Understanding the Quality of Wine Revealed In Chemistry

Alexandra Neff

2025-08-28

Abstract

Using a data set on wine quality based on various physical and chemical attributes, we fit an ordinal logit model to the response variable indicating wine quality as perceived by a panel of tasters; the ordinal logit model is the most appropriate. In spite of issues stemming from sparseness in the levels of the `quality` response variable, we found a model with attributes selected stepwise, achieving an prediction error rate as low as ~6% in testing on the subsetted Training Data; and we have generated predicitions for the Test Data set to be submitted for evaluation by the client, as well as created a dashboard allowing interactive customization of wine profiles to be predicted for quality by the model.

Context

Wine is a beverage rooted very deeply in human society, particularly in regions where grapes flourished and the practice of fermenting foodstuffs was popular. The Mediterranean has long included wine in everyday meals, in social events, and even in religious ritual, especially where Judaism and Christianity have become dominant faiths. This made wine an essential commodity in Europe, and as European cultures conquered and colonized most other nations on Earth, it spread from there.

As a commodity involved in cuisine and social gatherings, high-quality wines inevitably became associated with status, and the market factored the subjective quality of wines into their prices, cementing it as a sliding scale of wealth and status. The most delectable vintages of the most reliable grape cultivars became a part of opulence—ergo, they communicate a person's wealth in being possessed and consumed by them in the company of others.

As science and analysis of objective facts became popular, the quality of wine became a topic of study, with vintners hoping to produce better wines with clever adjustments to the growth and processing of the grapes, driven by scientific study, to make more efficient use of the costs put into the wine-making process. Success in these regards could lead to increased profits and greater market share.

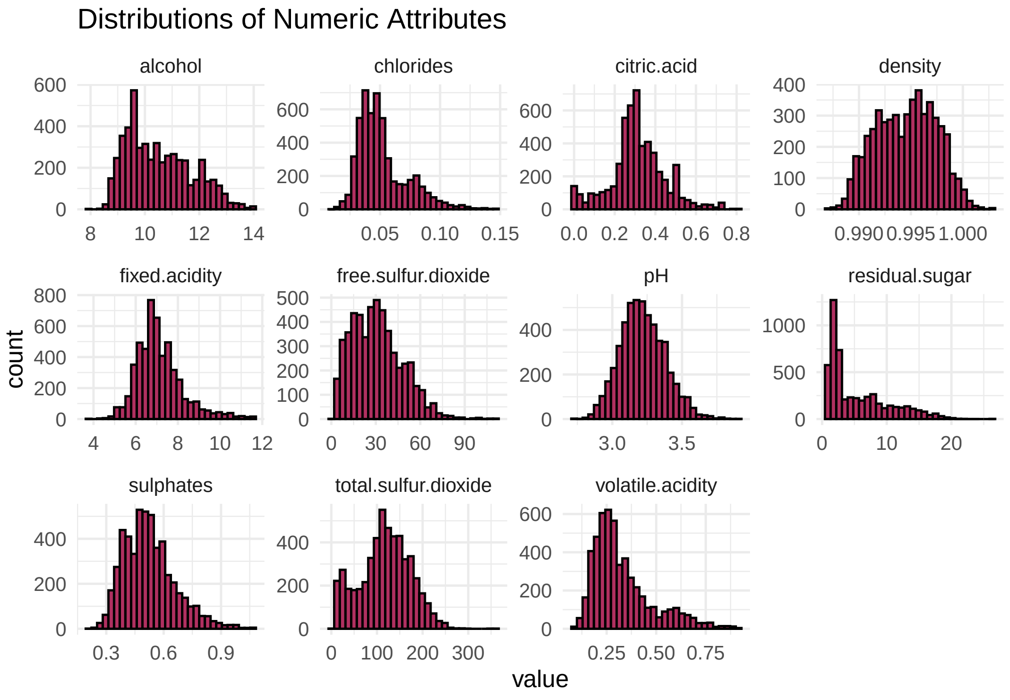
In this study, we explore just this approach to the quality of wine, analyzing how multiple objective attributes of a wine affect its rating by the objective but practiced tasting by sommelieres.

Analysis: Data Preparation and EDA

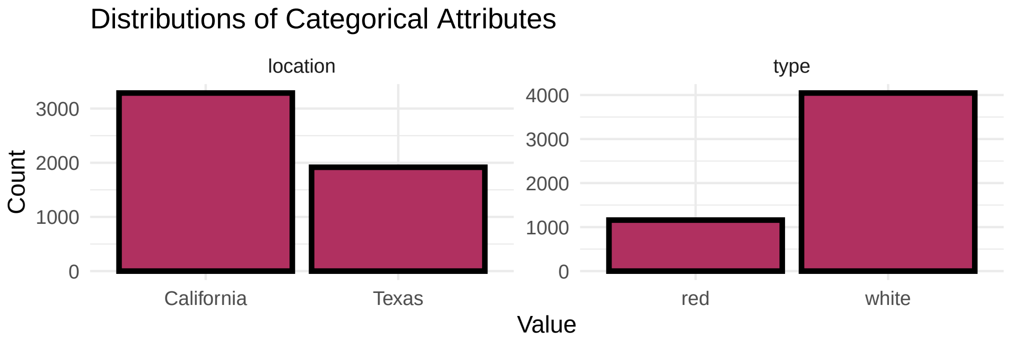
The training data file includes fifteen total variables for 5,463 records in Comma-Separated Values (CSV) format. The testing dataset contains 1,034 records of all of the same except for quality values. The target is the attribute included in the training data, `quality`; we aim to provide our evaluator with quality values predicted by our most accurate model for each of the records of the test dataset. Eleven of the other variables present in both datasets are numeric measures of chemical attributes, and the remaining three comprise each record's ID, wine type, and location.

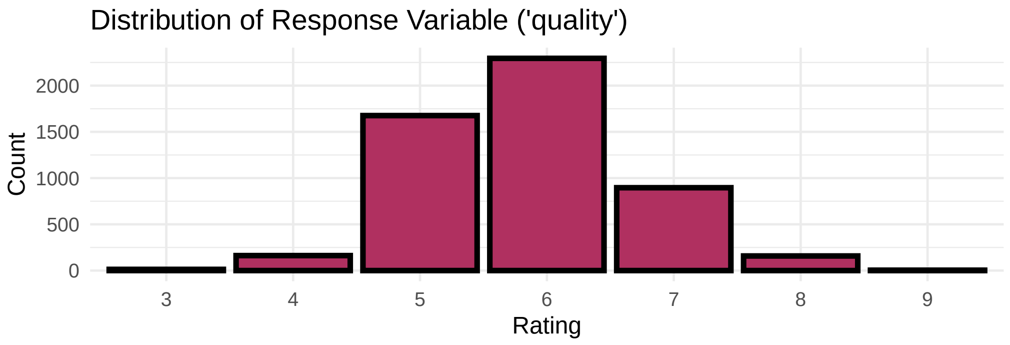
The data cleaning step is straightforward, requiring only a correction of the misspelling "Califormia" in the `location` attribute to address patently incorrect data. For outliers, the $3 \bullet \text{IQR}$ rule was selected because it was far less aggressive than $1.5 \bullet \text{IQR}$—too aggressive, and the procedure could remove too much information from the data.

The following plot matrix titled "Distributions of Numeric Attributes" shows the distributional shapes of the numeric data to be modeled.



The following three plots show the counts of wines in each category of the variables `type` and `location` and in each level of the response variable `quality`:





Analysis: Identifying & Predicting Relationships

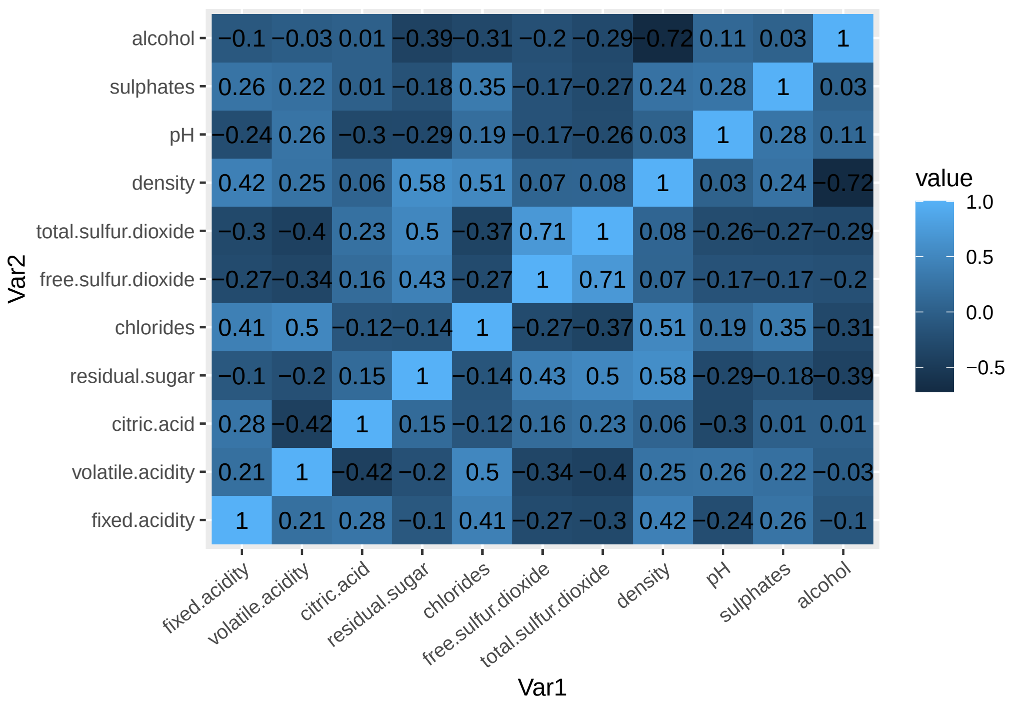
Nature of the Ordinal Logit Regression Model and Its Assumptions

One model well-suited to analyze this data, with the goal of predicting a whole number level from 0 to 10 in strictly an order of least to largest number value, is an *ordinal logit*, variably known as ordinal logistic regression or ordered logit regression. Recent research has been exploring uses of advanced and computation-intensive algorithms from the realm of Machine Learning, such as Random Forest and Gradient Boosting, for predicting ordinal response variables; however, much of that research and the associated R packages for such models are relatively new and conceptually advanced, and so for the sake of the statistical roots of Data Science as a field, this study will focus on the older techniques of regression.

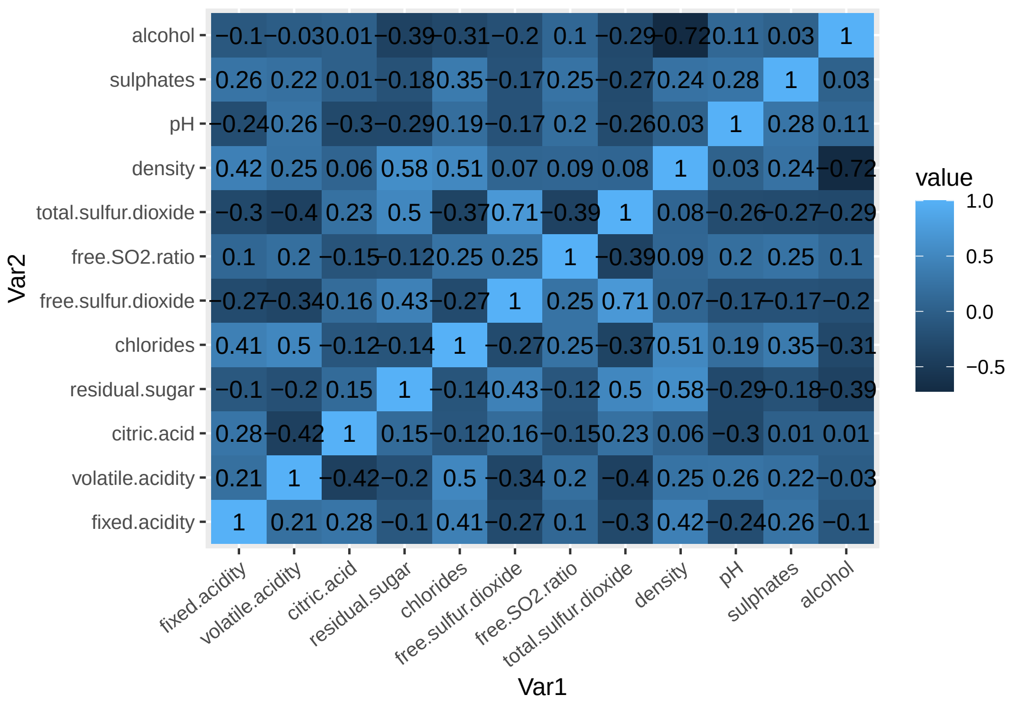
An ordinal logit model can take as arguments explanatory variables of types including "continuous, categorical, or ordinal"(St. Andrews CEED Math Support, n.d.); this is one of the model's assumptions.

This model also assumes that (paraphrased from St. Andrews PDF on Ordinal Regression):

1. ***The dependent (target/response) variable is ordinal:*** this is **satisfied**, with `quality` having the theoretically possible levels 0 to 10 in whole number steps, ordered from lowest, 0; to highest, 10.
2. ***At least one of the independent (explanatory) variables is/are categorical, or continuous, or also ordinal:*** this is **satisfied** by the two categoricals, `location` and `type`; and the eleven continuously numeric measures of chemical properties.
3. ***There must be no multicollinearity:*** this can be checked by creating a correlation matrix upon the explanatory variables.
4. ***Proportional Odds are assumed:*** this can be checked once the model has been run by running a Brant test.



We can see from the correlation matrix that there is no severe multicollinearity; no variable shows correlation coefficient near 1 with any other variable aside from itself. The most significant correlation we can see is a positive correlation (0.72) between `free.sulfur.dioxide` and `total.sulfur.dioxide`, indicating that together the two attributes constitute a duplication of information in the dataset. The second most significant correlation is an inverse one (-0.70) between density and alcohol; this makes chemical sense considering that ethanol alcohol's density is 79% of water's—as the alcohol in solution increases, the density of the solution decreases from the density of water(Marketing and Communications, n.d.).



By creating a new attribute, `free.SO2.ratio`, containing decimal values for each wine's free sulfur dioxide as a fraction of the total, the dataframe retains all of the information concerning sulfur dioxide, but with the option to remove either the free or total amount or both, with the ratio having insignificant correlation with either attribute from which it was calculated. This opens up options for the attribute selection process by which the model will be tuned.

The final step in data preprocessing for the ordinal regression is the scaling of it.

We can now proceed to the modeling phase of our analysis; when this is complete, the model may be assessed for proportional odds.

Preparing the Data and Applying the Model to the Split Training Data

Because the goal is to get predictions as close as possible to the actual qualities for the Test Data without knowing what those quality values are, the Test Data doesn't contain the quality column. If we proceeded as is typical, we would be flying blind or taking a shot in the dark, whichever metaphor one prefers.

To address this, we will begin by splitting the Training Data into training and testing sets using a 70-30 ratio using the `caTools` package. This will give us a chance to evaluate the margin of error of our predictions. If any tuning to model parameter inputs is needed, it can be done before retraining the model fresh—but with tuned parameters—on the entirety of the Training Data. This way we can ensure that the model in its default state won't overtrain on the Training Data in its complete state and that any changes in parameters that would be possible in a typical situation with full data can be done to some extent rather than not be done at all. We will use the `polr` package to accomplish modeling, per the procedure laid out by UCLA's Advanced Research Computing department (Computing, n.d.).

```
## [1] "RAW MODEL AND ITS ACCURACY"
```

```
## Call:
## polr(formula = quality ~ fixed.acidity + volatile.acidity + citric.acid +
##     residual.sugar + chlorides + free.sulfur.dioxide + free.SO2.ratio +
##     total.sulfur.dioxide + density + pH + sulphates + alcohol +
##     type + location, data = tune.train, Hess = T)
##
```

```
## Coefficients:
##                        Value Std. Error  t value
## fixed.acidity         0.32973    0.07304   4.5141
## volatile.acidity     -0.45773    0.05053  -9.0591
## citric.acid           0.01009    0.04167   0.2421
## residual.sugar        0.89296    0.11847   7.5376
## chlorides            -0.09594    0.05478  -1.7516
## free.sulfur.dioxide   0.11846    0.09143   1.2957
## free.SO2.ratio        0.66136    0.56929   1.1617
## total.sulfur.dioxide -0.06344    0.09384  -0.6761
## density              -1.10374    0.18523  -5.9589
## pH                    0.32187    0.05610   5.7377
## sulphates             0.26006    0.04096   6.3490
## alcohol               0.39093    0.09393   4.1619
## typewhite            -1.28658    0.23459  -5.4843
## locationTexas        -1.93139    0.08340 -23.1589
##
## Intercepts:
##      Value     Std. Error t value
## 0|1   -14.8480    2.6848    -5.5303
## 1|2   -13.5191    2.2373    -6.0425
## 2|3   -12.9363    2.0096    -6.4371
## 3|4    -8.7936    0.4405   -19.9609
## 4|5    -6.0345    0.3006   -20.0769
## 5|6    -2.4127    0.2799    -8.6209
## 6|7     0.5626    0.2764     2.0354
## 7|8     2.8836    0.2885     9.9970
## 8|9     6.6085    0.6397    10.3301
## 9|10  722.0877    0.6397  1128.7318
##
## Residual Deviance: 7296.621
## AIC: 7344.621
```

```
##   0   1   2   3   4   5   6   7   8   9  10
##   0   0   0   0   0 508 926 127   0   0   0
```

```
## [1] 0.4106342
```

```
## [1] 5.958406
```

We can see from the MAPE (Mean Absolute Percent Error) that our model in its raw state, taking all variables as inputs, is already quite accurate. Here we are using MAPE to quantify the accuracy of the predictions independent of scale; the MAPE value of 6.19% is the rate of incorrect predictions—ergo, the model is ~93.81% accurate. From the logic that all of the measurements included in the data have a real, physical or chemical relationship to the material—wine—being studied, we can conject that this model is the most realistic in terms of those real-world mechanisms. However, it is of interest if the model could be more accurate with only several of the most significant variables considered; it is possible that some chemical properties do not manifest in ways that are reliably or in any orderly way detected by the palate of a sommelier. There are chemical compounds that have no flavor and/or no texture (in terms of mouthfeel); perhaps citric acid, despite its fame as a component of the distinctly tart flavors of citrus fruits, has little effect on the overall quality of flavor of a wine. Apropos of the effects of acids on taste and mouthfeel, it is of some interest that the data contains no measure of tannins, the principal source of the astringent mouthfeel of some wines including and especially red wines. Red wines differ from whites in the winemaking process in that the juice is allowed to ferment with the grapes' seeds and skins immersed (Eckstein 2021). This analysis would be well worth the revision if new data containing measures of tannins were to become available.

To assess the ideal combination of variables, we apply stepwise selection in various directions and compare the resulting AIC values.

```
##                  df      AIC
## fit.wine.tune.0 24 7344.621
## fit.wine.0.fwd  22 7341.270
## fit.wine.0.back 22 7341.270
## fit.wine.0.foba 22 7341.270
```

Because of the inherent flexibility in a stepwise variable selection process moving 'both ways', the model `fit.wine.0.foba` is likely the ideal choice, even if the backwards-elimination method produces identical results, and console explorations of theses models' summaries suggest that to be the case.

Next, we cross-validate this model, using the validation set we set aside earlier as `tune.test`. The MAPE indicates this model is slightly less accurate than the raw model, but the results are more reliably significant thanks to the rigor of the stepwise selection process.

```
## [1] "FOR THE SELECTED MODEL:"
```

```
## Call:
## polr(formula = quality ~ fixed.acidity + volatile.acidity + residual.sugar +
##     chlorides + free.sulfur.dioxide + free.SO2.ratio + density +
##     pH + sulphates + alcohol + type + location, data = tune.train,
##     Hess = T)
##
## Coefficients:
##                         Value Std. Error t value
## fixed.acidity        0.33658    0.07184   4.685
## volatile.acidity    -0.46610    0.04688  -9.942
## residual.sugar       0.90024    0.11819   7.617
## chlorides           -0.09573    0.05477  -1.748
## free.sulfur.dioxide  0.07094    0.05208   1.362
## free.SO2.ratio       0.89907    0.39880   2.254
## density             -1.12170    0.18358  -6.110
## pH                   0.32179    0.05587   5.759
## sulphates            0.25988    0.04089   6.356
## alcohol              0.38752    0.09317   4.159
## typewhite           -1.32544    0.23032  -5.755
## locationTexas       -1.93003    0.08338 -23.147
##
## Intercepts:
##      Value    Std. Error t value
## 0|1  -14.3915   3.4487    -4.1731
## 1|2  -13.2481   2.7125    -4.8841
## 2|3  -12.5839   2.1722    -5.7931
## 3|4   -8.7572   0.4345   -20.1529
## 4|5   -5.9980   0.2894   -20.7239
## 5|6   -2.3750   0.2677    -8.8712
## 6|7    0.6002   0.2644     2.2704
## 7|8    2.9208   0.2771    10.5402
## 8|9    6.6359   0.6328    10.4862
## 9|10 543.4685   0.6328   858.8028
##
## Residual Deviance: 7297.27
## AIC: 7341.27
```

```
## For the selected features  fixed.acidity, volatile.acidity, residual.sugar, chlorides, free.sulfur.di
```

```
##
## We find the MAE for CV on the Validation Data: 0.4113
```

```
##
## And the MAPE for the same: 5.97
```

The Brant test on our chosen model presents a problem, however.

Likely because several theoretically possible levels of `quality` have no observations, and because via the function `range()` we find that the predictions by that model range from 6 to 8, the Brant Test algorithm fails to converge.

That being said, the model itself does converge and produces results. We can expect that the results can be generalized to wines that would be rated between 6 and 8, inclusive; this issue leaves the results on shaky but not necessarily crumbling ground.

```
## Warning: with 6 zero cell entries in the computation crosstab, test results may
## be inaccurate.
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Brant Test:
##                       chi-sq   df   pr(>chi)
## Omnibus              173.294   60    6.2e-13 ***
## fixed.acidity          0.235    5     0.9987
## volatile.acidity       0.619    5     0.9871
## residual.sugar         0.777    5     0.9785
## chlorides              1.972    5     0.8531
## free.sulfur.dioxide   51.359    5    7.3e-10 ***
## free.SO2.ratio        26.631    5    6.7e-05 ***
## density                2.241    5     0.8150
## pH                     3.564    5     0.6137
## sulphates              0.110    5     0.9998
## alcohol               17.079    5     0.0044 **
## typewhite             43.843    5    2.5e-08 ***
## locationTexas          3.493    5     0.6244
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## H0: Proportional odds assumption holds
```

This model formula is rudimentarily as follows:

$quality \sim fixed.acidity + volatile.acidity + residual.sugar + free.SO2.ratio + density + pH + sulphates + alcohol + typewhite + locationTexas$

However, because this model doesn't converge and—on account of the data causing sparseness in the theoretically possible levels of `quality`—violates the Proportional Odds assumption. A common approach to this is to use a Multinomial Logit that doesn't make this assumption, but this is inappropriate for our study because of the Independence of Irrelevant Alternatives assumption.

This assumption posits that any added option in the categorical—but in the case of a Multinomial Logit standing in for Ordinal Logit, unordered—response variable will decrease the likelihood of all others by equal fractions(Benson, Kumar, and Tomkins 2016). This, however, doesn't hold in the wine ratings realm, because the sparseness of the data so far presents the clear scenario of wines reaching higher ratings than in the past due to whatever improvements the vintners made—and as a result, the highest-scoring wines in current data would later seem less premium. A shopper's decision would be changed by what could be called an "Overton Window" of wine ratings having either or both sides of the frame or the center of it adjusted.

For this reason, and because the Ordinal Logit model gave highly accurate results, it may be most useful in the context and scope of this study to apply Box's famous late-seventies axiom, that "all models are wrong, but some are useful", to offer recommendations based on the model chosen by the both-ways stepwise variable selection. By doing so, we sacrifice only a marginal amount of prediction accuracy for a model with the least relevant explanatory variables removed to offer a more efficient route to wine quality optimization for vintners.

From this, we have a model with the following coefficients with their 95% confidence intervals and P-values shown in the table below. Since the numeric data is normalized to a mean of zero and standard deviation of one, the model coefficients represent log-odds changes in `quality` rating for changes in standard deviation of the explanatory variables. Peculiarly, the coefficient for free sulfur dioxide seems not to be significant, yet the stepwise selection process included it anyway.

```
##                      Estimate Std.Error  Z.value P.value CI.Lower CI.Upper
## fixed.acidity          0.3646    0.0606   6.0207  0.0000   0.2459   0.4833
## volatile.acidity      -0.4556    0.0392 -11.6217  0.0000  -0.5324  -0.3787
## residual.sugar         0.9270    0.0980   9.4595  0.0000   0.7349   1.1191
## chlorides             -0.0758    0.0453  -1.6743  0.0941  -0.1646   0.0129
## free.sulfur.dioxide    0.0676    0.0433   1.5592  0.1189  -0.0174   0.1525
## free.SO2.ratio         1.0151    0.3365   3.0168  0.0026   0.3556   1.6746
## density               -1.1391    0.1524  -7.4747  0.0000  -1.4378  -0.8404
## pH                     0.3297    0.0467   7.0639  0.0000   0.2383   0.4212
## sulphates              0.2768    0.0342   8.0944  0.0000   0.2098   0.3438
## alcohol                0.4047    0.0771   5.2515  0.0000   0.2536   0.5557
## typewhite             -1.2817    0.1884  -6.8023  0.0000  -1.6510  -0.9124
## locationTexas         -1.9872    0.0704 -28.2345  0.0000  -2.1252  -1.8493
```

RShiny Dashboard

For the purposes of wine optimization, we built an RShiny applet, titled "The Analytical Sommelier", to allow interactive forecasting of wine quality using our final model. The code for this was developed with the assistance of the Claude AI LLM by Anthropic.

The dashboard allows users to select any combination of attribute values from those present in the training data to customize a wine profile, and determine what the model predicts the rating would be, along with a probability distribution and model information for the prediction.

Further study with an emphasis on domain knowledge of winemaking could better inform the attribute slider ranges to allow for customizing the profile outside of the wines represented in the data.

Conclusions

From the data given, we have fit an ordinal logit model with a prediction accuracy, based on our testing using a subset of the full provided training data, of \$5.97%\$.

These results are generalizable to red and white wines produced in California and Texas, and specifically in the context of ratings made by the panel of sommeliers from which this data was derived. We, as the authors of this study, are not liable for successes or failures in the winemaking process resulting from actions inspired by this study's results.

Because of the issue with the Proportional Odds assumption, the results ought to be taken with a grain of salt despite the model's predictive accuracy. For a more robust modeling of the data, more complex techniques in the deep learning realm of data science may be promising. These results are intended to provide our client with rudimentary recommendations for experimenting with their winemaking processes; apropos of that, the model suggests that broadly speaking, red wines outperform whites and California wines outperform Texas wines. Further study would be prudent to investigate any interactions between type and location.

More depth in the data could also benefit this topic of study, with attention paid to wines from more locales, more types and/or separation of types into subtypes, and more information about the chemical makeup of a wine—such as quantities of tannins, which can affect flavor and mouthfeel; and aspects of the winemaking process(Eckstein 2021).

Further data collection and model type exploration are indicated for this data and the results suggest analytical quality optimization could prove profitable.

References

Benson, Austin R., Ravi Kumar, and Andrew Tomkins. 2016. "On the Relevance of Irrelevant Alternatives." In *Proceedings of the 25th International Conference on World Wide Web*, 963–73. WWW '16. Republic; Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee. https://doi.org/10.1145/2872427.2883025.

Bobbitt, Zach. 2020. "How to Remove Outliers from Multiple Columns in R." https://www.statology.org/remove-outliers-from-multiple-columns-in-r/.

———. 2022. "How to Split Data into Training & Test Sets in R (3 Methods)." https://www.statology.org/train-test-split-r/.

Computing, UCLA Advanced Research. n.d. https://stats.oarc.ucla.edu/r/dae/ordinal-logistic-regression/.

Eckstein, Danielle. 2021. "Official playbook to the difference between white wine & red wine." https://www.7cellars.com/blogs/7cellars/official-playbook-to-the-difference-between-white-wine-red-wine#:~:text=For%20white%20wines%2C%20the%20grapes,in%20the%20distinctive%20darker%20color.

GeeksforGeeks. 2022. "How to create correlation heatmap in R." https://www.geeksforgeeks.org/how-to-create-correlation-heatmap-in-r/.

Marketing, Grainger Engineering Office of, and Communications. n.d. "Liquids more dense than water or alcohol." https://van.physics.illinois.edu/ask/listing/1738#:~:text=A%3A,1.0%20g%2Fcc%20for%20water.

Mulani, Safa. 2022. "How to Normalize data in R [3 easy methods] | DigitalOcean." https://www.digitalocean.com/community/tutorials/normalize-data-in-r.

St. Andrews CEED Math Support, University of. n.d. "ORDINAL REGRESSION." https://www.st-andrews.ac.uk/media/ceed/students/mathssupport/ordinal%20logistic%20regression.pdf.

Appendix: All Code

```
knitr::opts_chunk$set(
  echo = FALSE,
  out.width = "0.8\\textwidth",
  fig.align = 'center',
  tidy=TRUE)

require(brant)
require(dplyr)
require(foreign)
require(ggplot2)
require(Hmisc)
require(MASS)
require(ordinal)
require(Metrics)
require(reshape2)
require(rstatix)
require(scales)
require(tidyverse)
require(gridExtra)
require(gofcat)
require(car)
require(nnet)

trainFull <- read.csv("data/WineTrainSet.csv")
testFull <- read.csv("data/WineTestSet.csv")
# identify numeric column names for numeric variable data wrangling steps
numCol <- c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar",
↪  "chlorides", "free.sulfur.dioxide", "total.sulfur.dioxide", "density", "pH",
↪  "sulphates", "alcohol")

# check NAs
sum(is.na(trainFull))
sum(is.na(testFull))

# find all levels of cat vars
unique(trainFull[["type"]])
unique(trainFull[["location"]])
unique(trainFull[["quality"]])

unique(testFull[["type"]])
unique(testFull[["location"]])


# fix cat var errors and check result
trainFull$location[trainFull$location == "Califormia"] <- "California"
testFull$location[testFull$location == "Califormia"] <- "California"

unique(trainFull[["location"]])
unique(testFull[["location"]])

# change data types of categoricals and ordinal response to factor
```

```r
trainFull$type <- as.factor(trainFull$type)
trainFull$location <- as.factor(trainFull$location)
trainFull$quality <- factor(trainFull$quality, c(0,1,2,3,4,5,6,7,8,9,10), ordered = TRUE)

testFull$type <- as.factor(testFull$type)
testFull$location <- as.factor(testFull$location)

# Outlier detection & removal
idOL <- function(data, colName) {
  colDat <- data[[colName]]
  q1 <- quantile(colDat, 0.25, na.rm = TRUE)
  q3 <- quantile(colDat, 0.75, na.rm = TRUE)
  iqr <- q3 - q1
  lwrBnd <- q1 - 3 * iqr
  uprBnd <- q3 + 3 * iqr
  allvalsOL <- which(colDat < lwrBnd | colDat > uprBnd)
  return(allvalsOL)
}

# Remove outliers
allrowsOL <- c()
for (col in numCol) {
  allvalsOL <- idOL(trainFull, col)
  allrowsOL <- c(allrowsOL, allvalsOL)
}

uniqrowsOL <- unique(allrowsOL)
trainClean <- trainFull[-uniqrowsOL, ]

cat("Original dataset:", nrow(trainFull), "rows,", ncol(trainFull), "columns\n")
cat("After outlier removal:", nrow(trainClean), "rows (removed", length(uniqrowsOL),
↪  "outliers)\n\n")

#data distribution plots, numeric vars
trainClean %>%
 select(all_of(numCol)) %>%
 pivot_longer(everything(), names_to = "variable", values_to = "value") %>%
 ggplot(aes(x = value)) +
 geom_histogram(fill = "maroon", color = "black") +
 facet_wrap(~ variable, scales = "free") +
 theme_minimal() +
 labs(title = "Distributions of Numeric Attributes")

#data plots, categoricals and response var
barCat <- trainClean %>%
 select(type, location) %>%
 pivot_longer(everything(), names_to = "variable", values_to = "value") %>%
 ggplot(aes(x = factor(value))) +
 geom_bar(fill = "maroon", color = "black", linewidth = 1.2) +
 facet_wrap(~ variable, scales = "free") +
 theme_minimal() +
 labs(title = "Distributions of Categorical Attributes",
      x = "Value", y = "Count")
```

```r
barOrd <- trainClean %>%
 select(quality) %>%
 ggplot(aes(x = factor(quality))) +
 geom_bar(fill = "maroon", color = "black", linewidth = 1.2) +
 theme_minimal() +
 labs(title = "Distribution of Response Variable ('quality')",
      x = "Rating", y = "Count")

grid.arrange(barCat, barOrd, ncol = 1)

# Correlation Matrix
# code in this section derived from GeeksforGeeks 2022
corrMat <- round(cor(trainClean[,2:12]),2)

melCorrMat <- melt(corrMat) #aka morgothMat

ggplot(data = melCorrMat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
  theme(axis.text.x = element_text(angle = 38, hjust = 1))

# add Sulfur Dioxide ratio column and re-check collinearity
trainClean2 <- trainClean %>%
 mutate(free.SO2.ratio = free.sulfur.dioxide / total.sulfur.dioxide) %>%
 relocate(free.SO2.ratio, .after = free.sulfur.dioxide)

corrMat2 <- round(cor(trainClean2[,2:13]),2)

melCorrMat2 <- melt(corrMat2) #aka morgothMat2

ggplot(data = melCorrMat2, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
  theme(axis.text.x = element_text(angle = 38, hjust = 1))
trainClean3 <- trainClean2 %>%
  mutate(
    # Scale continuous variables
    fixed.acidity = scale(fixed.acidity)[,1],
    volatile.acidity = scale(volatile.acidity)[,1],
    citric.acid = scale(citric.acid)[,1],
    residual.sugar = scale(residual.sugar)[,1],
    chlorides = scale(chlorides)[,1],
    free.sulfur.dioxide = scale(free.sulfur.dioxide)[,1],
    # free.SO2.ratio was scaled in its creation.
    free.SO2.ratio = free.SO2.ratio,
    total.sulfur.dioxide = scale(total.sulfur.dioxide)[,1],
    density = scale(density)[,1],
    pH = scale(pH)[,1],
    sulphates = scale(sulphates)[,1],
    alcohol = scale(alcohol)[,1],
    # factors are not scaled
    type = type,
    location = location,
    # quality has already been ordered
    quality = quality
  )

# split trainClean3
# code in this section derived from Bobbitt 2022
set.seed(1)
tune.train <- trainClean3 %>% dplyr::sample_frac(0.70)
tune.test  <- dplyr::anti_join(trainClean3, tune.train, by = 'ID')

# train model, predict, calculate MAE
print("RAW MODEL AND ITS ACCURACY")
fit.wine.tune.0 <- polr(formula = quality ~ fixed.acidity + volatile.acidity +
↪  citric.acid + residual.sugar + chlorides + free.sulfur.dioxide + free.SO2.ratio +
↪  total.sulfur.dioxide + density + pH + sulphates + alcohol + type + location, data =
↪  tune.train, Hess = T)
predict.wine.tune.0 <- predict(fit.wine.tune.0, newdata = tune.test, type = "class")

summary(fit.wine.tune.0)
summary(predict.wine.tune.0)

#calculate MAE, MAPE for raw model
mae.wine.0 <- mae(actual = as.numeric(tune.test$quality), predicted =
↪  as.numeric(predict.wine.tune.0))
mape.wine.0 <- mape(actual = as.numeric(tune.test$quality), predicted =
↪  as.numeric(predict.wine.tune.0))
mae.wine.0
mape.wine.0*100

# forward selection
fit.wine.0.fwd <- stepAIC(polr(quality ~ 1, data = tune.train, Hess = T), scope =
↪  list(lower = ~ 1, upper = ~ fixed.acidity + volatile.acidity + citric.acid +
↪  residual.sugar + chlorides + free.sulfur.dioxide + free.SO2.ratio +
↪  total.sulfur.dioxide + density + pH + sulphates + alcohol + type + location),
↪  direction = "forward")

# backward elimination
fit.wine.0.back <- stepAIC(fit.wine.tune.0, direction = "backward")

# forward and backward
fit.wine.0.foba <- stepAIC(fit.wine.tune.0, direction = "both")

# compare AIC values
AIC(fit.wine.tune.0, fit.wine.0.fwd, fit.wine.0.back, fit.wine.0.foba)

fit.wine.1 <- fit.wine.0.foba
#cross-validation on validation set
fit.1.feats <- attr(terms(fit.wine.1), "term.labels")
fit.1.form <- as.formula(paste("quality ~", paste(fit.1.feats, collapse = " + ")))

pred.1.val.probs <- predict(fit.wine.1, tune.test, type = "probs")
```

```r
pred.1.val.class <- apply(pred.1.val.probs, 1, which.max)

pred.1.val.MAE <- mae(actual = as.numeric(tune.test$quality), predicted =
↪  as.numeric(pred.1.val.class))
pred.1.val.MAPE <- mape(actual = as.numeric(tune.test$quality), predicted =
↪  as.numeric(pred.1.val.class))

print("FOR THE SELECTED MODEL:")
summary(fit.wine.1)
#summary(pred.1.val.probs)

cat("For the selected features ", paste(fit.1.feats, collapse = ", "), "\n")
cat("\nWe find the MAE for CV on the Validation Data:", round(pred.1.val.MAE, 4), "\n")
cat("\nAnd the MAPE for the same:", round(pred.1.val.MAPE, 4)*100, "\n")

#check proportional odds assessment
brant.test(fit.wine.1)

# prepare testing dataset-add free.SO2.ratio attribute
testFull2 <- testFull %>%
 mutate(free.SO2.ratio = free.sulfur.dioxide / total.sulfur.dioxide) %>%
 relocate(free.SO2.ratio, .after = free.sulfur.dioxide)

# train the final model on the whole Training Dataset
fit.wine.fin <- polr(fit.1.form, data = trainClean3, Hess = TRUE)
pred.wine.fin <- predict(fit.wine.fin, newdata = testFull2, type = "class")

# export predictions for Testing Dataset
predictions.wine <- data.frame(testFull$ID, pred.wine.fin)
colnames(predictions.wine) <- c("ID","quality")
write.csv(predictions.wine, file = "results/WineQualityPredictions.csv", row.names =
↪  FALSE)

# present model coefficients, their CIs, and their p-values
fit.wine.fin.coeffs <- summary(fit.wine.fin)$coefficients
fwf.predr.coeffs <- fit.wine.fin.coeffs[!grepl("\\|", rownames(fit.wine.fin.coeffs)), ]

fit.wine.fin.zvals <- fwf.predr.coeffs[, "Value"] / fwf.predr.coeffs[, "Std. Error"]
fit.wine.fin.pvals <- 2 * (1 - pnorm(abs(fit.wine.fin.zvals)))

fwf.tc <- qt(0.975, summary(fit.wine.fin)$df.residual) # t_crit

fit.wine.fin.ciLWR <- fwf.predr.coeffs[, "Value"] - 1.96 * fwf.predr.coeffs[, "Std.
↪  Error"]
fit.wine.fin.ciUPR <- fwf.predr.coeffs[, "Value"] + 1.96 * fwf.predr.coeffs[, "Std.
↪  Error"]

fwf.results <- data.frame(
  Estimate = round(fwf.predr.coeffs[, "Value"], 4),
  Std.Error = round(fwf.predr.coeffs[, "Std. Error"], 4),
  Z.value = round(fit.wine.fin.zvals, 4),
  P.value = round(fit.wine.fin.pvals, 4),
  CI.Lower = round(fit.wine.fin.ciLWR, 4),
  CI.Upper = round(fit.wine.fin.ciUPR, 4)
)

head(fwf.results, 12)

# Export model as RDS file to allow Dashboard to use it for predictions; do same for
↪  training data
saveRDS(fit.wine.fin, file = "WineQualDash/fwf.Rds")
write.csv(trainClean, file = "WineQualDash/trainClean.csv", row.names = FALSE)
```

NOTE: this code was developed with the assistance of Claude AI, a product of Anthropic PBC.