Abstract

The objective of this study is to explore useful learning models to predict red and white wine quality. First general patterns were analysed; when analysing distribution of quality most scored 5 and 6 dropping off either side, white wines are perceived as generally higher quality. There is a positive correlation between quality and alcohol content in both wines and no significant correlation between quality and sweetness. When analysing correlations, density had a high correlation with alcohol and residual sugar so this was removed for learning, residual Sugar and free sulfur dioxide were also removed for low correlation with quality.

Regarding models, random forest classifier was the most successful classification model with the highest f1 and k-fold scores, however logistic regression performed only a little worse and is less prone to overfitting. The regression models had very low accuracy so would be unsuitable to use however it may be an unsuitable model type in the first place as the outcome set is not highly suitable for treating regressively and there are very few data points for some qualities. Similar problems were found with predicting all qualities as the lack of data points meant low accuracy despite SMOTE. Predicting all quality outcomes proved inaccurate as well especially with low data sample values.

Google Collab link:
https://colab.research.google.com/drive/1IOr_Ca9Em1iAmhdYdguoymE3S7LPf42Z?usp=sharing

What was done and how

Quality Distribution

The first task was to describe the distribution of wine quality across samples. I firstly plotted the distribution separately side by side using subplots and a first glance showed that for both wines the vast majority of wines were classified between 5 and 7. In Red wine qualities 5 and 6 dominated, which then dropped down to about ⅓ as much at 7 and only a few at 8, whilst in white there was nearly twice as much at 6 than 5 before dropping down to 7, 8 and even a few 9s which red didn’t have. I then plotted red wines and white wines in the same graph showing quality as well as normalising the data as there are far more white data points than red. This showed the same information as before but perhaps more clearly, it seems like Red wines are generally centered around the 5 and 6 mark whilst Whites are dominating the 6 to 8 quality marks .In general white wines are more heavily distributed around higher qualities than red.

Alcohol Content vs Quality

I used standard deviations and the pandas cut function to add new columns to both data frames with the alcohol content discretized into L,M and H. Before assessing the data using the new variable I analysed alcohol against quality. When plotting them against each other in the combined dataset there is a clear positive correlation with an R value of 0.44 when running the Pearson's test and an extremely small P value (1.49 $^{-312}$) showing high statistical significance (under 0.05). When observing discretized alcohol content we see in both there are very few high alcohol content wines below quality 6, we also see from quality 4 upwards low alcohol content wines have less and less representation. High alcohol wines dominate the distribution in quality 7 and above in white wine.

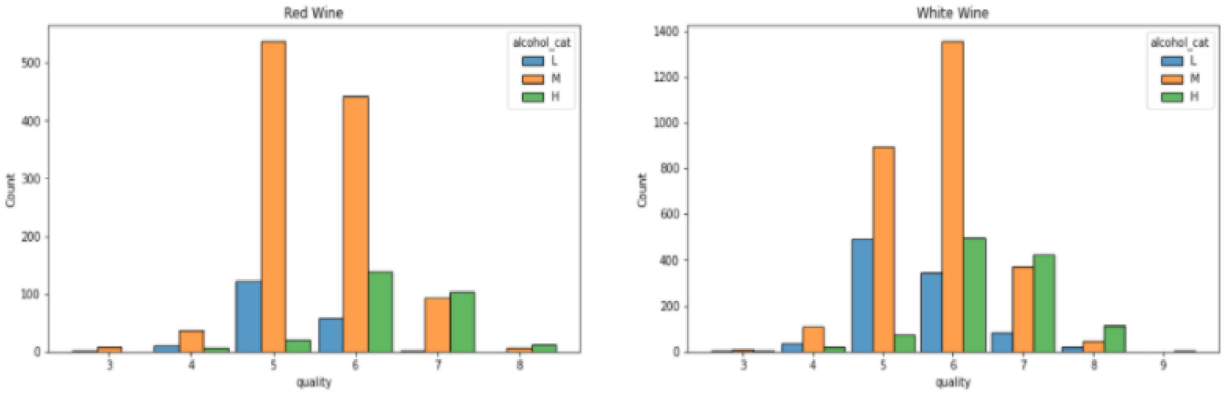


Figure 8: Distribution of wine quality with alcohol content

Sweetness vs Quality

When plotting the distribution of residual sugar in the wines on a histogram it was clear that white wines have wider distribution with many more wines at a higher level of residual sugar whilst the majority of red wines were clustered around the 2.5 mark. I normalised the data so that you could clearly compare the two on the same graph.

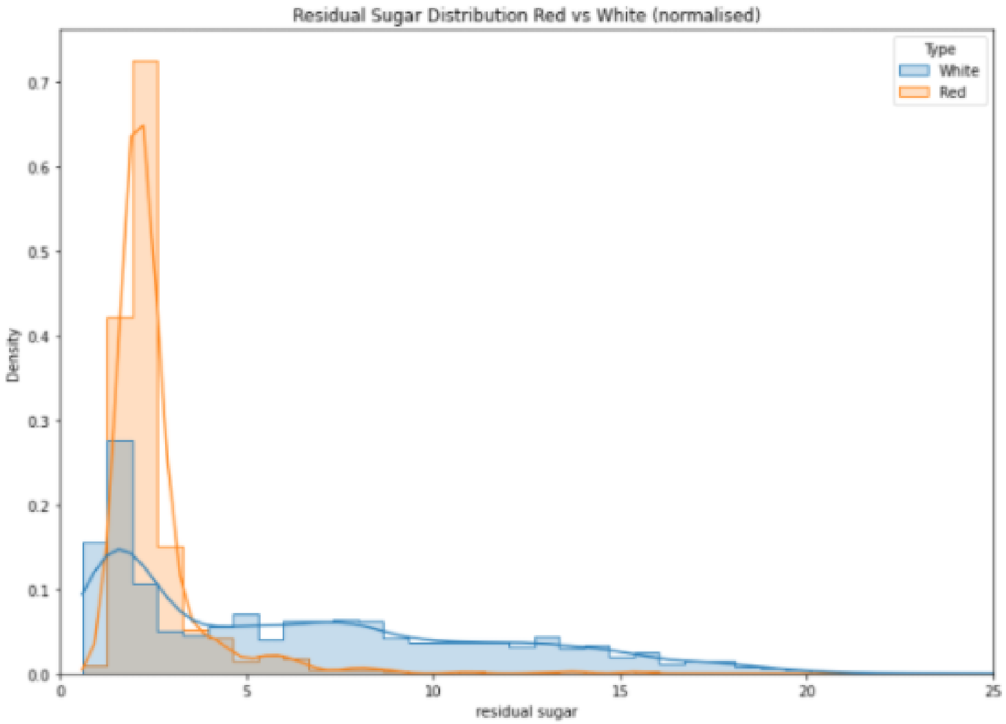
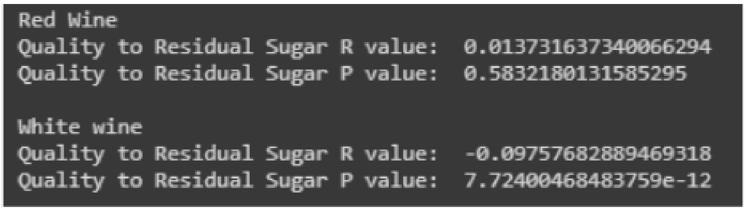


Figure 9: Distribution of wine quality with alcohol content

The residual sugar variable was discretized into either sweet or dry with the mean of each dataset being the threshold. Before using the isSweet variable to analyse residual sugar to quality, by just plotting quality vs residual sugar on a scatter graph you can see there is no obvious correlation between the two. When performing a Pearson test to analyse this, there is no significant correlation in red wine (P-value<0.05) however there is in white wine with high statistical significance and an R value of -0.097 meaning high quality white wines tend to be dryer.



```
Red Wine
Quality to Residual Sugar R value:  0.013731637340066294
Quality to Residual Sugar P value:  0.5832180131585295

White wine
Quality to Residual Sugar R value:  -0.09757682889469318
Quality to Residual Sugar P value:  7.72400468483759e-12
```

Figure 1: Pearsons results for quality to residual sugar

This can also be seen from the bar plots of the discretized residual sugar values where the dry bar is slightly higher. I conclude that there is a slight negative correlation between quality and sweetness in white wine but nothing significant in red wine.

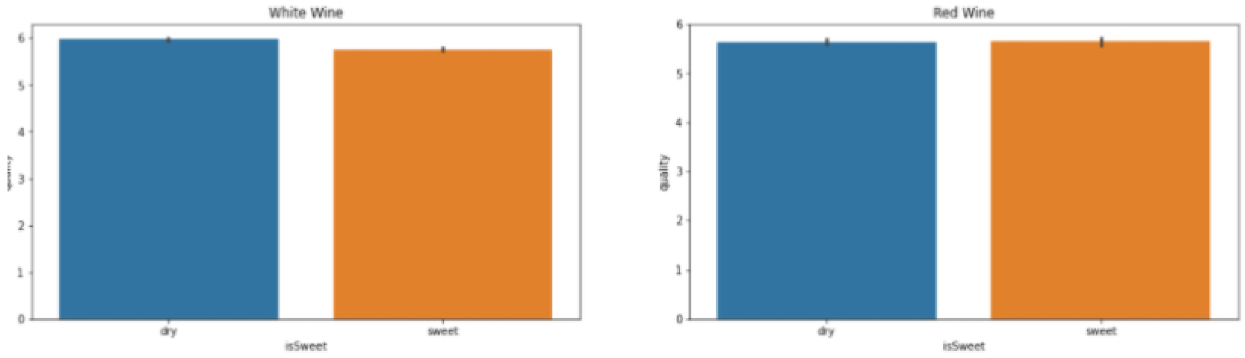


Figure 2: Average quality for dry and sweet wines

Analysing Correlations

To get a feel for the correlations in the dataset I printed a pairplot to show scatterplots of all the combinations of attributes for both wines. A few correlations stuck out at first for example a clear negative correlation between fixed acidity and pH in both wines, a positive correlation between residual sugar and density especially in white wine and a positive correlation between fixed acidity and density. Although this may have been a good initial view of the correlations it is confusing to take it all in and some less obvious correlations are easy to miss so a correlation matrix can give a more in depth correlation analysis.

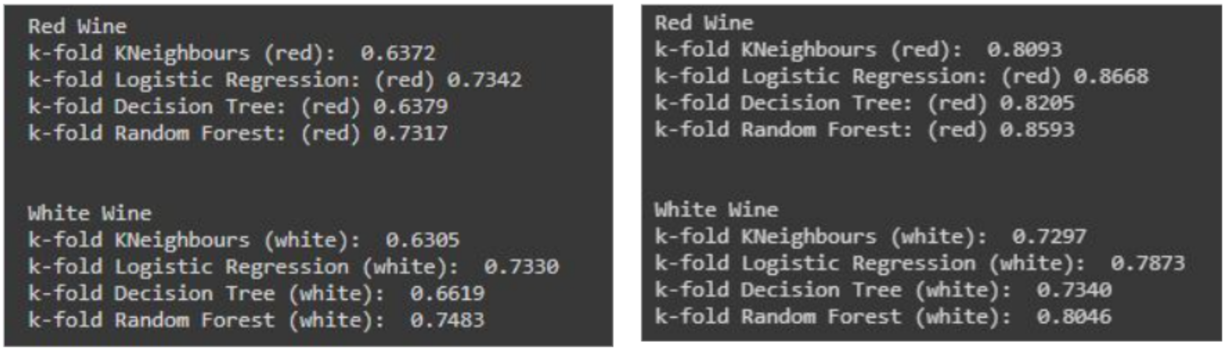
For the matrices I used pandas corr function and seaborns heatmaps to visualize. I first decided which correlation method was most appropriate (Pearsons vs Kendall vs Spearman). When researching Kendall it seemed similar to Spearmans although works best when your sample size is quite small and there are many tied ranks, which isn't really the case as there is a good amount of data and most of the attributes are on a continuous scale and do not contain many tied ranks. In most scenarios and unless you have a good reason to use Kendall it seems to be best to use spearmans or pearsons. When running the kendall matrix it mostly followed the pattern of the spearman's matrices except each R value was smaller although there may be some subtle differences.

Pearsons seemed more useful when examining correlation between two continuous variables whilst spearmans ranks the data and uses this to find correlations rather than using the raw data (A comparison of the Pearson and Spearman correlation methods, n.d.), the data I am comparing in the matrices is all continuous therefore it makes more obvious sense to stick with pearsons. When observing the scatterplots it seems evident that all the plots that have correlations are linear which works well for pearsons, spearmans probably wouldn't be a bad way to correlate the data either as the plots also seem to have mostly monotonic relationships however I conclude that I will use pearsons.

I observed the Pearsons matrix in order to deduce which attributes are most useful for learning. The matrix is not altogether the same when regarding white wine and red wine therefore it would make sense to analyse them separately as the machine learning models will be trained separately for white and red. In white wine there is a significant negative correlation between density and alcohol, and alcohol has a high positive correlation with quality. Since using two variables which are highly correlated with each other to predict outcome is redundant I will remove the density column. Also since residual sugar in red wine has no significant correlation to quality I will remove this as well. Free sulfur dioxide has no significant correlation to quality in white wine therefore I will remove this.

Quality Threshold

Once the initial analysis was complete I moved on to machine learning. Firstly for considering the model as a classification problem I had to discretise the quality outcome into low and high. After trying a few thresholds I settled on using (LQ<6 and HQ=>6). When using 7 as the threshold the LQ to HQ ratio was extremely skewed especially in red wine which meant that SMOTE was done to a higher degree to generate more HQ samples, although this didn't seem to affect the accuracy of the models as they were significantly higher than using 6 as the threshold when using the models on the test set *(Figure 3)* but it did mean that the vast majority of wines in the dataframe were then categories as 'LQ' which didn't seem right. The selection of the threshold is also somewhat subjective as what you should define as being a high or low quality wine should probably be up to wine experts, I think it might be more appropriate to have 3 categories with medium quality in the middle.



```
Red Wine
k-fold KNeighbours (red):  0.6372
k-fold Logistic Regression: (red) 0.7342
k-fold Decision Tree: (red) 0.6379
k-fold Random Forest: (red) 0.7317


White Wine
k-fold KNeighbours (white):  0.6305
k-fold Logistic Regression (white):  0.7330
k-fold Decision Tree (white):  0.6619
k-fold Random Forest (white):  0.7483
```

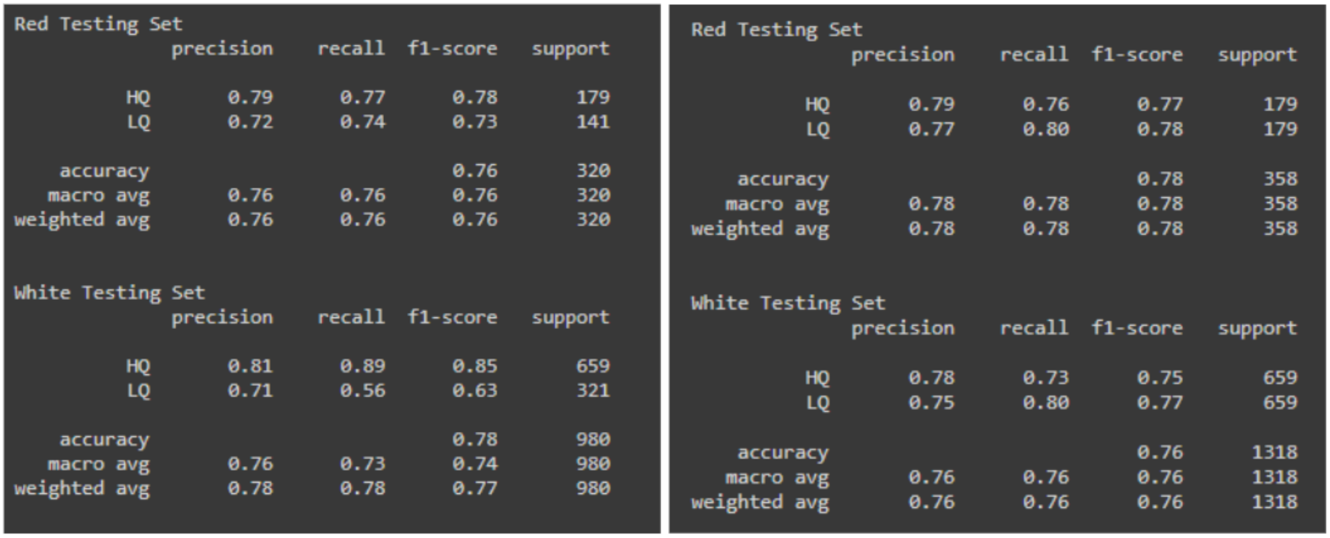
```
Red Wine
k-fold KNeighbours (red):  0.8093
k-fold Logistic Regression: (red) 0.8668
k-fold Decision Tree: (red) 0.8205
k-fold Random Forest: (red) 0.8593


White Wine
k-fold KNeighbours (white):  0.7297
k-fold Logistic Regression (white):  0.7873
k-fold Decision Tree (white):  0.7340
k-fold Random Forest (white):  0.8046
```

Figure 3: K-fold for each model (LQ<6 and HQ=>6 on left) (LQ<7 and HQ=>7 on right)

Pre processing

Before running the models some preprocessing had to be done. Firstly one hot encoding was used to encode the categorical columns in the data set (alcohol_cat and isSweet), then the data had to be balanced as there were more high quality wines especially in white wine, so SMOTE was used to level this out. Out of interest I ran a KNeighbors model before and after SMOTE to see the difference, initially I ran the models using SMOTE on the training and test sets which led to the SMOTE data being much more accurate but I then found out this was incorrect as you should only oversample the training sets.

As seen in Figure 4 SMOTE increased the accuracy in red wine for KNeighbours by about 2% whilst decreasing accuracy overall in white wine by about 2% however it did increase the LQ accuracy and recall by quite a lot, probably due to there not being as much data before SMOTE. Other models reacted differently to SMOTE. Both the decision tree and random forests had severe drops in accuracy predicting white wine quality after SMOTE. This is probably because Dthey are prone to overfitting which would be made worse through training on a largely oversampled set of data as white wine had a large discrepancy between the number of LQ and HQ wines however for red wine small increases in accuracy were found probably as oversampling was less extreme.



```
Red Testing Set
              precision    recall  f1-score   support

          HQ       0.79      0.77      0.78       179
          LQ       0.72      0.74      0.73       141

    accuracy                           0.76       320
   macro avg       0.76      0.76      0.76       320
weighted avg       0.76      0.76      0.76       320

White Testing Set
              precision    recall  f1-score   support

          HQ       0.81      0.89      0.85       659
          LQ       0.71      0.56      0.63       321

    accuracy                           0.78       980
   macro avg       0.76      0.73      0.74       980
weighted avg       0.78      0.78      0.77       980
```

```
Red Testing Set
              precision    recall  f1-score   support

          HQ       0.79      0.76      0.77       179
          LQ       0.77      0.80      0.78       179

    accuracy                           0.78       358
   macro avg       0.78      0.78      0.78       358
weighted avg       0.78      0.78      0.78       358

White Testing Set
              precision    recall  f1-score   support

          HQ       0.78      0.73      0.75       659
          LQ       0.75      0.80      0.77       659

    accuracy                           0.76      1318
   macro avg       0.76      0.76      0.76      1318
weighted avg       0.76      0.76      0.76      1318
```

Figure 4: Classification reports before (left) and after (right) SMOTE (K-Neighbours)

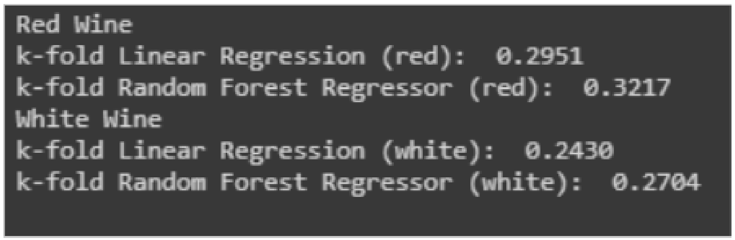
Running the classification models

I chose to try out some of the most common machine learning classification models so after doing some research about the options I picked the four you can see in Figure 3. The Random Forest model seemed to perform the best with any parameters or thresholds I tried so this is potentially the one to go with (Figure 3 left). It performed high on the test set with 74% on white and 73% on red and on the training set achieved 100% accuracy in the f1 score with both wines, this was an indicator of something that came up when researching which is that random forest is prone to overfitting so potentially if the trained model was used on a less similar dataset the accuracy would not be lower. The decision tree model was less accurate which is to be expected as random forest is similar but used as an improvement on decision trees to push out bias and group outcomes based on the most likely positive responses (Fuchs, 2017). Logistic regression could be a good alternative model to use if overfitting became a problem as its accuracy is not that far off from random forest.

Regarding hyper parameter tuning at first I used GridsearchCV to use cross validation scores to find the best combination of parameters for each model and plugged these in manually. This became tedious as whenever I wanted to rerun the models after changing something I had to re enter all the parameters. To solve this I used the same method but stored the best parameters in variables and used them instead.

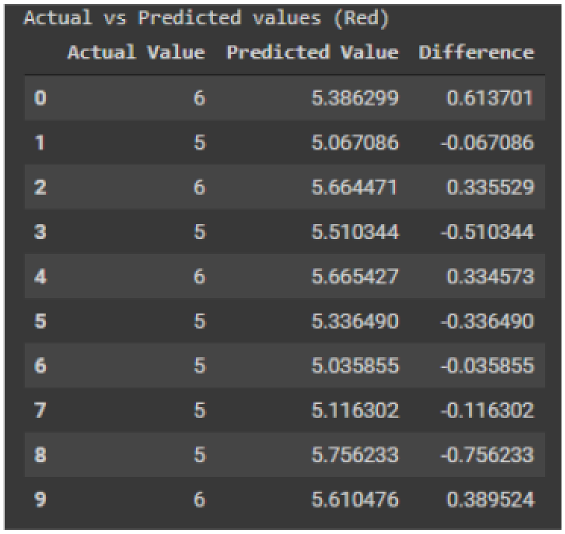
Running the regression models

I then treated it as a regression problem. This seemed to be much less accurate than the classifiers and I don't think this problem is as well suited to regression because the quality is at defined intervals. In order to analyse the accuracy of the models I used MSE and k-fold cross validation. For mean square error as well as K-fold random forest performed better however neither performed very well in k-fold as seen in figure 5. As the models are working regressively they predict values that are not natural numbers which may have played a part in why the models were classified so poorly, you can see this in figure 6. For further research it could be interesting to try rounding the predicted values to the nearest integer to see if this significantly increases accuracy.



```
Red Wine
k-fold Linear Regression (red):  0.2951
k-fold Random Forest Regressor (red):  0.3217
White Wine
k-fold Linear Regression (white):  0.2430
k-fold Random Forest Regressor (white):  0.2704
```

Figure 5: K-fold scores for the regression models



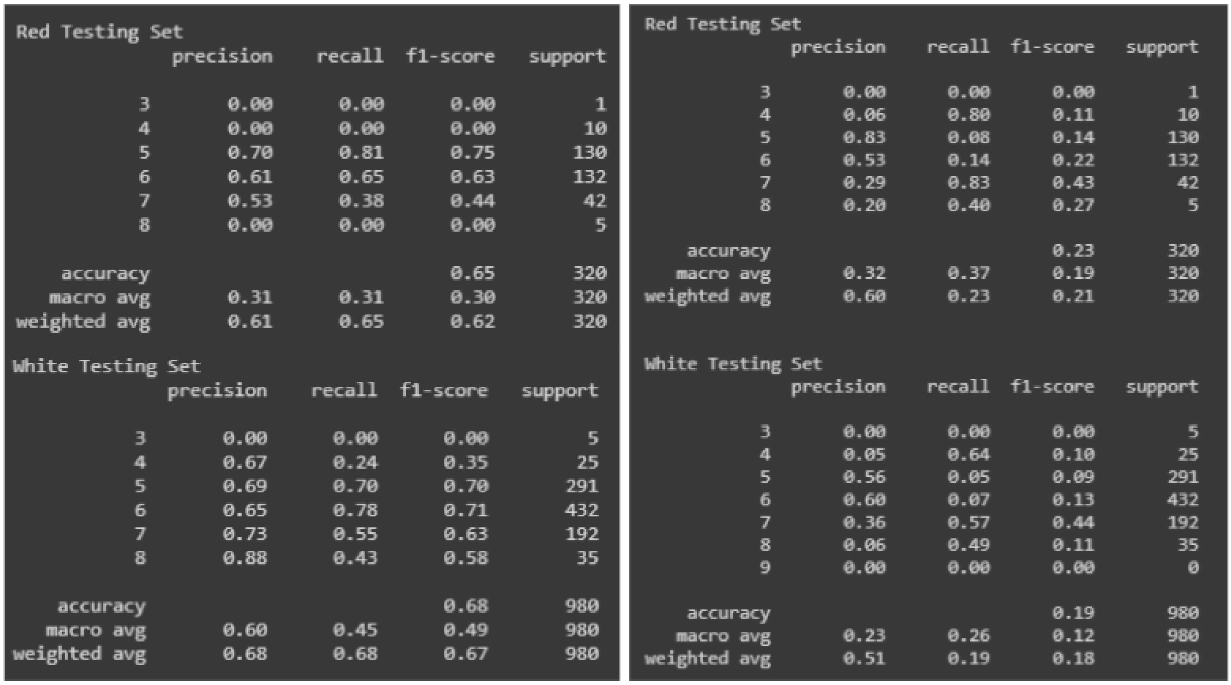
```
Actual vs Predicted values (Red)
   Actual Value  Predicted Value  Difference
0             6         5.386299    0.613701
1             5         5.067086   -0.067086
2             6         5.664471    0.335529
3             5         5.510344   -0.510344
4             6         5.665427    0.334573
5             5         5.336490   -0.336490
6             5         5.035855   -0.035855
7             5         5.116302   -0.116302
8             5         5.756233   -0.756233
9             6         5.610476    0.389524
```

Figure 6: Multiple linear regression actual values vs predicted

Predicting all possible values

The first dilemma I came across with this approach is the use of SMOTE. The dataset is very skewed in both red and white wines. For example in red wine there are 638 quality 6 wines but only 10 quality 3 wines. In white wine there are 2198 quality 6 wines but only 5 quality 9. This means that using SMOTE would be oversampling 5 values into 2198 values which gave me a gut feeling of not seeming sensible. After doing some research I found that oversampling to this extent can lead to overfitting (Yap et al., 2014) so I took the results I then got from the SMOTE models with some apprehension. It then turned out that SMOTE actually decreased the accuracy for both the models I tried for instance in random forest (Figure 7)

As seen in figure 7 although some accuracies are somewhat high for example in qualities 5 and 6 for both wines the wine qualities at the extremities have very low accuracies and recall often at 0. It should be noted that although SMOTE decreases the overall accuracy for some of the qualities such as 3 and 8 in red SMOTE actually increases the accuracy somewhat despite still being very low. After parameter tuning the highest performing model still only gets a k-fold score of 0.55 so I conclude that it is more appropriate to use a LQ/HQ threshold.



```
Red Testing Set
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       0.00      0.00      0.00        10
           5       0.70      0.81      0.75       130
           6       0.61      0.65      0.63       132
           7       0.53      0.38      0.44        42
           8       0.00      0.00      0.00         5

    accuracy                           0.65       320
   macro avg       0.31      0.31      0.30       320
weighted avg       0.61      0.65      0.62       320
White Testing Set
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.67      0.24      0.35        25
           5       0.69      0.70      0.70       291
           6       0.65      0.78      0.71       432
           7       0.73      0.55      0.63       192
           8       0.88      0.43      0.58        35

    accuracy                           0.68       980
   macro avg       0.60      0.45      0.49       980
weighted avg       0.68      0.68      0.67       980
```

```
Red Testing Set
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         1
           4       0.06      0.80      0.11        10
           5       0.83      0.08      0.14       130
           6       0.53      0.14      0.22       132
           7       0.29      0.83      0.43        42
           8       0.20      0.40      0.27         5

    accuracy                           0.23       320
   macro avg       0.32      0.37      0.19       320
weighted avg       0.60      0.23      0.21       320


White Testing Set
              precision    recall  f1-score   support

           3       0.00      0.00      0.00         5
           4       0.05      0.64      0.10        25
           5       0.56      0.05      0.09       291
           6       0.60      0.07      0.13       432
           7       0.36      0.57      0.44       192
           8       0.06      0.49      0.11        35
           9       0.00      0.00      0.00         0

    accuracy                           0.19       980
   macro avg       0.23      0.26      0.12       980
weighted avg       0.51      0.19      0.18       980
```

Figure 7: Predicting all qualities classification reports before (left) and after (right) SMOTE (Random Forest)

Conclusions

For future work I think the process of dropping certain columns could be automated and although this might take a lot more computing power and time it could ensure that the best combinations of columns to predict outcome could be found. I think that predicting all labels could be a good option if there was more data so it would be interesting to run some models with more data.

On reflection I think I could have written some functions to reduce the amount of code I wrote, for example storing all the different balanced and unbalanced x and y columns in variables soon became very messy and confusing so a method to run a bunch of models when inputting the model type and columns would be a good thing to do next time round. I also wasted a lot of time plugging in tuned parameters manually at the start which I would just automate to begin with in the future.

I found there was a lot for me to get my head round understanding the different models, the ways they could overfit and the effect of SMOTE and I'm still not sure whether I have understood whether I approached the regression problem appropriately as although it is very inaccurate it seems relevant to me that it could be predicting qualities in between the natural numbers for example with two quality 6 wines, one might have attributes tending towards a 7 which the model might be picking up on despite it being categorised a 6 so maybe the model predicting it a 6.4 could still be useful information.

In summary a plethora of different models were tested both with classification and regression using the preliminary analysis to decide if columns should be dropped. I conclude that the most appropriate model was the random forest classifier for LQ and HQ as it performed highest on both white and red wine and eliminates some of the overfitting that decision trees introduce whilst still performing well.

Bibliogrpahy

A comparison of the Pearson and Spearman correlation methods (n.d.). Available at: https://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics/regression/supporting-topics/basics/a-comparison-of-the-pearson-and-spearman-correlation-methods/ (accessed 17 April 2021).

Fuchs K (2017) Machine Learning: Classification Models. Available at: https://medium.com/fuzz/machine-learning-classification-models-3040f71e2529 (accessed 18 April 2021).

Yap BW, Rani KA, Rahman HAA, et al. (2014) An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets. In: *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013)*, 2014, pp. 13–22. Springer Singapore.