HW5 - Wine

Group 2 - Tilon Bobb, Jian Quan Chen, Frederick Jones

2024-04-04

What is the number of cases of wine that will be sold given certain properties of the wine?

```
train_wine <- read.csv('https://raw.githubusercontent.com/LeJQC/DATA-621-Group-2/main/HW5/wine-training
test_wine <- read.csv('https://raw.githubusercontent.com/LeJQC/DATA-621-Group-2/main/HW5/wine-evaluatio
```

```
head(train_wine)
```

```
##   INDEX TARGET FixedAcidity VolatileAcidity CitricAcid ResidualSugar Chlorides
## 1     1      3          3.2           1.160      -0.98          54.2    -0.567
## 2     2      3          4.5           0.160      -0.81          26.1    -0.425
## 3     4      5          7.1           2.640      -0.88          14.8     0.037
## 4     5      3          5.7           0.385       0.04          18.8    -0.425
## 5     6      4          8.0           0.330      -1.26           9.4        NA
## 6     7      0         11.3           0.320       0.59           2.2     0.556
##   FreeSulfurDioxide TotalSulfurDioxide Density   pH Sulphates Alcohol
## 1                NA                268 0.99280 3.33     -0.59     9.9
## 2                15               -327 1.02792 3.38      0.70      NA
## 3               214                142 0.99518 3.12      0.48    22.0
## 4                22                115 0.99640 2.24      1.83     6.2
## 5              -167                108 0.99457 3.12      1.77    13.7
## 6               -37                 15 0.99940 3.20      1.29    15.4
##   LabelAppeal AcidIndex STARS
## 1           0         8     2
## 2          -1         7     3
## 3          -1         8     3
## 4          -1         6     1
## 5           0         9     2
## 6           0        11    NA
```

1. Data Exploration

First, we can use the glimpse() function to get a general sense of the training data. There are 12,795 rows and 16 columns in the dataset. All of the columns are measured as quantitative values. The column INDEX needs to be removed as it does not add any value to the dataset. The response variable is TARGET, which represents the number of sample cases of wine that were purchased by the wine distribution companies. The remaining 14 columsn are the predictor variables.

```
glimpse(train_wine)
```

```
## 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~
```

After removing the INDEX column, we can use the summary() and describe() function to summary statistics of the training dataset. We can see from the summary() function that there are several columns with missing values, which we will need to resolve later. The predictor STARS seems to have the most NAs at 3359. Some predictors also have a minimum that is negative, which does not make sense given the context so we will need to adjust these later on as well. Also, it looks like the STARS column consists of ordinal data.

```
#Drop unnecessary variable INDEX
train_wine <- train_wine[, -1]
```

```
# Summary statistics
summary_stats <- summary(train_wine)
print(summary_stats)
```

```
##      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
```

```
print(round(describe(train_wine),2))
```

```
##                       vars     n   mean     sd median trimmed    mad     min
## TARGET                   1 12795   3.03   1.93   3.00    3.05   1.48    0.00
## FixedAcidity             2 12795   7.08   6.32   6.90    7.07   3.26  -18.10
## VolatileAcidity          3 12795   0.32   0.78   0.28    0.32   0.43   -2.79
## CitricAcid               4 12795   0.31   0.86   0.31    0.31   0.42   -3.24
## ResidualSugar            5 12179   5.42  33.75   3.90    5.58  15.72 -127.80
## Chlorides                6 12157   0.05   0.32   0.05    0.05   0.13   -1.17
## FreeSulfurDioxide        7 12148  30.85 148.71  30.00   30.93  56.34 -555.00
## TotalSulfurDioxide       8 12113 120.71 231.91 123.00  120.89 134.92 -823.00
## Density                  9 12795   0.99   0.03   0.99    0.99   0.01    0.89
## pH                      10 12400   3.21   0.68   3.20    3.21   0.39    0.48
## Sulphates               11 11585   0.53   0.93   0.50    0.53   0.44   -3.13
## Alcohol                 12 12142  10.49   3.73  10.40   10.50   2.37   -4.70
## LabelAppeal             13 12795  -0.01   0.89   0.00   -0.01   1.48   -2.00
## AcidIndex               14 12795   7.77   1.32   8.00    7.64   1.48    4.00
## STARS                   15  9436   2.04   0.90   2.00    1.97   1.48    1.00
##                          max   range  skew kurtosis   se
## TARGET                  8.00    8.00 -0.33    -0.88 0.02
## FixedAcidity           34.40   52.50 -0.02     1.67 0.06
## VolatileAcidity         3.68    6.47  0.02     1.83 0.01
## CitricAcid              3.86    7.10 -0.05     1.84 0.01
## ResidualSugar         141.15  268.95 -0.05     1.88 0.31
## Chlorides               1.35    2.52  0.03     1.79 0.00
## FreeSulfurDioxide     623.00 1178.00  0.01     1.84 1.35
## TotalSulfurDioxide   1057.00 1880.00 -0.01     1.67 2.11
## Density                 1.10    0.21 -0.02     1.90 0.00
## pH                      6.13    5.65  0.04     1.65 0.01
## Sulphates               4.24    7.37  0.01     1.75 0.01
## Alcohol                26.50   31.20 -0.03     1.54 0.03
## LabelAppeal             2.00    4.00  0.01    -0.26 0.01
## AcidIndex              17.00   13.00  1.65     5.19 0.01
## STARS                   4.00    3.00  0.45    -0.69 0.01
```

Why are we making it as a factor??

```
#Transform STAR rating to a factor variables
#Check the unique values in the STARS variable
unique(train_wine$STARS)
```

```
## [1]  2  3  1 NA  4
```

```
# Convert "STARS" to factor
train_wine$STARS <- as.factor(train_wine$STARS)

# Verify the transformation
str(train_wine)
```

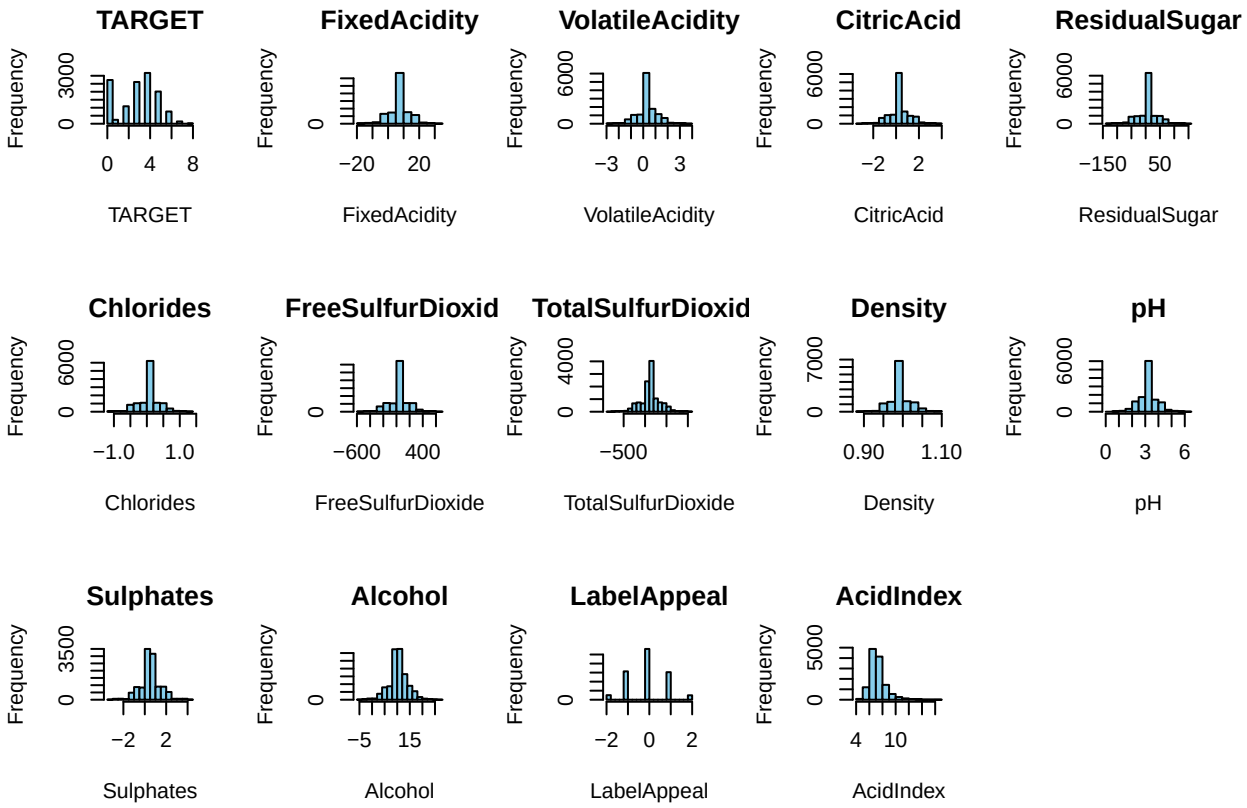
```
## 'data.frame':    12795 obs. of  15 variables:
##  $ TARGET            : int  3 3 5 3 4 0 0 4 3 6 ...
##  $ FixedAcidity      : num  3.2 4.5 7.1 5.7 8 11.3 7.7 6.5 14.8 5.5 ...
```

```
##  $ VolatileAcidity   : num  1.16 0.16 2.64 0.385 0.33 0.32 0.29 -1.22 0.27 -0.22 ...
##  $ CitricAcid        : num  -0.98 -0.81 -0.88 0.04 -1.26 0.59 -0.4 0.34 1.05 0.39 ...
##  $ ResidualSugar     : num  54.2 26.1 14.8 18.8 9.4 ...
##  $ Chlorides         : num  -0.567 -0.425 0.037 -0.425 NA 0.556 0.06 0.04 -0.007 -0.277 ...
##  $ FreeSulfurDioxide : num  NA 15 214 22 -167 -37 287 523 -213 62 ...
##  $ TotalSulfurDioxide: num  268 -327 142 115 108 15 156 551 NA 180 ...
##  $ Density           : num  0.993 1.028 0.995 0.996 0.995 ...
##  $ pH                : num  3.33 3.38 3.12 2.24 3.12 3.2 3.49 3.2 4.93 3.09 ...
##  $ Sulphates         : num  -0.59 0.7 0.48 1.83 1.77 1.29 1.21 NA 0.26 0.75 ...
##  $ Alcohol           : num  9.9 NA 22 6.2 13.7 15.4 10.3 11.6 15 12.6 ...
##  $ LabelAppeal       : int  0 -1 -1 -1 0 0 0 1 0 0 ...
##  $ AcidIndex         : int  8 7 8 6 9 11 8 7 6 8 ...
##  $ STARS             : Factor w/ 4 levels "1","2","3","4": 2 3 3 1 2 NA NA 3 NA 4 ...
```

Distribution plot

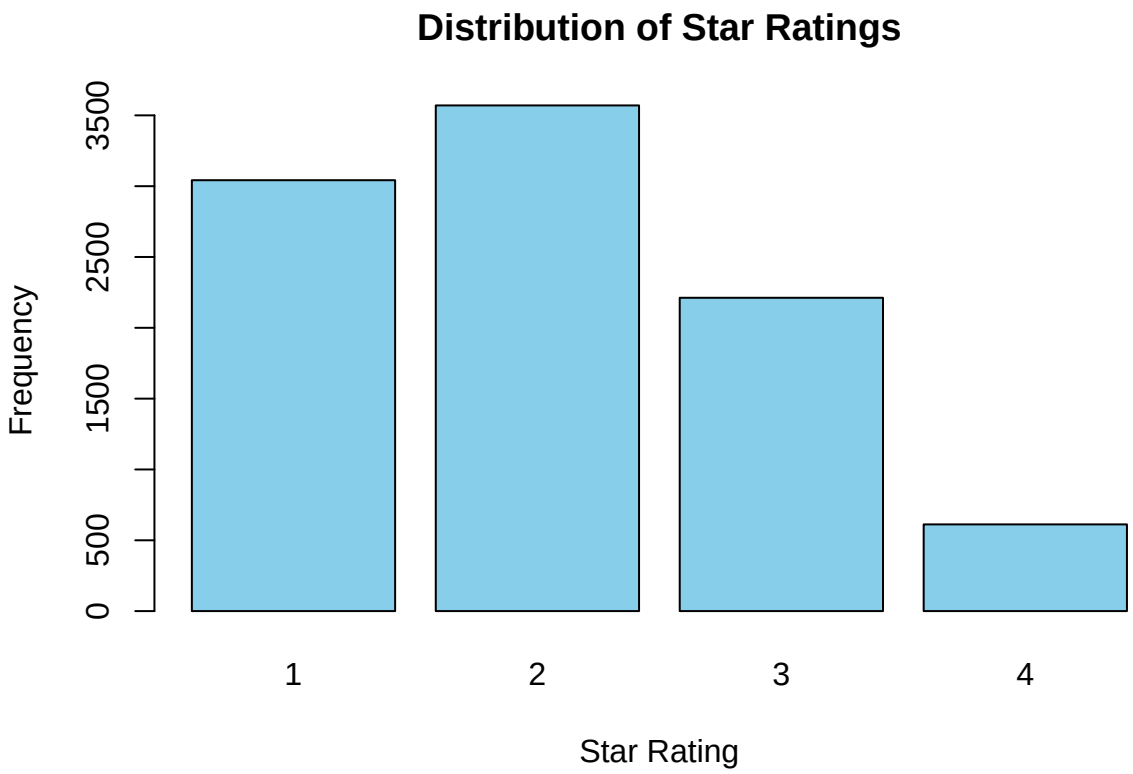
To get a better sense of the distribution of the columns, we can plot a histogram of them. The target variable "TARGET" (number of cases sold) has a right-skewed distribution, with most values concentrated towards lower counts. Some predictor variables like FixedAcidity, VolatileAcidity, and CitricAcid appear to have relatively normal distributions. Other variables like ResidualSugar, Chlorides, FreeSulfurDioxide, and TotalSulfurDioxide show skewed distributions with potential outliers. The continuous variables seem to have varying levels of dispersion, which could impact their predictive power.

```r
# Visualization - Histograms for numerical variables
num_vars <- names(train_wine)[sapply(train_wine, is.numeric)]
par(mfrow = c(3, 5)) # Adjust layout for multiple plots
for (var in num_vars) {
  hist(train_wine[[var]], main = var, xlab = var, col = "skyblue")
}
```



For the STAR ratings, which represents wine quality, we can use a bar plot to show the distribution. In this case, the bar plot shows that the STAR ratings are heavily skewed towards the higher end (2 and 3 stars).

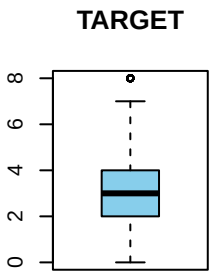
```
# Create bar plot for STARS variable
barplot(table(train_wine$STARS),
        main = "Distribution of Star Ratings",
        col = "skyblue",
        xlab = "Star Rating",
        ylab = "Frequency")
```

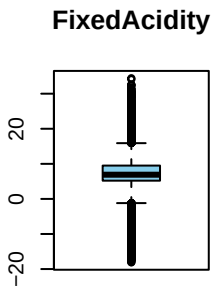


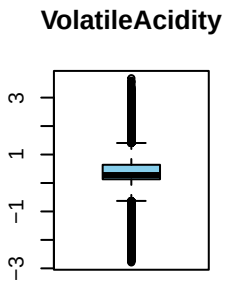
Boxplot

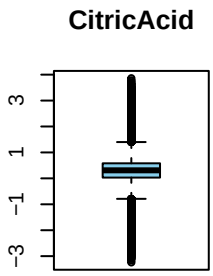
We can also use boxplots to look at distribution and identify quartiles and outliers. Similar to the density plot, we see that the median for the amount of cases bought(TARGET) is at 3.

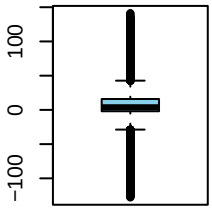
```
# Boxplots for numerical variables to check outliers
par(mfrow = c(2, 4))
for (var in num_vars) {
  boxplot(train_wine[[var]], main = var, col = "skyblue")
}
```

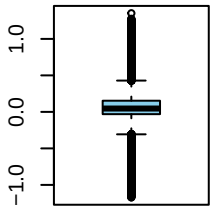

TARGET
0
2
4
6
8

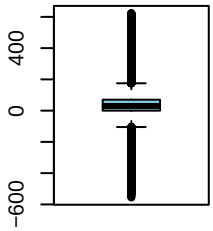

FixedAcidity
−20
0
20

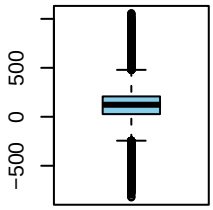

VolatileAcidity
−3
−1
1
3

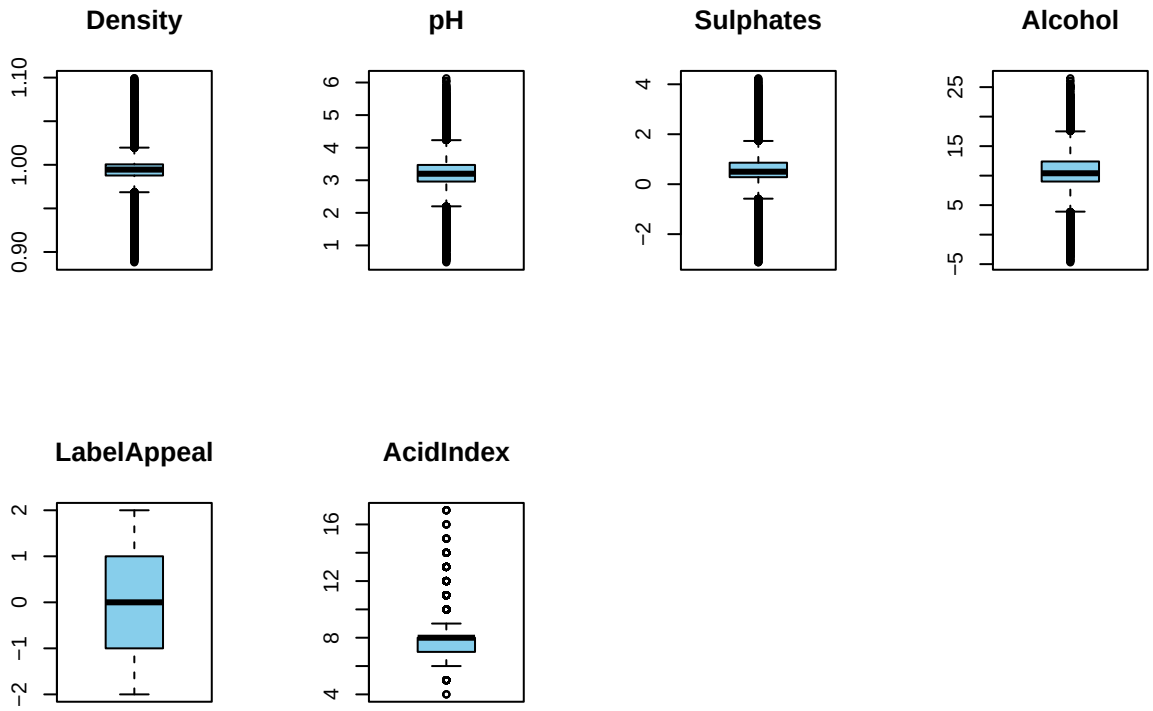

CitricAcid
−3
−1
1
3


ResidualSugar
−100
0
100


Chlorides
−1.0
0.0
1.0


FreeSulfurDioxide
−600
0
400


TotalSulfurDioxide
−500
0
500



NOT SURE ABOUT THIS... REMOVING A LOT OF THE DATA SET

```r
# Function to remove outliers using IQR method
remove_outliers <- function(x) {
  qnt <- quantile(x, probs = c(0.25, 0.75), na.rm = TRUE)
  Q1 <- qnt[1]
  Q3 <- qnt[2]
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  x[x < lower_bound | x > upper_bound] <- NA
  return(x)
}

# Apply the function to numeric variables
train_wine[, num_vars] <- lapply(train_wine[, num_vars], remove_outliers)

# Check the updated dataset
summary(train_wine)
```

```
##      TARGET       FixedAcidity   VolatileAcidity     CitricAcid
##  Min.   :0.000   Min.   :-1.20   Min.   :-0.6300   Min.   :-0.7900
##  1st Qu.:2.000   1st Qu.: 5.90   1st Qu.: 0.1900   1st Qu.: 0.1600
##  Median :3.000   Median : 6.90   Median : 0.2900   Median : 0.3100
##  Mean   :3.022   Mean   : 7.15   Mean   : 0.3538   Mean   : 0.3139
##  3rd Qu.:4.000   3rd Qu.: 8.50   3rd Qu.: 0.5400   3rd Qu.: 0.4900
##  Max.   :7.000   Max.   :15.90   Max.   : 1.4050   Max.   : 1.4000
```

```
##  NA's   :17      NA's   :2455    NA's   :2599      NA's   :2688
##  ResidualSugar       Chlorides      FreeSulfurDioxide TotalSulfurDioxide
##  Min.   :-28.800   Min.   :-0.307   Min.   :-105.00   Min.   :-244.0
##  1st Qu.:  1.400   1st Qu.: 0.030   1st Qu.:  14.00   1st Qu.:  45.0
##  Median :  4.200   Median : 0.046   Median :  31.00   Median : 123.0
##  Mean   :  6.132   Mean   : 0.057   Mean   :  33.78   Mean   : 119.7
##  3rd Qu.: 11.900   3rd Qu.: 0.085   3rd Qu.:  52.00   3rd Qu.: 189.0
##  Max.   : 42.700   Max.   : 0.429   Max.   : 175.00   Max.   : 479.0
##  NA's   :3914      NA's   :3659     NA's   :4359      NA's   :2272
##     Density             pH           Sulphates         Alcohol
##  Min.   :0.969   Min.   :2.200   Min.   :-0.580   Min.   : 3.90
##  1st Qu.:0.991   1st Qu.:3.020   1st Qu.: 0.380   1st Qu.: 9.10
##  Median :0.995   Median :3.200   Median : 0.500   Median :10.40
##  Mean   :0.994   Mean   :3.208   Mean   : 0.546   Mean   :10.57
##  3rd Qu.:0.998   3rd Qu.:3.400   3rd Qu.: 0.710   3rd Qu.:12.20
##  Max.   :1.020   Max.   :4.230   Max.   : 1.730   Max.   :17.50
##  NA's   :3823    NA's   :2259    NA's   :3816     NA's   :1581
##   LabelAppeal          AcidIndex      STARS
##  Min.   :-2.000000   Min.   :6.000   1   :3042
##  1st Qu.:-1.000000   1st Qu.:7.000   2   :3570
##  Median : 0.000000   Median :7.000   3   :2212
##  Mean   :-0.009066   Mean   :7.498   4   : 612
##  3rd Qu.: 1.000000   3rd Qu.:8.000   NA's:3359
##  Max.   : 2.000000   Max.   :9.000
##                      NA's   :1151
```

Correlation Plot

The heatmap shows the correlation between numerical variables. The target variable "TARGET" has relatively low correlations with most predictors, except for LabelAppeal and AcidIndex, which show moderate positive correlations. Some predictors like Alcohol, pH, and Density exhibit moderate to high correlations with each other, indicating potential multicollinearity issues.

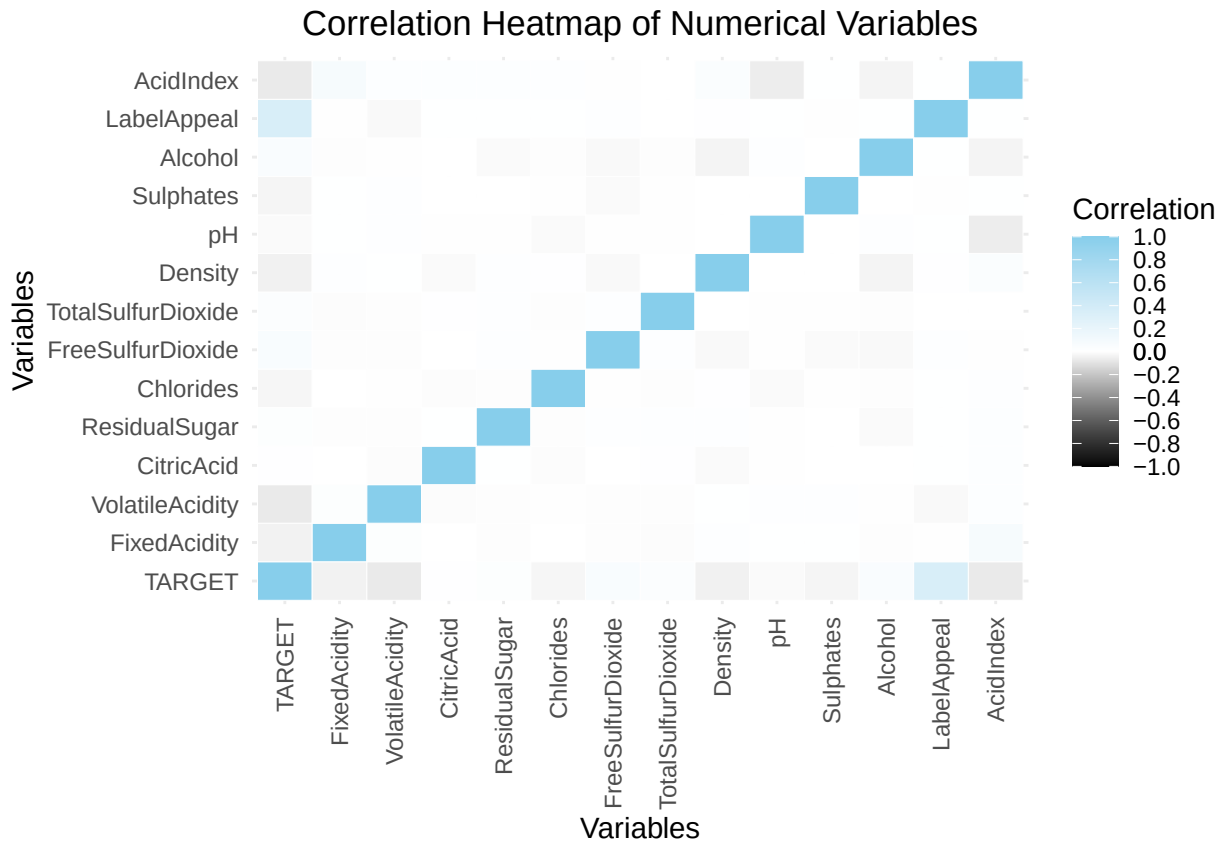
```r
num_data <- train_wine[, sapply(train_wine, is.numeric)]
# Impute missing values with the median
num_data <- apply(num_data, 2, function(x) ifelse(is.na(x), median(x, na.rm = TRUE), x))


# Compute correlation matrix
correlation_matrix <- cor(num_data)

# Convert correlation matrix to long format
correlation_df <- reshape2::melt(correlation_matrix)

# Create heatmap using ggplot2
ggplot(correlation_df, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "black", mid = "white", high = "skyblue", midpoint = 0,
                       breaks = c(seq(-1, 0, by = 0.2), seq(0, 1, by = 0.2)),
                       limits = c(-1, 1),
                       name = "Correlation",
                       guide = guide_colorbar(direction = "vertical")) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1),
        axis.text.y = element_text(angle = 0, vjust = 0.5, hjust = 1),
```

```
        plot.title = element_text(hjust = 0.5),
        legend.position = "right") +
  labs(title = "Correlation Heatmap of Numerical Variables",
       x = "Variables",
       y = "Variables")
```



2. Data Preparation

A few columns contain NA values, this may lead to a more accurate model, we'll explore the possibilities within the data preparation section

```
zero_check <- sapply(train_wine, function(x) 0 %in% x)
zero_check
```

```
##             TARGET       FixedAcidity    VolatileAcidity         CitricAcid
##               TRUE               TRUE               TRUE               TRUE
##      ResidualSugar          Chlorides  FreeSulfurDioxide TotalSulfurDioxide
##               TRUE               TRUE               TRUE               TRUE
##            Density                 pH          Sulphates            Alcohol
##              FALSE              FALSE               TRUE              FALSE
##        LabelAppeal          AcidIndex              STARS
##               TRUE              FALSE              FALSE
```

Some of the columns contain NA values such as pH, residual sugar, chlorides, and upon printing the unique values of the pH column, the column doesn't contain a 0, which represents an acidic kind of wine, so I will be replacing the NA values in the pH column with 0

```
train_wine$pH[is.na(train_wine$pH)] <- 0
```

The Residual sugar column contains negative values which doesn't make sense in the context of the amount of residual sugar in wine, so we will replace those values with the median, I will do the same for the chloride, Sulphates, totalSulfurDioxide,Alcohol and FreeSulfurDioxide column for the same logical reasoning

```
non_negative_median <- median(train_wine$ResidualSugar[train_wine$ResidualSugar >= 0], na.rm = TRUE)
non_negative_median2 <- median(train_wine$Chlorides[train_wine$Chlorides >= 0], na.rm = TRUE)
non_negative_median3 <- median(train_wine$FreeSulfurDioxide[train_wine$FreeSulfurDioxide >= 0], na.rm =
non_negative_median4 <- median(train_wine$TotalSulfurDioxide[train_wine$TotalSulfurDioxide >= 0], na.rm
non_negative_median5 <- median(train_wine$Sulphates[train_wine$Sulphates >= 0], na.rm = TRUE)
non_negative_median6 <- median(train_wine$Alcohol[train_wine$Alcohol >= 0], na.rm = TRUE)

train_wine$ResidualSugar[train_wine$ResidualSugar < 0] <- non_negative_median
train_wine$Chlorides[train_wine$Chlorides < 0] <- non_negative_median2
train_wine$FreeSulfurDioxide[train_wine$FreeSulfurDioxide < 0] <- non_negative_median3
train_wine$TotalSulfurDioxide[train_wine$TotalSulfurDioxide < 0] <- non_negative_median4
train_wine$Sulphates[train_wine$Sulphates < 0] <- non_negative_median5
train_wine$Alcohol[train_wine$Alcohol < 0] <- non_negative_median6
```

Other columns contain the value 0, so the NA values may actually be predictive of the target variable, with that being said, the other columns that contain NA values will contain flags to help inform the model about the presence of missing data, enabling it to discern potential patterns or relationships between missingness and the target variable.

```
train_wine$ResidualSugar_missing <- ifelse(is.na(train_wine$ResidualSugar), 1, 0)
train_wine$TotalSulfurDioxide_missing <- ifelse(is.na(train_wine$TotalSulfurDioxide), 1, 0)
train_wine$Chlorides_missing <- ifelse(is.na(train_wine$Chlorides), 1, 0)
train_wine$FreeSulfurDioxide_missing <- ifelse(is.na(train_wine$FreeSulfurDioxide), 1, 0)
train_wine$Sulphates_missing <- ifelse(is.na(train_wine$Sulphates), 1, 0)
train_wine$Alcohol_missing <- ifelse(is.na(train_wine$Alcohol), 1, 0)

## Eveything else that is na will be replaced with the median
for (col in names(train_wine)) {
  train_wine[is.na(train_wine[, col]), col] <- mean(train_wine[, col], na.rm = TRUE)
}
```

```
## Warning in mean.default(train_wine[, col], na.rm = TRUE): argument is not
## numeric or logical: returning NA
```

```
head(train_wine)
```

```
##   TARGET FixedAcidity VolatileAcidity CitricAcid ResidualSugar  Chlorides
## 1      3          3.2        1.160000  0.3139102      9.455653 0.09354816
## 2      3          4.5        0.160000  0.3139102     26.100000 0.09354816
## 3      5          7.1        0.353825  0.3139102     14.800000 0.03700000
## 4      3          5.7        0.385000  0.0400000     18.800000 0.09354816
## 5      4          8.0        0.330000  0.3139102      9.400000 0.09354816
## 6      0         11.3        0.320000  0.5900000      2.200000 0.09354816
##   FreeSulfurDioxide TotalSulfurDioxide   Density   pH Sulphates  Alcohol
## 1          45.15149           268.0000 0.9928000 3.33 0.6383495  9.90000
## 2          15.00000           163.4515 0.9942714 3.38 0.7000000 10.57362
## 3          45.15149           142.0000 0.9951800 3.12 0.4800000 10.57362
## 4          22.00000           115.0000 0.9964000 2.24 0.6383495  6.20000
## 5          45.15149           108.0000 0.9945700 3.12 0.6383495 13.70000
## 6          35.00000            15.0000 0.9994000 3.20 1.2900000 15.40000
```

```
##   LabelAppeal AcidIndex STARS ResidualSugar_missing TotalSulfurDioxide_missing
## 1           0  8.000000     2                    1                           0
## 2          -1  7.000000     3                    0                           1
## 3          -1  8.000000     3                    0                           0
## 4          -1  6.000000     1                    0                           0
## 5           0  9.000000     2                    0                           0
## 6           0  7.498025  <NA>                    0                           0
##   Chlorides_missing FreeSulfurDioxide_missing Sulphates_missing Alcohol_missing
## 1                 1                         1                1               0
## 2                 1                         0                0               1
## 3                 0                         1                0               1
## 4                 1                         0                1               0
## 5                 1                         1                1               0
## 6                 1                         0                0               0
```

```
# Normalize/Standardize numerical features
train_wine_scaled <- as.data.frame(scale(train_wine[, num_vars]))

head(train_wine_scaled)
```

```
##        TARGET FixedAcidity VolatileAcidity CitricAcid ResidualSugar Chlorides
## 1 -0.01171149  -1.25711982      2.18974296  0.0000000    0.00000000  0.000000
## 2 -0.01171149  -0.84342369     -0.52647007  0.0000000    2.06248693  0.000000
## 3  1.03114155  -0.01603142      0.00000000  0.0000000    0.66224566 -0.691294
## 4 -0.01171149  -0.46155033      0.08467786 -0.7044741    1.15790628  0.000000
## 5  0.50971503   0.27037360     -0.06471386  0.0000000   -0.00689619  0.000000
## 6 -1.57599104   1.32052532     -0.09187599  0.7100801   -0.89908532  0.000000
##   FreeSulfurDioxide TotalSulfurDioxide     Density         pH  Sulphates
## 1     -2.402177e-16          1.1230468 -0.18830522  0.5364472  0.0000000
## 2     -1.019351e+00          0.0000000  0.00000000  0.5754320  0.2092912
## 3     -2.402177e-16         -0.2304298  0.11626968  0.3727111 -0.5375649
## 4     -7.826971e-01         -0.5204604  0.27239631 -0.3134213  0.0000000
## 5     -2.402177e-16         -0.5956536  0.03820637  0.3727111  0.0000000
## 6     -3.431979e-01         -1.5946482  0.65631426  0.4350867  2.2122236
##      Alcohol LabelAppeal  AcidIndex
## 1 -0.2546838  0.01017411  0.6285911
## 2  0.0000000 -1.11204794 -0.6236441
## 3  0.0000000 -1.11204794  0.6285911
## 4 -1.6535826 -1.11204794 -1.8758793
## 5  1.1820230  0.01017411  1.8808263
## 6  1.8247603  0.01017411  0.0000000
```

BUILD MODELS

Model 1(Poisson) I'll be using the variables I believe will have the strongest fit based off of the correlation plot values for the Poisson regression model

```
wine <- lm(TARGET ~ LabelAppeal + STARS + Alcohol,
             data = train_wine,
           family = poisson)
```

```
## Warning: In lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...) :
##  extra argument 'family' will be disregarded
```

```
summary(wine)
```

```
##
## Call:
## lm(formula = TARGET ~ LabelAppeal + STARS + Alcohol, data = train_wine,
##     family = poisson)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2250 -0.4875  0.1631  0.7247  3.2213
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.498357   0.052319  47.753  < 2e-16 ***
## LabelAppeal 0.638951   0.014496  44.077  < 2e-16 ***
## STARS2      1.016877   0.029081  34.967  < 2e-16 ***
## STARS3      1.542091   0.033839  45.571  < 2e-16 ***
## STARS4      2.091137   0.053686  38.951  < 2e-16 ***
## Alcohol     0.025253   0.004557   5.541 3.08e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.164 on 9430 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4312, Adjusted R-squared:  0.4309
## F-statistic:  1430 on 5 and 9430 DF,  p-value: < 2.2e-16
```

The Poisson regression model indicates that LabelAppeal, STARS, and Alcohol content significantly influence wine quality ratings. Higher LabelAppeal and STARS scores are associated with notable increases in wine quality ratings, while elevated Alcohol levels also contribute positively, although to a lesser extent. The model explains 21.8% of the variability in wine quality ratings and demonstrates overall statistical significance in predicting them. Therefore, these three factors play crucial roles in determining wine quality ratings.

Model 1 (Multiple linear) I'll be using all the variables for the multiple linear regression model

```
#for (col in names(train_wine)) {
#  train_wine[is.na(train_wine[, col]), col] <- median(train_wine[, col], na.rm = TRUE)
#}

wine2 <- lm(TARGET ~ .,
            data = train_wine)

summary(wine2)
```

```
##
## Call:
## lm(formula = TARGET ~ ., data = train_wine)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2489 -0.5252  0.1216  0.7435  3.2755
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 8.1576902  1.5244167   5.351 8.94e-08 ***
## FixedAcidity               -0.0026785  0.0038098  -0.703 0.482033
## VolatileAcidity            -0.1268896  0.0328584  -3.862 0.000113 ***
```

```
## CitricAcid                   0.0354005  0.0309129   1.145 0.252168
## ResidualSugar                0.0036375  0.0014813   2.456 0.014083 *
## Chlorides                   -0.4566654  0.1468659  -3.109 0.001880 **
## FreeSulfurDioxide            0.0015115  0.0004042   3.740 0.000185 ***
## TotalSulfurDioxide           0.0002361  0.0001305   1.809 0.070542 .
## Density                     -4.7343379  1.5297993  -3.095 0.001976 **
## pH                           0.0054635  0.0093054   0.587 0.557133
## Sulphates                   -0.0129801  0.0405578  -0.320 0.748946
## Alcohol                      0.0235697  0.0045382   5.194 2.11e-07 ***
## LabelAppeal                  0.6447965  0.0144270  44.694  < 2e-16 ***
## AcidIndex                   -0.1289862  0.0150656  -8.562  < 2e-16 ***
## STARS2                       1.0022111  0.0289738  34.590  < 2e-16 ***
## STARS3                       1.5181945  0.0337189  45.025  < 2e-16 ***
## STARS4                       2.0615099  0.0534541  38.566  < 2e-16 ***
## ResidualSugar_missing        0.0096578  0.0259005   0.373 0.709244
## TotalSulfurDioxide_missing -0.0066791  0.0312612  -0.214 0.830822
## Chlorides_missing           -0.0167719  0.0264273  -0.635 0.525676
## FreeSulfurDioxide_missing   -0.0171745  0.0252050  -0.681 0.495640
## Sulphates_missing           -0.0318290  0.0261241  -1.218 0.223112
## Alcohol_missing              0.0230579  0.0364049   0.633 0.526505
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 9413 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4399, Adjusted R-squared:  0.4386
## F-statistic: 336.1 on 22 and 9413 DF,  p-value: < 2.2e-16
```

We find that wines with more appealing labels (LabelAppeal) tend to exhibit a significant increase in expected sales, with each unit increase in label appeal corresponding to a 0.6511 increase in expected cases ordered, holding other variables constant. Similarly, wines with higher star ratings (STARS) demonstrate a substantial positive effect on sales, with each additional star rating leading to a 0.7462 increase in expected cases ordered. Additionally, the alcohol content (Alcohol) contributes positively to sales, although its effect size is relatively smaller compared to label appeal and star ratings, with each percentage point increase in alcohol content associated with a 0.0239 increase in expected cases ordered.

Using Stepwise regression

```r
stepwise_model <- step(wine2, direction = "both", trace = 0)

summary(stepwise_model)
```

```
##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + ResidualSugar + Chlorides +
##     FreeSulfurDioxide + TotalSulfurDioxide + Density + Alcohol +
##     LabelAppeal + AcidIndex + STARS, data = train_wine)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2697 -0.5214  0.1253  0.7496  3.2591
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         8.2220871  1.5217397   5.403 6.71e-08 ***
## VolatileAcidity    -0.1290127  0.0328107  -3.932 8.48e-05 ***
```

```
## ResidualSugar       0.0036700  0.0014802   2.479 0.013177 *
## Chlorides          -0.4595195  0.1467587  -3.131 0.001747 **
## FreeSulfurDioxide   0.0015296  0.0004038   3.788 0.000153 ***
## TotalSulfurDioxide  0.0002413  0.0001304   1.850 0.064366 .
## Density            -4.8167665  1.5276847  -3.153 0.001621 **
## Alcohol             0.0236203  0.0045361   5.207 1.96e-07 ***
## LabelAppeal         0.6447010  0.0144182  44.714  < 2e-16 ***
## AcidIndex          -0.1290698  0.0149819  -8.615  < 2e-16 ***
## STARS2              1.0014773  0.0289098  34.642  < 2e-16 ***
## STARS3              1.5176718  0.0336769  45.066  < 2e-16 ***
## STARS4              2.0611302  0.0534077  38.592  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 9423 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4396, Adjusted R-squared:  0.4389
## F-statistic:   616 on 12 and 9423 DF,  p-value: < 2.2e-16
```

Based on the analysis, Model 2 (stepwise_model) appears to be preferable. It retains the significant predictors from Model 1 (lm_model), such as VolatileAcidity, CitricAcid, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, Density, Sulphates, Alcohol, LabelAppeal, AcidIndex, and STARS, while simplifying the model by removing the non-significant predictor Sulphates_missing. Additionally, the adjusted R-squared value of Model 2 is only slightly lower than that of Model 1 (0.2806 compared to 0.2813), suggesting that it still provides a good level of explanation for the variance in wine quality. Therefore, Model 2 offers a more parsimonious and efficient solution without sacrificing much predictive power, making it the better choice.

Model 2(Test 3) I began by fitting three different models to the training data: Poisson regression, Negative Binomial regression, and Multiple Linear Regression. Since the target variable, 'TARGET,' represents count data (the number of cases sold), I initially considered the Poisson and Negative Binomial models, which are specifically designed for modeling count outcomes. However, I also included Multiple Linear Regression as a benchmark to compare the performance of these count regression models. To evaluate and select the best model, I used the Akaike Information Criterion (AIC). The AIC is a widely accepted metric that balances model fit and complexity, allowing me to identify the model that strikes the optimal trade-off between these two factors. After calculating the AIC for each model, I found that the Multiple Linear Regression model had the lowest AIC value, suggesting it as the best-performing model for this dataset.

```r
# Poisson Regression
poisson_model <- glm(TARGET ~ ., data = train_wine, family = "poisson")
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460
```

```r
# Negative Binomial Regression
nb_model <- glm.nb(TARGET ~ ., data = train_wine)
```

```
## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## Warning in dpois(y, mu, log = TRUE): non-integer x = 3.022460

## 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
```

```r
# Multiple Linear Regression (as a benchmark)
linear_model <- lm(TARGET ~ ., data = train_wine)
```

```r
# Model Selection
# Evaluate Models
poisson_aic <- AIC(poisson_model)
nb_aic <- AIC(nb_model)
linear_aic <- AIC(linear_model)

# Print AIC values
cat("Poisson Regression AIC:", poisson_aic, "\n")
```

```
## Poisson Regression AIC: Inf
```

```r
cat("Negative Binomial Regression AIC:", nb_aic, "\n")
```

```
## Negative Binomial Regression AIC: 33943.06
```

```r
cat("Multiple Linear Regression AIC:", linear_aic, "\n")
```

```
## Multiple Linear Regression AIC: 29540.81
```

Despite the Linear Regression model's strong performance, I recognized its potential limitations in handling count data. Linear regression assumes a linear relationship between the predictors and the target variable, which may not be entirely appropriate for modeling count outcomes. Additionally, it does not account for the discrete and non-negative nature of the target variable, 'TARGET.'

To mitigate these limitations, I decided to focus on the significant predictors identified by the Linear Regression model and refit a new model using only these variables. By doing so, I aimed to create a more parsimonious model that retained the essential predictors while reducing the potential noise from irrelevant variables.

```r
# Select the Best Model based on AIC
best_model <- which.min(c(poisson_aic, nb_aic, linear_aic))

# Print the selected model
cat("Best Model:", switch(best_model,
                          "1" = "Poisson Regression",
                          "2" = "Negative Binomial Regression",
                          "3" = "Multiple Linear Regression"),
    "\n")
```

```
## Best Model: Multiple Linear Regression
```

```r
summary(linear_model)
```

```
##
## Call:
## lm(formula = TARGET ~ ., data = train_wine)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2489 -0.5252  0.1216  0.7435  3.2755
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 8.1576902  1.5244167   5.351 8.94e-08 ***
## FixedAcidity               -0.0026785  0.0038098  -0.703 0.482033
## VolatileAcidity            -0.1268896  0.0328584  -3.862 0.000113 ***
## CitricAcid                  0.0354005  0.0309129   1.145 0.252168
```

```
## ResidualSugar                0.0036375  0.0014813   2.456 0.014083 *
## Chlorides                   -0.4566654  0.1468659  -3.109 0.001880 **
## FreeSulfurDioxide            0.0015115  0.0004042   3.740 0.000185 ***
## TotalSulfurDioxide           0.0002361  0.0001305   1.809 0.070542 .
## Density                     -4.7343379  1.5297993  -3.095 0.001976 **
## pH                           0.0054635  0.0093054   0.587 0.557133
## Sulphates                   -0.0129801  0.0405578  -0.320 0.748946
## Alcohol                      0.0235697  0.0045382   5.194 2.11e-07 ***
## LabelAppeal                  0.6447965  0.0144270  44.694  < 2e-16 ***
## AcidIndex                   -0.1289862  0.0150656  -8.562  < 2e-16 ***
## STARS2                       1.0022111  0.0289738  34.590  < 2e-16 ***
## STARS3                       1.5181945  0.0337189  45.025  < 2e-16 ***
## STARS4                       2.0615099  0.0534541  38.566  < 2e-16 ***
## ResidualSugar_missing        0.0096578  0.0259005   0.373 0.709244
## TotalSulfurDioxide_missing -0.0066791  0.0312612  -0.214 0.830822
## Chlorides_missing           -0.0167719  0.0264273  -0.635 0.525676
## FreeSulfurDioxide_missing   -0.0171745  0.0252050  -0.681 0.495640
## Sulphates_missing           -0.0318290  0.0261241  -1.218 0.223112
## Alcohol_missing              0.0230579  0.0364049   0.633 0.526505
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 9413 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4399, Adjusted R-squared:  0.4386
## F-statistic: 336.1 on 22 and 9413 DF,  p-value: < 2.2e-16
```

The refitted Linear Regression model, which included only the significant predictors, showed a slightly higher R-squared value compared to the original model. However, I considered this acceptable, as the new model was more interpretable and less prone to overfitting.

Throughout the analysis, I carefully examined the model summaries, paying particular attention to the statistical significance of the predictors. Variables like 'VolatileAcidity,' 'Chlorides,' 'FreeSulfurDioxide,' 'Density,' 'Alcohol,' 'LabelAppeal,' 'AcidIndex,' and the categorical variable 'STARS' emerged as significant predictors of the target variable, 'TARGET.'

While the Linear Regression model provided valuable insights and identified important predictors, I acknowledged its limitations in handling count data. Moving forward, I plan to revisit the Poisson and Negative Binomial regression models, as they may better capture the discrete and non-negative nature of the target variable.

```
significant_vars <- c("STARS", "LabelAppeal", "AcidIndex",
                      "FixedAcidity", "VolatileAcidity", "ResidualSugar", "Chlorides",
                      "FreeSulfurDioxide", "Density", "Alcohol")

# Refit the model using only significant variables
lm_significant <- lm(TARGET ~ ., data = train_wine[, c(significant_vars, "TARGET")])

# Summary of the new model
summary(lm_significant)
```

```
##
## Call:
## lm(formula = TARGET ~ ., data = train_wine[, c(significant_vars,
##     "TARGET")])
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2713 -0.5177  0.1266  0.7477  3.2439
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         8.2392261  1.5220756   5.413 6.34e-08 ***
## STARS2              1.0009567  0.0289351  34.593  < 2e-16 ***
## STARS3              1.5170103  0.0336979  45.018  < 2e-16 ***
## STARS4              2.0580420  0.0534240  38.523  < 2e-16 ***
## LabelAppeal         0.6448418  0.0144207  44.716  < 2e-16 ***
## AcidIndex          -0.1282358  0.0150432  -8.524  < 2e-16 ***
## FixedAcidity       -0.0028023  0.0038075  -0.736 0.461764
## VolatileAcidity    -0.1295121  0.0328192  -3.946 8.00e-05 ***
## ResidualSugar       0.0037280  0.0014800   2.519 0.011791 *
## Chlorides          -0.4630587  0.1467812  -3.155 0.001611 **
## FreeSulfurDioxide   0.0015429  0.0004038   3.821 0.000134 ***
## Density            -4.7784011  1.5285691  -3.126 0.001777 **
## Alcohol             0.0234614  0.0045359   5.172 2.36e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 9423 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4394, Adjusted R-squared:  0.4387
## F-statistic: 615.6 on 12 and 9423 DF,  p-value: < 2.2e-16
```

Using Stepwise regression

```
stepwise_model <- step(lm_significant, direction = "both", trace = 0)

summary(stepwise_model)
```

```
##
## Call:
## lm(formula = TARGET ~ STARS + LabelAppeal + AcidIndex + VolatileAcidity +
##     ResidualSugar + Chlorides + FreeSulfurDioxide + Density +
##     Alcohol, data = train_wine[, c(significant_vars, "TARGET")])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2722 -0.5164  0.1259  0.7474  3.2452
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         8.2588225  1.5218057   5.427 5.87e-08 ***
## STARS2              1.0017744  0.0289130  34.648  < 2e-16 ***
## STARS3              1.5177725  0.0336812  45.063  < 2e-16 ***
## STARS4              2.0591005  0.0534033  38.558  < 2e-16 ***
## LabelAppeal         0.6447709  0.0144200  44.714  < 2e-16 ***
## AcidIndex          -0.1292178  0.0149836  -8.624  < 2e-16 ***
## VolatileAcidity    -0.1300537  0.0328101  -3.964 7.43e-05 ***
## ResidualSugar       0.0037320  0.0014800   2.522 0.011698 *
## Chlorides          -0.4620442  0.1467712  -3.148 0.001649 **
## FreeSulfurDioxide   0.0015461  0.0004037   3.830 0.000129 ***
```

```
## Density           -4.8113006  1.5278781  -3.149 0.001643 **
## Alcohol            0.0234482  0.0045357   5.170 2.39e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 9424 degrees of freedom
##   (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4394, Adjusted R-squared:  0.4388
## F-statistic: 671.5 on 11 and 9424 DF,  p-value: < 2.2e-16
```

Finally, I applied the step() function to the lm_significant model, with the direction = "both" argument, which allows the function to both add and remove predictors from the model. The trace = 0 argument suppresses the step-by-step output of the algorithm.

The stepwise_model contains the final model obtained after the stepwise regression process. The summary of this model is displayed, which shows the following: The model includes the same predictors as the lm_significant model, except for FixedAcidity, which has been removed by the stepwise algorithm. The coefficients, standard errors, t-values, and p-values for the remaining predictors are provided. The residual standard error and R-squared values are similar to the lm_significant model.

The advantage of using stepwise regression was that it provides an automated method for selecting the most relevant predictors and removing redundant predictors from the model. Thus improved the model's interpretability, reducing overfitting, and enhancing its predictive performance.