Predicting Wine Quality from Physicochemical Measurements

July 20, 2020
By Jeffrey P. Koskulics, Ph.D.

Problem Statement

The quality of wine is usually a subjective score judged by experts who conduct taste tests of wine samples. However, wine quality may be presumed to be dependent on objective measurements such as density, alcohol content, acidity and pH. This raises the question: can wine quality be estimated using physicochemical measurements? What should we expect from a predictive model?

In this project, I analyze a set of *vinho verde*, a lightly carbonated, low-alcohol content wine from Northern Portugal, of both red and white varieties. The dataset is the subject of an analysis on data mining.[1] The dataset is hosted by the UCI Machine Learning Repository in csv files:

The data set includes 4989 instances, each of which is measured in 11 physicochemical features. The wine samples that have been judged for quality on a scale of 0 to 10 representing very bad and very good wine, respectively. The quality score model will be treated as a multiclass classification problem where each score integer represents a discrete category.

Data Wrangling

The data are downloaded in CSV form from the UMI machine learning repository hosted by the University of California at Irvine.

https://archive.ics.uci.edu/ml/datasets/Wine+Quality

The data are in well-maintained condition. No missing or corrupted values are detected. A description of the measurements is separately provided.

[1] Cortez, Paulo, et al. "Modeling Wine Preferences by Data Mining from Physicochemical Properties." *Decision Support Systems*, North-Holland, 9 June 2009

Exploratory Data Analysis

The data are loaded and a few samples are observed. With the exception of wine type (red/white), all features are in numerical form. A pair-scatter plot of the features with each point colored by its quality score provides an indication of the distribution of values and possible correlation with features.

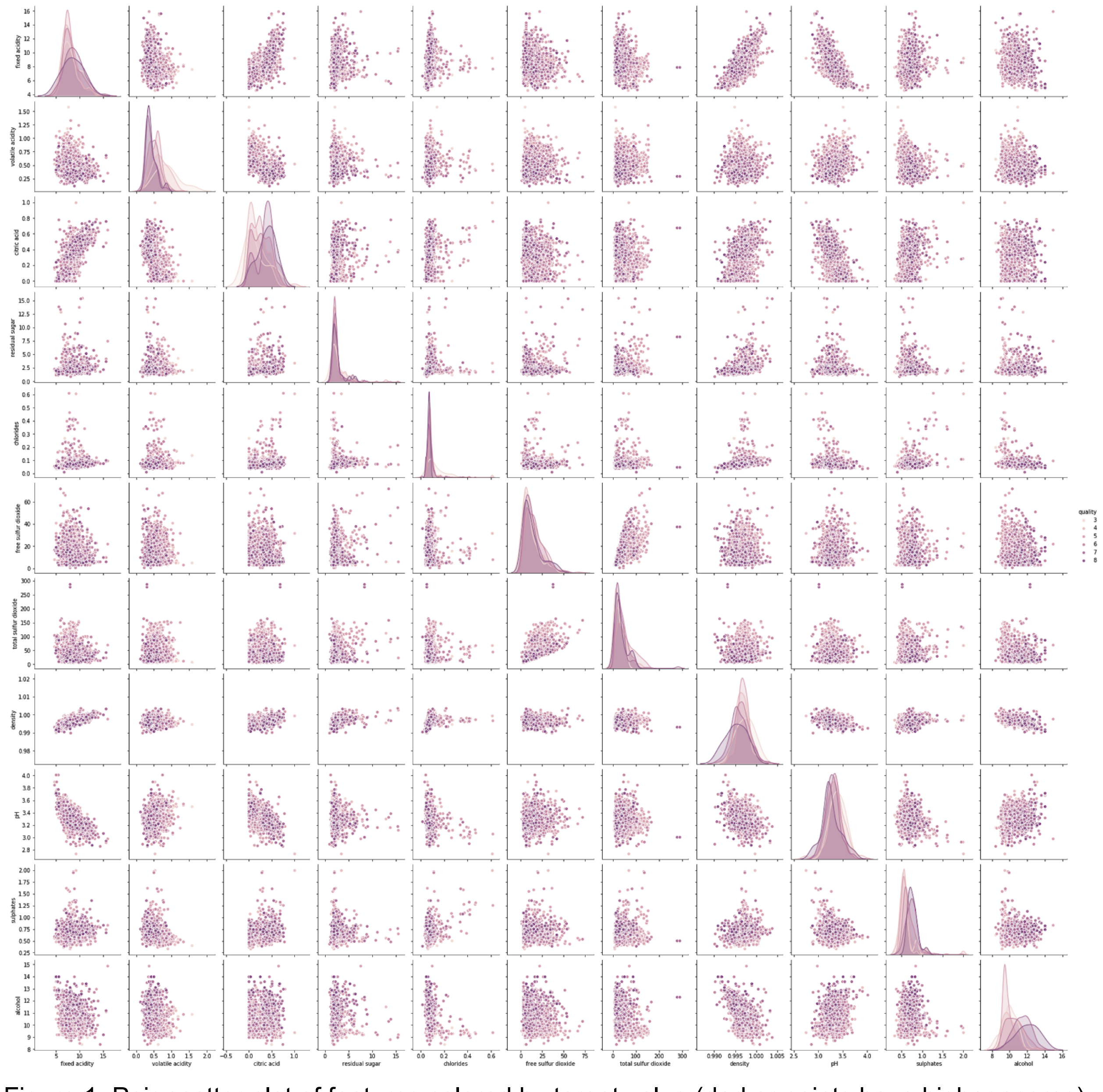


Figure 1. Pair-scatter plot of features colored by target value (darker points have higher scores).

Figure 1 plots pairs of features against each other. From the coloration, it's not apparent that quality correlates strongly with the target. Most of the distributions look reasonably well-clustered with few or no outlying values.

Imbalanced Target

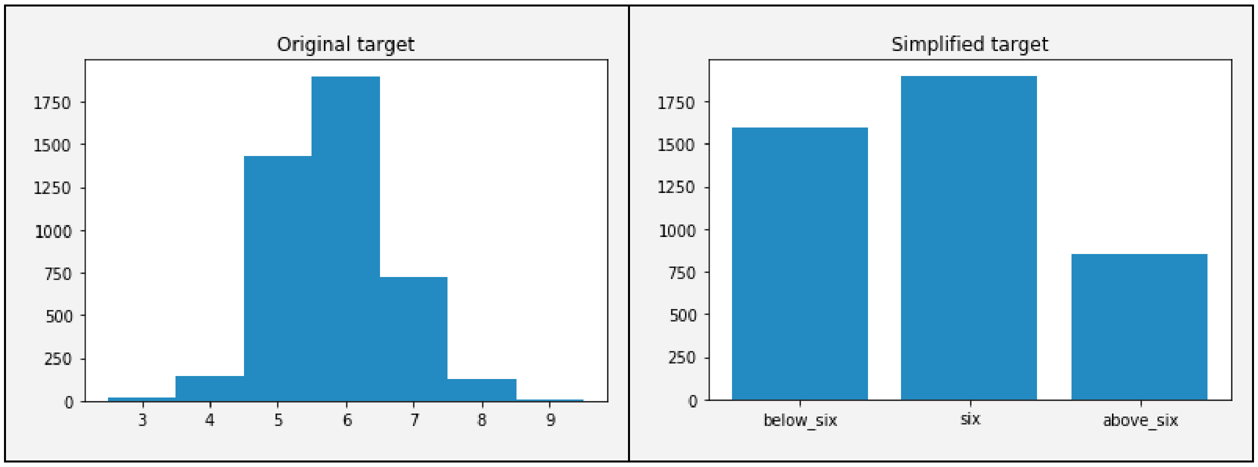


Figure 3. Histogram showing the original and simplified wine quality targets, left and right, respectively.

The data are not distributed uniformly along the span of values that we're interested in (0 to 10). Moreover, data with extremely low or high scores (3, 4, 8, and 9) are sparse. Most of the examples are in the narrow range from 5 to 7.

One approach to reducing imbalance is to retarget the data. Instead of asking the model to predict a precise score, it may be more accurate to ask if the score is below, equal to, or above the value of 6. The simplified target now has a significant number of data in each class. The classes remain somewhat imbalanced.

Splitting unbalanced data into training and test sets

The data are split into separate training and test sets by random sampling. Since this is a multiclass classification problem, we need to ensure that the target imbalance doesn't adversely impact either the training or test samples. The problem is that random sampling risks not picking any members of a very sparsely populated class which causes the training and test sets to have different numbers of classes.

We have to ensure that representative samples are collected from each class. For this reason, the train test split was performed with "stratification" turned on. Stratification ensures that samples are randomly selected from each category according to their relative weight in the population.

Model results

Logistic Regression models were trained and tested using the original or simplified targets. A variety of models were trained using both target schemes. Here a detailed examination of Random Forest is given as it's features were the best performing and were qualitatively similar to many of the other models tested.

Logistic Regression

Logistic regression models were trained on the original and simplified targets. A convergence warning was encountered after training on the original target. The warning disappeared when training on the simplified target. This shows that there are numerical stability problems when treating classes with very sparse data.

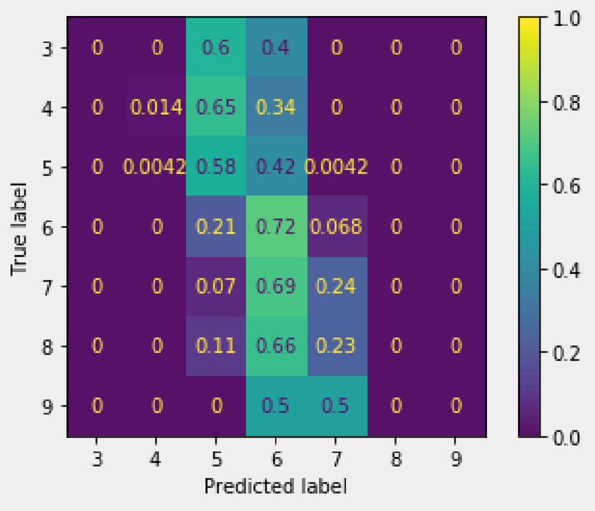
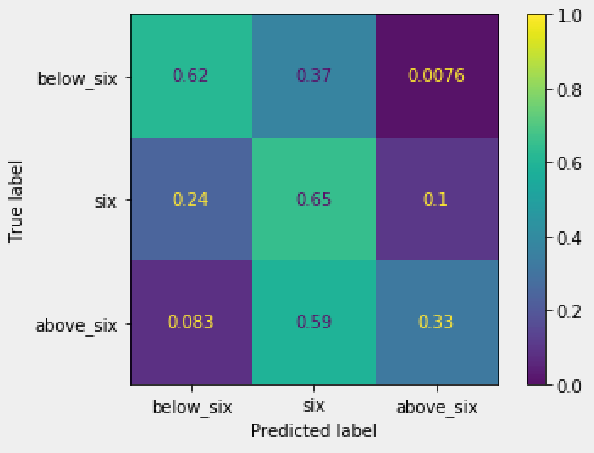
Original target	Simplified target
	

Figure 4. Test result confusion matrices of logistic regression models trained on the original and simplified targets, respectively. Balanced accuracy

The confusion matrix in Figure 4 shows that most predictions fall in a narrow column spanning scores 5, 6 and 7. Most of the predictions are centralized around 6, where the imbalanced training data scores are concentrated. Only a handful of 4's and no 3's, 8's or 9's are predicted.

On the other hand, there is indication that the model has at least some predictive power of wine quality scores. Here's one way to look at the confusion matrix. The value of diagonal elements (on the downward sloping diagonal line from top-left to bottom-right corners) represent true positive predictions and off-diagonal elements represent false negatives. Highly performing classifiers should have diagonal and off-diagonal elements approaching 1 and 0, respectively.

In figure 4, the matrix has a clear downward-left slope, but it only intersects the true positive diagonal around 5-6's. The model is correctly predicting in the right direction, but it misses a lot on the extreme values. The models biggest value is in avoiding gross misclassification, where a very low true quality is predicted as being very high. The dark upper right and lower left corner are exemplary of this predictive power.

Random forest

Random forests are models constructed from an ensemble of decision tree classifiers. Each tree is trained on a random bootstrap sample. The result of all trees is aggregated to produce a single final estimate. The method of bootstrap sampling can be used to construct trees that were trained on either unbalanced or balanced samples. A balanced bootstrap sample can be obtained by over- or under-sampling.

In this case, we compare a standard (unbalanced) forest with a balanced random forest from the imbalanced learn library (imblearn). The balanced random forest trees are trained on undersampled bootstrap samples.

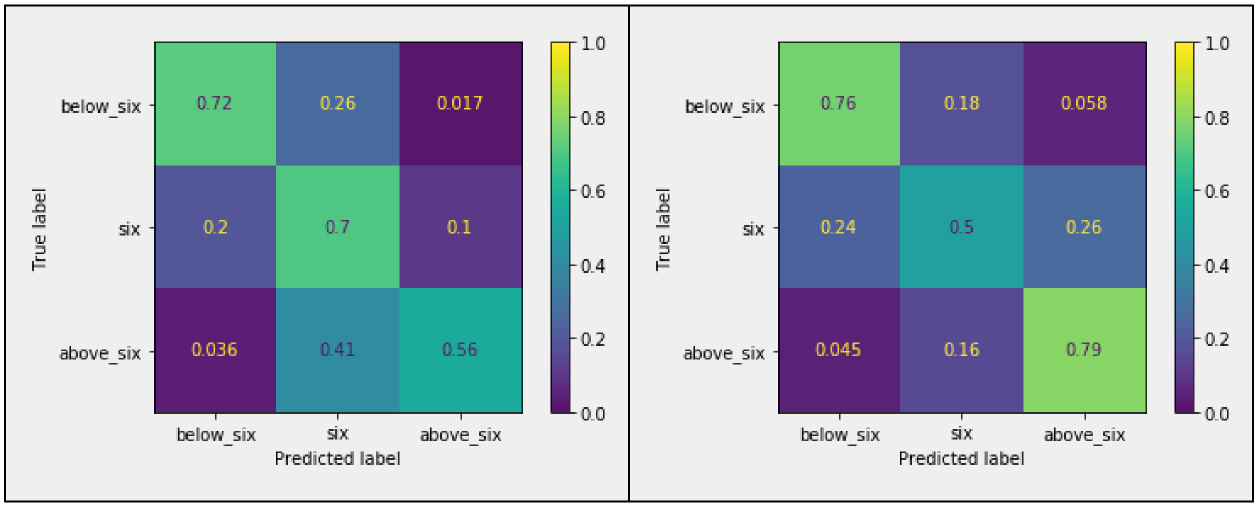


Figure 5. Test result confusion matrices for Random Forest models trained using the unbalanced (left) and balanced (right) bootstrap samples.

Eyeballing figure 5, the left confusion matrix (balanced random forest) appears more symmetric about the diagonal than the right. The balanced forest improves upon the scores of the less-populous targets at the expense of accuracy of the largest-population target (six).

Table 1: Random forest performance

Metric	Unbalanced	Balanced

Average accuracy	0.68	0.66
Balanced accuracy	0.65	0.69

There is a tradeoff between average and balanced accuracy on the unbalanced and balanced forests. Unbalanced models get the top performance for the largest target populations and balanced models do better on smaller targets.

Conclusions

It's clear that wine quality is predictable in some part by a simple set of physicochemical measurements. The best performance comes in distinguishing between very high and low, the model tends to underpredict. Part of the problem is in the data sparsity, being dominated by wines with scores of 5, 6, or 7. This problem is called imbalanced target.

The target imbalance was treated in two ways. Partial balancing is accomplished by simplifying the classification scheme into three classes: below, at, or above six. This has the effect of increasing the model accuracy and balanced accuracy. Target simplification also has the effect of communicating the limitations on the predictive power of the model to users who may expect more.

Full balancing was treated using random undersampling when drawing bootstrap samples for a random forest. Here a tradeoff becomes apparent where overall accuracy declines with undersampling, but the predictions become much more symmetric. Resolution of this tradeoff should consider the precise formulation of the question that the model is asked to answer. For example, if it's more important to identify true 6's the unbalanced model should be used. However, if the model is asked to eliminate the highest or lowest scoring samples, a balanced forest is more appropriate.