Wine Purchase Regression

Brian Weinfeld

CUNY - SPS

Abstract

I have been tasked with creating 6 regressions in an effort to predict the number of cases of wine purchased based on a number of predictors about the wine. The response variable is a count of the number of cases of wine purchased. There are 12795 observations, 14 predictors and 1 response variable.

I will begin by analysizing the initial raw data and then perform 2 multiple linear regressions, 2 poisson regression and 2 negative binomial regressions. From there I will select the best model to deploy.

Data Preparation

The raw data was read in and after an initial inspection only a few small modification needed to be made. I removed the INDEX column as it was unnecessary and modified LabelAppeal and STARS to be factors as indicated by their descriptions. The rest of the predictors are continuous or discrete.

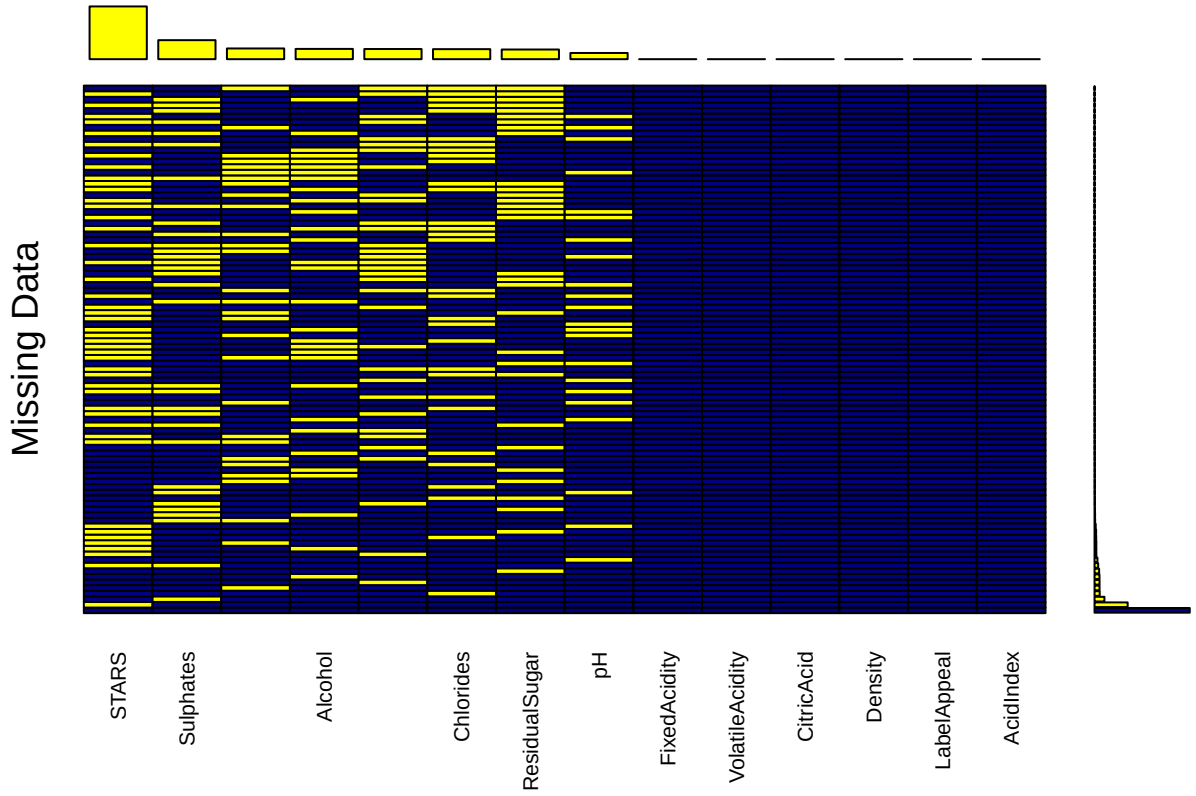
From there I explored the number of missing values in the data set.

```
wine %>%
  map_dbl(~sum(is.na(.))/nrow(wine)) %>%
  kable()
```

	x
TARGET	0.0000000
FixedAcidity	0.0000000
VolatileAcidity	0.0000000
CitricAcid	0.0000000
ResidualSugar	0.0481438
Chlorides	0.0498632
FreeSulfurDioxide	0.0505666
TotalSulfurDioxide	0.0533021
Density	0.0000000
pH	0.0308714
Sulphates	0.0945682
Alcohol	0.0510356
LabelAppeal	0.0000000
AcidIndex	0.0000000
STARS	0.2625244

There are a number of missing values in the data set but most of them comprise a relatively small proportion of the total observations. This should not be problematic to deal with. However, I am missing more than a quarter of all the STARS rankings. This is a sizeable amount of data. My first intuition is to remove this predictor but there is reason to believe that it will be highly significant as a predictor. There is strong evidence to suggest that people are not all that knowlege about wine purchases and will largely base their decisions on arguments from authority. Therefore, an external ranking system seems important. There is also another problem exhibited by the `aggr` plot

```
VIM::aggr(wine[, -1], col=c('navyblue', 'yellow'),
          numbers=TRUE, sortVars=TRUE,
          labels=names(wine[, -1]), cex.axis=.7,
          gap=3, ylab=c('Missing Data', 'Pattern'), combined=TRUE)
```




```
##
##  Variables sorted by number of missings:
##            Variable Count
##               STARS  3359
##           Sulphates  1210
##  TotalSulfurDioxide   682
##             Alcohol   653
##   FreeSulfurDioxide   647
##          Chlorides   638
##       ResidualSugar   616
##                  pH   395
##        FixedAcidity     0
##     VolatileAcidity     0
##          CitricAcid     0
##             Density     0
##         LabelAppeal     0
##           AcidIndex     0
```

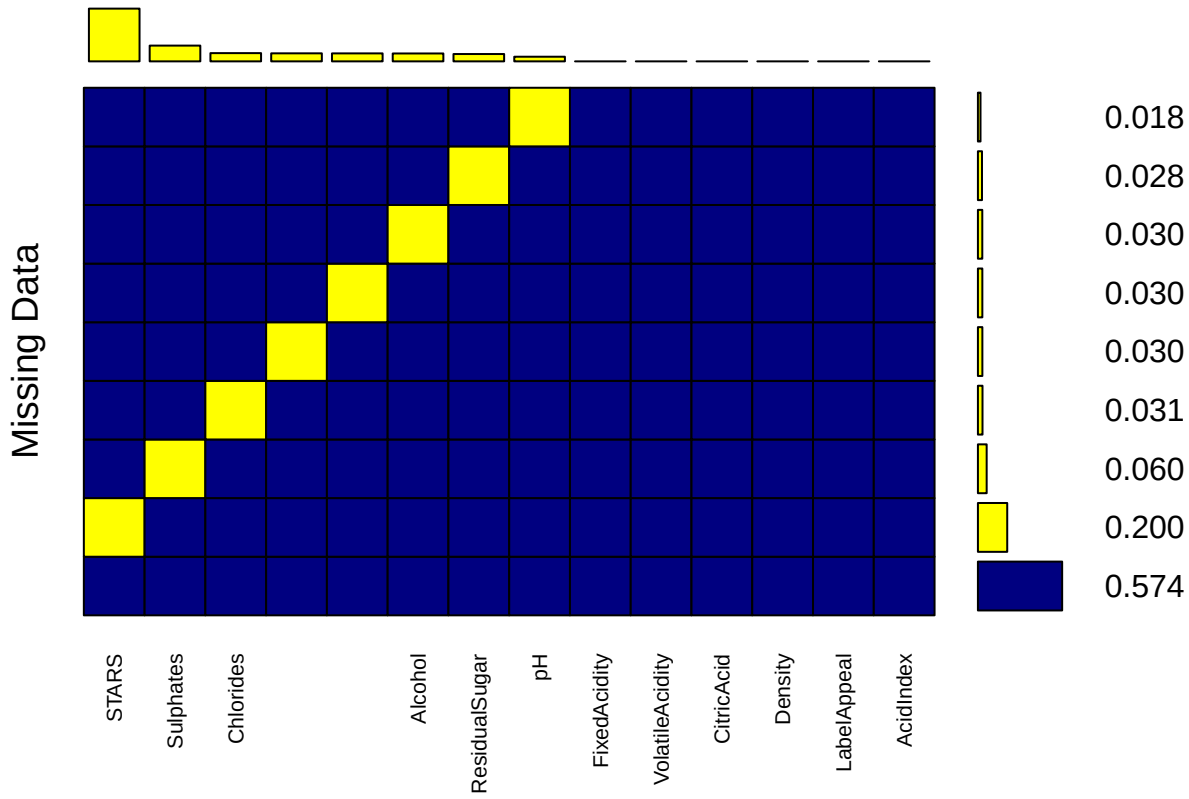
The above plot highlights all the missing values. As expected the two largest groups include the data with nothing missing followed by data only missing STARS. The problem, however, are most of the other observations. Although any given predictor is missing fewer than 10% of it's data, those missing observations are clumped together. That is, there are many observations that are missing multiple pieces of data. This is problematic as I do not feel comfortable imputed nearly half of the predictors for a wine. Imputation is powerful, but it is not magical.

As a result I decide to remove observations missing 2 or more predictors and impute the rest of the data. The choice for cutoff is somewhat arbitrary but I feel as if it gives a good balance of keeping as many observations

as possible without imputing too much missing data. The proportion of observations missing 2 or more predictors is also a small proportion of the total number of observations so I will not be losing too much valuable data.

```
temp.wine <- wine %>%
  filter(rowSums(is.na(wine)) <= 1)

VIM::aggr(temp.wine[, -1], col=c('navyblue', 'yellow'),
          numbers=TRUE, sortVars=TRUE,
          labels=names(temp.wine[, -1]), cex.axis=.7,
          gap=3, ylab=c('Missing Data', 'Pattern'), combined=TRUE)
```




```
##
##  Variables sorted by number of missings:
##            Variable Count
##               STARS  2239
##           Sulphates   669
##           Chlorides   350
##  TotalSulfurDioxide   341
##   FreeSulfurDioxide   338
##             Alcohol   335
##       ResidualSugar   311
##                  pH   197
##        FixedAcidity     0
##     VolatileAcidity     0
##          CitricAcid     0
##             Density     0
```

```
##          LabelAppeal     0
##            AcidIndex     0
```

With the data missing multiple predictors removed, I am ready to impute the data. In total, only about 10% of the observations were removed.

```
set.seed(123)
imputed.data <- mice::mice(temp.wine[, -1], m=5, maxit=50, method='pmm', seed=500, printFlag=FALSE)
wine.complete <- cbind(temp.wine[, 1], complete(imputed.data, 1))
```

From there I create a training and testing partition to test my various models.

```
set.seed(1)
part <- caret::createDataPartition(wine.complete$TARGET, p=0.8, list=FALSE)
wine.training <- wine.complete %>%
  filter(row_number() %in% part)
wine.testing <- wine.complete %>%
  filter(!row_number() %in% part)
```

Data Exploration

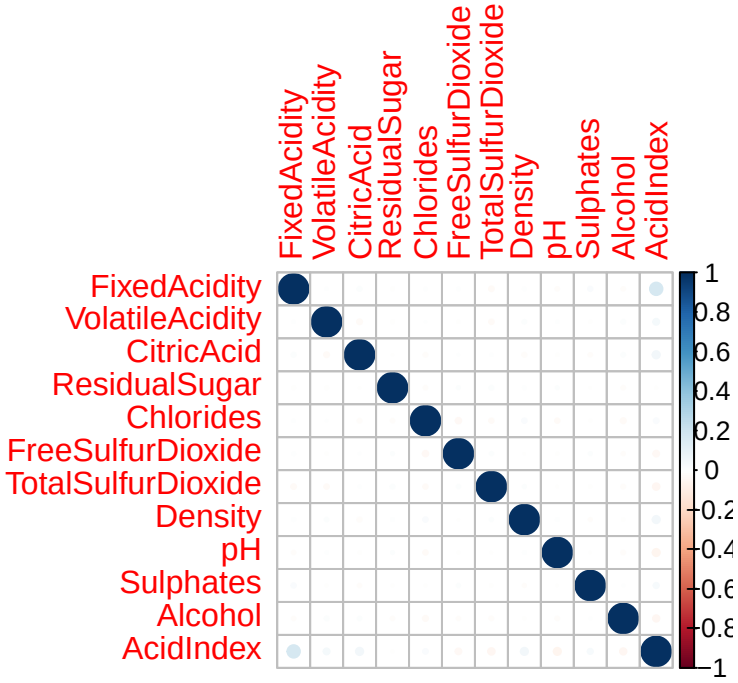
With the data imputed and seperated, I am ready to begin my data exploration. The first plot demonstrates the odd distribution of the response variable. A poisson distribution rquires an equal mean and variance and this data does not have that. The mean is 3.18 while the variance is about 3.5. In fact, the disproportionately high number of 0s indicate that most wines are not purchased, however, if the wine is purchased it is for an average of 4 cases. That is, there is a binomial question of 0 cases vs. 1 or more cases and a poission question of how many cases, given that at least 1 case is purchased.

We will keep this in mind during the model analysis.

```
ggplot(wine.training, aes(TARGET)) +
  geom_bar() +
  labs(x='Cases Purchased',
       y='Frequency',
       title='Distribution of Cases Purchased')
```

Correlation between the predictors does not appear to be an issue so it is unlikely that will need to addressed in our models.

```
corrplot(cor(wine.training[, c(-1, -13, -15)]), method='circle')
```



Multiple Linear Regression

In general, when working with count data a poisson regression is prefered. However, if the count is large enough a multiple linear regression is a valid substitution. Given that the TARGET count in this case only ranges from 0 to 8 it seems unlikely that a multiple linear regression will be sufficient. It may be useful though to compare these models to the other regressions.

Model 1

The first model is be created by initially supplying all the predictors and then using `stepAIC` to select the needed predictors

```
lm.1 <- lm(TARGET ~ ., data=wine.training)
MASS::stepAIC(lm.1, trace=0)
```

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + Density + pH + Sulphates + Alcohol +
##     LabelAppeal + AcidIndex + STARS, data = wine.training)
##
## Coefficients:
##         (Intercept)     VolatileAcidity           Chlorides
```

```
##          4.5637412          -0.1383355          -0.2171212
##  FreeSulfurDioxide  TotalSulfurDioxide             Density
##          0.0003472           0.0003122          -1.0101945
##                 pH           Sulphates             Alcohol
##         -0.0613021          -0.0342463           0.0205445
##      LabelAppeal-1        LabelAppeal0        LabelAppeal1
##          0.6344834           1.2628567           1.7915351
##       LabelAppeal2           AcidIndex              STARS2
##          2.5005038          -0.2944674           0.7759056
##             STARS3              STARS4
##          1.1710190           1.9174914
```

```
lm.1 <- update(lm.1, . ~ . -FixedAcidity -CitricAcid -ResidualSugar)
summary(lm.1)
```

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + Density + pH + Sulphates + Alcohol +
##     LabelAppeal + AcidIndex + STARS, data = wine.training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2799 -0.5789  0.3297  1.0369  4.9257
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         4.564e+00  6.362e-01   7.173 7.93e-13 ***
## VolatileAcidity    -1.383e-01  2.103e-02  -6.578 5.05e-11 ***
## Chlorides          -2.171e-01  5.228e-02  -4.153 3.32e-05 ***
## FreeSulfurDioxide   3.472e-04  1.113e-04   3.119  0.00182 **
## TotalSulfurDioxide  3.122e-04  7.183e-05   4.347 1.40e-05 ***
## Density            -1.010e+00  6.236e-01  -1.620  0.10529
## pH                 -6.130e-02  2.431e-02  -2.521  0.01171 *
## Sulphates          -3.425e-02  1.789e-02  -1.914  0.05561 .
## Alcohol             2.054e-02  4.448e-03   4.618 3.92e-06 ***
## LabelAppeal-1       6.345e-01  9.394e-02   6.754 1.52e-11 ***
## LabelAppeal0        1.263e+00  9.178e-02  13.760  < 2e-16 ***
## LabelAppeal1        1.792e+00  9.560e-02  18.740  < 2e-16 ***
## LabelAppeal2        2.501e+00  1.246e-01  20.067  < 2e-16 ***
## AcidIndex          -2.945e-01  1.295e-02 -22.736  < 2e-16 ***
## STARS2              7.759e-01  3.988e-02  19.456  < 2e-16 ***
## STARS3              1.171e+00  4.696e-02  24.937  < 2e-16 ***
## STARS4              1.917e+00  7.755e-02  24.724  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.567 on 8957 degrees of freedom
## Multiple R-squared:  0.2914, Adjusted R-squared:  0.2902
## F-statistic: 230.2 on 16 and 8957 DF,  p-value: < 2.2e-16
```

The diagnostics [SEE APPENDIX] have many causes for concern. The most immediately obvious issue is the heteroscedasticity seen in the variance and the lack of a normal fit on the qqplot. Transformational tools are problematic as well due to the fact that the response variable has 0s.

Model 2

For the 2nd model I will use BIC as a predictor selection method instead of AIC. BIC has a stronger penalty for predictors. This may help create a more simple and interpretable model. [SEE APPENDIX]

```
lm.2 <- lm(TARGET ~ ., wine.training)
stepAIC(lm.2, k=log(nrow(wine.training)), trace=0)
```

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + Alcohol + LabelAppeal + AcidIndex +
##     STARS, data = wine.training)
##
## Coefficients:
##        (Intercept)     VolatileAcidity           Chlorides
##          3.3376304          -0.1389540          -0.2157377
##  FreeSulfurDioxide  TotalSulfurDioxide             Alcohol
##          0.0003467           0.0003120           0.0207433
##      LabelAppeal-1        LabelAppeal0        LabelAppeal1
##          0.6400352           1.2683793           1.7979815
##       LabelAppeal2           AcidIndex              STARS2
##          2.4990924          -0.2944854           0.7757298
##             STARS3              STARS4
##          1.1734369           1.9185703
```

```
lm.2 <- update(lm.2, . ~ . -FixedAcidity -ResidualSugar -Density -pH -Sulphates, wine.training)
summary(lm.2)
```

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + CitricAcid + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + Alcohol + LabelAppeal +
##     AcidIndex + STARS, data = wine.training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2140 -0.5835  0.3290  1.0296  4.9500
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          3.340e+00  1.442e-01  23.165  < 2e-16 ***
## VolatileAcidity     -1.382e-01  2.105e-02  -6.565 5.51e-11 ***
## CitricAcid           2.400e-02  1.933e-02   1.241   0.2145
## Chlorides           -2.145e-01  5.229e-02  -4.103 4.12e-05 ***
## FreeSulfurDioxide    3.460e-04  1.114e-04   3.107   0.0019 **
## TotalSulfurDioxide   3.113e-04  7.186e-05   4.332 1.49e-05 ***
## Alcohol              2.064e-02  4.450e-03   4.638 3.57e-06 ***
## LabelAppeal-1        6.405e-01  9.397e-02   6.816 9.98e-12 ***
## LabelAppeal0         1.267e+00  9.180e-02  13.807  < 2e-16 ***
## LabelAppeal1         1.797e+00  9.563e-02  18.795  < 2e-16 ***
## LabelAppeal2         2.499e+00  1.247e-01  20.045  < 2e-16 ***
## AcidIndex           -2.955e-01  1.293e-02 -22.848  < 2e-16 ***
## STARS2               7.756e-01  3.990e-02  19.438  < 2e-16 ***
## STARS3               1.174e+00  4.697e-02  24.983  < 2e-16 ***
```

```
## STARS4              1.917e+00  7.758e-02  24.715  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.568 on 8959 degrees of freedom
## Multiple R-squared:  0.2905, Adjusted R-squared:  0.2894
## F-statistic: 262.1 on 14 and 8959 DF,  p-value: < 2.2e-16
```

```
anova(lm.1, lm.2, test='Chisq')
```

Res.Df	RSS	Df	Sum of Sq	Pr(>Chi)
8957	21999.57	NA	NA	NA
8959	22026.90	-2	-27.33554	0.0038305

There appears to be a statistically significant difference between the two models. The BIC selection method may have left the model with nearly all significant predictors, but it apparently performs statistically worse than the first model. This may indicate overfitting on the first model or that the removed predictors had some predictive value when taken together.

Poisson Regression

Model 3

Poisson regressions are best for count data like the TARGET response variable in this data set. However, there is a limitation to the poisson regression in that the mean and variance are equal. There is the possibility for overdispersion in real world data. I will need to examine that in the models.

For model 3 I used a poisson regression and selected the predictors using lasso. [SEE APPENDIX].

```
lm.3 <- glm(TARGET ~ . -FixedAcidity, wine.training, family=poisson)
summary(lm.3)
```

```
##
## Call:
## glm(formula = TARGET ~ . - FixedAcidity, family = poisson, data = wine.training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7545  -0.3557   0.1783   0.5746   2.4878
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          1.450e+00  2.306e-01   6.289 3.20e-10 ***
## VolatileAcidity     -4.323e-02  7.516e-03  -5.751 8.88e-09 ***
## CitricAcid           7.332e-03  6.884e-03   1.065 0.286825
## ResidualSugar        9.176e-05  1.749e-04   0.525 0.599781
## Chlorides           -6.491e-02  1.879e-02  -3.455 0.000551 ***
## FreeSulfurDioxide    1.075e-04  3.981e-05   2.701 0.006911 **
## TotalSulfurDioxide   9.976e-05  2.576e-05   3.873 0.000108 ***
## Density             -3.068e-01  2.235e-01  -1.373 0.169828
## pH                  -2.110e-02  8.718e-03  -2.420 0.015499 *
## Sulphates           -1.111e-02  6.390e-03  -1.739 0.082102 .
```

```
## Alcohol             5.875e-03  1.596e-03   3.681 0.000232 ***
## LabelAppeal-1       3.477e-01  4.669e-02   7.445 9.67e-14 ***
## LabelAppeal0        5.798e-01  4.563e-02  12.706  < 2e-16 ***
## LabelAppeal1        7.244e-01  4.635e-02  15.627  < 2e-16 ***
## LabelAppeal2        8.801e-01  5.171e-02  17.020  < 2e-16 ***
## AcidIndex          -1.043e-01  5.181e-03 -20.139  < 2e-16 ***
## STARS2              2.888e-01  1.557e-02  18.547  < 2e-16 ***
## STARS3              3.905e-01  1.716e-02  22.759  < 2e-16 ***
## STARS4              5.459e-01  2.437e-02  22.404  < 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: 14296  on 8973  degrees of freedom
## Residual deviance: 11394  on 8955  degrees of freedom
## AIC: 34889
##
## Number of Fisher Scoring iterations: 5
```

Examining the model indicates a very poor fit. Calculating the dispersion parameter indicates that dispersion does not appear to be a problem.

```
pchisq(lm.3$deviance, df=lm.3$df.residual, lower.tail=FALSE)
```

```
## [1] 1.287058e-63
```

Model 4

For the second poisson regression I will attempt to remove several of the non-statistically significant predictors. The lasso method for the previous model left numerous predictors that the other models found to not be significant and that exploratory analysis indicated did not have strong predictive power. I will attempt to simplify the model without losing predictive ability. [SEE APPENDIX]

```
lm.4 <- glm(TARGET ~ . -FixedAcidity -CitricAcid -ResidualSugar -Density -Sulphates,
            wine.training, family=poisson)
summary(lm.4)
```

```
##
## Call:
## glm(formula = TARGET ~ . - FixedAcidity - CitricAcid - ResidualSugar -
##     Density - Sulphates, family = poisson, data = wine.training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7238  -0.3539   0.1778   0.5740   2.5054
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          1.144e+00  7.026e-02  16.277  < 2e-16 ***
## VolatileAcidity     -4.351e-02  7.514e-03  -5.791 6.99e-09 ***
## Chlorides           -6.591e-02  1.878e-02  -3.510 0.000448 ***
## FreeSulfurDioxide    1.069e-04  3.980e-05   2.687 0.007218 **
## TotalSulfurDioxide   9.950e-05  2.574e-05   3.865 0.000111 ***
## pH                  -2.114e-02  8.717e-03  -2.426 0.015284 *
```

```
## Alcohol             5.906e-03  1.595e-03   3.703 0.000213 ***
## LabelAppeal-1       3.486e-01  4.669e-02   7.466 8.28e-14 ***
## LabelAppeal0        5.814e-01  4.563e-02  12.743  < 2e-16 ***
## LabelAppeal1        7.259e-01  4.635e-02  15.660  < 2e-16 ***
## LabelAppeal2        8.805e-01  5.171e-02  17.028  < 2e-16 ***
## AcidIndex          -1.047e-01  5.161e-03 -20.282  < 2e-16 ***
## STARS2              2.888e-01  1.557e-02  18.550  < 2e-16 ***
## STARS3              3.908e-01  1.715e-02  22.782  < 2e-16 ***
## STARS4              5.473e-01  2.436e-02  22.468  < 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: 14296  on 8973  degrees of freedom
## Residual deviance: 11400  on 8959  degrees of freedom
## AIC: 34888
##
## Number of Fisher Scoring iterations: 5
```

```
anova(lm.3, lm.4, test='Chisq')
```

Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
8955	11394.02	NA	NA	NA
8959	11400.34	-4	-6.321841	0.1763677

The anova test indicates that the more simple model is a valid substitution for the previous model without causing any lost of predictive ability.

```
pchisq(lm.4$deviance, df=lm.4$df.residual, lower.tail=FALSE)
```

```
## [1] 1.058372e-63
```

However, like the previous model this model still appears to be a poor fit.

Negative Binomial

Model 5

When a model appears as if it should follow a poisson distribution but the poisson model does not appear to be an appropriate fit, a negative biniomial model may address the underlying issues. I will fit a negative binomial model and select the predictors using `stepAIC`.

```
lm.5 <- glm.nb(TARGET ~ ., data=wine.training)
stepAIC(lm.5, trace=0)
```

```
##
## Call:  glm.nb(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + pH + Sulphates + Alcohol + LabelAppeal +
##     AcidIndex + STARS, data = wine.training, init.theta = 55688.87066,
##     link = log)
##
## Coefficients:
```

```
##        (Intercept)     VolatileAcidity           Chlorides
##          1.148e+00          -4.358e-02          -6.595e-02
##  FreeSulfurDioxide  TotalSulfurDioxide                  pH
##          1.076e-04           9.936e-05          -2.113e-02
##          Sulphates             Alcohol       LabelAppeal-1
##         -1.106e-02           5.916e-03           3.477e-01
##       LabelAppeal0        LabelAppeal1        LabelAppeal2
##          5.805e-01           7.250e-01           8.799e-01
##          AcidIndex              STARS2              STARS3
##         -1.044e-01           2.890e-01           3.907e-01
##              STARS4
##          5.466e-01
##
## Degrees of Freedom: 8973 Total (i.e. Null);  8958 Residual
## Null Deviance:       14300
## Residual Deviance: 11400     AIC: 34890
```

```
lm.5 <- update(lm.5, . ~ . -FixedAcidity -ResidualSugar)
summary(lm.5)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ VolatileAcidity + CitricAcid + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + Density + pH + Sulphates +
##     Alcohol + LabelAppeal + AcidIndex + STARS, data = wine.training,
##     init.theta = 55713.23098, link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.7621  -0.3532   0.1780   0.5747   2.4886
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.450e+00  2.306e-01   6.286 3.26e-10 ***
## VolatileAcidity    -4.328e-02  7.516e-03  -5.758 8.50e-09 ***
## CitricAcid          7.316e-03  6.884e-03   1.063 0.287902
## Chlorides          -6.504e-02  1.879e-02  -3.462 0.000536 ***
## FreeSulfurDioxide   1.077e-04  3.981e-05   2.706 0.006805 **
## TotalSulfurDioxide  1.000e-04  2.575e-05   3.883 0.000103 ***
## Density            -3.060e-01  2.235e-01  -1.369 0.171050
## pH                 -2.102e-02  8.717e-03  -2.412 0.015871 *
## Sulphates          -1.113e-02  6.390e-03  -1.741 0.081651 .
## Alcohol             5.863e-03  1.596e-03   3.674 0.000238 ***
## LabelAppeal-1       3.477e-01  4.669e-02   7.446 9.62e-14 ***
## LabelAppeal0        5.797e-01  4.564e-02  12.704  < 2e-16 ***
## LabelAppeal1        7.243e-01  4.635e-02  15.626  < 2e-16 ***
## LabelAppeal2        8.805e-01  5.171e-02  17.027  < 2e-16 ***
## AcidIndex          -1.043e-01  5.181e-03 -20.139  < 2e-16 ***
## STARS2              2.889e-01  1.557e-02  18.556  < 2e-16 ***
## STARS3              3.907e-01  1.715e-02  22.774  < 2e-16 ***
## STARS4              5.461e-01  2.437e-02  22.412  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(55713.23) family taken to be 1)
```

```
##
##     Null deviance: 14295  on 8973  degrees of freedom
## Residual deviance: 11394  on 8956  degrees of freedom
## AIC: 34890
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  55713
##           Std. Err.:  88470
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -34851.6
```

The model appears to suffer from a similar slate of issues in the diagnostics are in the previous models. [SEE APPENDIX]

Model 6

Exploring the TARGET in the training data reveals a disproportionately large amount of 0 values. 0s make up more than 20% of the total data, nearly double the proportional amount. This indicates that there is a hurdle in the number of cases purchased. That is, most wines have 0 purchases but amongst wines that have been purchased there is a fairly normal distribution.

```
wine.training %>%
  group_by(TARGET) %>%
  count()
```

TARGET	n
0	1610
1	152
2	760
3	1875
4	2389
5	1512
6	556
7	104
8	16

For the last model, I will fit a hurdle model. This will create two models – a logistic to determine whether bottles were purchased and a negative binomial indicating how many bottles were purchased. The predictors for the negative binomial distribution were selected based on the best poisson distribtuion discovered above (Model #4) while the logistic regression predictors were selected via a seperate regression analysis. [SEE APPENDIX]

```
lm.6 <- hurdle(TARGET ~ . -FixedAcidity -CitricAcid -ResidualSugar -Density -Sulphates |
                 . -FixedAcidity -ResidualSugar -Density -Alcohol, wine.training,
               dist='negbin')
summary(lm.6)
```

```
##
## Call:
## hurdle(formula = TARGET ~ . - FixedAcidity - CitricAcid - ResidualSugar -
##     Density - Sulphates | . - FixedAcidity - ResidualSugar - Density -
```

```
##     Alcohol, data = wine.training, dist = "negbin")
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -2.2238 -0.2945  0.1510  0.4698  3.1700
##
## Count model coefficients (truncated negbin with log link):
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         3.842e-01  8.351e-02   4.601 4.21e-06 ***
## VolatileAcidity    -1.053e-02  7.940e-03  -1.326  0.18486
## Chlorides          -1.573e-02  1.986e-02  -0.792  0.42827
## FreeSulfurDioxide   2.205e-05  4.141e-05   0.532  0.59440
## TotalSulfurDioxide -1.633e-05  2.634e-05  -0.620  0.53518
## pH                  1.063e-02  9.200e-03   1.155  0.24794
## Alcohol             7.112e-03  1.663e-03   4.276 1.91e-05 ***
## LabelAppeal-1       5.590e-01  6.101e-02   9.162  < 2e-16 ***
## LabelAppeal0        8.628e-01  5.994e-02  14.394  < 2e-16 ***
## LabelAppeal1        1.058e+00  6.052e-02  17.484  < 2e-16 ***
## LabelAppeal2        1.226e+00  6.478e-02  18.922  < 2e-16 ***
## AcidIndex          -1.748e-02  5.744e-03  -3.043  0.00234 **
## STARS2              1.104e-01  1.680e-02   6.574 4.91e-11 ***
## STARS3              1.968e-01  1.829e-02  10.761  < 2e-16 ***
## STARS4              3.116e-01  2.533e-02  12.302  < 2e-16 ***
## Log(theta)          1.745e+01  2.550e+00   6.844 7.71e-12 ***
## Zero hurdle model coefficients (binomial with logit link):
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         5.0980903  0.2730644  18.670  < 2e-16 ***
## VolatileAcidity    -0.2022335  0.0373057  -5.421 5.93e-08 ***
## CitricAcid          0.0524728  0.0342937   1.530 0.125991
## Chlorides          -0.3414480  0.0922837  -3.700 0.000216 ***
## FreeSulfurDioxide   0.0005195  0.0001968   2.639 0.008308 **
## TotalSulfurDioxide  0.0007198  0.0001270   5.669 1.44e-08 ***
## pH                 -0.1856717  0.0430200  -4.316 1.59e-05 ***
## Sulphates          -0.0930402  0.0318114  -2.925 0.003447 **
## LabelAppeal-1      -0.0822466  0.1606265  -0.512 0.608626
## LabelAppeal0       -0.2551787  0.1564156  -1.631 0.102803
## LabelAppeal1       -0.5055303  0.1629810  -3.102 0.001924 **
## LabelAppeal2       -0.5779396  0.2166341  -2.668 0.007635 **
## AcidIndex          -0.4124172  0.0210699 -19.574  < 2e-16 ***
## STARS2              0.9341205  0.0680435  13.728  < 2e-16 ***
## STARS3              1.1395231  0.0863380  13.198  < 2e-16 ***
## STARS4              1.5108329  0.1674202   9.024  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Theta: count = 37870399.0467
## Number of iterations in BFGS optimization: 61
## Log-likelihood: -1.588e+04 on 32 Df
```

SELECT MODELS

From the diagnostics of the models below it is clear that there is only one proper model. Model 6, the hurdle model, is not only a valid model but also accurately accounts for the disproportionate number of 0s in the training data. However, to be certain I compared the AIC, AICc and BIC values for each model. Model #6 easily performed the strongest in each of the three tests, further supporting the idea that Model #6 should be deployed.

```
data_frame('Model' = 1:6,
           'AIC' = c(AIC(lm.1), AIC(lm.2), AIC(lm.3), AIC(lm.4), AIC(lm.5), AIC(lm.6)),
           'AICc' = c(AICc(lm.1), AICc(lm.2), AICc(lm.3), AICc(lm.4), AICc(lm.5), AICc(lm.6)),
           'BIC' = c(useBIC(lm.1), useBIC(lm.2), useBIC(lm.3), useBIC(lm.4), useBIC(lm.5), useBIC(lm.6))
  kable()
```

Model	AIC	AICc	BIC
1	33550.02	33550.09	33677.85
2	33557.16	33557.22	33670.79
3	34889.21	34889.30	35024.15
4	34887.53	34887.59	34994.06
5	34889.60	34889.69	35024.54
6	31817.64	31817.87	32044.90

Examining the withheld testing data demonstrates that Model #6 properly predicts more than three times as many observations as any other model.

```
pred.1 <- predict(lm.1, newdata=wine.testing, interval='confidence')
a.1 <- sum(ifelse(pred.1[, 2] < wine.testing[, 1] & pred.1[, 3] > wine.testing[, 1], 1, 0))

pred.2 <- predict(lm.2, newdata=wine.testing, interval='confidence')
a.2 <- sum(ifelse(pred.2[, 2] < wine.testing[, 1] & pred.2[, 3] > wine.testing[, 1], 1, 0))

pred.3 <- predict(lm.3, newdata=wine.testing, type='link', se.fit=TRUE)
pred.3.upr <- pred.3$fit + (1.96*pred.3$se.fit)
pred.3.lwr <- pred.3$fit - (1.96*pred.3$se.fit)
pred.3.data <- data_frame(fit=pred.3$fit, lwr=pred.3.lwr, upr=pred.3.upr)
a.3 <- sum(ifelse(exp(pred.3.data[, 2]) < wine.testing[, 1] &
             exp(pred.3.data[, 3]) > wine.testing[, 1], 1, 0))

pred.4 <- predict(lm.4, newdata=wine.testing, type='link', se.fit=TRUE)
pred.4.upr <- pred.4$fit + (1.96*pred.4$se.fit)
pred.4.lwr <- pred.4$fit - (1.96*pred.4$se.fit)
pred.4.data <- data_frame(fit=pred.4$fit, lwr=pred.4.lwr, upr=pred.4.upr)
a.4 <- sum(ifelse(exp(pred.4.data[, 2]) < wine.testing[, 1] &
             exp(pred.4.data[, 3]) > wine.testing[, 1], 1, 0))

pred.5 <- predict(lm.5, newdata=wine.testing, type='link', se.fit=TRUE)
pred.5.upr <- pred.5$fit + (1.96*pred.5$se.fit)
pred.5.lwr <- pred.5$fit - (1.96*pred.5$se.fit)
pred.5.data <- data_frame(fit=pred.5$fit, lwr=pred.5.lwr, upr=pred.5.upr)
a.5 <- sum(ifelse(exp(pred.5.data[, 2]) < wine.testing[, 1] &
             exp(pred.5.data[, 3]) > wine.testing[, 1], 1, 0))

pred.6 <- predict(lm.6, newdata=wine.testing, type='prob', se.fit=TRUE)
```

```
y <- as_data_frame(pred.6) %>%
  mutate(obs = row_number()) %>%
  gather(key='prediction', value='prob', -obs) %>%
  group_by(obs) %>%
  filter(prob == max(prob)) %>%
  arrange(obs)

a.6 <- sum(ifelse(y == wine.testing[, 1], 1, 0))
data_frame('Model' = 1:6,
           'Correct' = c(a.1, a.2, a.3, a.4, a.5, a.6))
```

Model	Correct
1	191
2	165
3	218
4	191
5	214
6	717

Taking the total of all the information into account, there is no question that Model #6 is the best model. As such, I will be deploying this model on the evaluation data. Unlike the training data where I could remove entries with multiple missing values, I do not have that luxery with the evaluation data. I will impute everything missing.

```
set.seed(123)
wine.evaluation <- read_csv('C:\\Users\\Brian\\Desktop\\GradClasses\\Summer18\\621\\621week5\\wine-evalu
  dplyr::select(-IN) %>%
  mutate(LabelAppeal = factor(LabelAppeal),
         STARS = factor(STARS))

eval.imputed.data <- mice::mice(wine.evaluation[, -1], m=5, maxit=50,
                                method='pmm', seed=500, printFlag=FALSE)
wine.evaluation.complete <- cbind(wine.evaluation[, 1], complete(eval.imputed.data, 1))

write_csv(as_data_frame(round(predict(lm.6, newdata=wine.evaluation.complete,
                                      type='prob'), 2)), 'evaluation_predictions.csv')
```

Conclusion

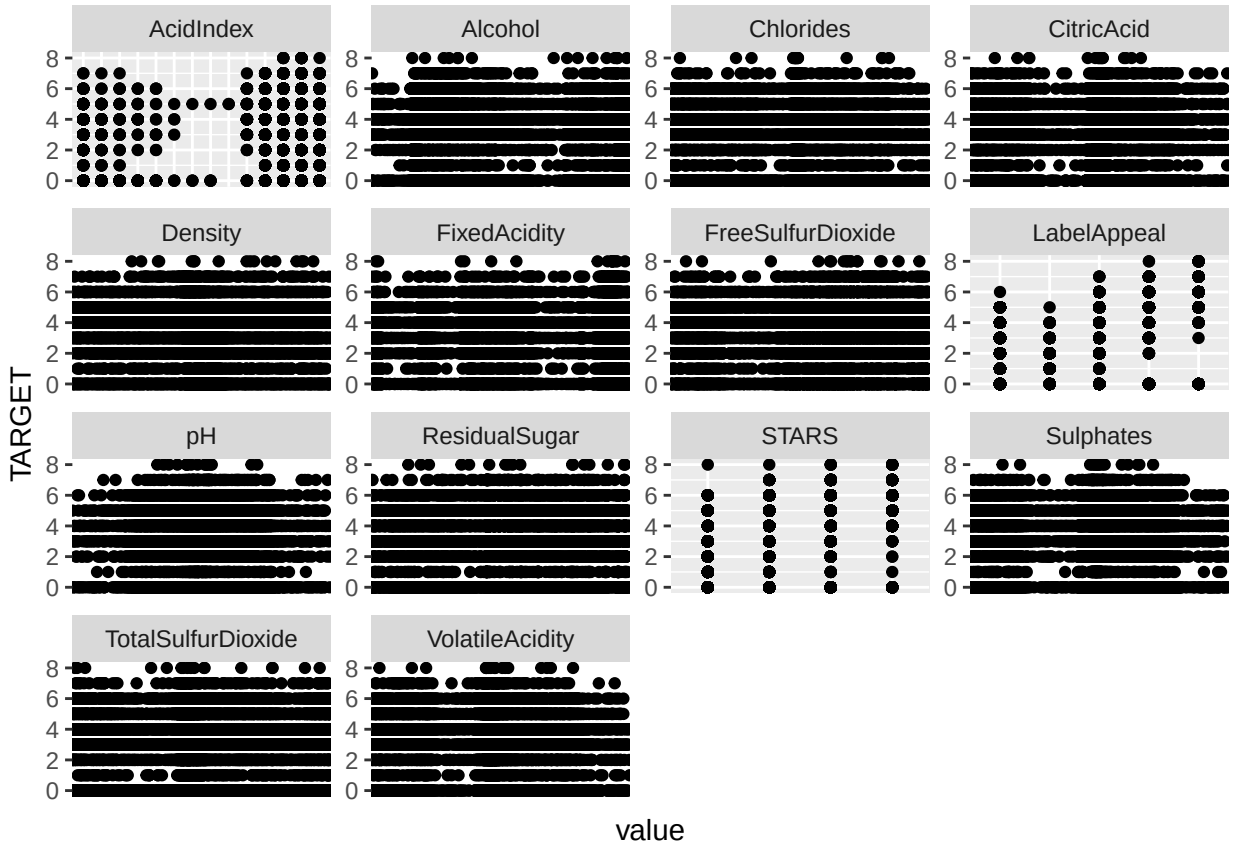
The analysis has successfully produced a model that can accurately predict the number of cases of wine purchased. This project underscores the importance of initial data exploration prior to the development of models. The disproportionately large number of 0 case entries highlights the need for a hurdle model. In fact, after examining a variety of different models, the hurdle model performed so much better than any other model that it was the only logical choice for deployment. Had this been an actual business situation I would have jumped right the hurdle methodology.

Appendix

Model 1

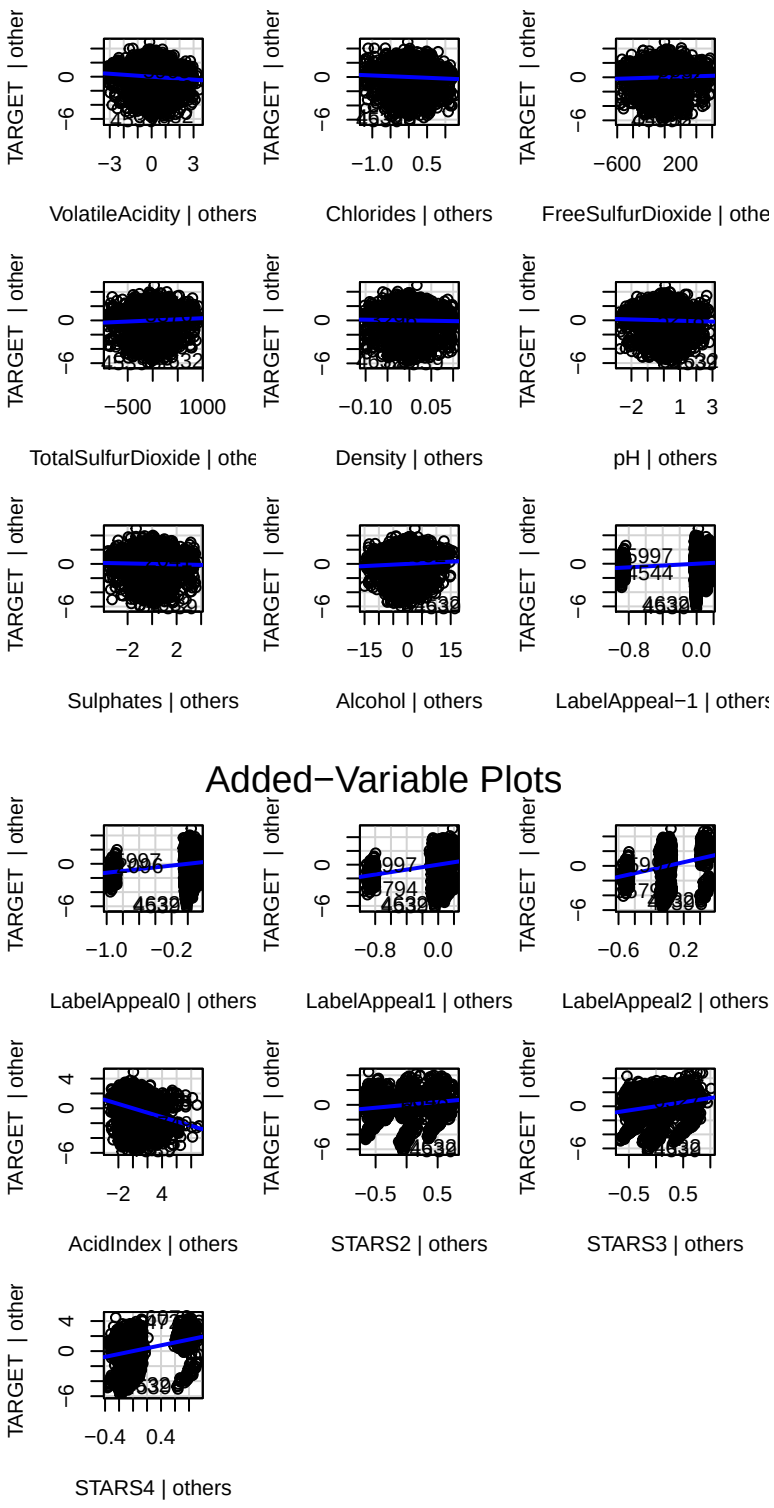
In order to perform a multiple linear regression their must be a linear relationship between the response variable and the predictor. The presence of this relationship is hard to determine due to the small variation in the reponse variable value. However, each predictor appears to be linearly related to the reponse.

```
wine.complete %>%
  gather(key='predictor', value='value', 2:15, -TARGET) %>%
  ggplot(aes(value, TARGET)) +
  geom_point() +
  geom_smooth(method='loess') +
  facet_wrap(~predictor, scales='free') +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank())
```

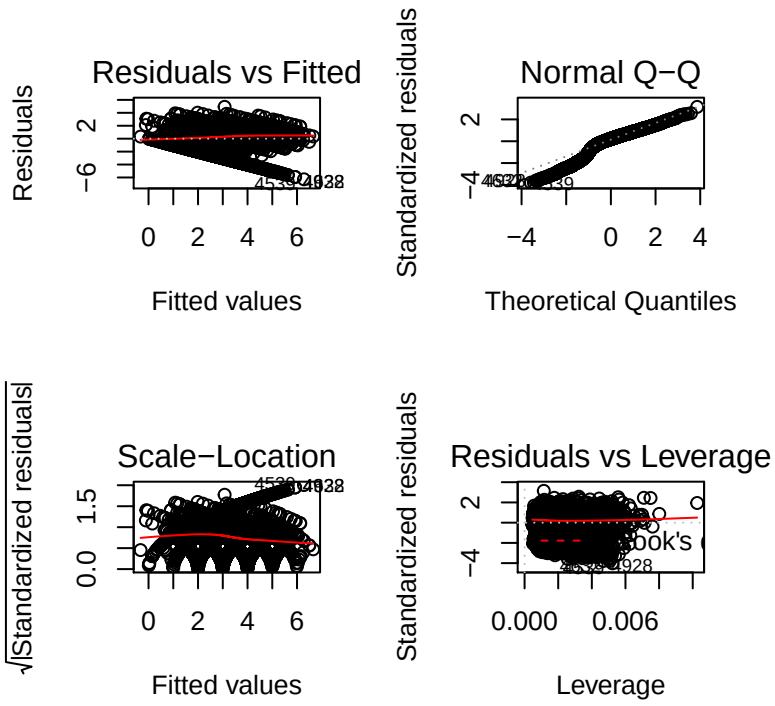


Examining the avPlots and the diagnostics shows that heteroscadacity is a problem with this model. As the values increase so does the variance in the residuals. This indicates that this is not a valid model. In addition, the qqplot shows that the residuals do not appear to be normally distributed and have long tails which are especially challenging for linear regression to model accurately.

```
car::avPlots(lm.1)
```




```
par(mfrow=c(2,2))
plot(lm.1)
```

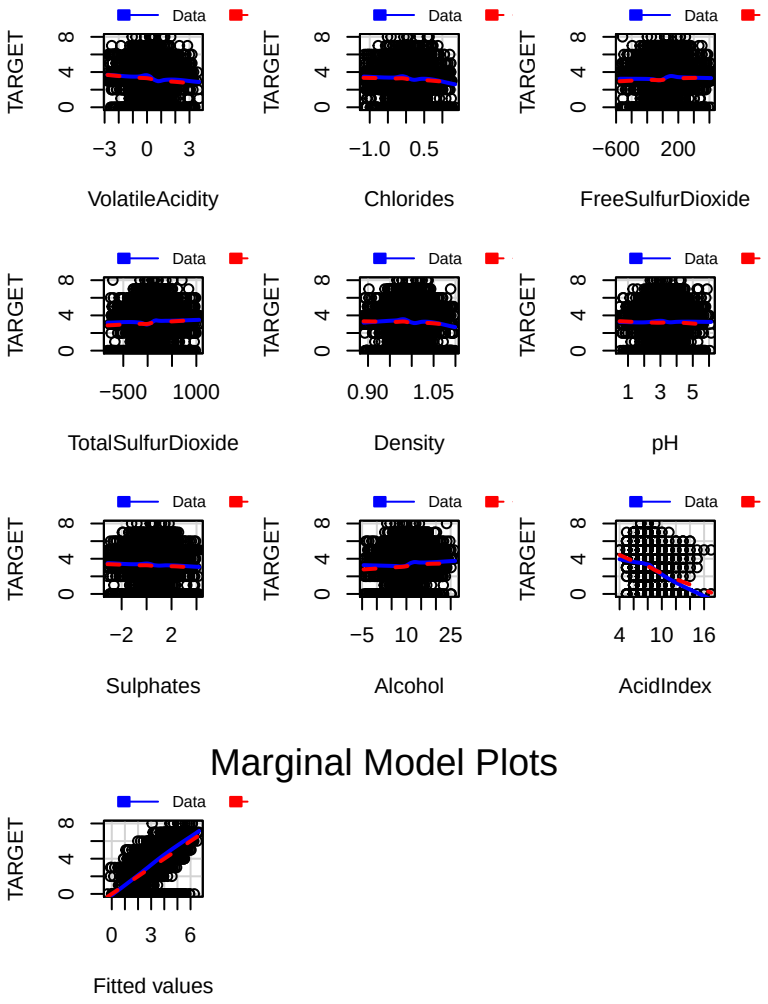


There does not appear to be an issue with mutli-collinearity in the data so it does not appear that there are any reasonable modifications that can be made to the raw data to make the multiple linear regression fit better.

```
car::vif(lm.1) %>%
  kable()
```

	GVIF	Df	GVIF^(1/(2*Df))
VolatileAcidity	1.004663	1	1.002329
Chlorides	1.004843	1	1.002419
FreeSulfurDioxide	1.003741	1	1.001869
TotalSulfurDioxide	1.005319	1	1.002656
Density	1.005422	1	1.002707
pH	1.005583	1	1.002788
Sulphates	1.002468	1	1.001233
Alcohol	1.010590	1	1.005281
LabelAppeal	1.152381	4	1.017887
AcidIndex	1.026764	1	1.013294
STARS	1.166877	3	1.026056

```
par(mfrow=c(2,1))
car::mmps(lm.1)
```

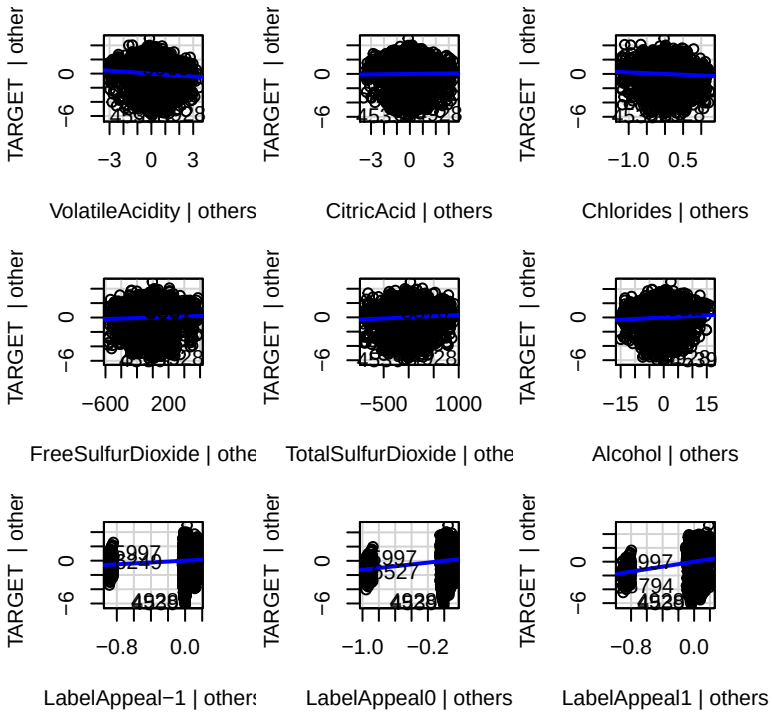


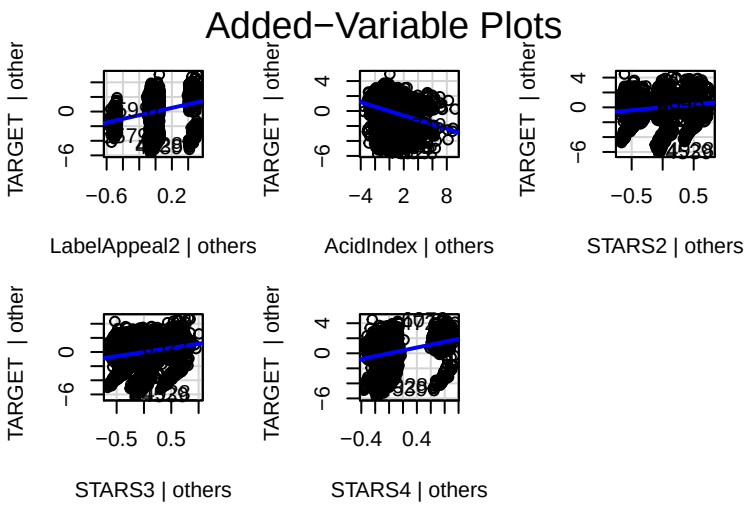
Finally, the marginal model plots appear to indicate a good fit.

Model 2

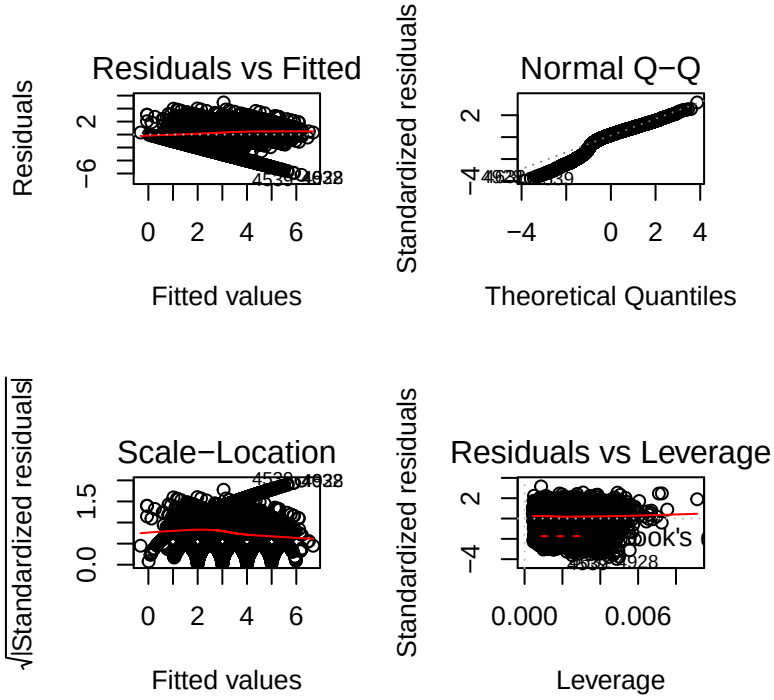
The second model attempts to further restrict the number of predictors used in order to create a more simple, interpretable model.

```
par(mfrow=c(2,1))
car::avPlots(lm.2)
```






```
par(mfrow=c(2,2))
plot(lm.2)
```



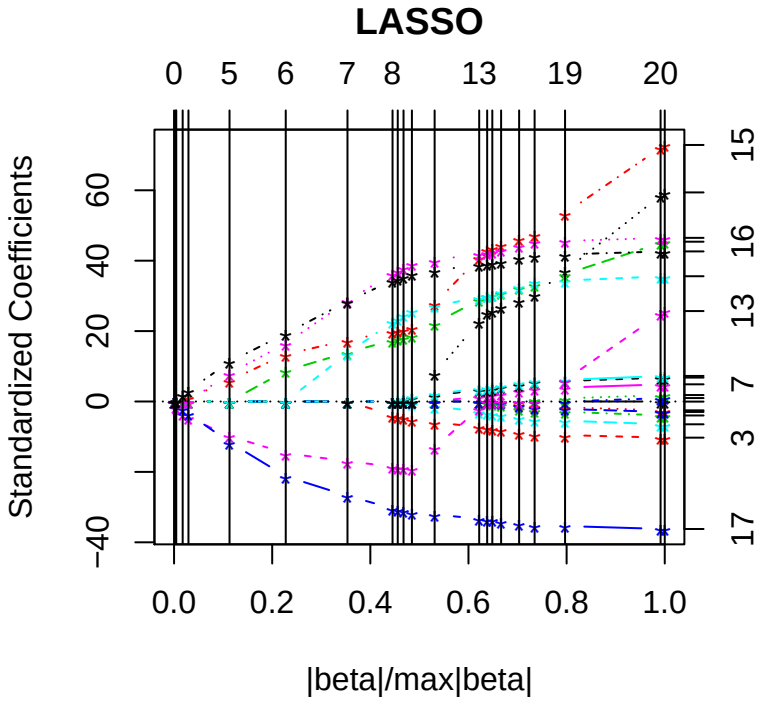
Just like with the first model, there appears to be an issue with non-constant variance and long tails in the qqplot. The anova test performed aboved indicates a reduction in predictive ability in this model compared

to the first one. It appears that I may have removed too many predictors. The marginal model plots and multi-collinearity check result in similar results to model 1.

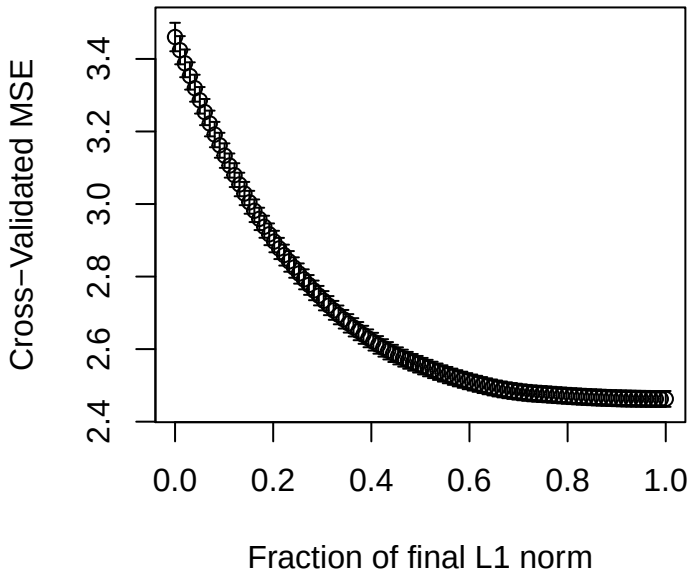
Model 3

This model's predictor selection was performed with the lasso method.

```
set.seed(123)
lasso <- lars(model.matrix(~ . -TARGET, wine.training), wine.training$TARGET)
plot(lasso)
```




```
cvlmod <- cv.lars(model.matrix(~ . -TARGET, wine.training), wine.training$TARGET)
```




```
cvlmod$index[which.min(cvlmod$cv)]
```

```
## [1] 0.989899
```

```
predict(lasso, s=0.989899, type='coef', mode='fraction')$coef
```

```
##        (Intercept)       FixedAcidity    VolatileAcidity
##       0.0000000000       0.0000000000      -0.1370187801
##         CitricAcid      ResidualSugar          Chlorides
##       0.0219868389       0.0002896797      -0.2138467299
##  FreeSulfurDioxide TotalSulfurDioxide            Density
##       0.0003426880       0.0003085241      -0.9801474520
##                 pH          Sulphates            Alcohol
##      -0.0605596730      -0.0334036729       0.0203411977
##      LabelAppeal-1       LabelAppeal0       LabelAppeal1
##       0.6109860242       1.2390453810       1.7675987802
##       LabelAppeal2          AcidIndex             STARS2
##       2.4741667747      -0.2950875661       0.7740221540
##             STARS3             STARS4
##       1.1694894726       1.9140150146
```

Exploration of the model's diagnostics reveal similar issues as the mutliple linear regression model.

```
AER::dispersiontest(lm.3)
```

```
##
##  Overdispersion test
##
## data:  lm.3
## z = -10.841, p-value = 1
## alternative hypothesis: true dispersion is greater than 1
```

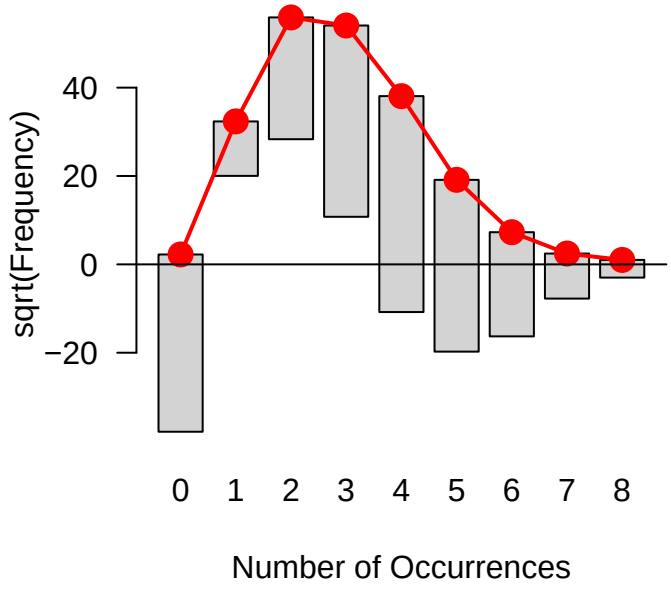
```
## sample estimates:
## dispersion
##  0.8315557
```

Dispersion is not an issue for this model so we will not need to adjust for that in our model.

```
summary(goodfit(wine.training$TARGET))
```

```
##
##   Goodness-of-fit test for poisson distribution
##
##                       X^2 df P(> X^2)
## Likelihood Ratio 5425.293  7        0
```

```
rootogram(table(wine.training$TARGET), fitted=table(c(trunc(fitted(lm.3)), 8)), type='hanging')
```



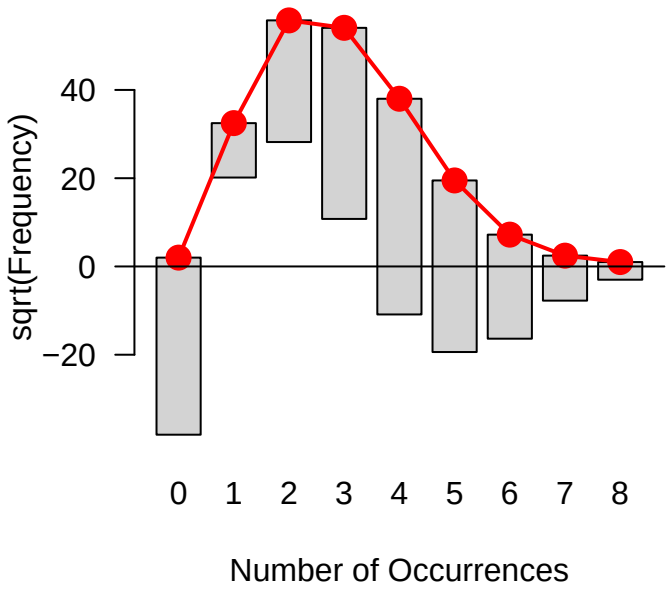
A goodfit test indicates that a poisson distribution is a poor choice for this model. The corresponding rootogram agrees with this assessment by underfitting predictions of 0 and overfitting predictions of 1 and 2 and 3.

Model 4

```
summary(fit <- goodfit(wine.training$TARGET))
```

```
##
##   Goodness-of-fit test for poisson distribution
##
##                       X^2 df P(> X^2)
## Likelihood Ratio 5425.293  7        0
```

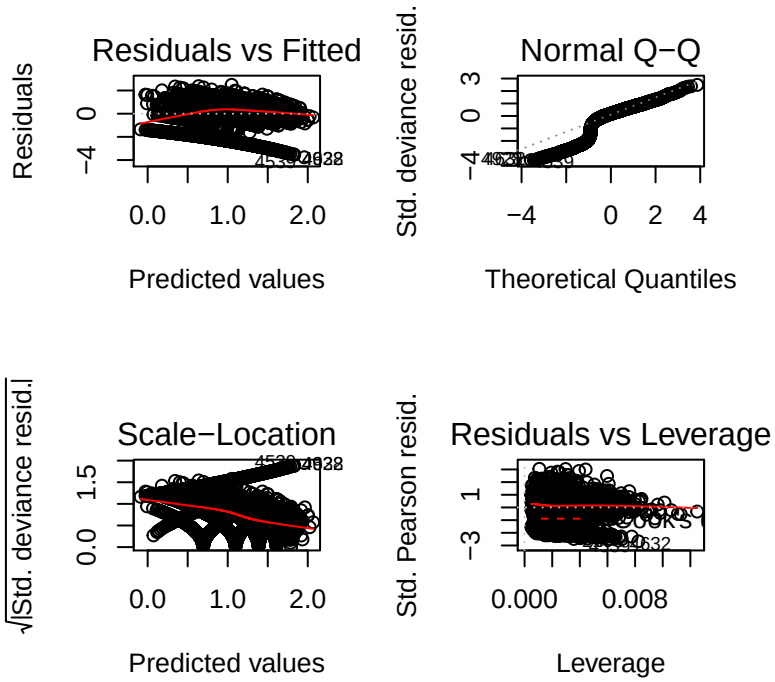
```
rootogram(table(wine.training$TARGET), fitted=table(c(trunc(fitted(lm.4)), 8)), type='hanging')
```



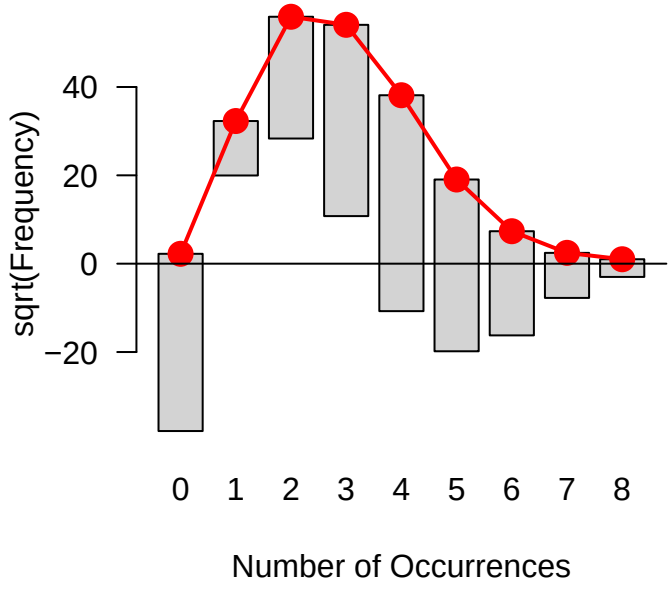
The modification of the predictors has not addressed the underlying issue that the model is underpredicting 0s and overpredicting 1, 2 and 3.

Model 5

```
par(mfrow=c(2,2))
plot(lm.5)
```




```
rootogram(table(wine.training$TARGET), fitted=table(c(trunc(fitted(lm.5)), 8)), type='hanging')
```

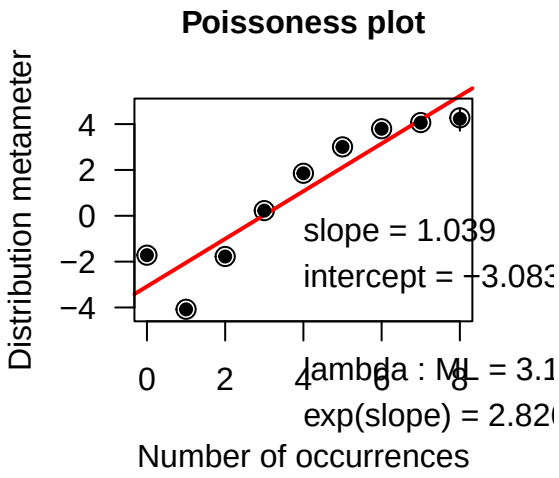


The diagnostics for model 5 paint a similarly poor fit as the previous models. All of them severly underrepresent the amount of 0 cases purchased. This all indicates that this data should be modeled with a hurdle model.

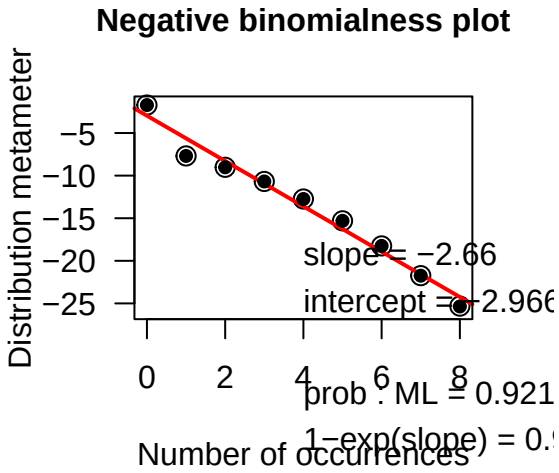
Model 6

In order to select the link functions, I performed an examination between a poisson distribution and a negative binomial distribution. The stronger performance of the negative binomial plot led me to select that as the link function.

```
distplot(wine.training$TARGET, type='poisson')
```




```
distplot(wine.training$TARGET, type='nbinomial')
```



The predictors for the logistic regression portion of the hurdle model were selected by performing a logistic regression on the data.

```
wine.bi <- wine.complete %>%
  mutate(TARGET_BI = factor(ifelse(TARGET > 0, 1, 0))) %>%
  dplyr::select(-TARGET)

lm.bi <- glm(TARGET_BI ~ ., data=wine.bi, family=binomial)
```

```
MASS::stepAIC(lm.bi, trace=0)
```

```
##
## Call:  glm(formula = TARGET_BI ~ VolatileAcidity + CitricAcid + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + pH + Sulphates +
##     LabelAppeal + AcidIndex + STARS, family = binomial, data = wine.bi)
##
## Coefficients:
##        (Intercept)     VolatileAcidity          CitricAcid
##          5.2137855          -0.1949509           0.0562673
##           Chlorides   FreeSulfurDioxide  TotalSulfurDioxide
##          -0.2733950           0.0006036           0.0007682
##                  pH           Sulphates       LabelAppeal-1
##          -0.1727857          -0.0893808          -0.2031410
##       LabelAppeal0        LabelAppeal1        LabelAppeal2
##          -0.3099805          -0.5131985          -0.6353142
##           AcidIndex              STARS2              STARS3
##          -0.4242696           0.8908109           1.0831065
##              STARS4
##           1.4307053
##
## Degrees of Freedom: 11215 Total (i.e. Null);  11200 Residual
## Null Deviance:       10580
## Residual Deviance: 9421  AIC: 9453
```

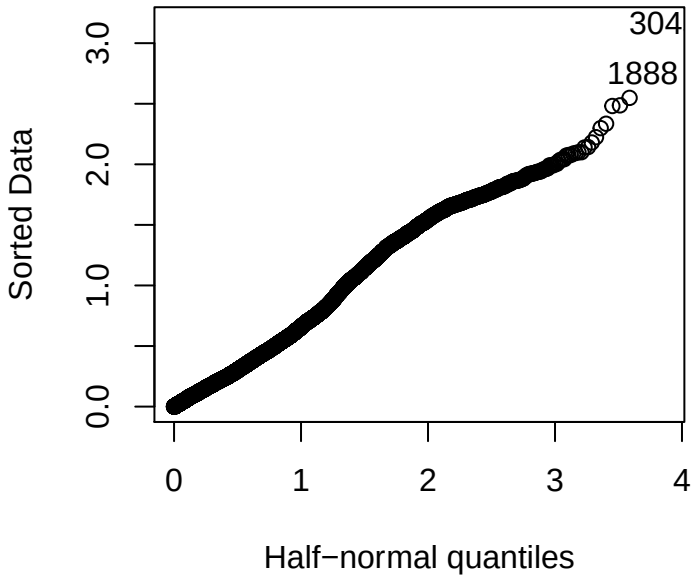
```
lm.bi <- update(lm.bi, . ~ . -FixedAcidity -ResidualSugar -Density -Alcohol)
summary(lm.bi)
```

```
##
## Call:
## glm(formula = TARGET_BI ~ VolatileAcidity + CitricAcid + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + pH + Sulphates +
##     LabelAppeal + AcidIndex + STARS, family = binomial, data = wine.bi)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6593   0.3494   0.4785   0.6307   2.2147
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         5.2137855  0.2443714  21.335  < 2e-16 ***
## VolatileAcidity    -0.1949509  0.0333830  -5.840 5.23e-09 ***
## CitricAcid          0.0562673  0.0306168   1.838 0.066093 .
## Chlorides          -0.2733950  0.0820062  -3.334 0.000857 ***
## FreeSulfurDioxide   0.0006036  0.0001760   3.430 0.000604 ***
## TotalSulfurDioxide  0.0007682  0.0001142   6.729 1.71e-11 ***
## pH                 -0.1727857  0.0386487  -4.471 7.80e-06 ***
## Sulphates          -0.0893808  0.0281740  -3.172 0.001512 **
## LabelAppeal-1      -0.2031410  0.1448540  -1.402 0.160801
## LabelAppeal0       -0.3099805  0.1414733  -2.191 0.028445 *
## LabelAppeal1       -0.5131985  0.1475762  -3.478 0.000506 ***
## LabelAppeal2       -0.6353142  0.1919931  -3.309 0.000936 ***
## AcidIndex          -0.4242696  0.0188336 -22.527  < 2e-16 ***
## STARS2              0.8908109  0.0607380  14.666  < 2e-16 ***
```

```
## STARS3              1.0831065  0.0767833  14.106  < 2e-16 ***
## STARS4              1.4307053  0.1484327   9.639  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10580.7  on 11215  degrees of freedom
## Residual deviance:  9420.9  on 11200  degrees of freedom
## AIC: 9452.9
##
## Number of Fisher Scoring iterations: 5
```

Exploring the fit of the model yields some promising results. First, the lack of fit in the qqplot has been addressed.

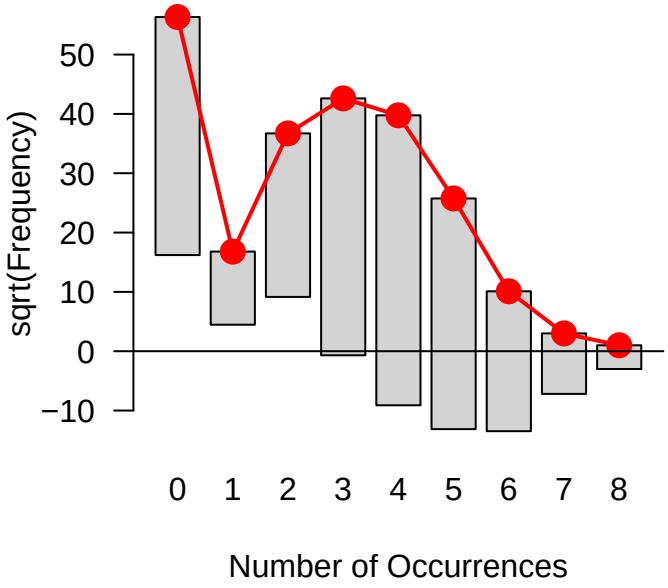
```
faraway::halfnorm(residuals(lm.6))
```



The rootogram as well shows a better fit. We are over fitting on 0, 1 and 2 but not nearly as much as the previous models.

```
y <- as_data_frame(predict(lm.6, newdata=wine.training, type='prob')) %>%
  mutate(obs = row_number()) %>%
  gather(key='prediction', value='prob', -obs) %>%
  group_by(obs) %>%
  filter(prob == max(prob))

rootogram(table(wine.training$TARGET), fitted=table(c(y$prediction, 8)), type='hanging')
```


sqrt(Frequency)
50
40
30
20
10
0
−10
0 1 2 3 4 5 6 7 8
Number of Occurrences