DATA621 - Spring2020 - Homework5

Abdelmalek Hajjam / Monu Chacko

Problem Statement

In this homework assignment, you will explore, analyze and model a data set containing information on approximately 12,000 commercially available wines. The variables are mostly related to the chemical properties of the wine being sold. The response 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. A large wine manufacturer is studying the data 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 that manufacturer will be able to adjust their wine offering to maximize sales.

Your objective is to build a count regression model to predict the number of cases of wine that will be sold given certain properties of the wine. HINT: Sometimes, the fact that a variable is missing is actually predictive of the target. You can only use the variables given to you (or variables that you derive from the variables provided).

Data Exploration

The data set includes 12,795 observations with 15 variables (including target variable). Of all 15 columns, 0 are discrete, 15 are continuous, and 0 are all missing. There are 8,200 missing values out of 191,925 data points. Below we'll display a few basic EDA techniques to gain insight into our wine dataset.

Basic statistics and Summary of Variables The data set includes 14 independent variables:

- `AcidIndex`: Proprietary method of testing total acidity of training by using a weighted average.
- `Alcohol`: Alcohol content of training.
- `Chlorides`: Chloride content of training.
- `CitricAcid`: Citric acid content of training.
- `Density`: Density of training.
- `FixedAcidity`: Fixed Acidity of training.
- `FreeSulfurDioxide`: Sulfur dioxide content of training.
- `LabelAppeal`: Marketing score indicating the appeal of label design for consumers.
- `ResidualSugar`: Residual sugar of training.
- `STARS`: training rating by a team of experts. Ranges from 1 (Poor) to 4 (Excellent) stars.
- `Sulphates`: Sulfate content of training.
- `TotalSulfurDioxide`: Total sulfur dDioxide of training.
- `VolatileAcidity`: Volatile acid content of training.
- `pH`: pH of training

Dependent variable is `TARGET` representing number of cases of training purchased.

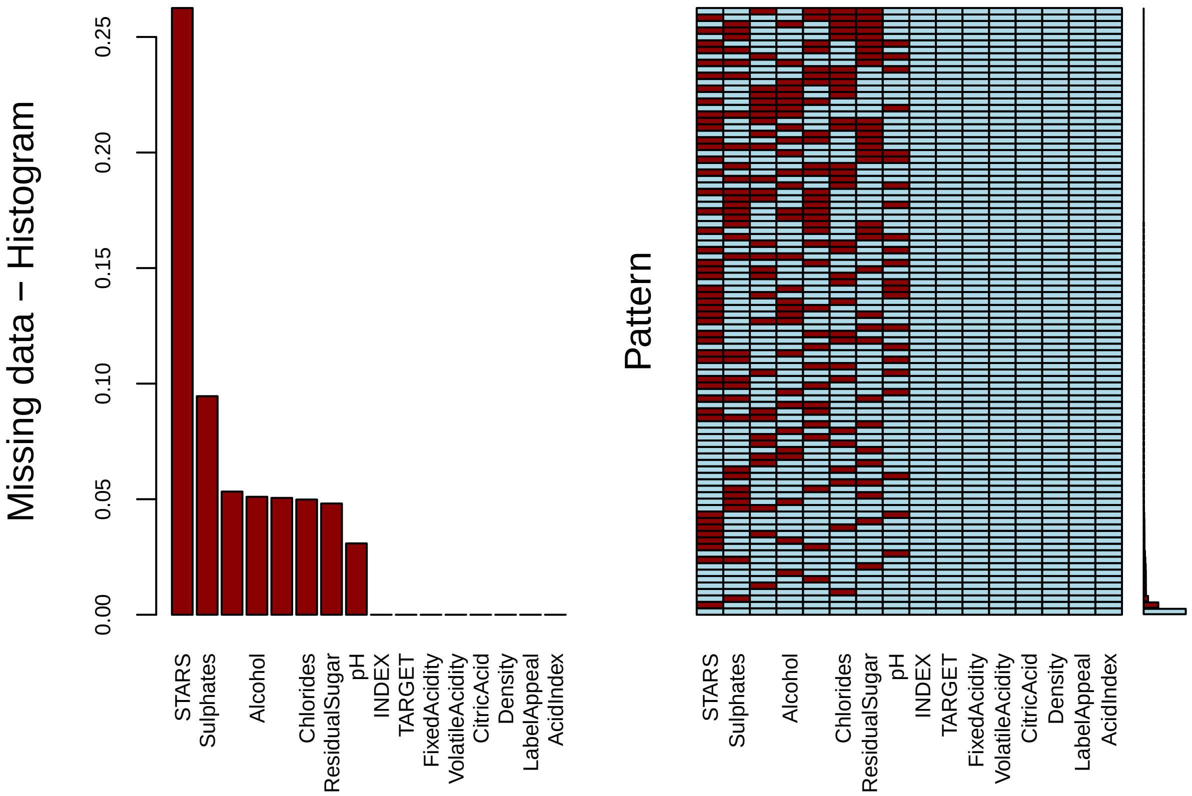
The table below shows summary of all variables.

Variable	Class	Min	Median	Mean	SD	Max
FixedAcidity	numeric	-18.1	6.9	7.076	6.318	34.4
VolatileAcidity	numeric	-2.79	0.28	0.3241	0.784	3.68
CitricAcid	numeric	-3.24	0.31	0.3084	0.8621	3.86
ResidualSugar	numeric	-127.8	3.9	5.419	33.75	141.2
Chlorides	numeric	-1.171	0.046	0.05482	0.3185	1.351
FreeSulfurDioxide	numeric	-555	30	30.85	148.7	623
TotalSulfurDioxide	numeric	-823	123	120.7	231.9	1057
Density	numeric	0.8881	0.9945	0.9942	0.02654	1.099
pH	numeric	0.48	3.2	3.208	0.6797	6.13
Sulphates	numeric	-3.13	0.5	0.5271	0.9321	4.24
Alcohol	numeric	-4.7	10.4	10.49	3.728	26.5
LabelAppeal	integer	-2	0	-0.009066	0.8911	2
AcidIndex	integer	4	8	7.773	1.324	17
STARS	integer	1	2	2.042	0.9025	4

Variable	Num of NAs	Num of Zeros	Num of Neg Values
FixedAcidity	0	39	1621
VolatileAcidity	0	18	2827
CitricAcid	0	115	2966
ResidualSugar	616	6	3136
Chlorides	638	5	3197
FreeSulfurDioxide	647	11	3036
TotalSulfurDioxide	682	7	2504
Density	0	0	0
pH	395	0	0
Sulphates	1210	22	2361
Alcohol	653	2	118
LabelAppeal	0	5617	3640
AcidIndex	0	0	0
STARS	3359	0	0

The Variables `LabelAppeal`, `AcidIndex` and `STARS` are categorical, and represented by numeric values in logical order. We Will then use them in modeling as numeric variables. All other variables are continous.

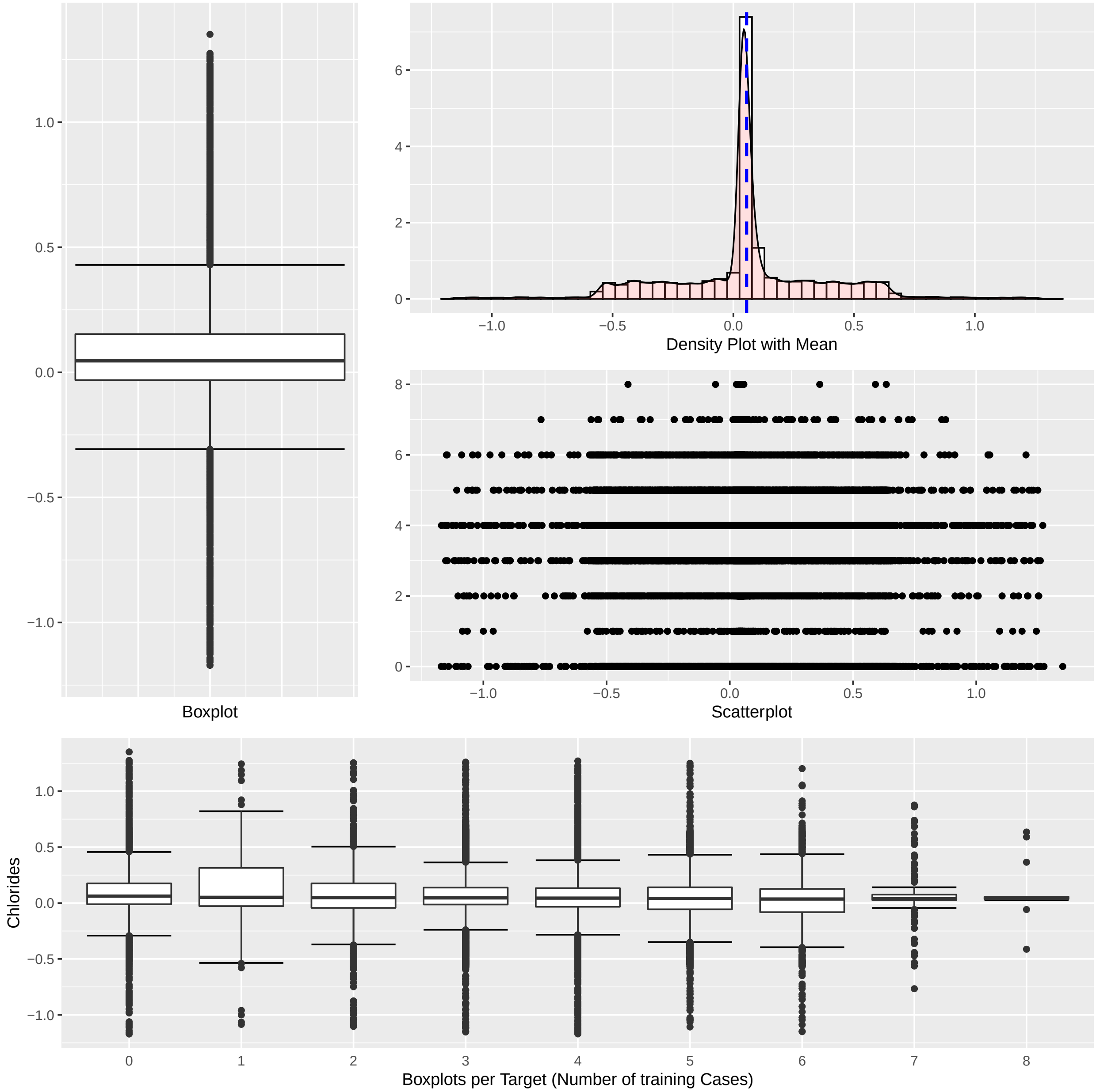
Missing Values We have 8 variables that have some sort of `NA` values. Most variables have negative values. Below plots show how the missing values are spread out within the data set. Approximately 25% of observations are missing a `STARS` value, and approximately 9% of observations have missing values.



```
##
##  Variables sorted by number of missings:
##            Variable      Count
##               STARS 0.26252442
##           Sulphates 0.09456819
##  TotalSulfurDioxide 0.05330207
##             Alcohol 0.05103556
##   FreeSulfurDioxide 0.05056663
##           Chlorides 0.04986323
##       ResidualSugar 0.04814381
##                  pH 0.03087143
##               INDEX 0.00000000
##              TARGET 0.00000000
##        FixedAcidity 0.00000000
##     VolatileAcidity 0.00000000
##          CitricAcid 0.00000000
##             Density 0.00000000
##         LabelAppeal 0.00000000
##           AcidIndex 0.00000000
```

Exploratory Plots Checking on the variables with different plots reveal that distribution for all variables are symmetrical and unimodal. We use scatter plot, density plot and box plot to inspect the variables. Box plots are also similar for most of the variables (excluding the last category).

We will use the variable `Chlorides` as an example for plotting. The plots will be the same for the other variables.



Correlations Correlation matrix below shows that there is very little correlation between variables. All can contribute in the modeling.

	TARGET	FixedAcidity	VolatileAcidity	CitricAcid	ResidualSugar	Chlorides	FreeSO2	TotalSO2	Density	pH	Sulphates	Alcohol	LabelAppeal	AcidIndex	STARS
TARGET	1	-0.05	-0.09	0.01	0.02	-0.04	0.04	0.05	-0.04	-0.01	-0.04	0.06	0.36	-0.25	0.56
FixedAcidity	-0.05	1	0.01	0.01	-0.02	0	0	-0.02	0.01	-0.01	0.03	-0.01	0	0.18	-0.01
VolatileAcidity	-0.09	0.01	1	-0.02	-0.01	0	-0.01	-0.02	0.01	0.01	0	0	-0.02	0.04	-0.03
CitricAcid	0.01	0.01	-0.02	1	-0.01	-0.01	0.01	0.01	-0.01	-0.01	-0.01	0.02	0.01	0.07	0
ResidualSugar	0.02	-0.02	-0.01	-0.01	1	-0.01	0.02	0.02	0	0.01	-0.01	-0.02	0	-0.01	0.02
Chlorides	-0.04	0	0	-0.01	-0.01	1	-0.02	-0.01	0.02	-0.02	0	-0.02	0.01	0.03	0
FreeSO2	0.04	0	-0.01	0.01	0.02	-0.02	1	0.01	0	0.01	0.01	-0.02	0.01	-0.04	-0.01
TotalSO2	0.05	-0.02	-0.02	0.01	0.02	-0.01	0.01	1	0.01	0	-0.01	-0.02	-0.01	-0.05	0.01
Density	-0.04	0.01	0.01	-0.01	0	0.02	0	0.01	1	0.01	-0.01	-0.01	-0.01	0.04	-0.02
pH	-0.01	-0.01	0.01	-0.01	0.01	-0.02	0.01	0	0.01	1	0.01	-0.01	0	-0.06	0
Sulphates	-0.04	0.03	0	-0.01	-0.01	0	0.01	-0.01	-0.01	0.01	1	0	0	0.03	-0.01
Alcohol	0.06	-0.01	0	0.02	-0.02	-0.02	-0.02	-0.02	-0.01	-0.01	0	1	0	-0.04	0.07
LabelAppeal	0.36	0	-0.02	0.01	0	0.01	0.01	-0.01	-0.01	0	0	0	1	0.02	0.33
AcidIndex	-0.25	0.18	0.04	0.07	-0.01	0.03	-0.04	-0.05	0.04	-0.06	0.03	-0.04	0.02	1	-0.09
STARS	0.56	-0.01	-0.03	0	0.02	0	-0.01	0.01	-0.02	0	-0.01	0.07	0.33	-0.09	1

Dependent Variable The dependent variable `TARGET` ranges from 0 (no cases purchased) to 8 cases of training purchased. The most common outcome is 4 cases at 25% of all observations followed closely with no purchase (0 cases) at 21%. Not counting the 0 outcome, it seems that the variable has unimodal, symmetrical distribution resembling normal distibution centered around 4.

Outcome	# of Observations	Percent of Total
0	2734	0.21
1	244	0.02
2	1091	0.09
3	2611	0.2
4	3177	0.25
5	2014	0.16
6	765	0.06
7	142	0.01
8	17	0

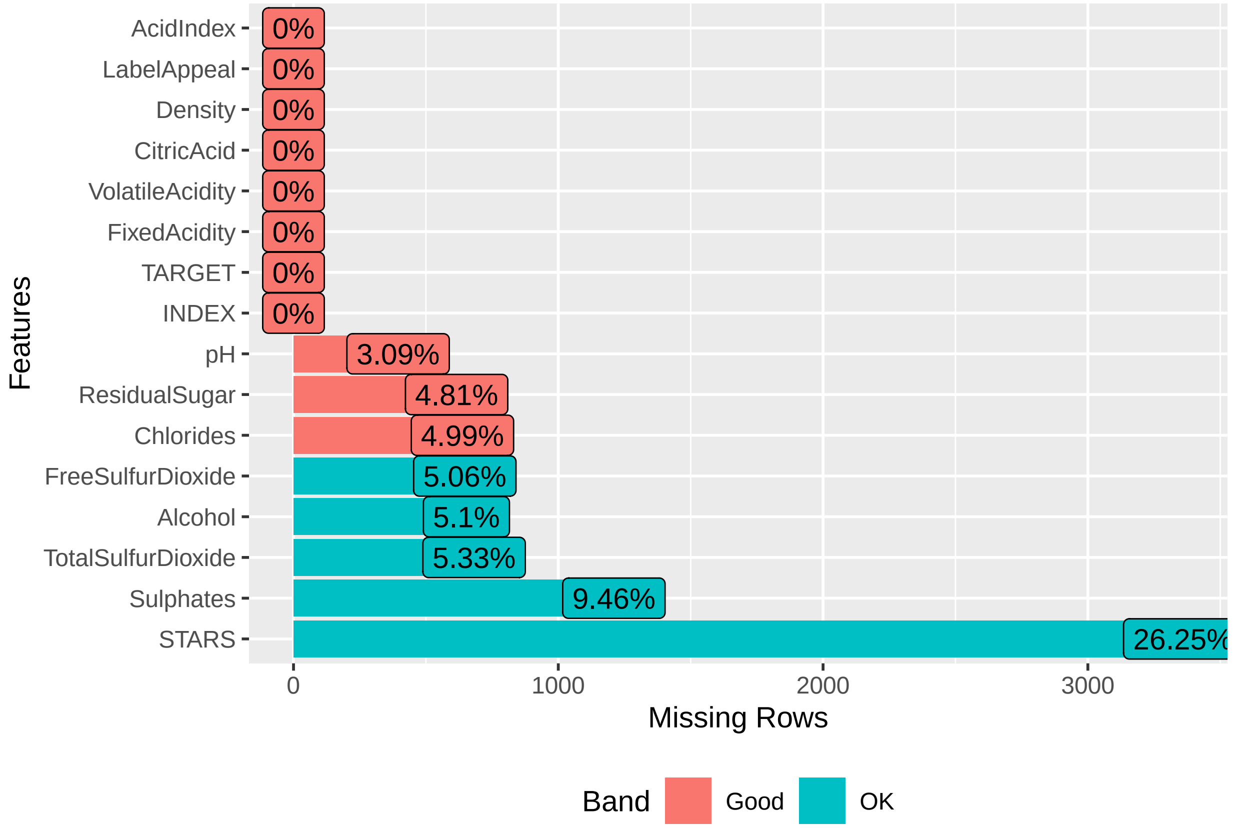
Data Preparation

All variables, as is, are good for modeling. Our dataset require some few transformations. The only concern we should have are the missing and negative values.

Consider Missing Values We have 2 variables, `LabelAppeal` and `AcidIndex`, that do not have any missing values. these are good to go.

The `STARS` variable has 3,359 missing values. It represents experts' rating and has meaning in modeling. We will replace the missing values for `STARS` with 0.

The Other variables that have missing values will be imputed. We will use the R package called `mice`. It has a handy method called `norm` that will perform the imputation.



Consider Negative Values The `Alcohol` variable has about 118 observations with negative values. Such a thing cannot be possible, because 0 is the minimum value for being non-alcoholic. Therefore we will transform this variable by taking absolute value for its observations.

Other variables contain negative values, but this must be how they chose to represent the data. Therefore, except for `STARS`, all negative values for all variables will not be changed and will stay the same.

Training/Testing Split We split our Dataset into a training (75% of observations) and testing (25% of observations) portions. We use the `caTools` R package to do that, based on the `TARGET` variable so that the target classes can be random, not bias, and having the same porportion between the 2 sets.

Modeling: Linear

We used two linear models, one full including all variables and another one using the `stepAIC` function.

The first resulted in R^2 of 0.5268, RMSE of 1.3184 and accuracy in predicting the outcomes in the testing set of 0.2853.

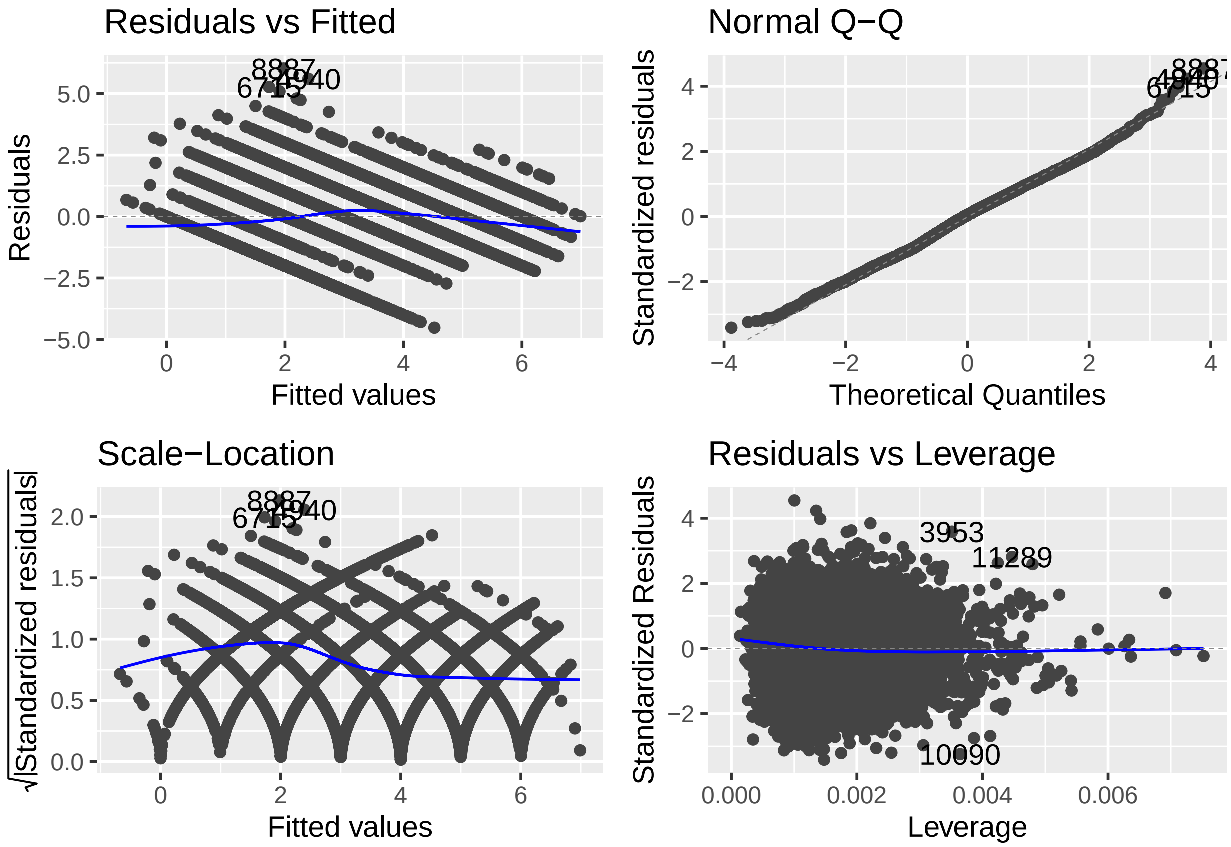
The second model who used the `stepAIC` function resulted in R^2 of 0.5266, RMSE of 1.3193 and accuracy of 0.2847 against the testing set.

According to those metrics, we concluded that the full model performed very slightly better than the stepwise model.

Here is the model summary.

```
## 
## Call:
## lm(formula = TARGET ~ . - INDEX, data = training.TRAIN)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -4.5219 -0.9513  0.0652  0.9084  6.0245 
## 
## Coefficients:
##                       Estimate  Std. Error t value             Pr(>|t|)    
## (Intercept)         3.70553113  0.52010041   7.125     0.00000000000112 ***
## FixedAcidity        0.00019648  0.00218902   0.090              0.92848    
## VolatileAcidity    -0.09152638  0.01732100  -5.284     0.00000012908170 ***
## CitricAcid          0.02355349  0.01575714   1.495              0.13500    
## ResidualSugar      -0.00007171  0.00040126  -0.179              0.85816    
## Chlorides          -0.12735379  0.04281180  -2.975              0.00294 ** 
## FreeSulfurDioxide   0.00029730  0.00009115   3.262              0.00111 ** 
## TotalSulfurDioxide  0.00014995  0.00005870   2.554              0.01066 *  
## Density            -0.60281825  0.51216041  -1.177              0.23922    
## pH                 -0.03063097  0.01988702  -1.540              0.12353    
## Sulphates          -0.02775911  0.01460659  -1.900              0.05740 .  
## Alcohol             0.01209422  0.00375157   3.224              0.00127 ** 
## LabelAppeal         0.41968294  0.01575483  26.638 < 0.0000000000000002 ***
## AcidIndex          -0.20248853  0.01067812 -18.963 < 0.0000000000000002 ***
## STARS               0.98375162  0.01199550  82.010 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1.326 on 9580 degrees of freedom
## Multiple R-squared:  0.5267, Adjusted R-squared:  0.526 
## F-statistic: 761.4 on 14 and 9580 DF,  p-value: < 0.00000000000000022
```

The diagnostic plots show us that the model performs really well. Because the dependent variable is a count variable, Some of the plots are not very useful for this dataset.



According to the below confusion matrix, the accuracy is fairly low at 28.3%; however, if we examine full confusion matrix below we can see that the model mostly errors only by 1 or 2 cases. There may be significant cost associated with this error.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2   3   4   5   6   7   8
##          0  23   2   0   2   0   0   0   0   0
##          1 319  29  67  82  23   7   0   0   0
##          2 251  26 117 180 108  26  10   2   0
##          3  86   3  74 234 226  74   9   0   0
##          4   5   1  14 128 296 196  42   3   0
##          5   0   0   1  27 130 155  76  14   1
##          6   0   0   0   0  11  45  51  15   1
##          7   0   0   0   0   0   1   3   2   2
##          8   0   0   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.2834
##                  95% CI : (0.2679, 0.2994)
##     No Information Rate : 0.2481
##     P-Value [Acc > NIR] : 0.000002815
##
##                   Kappa : 0.1624
```

```
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity          0.033626 0.475410  0.42857  0.35835   0.3728  0.30754
## Specificity          0.998410 0.841351  0.79399  0.81468   0.8383  0.90764
## Pos Pred Value       0.851852 0.055028  0.16250  0.33144   0.4321  0.38366
## Neg Pred Value       0.791680 0.988028  0.93710  0.83200   0.8020  0.87518
## Prevalence           0.213750 0.019062  0.08531  0.20406   0.2481  0.15750
## Detection Rate       0.007188 0.009062  0.03656  0.07312   0.0925  0.04844
## Detection Prevalence 0.008438 0.164687  0.22500  0.22062   0.2141  0.12625
## Balanced Accuracy    0.516018 0.658380  0.61128  0.58652   0.6056  0.60759
##                      Class: 6 Class: 7 Class: 8
## Sensitivity           0.26702 0.055556  0.00000
## Specificity           0.97607 0.998104  1.00000
## Pos Pred Value        0.41463 0.250000      NaN
## Neg Pred Value        0.95450 0.989348  0.99875
## Prevalence            0.05969 0.011250  0.00125
## Detection Rate        0.01594 0.000625  0.00000
## Detection Prevalence  0.03844 0.002500  0.00000
## Balanced Accuracy     0.62154 0.526830  0.50000
```

Modeling: Poisson

The linear model seemed to perform good with all variables. So for the poisson regression similar strategy was applied - a model with all variables and a model optimized by the stepwise method. There was no considerable imrovement using the stepwise method, so for comparison reasons below is the summary for the full model. RMSE for this model is 1.39855, slightly worse than for the linear model.

```
##
## Call:
## glm(formula = TARGET ~ . - INDEX, family = poisson, data = training.TRAIN)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9632  -0.7309   0.0679   0.5744   3.2461
##
## Coefficients:
##                       Estimate  Std. Error z value             Pr(>|z|)
## (Intercept)         1.43464543  0.22550632   6.362       0.000000000199 ***
## FixedAcidity       -0.00028279  0.00095340  -0.297              0.76676
## VolatileAcidity    -0.03212768  0.00753889  -4.262       0.000020297599 ***
## CitricAcid          0.00902046  0.00682187   1.322              0.18607
## ResidualSugar      -0.00006234  0.00017403  -0.358              0.72021
## Chlorides          -0.04405384  0.01860105  -2.368              0.01787 *
## FreeSulfurDioxide   0.00012067  0.00003949   3.055              0.00225 **
## TotalSulfurDioxide  0.00005587  0.00002575   2.170              0.03002 *
## Density            -0.22242253  0.22133123  -1.005              0.31493
## pH                 -0.01497739  0.00861743  -1.738              0.08220 .
## Sulphates          -0.01138039  0.00635560  -1.791              0.07336 .
## Alcohol             0.00290555  0.00163716   1.775              0.07594 .
## LabelAppeal         0.13013647  0.00699899  18.594 < 0.0000000000000002 ***
## AcidIndex          -0.08450998  0.00525577 -16.079 < 0.0000000000000002 ***
## STARS               0.31335960  0.00518622  60.422 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 17142  on 9594  degrees of freedom
## Residual deviance: 11061  on 9580  degrees of freedom
## AIC: 35046
##
## Number of Fisher Scoring iterations: 5
```

```
## [1] 1.398167
```

Comparing predicted values to the test data, with the folowing confusion matrix, shows that the model does not predict *no purchase* outcome (count is 0). Worse than that, it often predicts fewer cases than the test data indicates. Accuracy is very bad and lower than the one for the linear model.

```
##  Accuracy
## 0.2315625
```

```
##           Reference
## Prediction   0   1   2   3   4   5   6   7   8   9  10
##         0    0   0   0   0   0   0   0   0   0   0   0
##         1  206  20  36  38   7   4   0   0   0   0   0
##         2  420  37 164 284 178  47  12   2   0   0   0
##         3   57   4  65 230 306 114  13   1   0   0   0
##         4    1   0   7  75 182 155  37   2   0   0   0
##         5    0   0   1  24  96 104  51  12   1   0   0
##         6    0   0   0   2  18  54  32   5   0   0   0
##         7    0   0   0   0   5  20  30   9   1   0   0
##         8    0   0   0   0   2   5  15   4   0   0   0
##         9    0   0   0   0   0   1   0   1   2   0   0
##         10   0   0   0   0   0   0   1   0   0   0   0
```

Modeling: Negative Binomial

We create a negative binomial model with all variables with the help of the `MASS` R package. This model turned out to be nearly identical to the poisson model. code is on the bottom of the page.

Both models exhibited over-dispersion.

```
##
## Call:
## glm.nb(formula = TARGET ~ . - INDEX, data = training.TRAIN, init.theta = 49050.49378,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9631  -0.7309   0.0679   0.5744   3.2460
```

```
##
## Coefficients:
##                       Estimate  Std. Error z value            Pr(>|z|)
## (Intercept)         1.43465967  0.22551525   6.362        0.0000000002 ***
## FixedAcidity       -0.00028280  0.00095343  -0.297             0.76676
## VolatileAcidity    -0.03212851  0.00753919  -4.262        0.0000203030 ***
## CitricAcid          0.00902057  0.00682214   1.322             0.18608
## ResidualSugar      -0.00006233  0.00017404  -0.358             0.72024
## Chlorides          -0.04405475  0.01860179  -2.368             0.01787 *
## FreeSulfurDioxide   0.00012067  0.00003949   3.055             0.00225 **
## TotalSulfurDioxide  0.00005588  0.00002575   2.170             0.03002 *
## Density            -0.22242728  0.22134001  -1.005             0.31494
## pH                 -0.01497805  0.00861777  -1.738             0.08220 .
## Sulphates          -0.01138084  0.00635585  -1.791             0.07336 .
## Alcohol             0.00290552  0.00163722   1.775             0.07595 .
## LabelAppeal         0.13013584  0.00699926  18.593 < 0.0000000000000002 ***
## AcidIndex          -0.08451184  0.00525596 -16.079 < 0.0000000000000002 ***
## STARS               0.31336367  0.00518642  60.420 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(49050.49) family taken to be 1)
##
##     Null deviance: 17141  on 9594  degrees of freedom
## Residual deviance: 11061  on 9580  degrees of freedom
## AIC: 35048
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  49050
##           Std. Err.:  58783
## Warning while fitting theta: iteration limit reached
##
##  2 x log-likelihood:  -35015.8
```

Modeling: Zero-Inflated Negative Binomial

Poisson and negative binomial models do not account for the 0 outcome. So a zero-inflated negative binomial model was attempted using the `pscl` R package. RMSE for this model is 1.2727, the best one out of all models.

```
##
## Call:
## zeroinfl(formula = TARGET ~ . - INDEX, data = training.TRAIN, dist = "negbin")
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -2.1207 -0.4049 -0.0094  0.3709  5.8061
##
## Count model coefficients (negbin with log link):
##                       Estimate  Std. Error z value            Pr(>|z|)
## (Intercept)         1.39772713  0.23301531   5.998       0.00000000199 ***
## FixedAcidity        0.00034372  0.00098199   0.350               0.7263
## VolatileAcidity    -0.01186651  0.00778322  -1.525               0.1274
## CitricAcid          0.00292492  0.00700362   0.418               0.6762
## ResidualSugar      -0.00014976  0.00017918  -0.836               0.4033
## Chlorides          -0.02524922  0.01914007  -1.319               0.1871
## FreeSulfurDioxide   0.00001534  0.00003986   0.385               0.7004
## TotalSulfurDioxide -0.00003788  0.00002561  -1.479               0.1391
## Density            -0.23473137  0.22811571  -1.029               0.3035
## pH                  0.00290531  0.00888001   0.327               0.7435
## Sulphates          -0.00085515  0.00654214  -0.131               0.8960
## Alcohol             0.00685935  0.00167724   4.090        0.00004320043 ***
## LabelAppeal         0.23202653  0.00727758  31.882 < 0.0000000000000002 ***
## AcidIndex          -0.01809361  0.00566373  -3.195               0.0014 **
## STARS               0.10293969  0.00596307  17.263 < 0.0000000000000002 ***
## Log(theta)         20.67721778  0.88536301  23.355 < 0.0000000000000002 ***
##
## Zero-inflation model coefficients (binomial with logit link):
##                       Estimate  Std. Error z value             Pr(>|z|)
## (Intercept)        -4.43637559  1.56595577  -2.833             0.004611 **
## FixedAcidity       -0.00004499  0.00642667  -0.007             0.994414
## VolatileAcidity     0.15277230  0.04998353   3.056             0.002240 **
## CitricAcid         -0.01713255  0.04633634  -0.370             0.711574
## ResidualSugar      -0.00119371  0.00116782  -1.022             0.306699
## Chlorides           0.02518088  0.12416482   0.203             0.839290
## FreeSulfurDioxide  -0.00097050  0.00027193  -3.569             0.000358 ***
## TotalSulfurDioxide -0.00094462  0.00016974  -5.565         0.0000000262 ***
## Density             0.61432674  1.53705055   0.400             0.689393
## pH                  0.19853234  0.05812331   3.416             0.000636 ***
## Sulphates           0.10308116  0.04270171   2.414             0.015779 *
## Alcohol             0.02436295  0.01102603   2.210             0.027134 *
## LabelAppeal         0.75562643  0.04963533  15.224 < 0.0000000000000002 ***
## AcidIndex           0.43455751  0.03006399  14.454 < 0.0000000000000002 ***
## STARS              -2.38860729  0.07002385 -34.111 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Theta = 954996066.2911
## Number of iterations in BFGS optimization: 37
## Log-likelihood: -1.53e+04 on 31 Df
```

```
## [1] 1.272187
```

This model has the accuracy of 36.03%, again the best one out of all models. It predicts 0 outcomes (not ideally, but perhaps it can be improved with more research).

```
##  Accuracy
## 0.3603125
```

```
##           Reference
## Prediction   0   1   2   3   4   5   6   7   8
##          0 112   1   0   6   4   2   1   1   0
##          1 315  15  37  77  39  17   5   1   0
##          2 141  36 121 112  40  11   4   0   0
##          3 102   9 108 313 242  65   4   0   0
##          4  14   0   7 139 376 197  35   2   0
##          5   0   0   0   6  81 165  75  10   0
##          6   0   0   0   0  12  37  44  15   1
##          7   0   0   0   0   0  10  19   7   3
##          8   0   0   0   0   0   0   4   0   0
```

Model Selection

Considering log-likelihood of all models, it is clear that zero-inflated negative binomial model is the best option. More research in that direction will probably be beneficial.

	Log-Likelihood	DF
Linear	-16317	16
Poisson	-17508	15
NB	-17508	16
ZINB	-15299	31

Comparing all coefficients using full model with all methods, we see that usually the coefficients are similar in sign and in magnitude. We also notice that between NB and ZINB models, some small coefficients do change signs.

	Linear	Poisson	NB	ZINB (Count)
(Intercept)	3.706	1.435	1.435	1.398
FixedAcidity	0.000196	-0.000283	-0.000283	0.000344
VolatileAcidity	-0.09153	-0.03213	-0.03213	-0.01187
CitricAcid	0.02355	0.00902	0.009021	0.002925
ResidualSugar	-0.000072	-0.000062	-0.000062	-0.00015
Chlorides	-0.1274	-0.04405	-0.04405	-0.02525
FreeSulfurDioxide	0.000297	0.000121	0.000121	0.000015
TotalSulfurDioxide	0.00015	0.000056	0.000056	-0.000038
Density	-0.6028	-0.2224	-0.2224	-0.2347
pH	-0.03063	-0.01498	-0.01498	0.002905
Sulphates	-0.02776	-0.01138	-0.01138	-0.000855
Alcohol	0.01209	0.002906	0.002906	0.006859
LabelAppeal	0.4197	0.1301	0.1301	0.232
AcidIndex	-0.2025	-0.08451	-0.08451	-0.01809
STARS	0.9838	0.3134	0.3134	0.1029

Evaluation of our selected Zero-Inflated Model

We will only display the evaluation result of the first 50 observations from the evaluation set. The generated Prediction file will be saved into a csv file called Prediction_for_Eval.csv located at: https://github.com/theoracley/Data621/blob/master/Homework5/Prediction_For_Eval.csv

Index	Predicted Value	Predicted Outcome
3	1.903	2
9	3.82	4
10	2.511	3
18	2.503	3
21	0.7087	1
30	5.697	6
31	3.587	4
37	1.391	1
39	0.2432	0
47	1.43	1
60	2.86	3
62	0.1887	0
63	3.545	4
64	1.331	1
68	1.145	1
75	2.757	3
76	2.53	3
83	0.0482	0
87	3.72	4
92	5.44	5
98	2.319	2
106	1.648	2
107	0.5035	1
113	2.54	3
120	3.46	3
123	5.907	6
125	2.844	3
126	5.876	6
128	4.516	5
129	2.443	2
131	4.216	4
135	0.9734	1
141	4.204	4
147	3.222	3
148	1.28	1
151	3.748	4
156	3.16	3
157	3.384	3
174	1.709	2
186	0.4659	0
193	2.586	3
195	0.9954	1
212	0.6932	1
213	0.7003	1
217	2.961	3
223	3.83	4
226	3.158	3

Index	Predicted Value	Predicted Outcome
228	4.541	5
230	4.128	4
241	2.61	3

APPENDIX

```r
library(ggplot2)    # plotting
library(dplyr)      # data manipulation
library(gridExtra)  # display
library(knitr)      # display
library(kableExtra) # display
library(mice)       # imputation
library(caTools)    # train-test split
library(MASS)       # boxcox
library(Metrics)    # rmse
library(caret)      # confusion matrix
library(VIM)        # plotting NAs
library(ggfortify)  # plotting lm diagnostic
library(car)        # VIF
library(pander)
library(pscl)       # zero-inflated model
library(DataExplorer)

training <- read.csv("https://raw.githubusercontent.com/theoracley/Data621/master/Homework5/wine-trainin
colnames(training)[1] <- "INDEX"

# Basic statistic
nrow(training); ncol(training)
summary(training)

# Summary table
sumtbl = data.frame(Variable = character(),
                    Class = character(),
                    Min = integer(),
                    Median = integer(),
                    Mean = double(),
                    SD = double(),
                    Max = integer(),
                    Num_NAs = integer(),
                    Num_Zeros = integer(),
                    Num_Neg = integer())
for (i in c(3:16)) {
  sumtbl <- rbind(sumtbl, data.frame(Variable = colnames(training)[i],
                                     Class = class(training[,i]),
                                     Min = min(training[,i], na.rm=TRUE),
                                     Median = median(training[,i], na.rm=TRUE),
                                     Mean = mean(training[,i], na.rm=TRUE),
                                     SD = sd(training[,i], na.rm=TRUE),
                                     Max = max(training[,i], na.rm=TRUE),
                                     Num_NAs = sum(is.na(training[,i])),
                                     Num_Zeros = length(which(training[,i]==0)),
                                     Num_Neg = sum(training[,i]<0 & !is.na(training[,i]))))
}
colnames(sumtbl) <- c("Variable", "Class", "Min", "Median", "Mean", "SD", "Max",
                      "Num of NAs", "Num of Zeros", "Num of Neg Values")
pander(sumtbl[,1:7])
pander(sumtbl[,c(1,8:10)])
```

```
# Categorical variables
table(training$LabelAppeal)
table(training$AcidIndex)
table(training$STARS)

# Exploratory plots
mvariable <- "FixedAcidity"
mvariable <- "VolatileAcidity"
mvariable <- "CitricAcid"
mvariable <- "ResidualSugar"
mvariable <- "Chlorides"
mvariable <- "FreeSulfurDioxide"
mvariable <- "TotalSulfurDioxide"
mvariable <- "Density"
mvariable <- "pH"
mvariable <- "Sulphates"
mvariable <- "Alcohol"
mvariable <- "LabelAppeal"
mvariable <- "AcidIndex"
mvariable <- "STARS"
mvariable <- "Chlorides"
ploting.data <- as.data.frame(cbind(training[, mvariable], training$TARGET)); colnames(ploting.data) <-
box.plot <- ggplot(ploting.data, aes(x = 1, y = X)) + stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())
histogram.plot <- ggplot(ploting.data, aes(x = X)) + geom_histogram(aes(y=..density..), bins=50, colour=
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(X, na.rm=TRUE)), color="blue", linetype="dashed", size=1)
scotter.plot <- ggplot(ploting.data, aes(x=X, y=Y)) + geom_point() + xlab("Scatterplot") + ylab("")
box.plot.target <- ggplot(training, aes(x = as.factor(TARGET), y = Chlorides)) + stat_boxplot(geom ='err
  xlab("Boxplots per Target (Number of training Cases)") + ylab("Chlorides") + theme(axis.ticks.x=elemen
grid.arrange(box.plot, histogram.plot, scotter.plot, box.plot.target, layout_matrix=rbind(c(1,2,2),c(1,3


# Correlation matrix
Correlation.matrix <- cor(training[,2:16], use="pairwise.complete.obs")
Correlation.matrix <- round(Correlation.matrix, 2)
rownames(Correlation.matrix)[7:8] <- c("FreeSO2", "TotalSO2")
colnames(Correlation.matrix)[7:8] <- c("FreeSO2", "TotalSO2")
Correlation.matrix.out <- as.data.frame(Correlation.matrix) %>% mutate_all(function(z) {
  cell_spec(z, "latex", color = ifelse(z>0.5 | z<(-0.5),"blue","black"))
})
rownames(Correlation.matrix.out) <- colnames(Correlation.matrix.out)
Correlation.matrix.out %>%
  kable("latex", escape = F, align = "c", row.names = TRUE) %>%
  kable_styling("striped", full_width = F, font_size = 6) %>%
  row_spec(0, angle = 90)


# Dependent variable
outcome <- as.data.frame(table(training$TARGET))
outcome <- cbind(outcome, as.data.frame(round(table(training$TARGET)/sum(table(training$TARGET)),2))[,2]
colnames(outcome) <- c("Outcome", "# of Observations", "Percent of Total")
```

```
pander(outcome)


# IMPUTATION / TRANSFORMATION
plot_missing(training)

# Imputation
training$STARS[is.na(training$STARS)] <- 0
training$Alcohol <- abs(training$Alcohol)
trainingImputed <- mice(training, m=5, maxit=10, meth='norm', seed=500)
training <- complete(trainingImputed)

# Split into train and test sets
set.seed(88)
split <- sample.split(training$TARGET, SplitRatio = 0.75)
training.TRAIN <- subset(training, split == TRUE)
training.TEST <- subset(training, split == FALSE)


# LINEAR MODEL

# All variables
lmModel <- lm(TARGET ~ .-INDEX,data = training.TRAIN)
summary(lmModel)


# stepAIC
lmModel <- stepAIC(lmModel, trace=FALSE, direction='both')
summary(lmModel)
# Model returned by step AIC
lmModel <- lm(TARGET ~ VolatileAcidity + CitricAcid +
                Chlorides + FreeSulfurDioxide +
                TotalSulfurDioxide + Sulphates + Alcohol +
                LabelAppeal + AcidIndex + STARS,
              data = training.TRAIN)
summary(lmModel)
# Manual variations
lmModel <- lm(TARGET ~ VolatileAcidity + Chlorides +
                FreeSulfurDioxide +
                TotalSulfurDioxide + Sulphates + Alcohol +
                LabelAppeal + AcidIndex + STARS,
              data = training.TRAIN)
summary(lmModel)
lmModel <- lm(TARGET ~ VolatileAcidity + Chlorides +
                FreeSulfurDioxide +
                TotalSulfurDioxide + Alcohol +
                LabelAppeal + AcidIndex + STARS,
              data = trainingTRAIN)
summary(lmModel)

# Calculate RMSE
pred <- predict(lmModel, newdata=training.TEST)
rmse(training.TEST$TARGET, pred)
```

```r
# Confusion matrix
predRound <- as.factor(round(pred,0))
table(predRound)
levels(predRound) <- levels(as.factor(training.TEST$TARGET))
confusionMatrix(predRound, as.factor(training.TEST$TARGET))

autoplot(lmModel)

# Model plots
plot(lmModel$residuals, ylab="Residuals")
abline(h=0)

plot(lmModel$fitted.values, lmModel$residuals,
     xlab="Fitted Values", ylab="Residuals")
abline(h=0)

qqnorm(lmModel$residuals)
qqline(lmModel$residuals)

# POISSON and NB REGRESSION MODEL

# Poisson 1
poisson.Model <- glm (TARGET ~ .-INDEX, data = training.TRAIN, family = poisson)
summary(poisson.Model)
pred <- predict(poisson.Model, newdata=training.TEST, type='response')
rmse(training.TEST$TARGET, pred)
predRound <- as.factor(round(pred,0))
testData <- as.factor(training.TEST$TARGET)
levels(predRound) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "0")
levels(testData) <- c("0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10")
confusionMatrix(predRound, testData)

# Poisson 2
poisson.Model2 <- stepAIC(poisson.Model, trace=FALSE, direction='both')
summary(poisson.Model2)
pred <- predict(poisson.Model2, newdata=training.TEST, type='response')
rmse(training.TEST$TARGET, pred)
predRound <- as.factor(round(pred,0))
testData <- as.factor(training.TEST$TARGET)
levels(predRound) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "0")
levels(testData) <- c("0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10")
confusionMatrix(predRound, testData)

# Poisson 3
poisson.Model3 <- glm(TARGET ~ VolatileAcidity + Chlorides + FreeSulfurDioxide +
                  Sulphates + Alcohol + LabelAppeal +
                  AcidIndex + STARS, family = poisson, data = training.TRAIN)
summary(poisson.Model3)
pred <- predict(poisson.Model3, newdata=training.TEST, type='response')
rmse(training.TEST$TARGET, pred)
predRound <- as.factor(round(pred,0))
testData <- as.factor(training.TEST$TARGET)
levels(predRound) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "0")
levels(testData) <- c("0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10")
confusionMatrix(predRound, testData)

# NB
negativeBinomial.Model <- glm.nb(TARGET ~ .-INDEX, data = training.TRAIN)
summary(negativeBinomial.Model)
pred <- predict(negativeBinomial.Model, newdata=training.TEST, type='response')
rmse(training.TEST$TARGET, pred)
predRound <- as.factor(round(pred,0))
testData <- as.factor(training.TEST$TARGET)
levels(predRound) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "0")
levels(testData) <- c("0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10")
confusionMatrix(predRound, testData)

# Zero Inflated
ZeroInflated.Model <- zeroinfl(TARGET ~ .-INDEX, data = training.TRAIN, dist = "negbin")
summary(ZeroInflated.Model)
pred <- predict(ZeroInflated.Model, newdata=training.TEST, type='response')
rmse(training.TEST$TARGET, pred)
predRound <- as.factor(round(pred,0))
testData <- as.factor(training.TEST$TARGET)
confusionMatrix(predRound, testData)

# Deviance residuals
anova(poisson.Model, test="Chisq")
anova(poisson.Model2, test="Chisq")
anova(poisson.Model3, test="Chisq")
anova(zrModel, test="Chisq")

# VIF
vif(poisson.Model)
vif(negativeBinomial.Model)
vif(ZeroInflated.Model)

# Coefficients
coef <- as.data.frame(lmModel$coefficients)
coef <- cbind(coef, as.data.frame(poisson.Model$coefficients))
coef <- cbind(coef, as.data.frame(negativeBinomial.Model$coefficients))
coef <- cbind(coef, as.data.frame(ZeroInflated.Model$coefficients))

# Prediction
eval <- read.csv("https://raw.githubusercontent.com/theoracley/Data621/master/Homework5/wine-evaluation-
colnames(eval)[1] <- "INDEX"

sumtbl = data.frame(Variable = character(),
                    Class = character(),
                    Min = integer(),
                    Median = integer(),
                    Mean = double(),
                    SD = double(),
                    Max = integer(),
                    Num_NAs = integer(),
                    Num_Zeros = integer(),
                    Num_Neg = integer())
for (i in c(3:16)) {
  sumtbl <- rbind(sumtbl, data.frame(Variable = colnames(eval)[i],
                                     Class = class(eval[,i]),
                                     Min = min(eval[,i], na.rm=TRUE),
                                     Median = median(eval[,i], na.rm=TRUE),
                                     Mean = mean(eval[,i], na.rm=TRUE),
                                     SD = sd(eval[,i], na.rm=TRUE),
                                     Max = max(eval[,i], na.rm=TRUE),
                                     Num_NAs = sum(is.na(eval[,i])),
                                     Num_Zeros = length(which(eval[,i]==0)),
                                     Num_Neg = sum(eval[,i]<0 & !is.na(eval[,i]))))
}
colnames(sumtbl) <- c("Variable", "Class", "Min", "Median", "Mean", "SD", "Max",
                      "Num of NAs", "Num of Zeros", "Num of Neg Values")
sumtbl

eval$STARS[is.na(eval$STARS)] <- 0
eval$Alcohol <- abs(eval$Alcohol)

evalImputed <- mice(eval, m=5, maxit=10, meth='norm', seed=500)
eval <- complete(evalImputed)

pred <- predict(ZeroInflated.Model, newdata=eval, type="response")
results <- eval[, c("INDEX")]
results <- cbind(results, prob=round(pred,4))
results <- cbind(results, predict=round(pred,0))
colnames(results) <- c("Index", "Predicted Value", "Predicted Outcome")
pander(head(results, 100))

#Write the results to a Prediction file
write.csv(results, "Prediction_For_Eval.csv")
```