PREDICTIVE ANALYTICS

Syndicate Task #3 – Predicting Wine Quality

Syndicate 8
Tony Trinh (1099433)
Alvin Lee (1169514)
Varun Sharma (959620)
Jonno Lindsay (1026965

Table of Contents

1. Purpose

The purpose of this report is to identify if the predictive performance of supervised (Regression Tree & Neural Network) and unsupervised (K-means clustering & K-NN regression) machine learning models is more accurate than that of the simpler regression models such as linear & logistics regressions. Within that context, the data of interest is the relationships between wine quality and physiochemical properties.

On top of the standard predictive accuracy measures such as RMSE, MAE, MAPE & MASE, a specific loss function is created to target the premium wine market. This will help winemaker to more accurately predict the wine quality score during the production process in order to optimise their pricing and marketing strategy to maximise profit.

2. Methodology

Various Regression Tree with CP values ranging between 0.01 to 0.1 were modelled. As expected, when pruning the regression tree, the greater the CP value, the fewer physiochemical property segments featured.

In addition, multiple Neural Network (NN) were trained with one hidden layer and different number of nodes (4 & 5) to determine the stability of the models. It's well-known that NN may be unstable and using less layers / nodes (4 nodes in this case). It's also worth noting that using too many layers or nodes may reduce the model's predictive accuracy, possibly due to overfitting. Sensitivity analysis using Lek's profile was also run for all explanatory variables with the 7 groups as well as 2 groups.

Moving on to K-means clustering, the Elbow method indicated that the optimal number of clusters (k) is 7. Meanwhile, the Gap method demonstrates that k=2 is best. Lastly, the Silhouette method suggests that a k=2 would be most appropriate though 7 & 9 are also reasonable selections.

Finally, K-Nearest Neighbour model was utilised though its performance fell into the middle of the pack.

3. Analysis

3.1. Regression Tree

When conducting the VIP regression tree, 'Alc, 'Sulphates' and 'VA' are considered some of the most important variables, which aligns with what is represented in the regression tree. However, despite featuring as the second most important variable in the VIP regression tree, 'Density' does not feature as a segment on any regression tree, regardless of their CP value.

As outlined in Appendix 6.1, the predicted quality score ranges from 5.1 to 6.6, subject to the pathway followed. For example, wines with lower alcohol and sulphate levels are predicted to have a quality score of 5.1.

3.2. Neural Network

Garson's method depicts the most important variables in explaining quality are 'CA','Alc' and 'TSD'. The Olden's method demonstrates a slightly different picture, with 'Alc','FA' being the most important variables. Using Olden's method, 'CA' is considered one of the least important variables and likely there are positive and negative weights associated with this variable.

Lek's profile was performed and demonstrates the impact wine properties has on the final quality score. Higher levels of alcohol have a positive impact on quality score, across the middle and upper ranges. There's an inverse relationship between the level of 'TSD', 'VA' and 'pH' on quality score, as the values of these properties increase, the quality score decreases. The sharpest quality score reductions occur with the 'pH' and 'VA' variables.

3.3. K-Means Clustering

The k-means clustering k=7 (Appendix 6.2) indicates that those clusters with high quality scores (1, 5 and 6) tend to be high in alcohol. These three clusters also seem to indicate that lower 'TSD' and 'FSD' levels tend to produce better quality scores. Interestingly, cluster 7 has one of the lowest 'FSD' and 'TSD' levels but is also very low on the quality score. It may indicate that this characteristic in conjunction with a variable like alcohol might help determine quality score.

K-means with k=2 (Appendix 6.2) provides a clearer determination of drivers of wine quality. It appears wines higher in 'Alc', 'Sulphates', 'Ch', 'RS', 'CA' and 'FA' have a positive impact, whilst 'VA', 'FSD', 'TSD', and 'pH' have a negative impact.

3.4. K-Nearest Neighbour (kNN)

With kNN, it's rather hard to explain the causal impact of wine characteristics on quality score as each data point can belong to multiple groups and thus makes it hard to determine what is driving wine quality.

4. Winemaker's loss function

The winemaker loss function was developed (Appendix 6.3) based on the below pricing model, where it is assumes the price of wine increases significantly when the quality score is 7 or above. It is also assuming that the extra cost on marketing a high-quality wine is fixed at approximately $2 per bottle. It is evident from these assumptions that the loss of revenue due to underestimation significantly outweighs any expenditure loss due to marketing, therefore overestimation is preferred rather than underestimation for high quality wines.

Wine Quality Score	1	2	3	4	5	6	7	8	9	10
Wine Retail Price	$ 5-7	$ 5-7	$ 7-8	$ 7-8	$ 8-10	$ 8-10	$ 20	$ 30	$ 40	$ 50

Business use cases were used to develop the weighting matrix (Appendix 6.3) to allocate over and under estimation factors to different wine quality scores based on the difference between the predicted and actual quality score.

The predictive performance of the loss function is in-line with other standard predictive accuracy metrics like RMSE, MAE, MAPE & MASE (Appendix 6.4). Interestingly the 1 layer & 5 node NN model (seed 321) performs best for our specific loss function. However, this is just an anomaly given the instability of the 5 nodes neural network. The predictive performance of the loss function performs poorly for the regression tree and k-means methods due to the rigidity of final regions. It performs poorer for K-NN and the neural networks as these models tend to slightly underpredict actual wine scores 7 and above. Lastly, the predicted quality score used for the loss function is not rounded off to maintain the integrity of loss calculation.

5. Comparison to non-machine learning methods

As demonstrated previously, simple regression methods (linear & non-linear) determined that ‘FSD’, ’Sulphates’ and 'Alc’ had positive effects on red wine quality scores and ‘VA’, ’Ch’, ’TSD’, ’pH’ all have negative effects on wine quality. To an extent, analysis using machine learning methods indicated similar findings. Alcohol was the clearest driver on wine quality, with regression tree, neural networks and clustering models all indicating its importance. The other important variables then varied, depending on which method was performed. ‘TSD’ and ‘VA’ appeared to be other common variables with a negative impact on quality score.

Overall, none of the machine learning methods predicted particularly well relative to the non-machine learning methods when measured in term of predictive accuracy using both standard metrics and the winemaker’s loss function (Appendix 6.4). Some machine learning methods were unlikely to ever predict better due to the nature of their models. For example, regression tree cannot predict wine scores at a more granular level, which is evident by the actual quality scores which range from 3 to 8. This demonstrates that although predictive insights can be easily understood using this method, the segmentation of quality scores is too coarse-grained.

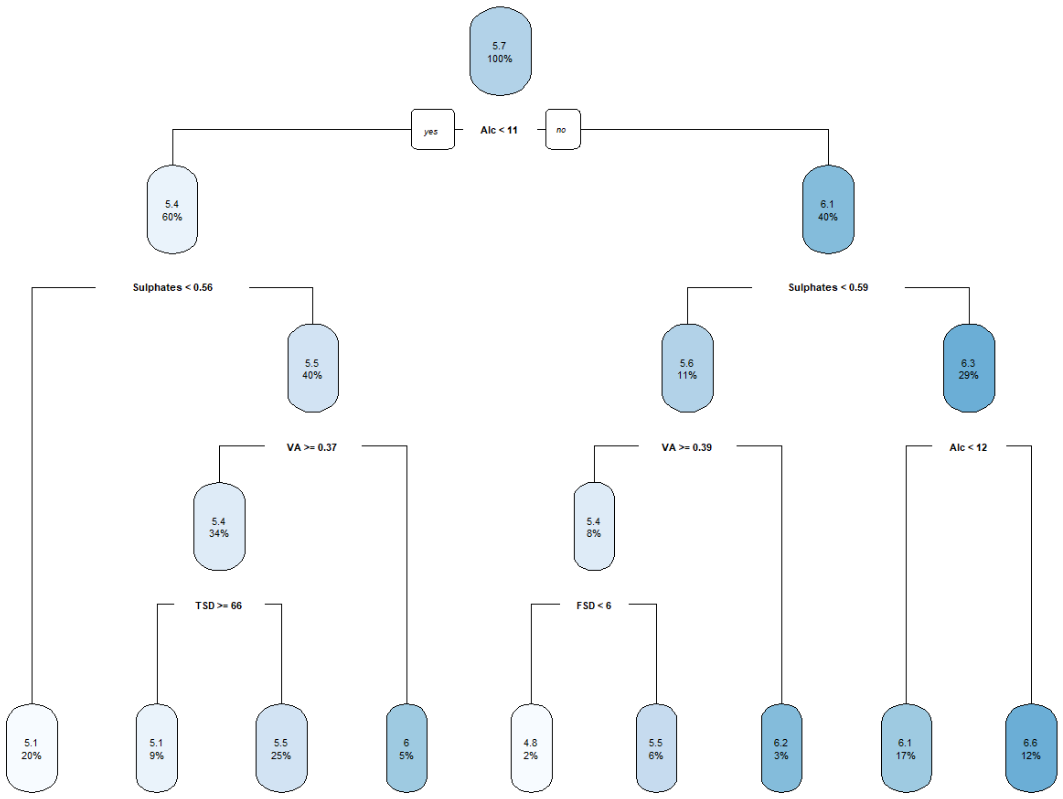
The k-means clustering also suffered a similar problem in that predictive scores are limited to the k selected. Utilizing the clusters as a pre-filter, the regression improved the loss function metrics but not well enough to trump any of the non-machine learning methods.

Out of the neural network models, the one with 1 layer and 4 nodes proved to provide the most stable predictions.

The reasons why machine learning methods perform worse than the non-machine learning methods may be due to a lack of data or overfitting. It may also be because these methods may not be well suited to this problem and would fare better for classification problems.

6. Appendix

6.1 Regression Tree



6.2 K-Means=7 & K-Means=2

Cluster	FA	VA	CA	RS	Ch	FSD	TSD	Density	pH	Sulphates	Alc	QS
1	6.3684	0.6117	0.0848	2.1916	0.0668	18.7161	41.6645	0.9941	3.5066	0.6246	11.7778	5.8903
2	7.9378	0.5250	0.3549	8.2459	0.0911	30.8378	104.1351	0.9985	3.2792	0.6716	10.1108	5.4324
3	8.1799	0.5268	0.2779	2.4092	0.0857	26.4672	85.6025	0.9972	3.2891	0.6220	9.8059	5.3320
4	8.5261	0.5213	0.5152	1.9826	0.3687	15.0000	59.9130	0.9971	3.0357	1.2822	9.4609	5.3478
5	11.1886	0.4207	0.5160	2.7348	0.0872	10.4238	31.3714	0.9988	3.1546	0.6943	10.4716	5.8905
6	8.4690	0.3458	0.4091	2.2955	0.0746	12.5728	28.6855	0.9956	3.2936	0.7332	11.4484	6.3005
7	7.5453	0.6418	0.1188	2.2482	0.0860	10.9597	33.2519	0.9967	3.3609	0.5937	9.8797	5.3325

Cluster 2	FA	VA	CA	RS	Ch	FSD	TSD	Density	pH	Sulphates	Alc	QS
1	7.295086	0.606275	0.142683	2.398274	0.080231	16.65073	48.01992	0.996108	3.380385	0.600239	10.33152	5.491368
2	9.75019	0.408802	0.45846	2.715114	0.097068	14.30608	42.42586	0.997548	3.213042	0.738783	10.62744	5.891635

6.3 Loss Function Code & Business Use Case

```
wine_maker_loss_function = function(error,actual){
  relative=100*(error/actual)
  lossDF=as.data.frame(t(rbind(relative,actual)))

  highQualityOverestimate
=lossDF[lossDF$actual>=7 & lossDF$relative <= 0, 1]
  highQualityUnderestimateWithin22
=lossDF[lossDF$actual>=7 & lossDF$relative > 0 &
lossDF$relative <= 22, 1]
```

```
  highQualityUnderestimateWithin22to29
=lossDF[lossDF$actual>=7 & lossDF$relative > 22 &
lossDF$relative <= 29, 1]
  highQualityUnderestimateWithin29to36
=lossDF[lossDF$actual>=7 & lossDF$relative > 29 &
lossDF$relative <= 36, 1]
  highQualityUnderestimateMoreThan36
=lossDF[lossDF$actual>=7 & lossDF$relative > 36, 1]

    lowQualityUnderestimate
=lossDF[lossDF$actual<7 & lossDF$relative>=0, 1]
  lowQualityOverestimateWithin22      =lossDF[lossDF$actual<7
& lossDF$relative < 0 & lossDF$relative >= -22, 1]
  lowQualityOverestimateWithin22to29  =lossDF[lossDF$actual<7
& lossDF$relative < -22 & lossDF$relative >= -29, 1]
  lowQualityOverestimateWithin29to36  =lossDF[lossDF$actual<7
& lossDF$relative < -29 & lossDF$relative >= -36, 1]
  lowQualityOverestimateMoreThan36    =lossDF[lossDF$actual<7
& lossDF$relative < -36, 1]

  loss= sum(1*abs(highQualityOverestimate)) +
      sum(8*abs(highQualityUnderestimateWithin22)) +
      sum(18*abs(highQualityUnderestimateWithin22to29)) +
      sum(28*abs(highQualityUnderestimateWithin29to36)) +
      sum(38*abs(highQualityUnderestimateMoreThan36)) +
      sum(1*abs(lowQualityUnderestimate)) +
      sum(8*abs(lowQualityOverestimateWithin22)) +
      sum(18*abs(lowQualityOverestimateWithin22to29)) +
      sum(28*abs(lowQualityOverestimateWithin29to36)) +
      sum(38*abs(lowQualityOverestimateMoreThan36))

  return(loss/length(error))
}
```

Example business case for loss function

Assume a wine has been predicted to be of the quality score of 7, winemaker will price the wine at $20 per bottle and spend $2 on the marketing expenses, expecting an additional profit of $8.

- If the actual score is less than 7, then the winemaker will lose an additional $2 spent on marketing and will have to sell the wine at marked down price (between $8 to $10, depending on the degree of error in estimation).
- If the actual score is 7 or higher, the winemaker will either, make $8 or will have revenue loss based on the degree of error estimation. If error is up to 22%, there is no loss, but if error increases beyond 22% loss of revenue up till 29% will be $18, as the actual wine score will be 8 and can be sold at $30. Similarly, if the error is between 29% and 36%, the actual wine score in that case is 9 and can be sold at $40 and hence the loss of revenue will be $28. In case the error is over 36%, the actual wine score will be 10 with revenue loss of $38 as the wine can be sold for $50.

- In case of underestimating when actual and predicted are less than 7 the loss is will be incurred due to incorrect pricing. Similarly, in case where actual and predicted are both overestimated and are above the score of 7 loss will be incurred due to lack of cost optimization.

Matrix Weighting Summary	0 - 22%	22% - 29%	29% - 36%	> 36%
Underestimate factor if actual > 7	8	18	28	38
Overestimate factor if actual > 7	1	1	1	1
Underestimate factor if actual < 7	1	1	1	1
Overestimate factor if actual < 7	2	4	4	4

6.4 Predictive Accuracy Comparison

	RMSE	MAE	MAPE	MASE	WineMakerLoss
Linear	0.5859	0.4650	8.6633	0.7294	84.6165
Stepwise	0.5872	0.4646	8.6563	0.7288	86.0358
Nonlinear	0.5835	0.4608	8.6200	0.7228	86.7770
RegTree	0.6290	0.4739	8.9016	0.7433	98.7678
NN_1_4_seed#321	0.5971	0.4644	8.7054	0.7285	89.5152
NN_1_4_seed#123	0.5992	0.4692	8.7790	0.7360	88.7988
NN_1_4_seed#888	0.6029	0.4631	8.7231	0.7264	91.1193
NN_1_5_seed#321	0.6158	0.4858	9.0150	0.7620	81.6527
NN_1_5_seed#123	0.5971	0.4644	8.7054	0.7285	89.5152
kMeans	0.6607	0.5316	9.9400	0.8339	150.0844
kMeansRegFull_2cluster	0.5895	0.4698	8.7434	0.7369	175.4709
kmregWithoutCA_2cluster	0.5862	0.4680	8.7128	0.7342	178.3517
kMeansRegFull_7cluster	0.6088	0.4871	9.0919	0.7641	189.1043
kmregWithoutCA_7cluster	0.6067	0.4888	9.1312	0.7667	194.3804
knn	0.6092	0.4929	9.3244	0.7732	102.4079