Predicting Wine Quality Using Linear Models

Introduction

Winemaking is a very complex process that requires consideration of both physical and chemical factors in order to produce wines of the highest quality. So, it is of interest to winemakers to try to optimize their wines to be enjoyed by the greatest number of people. In other words, winemakers want to create wines that are highly rated by most. This is beneficial not only for winemakers, but for consumers as well. It is useful, then, for winemakers to try to know how their wines will be rated, based on some of the measurable characteristics of the wine. This is the motivation behind the analysis we will conduct; we want to try to answer the following question: "How well can wine quality be predicted based on the various physiochemical properties of the wine?" If wine quality can be predicted well using the created model, then winemakers might be able to "score" their own wine without having to rely on professional tasters, saving them time and money.

The Data

The dataset consists of 6,497 observations of *vinho verde* (a type of Portuguese wine) samples of red and white wine and 13 columns, as shown in Table 1. 4,898 of the observations are for white wine; the rest correspond to red wine. As mentioned previously, it will be used to try to predict the quality rating for the wine (`quality`), given some (or possibly all) of the other variables in the dataset.

Table 1: Description of the dataset

Variable	Description	Type
fixed.acidity	the amount of tartaric acid in the wine, in g/dm^3	continuous
volatile.acidity	the amount of acetic acid in the wine, in g/dm^3	continuous
citric.acid	the amount of citric acid in the wine, in g/dm^3	continuous
residual.sugar	the amount of sugar left in the wine after fermentation finishes, in g/dm^3	continuous
chlorides	the amount of salt in the wine, in g/dm^3	continuous
free.sulfur.dioxide	the amount of free sulfur dioxide in the wine, in g/dm^3	continuous
tot.sulfur.dioxide	the amount of sulfur dioxide in the wine (in both its free and bound forms), in g/dm^3	continuous
density	the density of the wine, in g/cm^3	continuous
pH	the pH of the wine on a 0–14 scale	continuous
sulphates	the amount of sulphates added, in g/dm^3	continuous
alcohol	the percent alcohol content of the wine	continuous
type	the type of the wine (1 = white, 0 = red)	categorical
quality	the quality of the wine, scored on a 0–10 scale	categorical

Figure 1 below shows the distribution of `quality`, as well as a correlation plot between the variables. Over 99.9% of the observations have a `quality` score bewteen 3 and 8; only five wine samples have a score of 9. There were no samples that received scores of 0, 1, 2, or 10. The correlation plot shows that there are variables that seem to be correlated with `quality`, and it also shows that there are some variables that are correlated with each other as well.

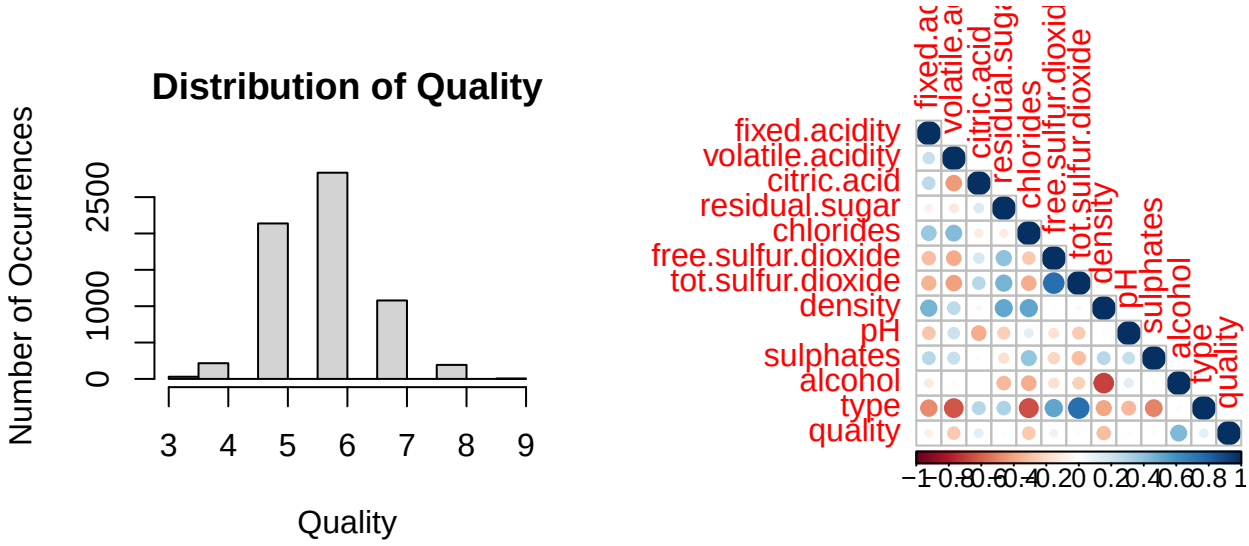

Figure 1: Unimodal distribution of wine quality (left); Correlation plot (right)

The Model

We will briefly describe our modeling method here; it is explained in greater detail in the "**Additional Work**" section on page 4. We first used the variable selection criteria BIC and C_p to create two models of only main effects. We also used the shrinkage methods of LASSO and ridge regression two obtain two more models. Then, we compared the 10-fold cross-validation MSEs (CV-MSEs) of the models to obtain the best one. The model selected by C_p performed the best, so we then considered all the possible interactions between the main effects included in the C_p model, performing forward and backward selection using AIC and BIC to obtain the best model with interactions that also followed the Principle of Marginality. The results are summarized in the table below. Note that forward selection using both AIC and BIC selected the same model, so we only show it once in the table as `fwd_aic`. For comparison, although it was not the focus of the second stage of our model selection process, we also include in the table the CV-MSEs for the LASSO and ridge models fit on the training set using all possible interactions. Surprisingly, even though the LASSO model is essentially unconstrained in the variables it can choose, it is actually the worst-performing model.

Table 2: 10-fold cross-validation scores for candidate models

Model	10-Fold CV-MSE
`fwd_aic`	0.5325936
`bwd_aic`	0.4795698
`bwd_bic`	0.4812950
LASSO	0.5330602
Ridge	0.5080792

Out of the models in the first three rows, we selected the model with the lowest CV-MSE, which is `bwd_aic`. It contains 56 variables, including the intercept. All the main effects are included, and all the main effects are interacted with each other at least once. Despite all these terms, the adjusted R^2 value is 0.3742, which is not very high. Because we used automated model selection methods to obtain our model, we will not be able to perform inference reliably using the calculated standard errors. If we wanted to, we could use the bootstrap to obtain standard errors for the coefficients, but since our main goal is prediction, we will not be performing inference.

Because the terms were selected using model selection processes, terms that have been omitted from the model are not necessarily useless for prediction. Similarly, terms that have been included are not necessarily "correct." That is, they may help to improve the predictive accuracy of the model, but they may not actually be a part of the population regression model. In theory, it is still possible to interpret the coefficients of the model, but the interpretations are not very practical. For example, the coefficient on `type` is about 133, which implies that (ignoring interactions), if all other variables are held constant, white wine scores higher than red wine by 133 points on average. Of course, this makes no sense, since `quality` is on a scale from 0 to 10, and the coefficients of the other variables in the model do offset the effect of the `type` coefficient, but this shows that our model is only fit for prediction, not for interpretation. The full details of the model can be found in **Appendix A**.

Model Discussion and Conclusions

Despite our best efforts with finding interaction terms and testing different models, the adjusted R^2 of our chosen model fit on the entire dataset is quite poor at 0.3742 (see the "**Additional Work**" section). So, our model does not fit the data well, meaning that a purely linear model approach is insufficient to predict wine ratings. This is not that surprising; the biggest limitation of our model is that it assumes the response variable `quality` is continuous, whereas `quality` is actually discrete. This also means that the common assumption of normality when performing least-squares regression does not hold with our data. However, the violation of normality is not a problem for us because the purpose of our analysis is to create the best predictive model, not to do any inference with coefficient estimates. Because of this, we believe that a multiclass logistic regression model, such as a proportional odds model, would be more suited to predicting wine quality.

We mentioned previously that our model contains 56 terms, including the intercept. Though a complex model does also introduce the possibility of overfitting, we are not too worried about this. We used cross-validation to determine the best model, and moreover, the MSE of the model applied to the test set is not too different from the MSE of the model built using the training set, so we conclude that our model generalizes well to unseen data (again, see the "**Additional Work**" section).

The large number of variables and interactions in our final model shows that wine quality is a very difficult thing to pin down. Even if not all the variables "matter," the fact that there are still 56 variables demonstrates the complexity of wine quality. This is also not surprising, since quality ratings are assigned by humans, not machines, and variables like citric acid or the amount of sulphates are likely not known to humans as they are tasting the wine. The conclusion is that wine is greater than the sum of its parts. That is, the quality of a wine cannot simply be quantified by the chemical makeup of the wine, which leads to our next topic: the usefulness of our model for winemakers.

Main Findings for Winemakers

With the final model, winemakers will be able to get a general sense of the quality of their wine, but it will not be as useful for providing an exact prediction. In other words, the model is not a replacement for a human wine taster, which does make sense because "quality" is inherently a subjective measurement made by humans. As will be shown later, the mean squared prediction error of our chosen model on the test data is about 0.44, so our model does reasonably well at prediction, even though the adjusted R^2 value is not very high. This is why we claim that the model can still be used to get a general idea of how a sample of wine will score. So, we can say that the model is accurate to a certain degree. We would also dissuade winemakers from interpreting the coefficients of the model too literally, as mentioned before. In reality, for example, the difference between `quality` for red wine and white wine is probably not very high, even though the coefficient of `type` is 133. All this means is that winemakers should only use the model for rough predictions and not for causal inference or to make any associations.

Additional Work

In this section, we give more details about our model selection procedure and check model diagnostics to assess the validity of the linear regression assumptions. We also consider other models using logistic regression that may be a better fit for the data, and explain why that is the case.

Procedure

Before we began, we first looked at the distribution of the explanatory variables in our dataset. Many of the variables had skewed distributions, so we applied the natural logarithm and square root transformations to make the distributions more symmetric. Some examples of this are shown in Figure 2 below. A more symmetric distribution for the explanatory variables could help ensure that the linear regression assumptions of linearity and homoscedasticity hold, which are the key assumptions that will allow us to make predictions with the model. We did observe some correlations between the explanatory variables, but since we will not be doing inference, we are not worried about multicollinearity.

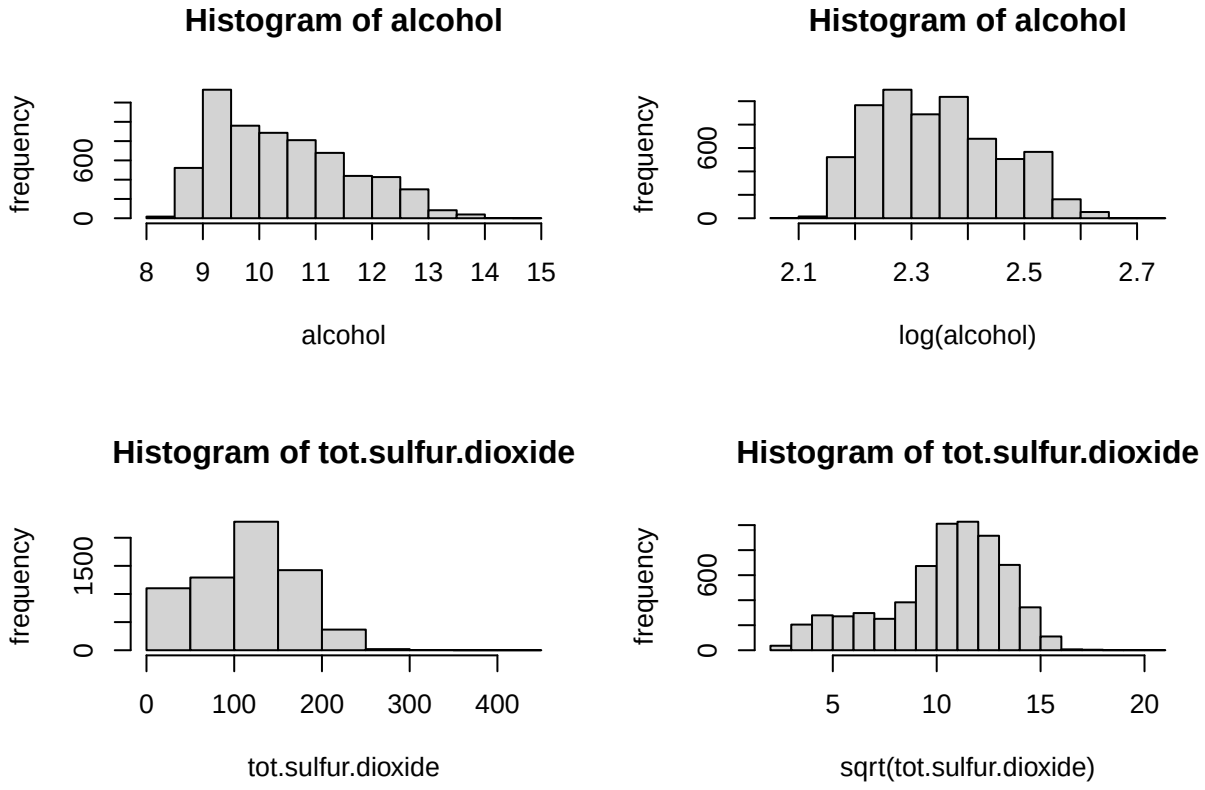

Figure 2: Distribution of `alcohol` (top left); distribution of `log(alcohol)` (top right); distribution of `tot.sulfur.dioxide` (bottom left); distribution of `sqrt(tot.sulfur.dioxide)` (bottom right).

We had many possible explanatory variables to choose from, so we decided to use model selection techniques, both with and without regularization, to determine our final model. Because our model was to be used for prediction, we did a 90/10 train/test split, so that we would have "new" data to test our model on. We did a 90/10 split because we had a lot of data in our dataset, so we could commit more data for use in training and still have a good sample with which to test our model. The correlation plot shown in Figure 1 from before showed the relationships between the main effects and `quality`. We then did some EDA to

look for possible interactions. To given some examples, Figure 3 below (left) shows a possible interaction between `volatile.acidity` and `type`. This is seen in the fact that the slope of the line corresponding to the relationship between `volatile.acidity` and `quality` differs based on wine type. Similarly, in Figure 3 (right), we see a possible interaction between `free.sulfur.dioxide` and `type`, as the slope of the two lines are again very different.

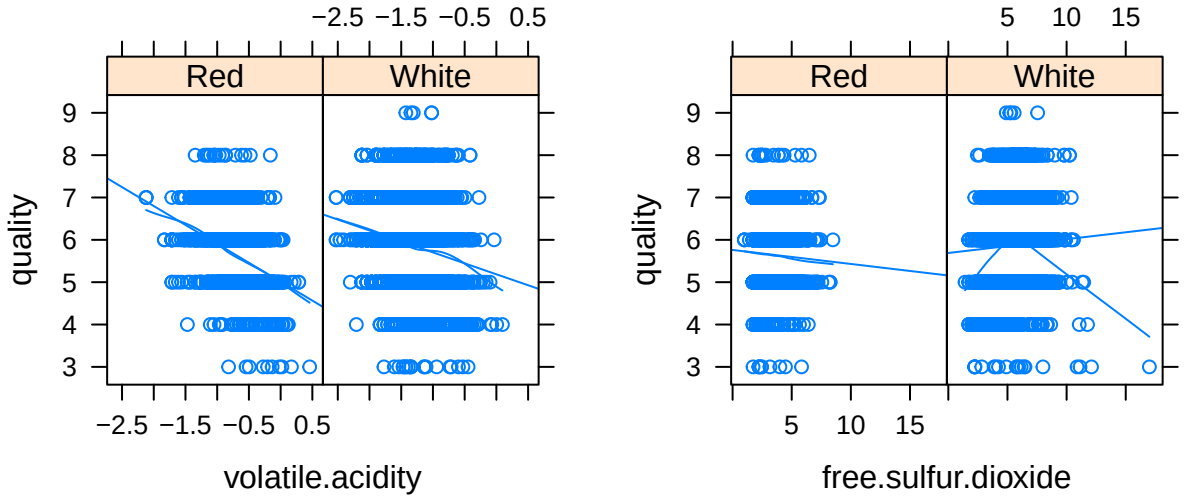

Figure 3: Examples of possible interactions

Because we noticed some possible interactions, we had to adopt a slightly different method of model selection. In general, it may not be that model selection techniques obey the Principle of Marginality. To account for this, we used a two-stage selection procedure. First, we selected a model containing only main effects. We performed an exhaustive search over the 12 main effects using BIC and Mallows's C_p as our criteria of choice, and obtained two models. BIC and C_p were chosen because they are commonly used criteria. The best model obtained using BIC removed some of the main effects, while the best model obtained using Mallows's C_p kept all of the main effects.

We also used ridge regression and the lasso to gain two more models, where the tuning parameter λ was selected using 10-fold cross-validation. We chose to consider models that use shrinkage because they often have lower variances (at the cost of more bias), which can lead to better predictions. We wanted to compare the predictive accuracy of these four models to determine which one we wanted to add interactions to. To do this, we used 10-fold cross-validation and compared the CV-MSEs, which are displayed in Table 3 below. We selected the model with the lowest CV-MSE, which was the linear model obtained using C_p.

Table 3: Cross-validation scores for candidate models containing only main effects

Method (Main Effects)	10-Fold CV-MSE
BIC	0.5347080
C_p	0.5322289
LASSO	0.5387922
Ridge	0.5339157

In the second stage of our model selection process, we considered interactions between the main effects chosen in the model selected by C_p. As the model selected by C_p contained all 12 main effects, there would be

$\binom{12}{2} - 1 = 65$ possible interaction terms, which was too many variables to run an exhaustive search on. The reason why we subtracted 1 is because the `type:type` interaction is exactly the same as the variable `type`, so we removed it from consideration beforehand. Since there were 65 possible interactions, we decided to use forward and backward selection (with AIC and BIC as our criteria), where the minimum model was the model with only main effects and the maximum model was the model with all 65 possible interactions (as well as the main effects). Carrying out the model selection process, we again obtained four models that, by construction, obeyed the Principle of Marginality. Interestingly, forward selection using both AIC and BIC did not add any interactions to the model, so the models obtained using forward selection were the same as the model of just main effects. Backward selection did remove some interactions, but many were still left in. Thus, in the end, we had only three different models, `fwd_aic`, `bwd_aic`, and `bwd_bic`.

At this point, although we had determined in the first stage that the LASSO and ridge regression models performed worse than the traditional linear regression models, we wanted to see (for comparison) how these models would perform with all the interactions available to them. So, we obtained two more models using LASSO and ridge regression that considered all the interactions. To see which of the models were best, we again conducted 10-fold cross validation and compared the CV-MSEs of the models. The model with the lowest CV-MSE ended up being the model chosen using backward selection with AIC as the criterion (`bwd_aic`), as shown in the table below. Once again, the linear regression model performed better than the models with shrinkage penalties. The CV-MSEs are shown in the table below (note that this table is the same as the one in the section "**The Model**" from before).

Table 4: Reproduction of Table 2

Method	10-Fold CV-MSE
`fwd_aic`	0.5325936
`bwd_aic`	0.4795698
`bwd_bic`	0.4812950
LASSO	0.5330602
Ridge	0.5080792

The improvement in the CV-MSE compared to the model with only main effects is about 0.05, which is not as insignificant as it seems, as the the observed values of quality lie on a 0 to 10 scale and we are considering the average of the *squared* errors (so the actual errors are higher). The model `bwd_aic` contains 56 terms, including the intercept, which is well below the 10% rule of thumb for the number of terms to include in a model, so we are not too concerned about the possibility of having an overfitted model.

We could now get a sense of the predictive accuracy of our model by considering the mean squared prediction error (MSPE) of our model using the data from the test set, which is displayed in Table 5 below. For reference, Table 5 also includes the MSPE of our model using the data in the training set.

Table 5: Mean squared prediction error on the test data

Dataset	MSPE
Train	0.4694684
Test	0.4475201

Interestingly, the MSPE on the test set is actually lower than the MSPE of the model on the training set. This is why we concluded that our model generalizes well to unseen data, which gives us further reassurance that our model does not suffer from overfitting.

Model Diagnostics and Alternatives

As with all model selection procedures, our two-stage method for model selection introduces some compromises. Recall that in the first stage, we selected only for main effects, and in the second stage, we selected for interactions. One might wonder why we needed to do this two-stage procedure if our selected model from the first stage contained all the main effects. The point is that if we tried to select main effects and interactions all in one step, some main effects could have been omitted from the model, which is why the two-stage method is necessary. But because of how we selected the variables, it could be that omitting some main effects (and thus some interactions as well) would actually improve the ability of the model to predict the data in the test set. In that sense, although we are confident in our chosen model, there could be other models that perform even better than it.

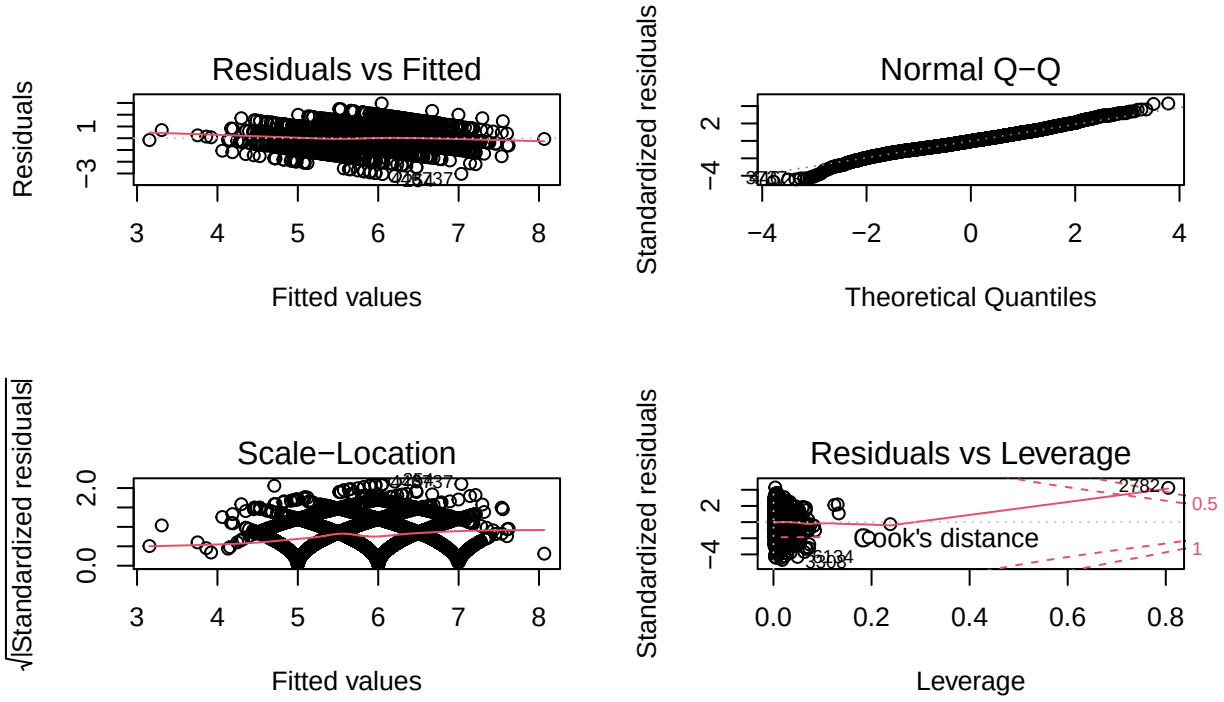

Figure 4: Diagnostic plots for our final model

Now, we consider some model diagnostics for our selected model, `bwd_aic`, checking to see if the assumptions of linear regression hold. We already know that the data cannot be normally distributed, so the normal Q-Q plot is not of much interest to us. Again, this is not a problem for us, since we only need the assumption of normality for inference, which we are not interested in doing, especially since the coefficient values themselves may not be very realistic (recall the coefficient of 133 on `type`, for example). Figure 4 above shows some diagnostic plots. Though there are some clear patterns in the residual plot, this is mostly an artifact of the discrete response variable. The scatter of points is still quite even about 0, so the OLS assumption of linearity is valid. Furthermore, the variance of the residuals does appear to be constant, so the assumption of homoscedasticity holds as well. The scale-location plot also shows some clear patterns, but again, this is simply due to the discrete response variable. From the plot of Standardized Residuals vs. Leverage, we do not see a clear relationship between leverage and the residuals. There is one point that has high leverage and a high standardized residual, data point 2782 (labeled in the plot), but upon investigation, the data point does not appear to be "incorrect" in any way, so we do not remove it from the sample. We conclude that it is just a strange wine sample. So overall, although the model has low predictive power, we see that there are no obvious issues with the linear modeling assumptions important to us.

We discussed earlier in our paper that another alternate model we could have considered is a proportional odds model. It may be the case that the different between a 1 and 2 rating, for example, is much more "extreme" than a difference between a 9 and 10 rating. The OLS model we used does not take this possibility into account, so a proportional odds model may fit better. Also, we can be sure that none of the predictions made by a proportional odds model will be outside the range of realistic values, which is a concern for our linear regression model. In our case, this concern is not too serious, since most of our data lies in the range between 3 and 8, so we do not have to worry much about predictions that fall outside of the acceptable range. Still, a model that actively takes the discrete categories into account is likely to improve prediction. Furthermore, treating the modeling problem as a classification problem, we can use the 0-1 loss function to get a sense of predictive accuracy, which may be a more accurate measurement than the raw MSPE that we used in our current model. Although the MSPE of our current model fit to the test set, 0.474, is not very high, it still introduces ambiguity into our predictions. For example, if the model predicts a quality rating of 5.5, it is not clear whether that should actually be a 5 or a 6. Thus, our next step should be to pursue a logistic regression model in the hopes of increasing our predictive accuracy even further. Our model is already reasonably accurate in terms of prediction, so a logistic regression model could do even better. Such a model may be more useful to winemakers as a way for them to rate their own wines without needing to hire a professional to do so for them.

References

The dataset is from the paper "Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009" by P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.

The dataset was found on the UC Irvine Machine Learning repository at the following link: https://archive.ics.uci.edu/ml/datasets/wine+quality.

Appendix A

The final model, fit on the entire dataset.

```
##
## Call:
## lm(formula = formula(bwd_aic), data = wine)
##
## Coefficients:
##                              (Intercept)                            fixed.acidity
##                                -1.229e+03                               -8.627e+00
##                           volatile.acidity                              citric.acid
##                                -1.478e+01                                4.252e+01
##                             residual.sugar                                chlorides
##                                -6.265e+01                               -6.027e+01
##                        free.sulfur.dioxide                     tot.sulfur.dioxide
##                                -7.853e-01                                4.858e-01
##                                   density                                       pH
##                                 1.256e+03                                1.799e+02
##                                 sulphates                                  alcohol
##                                -6.242e+00                                2.305e+02
##                                      type                  fixed.acidity:chlorides
##                                 1.333e+02                               -8.379e-01
##      fixed.acidity:free.sulfur.dioxide       fixed.acidity:tot.sulfur.dioxide
##                                 1.299e-01                               -5.448e-02
##                           fixed.acidity:pH                 fixed.acidity:sulphates
##                                 2.146e+00                                8.909e-01
##                          fixed.acidity:type           volatile.acidity:citric.acid
##                                 5.569e-01                                5.601e-01
##   volatile.acidity:free.sulfur.dioxide                volatile.acidity:density
##                                 5.788e-02                                6.243e+00
##                        volatile.acidity:pH              volatile.acidity:sulphates
##                                 6.825e-01                               -1.949e-01
##                volatile.acidity:alcohol                    volatile.acidity:type
##                                 2.240e+00                               -1.783e-01
##                   citric.acid:chlorides         citric.acid:free.sulfur.dioxide
##                                 8.740e-02                                8.139e-02
##         citric.acid:tot.sulfur.dioxide                     citric.acid:density
##                                -4.805e-02                               -4.762e+01
##                   citric.acid:sulphates                     citric.acid:alcohol
##                                -6.136e-01                                2.136e+00
##                         citric.acid:type                residual.sugar:chlorides
##                                 5.031e-01                               -2.114e-01
##      residual.sugar:tot.sulfur.dioxide                  residual.sugar:density
##                                 1.075e-02                                5.970e+01
##                residual.sugar:sulphates                  residual.sugar:alcohol
##                                -1.009e-01                                1.103e+00
##                     residual.sugar:type                       chlorides:density
##                                 2.529e-01                                6.464e+01
##                            chlorides:pH                    chlorides:sulphates
##                                 -7.782e-01                               -2.919e-01
## free.sulfur.dioxide:tot.sulfur.dioxide                  free.sulfur.dioxide:pH
##                                -3.482e-02                                7.537e-02
##          free.sulfur.dioxide:sulphates             free.sulfur.dioxide:alcohol
```

```
##                          1.464e-01                          2.626e-01
##        free.sulfur.dioxide:type  tot.sulfur.dioxide:sulphates
##                          3.012e-01                         -1.513e-01
##  tot.sulfur.dioxide:alcohol       tot.sulfur.dioxide:type
##                         -1.467e-01                          4.046e-02
##                       density:pH                density:alcohol
##                         -1.867e+02                         -2.290e+02
##                     density:type                    pH:sulphates
##                         -1.399e+02                          1.574e+00
##                            pH:type                    alcohol:type
##                          1.593e+00                         -1.445e+00
```

Appendix B

```
# loading necessary packages
library(leaps)
library(glmnet)
library(caret)
library(ggplot2)
library(GGally)
library(corrgram)
library(corrplot)
library(lattice)
library(gridExtra)
# loading data
setwd("C:/Users/pche/Desktop/Stat_151A/")
wine <- read.csv("winequality.csv", header = TRUE)
# cleaning up the names of variables
wine[12:13] <- wine[13:12]
names(wine)[7] <- "tot.sulfur.dioxide"
names(wine)[12] <- "type"
names(wine)[13] <- "quality"
names(wine)[7] <- "tot.sulfur.dioxide"
# applying transformations
wine$alcohol <- log(wine$alcohol)
wine$sulphates <- log(wine$sulphates)
wine$chlorides <- log(wine$chlorides)
wine$residual.sugar <- log(wine$residual.sugar)
wine$volatile.acidity <- log(wine$volatile.acidity)
wine$fixed.acidity <- log(wine$fixed.acidity)
wine$free.sulfur.dioxide <- sqrt(wine$free.sulfur.dioxide)
wine$tot.sulfur.dioxide <- sqrt(wine$tot.sulfur.dioxide)
wine$citric.acid <- sqrt(wine$citric.acid)
```

The Data

```
par(mfrow = c(1, 2))
# histogram of response variable quality
hist(wine$quality, main = "Distribution of Quality",
     ylab = "Number of Occurrences",
     xlab = "Quality")
# correlation plot between variables
corrplot(cor(wine[, 1:13]), type = "lower")
```

```
# 90/10 train/test split
set.seed(1)
split <- sample(1:nrow(wine), floor(9 * nrow(wine) / 10))
train <- wine[split, ]
test <- wine[-split, ]

# C_p and BIC
# exhaustive search over all possible models
selection <- regsubsets(quality ~ .,
```

```
                        data = train,
                        method = "exhaustive",
                        nvmax = 12,
                        nbest = 1)
# by BIC
coef(selection, which.min(summary(selection)$bic))
bic_main <- lm(quality ~ volatile.acidity + residual.sugar + chlorides +
                 free.sulfur.dioxide + tot.sulfur.dioxide + density +
                 sulphates + alcohol + type, data = train)
# by C_p
coef(selection, which.min(summary(selection)$cp))
cp_main <- lm(quality ~ fixed.acidity + volatile.acidity + residual.sugar +
                 citric.acid + residual.sugar + chlorides +
                 free.sulfur.dioxide + tot.sulfur.dioxide + density + pH +
                 sulphates + alcohol + type, data = train)


# Ridge and Lasso
# creating model matrix
x <- model.matrix(quality ~ ., data = train)[, -1]
y <- train$quality
set.seed(1)
# grid of lambda values
lambda_grid <- 10^seq(10, -2, length = 200)

# Lasso
cv_lasso <- cv.glmnet(x, y, alpha = 1, nfolds = 10, lambda = lambda_grid)
# the MSE is
min(cv_lasso$cvm)

# Ridge
cv_ridge <- cv.glmnet(x, y, alpha = 0, nfolds = 10, lambda = lambda_grid)
# the MSE is
min(cv_ridge$cvm)

# 10-fold CV-MSE for BIC model
data_ctrl <- trainControl(method = "cv", number = 10)
bic_cv <- train(formula(bic_main), data = train,
                trControl = data_ctrl,
                method = "lm")
# the MSE is the square of the RMSE (root MSE), so we square it
bic_mse <- (bic_cv$results[2]^2)[1, ]

# 10-fold CV-MSE for C_p model
data_ctrl <- trainControl(method = "cv", number = 10)
cp_cv <- train(formula(cp_main), data = train,
                trControl = data_ctrl,
                method = "lm")
# the MSE is the square of the RMSE (root MSE), so we square it
cp_mse <- (cp_cv$results[2]^2)[1, ]
```

Additional Work

Procedure

```
# data transformation histograms
par(mfrow = c(2, 2))
# histogram of alcohol
hist(exp(wine$alcohol),
     xlab = "alcohol",
     ylab = "frequency",
     main = "Histogram of alcohol")
# histogram of log(alcohol)
hist(wine$alcohol,
     xlab = "log(alcohol)",
     ylab = "frequency",
     main = "Histogram of alcohol")
# histogram of tot.sulfur.dioxide
hist(wine$tot.sulfur.dioxide^2,
     xlab = "tot.sulfur.dioxide",
     ylab = "frequency",
     main = "Histogram of tot.sulfur.dioxide")
# histogram of sqrt(tot.sulfur.dioxide)
hist(wine$tot.sulfur.dioxide,
     xlab = "sqrt(tot.sulfur.dioxide)",
     ylab = "frequency",
     main = "Histogram of tot.sulfur.dioxide")
```

```
# setting up interaction plots
labels = wine$type
labels[labels == 1] = "White"
labels[labels == 0] = "Red"

# interaction between volatile.acidity and type
p1 <- xyplot(quality ~ volatile.acidity|labels, data = wine, pch=16, cex=.3,
             panel = function(x, y) {
              panel.xyplot(x, y, type = c("p", "smooth", "r"))
              })

# interaction between free.sulfur.dioxide and type
p2 <- xyplot(quality ~ free.sulfur.dioxide|labels, data = wine, pch=16, cex=.3,
             panel = function(x, y) {
              panel.xyplot(x, y, type = c("p", "smooth", "r"))
              })

grid.arrange(p1,p2, ncol = 2)
```

```
# formula for model of main effects selected by BIC
formula(bic_main)
# formula for model of main effects selected by C_p
formula(cp_main)
```

```r
# first we make a data frame with only the variables selected by BIC.
# of course, we include the quality column as well

# smallest model
min_model <- cp_main
# biggest model
big_model <- lm(quality ~ .^2 - type:type + type, data = train)
# forward selection using AIC
fwd_aic <- step(min_model, direction = 'forward', scope = big_model, trace = 0, k = 2)

# backward selection using AIC
bwd_aic <- step(big_model, direction = 'backward', scope = min_model, trace = 0, k = 2)

# forward selection using BIC
fwd_bic <- step(min_model, direction = 'forward', scope = big_model, trace = 0,
                k = log(nrow(train)))

# backward selection using BIC
bwd_bic <- step(big_model, direction = 'backward', scope = min_model, trace = 0,
                k = log(nrow(train)))

# computing the CV-MSEs
set.seed(345)
# first we do fwd_aic
fwd_aic_cv <- train(formula(fwd_aic), data = train,
                trControl = data_ctrl,
                method = "lm")
# the MSE is the square of the RMSE (root MSE), so we square it
fwd_aic_mse <- (fwd_aic_cv$results[2]^2)[1, ]

# next is bwd_aic
bwd_aic_cv <- train(formula(bwd_aic), data = train,
                trControl = data_ctrl,
                method = "lm")
# the MSE is the square of the RMSE (root MSE), so we square it
bwd_aic_mse <- (bwd_aic_cv$results[2]^2)[1, ]

# finally, we do bwd_bic
bwd_bic_cv <- train(formula(bwd_bic), data = train,
                trControl = data_ctrl,
                method = "lm")
# the MSE is the square of the RMSE (root MSE), so we square it
bwd_bic_mse <- (bwd_bic_cv$results[2]^2)[1, ]


# LASSO and ridge model selection with all possible interactions, for comparison
set.seed(345)
x_2 <- model.matrix(quality ~ .^2 - type:type + type, data = train)[, -1]
# LASSO
cv_lasso_2 <- cv.glmnet(x_2, y, alpha = 1, nfolds = 10, lambda = lambda_grid)
# CV-MSE
min(cv_lasso_2$cvm)
```

```r
# Ridge
cv_ridge_2 <- cv.glmnet(x_2, y, alpha = 0, nfolds = 10, lambda = lambda_grid)
# CV-MSE
min(cv_ridge_2$cvm)
```

```r
# test MSPE
mean((test$quality - predict(lm(formula(bwd_aic), data = test), test))^2)
# train MSPE
mean((train$quality - predict(lm(formula(bwd_aic), data = train), train))^2)
```

```r
# diagnostic plots
par(mfrow = c(2,2))
plot(lm(formula(bwd_aic), data = wine))
```