Understanding the Quality of Wine Revealed In Chemistry

Iris Cannary

2024-08-22

Abstract

asdf

Context

The quality of wine can be mystifying despite being of central importance to the practice of making it. However, that quality may be no enigma whatsoever for vintners and sommeliers, whose professions require diverse experience with and deep understanding of wine; and the well-to-do, whose means allow them to have similarly diverse experiences with the potation and the social impetus to seek wines of the highest perceived quality.

Vintners seek to acquire a map to lead their grapes to becoming the most sought-after wines on the market. Wine consumers, like society as a whole, highly value wine as a social drink and a quenching garnish to their meals, and because of that value system pursue ever more pleasing and compatible wines. The most wealthy and high-status of these consumers add to that system the use of the most valuable wines as markers of their wealth and status—consider the banker in the film *The Big Short* asking a colleague "What's with the Dom?", referring to a celebration of a lucrative deal with the famously expensive champagne Dom Pérignon. One may also consider the scene in the limited series *Anatomy of A Scandal* during which a collegiate social club of aristocratic young men launch into a boisterous chant about wasting "Bolly", referring to the similarly expensive Bollinger. Sommeliers, then, have a reputation and business liasonship to maintain as middlefolk to the enterprising vintners pursuing firm and lasting footing in the market and the diverse and eager consumers pursuing the best wines they can acquire.

In this study, we aim to leverage the statistical sciences to uncover just *how* a wine, on account of its constituent substances and their combination, achieves a particular level of quality. As society continues its long march from a worldview grounded in religion, myth, and superstition to one built on a fundament of science, the work achieved by this study offers a valuable transition of the basis of wine quality from a subjective realm to an objective one. Foremost among the practical benefits to this are the improved cost efficiency for vintners to adjust their grape growing and winemaking techniques and the bolstered confidence of the consumer when choosing a wine.

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 data set contains 1,034 records of all of the same except for quality values. The target, of course, is quality, which makes the test set peculiar; eleven of the other variables are numeric measures of chemical attributes. The remaining three comprise each record's ID, and then the wine's type and location.

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

```
## [1] 0
```

```
sum(is.na(testFull))
```

```
## [1] 0
```

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

```
## [1] "white" "red"
```

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

```
## [1] "Texas"      "California" "Californa"
```

```
unique(trainFull[["quality"]])
```

```
## [1] 5 4 6 7 8 3 9
```

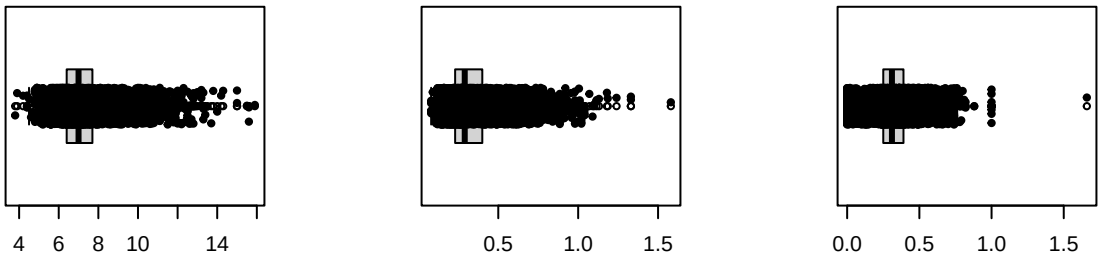
```
unique(testFull[["type"]])
```

```
## [1] "red"   "white"
```

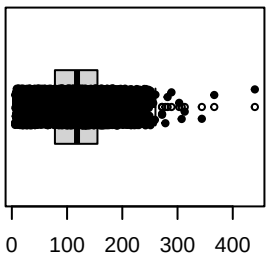
```
unique(testFull[["location"]])
```

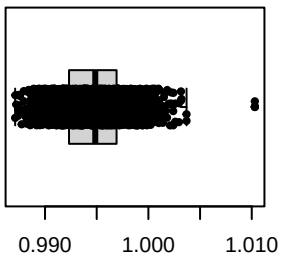
```
## [1] "California" "Texas"      "Californa"
```

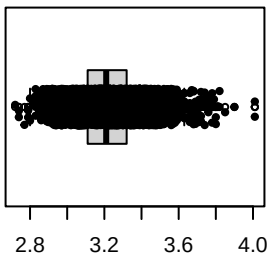
```
# find outliers 1
par(mfrow = c(2, 3))
invisible(lapply(2:7, function(i) {
  boxplot(trainFull[, i], horizontal = TRUE)
  stripchart(trainFull[, i], method = "jitter", pch = 19, add = TRUE)}))
```

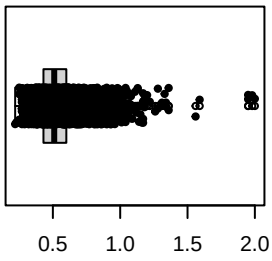


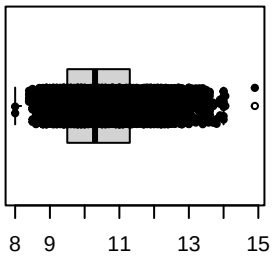

```
# find outliers 2
par(mfrow = c(2, 3))
invisible(lapply(8:12, function(i) {
  boxplot(trainFull[, i], horizontal = TRUE)
  stripchart(trainFull[, i], method = "jitter", pch = 19, add = TRUE)}))
```











The data is conveniently quite clean, having no null values at the outset. However, we can see that in the location variable for both data sets, some number of records contain the misspelled state name "California". We can also see by the boxplot-stripchart combination that several of the numerical variables show outliers. One's first instinct may be to only chase down the outliers in the attributes where they are obvious on the plot, but for consistency, every attribute will have any outliers removed subject to the conventional $1.5 \bullet \text{IQR}$ rule, eliminating any data more than 150% of the **i**nter**q**uartile **r**ange under the first quartile or over the third.

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

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

```
## [1] "Texas"      "California"
```

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

```
## [1] "California" "Texas"
```

```
# remove outliers
# code in this section derived from Bobbitt 2020
detect_outliers_1 <- function(x) {

  Q1 <- quantile(x, probs=.25)
  Q3 <- quantile(x, probs=.75)
  iqr = Q3-Q1

 upper_limit = Q3 + (iqr*1.5)
 lower_limit = Q1 - (iqr*1.5)

 x > upper_limit | x < lower_limit
}
```

```r
remove_outliers_1 <- function(df, cols = names(df)) {
  for (col in cols) {
    df <- df[!detect_outliers_1(df[[col]]),]
  }
}

remove_outliers_1(trainFull, c("fixed.acidity","volatile.acidity","citric.acid","residual.sugar","chlor
```

It appears that no outliers have been removed. It is unclear if this is a result of ineffectual code or the fact that none of the values that looked like outliers in the plots actually fit the definition. This may warrant further inspection.

If indeed there are no outliers per the $1.5 \bullet \text{IQR}$ definition, then the data is now clean, having had the locations standardized and being shown to have no NA/null values. We may proceed to exploring a suitable model and checking that the data satisfies the model's assumptions.

Analysis: Identifying & Predicting Relationships

Nature of the Model and its Assumptions

The ideal model 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 regression, variably known as ordinal logistic regression or ordered logit regression.

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

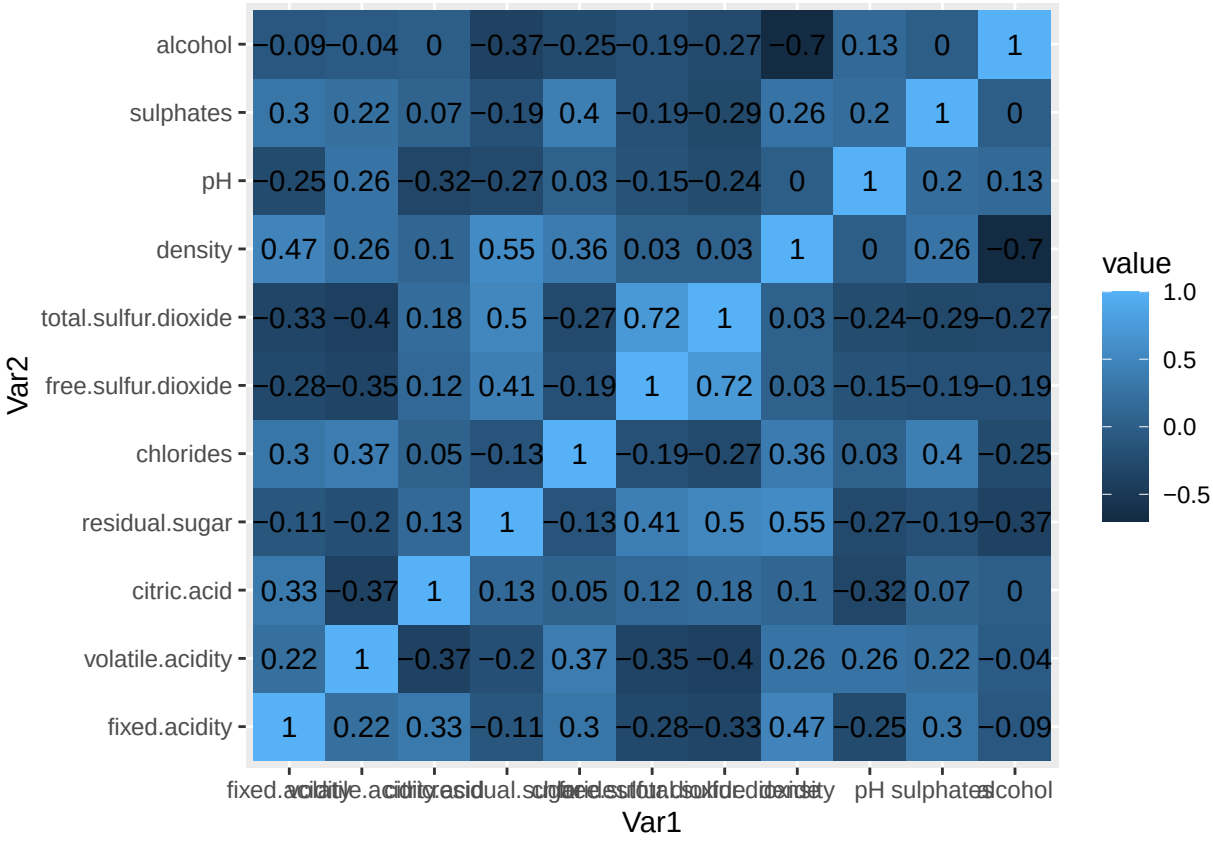
The model also assumes that (paraphrased from St. Andrews PDF on Ordinal Regression):

1. ***The dependent (target) variable is ordinal:*** this is **satisfied**, with 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 using the `brant` package.

```r
# Correlation Matrix
# code in this section derived from GeeksforGeeks 2022
corr_mat <- round(cor(trainFull[,2:12]),2)

melted_corr_mat <- melt(corr_mat)
#head(melted_corr_mat)

ggplot(data = melted_corr_mat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 4)
```



We can see from the correlation matrix that there is no multicollinearity; no variable shows significant correlation with any other variable aside from itself. 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 alone 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 (n.d.).

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

# train model
```

Revelations

n.d. https://stats.oarc.ucla.edu/r/dae/ordinal-logistic-regression/.

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.