

# 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-evaluation')
```

```
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 columns 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
```

Since the STAR column is an ordinal categorical variable we can transform it into 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)
```

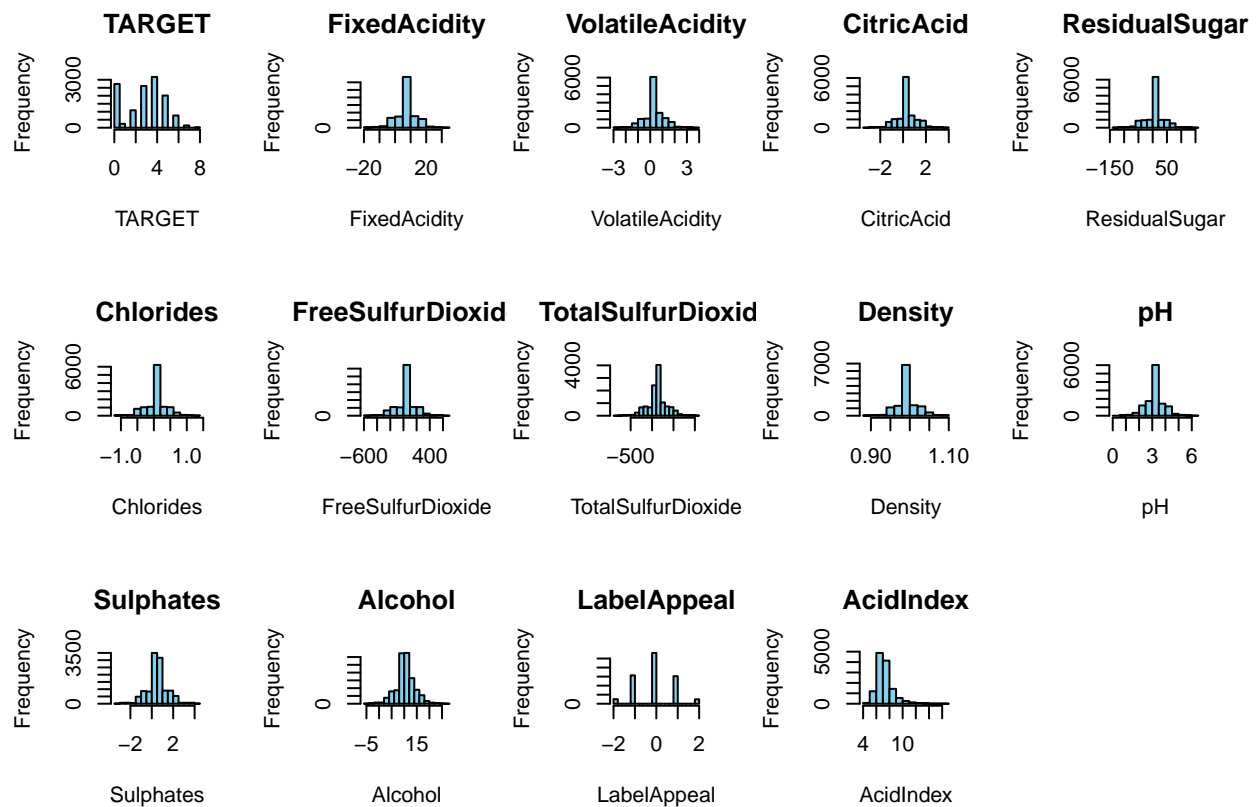
```
## 'data.frame':   12795 obs. of  15 variables:
## $ TARGET          : int  3 3 5 3 4 0 0 4 3 6 ...
## $ FixedAcidity     : num  3.2 4.5 7.1 5.7 8 11.3 7.7 6.5 14.8 5.5 ...
```

```
## $ 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

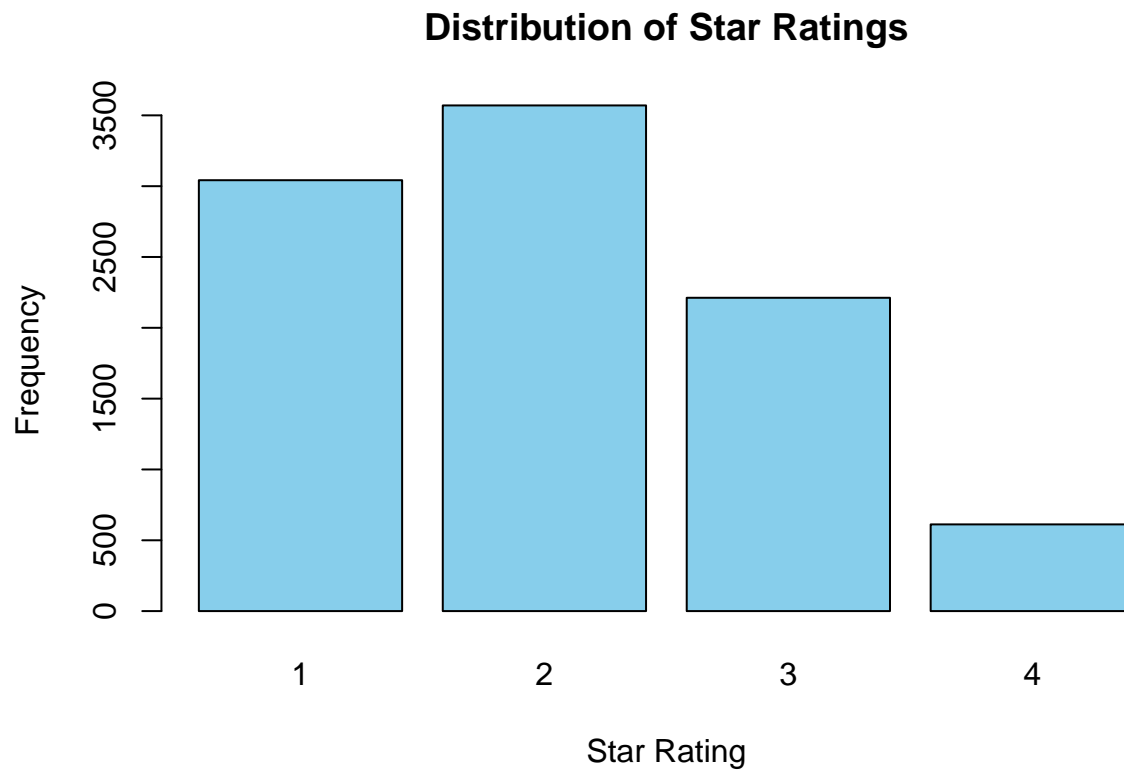
To get a better sense of the distribution of the columns, we can plot a histogram of them. The target variable “TARGET” (number of cases sold) has a right-skewed distribution, with most values concentrated towards lower counts. Some predictor variables like FixedAcidity, VolatileAcidity, and CitricAcid 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.

```
# 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).

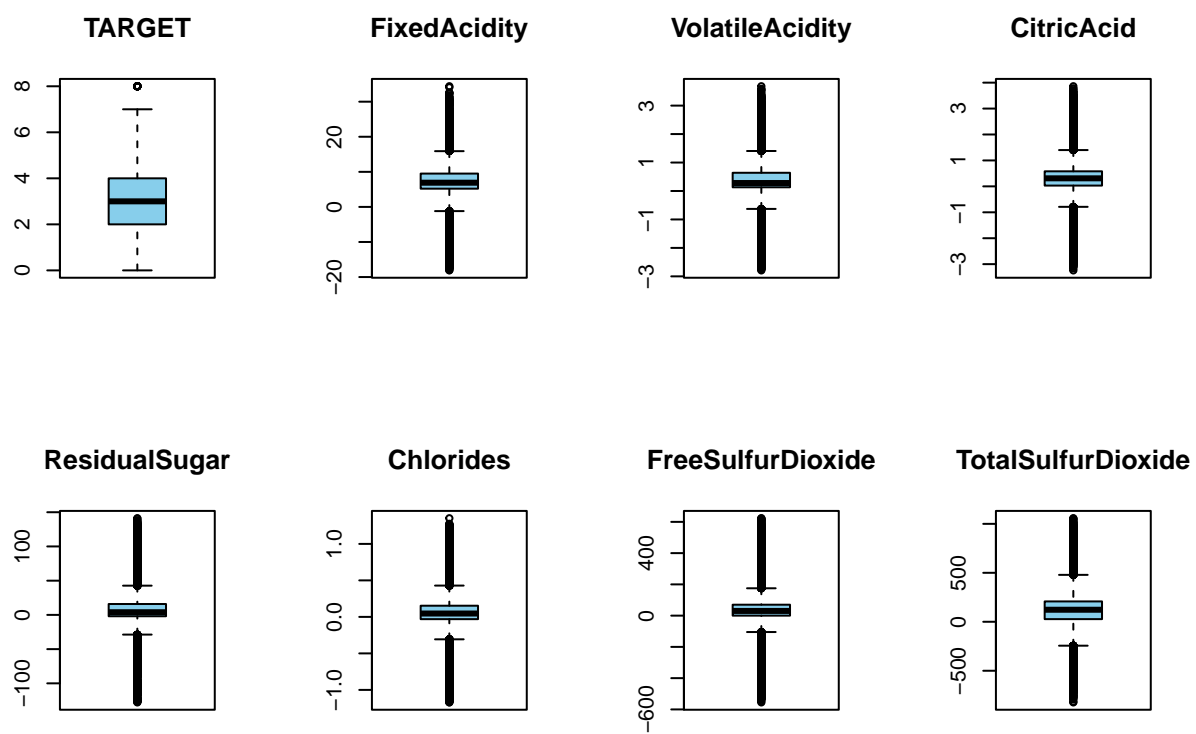
```
# Create bar plot for STARS variable
barplot(table(train_wine$STARS),
        main = "Distribution of Star Ratings",
        col = "skyblue",
        xlab = "Star Rating",
        ylab = "Frequency")
```

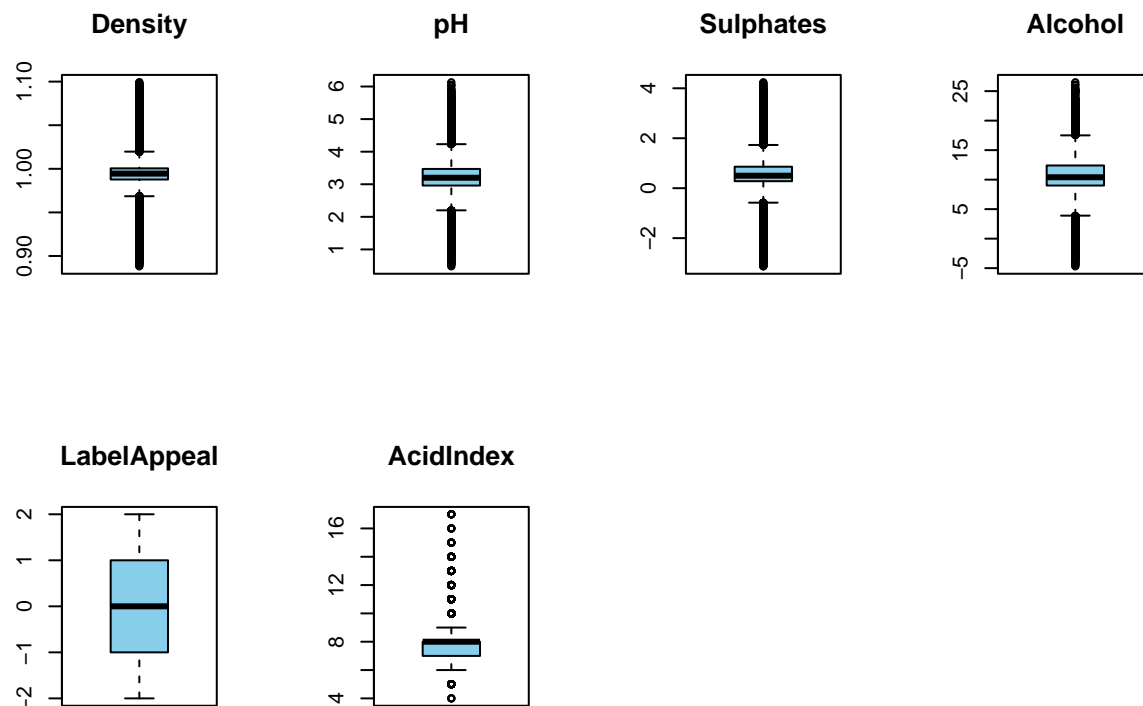


### Boxplot

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")
}
```





```
train_wine1 <- train_wine
```

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

## 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.

```
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)

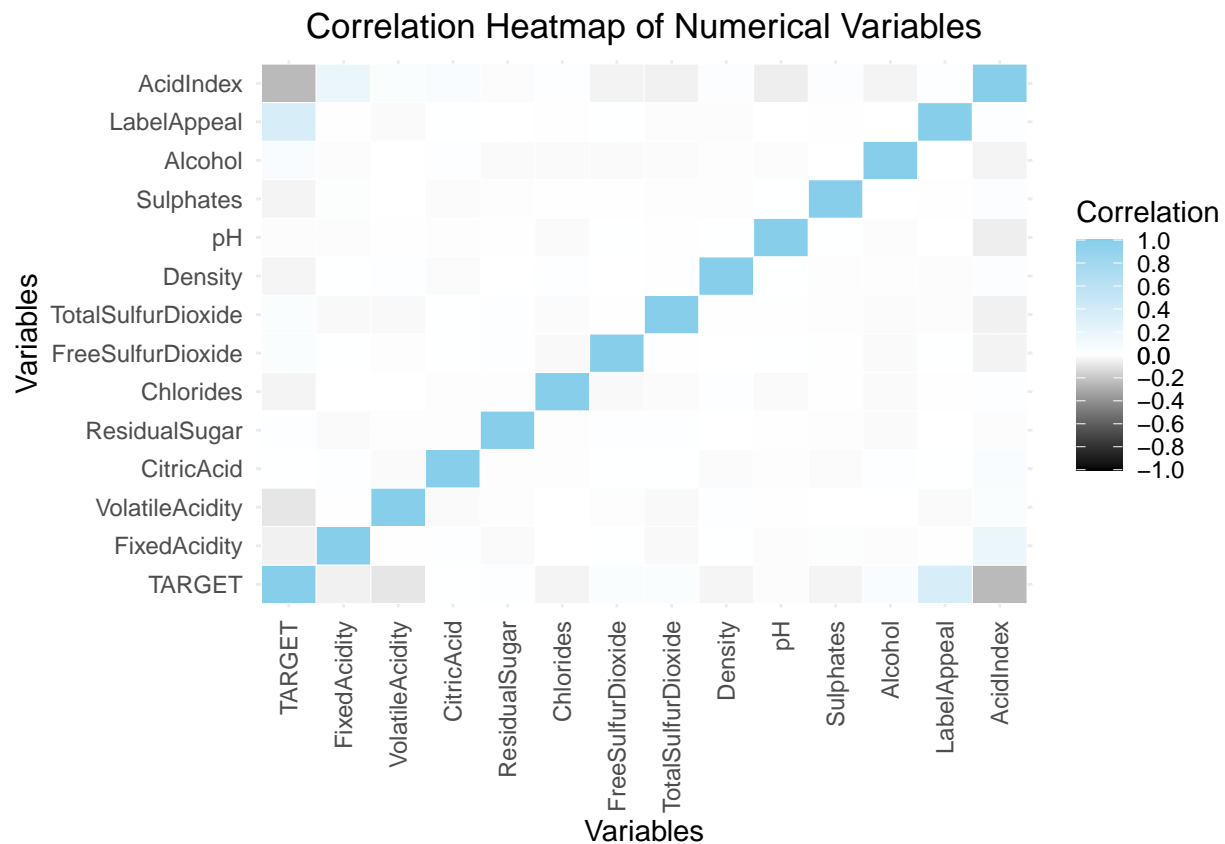
# Create heatmap using ggplot2
ggplot(correlation_df, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "black", mid = "white", high = "skyblue", midpoint = 0,
    breaks = c(seq(-1, 0, by = 0.2), seq(0, 1, by = 0.2)),
```



```

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

Let's prepare the data for modeling by identifying the columns that have missing values and negative values.

```

# Function to count NA values in each column
count_na <- function(df) {
  sapply(names(df), function(col_name) {
    result <- sum(is.na(df[[col_name]]))
  })
}

# Count NA values
print('Count of NAs:')

```

```
## [1] "Count of NAs:"
```

```
count_na(train_wine)
```

```
##          TARGET          FixedAcidity    VolatileAcidity          CitricAcid
##              0              0              0              0
##  ResidualSugar      Chlorides  FreeSulfurDioxide  TotalSulfurDioxide
##          616          638          647          682
##      Density          pH          Sulphates          Alcohol
##              0          395          1210          653
##  LabelAppeal      AcidIndex          STARS
##              0              0          3359
```

```
# Function to count negative values in each column
```

```
count_negative <- function(df) {
  sapply(names(df), function(col_name) {
    result <- sum(df[[col_name]] < 0)
  })
}
```

```
# Count negative values
```

```
print('Count of negative values:')
```

```
## [1] "Count of negative values:"
```

```
count_negative(train_wine)
```

```
## Warning in Ops.factor(df[[col_name]], 0): '<' not meaningful for factors
```

```
##          TARGET          FixedAcidity    VolatileAcidity          CitricAcid
##              0          1621          2827          2966
##  ResidualSugar      Chlorides  FreeSulfurDioxide  TotalSulfurDioxide
##              NA              NA              NA              NA
##      Density          pH          Sulphates          Alcohol
##              0              NA              NA              NA
##  LabelAppeal      AcidIndex          STARS
##          3640              0          NA
```

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 = TRUE)
non_negative_median4 <- median(train_wine$TotalSulfurDioxide[train_wine$TotalSulfurDioxide >= 0], na.rm = TRUE)
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)

## Everything 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
```

```
glimpse(train_wine)
```

```
## Rows: 12,795
## Columns: 21
## $ TARGET <int> 3, 3, 5, 3, 4, 0, 0, 4, 3, 6, 0, 4, 3, 7, 4~
## $ FixedAcidity <dbl> 3.2, 4.5, 7.1, 5.7, 8.0, 11.3, 7.7, 6.5, 14~
## $ VolatileAcidity <dbl> 1.160, 0.160, 2.640, 0.385, 0.330, 0.320, 0~
## $ CitricAcid <dbl> -0.98, -0.81, -0.88, 0.04, -1.26, 0.59, -0.~
## $ ResidualSugar <dbl> 54.20, 26.10, 14.80, 18.80, 9.40, 2.20, 21.~
## $ Chlorides <dbl> 0.063000, 0.063000, 0.037000, 0.063000, 0.1~
## $ FreeSulfurDioxide <dbl> 79.50877, 15.00000, 214.00000, 22.00000, 43~
## $ TotalSulfurDioxide <dbl> 268.0000, 150.0000, 142.0000, 115.0000, 108~
## $ Density <dbl> 0.99280, 1.02792, 0.99518, 0.99640, 0.99457~
## $ pH <dbl> 3.33, 3.38, 3.12, 2.24, 3.12, 3.20, 3.49, 3~
## $ Sulphates <dbl> 0.5750000, 0.7000000, 0.4800000, 1.8300000, ~
## $ Alcohol <dbl> 9.90000, 10.60758, 22.00000, 6.20000, 13.70~
## $ LabelAppeal <int> 0, -1, -1, -1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 2~
## $ AcidIndex <int> 8, 7, 8, 6, 9, 11, 8, 7, 6, 8, 5, 10, 7, 8,~
## $ STARS <fct> 2, 3, 3, 1, 2, NA, NA, 3, NA, 4, 1, 2, 2, 3~
## $ ResidualSugar_missing <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ TotalSulfurDioxide_missing <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0~
## $ Chlorides_missing <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ FreeSulfurDioxide_missing <dbl> 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0~
## $ Sulphates_missing <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0~
## $ Alcohol_missing <dbl> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

```
# Count NA values
print('Count of NAs:')
```

```
## [1] "Count of NAs:"
```

```
count_na(train_wine)
```

```
##          TARGET          FixedAcidity
##          0          0
##    VolatileAcidity          CitricAcid
##          0          0
##    ResidualSugar          Chlorides
##          0          0
##    FreeSulfurDioxide    TotalSulfurDioxide
##          0          0
##          Density          pH
##          0          0
##    Sulphates          Alcohol
##          0          0
##    LabelAppeal          AcidIndex
##          0          0
##          STARS    ResidualSugar_missing
##    3359          0
## TotalSulfurDioxide_missing    Chlorides_missing
##          0          0
## FreeSulfurDioxide_missing    Sulphates_missing
##          0          0
##    Alcohol_missing
##          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 -0.01509258 -0.613475120    1.066174586 -1.4945399    1.82861722 -0.4655856
## 2 -0.01509258 -0.407702191    -0.209312458 -1.2973424    0.46080061 -0.4655856
## 3  1.02313053  0.003843668    2.953895411 -1.3785414   -0.08924663 -0.5967961
## 4 -0.01509258 -0.217757948    0.077672127 -0.3113548    0.10546036 -0.4655856
## 5  0.50401898  0.146301850    0.007520339 -1.8193358   -0.35210107  0.0000000
## 6 -1.57242724  0.668648516   -0.005234531  0.3266372   -0.70257365  2.0223676
## FreeSulfurDioxide TotalSulfurDioxide    Density          pH  Sulphates
## 1      0.0000000      0.5091881 -0.05285768  0.25470317 -0.3947285
## 2     -0.7076548     -0.2975785  1.27054534  0.31222528 -0.1793348
## 3      1.4753554     -0.3522745  0.03682624  0.01311028 -0.5584278
## 4     -0.6308654     -0.5368736  0.08279867 -0.99927897  1.7678247
## 5     -0.4004975     -0.5847326  0.01384003  0.01311028  1.6644357
## 6     -0.4004975     -1.2205740  0.19584562  0.10514566  0.8373237
##    Alcohol LabelAppeal  AcidIndex
## 1 -0.2062209  0.01017411  0.1716684
## 2  0.0000000 -1.11204794 -0.5836606
## 3  3.3202797 -1.11204794  0.1716684
## 4 -1.2845723 -1.11204794 -1.3389897
## 5  0.9012752  0.01017411  0.9269974
## 6  1.3967339  0.01017411  2.4376554
```

### 3. 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
## -5.3268 -0.5073  0.1528  0.7234  3.2194
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.507936   0.042231  59.387 < 2e-16 ***
## LabelAppeal  0.655637   0.014443  45.396 < 2e-16 ***
## STARS2       1.010138   0.028969  34.869 < 2e-16 ***
## STARS3       1.539294   0.033703  45.672 < 2e-16 ***
## STARS4       2.160403   0.053479  40.397 < 2e-16 ***
## Alcohol      0.024749   0.003477   7.118 1.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.16 on 9430 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4424, Adjusted R-squared:  0.4421
## F-statistic: 1496 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 43.09% 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
## -5.2116 -0.5320  0.1073  0.7304  3.2441
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.7057206   0.4494948   10.469  < 2e-16 ***
## FixedAcidity     0.0011607   0.0018901    0.614  0.5392
## VolatileAcidity  -0.0846857   0.0150812   -5.615 2.02e-08 ***
## CitricAcid       0.0089719   0.0136755    0.656  0.5118
## ResidualSugar    0.0003033   0.0005727    0.530  0.5964
## Chlorides       -0.1851651   0.0604589   -3.063  0.0022 **
## FreeSulfurDioxide 0.0002116   0.0001280    1.653  0.0983 .
## TotalSulfurDioxide 0.0001317   0.0000816    1.614  0.1066
## Density        -0.9058991   0.4430536   -2.045  0.0409 *
## pH              0.0016649   0.0136459    0.122  0.9029
## Sulphates      -0.0289965   0.0204718   -1.416  0.1567
## Alcohol         0.0222956   0.0034271    6.506 8.13e-11 ***
## LabelAppeal     0.6656774   0.0142261   46.793  < 2e-16 ***
## AcidIndex      -0.1628359   0.0101775  -16.000  < 2e-16 ***
## STARS2          0.9773303   0.0285896   34.185  < 2e-16 ***
## STARS3          1.4852584   0.0332973   44.606  < 2e-16 ***
## STARS4          2.0918045   0.0527567   39.650  < 2e-16 ***
## ResidualSugar_missing 0.0762077   0.0550136    1.385  0.1660
## TotalSulfurDioxide_missing 0.0099005   0.0527390    0.188  0.8511
## Chlorides_missing -0.0035264   0.0541882   -0.065  0.9481
## FreeSulfurDioxide_missing 0.1110187   0.0547214    2.029  0.0425 *
## Sulphates_missing -0.0011549   0.0405624   -0.028  0.9773
## Alcohol_missing   0.0922902   0.0538365    1.714  0.0865 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.14 on 9413 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4618, Adjusted R-squared:  0.4605
## F-statistic: 367.1 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 approximately a 0.6448 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 approximately 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 approximately a 0.0236 increase in expected cases ordered.

Using Stepwise regression

```

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

summary(stepwise_model)

##
## Call:
## lm(formula = TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
##     TotalSulfurDioxide + Density + Alcohol + LabelAppeal + AcidIndex +
##     STARS + FreeSulfurDioxide_missing + Alcohol_missing, data = train_wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.2395 -0.5322  0.1092  0.7324  3.2417
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.704e+00  4.465e-01  10.533 < 2e-16 ***
## VolatileAcidity   -8.447e-02  1.507e-02  -5.606 2.13e-08 ***
## Chlorides        -1.862e-01  6.041e-02  -3.083 0.00206 **
## FreeSulfurDioxide  2.137e-04  1.279e-04   1.671 0.09472 .
## TotalSulfurDioxide 1.326e-04  8.155e-05   1.626 0.10399
## Density          -9.084e-01  4.428e-01  -2.052 0.04024 *
## Alcohol           2.225e-02  3.424e-03   6.499 8.52e-11 ***
## LabelAppeal       6.654e-01  1.422e-02  46.794 < 2e-16 ***
## AcidIndex        -1.620e-01  1.004e-02 -16.139 < 2e-16 ***
## STARS2            9.771e-01  2.856e-02  34.208 < 2e-16 ***
## STARS3            1.486e+00  3.328e-02  44.652 < 2e-16 ***
## STARS4            2.095e+00  5.272e-02  39.736 < 2e-16 ***
## FreeSulfurDioxide_missing 1.130e-01  5.469e-02   2.066 0.03888 *
## Alcohol_missing   9.145e-02  5.381e-02   1.700 0.08926 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.14 on 9422 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4615, Adjusted R-squared:  0.4607
## F-statistic: 621.1 on 13 and 9422 DF,  p-value: < 2.2e-16

```

Based on the analysis, Model 2 appears to be preferable. It retains significant predictors such as VolatileAcidity, ResidualSugar, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, Density, Alcohol, LabelAppeal, AcidIndex, and STARS, while simplifying the model. Additionally, the adjusted R-squared value of Model 2 (0.4389) is only slightly lower than that of Model 1 (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.

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

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

## 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

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

# Model Selection
cat("\n***** Poisson Model *****\n")

##
## ***** Poisson Model *****
summary(poisson_model)

##
## Call:
## glm(formula = TARGET ~ ., family = "poisson", data = train_wine)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.560e+00  2.055e-01   7.588 3.26e-14 ***
## FixedAcidity    3.092e-04  8.655e-04   0.357 0.720894
## VolatileAcidity -2.259e-02  6.910e-03  -3.269 0.001079 **
## CitricAcid      2.455e-03  6.227e-03   0.394 0.693340
## ResidualSugar   8.951e-05  2.608e-04   0.343 0.731410
## Chlorides      -4.953e-02  2.787e-02  -1.777 0.075549 .
## FreeSulfurDioxide 5.635e-05  5.829e-05   0.967 0.333725
## TotalSulfurDioxide 3.172e-05  3.721e-05   0.852 0.394042
## Density        -2.678e-01  2.024e-01  -1.323 0.185715
## pH             3.015e-05  6.214e-03   0.005 0.996129
## Sulphates      -9.054e-03  9.399e-03  -0.963 0.335403
## Alcohol        5.801e-03  1.572e-03   3.690 0.000224 ***
## LabelAppeal    1.826e-01  6.543e-03  27.910 < 2e-16 ***
## AcidIndex      -4.679e-02  4.902e-03  -9.545 < 2e-16 ***
## STARS2         3.208e-01  1.437e-02  22.323 < 2e-16 ***
## STARS3         4.351e-01  1.569e-02  27.731 < 2e-16 ***
## STARS4         5.395e-01  2.179e-02  24.754 < 2e-16 ***
## ResidualSugar_missing 1.970e-02  2.481e-02   0.794 0.427128
## TotalSulfurDioxide_missing 4.538e-03  2.388e-02   0.190 0.849283
## Chlorides_missing -4.261e-03  2.463e-02  -0.173 0.862687
## FreeSulfurDioxide_missing 2.863e-02  2.458e-02   1.165 0.244202
## Sulphates_missing -1.822e-03  1.856e-02  -0.098 0.921791
## Alcohol_missing  2.449e-02  2.434e-02   1.006 0.314422
## ---
```



```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
## Null deviance: 8597.2  on 9435  degrees of freedom
## Residual deviance: 5713.0  on 9413  degrees of freedom
## (3359 observations deleted due to missingness)
## AIC: 33850
##
## Number of Fisher Scoring iterations: 5

cat("\n***** Negative Binomial Regression *****\n")

##
## ***** Negative Binomial Regression *****

summary(nb_model)

##
## Call:
## glm.nb(formula = TARGET ~ ., data = train_wine, init.theta = 136927.4609,
## link = log)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.560e+00  2.055e-01   7.588 3.26e-14 ***
## FixedAcidity     3.092e-04  8.656e-04   0.357 0.720896
## VolatileAcidity  -2.259e-02  6.910e-03  -3.269 0.001079 **
## CitricAcid       2.455e-03  6.227e-03   0.394 0.693345
## ResidualSugar    8.951e-05  2.608e-04   0.343 0.731410
## Chlorides       -4.953e-02  2.787e-02  -1.777 0.075552 .
## FreeSulfurDioxide 5.635e-05  5.830e-05   0.967 0.333729
## TotalSulfurDioxide 3.172e-05  3.721e-05   0.852 0.394043
## Density        -2.678e-01  2.024e-01  -1.323 0.185719
## pH              3.010e-05  6.214e-03   0.005 0.996135
## Sulphates      -9.054e-03  9.399e-03  -0.963 0.335406
## Alcohol         5.801e-03  1.572e-03   3.690 0.000224 ***
## LabelAppeal     1.826e-01  6.543e-03  27.909 < 2e-16 ***
## AcidIndex      -4.679e-02  4.902e-03  -9.545 < 2e-16 ***
## STARS2          3.208e-01  1.437e-02  22.322 < 2e-16 ***
## STARS3          4.351e-01  1.569e-02  27.731 < 2e-16 ***
## STARS4          5.395e-01  2.179e-02  24.754 < 2e-16 ***
## ResidualSugar_missing 1.970e-02  2.481e-02   0.794 0.427136
## TotalSulfurDioxide_missing 4.538e-03  2.388e-02   0.190 0.849283
## Chlorides_missing -4.261e-03  2.463e-02  -0.173 0.862685
## FreeSulfurDioxide_missing 2.863e-02  2.458e-02   1.165 0.244206
## Sulphates_missing -1.822e-03  1.856e-02  -0.098 0.921785
## Alcohol_missing  2.449e-02  2.434e-02   1.006 0.314427
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(136927.5) family taken to be 1)
##
## Null deviance: 8597.0  on 9435  degrees of freedom
## Residual deviance: 5712.9  on 9413  degrees of freedom
```

```
## (3359 observations deleted due to missingness)
## AIC: 33852
##
## Number of Fisher Scoring iterations: 1
##
##
##          Theta: 136927
##        Std. Err.: 183967
## Warning while fitting theta: iteration limit reached
##
## 2 x log-likelihood: -33804.27
```

```
cat("\n***** Multiple Linear Regression *****\n")
```

```
##
## ***** Multiple Linear Regression *****
```

```
summary(linear_model)
```

```
##
## Call:
## lm(formula = TARGET ~ ., data = train_wine)
##
## Residuals:
```

	Min	1Q	Median	3Q	Max
	-5.2116	-0.5320	0.1073	0.7304	3.2441

```
##
## Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	4.7057206	0.4494948	10.469	< 2e-16 ***
FixedAcidity	0.0011607	0.0018901	0.614	0.5392
VolatileAcidity	-0.0846857	0.0150812	-5.615	2.02e-08 ***
CitricAcid	0.0089719	0.0136755	0.656	0.5118
ResidualSugar	0.0003033	0.0005727	0.530	0.5964
Chlorides	-0.1851651	0.0604589	-3.063	0.0022 **
FreeSulfurDioxide	0.0002116	0.0001280	1.653	0.0983 .
TotalSulfurDioxide	0.0001317	0.0000816	1.614	0.1066
Density	-0.9058991	0.4430536	-2.045	0.0409 *
pH	0.0016649	0.0136459	0.122	0.9029
Sulphates	-0.0289965	0.0204718	-1.416	0.1567
Alcohol	0.0222956	0.0034271	6.506	8.13e-11 ***
LabelAppeal	0.6656774	0.0142261	46.793	< 2e-16 ***
AcidIndex	-0.1628359	0.0101775	-16.000	< 2e-16 ***
STARS2	0.9773303	0.0285896	34.185	< 2e-16 ***
STARS3	1.4852584	0.0332973	44.606	< 2e-16 ***
STARS4	2.0918045	0.0527567	39.650	< 2e-16 ***
ResidualSugar_missing	0.0762077	0.0550136	1.385	0.1660
TotalSulfurDioxide_missing	0.0099005	0.0527390	0.188	0.8511
Chlorides_missing	-0.0035264	0.0541882	-0.065	0.9481
FreeSulfurDioxide_missing	0.1110187	0.0547214	2.029	0.0425 *
Sulphates_missing	-0.0011549	0.0405624	-0.028	0.9773
Alcohol_missing	0.0922902	0.0538365	1.714	0.0865 .

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.14 on 9413 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared: 0.4618, Adjusted R-squared: 0.4605
## F-statistic: 367.1 on 22 and 9413 DF, p-value: < 2.2e-16
```

#### AIC Values

```
# 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: 33850.11
```

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

```
## Negative Binomial Regression AIC: 33852.27
```

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

```
## Multiple Linear Regression AIC: 29279.14
```

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.

```
# 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
```

```
summary(linear_model)
```

```
##
## Call:
## lm(formula = TARGET ~ ., data = train_wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.2116 -0.5320  0.1073  0.7304  3.2441
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.7057206  0.4494948  10.469 < 2e-16 ***
## FixedAcidity      0.0011607  0.0018901   0.614  0.5392
## VolatileAcidity   -0.0846857  0.0150812  -5.615 2.02e-08 ***
## CitricAcid        0.0089719  0.0136755   0.656  0.5118
## ResidualSugar     0.0003033  0.0005727   0.530  0.5964
## Chlorides         -0.1851651  0.0604589  -3.063  0.0022 **
## FreeSulfurDioxide  0.0002116  0.0001280   1.653  0.0983 .
## TotalSulfurDioxide 0.0001317  0.0000816   1.614  0.1066
## Density           -0.9058991  0.4430536  -2.045  0.0409 *
## pH                0.0016649  0.0136459   0.122  0.9029
## Sulphates         -0.0289965  0.0204718  -1.416  0.1567
## Alcohol           0.0222956  0.0034271   6.506 8.13e-11 ***
## LabelAppeal       0.6656774  0.0142261  46.793 < 2e-16 ***
## AcidIndex         -0.1628359  0.0101775 -16.000 < 2e-16 ***
## STARS2            0.9773303  0.0285896  34.185 < 2e-16 ***
## STARS3            1.4852584  0.0332973  44.606 < 2e-16 ***
## STARS4            2.0918045  0.0527567  39.650 < 2e-16 ***
## ResidualSugar_missing 0.0762077  0.0550136   1.385  0.1660
## TotalSulfurDioxide_missing 0.0099005  0.0527390   0.188  0.8511
## Chlorides_missing -0.0035264  0.0541882  -0.065  0.9481
## FreeSulfurDioxide_missing 0.1110187  0.0547214   2.029  0.0425 *
## Sulphates_missing -0.0011549  0.0405624  -0.028  0.9773
## Alcohol_missing   0.0922902  0.0538365   1.714  0.0865 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.14 on 9413 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4618, Adjusted R-squared:  0.4605
## F-statistic: 367.1 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
## -5.2606 -0.5335  0.1049  0.7354  3.2056
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.7131653   0.4468440   10.548 < 2e-16 ***
## STARS2          0.9769849   0.0285714   34.194 < 2e-16 ***
## STARS3          1.4866743   0.0332917   44.656 < 2e-16 ***
## STARS4          2.0961419   0.0527401   39.745 < 2e-16 ***
## LabelAppeal     0.6649322   0.0142241   46.747 < 2e-16 ***
## AcidIndex      -0.1634126   0.0101463  -16.106 < 2e-16 ***
## FixedAcidity     0.0011812   0.0018899    0.625  0.53198
## VolatileAcidity -0.0850958   0.0150718  -5.646 1.69e-08 ***
## ResidualSugar    0.0003340   0.0005723    0.584  0.55951
## Chlorides       -0.1852183   0.0604318   -3.065  0.00218 **
## FreeSulfurDioxide 0.0002131   0.0001279    1.666  0.09578 .
## Density         -0.8838656   0.4428344   -1.996  0.04597 *
## Alcohol          0.0221731   0.0034248    6.474 1.00e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.14 on 9423 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.461, Adjusted R-squared:  0.4603
## F-statistic: 671.6 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 +
##       Chlorides + FreeSulfurDioxide + Density + Alcohol, data = train_wine[,
##       c(significant_vars, "TARGET")])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.2530 -0.5319  0.1054  0.7357  3.2207
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.7191608   0.4466384   10.566 < 2e-16 ***
## STARS2          0.9771930   0.0285641   34.211 < 2e-16 ***
## STARS3          1.4872472   0.0332820   44.686 < 2e-16 ***
## STARS4          2.0966189   0.0527317   39.760 < 2e-16 ***
## LabelAppeal     0.6647792   0.0142219   46.743 < 2e-16 ***
```

```
## AcidIndex          -0.1625572  0.0100370 -16.196 < 2e-16 ***
## VolatileAcidity    -0.0850665  0.0150705  -5.645 1.70e-08 ***
## Chlorides          -0.1861291  0.0604178  -3.081 0.00207 **
## FreeSulfurDioxide  0.0002143  0.0001279   1.676 0.09384 .
## Density            -0.8823296  0.4427863  -1.993 0.04633 *
## Alcohol            0.0221256  0.0034238   6.462 1.08e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.14 on 9425 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4609, Adjusted R-squared:  0.4604
## F-statistic: 805.9 on 10 and 9425 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.

### Model 3 (Poisson)

For this poisson model, instead of using variables that were statistically significant to the response variable, I am going to use the variables that had the highest coefficient from the previous poisson model. These variables were `STARS`, `LabelAppeal`, and `Density`. These variables had the largest effect on the response variable.

```
# Poisson Regression with selected variables
poisson_model_selected <- glm(TARGET ~ STARS + LabelAppeal + Density, data = train_wine, family = "poisson")

summary(poisson_model_selected)

##
## Call:
## glm(formula = TARGET ~ STARS + LabelAppeal + Density, family = "poisson",
##      data = train_wine)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.343662   0.200910   6.688 2.26e-11 ***
## STARS2       0.330709   0.014322  23.091 < 2e-16 ***
## STARS3       0.453370   0.015593  29.075 < 2e-16 ***
## STARS4       0.563550   0.021667  26.010 < 2e-16 ***
## LabelAppeal  0.179632   0.006528  27.517 < 2e-16 ***
## Density     -0.358261   0.201765  -1.776  0.0758 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
```

```
##
## Null deviance: 8597.2 on 9435 degrees of freedom
## Residual deviance: 5849.3 on 9430 degrees of freedom
## (3359 observations deleted due to missingness)
## AIC: 33952
##
## Number of Fisher Scoring iterations: 5
```

In this poisson model, the intercept when all the predictors are zero is 1.34. For the coefficients, the coefficient for STARS2 is 0.330709, which means that the log count of the TARGET is expected to increase by 0.330709 when the wine has 2 stars. STARS3 and STARS4 have a coefficient estimate of 0.45 and 0.56, respectively. This suggests that as the high STAR rated wines cause a higher increase in the cases of wine sold. The LabelAppeal coefficient is 0.18 so a one unit increase in LabelAppeal will cause a log change of 0.18 in the TARGET. In comparison, a one unit increase in Density, causes a log decrease of 0.36 in the TARGET. Like before, the STARS and LabelAppeal variable are statistically significant while the density variable is not with a p-value of 0.076.

Theoretically speaking, the STARS and LabelAppeal coefficients make sense since the rating and look of the wine should increase the amount of cases that are purchased. However, it is hard to gauge how density impacts the response variable. It seems that in all of the models we've done so far, an increase in density seems to cause a decrease in the units purchased.

The model's AIC is 33952, which is higher than the AIC of the previous model (33850.11) when we used all of the response variables. The residual deviance is quite large at 5849.3, suggesting that there may be room for improvement in the model. We can also check if the poisson model is a good fit for the data by looking at the mean and variance to get check for dispersion.

```
# Calculate the mean of the TARGET variable
mean_target <- mean(train_wine$TARGET, na.rm = TRUE)

# Calculate the variance of the TARGET variable
var_target <- var(train_wine$TARGET, na.rm = TRUE)

# Print the mean and variance
print(paste("Mean of TARGET: ", mean_target))

## [1] "Mean of TARGET: 3.02907385697538"

print(paste("Variance of TARGET: ", var_target))

## [1] "Variance of TARGET: 3.71089452283923"
```

In a poisson distribution, one of the assumptions is that the mean and variance are exactly equal. However, the mean and variance of the TARGET variable is relatively close but not equal. Since the variance is slightly greater than the mean, this could indicate overdispersion, which can suggest that a negative binomial regression model might be a better fit for this data.

### Model 3 (Negative Binomial Regression)

For this negative binomial regression model, we are going to use the same predictors as the previous poisson model. This is because the response variable had a variance greater than the mean suggesting overdispersion.

```
nb_model3 <- glm.nb(TARGET ~ STARS + LabelAppeal + Density, data = train_wine)

## 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(nb_model3)

##
## Call:
## glm.nb(formula = TARGET ~ STARS + LabelAppeal + Density, data = train_wine,
##       init.theta = 136299.622, link = log)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.343664   0.200913   6.688 2.27e-11 ***
## STARS2       0.330709   0.014322  23.091 < 2e-16 ***
## STARS3       0.453370   0.015593  29.075 < 2e-16 ***
## STARS4       0.563550   0.021667  26.009 < 2e-16 ***
## LabelAppeal  0.179632   0.006528  27.516 < 2e-16 ***
## Density     -0.358263   0.201768  -1.776  0.0758 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(136299.6) family taken to be 1)
##
## Null deviance: 8597.0 on 9435 degrees of freedom
## Residual deviance: 5849.3 on 9430 degrees of freedom
## (3359 observations deleted due to missingness)
## AIC: 33955
##
## Number of Fisher Scoring iterations: 1
##
##
##              Theta: 136300
##             Std. Err.: 184038
## Warning while fitting theta: iteration limit reached
##
## 2 x log-likelihood: -33940.66
```

The results of Negative Binomial regression model are similar to those of the poisson model. As before, the coefficients STARS and LabelAppeal are all statistically significant. The dispersion parameter for the Negative Binomial model is 136299.6, which is significantly larger than 1, indicating overdispersion in the data. However, since the difference in mean and variance is roughly equal to each other, this might be the reason why the results are very similar in the poisson and negative binomial models.

The AIC is 33955, which is a bit higher than that of the poisson model (33952). Since these results are almost the same, let's add another input into the model. I chose the variable AcidIndex since it showed statistical significance in the previous negative binomial regression model with all the predictor variables and removed the density variable.

```
nb_model3a <- glm.nb(TARGET ~ STARS + LabelAppeal + AcidIndex, data = train_wine)

## 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(nb_model3a)
```



```
##
## Call:
## glm.nb(formula = TARGET ~ STARS + LabelAppeal + AcidIndex, data = train_wine,
##       init.theta = 136188.1391, link = log)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.364532   0.038971  35.01  <2e-16 ***
## STARS2       0.322236   0.014349  22.46  <2e-16 ***
## STARS3       0.439408   0.015658  28.06  <2e-16 ***
## STARS4       0.547683   0.021726  25.21  <2e-16 ***
## LabelAppeal  0.182412   0.006537  27.91  <2e-16 ***
## AcidIndex   -0.048570   0.004827 -10.06  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(136188.1) family taken to be 1)
##
## Null deviance: 8597.0 on 9435 degrees of freedom
## Residual deviance: 5748.4 on 9430 degrees of freedom
## (3359 observations deleted due to missingness)
## AIC: 33854
##
## Number of Fisher Scoring iterations: 1
##
##
##             Theta: 136188
##           Std. Err.: 182918
## Warning while fitting theta: iteration limit reached
##
## 2 x log-likelihood: -33839.84
```

Looking at the coefficients, like in other models, a one unit increase in the STARS rating causes a log increase in the TARGET variable. Again, the coefficients STARS and LabelAppeal are statistically significant. The new predictor variable added, AcidIndex, has a coefficient of -0.048 which means a one unit increase in AcidIndex will cause a log decrease of 0.048 in the TARGET variable. This seems to correspond to the results of the other model since the coefficient of AcidIndex has been negative in all the models we've done so far. The p-value of AcidIndex is also less than 0.05 indicating it is statistically significant.

The AIC here is 33854, slightly lower than the previous binomial regression model (33955). The addition of AcidIndex as a predictor has improved the model fit, as indicated by the decrease in AIC.

### Model 3 (Multiple Linear Regression)

For this multiple linear regression model, we are going to use the variables that statistically significant after using the step() function. These variables are STARS, LabelAppeal, AcidIndex, VolatileAcidity, Chlorides, and Alcohol.

```
model3_linear <- lm(formula = TARGET ~ STARS + LabelAppeal + AcidIndex + VolatileAcidity +
  Chlorides + Alcohol, data = train_wine)

summary(model3_linear)
```

```
##
## Call:
## lm(formula = TARGET ~ STARS + LabelAppeal + AcidIndex + VolatileAcidity +
```

```
## Chlorides + Alcohol, data = train_wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.2472 -0.5330  0.1032  0.7381  3.2134
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.868913   0.090317  42.837 < 2e-16 ***
## STARS2        0.978261   0.028568  34.243 < 2e-16 ***
## STARS3        1.487803   0.033288  44.695 < 2e-16 ***
## STARS4        2.096113   0.052738  39.746 < 2e-16 ***
## LabelAppeal   0.665390   0.014222  46.785 < 2e-16 ***
## AcidIndex    -0.163705   0.010029 -16.322 < 2e-16 ***
## VolatileAcidity -0.085624  0.015073  -5.681 1.38e-08 ***
## Chlorides    -0.189035   0.060422  -3.129 0.00176 **
## Alcohol       0.022083   0.003424   6.449 1.18e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.141 on 9427 degrees of freedom
## (3359 observations deleted due to missingness)
## Multiple R-squared:  0.4606, Adjusted R-squared:  0.4601
## F-statistic: 1006 on 8 and 9427 DF, p-value: < 2.2e-16
# Assume 'model' is your fitted linear regression model
aic_value <- AIC(model3_linear)
print('AIC: ')

## [1] "AIC: "

print(aic_value)

## [1] 29272.63
```

For the coefficients, a unit increase in STARS causes an increase in the TARGET variable. A STAR rating of 4 has the largest impact on the response variable as a STAR rating of 4 causes an increase of 2.09 to the amount of cases of wine sold. LabelAppeal has a coefficient of 0.66, indicating an increase of 0.66 in the response variable for every single unit increase at LabelAppeal. These coefficients make sense as one would expect the cases of wine sold to increase as the wine has a better rating and the appeal of the wine is higher. For AcidIndex, VolatileAcidity, and Chlorides, increases in these variables has a negative impact on the TARGET variable. This seems to coincide with the other models where an increase in these variables causes a decrease in the response variable. All the coefficients are statistically significant in predicting the TARGET variable.

The model's R-squared value is 0.4601, indicating that 46% of the variability in the TARGET variable can be explained by these predictors. The overall model has a p-value less than 0.05, indicating that it is statistically significant in predicting the amount of cases purchased. The R-squared is similar to the previous linear models so we can keep the model for now. Also, the AIC seems to be smaller than the AIC of the count regression models suggesting this multiple linear regression model may be a better fit for the data.

## 4. Select Models

To select the best count regression model, we will first use the AIC to measure how well the model fits the data. Unlike metrics such as average squared error, AIC takes into account both the goodness of fit and the complexity of the model, helping to avoid overfitting. Model 2's poisson and negative binomial regression

model had a AIC of 33850 and 33852, respectively. Model 3's poisson and negative binomial regression model had a AIC of 33952 and 33854, respectively. The AIC of all of these count regression models are pretty similar as they are within 1% of each other. Since this is the case, we can pick the model that is more parsimonious, which is model 3's negative binomial regression model. While the multiple linear regression models had a lower AIC than the count regression models, it may not be the best choice with count data like the TARGET variable. Also, the distribution of the TARGET variable was not normally distributed which violates the assumption of linear regression.

```
nb_model3a <- glm.nb(TARGET ~ STARS + LabelAppeal + AcidIndex, data = train_wine)
```

```
## 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(nb_model3a)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ STARS + LabelAppeal + AcidIndex, data = train_wine,
##       init.theta = 136188.1391, link = log)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.364532   0.038971  35.01  <2e-16 ***
## STARS2       0.322236   0.014349  22.46  <2e-16 ***
## STARS3       0.439408   0.015658  28.06  <2e-16 ***
## STARS4       0.547683   0.021726  25.21  <2e-16 ***
## LabelAppeal  0.182412   0.006537  27.91  <2e-16 ***
## AcidIndex   -0.048570   0.004827 -10.06  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(136188.1) family taken to be 1)
##
##      Null deviance: 8597.0  on 9435  degrees of freedom
## Residual deviance: 5748.4  on 9430  degrees of freedom
## (3359 observations deleted due to missingness)
## AIC: 33854
##
## Number of Fisher Scoring iterations: 1
##
##              Theta: 136188
##             Std. Err.: 182918
## Warning while fitting theta: iteration limit reached
##
## 2 x log-likelihood: -33839.84
```

This Negative Binomial regression model looked at the three variables STARS, LabelAppeal, AcidIndex in relation to the response variable, TARGET. Since TARGET had a variance(3.7) that was greater than the mean(3.0), indicating overdispersion, it made sense that a negative binomial regression model would be more appropriate than using a poisson regression model. In addition, the observations are independent of each other so it meets the assumption of independence.

From the model, we can infer that the higher the STARS and LabelAppeal value, the higher the number of cases of wine purchased will be. On the other hand, an increase in AcidIndex seems to have a negative effect on the TARGET. This seems to correspond with the other models as well where an increase in the acidity of the wine lowers the number of cases of wine purchased. The model's coefficients were statistically significant and aligned with our understanding of the relationships between the predictors and the TARGET variable.

There were also 3359 observations that were deleted due to missingness, this was due to the missing values in the STAR column. Since most of the STAR ratings were either 2 or 3, we chose not to replace these NA values with the median or mean as it would create a bias in the data. Another take away from the results is that the residual deviance is less than the null deviance suggesting that the model is useful in predicting the response variable.

### Predictions using the evaluation data set

Next, let's deploy this negative on the test data set.

```
glimpse(test_wine)

## Rows: 3,335
## Columns: 16
## $ IN <int> 3, 9, 10, 18, 21, 30, 31, 37, 39, 47, 60, 62, 63, 6~
## $ TARGET <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ FixedAcidity <dbl> 5.4, 12.4, 7.2, 6.2, 11.4, 17.6, 15.5, 15.9, 11.6, ~
## $ VolatileAcidity <dbl> -0.860, 0.385, 1.750, 0.100, 0.210, 0.040, 0.530, 1~
## $ CitricAcid <dbl> 0.27, -0.76, 0.17, 1.80, 0.28, -1.15, -0.53, 1.14, ~
## $ ResidualSugar <dbl> -10.70, -19.70, -33.00, 1.00, 1.20, 1.40, 4.60, 31.~
## $ Chlorides <dbl> 0.092, 1.169, 0.065, -0.179, 0.038, 0.535, 1.263, --
## $ FreeSulfurDioxide <dbl> 23, -37, 9, 104, 70, -250, 10, 115, 35, 40, NA, -21~
## $ TotalSulfurDioxide <dbl> 398, 68, 76, 89, 53, 140, 17, 381, 83, 129, 583, -5~
## $ Density <dbl> 0.98527, 0.99048, 1.04641, 0.98877, 1.02899, 0.9502~
## $ pH <dbl> 5.02, 3.37, 4.61, 3.20, 2.54, 3.06, 3.07, 2.99, 3.3~
## $ Sulphates <dbl> 0.64, 1.09, 0.68, 2.11, -0.07, -0.02, 0.75, 0.31, 2~
## $ Alcohol <dbl> 12.30, 16.00, 8.55, 12.30, 4.80, 11.40, 8.50, 11.40~
## $ LabelAppeal <int> -1, 0, 0, -1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, -1, -1,~
## $ AcidIndex <int> 6, 6, 8, 8, 10, 8, 12, 7, 12, 7, 8, 10, 9, 8, 9, 11~
## $ STARS <int> NA, 2, 1, 1, NA, 4, 3, NA, NA, NA, 1, NA, 2, NA, NA~
```

Here is a some summary statistics on the evaluation set.

```
summary_stats <- summary(test_wine)
print(summary_stats)
```

##	IN	TARGET	FixedAcidity	VolatileAcidity
##	Min. : 3	Mode:logical	Min. : -18.200	Min. : -2.8300
##	1st Qu.: 4018	NA's:3335	1st Qu.: 5.200	1st Qu.: 0.0800
##	Median : 7906		Median : 6.900	Median : 0.2800
##	Mean : 8048		Mean : 6.864	Mean : 0.3103
##	3rd Qu.: 12061		3rd Qu.: 9.000	3rd Qu.: 0.6300
##	Max. : 16130		Max. : 33.500	Max. : 3.6100
##				
##	CitricAcid	ResidualSugar	Chlorides	FreeSulfurDioxide
##	Min. : -3.1200	Min. : -128.300	Min. : -1.15000	Min. : -563.00
##	1st Qu.: 0.0000	1st Qu.: -2.600	1st Qu.: 0.01600	1st Qu.: 3.00
##	Median : 0.3100	Median : 3.600	Median : 0.04700	Median : 30.00
##	Mean : 0.3124	Mean : 5.319	Mean : 0.06143	Mean : 34.95
##	3rd Qu.: 0.6050	3rd Qu.: 17.200	3rd Qu.: 0.17100	3rd Qu.: 79.25

```
## Max. : 3.7600 Max. : 145.400 Max. : 1.26300 Max. : 617.00
## NA's :168 NA's :138 NA's :152
## TotalSulfurDioxide Density pH Sulphates
## Min. : -769.00 Min. : 0.8898 Min. : 0.600 Min. : -3.0700
## 1st Qu.: 27.25 1st Qu.: 0.9883 1st Qu.: 2.980 1st Qu.: 0.3300
## Median : 124.00 Median : 0.9946 Median : 3.210 Median : 0.5000
## Mean : 123.41 Mean : 0.9947 Mean : 3.237 Mean : 0.5346
## 3rd Qu.: 210.00 3rd Qu.: 1.0005 3rd Qu.: 3.490 3rd Qu.: 0.8200
## Max. : 1004.00 Max. : 1.0998 Max. : 6.210 Max. : 4.1800
## NA's :157 NA's :104 NA's :310
## Alcohol LabelAppeal AcidIndex STARS
## Min. : -4.20 Min. : -2.00000 Min. : 5.000 Min. : 1.00
## 1st Qu.: 9.00 1st Qu.: -1.00000 1st Qu.: 7.000 1st Qu.: 1.00
## Median : 10.40 Median : 0.00000 Median : 8.000 Median : 2.00
## Mean : 10.58 Mean : 0.01349 Mean : 7.748 Mean : 2.04
## 3rd Qu.: 12.50 3rd Qu.: 1.00000 3rd Qu.: 8.000 3rd Qu.: 3.00
## Max. : 25.60 Max. : 2.00000 Max. : 17.000 Max. : 4.00
## NA's :185 NA's :841
```

Since we transformed the STARS variable to a factor in training data set, we will do so with the test data set as well.

```
unique(test_wine$STARS)
```

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

```
# Convert "STARS" to factor
```

```
test_wine$STARS <- as.factor(test_wine$STARS)
```

```
# Verify the transformation
```

```
str(test_wine$STARS)
```

```
## Factor w/ 4 levels "1","2","3","4": NA 2 1 1 NA 4 3 NA NA NA ...
```

Next, we deploy our count regression model to make predictions on the evaluation data set. We saved the predictions under pred. Here is a glimpse of the data:

```
test_wine$pred <- predict(nb_model3a, type = "response", newdata = test_wine)
```

```
glimpse(test_wine)
```

```
## Rows: 3,335
```

```
## Columns: 17
```

```
## $ IN <int> 3, 9, 10, 18, 21, 30, 31, 37, 39, 47, 60, 62, 63, 6~
## $ TARGET <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ FixedAcidity <dbl> 5.4, 12.4, 7.2, 6.2, 11.4, 17.6, 15.5, 15.9, 11.6, ~
## $ VolatileAcidity <dbl> -0.860, 0.385, 1.750, 0.100, 0.210, 0.040, 0.530, 1~
## $ CitricAcid <dbl> 0.27, -0.76, 0.17, 1.80, 0.28, -1.15, -0.53, 1.14, ~
## $ ResidualSugar <dbl> -10.70, -19.70, -33.00, 1.00, 1.20, 1.40, 4.60, 31.~
## $ Chlorides <dbl> 0.092, 1.169, 0.065, -0.179, 0.038, 0.535, 1.263, --
## $ FreeSulfurDioxide <dbl> 23, -37, 9, 104, 70, -250, 10, 115, 35, 40, -21~
## $ TotalSulfurDioxide <dbl> 398, 68, 76, 89, 53, 140, 17, 381, 83, 129, 583, -5~
## $ Density <dbl> 0.98527, 0.99048, 1.04641, 0.98877, 1.02899, 0.9502~
## $ pH <dbl> 5.02, 3.37, 4.61, 3.20, 2.54, 3.06, 3.07, 2.99, 3.3~
## $ Sulphates <dbl> 0.64, 1.09, 0.68, 2.11, -0.07, -0.02, 0.75, 0.31, 2~
## $ Alcohol <dbl> 12.30, 16.00, 8.55, 12.30, 4.80, 11.40, 8.50, 11.40~
## $ LabelAppeal <int> -1, 0, 0, -1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, -1, -1, ~
```

```
## $ AcidIndex      <int> 6, 6, 8, 8, 10, 8, 12, 7, 12, 7, 8, 10, 9, 8, 9, 11~
## $ STARS          <fct> NA, 2, 1, 1, NA, 4, 3, NA, NA, NA, 1, NA, 2, NA, NA~
## $ pred           <dbl> NA, 4.036380, 2.653745, 2.211254, NA, 5.507260, 3.3~
```