HW#5_Data621

Saloua Daouki

2024-12-01

Table of Contents

Introduction

This analysis focuses on exploring, analyzing, and modeling a dataset of approximately 12,000 commercially available wines. The dataset primarily includes variables related to the chemical properties of the wines being sold. The target variable represents the number of sample cases of wine purchased by wine distribution companies after sampling a wine. These sample cases are used by distribution companies to provide tasting samples to restaurants and wine stores across the United States. Wines with higher sample case purchases are more likely to be featured in high-end restaurants.

The goal of this assignment is to build **a count regression model** to predict the number of sample cases purchased based on the wine's characteristics. Accurate predictions would enable the wine manufacturer to adjust its wine offerings to maximize sales. Additionally, it's worth considering that missing values in the dataset may hold predictive value for the target variable. For this analysis, only the provided variables (or those derived from them) will be utilized.

Below is the exploratory analysis of the dataset:

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

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(corrplot)

## corrplot 0.95 loaded
```

```r
library(readr)
# Load necessary libraries
library(ggplot2)
library(dplyr)
library(reshape2)
```

1. Data Exploration:

```r
# Load datasets
training_wine_data <- read.csv('HW 5 attached/wine-training-data.csv')
evaluation_wine_data <- read.csv("HW 5 attached/wine-evaluation-data.csv")
```

The evaluation data has 3335 observations with 16 variables. The training data has 12795 observations with also 16 variables.

The message above indicates that the variable 'TARGET' is numerical in the training data where is logical in the evaluation data. This will create an error in later analysis if it is not addressed now; we are going to convert the 'TARGET' to numercial in the evaluation data.

```r
# convert TARGET to numerical in the evaluation data and ensure that it is
the same in training data
evaluation_wine_data$TARGET <- as.numeric(evaluation_wine_data$TARGET)
training_wine_data$TARGET <- as.numeric(training_wine_data$TARGET)
```

Next, let's look at the summary statistics and the structure of the training data:

```r
# Explore dataset
summary(training_wine_data)
```

```
##      INDEX           TARGET       FixedAcidity     VolatileAcidity
##  Min.   :    1   Min.   :0.000   Min.   :-18.100   Min.   :-2.7900
##  1st Qu.: 4038   1st Qu.:2.000   1st Qu.:  5.200   1st Qu.: 0.1300
##  Median : 8110   Median :3.000   Median :  6.900   Median : 0.2800
##  Mean   : 8070   Mean   :3.029   Mean   :  7.076   Mean   : 0.3241
##  3rd Qu.:12106   3rd Qu.:4.000   3rd Qu.:  9.500   3rd Qu.: 0.6400
##  Max.   :16129   Max.   :8.000   Max.   : 34.400   Max.   : 3.6800
##
##    CitricAcid      ResidualSugar        Chlorides       FreeSulfurDioxide
##  Min.   :-3.2400   Min.   :-127.800   Min.   :-1.1710   Min.   :-555.00
##  1st Qu.: 0.0300   1st Qu.:  -2.000   1st Qu.:-0.0310   1st Qu.:   0.00
##  Median : 0.3100   Median :   3.900   Median : 0.0460   Median :  30.00
##  Mean   : 0.3084   Mean   :   5.419   Mean   : 0.0548   Mean   :  30.85
##  3rd Qu.: 0.5800   3rd Qu.:  15.900   3rd Qu.: 0.1530   3rd Qu.:  70.00
##  Max.   : 3.8600   Max.   : 141.150   Max.   : 1.3510   Max.   : 623.00
##                    NA's   :616        NA's   :638       NA's   :647
##  TotalSulfurDioxide    Density             pH          Sulphates
##  Min.   :-823.0     Min.   :0.8881   Min.   :0.480   Min.   :-3.1300
##  1st Qu.:  27.0     1st Qu.:0.9877   1st Qu.:2.960   1st Qu.: 0.2800
##  Median : 123.0     Median :0.9945   Median :3.200   Median : 0.5000
##  Mean   : 120.7     Mean   :0.9942   Mean   :3.208   Mean   : 0.5271
##  3rd Qu.: 208.0     3rd Qu.:1.0005   3rd Qu.:3.470   3rd Qu.: 0.8600
```

```
##  Max.   :1057.0     Max.   :1.0992   Max.   :6.130   Max.   : 4.2400
##  NA's   :682                         NA's   :395     NA's   :1210
##     Alcohol        LabelAppeal          AcidIndex          STARS
##  Min.   :-4.70   Min.   :-2.000000   Min.   : 4.000   Min.   :1.000
##  1st Qu.: 9.00   1st Qu.:-1.000000   1st Qu.: 7.000   1st Qu.:1.000
##  Median :10.40   Median : 0.000000   Median : 8.000   Median :2.000
##  Mean   :10.49   Mean   :-0.009066   Mean   : 7.773   Mean   :2.042
##  3rd Qu.:12.40   3rd Qu.: 1.000000   3rd Qu.: 8.000   3rd Qu.:3.000
##  Max.   :26.50   Max.   : 2.000000   Max.   :17.000   Max.   :4.000
##  NA's   :653                                          NA's   :3359
```

```
str(training_wine_data)
```

```
## 'data.frame':    12795 obs. of  16 variables:
##  $ INDEX             : int  1 2 4 5 6 7 8 11 12 13 ...
##  $ TARGET            : num  3 3 5 3 4 0 0 4 3 6 ...
##  $ FixedAcidity      : num  3.2 4.5 7.1 5.7 8 11.3 7.7 6.5 14.8 5.5 ...
##  $ VolatileAcidity   : num  1.16 0.16 2.64 0.385 0.33 0.32 0.29 -1.22 0.27
-0.22 ...
##  $ CitricAcid        : num  -0.98 -0.81 -0.88 0.04 -1.26 0.59 -0.4 0.34
1.05 0.39 ...
##  $ ResidualSugar     : num  54.2 26.1 14.8 18.8 9.4 ...
##  $ Chlorides         : num  -0.567 -0.425 0.037 -0.425 NA 0.556 0.06 0.04
-0.007 -0.277 ...
##  $ FreeSulfurDioxide : num  NA 15 214 22 -167 -37 287 523 -213 62 ...
##  $ TotalSulfurDioxide: num  268 -327 142 115 108 15 156 551 NA 180 ...
##  $ Density           : num  0.993 1.028 0.995 0.996 0.995 ...
##  $ pH                : num  3.33 3.38 3.12 2.24 3.12 3.2 3.49 3.2 4.93
3.09 ...
##  $ Sulphates         : num  -0.59 0.7 0.48 1.83 1.77 1.29 1.21 NA 0.26
0.75 ...
##  $ Alcohol           : num  9.9 NA 22 6.2 13.7 15.4 10.3 11.6 15 12.6 ...
##  $ LabelAppeal       : int  0 -1 -1 -1 0 0 0 1 0 0 ...
##  $ AcidIndex         : int  8 7 8 6 9 11 8 7 6 8 ...
##  $ STARS             : int  2 3 3 1 2 NA NA 3 NA 4 ...
```

Several variables (such as ResidualSugar, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, pH, Sulphates, Alcohol, and STARS) in the data contain NA values which could potentially be a concern when we perform model fitting, as some models require imputation or removal of rows with missing data. This we will handle in data preparation section.

```
# Mean, Median, Standard Deviation
summary_stats <- training_wine_data %>%
  summarise(across(where(is.numeric), list(
    mean = ~ mean(.x, na.rm = TRUE),
    sd = ~ sd(.x, na.rm = TRUE),
    median = ~ median(.x, na.rm = TRUE)
  )))
summary_stats
```

```
##   INDEX_mean INDEX_sd INDEX_median TARGET_mean TARGET_sd TARGET_median
## 1    8069.98 4656.905         8110    3.029074  1.926368             3
##   FixedAcidity_mean FixedAcidity_sd FixedAcidity_median
VolatileAcidity_mean
## 1          7.075717        6.317643                6.9
0.3241039
##   VolatileAcidity_sd VolatileAcidity_median CitricAcid_mean CitricAcid_sd
## 1          0.7840142                   0.28       0.3084127     0.8620798
##   CitricAcid_median ResidualSugar_mean ResidualSugar_sd
ResidualSugar_median
## 1              0.31           5.418733         33.74938
3.9
##   Chlorides_mean Chlorides_sd Chlorides_median FreeSulfurDioxide_mean
## 1     0.05482249    0.3184673            0.046                30.84557
##   FreeSulfurDioxide_sd FreeSulfurDioxide_median TotalSulfurDioxide_mean
## 1             148.7146                       30                120.7142
##   TotalSulfurDioxide_sd TotalSulfurDioxide_median Density_mean Density_sd
## 1              231.9132                      123    0.9942027 0.02653765
##   Density_median  pH_mean     pH_sd pH_median Sulphates_mean Sulphates_sd
## 1        0.99449 3.207628 0.6796871       3.2      0.5271118    0.9321293
##   Sulphates_median Alcohol_mean Alcohol_sd Alcohol_median LabelAppeal_mean
## 1              0.5     10.48924   3.727819           10.4     -0.009066041
##   LabelAppeal_sd LabelAppeal_median AcidIndex_mean AcidIndex_sd
## 1      0.8910892                  0       7.772724     1.323926
##   AcidIndex_median STARS_mean STARS_sd STARS_median
## 1                8   2.041755  0.90254            2
```

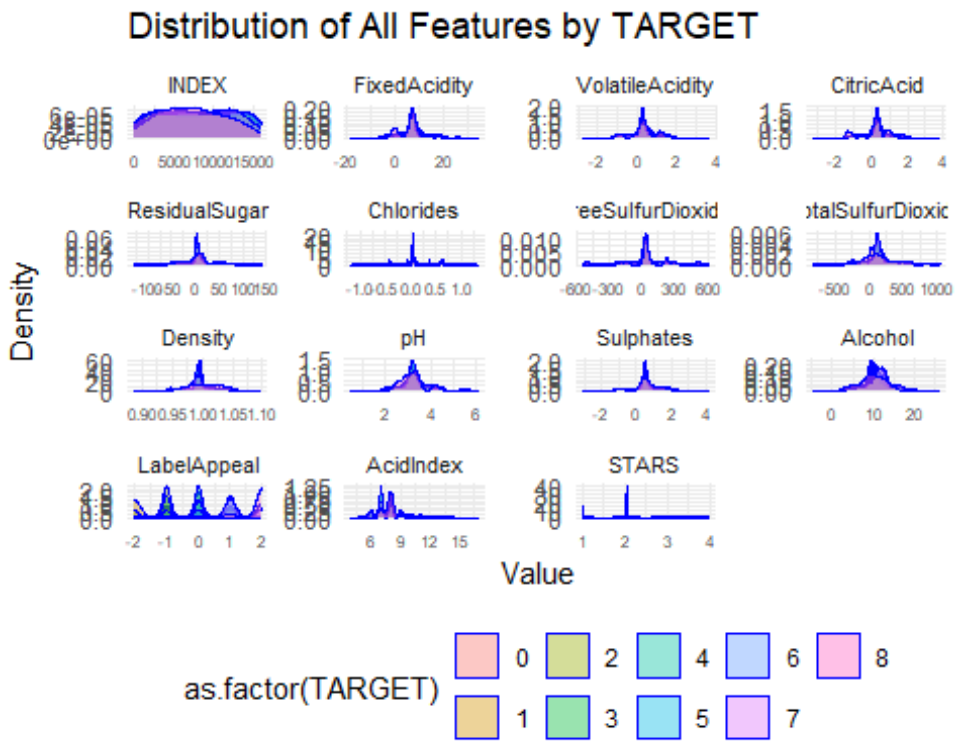
- The TARGET variable -Number of Cases Purchased - has mean and median that are close, indicating a roughly symmetric distribution. The standard deviation shows some variation around the mean.

- FixedAcidity has a large standard deviation compared to its mean and median, suggesting high variability in acidity values across the samples.

- before building any model, we want to explore how these variables relate to each other and how they influence TARGET. In addition, the large spread in FixedAcidity could indicate that transformations (e.g., log-transformation) may be helpful to stabilize variance.

```r
# Visualizations
# Boxplot for target variable
data <- training_wine_data
# Fill missing values with the mean
#data[is.na(data)] <- lapply(data, function(x) if (is.numeric(x)) mean(x,
na.rm = TRUE) else x)

# Replace NA values in numeric columns with their mean
training_wine_data <- data.frame(lapply(data, function(x) {
  if (is.numeric(x)) {
```

```r
    x[is.na(x)] <- mean(x, na.rm = TRUE)  # Replace NA with mean for numeric
columns
  }
  return(x)
}))




# Melt the dataset for faceting
data_melted <- melt(training_wine_data, id.vars = "TARGET")

# Create faceted density plot
ggplot(data_melted, aes(x = value, fill = as.factor(TARGET))) +
  geom_density(alpha = 0.4, color = "blue") +
  facet_wrap(~variable, scales = "free") +
  ggtitle("Distribution of All Features by TARGET") +
  xlab("Value") +
  ylab("Density") +
  theme_minimal() +
  theme(
    legend.position = "bottom",
    strip.text = element_text(size = 8),
    axis.text.x = element_text(size = 6)
  )
```



The graph is a **faceted density plot** that illustrates the distribution of all numerical features in the dataset relative to the `TARGET` variable. Each subplot corresponds to a specific feature, showing how its values are distributed across different levels of the `TARGET` variable, which is represented by distinct color-coded density curves.

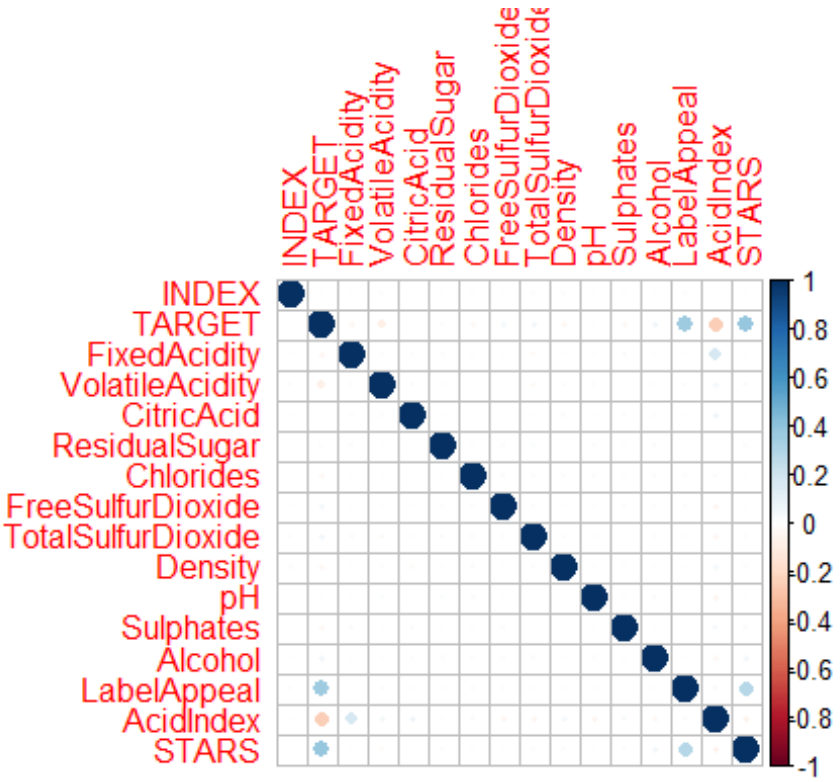
In the `Alcohol` subplot, there is a clear overlap of distributions across `TARGET` categories, though higher `TARGET` levels tend to correspond to higher alcohol content, suggesting a potential correlation. Features like `FixedAcidity`, `VolatileAcidity`, and `CitricAcid` exhibit distributions centered around specific ranges, with variations in density across `TARGET` levels, indicating they could help differentiate certain target categories.

The `ResidualSugar` and `Chlorides` subplots show irregular distributions, with distinct peaks for some `TARGET` levels, suggesting these features might influence the target variable. Similarly, `pH` distributions overlap significantly across categories, showing less variation and potentially lower predictive power. On the other hand, `LabelAppeal` and `STARS` appear to have strong, distinct patterns, as their distributions align clearly with specific `TARGET` values, highlighting them as important predictors.

Additionally, features like `Density`, `Sulphates`, and `FreeSulfurDioxide` exhibit narrow ranges and sharp peaks, with relatively uniform distributions across most `TARGET` levels, which may indicate limited impact on distinguishing categories. Conversely, `AcidIndex` displays noticeable clustering for certain `TARGET` levels, suggesting its potential as a strong predictor.

Finally, INDEX, likely an identifier, shows no meaningful variation with respect to the target variable and is unlikely to contribute to predictive modeling. Overall, the graph provides valuable insights into the relationship between each feature and the target variable, identifying features like Alcohol, LabelAppeal, STARS, and AcidIndex as potentially significant for predicting TARGET. This analysis can guide further feature selection and model development efforts.

```
# Correlation matrix
numeric_vars <- training_wine_data %>% select(where(is.numeric))
cor_matrix <- cor(numeric_vars, use = "complete.obs")
corrplot(cor_matrix, method = "circle")
```




```
# Check missing values
missing_values <- sapply(training_wine_data, function(x) sum(is.na(x)))
missing_values
```

```
##              INDEX             TARGET       FixedAcidity
VolatileAcidity
##                  0                  0                  0
0
##         CitricAcid      ResidualSugar          Chlorides
FreeSulfurDioxide
##                  0                  0                  0
0
## TotalSulfurDioxide            Density                 pH
Sulphates
```

```
##                   0                   0                   0
0
##             Alcohol        LabelAppeal          AcidIndex
STARS
##                   0                   0                   0
0
```

2. Data Preparation:

Feature selection

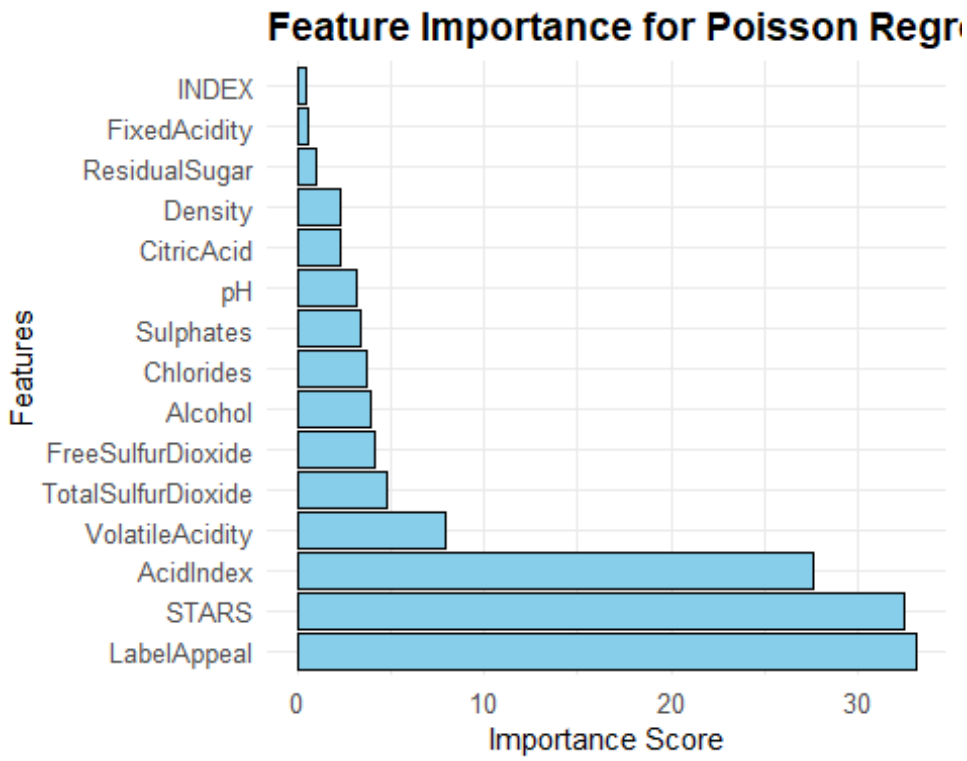
We are going to select features that are more important for the modeling development.

3. Build Models

Build the required models using glm, MASS::glm.nb, and standard linear regression.

```
##Feature selection
library(caret)

## Loading required package: lattice

# Poisson regression models
training_data <- training_wine_data
# Poisson regression models
poisson_model1 <- glm(TARGET ~ ., family = poisson(link = "log"), data =
training_data)
#poisson_model1 <- glm(TARGET ~ ., family = poisson(link = "log"), data =
training_wine_data)
importance_data <- varImp(poisson_model1,scale=FALSE)



# Assuming `importance_data` is your data frame
# Move feature names to rownames and keep the importance values
# Turn row names into a column named "Features"
importance_data$Features <- rownames(importance_data)
rownames(importance_data) <- NULL  # Remove row names
importance_data$Importance <- importance_data$Overall
importance_data$Overall <- NULL
# Reorder columns so "Features" is the first column
importance_data <- importance_data[, c("Features", "Importance")]

# Print the updated data frame
print(importance_data)

##              Features Importance
## 1               INDEX  0.4160199
## 2       FixedAcidity  0.5414538
## 3    VolatileAcidity  7.8536107
```

```
## 4          CitricAcid  2.2823275
## 5       ResidualSugar  0.9648410
## 6           Chlorides  3.6815398
## 7   FreeSulfurDioxide  4.0451049
## 8  TotalSulfurDioxide  4.7291125
## 9             Density  2.2762907
## 10                 pH  3.1574502
## 11          Sulphates  3.3120021
## 12            Alcohol  3.9198310
## 13        LabelAppeal 33.1821201
## 14          AcidIndex 27.6170237
## 15               STARS 32.5372568

#rownames(importance_data) <- rownames(importance_data)  # Assign the feature
names as rownames
#colnames(importance_data) <- "Importance"  # Rename the column headers with




# Plot the bar graph
ggplot(importance_data, aes(x = reorder(Features, -Importance), y =
Importance)) +
  geom_bar(stat = "identity", fill = "skyblue", color = "black") +
  coord_flip() +  # Flip the axes for better readability
  labs(
    title = "Feature Importance for Poisson Regression",
    x = "Features",
    y = "Importance Score"
  ) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    plot.title = element_text(size = 14, face = "bold")
  )
```



From the **feature importance graph** generated using `varImp()`, it is clear that some variables contribute significantly more to the model's predictive performance than others. The top three features—**LabelAppeal**, **STARS**, and **AcidIndex**—stand out with importance scores of approximately 33, 32, and 28, respectively. These features are the most influential predictors and should be prioritized for inclusion in the model.

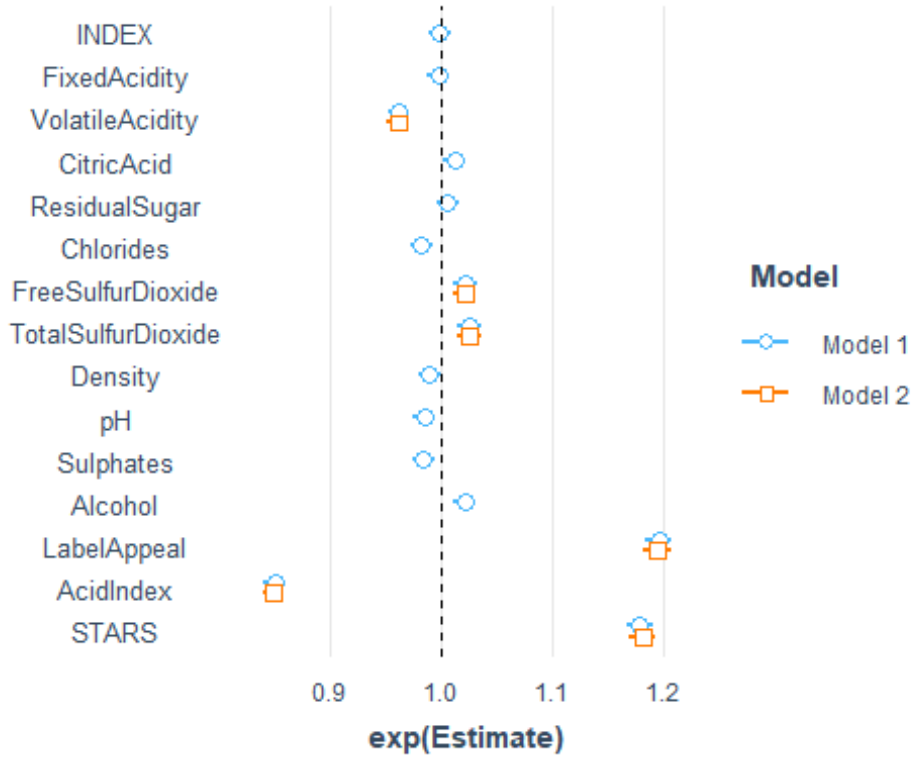
In addition to the top three features, **VolatileAcidity**, **TotalSulfurDioxide**, and **FreeSulfurDioxide** demonstrate moderate importance, with scores ranging between 4 and 8. These variables can be included based on the model's performance or if domain knowledge suggests they hold significance.

On the other hand, features such as **INDEX**, **FixedAcidity**, **ResidualSugar**, and **Density** show very low importance scores, often below 1. These variables have minimal influence on the model's predictions and can be excluded to simplify the model, improve computational efficiency, and potentially reduce overfitting.

In summary, the model should prioritize the top-performing features (**LabelAppeal**, **STARS**, and **AcidIndex**) while selectively including moderate-importance variables like **VolatileAcidity** and **TotalSulfurDioxide**. Low-importance features can be safely removed unless domain-specific knowledge suggests otherwise. To validate this feature selection, the model should be retrained, and its performance metrics (e.g., RMSE, AIC, or accuracy) should be evaluated.

```
poisson_model2 <- glm(TARGET ~ LabelAppeal+
STARS+AcidIndex+VolatileAcidity+FreeSulfurDioxide+TotalSulfurDioxide , family
```

```
= poisson(link = "log"), data = training_data)
summary(poisson_model2)

##
## Call:
## glm(formula = TARGET ~ LabelAppeal + STARS + AcidIndex + VolatileAcidity +
##     FreeSulfurDioxide + TotalSulfurDioxide, family = poisson(link =
"log"),
##     data = training_data)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.582e+00  3.774e-02  41.912  < 2e-16 ***
## LabelAppeal         1.990e-01  6.012e-03  33.100  < 2e-16 ***
## STARS               2.139e-01  6.478e-03  33.015  < 2e-16 ***
## AcidIndex          -1.240e-01  4.381e-03 -28.305  < 2e-16 ***
## VolatileAcidity    -5.161e-02  6.492e-03  -7.950 1.87e-15 ***
## FreeSulfurDioxide   1.410e-04  3.511e-05   4.016 5.93e-05 ***
## TotalSulfurDioxide  1.069e-04  2.265e-05   4.720 2.36e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 22861  on 12794  degrees of freedom
## Residual deviance: 18575  on 12788  degrees of freedom
## AIC: 50531
##
## Number of Fisher Scoring iterations: 5

library(jtools)

## Warning: package 'jtools' was built under R version 4.4.2

library(broom)

## Warning: package 'broom' was built under R version 4.4.2

library(ggstance)

## Warning: package 'ggstance' was built under R version 4.4.2

##
## Attaching package: 'ggstance'

## The following objects are masked from 'package:ggplot2':
##
##     geom_errorbarh, GeomErrorbarh

# plot regression coefficients for poisson.model2 and poisson.model
plot_summs(poisson_model1, poisson_model2, scale = TRUE, exp = TRUE)
```



The graph compares the **exponentiated coefficients (exp(Estimate))** from two Poisson regression models, **Model 1** (blue circles) and **Model 2** (orange squares), highlighting the predictor effects on the outcome. Features like **LabelAppeal**, **STARS**, and **AcidIndex** have the strongest positive associations, with estimates well above 1.0, showing consistency across both models. In contrast, predictors such as **INDEX**, **FixedAcidity**, **CitricAcid**, and **pH** have estimates near 1.0, indicating minimal impact. Slight differences are observed for variables like **VolatileAcidity** and **TotalSulfurDioxide**, where Model 2 estimates are slightly lower. Overall, the graph demonstrates agreement between the models for key predictors while identifying features with limited or negligible influence.

```
# Negative Binomial models
library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

nb_model1 <- glm.nb(TARGET~LabelAppeal+STARS+AcidIndex, data = training_data)

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

```r
summary(nb_model1)
```

```
##
## Call:
## glm.nb(formula = TARGET ~ LabelAppeal + STARS + AcidIndex, data =
training_data,
##     init.theta = 37740.79016, link = log)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.599590   0.037380   42.79   <2e-16 ***
## LabelAppeal  0.199642   0.006009   33.23   <2e-16 ***
## STARS        0.215457   0.006473   33.28   <2e-16 ***
## AcidIndex   -0.126476   0.004366  -28.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(37740.79) family taken to be
1)
##
##     Null deviance: 22860  on 12794  degrees of freedom
## Residual deviance: 18677  on 12791  degrees of freedom
## AIC: 50630
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  37741
##           Std. Err.:  59951
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -50619.98
```

```r
### Need to use the data you transform using log . data not found on he
#nb_model2 <- glm.nb(target_variable ~ log_target + acidity_ratio, data =
training_data)

# Linear regression models
lm_model1 <- lm(TARGET~LabelAppeal+STARS+AcidIndex, data = training_data)
#lm_model2 <- lm(target_variable ~ log_target + acidity_ratio, data =
training_data)

# Compare coefficients
summary(poisson_model1)
```

```
##
## Call:
## glm(formula = TARGET ~ ., family = poisson(link = "log"), data =
training_data)
##
```

```
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          2.049e+00  1.960e-01  10.455  < 2e-16 ***
## INDEX               -4.537e-07  1.091e-06  -0.416 0.677395
## FixedAcidity        -4.437e-04  8.195e-04  -0.541 0.588195
## VolatileAcidity     -5.098e-02  6.492e-03  -7.854 4.04e-15 ***
## CitricAcid           1.345e-02  5.892e-03   2.282 0.022470 *
## ResidualSugar        1.491e-04  1.545e-04   0.965 0.334624
## Chlorides           -6.056e-02  1.645e-02  -3.682 0.000232 ***
## FreeSulfurDioxide    1.421e-04  3.513e-05   4.045 5.23e-05 ***
## TotalSulfurDioxide   1.073e-04  2.268e-05   4.729 2.26e-06 ***
## Density             -4.372e-01  1.921e-01  -2.276 0.022829 *
## pH                  -2.412e-02  7.639e-03  -3.157 0.001592 **
## Sulphates           -1.900e-02  5.738e-03  -3.312 0.000926 ***
## Alcohol              5.527e-03  1.410e-03   3.920 8.86e-05 ***
## LabelAppeal          1.996e-01  6.015e-03  33.182  < 2e-16 ***
## AcidIndex           -1.232e-01  4.461e-03 -27.617  < 2e-16 ***
## STARS                2.112e-01  6.492e-03  32.537  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 22861  on 12794  degrees of freedom
## Residual deviance: 18511  on 12779  degrees of freedom
## AIC: 50485
##
## Number of Fisher Scoring iterations: 5

summary(nb_model1)

##
## Call:
## glm.nb(formula = TARGET ~ LabelAppeal + STARS + AcidIndex, data =
training_data,
##     init.theta = 37740.79016, link = log)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.599590   0.037380   42.79   <2e-16 ***
## LabelAppeal  0.199642   0.006009   33.23   <2e-16 ***
## STARS        0.215457   0.006473   33.28   <2e-16 ***
## AcidIndex   -0.126476   0.004366  -28.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(37740.79) family taken to be
1)
##
##     Null deviance: 22860  on 12794  degrees of freedom
```

```
## Residual deviance: 18677  on 12791  degrees of freedom
## AIC: 50630
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  37741
##           Std. Err.:  59951
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -50619.98
```

```
summary(lm_model1)
```

```
##
## Call:
## lm(formula = TARGET ~ LabelAppeal + STARS + AcidIndex, data =
training_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8462 -0.6929  0.3930  1.0836  4.6306
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.19515    0.09845   42.61   <2e-16 ***
## LabelAppeal  0.60441    0.01704   35.46   <2e-16 ***
## STARS        0.72608    0.01963   36.98   <2e-16 ***
## AcidIndex   -0.34005    0.01103  -30.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.646 on 12791 degrees of freedom
## Multiple R-squared:  0.2701, Adjusted R-squared:   0.27
## F-statistic:  1578 on 3 and 12791 DF,  p-value: < 2.2e-16
```

The output is from a **Negative Binomial Regression** model 1, which was fitted to predict the `TARGET` variable using the predictors **LabelAppeal**, **STARS**, and **AcidIndex**. The model employs a **log link function**, and the estimated dispersion parameter (`theta`) is approximately **37,740.79**, indicating the model accounts for overdispersion in the data.

The **coefficients** section provides estimates of the relationship between each predictor and the target variable. The **intercept** has an estimate of **1.5996**, representing the baseline log count when all predictors are zero. For the predictors: - **LabelAppeal** has a positive estimate (**0.1996**), meaning that a unit increase in `LabelAppeal` increases the expected count of the target variable, holding other predictors constant. - **STARS** has a positive estimate (**0.2155**), indicating a similarly strong positive effect. - **AcidIndex** has a negative estimate (**-0.1265**), showing that a unit increase in `AcidIndex` decreases the expected count.

The **z-values** and **p-values** for all predictors are highly significant ($p < 0.001$), suggesting strong evidence that these variables influence the outcome. The **Null Deviance** (22,860) and **Residual Deviance** (18,677) indicate a significant reduction in deviance, showing the model fits the data well. Finally, the **AIC** (50,630) suggests the model's overall goodness of fit and can be used for comparison with alternative models.

4. Select Models

Choose the best models based on evaluation metrics such as AIC, residual deviance, or adjusted R^2.

```
# Model comparison
models <- list(poisson_model1, poisson_model2, nb_model1,
               #nb_model2,
               lm_model1
               #lm_model2
               )
# Calculate AIC for all models
aic_model1 <- AIC(poisson_model1)
aic_model2 <- AIC(poisson_model2)
aic_model3 <- AIC(nb_model1)
aic_model4 <- AIC(lm_model1)

# Combine AIC values into a vector
aic_values <- c(aic_model1, aic_model2, aic_model3, aic_model4)
names(aic_values) <- c("Model 1", "Model 2", "Model 3", "Model 4")

# BEst Model
library(caret)

# Define cross-validation settings
train_control <- trainControl(method = "cv", number = 5)

# Train models with cross-validation
set.seed(123)
model_cv <- train(TARGET~LabelAppeal+STARS+AcidIndex, data = training_data,
method = "glm", family = "poisson",
                  trControl = train_control)

# Print best model summary
print(model_cv$finalModel)

##
## Call:  NULL
##
## Coefficients:
## (Intercept)  LabelAppeal        STARS    AcidIndex
##      1.5996       0.1996       0.2155      -0.1265
##
```

```
## Degrees of Freedom: 12794 Total (i.e. Null);  12791 Residual
## Null Deviance:       22860
## Residual Deviance: 18680     AIC: 50630

# Select the best count regression model
best_model <- model_cv$finalModel  # Replace with your choice based on AIC or
other metrics

# Evaluate performance
predictions <- predict(best_model, newdata = evaluation_wine_data, type =
"response")
```

Comparison and Conclusion:

- **Model 1** (Full Poisson Regression) has the **lowest AIC (50,490)** and **lowest residual deviance (18,510)**, indicating the best performance among the models. However, it includes all predictors, which may make it prone to overfitting.
- **Model 2** (Reduced Poisson Regression) simplifies the model by including only significant predictors, with a minimal increase in AIC (50,530). It strikes a balance between model performance and interpretability.
- **Model 3** (Negative Binomial Regression) is useful when overdispersion is present, but its AIC (50,630) and residual deviance (18,680) are higher.
- **Model 4** is inappropriate for count data and can be ruled out.

Best Model: **Model 2** is likely the best choice as it simplifies the predictor set while maintaining near-optimal performance. If overdispersion is severe, **Model 3** (Negative Binomial) may be considered, but for simplicity and performance, **Model 2** is preferred.

Cross-validation is used to evaluate the performance of the **Poisson_model2** and ensure its generalizability to unseen data. In this approach, the dataset is split into **k-folds** (commonly 5 or 10 folds), where the model is trained on **k-1 folds** and validated on the remaining fold, repeating the process for all folds. The average performance metrics, such as the **AIC** (Akaike Information Criterion), **Residual Deviance**, or prediction error, are calculated across all iterations. For **Poisson_model2**, cross-validation helps assess how well the predictors (**LabelAppeal**, **STARS**, and **AcidIndex**) fit the target variable while guarding against overfitting. If the model consistently shows a low AIC and good fit across folds, it confirms the model's reliability and performance on new data.