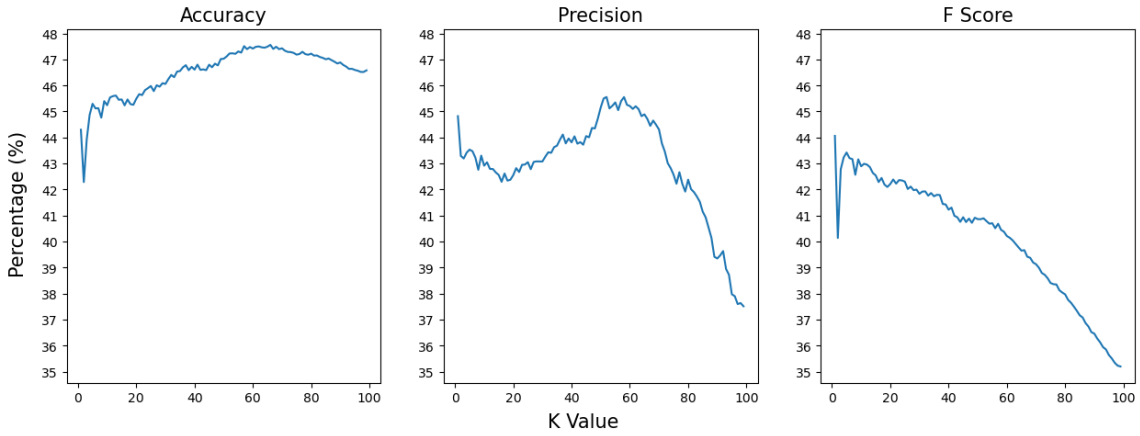

Figure 5: Accuracy, precision and f-score for k values ranging 1 to 100, sampled 300 times each and averaged. Generated with seed = 12345.

2.3 Optimizing Train/Test Split

Code section 2.2.2 was modified in order to iterate over different train/test data splits rather than k values and using the static k value of 10 found in section 2.2. The relationship between evaluation metrics and test/train split are presented in figure 6. For all metrics it is clear that 20% test and 80% train is optimal. That is to say having 20% allocated as testing data is best as it carries the highest percentage across all metrics indicating the strongest model.

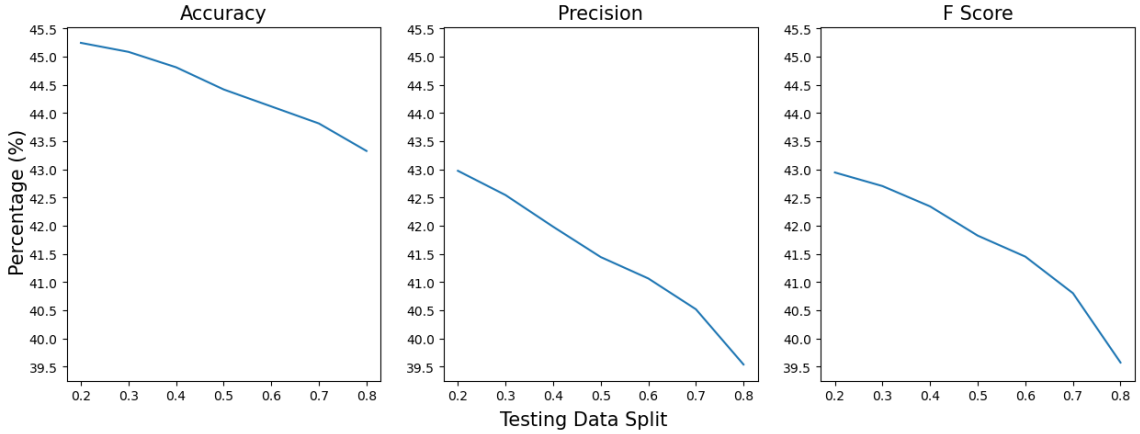

Figure 6: Accuracy, precision and f-score for test/train splits ranging 20% to 80%, sampled 300 times each and averaged when k = 10. Generated with seed = 12345.

3 K-means Clustering Model

3.1 Training

Code section 3.1 is responsible for fitting the k-means clustering model to the wine data. All parameters except the number of clusters (k value) were left at default as other values didn't provide any improvement and in some cases caused longer processing times. The number of clusters was set to 6 to match the number of unique wine qualities present in this seed. In theory, if a strong correlation between variables and quality exists then each cluster should correspond directly to a unique quality grade. This unfortunately was not the case as the model is quite weak and there is not a strong relationship between the tested variables and the subjective measure quality. In addition to this, the very low count of quality grades 3, 4, 8 and 9 also hinders model performance.

3.2 Optimizing k-value

One method for finding the optimal cluster number in k-means clustering is known as the elbow method. This method visualises the point at which increasing k value has diminishing returns on minimizing the distance between centroids. Code section 3.2 does this and the result is figure 7. It is clear that the average distance decreases rapidly till around k = 6 after which decreases in centroid distance diminish. This means that the k value in use in section 3.1 is optimal because it doesn't needlessly increase processing time, attains a low average distance to centroids and matches the number of clusters to the number of unique wine qualities.

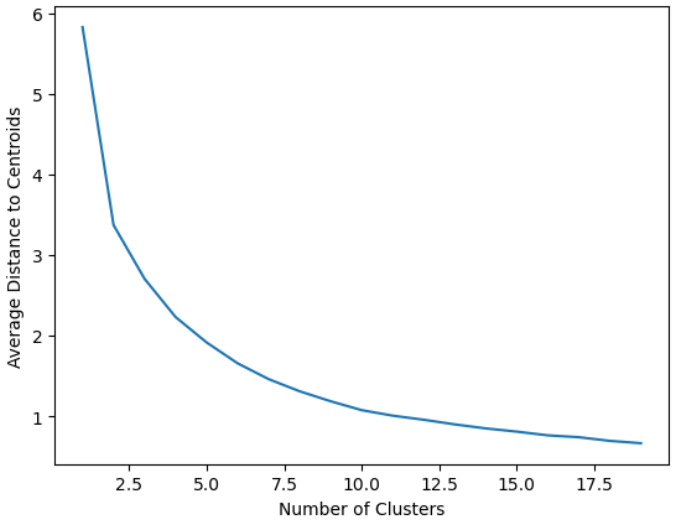

Figure 7: Average centroid distance as a function of k value.

3.3 Confusion Matrix

In order to evaluate the meaning and performance of the cluster model a confusion matrix has been generated. This allows a comparison of target quality and predicted cluster labels. Unfortunately,

the same issue with the k-NN model is present here with the k-Means clustering model. While to a lesser degree, the model is still over predicting the wine quality, 6. The model fails to predict any target quality aside from 6 with any meaningful accuracy. Even then, predictions are muddied by a massive number of false positive quality 6 predictions. This model would not be suitable for predicting a wines quality. This is unlikely to be due to the approach but rather the subjective nature of the variable quality.

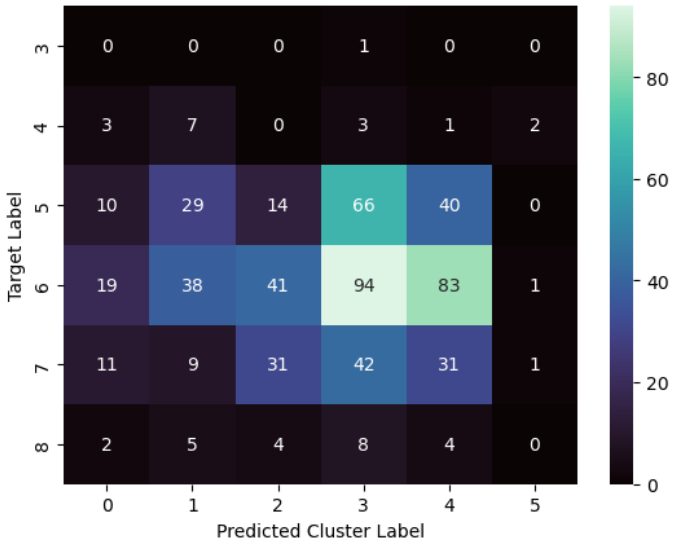

Figure 8: Confusion matrix for k-means clustering model. Generated with k = 6 and seed = 1234567

References

[Cournapeau, 2023] Cournapeau, D. (2023). scikit-learn: machine learning in Python — scikit-learn 1.3.2 documentation. [Online; accessed 27. Oct. 2023].

[McKinney, 2023] McKinney, W. (2023). Pandas Documentation — pandas 2.1.0 documentation. https://pandas.pydata.org/docs/index.html.

[Xia, 2023] Xia, F. (2023). Course modules: Practical Data Science with Python (2350). https://rmit.instructure.com/courses/107387/modules.