Business Analytics and Data Mining

DATA621 Homework 5

William Outcault, Mengqin Cai, Philip Tanofsky, Robert Welk, Zhi Ying Chen

13 December 2020

Contents

Overview

In this homework assignment, you will explore, analyze and model a data set containing information on approximately 12,000 commercially available wines. The variables are mostly related to the chemical properties of the wine being sold. The response variable is the number of sample cases of wine that were purchased by wine distribution companies after sampling a wine. These cases would be used to provide tasting samples to restaurants and wine stores around the United States. The more sample cases purchased, the more likely is a wine to be sold at a high end restaurant.

A large wine manufacturer is studying the data in order to predict the number of wine cases ordered based upon the wine characteristics. If the wine manufacturer can predict the number of cases, then that manufacturer will be able to adjust their wine offering to maximize sales. The objective is to build a count regression model to predict the number of cases of wine that will be sold given certain properties of the wine.

Below is a short description of the variables of interest in the data set:

- INDEX: Identification Variable (No theoretical effect)
- TARGET: Number of Cases Purchased (No theoretical effect)
- AcidIndex: Proprietary method of testing total acidity of wine by using a weighted average
- Alcohol: Alcohol Content
- Chlorides: Chloride content of wine
- CitricAcid: Citric Acid Content
- Density: Density of Wine
- FixedAcidity: Fixed Acidity of Wine
- FreeSulfurDioxide: Sulfur Dioxide content of wine
- LabelAppeal: Marketing Score indicating the appeal of label design for consumers. High numbers suggest customers like the label design. Negative numbers suggest customers don't like the design. (Theoretical effect: Many consumers purchase based on the visual appeal of the wine label design. Higher numbers suggest better sales.)
- ResidualSugar: Residual Sugar of wine
- STARS Wine rating by a team of experts. 4 Stars = Excellent, 1 Star = Poor. (Theoretical effect: A high number of stars suggests high sales)
- Sulphates: Sulfate content of wine
- TotalSulfurDioxide: Total Sulfur Dioxide of Wine
- VolatileAcidity: Volatile Acid content of wine
- pH: pH of wine

Data Exploration

Exploring 12,000 commercially available wines, specifically the chemical properties of the wine being sold. Dependent variable is the number of sample cases of wine purchased by wine companies after sampling.

Summary Statistics

Get an overview of raw data structure.

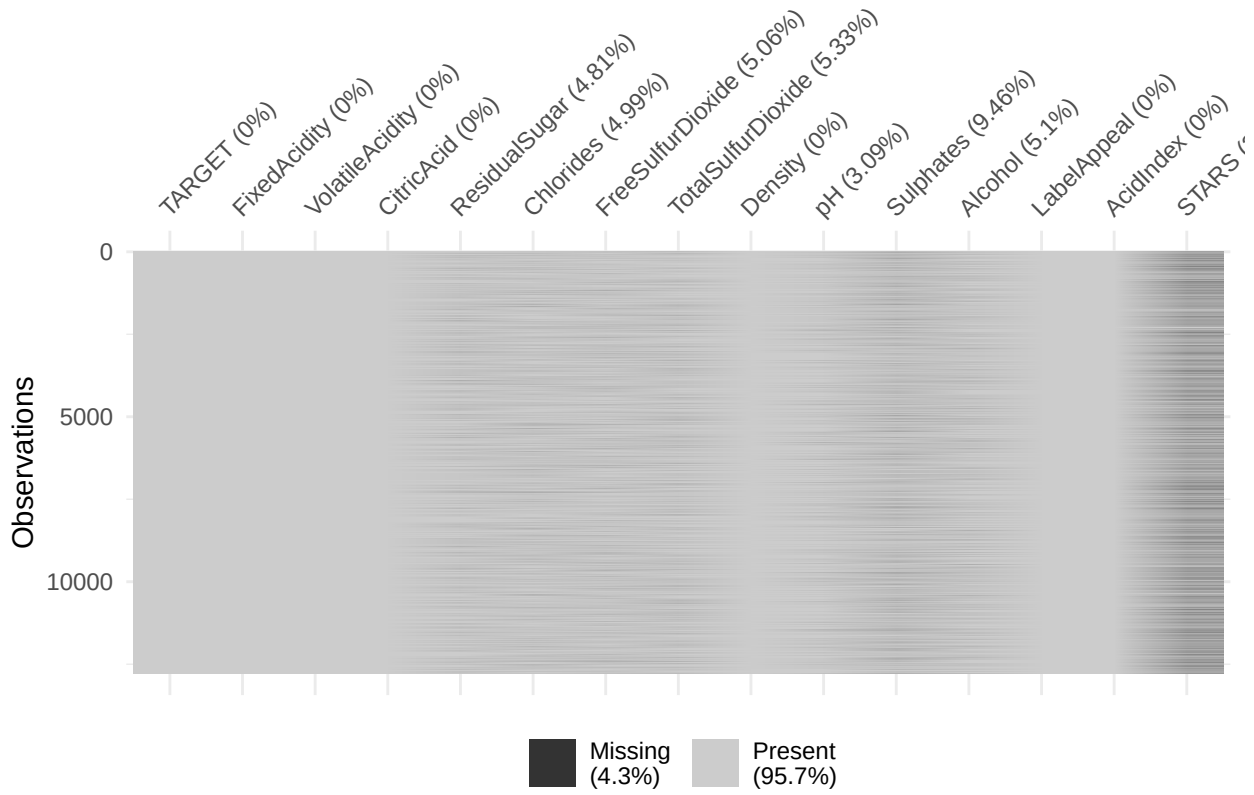
```
## Rows: 12,795
## Columns: 15
## $ TARGET             <int> 3, 3, 5, 3, 4, 0, 0, 4, 3, 6, 0, 4, 3, 7, 4, 0, ...
## $ FixedAcidity       <dbl> 3.2, 4.5, 7.1, 5.7, 8.0, 11.3, 7.7, 6.5, 14.8, 5...
## $ VolatileAcidity    <dbl> 1.160, 0.160, 2.640, 0.385, 0.330, 0.320, 0.290,...
## $ CitricAcid         <dbl> -0.98, -0.81, -0.88, 0.04, -1.26, 0.59, -0.40, 0...
```

```
## $ ResidualSugar      <dbl> 54.20, 26.10, 14.80, 18.80, 9.40, 2.20, 21.50, 1...
## $ Chlorides          <dbl> -0.567, -0.425, 0.037, -0.425, NA, 0.556, 0.060,...
## $ FreeSulfurDioxide  <dbl> NA, 15, 214, 22, -167, -37, 287, 523, -213, 62, ...
## $ TotalSulfurDioxide <dbl> 268, -327, 142, 115, 108, 15, 156, 551, NA, 180,...
## $ Density            <dbl> 0.99280, 1.02792, 0.99518, 0.99640, 0.99457, 0.9...
## $ pH                 <dbl> 3.33, 3.38, 3.12, 2.24, 3.12, 3.20, 3.49, 3.20, ...
## $ Sulphates          <dbl> -0.59, 0.70, 0.48, 1.83, 1.77, 1.29, 1.21, NA, 0...
## $ Alcohol            <dbl> 9.9, NA, 22.0, 6.2, 13.7, 15.4, 10.3, 11.6, 15.0...
## $ LabelAppeal        <int> 0, -1, -1, -1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 2, 0, ...
## $ AcidIndex          <int> 8, 7, 8, 6, 9, 11, 8, 7, 6, 8, 5, 10, 7, 8, 9, 8...
## $ STARS              <int> 2, 3, 3, 1, 2, NA, NA, 3, NA, 4, 1, 2, 2, 3, NA,...
```

All variables are numeric. `TARGET` is the response variable. The row index variable was removed. The remaining variables will be assessed for suitability as predictors in various regression models. There are 12,795 cases in the training dataset.

```
##      TARGET       FixedAcidity     VolatileAcidity     CitricAcid
##  Min.   :0.000   Min.   :-18.100   Min.   :-2.7900   Min.   :-3.2400
##  1st Qu.:2.000   1st Qu.:  5.200   1st Qu.: 0.1300   1st Qu.: 0.0300
##  Median :3.000   Median :  6.900   Median : 0.2800   Median : 0.3100
##  Mean   :3.029   Mean   :  7.076   Mean   : 0.3241   Mean   : 0.3084
##  3rd Qu.:4.000   3rd Qu.:  9.500   3rd Qu.: 0.6400   3rd Qu.: 0.5800
##  Max.   :8.000   Max.   : 34.400   Max.   : 3.6800   Max.   : 3.8600
##
##  ResidualSugar        Chlorides       FreeSulfurDioxide TotalSulfurDioxide
##  Min.   :-127.800   Min.   :-1.1710   Min.   :-555.00   Min.   :-823.0
##  1st Qu.:  -2.000   1st Qu.:-0.0310   1st Qu.:   0.00   1st Qu.:  27.0
##  Median :   3.900   Median : 0.0460   Median :  30.00   Median : 123.0
##  Mean   :   5.419   Mean   : 0.0548   Mean   :  30.85   Mean   : 120.7
##  3rd Qu.:  15.900   3rd Qu.: 0.1530   3rd Qu.:  70.00   3rd Qu.: 208.0
##  Max.   : 141.150   Max.   : 1.3510   Max.   : 623.00   Max.   :1057.0
##  NA's   :616        NA's   :638       NA's   :647       NA's   :682
##     Density             pH          Sulphates          Alcohol
##  Min.   :0.8881   Min.   :0.480   Min.   :-3.1300   Min.   :-4.70
##  1st Qu.:0.9877   1st Qu.:2.960   1st Qu.: 0.2800   1st Qu.: 9.00
##  Median :0.9945   Median :3.200   Median : 0.5000   Median :10.40
##  Mean   :0.9942   Mean   :3.208   Mean   : 0.5271   Mean   :10.49
##  3rd Qu.:1.0005   3rd Qu.:3.470   3rd Qu.: 0.8600   3rd Qu.:12.40
##  Max.   :1.0992   Max.   :6.130   Max.   : 4.2400   Max.   :26.50
##                   NA's   :395     NA's   :1210      NA's   :653
##   LabelAppeal          AcidIndex          STARS
##  Min.   :-2.000000   Min.   : 4.000   Min.   :1.000
##  1st Qu.:-1.000000   1st Qu.: 7.000   1st Qu.:1.000
##  Median : 0.000000   Median : 8.000   Median :2.000
##  Mean   :-0.009066   Mean   : 7.773   Mean   :2.042
##  3rd Qu.: 1.000000   3rd Qu.: 8.000   3rd Qu.:3.000
##  Max.   : 2.000000   Max.   :17.000   Max.   :4.000
##                                       NA's   :3359
```

Plot of missingness of dataset:



The summary statistics highlight several key concepts regarding the structure of the dataset.

1. Missing values: the STARS variable has significant amount of missing values. Other variables with missing values include Alcohol, Sulphates, pH, Sulfur Dioxide, Free Sulfur Dioxide, Sugar and Chlorides. As will be shown in later sections, the presence of missing data seems to be a contributing factor for sales. Based on the size of the dataset, it is reasonable to infer that the included wineries are diverse in terms of geography and production scale. For a myriad of reasons, this means that not all wineries will have access to laboratory analysis and ratings from an expert which may negatively impact the ability to sell.

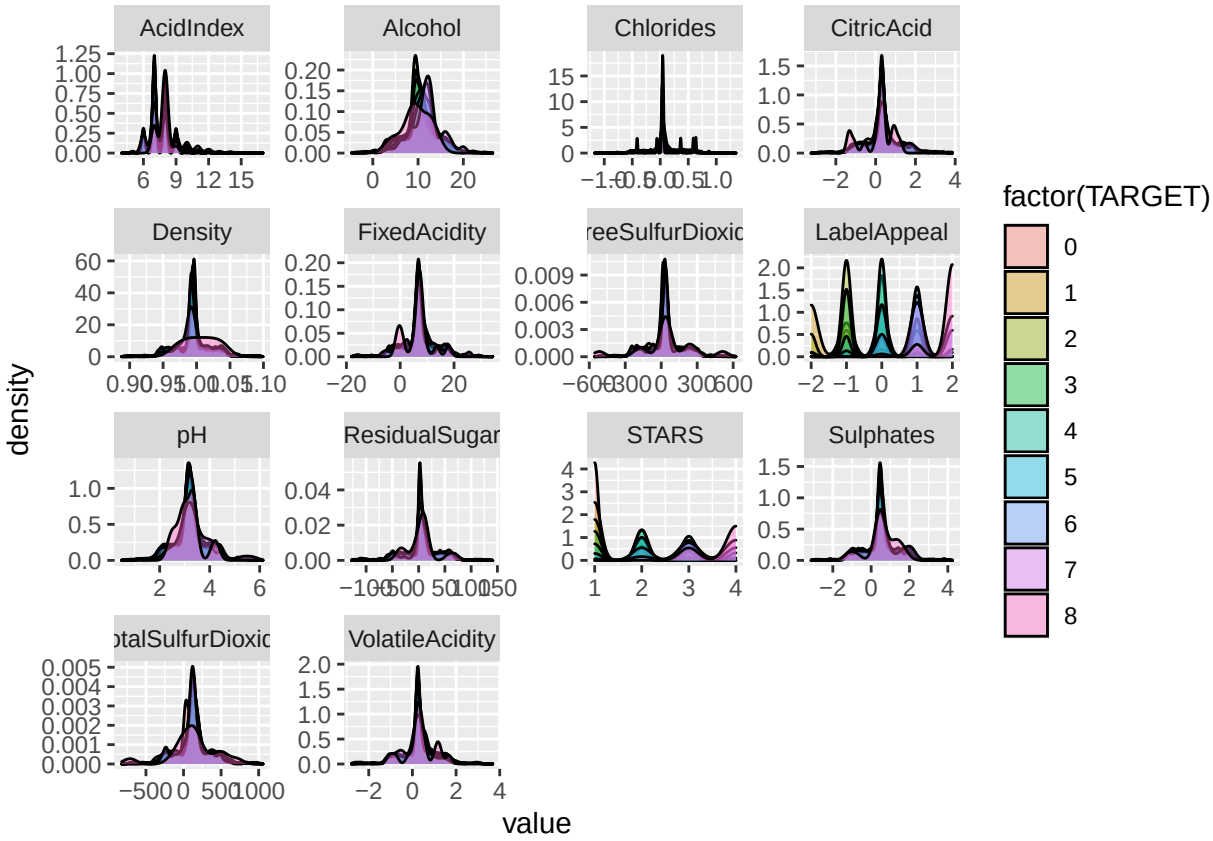
2. Negative values: Some variables contain unexpected negative values. In the Alcohol variable, for example, a negative value does not make physical sense in most commonly used units of measure (abv, proof). Since there is no information provided regarding the source of the data, or units of measurement to conduct the analysis, these values will be assumed to be correct.

Plots

A series of visualizations were used to reveal relationships between the target and predictors.

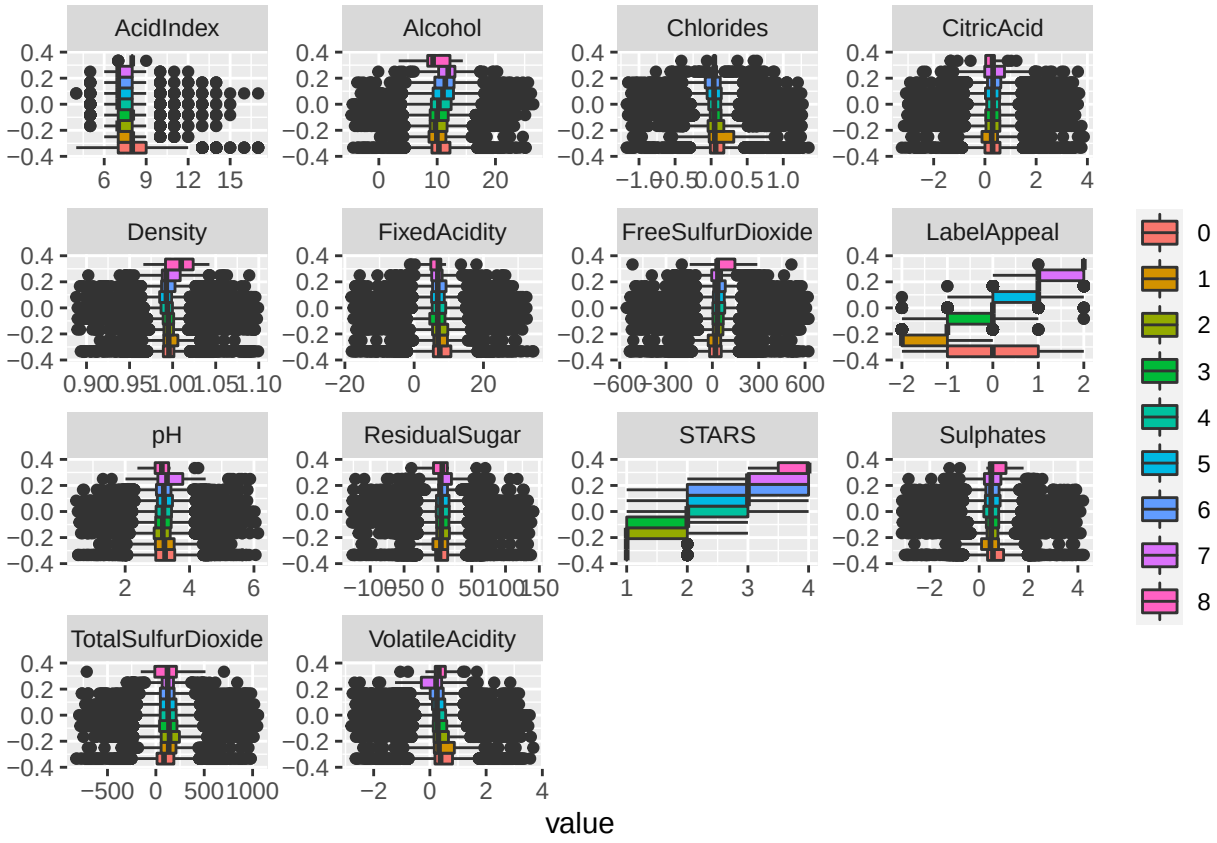
Density Plots

The variables that have the strongest relationships with the target are: `LabelAppeal` and `STARS`. These two are also the only variables that are not based on chemical analysis of wine samples.



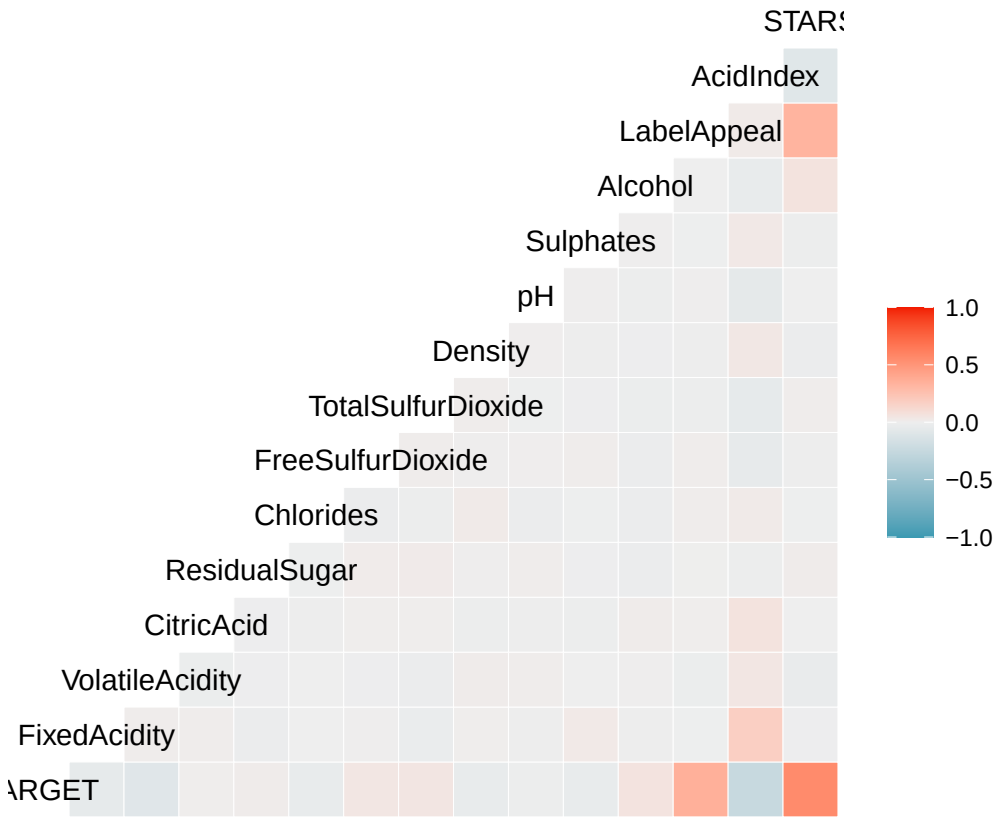
Boxplots

The dodged boxplot of each variable against the target variable highlights differences between target boxes which could mean the variable is useful for prediction. A dodged boxplot without overlapping boxes likely indicates a correlation in the value of the predictor variable to the target classes.



Correlation Matrix

Once again, there seems to be weak predictive value with the chemical analysis. No variables are highly correlated to sales, however `STARS` and `LabelAppeal` do have some correlation. In addition, `AcidIndex` has some negative correlation with sales. `LabelAppeal` has some correlation with `STARS`. Perhaps wine experts are biased towards appearance rather than wine quality.



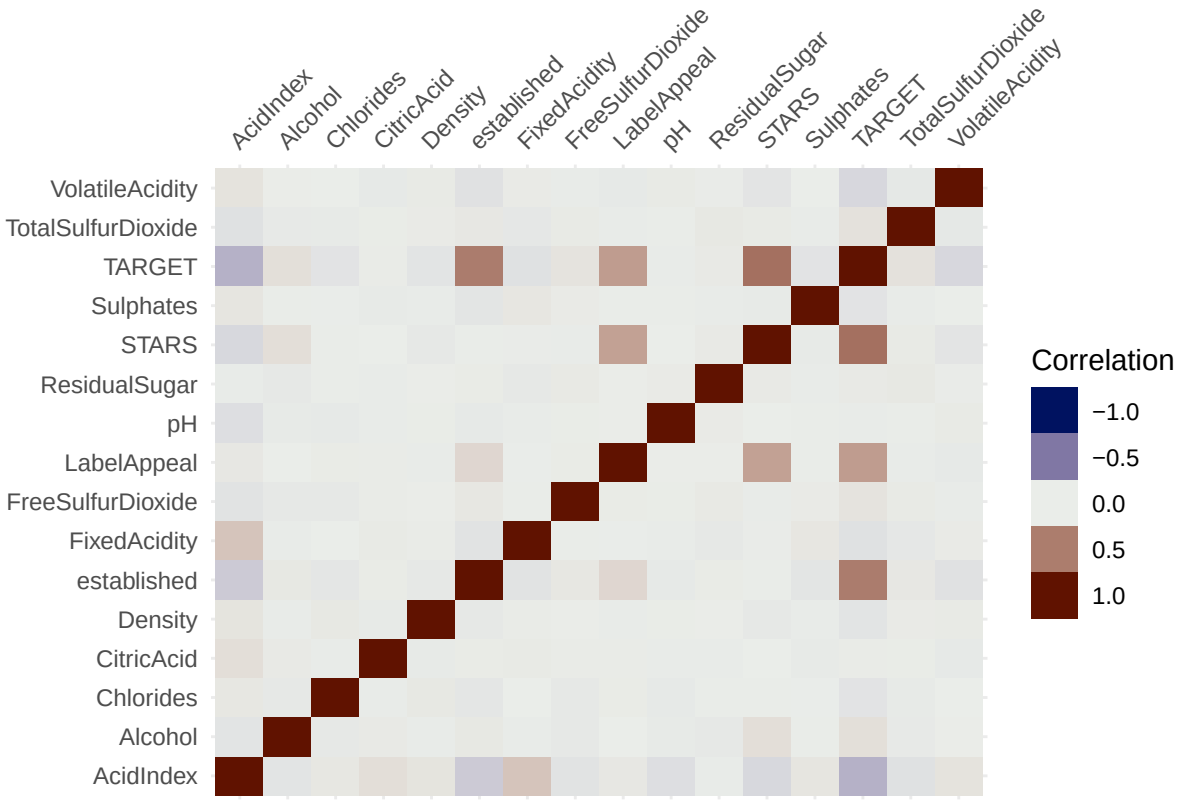
Data Preparation

Based on the data analysis done in the previous section, several steps will be taken to prepare the dataset for regression, including introduction of a new variable, handling of missing values, and datatype conversion.

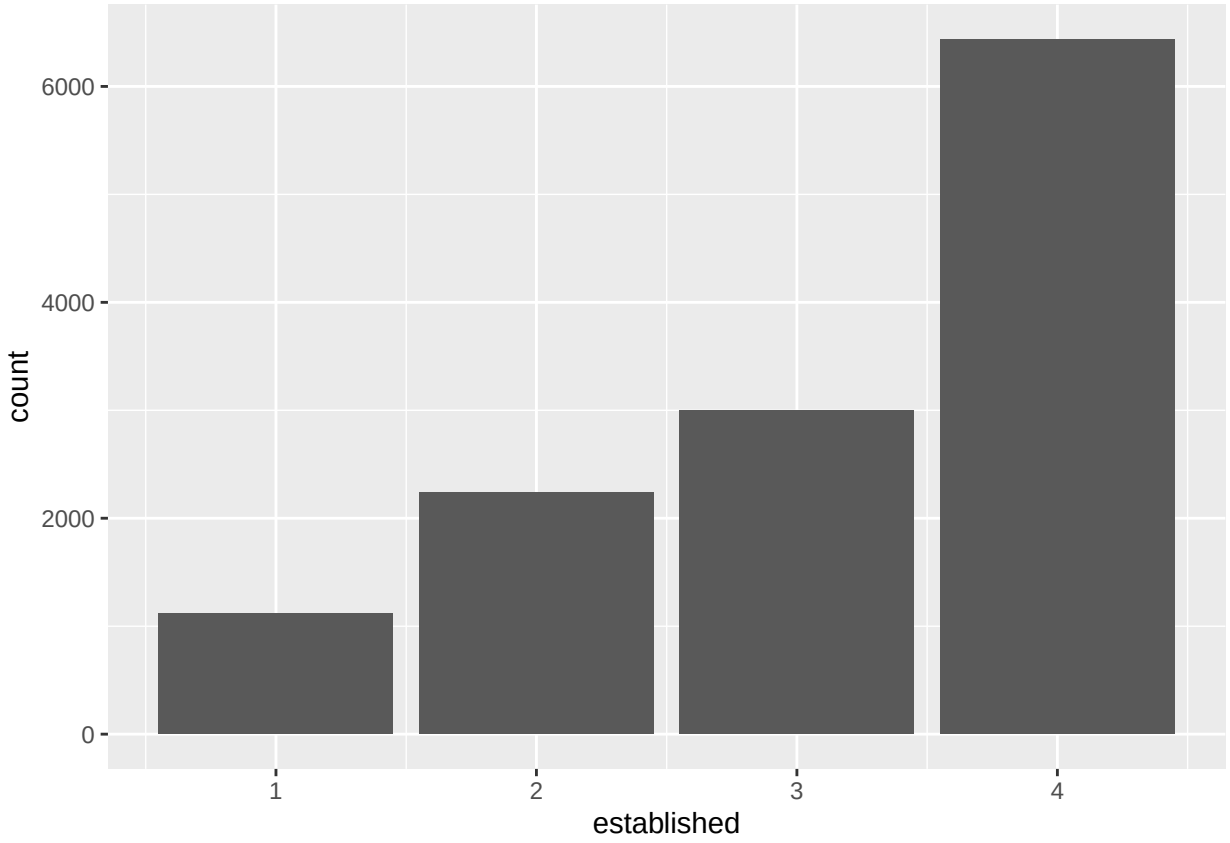
New Variable

The presence of a missing value could be influential to a buyer, causing them to not purchase cases from a supplier that cannot provide chemical analysis or expert rating. As discussed above, perhaps missing data means the wine manufacturer is not established, making the purchaser less likely to be receptive.

We notice the derived `established` variable does have a positive correlation with the target variable.


AcidIndex
Alcohol
Chlorides
CitricAcid
Density
established
FixedAcidity
FreeSulfurDioxide
LabelAppeal
pH
ResidualSugar
STARS
Sulphates
TARGET
TotalSulfurDioxide
VolatileAcidity
Correlation
−1.0
−0.5
0.0
0.5
1.0

```
## [1] 0.5030543
```



Replace Missing Values

The `STARS` and `established` variables are treated as factors. The reason for `STARS` being a factor is because the missing values may have significance to the target variable therefore should be treated as its own value.

Missing values were imputed using a predictive mean matching algorithm from the R `mice` package. The imputed data is then filled into both the training and evaluation sets.

Split into Train/Test

The provided training dataset was split into a train and test set using an 80/20 split.

Build Models

In this section, a series of models will be built and diagnostic metrics will be calculated. For each model a summary and brief analysis is provided.

Poisson Model 1: Stepwise

In the poisson model there is an assumption that the mean equals variance. The first model uses the generalized linear model family poisson and a stepwise selection algorithm to include only statistically significant coefficients in the model.

The model output shows deviance residuals are centered around 0 and are generally symmetrical around the median, which indicates a good fit. There are chemical variables included in the model which although are significant, have small coefficient values. The variables `STARS`, `established`, and `LabelAppeal` seem to be the best predictors, which is consistent with the visualizations from above. Overdispersion does not appear to be an issue with this poisson model as the value of sigma = Residual Deviance/Degrees of Freedom is near one(~0.977).

```
##
## Call:
## glm(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + Alcohol + LabelAppeal + AcidIndex +
##     STARS + established, family = poisson, data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -3.12272  -0.70764  -0.02958   0.47315   3.04915
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          5.002e-01  5.448e-02   9.183  < 2e-16 ***
## VolatileAcidity     -2.885e-02  7.282e-03  -3.962 7.44e-05 ***
## Chlorides           -3.632e-02  1.813e-02  -2.003  0.04515 *
## FreeSulfurDioxide    9.198e-05  3.833e-05   2.400  0.01641 *
## TotalSulfurDioxide   7.575e-05  2.472e-05   3.064  0.00218 **
## Alcohol              3.684e-03  1.533e-03   2.403  0.01627 *
## LabelAppeal          1.426e-01  6.877e-03  20.731  < 2e-16 ***
## AcidIndex           -7.109e-02  5.016e-03 -14.172  < 2e-16 ***
## STARS2               4.665e-01  1.516e-02  30.768  < 2e-16 ***
## STARS3               6.009e-01  1.690e-02  35.569  < 2e-16 ***
## STARS4               7.038e-01  2.364e-02  29.776  < 2e-16 ***
## established2         4.141e-02  3.835e-02   1.080  0.28020
## established3         8.969e-01  3.386e-02  26.491  < 2e-16 ***
## established4         8.772e-01  3.296e-02  26.617  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 18252  on 10237  degrees of freedom
## Residual deviance: 10002  on 10224  degrees of freedom
## AIC: 35598
##
## Number of Fisher Scoring iterations: 6
```

Poisson Model 2: Overdispersion

In Poisson Model 1, residual deviance divided by degrees of freedom gives sigma which if greater than 1 means overdispersion. This means the standard errors cannot be trusted. In the next model, overdispersion is accounted for.

```
##
## Call:
## glm(formula = TARGET ~ LabelAppeal + established + STARS, family = quasipoisson,
```

```
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1811  -0.7315  -0.0457   0.4565   3.0557
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.036887   0.029032  -1.271    0.204
## LabelAppeal   0.137146   0.006189  22.160   <2e-16 ***
## established2  0.033246   0.034596   0.961    0.337
## established3  0.921112   0.030523  30.177   <2e-16 ***
## established4  0.899293   0.029717  30.261   <2e-16 ***
## STARS2        0.484618   0.013645  35.515   <2e-16 ***
## STARS3        0.630808   0.015143  41.656   <2e-16 ***
## STARS4        0.739549   0.021198  34.888   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 0.8140267)
##
##     Null deviance: 18252  on 10237  degrees of freedom
## Residual deviance: 10265  on 10230  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 6

## The following objects are masked from test (pos = 3):
##
##     AcidIndex, Alcohol, Chlorides, CitricAcid, Density, established,
##     FixedAcidity, FreeSulfurDioxide, LabelAppeal, pH, ResidualSugar,
##     STARS, Sulphates, TARGET, TotalSulfurDioxide, VolatileAcidity
```

Poisson Model 3: Hurdle

A hurdle model is used to account for the large presence of zeroes in the target and the subsequent deviance from a true poisson distribution. The hurdle model calculates different sets of coefficients for instances where the target equals zero and for instances where the target does not equal zero. The model output shows deviance residuals once again centered around 0, but this time with a right skew. There are two sets of coefficients, the first is for the positive-count process, the second is for the zero-count process.

```
##
## Call:
## hurdle(formula = TARGET ~ LabelAppeal + established + STARS, data = train,
##     dist = "poisson")
##
## Pearson residuals:
##      Min       1Q   Median       3Q      Max
## -1.94441 -0.36802 -0.04123  0.35846  4.38541
##
## Count model coefficients (truncated poisson with log link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.976759   0.036841  26.513  < 2e-16 ***
```

```
## LabelAppeal  0.233461   0.007402  31.539  < 2e-16 ***
## established2 0.043263   0.042725   1.013    0.311
## established3 0.200785   0.037393   5.370 7.89e-08 ***
## established4 0.188087   0.036494   5.154 2.55e-07 ***
## STARS2       0.139954   0.016150   8.666  < 2e-16 ***
## STARS3       0.243697   0.017659  13.800  < 2e-16 ***
## STARS4       0.328166   0.024361  13.471  < 2e-16 ***
## Zero hurdle model coefficients (binomial with logit link):
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.36643    0.08712 -15.684   <2e-16 ***
## LabelAppeal   -0.58013    0.03945 -14.704   <2e-16 ***
## established2  -0.01008    0.10075  -0.100    0.920
## established3   2.80386    0.12010  23.345   <2e-16 ***
## established4   2.73839    0.10192  26.868   <2e-16 ***
## STARS2         2.08583    0.07929  26.306   <2e-16 ***
## STARS3        18.96904  378.97349   0.050    0.960
## STARS4        19.18841  727.30057   0.026    0.979
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 15
## Log-likelihood: -1.61e+04 on 16 Df

## [1]  0.6341575  8.5781250  1.9579545  0.7888942  0.5373721  0.6125848  1.0709571
## [8]  3.5454545 15.5714286
```

Binomial Model 1: Select Variables

Negative binomial regression can be used for over-dispersed count data. The same predictors are used here that were used in the hurdle model and quasi-poisson model.

As seen by the Residual Deviance to Degrees of Freedom ration, dispersion is effectively dealt with in this model. It also has relatively large coefficient values, especially when compared to corresponding standard errors. Once again residuals are centered around 0 and are symmetrical around the median value.

```
##
## Call:
## glm.nb(formula = TARGET ~ LabelAppeal + established + STARS,
##     data = train, init.theta = 43804.617, link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1810  -0.7315  -0.0457   0.4564   3.0555
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.03689    0.03218  -1.147    0.252
## LabelAppeal   0.13714    0.00686  19.992   <2e-16 ***
## established2  0.03325    0.03834   0.867    0.386
## established3  0.92112    0.03383  27.227   <2e-16 ***
## established4  0.89930    0.03294  27.303   <2e-16 ***
## STARS2        0.48462    0.01512  32.042   <2e-16 ***
## STARS3        0.63081    0.01679  37.582   <2e-16 ***
```

```
## STARS4        0.73956    0.02350  31.476   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(43804.62) family taken to be 1)
##
##     Null deviance: 18251  on 10237  degrees of freedom
## Residual deviance: 10265  on 10230  degrees of freedom
## AIC: 35851
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  43805
##           Std. Err.:  41298
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -35833.35

## [1] 35851.34

##  Accuracy
## 0.2827532 0.0000000 0.5576923 0.4346457 0.3969466 0.2704403

## [1] 0.8140717
```

Binomial Model 2: Expanded

In the next binomial model, we expand on the previous negative binomial model by including some of the chemical variables. In particular, two of the variables relating to acid content appeared to be the best predictors considering the output from the stepwise poisson model.

Comparing AIC scores of the two negative binomial distributions, we see some evidence that the extra variables that were added did not necessarily improve the quality of the model. Their coefficients are lower in magnitude, but do have statistically significant p-values.

```
##
## Call:
## glm.nb(formula = TARGET ~ LabelAppeal + established + STARS +
##     AcidIndex + VolatileAcidity + TotalSulfurDioxide, data = train,
##     init.theta = 44674.68544, link = log)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -3.14359  -0.71250  -0.02695   0.47246   3.06318
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         5.461e-01  5.138e-02  10.628  < 2e-16 ***
## LabelAppeal         1.426e-01  6.877e-03  20.729  < 2e-16 ***
## established2        4.111e-02  3.835e-02   1.072  0.28370
## established3        8.984e-01  3.385e-02  26.539  < 2e-16 ***
## established4        8.785e-01  3.295e-02  26.657  < 2e-16 ***
```

```
## STARS2              4.669e-01  1.516e-02  30.795  < 2e-16 ***
## STARS3              6.033e-01  1.687e-02  35.754  < 2e-16 ***
## STARS4              7.066e-01  2.359e-02  29.958  < 2e-16 ***
## AcidIndex          -7.208e-02  5.011e-03 -14.383  < 2e-16 ***
## VolatileAcidity    -2.891e-02  7.283e-03  -3.970  7.2e-05 ***
## TotalSulfurDioxide  7.585e-05  2.471e-05   3.069  0.00215 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(44674.69) family taken to be 1)
##
##     Null deviance: 18251  on 10237  degrees of freedom
## Residual deviance: 10018  on 10227  degrees of freedom
## AIC: 35610
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  44675
##           Std. Err.:  42207
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -35586.38

## [1] 35610.38

##  Accuracy
## 0.2839265 0.0000000 0.5576923 0.4062992 0.4249364 0.2830189

## [1] 0.7945887
```

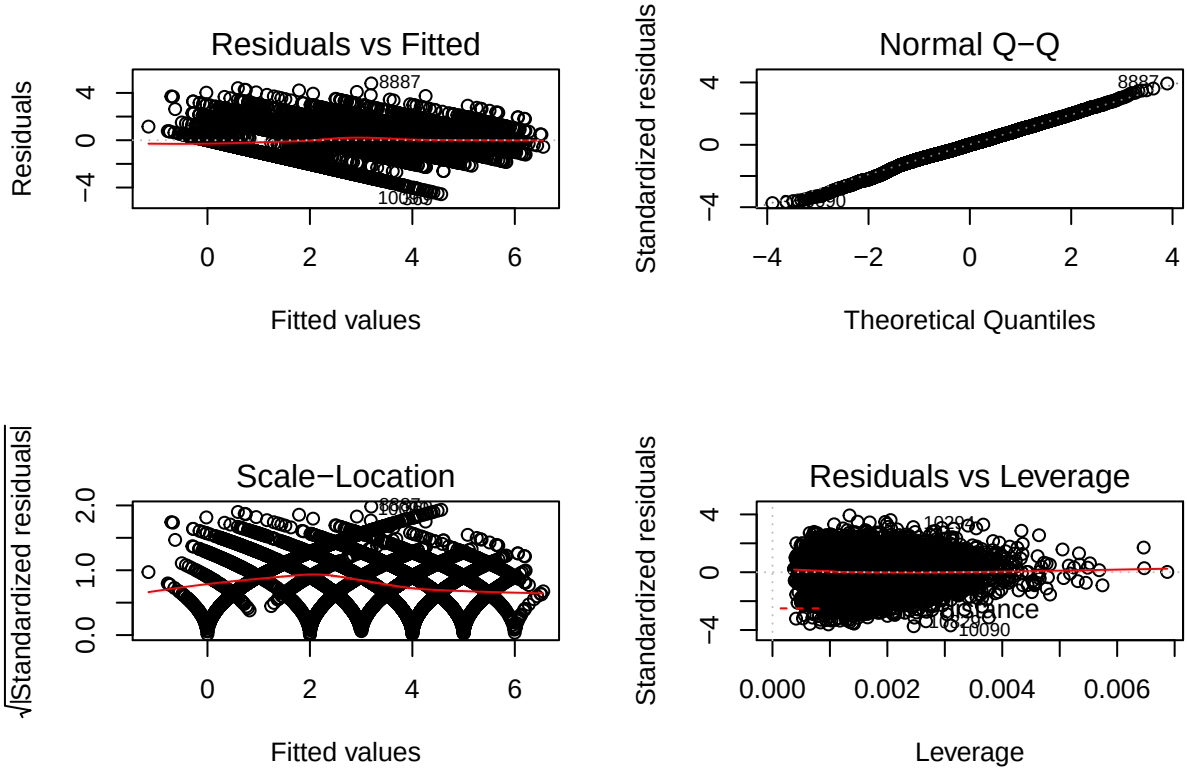
Linear Model 1: Stepwise

Next, multiple linear models are built. For the sake of comparison to generalized linear models, a stepwise model is built.

We see a similar output in the linear model compared to the poisson model. Some of the variables have a marginally more important role in the linear model, namely `Chlorides` and `Alcohol`, but once again the major predictors are not the chemical variables. Label appeal, presence of missing information, and expert opinion seem to have a greater effect. The OLS diagnostic plots suggest a valid model with residuals constant variance with normal distribution and the effect of influential outliers appears minimal. A shortfall of OLS modelling for count data is in the range of predictions (shown below). OLS has negative predictions and also does not predict well towards the high end of counts.

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + ResidualSugar + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + Density + Alcohol +
##     LabelAppeal + AcidIndex + STARS + established, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5639 -0.7654  0.0141  0.8125  4.8022
```

```
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.775e+00  4.582e-01   6.056 1.44e-09 ***
## VolatileAcidity     -9.065e-02  1.543e-02  -5.877 4.31e-09 ***
## ResidualSugar        5.118e-04  3.551e-04   1.441  0.14955
## Chlorides           -1.149e-01  3.825e-02  -3.005  0.00266 **
## FreeSulfurDioxide    2.500e-04  8.175e-05   3.058  0.00223 **
## TotalSulfurDioxide   2.072e-04  5.238e-05   3.956 7.69e-05 ***
## Density             -6.827e-01  4.530e-01  -1.507  0.13183
## Alcohol              1.397e-02  3.257e-03   4.288 1.82e-05 ***
## LabelAppeal          4.388e-01  1.425e-02  30.787  < 2e-16 ***
## AcidIndex           -1.721e-01  9.313e-03 -18.476  < 2e-16 ***
## STARS2               1.172e+00  2.904e-02  40.358  < 2e-16 ***
## STARS3               1.833e+00  3.632e-02  50.476  < 2e-16 ***
## STARS4               2.454e+00  5.961e-02  41.168  < 2e-16 ***
## established2         2.783e-02  5.025e-02   0.554  0.57972
## established3         1.761e+00  4.935e-02  35.690  < 2e-16 ***
## established4         1.681e+00  4.578e-02  36.727  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.223 on 10222 degrees of freedom
## Multiple R-squared:  0.5964, Adjusted R-squared:  0.5958
## F-statistic:  1007 on 15 and 10222 DF,  p-value: < 2.2e-16
```



```
## [1] 1.241026
```

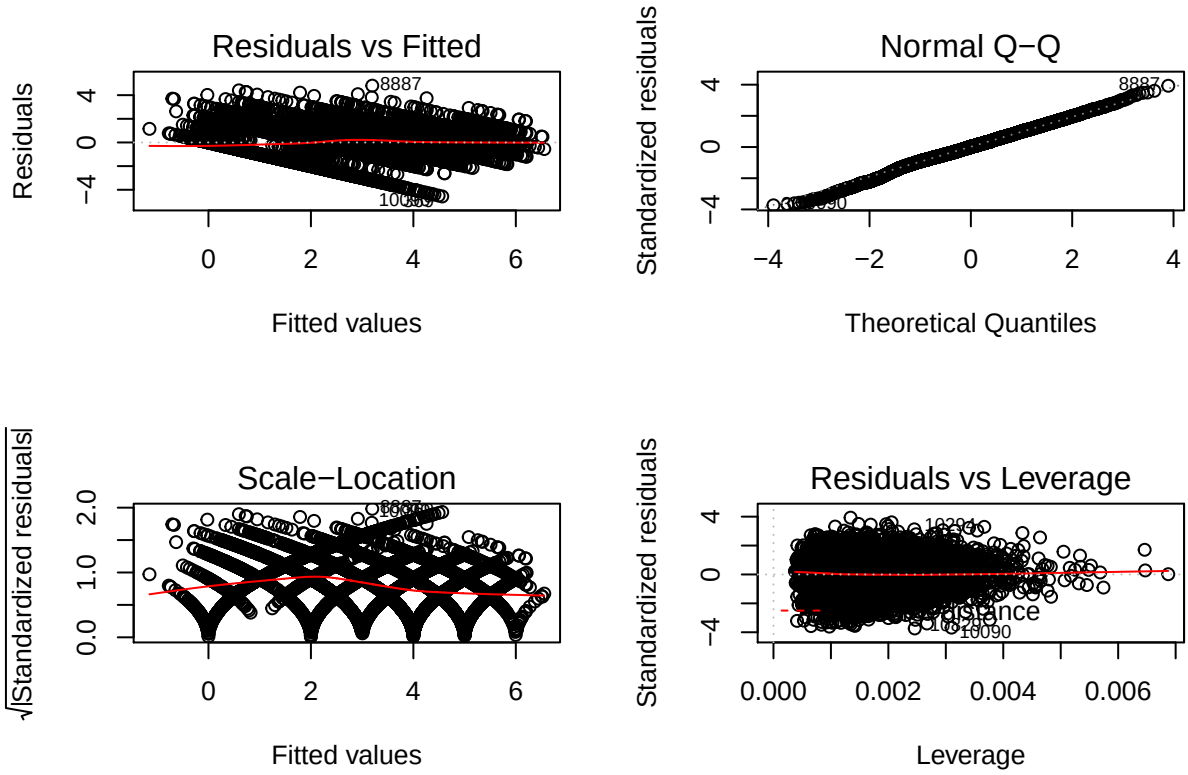
```
## [1] -0.5761142  6.2670249
```

Linear Model 2: Select Variables

The last model built is based on the non-chemical variables.

As seen in the negative binomial models, removing the chemical variables does not seem to be detrimental to the model output. R-squared values and RMSE for the two linear models are very similar. The diagnostic plots suggest this is a valid model, as the assumption of linear regression are satisfied. Overall this model is preferable to the previous OLS model since it performs nearly as well using fewer predictors.

```
##
## Call:
## lm(formula = TARGET ~ LabelAppeal + established + STARS, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6226 -0.8208  0.0359  0.8493  4.8353
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.820808   0.042924  19.123   <2e-16 ***
## LabelAppeal  0.417083   0.014519  28.727   <2e-16 ***
## established2 0.005834   0.051334   0.114     0.91
## established3 1.832672   0.050292  36.441   <2e-16 ***
## established4 1.744706   0.046670  37.384   <2e-16 ***
## STARS2       1.222964   0.029573  41.355   <2e-16 ***
## STARS3       1.920979   0.036855  52.122   <2e-16 ***
## STARS4       2.564449   0.060640  42.290   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.25 on 10230 degrees of freedom
## Multiple R-squared:  0.5781, Adjusted R-squared:  0.5779
## F-statistic:  2003 on 7 and 10230 DF,  p-value: < 2.2e-16
```




```
## [1] 1.266798
```

Select Model

The 7 models developed above are evaluated for suitability and the best model will be used to make predictions on the evaluation dataset. In the previous section, each model was briefly analyzed based on its summary output. In this section models will be compared to each other based on accuracy of predictions and will be calculated based on the testing dataset in order to remove bias from the results. Other metrics such as RMSE, R-squared, and AIC are not applicable to each of the regression methods.

Vuong tests for comparison

A Vuong test is used to compare Poisson, Hurdle Poisson, and negative binomial regression models. Of the applicable models, the hurdle model performed the best in the Vuong testing.

Test1: stepwise Poisson vs. Hurdle

```
## Vuong Non-Nested Hypothesis Test-Statistic:
## (test-statistic is asymptotically distributed N(0,1) under the
##  null that the models are indistinguishible)
## -------------------------------------------------------------
##               Vuong z-statistic             H_A    p-value
## Raw                   -37.45160 model2 > model1 < 2.22e-16
```

```
## AIC-corrected         -37.40709 model2 > model1 < 2.22e-16
## BIC-corrected         -37.24609 model2 > model1 < 2.22e-16
```

Test2: Hurdle vs binomial (select predictors)

```
## Vuong Non-Nested Hypothesis Test-Statistic:
## (test-statistic is asymptotically distributed N(0,1) under the
##  null that the models are indistinguishible)
## -------------------------------------------------------------
##                Vuong z-statistic             H_A    p-value
## Raw                    40.37590 model1 > model2 < 2.22e-16
## AIC-corrected          40.19785 model1 > model2 < 2.22e-16
## BIC-corrected          39.55389 model1 > model2 < 2.22e-16
```

Test3: Hurdle vs binomial (expanded predictors)

```
## Vuong Non-Nested Hypothesis Test-Statistic:
## (test-statistic is asymptotically distributed N(0,1) under the
##  null that the models are indistinguishible)
## -------------------------------------------------------------
##                Vuong z-statistic             H_A    p-value
## Raw                    37.62170 model1 > model2 < 2.22e-16
## AIC-corrected          37.51044 model1 > model2 < 2.22e-16
## BIC-corrected          37.10803 model1 > model2 < 2.22e-16
```

Model Selection

Based on the discussion of model outputs above and Vuong testing, the most appropriate model for the dataset is the Hurdle model:

This model is suitable for count data, accounts for high occurrence of zero-counts, and uses a small number of predictors to achieve similar performance to more complicated models. We found that chemical constituents of wine were not important to wine purchasers. Instead, labeling and expert opinion were much more valued. In addition, the absence of data seems to be important. We speculate that lack of data indicates a winery that is not established (or cannot prove that it is established), and this is a deterrent to the purchaser.

Now we are ready to write the predictions CSV file. Output of the hurdle model predictions is available in file `HW5_predictions.csv`.

Appendix

```
library(tidyverse)
library(fastDummies)
library(visdat)
library(MASS)
library(MLmetrics)
library(caret)
library(missForest)
library(mice)
library(pscl)
library(AER)
library(corrplot)
```

Data Exploration

```
training_data <- read.csv("https://raw.githubusercontent.com/willoutcault/DATA621/master/homework5/wine
eval_data <- read.csv("https://raw.githubusercontent.com/willoutcault/DATA621/master/homework5/wine-eva

training_index <- training_data[,1]
training_data <- training_data[,-1]
eval_index <- eval_data[,1]
eval_data <- eval_data[,-c(1,2)]
```

Summary Statistics

```
glimpse(training_data)
```

```
glimpse(eval_data)
```

```
summary(training_data)
```

```
summary(eval_data)
```

```
vis_miss(training_data)
```

Plots Density Plots

```
training_data %>%
  dplyr::select_if(is.numeric) %>%
  gather("attribute", "value", -TARGET) %>%
  ggplot(aes(x=value, fill=factor(TARGET)))+
    geom_density(position = 'dodge', alpha=0.4)+
    facet_wrap(~attribute, scales="free")
```

Boxplots

```
training_data %>%
  dplyr::select_if(is.numeric) %>%
  gather("attribute", "value", -TARGET) %>%
  ggplot(aes(x=value, fill=factor(TARGET))) +
    geom_boxplot(position = 'dodge') +
    facet_wrap(~attribute, scales="free") +
    theme(legend.title=element_blank())
```

Correlation Matrix

```
library(GGally)
ggcorr(training_data)
```

Data Preparation

New Variable

```
has_NA <- colnames(training_data[apply(training_data, 2, anyNA)])
in_testing <- !complete.cases(training_data[,has_NA[-length(has_NA)]])
in_testing <- ifelse(in_testing==T, 1, 0)
training_data$established <- ifelse(complete.cases(training_data), 4, 3)
training_data$established <- ifelse(in_testing==T & !is.na(training_data$STARS), 3, training_data$establ
training_data$established <- ifelse(in_testing==F & is.na(training_data$STARS), 2, training_data$establi
training_data$established <- ifelse(in_testing==T & is.na(training_data$STARS), 1, training_data$establi

# Create Variable for Eval Data
in_testing <- !complete.cases(eval_data[,has_NA[-length(has_NA)]])
in_testing <- ifelse(in_testing==T, 1, 0)
eval_data$established <- ifelse(complete.cases(eval_data), 4, 3)
eval_data$established <- ifelse(in_testing==T & !is.na(eval_data$STARS), 3, eval_data$established)
eval_data$established <- ifelse(in_testing==F & is.na(eval_data$STARS), 2, eval_data$established)
eval_data$established <- ifelse(in_testing==T & is.na(eval_data$STARS), 1, eval_data$established)

# Visualize Training Data
vis_cor(training_data)
cor(training_data$TARGET,training_data$established)
training_data %>% ggplot(aes(x=established, fill=TARGET))+
  geom_bar()
```

Replace Missing Values

```
training_data$STARS <- as.factor(training_data$STARS)
eval_data$STARS <- as.factor(eval_data$STARS)

training_data$established <- as.factor(training_data$established)
eval_data$established <- as.factor(eval_data$established)

temp_train_data <- mice(training_data,m=5,meth="pmm",maxit=10,seed=500,print=F,
                        defaultMethod = c("pmm", "logreg", "polyreg", "polr"))
temp_eval_data <- mice(eval_data,m=5,meth="pmm",maxit=10,seed=500,print=F,
                       defaultMethod = c("pmm", "logreg", "polyreg", "polr"),)
```

```
clean_train_data <- complete(temp_train_data)
clean_eval_data <- complete(temp_eval_data)
```

Split into Train/Test

```
set.seed(123)
trainIndex <-createDataPartition(clean_train_data$TARGET, p = 0.8,list = FALSE,times = 1)
train <- clean_train_data[trainIndex,]
test <- clean_train_data[-trainIndex,]
```

Build Models Poisson Model 1: Stepwise

```
model1 = glm(TARGET ~ ., data = train, family = poisson) %>% stepAIC(trace=F, direction ='both')
summary(model1)

mu<-predict(model1, type = "response")
```

```r
# calculate AIC
mod1AIC <- model1$aic

## use the test data set to make predicts and calculate metrics from the confusion matrix
mod1.predict.probs <- predict.glm(model1, type="response", newdata=test)
#glm_predict.full <- ifelse(glm_full.probs > 0.5, '1','0')
attach(test)
#table(glm_predict.full, test$TARGET_FLAG)

mod1.predict.preds <- round(mod1.predict.probs)

# now can use the caret function
cm.var <- caret::confusionMatrix(factor(mod1.predict.preds), factor(test$TARGET), positive='1')
#cm.var$table

# print metrics
mod1.CMmetrics <- c(cm.var$overall[c(1)], cm.var$byClass[c(1,2,5,6,7)])

# Dispersion Statistic
E2 <- resid(model1, type = "pearson")
N  <- nrow(train)
p  <- length(coef(model1)) + 1 # '+1' is due to theta
mod1.dispersion <- dispesion <-sum(E2^2) / (N - p)
```

Poisson Model 2: Overdispersion

```r
model2 = glm(TARGET ~ LabelAppeal + established + STARS, data = train, family = quasipoisson)
summary(model2)

# calculate AIC
mod2AIC <- model2$aic

## use the test data set to make predicts and calculate metrics from the confusion matrix
mod2.predict.probs <- predict.glm(model2, type="response", newdata=test)
#glm_predict.full <- ifelse(glm_full.probs > 0.5, '1','0')
attach(test)
#table(glm_predict.full, test$TARGET_FLAG)

mod2.predict.preds <- round(mod2.predict.probs)

# now can use the caret function
cm.var <- caret::confusionMatrix(factor(mod2.predict.preds), factor(test$TARGET), positive='1')
#cm.var$table

# print metrics
mod2.CMmetrics <- c(cm.var$overall[c(1)], cm.var$byClass[c(1,2,5,6,7)])

# Dispersion Statistic
E2 <- resid(model2, type = "pearson")
N  <- nrow(train)
p  <- length(coef(model2)) + 1 # '+1' is due to theta
mod2.dispersion <- dispesion <-sum(E2^2) / (N - p)
```

Poisson Model 3: Hurdle

```
# Hurdle regression
library(pscl)
model3 <- hurdle(TARGET ~ LabelAppeal + established + STARS, data=train, dist="poisson")
summary(model3)

# predict expected mean count
mu<-predict(model3, type = "response")
# sum the probabilities of a 0 count for each mean
acc <- NA
for(i in 1:9){
  acc[i] <- round(sum(dpois(x=(i-1),lambda=mu)))/sum(train$TARGET == (i-1))
}
acc



# calculate AIC
mod3AIC <- model3$aic

## use the test data set to make predicts and calculate metrics from the confusion matrix
mod3.predict.probs <- predict(model3, type="response", newdata=test)
#glm_predict.full <- ifelse(glm_full.probs > 0.5, '1','0')
attach(test)
#table(glm_predict.full, test$TARGET_FLAG)

mod3.predict.preds <- round(mod3.predict.probs)

# now can use the caret function
cm.var <- caret::confusionMatrix(factor(mod3.predict.preds), factor(test$TARGET), positive='1')
#cm.var$table

# print metrics
mod3.CMmetrics <- c(cm.var$overall[c(1)], cm.var$byClass[c(1,2,5,6,7)])

# Dispersion Statistic
E2 <- resid(model3, type = "pearson")
N  <- nrow(train)
p  <- length(coef(model3)) + 1 # '+1' is due to theta
mod3.dispersion <- dispesion <-sum(E2^2) / (N - p)
```

Binomial Model 1: Select Variables

```
library(MASS)
model4 <- glm.nb(TARGET ~ LabelAppeal + established + STARS, data=train)
summary(model4)

## AIC
model4$aic

## use the test data set to make predicts and calculate metrics from the confusion matrix
mod4.predict.probs <- predict.glm(model4, type="response", newdata=test)
#glm_predict.full <- ifelse(glm_full.probs > 0.5, '1','0')
```

```
attach(test)
#table(glm_predict.full, test$TARGET_FLAG)

mod4.predict.preds <- round(mod4.predict.probs)

# now can use the caret function
cm4 <- caret::confusionMatrix(factor(mod4.predict.preds), factor(test$TARGET), positive='1')
# cm4$table

# print metrics
c(cm4$overall[c(1)], cm4$byClass[c(1,2,5,6,7)])

# Dispersion Statistic
E2 <- resid(model4, type = "pearson")
N  <- nrow(train)
p  <- length(coef(model4)) + 1 # '+1' is due to theta
sum(E2^2) / (N - p)
```

Binomial Model 2: Expanded

```
library(MASS)
model5 <- glm.nb(TARGET ~ LabelAppeal + established + STARS + AcidIndex + VolatileAcidity + TotalSulfurl
summary(model5)

## AIC
model5$aic

## use the test data set to make predicts and calculate metrics from the confusion matrix
mod5.predict.probs <- predict.glm(model5, type="response", newdata=test)
#glm_predict.full <- ifelse(glm_full.probs > 0.5, '1','0')
attach(test)
#table(glm_predict.full, test$TARGET_FLAG)

mod5.predict.preds <- round(mod5.predict.probs)

# now can use the caret function
cm5 <- caret::confusionMatrix(factor(mod5.predict.preds), factor(test$TARGET), positive='1')
#cm5$table

# print metrics
c(cm5$overall[c(1)], cm5$byClass[c(1,2,5,6,7)])

# Dispersion Statistic
E2 <- resid(model5, type = "pearson")
N  <- nrow(train)
p  <- length(coef(model5)) + 1 # '+1' is due to theta
sum(E2^2) / (N - p)
```

Linear Model 1: Stepwise

```
library(ModelMetrics)
model6 <- lm(TARGET ~ ., data = train) %>% stepAIC(direction = "both", trace=FALSE)
summary(model6)
```

```
par(mfrow = c(2, 2))
plot(model6)

#calculate RMSE
predictions <- predict.lm(model6, newdata = test[,-1])
rmse(test[,1], predictions)

# show range of predictions
range(predictions)
```

Linear Model 2: Select Variables

```
model7 <- lm(TARGET ~ LabelAppeal + established + STARS, data = train)
summary(model7)

par(mfrow = c(2, 2))
plot(model6)

#calculate RMSE
predictions <- predict.lm(model7, newdata = test[,-1])
rmse(test[,1], predictions)
```

Select Model Vuong tests for comparison

Test1: stepwise Poisson vs. Hurdle

```
vuong(model1, model3)
```

Test2: Hurdle vs binomial(select predictors)

```
vuong(model3, model4)
```

Test3: Hurdle vs binomial(expanded predictors)

```
vuong(model3, model5)
```

Model Selection `HW5_predictions.csv`.

```
pred <- predict(model3, newdata = clean_eval_data, type = "response")
pred <- round(pred)
predictions <- cbind(Prediction=pred, clean_eval_data)
write.csv(predictions, "HW5_predict.csv")
```