STAT 517 FINAL PROJECT WINE QUALITY PREDICTION

Yiqun Hu

$December 7, 2019$

***Abstract*—The project is focusing on the wine quality dataset, using different regression models to fit the dataset, trying to find out how the outcome is affected by the predictors, and which model fits the data best by comparing the test MSE. By fitting the models, can conclude how accurate my prediction is. The original dataset is available at https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/.**

I. INTRODUCTION

This project focuses on the wine quality dataset from the UCI machine learning depository. Two sets of datasets are included, white wine and red wine. White wine dataset is being used here in this project, since it contains more observations. This dataset has 11 predictors and one response variable.

The response variable I chose from the dataset is the quality of wine, it is a score between 0 and 10. 0 means poor wine quality, whereas 10 means good wine quality. The predictors are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol. They are all numerical, and are obtained from lab test results. Here are some explanations.

Fixed acidity: most acids involved with wine or fixed or nonvolatile (do not evaporate readily).

Volatile acidity: the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste.

Citric acid: the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste.

Residual sugar: the amount of sugar remaining after fermentation stops, it's rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet.

Chlorides: the amount of salt in the wine.

Free sulfur dioxide: the free form of SO_2 exists in equilibrium between molecular SO_2 (as a dissolved gas) and bisulfite ion; it prevents microbial growth and the oxidation of wine.

Total sulfur dioxide: amount of free and bound forms of SO_2; in low concentrations, SO_2 is mostly undetectable in wine, but at free SO_2 concentrations over 50 ppm, SO_2 becomes evident in the nose and taste of wine.

Density: the density of water is close to that of water depending on the percent alcohol and sugar content.

PH: describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic); most wines are between 3-4 on the pH scale.

Sulphates: a wine additive which can contribute to sulfur dioxide gas (SO_2) levels, wich acts as an antimicrobial and antioxidant.

Alcohol: the percent alcohol content of the wine.

This project is to predict the wine quality based on these predictors using statistical learning methods. I am interested in the questions that which predictors are significant to predict wine quality, how the outcome is affected by the predictors, and how well the models fit the data.

II. METHODOLOGY

I did a basic analysis on the dataset. Here is the distribution of the wine quality. Clearly, most wine falls in to level 5, 6, and 7. So, when I predict wine quality, there is less chance to get poor wine quality or good wine quality.

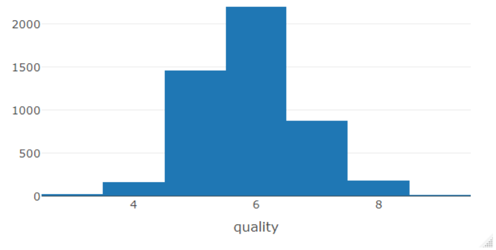


Fig. 1: Distribution of the wine quality.

Meanwhile, I also did some boxplots and tried to figure out if there is any relationship between the quality and predictors. Since there is no trend on the median, no obvious relationship was observed.

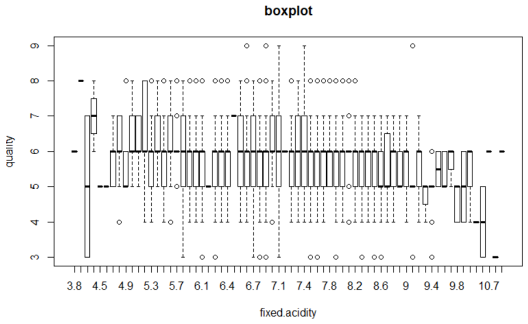


Fig. 2: Boxplot: quality vs fixed acidity

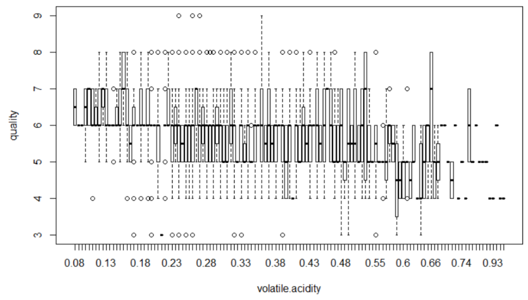


Fig. 3: Boxplot: quality vs volatile acidity .

III. Regression Methods

Linear Regression:

First of all, I would like to fit the data with all predictors. By fitting the linear regression model, we may encounter some problems like correlation of error terms, outliers, collinearity between predictors, or the model does not work if the function is highly non-linear. Even though this method is almost never correct, it is the one I want to start my project with, to give me a rough picture on this unknown function.

Linear regression, also known as ordinary least squares(OLS), is the approach for predicting a quantitative response Y based on predictor variable X's. It assumes that there is approximately a linear relationship between X's and Y. By minimizing residual sum of squares, it helps with estimating coefficients.

Ridge and Lasso:

Ridge regression and Lasso regression are approaches based on OLS, adding a shrinkage penalty term. They both use cross validation to figure out the best tuning parameter λ. When collinearity occurs, OLS does not work well due to the high variance. Then, we need to use Ridge or Lasso method to improve the model.

Ridge regression has the computation advantage because for any given λ, only one model needs to be fit. Lasso regression outperforms Ridge when there is a small number of predictors.

Regression Tree:

A regression tree is built through a process known as binary recursive partitioning, which is an iterative process that splits the data into partitions or branches, and then continues splitting each partition into smaller groups as the method moves up each branch.

The regression tree is easy to interpret and has a nice graphical representation. It makes prediction fast, and easy to understand what variables are important in making the prediction. Decision tree method does not work better than ordinary least squares, however, it gives us the most important predictor by looking at the top of the tree.

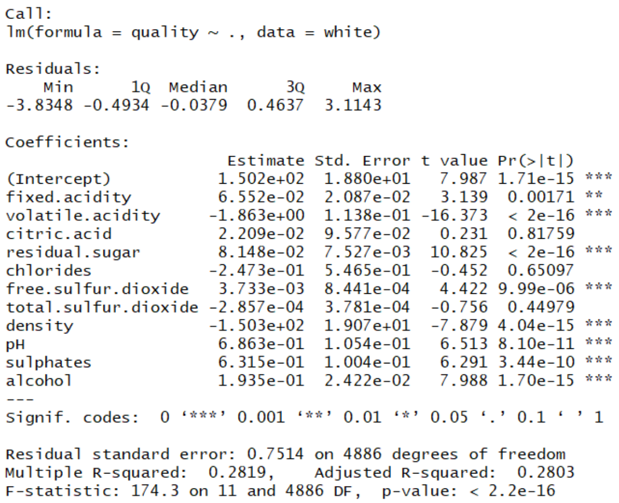
Random Forest:

Bootstrap aggregation, or bagging, is a general-purpose procedure for reducing the variance of a statistical learning method; we introduce it here because it is particularly useful and frequently used in the context of decision trees. Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees. This reduces the variance when we average the trees.

IV. Data Anlysis

Linear Regression:

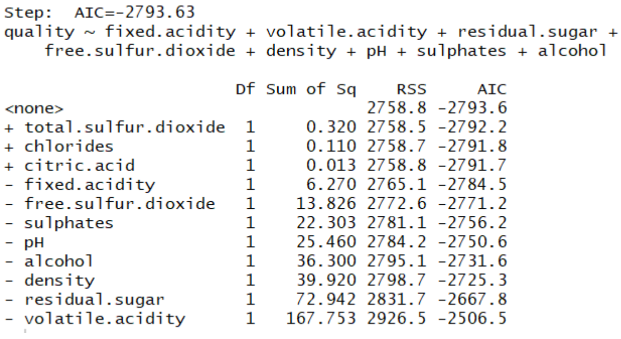
Here is the summary after fitting the model. As we can see that there are three predictors are significant, since their p-values are less than 0.05. The predictors are citric acid, chlorides, and total sulfur dioxide. Adjusted R-square is 0.2803 means only 28.03 percent of variance has been explain, which is not a high percentage.



```
Call:
lm(formula = quality ~ ., data = white)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.8348 -0.4934 -0.0379  0.4637  3.1143 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)           1.502e+02  1.880e+01   7.987 1.71e-15 ***
fixed.acidity         6.552e-02  2.087e-02   3.139  0.00171 ** 
volatile.acidity     -1.863e+00  1.138e-01 -16.373  < 2e-16 ***
citric.acid           2.209e-02  9.577e-02   0.231  0.81759    
residual.sugar        8.148e-02  7.527e-03  10.825  < 2e-16 ***
chlorides            -2.473e-01  5.465e-01  -0.452  0.65097    
free.sulfur.dioxide   3.733e-03  8.441e-04   4.422 9.99e-06 ***
total.sulfur.dioxide -2.857e-04  3.781e-04  -0.756  0.44979    
density              -1.503e+02  1.907e+01  -7.879 4.04e-15 ***
pH                    6.863e-01  1.054e-01   6.513 8.10e-11 ***
sulphates             6.315e-01  1.004e-01   6.291 3.44e-10 ***
alcohol               1.935e-01  2.422e-02   7.988 1.70e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7514 on 4886 degrees of freedom
Multiple R-squared:  0.2819,	Adjusted R-squared:  0.2803 
F-statistic: 174.3 on 11 and 4886 DF,  p-value: < 2.2e-16
```

Fig. 4: Linear Regression Output

After ran a step-wise selection from both directions, forward and backward. The final model obtained contained eight predictors, three were eliminated. They are total sulfur dioxide, chlorides, citric acid, which agrees with the previous results.



```
Step:  AIC=-2793.63
quality ~ fixed.acidity + volatile.acidity + residual.sugar + 
    free.sulfur.dioxide + density + pH + sulphates + alcohol

                       Df Sum of Sq    RSS     AIC
<none>                              2758.8 -2793.6
+ total.sulfur.dioxide  1     0.320 2758.5 -2792.2
+ chlorides             1     0.110 2758.7 -2791.8
+ citric.acid           1     0.013 2758.8 -2791.7
- fixed.acidity         1     6.270 2765.1 -2784.5
- free.sulfur.dioxide   1    13.826 2772.6 -2771.2
- sulphates             1    22.303 2781.1 -2756.2
- pH                    1    25.460 2784.2 -2750.6
- alcohol               1    36.300 2795.1 -2731.6
- density               1    39.920 2798.7 -2725.3
- residual.sugar        1    72.942 2831.7 -2667.8
- volatile.acidity      1   167.753 2926.5 -2506.5
```

Fig. 5: Final Linear Regression Output

Furthermore, a correlation table may could help to recognize relationship and collinearity between quality and predictors.

From the table, quality has no strong relationship with citric acid, free sulfur, dioxide and sulphates, which doesn't match the results we got earlier.

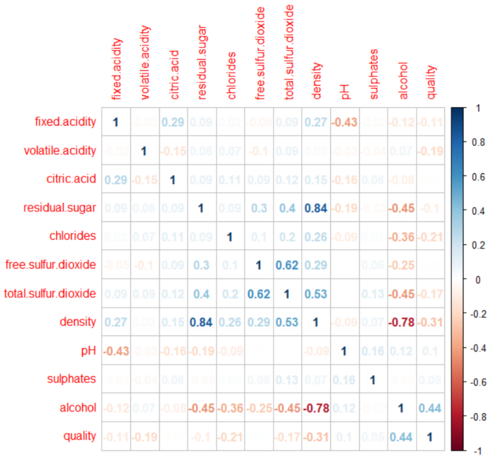


Fig. 6: Correlation Table between quality and predictors

KNN:

For k-nearest neighbours, 5 kmax, 2 distance, and 3 kernel values will be used. For the distance value, 1 is the Manhattan distance, and 2 is the Euclidian distance.

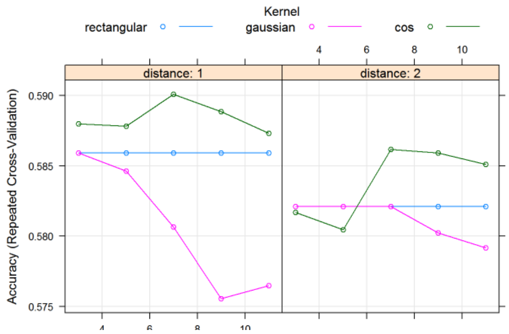
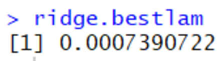


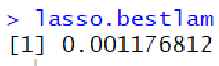
Fig. 7: KNN(k-nearest neighbours)

The cos kernel using a distance of 1 outperforms the alternatives. The best value for k is 7.

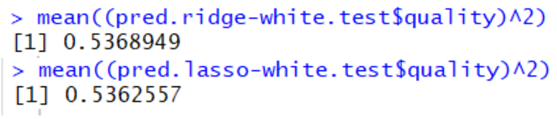
Ridge and Lasso:



```
> ridge.bestlam
[1] 0.0007390722
```



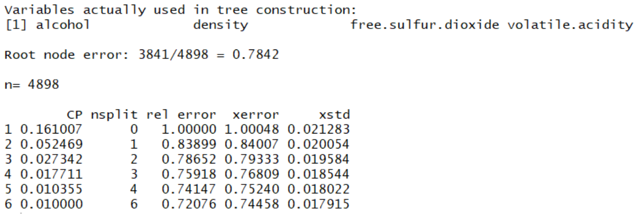
```
> lasso.bestlam
[1] 0.001176812
```



```
> mean((pred.ridge-white.test$quality)^2)
[1] 0.5368949
> mean((pred.lasso-white.test$quality)^2)
[1] 0.5362557
```

Based the tuning parameters, these test MSE were obtained, which are really close. Comparing to test MSE of OLS, there is not much difference. Therefore, three models function similar.

Tree:



```
Variables actually used in tree construction:
[1] alcohol              density              free.sulfur.dioxide volatile.acidity

Root node error: 3841/4898 = 0.7842

n= 4898

        CP nsplit rel error  xerror     xstd
1 0.161007      0   1.00000 1.00048 0.021283
2 0.052469      1   0.83899 0.84007 0.020054
3 0.027342      2   0.78652 0.79333 0.019584
4 0.017711      3   0.75918 0.76809 0.018544
5 0.010355      4   0.74147 0.75240 0.018022
6 0.010000      6   0.72076 0.74458 0.017915
```

Fig. 8: Tree Output

Based on the table, four variables are used to grow the tree: alcohol, density, free sulfur dioxide and volatile acidity. The following graph shows the size of the tree.

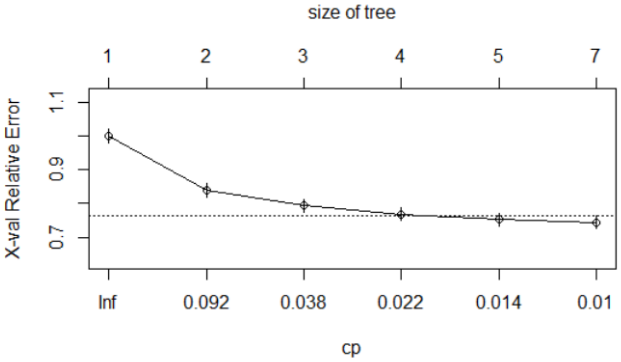


Fig. 9: Tree Output

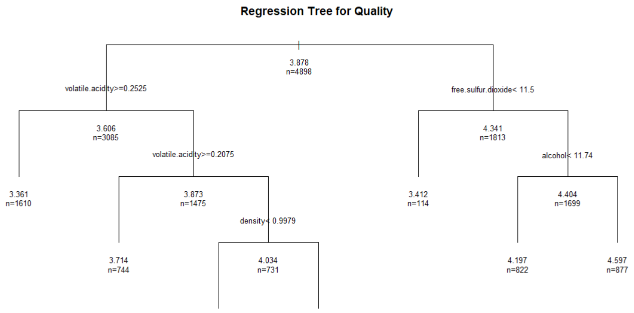


Fig. 10: Regression Tree

The tree indicates that alcohol is the most important predictors since it is the first split. It makes sense that we are predicting the quality of wine.

The test MSE of regression tree is 0.5546. Not much difference can be seen from previous methods. Therefore, there is no evidence shows which approach works better.

Random Forest:

An mtry of 1 is like using univariate decision trees. I ran the train function again using an ntree of 1000 to see if the result of 1 would stick, and it did. So, I'm going to keep mtry = 1 as the best value.

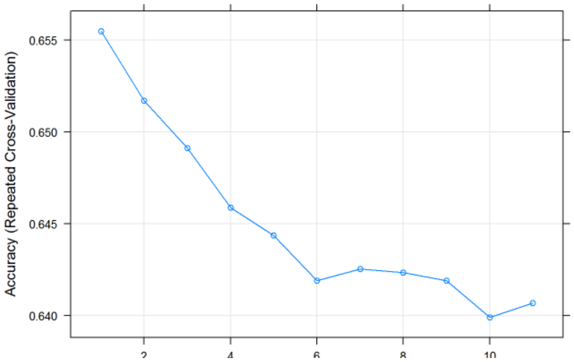
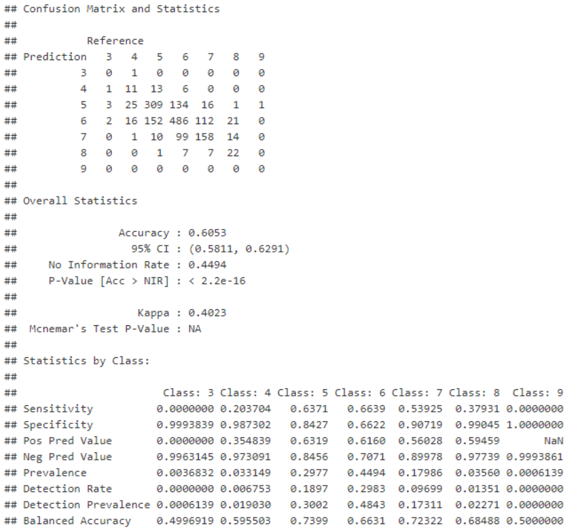


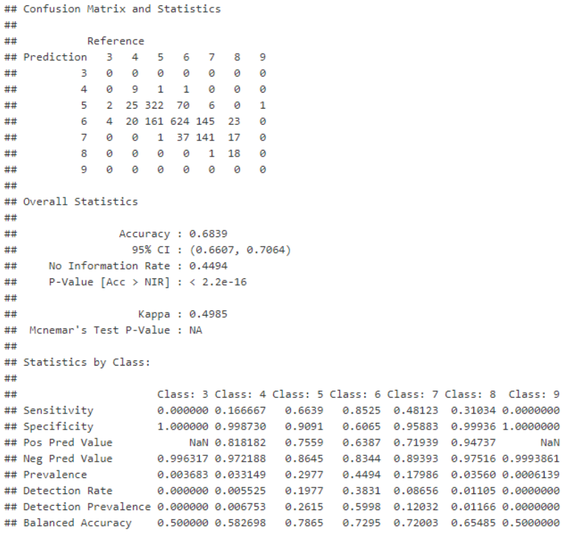
Fig. 11: Random Forest

V. Model Selection



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8   9
##          3   0   1   0   0   0   0   0
##          4   1  11  13   6   0   0   0
##          5   3  25 309 134  16   1   1
##          6   2  16 152 486 112  21   0
##          7   0   1  10  99 158  14   0
##          8   0   0   1   7   7  22   0
##          9   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6053
##                  95% CI : (0.5811, 0.6291)
##     No Information Rate : 0.4494
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4023
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                       Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8  Class: 9
## Sensitivity          0.0000000 0.203704   0.6371   0.6639  0.53925  0.37931 0.0000000
## Specificity          0.9993839 0.987302   0.8427   0.6622  0.90719  0.99045 1.0000000
## Pos Pred Value       0.0000000 0.354839   0.6319   0.6160  0.56028  0.59459       NaN
## Neg Pred Value       0.9963145 0.973091   0.8456   0.7071  0.89978  0.97739 0.9993861
## Prevalence           0.0036832 0.033149   0.2977   0.4494  0.17986  0.03560 0.0006139
## Detection Rate       0.0000000 0.006753   0.1897   0.2983  0.09699  0.01351 0.0000000
## Detection Prevalence 0.0006139 0.019030   0.3002   0.4843  0.17311  0.02271 0.0000000
## Balanced Accuracy    0.4996919 0.595503   0.7399   0.6631  0.72322  0.68488 0.5000000
```

Fig. 12: KNN output



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8   9
##          3   0   0   0   0   0   0   0
##          4   0   9   1   1   0   0   0
##          5   2  25 322  70   6   0   1
##          6   4  20 161 624 145  23   0
##          7   0   0   1  37 141  17   0
##          8   0   0   0   0   1  18   0
##          9   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6839
##                  95% CI : (0.6607, 0.7064)
##     No Information Rate : 0.4494
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4985
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8  Class: 9
## Sensitivity          0.000000 0.166667   0.6639   0.8525  0.48123  0.31034 0.0000000
## Specificity          1.000000 0.998730   0.9091   0.6065  0.95883  0.99936 1.0000000
## Pos Pred Value            NaN 0.818182   0.7559   0.6387  0.71939  0.94737       NaN
## Neg Pred Value       0.996317 0.972188   0.8645   0.8344  0.89393  0.97516 0.9993861
## Prevalence           0.003683 0.033149   0.2977   0.4494  0.17986  0.03560 0.0006139
## Detection Rate       0.000000 0.005525   0.1977   0.3831  0.08656  0.01105 0.0000000
## Detection Prevalence 0.000000 0.006753   0.2615   0.5998  0.12032  0.01166 0.0000000
## Balanced Accuracy    0.500000 0.582698   0.7865   0.7295  0.72003  0.65485 0.5000000
```

Fig. 13: Random Forest output

Only one model performed better than benchmark accuracy, the Random Forest model. Random Forest returned an accuracy of 68.42.3. Ridge, Lasso, Regression tree, K-nearest neighbours performed statistically worse.

VI. Conclusions

A model that is only accurate at identifying average quality wines is of limited use. With this dataset, it's hard to say if a model can be found that accurately identifies the low and high quality wines. Only more work done with this dataset can answer that.

The benchmark model has lower overall accuracy then I was able to achieve, but the benchmark accuracy for white wine is more balanced across the classes. However, that model is of limited overall use as well.

VII. Reference

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553. ISSN: 0167-9236.

https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/

https://www.statmethods.net/advstats/cart.html

http://www.rpubs.com/jasonchanhku/wine

https://rstudio-pubs-static.s3.amazonaws.com/175762_83cf2d7b322c4c63bf9ba2487b79e77e.html

http://rstudio-pubs-static.s3.amazonaws.com/80458_5000e31f84df449099a872ccf40747b7.html

https://www.kaggle.com/piyushgoyal443/red-wine-dataset#wineQualityReds.csv

VIII. R Code

```
set.seed(1)
library(glmnet)
library(caret)
library(MASS)
library(boot)
library(ISLR)
library(tree)
library(rpart)
library(plotly)
library(rpart.plot)
library(corrplot)
read in data
white wine
white.url ¡- ”https://archive.ics.uci.edu/ml/
machine-learning-databases/wine-quality/winequality-white.csv”
white.raw ¡- read.csv(white.url, header = TRUE, sep = ”;”)
white ¡- white.raw
str(white)
table(whitequality)
plot_ly(data = white, x = quality, type = ”histogram”)
boxplot(quality fixed.acidity, data = white, main =
”boxplot”, xlab = ”fixed.acidity”, ylab = ”quality”)
boxplot(quality volatile.acidity, data = white, xlab =
”volatile.acidity”, ylab = ”quality”)
boxplot(quality free.sulfur.dioxide, data =
white, xlab = ”free.sulfur.dioxide”, ylab = ”quality”)
head(white)
M < −cor(white)
head(round(M, 2))
corrplot(M, method = ”number”)
fitalinearmodelwithallpredictors
lm.fit = lm(quality ., data = white)
summary(lm.fit)
step < −stepAIC(lm.fit, direction = ”both”); stepwise
stepanova display results
data split
train=sample(1:dim(white)[1],dim(white)[1]/2)
test=-train
white.train=white[train,]
white.test=white[test,]
white.train
ridge
train.mat=model.matrix(quality .,data=white.train)
test.mat=model.matrix(quality .,data=white.test)
grid=10^seq(10, −10, length = 100)
ridge.fit = glmnet(train.mat, white.trainquality,
alpha=0,lambda=grid)
ridge.cv=cv.glmnet(train.mat,white.trainquality,
alpha = 0, lambda = grid)
ridge.bestlam = ridge.cvlambda.min
ridge.bestlam
pred.ridge=predict(ridge.fit,s=ridge.bestlam,newx=test.mat)
mean((pred.ridge-white.testquality)^2)
lasso
lasso.fit = glmnet(train.mat, white.trainquality,
alpha=1,lambda=grid)
lasso.cv=cv.glmnet(train.mat,white.trainquality,
alpha = 1, lambda = grid)
lasso.bestlam = lasso.cvlambda.min
lasso.bestlam
pred.lasso=predict(lasso.fit,s=lasso.bestlam,newx=test.mat)
mean((pred.lasso-white.testquality)^2)
regressiontree
tree.white < −tree(quality ., data = white.train)
summary(tree.white)
plot(tree.white)
text(tree.white, pretty = 0)
yhat = predict(tree.white, newdata = white.test)
mean((yhat − white.testquality)^2)
yhat
fit = rpart(quality fixed.acidity + volatile.acidity +
residual.sugar + free.sulfur.dioxide + density + pH +
sulphates + alcohol, method = ”anova”, data = white)
printcp(fit)
plotcp(fit)
summary(fit)
plot(fit, uniform = TRUE, main =
”RegressionTreeforQuality”)
text(fit, use.n = TRUE, all = TRUE, cex = .8)
Randomforest
rf.white = randomForest(quality ., data = white,
subset = white.train, mtry = 5, importance = TRUE)
yhat.rf = predict(rf.white, newdata = white[−train, ])
mean((yhat.rf − white.test)^2)
```