Classification of Red Wine Sensory Scores

Simen Håpnes

(Dated: Tuesday 17th December, 2019)

A numerical study regarding the classification of red wine preference scores is presented. A large data set (1599 data points) of red wine from the *vinho verde* district in Portugal is analysed by applying three ensemble machine learning techniques. The predictors at hand, were 11 various physicochemical properties of the wines. The response variable was a wine preference score from 0-10, based on the median value of at least three blind testes. The data mining techniques were *random forests*, *bagging* and *gradient boosting.* Due to the imbalanced nature of the outcomes, the most difficult task was to identify the worst and the best wines. Random forest was the method which achieved the best overall accuracy of predicting the exact correct score, with an accuracy of 66.9%. Confusion matrices from all three models showed that the misclassified wines were more likely to be identified as a more common score (towards 5 and 6). Out of all the predictors: alcohol, sulphates, and volatile acidity was the three most important ones in the random forest and gradient boosting models. The lowest and highest scored wines showed noticeable differences in the values of these three variables. It was seemingly wines with higher alcohol percentages, lower volatile acidity, and higher levels of sulphates, which predicted a good wine.

I. INTRODUCTION

The region *vinho verde*, located in the North West part of Portugal, is the largest wine region in its country and one of the largest regions in the world. The data set of the wines from this district were donated to UCI machine learning repository in 2009, and it consists of 1599 red wines and 4898 white wines. The data was collected during the period May 2004 - Feb. 2007.

A previous study on this data set has been conducted [P. Cortez]. Cortez et. al. applied three regression techniques that would perform simultaneous variable and model selection, namely multiple regression (MR), neural network (NN) and support vector machine (SVM). The latter, SVM, achieved promising results and outperformed the other models. They also computed the confusion matrix for various fixed error tolerances. Lastly, they plotted the relative importances of the input variables in the SVM model. As a proposal for further analysis, the data sets could be viewed as either classification or regression tasks. Interesting analyses would be to apply outlier detection algorithms to identify poor or excellent wines. Lastly, feature selection methods could be applied to identify which of the variables were most relevant.

The study presented, only considers the red wine data set. The techniques that will be applied, consists of bagging with decision trees, random forests and gradient boosting. The aim by applying these three methods is to study the methods themselves, and to explore which of these methods provide the best predictive ability for this data set. The overall aim is, as proposed by Cortez et. al., to detect the excellent and poor wines.

In the theory section, the necessary theory behind the three models will be introduced. The method section will in detail cover the processing of the data set, how the algorithms were implemented and the techniques used for tuning of parameters. In the result and discussion sections, the results of this research will be presented and examined. Finally, the conclusion section will summarize the key insights and conclusions. For reproducibility and further research, the codes and algorithms, figures and results, may also be found in my GitHub repository at `https://github.com/simehaa/WineQuality`.

II. THEORY

This section will introduce the three data mining methods that will be applied in the research presented. All three of the methods: bagging, random forests and gradient boosting are ensemble methods which typically uses decision trees as base learners.

The correlation matrix $corr(X)$ is a $p \times p$ matrix, where p is the number of variables that is considered. Each entry in the matrix is evaluated by considering the two column vectors X_i and X_j of length N, where N is the number of data points. This matrix element is the correlation between variable i and j in the data and its given by

$$\rho_{i,j} = \frac{cov(X_i, X_j)}{\sigma_{Xi}\sigma_{Xj}} = \frac{E[(X_i - \mu_{Xi})(X_j - \mu_{Xj})]}{\sigma_{Xi}\sigma_{Xj}} \in [-1, 1]. \tag{1}$$

Correlation values closer to -1 and 1 signifies a stronger correlation. Note also that the matrix is symmetric and that all diagonal elements are 1.

A. Bagging

Bagging is an abbreviation for bootstrap aggregating, and it is essentially an ensemble methods of many base learners which each is applied to a bootstrap sample of the training data. The most common base learner for bagging is decision trees. This subsection will cover the main concepts of decision trees, bootstrap samples, tuning parameters and the general algorithm.

Decision trees are one of the most easily understood and interpretable machine learning methods. The goal of a single tree is to predict an output value, which in the classification case is a categorical value. The way it works is to sequentially split the data set into two subsets that are called nodes. Each node can again be split into two new nodes. The number of splits required to reach the leaf which is farthest out from the stem is referred to as the maximum depth of the tree. The depth is one of the most important tuning parameters of a decision tree.

The rule that is used for the split is based on a criterion, and most commonly that criterion is either the *gini impurity* or the *information gain*. When considering decision trees, there are two central algorithms, the so-called CART algorithm (Classification and Regression Tree) and ID3 algorithm (Iterative Dichotomiser 3). CART uses gini impurity and ID3 uses information gain, and the research presented will apply the CART algorithm, when using decision trees.

Drawing bootstrap samples to produce each single model is done by picking data points *with* replacements. This means that the same data point can occur several times in the bootstrap samples, and it also means that some samples will most likely not be included in the bootstrap samples. The expected number of unique data points is given by

$$1 - e^{-1} \approx 63.8\%. \tag{2}$$

In bagging with decision trees, there are two important tuning parameters. The first one is the number of trees, B and the second one is maximum depth of each individual tree. The output value from an input is decided by aggregating the results from all trees, and this can be done by either consensus voting or probability voting. In the former, each tree rounds off its output to produce a single vote for one of the output categories. In the latter, all the trees' probabilities for all the various categories are aggregated: the output value is simply the category with the highest aggregated probability.

Bagging is a data mining method which certainly is computationally slower than a decision tree, so in what way does it provide better results? The bias of a bagging model is unaffected, whereas the variance can be decreased, and the total effect is a reduced prediction error. This also means that a bagged model is more resistant to overfitting.

The algorithm for bagging is outlined as a special case of the algorithm in the next subsection: *Random Forests*, because of these models' similarity.

B. Random Forests

Random forests is a method which is very similar to bagging with decision trees. RF always considers decision trees, and has an additional tuning parameter in addition to bagging. For each individual tree, RF picks a subset out of the p features. The number of features for each tree, will be referred to as m, and for classification problems m is typically

$$m \approx \lfloor \sqrt{p} \rfloor. \tag{3}$$

But of course, m can be tuned and adjusted by using e.g. cross-validation. All in all, there are typically three tuning parameters of RF: $m \leq p$ number of features per tree, max depth of each tree and the total number of trees.

The algorithm for RF for classification can be summarized in these steps[1]:

1. For $b = 1, B$:
   (a) Draw a bootstrap sample $\mathbb{Z}^*$ of size N from the training data (draw data points with replacement).
   (b) Grow a RF tree T_b to the bootstrapped data, by recursively repeating the for each terminal node of the tree, until the minimum node size is reached:
      i. Select $m \leq p$ features.
      ii. Grow the tree by picking the best split point among m and for the child nodes and so on.
2. Output the ensemble of trees $\{T_b\}_1^B$.

The algorithm for bagging with decision trees is almost identical to the algorithm above, but in bagging $m = p$, i.e. all the predictors are used. With the ensemble of trees, a prediction for a new data point can be made by

$$\hat{C}_{rf}^B(x) = \textit{majority vote}\left\{\hat{C}_b(x)\right\}_1^B. \tag{4}$$

Another possibility is to average over all the trees' probabilities for each category, and then choose the category with highest aggregated probabiliy.

C. Gradient Boosting

Gradient boosting (GB) is the final ensemble method that will be applied as a classification algorithm. GB is a so-called *forward stagewise additive model*, which means that it sequentially adds more terms to the model function. GB will start of with a weak learner which typically is a decision tree with a small depth. A weak learner is a base learner with very bad predictive abilities, i.e. only slightly better than just random guessing. The goal of GB is to minimize a pre specified loss function wrt. the model function. This problem can be solved by applying a steepest decent algorithm. The general idea is to determine the gradient of a given loss function[1]

$$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f_{m-1}}. \tag{5}$$

If the only goal was to minimize the training loss, then steepest decent would be an excellent choice of method. However, we want to uncover a final model function $f_M(x)$ which is also applicable to new data. In classification problems, the loss function will be deviance, where its derivative is given by

$$-g_{ikm} = I(y_i = \mathcal{G}_k) - p_k(x_i). \tag{6}$$

$$p_k(x) = \frac{e^{f_k(x)}}{\sum_{l=1}^K e^{f_l(x)}} \tag{7}$$

Notice, that the g_{im} from equation 5 is now g_{ikm}, where a new index k represents a tree. At each iteration in classification GB, K trees must be fitted, one for each category.

The algorithm of GB for a K-class classification problem can be summarized in these steps[1]:

1. Initialize $f_{k0}(x) = 0$ for
2. for $m = 1, M$:
   (a) Set $p_k(x)$ according to equation 7 for $k = 1, K$.
   (b) For $k = 1, K$:
      i. Compute $r_{ikm} = y_{ik} - p_k(x_i)$ for $i = 1, 2, ..., N$.
      ii. Fit a regression tree to the targets g_{ikm} for $i = 1, 2, ..., N$.
      iii. For $j = 1, 2, ..., J_m$, compute

$$\gamma_{jkm} = \frac{K-1}{K} \frac{\sum_{x_i \in R_{ikm} r_{ikm}}}{\sum_{x_i \in R_{ikm} |r_{ikm}|(1-|r_{ikm}|)}}. \qquad (8)$$

      iv. Update

$$f_{km}(x) = f_{k,m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jkm} I(x \in R_{jkm}). \qquad (9)$$

3. Output $\hat{f}_k(x) = f_{kM}(x)$, $k = 1, 2, ..., K$.

J_m represents the total number of nodes allowed in a single tree, e.g. $J_m = 2$ represents a single split. The learning rate ν is not necessary, but introduces shrinkage, and it will help to prevent overfitting. Two important tuning parameters are the total number of iterations M and the learning rate, and they affect each other: with a smaller learning rate, a higher number of iterations are needed. A third tuning parameter is J_m. According to[1]: $4 \leq J_m \leq 8$ does by experience work well. A simple choice is to simply set $Jm = 6$. Alternatively, one could tune J_m with a validation set, but it seldom provides a significant improvement over using $J_m \approx 6$[1].

D. Metrics

The metrics are the methods for evaluation of the models. Because the problem at hand is a classification problem, the accuracy will be used. One type of accuracy is the overall accuracy, which simply is the fraction of correctly classified data points. A more interesting overview can be given by studying the specific accuracy for each of the output categories. This will give more insight than the overall accuracy, because it will also reveal how well the less populated categories are predicted.

An even more nuanced picture of the accuracy is given by the confusion matrix. This matrix shows the distribution of predicted outputs for each input category. A perfect model would have all data points along the diagonal.

The random forest and gradient boosting methods both performs a feature analysis: RF picks samples of the predictors and GB actually finds a factor for each of the variables in each iteration. These models are therefore able to provide a feature importance. This can be provided directly from *Scikit-learn's* ensemble package[3].

III. METHOD

A. Analysis of the Data Set

The numerical work was conducted by using the popular programming language *Python 3*. An incredible useful package for many different machine learning techniques is Scikit-learn[3], and this package will be applied for many of the numerical algorithms in the study presented.

The data set was analyzed in three ways in order to obtain a good overview over the data. The distribution of the response variable was plotted as a histogram. For the predictors, the correlation matrix was plotted and a table showing the min, max and mean of all values were computed.

A first analysis of the ouput variables was done. The response variable, y was analyzed by looking at the distribution of the wine scores. A key aspect of the y data, was that is was not balanced, i.e. the majority of the data points contained a score of 5 and 6 and all scores were in the range $[3, 8]$. This imbalance, could lead to models which would have a very weak ability to identify the boundary scores (especially scores of 3 and 8). It would be interesting to see if the models are able to identify the few bad or few good wines. Therefore two approaches for the data set were applied. The first approach was to not do anything with the data at all and see what results the models yield. The second approach was to balance the data set, by randomly resampling data points in the training set: so that the number of data points in each category were equal. Note: this was **not** done for the test set, because the models will predict the same outputs for the same inputs and thus lead to a wrong accuracy score in the evaluation of the models.

A better look at the variables could be important, because there could be correlations between some of them. The approach to investigate the possible correlations was to compute and plot the correlation matrix for all features, including the output. One last aspect to ensure a proper analysis is to look at the data set before processing it, to actually see that the numbers in each feature is valid and makes sense (e.g. pH values cannot be 100). This was done by looking at the minimum, maximum and mean value of all predictors before standardizing the data.

B. Processing of the Data

Before the models were used, the data set was processed. A very important aspect is to split the data set into a training part (2/3) and a test part (1/3). The purpose of the test set is to be set aside and only be used as a final evaluation of the methods, this means that the test set should not be used for tuning of the model parameters.

The input data X, was standardized, by adjusting the predictor columns to obtain a zero mean and one standard deviation. The methods should not know any information about the completely independent test set, therefore both sets were standardized according to the training set.

The data was very imbalanced, i.e. there was a lot more data points with qualities (output variable) of 5 and 6. A histogram which shows the distribution of outputs was plotted. In order for the models to better predict the wines with high and low qualities, the training data was upscaled. This was done by a so-called randomOverSampler, which randomly copyies data points from minority categories until the number of outputs in each class were equal.

The final analyses of the input data, were done by computing and plotting the correlation matrix of all features (including the output) and by plotting a histogram of the output variables.

C. Implementing the Models

The three models: bagging, random forests and gradient boosting were implemented by using a popular Machine Learning package in Python, namely *Scikit-learn*[3]. The classes used for the three methods were BaggingClassifier, RandomForestClassifier and GradientBoostingClassifier. The general idea was to

1. tune some selected input variables by using GridSearchCV, which uses 5-fold cross validation on the test set and finds the best set of tuning parameters by evaluating all the possible models.

2. predict the test data with the tuned models.

3. evaluate the results by plotting confusion matrices, computing the overall accuracy and the accuracy per output category.

The three methods was used along with *scikit-learn's* class: *GridSearchCV*. This method performs a grid search over all combinations of parameters. This combination is specified as a dictionary with parameters and the respective values for those parameters. This grid search was done for both the normal training data and the upsampled training data (with balanced output values). The grid search was performed with 5-fold CV for each model and the scoring metric was "balanced accuracy" which aims to optimize the average accuracy on all categories.

The evaluation of the models included plotting of confusion matrices, computing of the overall accuracy score and the accuracy per category. For RF and GB, *scikit-learn* has the functionality to show relative feature importance (RFI) for the final model. For these two models, the RFI was plotted.

IV. RESULTS

A. The Red Wine Data Set

In table I, the minimum, maximum, and mean of each predictor is shown.

TABLE I. Summary of the min, max and mean values of the physicochemical variables of the entire red wine data set.

Predictor (unit)	Min	Max	Mean
fixed acidity (g(tartaric acid)$/l$)	4.6	15.9	8.32
volatile acidity (g(acetic acid)$/l$)	0.1	1.6	0.5
citric acid (g/l)	0.0	1.0	0.3
residual sugar (g/l)	0.9	15.5	2.5
chlorides (g(sodium chloride)$/l$)	0.01	0.61	0.09
free sulfur dioxide (mg/l)	1	72	16
total sulfur dioxide (mg/l)	6	289	47
density (g/l)	0.990	1.004	0.997
pH	2.7	4.0	3.3
sulphates (g(potassium sulphate)$/l$)	0.3	2.0	0.7
alcohol ($vol.$%)	8.4	14.9	10.4

Figure 1 shows the distribution of the various scores in the enitre data set.

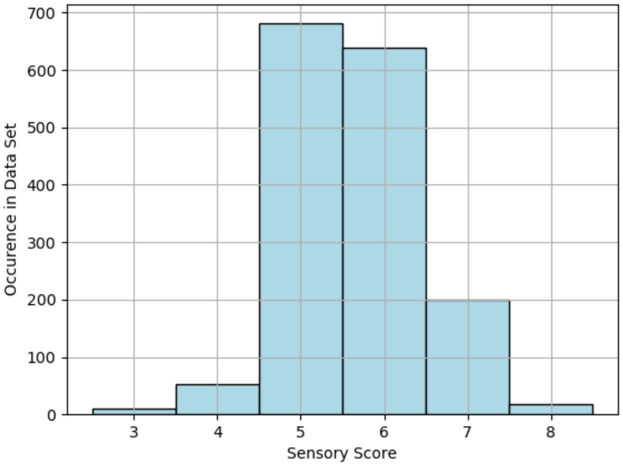

FIG. 1. The histogram of the response variable: sensory score for the enitre data set.

Figure 2 shows the computed correlation matrix of the input features.

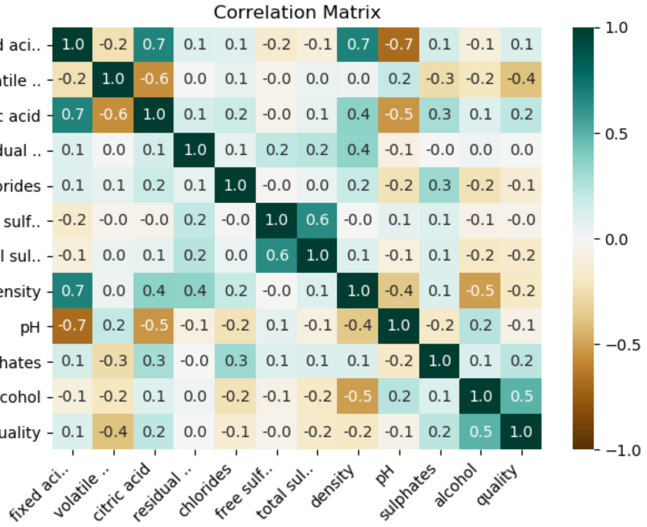

FIG. 2. The correlation matrix of all features (including output) plotted as a heatmap. Values closer to -1 or 1 signifies that the variables are more correlated.

B. Model Accuracy

Table II shows the final accuracy (with tolerance $T = 0.5$) for the three models.

TABLE II. Overall accuracy for all three models.

	Model		
	Bagging	Random Forest	Gradient Boosting
Accuracy [%]	64.8	66.9	64.4

Table III shows the accuracy per score (category) for all three models and for tolerances $T = 0.5$ (must be correct category) and $T = 1.0$ (within correct score ±1). The data set for these results were the normal data set.

TABLE III. Accuracy per score category for the normal data set and for the three ensemble methods.

	Accuracy per score					
Model (data set)	3	4	5	6	7	8
Bagging ($T = 0.5$)	0	4	79	69	55	20
Bagging ($T = 1.0$)	0	74	98	100	97	60
Random Forest ($T = 0.5$)	0	0	78	68	53	20
Random Forest ($T = 1.0$)	0	70	98	100	97	60
Gradient Boosting ($T = 0.5$)	0	13	73	68	50	20
Gradient Boosting ($T = 1.0$)	0	61	96	97	97	60

Table IV shows the accuracy per score (category) for all three models and for tolerances $T = 0.5$ (must be correct category) and $T = 1.0$ (within correct score ±1). The data set for these results were the **upscaled** data set.

TABLE IV. Accuracy per score category for the **upscaled data set** and for the three ensemble methods.

	Accuracy per score					
Model (tolerance)	3	4	5	6	7	8
Bagging ($T = 0.5$)	0	5	79	62	53	12
Bagging ($T = 1.0$)	100	76	99	97	95	50
Random Forest ($T = 0.5$)	0	0	80	69	49	12
Random Forest ($T = 1.0$)	33	76	100	99	99	62
Gradient Boosting ($T = 0.5$)	0	0	72	68	57	12
Gradient Boosting ($T = 1.0$)	33	76	99	97	93	75

C. Confusion Matrices

In figure 3, 4 and 5, the confusion matrices for each respective model is shown. The models are evaluated on the test set and trained on the upscaled data set.

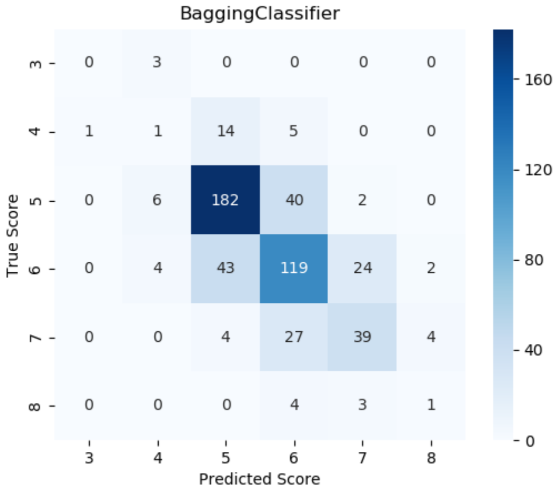

FIG. 3. Confusion matrix for the bagging model, obtained on the test set and by using the **upscaled** data set for training.

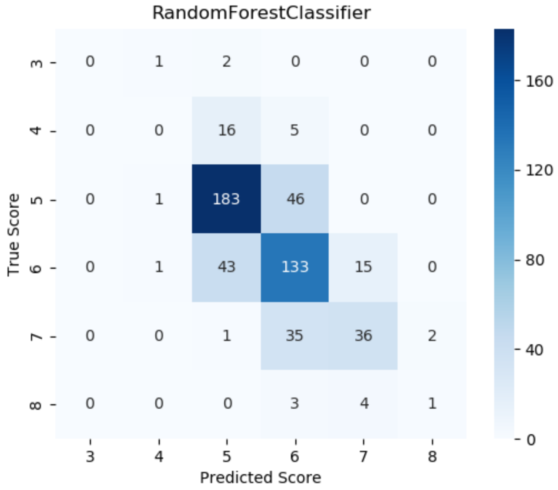

FIG. 4. Confusion matrix for the random forest model, obtained on the test set and by using the **upscaled** data set for training.

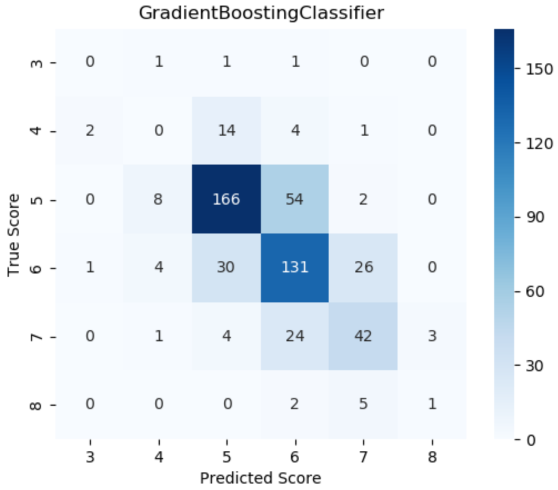

FIG. 5. Confusion matrix for the gradient boosting model, obtained on the test set and by using the **upscaled** data set for traininig.

D. Feature Importance

In figure 6 and 7, the relative feature importance for the random forest classifier model is shown. The figures show the normal and upscaled data set respectively.

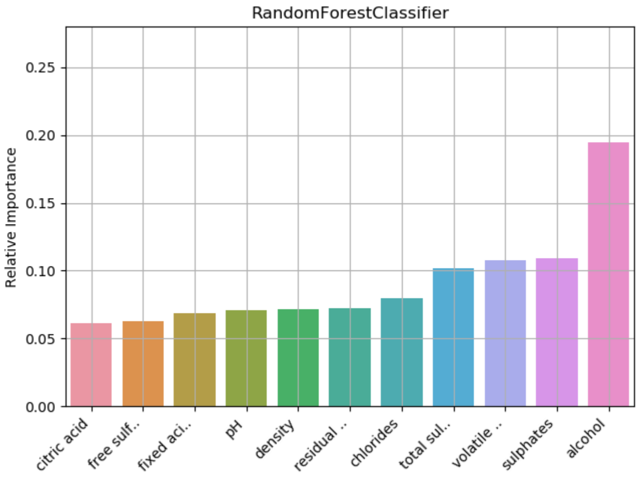

FIG. 6. Relative feature importance for the random forest model, by training with the normal data set.

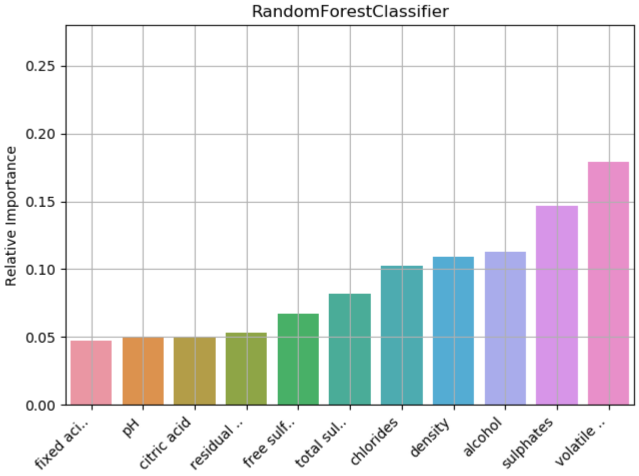

FIG. 7. Relative feature importance for the random forest model, by training with the **upscaled** data set.

In figure 8 and 9, the relative feature importance for the gradient boosting classifier model is shown. The figures show the normal and upscaled data set respectively.

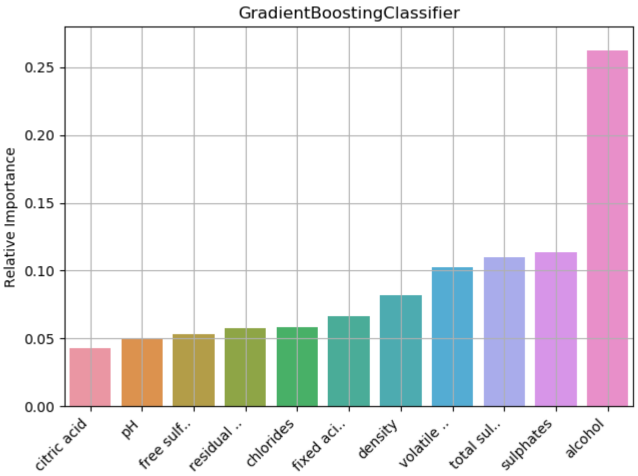

FIG. 8. Relative feature importance for the gradient boosting model, by training with the normal data set.

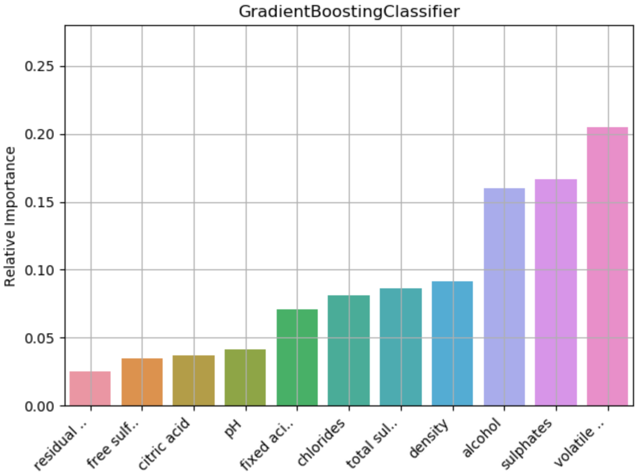

FIG. 9. Relative feature importance for the gradient boosting model, by training with the **upscaled** data set.

Table V shows a summary of the variables *volatile acidity*, *alcohol*, and *sulphates* for the outer categories: wines with scores 3 and 8.

TABLE V. Mean and standard deviation of the three variables *volatile acidity*, *alcohol*, and *sulphates* for the outer categories.

	Volatile Acidity	Alcohol	Sulphates
Quality	$(g(CH_3COOH)/dm^3)$	(vol.%)	$(g(K_2SO_4)/dm^3)$
8	0.42 ± 0.14	12.1 ± 1.2	0.8 ± 0.1
3	0.88 ± 0.31	10.0 ± 0.8	0.6 ± 0.1

V. DISCUSSION

A. Data Set

Table I shows the min, max, and mean values of each predictor. The purpose of this table is that the reader can get an overview over these variables.

In figure 1, a histogram over the output variable (quality score) is plotted. This figure shows the imbalance of the data set, which is a key aspect. There are by far most wines with a score of 5 and 6, followed by 7. The are very few data points of the outer categories 3 and 8. An interesting aspect in the following analyses is whether or not the models are able to identify the few bad or few good wines: this is perhaps one of the most valuable insights for the wine industry.

Figure 2 shows the correlation matrix. There are varying degrees of correlation, and the strongest correlations are

- *fixed acidity* and *citric acid* (0.7).
- *fixed acidity* and *density* (0.7).
- *fixed acidity* and *pH* (-0.7).
- *citric acid* and *volatile acidity* (-0-6).
- *free sulfur dioxide* and *total sulfur dioxide* (0.6).
- *pH* and *citric acid* (-0.5).
- *alcohol* and *density* (-0.5).

The predictors which are most correlated with the response (wine quality) are *alcohol* and *volatile acidity*.

B. Models

Table II shows the overall accuracy for the three ensemble methods. Bagging and GB achieves a similar accuracy of 64.8% and 64.4% respectively, whereas RF achieves the highest accuracy of 66.9%. All of these accuracies is better than the corresponding results in the previous research by [P. Cortez]: with their best model, support vector machines, they achieved an accuracy of 62.4%.

Table III shows the accuracy per score for the normal (imbalanced) data set. It can be seen that the scores of 5 and 6 achieves the highest accuracies, followed by the score 7. This seems to follow the balance in the data set: the more data points, the better the accuracy. Another feature of the table is that the tolerance $T = 1.0$ has very high accuracies for the scores 5, 6, and 7 and somewhat high accuracy for the scores 4 and 8. Not a single one of the worst wines (with score 3) are identified as either a 3 or a 4. In other words, it seems that the models are not so good in identifying the good wines (score 8), and very bad in identifying the bad wines (score 3).

Table IV shows the accuracy per score for the upscaled data set. None of the three models are still able to identify a 3 as a 3. However, all models are able to identify some of the bad wines as a 4 (at tolerance $T = 1.0$). Bagging was best to identify the bad wines, where 100% of the score 3 wines was identified as either a 3 or a 4. Gradient Boosting was best to identify the good wines, where 75% of the score 8 wines was identified as either a 7 or 8.

In figure 3, 4, and 5, the confusion matrices for the bagging, random forest, and gradient boosting are shown. All of these figures are for the upscaled (balanced) data set. These confusion matrices shows the same results as table IV, but with more information because we can also see what the misclassified data points are classified as. For all models, the misclassified data points tends to be more towards the middle, e.g. a misclassified 7 is more likely to be a 6 rather than an 8. The insights from these confusion matrices is that the models tends to classify correct or a more midrange score. In other words, the models are not very good at identifying the good or bad wines, even though the data set have been upscaled. This could be due to the fact that the upscaling only reuse data points, and what could improve the models is to actually have more unique data points, especially of the good and bad wines.

C. Feature Importance

There are four figures which shows the feature importance. Figure 6 and 8 shows the relative feature importance for a RF model and a GB model with the normal data set. Both of these figures has the highest relative importance for *alcohol*, followed by *sulphates*. The two next variables are *volatile acidity* and *total sulfur dioxide*, but in different order for the two models. Understandably, *alcohol* could be an important variable, especially when considering the wide range of alcohol values in the data set (between 8.4 % and 14.9 %). A possible explanation for why sulphates could be an important variable, is that it has an anti-oxidising effect, and this could be important in the aging of the wine. Sulphates could also be important when considering what happens to the wine when it is exposed to air. These insights could be interesting in further analyses.

Figure 7 and 9 shows the feature importance for a RF model and a GB model on the upscaled data set. These two figures could give some insights about which features that are important when trying to identify either the bad wines, the good wines, or both. By contrasting these two figures with both figure 6 and 8, the most important variable is now *volatile acidity*, and alcohol has now dropped to being only the third most important variable. This is a very interesting result, which might mean that *volatile acidity* has a noticeable effect on the wine quality.

The three most important variables, are also listed with mean and standard deviation for the specific bad and good wines in table V. Since RF and GB identified these variables with highest relative importance, it is possible two attempt to outline a relationship between these values and wine quality. The table shows a distinct separation in alcohol percentage, where the score 8 wines had a level of 12.1% which is higher than the score 3 wines, with 10.0% (separated with their uncertainties). The distinction in volatile acidity and sulphates does have a overlap in their uncertainties, but the distinction is still noticeable. It seems that wines with *lower* volatile acidity and *higher* level of sulphates score better. These results were found by contrasting the imbalanced and balanced data set, and this was not mentioned nor found in the previous research conducted by [P. Cortez].

Lastly, it is worth to mention that human tasting is a very complex system and it is not perfectly understood. There are many factors at play, and the insights of these three most important variables does not paint the entire picture. The taste of red wine is a matter of subjective preference, and this might affect the quality of the data set. Regardless, the aim is to predict this preference of certified Portuguese wine experts, and that is the framework to work within. It is surprising that alcohol proved the most important feature in RF and GB for the imbalanced data set. It was also interesting that higher alcohol levels correlated with higher wine scores.

D. Comparison of the Methods

Out of the three models bagging, random forest, and gradient boosting, it is actually expected that RF performs better than bagging. It is also expected that GB is able to perform even better than RF, because of its ability to reduce bias. The results must be split in two separate categories: overall accuracy and accuracy per category. For both of these metrics, it is worth to mention the similarities in the results. However, when comparing the various methods, one must attempt to contrast the small differences that there are.

When considering the overall accuracy. RF performs best. This is not consistent with the expectations based on theory, because GB should be able to perform even better. A possible explanation could be that the models are not tuned properly or that the data set is not large enough.

The accuracy per category is a little different from overall accuracy. All models perform very well on the scores 5, 6 and 7. The accuracy score on the categories 3 and 8 might be unreliable results, because of the few data points. But, the difficult task relates to the very relevant problem of actually identifying these wines. The upscaled data set must be considered, because the imbalanced data set performed poorly on the outer scores. In addition, the tolerance $T = 1.0$ will also be considered, because the aim is to identify bad or good (e.g. a score 3 wine should at most be classified as a 4). As mentioned earlier, bagging was best in identifying the bad wines, whereas GB was best in identifying the good ones.

E. Further Improvements

The most important future improvement when studying this data set with these exact methods, is to perform a more nuanced tuning of the model parameters. On the other side, it could be interesting to bring in other machine learning methods.

It would be very interesting to analyse a larger data set, perhaps including wine from all around the world instead of only the Vinho Verde district. Bringing in more wines does also mean a more broad and useful result.

Lastly, a more detailed study of the feature importances could be valuable for the wine industry.

VI. CONCLUSION

Three ensemble machine learning methods, bagging, random forest, and gradient boosting, were applied to analyse red wine quality scores. The data set contained 11 physicochemical predictors and a categorical output: quality score (0-10) which was determined with blind tests of at least three certified wine experts.

Due to the imbalanced nature of the outcomes, the most difficult task was to identify the worst and the best wines. The best overall accuracy of predicting the exact correct score was 66.9%, and it was achieved by the RF model. Confusion matrices from the three models showed that the misclassified wines was more likely to be identified as a more common score (5 and 6 being the most common scores).

Out of the 11 physicochemical variables: alcohol, sulphates, and volatile acidity were the three most important ones. There were noticeable differences in the values of these three variables, when contrasting score 3 and score 8 wines. It was seemingly wines with higher alcohol percentages, lower volatile acidity, and higher levels of sulphates, which predicted a good wine.

REFERENCES

[1] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning.* Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

[P. Cortez] P. Cortez, A. Cerdeira, F. A. T. M. J. R. Modeling wine preferences by data mining from physicochemical properties. in decision support systems, elsevier, 47(4):547-553. issn: 0167-9236.

[3] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.