Sales Prediction and Forecasting Report

Wine Sales

Singh, Gurjeet

This report contains analysis done while building various regression models to predict the number of wine cases ordered based upon the wine characteristics.

Table of Contents

Table of Figures

Introduction:

The purpose of this project is to assist a large wine manufacturer in studying wine data set in order to predict the number of wine cases ordered based upon the wine characteristics. If the wine manufacturer can predict the number of cases, then it will be able to adjust their wine offerings to maximize sales.

For our purposes, we use the data set that contains information on approximately 12,000 commercially available wines. The variables are mostly related to the chemical properties of the wine being sold. The target variable is the number of sample cases of wine that were purchased by wine distribution companies after sampling a wine. These cases would be used to provide tasting samples to restaurants and wine stores around the United States. The more sample cases purchased, the more likely is a wine to be sold at a high end restaurant.

Our goal is to build models to predict the number of cases of wine that will be sold given certain properties of the wine.

The training dataset contains 12,795 observations and 16 explanatory variables of which 16 are numerical variables. The test dataset contains 3,335 observations and 16 explanatory variables. In the test dataset, there is no "TARGET" values. We will be using our selected model to score the test data file to predict the number of cases of wine that will be sold.

Figure 1 gives the basic descriptions of each field and how those affect the prediction.

Figure 1: Table - Data Dictionary

VARIABLE NAME	DEFINITION	THEORETICAL EFFECT
INDEX	Identification Variable (do not use)	None
TARGET	Number of Cases Purchased	None
AcidIndex	Proprietary method of testing total acidity of wine by using a weighted average	
Alcohol	Alcohol Content	
Chlorides	Chloride content of wine	
CitricAcid	Citric Acid Content	
Density	Density of Wine	
FixedAcidity	Fixed Acidity of Wine	
FreeSulfurDioxide	Sulfur Dioxide content of wine	
LabelAppeal	Marketing Score indicating the appeal of label design for consumers. High numbers suggest customers like the label design. Negative numbers suggest customes don't like the design.	Many consumers purchase based on the visual appeal of the wine label design. Higher numbers suggest better sales.
ResidualSugar	Residual Sugar of wine	
STARS	Wine rating by a team of experts. 4 Stars = Excellent, 1 Star = Poor	A high number of stars suggests high sales
Sulphates	Sulfate conten of wine	
TotalSulfurDioxide	Total Sulfur Dioxide of Wine	
VolatileAcidity	Volatile Acid content of wine	
pH	pH of wine	

Now that we have some context for our analysis and dataset, let's look at the results in the next section.

Section 1: Data Exploration

The first step towards any modeling project is the Data Exploration i.e. Exploratory Data Analysis (EDA). This helps us to understand and analyze the data set to summarize the main characteristics of variables. For this purposes, we will look into the basic statistics to understand the data and examine the distributions of the variables.

Section 1.1: Statistics

Figure 2 shows the statistical values of the numerical variables in the wine data set. The number of observations in the data is 12,795 records. We can clearly see that there are missing values (NAs) in the data for variables, ResidualSugar, Chlorides, FreeSulfurDioxide, TotalSulfurDioxide, pH, Sulphates, Alcohol, and STARS. In Table 2, we can also see outliers in various variables in the data. For instance, the variables, ResidualSugar, FreeSulfurDioxide, and TotalSulfurDioxide has a minimum and maximum values so far from the mean and median values. We will cap the upper and lower values of these variables during the data preparation stage. Except a few variables, all the variables have negative values. Some of the variables negative value doesn't make sense. It is hard to conclude if these are accurate or entered in error because there are so many. It is my understanding that this data has been standardized at some point. Hence, we are seeing negative values.

We explained how we fixed the missing values and outliers in Section 2: Data Preparation.

Figure 2: Table - Statistical Values of Numerical Variables

Variable Names	nobs	NAs	Minimum	Maximum	1. Quartile	3. Quartile	Mean	Median	SE Mean	Variance	Stdev
INDEX	12795	0	1.000	16129.000	4037.500	12106.500	8069.980	8110.000	41.170	21686770.000	4656.905
TARGET	12795	0	0.000	8.000	2.000	4.000	3.029	3.000	0.017	3.711	1.926
FixedAcidity	12795	0	-18.100	34.400	5.200	9.500	7.076	6.900	0.056	39.913	6.318
VolatileAcidity	12795	0	-2.790	3.680	0.130	0.640	0.324	0.280	0.007	0.615	0.784
CitricAcid	12795	0	-3.240	3.860	0.030	0.580	0.308	0.310	0.008	0.743	0.862
ResidualSugar	12795	616	-127.800	141.150	-2.000	15.900	5.419	3.900	0.306	1139.021	33.749
Chlorides	12795	638	-1.171	1.351	-0.031	0.153	0.055	0.046	0.003	0.101	0.318
FreeSulfurDioxide	12795	647	-555.000	623.000	0.000	70.000	30.846	30.000	1.349	22116.020	148.715
TotalSulfurDioxide	12795	682	-823.000	1057.000	27.000	208.000	120.714	123.000	2.107	53783.740	231.913
Density	12795	0	0.888	1.099	0.988	1.001	0.994	0.994	0.000	0.001	0.027
pH	12795	395	0.480	6.130	2.960	3.470	3.208	3.200	0.006	0.462	0.680
Sulphates	12795	1210	-3.130	4.240	0.280	0.860	0.527	0.500	0.009	0.869	0.932
Alcohol	12795	653	-4.700	26.500	9.000	12.400	10.489	10.400	0.034	13.897	3.728
LabelAppeal	12795	0	-2.000	2.000	-1.000	1.000	-0.009	0.000	0.008	0.794	0.891
AcidIndex	12795	0	4.000	17.000	7.000	8.000	7.773	8.000	0.012	1.753	1.324
STARS	12795	3359	1.000	4.000	1.000	3.000	2.042	2.000	0.009	0.815	0.903

Section 1.2: Examining Distributions

We continue our data exploration by examining the distribution after fixing the missing values to get an appropriate distribution. So, if you notice any variable name starting with "IMP_", it means that the variable is imputed.

Figure 3 shows the distributions of values for the variable, TARGET and TARGET classified by STARS. From the left histogram, we see that our data is zero inflated. There are total of 2,734 records (21.36% approx.) with TARGET equals to zero out of the total population. This seems to be a lot of zero. This provides us with an indication that we will need to use Zero Inflated Poisson and Zero Inflated Negative Binomial to deal with zero inflation. We will discuss each of these models in the model building section.

In Figure 3, we notice that STARS seems to be highly predictable. If the STARS are missing. There's hardly any cases of wine sold. More wine cases are sold if there's a STARS associated with it.

Figure 3: Histogram: TARGET

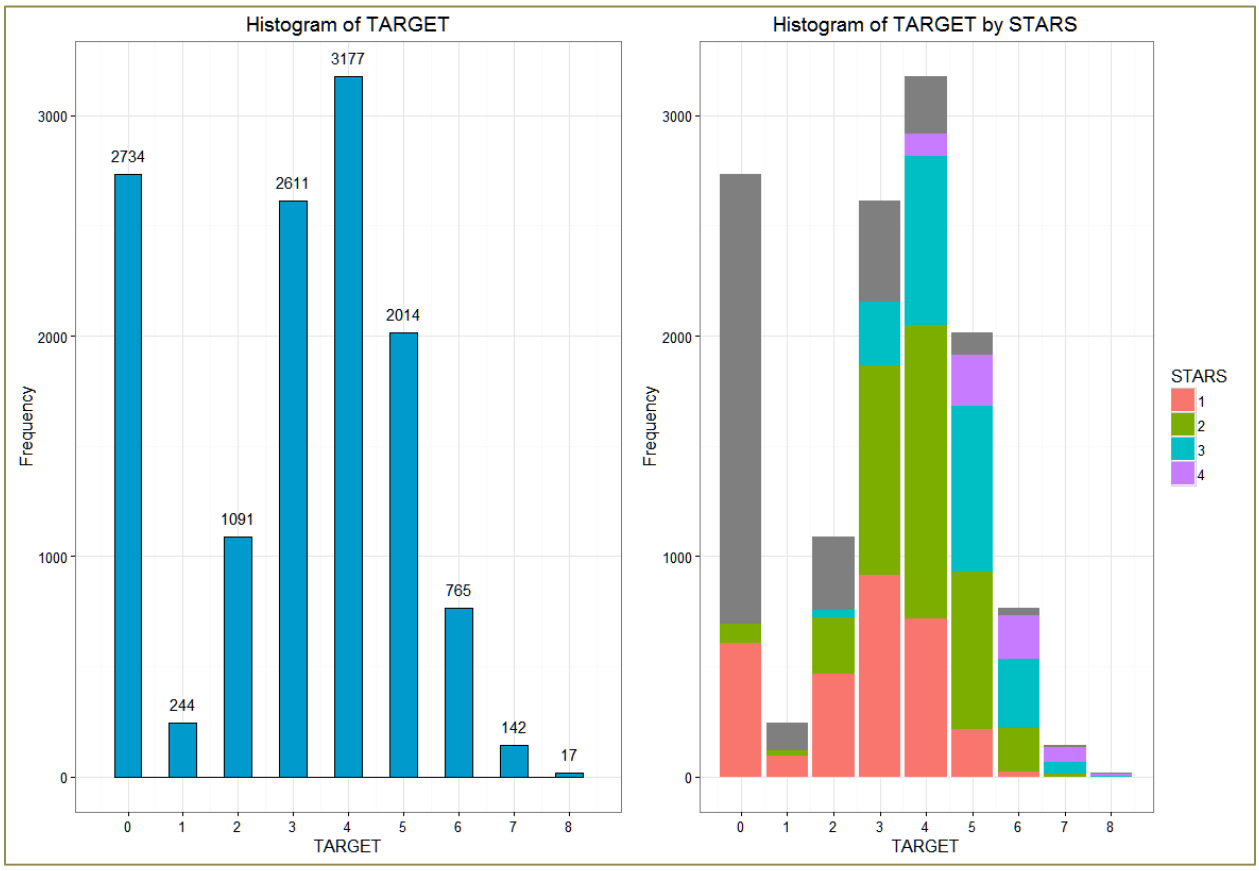


Figure 4 shows the distributions of values for the variables, FixedAcidity and VolatileAcidity. Each of these distribution plots shows a very high peak in the middle of the histogram which is surrounded by the smaller values. We also notice negative values. Our doubts is confirmed that these values are standardized and hence we see negative values in the data. Further investigation may be required to understand such negative values in the data. However, for our purposes, we will consider this data normal and proceed forward.

Figure 4: Histogram: FixedAcidity and VolatileAcidity

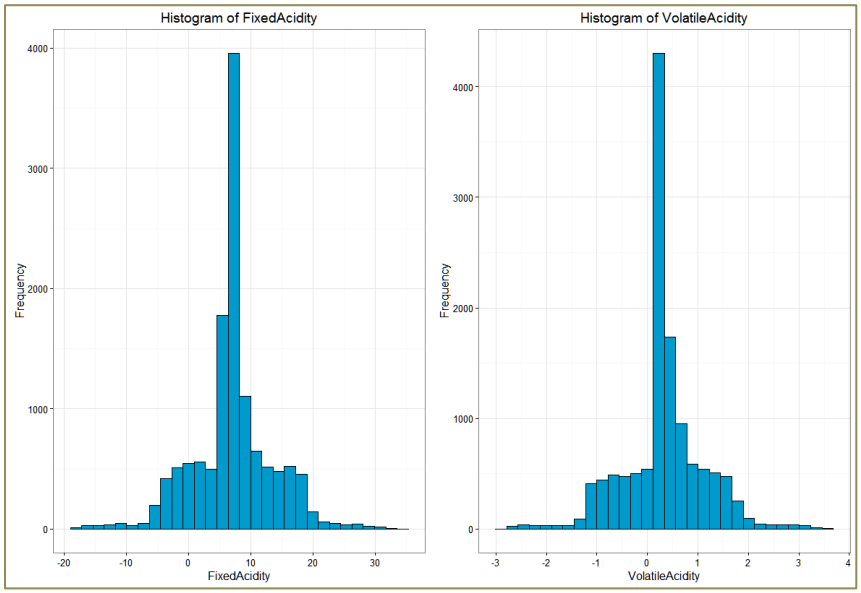


Figure 5 shows the distributions of values for the variables, CitricAcid and IMP_ResidualSugar. Again, each of these distribution plots shows a very high peak in the middle of the histogram which is surrounded by the smaller values.

Figure 5: Histogram: CitricAcid and IMP_ResidualSugar

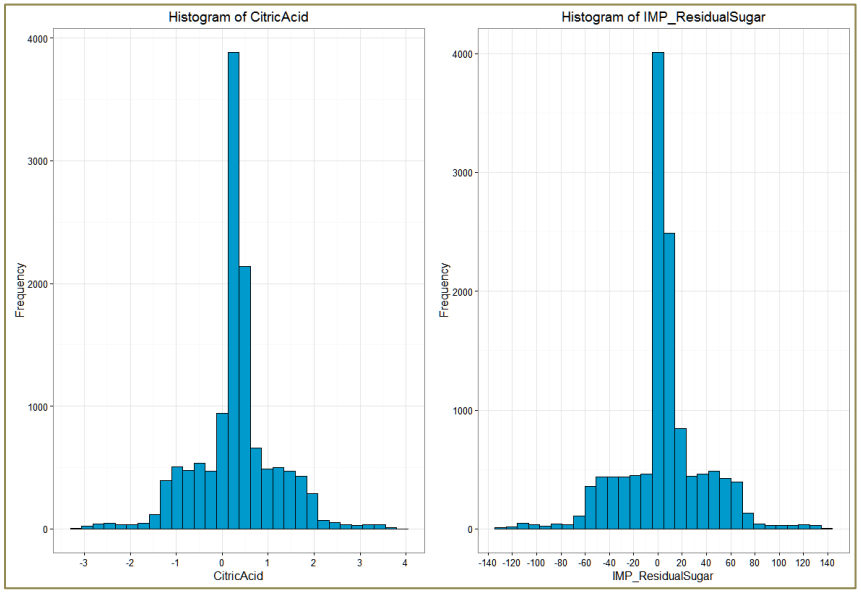


Figure 6 shows the distributions of values for the variables, IMP_Chlorides and IMP_FreeSulfurDioxide. Again, each of these distribution plots shows a very high peak in the middle of the histogram which is surrounded by the smaller values.

Figure 6: Histogram: IMP_Chlorides and IMP_FreeSulfurDioxide

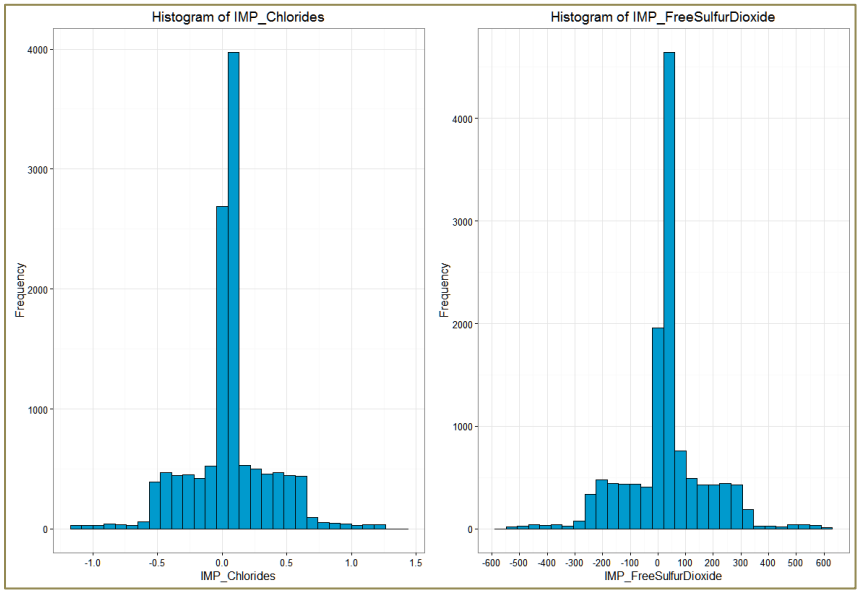


Figure 7 shows the distributions of values for the variables, IMP_TotalSulfurDioxide and Density. Again, each of these distribution plots shows a very high peak in the middle of the histogram which is surrounded by the smaller values.

Figure 7: Histogram: IMP_TotalSulfurDioxide and Density

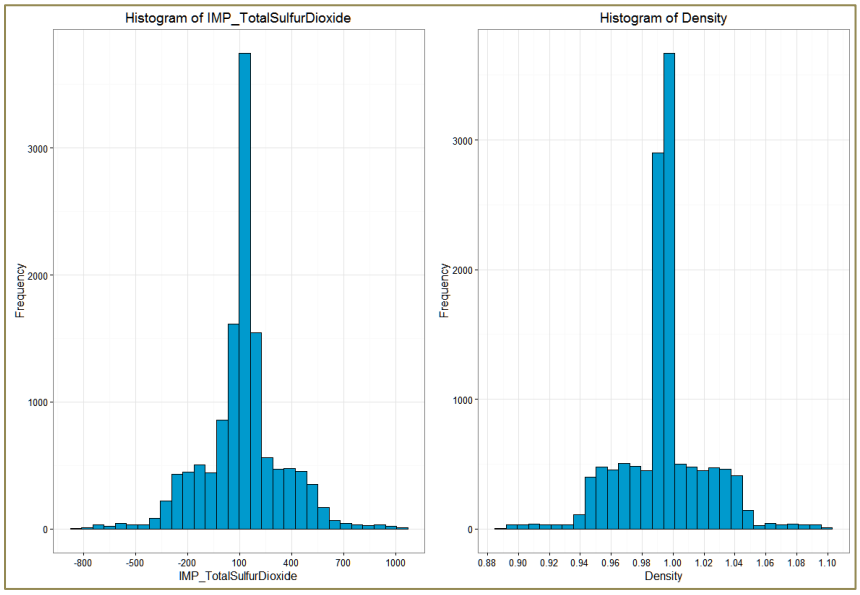


Figure 8 shows the distributions of values for the variables, IMP_pH and IMP_Sulphates. Again, each of these distribution plots shows a very high peak in the middle of the histogram which is surrounded by the smaller values.

Figure 8: Histogram: IMP_pH and IMP_Sulphates

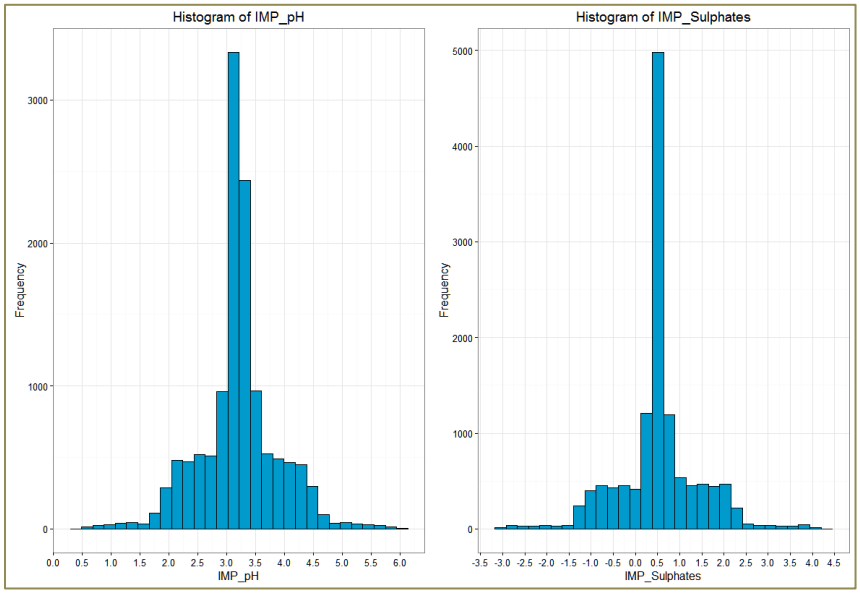


Figure 9 shows the distributions of values for the variables, IMP_Alcohol and LabelAppeal. Again, the IMP_Alcohol distribution plot shows a very high peak in the middle of the histogram which is surrounded by the smaller values.

Figure 9: Histogram: IMP_Alcohol and LabelAppeal

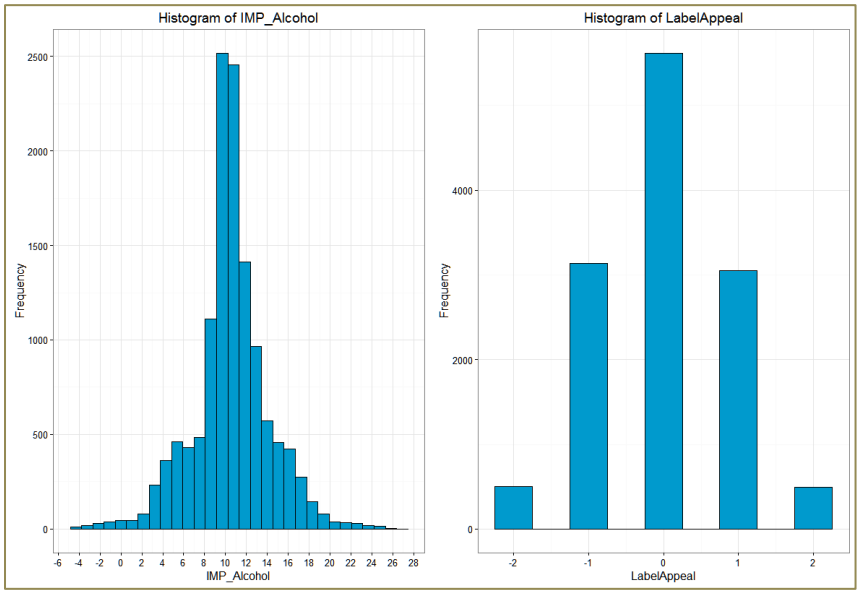


Figure 10 shows the distributions of values for the variables, AcidIndex and IMP_STARS.

Figure 10: Histogram: AcidIndex and IMP_STARS

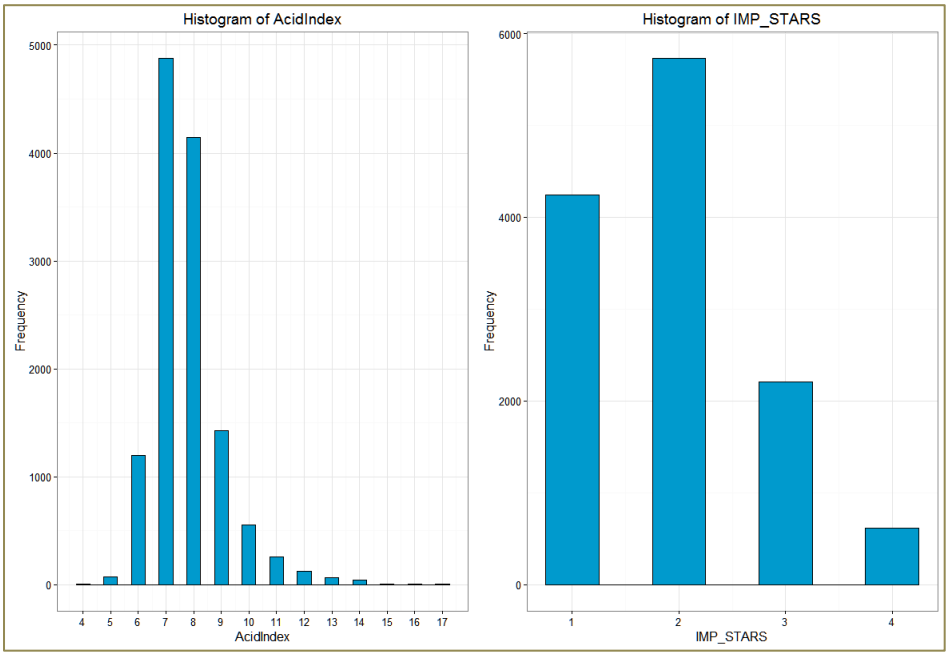
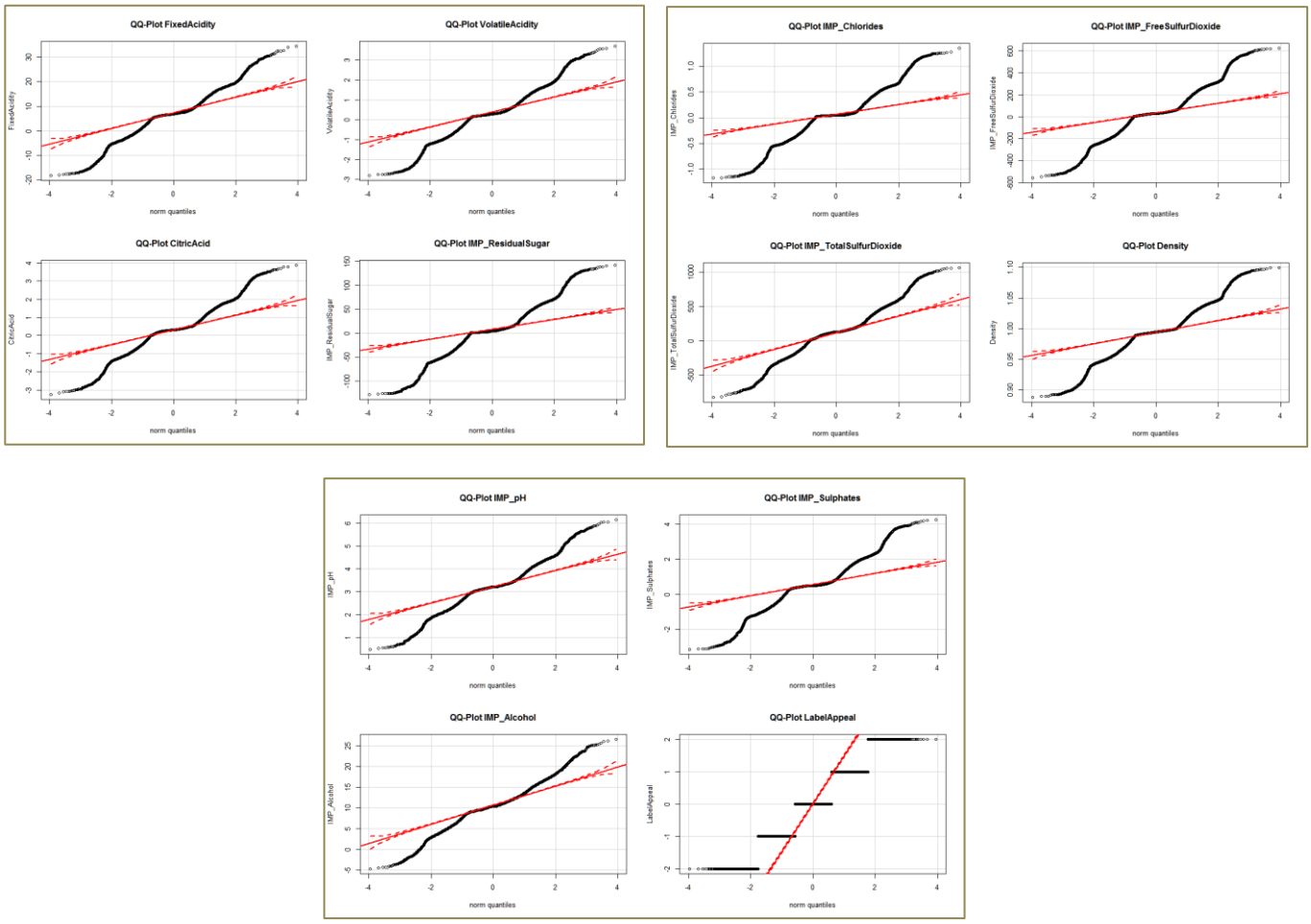


Figure 11 shows the Q-Q plots of the variables. It seems like all exhibit the same behavior. The line curve up, then back down, and then back up again. All the plots shows the same pattern and differ only slightly in the sharpness of their curves. LabelAppeal has a different pattern because it is not a continuous variable.

Figure 11: Q-Q Plots



Section 2: Data Preparation

In Section 1: Data Exploration, we noticed that we had some missing values and a few outliers in the data. This section explains the methodology we took to fix the missing values and transform data to eliminate outliers. First, we fixed the missing values by using the Decision Tree concepts for variable STARS. Figure 12 shows the Decision Tree to impute STARS. We imputed "1" for observations where LabelAppeal is less than -0.5 i.e. where LabelAppeal is -1 or -2. We imputed "2" for the rest of the observations with missing data i.e. LabelAppeal >= -0.5

During our data exploration, we notice that various variables had negative values. There wasn't any patterns that we found to fix those. We assume that this data was already standardized at some point, so we do not try to fix any negative values.

Figure 12: Decision Tree: Impute STARS

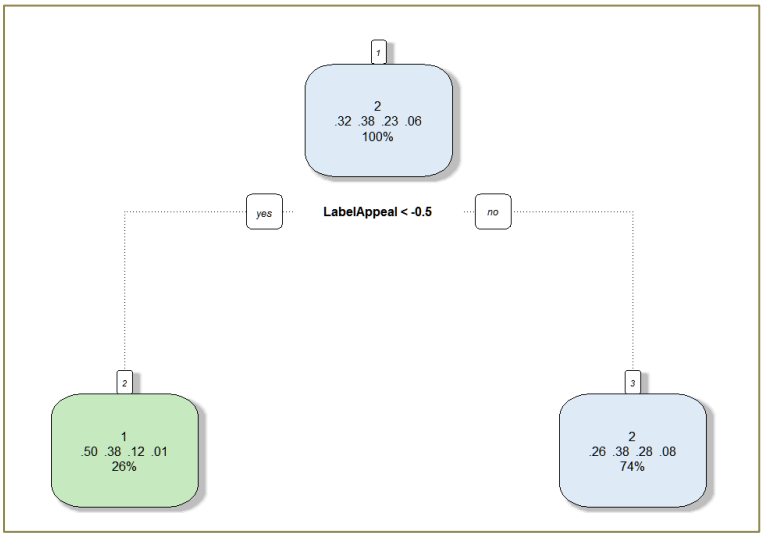


Figure 13 shows all the other variables and the values we used to impute. We used Median values to impute variables listed in the figure. Figure 13 also shows three (3) variables whose min and max values were cap to eliminate some high extreme ends.

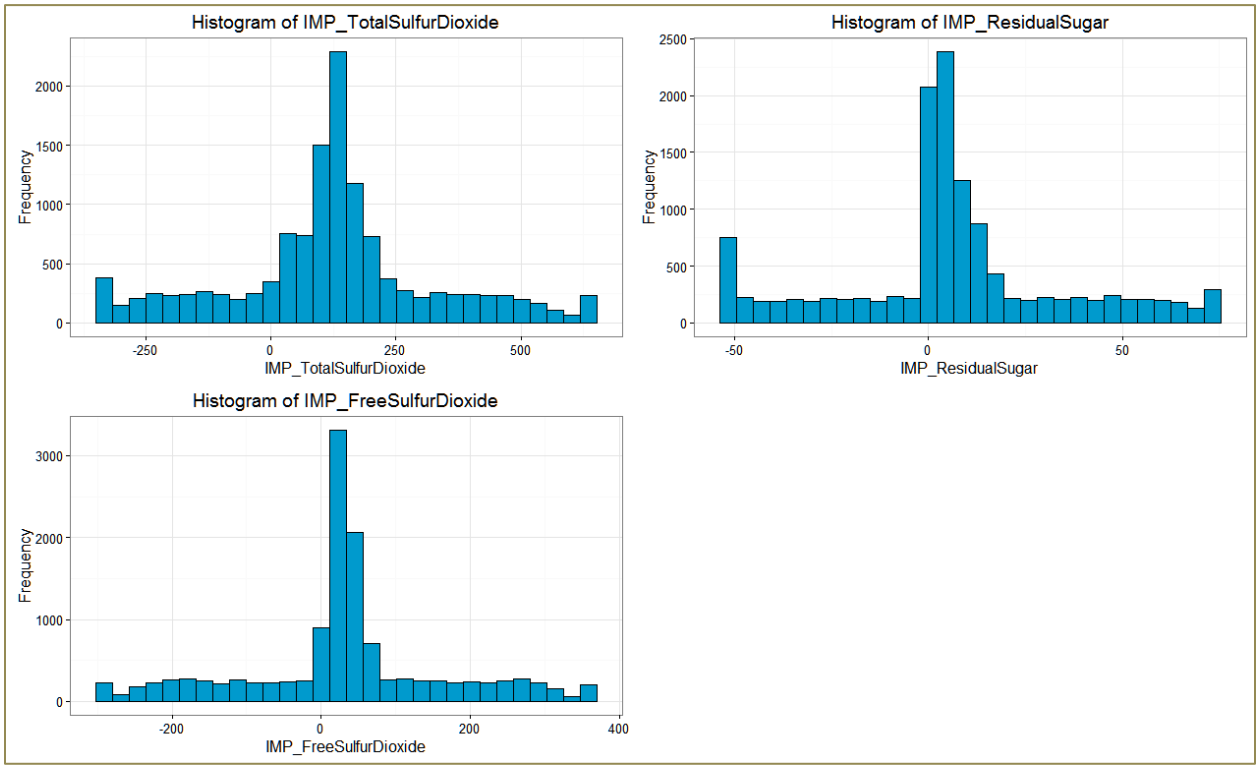
Figure 13: Table – Imputed Values

Variable Name	Median Value	New Variable Name
ResidualSugar	3.9	IMP_ResidualSugar
Chlorides	0.05	IMP_Chlorides
FreeSulfurDioxide	30	IMP_FreeSulfurDioxide
TotalSulfurDioxide	123	IMP_TotalSulfurDioxide
pH	3.2	IMP_pH
Sulphates	0.5	IMP_Sulphates
Alcohol	10.4	M_Sulphates

Variable Name	Min. Value	Max. Value
IMP_TotalSulfurDioxide	-350	620
IMP_ResidualSugar	-50	75
IMP_FreeSulfurDioxide	-300	350

Figure 14 shows the new distribution for IMP_TotalSulfurDioxide, IMP_ResidualSugar, and IMP_FreeSulfurDioxide after fixing the extreme points. If you compare this with Figures 5, Figure 6, and Figure 7, you will notice that for these variables the distribution is pulled together and look way better than before.

Figure 14: Histogram - Distribution



We have also created indicator variables to flag the change of the value. Zero ("0") for original value and one ("1") for imputed value. Figure 15 shows all the variables including the imputed variables and indicator variables. The variables that are highlighted in yellow had missing values and we imputed those by creating corresponding variables highlighted in green.

Figure 15: Table – Statistical values with Indicator variables

Variable Names	nobs	NAs	Minimum	Maximum	1. Quartile	3. Quartile	Mean	Median	Variance	Stdev
INDEX	12795	0	1.000	16129.000	4037.500	12106.500	8069.980	8110.000	21686765.176	4656.905
TARGET	12795	0	0.000	8.000	2.000	4.000	3.029	3.000	3.711	1.926
FixedAcidity	12795	0	-18.100	34.400	5.200	9.500	7.076	6.900	39.913	6.318
VolatileAcidity	12795	0	-2.790	3.680	0.130	0.640	0.324	0.280	0.615	0.784
CitricAcid	12795	0	-3.240	3.860	0.030	0.580	0.308	0.310	0.743	0.862
ResidualSugar	12795	616	-127.800	141.150	-2.000	15.900	5.419	3.900	1139.021	33.749
Chlorides	12795	638	-1.171	1.351	-0.031	0.153	0.055	0.046	0.101	0.318
FreeSulfurDioxide	12795	647	-555.000	623.000	0.000	70.000	30.846	30.000	22116.020	148.715
TotalSulfurDioxide	12795	682	-823.000	1057.000	27.000	208.000	120.714	123.000	53783.737	231.913
Density	12795	0	0.888	1.099	0.988	1.001	0.994	0.994	0.001	0.027
pH	12795	395	0.480	6.130	2.960	3.470	3.208	3.200	0.462	0.680
Sulphates	12795	1210	-3.130	4.240	0.280	0.860	0.527	0.500	0.869	0.932
Alcohol	12795	653	-4.700	26.500	9.000	12.400	10.489	10.400	13.897	3.728
LabelAppeal	12795	0	-2.000	2.000	-1.000	1.000	-0.009	0.000	0.794	0.891
AcidIndex	12795	0	4.000	17.000	7.000	8.000	7.773	8.000	1.753	1.324
STARS	12795	3359	1.000	4.000	1.000	3.000	2.042	2.000	0.815	0.903
IMP_ResidualSugar	12795	0	-50.000	75.000	0.900	14.900	5.894	3.900	835.120	28.898
M_ResidualSugar	12795	0	0.000	1.000	0.000	0.000	0.048	0.000	0.046	0.214
IMP_Chlorides	12795	0	-1.171	1.351	0.000	0.128	0.055	0.048	0.096	0.310
M_Chlorides	12795	0	0.000	1.000	0.000	0.000	0.050	0.000	0.047	0.218
IMP_FreeSulfurDioxide	12795	0	-300.000	350.000	5.000	64.000	30.524	30.000	17755.967	133.252
M_FreeSulfurDioxide	12795	0	0.000	1.000	0.000	0.000	0.051	0.000	0.048	0.219
IMP_TotalSulfurDioxide	12795	0	-350.000	620.000	34.000	198.000	121.431	123.000	42736.339	206.728
M_TotalSulfurDioxide	12795	0	0.000	1.000	0.000	0.000	0.053	0.000	0.050	0.225
IMP_pH	12795	0	0.480	6.130	2.970	3.450	3.207	3.200	0.448	0.669
M_pH	12795	0	0.000	1.000	0.000	0.000	0.031	0.000	0.030	0.173
IMP_Sulphates	12795	0	-3.130	4.240	0.340	0.770	0.525	0.500	0.787	0.887
M_Sulphates	12795	0	0.000	1.000	0.000	0.000	0.095	0.000	0.086	0.293
IMP_Alcohol	12795	0	-4.700	26.500	9.100	12.200	10.485	10.400	13.188	3.631
M_Alcohol	12795	0	0.000	1.000	0.000	0.000	0.051	0.000	0.048	0.220
IMP_STARS	12795	0	1.000	4.000	1.000	2.000	1.937	2.000	0.692	0.832
M_STARS	12795	0	0.000	1.000	0.000	1.000	0.263	0.000	0.194	0.440

Figure 16 shows all the variables that were dropped prior to model building.

Figure 16: Table – Dropped Variables

Variables Dropped	
ResidualSugar	Chlorides
pH	Sulphates
FreeSulfurDioxide	TotalSulfurDioxide
Alcohol	STARS

Section 3: Model Building

For the model building, we started the approach of adding one variable at a time and observed how it benefitted the mode. If the variable was not statistically significant or adding much value to the model, it was dropped. The reason for this approach was that we do not know much on the wine chemistry except that it has grape and alcohol. To be safe, we wanted to test out all the variables and keep only those that performed well. After trying different combinations (of course, combinations that made sense) for each category of distribution, we selected five models that gave us decent metrics and also logically made sense.

This section explains the five (5) models (Model 1, Model 2, Model 3, Model 4, and Model 5) that we have created to predict the number of cases of wine that will be sold given certain properties of the wine. Out of these models, we have selected one (1) that fits the best and also logically correct to use for predicting the number of cases of wine.

We have further split our wine data into two sets, train and test. Seventy (70) percent of the data is split into train and 30 percent to test. This further split will help us better test the accuracy of our models and help us decide which one to pick based on various metrics.

Figure 17 shows us the Mean is higher than the Variance for the TARGET variable (for non-zero values). This means that we will need to deal with the underdispersion.

Figure 17: TARGET – Mean and Variance

Models	Data Type	Mean	Variance	Mean (TARGET > 0)	Variance (TARGET > 0)	Note
Wine Data	Original Raw data	3.029074	3.710895	3.852202	1.548233	Mean is higher than Variance. Shows that we need to deal with underdispersion.
train.df	Training Data	3.037915	3.675408	3.845130	1.547865	
test.df	Test Data	3.008563	3.793594	3.868869	1.549221	

Section 3.1: Model 1 - Poisson

Figure 18 shows the summary output of the Model 1. Model 1 is created using Poisson distribution. We started with a simple model with one variable and then added additional variable as they add value to the model. This model shows that IMP_Cholorides is not statistically significant because the p-value is over 0.05. We also noticed that IMP_Alcohol and IMP_TotalSulfurDioxide are statistically significant but their coefficients are almost close to zero (0). We will remove these variables and will refit the next model using Negative Binomial in Model 2.

The residual deviance statistic of Model 1 is **9548.1** with **8932 degrees of freedom (df)**. The **Value/DF** = **9548.1/8932** = **1.069** which is very close to 1. This means that model fits but not that well because the Value/DF should be less than 1. This slight lack of fitting could be due to the zero inflation in the data.

Figure 18: Model 1: Poisson Output

```
Call:
glm(formula = TARGET ~ IMP_STARS + M_STARS + LabelAppeal + AcidIndex +
    VolatileAcidity + IMP_Alcohol + IMP_Chlorides + IMP_TotalSulfurDioxide,
    family = poisson, data = train.df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.2072  -0.6134   0.0142   0.4396   3.6629

Coefficients:
                        Estimate  Std. Error z value             Pr(>|z|)
(Intercept)           1.51550182  0.04891232  30.984 < 0.0000000000000002 ***
IMP_STARS             0.17589492  0.00729912  24.098 < 0.0000000000000002 ***
M_STARS              -0.95415985  0.02039813 -46.777 < 0.0000000000000002 ***
LabelAppeal           0.15807792  0.00749388  21.094 < 0.0000000000000002 ***
AcidIndex            -0.08556870  0.00537214 -15.928 < 0.0000000000000002 ***
VolatileAcidity      -0.03018128  0.00778143  -3.879             0.000105 ***
IMP_Alcohol           0.00353624  0.00166695   2.121             0.033889 *
IMP_Chlorides        -0.03758389  0.01961416  -1.916             0.055345 .
IMP_TotalSulfurDioxide  0.00007935  0.00002926   2.712             0.006696 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 15769.2  on 8940  degrees of freedom
Residual deviance:  9548.1  on 8932  degrees of freedom
AIC: 31981

Number of Fisher Scoring iterations: 6
```

Figure 19 shows the equation of the Model 1. This model gives an intercept of 1.5155 and the coefficient of the predictor variables shown in Figure 19. For instance, the coefficient of IMP_STARS is 0.17589 approx. This model tells us that for each 1 unit increase in IMP_STARS increases the expected number of wine cases sale (TARGET) by a factor of exp(0.17589) = 1.19231 or about {100 * (exp(0.17589) -1)} = 19.23 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 0.15808 approx. This model tells us that for each 1 unit increase in LabelAppeal increases the expected number of wine cases sale (TARGET) by a factor of exp(0.15808) = 1.17126 or about {100 * (exp(0.15808) -1)} = 17.12 percent, while holding all the other variables constant.

The coefficient of AcidIndex is -0.08557 approx. This model tells us that for each 1 unit increase in AcidIndex decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.08557) = 0.91799 or about {100 * (exp(-0.08557) -1)} = 8.2 percent, while holding all the other variables constant.

The coefficient of VolatileAcidity is -0.03018 approx. This model tells us that for each 1 unit increase in VolatileAcidity decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.03018) = 0.97026 or about {100 * (exp(-0.03018) -1)} = 2.97 percent, while holding all the other variables constant.

The coefficient of IMP_Alcohol is 0.00354 approx. This model tells us that for each 1 unit increase in IMP_Alcohol increases the expected number of wine cases sale (TARGET) by a factor of exp(0.00354) = 1.00354 or about {100 * (exp(0.00354) -1)} = 0.35 percent, while holding all the other variables constant.

The coefficient of IMP_Chlorides is 0.03758 approx. This model tells us that for each 1 unit increase in IMP_Chlorides decreases the expected number of wine cases sale (TARGET) by a factor of exp(0.03758) = 0.96311 or about {100 * (exp(0.03758) -1)} = 3.68 percent, while holding all the other variables constant.

The coefficient of IMP_TotalSulfurDioxide is 0.00008 approx. This model tells us that for each 1 unit increase in IMP_TotalSulfurDioxide increases the expected number of wine cases sale (TARGET) by a factor of exp(0.00008) = 1.00007 or about {100 * (exp(0.00008) -1)} = 0.007 percent, while holding all the other variables constant.

Figure 19: Model 1: Equation

```
TARGET = 1.5155                          +
         0.17589 * IMP_STARS             -
         0.95416 *  M_STARS              +
         0.15808 *  LabelAppeal          -
         0.08557 *  AcidIndex            -
         0.03018 *  VolatileAcidity      +
         0.00354 *  IMP_Alcohol          -
         0.03758 *  IMP_Chlorides        +
         0.00008 *  IMP_TotalSulfurDioxide
```

Figure 20 shows the coefficients, expected count, and percentage values of the Model 1. We can clearly see that IMP_TotalSulfurDioxide effect is very negligible.

Figure 20: Model 1: Poisson Coefficients

	Coefficients	Count	Percent
(Intercept)	1.51550181828	4.5517047	355.170468245
IMP_STARS	0.17589492309	1.1923128	19.231276753
M_STARS	-0.95415985301	0.3851356	-61.486442083
LabelAppeal	0.15807792317	1.1712575	17.125745925
AcidIndex	-0.08556869884	0.9179901	-8.200992396
VolatileAcidity	-0.03018128054	0.9702696	-2.973037340
IMP_Alcohol	0.00353623956	1.0035425	0.354249943
IMP_Chlorides	-0.03758388544	0.9631136	-3.688637688
IMP_TotalSulfurDioxide	0.00007934709	1.0000794	0.007935024

Figure 21 shows the statistics of the Model 1. We see that the Mean Square Error (MSE) and Mean Absolute Error (MAE) for the test data (i.e. out of sample) is way higher than the training data (i.e. in-sample). This means that this model is overfitting. The model tells us that average error in the prediction is about 1 wine case i.e. MAE = 1.05 for test data.

Figure 21: Model 1: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_1	Poisson	31980.77	0.573410	1.796864	0.512895	1.056042

Section 3.2: Model 2 – Negative Binomial

Figure 22 shows the summary output of the Model 2. Model 2 is created using Negative Binomial distribution. We ran this model using the same variables in the Model 1. We noticed that variables, IMP_Cholorides, IMP_Alcohol, and IMP_TotalSulfurDioxide, were not adding a lot of value to the model. Also, IMP_Cholorides was not statistically significant. Hence, we dropped these variables and re-ran the model. All the remaining variables are statistically significant because the p-value is less than 0.05. It appears that all variables are contributing to the model. We will find out more once we interpret each parameter.

The residual deviance statistic of Model 2 is **9563.1** with **8935 degrees of freedom (df)**. The **Value/DF** = **9563.1/8935** = **1.07** which is very close to 1. This means that model fits but not that well because the Value/DF should be less than 1. This slight lack of fitting could be due to the zero inflation in the data.

Figure 22: Model 2: Negative Binomial Output

```
Call:
glm.nb(formula = TARGET ~ IMP_STARS + M_STARS + LabelAppeal +
    AcidIndex + VolatileAcidity, data = train.df, init.theta = 40622.08892,
    link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.1853  -0.6233   0.0084   0.4531   3.6825

Coefficients:
                Estimate Std. Error z value             Pr(>|z|)
(Intercept)     1.565651   0.044910  34.862 < 0.0000000000000002 ***
IMP_STARS       0.177444   0.007285  24.357 < 0.0000000000000002 ***
M_STARS        -0.956033   0.020393 -46.880 < 0.0000000000000002 ***
LabelAppeal     0.157509   0.007493  21.021 < 0.0000000000000002 ***
AcidIndex      -0.086563   0.005365 -16.134 < 0.0000000000000002 ***
VolatileAcidity -0.030361   0.007779  -3.903            0.0000949 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for Negative Binomial(40622.09) family taken to be 1)

    Null deviance: 15768.4  on 8940  degrees of freedom
Residual deviance:  9563.1  on 8935  degrees of freedom
AIC: 31992

Number of Fisher Scoring iterations: 1


              Theta:  40622
          Std. Err.:  41049
Warning while fitting theta: iteration limit reached

 2 x log-likelihood:  -31978.47
```

Figure 23 shows the equation of the Model 2. This model gives an intercept of 1.5656 and the coefficient of the predictor variables shown in Figure 23. For instance, the coefficient of IMP_STARS is 0.17744 approx. This model tells us that for each 1 unit increase in IMP_STARS increases the expected number of wine cases sale (TARGET) by a factor of exp(0.17744) = 1.19416 or about {100 * (exp(0.17744) -1)} = 19.41 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 0.15751 approx. This model tells us that for each 1 unit increase in LabelAppeal increases the expected number of wine cases sale (TARGET) by a factor of exp(0.15751) = 1.17059 or about {100 * (exp(0.15751) -1)} = 17.06 percent, while holding all the other variables constant.

The coefficient of AcidIndex is -0.08656 approx. This model tells us that for each 1 unit increase in AcidIndex decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.08656) = 0.91708 or about {100 * (exp(-0.0865) -1)} = 8.3 percent, while holding all the other variables constant.

The coefficient of VolatileAcidity is -0.03036 approx. This model tells us that for each 1 unit increase in VolatileAcidity decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.03036) = 0.97009 or about {100 * (exp(-0.03036) -1)} = 2.99 percent, while holding all the other variables constant.

Figure 23: Model 2: Equation

$$
\begin{aligned}
\text{TARGET} = 1.56565 \\
&+ 0.17744 * \text{IMP\_STARS} \\
&- 0.95603 * \text{M\_STARS} \\
&+ 0.15751 * \text{LabelAppeal} \\
&- 0.08656 * \text{AcidIndex} \\
&- 0.03036 * \text{VolatileAcidity}
\end{aligned}
$$

Figure 24 shows the coefficients, expected count, and percentage values of the Model 2.

Figure 24: Model 2: Negative Binomial Coefficients

	Coefficients	Count	Percent
(Intercept)	1.56565095	4.7857892	378.578924
IMP_STARS	0.17744414	1.1941614	19.416136
M_STARS	-0.95603289	0.3844149	-61.558512
LabelAppeal	0.15750896	1.1705912	17.059125
AcidIndex	-0.08656264	0.9170781	-8.292189
VolatileAcidity	-0.03036089	0.9700954	-2.990462

Figure 25 shows the statistics of the Model 2. Again, we see that the Mean Square Error (MSE) and Mean Absolute Error (MAE) for the test data (i.e. out of sample) is way higher than the training data (i.e. in-sample). This means that this model is overfitting as well. The model tells us that average error in the prediction is about 1 wine case i.e. MAE = 1.05 for test data. The MAE of this model seems to be fairly close to Model 1.

Figure 25: Model 2: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_2	Neg. Binomial	31992.47	0.575165	1.802524	0.513362	1.058692

Section 3.3: Model 3 – Zero Inflated Poisson

Figure 26 shows the summary output of the Model 3. Model 3 is created using Zero Inflated Poisson distribution. We started with a simple model with one variable and then added additional variable as they add value to the model. In the figure below, the first block of output in the model call contains Poisson regression coefficients for each of the variables along with the standard errors, z-scores, and p-value for the coefficients. The second block corresponds to the inflation model. Since we know that our data contains excess zeros i.e. zero inflation, this block includes logit coefficients for predicting excess zeros along with their standard errors, z-scores, and p-values.

The first blocks provide Poisson Regression Coefficients to predict the number of wine cases sold. The second block Logit Coefficients provides coefficients to predict the LOG ODDS that of a wine case not sold.

Figure 26: Model 3: ZIP Output

```
Call:
zeroinfl(formula = TARGET ~ LabelAppeal + AcidIndex + IMP_Alcohol + IMP_STARS +
    M_STARS | VolatileAcidity + LabelAppeal + AcidIndex + IMP_TotalSulfurDioxide +
    IMP_pH + IMP_Sulphates + IMP_Alcohol + IMP_STARS + M_STARS, data = train.df)

Pearson residuals:
      Min        1Q    Median        3Q       Max
-2.000848 -0.398366 -0.003055  0.382786  6.307682

Count model coefficients (poisson with log link):
             Estimate Std. Error z value             Pr(>|z|)
(Intercept)  1.160645   0.051870  22.376 < 0.0000000000000002 ***
LabelAppeal  0.226027   0.007686  29.407 < 0.0000000000000002 ***
AcidIndex   -0.022432   0.005847  -3.836              0.000125 ***
IMP_Alcohol  0.006575   0.001708   3.850              0.000118 ***
IMP_STARS    0.119220   0.007705  15.472 < 0.0000000000000002 ***
M_STARS     -0.124404   0.022343  -5.568          0.0000000258 ***

Zero-inflation model coefficients (binomial with logit link):
                          Estimate Std. Error z value             Pr(>|z|)
(Intercept)            -5.1897502  0.3861190 -13.441 < 0.0000000000000002 ***
VolatileAcidity         0.1885154  0.0526203   3.583              0.000340 ***
LabelAppeal             1.2245524  0.0694771  17.625 < 0.0000000000000002 ***
AcidIndex               0.4426177  0.0300652  14.722 < 0.0000000000000002 ***
IMP_TotalSulfurDioxide -0.0011424  0.0001977  -5.778         0.00000000755 ***
IMP_pH                  0.2197314  0.0614056   3.578              0.000346 ***
IMP_Sulphates           0.0993321  0.0458211   2.168              0.030172 *
IMP_Alcohol             0.0237497  0.0116908   2.031              0.042205 *
IMP_STARS              -1.5964480  0.0969508 -16.467 < 0.0000000000000002 ***
M_STARS                 3.8688698  0.1191785  32.463 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 24
Log-likelihood: -1.438e+04 on 16 Df
```

Figure 27 shows the equation of the Model 3. This model gives an intercept of 1.16064 for the Poisson Regression and the -5.1897 for the Logit Regression. All the coefficients of the predictor variables shown in Figure 27.

The interpretation of the Poisson Regression parameter is as follows:

The coefficient of IMP_STARS is 0.11922 approx. This model tells us that for each 1 unit increase in IMP_STARS increases the expected number of wine cases sale (TARGET) by a factor of exp(0.11922) = 1.12662 or about {100 * (exp(0.11922) -1)} = 12.66 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 0.22603 approx. This model tells us that for each 1 unit increase in LabelAppeal increases the expected number of wine cases sale (TARGET) by a factor of exp(0.22603) = 1.25361 or about {100 * (exp(0.22603) -1)} = 25.36 percent, while holding all the other variables constant.

The coefficient of AcidIndex is -0.02243 approx. This model tells us that for each 1 unit increase in AcidIndex decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.02243) = 0.97782 or about {100 * (exp(-0.02243) -1)} = 2.22 percent, while holding all the other variables constant.

The coefficient of IMP_Alcohol is 0.00657 approx. This model tells us that for each 1 unit increase in IMP_Alcohol increases the expected number of wine cases sale (TARGET) by a factor of exp(0.00657) = 1.00659 or about {100 * (exp(0.00657) -1)} = 0.66 percent, while holding all the other variables constant.

The interpretation of the Logit Regression parameter is as follows:

The coefficient of VolatileAcidity is 0.18852 approx. This model tells us that for each 1 unit increase in VolatileAcidity, the log of the odds ratio for the response variable i.e. TARGET would be in the “Certain Zero” group would increase by a factor of exp(0.18852) = 1.20746 or about {100 * (exp(0.18852) -1)} = 20.75 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 1.22455 approx. This model tells us that for each 1 unit increase in LabelAppeal, the log of the odds ratio for the response variable i.e. TARGET would be in the “Certain Zero” group would increase by a factor of exp(1.22455) = 3.40264 or about {100 * (exp(1.22455) -1)} = 240.26 percent, while holding all the other variables constant.

The coefficient of IMP_TotalSulfurDioxide is -0.00114 approx. This model tells us that for each 1 unit increase in IMP_TotalSulfurDioxide, the log of the odds ratio for the response variable i.e. TARGET would be in the “Certain Zero” group would decrease by a factor of exp(-0.00114) = 0.99886 or about {100 * (exp(-0.00114) -1)} = 0.11 percent, while holding all the other variables constant.

Similarly, we would interpret all the remaining variables in the zero-inflated model. Figure 28 provides all the values.

Figure 27: Model 3: Equation

Poisson Regression Coefficient Equation

TARGET = (1.160645) +
LabelAppeal * (0.226027) +
AcidIndex * (-0.022432) +
IMP_Alcohol * (0.006575) +
IMP_STARS * (0.119220) +
M_STARS * (-0.124404)

Logit Regression Coefficient Equation

TARGET = (-5.1897502) +
VolatileAcidity * (0.1885154) +
LabelAppeal * (1.2245524) +
AcidIndex * (0.4426177) +
IMP_TotalSulfurDioxide * (-0.0011424) +
IMP_pH * (0.2197314) +
IMP_Sulphates * (0.0993321) +
IMP_Alcohol * (0.0237497) +
IMP_STARS * (-1.5964480) +
M_STARS * (3.8688698)

Figure 28 shows the coefficients, expected count, and percentage values of the Model 3.

Figure 28: Model 3: ZIP Coefficients – Poisson and Logistic

Poisson Regression Coefficient

	Coefficients	Count	Percent
(Intercept)	1.160645046	3.1919916	219.1991594
LabelAppeal	0.226026730	1.2536092	25.3609174
AcidIndex	-0.022432031	0.9778177	-2.2182304
IMP_Alcohol	0.006575282	1.0065969	0.6596946
IMP_STARS	0.119219839	1.1266176	12.6617566
M_STARS	-0.124404284	0.8830228	-11.6977223

Logit Regression Coefficient

	Coefficients	Odds Ratio - Zero	Percent
(Intercept)	-5.189750159	0.005573399	-99.4426601
VolatileAcidity	0.188515373	1.207455645	20.7455645
LabelAppeal	1.224552382	3.402642659	240.2642659
AcidIndex	0.442617744	1.556777134	55.6777134
IMP_TotalSulfurDioxide	-0.001142382	0.998858271	-0.1141729
IMP_pH	0.219731376	1.245742050	24.5742050
IMP_Sulphates	0.099332139	1.104433064	10.4433064
IMP_Alcohol	0.023749682	1.024033951	2.4033951
IMP_STARS	-1.596447981	0.202614933	-79.7385067
M_STARS	3.868869755	47.888230034	4688.8230034

Figure 29 shows the statistics of the Model 3. In this model, we see that the Mean Square Error (MSE) and Mean Absolute Error (MAE) for the test data (i.e. out of sample) is higher than the training data (i.e. in-sample) but not by much as it was in Model 1 and Model 2. This means that this model is slightly overfitting. The model tells us that average error in the prediction is about 1 wine case i.e. MAE = 1.00 for test data. The MAE of this model seems to be fairly close to Model 1.

Figure 29: Model 3: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_3	ZIP	28795.32	1.639497	1.710035	0.970132	1.002513

Section 3.4: Model 4 – Zero Inflated Negative Binomial

Figure 30 shows the summary output of the Model 4. Model 4 is created using Negative Binomial distribution. We started with a simple model with one variable and then added additional variable as they add value to the model. In the figure below, the first block of output in the model call contains Negative Binomial regression coefficients for each of the variables along with the standard errors, z-scores, and p-value for the coefficients. The second block corresponds to the inflation model. Since we know that our data contains excess zeros i.e. zero inflation, this block includes logit coefficients for predicting excess zeros along with their standard errors, z-scores, and p-values.

The first blocks provide Negative Binomial Regression Coefficients to predict the number of wine cases sold. The second block Logit Coefficients provides coefficients to predict the LOG ODDS that of a wine case not sold.

Figure 30: Model 4: ZINB Output

```
Call:
zeroinfl(formula = TARGET ~ LabelAppeal + AcidIndex + IMP_Alcohol + VolatileAcidity + IMP_STARS +
    M_STARS | VolatileAcidity + LabelAppeal + AcidIndex + IMP_pH + IMP_STARS + M_STARS,
    data = train.df, dist = "negbin", EM = TRUE)

Pearson residuals:
    Min      1Q  Median      3Q     Max
-1.9402 -0.4066 -0.0023  0.3847  6.1019

Count model coefficients (negbin with log link):
                Estimate Std. Error z value             Pr(>|z|)
(Intercept)     1.168252   0.051898  22.511 < 0.0000000000000002 ***
LabelAppeal     0.226200   0.007685  29.434 < 0.0000000000000002 ***
AcidIndex      -0.022406   0.005853  -3.828             0.000129 ***
IMP_Alcohol     0.006264   0.001701   3.683             0.000230 ***
VolatileAcidity -0.015752   0.008028  -1.962             0.049751 *
IMP_STARS       0.119052   0.007714  15.433 < 0.0000000000000002 ***
M_STARS        -0.122950   0.022330  -5.506          0.0000000367 ***
Log(theta)     12.275104   4.528173   2.711             0.006712 **

Zero-inflation model coefficients (binomial with logit link):
                Estimate Std. Error z value             Pr(>|z|)
(Intercept)     -5.10659    0.36225 -14.097 < 0.0000000000000002 ***
VolatileAcidity  0.19313    0.05254   3.676             0.000237 ***
LabelAppeal      1.22644    0.06908  17.753 < 0.0000000000000002 ***
AcidIndex        0.45095    0.03004  15.010 < 0.0000000000000002 ***
IMP_pH           0.22941    0.06106   3.757             0.000172 ***
IMP_STARS       -1.59895    0.09685 -16.509 < 0.0000000000000002 ***
M_STARS          3.86315    0.11862  32.567 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Theta = 214293.9341
Number of iterations in BFGS optimization: 1
Log-likelihood: -1.44e+04 on 15 Df
```

Figure 31 shows the equation of the Model 3. This model gives an intercept of 1.16830 for the Negative Binomial Regression and the -5.10487 for the Logit Regression. All the coefficients of the predictor variables shown in Figure 31.

The interpretation of the Negative Binomial Regression parameter is as follows:

The coefficient of IMP_STARS is 0.11905 approx. This model tells us that for each 1 unit increase in IMP_STARS increases the expected number of wine cases sale (TARGET) by a factor of exp(0.11905) = 1.12642 or about {100 * (exp(0.11905) -1)} = 12.64 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 0.22619 approx. This model tells us that for each 1 unit increase in LabelAppeal increases the expected number of wine cases sale (TARGET) by a factor of exp(0.22619) = 1.25382 or about {100 * (exp(0.22619) -1)} = 25.38 percent, while holding all the other variables constant.

The coefficient of AcidIndex is -0.02241 approx. This model tells us that for each 1 unit increase in AcidIndex decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.02241) = 0.97784 or about {100 * (exp(-0.02241) -1)} = 2.22 percent, while holding all the other variables constant.

The coefficient of IMP_Alcohol is 0.00626 approx. This model tells us that for each 1 unit increase in IMP_Alcohol increases the expected number of wine cases sale (TARGET) by a factor of exp(0.00626) = 1.00628 or about {100 * (exp(0.00657) -1)} = 0.63 percent, while holding all the other variables constant.

The coefficient of VolatileAcidity is -0.01575 approx. This model tells us that for each 1 unit increase in VolatileAcidity decreases the expected number of wine cases sale (TARGET) by a factor of exp(-0.01575) = 0.98437 or about {100 * (exp(-0.01575) -1)} = 1.56 percent, while holding all the other variables constant.

The interpretation of the Logit Regression parameter is as follows:

The coefficient of VolatileAcidity is 0.19313 approx. This model tells us that for each 1 unit increase in VolatileAcidity, the log of the odds ratio for the response variable i.e. TARGET would be in the “Certain Zero” group would increase by a factor of exp(0.19313) = 1.21304 or about {100 * (exp(0.19313) -1)} = 21.30 percent, while holding all the other variables constant.

The coefficient of LabelAppeal is 1.22644 approx. This model tells us that for each 1 unit increase in LabelAppeal, the log of the odds ratio for the response variable i.e. TARGET would be in the “Certain Zero” group would increase by a factor of exp(1.22644) = 3.40908 or about {100 * (exp(1.22644) -1)} = 240.90 percent, while holding all the other variables constant.

The coefficient of AcidIndex is 0.45095 approx. This model tells us that for each 1 unit increase in AcidIndex, the log of the odds ratio for the response variable i.e. TARGET would be in the "Certain Zero" group would increase by a factor of exp(0.45095) = 1.56980 or about {100 * (exp(0.45095) -1)} = 56.97 percent, while holding all the other variables constant.

The coefficient of IMP_pH is 0.22941 approx. This model tells us that for each 1 unit increase in IMP_pH, the log of the odds ratio for the response variable i.e. TARGET would be in the "Certain Zero" group would increase by a factor of exp(0.22941) = 1.25786 or about {100 * (exp(0.22941) -1)} = 25.78 percent, while holding all the other variables constant.

The coefficient of IMP_STARS is -1.59895 approx. This model tells us that for each 1 unit increase in IMP_STARS, the log of the odds ratio for the response variable i.e. TARGET would be in the "Certain Zero" group would decrease by a factor of exp(-1.59895) = 0.20211 or about {100 * (exp(-1.59895) -1)} = 79.78 percent, while holding all the other variables constant.

Figure 31: Model 4: Equation

Negative Binomial Regression Coefficient Equation

```
TARGET =  (1.168309)       +
          LabelAppeal      *  (0.226210)      +
          AcidIndex        *  (-0.022410)     +
          IMP_Alcohol      *  (0.006262)      +
          VolatileAcidity  *  (-0.015760)     +
          IMP_STARS        *  (0.119048)      +
          M_STARS          *  (-0.122895)
```

Logit Regression Coefficient Equation

```
TARGET =  (-5.10487)                          +
          VolatileAcidity      * (0.19299)    +
          LabelAppeal          * (1.22646)    +
          AcidIndex            * (0.45084)    +
          IMP_pH               * (0.22918)    +
          IMP_STARS            * (-1.59889)   +
          M_STARS              * (3.86298)
```

Figure 32 shows the coefficients, expected count, and percentage values of the Model 4.

Figure 32: Model 4: ZINB Coefficients – Negative Binomial and Logistic

Negative Binomial Regression Coefficient

	coefficients	Count	Percent
(Intercept)	1.168251822	3.2163649	221.6364943
LabelAppeal	0.226199690	1.2538260	25.3826016
AcidIndex	-0.022406111	0.9778430	-2.2156958
IMP_Alcohol	0.006264055	1.0062837	0.6283715
VolatileAcidity	-0.015752098	0.9843713	-1.5628682
IMP_STARS	0.119052311	1.1264288	12.6428841
M_STARS	-0.122949832	0.8843080	-11.5691974

Logit Regression Coefficient

	Coefficients	Odds Ratio - Zero	Percent
(Intercept)	-5.1065872	0.006056718	-99.39433
VolatileAcidity	0.1931264	1.213036066	21.30361
LabelAppeal	1.2264433	3.409083011	240.90830
AcidIndex	0.4509471	1.569798207	56.97982
IMP_pH	0.2294104	1.257858213	25.78582
IMP_STARS	-1.5989509	0.202108443	-79.78916
M_STARS	3.8631537	47.615280727	4661.52807

Figure 33 shows the statistics of the Model 4. In this model, we see that the Mean Square Error (MSE) and Mean Absolute Error (MAE) for the test data (i.e. out of sample) is again higher than the training data (i.e. in-sample) but not by much as it was in Model 1 and Model 2. This means that this model is slightly overfitting. The model tells us that average error in the prediction is about 1 wine case i.e. MAE = 1.00 for test data. The MAE of this model seems to be fairly close to the rest of the models.

Figure 33: Model 4: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_4	ZIP - Neg. Bion.	28833.74	1.648852	1.722139	0.974904	1.007088

Section 3.5: Model 5 – Linear Regression

Figure 34 shows the output of the Model 5. Residuals in Figure 34 are essentially the difference between the actual observed response values and the response values that the model predicted. The Residuals section of the model help us assess how well the model fits the data, we should look for a symmetrical distribution across these points on the mean value zero (0). In our case, we can see that the distribution of the residuals appears to be symmetrical since the Median value i.e. 0.0583 is very close to zero (0).

The coefficients of each parameter are behaving correctly (as per the theoretical effects in Data Dictionary). For instance, while holding the other predictors constant, if there is one (1) unit increase in STARS, the number of wine cases sold increase by 0.6633. Similarly, if a VolatileAcidity increase by one (1) unit, the number of wine cases sold decrease by 0.0931.

*** Significance stars in coefficients in Figure 34 shows that the predictor variables are statistically significant and it's unlikely that no relationship exists between the TARGET and predictor variables.

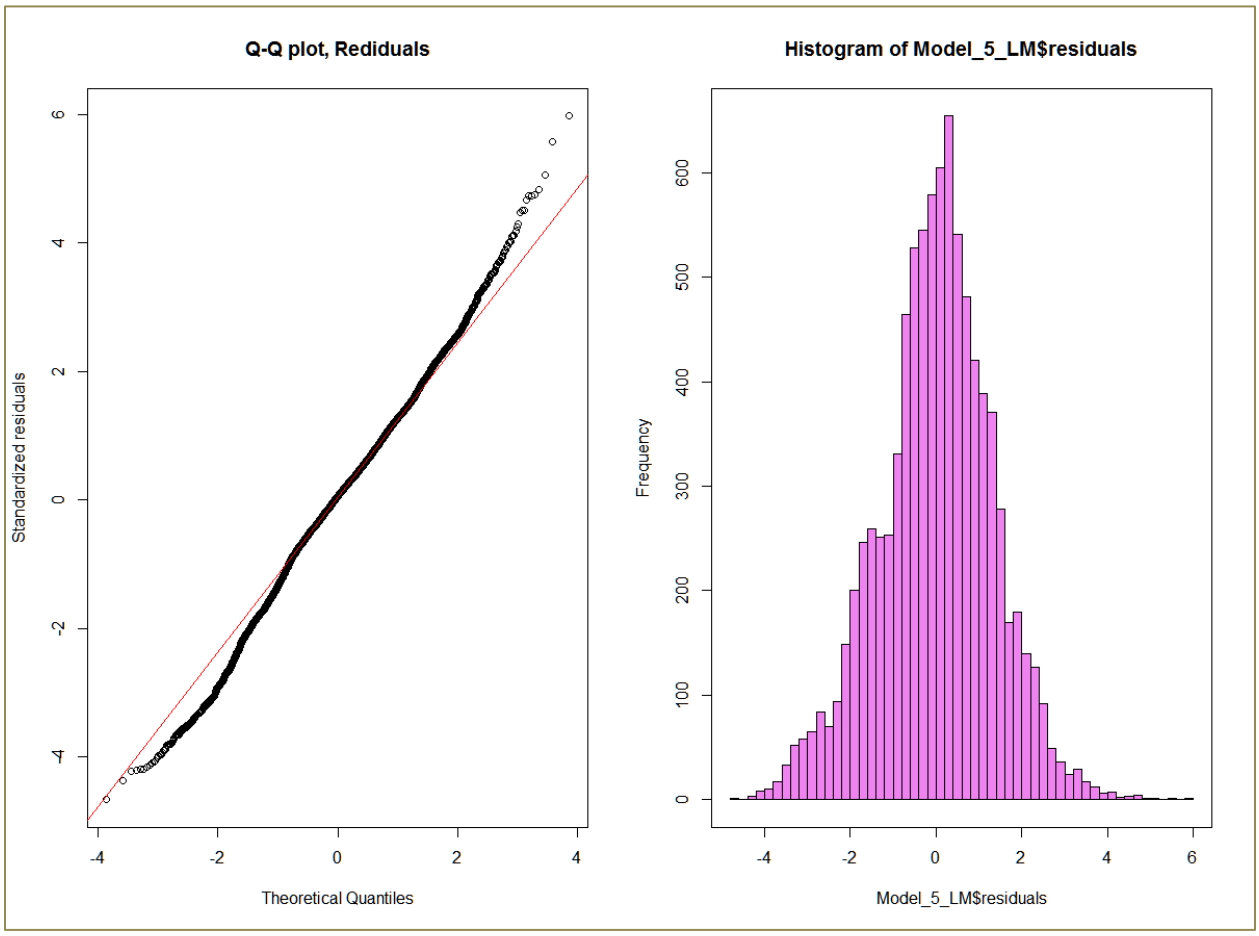
Figure 34: Model 5: Linear Regression Output

```
Call:
lm(formula = TARGET ~ IMP_STARS + M_STARS + LabelAppeal + AcidIndex +
    VolatileAcidity + IMP_Chlorides + IMP_TotalSulfurDioxide +
    IMP_Alcohol + M_Alcohol, data = train.df)

Residuals:
    Min      1Q  Median      3Q     Max
-4.6683 -0.7753  0.0583  0.8495  5.9698

Coefficients:
                        Estimate  Std. Error t value             Pr(>|t|)
(Intercept)           3.83150118  0.10436035  36.714 < 0.0000000000000002 ***
IMP_STARS             0.66329482  0.01902108  34.872 < 0.0000000000000002 ***
M_STARS              -1.97942935  0.03339059 -59.281 < 0.0000000000000002 ***
LabelAppeal           0.42435045  0.01754008  24.193 < 0.0000000000000002 ***
AcidIndex            -0.21814047  0.01080926 -20.181 < 0.0000000000000002 ***
VolatileAcidity      -0.09310473  0.01799780  -5.173          0.000000235 ***
IMP_Chlorides        -0.12267473  0.04527939  -2.709              0.00676 **
IMP_TotalSulfurDioxide  0.00022616  0.00006764   3.344              0.00083 ***
IMP_Alcohol           0.01328483  0.00388012   3.424              0.00062 ***
M_Alcohol             0.08184780  0.06390378   1.281              0.20030
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.333 on 8931 degrees of freedom
Multiple R-squared:  0.5169,    Adjusted R-squared:  0.5164
F-statistic:  1062 on 9 and 8931 DF,  p-value: < 0.00000000000000022
```

In Figure 34, we also look at the Adjusted R-Squared statistic to measure how well our model is fitting the actual data. The R-Squared is a measure of how close the data are to the fitted regression line. In general, the higher the R-Squared, the better the model fits the data. The Adjusted R-Squared we got is **0.5164** which is roughly 51.64% (approx.) of the variance found in the response variable (TARGET) can be explained by the predictor variables.

Using Figure 35, we assess the Goodness-Of-Fit (GOF) of the model. Figure 35 shows the distribution using Q-Q plot and histogram of residuals and we see slight evidence of non-normality towards the upper and lower end. The histogram shows that the residuals form a bell-shaped which usually present a normally distributed. Hence, our model pass the one of the criteria for GOF.

Figure 35: Model 5: GOF - Q-Q plot and Histogram of Residuals



Another assumption for Linear Regression Model is to pass the homoscedasticity assumption. This means that the variance of the error term (residuals) is constant for all combination of independent (predictor) variables. Figure 36 shows the structure of residuals against each predictor variables using scatterplot. We notice that there's no structure in these plots. Hence, we validate the homoscedasticity assumption.

Figure 36: Model 5: GOF - Scatterplot of Residuals and Predictor Variables

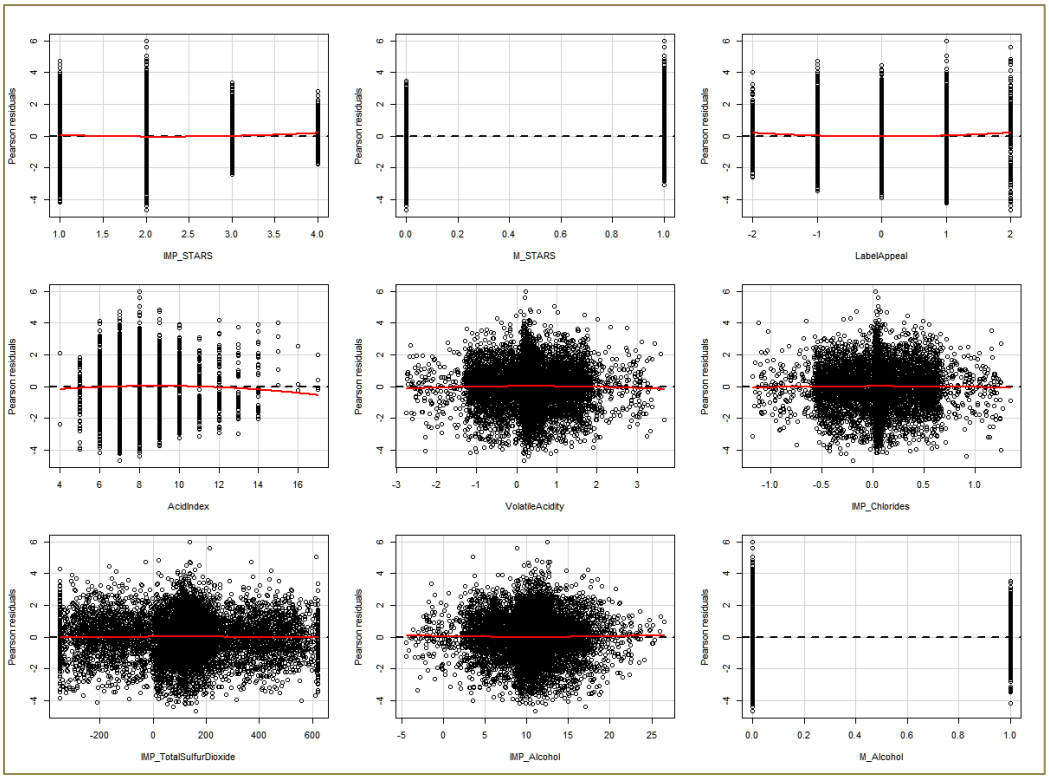


Figure 36 shows the statistics of the Model 5. In this model, we see that the Mean Square Error (MSE) and Mean Absolute Error (MAE) for the test data (i.e. out of sample) is again higher than the training data (i.e. in-sample) but not by much as it was in Model 1 and Model 2. This means that this model is also slightly overfitting. The model tells us that average error in the prediction is about 1 wine case i.e. MAE = 1.07 for test data.

Figure 36 shows the very interesting fact here. Even though the data did not follow the normal distribution, the model prediction was very close to other models with very similar error rate. Honestly, I did not expect it to be so close to other models. Accepting this model as my final model would not been an issue. However, I wouldn't use this model because I do not know what this model did for zero values.

Figure 37: Model 5: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_5	Linear Regression	30527.42	1.775320	1.874208	1.035863	1.073187

Section 4: Model Comparison and Selection

Figure 38 shows variables used in each of our five models i.e. Model 1, Model 2, Model 3, Model 4 and Model 5. Any cell that is grayed out means that variable are not used in that model.

Figure 38: Model Selection: Model 1, Model 2, Model 3, Model 4, Model 5

Model 1 - Poisson	Model 2 - Neg. Binomial	Model 3 - ZIP -Count	Model 3 - ZIP - Zero	Model 4 - ZINB - Count	Model 4 - ZINB - Zero	Model 5 - Linear
AcidIndex	AcidIndex	AcidIndex	AcidIndex	AcidIndex	AcidIndex	AcidIndex
IMP_Alcohol		IMP_Alcohol	IMP_Alcohol	IMP_Alcohol		IMP_Alcohol
IMP_Chlorides						IMP_Chlorides
			IMP_pH		IMP_pH	
IMP_STARS	IMP_STARS	IMP_STARS	IMP_STARS	IMP_STARS	IMP_STARS	IMP_STARS
			IMP_Sulphates			
IMP_TotalSulfurDioxide			IMP_TotalSulfurDioxide			IMP_TotalSulfurDioxide
LabelAppeal	LabelAppeal	LabelAppeal	LabelAppeal	LabelAppeal	LabelAppeal	LabelAppeal
M_STARS	M_STARS	M_STARS	M_STARS	M_STARS	M_STARS	M_STARS
VolatileAcidity	VolatileAcidity		VolatileAcidity	VolatileAcidity	VolatileAcidity	VolatileAcidity
						M_Alcohol

Figure 39 shows the statistics of all five models. We notice that Model 1 and Model 2 has beaten the other models in the training dataset. However, they did not perform poorly in the test data set. Model 3 and Model are fairly close with their stats. Out of these models, I would select Model 3 as my final model because both MSE and MAE is lowest in the test data. Also, all the variables were intuitive and statistically significant.

Figure 39: Model Selection: Statistics

Models	Model Type	AIC	MSE - Train data	MSE - Test Data	MAE - Train Data	MAE - Test Data
Model_1	Poisson	31980.77	0.573410	1.796864	0.512895	1.056042
Model_2	Neg. Binomial	31992.47	0.575165	1.802524	0.513362	1.058692
Model_3	ZIP	28795.32	1.639497	1.710035	0.970132	1.002513
Model_4	ZIP - Neg. Bion.	28833.74	1.648852	1.722139	0.974904	1.007088
Model_5	Linear Regression	30527.42	1.775320	1.874208	1.035863	1.073187

Section 5: Model Testing and Scoring

Next, we use our Model 2 to score the Wine Test data file ("wine_test data file"). Prior to running our model, we will clean the test data in a similar way as we cleaned our training dataset. We will fix missing values as well outliers in the data set. Note, a separate protocol document will accompany the stand-alone R program. This document will list the directions on how to import file and acquire results.

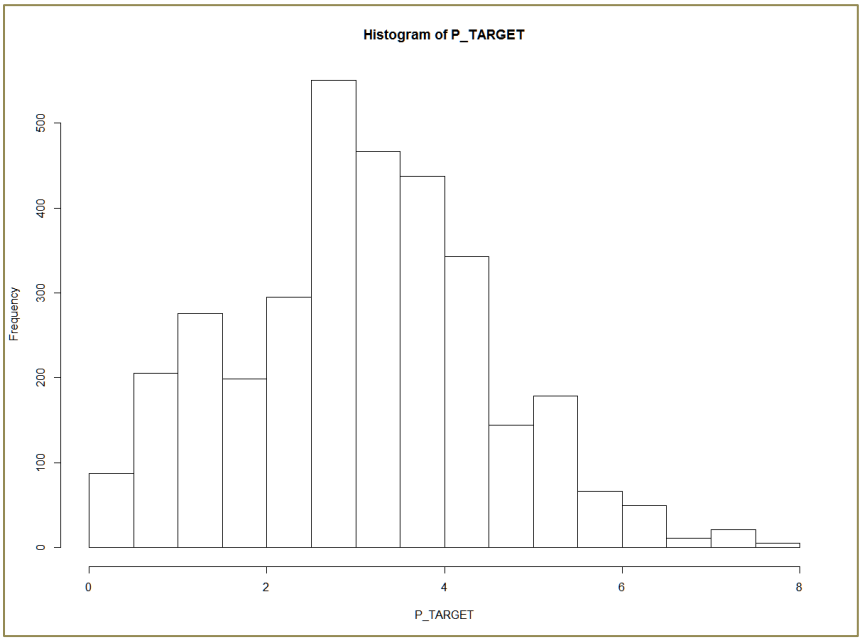
Figure 40 shows the summary statistics of the predicted probability and amount of payouts. We notice that we have all the observations i.e. 3335 observations, and there are no missing values. Our model predicted a minimum of 0.011 i.e. 0 wine cases (after rounding) and maximum of 7.84 i.e. 8 wine cases (after rounding). In my opinion, the results seem to be OK.

Figure 40: Wine Test Data Result Stats

Variable	nobs	NAs	Minimum	Maximum	1. Quartile	3. Quartile	Mean	Median
P_TARGET	3335	0	0.01193	7.840007	2.0935	3.967623	3.061564	3.037022

Figure 41 shows the distribution of our predicted values. We do not see any abnormality with this distribution. Therefore, we will accept this score data result and use it as our final output.

Figure 41: Histogram: Wine Test Data Predicted Values



Conclusion:

In conclusion, I would like to state that we created total five (5) models for this project i.e. Model 1 - Poisson, Model 2 - Negative Binomial, Model 3 - Zero Inflated Poisson, Model 4 – Zero Inflated Negative Binomial, and Model 5 – Linear Regression. Model 3, Model 4 and Model 5 were fairly close with the metrics when ran against the 30% of the data set i.e. test data (actual data was split into train and test). Even though Model 1 and Model 2 performed really well in the train data. However, they performed horribly in the test data. Also, I was surprised to Linear Regression Model to predict pretty close to other models. However, we picked a model that was logically sound as well provided us with best metrics i.e. Model 3 – ZIP model.

We are confident that Model 3 – ZIP is suitable for predicting the number of wine cases based upon the wine characteristics so that the wine manufacturer can adjust their wine offering to maximize sales.

Appendix I: Model Development R Code

```
#---------------------------------------------------------------------
# Sales Prediction and Forecasting - Wine Sales
# Singh, Gurjeet
#---------------------------------------------------------------------

library(readr)
library(car)
library(fBasics)
library(ggplot2)
library(corrplot)
library(plyr)
library(gmodels)
library(MASS)
library(gridExtra)

library(rpart)
library(rpart.plot)
library(RGtk2)
library(rattle)
library(RColorBrewer)
library(pscl)

options(scipen = 999)
#---------------------------------------------------------------------------
---
## 1 - DATA EXPLORATION
#---------------------------------------------------------------------------
---

wine <- read_csv(paste0("~/Desktop/Desktop_Folders/Gurjeet/Master_Degree",
                        "/Course/Predict 411 Section 55/Unit 3/",
                        "b.Assignments/b.1.Wine_Sales_Project/",
                        "SinghGurjeet_Unit3_Project/wine.csv"))


wine.Data <- wine
colnames(wine.Data)[1] <- "INDEX"

#Understand the stats and summary
str(wine.Data)
summary(wine.Data)

#reviewing the stats
View(t(basicStats(wine.Data[sapply(wine.Data,is.numeric)])))
      # View(char_stats)

WineTreeSTARS <- rpart(STARS ~
                         AcidIndex
                       +Alcohol
                       +Chlorides
                       +CitricAcid
                       +Density
                       +FixedAcidity
                       +FreeSulfurDioxide
```

```
                        +LabelAppeal
                        +ResidualSugar
                        +Sulphates
                        +TotalSulfurDioxide
                        +VolatileAcidity
                        +pH, data = wine.Data, method = 'class')

fancyRpartPlot(WineTreeSTARS)

#check the mean and variance of TARGET
mean(wine.Data[which(wine.Data$TARGET >0),"TARGET"])
var(wine.Data[which(wine.Data$TARGET >0),"TARGET"])

#---------------------------------------
#clean missing values with median values
#---------------------------------------

summary(wine.Data)

##clean missing values with median values
wine.Data$IMP_ResidualSugar <- ifelse(is.na(wine.Data$ResidualSugar),
                                      3.9,
                                      wine.Data$ResidualSugar)
wine.Data$M_ResidualSugar <- ifelse(is.na(wine.Data$ResidualSugar),
                                    1, 0)

wine.Data$IMP_Chlorides <-ifelse(is.na(wine.Data$Chlorides),
                                      0.05,
                                      wine.Data$Chlorides)
wine.Data$M_Chlorides <- ifelse(is.na(wine.Data$Chlorides),
                                    1, 0)


wine.Data$IMP_FreeSulfurDioxide <-ifelse(is.na(wine.Data$FreeSulfurDioxide),
                                         30,
                                         wine.Data$FreeSulfurDioxide)
wine.Data$M_FreeSulfurDioxide <- ifelse(is.na(wine.Data$FreeSulfurDioxide),
                                        1, 0)



wine.Data$IMP_TotalSulfurDioxide <-
ifelse(is.na(wine.Data$TotalSulfurDioxide),
                                           123,
                                           wine.Data$TotalSulfurDioxide)

wine.Data$M_TotalSulfurDioxide <- ifelse(is.na(wine.Data$TotalSulfurDioxide),
                                         1, 0)


wine.Data$IMP_pH <-ifelse(is.na(wine.Data$pH),
                                           3.2,
                                           wine.Data$pH)

wine.Data$M_pH <- ifelse(is.na(wine.Data$pH),
                                         1, 0)
```

```
wine.Data$IMP_Sulphates <-ifelse(is.na(wine.Data$Sulphates),
                                          0.50,
                                          wine.Data$Sulphates)

wine.Data$M_Sulphates <- ifelse(is.na(wine.Data$Sulphates),
                                          1, 0)


wine.Data$IMP_Alcohol <-ifelse(is.na(wine.Data$Alcohol),
                                          10.40,
                                          wine.Data$Alcohol)

wine.Data$M_Alcohol <- ifelse(is.na(wine.Data$Alcohol),
                                          1, 0)



wine.Data$IMP_STARS <-ifelse(is.na(wine.Data$STARS) & (wine.Data$LabelAppeal
< -0.5), 1,
                        ifelse(is.na(wine.Data$STARS) &
(wine.Data$LabelAppeal >= -0.5),2,
                               wine.Data$STARS))

wine.Data$M_STARS <- ifelse(is.na(wine.Data$STARS),
                                          1, 0)


#options(scipen = 999)
View(t(basicStats(wine.Data[sapply(wine.Data,is.numeric)])))

#-----------------------------------------
##Histograms - Exploration
#-----------------------------------------

#---------------
# TARGET
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=TARGET))+
          geom_histogram(fill = "deepskyblue3",
                         colour="black", binwidth = 0.5) +
          stat_bin(binwidth=1, geom="text",
                      aes(label=..count..), vjust=-1) +
          labs(title = "Histogram of TARGET",
                      x = "TARGET", y = "Frequency" ) +
          scale_x_continuous(breaks = seq(0, 10,by = 1))+
          theme_bw()

plot2 <- ggplot(wine.Data,
                mapping = aes(x=TARGET,
                         fill=as.factor(STARS),
                         group = as.factor(STARS)))+
          geom_bar()+
          theme_bw() +
          labs(title = "Histogram of TARGET by STARS",
               x = "TARGET", y = "Frequency") +
          scale_fill_discrete(name="STARS") +
        scale_x_continuous(breaks = seq(0, 10,by = 1))
```

```
grid.arrange(plot1, plot2, nrow = 1)

#---------------
# FixedAcidity
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=FixedAcidity))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of FixedAcidity",
       x = "FixedAcidity",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-100, 100,by = 10))+
  theme_bw()

#---------------
# VolatileAcidity
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=VolatileAcidity))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of VolatileAcidity",
       x = "VolatileAcidity",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-5, 5,by = 1))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)

#---------------
# CitricAcid
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=CitricAcid))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of CitricAcid",
       x = "CitricAcid",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-6, 6,by = 1))+
  theme_bw()


#---------------
# IMP_ResidualSugar
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=IMP_ResidualSugar))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_ResidualSugar",
       x = "IMP_ResidualSugar",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-200, 200,by = 20))+
  theme_bw()


grid.arrange(plot1, plot2, nrow = 1)
#---------------
```

```
# IMP_Chlorides
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=IMP_Chlorides))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_Chlorides",
       x = "IMP_Chlorides",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-5, 5,by = 0.5))+
  theme_bw()

#---------------
# IMP_FreeSulfurDioxide
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=IMP_FreeSulfurDioxide))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_FreeSulfurDioxide",
       x = "IMP_FreeSulfurDioxide",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-1000, 1000,by = 100))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)
#---------------
# IMP_TotalSulfurDioxide
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=IMP_TotalSulfurDioxide))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_TotalSulfurDioxide",
       x = "IMP_TotalSulfurDioxide",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-1100, 1100,by = 300))+
  theme_bw()

#---------------
# Density
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=Density))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of Density",
       x = "Density",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-3, 3,by = 0.02))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)
#---------------
# IMP_pH
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=IMP_pH))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_pH",
       x = "IMP_pH",
```

```
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-10, 10,by = 0.5))+
  theme_bw()

#---------------
# IMP_Sulphates
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=IMP_Sulphates))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_Sulphates",
       x = "IMP_Sulphates",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-10, 10,by = 0.5))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)
#---------------
# IMP_Alcohol
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=IMP_Alcohol))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_Alcohol",
       x = "IMP_Alcohol",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-50, 50,by = 2))+
  theme_bw()


#---------------
# LabelAppeal
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=LabelAppeal))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black", binwidth = 0.5) +
  labs(title = "Histogram of LabelAppeal",
       x = "LabelAppeal",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-5, 5,by = 1))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)
#---------------
# AcidIndex
#---------------
plot1 <- ggplot(wine.Data, mapping = aes(x=AcidIndex))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black", binwidth = 0.5) +
  labs(title = "Histogram of AcidIndex",
       x = "AcidIndex",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-50, 50,by = 1))+
  theme_bw()

#---------------
# IMP_STARS
```

```
#---------------
plot2 <- ggplot(wine.Data, mapping = aes(x=IMP_STARS))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black", binwidth = 0.5) +
  labs(title = "Histogram of IMP_STARS",
       x = "IMP_STARS",
       y = "Frequency" ) +
  scale_x_continuous(breaks = seq(-5, 5,by = 1))+
  theme_bw()

grid.arrange(plot1, plot2, nrow = 1)


#---------------------------------------------
# Q-Q Plots below
#---------------------------------------------

par(mfrow=c(2,2))
#---------------
# FixedAcidity
#---------------

with(wine.Data,
     qqPlot(FixedAcidity,
            main="QQ-Plot FixedAcidity",
            col = "gray0"));

#---------------
# VolatileAcidity
#---------------

with(wine.Data,
     qqPlot(VolatileAcidity,
            main="QQ-Plot VolatileAcidity",
            col = "gray0"));


#---------------
# CitricAcid
#---------------

with(wine.Data,
     qqPlot(CitricAcid,
            main="QQ-Plot CitricAcid",
            col = "gray0"));


#---------------
# IMP_ResidualSugar
#---------------


with(wine.Data,
     qqPlot(IMP_ResidualSugar,
            main="QQ-Plot IMP_ResidualSugar",
            col = "gray0"));
```

```
#---------------
# IMP_Chlorides
#---------------

with(wine.Data,
     qqPlot(IMP_Chlorides,
            main="QQ-Plot IMP_Chlorides",
            col = "gray0"));

#---------------
# IMP_FreeSulfurDioxide
#---------------

with(wine.Data,
     qqPlot(IMP_FreeSulfurDioxide,
            main="QQ-Plot IMP_FreeSulfurDioxide",
            col = "gray0"));

#---------------
# IMP_TotalSulfurDioxide
#---------------
with(wine.Data,
     qqPlot(IMP_TotalSulfurDioxide,
            main="QQ-Plot IMP_TotalSulfurDioxide",
            col = "gray0"));


#---------------
# Density
#---------------

with(wine.Data,
     qqPlot(Density,
            main="QQ-Plot Density",
            col = "gray0"));


#---------------
# IMP_pH
#---------------

with(wine.Data,
     qqPlot(IMP_pH,
            main="QQ-Plot IMP_pH",
            col = "gray0"));

#---------------
# IMP_Sulphates
#---------------

with(wine.Data,
     qqPlot(IMP_Sulphates,
            main="QQ-Plot IMP_Sulphates",
            col = "gray0"));

#---------------
```

```
# IMP_Alcohol
#---------------

with(wine.Data,
     qqPlot(IMP_Alcohol,
            main="QQ-Plot IMP_Alcohol",
            col = "gray0"));

#---------------
# LabelAppeal
#---------------

with(wine.Data,
     qqPlot(LabelAppeal,
            main="QQ-Plot LabelAppeal",
            col = "gray0"));

#---------------
# AcidIndex
#---------------

with(wine.Data,
     qqPlot(AcidIndex,
            main="QQ-Plot AcidIndex",
            col = "gray0"));

#---------------
# IMP_STARS
#---------------

with(wine.Data,
     qqPlot(IMP_STARS,
            main="QQ-Plot IMP_STARS",
            col = "gray0"));

par(mfrow=c(1,1))


#-----------------------------------------------------------------------------
--
## 2 - DATA PREPARATION
#-----------------------------------------------------------------------------
--

#------------------------------
## Fix extreme values for:
#     - IMP_TotalSulfurDioxide
#     - IMP_FreeSulfurDioxide
#     - IMP_ResidualSugar

# These have mean/median very low but min and max very high
#------------------------------


wine.Data$IMP_TotalSulfurDioxide = ifelse(wine.Data$IMP_TotalSulfurDioxide <
-350, -350,
```

```
ifelse(wine.Data$IMP_TotalSulfurDioxide > 620, 620,

wine.Data$IMP_TotalSulfurDioxide))

wine.Data$IMP_ResidualSugar = ifelse(wine.Data$IMP_ResidualSugar < -50, -50,
                                     ifelse(wine.Data$IMP_ResidualSugar
> 75, 75,

wine.Data$IMP_ResidualSugar))

wine.Data$IMP_FreeSulfurDioxide = ifelse(wine.Data$IMP_FreeSulfurDioxide < -
300, -300,

ifelse(wine.Data$IMP_FreeSulfurDioxide > 350, 350,

wine.Data$IMP_FreeSulfurDioxide))


#---------------------------------------
# Histograms to show new distribution
#---------------------------------------
plot1 <- ggplot(wine.Data, mapping = aes(x=IMP_TotalSulfurDioxide))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
    labs(title = "Histogram of IMP_TotalSulfurDioxide",
       x = "IMP_TotalSulfurDioxide",
       y = "Frequency" ) +
  theme_bw()

plot2 <- ggplot(wine.Data, mapping = aes(x=IMP_ResidualSugar))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
    labs(title = "Histogram of IMP_ResidualSugar",
       x = "IMP_ResidualSugar",
       y = "Frequency" ) +
  theme_bw()

plot3 <- ggplot(wine.Data, mapping = aes(x=IMP_FreeSulfurDioxide))+
  geom_histogram(fill = "deepskyblue3",
                 colour="black") +
  labs(title = "Histogram of IMP_FreeSulfurDioxide",
       x = "IMP_FreeSulfurDioxide",
       y = "Frequency" ) +
  theme_bw()

grid.arrange(plot1, plot2, plot3, nrow = 2)
#--------------------------
##Make a copy of the data ---START FROM HERE
#--------------------------

View(t(basicStats(wine.Data[sapply(wine.Data,is.numeric)])))

# Save the R data frame as an .RData object
saveRDS(wine.Data,file="wine.Data_Fixed.RData" );
```

```
# Read (or reload) the .RData object as an R data frame
wine.Data_fixed <- readRDS(file= "wine.Data_Fixed.RData");

#--------------------------
##creating a drop list
#-------------------------------------------------------------------------------
---
names(wine.Data)


#creating a drop list to remove not required variables.
drop.list <- c('ResidualSugar','Chlorides', 'FreeSulfurDioxide',
               'TotalSulfurDioxide','pH', 'Sulphates','Alcohol','STARS'
               )

#droping the variables
wine.Data_fixed <- wine.Data_fixed[,!(names(wine.Data_fixed) %in% drop.list
)]

names(wine.Data_fixed)
summary(wine.Data_fixed)
View(wine.Data_fixed)

View(t(basicStats(wine.Data_fixed[sapply(wine.Data_fixed,is.numeric)])))
#-----------------------------------------------------------------
#Requirement # Add a train/test flag to split the sample
#-----------------------------------------------------------------


wine.Data_fixed$u <- runif(n=dim(wine.Data_fixed)[1],min=0,max=1);
wine.Data_fixed$train <- ifelse(wine.Data_fixed$u<0.70,1,0);

#Save the R data frame as an .RData object
saveRDS(wine.Data_fixed,file="wine.Data_fixed_train.RData" );

# Check the counts on the train/test split
table(wine.Data_fixed$train)

# Check the train/test split as a percentage of whole
table(wine.Data_fixed$train)/dim(wine.Data_fixed)[1]


#-------------------------------------------------------------------------------
--
## 3- Build Models
#-------------------------------------------------------------------------------
---

options(scipen = 999)

# Read (or reload) the .RData object as an R data frame
wine.Data_fixed_Final <- readRDS(file= "wine.Data_fixed_train.RData");

names(wine.Data_fixed_Final)
View(wine.Data_fixed_Final)

# Create train/test split;
```

```
train.df <- subset(wine.Data_fixed_Final, train==1);
test.df <- subset(wine.Data_fixed_Final, train==0);

View(t(basicStats(wine.Data_fixed_Final[sapply(wine.Data_fixed_Final,is.numer
ic)])))

##Checking Mean and Variance - SHowing Variance over Mean - OverDispersion
mean(wine.Data_fixed_Final[which(wine.Data_fixed_Final$TARGET >0),"TARGET"])
var(wine.Data_fixed_Final[which(wine.Data_fixed_Final$TARGET >0),"TARGET"])

mean(train.df[which(train.df$TARGET >0),"TARGET"])
var(train.df[which(train.df$TARGET >0),"TARGET"])

mean(test.df[which(test.df$TARGET >0),"TARGET"])
var(test.df[which(test.df$TARGET >0),"TARGET"])

# #------------------------------
# ### ##Model_1_POIS - FINAL
# #------------------------------

names(train.df)

# Define the upper model as the FULL model
Model_1_POIS <- glm(TARGET ~ IMP_STARS + M_STARS +
                      LabelAppeal + AcidIndex +
                      VolatileAcidity + IMP_Alcohol +
                      IMP_Chlorides + IMP_TotalSulfurDioxide
                    ,family = poisson
                    ,data = train.df )
summary(Model_1_POIS)

# #------------------------------
# ### ##Model_2_NB - Negative Binomial - FINAL
# #------------------------------

names(train.df)

# Define the upper model as the FULL model
Model_2_NB <- glm.nb(TARGET ~ IMP_STARS + M_STARS + LabelAppeal +
                      AcidIndex + VolatileAcidity
                    ,data = train.df )
summary(Model_2_NB)

# #------------------------------
# ### ##Model_3_ZIP - Zero Inflated POisson - FINAL
# #------------------------------

names(train.df)
# Define the upper model as the FULL model
Model_3_ZIP <- zeroinfl(TARGET ~  LabelAppeal + AcidIndex +
                          IMP_Alcohol +
                          IMP_STARS + M_STARS
                        |  VolatileAcidity +
                          LabelAppeal + AcidIndex +
                          IMP_TotalSulfurDioxide  + IMP_pH +
                          IMP_Sulphates +  IMP_Alcohol +
                          IMP_STARS + M_STARS
```

```
                          ,data = train.df )
summary(Model_3_ZIP)

# #------------------------------
# ### ##Model_4_ZIP_NB - Zero Inflated Negative Binomial Distribution _ FINAL
# #------------------------------

names(train.df)

# Define the upper model as the FULL model
Model_4_ZIP_NB <- zeroinfl(TARGET ~  LabelAppeal + AcidIndex +
                           IMP_Alcohol + VolatileAcidity +
                           IMP_STARS + M_STARS
                         |  VolatileAcidity +
                           LabelAppeal + AcidIndex +
                           IMP_pH +
                           IMP_STARS + M_STARS
                         ,data = train.df, dist="negbin", EM=TRUE )
summary(Model_4_ZIP_NB)

# #------------------------------
# ### ##Model_5_LM
# #------------------------------


# Define the upper model as the FULL model
Model_5_LM <- lm(TARGET ~ IMP_STARS   +  M_STARS +
                    LabelAppeal +  AcidIndex +
                    VolatileAcidity + IMP_Chlorides +
                    IMP_TotalSulfurDioxide + IMP_Alcohol + M_Alcohol
                  ,data = train.df )
summary(Model_5_LM)


#------------------------------
##Model_5_lm - Assessing the Goodness-Of-Fit in OLS Regression
#------------------------------

# Validating the normality assumption:
par(mfrow = c(1,2))
#Creating 2 Q-Q plots to evaluate the distribution of
# SalePrice and L_SalePrice
qqnorm(Model_5_LM$residuals, main = "Q-Q plot, Rediduals",
       xlab="Theoretical Quantiles", col = "black",
       ylab="Standardized residuals",datax=FALSE)

qqline(Model_5_LM$residuals, datax=FALSE, distribution=qnorm,
       probs=c(0.25,0.75),qtype=7, col = "red")

hist(Model_5_LM$residuals, breaks = "FD", col = "violet"); box();
par(mfrow = c(1,1))

#Validating the homoscedasticity assumption (equal variance):
residualPlots(Model_5_LM)
# par(mfrow = c(1,1))
# residualPlot(Model_1_lm)
```

```
# #-------------------------------------------------------------------
# ## Predictive Adequcy
# #-------------------------------------------------------------------

#-----------------------------------
# CHECK FOR Summary Statistic
#-----------------------------------
summary(Model_1_POIS)
summary(Model_2_NB)
summary(Model_3_ZIP)
summary(Model_4_ZIP_NB)
summary(Model_5_LM)

#-----------------------------------
# Convert Coefficients into count and %
#-----------------------------------

#--Model 1
expCoeff_Model_1_POIS<- as.data.frame(cbind(Model_1_POIS$coefficients,
                                            exp(Model_1_POIS$coefficients),
                                            100*(exp(Model_1_POIS$coefficients)-
1)))
names(expCoeff_Model_1_POIS) <- c("Coefficients","Count", "Percent")
View(expCoeff_Model_1_POIS)

#-- Model 2

expCoeff_Model_2_NB <- as.data.frame(cbind(Model_2_NB$coefficients,
                                           exp(Model_2_NB$coefficients),
                                           100*(exp(Model_2_NB$coefficients)-1)))
names(expCoeff_Model_2_NB) <- c("Coefficients","Count", "Percent")
View(expCoeff_Model_2_NB)

#Model 3

  names(Model_3_ZIP)
  #--Count
  Model_3_ZIP$coefficients$count
expCoeff_Model_3_ZIP_Count<-
as.data.frame(cbind(Model_3_ZIP$coefficients$count,
                                            exp(Model_3_ZIP$coefficients$count),

100*(exp(Model_3_ZIP$coefficients$count)-1)))
  names(expCoeff_Model_3_ZIP_Count) <- c("Coefficients","Count", "Percent")
  View(expCoeff_Model_3_ZIP_Count)

  #--Zero
  Model_3_ZIP$coefficients$zero

Odds_Ratio_Model_3_ZIP_Zero <-
as.data.frame(cbind(Model_3_ZIP$coefficients$zero,
                                            exp(Model_3_ZIP$coefficients$zero),

100*(exp(Model_3_ZIP$coefficients$zero)-1)))


    #as.data.frame(exp(Model_3_ZIP$coefficients$zero))
```

```
  names(Odds_Ratio_Model_3_ZIP_Zero) <- c("Coefficients","Odds Ratio - Zero",
"Percent")
  View(Odds_Ratio_Model_3_ZIP_Zero)


#Model 4

  summary(Model_4_ZIP_NB)
  names(Model_4_ZIP_NB)

  #--Count
  Model_4_ZIP_NB$coefficients$count
expCoeff_Model_4_ZIP_NB_Count <-
as.data.frame(cbind(Model_4_ZIP_NB$coefficients$count,

exp(Model_4_ZIP_NB$coefficients$count),

100*(exp(Model_4_ZIP_NB$coefficients$count)-1)))
  names(expCoeff_Model_4_ZIP_NB_Count) <- c("coefficients","Count",
"Percent")
  View(expCoeff_Model_4_ZIP_NB_Count)

  #--Zero
  Model_4_ZIP_NB$coefficients$zero

  Odds_Ratio_Model_4_ZIP_NB_Zero <-
as.data.frame(cbind(Model_4_ZIP_NB$coefficients$zero,

exp(Model_4_ZIP_NB$coefficients$zero),

100*(exp(Model_4_ZIP_NB$coefficients$zero)-1)))

    #as.data.frame(exp(Model_4_ZIP_NB$coefficients$zero))
  names(Odds_Ratio_Model_4_ZIP_NB_Zero) <- c("Coefficients","Odds Ratio -
Zero", "Percent")
  View(Odds_Ratio_Model_4_ZIP_NB_Zero)

#Model 5

  expCoeff_Model_5_LM      <- as.data.frame(cbind(Model_5_LM$coefficients))
  names(expCoeff_Model_5_LM) <- c("Coefficients")
  View(expCoeff_Model_5_LM)


#-----------------------------------
# CHECK FOR AIC Values for Each Model
#-----------------------------------
AIC(Model_1_POIS)
AIC(Model_2_NB)
AIC(Model_3_ZIP)
AIC(Model_4_ZIP_NB)
AIC(Model_5_LM)

#-----------------------------------
# Calculate MSE and MAE Values for Each Model
# for Train and Test
#-----------------------------------
```

```
#-------------------
#Model_1_POIS
#-------------------

#---TRAIN
#MSE
mse.Model_1_POIS <- mean(Model_1_POIS$residuals^2)

##MAE
mae.Model_1_POIS <- mean(abs(Model_1_POIS$residuals))

#---TEST
##-get the predicted/fitted values
Pred.Model_Pois <- predict(Model_1_POIS,
                           newdata = test.df, type = "response")

#MSE
mse.Model_POIS_TEST <- mean((test.df$TARGET - Pred.Model_Pois)^2)
#MAE
mae.Model_POIS_TEST <- mean(abs(test.df$TARGET - Pred.Model_Pois))


#-------------------
#Model_2_NB
#-------------------

#--TRAIN
##MSE
mse.Model_2_NB <- mean(Model_2_NB$residuals^2)
##MAE
mae.Model_2_NB <- mean(abs(Model_2_NB$residuals))

#--TEST

##-get the predicted/fitted values
Pred.Model_NB <- predict(Model_2_NB, newdata = test.df, type = "response")

#MSE
mse.Model_NB_TEST <- mean((test.df$TARGET - Pred.Model_NB)^2)
#MAE
mae.Model_NB_TEST <- mean(abs(test.df$TARGET - Pred.Model_NB))


#-------------------
#Model_3_ZIP
#-------------------

summary(Model_3_ZIP)


#train.df
TEMP_ZIP <- with(train.df ,
                                      (1.160645)      +
                     LabelAppeal      *  (0.226027)      +
                     AcidIndex        *  (-0.022432)     +
                     IMP_Alcohol      *  (0.006575)      +
                     IMP_STARS        *  (0.119220)     +
```

```
                      M_STARS         *   (-0.124404))

P_SCORE_Zip_ALL <- exp(TEMP_ZIP)

TEMP_Zip_Zero <- with(train.df,
                                                (-5.1897502)     +
                        VolatileAcidity         *    (    0.1885154    )        +
                        LabelAppeal             *    (    1.2245524    )        +
                        AcidIndex                 *  (    0.4426177    )        +
                        IMP_TotalSulfurDioxide  *    (-0.0011424 )         +
                        IMP_pH                  *    (    0.2197314    )        +
                        IMP_Sulphates             *  (    0.0993321    )        +
                        IMP_Alcohol             *    (    0.0237497)         +
                        IMP_STARS                 *  (-1.5964480 )         +
                        M_STARS                 *    (    3.8688698    ))

P_SCORE_Zip_ZERO <- exp(TEMP_Zip_Zero)/(1+exp(TEMP_Zip_Zero))

P_SCORE_Zip <- P_SCORE_Zip_ALL * (1 - P_SCORE_Zip_ZERO)

mse.Model_ZIP <- mean((train.df$TARGET - P_SCORE_Zip)^2)
mae.Model_ZIP <- mean(abs(train.df$TARGET - P_SCORE_Zip))


#test.df

TEMP_ZIP_test <- with(test.df ,
                                          (1.160645)      +
                        LabelAppeal     *  (0.226027)      +
                        AcidIndex       *  (-0.022432)     +
                        IMP_Alcohol     *  (0.006575)      +
                        IMP_STARS       *  (0.119220)      +
                        M_STARS         *  (-0.124404))

P_SCORE_Zip_ALL_test <- exp(TEMP_ZIP_test)

TEMP_Zip_Zero_test <- with(test.df,
                                                  (-5.1897502)      +
                             VolatileAcidity        *    (    0.1885154    )
+
                             LabelAppeal            *    (    1.2245524    )
+
                             AcidIndex            *  (    0.4426177    )
+
                             IMP_TotalSulfurDioxide  *  (-0.0011424 )       +
                             IMP_pH                  *  (    0.2197314    )
+
                             IMP_Sulphates           *  (    0.0993321    )
+
                             IMP_Alcohol               *    (    0.0237497)
+
                             IMP_STARS               *  (-1.5964480 )       +
                             M_STARS                   *    (    3.8688698
))

P_SCORE_Zip_ZERO_test <- exp(TEMP_Zip_Zero_test)/(1+exp(TEMP_Zip_Zero_test))
```

```
P_SCORE_Zip_test <- P_SCORE_Zip_ALL_test * (1 - P_SCORE_Zip_ZERO_test)

mse.Model_ZIP_test <- mean((test.df$TARGET - P_SCORE_Zip_test)^2)
mae.Model_ZIP_test <- mean(abs(test.df$TARGET - P_SCORE_Zip_test))


#--------------------
#Model_4_ZIP_NB
#--------------------

#train.df
summary(Model_4_ZIP_NB)
TEMP_ZIP_NB <- with(train.df ,
                                    (1.168309)      +
                    LabelAppeal     *  (0.226210)      +
                    AcidIndex       *  (-0.022410)     +
                    IMP_Alcohol     *  (0.006262)      +
                    VolatileAcidity *  (-0.015760)     +
                    IMP_STARS       *  (0.119048)      +
                    M_STARS         *  (-0.122895))

P_SCORE_Zip_ALL_NB <- exp(TEMP_ZIP_NB)

TEMP_Zip_Zero_NB <- with(train.df,
                                                ( -5.10487)     +
                        VolatileAcidity         *   (   0.19299 )       +
                        LabelAppeal             *   (   1.22646 )       +
                        AcidIndex                 * (   0.45084 )       +
                        IMP_pH                  *   (   0.22918 )       +
                        IMP_STARS                 * (-1.59889   )       +
                        M_STARS                 *   (   3.86298 ))

P_SCORE_Zip_ZERO_NB <- exp(TEMP_Zip_Zero_NB)/(1+exp(TEMP_Zip_Zero_NB))

P_SCORE_Zip_NB <- P_SCORE_Zip_ALL_NB * (1 - P_SCORE_Zip_ZERO_NB)

mse.Model_ZIP_NB <- mean((train.df$TARGET - P_SCORE_Zip_NB)^2)
mae.Model_ZIP_NB <- mean(abs(train.df$TARGET - P_SCORE_Zip_NB))

#test.df

TEMP_ZIP_NB_test <- with(test.df ,
                                       (1.168309)      +
                       LabelAppeal     *  (0.226210)      +
                       AcidIndex       *  (-0.022410)     +
                       IMP_Alcohol     *  (0.006262)      +
                       VolatileAcidity *  (-0.015760)     +
                       IMP_STARS       *  (0.119048)      +
                       M_STARS         *  (-0.122895))

P_SCORE_Zip_ALL_NB_test <- exp(TEMP_ZIP_NB_test)

TEMP_Zip_Zero_NB_test <- with(test.df,
                                                  ( -5.10487)       +
                            VolatileAcidity         *   (   0.19299 )       +
                            LabelAppeal             *   (   1.22646 )       +
                            AcidIndex                 * (   0.45084 )       +
```

```
                              IMP_pH                    * (    0.22918 )       +
                              IMP_STARS                 * (-1.59889    )       +
                              M_STARS                 *   (    3.86298 ))

P_SCORE_Zip_ZERO_NB_test <-
exp(TEMP_Zip_Zero_NB_test)/(1+exp(TEMP_Zip_Zero_NB_test))

P_SCORE_Zip_NB_test <- P_SCORE_Zip_ALL_NB_test * (1 -
P_SCORE_Zip_ZERO_NB_test)

mse.Model_ZIP_NB_test <- mean((test.df$TARGET - P_SCORE_Zip_NB_test)^2)
mae.Model_ZIP_NB_test <- mean(abs(test.df$TARGET - P_SCORE_Zip_NB_test))



#
# #--------------------
# ### ##Model_5_LM
# #--------------------

##MSE
mse.Model_5_LM <- mean(Model_5_LM$residuals^2)

##MAE
mae.Model_5_LM <- mean(abs(Model_5_LM$residuals))



##-get the predicted/fitted values
Pred.Model_LM <- predict(Model_5_LM, newdata = test.df)

#MSE
mse.Model_LM_TEST <- mean((test.df$TARGET - Pred.Model_LM)^2)
#MAE
mae.Model_LM_TEST <- mean(abs(test.df$TARGET - Pred.Model_LM))




#-------------------------------------
# CHECK FOR MSE Values for Each Model
#-------------------------------------

#Model_1
  mse.Model_1_POIS
  mse.Model_POIS_TEST

  mae.Model_1_POIS
  mae.Model_POIS_TEST

#Model_2
  mse.Model_2_NB
  mse.Model_NB_TEST
```

```
  mae.Model_2_NB
  mae.Model_NB_TEST

#Model_3
  mse.Model_ZIP
  mse.Model_ZIP_test

  mae.Model_ZIP
  mae.Model_ZIP_test

#Model_4
  mse.Model_ZIP_NB
  mse.Model_ZIP_NB_test

  mae.Model_ZIP_NB
  mae.Model_ZIP_NB_test

#Model_5
  mse.Model_5_LM
  mse.Model_LM_TEST

  mae.Model_5_LM
  mae.Model_LM_TEST

  #---------------------------------
  # THE END
  #---------------------------------
```

Appendix II: Stand-Alone R Code

```
#---------------------------------------------------------------------
# Sales Prediction and Forecasting - Wine Sales
# Singh, Gurjeet
# Stand-Alone program
#---------------------------------------------------------------------

library(readr)
library(fBasics)

options(scipen = 999)
#---------------------------------------------------------------------------
## 1 - Importing a Test File and check import
#---------------------------------------------------------------------------

View(wine_test)
summary(wine_test)
str(wine_test)
colnames(wine_test)[1] <- "INDEX"

#---------------------------------------------------------------------------
## 2 - DATA PREPARATION
#---------------------------------------------------------------------------

#---------------------------------------
#clean missing values with median values
#---------------------------------------

summary(wine_test)
##clean missing values with median values

wine_test$IMP_ResidualSugar <- ifelse(is.na(wine_test$ResidualSugar),
                                      3.9,
                                      wine_test$ResidualSugar)
wine_test$M_ResidualSugar <- ifelse(is.na(wine_test$ResidualSugar),
                                    1, 0)

wine_test$IMP_Chlorides <-ifelse(is.na(wine_test$Chlorides),
                                 0.05,
                                 wine_test$Chlorides)
wine_test$M_Chlorides <- ifelse(is.na(wine_test$Chlorides),
                                1, 0)


wine_test$IMP_FreeSulfurDioxide <-ifelse(is.na(wine_test$FreeSulfurDioxide),
                                         30,
                                         wine_test$FreeSulfurDioxide)
wine_test$M_FreeSulfurDioxide <- ifelse(is.na(wine_test$FreeSulfurDioxide),
                                        1, 0)



wine_test$IMP_TotalSulfurDioxide <-
ifelse(is.na(wine_test$TotalSulfurDioxide),
                                          123,
                                          wine_test$TotalSulfurDioxide)
```

```
wine_test$M_TotalSulfurDioxide <- ifelse(is.na(wine_test$TotalSulfurDioxide),
                                         1, 0)


wine_test$IMP_pH <-ifelse(is.na(wine_test$pH),
                          3.2,
                          wine_test$pH)

wine_test$M_pH <- ifelse(is.na(wine_test$pH),
                         1, 0)

wine_test$IMP_Sulphates <-ifelse(is.na(wine_test$Sulphates),
                                 0.50,
                                 wine_test$Sulphates)

wine_test$M_Sulphates <- ifelse(is.na(wine_test$Sulphates),
                                1, 0)


wine_test$IMP_Alcohol <-ifelse(is.na(wine_test$Alcohol),
                               10.40,
                               wine_test$Alcohol)

wine_test$M_Alcohol <- ifelse(is.na(wine_test$Alcohol),
                              1, 0)



wine_test$IMP_STARS <-ifelse(is.na(wine_test$STARS) & (wine_test$LabelAppeal
< -0.5), 1,
                             ifelse(is.na(wine_test$STARS) &
(wine_test$LabelAppeal >= -0.5),2,
                                    wine_test$STARS))

wine_test$M_STARS <- ifelse(is.na(wine_test$STARS),
                            1, 0)


# These have mean/median very low but min and max very high
#------------------------------


wine_test$IMP_TotalSulfurDioxide = ifelse(wine_test$IMP_TotalSulfurDioxide <
-350, -350,

ifelse(wine_test$IMP_TotalSulfurDioxide > 620, 620,

wine_test$IMP_TotalSulfurDioxide))

wine_test$IMP_ResidualSugar = ifelse(wine_test$IMP_ResidualSugar < -50, -50,
                                     ifelse(wine_test$IMP_ResidualSugar > 75,
75,
                                            wine_test$IMP_ResidualSugar))

wine_test$IMP_FreeSulfurDioxide = ifelse(wine_test$IMP_FreeSulfurDioxide < -
300, -300,
```

```
ifelse(wine_test$IMP_FreeSulfurDioxide > 350, 350,

wine_test$IMP_FreeSulfurDioxide))

summary(wine_test)

#-------------------------------------------------------------------------------
--
## 3- MODEL Deployment
#-------------------------------------------------------------------------------
---
#-------------------------
## Exporting Model - Poisson and Logistic Model
#-------------------------

#wine_test
TEMP_ZIP_Wine_test <- with(wine_test ,
                       (1.160645)       +
                         LabelAppeal     *  (0.226027)      +
                         AcidIndex       *  (-0.022432)     +
                         IMP_Alcohol     *  (0.006575)      +
                         IMP_STARS       *  (0.119220)      +
                         M_STARS         *  (-0.124404))

P_SCORE_Zip_ALL_Wine_test <- exp(TEMP_ZIP_Wine_test)

TEMP_Zip_Zero_Wine_test <- with(wine_test,
                            (-5.1897502)   +
                              VolatileAcidity            *   (   0.1885154   )
+
                              LabelAppeal                *   (   1.2245524   )
+
                              AcidIndex                * (   0.4426177   )
+
                              IMP_TotalSulfurDioxide   * (-0.0011424 )       +
                              IMP_pH                   * (   0.2197314   )
+
                              IMP_Sulphates            * (   0.0993321   )
+
                              IMP_Alcohol                *   (   0.0237497)
+
                              IMP_STARS                * (-1.5964480 )       +
                              M_STARS                    *   (   3.8688698
))

P_SCORE_Zip_ZERO_Wine_test <-
exp(TEMP_Zip_Zero_Wine_test)/(1+exp(TEMP_Zip_Zero_Wine_test))

P_TARGET <- P_SCORE_Zip_ALL_Wine_test * (1 - P_SCORE_Zip_ZERO_Wine_test)


hist(P_TARGET)

summary(P_TARGET)
```

```
#---------------------------------
## Exporting Model - Linear Model
#---------------------------------

P_TARGET

View(t(basicStats(P_TARGET)))
View(t(basicStats(round(P_TARGET,0))))

#---------------------------------
## Creating Scoring Ouput file
#---------------------------------

#Integer values
FINAL_Submission <- with(wine_test,
                         cbind.data.frame(INDEX,
                                          round(P_TARGET,0)))

colnames(FINAL_Submission) <- c("INDEX", "P_TARGET")
write.csv(FINAL_Submission,
          "Singh_Gurjeet_Wine_Sales_Test_Score_IntegerValues.csv")

#Decimal values
FINAL_Submission_dec <- with(wine_test,
                         cbind.data.frame(INDEX, P_TARGET))

colnames(FINAL_Submission_dec) <- c("INDEX", "P_TARGET")
write.csv(FINAL_Submission_dec,
          "Singh_Gurjeet_Wine_Sales_Test_Score_DecimalValues.csv")
```