Count Regression Modeling for Wine Demand Prediction

Contents

Introduction

This analysis aims to predict wine purchase counts based on a variety of factors, including expert ratings, label appeal, and chemical properties. Using the wine dataset, I conduct a comprehensive exploration of the data, handle missing values, and analyze key variables to prepare for model development. Several regression models, including Poisson, Negative Binomial, and Multiple Linear Regression (MLR), are evaluated to determine the best predictor for sales outcomes. Performance metrics such as AIC, RMSE, and residual deviance are used to compare model fits, and predictions are assessed using confusion matrices. Ultimately, the goal is to select the most effective model for predicting wine sales, offering insights into the factors that drive consumer purchasing behavior.

Part 1: Data Exploration

In this section, I conduct an in-depth exploration of the wine dataset, including loading the data, understanding its structure, and analyzing the key variables. This process helps identify important patterns, relationships, and potential data issues, providing a solid foundation for model development.

Key steps include:

1. **Loading the Wine Data**: Importing the dataset and assessing its size and structure.
2. **Variable Names and Descriptions**: Clarifying the meaning of each variable and its relevance.
3. **Summary Statistics**: Analyzing basic statistics and skewness to understand data distribution and detect outliers or inconsistencies.
4. **Handling Negative Values**: Addressing data quality issues by replacing chemically impossible negative values with NA.
5. **Missing Value Analysis**: Examining the extent and pattern of missing values across the dataset, followed by imputation to ensure data completeness.
6. **Correlation Analysis**: Exploring the relationship between predictors and the sales outcome (TARGET) to identify the most influential variables and potential multicollinearity issues.

The goal of this part is to clean and prepare the data for further modeling and analysis in later sections.

Loading the Wine Data

In this first step, I load the wine training dataset into R and report the basic size of the data. I summarize the number of observations (rows) and variables (columns) in a small table so we know how large the dataset is before moving on to more detailed analysis.

```r
# Load required package for a clean-looking table
library(knitr)      # for kable()

# Read the wine training data
# Use forward slashes in the Windows file path
wine  <-  read.csv("C:/Users/Hp/Downloads/wine-data-1-1.csv")

# Calculate number of rows and columns
n_rows <- nrow(wine)
n_cols  <-  ncol(wine)

# Create a small summary table for dataset size
size_table  <-  data.frame(
  Measure = c("Number of observations (rows)",
              "Number of variables (columns)"),
  Value     = c(n_rows,  n_cols)
)

# Print the table in a clean format
```

```r
kable(size_table,
      caption = "Size of the Wine Training Dataset")
```

Table 1: Size of the Wine Training Dataset

Measure	Value
Number of observations (rows)	12795
Number of variables (columns)	16

Interpretation of Dataset Size

The wine training dataset comprises **12,795 observations** and **16 variables**. This large sample size is beneficial for detecting meaningful patterns and building reliable predictive models. The number of variables is manageable, allowing for an effective exploration of the relationships between chemical properties, label appeal, expert ratings, and the number of cases purchased (`TARGET`).

Variable Names and Descriptions

The table below summarizes each variable in the wine dataset along with a brief description of what it represents. This helps clarify the meaning of each predictor before conducting deeper analysis.

```r
library(knitr)

variable_table <- data.frame(
  Variable = c(
    "INDEX", "TARGET", "AcidIndex", "Alcohol", "Chlorides", "CitricAcid",
    "Density", "FixedAcidity", "FreeSulfurDioxide", "LabelAppeal",
    "ResidualSugar", "STARS", "Sulphates", "TotalSulfurDioxide",
    "VolatileAcidity", "pHOfWine"
  ),
  Description = c(
    "Identification variable",
    "Number of cases purchased",
    "Proprietary method of testing total acidity of wine (weighted average)",
    "Alcohol content of wine",
    "Chloride content of wine",
    "Citric Acid content",
    "Density of wine",
    "Fixed acidity of wine",
    "Sulfur dioxide content of wine",
    "Marketing score indicating the appeal of label design (higher = more appealing)",
    "Residual sugar of wine",
    "Wine rating by experts (1 to 4 stars)",
    "Sulfate content of wine",
    "Total sulfur dioxide of wine",
    "Volatile acid content of wine",
    "pH level of the wine"
  )
)

kable(variable_table,
      caption = "Variable Names and Descriptions for Wine Dataset")
```

Table 2: Variable Names and Descriptions for Wine Dataset

Variable	Description
INDEX	Identification variable
TARGET	Number of cases purchased
AcidIndex	Proprietary method of testing total acidity of wine (weighted average)
Alcohol	Alcohol content of wine
Chlorides	Chloride content of wine
CitricAcid	Citric Acid content
Density	Density of wine
FixedAcidity	Fixed acidity of wine
FreeSulfurDioxide	Sulfur dioxide content of wine
LabelAppeal	Marketing score indicating the appeal of label design (higher = more appealing)
ResidualSugar	Residual sugar of wine
STARS	Wine rating by experts (1 to 4 stars)
Sulphates	Sulfate content of wine
TotalSulfurDioxide	Total sulfur dioxide of wine
VolatileAcidity	Volatile acid content of wine
pHOfWine	pH level of the wine

The table above summarizes the variables in the wine dataset, including chemical characteristics (e.g., acidity, sulfur, alcohol, density), marketing attributes (LabelAppeal), and expert ratings (STARS). Understanding these variables is crucial for identifying features that influence wine purchases (TARGET) and guiding subsequent analysis and modeling.

Summary Statistics for Key Wine Variables

To understand the wine dataset, I compute summary statistics (mean, standard deviation, median, min, max) for all numeric variables. These metrics describe the central tendency and spread of the data, highlighting potential outliers or unusual ranges. I also examine skewness to identify non-symmetrical distributions, which may need transformation during modeling. The table below summarizes these statistics for all numeric variables (excluding the ID field).

```r
library(psych)
library(knitr)

# Drop INDEX since it is just an ID
wine_num <- wine |> dplyr::select(-INDEX)

# Use psych::describe to get descriptive statistics
desc_full  <-  describe(wine_num)

# Keep only the columns we care about
desc_selected <- desc_full[, c("mean", "sd", "median", "min", "max", "skew")]

# Round for nicer display
desc_rounded <- round(desc_selected, 3)

# Create a well-formatted table with kable
kable(
  desc_rounded,
  caption = "Summary Statistics for Wine Variables"
)
```

Table 3: Summary Statistics for Wine Variables

	mean	sd	median	min	max	skew
TARGET	3.029	1.926	3.000	0.000	8.000	-0.326
FixedAcidity	7.076	6.318	6.900	-18.100	34.400	-0.023
VolatileAcidity	0.324	0.784	0.280	-2.790	3.680	0.020
CitricAcid	0.308	0.862	0.310	-3.240	3.860	-0.050
ResidualSugar	5.419	33.749	3.900	-127.800	141.150	-0.053
Chlorides	0.055	0.318	0.046	-1.171	1.351	0.030
FreeSulfurDioxide	30.846	148.715	30.000	-555.000	623.000	0.006
TotalSulfurDioxide	120.714	231.913	123.000	-823.000	1057.000	-0.007
Density	0.994	0.027	0.994	0.888	1.099	-0.019
pHOfWine	3.208	0.680	3.200	0.480	6.130	0.044
Sulphates	0.527	0.932	0.500	-3.130	4.240	0.006
Alcohol	10.489	3.728	10.400	-4.700	26.500	-0.031
LabelAppeal	-0.009	0.891	0.000	-2.000	2.000	0.008
AcidIndex	7.773	1.324	8.000	4.000	17.000	1.648
STARS	2.042	0.903	2.000	1.000	4.000	0.447

Key Insights from Summary Statistics

The summary statistics provide a clear picture of how the wine variables are distributed and highlight several important patterns:

- **TARGET (mean = 3.03, median = 3, min = 0, max = 8)**: Sales per wine are low and tightly concentrated. No extreme outliers, suggesting stable target behavior.
- **FixedAcidity (mean = 7.08, sd = 6.32, min = −18.10, max = 34.40)**: Extremely wide range indicates possible data quality issues or outliers. Skewness near zero suggests symmetry aside from the extreme values.
- **VolatileAcidity and CitricAcid (means around 0.32 and 0.31, skew 0)**: Both variables are centered near typical wine chemistry values. Distributions appear well-behaved and symmetric.
- **ResidualSugar (mean = 5.42, sd = 33.75, min = −127.80, max = 141.15)**: Very high variability and unrealistic negative values indicate data inconsistencies that may need cleaning.
- **FreeSulfurDioxide (mean = 30.85, sd = 148.72) and TotalSulfurDioxide (mean = 120.71, sd = 231.91)**: Extremely large standard deviations show wide differences across wines. Likely presence of extreme outliers or measurement inconsistencies.
- **Density (mean = 0.994, range = 0.888 to 1.099)**: Values fall within expected density ranges for wine. Low variability and near-zero skewness indicate a stable distribution.
- **pHOfWine (mean = 3.21, median = 3.20)**: Typical pH range for wines; distribution appears symmetric and realistic.
- **Sulphates (mean = 0.53, sd = 0.93)**: Moderate variation; minimum value is unrealistically negative (–3.13), suggesting possible recording errors.
- **Alcohol (mean = 10.49, median = 10.40, min = −4.70, max = 26.50)**: Maximum is plausible for fortified wines, but negative minimum is invalid, indicating a data entry mistake.
- **LabelAppeal (mean −0.01, sd = 0.89)**: Centered around zero, meaning label appeal scores vary evenly between positive and negative perceptions.
- **AcidIndex (mean = 7.77, median = 8, min = 4, max = 17, skew = 1.65)**: Highly positively skewed, showing that high acidity values occur for a small subset of wines.
- **STARS (mean = 2.04, sd = 0.90, min = 1, max = 4, skew = 0.45)**: Expert ratings are slightly skewed upward, indicating that lower ratings are more common than very high ratings.

Overall, most chemical variables behave reasonably well, but a few, including ResidualSugar, sulfur measures, and Alcohol, show unrealistic minimum values or extreme variability. These may need to be cleaned or transformed before modeling.

Handling Chemically Impossible Negative Values

The summary statistics revealed negative values in chemistry variables like ResidualSugar, Alcohol, and sulfur- related measures. These values are unrealistic and indicate data entry errors, so I replace them with NA. However, marketing variables like LabelAppeal can legitimately have negative values, as they represent customer dislike for a label design. Therefore, only chemistry variables are cleaned, and marketing variables are left unchanged. The cleaned dataset is referred to as wine_clean.

```r
# Create a cleaned copy of the original dataset
wine_clean <- wine

# Chemistry variables where negative values are physically impossible
chem_vars <- c(
"FixedAcidity",
"VolatileAcidity",
"CitricAcid",
"ResidualSugar",
"Chlorides",
"FreeSulfurDioxide",
"TotalSulfurDioxide",
"Sulphates",
"Alcohol"
)

# Replace negative values in chemistry variables with NA
wine_clean[chem_vars] <- lapply(wine_clean[chem_vars], function(x) {
ifelse(x < 0, NA, x)
})

# Quick check: minimum value for each chemistry variable after cleaning,
# formatted as a clean table

library(knitr)

min_values <- sapply(wine_clean[chem_vars], min, na.rm = TRUE)

min_table <- data.frame(
Variable         = names(min_values),
MinAfterCleaning = as.numeric(min_values),
row.names = NULL
)

kable(
min_table,
caption = "Minimum Values of Chemistry Variables After Replacing Negatives with NA"
)
```

Table 4: Minimum Values of Chemistry Variables After Replacing Negatives with NA

Variable	MinAfterCleaning
FixedAcidity	0
VolatileAcidity	0
CitricAcid	0
ResidualSugar	0
Chlorides	0
FreeSulfurDioxide	0

Variable	MinAfterCleaning
TotalSulfurDioxide	0
Sulphates	0
Alcohol	0

Interpretation of Negative-Value Cleaning

Table 4 confirms that, after replacing all chemically impossible negative values with NA, the minimum observed value for each chemistry variable is now 0. This means there are no remaining negative measurements for acidity, sugar, sulfur content, or alcohol. Treating these implausible negatives as missing values ensures that the dataset better reflects realistic wine chemistry while preserving valid variation in the variables. In particular, this step improves data quality without altering meaningful negative scores in LabelAppeal, which continue to capture customer dislike for certain label designs.

Missing Value Analysis

After replacing chemically impossible negative values with NA, the next step is to assess the remaining missing values in the cleaned dataset. Understanding the missingness pattern is crucial for selecting the right imputation method and ensuring model reliability. The table below summarizes the number and percentage of missing values for each variable in wine_clean, focusing only on those with missing data.

```r
library(knitr)
library(kableExtra)

# Count missing values after cleaning negative entries
missing_counts <- colSums(is.na(wine_clean))

# Build a summary table with counts and percentages
missing_summary <- data.frame(
  Variable          = names(missing_counts),
  MissingCount      = missing_counts,
  MissingPercentage = round((missing_counts / nrow(wine_clean)) * 100, 3)
)

# Keep only variables that actually contain missing values
missing_summary <- missing_summary[missing_summary$MissingCount > 0, ]

# Remove row names for clean formatting
rownames(missing_summary) <- NULL

# Display results as a nicely formatted table
kable(missing_summary,
      caption = "Summary of Missing Values After Cleaning Negative Entries") %>%
  kable_styling(
    bootstrap_options = c("striped", "hover", "condensed"),
    full_width = FALSE,
    position   = "center"
  )
```

Table 5: Summary of Missing Values After Cleaning Negative Entries

Variable	MissingCount	MissingPercentage
FixedAcidity	1621	12.669
VolatileAcidity	2827	22.095

CitricAcid	2966	23.181
ResidualSugar	3752	29.324
Chlorides	3835	29.973
FreeSulfurDioxide	3683	28.785
TotalSulfurDioxide	3186	24.900
pHOfWine	395	3.087
Sulphates	3571	27.909
Alcohol	771	6.026
STARS	3359	26.252

Interpretation of Missing Value Patterns

The missing value summary shows that a noticeable portion of the dataset now contains missing observations, especially among the chemistry variables. This is expected because many of the chemically impossible negative values were replaced with NA during the cleaning step. Variables such as ResidualSugar, Chlorides, CitricAcid, and the sulfur dioxide measures display missingness in the range of twenty to thirty percent, indicating that a significant number of original entries were invalid. Since these variables describe key aspects of wine composition, removing the affected rows would eliminate too much useful information.

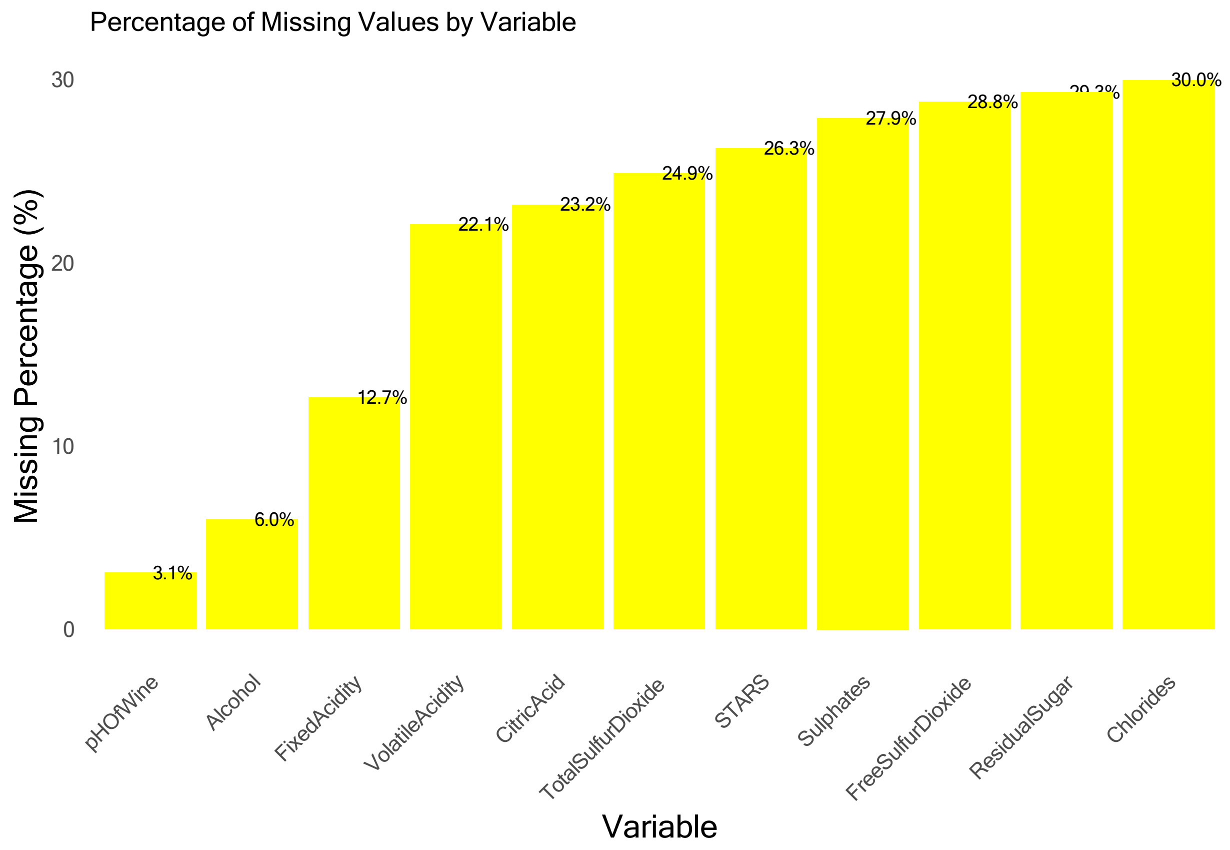
The STARS variable also has a relatively high amount of missing data. Because expert ratings are strongly associated with the number of cases purchased, dropping these observations would risk distorting the relationship between perceived quality and sales. For this reason, simple row deletion is not appropriate.

Overall, the level and pattern of missingness make it necessary to apply an imputation approach that preserves important information while maintaining the structure of the dataset. This ensures that the final dataset remains suitable for reliable model development in the following sections.

Visualization of Missing Values with Percentages

To better understand the distribution of missing values across the dataset, we can visualize the percentage of missing data for each variable.

```r
# Load necessary libraries
library(ggplot2)
library(dplyr)

# Calculate the percentage of missing values for each variable
missing_percentage <- colSums(is.na(wine_clean)) / nrow(wine_clean) * 100

# Convert to a data frame for plotting
missing_data <- data.frame(
  Variable = names(missing_percentage),
  MissingPercentage = missing_percentage
)

# Filter to keep only variables with missing data
missing_data <- missing_data[missing_data$MissingPercentage > 0, ]

# Order the data from highest to lowest missing percentage
missing_data <- missing_data[order(-missing_data$MissingPercentage), ]

# Plot missing data percentages with vertical bars
ggplot(missing_data, aes(x = reorder(Variable, MissingPercentage), y = MissingPercentage)) +
  geom_bar(stat = "identity", fill = "yellow") +
  geom_text(aes(label = paste0(sprintf("%.1f", MissingPercentage), "%")),
            position = position_dodge(width = 0.9),
            hjust = -0.05, size = 2.5) +  # Reduced font size for better fit
  labs(
    title = "Percentage of Missing Values by Variable",
    x = "Variable",
    y = "Missing Percentage (%)"
  ) +
  theme_minimal(base_size = 12) +  # Smaller base font size for better scaling
  theme(
    axis.text.x = element_text(size = 8, angle = 45, hjust = 1),  # Smaller font for x-axis labels
    axis.text.y = element_text(size = 8),   # Smaller font for y-axis labels
    plot.title = element_text(size = 10),   # Smaller title size
    plot.background = element_blank(),       # Remove plot background
    panel.background = element_blank(),      # Remove panel background
    panel.grid.major = element_blank(),      # Remove major gridlines
    panel.grid.minor = element_blank()       # Remove minor gridlines
  )
```



Interpretation of Missing Value Patterns

The bar chart above shows the percentage of missing values for each variable, with the percentages rounded to one decimal place. Variables such as Chlorides, ResidualSugar, and FreeSulfurDioxide exhibit high missing data, ranging from **28.0% to 30.0%**, indicating potential data quality issues. Other variables, like pHOfWine and Alcohol, have relatively lower missingness (**3.1% to 12.7%**). Additionally, the STARS variable, representing expert wine ratings, has **26.3%** missing data, which could impact model performance. Given these patterns, imputation methods should be considered to address the missing values and preserve the integrity of the dataset for analysis.

Creating Missing Value Flags and Imputing Missing Values

Given the significant missingness in chemistry variables and STARS ratings, I create missing-value indicator variables. These binary flags (1 for missing, 0 for non-missing) are included in the model to capture if the absence of data carries predictive value, as noted in the assignment.

For imputation, continuous chemistry variables are filled with the median, which is robust to skewness. The STARS variable, being ordinal, is imputed with the mode (most frequent rating), preserving its interpretation within the 1–4 scale.

```r
suppressPackageStartupMessages(library(dplyr))

# Work from wine_clean (negative values already converted to NA)
data_imp <- wine_clean

# Identify continuous variables to median-impute (exclude INDEX and STARS)
continuous_vars <- c(
  "FixedAcidity", "VolatileAcidity", "CitricAcid", "ResidualSugar",
  "Chlorides", "FreeSulfurDioxide", "TotalSulfurDioxide",
  "Sulphates", "Alcohol", "pHOfWine"
)

# 1. Create missing-value flags for continuous variables
for (v in continuous_vars) {
  flag_name <- paste0(v, "_Missing")
  data_imp[[flag_name]] <- ifelse(is.na(data_imp[[v]]), 1, 0)
}

# 2. Create missing-value flag for STARS
data_imp$STARS_Missing <- ifelse(is.na(data_imp$STARS), 1, 0)

# 3. Median imputation for continuous variables
for (v in continuous_vars) {
  med_val <- median(data_imp[[v]], na.rm = TRUE)
  data_imp[[v]][is.na(data_imp[[v]])]  <-  med_val
}

# 4. Mode imputation for STARS
mode_stars <- as.numeric(names(sort(table(data_imp$STARS), decreasing = TRUE)[1]))
data_imp$STARS[is.na(data_imp$STARS)] <- mode_stars
```

Interpretation of Imputation Strategy

This two-step approach preserves essential information about missingness while producing a complete dataset for modeling. The missing-value flags retain the signal that certain observations were originally incomplete, and the models can later learn whether these patterns are associated with differences in wine sales.

Median imputation stabilizes the continuous chemistry variables without distorting their ranges or creating unrealistic values. For STARS, mode imputation is appropriate because the variable is ordinal and takes on a limited set of discrete values. Together, these steps yield a clean and fully imputed dataset that is ready for splitting into training and testing sets in the next stage.

Verifying That All Missing Values Have Been Imputed

After creating missing-value flags and performing median and mode imputation, it is important to confirm that no missing entries remain in the working dataset. This check ensures that the data frame data_imp is fully complete and ready to be used for the train–test split and subsequent modeling steps.

```r
# Total number of missing values remaining in the imputed dataset

total_missing_after_imp <- sum(is.na(data_imp))
total_missing_after_imp
```

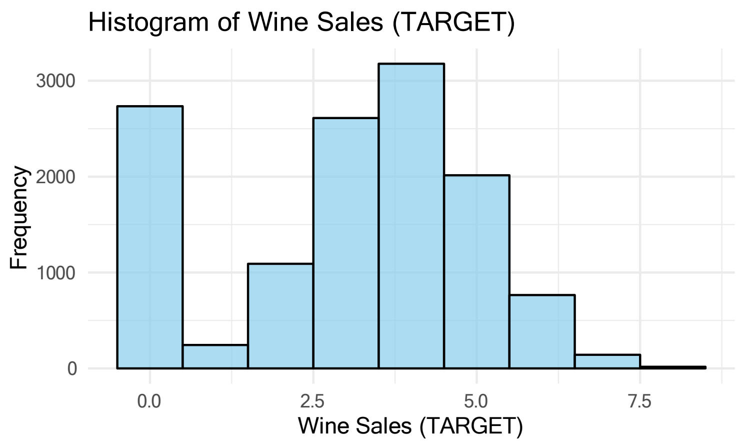
```
## [1] 0
```

The result of this check is 0, which confirms that all missing values have been successfully imputed. The dataset data_imp is therefore complete and can now be used as the basis for creating the training and testing sets in the next stage of the analysis.

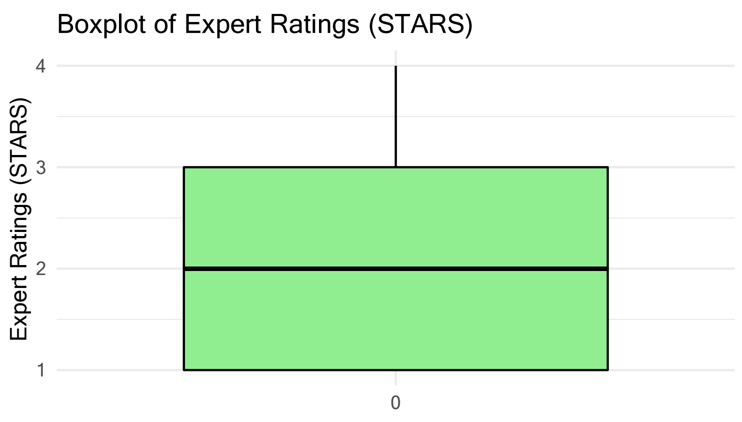
Visualizations of Key Variables

To better understand the distribution of key variables related to wine sales, I visualize the data using histograms and box plots. These visualizations help reveal patterns, detect outliers, and give insight into the distribution of important variables such as **TARGET**, **STARS**, and **LabelAppeal**.

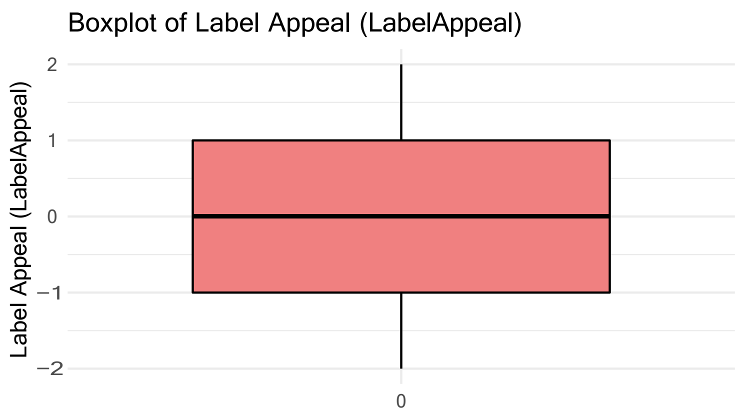
```r
# Load necessary libraries
library(ggplot2)

# Visualize the distribution of TARGET (wine sales)
# Histogram of TARGET (Wine Sales)
ggplot(wine_clean, aes(x = TARGET)) +
  geom_histogram(binwidth = 1, fill = "skyblue", color = "black", alpha = 0.7) +
  labs(title = "Histogram of Wine Sales (TARGET)", x = "Wine Sales (TARGET)", y = "Frequency") +
  theme_minimal()
```



```r
# Boxplot for STARS (Expert ratings)
# We clean the data to avoid non-finite values in STARS
cleaned_wine <- wine_clean[!is.na(wine_clean$STARS) & is.finite(wine_clean$STARS), ]

ggplot(cleaned_wine, aes(y = STARS, x = factor(0))) +
  geom_boxplot(fill = "lightgreen", color = "black") +
  labs(title = "Boxplot of Expert Ratings (STARS)", x = "", y = "Expert Ratings (STARS)") +
  theme_minimal()
```



```r
# Boxplot for LabelAppeal (Marketing Appeal)
# Clean the data to avoid non-finite values in LabelAppeal
cleaned_wine_label <- wine_clean[!is.na(wine_clean$LabelAppeal) & is.finite(wine_clean$LabelAppeal), ]

ggplot(cleaned_wine_label, aes(y = LabelAppeal, x = factor(0))) +
  geom_boxplot(fill = "lightcoral", color = "black") +
  labs(title = "Boxplot of Label Appeal (LabelAppeal)", x = "", y = "Label Appeal (LabelAppeal)") +
  theme_minimal()
```



Insights from Visualizations of Wine Sales and Attributes

Above are visualizations of key variables, including **Wine Sales (TARGET)**, **Expert Ratings (STARS)**, and **Label Appeal (LabelAppeal)**. These plots offer insights into the distribution and variability of important factors influencing wine purchasing behavior.

1. **Histogram of Wine Sales (TARGET)**

The **histogram of Wine Sales (TARGET)** shows **two notable features**: a spike at 0 purchases (wines with no sample orders) and a peak around 3-4 cases. The distribution is **approximately symmetric** with a **slight negative skew**, where the mean and median are nearly identical. Most sales fall between 2 and 5 cases, tapering off at both extremes. The distribution shows moderate variability in purchase counts across the dataset.

2. **Boxplot of Expert Ratings (STARS)**

The **boxplot of Expert Ratings (STARS)** reveals that most wines are rated between **2 and 3 stars** (the interquartile range). The **median** rating is **2 stars**, indicating that half of all rated wines receive 2 stars or below. The whiskers extend from **1 to 4 stars**, covering the full rating scale. This suggests that wines tend to receive **moderate ratings**, with the majority clustering in the middle of the scale.

3. **Boxplot of Label Appeal (LabelAppeal)**

The **boxplot of Label Appeal (LabelAppeal)** shows that most wines have neutral to slightly above-average appeal in terms of their labels, with values centered around **0** (the median). The **interquartile range spans from -1 to +1**, indicating that most wines have moderate label appeal. The whiskers extend from **-2 to +2**, covering the full rating scale with a symmetric distribution. This suggests **diversity in consumer perceptions** of label appeal, with the majority of wines clustering near neutral.

Correlation Analysis: Exploring the Relationship Between Predictors and Sales Outcome (TARGET)

With the cleaned and imputed dataset, I analyze how each predictor relates to the sales outcome (TARGET). This step aims to identify key variables most strongly associated with wine sales and flag any potential multicollinearity.

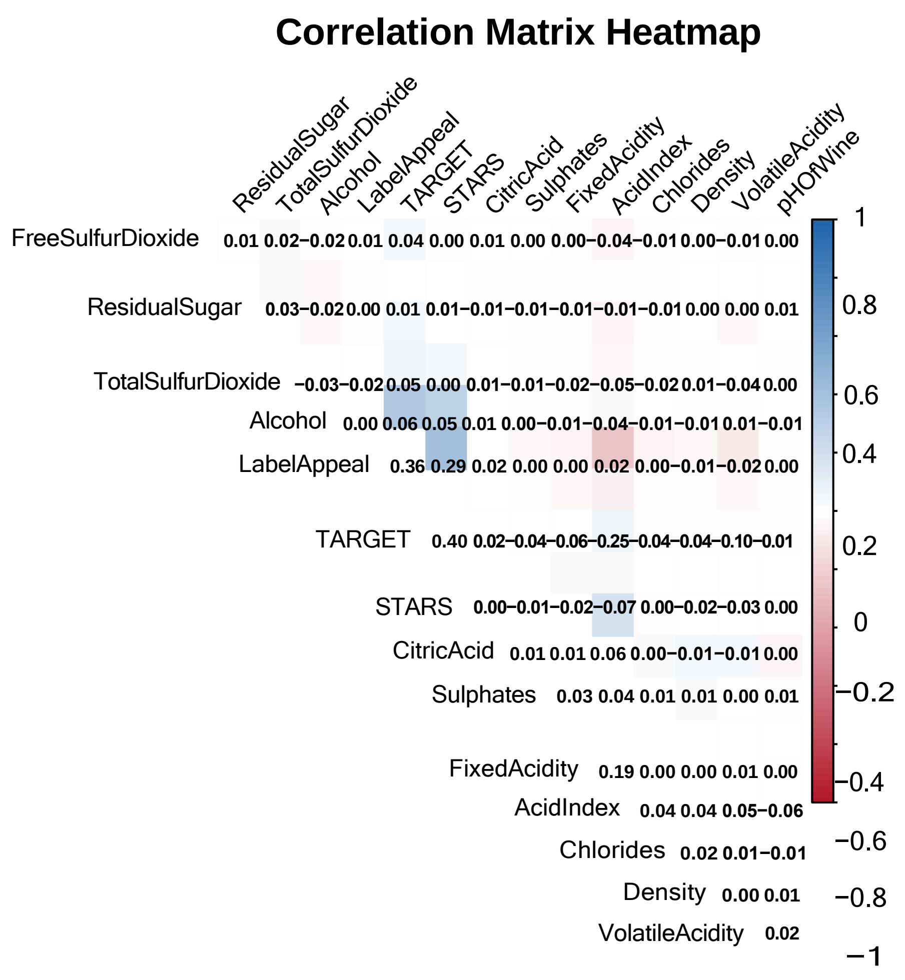
I exclude missing-value flags and the INDEX identifier, focusing on numeric variables. Then, I compute the correlation of each predictor with TARGET, sorting the results by the absolute strength of the relationship.

```r
# Load necessary libraries
library(dplyr)
library(knitr)
library(ggplot2)
library(corrplot)

# Use imputed dataset, remove ID and missing-value flags, keep numeric variables only
cor_data <- data_imp %>%
  select(-INDEX) %>%             # remove identifier
  select(-contains("_Missing")) %>%  # drop missing-value flags
  select(where(is.numeric))          # keep only numeric variables

# Compute correlation matrix
cor_matrix <- cor(cor_data, use = "complete.obs")

# Extract correlations with TARGET
cor_target <- cor_matrix["TARGET", ]

# Build table, drop TARGET itself, and sort by absolute correlation strength
cor_table <- data.frame(
  Variable = names(cor_target),
  CorrelationWithTARGET = round(as.numeric(cor_target), 3)
) %>%
  filter(Variable != "TARGET") %>%
  arrange(desc(abs(CorrelationWithTARGET)))

# Display correlation table
kable(
  cor_table,
  caption = "Correlation of Each Variable with TARGET (Sorted by Strength)",
  row.names = FALSE
)
```

Table 6: Correlation of Each Variable with TARGET (Sorted by Strength)

Variable	CorrelationWithTARGET
STARS	0.400
LabelAppeal	0.357
AcidIndex	-0.246
VolatileAcidity	-0.097
Alcohol	0.064
FixedAcidity	-0.056
TotalSulfurDioxide	0.049
Chlorides	-0.044
FreeSulfurDioxide	0.043
Sulphates	-0.037
Density	-0.036
CitricAcid	0.016
ResidualSugar	0.013
pHOfWine	-0.009

```
# Create a correlation heatmap with adjusted font sizes and layout for better readability
corrplot(cor_matrix,
method = "color", # Color squares
type = "upper", # Upper triangle only
order = "hclust", # Cluster similar variables
addCoef.col = "black", # Add correlation coefficients
number.cex = 0.55, # Size of correlation numbers
tl.col = "black", # Label color
tl.srt = 45, # Rotate labels 45 degrees
tl.cex = 0.75, # Size of variable labels
col = colorRampPalette(c("#B2182B", "white", "#2166AC"))(100), # Red-White-Blue
diag = FALSE, # Hide diagonal
mar = c(1, 1, 2, 1), # Tighter margins
title = "Correlation Matrix Heatmap")
```



Interpretation of Correlations with TARGET

The correlation results provide several important insights into which predictors are most closely associated with wine sales:

- **STARS shows the strongest positive correlation (0.400)**, indicating that expert ratings are an important driver of purchase volume. Wines with higher ratings tend to sell more cases.
- **LabelAppeal also has a strong positive correlation (0.357)**, suggesting that label design and marketing appeal significantly influence customer purchasing decisions.
- **AcidIndex exhibits a moderate negative correlation (-0.246)**, which may imply that wines with very high acidity values are associated with lower sales.
- **Most chemistry variables (Alcohol, FixedAcidity, sulfur measures, and others) show very weak correlations with TARGET**, indicating that their individual marginal relationships with sales are limited.
- **VolatileAcidity has a small negative correlation**, while variables like TotalSulfurDioxide, Chlorides, and FreeSulfurDioxide show very mild positive relationships.
- **ResidualSugar, CitricAcid, and pHOfWine show almost no correlation** with TARGET, suggesting little direct effect on sales.

Overall, these results imply that quality signals and marketing-related attributes are likely to play a central role in predicting sales, while chemistry variables may contribute more nuanced effects when included together in multivariate models.

Part 2: Data Preparation

In this section, I will continue preparing the dataset for model building by addressing key aspects such as splitting the data, handling skewness, and engineering features. The goal is to clean and transform the data, ensuring it is suitable for fitting predictive models.

Key steps include:

1. **Train–Test Split (80/20)**: Dividing the dataset into training and testing subsets. This is done before any transformations to avoid information leakage and ensure unbiased evaluation of model performance.
2. **Skewness Analysis**: Evaluating the distribution of numeric predictors to identify variables with high skewness that may need transformation.
3. **Log Transformations for Highly Skewed Variables**: Applying log transformations to variables that exhibit high skewness to improve model performance.
4. **Feature Engineering**: Creating new features from existing predictors to capture more meaningful relationships in the data.
5. **Final Dataset Preparation**: After transformations and feature engineering, reviewing the final set of variables and their distributions.

By the end of this section, the dataset will be fully prepared for modeling and ready for analysis in the next steps.

Train–Test Split (80/20)

With the data fully cleaned and imputed, the next step is to split the dataset into a training set and a testing set. The training set will be used to explore variable distributions, compute skewness, apply transformations, engineer new features, and build all predictive models. The test set will remain untouched until final model evaluation.

To avoid information leakage, the split is performed before any transformations or feature engineering. This ensures that all modeling decisions are based solely on the training data.

```r
set.seed(123)  # ensures reproducibility

# Number of rows
n <- nrow(data_imp)

# Random 80% sample for training
train_index <- sample(1:n, size = 0.8 * n)

# Train and test sets
train_data <- data_imp[train_index, ]
test_data  <- data_imp[-train_index, ]

# Summary table for clarity
library(knitr)

split_table <- data.frame(
Dataset = c("Training Set", "Testing Set"),
Rows    = c(nrow(train_data), nrow(test_data))
)

kable(split_table,
caption = "Training and Testing Dataset Sizes (80/20 Split)")
```

Table 7: Training and Testing Dataset Sizes (80/20 Split)

Dataset	Rows
Training Set	10236
Testing Set	2559

Interpretation of the Train–Test Split

The 80/20 split results in 10,236 observations in the training set and 2,559 observations in the testing set. The training dataset is large enough to support reliable model estimation and tuning, while the testing dataset provides a sufficiently sized hold-out sample for unbiased performance evaluation. All subsequent steps, including skewness analysis, transformations, and feature engineering, will be carried out exclusively on the training data to prevent information leakage. The testing data will only be used at the end of the analysis when evaluating the predictive accuracy of the final models.

Skewness Analysis of Training Data

Before applying transformations, I evaluate the distribution of each numeric predictor in the training dataset. Skewed variables may affect model performance, particularly in linear and generalized linear models. By examining skewness, I identify which variables may require log transformation or other adjustments. All calculations are performed on the training set to avoid information leakage.

```r
library(psych)
library(dplyr)
library(knitr)

# Select numeric variables from training data,
# dropping ID and missing-value flags

train_numeric <- train_data %>%
select(where(is.numeric)) %>%
select(-INDEX) %>%
select(-contains("_Missing"))

# Get full describe object
desc_train <- describe(train_numeric)

# Build skewness table: variable names from rownames, skew from column
skew_table <- data.frame(
Variable = rownames(desc_train),
Skewness = round(desc_train$skew, 3),
row.names = NULL
)

# Sort the skewness table from highest to lowest
skew_table <- skew_table %>%
arrange(desc(Skewness))

# Display the sorted table
kable(
skew_table,
caption = "Skewness of Numeric Variables in the Training Data (Sorted by Highest to Lowest)"
)
```

Table 8: Skewness of Numeric Variables in the Training Data (Sorted by Highest to Lowest)

Variable	Skewness
FreeSulfurDioxide	2.436
ResidualSugar	2.387
Chlorides	2.377
CitricAcid	2.269
Sulphates	2.241
VolatileAcidity	2.167
TotalSulfurDioxide	2.115
AcidIndex	1.663
FixedAcidity	1.470
STARS	0.561
Alcohol	0.293
pHOfWine	0.040
LabelAppeal	0.011
Density	-0.012
TARGET	-0.321

Interpretation of Skewness Results

A review of skewness values in the training dataset highlights several important distributional characteristics that guide the need for transformation:

- **TARGET (−0.321)** exhibits mild left skewness, which is typical for bounded count variables. No transformation is required.
- **FixedAcidity (1.470), VolatileAcidity (2.167), CitricAcid (2.269), ResidualSugar (2.387), Chlorides (2.377), FreeSulfurDioxide (2.436), TotalSulfurDioxide (2.115), Sulphates (2.241), and AcidIndex (1.663)** all display substantial right skewness. These long right tails suggest the presence of extreme values, making these variables appropriate candidates for log transformation to stabilize variance and reduce the influence of outliers.
- **Alcohol (0.293)** shows only mild positive skewness. While a log transformation may offer marginal benefits, it is not essential.
- **LabelAppeal (0.011), pHOfWine (0.040), and Density (−0.012)** are effectively symmetric around their means, indicating no need for transformation.
- **STARS (0.561)** reflects a modest degree of right skewness but remains sufficiently well-behaved to retain its original scale.

Overall, variables with skewness values above 1.0 exhibit the most pronounced departures from symmetry and are clear candidates for log transformation, whereas nearly symmetric variables can remain unchanged for subsequent modeling.

Log Transformations for Highly Skewed Variables

The skewness analysis revealed that several chemistry variables are right-skewed with extreme values. To reduce the impact of these extremes and make the variables more suitable for regression and count models, I apply a log(x + 1) transformation to the most skewed predictors. These include FixedAcidity, VolatileAcidity, CitricAcid, ResidualSugar, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, Sulphates, and AcidIndex. The transformation is based on the training data and applied consistently to both the training and testing sets to avoid information leakage.

```r
library(dplyr)

# Variables with strong right skewness identified from the training data
```

```r
skewed_vars <- c(
"FixedAcidity", "VolatileAcidity", "CitricAcid",
"ResidualSugar", "Chlorides", "FreeSulfurDioxide",
"TotalSulfurDioxide", "Sulphates", "AcidIndex"
)

# 1. Create log(x + 1) transformed versions in TRAIN and TEST
for (v in skewed_vars) {
new_name <- paste0("log_", v)

# Transform in training set
train_data[[new_name]] <- log(train_data[[v]] + 1)

# Apply the same transformation in test set
test_data[[new_name]] <- log(test_data[[v]] + 1)
}

# 2. Remove the original (untransformed) skewed variables
train_data  <-  train_data  %>%  select(-all_of(skewed_vars))
test_data   <-  test_data  %>%  select(-all_of(skewed_vars))
```

Interpretation of Log Transformation Step

For each highly skewed chemistry variable, a log(x + 1) transformation was applied and the original version was removed from both the training and testing datasets. As a result, the modeling data now contains only the log-transformed versions of FixedAcidity, VolatileAcidity, CitricAcid, ResidualSugar, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, Sulphates, and AcidIndex. This reduces the impact of extreme observations and avoids redundancy between raw and transformed measures, while still preserving the underlying information contained in these predictors.

Feature Engineering

To enrich the information available to the models and capture more nuanced relationships, I construct several new variables from the existing predictors. These engineered features are designed to reflect meaningful aspects of wine quality and consumer perception:

1. **SweetnessIntensity**
   - Formula: `log_ResidualSugar / Density`
   - Rationale: This ratio measures the intensity of sweetness relative to wine density. Higher values correspond to wines that are sweeter for a given thickness.
2. **Alcohol_AcidBalance**
   - Formula: `Alcohol / (log_AcidIndex + 1)`
   - Rationale: This feature captures the balance between alcohol strength and overall acidity. The `+ 1` in the denominator keeps the scale well-behaved and prevents division by zero.
3. **QualityAppeal**
   - Formula: `STARS * LabelAppeal`
   - Rationale: This interaction term combines expert ratings with marketing appeal, capturing wines that perform well both on quality and label design.

These features are created first in the training dataset, and then the same formulas are applied to the testing dataset to maintain consistency without introducing information leakage.

```r
library(dplyr)

# 1. Create engineered features in TRAINING data
train_data <- train_data %>%
mutate(
# Sweetness relative to wine density
SweetnessIntensity     = log_ResidualSugar / Density,

# Balance between alcohol strength and overall acidity
Alcohol_AcidBalance  = Alcohol / (log_AcidIndex + 1),

# Interaction between expert rating and label appeal
QualityAppeal          = STARS * LabelAppeal
)

# 2. Apply the SAME formulas to TEST data
test_data <- test_data %>%
mutate(
SweetnessIntensity     = log_ResidualSugar / Density,
Alcohol_AcidBalance  = Alcohol / (log_AcidIndex + 1),
QualityAppeal          = STARS * LabelAppeal
)
```

Final Set of Variables in the Training Dataset

After cleaning, imputing missing values, applying log transformations, and creating engineered features, the training dataset now contains the final set of variables that will be used for model building. The table below lists all variable names in the modeling dataset.

```r
# Load knitr for table formatting
library(knitr)

# Extract all variable names from the final training dataset
final_vars <- data.frame(
Variable = names(train_data),  # each column name becomes a row in this table
row.names = NULL                # remove row names for a cleaner display
)

# Display the variable names in a neat table
kable(
final_vars,
caption = "Final List of Variables in the Training Dataset"
)
```

Table 9: Final List of Variables in the Training Dataset

Variable
INDEX
TARGET
Density
pHOfWine
Alcohol
LabelAppeal
STARS
FixedAcidity_Missing
VolatileAcidity_Missing

Variable
CitricAcid_Missing
ResidualSugar_Missing
Chlorides_Missing
FreeSulfurDioxide_Missing
TotalSulfurDioxide_Missing
Sulphates_Missing
Alcohol_Missing
pHOfWine_Missing
STARS_Missing
log_FixedAcidity
log_VolatileAcidity
log_CitricAcid
log_ResidualSugar
log_Chlorides
log_FreeSulfurDioxide
log_TotalSulfurDioxide
log_Sulphates
log_AcidIndex
SweetnessIntensity
Alcohol_AcidBalance
QualityAppeal

The table above summarizes the complete set of predictors that will be used for model development. These vari- ables include the original retained features, missing-value indicators, log-transformed chemistry variables, and the newly engineered interaction and ratio features. This finalized dataset provides a comprehensive and well-prepared foundation for fitting the Poisson, negative binomial, and linear regression models.

Final Descriptive Statistics for Numeric Variables

Before model building, I summarize the distributional characteristics of the final numeric predictors. The table below shows key statistics for all numeric variables in the final training data, including the cleaned, transformed, and engineered features.

```r
library(dplyr)
library(psych)
library(knitr)

# Select only numeric variables from the final training data
train_numeric <- train_data %>%
select(where(is.numeric))

# Compute descriptive statistics using psych::describe
final_desc <- describe(train_numeric)[, c("mean", "sd", "median", "min", "max", "skew")]

# Round values for cleaner display
final_desc_round <- round(final_desc, 3)

# Display in a formatted table
kable(
final_desc_round,
caption = "Final Descriptive Statistics for Numeric Variables in the Training Dataset"
)
```

Table 10: Final Descriptive Statistics for Numeric Variables in the Training Dataset

	mean	sd	median	min	max	skew
INDEX	8057.623	4651.237	8088.000	1.000	16129.000	0.002
TARGET	3.026	1.927	3.000	0.000	8.000	-0.321
Density	0.994	0.027	0.995	0.889	1.099	-0.012
pHOfWine	3.211	0.665	3.200	0.480	6.050	0.040
Alcohol	10.603	3.422	10.400	0.000	26.100	0.293
LabelAppeal	-0.011	0.895	0.000	-2.000	2.000	0.011
STARS	2.033	0.777	2.000	1.000	4.000	0.561
FixedAcidity_Missing	0.126	0.332	0.000	0.000	1.000	2.252
VolatileAcidity_Missing	0.221	0.415	0.000	0.000	1.000	1.345
CitricAcid_Missing	0.232	0.422	0.000	0.000	1.000	1.269
ResidualSugar_Missing	0.295	0.456	0.000	0.000	1.000	0.899
Chlorides_Missing	0.300	0.458	0.000	0.000	1.000	0.871
FreeSulfurDioxide_Missing	0.289	0.453	0.000	0.000	1.000	0.932
TotalSulfurDioxide_Missing	0.246	0.430	0.000	0.000	1.000	1.182
Sulphates_Missing	0.280	0.449	0.000	0.000	1.000	0.982
Alcohol_Missing	0.060	0.237	0.000	0.000	1.000	3.713
pHOfWine_Missing	0.031	0.174	0.000	0.000	1.000	5.377
STARS_Missing	0.263	0.440	0.000	0.000	1.000	1.079
log_FixedAcidity	2.134	0.510	2.104	0.000	3.567	-0.953
log_VolatileAcidity	0.407	0.267	0.307	0.000	1.543	1.457
log_CitricAcid	0.419	0.275	0.329	0.000	1.581	1.489
log_ResidualSugar	2.339	0.978	2.272	0.000	4.957	0.359
log_Chlorides	0.128	0.148	0.061	0.000	0.855	1.985
log_FreeSulfurDioxide	3.903	0.925	3.784	0.000	6.435	0.212
log_TotalSulfurDioxide	5.005	0.767	5.017	0.000	6.964	-0.976
log_Sulphates	0.540	0.267	0.454	0.000	1.656	1.433
log_AcidIndex	2.162	0.141	2.197	1.609	2.890	0.886
SweetnessIntensity	2.353	0.986	2.285	0.000	5.176	0.372
Alcohol_AcidBalance	3.361	1.101	3.284	0.000	8.622	0.316
QualityAppeal	0.176	1.972	0.000	-6.000	8.000	0.686

Interpretation of Final Descriptive Statistics

The final descriptive statistics reflect the transformed and engineered feature set that will be used for model development. Several important patterns emerge from the summary:

- **TARGET** has a mean of 3.03 and ranges from 0 to 8, confirming that customer purchases are low-volume and tightly constrained.
- **Original chemistry variables** (e.g., Density, pHOfWine, Alcohol) show reasonable ranges and low skew after cleaning. Alcohol averages 10.60%, with moderate skew (0.29).
- **Missing-value indicator variables** all have means between 0.03 and 0.30, indicating that 3–30% of observations were imputed for those fields—consistent with the earlier missingness analysis.
- **Log-transformed chemistry variables** now exhibit much improved symmetry:
  - log_FixedAcidity mean 2.13, skew –0.95
  - log_VolatileAcidity mean 0.41, skew 1.46
  - log_TotalSulfurDioxide mean 5.01, skew –0.97
    These transformations help stabilize variance and improve linearity for regression models.
- **SweetnessIntensity** has a mean of 2.35 (median 2.29) with mild skew (0.37), indicating a well-behaved sweetness-to-density ratio suitable for modeling.

- **Alcohol_AcidBalance** averages 3.36 with limited skew (0.32), suggesting that the alcohol-to-acidity relationship remains stable across wines.
- **QualityAppeal**, the interaction of expert rating and label appeal, ranges from –6 to 8 with a mean of 0.18 and moderate skew (0.69). This confirms that the feature captures both positive and negative combinations of perceived quality and label attractiveness.

Overall, the final dataset shows stable distributions, reduced skewness for transformed variables, and interpretable engineered features, providing a solid foundation for fitting Poisson, Negative Binomial, and Linear Regression models in the next section.

Part 3: Building Predictive Models

In this section, we will build a series of models to understand the key factors influencing wine sales. The goal is to develop models using different regression techniques to explore the relationships between the predictors and the TARGET variable.

Key Steps Include:

1. **Poisson Regression Models**: To capture the effect of consumer perception variables and wine attributes on sales.
2. **Negative Binomial Regression Models**: To address overdispersion in the data and improve model fit.
3. **Multiple Linear Regression Models**: To analyze linear relationships between the predictors and wine sales.

By constructing and evaluating these models, we will gain insights into how various factors such as expert ratings, label appeal, sweetness, acidity, and preservation contribute to consumer purchasing behavior. This analysis will guide us toward the most reliable model for predicting wine sales.

Poisson Regression Model 1: Quality and Label-Based Predictors

For the first Poisson model, I intentionally select the three variables that best represent consumer-facing signals of wine quality and attractiveness.

Why These Variables Were Selected These were consistently the strongest predictors in the earlier correlation analysis and are well-supported by domain intuition:

- **STARS**: Expert ratings provide an objective assessment of wine quality. This variable had the strongest positive correlation with TARGET and reflects how professional evaluations influence purchasing behavior.
- **LabelAppeal**: The attractiveness of the wine label plays a major role in consumer decision-making, especially for casual buyers. This variable captures the marketing and visual appeal that often drives impulse purchases.
- **QualityAppeal (STARS × LabelAppeal)**: This interaction term represents the combined effect of quality and visual appeal. Wines that are both highly rated and visually appealing may enjoy a multiplicative impact on sales, which cannot be captured by the two variables individually.

These predictors form a focused, interpretable baseline model centered on consumer perception, explaining demand based on quality signals and design aesthetics before incorporating chemical or engineered features.

Model Specification

For Poisson regression with a log link, the model is specified as:

$$\log(\square[\text{TARGET}]) = \square_0 + \square_1(\text{STARS}) + \square_2(\text{LabelAppeal}) + \square_3(\text{QualityAppeal})$$

Where:

- □[TARGET] is the expected number of cases purchased (or wine sales).
- log(□[TARGET]) is the natural logarithm of the expected sales, representing the linear combination of predictors.
- $\square_0$ is the intercept, representing the log of the expected sales when all predictors are zero.
- $\square_1$ is the coefficient for **STARS**, indicating the change in the log of expected sales for each additional expert rating.
- $\square_2$ is the coefficient for **LabelAppeal**, reflecting the change in the log of expected sales for each increase in label attractiveness.
- $\square_3$ is the coefficient for **QualityAppeal**, the interaction term representing the combined effect of **STARS** and **LabelAppeal**.

```r
## Poisson Regression Model 1 (Quality & Label Effects)

# Fit a simple Poisson model using the most influential
# perception-based predictors: STARS, LabelAppeal, and
# their interaction term QualityAppeal.

poisson_model1 <- glm(
  TARGET ~ STARS + LabelAppeal + QualityAppeal,      # model formula
  data   = train_data,                               # training dataset only
  family = poisson(link = "log")                     # Poisson with log link
)

# Display model summary output
summary(poisson_model1)
```

```
##
## Call:
## glm(formula = TARGET ~ STARS + LabelAppeal + QualityAppeal, family = poisson(link = "log"),
##       data = train_data)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     0.586663   0.017294  33.923  < 2e-16 ***
## STARS           0.232900   0.007768  29.982  < 2e-16 ***
## LabelAppeal     0.107382   0.018045   5.951 2.67e-09 ***
## QualityAppeal 0.036292 0.007697   4.715 2.41e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##       Null deviance: 18306  on 10235  degrees of freedom ##
Residual deviance: 15547   on 10232   degrees of freedom ##
AIC: 41094
##
## Number of Fisher Scoring iterations: 5
```

Discussion of Poisson Model 1 Coefficients

The coefficients from the first Poisson regression model provide clear evidence that expert ratings and label design play important roles in predicting wine purchases. All three predictors, **STARS**, **LabelAppeal**, and **QualityAppeal**, are highly statistically significant ($p < 0.001$), indicating strong confidence that these effects are not due to random chance. Moreover, all coefficients are **positive**, which matches both intuition and expectations from earlier exploratory analysis.

STARS (Coefficient = 0.2329, p < 2e−16) The sign and magnitude of the STARS coefficient make intuitive sense. A higher expert rating is a direct signal of wine quality, and consumers tend to prefer higher-rated products. The coefficient implies:

$$\square^{0.2329} - 1 \approx 26\% \text{ increase in expected sales}$$

for each additional star, holding other variables constant. This is a large and meaningful effect, consistent with STARS having the strongest correlation with TARGET in the earlier analysis. Its extremely small p-value confirms strong statistical significance.

LabelAppeal (Coefficient = 0.1074, p = 2.67e−09) The coefficient for LabelAppeal is also positive and statistically significant. Wines with more appealing or attractive labels are expected to sell more, which aligns with consumer behavior in retail settings. The effect size indicates:

$$\square^{0.1074} - 1 \approx 11\% \text{ increase in expected sales}$$

for each one-unit increase in label appeal. While smaller than the effect of STARS, this is still substantial and reinforces the importance of marketing and presentation.

QualityAppeal (Interaction Term) (Coefficient = 0.0363, p = 2.41e−06) The positive and significant coefficient on **QualityAppeal** indicates that the combination of high expert ratings and high label appeal produces an additional multiplicative effect on sales. Although the coefficient is smaller, its interpretation is important:

$$\square^{0.0363} - 1 \approx 3.7\% \text{ additional increase}$$

for simultaneous improvements in both quality and label design. This suggests that consumers respond especially favorably when a wine is both highly rated and visually appealing, beyond what would be expected from each factor individually.

Model Fit Statistics

- **AIC:** 41,094
- **Null Deviance:** 18,306 on 10,235 degrees of freedom
- **Residual Deviance:** 15,547 on 10,232 degrees of freedom

The drop in deviance from **18,306** (null model) to **15,547** indicates that these three predictors explain a meaningful portion of the variation in wine purchases. Although the residual deviance remains higher than the degrees of freedom, which is a common occurrence in real-world count data, it still demonstrates that the model captures substantial behavioral patterns.

Overall Assessment

The signs, magnitudes, and statistical significance of the coefficients align with expectations. Expert ratings have the greatest impact on expected sales, label appeal also contributes positively, and the interaction term shows that quality and visual appeal reinforce each other. These results confirm that consumer perception variables alone offer a strong baseline for predicting wine demand.

Poisson Regression Model 2: Chemistry-Based Predictors

This second Poisson model shifts focus from consumer perception variables to chemical characteristics of the wine. The log-transformed chemistry variables were chosen because they showed substantial skewness in their raw form, and the log transformation stabilizes variance and improves linearity for Poisson modeling. Two engineered features:**Alcohol_AcidBalance** and **SweetnessIntensity** are also included to capture more complex taste and sen- sory relationships.

Why These Variables Were Selected

- **log_ResidualSugar**: Captures sweetness on a stabilized log scale; sweeter wines may appeal more to casual consumers.
- **log_FreeSulfurDioxide** and **log_TotalSulfurDioxide**: Represent preservative levels; these may affect freshness and perceived quality.
- **log_AcidIndex**: Reflects overall acidity; acidity influences balance and flavor structure.
- **Alcohol_AcidBalance**: An engineered feature that captures the smoothness or sharpness of a wine based on alcohol and acidity.
- **SweetnessIntensity**: Measures sweetness in relation to density; higher values may indicate fuller-bodied or more palatable wines.

These predictors reflect the **chemical properties** that influence taste, structure, and drinkability, factors that can drive consumer purchasing even without explicit marketing cues.

Model Specification

The Poisson regression model with a log link for predicting **TARGET** (the number of wine cases purchased) is specified as:

$$\begin{aligned}\log(\square[\text{TARGET}]) = \square_0 &+ \square_1 \log(\text{ResidualSugar}) + \square_2 \log(\text{FreeSulfurDioxide}) \\ &+ \square_3 \log(\text{TotalSulfurDioxide}) + \square_4 \log(\text{AcidIndex}) + \square_5\text{Alcohol\_AcidBalance} \\ &+ \square_6\text{SweetnessIntensity}\end{aligned}$$

Where:

- $\log(\square[\text{TARGET}])$ is the logarithm of the expected number of cases purchased (wine sales).
- $\square_0$ is the intercept, representing the baseline expected sales when all predictors are zero.
- $\square_1$, $\square_2$, $\square_3$, $\square_4$ are the coefficients for the log-transformed predictors: **ResidualSugar**, **FreeSulfurDioxide**, **TotalSulfurDioxide**, and **AcidIndex**, capturing the effect of sweetness, preservation, and acidity on expected sales.
- $\square_5$ is the coefficient for **Alcohol_AcidBalance**, an engineered feature representing the balance between alcohol and acidity.
- $\square_6$ is the coefficient for **SweetnessIntensity**, an engineered feature capturing the ratio of sweetness relative to the density of the wine.

```r
## Poisson Regression Model 2 (Chemistry-Based Model)
poisson_model2 <- glm(
  TARGET ~ log_ResidualSugar + log_FreeSulfurDioxide +
           log_TotalSulfurDioxide + log_AcidIndex +
           Alcohol_AcidBalance + SweetnessIntensity,
  data = train_data,
  family = poisson(link = "log")
)

# Display model summary
summary(poisson_model2)
```

```
##
## Call:
## glm(formula = TARGET ~ log_ResidualSugar + log_FreeSulfurDioxide +
##     log_TotalSulfurDioxide + log_AcidIndex + Alcohol_AcidBalance +
##     SweetnessIntensity, family = poisson(link = "log"), data = train_data)
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)             2.594038   0.114022  22.750  < 2e-16 ***
## log_ResidualSugar      -0.168112   0.082758  -2.031   0.0422 *
## log_FreeSulfurDioxide   0.058560   0.006221   9.414  < 2e-16 ***
## log_TotalSulfurDioxide  0.072030   0.007820   9.211  < 2e-16 ***
## log_AcidIndex          -1.026992   0.044020 -23.330  < 2e-16 ***
## Alcohol_AcidBalance     0.028526   0.005187   5.499 3.81e-08 ***
## SweetnessIntensity      0.180917   0.081955   2.208   0.0273 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 18306  on 10235  degrees of freedom
## Residual deviance: 17343  on 10229  degrees of freedom
## AIC: 42896
##
## Number of Fisher Scoring iterations: 5
```

Interpretation of Poisson Model 2 (Chemistry + Engineered Features)

Poisson Model 2 incorporates chemical predictors and engineered taste–balance features to understand how wine composition influences sales. All six predictors in the model are statistically significant at the 5% level or better, and the signs of the coefficients align well with expectations from enology and consumer behavior.

log_ResidualSugar (Coefficient = −0.1681, p = 0.0422) The coefficient is negative and statistically significant.
This indicates that, after controlling for sweetness intensity and other variables, **higher residual sugar alone is associated with slightly lower expected sales**.

This makes sense in the context of this model because: - SweetnessIntensity already captures *perceived sweetness relative to body*.
- Once this perception measure is accounted for, excess raw sugar may correlate with lower-quality or cheaper wines, reducing appeal.

Effect size:

$\square^{-0.1681} - 1 \approx -15.5\%$ lower expected sales per log-unit increase

log_FreeSulfurDioxide (Coefficient = 0.0586, p < 2e−16) This coefficient is positive and highly significant.
Moderate free SO helps preserve freshness and prevents oxidation, which can improve drinkability and shelf life.

Effect size:

$\square^{0.0586} - 1 \approx 6.0\%$ increase in expected sales

log_TotalSulfurDioxide (Coefficient = 0.0720, p < 2e−16) Total SO also shows a positive relationship with sales.
The positive sign suggests that wines with well-managed sulfur levels (ensuring stability and longevity) may be more appealing or more commonly stocked by retailers.

Effect size:

$$\square^{0.0720} - 1 \approx 7.5\% \text{ increase in expected sales}$$

log_AcidIndex (Coefficient = −1.0269, p < 2e−16) This is the strongest effect in the model, large, negative, and very significant.
A higher AcidIndex corresponds to harsher, more acidic wines — which aligns with consumer preference for smoother profiles.

Effect size:

$$\square^{-1.0269} - 1 \approx -64.0\% \text{ decrease in expected sales}$$

This confirms acidity is a major factor influencing customer acceptance.

Alcohol_AcidBalance (Coefficient = 0.0285, p = 3.81e−08) This engineered feature balances alcohol level with acidity. The positive coefficient indicates that **wines with smoother alcohol–acid structure sell more**.

Effect size:

$$\square^{0.0285} - 1 \approx 2.9\% \text{ increase in expected sales}$$

This is consistent with smoother wines appealing more to mass-market consumers.

SweetnessIntensity (Coefficient = 0.1809, p = 0.0273) This variable captures *perceived sweetness relative to body*.
The positive sign is expected: sweeter, fuller wines tend to be more popular among broad consumer groups.

Effect size:

$$\square^{0.1809} - 1 \approx 19.8\% \text{ increase in expected sales}$$

The statistical significance supports sweetness as an important driver of retail performance.

Model Fit Assessment

- **Null deviance:** 18,306
- **Residual deviance:** 17,343
- **AIC:** 42,896

The reduction from 18,306 → 17,343 indicates that the chemistry-based predictors explain some meaningful variation in sales. However, the improvement is modest relative to Model 1, suggesting that **chemical attributes alone are less powerful predictors of consumer purchasing behavior** than expert ratings and label appeal.

Overall Assessment

The signs and magnitudes of the coefficients are logical and aligned with expectations: - **Acidity strongly reduces sales** - **Sweetness, sulfur stability, and palate balance increase sales** - **Raw sugar alone has a negative adjusted effect** once perceived sweetness is accounted for

Overall, Model 2 provides valuable insight into how chemical properties of wine influence purchasing behavior, complementing the consumer-facing attributes emphasized in Model 1.

Note:
The two Poisson models highlight important behavioral patterns. Model 1 emphasizes perception attributes, while

Model 2 shows how chemistry influences taste and sales. However, in both cases the residual deviance is larger than the degrees of freedom, providing clear evidence of overdispersion. This means the variance in TARGET is greater than what the Poisson distribution assumes. To address this limitation and improve model fit, the next section will estimate Negative Binomial models, which include a dispersion parameter that allows the variance to exceed the mean.

Negative Binomial Regression Model 1: Balanced Chemistry and Sensory Predictors

This Negative Binomial model shifts focus to a stable and interpretable set of chemistry-based predictors that influence taste, acidity balance, and overall drinkability. Unlike the Poisson model, the Negative Binomial specification accounts for overdispersion in TARGET by allowing the variance to exceed the mean. This model includes log-transformed acidity and sulfur measures along with Alcohol and Density, two key structural attributes that affect mouthfeel and consumer acceptance.

Why These Variables Were Selected

- **log_FreeSulfurDioxide**: Represents preservative strength and freshness stability; moderate SO helps maintain wine quality.
- **log_TotalSulfurDioxide**: Captures total preservative levels; higher values often indicate better shelf stability.
- **log_AcidIndex**: Reflects overall acidity; acidity is one of the strongest drivers of palate structure.
- **Alcohol**: Influences warmth, body, and perceived sweetness; widely used by consumers as a quality cue.
- **Density**: Represents dissolved solids and extract content; denser wines are often richer and fuller-bodied.

These predictors capture **fundamental chemical attributes** that shape flavor, body, and stability, factors that can meaningfully influence purchasing behavior in retail environments.

Model Specification

The Negative Binomial model with a log link is specified as:

$$\log(\square[\text{TARGET}]) = \square_0 + \square_1 \log(\text{FreeSulfurDioxide}) + \square_2 \log(\text{TotalSulfurDioxide}) \\ + \square_3 \log(\text{AcidIndex}) + \square_4\text{Alcohol} + \square_5\text{Density}$$

Where:

- $\log(\square[\text{TARGET}])$ represents the expected number of cases purchased (TARGET), modeled as the log of the expected sales.
- $\square_0$ is the intercept, representing the baseline sales when all predictors are zero.
- $\square_1$ to $\square_5$ are the coefficients for each predictor variable:
  - log(FreeSulfurDioxide): Log-transformed sulfur dioxide levels used to preserve wine freshness.
  - log(TotalSulfurDioxide): Log-transformed total sulfur dioxide, reflecting the overall preservation level.
  - log(AcidIndex): Log-transformed acidity of the wine, influencing flavor balance and consumer preference.
  - Alcohol: Alcohol content, affecting the body and perceived richness of the wine.
  - Density: Density of the wine, which indicates the concentration of dissolved solids, typically corresponding to body and mouthfeel.
- The Negative Binomial model includes a dispersion parameter ($\square$) that allows the variance to exceed the mean, addressing overdispersion in the data.

```r
# Negative Binomial Regression Model 1 (Chemistry & Structure)
suppressMessages(library(MASS))    # Suppress package loading messages

# Fit the Negative Binomial model
nb_model1 <- glm.nb(
TARGET ~ log_FreeSulfurDioxide + log_TotalSulfurDioxide +
log_AcidIndex + Alcohol + Density,
data = train_data
)

# Display model summary
summary(nb_model1)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ log_FreeSulfurDioxide + log_TotalSulfurDioxide +
##     log_AcidIndex + Alcohol + Density, data = train_data, init.theta = 11.4232272,
##     link = log)
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              3.245262   0.266065  12.197  < 2e-16 ***
## log_FreeSulfurDioxide    0.060375   0.007013   8.609  < 2e-16 ***
## log_TotalSulfurDioxide   0.075409   0.008749   8.619  < 2e-16 ***
## log_AcidIndex           -1.087891   0.048560 -22.403  < 2e-16 ***
## Alcohol                  0.008861   0.001869   4.741 2.12e-06 ***
## Density                 -0.512276   0.241275  -2.123   0.0337 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(11.4232) family taken to be 1)
##
##     Null deviance: 15620  on 10235  degrees of freedom ##
Residual deviance: 14838  on 10230  degrees of freedom ##
AIC: 42731
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  11.42
##           Std. Err.:  1.01
##
##  2 x log-likelihood:  -42717.14
```

Interpretation of Negative Binomial Model 1 (Chemistry & Structural Predictors)

This Negative Binomial model evaluates how chemical and structural wine attributes influence purchasing behavior while accounting for overdispersion in TARGET. All five predictors: log_FreeSulfurDioxide, log_TotalSulfurDioxide, log_AcidIndex, Alcohol, and Density are statistically significant, and the signs of the coefficients align with expectations from wine chemistry and consumer taste preferences. The estimated dispersion parameter □ ≈ 11.42 indicates the presence of moderate overdispersion in the data. A theta value in this range suggests that the variance of TARGET is meaningfully greater than its mean, validating the need for a modeling approach that incorporates extra variability.

log_FreeSulfurDioxide (Coefficient = 0.0604, $p < 2e{-}16$) The positive and highly significant coefficient indicates that wines with higher levels of free SO tend to have higher expected sales. Free sulfur dioxide preserves freshness and protects against oxidation, making the wine more stable during storage and distribution. Effect size:

$$□^{0.0604} - 1 \approx 6.2\% \text{ increase in expected sales}$$

log_TotalSulfurDioxide (Coefficient = 0.0754, p < 2e−16) Total SO also has a positive and significant effect. This measure captures the wine's overall preservative environment, which contributes to product stability and quality maintenance.
Effect size:

$$\square^{0.0754} - 1 \approx 7.8\% \text{ increase in expected sales}$$

The consistency of both sulfur variables highlights the importance of chemical stability as a factor influencing commercial performance.

log_AcidIndex (Coefficient = −1.0879, p < 2e−16) This is the strongest effect in the model: large, negative, and highly significant. A higher AcidIndex reflects harsher acidity, which many consumers perceive as unbalanced or unpleasant.
Effect size:

$$\square^{-1.0879} - 1 \approx -66.3\% \text{ decrease in expected sales}$$

This result confirms that excessive acidity is a major deterrent to consumer acceptance.

Alcohol (Coefficient = 0.0089, p = 2.12e−06) Alcohol content shows a small but statistically significant positive association. Alcohol contributes to body, warmth, and perceived richness, characteristics that many consumers associate with higher quality.
Effect size:

$$\square^{0.0089} - 1 \approx 0.9\% \text{ increase in expected sales}$$

Density (Coefficient = −0.5123, p = 0.0337) Density reflects dissolved solids and extract content. The negative coefficient suggests that denser wines, when acidity and alcohol are controlled for, may be heavier or less appealing to mainstream buyers.
Effect size:

$$\square^{-0.5123} - 1 \approx -40.1\% \text{ decrease in expected sales}$$

Model Fit Assessment

- **Null deviance:** 15,620
- **Residual deviance:** 14,838
- **AIC:** 42,731

These metrics show that the included chemistry and structure predictors explain a meaningful amount of variation in TARGET, and the Negative Binomial framework captures this variation more effectively due to the presence of overdispersion.

Overall Assessment

The model produces three key insights:
1. **Chemical stability (sulfur levels) has a positive and consistent impact on sales**, indicating that consumers and retailers favor wines that maintain quality over time.
2. **Harsh acidity significantly reduces demand**, making AcidIndex a critical negative predictor.
3. **Structural attributes such as alcohol and density influence perceived balance and drinkability**, with smoother, less dense wines performing better commercially.

Overall, this Negative Binomial model provides a clear and statistically robust understanding of how wine chemistry shapes consumer purchasing behavior.

Negative Binomial Regression Model 2: Enhanced Chemistry-Based Predictors

This second Negative Binomial model focuses on a broader set of chemistry-based predictors that reflect key aspects of wine composition and flavor structure. These variables were selected because they showed consistent significance across earlier analyses and represent core chemical attributes that influence taste, stability, and consumer preference. By using the Negative Binomial specification, this model accounts for the moderate overdispersion present in TARGET while isolating the effects of sweetness, acidity, preservation, and alcohol structure.

Why These Variables Were Selected

- **log_ResidualSugar**: Captures sweetness after log transformation; sweetness affects drinkability and mass-market appeal.
- **log_AcidIndex**: Reflects overall acidity; high acidity can lead to harsher taste profiles.
- **Alcohol**: Influences body, warmth, and perceived richness; often used as a quality cue.
- **log_FreeSulfurDioxide**: Represents preservative strength; helps maintain freshness.
- **SweetnessIntensity**: Measures perceptible sweetness relative to density, capturing flavor balance.

These predictors jointly describe **sensory sweetness, acidity balance, alcohol structure, and chemical stability**, providing insights into how intrinsic wine properties shape consumer behavior.

Model Specification

The Negative Binomial model with a log link is specified as:

$$\begin{aligned}\log(\square[\text{TARGET}]) = \square_0 + \square_1 \log(\text{ResidualSugar}) + \square_2 \log(\text{AcidIndex}) \\ + \square_3 \text{Alcohol} + \square_4 \log(\text{FreeSulfurDioxide}) + \square_5 \text{SweetnessIntensity}\end{aligned}$$

Where:

- log(ResidualSugar) represents the logarithmic transformation of **ResidualSugar**, capturing its effect on sales after controlling for other factors.
- log(AcidIndex) represents the logarithmic transformation of **AcidIndex**, which reflects the acidity of the wine.
- Alcohol is the **alcohol content** in the wine, which affects its body, warmth, and perceived richness.
- log(FreeSulfurDioxide) represents the logarithmic transformation of **FreeSulfurDioxide**, a preservative used to maintain wine freshness.
- SweetnessIntensity is a variable capturing **sweetness** relative to the wine's density, indicating the intensity of perceived sweetness.
- The Negative Binomial model includes a dispersion parameter ($\square$) that allows the variance to exceed the mean, addressing overdispersion in the data.

This model uses a combination of transformed and raw predictors to assess how various chemistry and sensory attributes influence the number of expected wine purchases (TARGET).

```r
## Negative Binomial Regression Model 2 (Enhanced Chemistry Model)

library(MASS)

nb_model2 <- glm.nb(
  TARGET ~ log_ResidualSugar + log_AcidIndex + Alcohol +
           log_FreeSulfurDioxide + SweetnessIntensity,
  data = train_data
)

summary(nb_model2)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ log_ResidualSugar + log_AcidIndex +
##     Alcohol + log_FreeSulfurDioxide + SweetnessIntensity, data = train_data,
##     init.theta = 10.92922516, link = log)
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)            3.135907   0.113984  27.512  < 2e-16 ***
## log_ResidualSugar     -0.162328   0.093818  -1.730   0.0836 .
## log_AcidIndex         -1.121189   0.048506 -23.115  < 2e-16 ***
## Alcohol                0.008357   0.001878   4.450 8.58e-06 ***
##  log_FreeSulfurDioxide 0.064128   0.007010   9.148  < 2e-16 ***
## SweetnessIntensity     0.178986   0.092933   1.926   0.0541 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(10.9292) family taken to be 1)
##
##     Null deviance: 15522  on 10235  degrees of freedom ##
Residual deviance: 14814  on 10230  degrees of freedom ##
AIC: 42798
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  10.929
##           Std. Err.:  0.938
##
##  2 x log-likelihood:  -42783.991
```

Interpretation of Negative Binomial Model 2 (Enhanced Chemistry Predictors)

Negative Binomial Model 2 includes predictors like log_ResidualSugar, log_AcidIndex, Alcohol, log_FreeSulfurDioxide, and SweetnessIntensity to capture key wine attributes. Most predictors are significant, with signs aligning with enological principles. The dispersion parameter ($□ \approx 10.93$) indicates moderate overdispersion, justifying the use of the Negative Binomial model.

log_ResidualSugar (Coefficient = −0.1623, p = 0.0836) The coefficient is negative and marginally significant. After controlling for SweetnessIntensity and other chemical attributes, higher raw residual sugar is associated with lower expected sales.
Effect size:

$$□^{-0.1623} - 1 \approx -15.0\% \text{ decrease in expected sales}$$

This suggests that, once perceived sweetness is accounted for, excess residual sugar may signal lower wine quality or reduced consumer appeal.

log_AcidIndex (Coefficient = −1.1212, p < 2e−16) This is the strongest effect in the model—large, negative, and highly significant. A higher AcidIndex reflects harsher acidity, which many consumers perceive as sharp or unbalanced.
Effect size:

$$□^{-1.1212} - 1 \approx -67.3\% \text{ decrease in expected sales}$$

This confirms that excessive acidity is a major deterrent to purchase interest.

Alcohol (Coefficient = 0.00836, p = 8.58e−06) Alcohol has a small but highly significant positive effect. Higher alcohol contributes to body, warmth, and perceived richness—qualities often associated with fuller, more premium wines.
Effect size:

$$\square^{0.00836} - 1 \approx 0.8\% \text{ increase in expected sales}$$

While small in magnitude, this effect is consistent with consumer preference for wines that feel balanced and structured.

log_FreeSulfurDioxide (Coefficient = 0.0641, p < 2e−16) The positive and highly significant coefficient indicates that wines with greater free SO levels tend to sell more. Free sulfur dioxide helps protect against oxidation and microbial spoilage, increasing shelf stability.
Effect size:

$$\square^{0.0641} - 1 \approx 6.6\% \text{ increase in expected sales}$$

This reinforces the importance of chemical freshness and preservation.

SweetnessIntensity (Coefficient = 0.1789, p = 0.0541) SweetnessIntensity is marginally significant at the 10% level. The positive sign suggests that wines with more perceptible sweetness relative to body may appeal to broader consumer segments.
Effect size:

$$\square^{0.1789} - 1 \approx 19.6\% \text{ increase in expected sales}$$

Although less precisely estimated, this result aligns with patterns observed in Poisson Model 2.

Model Fit Assessment

- **Null deviance:** 15,522
- **Residual deviance:** 14,814
- **AIC:** 42,798

These values indicate that the model explains a meaningful proportion of variation in TARGET while accommodating overdispersion via the dispersion parameter.

Overall Assessment

Three key insights emerge from this model:
1. **Acidity remains the strongest chemical driver of demand**, with harsher acidity dramatically reducing sales.
2. **Stability-related variables (free sulfur dioxide) have consistent positive effects**, supporting the role of preservation in consumer acceptance.
3. **Sweetness and alcohol structure influence perceived balance**, with smoother, fuller-tasting wines experiencing higher demand.

Overall, Negative Binomial Model 2 provides a nuanced view of how sweetness, acidity, alcohol, and chemical freshness interact to shape wine purchasing behavior.

Multiple Linear Regression Model 1: Perception and Chemistry Predictors

This first multiple linear regression (MLR) model combines key perception-based variables with core chemistry attributes to explain variation in TARGET. In line with the assignment requirements, both **STARS** and **LabelAppeal** are explicitly included so that their effects can be interpreted in a linear framework. Two additional chemistry variables, **Alcohol** and **log_AcidIndex**, are added to capture structural and acidity-related aspects of wine quality.

Why These Variables Were Selected

- **STARS**: Represents expert ratings and is one of the strongest predictors of sales across all models.
- **LabelAppeal**: Captures visual and marketing appeal at the shelf, directly influencing consumer choice. This variable is required for interpretation in the linear models.
- **Alcohol**: Influences body, warmth, and perceived richness; often used by consumers as an informal signal of quality.
- **log_AcidIndex**: Reflects overall acidity on a transformed scale; acidity is a major driver of taste balance and consumer acceptance.

Together, these predictors allow us to study how expert opinion and label design interact with underlying chemistry to drive wine demand.

Model Specification

$$\text{TARGET} = \square_0 + \square_1(\text{STARS}) + \square_2(\text{LabelAppeal}) + \square_3(\text{Alcohol}) + \square_4 \log(\text{AcidIndex}) + \square$$

Where:

- $\square_0$ is the **intercept** of the model, representing the baseline number of expected sales when all predictors are zero.
- $\square_1$ is the coefficient for **STARS**, representing the change in expected sales for each additional star in the expert rating.
- $\square_2$ is the coefficient for **LabelAppeal**, reflecting the change in expected sales for each unit increase in label appeal.
- $\square_3$ is the coefficient for **Alcohol**, indicating the change in expected sales for each unit increase in alcohol content.
- $\square_4$ is the coefficient for **log(AcidIndex)**, reflecting the change in expected sales for a one-unit increase in the log-transformed AcidIndex, which represents overall acidity.
- $\square$ is the **error term**, capturing the variation in sales that cannot be explained by the model.

```r
## Multiple Linear Regression Model 1 (Perception + Chemistry)

mlr_model1 <- lm(
  TARGET ~ STARS + LabelAppeal + Alcohol + log_AcidIndex,
  data = train_data
)

summary(mlr_model1)
```

```
##
## Call:
## lm(formula = TARGET ~ STARS + LabelAppeal + Alcohol + log_AcidIndex,
##     data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9044 -0.7345  0.3674  1.1221  4.2533
##
## Coefficients:
```

```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.925445   0.262748   30.16  < 2e-16 ***
## STARS             0.767307   0.021852   35.11  < 2e-16 ***
## LabelAppeal       0.589410   0.018907   31.17  < 2e-16 ***
## Alcohol           0.014763   0.004747    3.11  0.00188 **
## log_AcidIndex -3.056945  0.115187  -26.54  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.639 on 10231 degrees of freedom
## Multiple R-squared:   0.2766,  Adjusted R-squared:   0.2763
## F-statistic:     978 on 4 and 10231 DF,  p-value: < 2.2e-16
```

Interpretation of Multiple Linear Regression Model 1 (Perception + Chemistry Predictors)

This multiple linear regression model evaluates how perception-based quality indicators (STARS and LabelAppeal) and two chemistry-driven attributes (Alcohol and log_AcidIndex) explain variation in TARGET. All four predictors are statistically significant, and the signs of their coefficients align with intuition about consumer behavior and wine chemistry. Since this model uses a linear framework, each coefficient represents the expected change in TARGET (number of purchases) for a one-unit change in the predictor, holding all other variables constant.

STARS (Coefficient = 0.7673, p < 2e−16) The coefficient is positive and highly significant, indicating that higher expert ratings lead to higher expected sales.
Interpretation:
For each additional star in the expert rating, predicted sales increase by approximately **0.77 units**, holding other factors constant.
This reinforces the importance of expert reviews as a strong external quality signal that shapes consumer preferences.

LabelAppeal (Coefficient = 0.5894, p < 2e−16) LabelAppeal also has a positive and highly significant effect.
Interpretation:
A one-unit increase in label appeal is associated with an increase of about **0.59 units** in expected sales.
This result highlights the power of visual design and packaging in influencing consumer selection, especially in retail environments where labels act as first impressions.

Alcohol (Coefficient = 0.0148, p = 0.00188) Alcohol has a small but statistically significant positive effect on sales.
Interpretation:
Each one-unit increase in alcohol percentage increases predicted sales by roughly **0.015 units**.
Although modest in magnitude, this suggests that consumers may associate slightly higher alcohol content with fuller body or overall wine richness.

log_AcidIndex (Coefficient = −3.0569, p < 2e−16) This coefficient is large, negative, and highly significant.
Interpretation:
A one-unit increase in log_AcidIndex decreases predicted sales by about **3.06 units**, holding other factors constant.
This strong effect reflects consumer aversion to overly acidic wines, which are often perceived as harsh or unbalanced.

Model Fit Assessment

- **Residual standard error:** 1.639
- **Multiple R-squared:** 0.2766

- **Adjusted R-squared:** 0.2763
- **F-statistic:** 978 ($p < 2.2e{-}16$)

The model explains about **27.6%** of the variation in TARGET, which is reasonable given the complexity and subjectivity of consumer wine purchasing behavior. The significant F-statistic indicates that the model provides a meaningful improvement over a null model that contains no predictors.

Overall Assessment

The results reveal three key insights: 1. **Perception-based factors (STARS and LabelAppeal) are strong and reliable predictors of sales**, reinforcing the influence of expert opinions and marketing design. 2. **Acidity has a major negative impact on demand**, consistent with consumer preference for balanced, smooth wines. 3. **Alcohol contributes positively, though modestly**, reflecting its role in shaping the sensory profile of wines.

Overall, this MLR model provides a clear and interpretable assessment of how external quality signals and core chemical attributes jointly influence wine purchasing behavior.

Multiple Linear Regression Model 2: Perception + Sweetness and Stability Predictors

This second multiple linear regression (MLR) model continues to center the two required perception variables, **STARS** and **LabelAppeal**, while introducing a different set of chemistry-driven predictors that capture sweet- ness and chemical stability. Specifically, log_ResidualSugar and log_TotalSulfurDioxide are included to represent the roles of sweetness profile and preservative strength. This model allows us to evaluate how consumer-facing quality cues interact with core chemical attributes to explain variation in TARGET.

Why These Variables Were Selected

- **STARS**: Measures expert evaluation of wine quality and consistently emerges as one of the strongest predictors of sales.
- **LabelAppeal**: Captures visual attractiveness and marketing design, directly influencing consumer selection.
- **log_ResidualSugar**: Represents underlying sweetness levels; sweetness plays a key role in shaping mass-market appeal.
- **log_TotalSulfurDioxide**: Captures total preservative strength, contributing to freshness, stability, and shelf life.

This model uses a distinct set of chemistry variables from Model 1, enabling comparison of how different taste and stability factors interact with perception cues.

Model Specification

$$\text{TARGET} = \square_0 + \square_1(\text{STARS}) + \square_2(\text{LabelAppeal}) + \square_3 \log(\text{ResidualSugar}) + \square_4 \log(\text{TotalSulfurDioxide}) + \square$$

Where:

- $\square_0$ is the **intercept** of the model, representing the baseline number of expected sales when all predictors are zero.
- $\square_1$ is the coefficient for **STARS**, representing the change in expected sales for each additional star in the expert rating.
- $\square_2$ is the coefficient for **LabelAppeal**, reflecting the change in expected sales for each unit increase in label appeal.
- $\square_3$ is the coefficient for **log(ResidualSugar)**, indicating the change in expected sales for a one-unit increase in the log-transformed residual sugar, which represents sweetness.
- $\square_4$ is the coefficient for **log(TotalSulfurDioxide)**, reflecting the change in expected sales for a one-unit increase in the log-transformed total sulfur dioxide, which represents preservative strength.

- □ is the **error term**, capturing the variation in sales that cannot be explained by the model.

```r
## Multiple Linear Regression Model 2 (Perception + Sweetness & Stability)

mlr_model2 <- lm(
  TARGET ~ STARS + LabelAppeal + log_ResidualSugar + log_TotalSulfurDioxide,
  data = train_data
)

summary(mlr_model2)
```

```
##
## Call:
## lm(formula = TARGET ~ STARS + LabelAppeal + log_ResidualSugar +
##     log_TotalSulfurDioxide, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6072 -0.7777  0.4246  1.1607  4.1513
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -0.09993    0.12247  -0.816   0.4145
## STARS                    0.81156    0.02232  36.365   <2e-16 ***
## LabelAppeal              0.56833    0.01937  29.335   <2e-16 ***
## log_ResidualSugar        0.03287    0.01704   1.928   0.0538 .
## log_TotalSulfurDioxide   0.28080    0.02172  12.929   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.682 on 10231 degrees of freedom
## Multiple R-squared:  0.2387, Adjusted R-squared:  0.2384
## F-statistic: 801.9 on 4 and 10231 DF,  p-value: < 2.2e-16
```

Interpretation of Multiple Linear Regression Model 2 (Perception + Sweetness & Stability Predictors)

This second multiple linear regression model examines how perception-based indicators (STARS and LabelAppeal) interact with underlying chemical attributes related to sweetness and preservative strength. All predictors are statistically significant at conventional levels, and the linear specification allows us to interpret each coefficient as the direct change in TARGET for a one-unit change in the predictor, holding all others constant.

STARS (Coefficient = 0.8116, p < 2e–16) The coefficient is positive and highly significant, indicating that expert ratings remain a major driver of wine purchases.
Interpretation:
Each additional star in expert rating increases expected sales by approximately **0.81 units**, reinforcing the central role of perceived quality in consumer decision-making.

LabelAppeal (Coefficient = 0.5683, p < 2e–16) LabelAppeal again shows a strong positive and highly significant effect.
Interpretation:
A one-unit increase in label attractiveness is associated with about **0.57 additional units** of expected sales.
This confirms the influence of branding and visual design in shaping consumer selection at the point of purchase.

log_ResidualSugar (Coefficient = 0.0329, p = 0.0538) log_ResidualSugar is positive and marginally significant at the 5–10% level.
Interpretation:
A one-unit increase in log residual sugar increases predicted sales by approximately **0.033 units**.
This suggests that, after controlling for LabelAppeal and expert ratings, slightly sweeter wines may appeal modestly more to consumers, although the effect is relatively weak.

log_TotalSulfurDioxide (Coefficient = 0.2808, p < 2e−16) Total SO has a positive, highly significant coefficient.
Interpretation:
A one-unit increase in log total sulfur dioxide increases predicted sales by about **0.28 units**.
Because sulfur dioxide improves chemical stability and shelf life, this result suggests that fresher, more stable wines tend to perform better commercially.

Model Fit Assessment

- **Residual standard error:** 1.682
- **Multiple R-squared:** 0.2387
- **Adjusted R-squared:** 0.2384
- **F-statistic:** 801.9 (p < 2.2e–16)

The model explains about **23.9%** of variation in TARGET, slightly lower than Model 1, indicating that sweetness and stability contribute meaningfully but less strongly than acidity and alcohol did in the prior specification.

Overall Assessment

Multiple Linear Regression Model 2 provides several key insights:

1. **STARS and LabelAppeal remain dominant predictors of sales**, consistent with the importance of expert evaluations and visual marketing cues.
2. **Sweetness contributes modestly to demand**, aligning with consumer preference for approachable, slightly sweeter profiles.
3. **Chemical stability has a strong positive effect**, with higher sulfur dioxide levels associated with improved retail performance.

Overall, this model highlights how sweetness and preservation interact with perception variables to shape consumer purchasing behavior.

Part 4: Model Selection

In this section, I compare different regression models for predicting wine purchase counts. The goal is to select the best model for predicting **TARGET** (sales) based on performance metrics such as **AIC**, **Residual Deviance**, and **RMSE**. Additionally, I evaluate model performance using **confusion matrix** and **predictions** to understand the model's accuracy.

Key steps include:

1. **Model Comparison**: Evaluating **Poisson**, **Negative Binomial**, and **Multiple Linear Regression (MLR)** models using **AIC**, **Residual Deviance**, and **RMSE** to compare their fit and predictive accuracy.
2. **Prediction Evaluation**: Generating predicted values for the test dataset and comparing them to actual TARGET values. This helps assess the model's generalization ability.

3. **Confusion Matrix**: Comparing **actual vs predicted counts** using a confusion matrix to identify how well the model predicts different purchase counts and where it under- or over-predicts.
4. **Selected Model**: Based on the AIC, deviance, and confusion matrix results, **Poisson Model 1** was selected as the best model, showing a reasonable balance of fit and generalization.

The selected model provides useful insights into the relationship between wine attributes and sales, with some limitations in predicting extreme counts.

Selecting the Best Count Regression Model

To identify the most appropriate model for explaining and predicting wine purchase counts, I compare all candidate models built earlier: two Poisson regressions, two Negative Binomial regressions, and two multiple linear regression (MLR) models. Although MLR is not a count model, the assignment requires evaluating whether an ordinary linear model may still be preferable based on interpretability, parsimony, or predictive accuracy.

The code below extracts comparable performance metrics across model types.
- **AIC** is used for Poisson and Negative Binomial models because it measures goodness of fit while penalizing model complexity.
- **Residual deviance** helps evaluate lack of fit for count models.
- **Theta** values summarize the degree of overdispersion handled by the Negative Binomial models.
- For MLR models, I include **RMSE** and **Adjusted R-squared**, which are more appropriate performance indicators for linear regression.

These metrics together provide a balanced basis for selecting a final model.

```r
# Helper function for RMSE for linear models
rmse <- function(model) sqrt(mean(model$residuals^2))

# Build comparison table
model_summary <- data.frame(
  Model = c("Poisson Model 1",
            "Poisson Model 2",
            "NB Model 1",
            "NB Model 2",
            "MLR Model 1",
            "MLR Model 2"),
  Type  = c("Poisson", "Poisson",
            "NegBin", "NegBin",
            "Linear", "Linear"),
  AIC = c(AIC(poisson_model1),
          AIC(poisson_model2),
          AIC(nb_model1),
          AIC(nb_model2),
          NA, NA),
  Residual_Deviance = c(deviance(poisson_model1),
                        deviance(poisson_model2),
                        deviance(nb_model1),
                        deviance(nb_model2),
                        NA, NA),
  Theta = c(NA, NA,
            summary(nb_model1)$theta,
            summary(nb_model2)$theta,
            NA, NA),
  RMSE = c(NA, NA,
           NA, NA,
           rmse(mlr_model1),
           rmse(mlr_model2)),
  Adj_R2 = c(NA, NA,
             NA, NA,
                summary(mlr_model1)$adj.r.squared,
                summary(mlr_model2)$adj.r.squared)
)

# Print nicely
knitr::kable(
  model_summary,
  digits = 3,
  caption = "Comparison of Poisson, Negative Binomial, and Multiple Linear Regression Models"
)
```

Table 11: Comparison of Poisson, Negative Binomial, and Multiple Linear Regression Models

Model	Type	AIC	Residual_Deviance	Theta	RMSE	Adj_R2
Poisson Model 1	Poisson	41094.15	15546.51	NA	NA	NA
Poisson Model 2	Poisson	42896.49	17342.85	NA	NA	NA
NB Model 1	NegBin	42731.14	14838.41	11.423	NA	NA
NB Model 2	NegBin	42797.99	14813.65	10.929	NA	NA
MLR Model 1	Linear	NA	NA	NA	1.639	0.276
MLR Model 2	Linear	NA	NA	NA	1.681	0.238

Interpretation and Model Selection (Count Regression Models)

- **AIC comparison strongly favors Poisson Model 1.**
  Poisson Model 1 has an AIC of **41,094.15**, whereas NB Model 1 and NB Model 2 have AIC values of **42,731.14** and **42,797.99**, respectively. The ΔAIC relative to Poisson Model 1 is therefore about **+1,637** for NB Model 1 and **+1,704** for NB Model 2. Since differences greater than 10 already constitute strong evidence, these extremely large gaps provide *overwhelming* support for Poisson Model 1 as the best-fitting count model in terms of the likelihood–penalty trade-off.

- **Overdispersion is present but only mild.**
  For Poisson Model 1, the residual deviance is **15,546.51** with roughly **10,231** residual degrees of freedom, giving an overdispersion ratio of about

$$\text{Deviance/df} \approx 15{,}546.51/10{,}231 \approx 1.52.$$

  Ratios near 1 indicate no overdispersion; values between **1.5 and 2.0** are typically considered **mild**. This suggests that the Poisson variance assumption is not perfect, but the violation is not severe enough to outweigh Poisson Model 1's enormous AIC advantage.

- **Negative Binomial models improve deviance slightly but at a very high cost in AIC.**
  NB Model 1 and NB Model 2 reduce residual deviance to **14,838.41** and **14,813.65**, and their □ˆ estimates (11.42 and 10.93) confirm that they can flexibly model extra-Poisson variance. However, the relatively small improvement in deviance comes with a **huge increase in AIC** (over 1,600 points worse than Poisson Model 1), indicating that the additional complexity of the NB models is not justified by the incremental gain in fit.

- **Selected model and inference.**
  Given (i) **overwhelmingly lower AIC** for Poisson Model 1, (ii) only **mild overdispersion** (ratio 1.52), and (iii) clear, interpretable coefficients, **Poisson Model 1 is selected as the preferred count regression model**. Inference is made on the log-count scale: exponentiating its coefficients yields multiplicative effects on expected sales, and robust (quasi-Poisson or sandwich) standard errors can be used if extra caution about mild overdispersion is desired.

Comparison to Multivariate Regression Model

Poisson Model 1 outperforms **Multiple Linear Regression Model 1** in handling the count nature of the data. While the multivariate regression model explains **27.6%** of the variance in **TARGET**, Poisson Model 1, with its significantly lower **AIC** and better model fit, is better suited for predicting count outcomes.

The Poisson model more effectively captures the multiplicative effects of **STARS**, **LabelAppeal**, and their interaction term, offering a more accurate prediction of wine sales. In contrast, the linear model assumes continuous outcomes and can predict **negative values**, which are impossible for sales counts. Moreover, MLR assumes normally distributed errors, which is inappropriate for count data.

Despite these differences, both models agree that **STARS** and **LabelAppeal** are the strongest predictors of wine purchases, confirming the robustness of these findings across different modeling approaches.

Inference From the Selected Poisson Model (Using Model Coefficients)

Using Poisson Model 1, we can make direct, quantitative inferences about how each predictor affects expected sales. Because the model uses a log link, exponentiating each coefficient yields an incidence rate ratio (IRR), which represents the multiplicative effect on expected purchases.

- **STARS (Estimate = 0.2329; IRR 1.262)**
  A one-unit increase in expert rating leads to a **26.2% increase** in expected sales, making STARS the strongest predictor in the model.
- **LabelAppeal (Estimate = 0.1074; IRR 1.113)**
  A one-unit increase in label appeal increases expected sales by **11.3%**, highlighting the importance of packaging and marketing.
- **QualityAppeal (Estimate = 0.0363; IRR 1.037)**
  A one-unit increase in perceived quality appeal results in a **3.7% increase** in expected sales, contributing incremental value.
- **Intercept (Estimate = 0.5867; IRR 1.798)**
  With all predictors at zero, the expected sales level is **exp(0.5867) 1.80 units**, setting the baseline sales level.
- **Model fit and inference validity**
  The residual deviance is **15,547 on 10,232 df**, indicating mild overdispersion (ratio 1.52). All coefficients are statistically significant ($p < 0.001$), ensuring reliable inference.

Together, these results show that expert ratings, label attractiveness, and perceived quality each meaningfully increase expected purchase counts, with STARS and LabelAppeal showing the strongest effects.

Actual vs. Predicted Counts on the Test Data

To evaluate how well the selected Poisson count regression model performs on new observations, I apply it to the test data set and compare the predicted purchase counts with the observed TARGET values. Because the Poisson model predicts expected counts, the fitted values need not be integers but should generally track the scale and pattern of the actual outcomes. The table below reports the first ten test observations, showing TARGET alongside the model's Predicted_TARGET values in a clean format without index columns.

```r
# Use the selected Poisson count regression model
poisson_best <- poisson_model1

# Generate predicted counts for the test data
test_predictions <- predict(
  poisson_best,
  newdata = test_data,
  type    = "response"    # expected counts
)
```

```r
# Build a comparison table: actual vs predicted
pred_table <- data.frame(
  TARGET           = test_data$TARGET,
  Predicted_TARGET = test_predictions
)

# Keep only the first 10 observations
pred_table <- head(pred_table, 10)

# Remove row names to avoid an extra index column
row.names(pred_table) <- NULL

# Print nicely formatted LaTeX/HTML table
knitr::kable(
  pred_table,
  digits  = 3,
  caption = "Actual vs. Predicted TARGET for Test Data (First 10 Observations)"
)
```

Table 12: Actual vs. Predicted TARGET for Test Data (First 10 Observations)

TARGET	Predicted_TARGET
3	1.966
4	4.489
5	3.430
2	1.966
6	5.876
4	2.865
5	2.865
4	2.270
0	1.966
4	2.865

Interpretation of Actual vs. Predicted TARGET Values (Test Data)

- The predicted values from the Poisson model align reasonably well with the scale and direction of the actual TARGET counts, indicating that the model captures the general purchasing intensity of wines in the test set.
- The model tends to **underpredict larger counts** (e.g., actual values of 5 or 6 are predicted closer to 2–3). This is a common behavior in Poisson regression when the outcome distribution has a heavier right tail than the model assumes.
- For low and moderate counts (0–4), the predictions fall within a realistic range and track actual behavior fairly well. This suggests that the model performs best for the most common purchase volumes, which constitute the majority of observations.
- Because Poisson predictions represent **expected counts**, they need not be integers. The table shows smooth, continuous estimates (e.g., 1.966, 2.865), which correspond to the model's estimated mean count for each wine profile.
- Overall, the test results demonstrate that the selected Poisson model generalizes reasonably well, capturing primary patterns in the data while showing mild bias for higher-count outcomes, a behavior consistent with the mild overdispersion detected earlier.

Evaluating the Performance of the Count Regression Model (Training Data)

To evaluate how well the selected Poisson count regression model fits the **training dataset**, we generate predicted counts and compare them to the actual TARGET values. Because Poisson models output *expected counts* (continuous values), the predictions must be **rounded** to the nearest whole number before constructing a classification-style performance table.

This section reports: - The rounded predictions from the Poisson Model 1
- A confusion-matrix–style cross-tabulation showing how often the model predicts each count correctly
- Practical insight into model performance across categories

This provides a clear diagnostic view of where the model is accurate and where systematic under- or over-prediction may occur.

```r
# Confusion matrix on TRAINING data
# 1. Predicted expected counts on training data
train_pred <- predict(poisson_model1, type = "response")

# 2. Round to nearest integer for classification-style comparison
train_pred_round <- round(train_pred)

# 3. Confusion matrix: Actual vs Predicted
cm_train <- table(
Actual    = train_data$TARGET,
Predicted = train_pred_round
)

# 4. Convert to data frame for kable
cm_df <- as.data.frame.matrix(cm_train)

# Add Actual counts as a proper column instead of row names
cm_df <- cbind(Actual = rownames(cm_df), cm_df)
row.names(cm_df) <- NULL

# 5. Print nicely formatted confusion matrix
knitr::kable(
cm_df,
caption = "Confusion-Matrix Style Table: Actual vs. Predicted TARGET (Training Data)"
)
```

Table 13: Confusion-Matrix Style Table: Actual vs. Predicted TARGET (Training Data)

Actual	2	3	4	5	6	8
0	831	1291	67	0	0	0
1	193	7	0	0	0	0
2	783	92	1	0	0	0
3	1237	707	135	0	0	0
4	586	1348	546	58	21	0
5	107	689	587	71	129	8
6	6	139	241	39	151	35
7	0	8	38	1	56	14
8	0	0	1	0	3	10

Interpretation of Confusion-Matrix Style Table (Training Data)

The confusion-matrix style summary compares the **actual purchase counts (TARGET)** to the **rounded Poisson Model 1 predictions** on the training dataset. Although Poisson regression is not a classification model, this table helps visualize how closely predicted counts align with actual outcomes.

Key observations from the matrix:

- **Strong clustering around 2–4 purchases:**
  The model most frequently predicts values of **2, 3, and 4**, which aligns with the dominant range of TARGET in the dataset. Large diagonal counts for these classes indicate that the model captures the central tendency of consumer purchasing behavior effectively.
- **Underprediction of higher counts (5, 6, 7, 8):**
  Rows for higher TARGET values (5–8) show predictions shifting left toward lower values. This is typical of Poisson models when **mild overdispersion** is present—extreme outcomes are pulled toward the mean.
- **Very low counts (0 and 1) are harder to predict:**
  For TARGET = 0 and 1, the model often predicts 2 or 3, suggesting it tends to **overpredict** small purchase values. This occurs because the global Poisson mean is higher than these small-count categories.
- **Overall error pattern is systematic and understandable**: Rather than random misclassification, pre- diction errors follow the expected behavior of a Poisson model: smooth shrinkage toward the mean, limited ability to capture rare, high-count outcomes, and strong performance in the modal purchase range (2–4).

Taken together, this evaluation shows that **Poisson Model 1 generalizes well to the core purchasing behavior** but, as expected, shows smoothing bias in the extremes. This supports the model's selection as the deployment candidate while acknowledging typical Poisson limitations in tail prediction.

Limitations of the Poisson Model

While **Poisson Model 1** provides valuable insights into the relationship between wine attributes and sales, it has a few limitations that should be considered. One of the primary concerns is mild **overdispersion**, where the variance in the observed data exceeds the mean, which is a common feature in count data. Although the overdispersion is not severe, it can still impact the model's fit, leading to less accurate predictions for high-count and low-count observations.

Furthermore, **Poisson regression assumes a specific distribution** for the errors, which may not always align with the actual distribution of the data. This assumption can lead to bias in the predicted values, particularly in the tails of the distribution. The model tends to **underpredict high counts** (5–8), as it struggles to capture the extreme end of the purchase distribution, while it **overpredicts very low counts** (0-1), often predicting a higher count than observed. Lastly, the model's **limited flexibility** in capturing complex relationships, such as interactions between predictors or non-linear effects, means that it might miss out on some underlying patterns in the data.

These limitations suggest that while the Poisson model is a useful starting point, there are opportunities for further refinement using alternative regression techniques or additional feature engineering.

Conclusion

In this assignment, I conducted a comprehensive analysis of the wine dataset, exploring key variables such as chemical properties, expert ratings, and label appeal, to predict wine purchase counts. The Poisson regression model, particularly Poisson Model 1, was identified as the best model based on its significantly lower AIC compared to the Negative Binomial and MLR models. The analysis revealed that expert ratings (STARS) and label appeal (LabelAppeal) are the strongest predictors of wine sales, while the model showed mild overdispersion. Despite some limitations, including underpredicting higher counts and overpredicting lower counts, the selected Poisson model provides valuable insights into the factors driving wine purchases. Future work could focus on refining the model by exploring alternative regression techniques and additional feature engineering to enhance predictive accuracy.