STA 141A - Final Project

Wine(White) Quality Report

DECEMBER 13TH, 2019

Task Distribution:

Xin Ye: Introduction, Data Analysis Plan, Visualization, Linear Regression, Diagnostic, Conclusion
Yutian Yang: Linear Regression, Classification, Visualization, Conclusion
Michelle Chen: Introduction, Visualization, Formatting

Ⅰ Introduction

1.1 Background

In recent years, consumers' interest in wine has increased. More and more companies are investing in new technologies aiming to promote wine production and marketing. Among all, Controlling quality of wine is one of the key strategies that the wine industry focus on. Therefore, quality certification and assessment of the wine become significant steps for the market to evaluate and improve wine quality. The wine industry need to predict human wine taste preferences that are based on easily available analytical tests at the certification step. A large dataset is considered, with white vinho verde samples, from Portugal.

1.2 Our goal

In this project, we are interested in building a model to predict white wine quality with the most significant variables. We analyze the white vinho verde wine quality data of the North of Portugal with multinomial regression and classification methods. Different techniques would be applied, under a computationally efficient procedure that performs simultaneous variable and model selection. We would pick the best model to support the oenologist wine tasting evaluations and improve wine production.

Ⅱ Data Analysis Plan

The primary interest in this project is to build a predictive model for wine quality. Therefore, determine which several factors will have the most significant impact on wine quality becomes our first task. Moreover, we are interested in if there are interactions between our variables. To solve these problems, we manually implemented the model selection methods, such as Forward and Backward Selection by recording the Akaike Information Criteria(AIC) and Bayesian Information Criteria(BIC) results to avoid overfitting or underfitting. We finally choose the best linear regression model with the reasonable adjusted R^2 and least AIC and BIC results to ensure our model includes the most significant variables.

According to the best model we selected, we add appropriate interaction terms according to the corrogram, correlations between among significant variables and F-test to finalize our linear regression model. Then, we fit our finalized linear model to linear discriminant analysis(LDA) and multinomial logistic regression(MLR). Using test-training split methods with the ratio of 2:8, we validate our prediction model based on the accuracy of prediction. Meanwhile, we use k Nearest-Neighbour(kNN) classification, the non-parametric method, to gain a high accuracy model since the sample size is sufficiently large. Although we may not gain any practical inference from this model, we may use the results to compare with our finalized model as well as provide the wine industry a good-to-go classification model with the optimal k value to directly use.

2.1 Data Summary

The white wine quality data describes 4898 wine samples and 12 features of wine quality measurements, all obtained in the 2009s by Paulo Cortez. Due to privacy and logistic issues, only physicochemical and sensory variables are available to sample during the wine manufacturer process. Thus, we take the 11 physicochemical variables as our input and choose the discrete wine quality score as our output for our model and all of them are numerical.

Attributes**:**

Physicochemical Inputs	units
Fixed acidity	g(tartaric acid)/dm^3
Volatile acidity	g(acetic acid)/dm^3
Citric acid	g/dm^3
Residual sugar	g/dm3
Chlorides	g(sodium chloride)/dm3
Free sulfur dioxide	mg/dm3
Total sulfur dioxide	mg/dm3
Density	g/cm3
pH	\
Sulphates	g(potassium sulphate)/dm3
Alcohol	vol.%

Table 1

Output**:**

Wine Quality: discretely range from 0 to 10

2.2 Data Visualization

We first looked at the pairwise correlations between each variable.

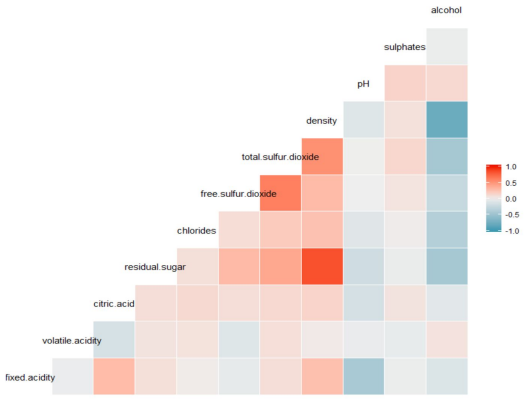

Figure 1: Corrogram

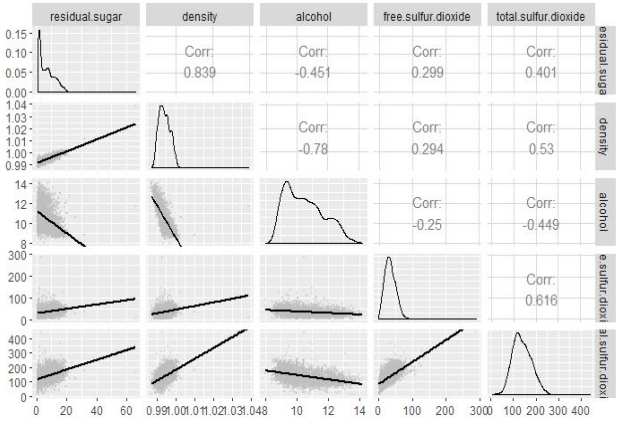

Figure 2: Pairwise Correlations

In this step, we were interested in finding the relationship between each feature. By creating the Corrogram for 12 observation variables, see Figure 1, we were able to identify their correlation value via opacity of the color. Thus, we found the density vs. residual sugar has the largest positive correlation and density vs. alcohol has the largest negative correlation(show high opacity of red color and high opacity of blue color). Thus, we assumed that there may be an interaction between density and residual sugar as well as density and alcohol to be assessed in our regression part.

To take a further look at detailed pairwise correlations among these three variables, we created a Pairwise Correlations graph. From Figure 2, we observed that all correlations among engaged products, wine features that are dense with residual sugar and dense with alcohol are near to 1, which shows strong positive linear relationships.

Ⅲ Main Analysis

3.1 Regression and Model Selection

3.1.1 Regression Model with Individual Variable

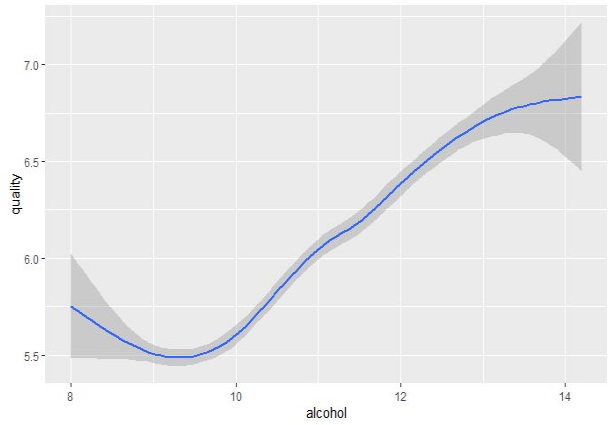

Figure 3: Quality vs Alcohol using "auto" smoother

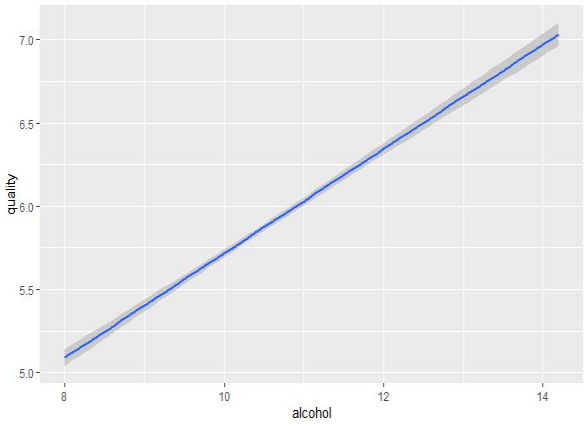

Figure 4: Quality vs Alcohol using "lm" smoother

At this part, we analyze and find the most important variable for wine quality. We first applied each variable individually into the linear regression model to find their relationship with quality. From linear regression models with single variable, we conclude that alcohol is the most important variable for wine quality because it gives the highest r square (0.144).

Based on that, we applied all variable without adding interaction into the model, we found out that citric acid, chlorides, and total sulfur dioxide are not significant, so we eliminated them from the regression model.

3.1.2 Model Selection

Then, we manually implemented the Forward and Backward Selection. We start with an empty model by adding one variable to the model and record the Akaike Information Criteria(AIC) and Bayesian Information Criteria(BIC) results. We then keep adding until the AIC and BIC results no longer decreases and keep the model. Then we start with a full model by deleting one variable from the model and also stop until the AIC and BIC results no longer decreases and keep the model. Finally,we choose the same linear regression model drawn from both model selection results.

wine.lm1 = lm(quality ~ fixed.acidity + volatile.acidity + citric.acid + residual.sugar + chlorides + free.sulfur.dioxide +total.sulfur.dioxide + density + pH + sulphates + alcohol , data = `winequality.white`) (AIC = 11108.29, BIC = 11173.25, R_{adj}^2 = 28.06%)

3.1.3 Interaction Model

According to our corrogram, we noticed that there are correlations between alcohol and density, density and residual sugar, and density and total sulfur dioxide so we added them as interaction terms in our model.

wine.lm2 = lm(quality ~ fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide + density + pH + sulphates + alcohol + alcohol*density + density*residual.sugar +density*total.sulfur.dioxide, data = `winequality.white`)

3.1.4 Diagnostics for the final model

This is the final prediction model we drawn from our analysis. Therefore, we use Shapiro-Wilks test and Non-constant Variance test and find that both of the assumptions are violated. Unfortunately, after trying both box-cox transformation(using λ=0.946) and bootstrap, we still got the same results.While the Lasso Regression can be used even if the assumptions are violated, it is not very good at handling variables which show correlation between them. Therefore, we may need to learn more advanced techniques in the future to handle this situation. For now, We decide to continue to use the final model to complete the following classification task by using LDA and Multinomial Logistic Regression.

3.2 Classification

We use the model in regression part to conduct a classification task in order to predict wine quality. We used three methods: Linear Discrimination Analysis, Multinomial Logistics Regression, and k Nearest-Neighbor to predict and compare each method.

We first divided quality into three general category, and use 0 to represent “low” quality wine, 1 to represent “fair” quality wine, and 2 to represent “high” quality wine. Quality 0 contains original data of wine quality within 1 to 4. Quality 1 contains original data of wine quality with 5 to 6. Finally, quality 2 contains original data of wine quality within 7 to 9. We use training-test split methods, taking 80% of each quality level to be the training dataset and the remaining 20% data to to be the test dataset.

3.2.1 Linear Discrimination Analysis (LDA)

We used the model in our regression part as a prediction model to train our data. From this model, we got the following confusion matrix:

Pred True	0	1	2
0	1	4	1
1	24	830	78
2	0	31	11

Table 2: LDA with final predictive model

The error rate of using LDA is about 0.14, which means 86% of the test data were correctly predicted. However, this method requires the assumption of normal distribution since it is based on the linear regression model. Our dataset appears to violate some of the assumptions so we did not expect this model to be very accurate.

3.2.2 Multinomial Logistic Regression

We used the model in our regression part once again as the predict model. However, instead of using binomial logistic classification, we conducted a multinomial logistic classification and got the following confusion matrix:

Pred True	0	1	2
0	0	5	1
1	5	859	68
2	0	29	13

Table 3: MLR with final predictive model

The error rate using multinomial logistic regression is 0.11, which means 89% of our predictions are correct. As we using multinomial regression instead of linear regression to predict our data, the model is performing better just as we expected.

3.2.3 kNN

We have tested multiple k's in order to find the best k. From k=10 to k=100, we found out that k=70 is the best k since it gives the lowest error rate.

Pred True	0	1	2
0	0	6	0
1	0	896	36
2	0	35	7

Table 4 (kNN model with k=70)

As we expect, the kNN model gives the lowest error rate of 0.08, which means 92% of our test data were correctly predicted. Comparing to other k's, we believe larger k's gives better prediction because the size of our test dataset is relatively large. However, when k is too large, the bias increases, therefore our predict model might not be accurate. From that, we think k=70 is the best k.

Ⅳ Conclusion

Our study provides an effective predictive model with multiple important variables to predict wine quality of white wine. To analyze the most important variable that affects the quality of white wine, we concluded that alcohol is the most important variable in the model. By adding the rest of the variables into the model, we found out that citric acid, chlorides, and total sulfur dioxide are not significant, so we removed them from the regression model. After looking at our corrogram (figure 1), it suggests that alcohol and density, density and residual sugar, and density and total sulfur dioxide are highly correlated, so we added them as interaction terms in our model.

The results of our study may offer a practical inference for the wine industry. Nowadays, wine evaluation tests are mostly based on a human's subjective sensory judgement, which might be biased varying among people. Our model provides the possibility of improving the accuracy of the test. With more of the sample data to be analyzed, our model may have a potential possibility to generally replace some human based tests by simply analyze the physicochemical data of the wine. From economy perspective, it may solve the problem of the scarcity of oenology experts and improve the economic efficiency for wine industry.

Ⅴ References

A. Asuncion, D. Newman
UCI Machine Learning Repository, University of California, Irvine (2019)
https://archive.ics.uci.edu/ml/datasets/Wine+Quality

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.
Modeling wine preferences by data mining from physicochemical properties
https://www.sciencedirect.com/science/article/pii/S0167923609001377?via%3Dihub

Ⅵ R Appendix

```
library(car)
library(ggplot2)
winequality.white <- read.csv("C:/Users/838748635/Downloads/winequality-white.csv", sep=";")
attach(winequality.white)
table(`winequality.white`$quality)
wine_low_index = which(winequality.white$quality <= 4)
wine_low = winequality.white[wine_low_index,]
wine_low[,"quality"] = 0
wine_high_index = which(winequality.white$quality >6) #including index with mid and low quality
wine_high = winequality.white[wine_high_index,]
wine_high[,"quality"] = 2
wine_mid = winequality.white[-c(wine_low_index,wine_high_index),]
wine_mid[,"quality"] = 1
new_winequality.white = rbind(wine_low,wine_mid,wine_high)
wine_mid_index = which(new_winequality.white$quality == 1)
pairs(~ fixed.acidity + volatile.acidity + residual.sugar + free.sulfur.dioxide + density + pH + sulphates +
alcohol+quality, data = `new_winequality.white`, main = " pair plot")
# individual r square:
# fixed.acidity: 0.007839
# volatile.acidity: 0.004935
# citric.acid: 0.001278
# residual.sugar: 0.01296
# chlorides: 0.03347
# free.sulfur.dioxide: 0.0009707
# total.sulfur.dioxide: 0.02728
# density: 0.07926
# pH: 0.008183
# sulphates: 0.002108
# alcohol: 0.144 comparing that
ggwine = ggplot(winequality.white,aes(log10(alcohol),density))
ggwine + geom_point(alpha=I(1/5)) +geom_smooth(method="lm")+ facet_wrap(quality)
#slope of better high alcohol is steeper
library(GGally)
ggpairs(winequality.white, c(4,8,11,6,7), mapping = ggplot2::aes(alpha = 0.5),
lower=list(continuous = wrap("smooth", alpha=0.3,size=0.2,color="grey")))
## ggplot using general smoother
ggplot(winequality.white,aes(alcohol,quality))+geom_smooth()
ggplot(winequality.white,aes(sulphates,quality))+geom_smooth(method="lm")
#apply all factor without interaction, citric acid, chlorides, total.sulfur.dioxide are not significant
wine.lm1 = lm(quality ~   fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide + density +
pH + sulphates + alcohol , data = `winequality.white`)
summary(wine.lm1)
plot(wine.lm1)
AIC(wine.lm1)
BIC(wine.lm1)
```

```
ei = wine.lm1$residuals
##use Shapiro-Wilks test to test the normality of the Linear Model
the.SWtest = shapiro.test(ei)
the.SWtest
## non-constant test to test the if our data has constant error
ncvTest(wine.lm1)
## Box-cox Transformation
library(EnvStats)
L1 = boxcox(wine.lm1 ,objective.name = "PPCC",optimize = TRUE)$lambda
L2 = boxcox(wine.lm1 ,objective.name = "Shapiro-Wilk",optimize = TRUE)$lambda
YT = (winequality.white$quality^(L2)-1)/L2
##Transformed data
t.data = cbind.data.frame(winequality.white,YT)
t.model = lm(YT ~ fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide + density + pH +
sulphates + alcohol ,data = t.data)
summary(t.model)
ei = t.model$residuals
##use Shapiro-Wilks test to test the normality of the transformed Model
the.SWtest = shapiro.test(ei)
the.SWtest
## non-constant test to test the if our data has constant error
ncvTest(t.model)
#add interaction based on corrogram
wine.lm2 = lm(quality ~   fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide + density +
pH + sulphates + alcohol + alcohol*density + density*residual.sugar +density*total.sulfur.dioxide, data =
`winequality.white`)
summary(wine.lm2)
plot(wine.lm1)
plot(wine.lm2)
ks.test(wine.lm1$residuals,rnorm(1000))
# Residual bootstrap (resample residuals)
bootresid = function(lmobj) {   # an lm object
  n = length(lmobj$resid)
  # Sample row numbers (i) rather than values (e_i)
  idx = sample(n, n, replace = TRUE)
  # Use row numbers to get new residuals (e_samp_i).
  res_samp = lmobj$resid[idx]
  # y_samp_i =  b_0 + b_1 * x_i    + e_samp_i
  y_samp =  lmobj$fitted     + res_samp
  # take x variable to be the original x variable
  x = lmobj$model[,2]
  # create the data frame for the resampled data
  data.boot = data.frame(x,y_samp)
  # Fit the same model with new data (y_samp_i, x_i).
  boot.lm = lm(y_samp ~ x, data.boot)

  return (coef(boot.lm))
}
```

```
# Boostrap distribution of the regression coefficients
N = 1000  # Number of bootstrap samples
# bootsrap samples of coefficients (b0_boot, b1_boot)
coef.boot = replicate(N, bootresid(wine.lm1))
coef.boot = data.frame(t(coef.boot))
colnames(coef.boot) = c("b0_boot","b1_boot")
#############
## plot the sampling distribution of (b0_boot,b1_boot),
## also known as the bootstrap distribution of (b0,b1)
library(ggplot2)
boot.pl = ggplot(data=coef.boot, aes(b0_boot,b1_boot))
# scatter plot together with the contour plot of the density
boot.pl + geom_point() + stat_density2d()
## Confidence intervals
## Bootstrap confidence interval
alpha = 0.05
## Theoretical confidence intervals based on Gaussian linear regression model
beta.conf.th = confint(wine.lm1,level=1-alpha)  # confidence coefficient is (1-alpha)
beta.conf.boot = sapply(1:2, function(i) quantile(coef.boot[,i],c(alpha/2,1-alpha/2)))
#express the confidence intervals in an easily interpretable format
beta.conf.boot = t(beta.conf.boot)
rownames(beta.conf.boot) = c("quality_boot","alcohol_boot")
#classification
library(MASS)
set.seed(100)
test_low_quality_idx = sample(wine_low_index,round(0.2*(length(wine_low_index))))
test_mid_quality_idx = sample(wine_mid_index,round(0.2*(length(wine_mid_index))))
test_high_quality_idx = sample(wine_high_index,round(0.2*(length(wine_high_index))))
test_idx  = c(test_low_quality_idx,test_mid_quality_idx,test_high_quality_idx)
testdata = new_winequality.white[test_idx,]
traindata = new_winequality.white[-test_idx,]

winequality.lda = lda(quality~fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide +
density + pH + sulphates + alcohol + alcohol*density + density*residual.sugar
+density*total.sulfur.dioxide,traindata)
winequality.lda
winequality.lda$means

winequality.pred.lda = predict(winequality.lda,testdata,type = "response")
winequality.confusion.lda = table(true = testdata$quality, predicted = winequality.pred.lda$class)
winequality.confusion.lda

winequality.pred.lda_error = (winequality.confusion.lda[1,2] +winequality.confusion.lda[1,3]
+winequality.confusion.lda[2,1]+winequality.confusion.lda[2,3]+winequality.confusion.lda[3,1]+winequa
lity.confusion.lda[3,2])/sum(winequality.confusion.lda)
winequality.pred.lda_error
#classification
library(nnet)
```

```
winequality.glm = multinom(quality~fixed.acidity + volatile.acidity + residual.sugar +free.sulfur.dioxide
+ density + pH + sulphates + alcohol + alcohol*density + density*residual.sugar
+density*total.sulfur.dioxide, traindata)
winequality.glm
summary(winequality.glm)
winequality.pred.glm = predict(winequality.glm, testdata, type = "class")
winequality.glm.con = table(true = testdata$quality, model = winequality.pred.glm)
winequality.glm.con
winequality.pred.glm_error = (winequality.glm.con[1,2] +winequality.glm.con[1,3]
+winequality.glm.con[2,1]+winequality.glm.con[2,3]+winequality.glm.con[3,1]+winequality.glm.con[3,2
])/sum(winequality.confusion.lda)
winequality.pred.glm_error
set.seed(100)
library(class)

knn_pred1 = knn(
  train =
traindata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphat
es","alcohol")],
  test  =
testdata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphate
s","alcohol")],
  cl    = traindata$quality,
  k     = 30
)
# Confusion matrix.
knn_con1 = table(true = testdata$quality, model = knn_pred1)
knn_con1
#Misclassification error rate
winequality.pred.knn_error1 = (knn_con1[1,2] +
knn_con1[1,3]+knn_con1[2,1]+knn_con1[2,3]+knn_con1[3,1]+knn_con1[3,2])/sum(knn_con1)
winequality.pred.knn_error1

knn_pred2 = knn(
  train =
traindata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphat
es","alcohol")],
  test  =
testdata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphate
s","alcohol")],
  cl    = traindata$quality,
  k     = 50
)
# Confusion matrix.
knn_con2 = table(true = testdata$quality, model = knn_pred2)
knn_con2
#Misclassification error rate
```

```
winequality.pred.knn_error2 = (knn_con2[1,2] +
knn_con2[1,3]+knn_con2[2,1]+knn_con2[2,3]+knn_con2[3,1]+knn_con2[3,2])/sum(knn_con2)
winequality.pred.knn_error2

knn_pred3 = knn(
  train =
traindata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphat
es","alcohol")],
  test  =
testdata[c("fixed.acidity","volatile.acidity","residual.sugar","free.sulfur.dioxide","density","pH","sulphate
s","alcohol")],
  cl    = traindata$quality,
  k     = 70
)
# Confusion matrix.
knn_con3 = table(true = testdata$quality, model = knn_pred3)
knn_con3
#Misclassification error rate
winequality.pred.knn_error3 = (knn_con3[1,2] +
knn_con3[1,3]+knn_con3[2,1]+knn_con3[2,3]+knn_con3[3,1]+knn_con3[3,2])/sum(knn_con3)
winequality.pred.knn_error3
```