A Pour Choice: Classifying Portuguese Wine Quality with Supervised Learning

Benjamin Berkman (bjb433), Zixiao Chen (zc2194), Stanley Sukanto (ss14358), David Trakhtenberg (dt2229)
DS-GA-1001 | New York University

https://github.com/dtrakht/1001-Team-Project

Abstract: Wine sales have dramatically increased during the coronavirus pandemic. According to the Washington Post, Americans purchased nearly 30 percent more wine in June 2020 than in March 2020[1]. Given the increased demand, quick and accurate classification of a wine's quality is important in order to ensure a positive consumer experience and fair price at market. This paper studies if the physicochemical properties of wine can sufficiently predict its quality through supervised learning approaches. This has the potential to aid wine makers (oenologists), evaluators, and consumers. We show that a high-performing random forest algorithm is able make these classifications, and we discuss its business implications and deployment potential.

1 Business Understanding

Wine is a unique good that has objective physicochemical properties yet a somewhat subjective quality rating that is marketed to consumers. These physicochemical properties are readily measured via laboratory tests and include measures of alcohol percentage, pH, and residual sugar, among other features explained in this paper. The median value of three assessors' blind tests determines a wine's quality score (Cortez, Cerdeira and Almeida). These scores are marketed to the public, but there is often little transparency in what contributes to a good score. Thus, there exists a need to model the effect of these properties on a wine's quality and to be able to predict a wine's quality given its properties.

A machine learning solution has the potential to address several business problems. First, a supervised learning model will be able to identify which physicochemical properties of wine most directly influence its quality. This may allow oenologists to directly increase certain physicochemical properties in the wine making process. For example, residual sugar – one of the features studied – can be controlled via sugar fermentation in yeasts (Cortez, Cerdeira and Almeida). Additionally, this can serve as a quality control mechanism through the wine scoring process. If the median value of the three assessors' scores significantly differs from the projected score of the model, it could serve as a reason for re-evaluating the quality. Ultimately, human evaluators may be rendered unnecessary given enough training feedback into the learning model. Moreover, a machine learning solution may help test if wine assessors in training are scoring wine on par with the model. This paper also employs a brief unsupervised clustering approach to group similar wines. This may benefit consumers via targeted marketing. A vendor could employ a recommendation system,

[1] https://www.washingtonpost.com/lifestyle/food/wine-sales-have-surged-during-the-pandemic--but-not-for-small-producers/2020/06/19/c7c49cba-b0b6-11ea-8f56-63f38c990077_story.html

recommending wines with similar physicochemical properties given a customer enjoyed a specific wine.

As discussed in the next section, the data in this paper is limited to Portuguese white and red vinho verde wines introduced in 2009[2]. However, this opportunity has the potential to be expanded to all wines that have both measured physicochemical properties and evaluation scores. With technological increases in the wine industry, these properties are increasingly measurable. Additionally, as wine is no longer seen exclusively as a luxury good, a variety of consumer-facing applications, such as CellarTracker[3], provide millions of user ratings for millions of wines. There does not appear to exist a database that maps these user ratings to the physicochemical properties of their respective wines, but this would serve as a valuable future research topic.

The state of the art in solving this problem has largely been limited to ad-hoc studies of small datasets of varying physicochemical properties such as the Portuguese dataset in this discussion. For example, Hopfer et al. measured five quality proxies against sensory attributes, volatile compounds, and elemental composition in Californian wine (Hopfer, Nelson and Ebeler). In other cases, hierarchical clustering was applied to both physicochemical and sensory assessments of Brazilian wine (de Castilhosa, Cattelana and Conti-Silva). However, there does not appear to be a large-scale machine learning model in production that predicts wine quality given its physicochemical properties.

2 Data Understanding

We acquired the datasets from the UCI Machine Learning Repository. The datasets present red and white wine quality with corresponding physicochemical properties. Given their distinct flavor profiles and use cases, the red and white wines are stored in two separate datasets. This paper also includes separate analysis for red and white wines. The red and white wine datasets have 1,599 and 4,898 rows, respectively. Each row represents a distinct wine; therefore, the observations are independent. There are 12 variables in each dataset: 11 attributes describing the wines' physicochemical properties and one target variable, quality (scored from 0 to 10). The 11 features are described in detail in the appendix (Table 5).

Before any data preparation or model building, we conducted exploratory data analysis (EDA) and visualized relevant results. We first evaluated the descriptive statistics of both red wine and white wine datasets (Table 4 in appendix), and then analyzed the correlation between each attribute and the target variable. From a correlation heat map, we observed that red wine quality has a weak positive correlation with alcohol, sulphates, citric acid, and fixed acidity, and has a weak negative correlation with other attributes. Similarly, white wine quality has a weak positive correlation to alcohol, sulphates, pH, and free sulfur dioxide, and has a weak negative correlation with other features (Figure 2 in appendix). We also analyzed the relationship between the features and the target variable via bar charts. If there was any clear increasing or decreasing pattern in the

2 https://archive.ics.uci.edu/ml/datasets/wine+quality

3 https://www.cellartracker.com/

bar chart, we concluded that the input variable has influence on the target variable. This analysis largely confirmed the findings from the correlation heat maps. Several of these relationships are plotted in the appendix (Figure 3).

We also analyzed feature importance and listed the top five features in regard to both Area under the Receiver Operator Curve (AUC) and Mutual Information (MI) (Table 6 in appendix). Based on AUC, the top five features for red wine are volatile acidity, citric acid, total sulfur dioxide, sulphates, and alcohol (for white: chlorides, total sulfur dioxide, density, pH, and alcohol). A Decision Tree was learned to estimate the MI of each feature. Based on MI, the top five features for red wine are volatile acidity, total sulfur dioxide, density, sulphates, and alcohol (for white: volatile acidity, citric acid, total sulfur dioxide, sulphates, and alcohol). Thus, it stands to reason that the volume percent of alcohol in a wine, for example, will affect the sensory taste. An oenologist using the results of this simple exploratory analysis may opt to increase alcohol content, holding all else equal, in an effort to make higher quality wine.

Clustering, an unsupervised learning technique, served as a useful approach to understand the similarity of the wines in the dataset. Red wines and white wines were clustered separately using both hierarchical clustering and k-means clustering. For this specific exercise, the quality of the wines was not considered, only the physicochemical properties. The hierarchical clustering demonstrated that the red wines could efficiently be clustered into two clusters, while the white wines fit nicely into three clusters. K-means clustering then separated the red and white wines into two and three clusters, respectively.

As shown in the appendix (Figure 4), when plotted against density and pH, the clusters do form relatively strong boundaries. As demonstrated by Provost and Fawcett in their discussion of whiskey analytics, this clustering can help inform recommendations (Provost and Fawcett). If a customer enjoyed a specific wine, a vendor could recommend a wine with similar properties (and therefore within the same dendrogram node or k-means cluster) which in turn may have a similar taste profile. If a vendor had a limited budget, it could stock only wines within one cluster (to serve as a specialized vendor), or consciously pick wines from each cluster (to ensure a diverse selection).

With a stronger domain knowledge of the features, we then examined feature independence, a necessary assumption for several predictive modeling algorithms (e.g., Logistic Regression). Multicollinearity occurs when one independent variable can be predicted from the other independent variables. When multicollinearity is present, small changes in the data may drastically change the coefficient estimates of the regression model. However, it is important to note that multicollinearity does not affect the predictive power of a model.

One way to detect multicollinearity is to use the variance inflation factor (VIF), which is calculated as follows: $VIF = \frac{1}{1-R_j^2}$, where R_j^2 is the R^2 value obtained from regressing the j^{th} predictor on all the other predictors. A VIF higher than 5 or 10 usually indicates a multicollinearity problem. Table 7: VIF in the appendix shows the VIF of the independent variables that have been standardized and transformed. We observed that for red wine, fixed acidity is the feature with the highest VIF, followed closely by density. A solution to

dealing with multicollinearity is to remove several features with the highest VIFs. The appendix shows the VIF of the remaining features after removing "fixed acidity". All of the remaining features have VIF less than five, indicating that there is no multicollinearity between these features. For white wine, we observed that density is the only feature with VIF greater than five.

Since our goal was only to correctly predict the quality of wine given the wine features, the presence of multicollinearity was not a huge problem as we are not concerned with performing statistical inferences on the model (e.g., estimating individual feature importance). A five-fold cross-validation was utilized to determine whether or not fixed acidity and density, for red wine and white wine respectively, should be excluded when building the model. To test this, we compared the performance of two logistic regression models, the first with all of the features available and the second model with all features available except for the feature with the highest VIF. Using AUC as the evaluation metric, the first model performed slightly better for both red (.871 ± .06 vs. .869 ± .06) and white wine (.795 ± .04 vs. .793 ± .04).

3 Data Preparation

The dataset arrived largely pre-packaged for modeling, though we did consider several strategic decisions and feature engineering options. The physicochemical properties and qualities were collected from 2004 to 2007 by a Portuguese professional organization focused on improving the excellence and the brand value of vinho verde wines (Cortez, Cerdeira and Almeida). The data was then stored in a database and provided via UCI to machine learning researchers. The data in its native format contains no missing values. Both the feature and target variables all contain float or numeric field types.

The various feature variables are, however, on different scales (e.g., among the red wines, volatile acidity ranges from .12 to 1.58 while total sulfur dioxide ranges from 6 to 289). One of the first data preparation steps was thus to use Scikit-learn's StandardScaler[4] transformer to standardize features by removing the mean and scaling to unit variance. The data is also natively broken out by red and white wines in two separate files. As the data is hosted in .csv format (with ';' delimiters), it can easily be read into a python notebook.

A first consideration involved whether to combine the red and white wine datasets into a single dataset. On one hand, red and white wines have different tasting notes so they arguably should be analyzed separately. On the other hand, a model that can successfully aggregate the two could serve as a more robust prediction system. Ultimately, we did not combine the two datasets with the rationale that the average customer is more likely to search for a "good red wine" instead of simply a "good wine".

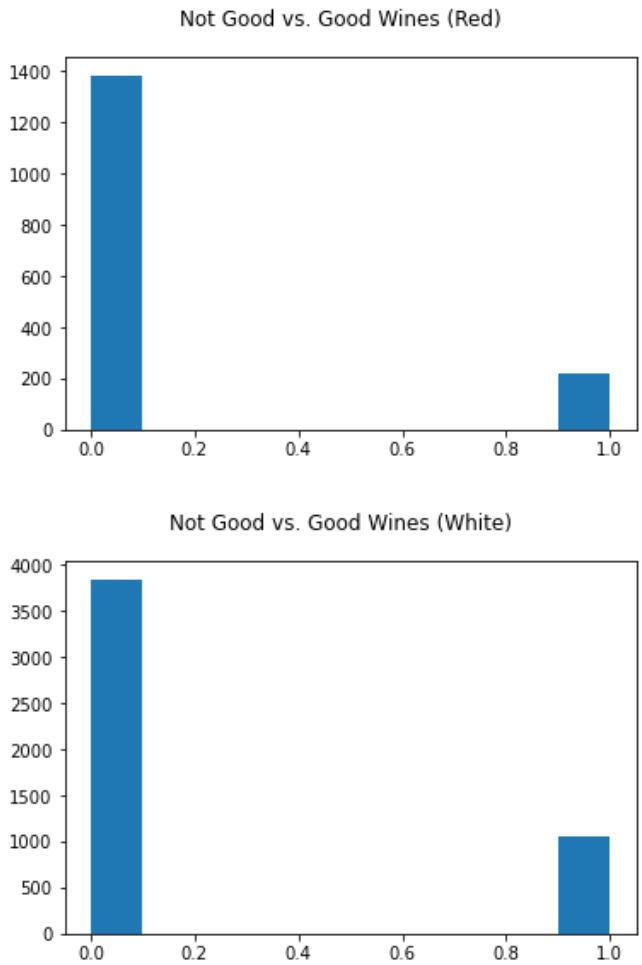
The target variable is defined as the quality of each wine, scored by wine assessors on a scale from 0 (very bad) to 10 (excellent). We decided to frame this business problem as a binary classification example. Instead of developing a

4 https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

system to predict the specific numeric quality of a wine, we instead opted to predict simply if a wine would be "good" or not (i.e., "poor quality"). This approach allowed our team to employ a more diverse set of algorithms. Additionally, classifying a wine as good or of poor quality simplifies the business implications. Still, the wines that the models predict with a very strong

probability of being a good wine may be indicative of being a particularly good bottle, and therefore the model should be able to still segregate minimally good versus exceptional bottles. We classified all wines scored with a quality less than 7 to be a not good wine (binary class 0), and those with a quality greater than or equal to 7 to be a good wine (binary class 1). 14% of reds and 22% of white wines are then coded as "good".

Figure 1: Red and white wine target class distribution



Finally, during EDA, we discovered that many of the features have a right-skewed distribution. This may be due to the chemical nature of the features. With residual sugar, for example, wine has varying levels of sugar after fermentation ends but various instances bunch up around 1 g/L. To normalize the skewed features, a log transformation proved most successful. Other methods may include a square root transform as well as a box-cox transform. Specifically, for our analysis, log-transforms of alcohol, residual sugar, and free sulfur dioxide remedied the adverse effects of the tail region consisting of outliers.

As a final data preparation step, the two datasets were split into training and test sets. Following data splitting standards, two thirds of the data were randomly assigned to the training set; the remaining third was assigned to the test set.

4 Modeling & Evaluation

By framing this business scenario as a binary classification problem instead of a regression problem, we enabled the use of a number of classification algorithms. Given the relatively small size of the data, we were not terribly constrained by computational time to train, test, tune, and retest a number of algorithms. With that in mind, we elected initially to train and test random forest (RF), logistic regression (LR), Gaussian naive Bayes (GNB), support vector machine (SVM), and K-nearest neighbor (K-NN) classifiers. We also separately evaluated the performance of gradient-boosting a decision tree.

We considered the assumptions, benefits, and possible issues with each algorithm. For example, random forests are

extremely robust models that perform well in many situations and do not make many assumptions about the underlying data distribution; however, training time may be slow (as is the case for support vector machines). Logistic regression handles class imbalance and small datasets well but requires little multicollinearity and a linear relationship between the logit of the label and each independent feature. The GNB classifier assumes conditional independence and that the independent variables are conditionally normally distributed, but also often performs well; however, it may be unlikely that the physicochemical properties are conditionally independent. K-NN is a simple, interpretable algorithm that performs well on small datasets, and may nicely fit the structured groupings of the physicochemical properties.

Before training or evaluating any models, choosing an appropriate evaluation framework was necessary. Accuracy was not appropriate in this case due to the target class imbalance. We ultimately selected AUC for the following reasons:

- It is base rate invariant: This means we can compare the AUC performance on the white and red wine datasets.
- It is class distribution independent: This means the skewed class distribution of the target variable will not seriously influence the prediction result.
- It is easily interpretable and nicely bounded and thus effective for communicating to business stakeholders
- It is advantageous for ranking predictions. This benefits our business case by allowing a vendor using the model to identify wines that may be exceptionally good, even if they are only rated as "good" by an oenologist. Additionally, a vendor concerned with only stocking good wines can select wines that the model predicts with high probability are of good quality.

In pursuit of an optimal model, we aimed for generalization while avoiding overfitting. Cross-validation helped assess how the models performed against different folds of the data. This method ensured that our models generalize to various holdout sets – not just one test set. To this end, we outlined the following evaluation process:

1. Trained baseline model and evaluated performance using k-fold cross validation.
2. Performed hyperparameter tuning for each model using k-fold grid search with a defined range of possible hyperparameters and recorded respective parameters that maximize AUC.
3. Retrained each model with the optimal hyperparameters and evaluated performance, again using k-fold cross validation.

In selecting a "k" for cross validation, we considered the size of the red and white datasets and the train/test split. With a relatively small dataset, we chose k = 5 which allowed us to maintain a balance between bias (low value of k) and variance (high value of k), and to train and test our models on each fold of the data.

We trained the five models with default hyperparameters as base models, evaluated performance via AUC, and plotted the ROC curve for each model (Figure 5 in appendix). The ROC curve plots true positive rate against false positive rate at different classification thresholds. We then calculated the mean AUC and standard deviation across the 5 folds, shown in Table 1. The random forest performed notably better than the other four models, and the ROC curves (Figure 5 in appendix) indicate it outperforms at a majority of classification thresholds.

Table 1: Baseline model performance

	Red Wines		White Wines	
Model	Mean AUC	Mean Standard Deviation	Mean AUC	Mean Standard Deviation
RF	0.909	0.06	.894	0.03
LR	0.871	0.06	.795	0.04
GNB	0.861	0.05	.787	0.05
SVM	0.868	0.10	.835	0.02
K-NN	0.806	0.09	.815	0.04

To perform hyperparameter tuning, we employed grid search to take a range of values for each hyperparameter and construct a grid of all possible combinations in an attempt to maximize AUC. For example, with LR, we tuned 'C' (the regularization weight), 'penalty' (specific norm for penalization), and 'solver' (specific algorithm used), which converged to an optimal set at C = 0.1, penalty = 'L1', and solver = 'liblinear' for red wine and C = 10, penalty = 'L2', and solver = 'saga' for white wine. An optimal set is defined by the hyperparameters that yield the highest AUC. In defining the range for 'C' (and other continuous hyperparameters), we made sure to select values that would span model complexity from low to high (thus varying bias and variance). Applying this approach to each algorithm, we were able to find the optimal set of hyperparameters to retrain the models. A summary of the tested and optimal hyperparameters can be found in the appendix (Table 8). The ROC curves are provided in the appendix (Figure 6).

As a final tuning step, we also evaluated if we could notably improve the random forest's performance using a gradient boosting method (XGB). In this algorithm, an initial decision tree is fit, then a decision tree is fit to the error of the prior model. This is repeated until a stopping criterion is met. Despite the strong performance often seen through this approach, the gradient boosted model did not improve the AUC of the classification and is not justified given its increase in training and tuning time.

Following hyperparameter tuning, the random forest model returned the highest AUC (Table 2). One explanation might be related to the shape of our input data. Random forest is able to work with data with n instances and m features (an n by m matrix). This may explain why random forest outperforms SVM in our case, since when input data is not an n by n matrix, SVM will have to construct an n by n matrix as an intermediate step. Our datasets are 12 by 1,599 and 12 by 4,898 matrices, which may work better with random forests.

Table 2: Model performance after grid search

	Red Wines		White Wines	
Model	Mean AUC	Mean Standard Deviation	Mean AUC	Mean Standard Deviation
RF	0.911	0.06	.894	0.02
LR	0.874	0.06	.795	0.04
GNB	0.869	0.05	.787	0.05
SVM	0.868	0.10	.835	0.02
K-NN	0.874	0.09	.882	0.03

Another useful trait of the random forest is that it is parallelizable. This means the process can be split into several machines to run. As we used grid search for hyperparameter tuning, a time-consuming method in general, the computation time of a random forest might not be as large of an issue. Logistic regression models also handle imbalance robustly, and the LR model was the second strongest on our datasets.

Random forests are often difficult to interpret. In an effort to better understand our "black-box" model, we employed the LIME (Local Interpretable Model-Agnostic Explanations) technique. LIME's authors posit two types of trust when one explains a machine learning model: trusting a prediction and trusting a model. Users of a model will gain

trust when it is easier to interpret (Ribeiro, Singh and Guestrin). Further, we are concerned with local fidelity: features that are significant in the global problem may not be significant locally.

LIME perturbs data around a local instance to build a model, so we used this explanation in a local sense, not a global one. We applied LIME to the tuned random forest classifier. For the red wine dataset, randomizing which test set instance to explain, we found that high sulphate content has positive correlation with high red wine quality and low alcohol content negatively correlates with high red wine quality. For the white wine dataset, we similarly found that low alcohol content negatively correlates with high white wine quality and a high level of chlorides negatively correlates with high white wine quality (Figure 7 in appendix). This supports our previous mutual information analysis. With a more confident understanding of how our physicochemical properties impact the quality of vinho verde, we can begin to inform our business use case.

Next, a lift curve informed us how much better our models performed over randomly guessing. The y-axis of a lift curve, Lift, is obtained from dividing the hit rate of the model by the hit rate of a random classifier at a particular point, x. The x-axis of a lift curve is the percentage of the test population that is targeted, ranked using the classifier. The curve is particularly useful when a firm faces a budgetary constraint and is not able to target all of the population. Assume that there is a wine distributor that wants to stock up on vinho verde wines, instead of randomly guessing which wines to purchase. The seller can utilize the lift curve, pick the model with the highest lift at a particular point k based on her budget, and choose the top k% of wines ranked by the classifier. Doing so will provide the seller with the most bang for her buck.

Figure 8 in the appendix shows the lift curves of both the red and white wines. For both, random forest and k-NN perform very well in comparison with the other classifiers. We observe a very intriguing result, in which k-NN is superior up through about 20% of the test instances, after which RF takes over and performs better. Thus, small wine sellers who can only afford to choose from a few vinho verde wine variants (less than 20% of the total variants) should opt for the k-NN classifier, and medium to large wine sellers should use RF when picking wines to build their inventories.

We also analyzed a confusion matrix at the 0.5 classification threshold to measure false negatives and false positives. In our business case, we would like to know how many poor-quality wines are mislabeled as good wine (false positive), and how many good wines are mislabeled as poor-quality wine (false negative). For example, selling mislabeled poor wine as good wine may jeopardize the brand reputation of a vendor. Selling mislabeled good wine as poor-quality wine would lead to mispricing and failure in cost management. Therefore, for quality control and management reasons, we would like to see the proportion of FPR and FNR. As shown in Table 3, both models have much higher false positive rates than false negative rates. This has the potential to hurt business (underpricing wine) but please customers (underpricing good wine). Future exploration should analyze different classification thresholds to better handle these Type I and Type II errors.

Table 3: Red and white wine confusion matrices

Red Wine	Actual: Poor	Actual: Good
Predicted: Poor	438	13 (FPR: 29%)
Predicted: Good	46 (FNR: 10%)	31

White Wine	Actual: Poor	Actual: Good
Predicted: Poor	1214	142 (FPR: 17%)
Predicted: Good	159 (FNR: 12%)	202

We conducted a cost-benefit analysis based on the confusion matrix to assess the cost associated with mislabeling. Assume a bottle of high-quality wine is 100 dollars and a bottle of poor-quality wine is 10 dollars. Let the correct classification case be the base scenario where cost/benefit equals 0. Mislabeling a good wine as a poor-quality wine (false negative) and selling it would cause direct revenue loss of 90 dollars for the retailer. Although the cost of mislabeling a poor-quality wine as a good wine (false positive) cannot be monetized, this type of mislabeling would jeopardize the brand name and the trust from customers.

To provide one example in which this model may solve a business problem, we focused on the specific use case of applying our data mining to a wine distributor, fictitiously named Wine Lovers. Our analysis is specifically for Portuguese vinho verde wines, though future work could focus on a larger, broader wine dataset. By implementing our tuned random forest to Wine Lovers' purchasing department, the department can find mismatches in price and predicted quality. For example, if bottle A received a "Good" quality classification, the distributor could sell bottle A to wine shops for an increased value as consumer demand for that bottle should in theory increase due to its higher quality.

If consumer taste does not drive demand, then the perceived value of increasing bottle A's price may drive demand aided by a quantifiable justification (i.e., our model's classification). The benefits of hierarchical clustering (explained earlier) can then be applied by Wine Lovers as well. By choosing a lower and a higher priced bottle from each cluster, Wine Lovers can provide a diversified set of wines – for example, a low-priced wine that may taste similarly to high priced wine given its similar physicochemical properties. This would allow consumers to sample a diverse set of wines.

5. Deployment

After the requisite featuring engineering, modeling, and evaluation, Wine Lovers can deploy the optimal model on its inventory as a method to set prices when selling to wine vendors. Specifically, it can use the model to quickly analyze the physicochemical properties of its stock, predict the quality for each bottle, and use the predictions and the prediction probabilities to establish the value of each wine. In turn, it could calculate its expected profits, breakeven point, and return on investment.

Due to the smaller scale of its wine inventory, Wine Lovers could compute batch predictions either ad hoc or on a continuously scheduled basis. This offline deployment allows for a rush-free serving container as opposed to an online one that streams in real-time. To be able to scale the pipeline based on demand, Wine Lovers could implement a containerized solution like Kubernetes where all the model artifacts are packaged in appropriate containers. They could package the random forest model along with metadata files and model parameter files. Wine Lovers' data processing would not require the implementation of Spark, but a

parallelized solution could be useful for larger wine distributors that sell wine from all over the world.

Additionally, a scalable database solution such as Cassandra would allow for the model repository to be stored with the full set of physicochemical features in a separate database. The batch predictions would be available for SQL querying once loaded by the serving container. To ensure that the model is performing optimally, we must monitor its predictions and ensure the production environment is operating on accurate and relevant data. Wine Lovers could log metrics on its model by sampling a percentage of predictions and comparing that sample with wine assessors' quality score. To create a data science feedback product, we could interact with wine shops to see how their customers enjoyed individual wines and score our model appropriately. This would inform future model tuning and how the data generating process may be improved.

As we are implementing a model for a substance (alcohol), we have to be mindful of the ethical ramifications. Through our MI analysis as well as the LIME explanation model, we found that alcohol is one of the most predictive properties of wine quality. If we employ our model to help winemakers create wine that is of better quality by increasing alcohol content, we must be mindful of how we market that wine to consumers as it can pose a bigger health threat.

A potential risk associated with this model is that of concept drift. If the model was trained on 2019's wines, for example, and then deployed in 2020, there could be external factors that influence a wine's quality. For example, if the weather in 2019 was drastically different than that in 2020, the physicochemical properties leading to good wine in 2019 may be different than those in 2020. A distributor must thus be conscious of other components affecting a wine's quality, such as weather and market trends. To mitigate this, ideally the model in deployment would be trained on a year with similar weather and soil conditions.

6. Conclusion & Future Work

In this paper we have introduced the Portuguese white and red vinho verde wine dataset. Framed as a binary classification problem, we sought to develop a model to identify if a wine would be of good or poor quality given its physicochemical properties. Using AUC as an evaluation metric, we developed a strong-performing random forest model. We suggested several use cases for such a model. For example, a wine distributor could use the physicochemical properties to evaluate the qualities of the wine and thus use that information to set prices it markets to vendors. As Wine Lovers grows its business due to increased demand, it could benefit from improvements to its classification pipeline that is out of the scope of this paper. For example, to detect the few excellent or very poor wines in the dataset, Wine Lovers could employ an outlier detection algorithm. Additionally, to achieve a better sense of false positives and false negatives, it could analyze different classification thresholds. The deployment of a wine quality classification model is a novel concept that is largely unimplemented. Through future work, however, this has the potential to enhance the wine purchasing experiences of many.

Works Referenced:

Cortez, Paulo, et al. "Modeling wine preferences by data mining from physicochemical properties." *Decision Support Systems* (2009): 4800-058.

de Castilhosa, Maurício, et al. "Influence of two different vinification procedures on the physicochemical and sensory properties of Brazilian non-Vitis vinifera red wines." *LWT - Food Science and Technology* (2013): 360-366.

Hopfer, Helen, et al. "Correlating Wine Quality Indicators to Chemical and Sensory Measurements ." *Molecules* (2015): 8453-8483.

Provost, Foster and Tom Fawcett. *Data Science for Business*. O'Reilly, 2013.

Ribeiro, Marco Tulio, Sameer Singh and Carlos Guestrin. *"Why Should I Trust You?" Explaining the Predictions of Any Classifier*. arXiv:1602.04938 , 2016.

Appendix:

Team Contributions:

Ben: Business understanding, clustering, random forest and gradient boosting models
Stanley: VIF analysis, Naïve Bayes model, lift analysis
David: Data preparation, deployment, logistic regression model, LIME analysis
Zixiao: Data understanding, model evaluation, K-NN&SVM models

Data Understanding:

Table 4: Red (top) and white (bottom) wine descriptive statistics

	fixed acidity	*volatile acidity*	*citric acid*	*residual sugar*	*chlorides*	*free sulfur dioxide*	*total sulfur dioxide*	*density*	*pH*	*sulphates*	*alcohol*	*quality*
count	1599	1599	1599	1599	1599	1599	1599	1599	1599	1599	1599	1599
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.636023
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668	0.807569
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.000000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.000000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.000000
max	15.90	1.58000	1.000000	15.500000	0.611000	72.000000	289.000	1.003690	4.010000	2.000000	14.900000	8.000000

	fixed acidity	*volatile acidity*	*citric acid*	*residual sugar*	*chlorides*	*free sulfur dioxide*	*total sulfur dioxide*	*density*	*pH*	*sulphates*	*alcohol*	*quality*
count	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898
mean	6.85478	0.278241	0.334192	6.391415	0.045772	35.308085	138.3606	0.994027	3.188267	0.489847	10.514267	5.877909
std	0.84386	0.100795	0.121020	5.072058	0.021848	17.007137	42.498065	0.002991	0.151001	0.114126	1.230621	0.885639
min	3.80000	0.080000	0.000000	0.600000	0.009000	2.000000	9.0000	0.987110	2.720000	0.220000	8.000000	3.000000
25%	6.30000	0.210000	0.270000	1.700000	0.036000	23.000000	108.0000	0.991723	3.090000	0.410000	9.500000	5.000000
50%	6.80000	0.260000	0.320000	5.200000	0.043000	34.000000	134.0000	0.993740	3.180000	0.470000	10.400000	6.000000
75%	7.30000	0.320000	0.390000	9.900000	0.050000	46.000000	167.0000	0.996100	3.280000	0.550000	11.400000	6.000000
max	14.200	1.100000	1.660000	65.800000	0.346000	289.0000	440.0000	1.038980	3.820000	1.080000	14.200000	9.000000

Table 5: Dataset features and descriptions

Attribute	Description	Unit
Fixed Acidity	Acids enhance the fundamental tastes in wine, such as sourness and tartness. Most acids in wine are fixed acids.[5]	mg/L
Volatile Acidity	The steam distillable acids in wine. Usually kept at low level to avoid undesirable aromas.[6]	g/L
Citric Acid	A less common acid in wine. Used for stabilization purposes to prevent ferric hazes and should be kept below 1 g/L.[7]	g/L
Residual Sugar	The natural sugar leftover after alcoholic fermentation finishes.[8]	g/L
Chlorides	The major contributor to saltiness in wine. The common range is from 2 to 4 g/L.[9]	g/L

[5] https://waterhouse.ucdavis.edu/whats-in-wine/fixed-acidity
[6] https://waterhouse.ucdavis.edu/whats-in-wine/volatile-acidity
[7] https://www.randoxfood.com/why-is-testing-for-citric-acid-important-in-winemaking/
[8] https://winefolly.com/deep-dive/what-is-residual-sugar-in-wine/
[9] https://www.researchgate.net/publication/276444447_Chloride_concentration_in_red_wines_Influence_of_terroir_and_grape_type

Free Sulfur Dioxide	The amount of free SO2 in wine. Used for preventing microbial growth and oxidation. [10]	mg/L
Total Sulfur Dioxide	The portion of free SO2 and bound SO2 to other chemicals.[11]	mg/L
Density	The weight per volume of a material. Density of wine is mainly determined by the concentration of alcohol, sugar, and glycerol.[12]	g/mL
pH	Measures ripeness versus acidity. The common pH range for wine is from 3.3 to 3.6. The higher the pH, the lower the acidity.[13]	1-14
Sulphates	The amount of salts of sulfuric acid in wine. Used for inhibiting microbial growth.[14]	g/L
Alcohol	The percent alcohol content of the wine.	%
Quality	A score measures wine quality.	0-10

Red Wine: *White Wine:*

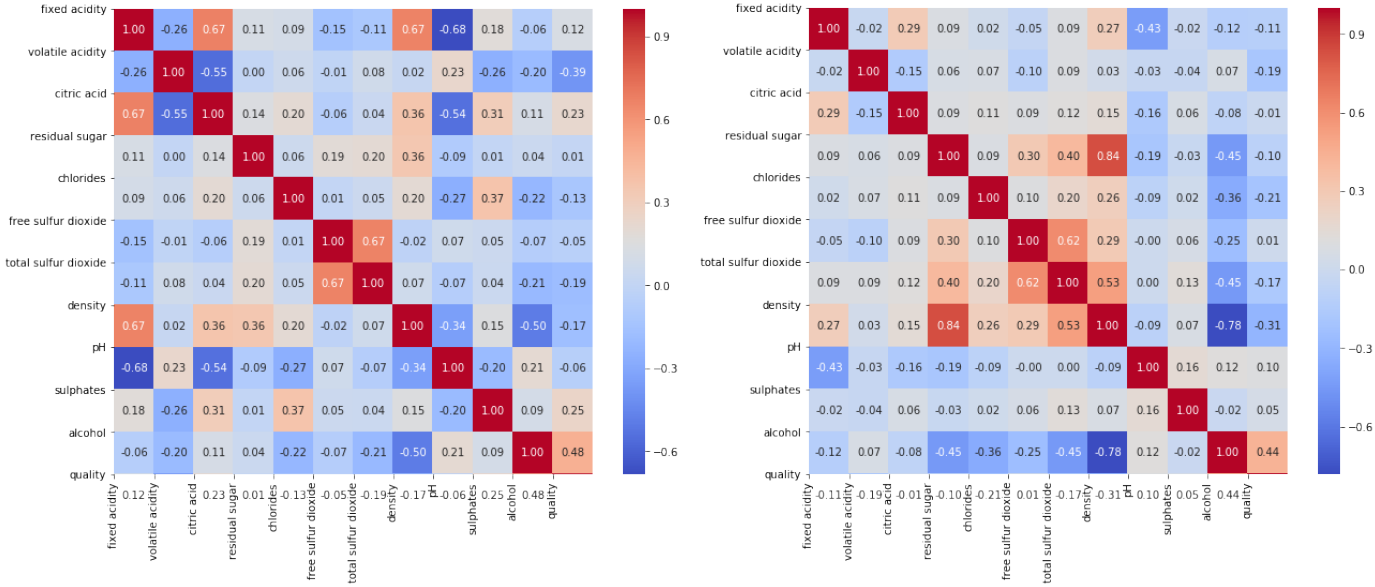


Figure 2: Correlation Heat Maps

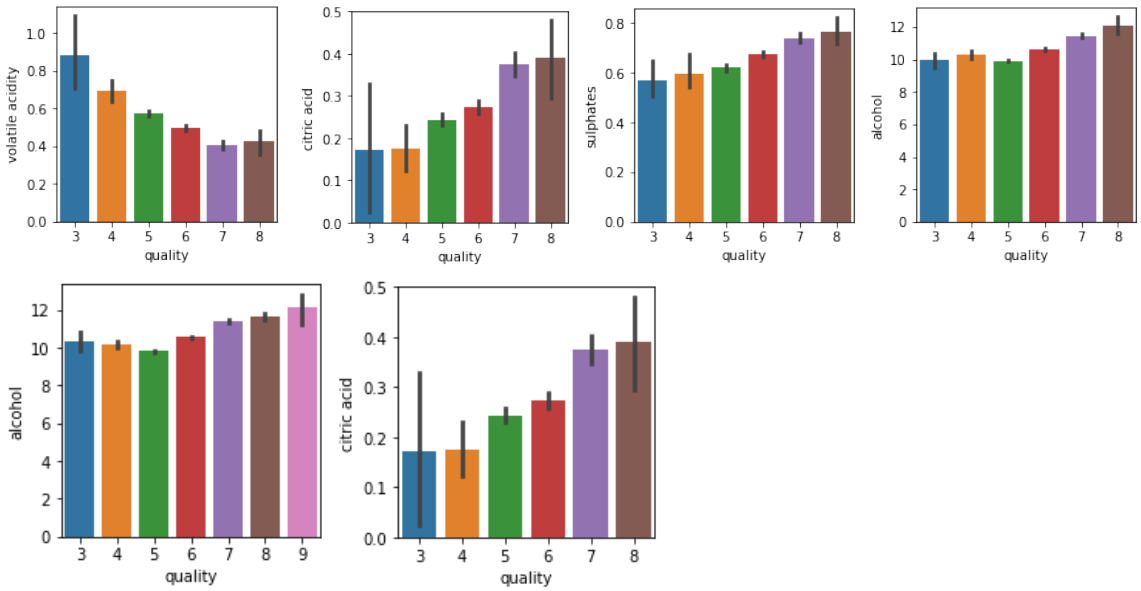


Figure 3:Red and white quality vs selected features

[10] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3472855/
[11] https://www.extension.iastate.edu/wine/total-sulfur-dioxide-why-it-matters-too
[12] https://www.etslabs.com/analyses/DEN
[13] https://www.winespectator.com/articles/what-do-ph-and-ta-numbers-mean-to-a-wine-5035
[14] https://www.winespectator.com/articles/difference-between-sulfites-sulfates-wine-54706

Table 6: Feature Importance

	AUC		MI	
Feature	**Red**	**White**	**Red**	**White**
fixed acidity	0.604954	0.542719	0.061719	0.068821
volatile acidity	0.745092	0.546597	0.126417	0.095679
citric acid	0.678216	0.500781	0.054296	0.095796
residual sugar	0.550760	0.562697	0.047283	0.070919
chlorides	0.632258	0.690441	0.046016	0.054775
free sulfur dioxide	0.575003	0.508142	0.025598	0.080013
total sulfur dioxide	0.644923	0.617750	0.112598	0.081161
density	0.627400	0.713980	0.079949	0.072746
pH	0.556085	0.566596	0.039620	0.079188
sulphates	0.738833	0.515299	0.141406	0.082562
alcohol	0.822491	0.752305	0.2650	0.218342

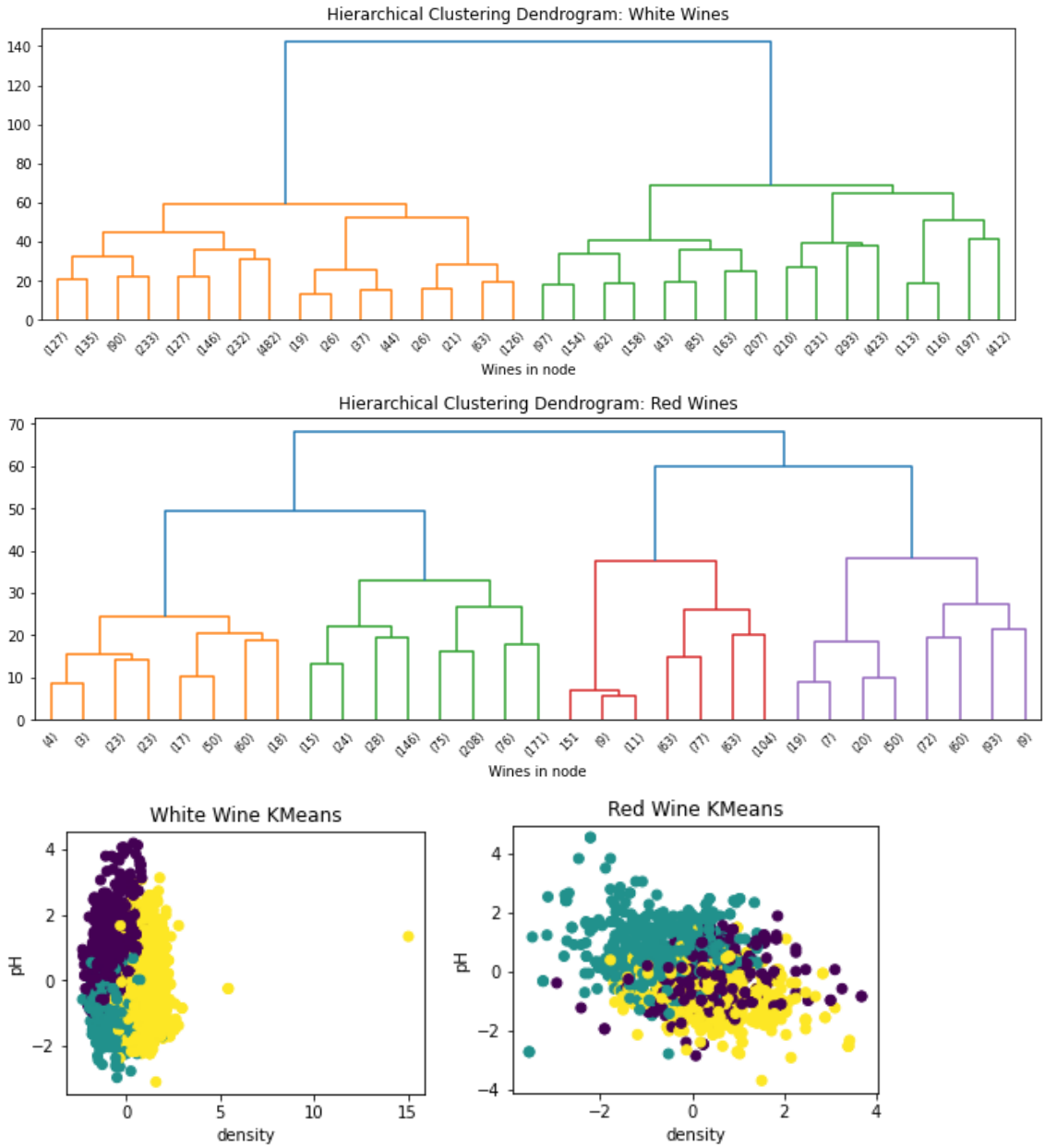


Figure 4: Clustering

Table 7: VIF (red, white)

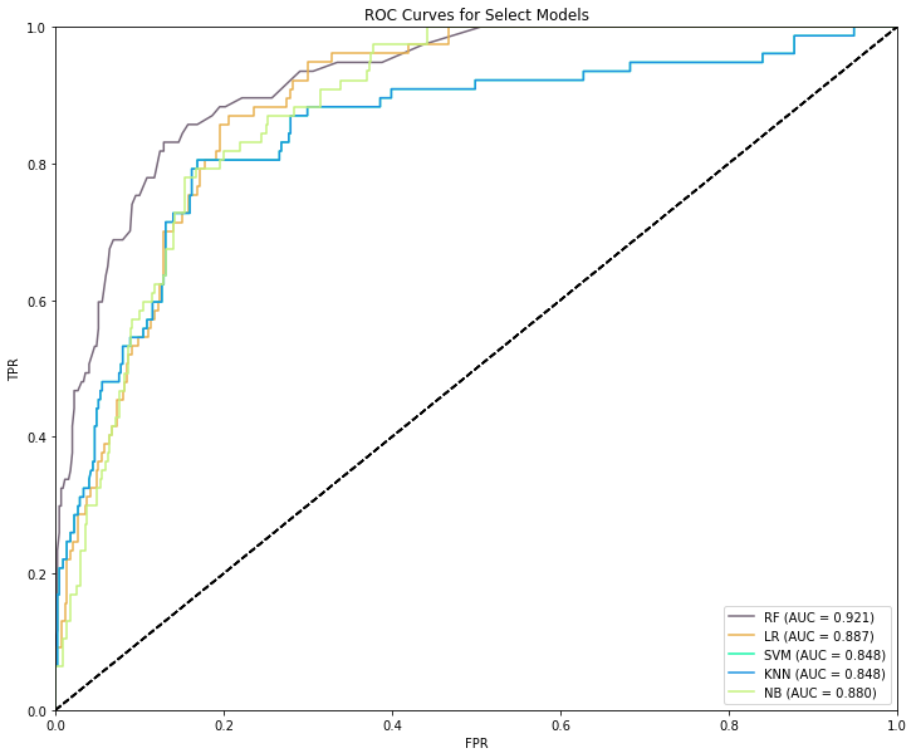
Feature	VIF
Fixed Acidity	7.814194
Volatile Acidity	1.866593
Citric Acid	3.084088
Residual Sugar	2.002017
Chlorides	1.448900
Free Sulfur Dioxide	1.969745
Total Sulfur Dioxide	2.240458
Density	7.528407
pH	3.324713
Sulphates	1.393500
Alcohol	3.474619

Feature	VIF
Volatile Acidity	1.866445
Citric Acid	2.834463
Residual Sugar	1.598921
Chlorides	1.379659
Free Sulfur Dioxide	1.961269
Total Sulfur Dioxide	2.170959
Density	2.747043
pH	1.621783
Sulphates	1.340303
Alcohol	2.316157

Feature	VIF
Fixed Acidity	1.828567
Volatile Acidity	1.156441
Citric Acid	1.153274
Residual Sugar	4.542915
Chlorides	1.393057
Free Sulfur Dioxide	1.736008
Total Sulfur Dioxide	2.199714
Density	10.357201
pH	1.586563
Sulphates	1.098830
Alcohol	4.587571

Feature	VIF
Fixed Acidity	1.348393
Volatile Acidity	1.155812
Citric Acid	1.143687
Residual Sugar	1.401792
Chlorides	1.380040
Free Sulfur Dioxide	1.718938
Total Sulfur Dioxide	2.178811
pH	1.325385
Sulphates	1.068640
Alcohol	1.749781

Modeling and Evaluation:



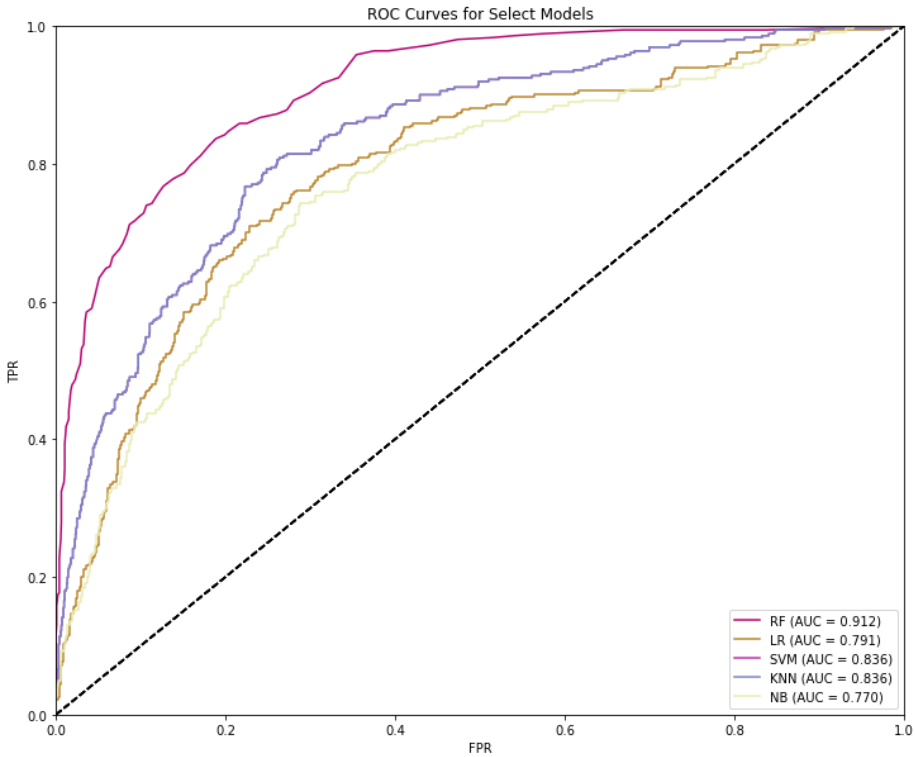


Figure 5: Baseline model ROC curves (red, white)

Table 8: Grid Search Results

Algorithm	Hyperparameter	Red Optimal Set	White Optimal Set
RF	Criterion	Gini	Entropy
	Max_Depth	50	None
	Min_Samples_Split	2	2
	Min_Samples_Leaf	1	1
	Class_Weight	Balanced	Balanced
LR	C	0.1	1
	Penalty	L2	L2
	Solver	Liblinear	Liblinear
GNB	Var_Smoothing	46.416	0.000534
SVM	C	1	1
	Gamma	Scale	Auto
	Kernel	RBF	RBF
K-NN	Algorithm	Auto	Auto
	Leaf_Size	1	1
	N_Jobs	-1	-1
	N_Neighbors	9	9
	Weights	Distance	Distance

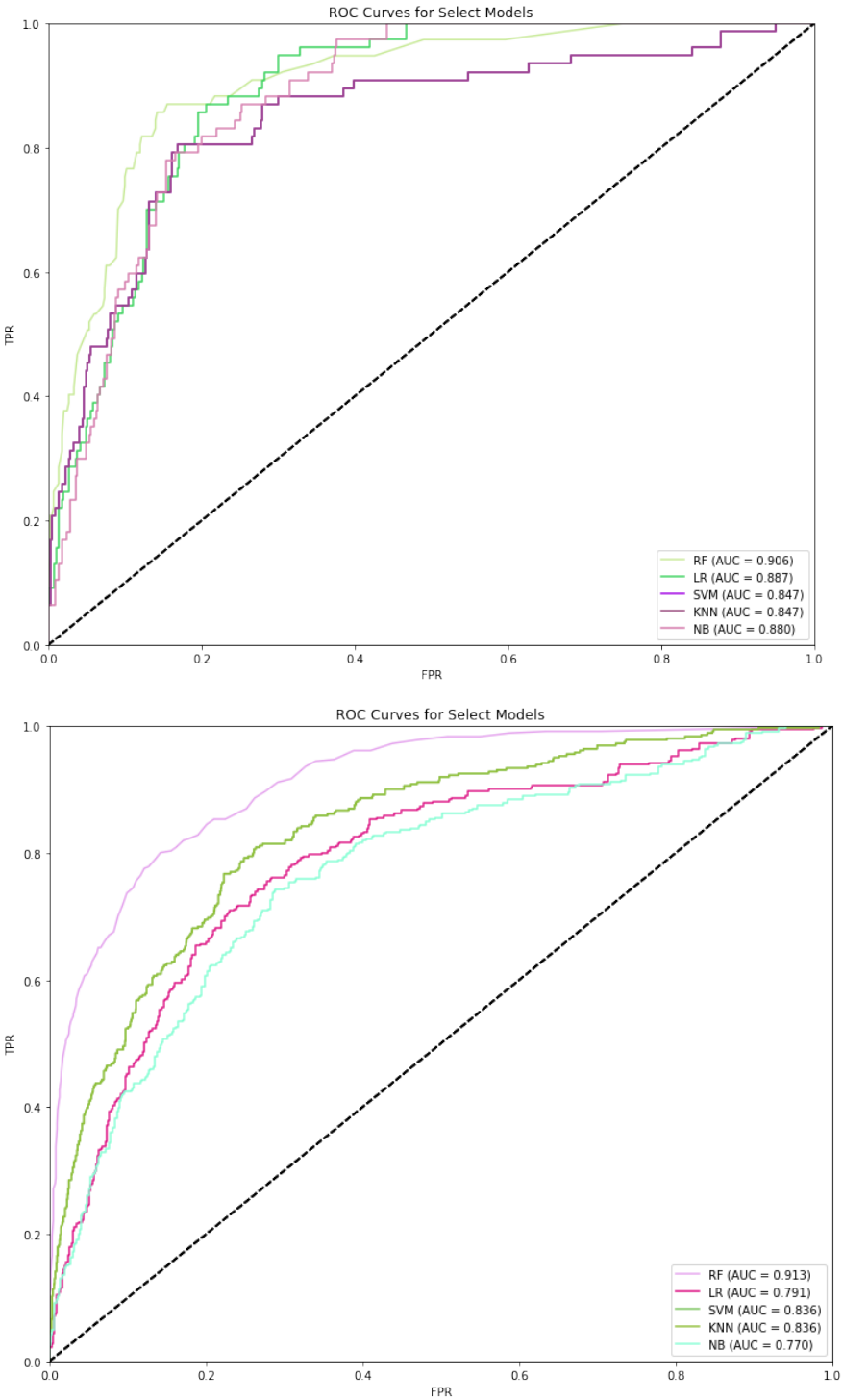


Figure 6: Tuned model ROC curves (red, white)

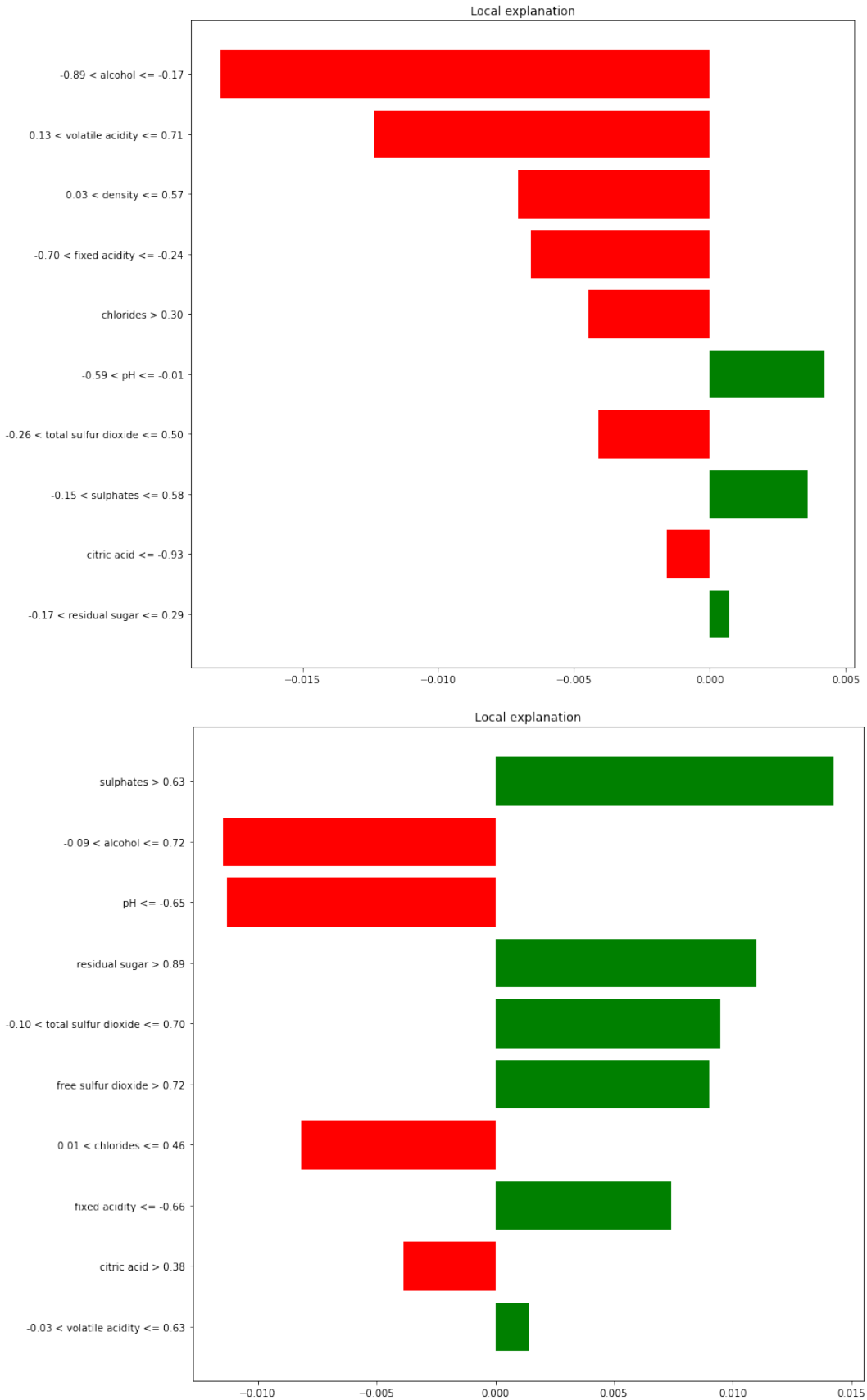
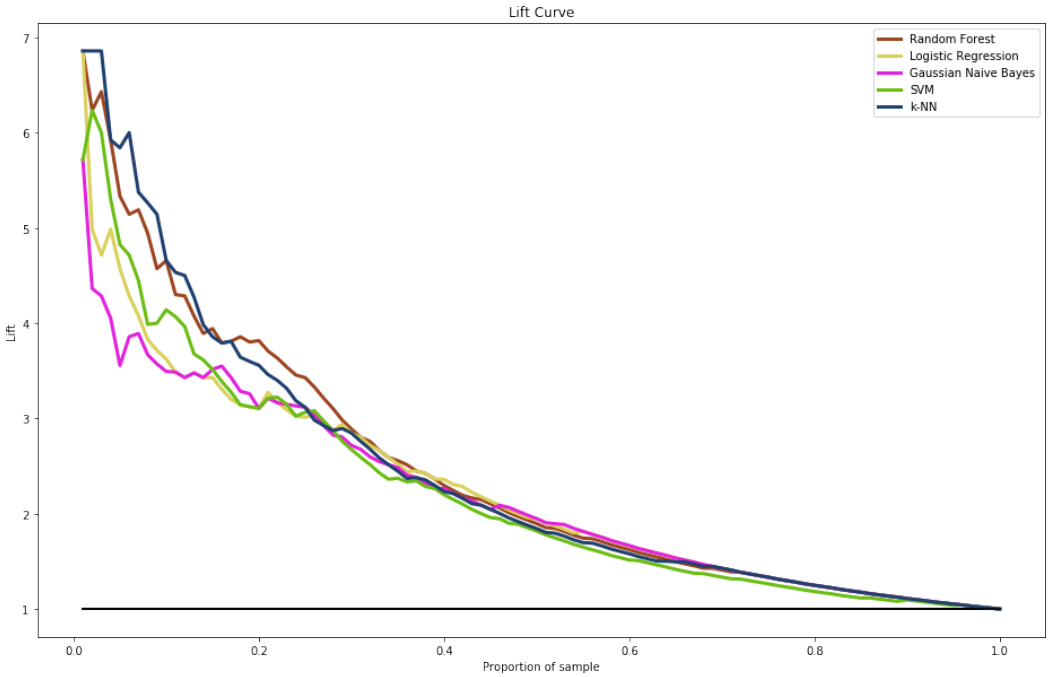


Figure 7: LIME models (red, white)



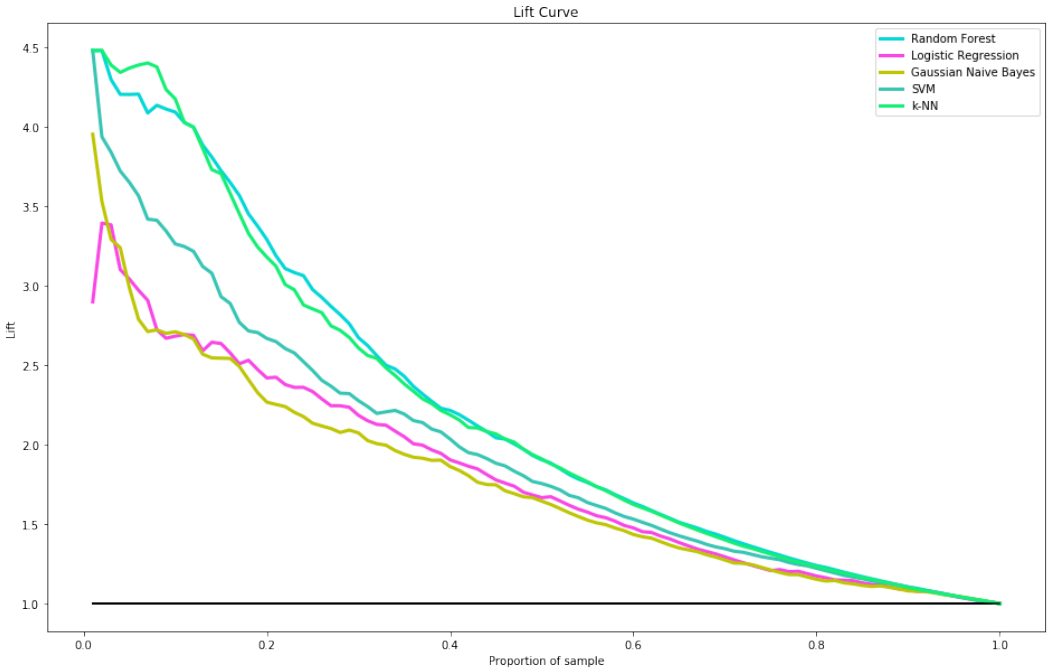


Figure 8: Lift curves (red, white)