Final Results

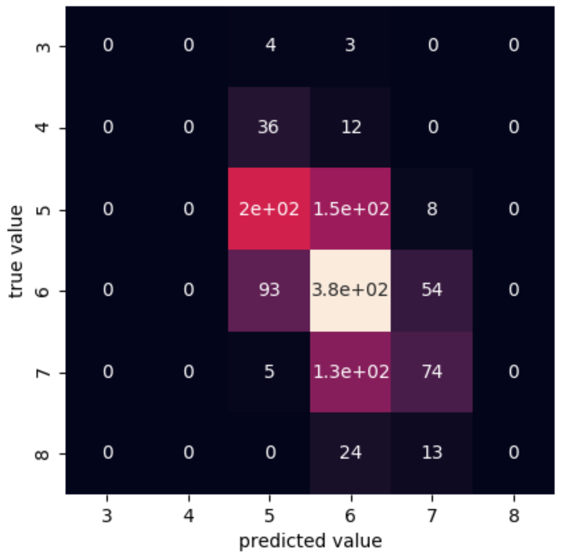
Training Results

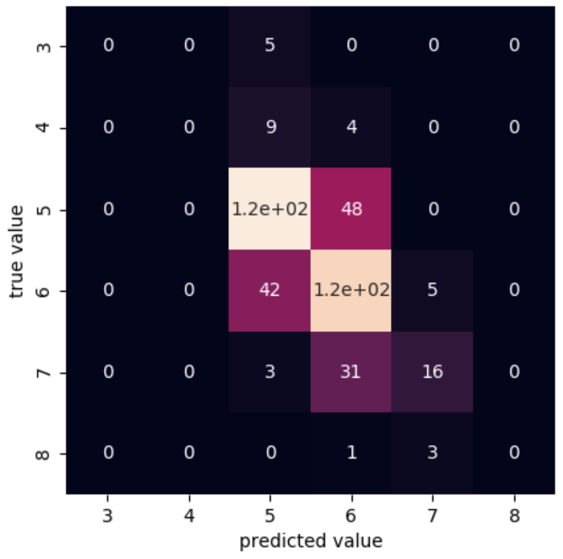
In order to improve on my fairly disappointing initial results, I had to tackle the issue of over fitting in the models. At Tiff's recommendation, I used grid search to tune my hyper parameters in an effort to create a more useable model. I began with a randomized search using a wide range of possible values, then based on the output of this I reduced the ranges and found the following sets of hyper parameters using an extensive search;

White: {n_estimators=800, min_samples_split=11, min_samples_leaf=2, max_features='sqrt', max_depth=70, bootstrap=True}

Red: {n_estimators=1000, min_samples_split=5, min_samples_leaf=4, max_features='sqrt', max_depth=70, bootstrap=True}

However, when I tested the models made with these parameters, I found little change in the test accuracies and precisions (accuracy: white ~55%, red ~63%; precision: white ~58%, red ~65%), and the overfitting issue remained (as shown in the confusion matrices below; white left, red right).

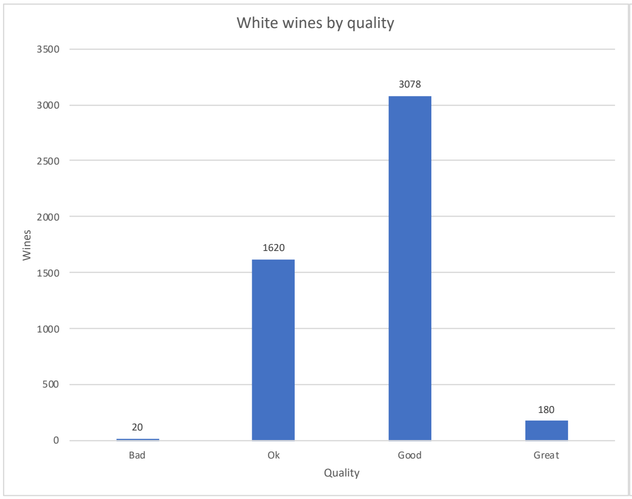


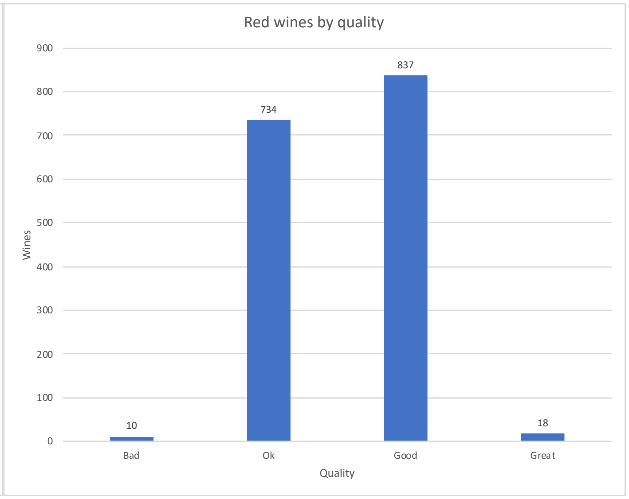


This was quite disappointing, as I continued to receive poor predictions despite my efforts. I thus decided to alter my approach, I broke up the ratings into 4 classes:

quality<=3 is Bad, 3<quality<=5 is Ok, 5<quality<=7 is Good, quality>7 is Great.

This doesn't fix the massive class imbalance (as seen in the graphs below), however it does mean that the accuracy required for usability is decreased and the model is no longer required to perform extrapolation to predict extreme values.



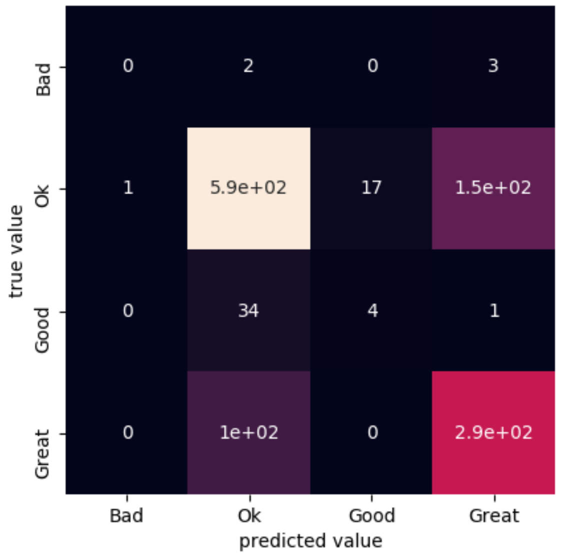


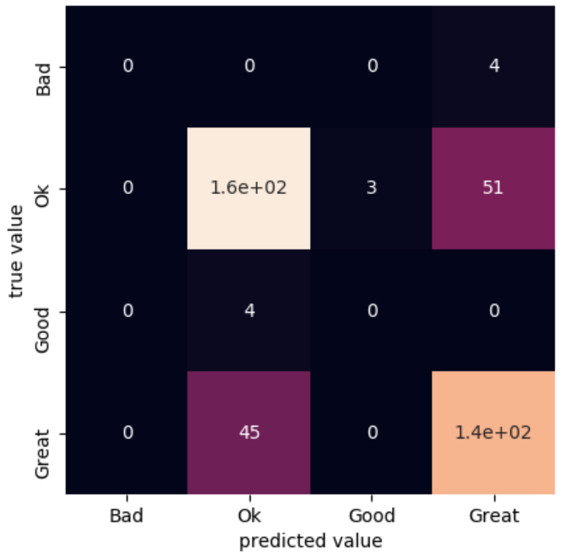
I repeated the process of random and grid search to tune the hyper-parameters; I scored variations based on pecision_macro, resulting in the following set of values:

White: {n_estimators=600, min_samples_split=7, min_samples_leaf=7, max_features='sqrt', max_depth=15, bootstrap=True, class_weight="balanced"}

Red: {n_estimators=600, min_samples_split=2, min_samples_leaf=3, max_features='auto', max_depth=10, bootstrap=True, class_weight="balanced_subsample"}

These values resulted in test accuracies of ~75% for white and ~75% for red (training accuracies of ~86% for both), precision values of ~42% for white and ~39% for red. There is still a large amount of misclassification, as seen in the confusion matrices:





Through this testing, I learned that the values assigned to n_estimators, min_samples_split, and min_samples_leaf had the greatest effect on the model. The importance of n_estimators is simple to understand as the larger this value is, the greater the variability the model is equipped to handle. The other two must be optimized to find the balance between under and over fitting;

when these are reduced, the model is more sensitive to outliers as it takes fewer values to create a branch/leaf, but when the values are too high then some

For this random forest classifier, the training attempts to minimize the Gini impurity. This value represents how likely a new data point is to be misclassified. It is calculated as the sum of the probability of any class being predicted multiplied by the probability of the class not being the one selected.

This final model is still flawed, however it is more useable than my initial regressor. I have deviated from my initial goal, but this project has not been a total failure. To extend this project, I would like to search for similar datasets that I might be able to use to improve my model's accuracy, especially with respect to extreme values. It might also be interesting to create some sort of visualization of how the physiochemical properties affect/are reliant on each other.

Demonstration Proposal

I intend to create a relatively simple web-app which will allow users to select values for all the physiochemical properties of a red or white wine (there will also be a selector for red/white), these values will be run through my trained model to estimate the quality of a wine with set property values. I will create this app using Flask and will do my best to design an appealing landing page as a front end (I will likely use some online templates as inspiration as front-end development is a significant weakness of mine). I intend to load the files for this onto McGill's student server and have them accessible at cs.mcgill.ca/~gpope (my personal domain). I have very minimal web-app development experience, as such I will make heavy use of online tutorials and examples and will also attend any workshops on the topic if there are some available.