# CUNY DATA 621 - Business Analytics and Data Mining

Homework 5 - Wine

*Walt Wells, 2018*

## Problem

Our goal is to explore, analyze and model a dataset 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.

Your 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.

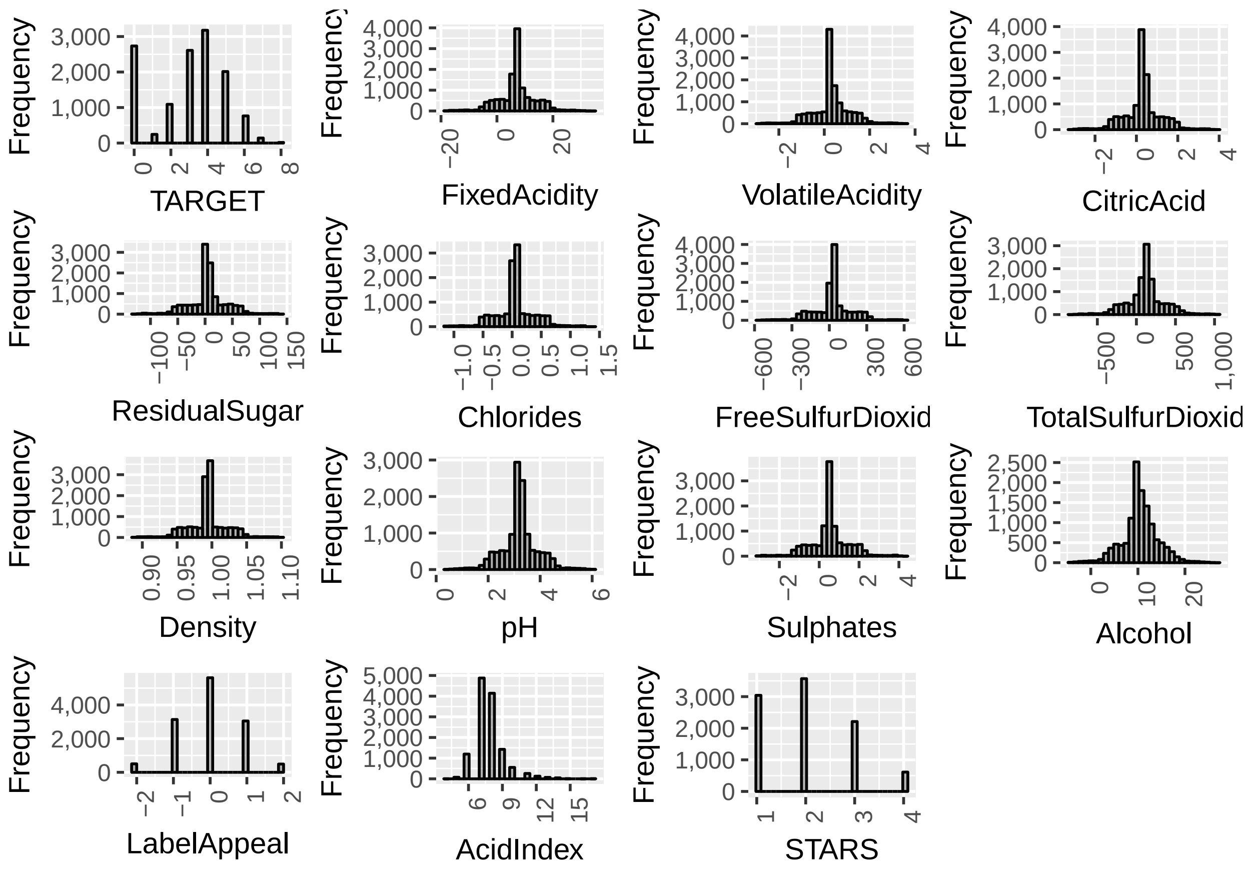
## 1. DATA EXPLORATION

Below we'll display a few basic EDA techniques to gain insight into our wine dataset.

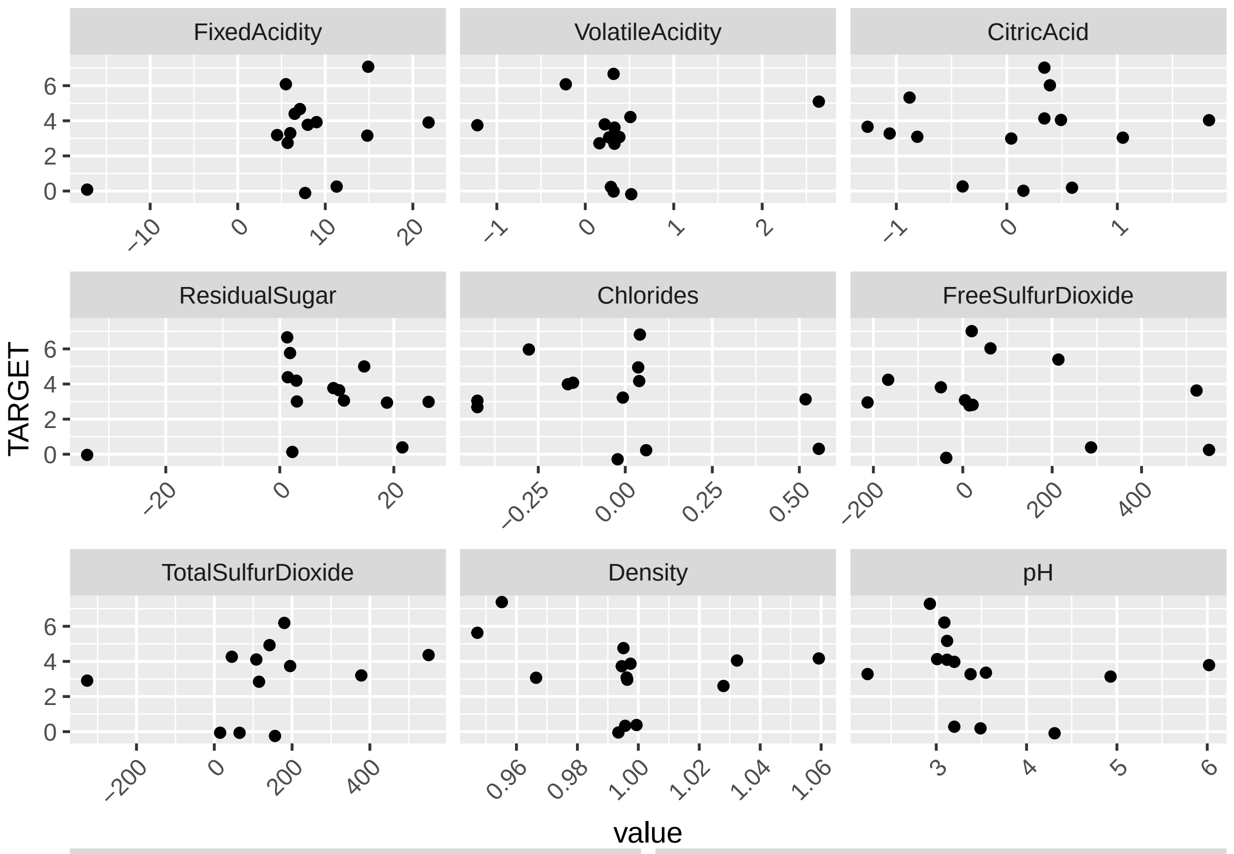
### Basic Statistics

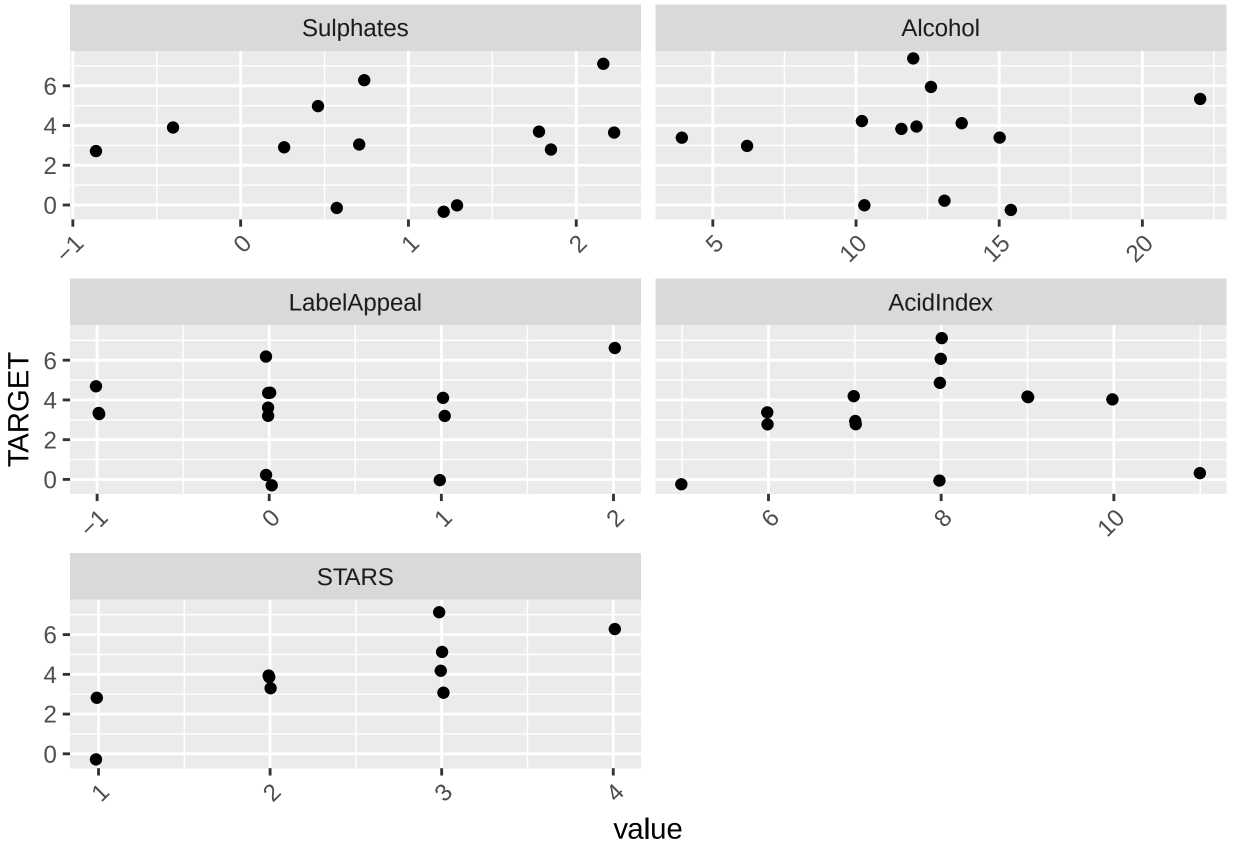
The data is 1.3 Mb in size. There are 12,795 rows and 15 columns (features). Of all 15 columns, 0 are discrete, 15 are continuous, and 0 are all missing. There are 8,200 missing values out of 191,925 data points.

**Histogram of Variables**



**Relationship of Predictors to Target**





## 2. DATA PREPARATION

### Negative Values

As discussed during a class, there are some wine quality measures that are negative that should not be. We will simply take the absolute value of these for now. The alternative would be to center by adding the min of each variable. Since we are given little information about the source of this dataset, and why these quality measures are so off, it is difficult to ascertain the best overall approach. Lacking that, we take the absolute value.

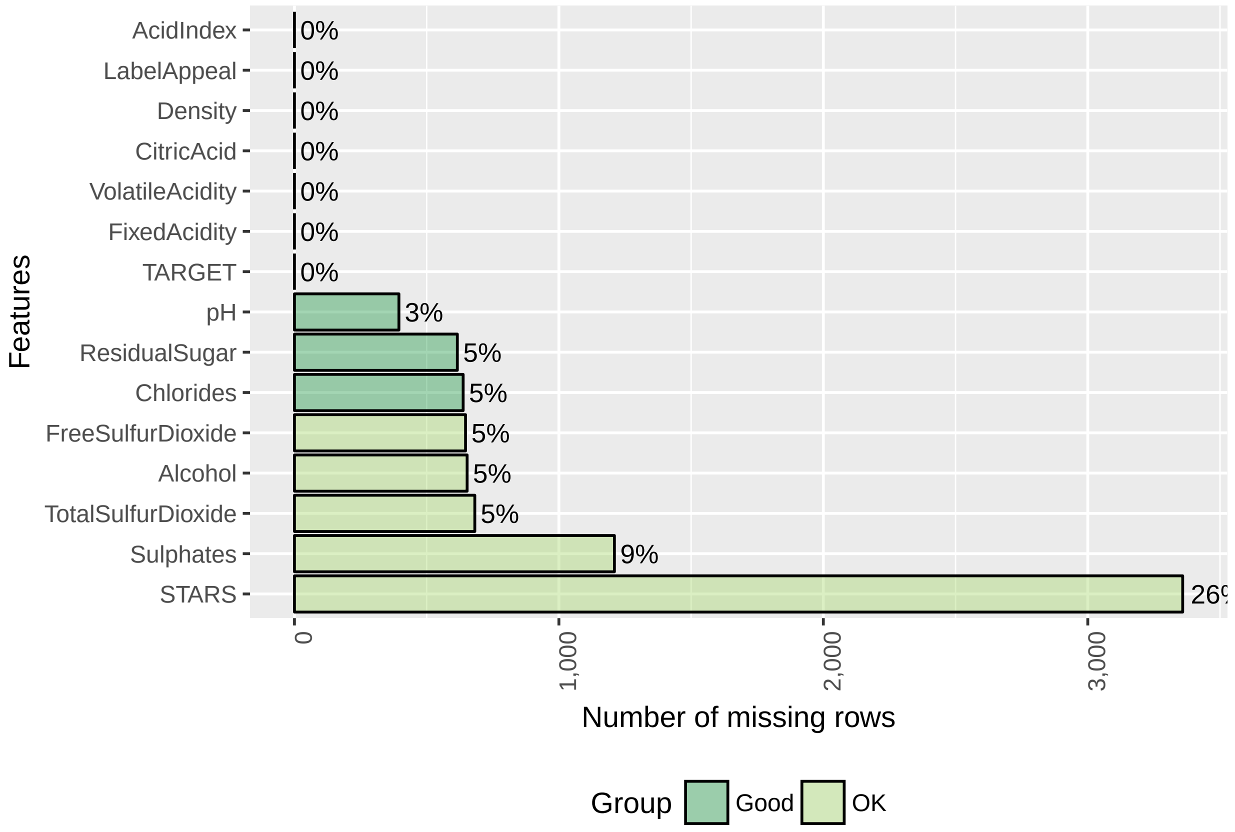
```
train$FixedAcidity <- abs(train$FixedAcidity)
test$FixedAcidity <- abs(test$FixedAcidity)

train$VolatileAcidity <- abs(train$VolatileAcidity)
test$VolatileAcidity <- abs(test$VolatileAcidity)

train$CitricAcid <- abs(train$CitricAcid)
test$CitricAcid <- abs(test$CitricAcid)

train$ResidualSugar <- abs(train$ResidualSugar)
test$ResidualSugar <- abs(test$ResidualSugar)

train$Chlorides <- abs(train$Chlorides)
test$Chlorides <- abs(test$Chlorides)

train$FreeSulfurDioxide <- abs(train$FreeSulfurDioxide)
test$FreeSulfurDioxide <- abs(test$FreeSulfurDioxide)

train$TotalSulfurDioxide <- abs(train$TotalSulfurDioxide)
test$TotalSulfurDioxide <- abs(test$TotalSulfurDioxide)

train$Sulphates <- abs(train$Sulphates)
test$Sulphates <- abs(test$Sulphates)
```

For LabelAppeal, we will add the min.

```
train$LabelAppeal <- train$LabelAppeal + abs(min(train$LabelAppeal))
test$LabelAppeal <- test$LabelAppeal + abs(min(test$LabelAppeal))
```

### Plot and Review Missing

```
plot_missing(train)
```



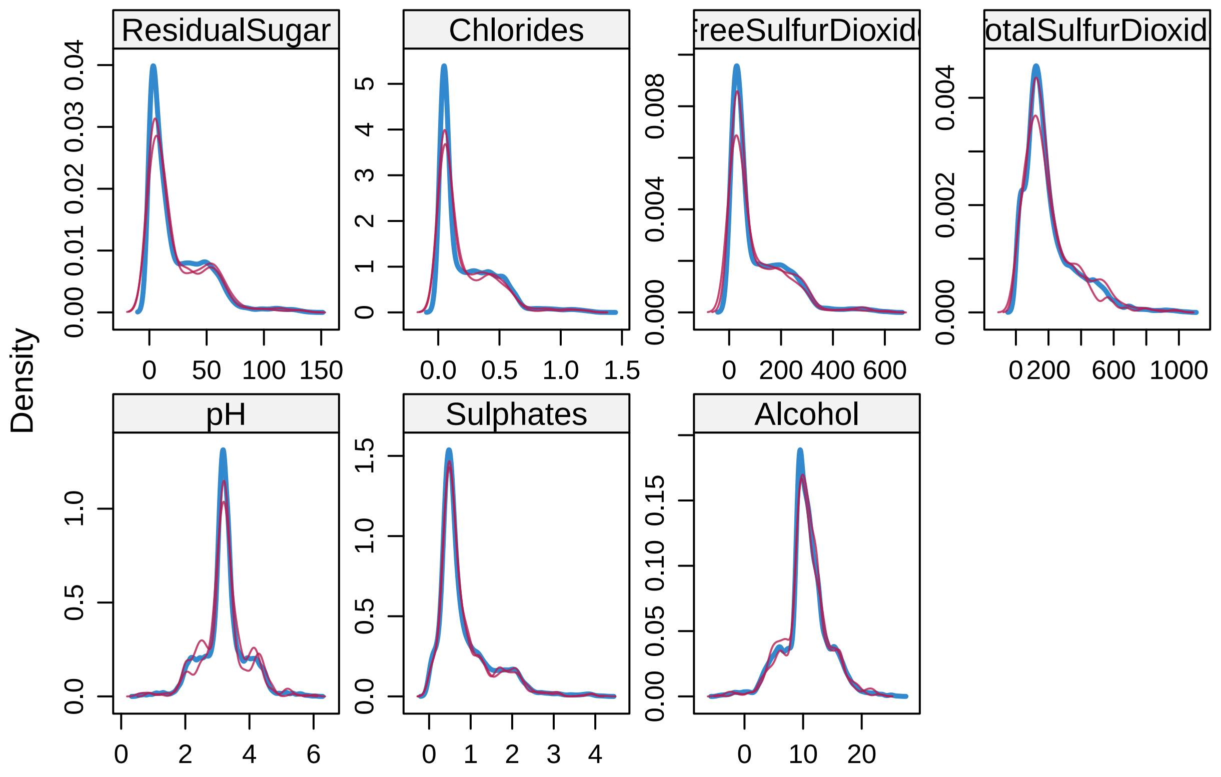
We need to make decisions about pH, ResidualSugar, Chlorides, Free SulfurDioxide, Alcohol, TotalSulfurDioxide, Sulphates, and STARS. After reviewing the test set, we are missing the same variables at about the same rate.

Of particular interest is our STARS. It's entirely likely, with 26% missing that this is a function of critics not getting around to reviewing, or it potentially being subpar. To manage this (and because it appears that STARS rating is indeed predictive of selling and missing values are not random) we will simply assign NAs a 0.

```
train$STARS[is.na(train$STARS)] <- 0
test$STARS[is.na(test$STARS)] <- 0
```

Elsewhere, let's look at using MICE for imputation.

```
mice_imputes <- mice(train, m = 2, maxit = 2, print = FALSE)
densityplot(mice_imputes)
```



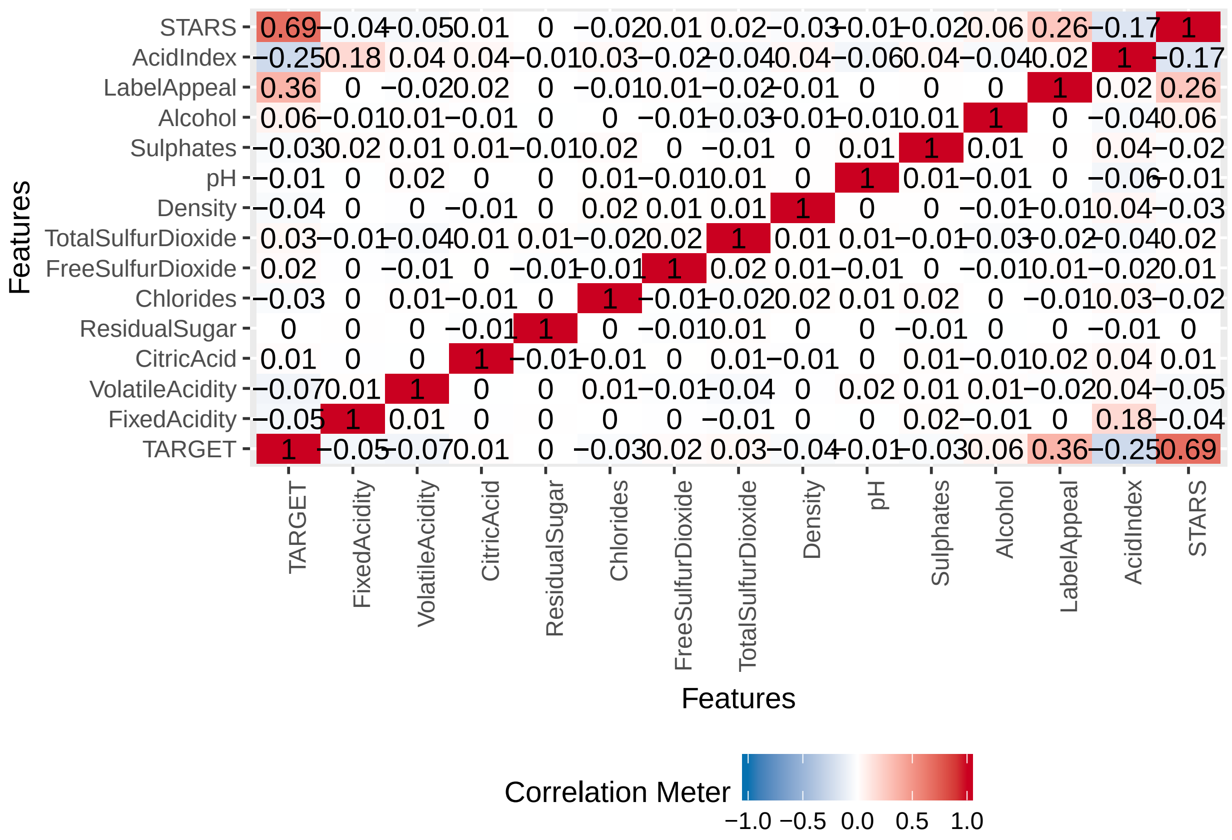
We can see that each of the remaining variables with missing values seem to be MAR, as the mice imputation distributions roughly match the existing.

We'll also run the mice imputation again on both the train and test set. Instead of using it for our models, however, we'll simplify our run and fill in our data. This is not a good method, as it doesn't account for variability, but it should do fine for the sake of this exercise.

```
mice_train <-  mice(train, m = 1, maxit = 1, print = FALSE)
train <- complete(mice_train)

mice_test <- mice(test, m = 1, maxit = 1, print = FALSE)
test <- complete(mice_test)
```

**Correlation Review**

```
plot_correlation(train)
```



And finally, after our analysis, (and so we can use it in in our model), we'll update STARS to become a factor variable.

```
train$STARS <- as.factor(train$STARS)
test$STARS <- as.factor(test$STARS)
```

## 3. BUILD MODELS

### Create Holdout

```
set.seed(121)
split <- sample(1:nrow(train), .8*nrow(train))

holdout <- train[-split,]
train <- train[split,]
```

### Model 1: Poisson

```
p1 <- glm(TARGET ~ . , data=train, family="poisson")
#summary(p1)
```

### Model 2: Poisson Reduced

```
p2 <- glm(TARGET ~ VolatileAcidity + Chlorides + TotalSulfurDioxide + Sulphates + Alcohol + LabelAppeal
#summary(p2)
```

### Model 3: Negative Binomial

```
nb1 <- glm.nb(TARGET ~ . , data=train)
```

```
## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached
```

```
#summary(nb1)
```

### Model 4: Negative Binomial Reduced

```
nb2 <- glm.nb(TARGET ~ VolatileAcidity + Chlorides + TotalSulfurDioxide + Sulphates + Alcohol + LabelApp
```

```
## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached
```

```
#summary(nb2)
```

### Model 5: Zero Dispersion Counts

We see that there are an inflated number of 0s in our counts target, so let's try this model across the negative binomial Distribution.

```
zMod1 <- zeroinfl(TARGET ~ . |STARS, data=train, dist="negbin")
#summary(zMod1)
```

### Model 6: Zero Dispersion Counts: Reduced

```
zMod2 <- zeroinfl(TARGET ~ VolatileAcidity + Chlorides + Density + Alcohol + LabelAppeal + AcidIndex | S
#summary(zMod2)
```

## 4. SELECT MODELS

To aid in model selection, let's test each of our models agains the holdout validation set.

```
getMAE <- function(x) {
    mean(abs(holdout$TARGET - x))
}

getRMSE <- function(x) {
    sqrt(mean((holdout$TARGET - x)^2))
}


results <- data.frame(Model = c("Poisson1",
                                "Poisson2",
                                "NegBinom1",
                                "NegBinom2",
                                "ZeroCounts1",
                                "ZeroCounts2"),
                     MAE = c(getMAE(predict.lm(p1, holdout)),
                             getMAE(predict.lm(p2, holdout)),
                             getMAE(predict.lm(nb1, holdout)),
                             getMAE(predict.lm(nb2, holdout)),
                             getMAE(predict(zMod1, holdout,
                                            type="response")),
                             getMAE(predict(zMod2, holdout,
                                            type="response"))
                             ),
                     RMSE = c(getRMSE(predict.lm(p1, holdout)),
                              getRMSE(predict.lm(p2, holdout)),
                              getRMSE(predict.lm(nb1, holdout)),
                              getRMSE(predict.lm(nb2, holdout)),
                              getRMSE(predict(zMod1, holdout,
                                            type="response")),
                              getRMSE(predict(zMod2, holdout,
                                            type="response"))
                              )
)

knitr::kable(results)
```

| Model | MAE | RMSE |
|---|---|---|
| Poisson1 | 2.233215 | 2.608078 |
| Poisson2 | 2.233099 | 2.607781 |
| NegBinom1 | 2.233215 | 2.608078 |
| NegBinom2 | 2.233099 | 2.607781 |
| ZeroCounts1 | 1.046420 | 1.349841 |
| ZeroCounts2 | 1.078326 | 1.404224 |

Here we see a preference for our full zero-counts Model. We'll make our predictions using our the entire model.

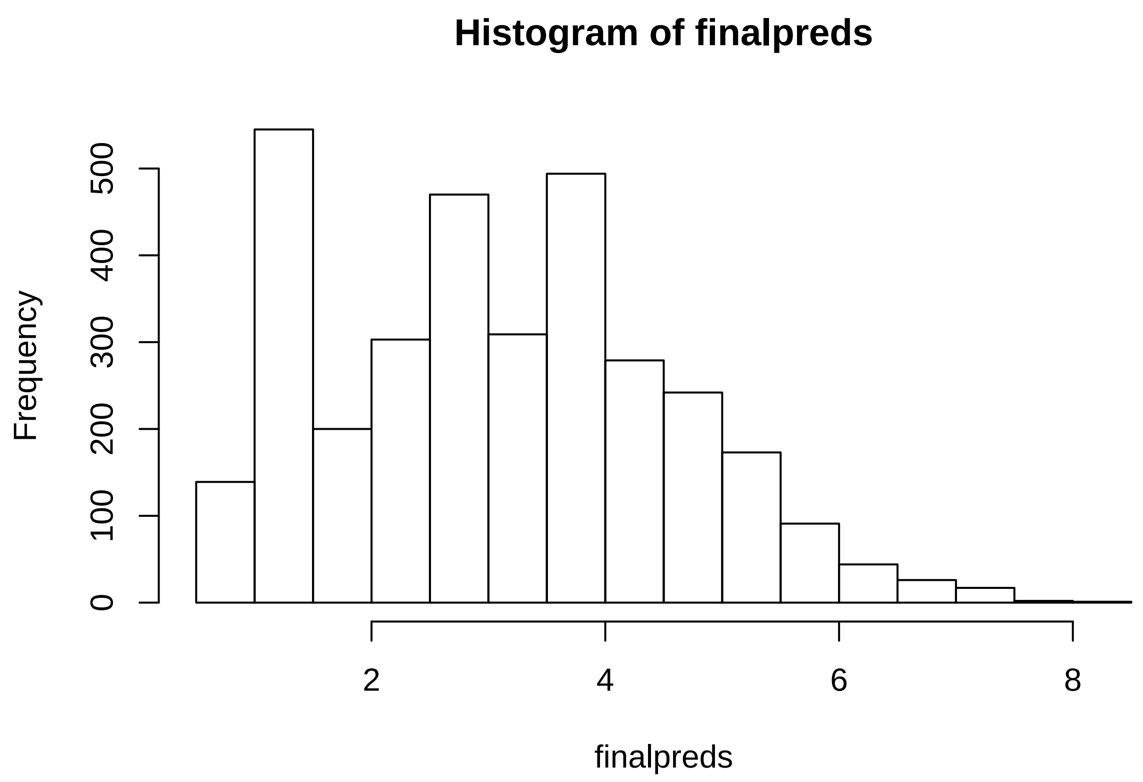
```
summary(zMod1)
```

```
## Warning in sqrt(diag(object$vcov)): NaNs produced

##
## Call:
## zeroinfl(formula = TARGET ~ . | STARS, data = train, dist = "negbin")
##
## Pearson residuals:
##      Min       1Q   Median       3Q      Max
## -2.45848 -0.54525  0.01476  0.43179  2.78664
##
## Count model coefficients (negbin with log link):
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        1.271e+00  2.249e-01   5.651 1.60e-08 ***
## FixedAcidity       4.335e-04  1.190e-03   0.364   0.7157
## VolatileAcidity   -1.936e-02  1.074e-02  -1.802   0.0716 .
## CitricAcid        -5.727e-04  9.482e-03  -0.060   0.9518
## ResidualSugar       7.409e-05  2.332e-04   0.318   0.7507
## Chlorides          -3.104e-02  2.482e-02  -1.251   0.2110
## FreeSulfurDioxide  -9.467e-06  5.250e-05  -0.180   0.8569
## TotalSulfurDioxide -8.350e-06  3.430e-05  -0.243   0.8077
## Density            -4.111e-01  2.193e-01  -1.874   0.0609 .
## pH                  3.845e-03  8.588e-03   0.448   0.6543
## Sulphates           4.916e-04  8.943e-03   0.055   0.9562
## Alcohol             6.577e-03  1.566e-03   4.201 2.65e-05 ***
## LabelAppeal         2.223e-01  7.086e-03  31.376  < 2e-16 ***
## AcidIndex          -3.136e-02  5.602e-03  -5.597 2.18e-08 ***
## STARS1              5.219e-02  2.368e-02   2.204   0.0275 *
## STARS2              1.829e-01  2.193e-02   8.339  < 2e-16 ***
## STARS3              2.798e-01  2.316e-02  12.082  < 2e-16 ***
## STARS4              3.799e-01  2.886e-02  13.163  < 2e-16 ***
## Log(theta)          1.521e+01         NA      NA       NA
##
## Zero-inflation model coefficients (binomial with logit link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.36392    0.04139   8.792   <2e-16 ***
## STARS1       -2.05760    0.07657 -26.873   <2e-16 ***
## STARS2      -17.82163         NA      NA       NA
## STARS3      -20.02237  443.24089  -0.045    0.964
## STARS4      -20.02170  851.69797  -0.024    0.981
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Theta = 4030189.8146
## Number of iterations in BFGS optimization: 50
## Log-likelihood: -1.66e+04 on 24 Df
```

## Make Predictions

We make our final predictions, create a dataframe with the prediction. We see that our predictions have a similar shape to our training Target variable.

```
finalpreds <- predict(zMod1, test)
finaldf <- cbind(TARGET_FLAG=finalpreds)

hist(finalpreds)
```



```r
write.csv(finaldf, 'HW5preds.csv', row.names = FALSE)
```

## Appendix

- For full output code visit: https://github.com/wwells/CUNY_DATA_621/blob/master/HW/HW5/HW5_WWells.Rmd
- For predicted values over test set visit: https://github.com/wwells/CUNY_DATA_621/blob/master/HW/HW5/HW5preds.csv