Data 621 Homework 5

Mark Gonsalves, Joshua Hummell, Claire Meyer, Chinedu Onyeka, Rathish Parayil Sasidharan

5/2/2022

```r
library(Amelia)
#library(rpart.plot)
#library(ggfortify)
#library(gridExtra)
#library(forecast)
#library(fpp2)
#library(fma)
library(kableExtra)
#library(e1071)
#library(mlbench)
library(ggcorrplot)
#library(DataExplorer)
library(timeDate)
library(caret)
#library(GGally)
library(corrplot)
library(RColorBrewer)
library(tidyverse)
library(caTools)
library(visdat)
library(dplyr)
#library(reshape2)
#library(mixtools)
#library(tidymodels)
#(ggpmisc)
#library(regclass)
#library(skimr)
#library(RANN)
#library(Hmisc)
library(MASS)
```

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

Variable Name	Definition
INDEX	Identification Variable (do not use)
TARGET	Number of Cases Purchased
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 customer
ResidualSugar	Residual Sugar of wine
STARS	Wine rating by a team of experts. 4 Stars = Excellent, 1 Star = Poor
Sulphates	Sulfate conten of wine
TotalSulfurDioxide	Total Sulfur Dioxide of Wine
VolatileAcidity	Volatile Acid content of wine
pH	pH of wine

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. HINT: Sometimes, the fact that a variable is missing is actually predictive of the target. You can only use the variables given to you (or variables that you derive from the variables provided). Below is a short description of the variables of interest in the data set

1. Data Exploration

Dataset

First we load the datasets.

```
url_train <- "https://raw.githubusercontent.com/chinedu2301/data621-business-analytics-data-mining/main/
url_eval  <- "https://raw.githubusercontent.com/chinedu2301/data621-business-analytics-data-mining/main/
training_df <- read.csv(url_train) %>% as.tibble()
eval_df <- read.csv(url_eval) %>% as.tibble()
```

Then we get the dimension of the training dataset.

```
dim(training_df)
```

```
## [1] 12795    16
```

The wine data set contains 16 variables including the target variable 'TARGET' variable and 12795 observations.

Then we get glimpse() of the training dataset.

```
glimpse(training_df)
```

```
## Rows: 12,795
## Columns: 16
## $ INDEX              <int> 1, 2, 4, 5, 6, 7, 8, 11, 12, 13, 14, 15, 16, 17, 19~
## $ TARGET             <int> 3, 3, 5, 3, 4, 0, 0, 4, 3, 6, 0, 4, 3, 7, 4, 0, 0, ~
## $ FixedAcidity       <dbl> 3.2, 4.5, 7.1, 5.7, 8.0, 11.3, 7.7, 6.5, 14.8, 5.5,~
## $ VolatileAcidity    <dbl> 1.160, 0.160, 2.640, 0.385, 0.330, 0.320, 0.290, -1~
## $ CitricAcid         <dbl> -0.98, -0.81, -0.88, 0.04, -1.26, 0.59, -0.40, 0.34~
## $ ResidualSugar      <dbl> 54.20, 26.10, 14.80, 18.80, 9.40, 2.20, 21.50, 1.40~
## $ Chlorides          <dbl> -0.567, -0.425, 0.037, -0.425, NA, 0.556, 0.060, 0.~
## $ FreeSulfurDioxide  <dbl> NA, 15, 214, 22, -167, -37, 287, 523, -213, 62, 551~
## $ TotalSulfurDioxide <dbl> 268, -327, 142, 115, 108, 15, 156, 551, NA, 180, 65~
## $ Density            <dbl> 0.99280, 1.02792, 0.99518, 0.99640, 0.99457, 0.9994~
## $ pH                 <dbl> 3.33, 3.38, 3.12, 2.24, 3.12, 3.20, 3.49, 3.20, 4.9~
## $ Sulphates          <dbl> -0.59, 0.70, 0.48, 1.83, 1.77, 1.29, 1.21, NA, 0.26~
## $ Alcohol            <dbl> 9.9, NA, 22.0, 6.2, 13.7, 15.4, 10.3, 11.6, 15.0, 1~
## $ LabelAppeal        <int> 0, -1, -1, -1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 2, 0, 0, ~
## $ AcidIndex          <int> 8, 7, 8, 6, 9, 11, 8, 7, 6, 8, 5, 10, 7, 8, 9, 8, 9~
## $ STARS              <int> 2, 3, 3, 1, 2, NA, NA, 3, NA, 4, 1, 2, 2, 3, NA, NA~
```

We see that data set contains only numerical variables, some of them are discrete with limited number of values. Since the Index column had no impact on the target variable, it can be dropped from training and evaluation data.

```
headers <- c("INDEX", "TARGET", "FixedAcidity", "VolatileAcidity", "CitricAcid", "ResidualSugar", "Chlo
colnames(training_df) <- headers
head(training_df)
```

```
## # A tibble: 6 x 16
##   INDEX TARGET FixedAcidity VolatileAcidity CitricAcid ResidualSugar Chlorides
##   <int>  <int>        <dbl>           <dbl>      <dbl>         <dbl>     <dbl>
## 1     1      3          3.2           1.16       -0.98          54.2    -0.567
## 2     2      3          4.5           0.16       -0.81          26.1    -0.425
## 3     4      5          7.1           2.64       -0.88          14.8     0.037
## 4     5      3          5.7           0.385       0.04          18.8    -0.425
## 5     6      4          8             0.33       -1.26           9.4    NA
## 6     7      0         11.3           0.32        0.59           2.2     0.556
## # ... with 9 more variables: FreeSulfurDioxide <dbl>, TotalSulfurDioxide <dbl>,
## #   Density <dbl>, pH <dbl>, Sulphates <dbl>, Alcohol <dbl>, LabelAppeal <int>,
## #   AcidIndex <int>, STARS <int>
```

```
head(eval_df)
```

```
## # A tibble: 6 x 16
##      IN TARGET FixedAcidity VolatileAcidity CitricAcid ResidualSugar Chlorides
##   <int> <lgl>         <dbl>           <dbl>      <dbl>         <dbl>     <dbl>
## 1     3 NA              5.4          -0.86        0.27         -10.7      0.092
## 2     9 NA             12.4           0.385      -0.76         -19.7      1.17
## 3    10 NA              7.2           1.75        0.17         -33        0.065
## 4    18 NA              6.2           0.1         1.8            1       -0.179
## 5    21 NA             11.4           0.21        0.28           1.2      0.038
## 6    30 NA             17.6           0.04       -1.15           1.4      0.535
## # ... with 9 more variables: FreeSulfurDioxide <dbl>, TotalSulfurDioxide <dbl>,
## #   Density <dbl>, pH <dbl>, Sulphates <dbl>, Alcohol <dbl>, LabelAppeal <int>,
## #   AcidIndex <int>, STARS <int>
```

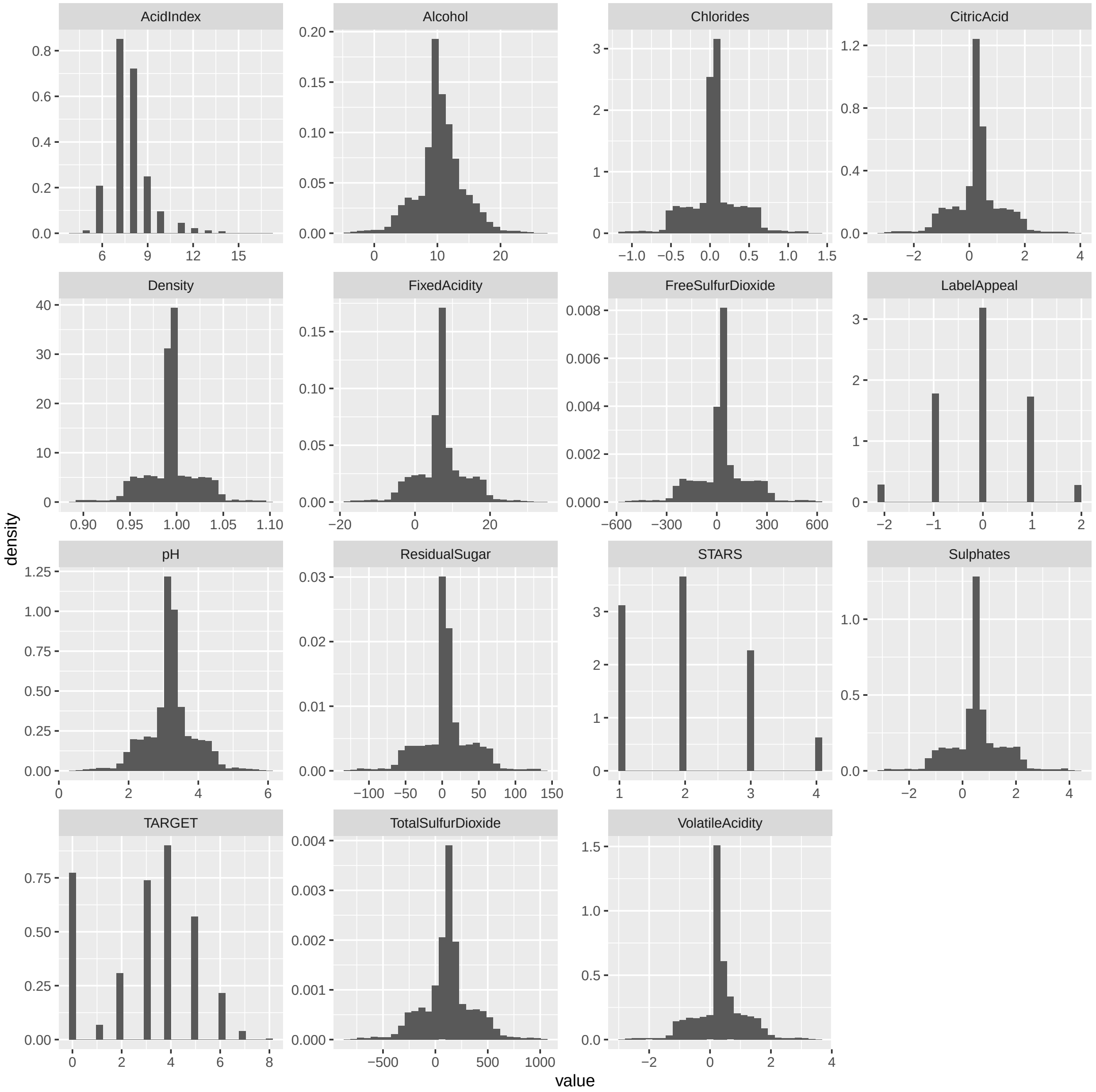
```
df_train <- training_df %>% dplyr::select(-c(INDEX))

df_eval <- eval_df %>% dplyr::select(-IN)
```

Let’s look at summary statistics.

```
##      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
```

And then let’s look at the distribution of each variable in the dataset.

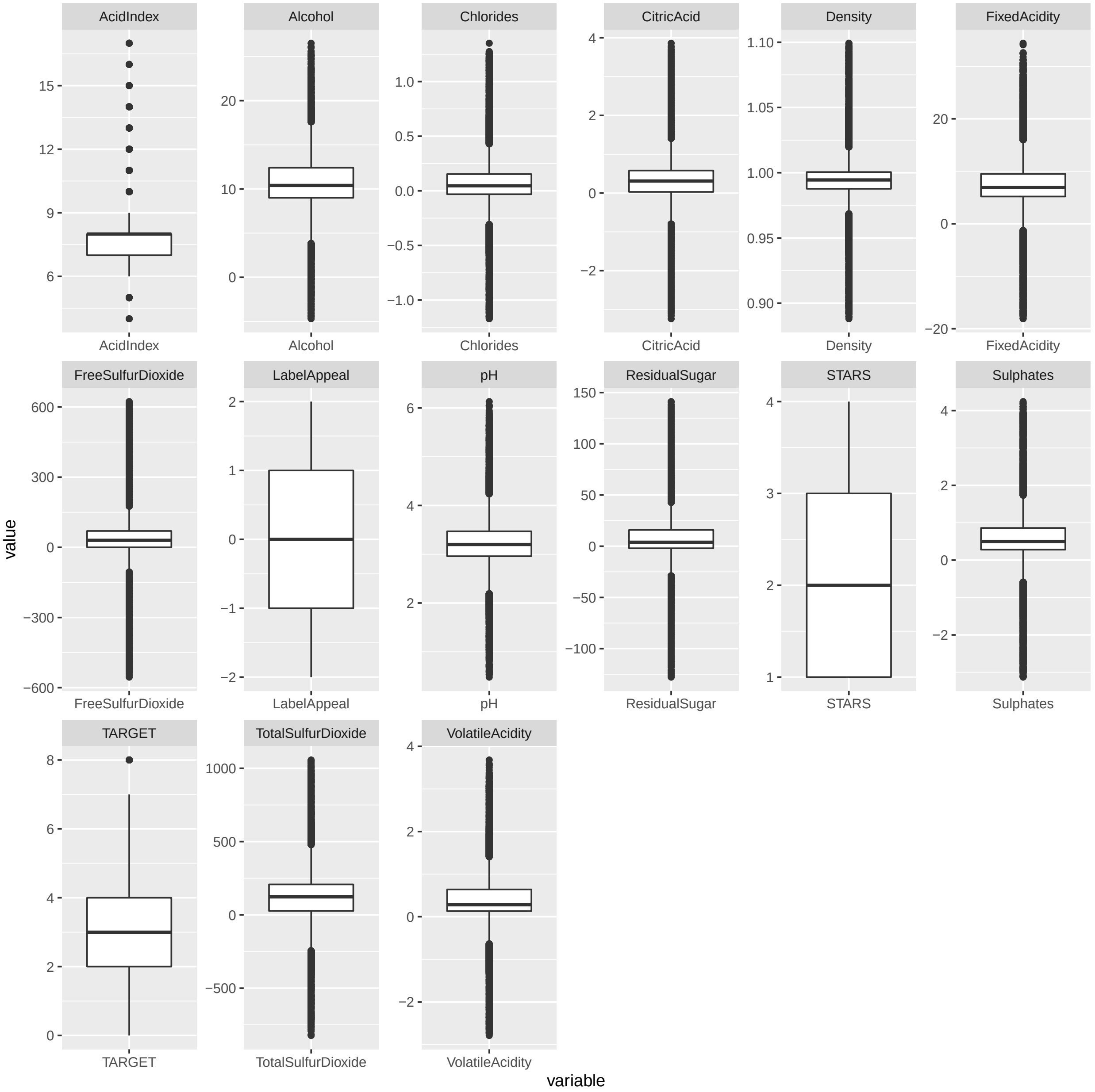


We see that most variables are somewhat normally distributed.

The distribution profiles show right skew in variables ‘AcidIndex’, and ‘STARS’.

Also we notice that some of these variables like STARS, Target, LabelAppeal etc. have discrete values, meaning they are categorical.

We analyze the spread of each variables using a box-plot.

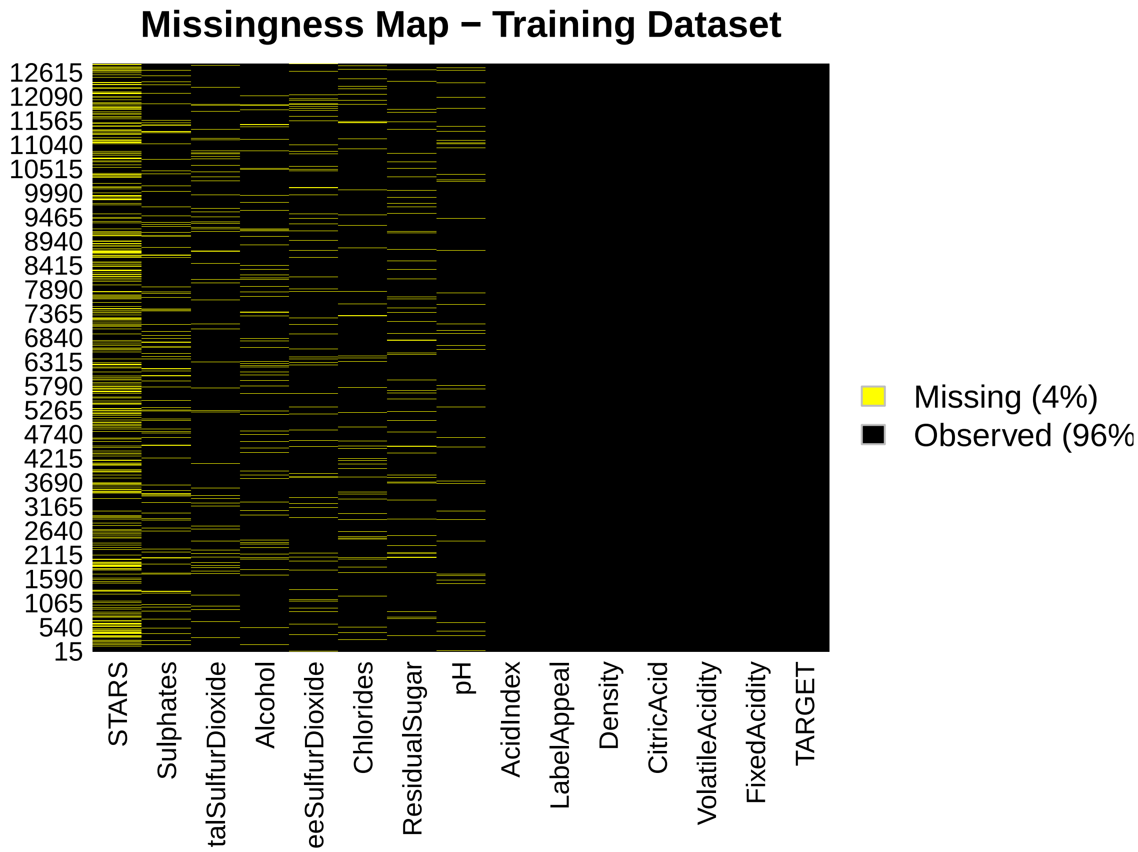


There are not many outliers in the variables.

We have already noticed that there are many missing values in the dataset. Let's analyze the distribution of missing values.

```
##         values                ind
## 1  0.2625244236              STARS
## 2  0.0945681907          Sulphates
## 3  0.0536928488 TotalSulfurDioxide
## 4  0.0517389605  FreeSulfurDioxide
## 5  0.0510355608            Alcohol
## 6  0.0498632278          Chlorides
## 7  0.0483782728      ResidualSugar
## 8  0.0308714342                 pH
## 9  0.0001563111       FixedAcidity
## 10 0.0000000000            TARGET
## 11 0.0000000000   VolatileAcidity
## 12 0.0000000000        CitricAcid
## 13 0.0000000000           Density
## 14 0.0000000000       LabelAppeal
## 15 0.0000000000         AcidIndex
```

```
missmap(df_train, col = c("yellow", "black"), main = "Missingness Map - Training Dataset")
```



We see STARS has lot of missing values, almost 26%, which we can replace with zero.

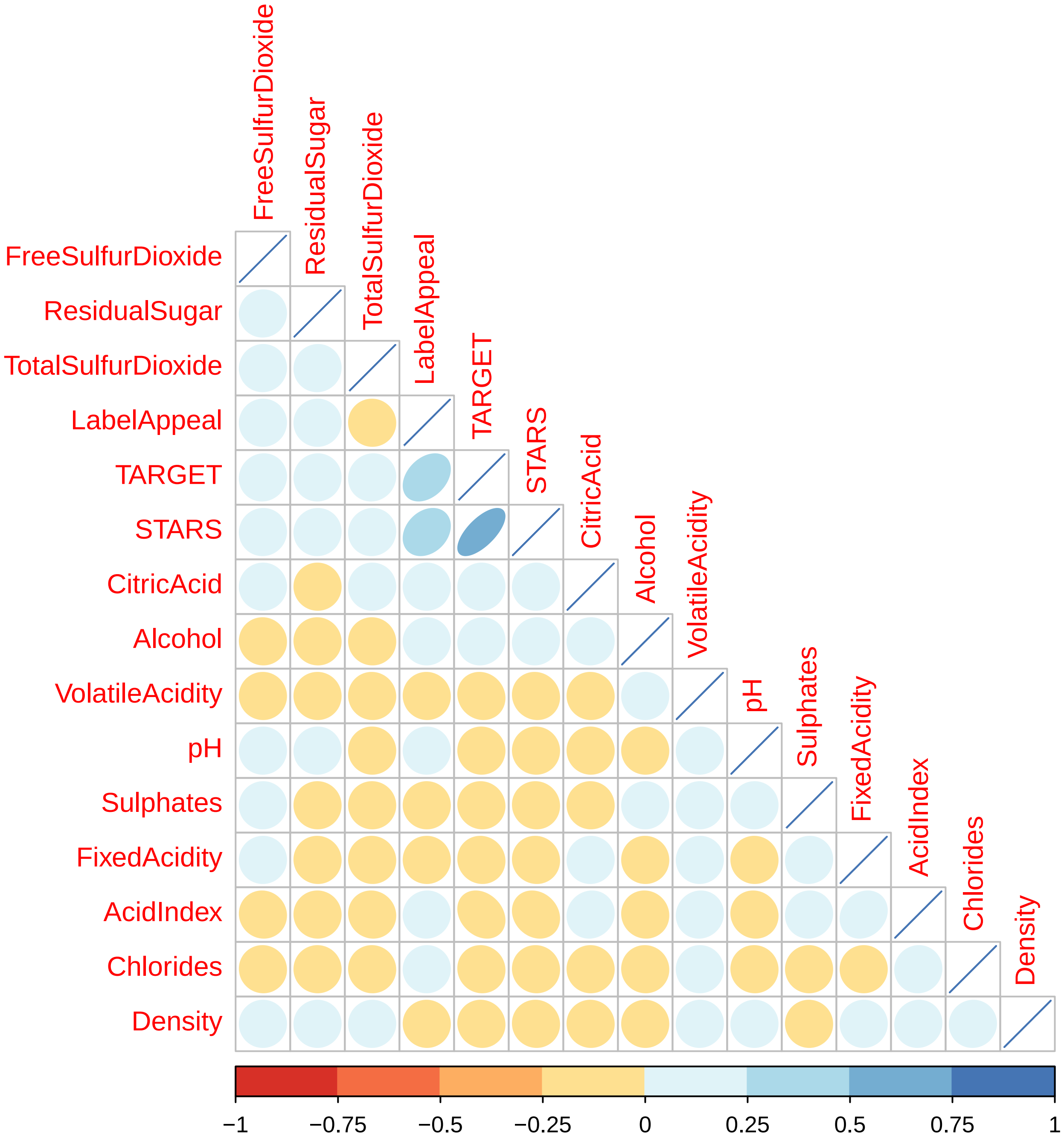
```
df_train["STARS"][is.na(df_train["STARS"])] <- 0
df_eval["STARS"][is.na(df_eval["STARS"])] <- 0
```

Then, let’s look at the correlation with Target.

```
##          values             ind
## 1  0.685381473           STARS
## 2  0.356500469     LabelAppeal
## 3  0.008684633      CitricAcid
## 4 -0.035517502         Density
## 5 -0.049010939    FixedAcidity
## 6 -0.088793212 VolatileAcidity
## 7 -0.246049449       AcidIndex
```

We see that 'STARS', 'LabelAppeal', and 'AcidIndex' have the highest correlation with 'TARGET'.

We create a correlation plot to check for multicollinearity.



We see that the features have very low correlations with each other, meaning that there is not much multicolinearity present in the dataset.

This means that the assumptions of linear regression are more likely to be met.

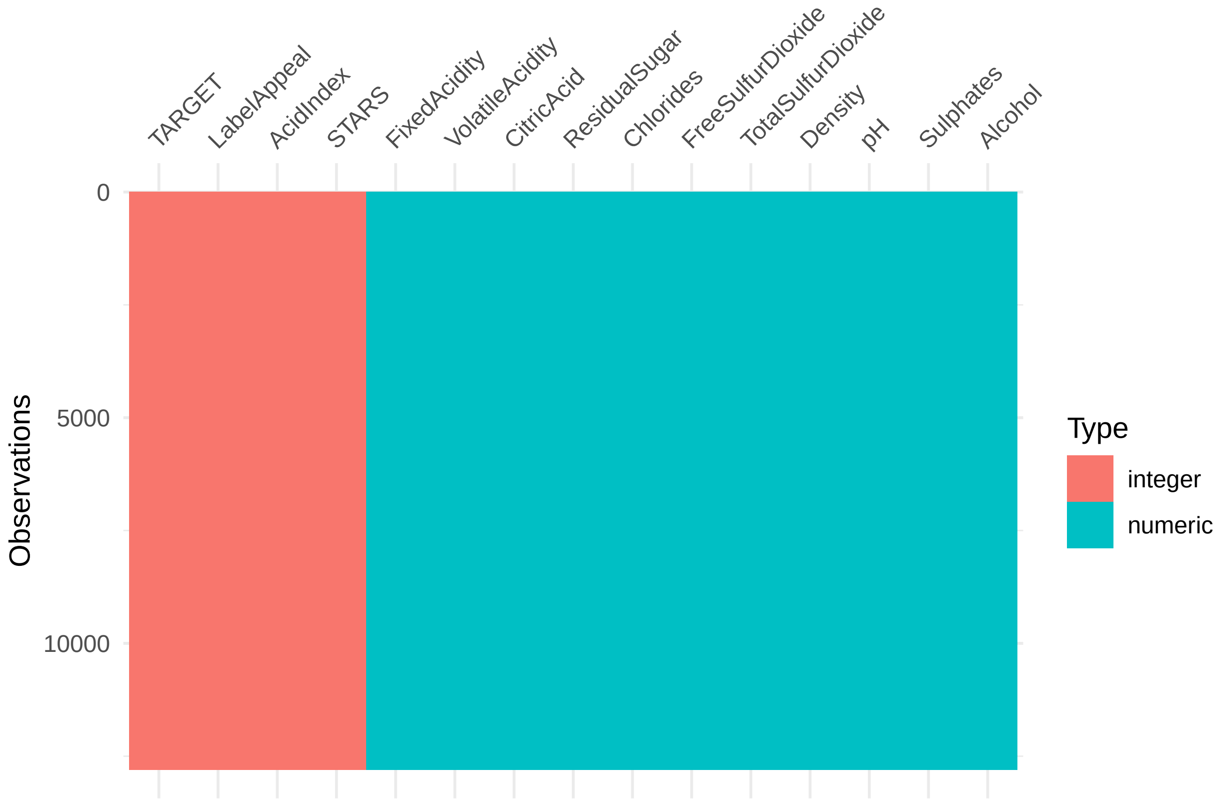
2. Data Preparation

First we can address all missing values in the dataset and replace with the mean:

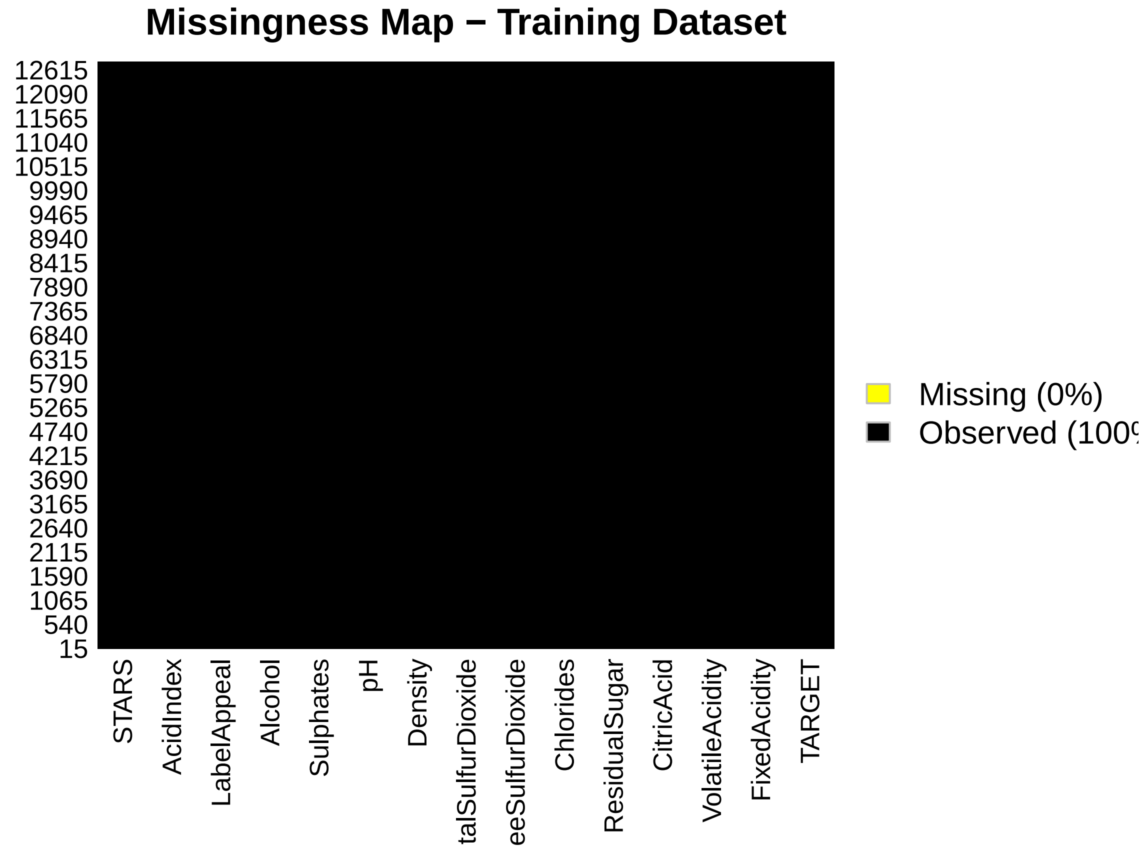
```
is_missing <- function(x){
  missing_strs <- c(’’, ’null’, ’na’, ’nan’, ’inf’, ’-inf’, ’-9’, ’unknown’, ’missing’)
  ifelse((is.na(x) | is.nan(x) | is.infinite(x)), TRUE,
         ifelse(trimws(tolower(x)) %in% missing_strs, TRUE, FALSE))
}
```

```
clean_df$STARS[is_missing(clean_df$STARS)] <- median(clean_df$STARS, na.rm = TRUE)
clean_df$Sulphates[is_missing(clean_df$Sulphates)] <- mean(clean_df$Sulphates, na.rm = TRUE)
clean_df$TotalSulfurDioxide[is_missing(clean_df$TotalSulfurDioxide)] <- mean(clean_df$TotalSulfurDioxid
clean_df$FreeSulfurDioxide[is_missing(clean_df$FreeSulfurDioxide)] <- mean(clean_df$FreeSulfurDioxide, 
clean_df$Alcohol[is_missing(clean_df$Alcohol)] <- mean(clean_df$Alcohol, na.rm = TRUE)
clean_df$Chlorides[is_missing(clean_df$Chlorides)] <- mean(clean_df$Chlorides, na.rm = TRUE)
clean_df$ResidualSugar[is_missing(clean_df$ResidualSugar)] <- mean(clean_df$ResidualSugar, na.rm = TRUE
clean_df$pH[is_missing(clean_df$pH)] <- mean(clean_df$pH, na.rm = TRUE)
clean_df$FixedAcidity[is_missing(clean_df$FixedAcidity)] <- mean(clean_df$FixedAcidity, na.rm = TRUE)
# assign the clean dataframe to training
training = clean_df
```

```
vis_dat(training)
```



```
missmap(training, col = c("yellow", "black"), main = "Missingness Map - Training Dataset")
```



We do the same for the evaluation dataset.

```
df_eval$STARS[is_missing(df_eval$STARS)] <- median(df_eval$STARS, na.rm = TRUE)
df_eval$Sulphates[is_missing(df_eval$Sulphates)] <- mean(df_eval$Sulphates, na.rm = TRUE)
df_eval$TotalSulfurDioxide[is_missing(df_eval$TotalSulfurDioxide)] <- mean(df_eval$TotalSulfurDioxide, n
df_eval$FreeSulfurDioxide[is_missing(df_eval$FreeSulfurDioxide)] <- mean(df_eval$FreeSulfurDioxide, na.
df_eval$Alcohol[is_missing(df_eval$Alcohol)] <- mean(df_eval$Alcohol, na.rm = TRUE)
df_eval$Chlorides[is_missing(df_eval$Chlorides)] <- mean(df_eval$Chlorides, na.rm = TRUE)
df_eval$ResidualSugar[is_missing(df_eval$ResidualSugar)] <- mean(df_eval$ResidualSugar, na.rm = TRUE)
df_eval$pH[is_missing(df_eval$pH)] <- mean(df_eval$pH, na.rm = TRUE)
df_eval$FixedAcidity[is_missing(df_eval$FixedAcidity)] <- mean(df_eval$FixedAcidity, na.rm = TRUE)
df_eval$VolatileAcidity[is_missing(df_eval$VolatileAcidity)] <- mean(df_eval$VolatileAcidity, na.rm = TI
df_eval$CitricAcid[is_missing(df_eval$CitricAcid)] <- mean(df_eval$CitricAcid, na.rm = TRUE)
df_eval$Density[is_missing(df_eval$Density)] <- mean(df_eval$Density, na.rm = TRUE)
df_eval$LabelAppeal[is_missing(df_eval$LabelAppeal)] <- mean(df_eval$LabelAppeal, na.rm = TRUE)
df_eval$AcidIndex[is_missing(df_eval$AcidIndex)] <- mean(df_eval$AcidIndex, na.rm = TRUE)
evaluation = df_eval
```

Then we split the dataset into test and train.

```
set.seed(101)

# Split the sample
```

```
sample <- sample.split(training$TARGET, SplitRatio = 0.8)

# Training sample data
wine_train <- subset(training, sample == TRUE)

# Test sample data
wine_test <- subset(training, sample == FALSE)
```

3. Build Models

Poisson Regression Model 1: In this Poisson Regression model, we will include all variables.

```
prmodel1 <- glm(TARGET ~ ., data = wine_train, family = poisson)
summary(prmodel1)
```

```
##
## Call:
## glm(formula = TARGET ~ ., family = poisson, data = wine_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8452  -0.7181   0.0620   0.5877   3.2218
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.480e+00  2.196e-01   6.739 1.59e-11 ***
## FixedAcidity       -5.073e-04  9.150e-04  -0.554 0.579271
## VolatileAcidity    -3.613e-02  7.300e-03  -4.949 7.46e-07 ***
## CitricAcid          1.020e-02  6.569e-03   1.553 0.120474
## ResidualSugar       1.903e-04  1.731e-04   1.099 0.271553
## Chlorides          -4.267e-02  1.831e-02  -2.331 0.019754 *
## FreeSulfurDioxide   1.423e-04  3.919e-05   3.632 0.000281 ***
## TotalSulfurDioxide  9.694e-05  2.555e-05   3.793 0.000149 ***
## Density            -2.511e-01  2.156e-01  -1.164 0.244324
## pH                 -1.560e-02  8.538e-03  -1.827 0.067719 .
## Sulphates          -1.358e-02  6.470e-03  -2.098 0.035876 *
## Alcohol             3.205e-03  1.584e-03   2.023 0.043048 *
## LabelAppeal         1.322e-01  6.779e-03  19.500  < 2e-16 ***
## AcidIndex          -8.666e-02  5.081e-03 -17.056  < 2e-16 ***
## STARS               3.119e-01  5.047e-03  61.799  < 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: 18291  on 10236  degrees of freedom
## Residual deviance: 11767  on 10222  degrees of freedom
## AIC: 37355
##
## Number of Fisher Scoring iterations: 5
```

Poisson Regression Model 2: In this model we will only look at significant variables.

```
prmodel2 <- glm(TARGET ~ . -CitricAcid -FixedAcidity -Chlorides - ResidualSugar -Density - TotalSulfurDi
summary(prmodel2)
```

```
##
## Call:
## glm(formula = TARGET ~ . - CitricAcid - FixedAcidity - Chlorides -
##     ResidualSugar - Density - TotalSulfurDioxide - FreeSulfurDioxide -
##     Alcohol - pH - Sulphates, family = poisson, data = wine_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8625  -0.7033   0.0595   0.5837   3.2731
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      1.232415   0.040848  30.171  < 2e-16 ***
## VolatileAcidity -0.036623   0.007298  -5.018 5.21e-07 ***
## LabelAppeal      0.131550   0.006775  19.416  < 2e-16 ***
## AcidIndex       -0.088304   0.004994 -17.683  < 2e-16 ***
## STARS            0.313706   0.005029  62.379  < 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: 18291  on 10236  degrees of freedom
## Residual deviance: 11818  on 10232  degrees of freedom
## AIC: 37386
##
## Number of Fisher Scoring iterations: 5
```

Negative Binomial Regression Model 1: In this Negative Binomial Regression model, we will include all variables.

```
nbrm1 <- glm.nb(TARGET ~ ., data = wine_train)
summary(nbrm1)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ ., data = wine_train, init.theta = 48949.4532,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8451  -0.7180   0.0620   0.5877   3.2217
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.480e+00  2.196e-01   6.739 1.59e-11 ***
## FixedAcidity       -5.074e-04  9.151e-04  -0.554 0.579275
## VolatileAcidity    -3.613e-02  7.300e-03  -4.949 7.46e-07 ***
## CitricAcid          1.020e-02  6.570e-03   1.553 0.120484
## ResidualSugar       1.903e-04  1.731e-04   1.099 0.271554
## Chlorides          -4.267e-02  1.831e-02  -2.331 0.019755 *
## FreeSulfurDioxide   1.423e-04  3.919e-05   3.632 0.000281 ***
## TotalSulfurDioxide  9.694e-05  2.556e-05   3.793 0.000149 ***
## Density            -2.511e-01  2.156e-01  -1.164 0.244339
## pH                 -1.560e-02  8.539e-03  -1.827 0.067714 .
## Sulphates          -1.358e-02  6.470e-03  -2.098 0.035878 *
## Alcohol             3.205e-03  1.584e-03   2.023 0.043059 *
## LabelAppeal         1.322e-01  6.779e-03  19.499  < 2e-16 ***
## AcidIndex          -8.666e-02  5.081e-03 -17.056  < 2e-16 ***
## STARS               3.119e-01  5.047e-03  61.797  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(48949.45) family taken to be 1)
##
##     Null deviance: 18290  on 10236  degrees of freedom
## Residual deviance: 11767  on 10222  degrees of freedom
## AIC: 37358
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  48949
##           Std. Err.:  56490
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -37325.54
```

We see Citric Acid, Residual Sugar, Free Sulfur Dioxide, Total Sulfur Dioxide, Alcohol and Stars are significant variables.

Negative Binomial Regression Model 2: In this Negative Binomial Regression Model, we will look at those significant variables.

```
nbrm2 <- glm.nb(TARGET ~ . +CitricAcid +ResidualSugar +TotalSulfurDioxide +FreeSulfurDioxide +Alcohol +
summary(nbrm2)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ . + CitricAcid + ResidualSugar + TotalSulfurDioxide +
##     FreeSulfurDioxide + Alcohol + STARS, data = wine_train, init.theta = 48949.4532,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8451  -0.7180   0.0620   0.5877   3.2217
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.480e+00  2.196e-01   6.739 1.59e-11 ***
## FixedAcidity       -5.074e-04  9.151e-04  -0.554 0.579275
## VolatileAcidity    -3.613e-02  7.300e-03  -4.949 7.46e-07 ***
## CitricAcid          1.020e-02  6.570e-03   1.553 0.120484
## ResidualSugar       1.903e-04  1.731e-04   1.099 0.271554    
## Chlorides          -4.267e-02  1.831e-02  -2.331 0.019755 *  
## FreeSulfurDioxide   1.423e-04  3.919e-05   3.632 0.000281 ***
## TotalSulfurDioxide  9.694e-05  2.556e-05   3.793 0.000149 ***
## Density            -2.511e-01  2.156e-01  -1.164 0.244339    
## pH                 -1.560e-02  8.539e-03  -1.827 0.067714 .  
## Sulphates          -1.358e-02  6.470e-03  -2.098 0.035878 *  
## Alcohol             3.205e-03  1.584e-03   2.023 0.043059 *  
## LabelAppeal         1.322e-01  6.779e-03  19.499  < 2e-16 ***
## AcidIndex          -8.666e-02  5.081e-03 -17.056  < 2e-16 ***
## STARS               3.119e-01  5.047e-03  61.797  < 2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## (Dispersion parameter for Negative Binomial(48949.45) family taken to be 1)
## 
##     Null deviance: 18290  on 10236  degrees of freedom
## Residual deviance: 11767  on 10222  degrees of freedom
## AIC: 37358
## 
## Number of Fisher Scoring iterations: 1
## 
## 
##               Theta:  48949 
##           Std. Err.:  56490 
## Warning while fitting theta: iteration limit reached 
## 
##  2 x log-likelihood:  -37325.54
```

Multiple Linear Regression Model 1: In this Multiple Linear Regression model, we will look at all variables.

```
mlr1 <- lm(TARGET ~ ., data = wine_train)
summary(mlr1)
```

```
## 
## Call:
## lm(formula = TARGET ~ ., data = wine_train)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -4.3176 -0.9452  0.0624  0.9259  5.9751 
## 
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)    
## (Intercept)         3.800e+00  5.025e-01   7.561 4.34e-14 ***
## FixedAcidity       -6.296e-04  2.099e-03  -0.300  0.76428    
## VolatileAcidity    -1.005e-01  1.670e-02  -6.018 1.83e-09 ***
## CitricAcid          2.642e-02  1.515e-02   1.743  0.08130 .  
## ResidualSugar       5.676e-04  3.986e-04   1.424  0.15449    
## Chlorides          -1.211e-01  4.197e-02  -2.885  0.00392 ** 
## FreeSulfurDioxide   3.752e-04  9.016e-05   4.161 3.19e-05 ***
## TotalSulfurDioxide  2.596e-04  5.850e-05   4.438 9.19e-06 ***
## Density            -6.685e-01  4.950e-01  -1.351  0.17687    
## pH                 -3.739e-02  1.957e-02  -1.910  0.05611 .
## Sulphates          -3.411e-02  1.488e-02  -2.292  0.02190 *
## Alcohol             1.426e-02  3.615e-03   3.945 8.04e-05 ***
## LabelAppeal         4.288e-01  1.528e-02  28.068  < 2e-16 ***
## AcidIndex          -2.050e-01  1.023e-02 -20.038  < 2e-16 ***
## STARS               9.783e-01  1.165e-02  83.962  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.324 on 10222 degrees of freedom
## Multiple R-squared:  0.5287, Adjusted R-squared:  0.5281
## F-statistic: 819.1 on 14 and 10222 DF,  p-value: < 2.2e-16
```

Here we see an adjusted R-square of 0.5281.

Multiple Linear Regression Model 2: In this model, we will look at those significant variables.

```
mlr2 <- lm(TARGET ~ . -CitricAcid -FixedAcidity -Chlorides - ResidualSugar -Density - TotalSulfurDioxide
summary(mlr2)
```

```
##
## Call:
## lm(formula = TARGET ~ . - CitricAcid - FixedAcidity - Chlorides -
##     ResidualSugar - Density - TotalSulfurDioxide - FreeSulfurDioxide -
##     Alcohol - pH - Sulphates, data = wine_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3442 -0.9538  0.0800  0.9253  6.0589
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.21774    0.08415  38.237  < 2e-16 ***
## VolatileAcidity -0.10170    0.01675  -6.073  1.3e-09 ***
## LabelAppeal      0.42627    0.01532  27.827  < 2e-16 ***
## AcidIndex       -0.20994    0.01004 -20.902  < 2e-16 ***
## STARS            0.98538    0.01166  84.525  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.328 on 10232 degrees of freedom
## Multiple R-squared:  0.5251, Adjusted R-squared:  0.5249
## F-statistic:  2829 on 4 and 10232 DF,  p-value: < 2.2e-16
```

We see that the adjusted R-squared value of 0.5249 was actually worse than our first MLR model.

4. Select Models

```
model_test <- function(model, wine_test, trainY) {
  # Evaluate Model 1 with testing data set
  predictedY <- predict(model, newdata=wine_test)
```

	RMSE	Rsquared	MAE	aic	bic	m
prmodel1_eval	2.59206392755416	0.523083117977806	2.25869826962797	37355.3208660464	37463.8273243502	p
prmodel2_eval	2.59146396280063	0.524553446289377	2.25909608477653	37385.8401566517	37422.0089760863	p
nbrm1_eval	2.59206300547322	0.523082810276954	2.25869629078798	37357.5433390294	37473.2835612201	n
nbrm2_eval	2.59206300547322	0.523082810276954	2.25869629078798	37357.5433390294	37473.2835612201	n
mlr1_eval	1.32741201248383	0.524762584796396	1.0625722003606	34807.8425223109	34923.5827445016	m
mlr2_eval	1.3255535618419	0.526080896568868	1.0615300940943	34865.1042764434	34908.5068597649	m

```
  model_results <- data.frame(obs = trainY, pred=predictedY)
  colnames(model_results) = c('obs', 'pred')

  # This grabs RMSE, Rsquaredand MAE by default
  model_eval <- defaultSummary(model_results)

  # Add AIC score to the results
  if ('aic' %in% model) {
    model_eval[4] <- model$aic
  } else {
    model_eval[4] <- AIC(model)
  }

  names(model_eval)[4] <- 'aic'

  # Add BIC score to the results
  model_eval[5] <- BIC(model)
  names(model_eval)[5] <- 'bic'


  model_eval[6] <- paste0(deparse(substitute(model)))
  names(model_eval)[6] <- "model"

  return(model_eval)}
```

```
trainY <- wine_test %>% dplyr::select(TARGET)

models = list(prmodel1, prmodel2, nbrm1, nbrm2,mlr1,mlr2)


prmodel1_eval = model_test(prmodel1, wine_test, trainY)
prmodel2_eval = model_test(prmodel2, wine_test, trainY)
nbrm1_eval= model_test(nbrm1, wine_test, trainY)
nbrm2_eval= model_test(nbrm2, wine_test, trainY)
mlr1_eval= model_test(mlr1, wine_test, trainY)
mlr2_eval= model_test(mlr2, wine_test, trainY)

models_summary <- rbind(prmodel1_eval, prmodel2_eval, nbrm1_eval, nbrm2_eval, mlr1_eval,mlr2_eval)
kable(models_summary) %>%
  kable_styling(bootstrap_options = "basic", position = "center")
```

```
models_summary
```

```
##                 RMSE                Rsquared              MAE
## prmodel1_eval "2.59206392755416" "0.523083117977806" "2.25869826962797"
## prmodel2_eval "2.59146396280063" "0.524553446289377" "2.25909608477653"
## nbrm1_eval    "2.59206300547322" "0.523082810276954" "2.25869629078798"
## nbrm2_eval    "2.59206300547322" "0.523082810276954" "2.25869629078798"
## mlr1_eval     "1.32741201248383" "0.524762584796396" "1.0625722003606"
## mlr2_eval     "1.3255535618419"  "0.526080896568868" "1.0615300940943"
##               aic                bic                model
## prmodel1_eval "37355.3208660464" "37463.8273243502" "prmodel1"
## prmodel2_eval "37385.8401566517" "37422.0089760863" "prmodel2"
## nbrm1_eval    "37357.5433390294" "37473.2835612201" "nbrm1"
## nbrm2_eval    "37357.5433390294" "37473.2835612201" "nbrm2"
## mlr1_eval     "34807.8425223109" "34923.5827445016" "mlr1"
## mlr2_eval     "34865.1042764434" "34908.5068597649" "mlr2"
```

This table showcases the RMSE, R2, MAE, AIC and BIC for the six models. The Linear regressions, mlr1 and mlr2, had the best performances based on RMSE and R2.. Also, mlr1 had the best aic and mlr2 had the best bic.

Both RMSE an R2 were not significantly different across the 6 models, so we chose MLR 1 as our final model since it had the lowest AIC.

Top Model Evaluation

```
eval_data <- df_eval %>% dplyr::select(-TARGET)
predictions <- predict(mlr1, eval_data)

eval_data$TARGET <- predictions

write.csv(eval_data, 'eval_predictions.csv', row.names=FALSE)

head(eval_data)
```

```
## # A tibble: 6 x 15
##   FixedAcidity VolatileAcidity CitricAcid ResidualSugar Chlorides
##          <dbl>           <dbl>      <dbl>         <dbl>     <dbl>
## 1          5.4          -0.86        0.27         -10.7     0.092
## 2         12.4           0.385      -0.76         -19.7     1.17
## 3          7.2           1.75        0.17         -33       0.065
## 4          6.2           0.1         1.8            1      -0.179
## 5         11.4           0.21        0.28           1.2     0.038
## 6         17.6           0.04       -1.15           1.4     0.535
## # ... with 10 more variables: FreeSulfurDioxide <dbl>,
## #   TotalSulfurDioxide <dbl>, Density <dbl>, pH <dbl>, Sulphates <dbl>,
## #   Alcohol <dbl>, LabelAppeal <dbl>, AcidIndex <dbl>, STARS <int>,
## #   TARGET <dbl>
```