Wine

Introduction

In this report we are analyzing various characteristics of wine in order to predict how many sample cases wine distribution companies will purchase. Wine distribution companies buy sample cases of wine in order to provide tasting samples to restaurants and wine shops. A large wine manufacturer is interested in learning which characteristics of wine are most desirable so that they can adjust their wine offering to maximize sales. Maximizing the number of purchased sample cases increases the likelihood of the wine being sold at high-end restaurants and stores. We will be using the Wine data set to build various models, including Poisson, Negative Binomial, Zero Inflated Poisson, Zero Inflated Negative Binomial, Linear Regression, Hurdle, and Ensemble models. In order to create a model that most accurately predicts the number of wine cases purchased, we will prepare the data using imputation and binning, and we will select the variables using stepwise automated variable selection and decision trees. We will present several models, discuss the merits and shortcomings of each, and ultimately select one model that best predicts how many sample cases wine distribution companies will purchase.

Data Exploration

The Wine data set contains 12,795 observations with 14 numeric variables, 1 target variable, and 1 index variable concerning commercially available wines. For this report, we will be focusing on predicting the TARGET variable, which signifies how many sample cases of wine a wine distribution company has purchased. In **Figure 1**, we can see that the TARGET variable follows a mostly normal distribution but is zero-inflated. Because the TARGET variable consists of discrete values and has an excessive amount of zeros, linear regression is probably not the best choice to model this variable, but we will try this method along with various other modeling techniques to try to create the most accurate predictions.

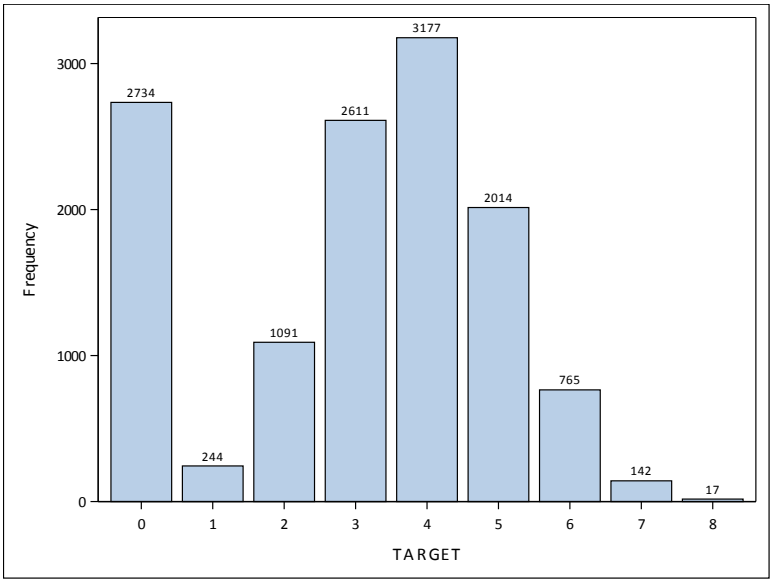

Figure 1: Distribution of TARGET

In **Table 1**, we see the mean and variance of TARGET. Because the mean is larger than the variance, the variable is considered underdispersed. Knowing that the distribution of the target

variable is underdispersed and zero-inflated may help us later on when we try to create and improve upon our predictive models.

Analysis Variable : TARGET	
Mean	Variance
3.8522016	1.5482330

Table 1: TARGET Summary

In order to predict how many sample cases of wine are purchased, we must first look at the data we have. In **Table 2**, we can see that most of the variables contain all 12,795 data points. Of the 14 numeric variables in the data set, eight have missing data: ResidualSugar, Chlorides, FreeSufurDioxide, TotalSulfurDioxide, pH, Sulphates, Alcohol, and STARS. Most of these variables are missing 5 percent or less of their data, while Sulphates is missing approximately 10 percent and STARS is missing a little over a quarter of its data. We believe that we can fix all of these variables with imputation. We will discuss our methods of imputation in the next section.

Variable	Minimum	Maximum	Mean	Median	N	N Miss
TARGET	0	8.0000000	3.0290739	3.0000000	12795	0
FixedAcidity	-18.1000000	34.4000000	7.0757171	6.9000000	12795	0
VolatileAcidity	-2.7900000	3.6800000	0.3241039	0.2800000	12795	0
CitricAcid	-3.2400000	3.8600000	0.3084127	0.3100000	12795	0
ResidualSugar	-127.8000000	141.1500000	5.4187331	3.9000000	12179	616
Chlorides	-1.1710000	1.3510000	0.0548225	0.0460000	12157	638
FreeSulfurDioxide	-555.0000000	623.0000000	30.8455713	30.0000000	12148	647
TotalSulfurDioxide	-823.0000000	1057.00	120.7142326	123.0000000	12113	682
Density	0.8880900	1.0992400	0.9942027	0.9944900	12795	0
pH	0.4800000	6.1300000	3.2076282	3.2000000	12400	395
Sulphates	-3.1300000	4.2400000	0.5271118	0.5000000	11585	1210
Alcohol	-4.7000000	26.5000000	10.4892363	10.4000000	12142	653
LabelAppeal	-2.0000000	2.0000000	-0.0090660	0	12795	0
AcidIndex	4.0000000	17.0000000	7.7727237	8.0000000	12795	0
STARS	1.0000000	4.0000000	2.0417550	2.0000000	9436	3359

Table 2: Overview of Variables in Wine Data Set

As we examine the table above, we see that many variables contain negative values. This does not seem very intuitive—how can wine have a negative amount of alcohol, for instance? Although these negative values do not make sense right now, we will have to wait to see what the distributions of these variables look like before deciding what to do. Also, we notice that some of the variables, such as FreeSufurDioxide and TotalSulfurDioxide, have very high and very low values. These extreme values might be indicative of outliers, which is something we will have to look for when we examine distribution plots.

After looking at summary tables to identify the missing values in the Wine data set, we continue our data exploration by looking at the distribution and probability plots for the variables. We do this to understand the data better and get a sense of possible outliers. Something we notice right away from looking at the plots is that the distributions look very similar for almost all of the variables representing chemical properties of wine. For instance, **Figures 2 and 3** depict the distributions of a couple variables with missing values, and **Figures 4 and 5** show the plots of a couple variables without any missing values.

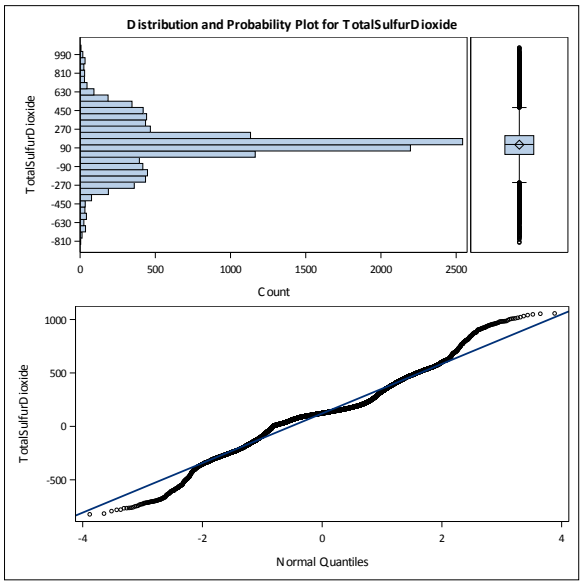

Figure 2: Distribution of TotalSulfurDioxide

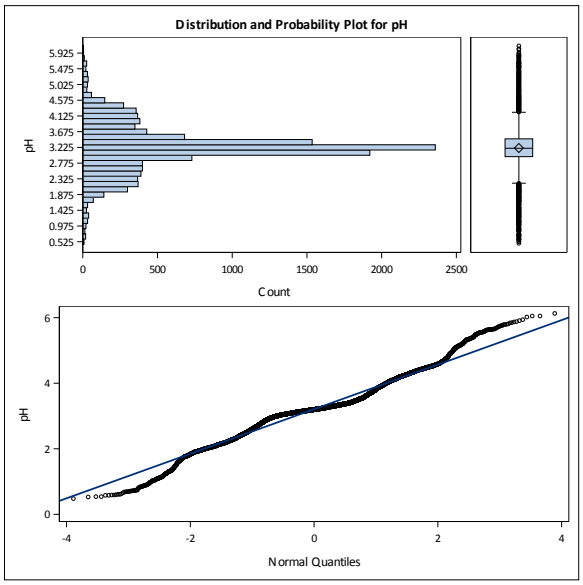

Figure 3: Distribution of pH

Each of these distribution and probability plots has a very stark peak in the middle of the histogram that is surrounded by smaller values and then very small values. The Q-Q plots all exhibit the same behavior: they begin below the 45-degree line then curve up, then back down, then back up, etc. The Q-Q plots "wobble" in the same way and differ only slightly in the sharpness of their curves. As for the box plots, all show a fairly small interquartile range (IQR) and have a lot of values that extend far beyond the upper and lower whiskers. Because all of the data points beyond the whiskers are very close together and seem to form a solid line of values, there are no obvious outliers we can confidentially remove or cap. We also notice that the negative values we found illogical in **Table 2** appear correct since there are a lot of negative values that help form the distributions. While it still seems counterintuitive that some variables, such as alcohol, can have a negative value, we will accept these negative values as correct and avoid tinkering with them to make them positive. Consulting an expert can perhaps provide insight into why these negative values make sense.

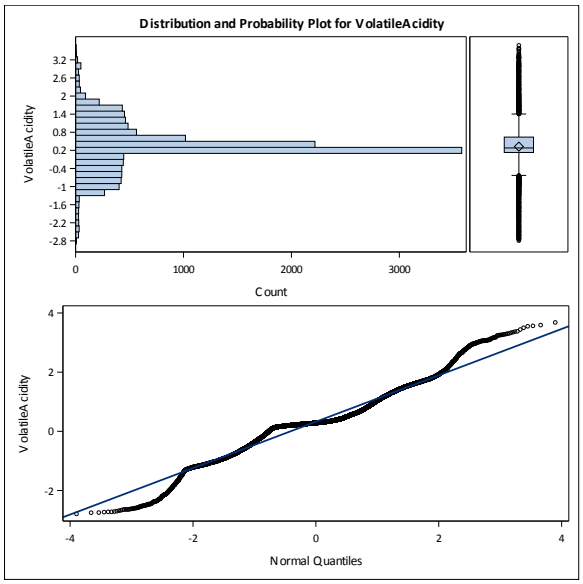

Figure 4: Distribution of VolatileAcidity

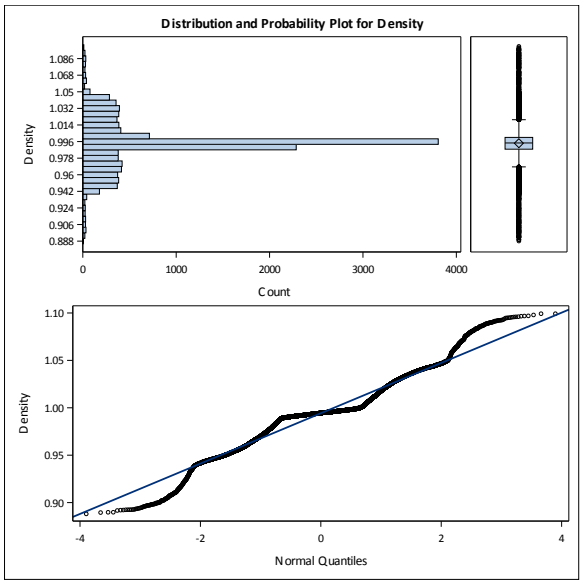

Figure 5: Distribution of Density

After examining the distribution and probability plots for each variable, we created a correlation matrix to see which variables correlate most strongly with TARGET. We notice that most variables have very poor correlation with TARGET, i.e., less than 10 percent, but that a few variables show some stronger correlations. In **Table 3** we selected just the variables that have the strongest correlations with TARGET. We can see that STARS has the strongest correlation at 0.55879. If you recall from **Table 2**, about a fourth of all records are missing values for STARS. Perhaps the correlation will change once we impute the missing values.

Pearson Correlation Coefficients Prob > |r| under H0: Rho=0				
	TARGET	**AcidIndex**	**LabelAppeal**	**STARS**
TARGET	1.00000	-0.24605 <.0001	0.35650 <.0001	0.55879 <.0001
AcidIndex	-0.24605 <.0001	1.00000	0.02475 0.0051	-0.08626 <.0001
LabelAppeal	0.35650 <.0001	0.02475 0.0051	1.00000	0.33479 <.0001
STARS	0.55879 <.0001	-0.08626 <.0001	0.33479 <.0001	1.00000

Table 3: Correlation Matrix of Select Variables

From the above matrix, we also can see that there is a fairly high correlation, 0.33479, between LabelAppeal and STARS. This suggests that wine with a higher star rating is likely to have a more attractive label. We may be able to use LabelAppeal to help impute the missing STARS values.

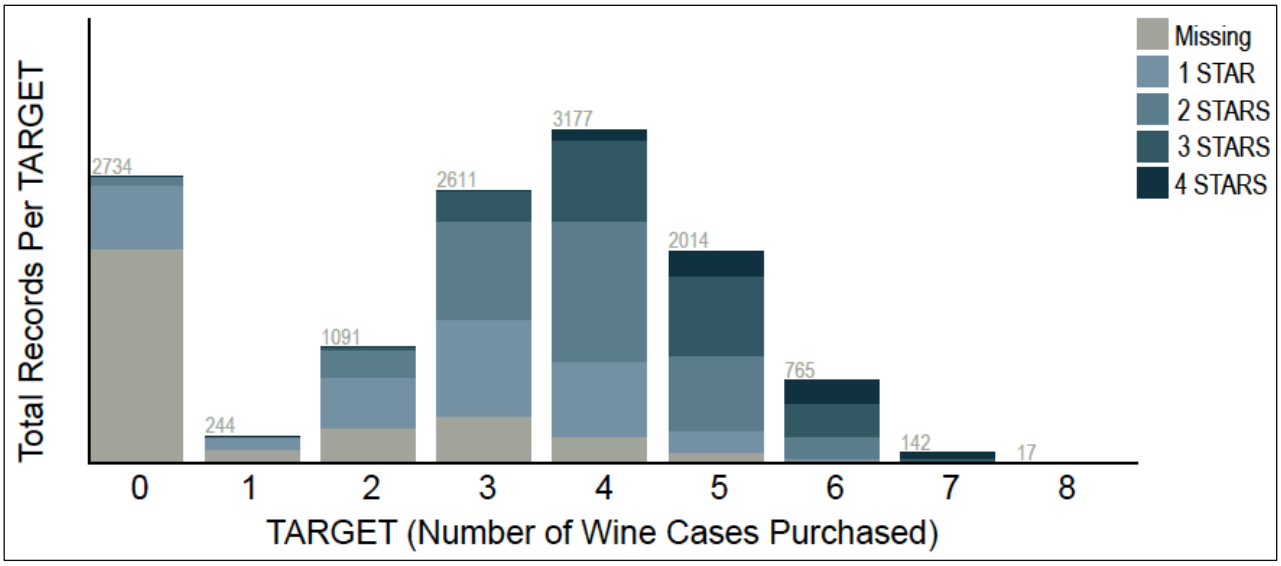

Figure 6: Distribution of TARGET by STARS

Because the number of STARS seems predictive of TARGET, we decided to visualize TARGET by STARS using Weka. In **Figure 6**, the STARS categories for column TARGET = 8 are not discernable in the plot, but a frequency table we created during our data exploration revealed that the majority of TARGET = 8 records were STARS = 4, with a few STARS = 3 and a couple STARS = missing values. **Figure 6** also shows that the majority of TARGET = 0 records have

missing values for STARS. The plot shows that as the number of cases purchased increases, the STARS rating generally increases.

TARGET_FLAG	N Obs	Variable	N Miss
0	2734	AcidIndex	0
		Alcohol	137
		Chlorides	141
		CitricAcid	0
		Density	0
		FixedAcidity	0
		FreeSulfurDioxide	139
		LabelAppeal	0
		ResidualSugar	127
		STARS	2038
		Sulphates	279
		TotalSulfurDioxide	140
		VolatileAcidity	0
		pH	101
1	10061	AcidIndex	0
		Alcohol	516
		Chlorides	497
		CitricAcid	0
		Density	0
		FixedAcidity	0
		FreeSulfurDioxide	508
		LabelAppeal	0
		ResidualSugar	489
		STARS	1321
		Sulphates	931
		TotalSulfurDioxide	542
		VolatileAcidity	0
		pH	294

Table 4: Summary Table of TARGET_FLAG

While conducting our data exploration, we decided to create a new variable, TARGET_FLAG, to further explore the relationship between the number of sample cases purchased and the variables. We set TARGET_FLAG = 0 for records where no cases of wine were purchased, and we set TARGET_FLAG = 1 for records were one or more cases of wine were purchased. In **Table 4,** we can see that 2734 records indicate that no cases of wine were sold, while 10,061 records indicate that cases of wine were sold. Given that the number of observations for TARGET_FLAG = 0 is approximately a third of the number of observations for TARGET_FLAG = 1, we expect the number of missing observations to follow this same logic. In general, the number of observations missing for each variable is three times higher for TARGET_FLAG = 1 than for TARGET_FLAG = 0 except for the variable STARS. When TARGET_FLAG = 0, the number of missing values for STARS = 2038, which is nearly twice as much as the 1321 missing values for STARS when TARGET_FLAG = 1. This is indicative that the missing data for STARS is likely highly predictive of the target variable.

Looking at the frequencies and distributions of the variables has given us some ideas how we might approach the data preparation, model creation, and model selection stages in this project. In the next section we will discuss in detail how we used the observations we made during our data exploration in order to prepare the data for modeling.

Data Preparation

The first step we took in preparing the data was to fix the missing values. **Table 5** shows just the variables with missing values. Although we created many different models with different variables, we consistently imputed the missing values for each variable except STARS in the same way. We imputed the mean values for ResidualSugar, Chlorides, FreeSulphurDioxide, and TotalSulphurDioxide, and we used the median values for pH, Sulphates, and Alcohol. We probably could have imputed using the means for each variable without causing drastic differences in results, but at the time we were preparing the data, we decided on an ad hoc decision rule to use the median if the mean and median were the same to the first decimal place.

Variable	Minimum	Maximum	Mean	Median	N	N Miss
ResidualSugar	-127.8000000	141.1500000	5.4187331	3.9000000	12179	616
Chlorides	-1.1710000	1.3510000	0.0548225	0.0460000	12157	638
FreeSulfurDioxide	-555.0000000	623.0000000	30.8455713	30.0000000	12148	647
TotalSulfurDioxide	-823.0000000	1057.00	120.7142326	123.0000000	12113	682
pH	0.4800000	6.1300000	3.2076282	3.2000000	12400	395
Sulphates	-3.1300000	4.2400000	0.5271118	0.5000000	11585	1210
Alcohol	-4.7000000	26.5000000	10.4892363	10.4000000	12142	653
STARS	1.0000000	4.0000000	2.0417550	2.0000000	9436	3359

Table 5: Overview of Variables with Missing Values

For STARS, we changed several times how we imputed the missing values. We first tried imputing values based on the values for LabelAppeal. We created a frequency table of STARS by LabelAppeal, shown in **Table 6**, to determine which values to impute for the missing STARS. Our method was to use the column percentage to assign the LabelAppeal value most frequently associated with a STARS rating to the missing STARS observations. For instance, we saw that 40.28 percent of LabelAppeal = -2 had a STARS rating of 1, so we would impute STARS = 1 for any observation that was missing its STARS data but had LabelAppeal = -2. By this logic, we would impute STARS = 1 when LabelAppeal = -2 or -1, STARS = 2 when LabelAppeal = 0 or 1, and STARS = 3 when LabelAppeal = 2. However, a transcription error caused us to assign STARS = 2 for all missing values except where LabelAppeal = -1 or 0. We will call out the model that used this accidental imputation technique when we discuss creating our models in the next section. Although this imputation technique was an error, it still yielded pretty good results and helped us realize that imputing “2” for more of the missing STARS data was better than imputing fewer of the missing STARS with “2”.

Table of STARS by LabelAppeal						
STARS	**LabelAppeal**					
Frequency Percent Row Pct Col Pct	**-2**	**-1**	**0**	**1**	**2**	**Total**
.	210 1.64 6.25 41.67	988 7.72 29.41 31.51	1411 11.03 42.01 25.12	651 5.09 19.38 21.36	99 0.77 2.95 20.20	3359 26.25
1	203 1.59 6.67 40.28	1008 7.88 33.14 32.14	1334 10.43 43.85 23.75	448 3.50 14.73 14.70	49 0.38 1.61 10.00	3042 23.77
2	70 0.55 1.96 13.89	849 6.64 23.78 27.07	1669 13.04 46.75 29.71	873 6.82 24.45 28.64	109 0.85 3.05 22.24	3570 27.90
3	21 0.16 0.95 4.17	262 2.05 11.84 8.35	1011 7.90 45.71 18.00	766 5.99 34.63 25.13	152 1.19 6.87 31.02	2212 17.29
4	0 0.00 0.00 0.00	29 0.23 4.74 0.92	192 1.50 31.37 3.42	310 2.42 50.65 10.17	81 0.63 13.24 16.53	612 4.78
Total	504 3.94	3136 24.51	5617 43.90	3048 23.82	490 3.83	12795 100.00

Table 6: STARS by LabelAppeal

Another imputation technique we tried for the missing values of STARS was to create a decision tree in R. The decision tree in **Figure 7** shows that we should impute “1” for observations where LabelAppeal is less than -0.5, i.e., where LabelAppeal = -1 or -2, and we should impute “2” for the rest of the observations with missing data. We tried this for one of our models and will discuss the results in the next couple sections.

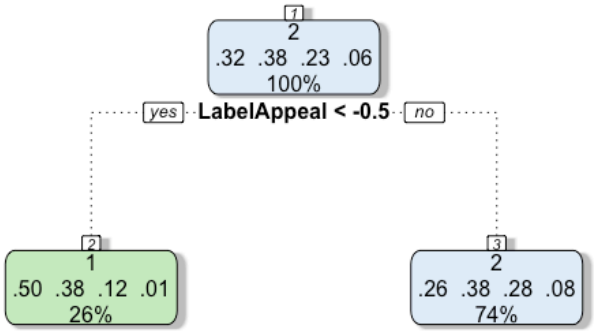

Figure 7: Decision Tree for STARS Imputation

A final technique we used for imputing the missing values for STARS was to use just the median, “2”, for all of the missing data. We used this technique for just one of our sets of models. For the models were we imputed “2” for all the missing STARS, we also binned many of the other

variables after we imputed missing values. Although we consistently imputed the same means and medians for each variable with missing data, we decided to take our data preparation one step further by grouping the data for every variable except IMP_STARS, LabelAppeal, and AcidIndex. We did not bin these three variables because they already had a very manageable number of bins: 4, 5, and 14, respectively. For all the other variables, we started binning the data after we had imputed all the missing values. We created three bins for each: a bin containing all the values less than the IQR, a bin containing the values within the IQR, and a bin containing all the values above the IQR. We used the quantile tables we examined during our data exploration stage, such as the quantile table for FixedAcidity in **Table 7**. For instance, we created the first bin for FixedAcidity so that it contained all values up to and including 5.2, the second bin with all values greater than 5.2 and up to and including 9.5, and the third bin with all the values greater than 9.5. Only now do we realize that we should have included the 25th percentile value in the IQR bin rather than having it fall in the bin for all the values below the IQR. This was a small error that likely did not affect the models too much but is something we would change if we were to prepare the data again.

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	34.4
99%	24.4
95%	17.8
90%	15.6
75% Q3	9.5
50% Median	6.9
25% Q1	5.2
10%	-1.2
5%	-3.6
1%	-10.9
0% Min	-18.1

Table 7: Quantiles for FixedAcidity

We chose to create only three bins for each variable based on the kind of distribution plots we examined in **Figures 2 to 5**. These plots seemed naturally divided in three sections: the small values, the middle values, and the large values, with the majority of the data falling into the middle values. In **Figures 8 and 9**, we can see how the binned data looks for a couple variables.

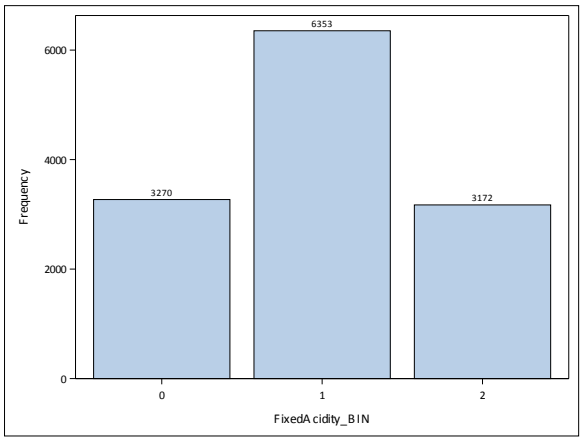

Figure 8: Distribution of FixedAcidity_BIN

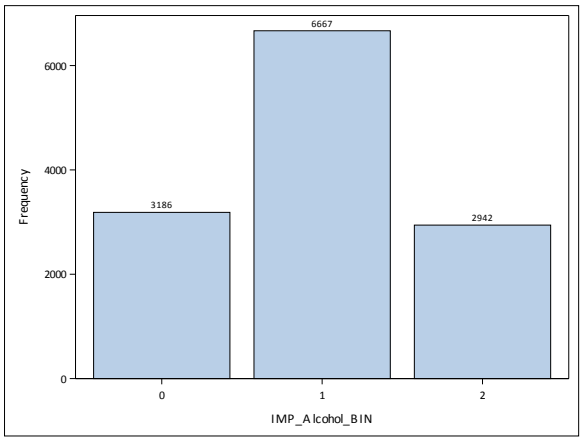

Figure 9: Distribution of IMP_Alcohol_BIN

In addition to wanting to change our binning process to include the 25th percentile value in the IQR bin, we think we should have probably based our binning on the updated quantile tables after we had imputed the missing values. We instead used the values from the quantile tables we examined during our data exploration stage, prior to our data imputations. **Figure 9** makes this process flaw evident since it shows that Bin 1 contains 6667 observations, which is more than half of 12,795 total observations. Bins 0 and 2 should also be equal since they are supposed to contain the lowest 25 percent and highest 25 percent of the data, respectively, but they clearly do not have the same number of observations. Although we are discovering these process flaws now, we still think that the models we created using binned data were successful. We will discuss why in a later section.

Model Creation

After exploring the data and preparing them so that they would be ready for modeling, we began selecting variables to include in the models. We built many distinct models using the Poisson, Negative Binomial, Zero Inflated Poisson, Zero Inflated Negative Binomial, Linear Regression, Hurdle, and Ensemble modeling techniques. For each change in our data preparation stage, we created a new set of these seven models using the updated data. We created models using three main data preparation techniques, and while we will not discuss all seven models for each of the three techniques, we will discuss at least one model for each data preparation technique for the sake of comparison. We will also provide a summary of the data preparation we performed when we introduce a new model.

For the first set of models, we used the data set where we meant to impute STARS = 1 when LabelAppeal = -2 or -1, STARS = 2 when LabelAppeal = 0 or 1, and STARS = 3 when LabelAppeal = 2, but where we actually accidentally imputed STARS = 2 for all missing values except where LabelAppeal = -1 or 0. We imputed the mean values for ResidualSugar, Chlorides, FreeSulphurDioxide, and TotalSulphurDioxide, and we used the median values for pH, Sulphates, and Alcohol. We will refer to this data set as the Model A data set.

In order to decide which variables to include in the Model A data set models, we used Weka to create a decision tree. We imported the original data set without the imputations into Weka, but we might change this in the future to try to create a better tree. We discretized all the variables except STARS, LabelAppeal, and AcidIndex into 10 bins each to enable the decision tree software to run smoothly. The resulting tree contained 612 leaves—far too many to try to analyze. We decided to prune this tree by specifying that each leaf must contain at least 120 observations,

or approximately 1 percent of the entire data set. The resulting tree contained only 37 leaves and was much more interpretable. The output for the pruned decision tree is in **Appendix A**.

When creating our seven models, we included only the variables that appeared in the pruned tree—plus a flag variable for the missing imputed STARS since we suspected variables with missing STARS were predictive. The formula for Model A ZIP is as follows:

```
TEMP =      1.3001
            + LabelAppeal*0.2408
            + AcidIndex*-0.0367
            + IMP_STARS*0.1075
            + M_IMP_STARS*-0.1102
            + IMP_Alcohol*0.0062
            + VolatileAcidity*-0.0173

P_SCORE_ZIP_ALL = exp(TEMP)

TEMP =      - 0.6724
            + IMP_STARS*-1.6165
            + LabelAppeal*1.1608
            + M_IMP_STARS*3.1521

P_SCORE_ZERO = exp(TEMP)/(1+exp(TEMP))

P_ZIP = P_SCORE_ZIP_ALL * (1-P_SCORE_ZERO)
```

The first formula for TEMP predicts the number of cases of wine purchased assuming that the wine is purchased. The formula suggests that the number of cases of wine purchased increases when the values for LabelAppeal, IMP_STARS, and IMP_Alcohol increase while the values for AcidIndex, M_IMP_STARS, and VolatileAcidity decrease. This suggests that wine distribution companies prefer wine with attractive labels, high star ratings, and more alcohol. A higher amount of acidity, represented by both the AcidIndex and VolatileAcidity, is not preferred. The fact that missing STARS decreases the likelihood of a wine being purchased indicates that unranked wines do not sell as well.

The second formula, for P_SCORE_ZIP_ALL, converts TEMP into a variable representing the number of cases of wine purchased assuming that the wine is purchased. The third formula, also called TEMP, calculates the logit value that wine is not purchased. (We chose to use only the three variables shown above rather than the six in the original equation because we found these variables were the most predictive. Including all six did not appear to change the results too much.) The fourth formula, for P_SCORE_ZERO, converts the logit into the probability that wine is not purchased. The fifth formula, for P_ZIP, calculates the expected number of cases of wine purchased. We performed a data quality check by comparing the probability values we output when we used PROC GENMOD with the values from the score code. **Table 8** shows that the probability values are nearly identical, save for slight rounding errors.

Obs	X_GENMOD_PZERO	P_SCORE_ZERO
1	0.01973	0.01974
2	0.00125	0.00125
3	0.00125	0.00125
4	0.03078	0.03078
5	0.01973	0.01974
6	0.70332	0.70333
7	0.70332	0.70333
8	0.01260	0.01260
9	0.70332	0.70333
10	0.00079	0.00079

Table 8: Data Quality Check for Model A ZIP Probabilities

We used the Model A data set to create six more models but will not discuss these here due to space constraints. The formulas are available in the code file.

For all the Model B data set models, we performed the same imputations as in the Model A data set, except we changed how we imputed the missing STARS. We created a decision tree in R, previously shown in **Figure 7**, to determine that we should impute "1" for observations where LabelAppeal = -1 or -2, and "2" for the rest of the observations with missing data. We used stepwise automated variable selection on a linear regression model first in order to determine which variables to include in the six other models. The formula for Model B ZIP is as follows:

```
TEMP =      1.5441
            + VolatileAcidity*-0.0173
            + IMP_Chlorides*-0.0247
            + IMP_FreeSulfurDioxide*0
            + IMP_TotalSulfurDioxide*0
            + Density*-0.2793
            + IMP_pH*0.0003
            + IMP_Sulphates*-0.0019
            + IMP_Alcohol*0.0061
            + LabelAppeal*0.2286
            + AcidIndex*-0.0359
            + IMP_STARS*0.1209
            + M_IMP_STARS*-0.1229

P_SCORE_ZIP_ALL = exp(TEMP)

TEMP =      -0.851
            + IMP_STARS*-1.5926
            + LabelAppeal*1.3092
            + M_IMP_STARS*4.0468

P_SCORE_ZERO = exp(TEMP)/(1+exp(TEMP))

P_ZIP= P_SCORE_ZIP_ALL * (1-P_SCORE_ZERO)
```

The first formula for TEMP suggests that the number of cases of wine purchased increases when the values for IMP_pH, IMP_Alcohol, LabelAppeal, and IMP_STARS increase but decreases when the values for VolatileAcidity, IMP_Chlorides, Density, IMP_Sulphates, AcidIndex, and M_IMP_STARS increase. IMP_FreeSulfurDioxide and IMP_TotalSulfurDioxide are both multiplied by 0, suggesting that they do not have an effect on the number of cases purchased, but they likely have a very small positive coefficient that was too small to display in SAS. The variables in Model B ZIP that also appear in Model A ZIP exhibit the same effect in the first TEMP formula, and variables specific to Model B ZIP suggest that wine distribution companies prefer wine with a lower amount of chlorides, density, and sulphates, but a higher pH. As mentioned earlier, IMP_FreeSulfurDioxide and IMP_TotalSulfurDioxide seem to have little effect on the number of cases purchased. The explanation for the remaining formulas for Model B ZIP is the same as what we discussed for Model A ZIP. We likewise performed the same data quality checks we mentioned in our discussion of Model A ZIP for Model B ZIP. In addition to creating Model B ZIP, we created six more models but will not discuss these here. The formulas are available in the code file.

For all the Model C data set models, we performed the same imputations as in the Model A and B data sets, except we imputed all the missing values for STARS as “2” and binned the data for all variables except IMP_STARS, LabelAppeal, and AcidIndex into the lower than IQR, IQR, and higher than IQR bins we discussed in the data preparation section. We used stepwise automated variable selection on a linear regression model in order to determine which variables to include in all seven models. For the set of Model C models, we will provide formulas for each of the modeling techniques we used and discuss their coefficients.

The first model we created using the Model C data set was a linear regression model. We created this model first since it allowed us to easily select which variables to include in all of the models. The formula for Model C Regression is:

P_REGRESSION = 3.62065
+ VolatileAcidity_BIN*-0.09027
+ CitricAcid_BIN*0.03814
+ IMP_Chlorides_BIN*-0.03157
+ IMP_FreeSulfurDioxide_BIN*0.04559
+ IMP_TotalSulfurDioxide_BIN*0.06855
+ IMP_pH_BIN*-0.04078
+ IMP_Sulphates_BIN*-0.03493
+ IMP_Alcohol_BIN*0.0506
+ LabelAppeal*0.46497
+ AcidIndex*-0.20442
+ IMP_STARS*0.78136
+ M_IMP_STARS*-2.24872

In addition to the formula above, we specified that any negative values of the predicted target, P_REGRESSION, should be set to 0 since it is impossible to purchase negative cases of wine. The above formula is suggesting that the estimated number of cases purchased increases as the variables CitricAcid_BIN, IMP_FreeSulfurDioxide_BIN, IMP_TotalSulfurDioxide_BIN, IMP_Alcohol_BIN, LabelAppeal, and IMP_STARS increase, and decreases as VolatileAcidity_BIN, IMP_Chlorides_BIN, IMP_pH_BIN, IMP_Sulphates_BIN, AcidIndex, M_IMP_STARS increase. Although the data is binned, the signs for the coefficients match the signs that appear in Model A ZIP and Model B ZIP, with the exception of the pH variables. Model B ZIP showed IMP_pH as having a positive effect on the predicted target, whereas Model C

Regression shows IMP_pH_BIN having a negative effect. This is not something to be concerned about since we used different data sets and different modeling techniques for these models, so there will naturally be some differences. We also are not sommeliers, so we do not have the expertise to comment on the differences in the coefficients for the chemical properties of wines. The only coefficients we can confidently comment on are the variables related to STARS and LabelAppeal since these are easy to interpret, even from a layman perspective.

The next two models we created, Model C Negative Binomial and Model C Poisson, yielded the same coefficients. In fact, regardless of the data preparation techniques, all of the models we created showed that the Negative Binomial and Poisson models yielded the same coefficients for a particular data set. The formula for Model C Negative Binomial and Model C Poisson is:

```
TEMP =      1.4978
            + VolatileAcidity_BIN*-0.0289
            + CitricAcid_BIN*0.0131
            + IMP_Chlorides_BIN*-0.0091
            + IMP_FreeSulfurDioxide_BIN*0.0159
            + IMP_TotalSulfurDioxide_BIN*0.0241
            + IMP_pH_BIN*-0.0148
            + IMP_Sulphates_BIN*-0.013
            + IMP_Alcohol_BIN*0.0122
            + LabelAppeal*0.1584
            + AcidIndex*-0.082
            + IMP_STARS*0.1887
            + M_IMP_STARS*-1.0247

P_NBPOI = exp(TEMP)
```

All of the signs for the coefficients in the model above match the signs for the coefficients in the Model C Regression formula. It makes sense that more attractive labels and higher star ratings increase the target values while more missing star ratings decrease the number of cases sold.

Another model we created is the Zero Inflated Poisson model. The formula for Model C ZIP is:

```
TEMP =      1.3285
            + VolatileAcidity_BIN*-0.0141
            + CitricAcid_BIN*0.0063
            + IMP_Chlorides_BIN*-0.0067
            + IMP_FreeSulfurDioxide_BIN*0.0063
            + IMP_TotalSulfurDioxide_BIN*-0.0006
            + IMP_pH_BIN*-0.0007
            + IMP_Sulphates_BIN*-0.0028
            + IMP_Alcohol_BIN*0.0284
            + LabelAppeal*0.2327
            + AcidIndex*-0.0334
            + IMP_STARS*0.1062
            + M_IMP_STARS*-0.1828

P_SCORE_ZIP_ALL = exp(TEMP);

TEMP =      2.5005
```

+ IMP_STARS*-4.1348
+ LabelAppeal*0.7496
+ M_IMP_STARS*6.2279

P_SCORE_ZERO = exp(TEMP)/(1+exp(TEMP))

P_ZIP = P_SCORE_ZIP_ALL * (1-P_SCORE_ZERO)

All of the signs for the coefficients in Model C ZIP match the signs for the coefficients in the formulas for the previous Model C data set models, with the exception of IMP_TotalSulfurDioxide_BIN. The coefficient for this variable became negative. Since we do not have expertise in the chemical properties of wines, we cannot conclude if this change is significant. It would be prudent to consult a wine connoisseur before deploying this model.

Another model we created is the Zero Inflated Negative Binomial model. The formula for Model C ZINB is:

TEMP = 1.3233
+ VolatileAcidity_BIN*-0.0136
+ CitricAcid_BIN*0.0058
+ IMP_Chlorides_BIN*-0.0062
+ IMP_FreeSulfurDioxide_BIN*0.0065
+ IMP_TotalSulfurDioxide_BIN*-0.0011
+ IMP_pH_BIN*-0.0006
+ IMP_Sulphates_BIN*-0.002
+ IMP_Alcohol_BIN*0.0279
+ LabelAppeal*0.2336
+ AcidIndex*-0.0324
+ IMP_STARS*0.1057
+ M_IMP_STARS*-0.1828

P_SCORE_ZINB_ALL = exp(TEMP);

TEMP = 0.5258
+ IMP_STARS*-2.2112
+ LabelAppeal*0.7178
+ M_IMP_STARS*4.3504

P_SCORE_ZERO = exp(TEMP)/(1+exp(TEMP));

P_ZINB = P_SCORE_ZINB_ALL * (1-P_SCORE_ZERO);

All of the signs for the coefficients in Model C ZINB match the signs for the coefficients in Model C ZIP. Again, we can really only comment on the fact that an increase in LabelAppeal and IMP_STARS increases the predicted target, while an increase in M_IMP_STARS decreases the predicted target. We have seen this consistently in the formulas for each model so far.

For the next model, Model C Hurdle, we had to combine a logistic regression and a Poisson model to create the hurdle model. First is the formula for the logistic regression model:

P_LOGIT_PROB = 2.0853
+ VolatileAcidity_BIN*-0.1793
+ CitricAcid_BIN*0.0766
+ IMP_Chlorides_BIN*-0.0136
+ IMP_FreeSulfurDioxide_BIN*0.0978
+ IMP_TotalSulfurDioxide_BIN*0.2495
+ IMP_pH_BIN*-0.18
+ IMP_Sulphates_BIN*-0.1156
+ IMP_Alcohol_BIN*-0.1108
+ LabelAppeal*-0.4694
+ AcidIndex*-0.3966
+ IMP_STARS*2.5549
+ M_IMP_STARS*-4.3646

if P_LOGIT_PROB > 1000 then P_LOGIT_PROB = 1000
if P_LOGIT_PROB < -1000 then P_LOGIT_PROB = -1000

P_LOGIT_PROB = exp(P_LOGIT_PROB) / (1+exp(P_LOGIT_PROB))

After creating the logistic regression model, we created the Poisson model. The formula is:

P_GENMOD_HURDLE = 0.9024
+ VolatileAcidity_BIN*-0.0114
+ CitricAcid_BIN*0.0033
+ IMP_Chlorides_BIN*-0.006
+ IMP_FreeSulfurDioxide_BIN*0.0062
+ IMP_TotalSulfurDioxide_BIN*-0.0106
+ IMP_pH_BIN*0.0059
+ IMP_Sulphates_BIN*0.0001
+ IMP_Alcohol_BIN*0.0394
+ LabelAppeal*0.2949
+ AcidIndex*-0.0213
+ IMP_STARS*0.1216
+ M_IMP_STARS*-0.209

P_GENMOD_HURDLE = exp(P_GENMOD_HURDLE)

We tied the predicted values together with the following formula:

P_HURDLE = P_LOGIT_PROB * (P_GENMOD_HURDLE+1)

From the logistic regression and Poisson formulas that compose Model C Hurdle, we see that only a few coefficient signs changed, and some of the magnitudes changed. We cannot comment much on these changes, aside from the fact that it is interesting that LabelAppeal has a negative coefficient in the logistic regression formula, whereas it has always had a positive coefficient in previous models. This is something that we might want to look into with the help of an expert since it seems counterintuitive that a more attractive label has a negative influence on sales.

After creating the Linear Regression, Poisson, Negative Binomial, Zero Inflated Poisson, Zero Inflated Negative Binomial, and Hurdle models for the Model C data set, we tied all of the results of these models together by creating an Ensemble model. The equation for Model C Ensemble is:

P_ENSEMBLE = (P_REGRESSION + P_NBPOI + P_ZIP + P_ZINB + P_HURDLE)/5

We divided only by 5 instead of 6 since the Poisson and Negative Binomial models yielded the same results.

Given that some of the coefficient signs switched for the various models that we built, we would want to investigate the data further before deploying the models. However, since pretty much all of the formulas suggested that an increase in LabelAppeal and IMP_STARS had a positive effect on the predicted estimate of wine cases purchased while an increase in M_IMP_STARS had a negative effect on the predicted target, we feel confident that the models are fairly stable and fine to deploy.

In this section we have looked at a lot of different models. In the next section we will discuss which of the models explored above we consider the best model for predicting the number of sample cases of wine purchased by wine distributors.

Model Selection

While examining all the models we created, we chose Model C Hurdle as the best model. The criteria we used for selecting the best model were the Mean Error (ME) and the Root Mean Square Error (RMSE). We compared the ME and RMSE values of each model created from the three model data sets we discussed in the previous sections. (Although we only provided the ZIP formula in the previous section for the Model A and Model B data sets, we have included the error rates for each model we created from both data sets in the table below to show how all the models performed.) **Table 9** shows the errors for each type of model we created using each of the three data sets, as well as what a simple average error would be for the data using either ME or RMSE. All of the models we created performed much better than the ME and RMSE values in the Error Type Mean column.

Model Data Set	Error Type	Error Type Mean	Error REG	Error NBPOI	Error ZIP	Error ZINB	Error HURDLE	Error ENSEMBLE
A	ME	1.55924	1.03165	1.0317	1.01444	1.01503	1.0009	1.00959
	RMSE	1.92629	1.33838	1.32021	1.3182	1.3164	1.30774	1.30454
B	ME	1.55924	1.04529	1.03518	1.00617	1.00856	0.98968	1.00605
	RMSE	1.92629	1.34225	1.31903	1.30853	1.30717	1.28749	1.29394
C	ME	1.55924	1.02627	1.0342	1.00208	1.00428	0.9766	0.99646
	RMSE	1.92629	1.31009	1.31646	1.29936	1.29933	1.27348	1.2811

Table 9: ME and RMSE Comparison for Models

As the table depicts, the Hurdle models generally performed the best for each model data set. Model A Ensemble, however, scored a better RMSE than Model A Hurdle, which breaks from the pattern of the Hurdle models scoring lowest for both ME and RMSE. Overall, the Hurdle model based on the Model C data set scored lowest for both ME and RMSE of all the models presented, thus making this model the best and least-error prone.

Although we ultimately decided to select the best model using ME and RMSE, we did consider evaluating the models on the sum of how many cases of wine each model estimated compared to the sum of how many cases of wine were actually purchased. **Table 10** shows the sum of TARGET, which is 38,757 cases of wine, and the estimated sums for each modeling technique.

Ideally the model that had the sum closest to the TARGET sum would be deemed the best model. However, we decided that it was more important to focus on how well the models predicted each individual TARGET value rather than the sum of the TARGET values because not all the wines are created equally. For instance, the table below suggests that Model A ZIP is most accurate since it predicted only four extra cases than the sum of the target variable. The sum, however, does not reveal what kinds of wine were purchased. Perhaps a wine distribution company wanted to purchase mostly 3- or 4-star wines, but ended up purchasing mostly 1- or 2-star wines. There is no way to know how accurate the model performed from just looking at the sums. A model may be good at predicting the total cases of wine to purchase, but it does not mean it is good at predicting the correct kinds of wine to purchase.

	Model A Data Set	Model B Data Set	Model C Data Set
Variable	**Sum**	**Sum**	**Sum**
TARGET	38757	38757	38757
P_REGRESSION	38741	38891	38839
P_NBPOI	38585	38712	38422
P_ZIP	38761	38990	38499
P_ZINB	38577	38636	38065
P_HURDLE	38840	38909	39000
P_ENSEMBLE	38656	38838	38168

Table 10: TARGET Sums Comparison for Models

Because using the sums for each model's estimates has its flaws, we decided that using the ME and RMSE values in **Table 9** would be best. However, a wine manufacturer might decide otherwise and deploy the model that best predicts the sum of the training data target values. We would advise investigating the data more closely before making such a business decision.

Conclusion

Our goal with this project was to create a model that best predicted how many cases of wine would be purchased based on its characteristics. We created numerous Poisson, Negative Binomial, Zero Inflated Poisson, Zero Inflated Negative Binomial, Linear Regression, Hurdle, and Ensemble models based on the Wine data set provided. We prepared the data using imputation and binning, and we used stepwise automated variable selection and decision trees to help choose variables for our models. Our best model surpassed the other models by having the lowest ME and RMSE values. In our Data Preparation section we mentioned a few changes we would like to explore, such as using the quantile values produced after imputing the missing values, and correcting the IQR bin in the Model C data set to include the 25th percentile value rather than having that value fall into the less-than-IQR bin. We would like to further investigate how these changes might affect the model's accuracy, and we are curious how a model might improve if we also incorporate some other techniques, such as using the STARS values suggested by the R decision tree or the variables suggested by the Weka decision tree. In the meantime, we are confident that Model C Hurdle is suitable for predicting the number of sample cases of wine that a wine distribution company will purchase.

Appendix A

```
=== Run information ===

Scheme:weka.classifiers.trees.J48 -C 0.25 -M 120
Relation:     wine-weka.filters.unsupervised.attribute.Discretize-F-B10-M-1.0-R3-13-
weka.filters.unsupervised.attribute.Discretize-F-B9-M-1.0-R2-
weka.filters.unsupervised.attribute.Discretize-F-B5-M-1.0-R14-
weka.filters.unsupervised.attribute.Discretize-F-B4-M-1.0-R16-
weka.filters.unsupervised.attribute.Discretize-F-B4-M-1.0-R16-
weka.filters.unsupervised.attribute.Discretize-F-B14-M-1.0-R15-
weka.filters.unsupervised.attribute.Remove-R1
Instances:    12795
Attributes:   15
              TARGET
              FixedAcidity
              VolatileAcidity
              CitricAcid
              ResidualSugar
              Chlorides
              FreeSulfurDioxide
              TotalSulfurDioxide
              Density
              pH
              Sulphates
              Alcohol
              LabelAppeal
              AcidIndex
              STARS
Test mode:evaluate on training data

=== Classifier model (full training set) ===

J48 pruned tree
------------------

LabelAppeal = '(-inf--0.5]'
|   AcidIndex <= 8
|   |   STARS = '(-inf-1.5]': '(0.5-2.5]' (1430.35/851.61)
|   |   STARS = '(1.5-2.5]': '(2.5-3.5]' (1101.7/665.98)
|   |   STARS = '(2.5-3.5]': '(2.5-3.5]' (377.23/243.07)
|   |   STARS = '(3.5-inf)': '(3.5-4.5]' (35.72/21.38)
|   AcidIndex > 8: '(-inf-0.5]' (695.0/404.0)
LabelAppeal = '(-0.5-0.5]'
|   STARS = '(-inf-1.5]'
|   |   AcidIndex <= 9
|   |   |   VolatileAcidity = '(-inf--0.7225]': '(2.5-3.5]' (160.13/104.74)
|   |   |   VolatileAcidity = '(-0.7225--0.135]': '(2.5-3.5]' (159.67/100.69)
|   |   |   VolatileAcidity = '(-0.135-0.1775]': '(3.5-4.5]' (161.3/95.59)
|   |   |   VolatileAcidity = '(0.1775-0.2325]': '(3.5-4.5]' (155.25/86.23)
```

Appendix A

```
|   |   |   VolatileAcidity = '(0.2325-0.2825]': '(2.5-3.5]' (161.84/102.37)
|   |   |   VolatileAcidity = '(0.2825-0.3575]': '(2.5-3.5]' (153.67/94.59)
|   |   |   VolatileAcidity = '(0.3575-0.515]': '(-inf-0.5]' (158.91/102.91)
|   |   |   VolatileAcidity = '(0.515-0.8275]': '(-inf-0.5]' (155.11/87.96)
|   |   |   VolatileAcidity = '(0.8275-1.355]': '(3.5-4.5]' (171.55/115.11)
|   |   |   VolatileAcidity = '(1.355-inf)': '(-inf-0.5]' (163.72/106.54)
|   |   AcidIndex > 9: '(-inf-0.5]' (180.36/60.0)
|   STARS = '(1.5-2.5]'
|   |   AcidIndex <= 8: '(3.5-4.5]' (1831.23/1058.7)
|   |   AcidIndex > 8
|   |   |   AcidIndex <= 9: '(3.5-4.5]' (224.39/148.43)
|   |   |   AcidIndex > 9: '(-inf-0.5]' (173.28/95.27)
|   STARS = '(2.5-3.5]': '(3.5-4.5]' (1350.16/829.26)
|   STARS = '(3.5-inf)': '(4.5-5.5]' (256.41/173.31)
LabelAppeal = '(0.5-1.5]'
|   STARS = '(-inf-1.5]'
|   |   AcidIndex <= 8: '(3.5-4.5]' (432.38/271.28)
|   |   AcidIndex > 8: '(-inf-0.5]' (137.3/53.18)
|   STARS = '(1.5-2.5]'
|   |   Alcohol = '(-inf-5.716667]': '(3.5-4.5]' (122.91/76.12)
|   |   Alcohol = '(5.716667-8.475]': '(4.5-5.5]' (104.49/55.1)
|   |   Alcohol = '(8.475-9.275]': '(3.5-4.5]' (119.43/62.79)
|   |   Alcohol = '(9.275-9.716667]': '(3.5-4.5]' (140.92/73.33)
|   |   Alcohol = '(9.716667-10.35]': '(4.5-5.5]' (119.83/77.84)
|   |   Alcohol = '(10.35-11.016667]': '(4.5-5.5]' (126.82/77.05)
|   |   Alcohol = '(11.016667-11.925]': '(4.5-5.5]' (98.16/58.26)
|   |   Alcohol = '(11.925-13.025]': '(4.5-5.5]' (101.33/61.64)
|   |   Alcohol = '(13.025-15.35]': '(4.5-5.5]' (87.33/49.67)
|   |   Alcohol = '(15.35-inf)': '(4.5-5.5]' (88.87/51.81)
|   STARS = '(2.5-3.5]'
|   |   AcidIndex <= 8: '(4.5-5.5]' (809.3/459.56)
|   |   AcidIndex > 8: '(-inf-0.5]' (164.74/94.43)
|   STARS = '(3.5-inf)': '(4.5-5.5]' (394.19/259.98)
LabelAppeal = '(1.5-inf)': '(5.5-6.5]' (490.0/307.0)

Number of Leaves  :   37

Size of the tree :    49


Time taken to build model: 0.3 seconds

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances        5180               40.4846 %
Incorrectly Classified Instances      7615               59.5154 %
Kappa statistic                          0.255
```

Appendix A

Mean absolute error 0.1781
Root mean squared error 0.297
Relative absolute error 87.753 %
Root relative squared error 93.2378 %
Total Number of Instances 12795

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.335	0.098	0.482	0.335	0.395	0.745	'(-inf-0.5]'
	0.312	0.051	0.417	0.312	0.357	0.878	'(0.5-2.5]'
	0.33	0.153	0.355	0.33	0.342	0.739	'(2.5-3.5]'
	0.607	0.303	0.398	0.607	0.48	0.728	'(3.5-4.5]'
	0.434	0.117	0.41	0.434	0.422	0.79	'(4.5-5.5]'
	0.239	0.026	0.373	0.239	0.292	0.889	'(5.5-6.5]'
	0	0	0	0	0	0.947	'(6.5-7.5]'
	0	0	0	0	0	0.976	'(7.5-inf)'
Weighted Avg.	0.405	0.153	0.405	0.405	0.395	0.772	

=== Confusion Matrix ===

a	b	c	d	e	f	g	h	<-- classified as
917	114	431	897	267	108	0	0	a = '(-inf-0.5]'
230	417	583	102	3	0	0	0	b = '(0.5-2.5]'
398	361	861	963	26	2	0	0	c = '(2.5-3.5]'
239	92	449	1927	457	13	0	0	d = '(3.5-4.5]'
95	17	96	821	875	110	0	0	e = '(4.5-5.5]'
22	0	3	131	426	183	0	0	f = '(5.5-6.5]'
3	0	0	4	76	59	0	0	g = '(6.5-7.5]'
0	0	0	0	2	15	0	0	h = '(7.5-inf)'