IDV_Learners_Proj

Coco Ying

6/21/2020

1. Overview

1.1 Introduction

This is the IDV_Learners capstone project for HarvardX: PH125.9x Data Science: Capstone. The goal of this project is to apply machine learning techniques taught throughout HarvardX Data Science Professional Certificate Program to a self-selected dataset and create a project of our own. This is a wine quality prediction project using data analysis techniques and machine learning algorithms to effectively produce accurate results. This report includes the details of this project from data cleaning, data exploration, data visualization, model building, model performance, and conclusion.

1.2 Dataset

According to the project overview page, UCI Machine Learning Repository and Kaggle are two suggested websites in dataset selection. The chosen dataset is a multivariate "Wine Quality Data Set" from UCI. The complete dataset contains two separate red and white data sets of 12 chemical attributes of wine quality in numerical form. The **white wine data set** is the chosen dataset for this project, with more description in the next chapter.

Dataset Link of the Selected Wine Quality Dataset

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

Dataset Reference

- https://archive.ics.uci.edu/ml/index.php
- https://www.kaggle.com/datasets

1.3 Purpose

The purpose of this project is to create an accurate wine quality prediction system using multiple machine learning techniques. The selected evaluation criterion or loss function is RMSE (Root Mean Square Error). The main goal in model construction is to generate a model that is low in RMSE and accurate in prediction. The following is the definition of RMSE:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}.$$

2. Methods and Analysis

2.1 Data Preparation

2.1.1 Loading Package

The following packages are required in generating models of this report (tidyverse, caret, data.table, ggplot2, hrbrthemes, corrplot, factoextra, class, e1071, rpart, rpart.plot, and gridExtra):

```
## Loading required package: tidyverse

##   Attaching packages                                  tidyverse 1.3.0

##   ggplot2 3.3.2      purrr   0.3.3
##   tibble  3.0.1      dplyr   0.8.5
##   tidyr   1.1.0      stringr 1.4.0
##   readr   1.3.1      forcats 0.4.0

##   Conflicts                                   tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift

## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose

## Loading required package: hrbrthemes

## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.

##       Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and

##       if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow

## Loading required package: corrplot

## corrplot 0.84 loaded

## Loading required package: factoextra

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

## Loading required package: class

## Loading required package: e1071
## Loading required package: rpart
```

```
## Loading required package: rpart.plot
```

```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

2.1.2 Data Loading

The data preparation process includes data loading, cleaning and partitioning. For data loading, both red and white wine datasets are loaded. In order to determine which dataset should be chosen for this project, we determine the dimensions and attributes of both red and white. The attributes of both red and white are the same 12 attributes (fixed.acidity, volatile.acidity, citric.acid, residual.sugar, chlorides, free.sulfur.dioxide, total.sulfur.dioxide, density, pH, sulphates, alcohol, and quality). Whereas the numeber of data for red is 1599 and 4898 for white. Since there are much more data for white wines, the selected dataset for modeling is the **white wine data set**.

```
# Load Red Wine Data
red_wine <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.cs
r_wine <- read.csv(file=red_wine, sep=";")
head(r_wine)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.4             0.70        0.00            1.9     0.076
## 2           7.8             0.88        0.00            2.6     0.098
## 3           7.8             0.76        0.04            2.3     0.092
## 4          11.2             0.28        0.56            1.9     0.075
## 5           7.4             0.70        0.00            1.9     0.076
## 6           7.4             0.66        0.00            1.8     0.075
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  11                   34  0.9978 3.51      0.56     9.4
## 2                  25                   67  0.9968 3.20      0.68     9.8
## 3                  15                   54  0.9970 3.26      0.65     9.8
## 4                  17                   60  0.9980 3.16      0.58     9.8
## 5                  11                   34  0.9978 3.51      0.56     9.4
## 6                  13                   40  0.9978 3.51      0.56     9.4
##   quality
## 1       5
## 2       5
## 3       5
## 4       6
## 5       5
## 6       5
```

```
dim(r_wine)
```

```
## [1] 1599   12
```

```
# Load White Wine Data
white_wine <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white
w_wine <- read.csv(file=white_wine, sep=";")
head(w_wine)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.0             0.27        0.36           20.7     0.045
## 2           6.3             0.30        0.34            1.6     0.049
## 3           8.1             0.28        0.40            6.9     0.050
## 4           7.2             0.23        0.32            8.5     0.058
## 5           7.2             0.23        0.32            8.5     0.058
## 6           8.1             0.28        0.40            6.9     0.050
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  45                  170  1.0010 3.00      0.45     8.8
## 2                  14                  132  0.9940 3.30      0.49     9.5
## 3                  30                   97  0.9951 3.26      0.44    10.1
## 4                  47                  186  0.9956 3.19      0.40     9.9
## 5                  47                  186  0.9956 3.19      0.40     9.9
## 6                  30                   97  0.9951 3.26      0.44    10.1
##   quality
## 1       6
## 2       6
## 3       6
## 4       6
## 5       6
## 6       6
```

```
dim(w_wine)
```

```
## [1] 4898   12
```

2.1.3 Data Cleaning and Partitioning

The NA values of data are then examined through the data cleaning process. In this case, the white wine dataset does not have any NA values. Since no values has to be eliminated from the original data set, we move on to data partitioning. The data is partitioned into 90% training and 10% testing set and referred to as "train" and "test" sets.

```
# Data Cleaning
sapply(w_wine, function(x) sum(is.na(x))) #No NA values for the combined dataset
```

```
##        fixed.acidity     volatile.acidity          citric.acid
##                    0                    0                    0
##       residual.sugar            chlorides  free.sulfur.dioxide
##                    0                    0                    0
## total.sulfur.dioxide              density                   pH
##                    0                    0                    0
##            sulphates              alcohol              quality
##                    0                    0                    0
```

```
# Data Partition (10% Testing)
set.seed(1, sample.kind="Rounding") #Version used: R 3.6.2
test_index <- createDataPartition(y = w_wine$quality, times = 1, p = 0.1, list = FALSE)
train <- w_wine[-test_index,]
test <- w_wine[test_index,]
```

2.2 Data Exploration and Visualization

2.2.1 Data Summary

The first step in data exploration and visualization is to check number of data in test and train sets and summarize data of **train** set. The results show that number of data of test set is indeed approximately 10% of the white wine set. Then, the summary show all 12 attributes as well as statistical summary of the dataset.

```
# Check testing percentage is correct (approx = 10%)
dim(test)[1]/(dim(train)[1]+dim(test)[1])
```

```
## [1] 0.1000408
```

```
# Overview/Summary of Data
head(train)
```

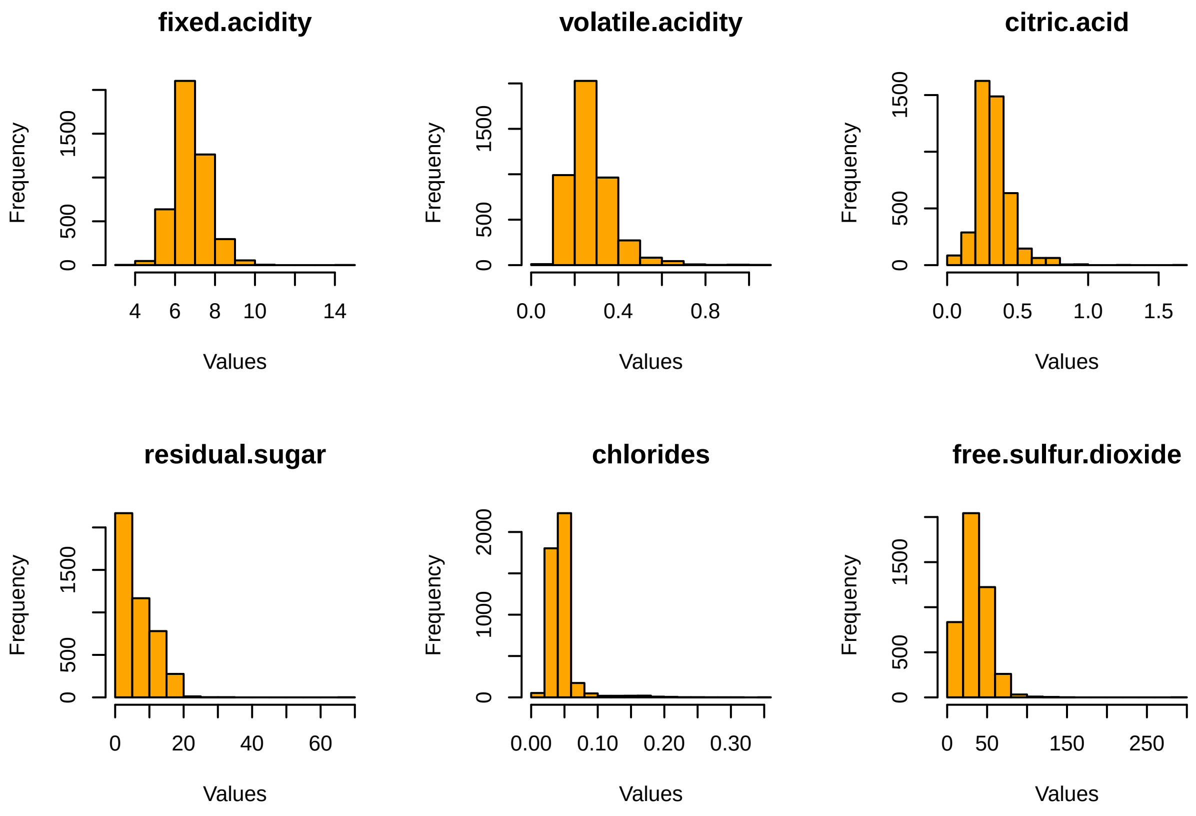
```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.0             0.27        0.36           20.7     0.045
## 2           6.3             0.30        0.34            1.6     0.049
## 3           8.1             0.28        0.40            6.9     0.050
## 4           7.2             0.23        0.32            8.5     0.058
## 5           7.2             0.23        0.32            8.5     0.058
## 6           8.1             0.28        0.40            6.9     0.050
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  45                  170  1.0010 3.00      0.45     8.8
## 2                  14                  132  0.9940 3.30      0.49     9.5
## 3                  30                   97  0.9951 3.26      0.44    10.1
## 4                  47                  186  0.9956 3.19      0.40     9.9
## 5                  47                  186  0.9956 3.19      0.40     9.9
## 6                  30                   97  0.9951 3.26      0.44    10.1
##   quality
## 1       6
## 2       6
## 3       6
## 4       6
## 5       6
## 6       6
```

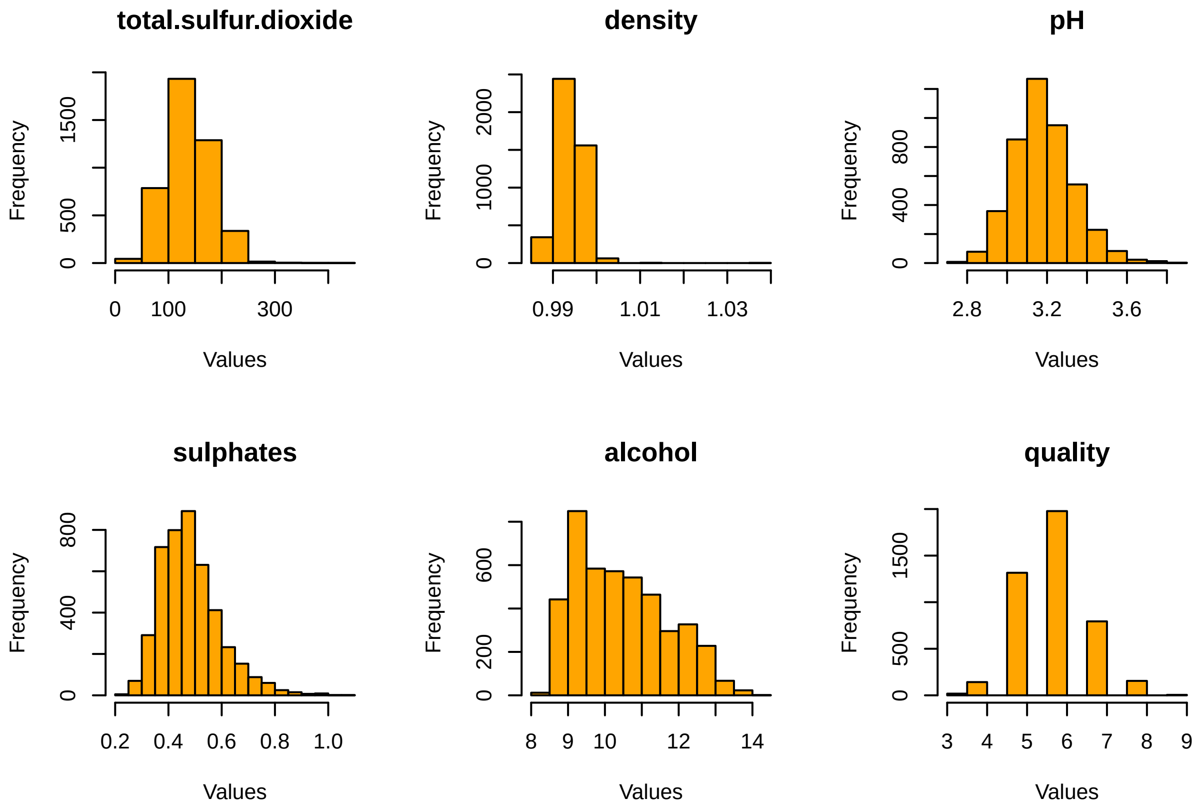
```
summary(train)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700   1st Qu.: 1.700
##  Median : 6.800   Median :0.2600   Median :0.3200   Median : 5.200
##  Mean   : 6.852   Mean   :0.2783   Mean   :0.3349   Mean   : 6.389
##  3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900   3rd Qu.: 9.900
##  Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800
##    chlorides         free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  2.00      Min.   :  9.0        Min.   :0.9871
##  1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108.0        1st Qu.:0.9917
##  Median :0.04300   Median : 34.00      Median :134.0        Median :0.9938
##  Mean   :0.04581   Mean   : 35.39      Mean   :138.5        Mean   :0.9940
##  3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:168.0        3rd Qu.:0.9961
##  Max.   :0.34600   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
##  1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.180   Median :0.4700   Median :10.40   Median :6.000
##  Mean   :3.188   Mean   :0.4877   Mean   :10.51   Mean   :5.879
##  3rd Qu.:3.280   3rd Qu.:0.5400   3rd Qu.:11.40   3rd Qu.:6.000
##  Max.   :3.820   Max.   :1.0600   Max.   :14.20   Max.   :9.000
```

2.2.2 Data Visualization- Histogram

In order to train a proper model for this dataset, we have to understand the nature of this dataset. To do so, first, we visualize each of the 12 attributes to find out its distribution. The results below are the overview of 12 attributes in histogram form.



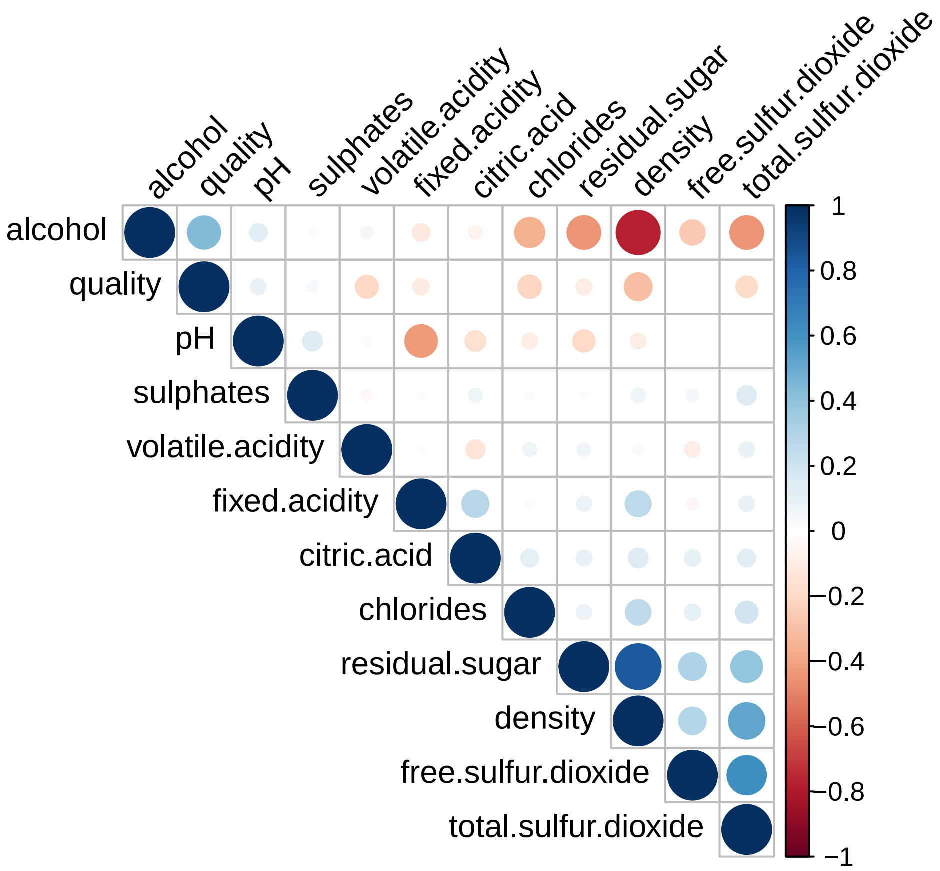


As we can see, most of the attributes show somewhat of a normal distribution pattern. The first 11 attributes are continuous, while quality rankings are discrete ranked from 3 to 9. Therefore, both regression and classification are suitable for model building of this dataset.

2.2.2 Data Visualization- Correlation

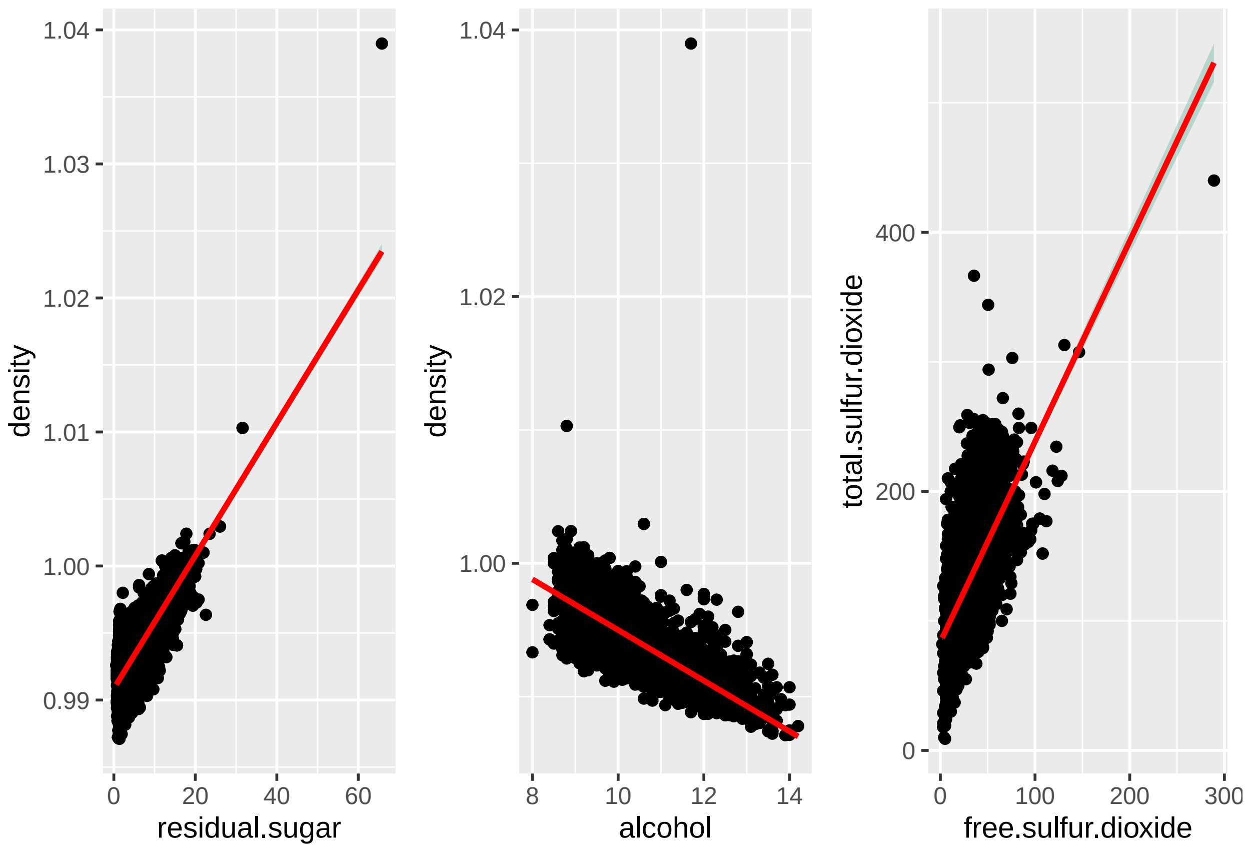
Since we have concluded that both regression and classification may be suitable for model building of this dataset, we could examine relationships between individual attributes through finding their correlation between each other. The following shows the correlation table and the visualized correlation plot.

```
##                      fixed.acidity volatile.acidity  citric.acid residual.sugar
## fixed.acidity           1.00000000      -0.01604045  0.287242805     0.08643643
## volatile.acidity       -0.01604045       1.00000000 -0.139482086     0.06676667
## citric.acid             0.28724280      -0.13948209  1.000000000     0.09599766
## residual.sugar          0.08643643       0.06676667  0.095997664     1.00000000
## chlorides               0.02515364       0.07107817  0.119320308     0.08781050
## free.sulfur.dioxide    -0.04689160      -0.09210804  0.101011322     0.30125402
## total.sulfur.dioxide    0.09197144       0.09312334  0.121406491     0.39769184
## density                 0.26257693       0.03798678  0.147271289     0.83964867
## pH                     -0.42063771      -0.02796592 -0.166204068    -0.19185189
## sulphates              -0.01609415      -0.03724718  0.071268944    -0.02751611
## alcohol                -0.11933949       0.05649555 -0.069829934    -0.44917838
## quality                -0.10339143      -0.20474400 -0.008456315    -0.09706125
##                       chlorides free.sulfur.dioxide total.sulfur.dioxide
## fixed.acidity        0.02515364        -0.046891598         0.0919714403
## volatile.acidity     0.07107817        -0.092108043         0.0931233443
## citric.acid          0.11932031         0.101011322         0.1214064909
## residual.sugar       0.08781050         0.301254019         0.3976918406
## chlorides            1.00000000         0.100206188         0.1937547907
## free.sulfur.dioxide   0.10020619         1.000000000         0.6161863516
## total.sulfur.dioxide  0.19375479         0.616186352         1.0000000000
## density               0.25495606         0.295629518         0.5281052013
## pH                   -0.09480989        -0.006531336        -0.0005904226
## sulphates             0.02253177         0.057720770         0.1409539649
## alcohol              -0.35763556        -0.250366490        -0.4499004259
## quality              -0.21254694        -0.000497408        -0.1809069324
##                          density            pH   sulphates     alcohol
## fixed.acidity         0.26257693 -0.4206377052 -0.01609415 -0.11933949
## volatile.acidity      0.03798678 -0.0279659211 -0.03724718  0.05649555
## citric.acid           0.14727129 -0.1662040679  0.07126894 -0.06982993
## residual.sugar        0.83964867 -0.1918518930 -0.02751611 -0.44917838
## chlorides             0.25495606 -0.0948098865  0.02253177 -0.35763556
## free.sulfur.dioxide   0.29562952 -0.0065313357  0.05772077 -0.25036649
## total.sulfur.dioxide  0.52810520 -0.0005904226  0.14095396 -0.44990043
## density               1.00000000 -0.0915242378  0.07679420 -0.77799287
## pH                   -0.09152424  1.0000000000  0.14839634  0.12027440
## sulphates             0.07679420  0.1483963352  1.00000000 -0.02928405
## alcohol              -0.77799287  0.1202744036 -0.02928405  1.00000000
## quality              -0.30603594  0.0931438975  0.04662307  0.43830317
##                           quality
## fixed.acidity        -0.103391429
## volatile.acidity     -0.204743999
## citric.acid          -0.008456315
## residual.sugar       -0.097061252
## chlorides            -0.212546944
## free.sulfur.dioxide  -0.000497408
## total.sulfur.dioxide -0.180906932
## density              -0.306035941
## pH                    0.093143897
## sulphates             0.046623075
## alcohol               0.438303171
## quality               1.000000000
```

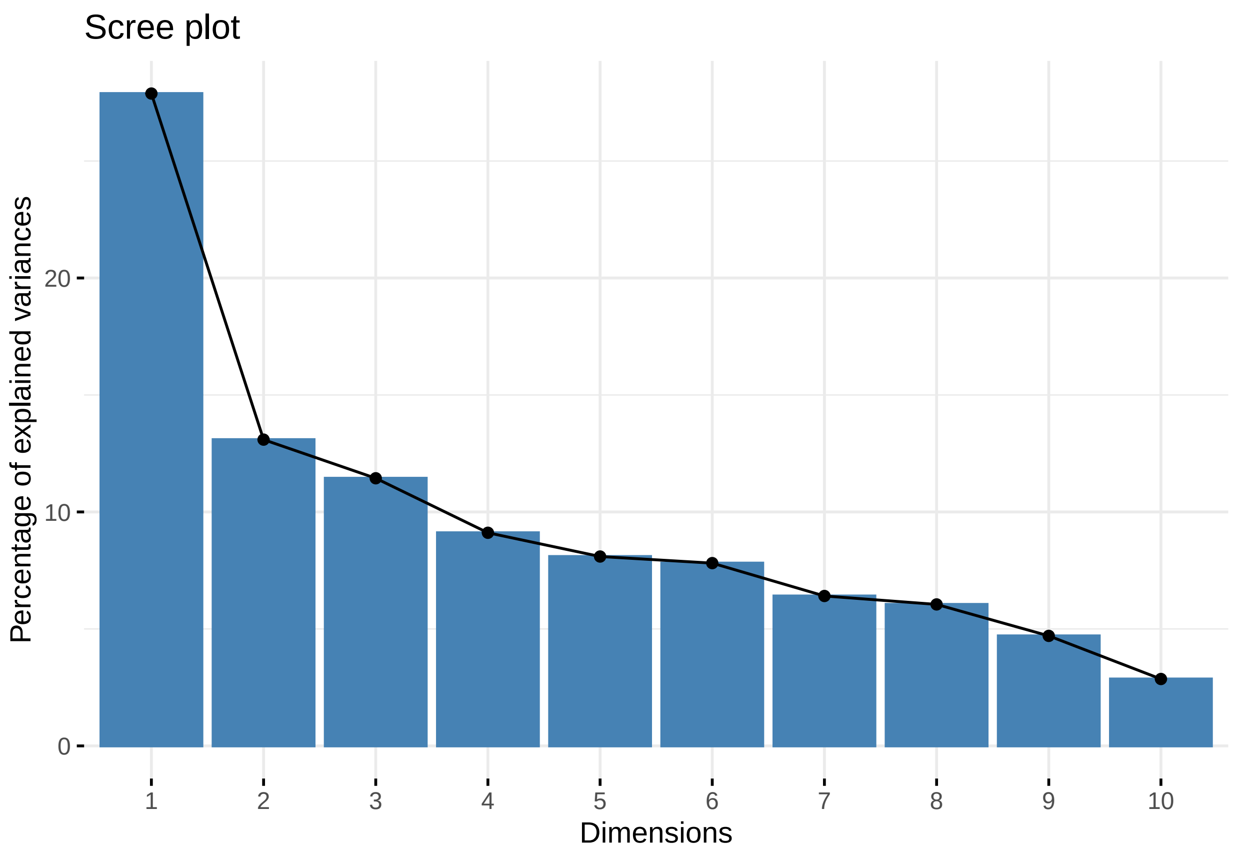


As we can see above, there are several correlations that are dark in red and blue (negative and positive correlations). The top three relations are chosen and plotted against each other, as shown below.

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



Then, we determine whether or not the attributes should be trimmed using PCA (Principle component analysis). Since contributions of each individual component are not significant, we make a decision of not using PCA in our model.



2.3 Summary of Methods and Analysis

- Relationship between attributes are visible, but there are no single attribute that strongly affects wine quality, instead we should take all attributes in consideration during model building
- No need of dimension reduction using PCA since number of attributes are limited and individual contributions are small
- Suiting model types include classification and regression, but regression would probably be superior to classification since all 11 attributes are numerical instead of categorical.

3. Results

The results section includes evaluation metrics, modeling process, and model results. The modeling techinque used in this section consist of naive, regression, and classification methods. In precise, naive approach, Knn, regression tree, linear regression, and SVR (Support Vector Regression) are used. The results differ from the use of methods, with SVR being the best result of all.

3.1 Evaluation Metric/Loss Function

The evaluation metric or loss function used in this project is RMSE, a popular indicator of differences between predicted values and true values. RMSE is commonly used in the evaluation of models in machine learning. The equation of RMSE is stated below:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

3.2 Modeling

3.2.1 Model 1: Naive Model

The first step of modeling is to generate a **naive model** through calculating mean of train set and use this mean to predict results of test set. The RMSE for the naive model is 0.90455.

```r
# Model 1: Naive Model (RMSE=0.90455)
#Starting RMSE
mu_hat <- mean(train$quality)
RMSE_1<-RMSE(test$quality, mu_hat)
rmse_table<- data.table(Method="Naive", Type="Mean", RMSE=RMSE_1)
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524

3.2.2 Model 2: Knn

The second model we consider is **Knn (K-nearest-neighbor)**, a classification method with the goal of grouping datapoint through lexamining the datapoints around it. The "K" in Knn is the number of nearest neighbors considered. In order to find an appropriate Knn model, the k value has to be optimized. In this case, the optimized k value is 49. Which is the k-value used in the modeling process as shown in the code below. As we can see, this optimization method is not suiting for this dataset, since RMSE obtained from this model is 2.1806. The possible reason that this model is inappropriate is that Knn is a classification method that does not deal with dimentionality well. It is a type of instance-based (lazy learning) method that has a strength in approximation.

```r
# Model 2: Optimized Knn (RMSE=2.1806196)
set.seed(1)
train_knn<- train(quality ~ ., method = "knn",
                  data = train,
                  tuneGrid = data.frame(k = seq(3, 51, 2)))
train_knn #use k=49
```

```
## k-Nearest Neighbors
##
## 4408 samples
##   11 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 4408, 4408, 4408, 4408, 4408, 4408, ...
## Resampling results across tuning parameters:
##
##   k   RMSE       Rsquared   MAE
##    3  0.8943624  0.1491361  0.6449355
##    5  0.8670105  0.1429979  0.6575846
##    7  0.8528516  0.1407273  0.6592755
##    9  0.8464953  0.1367906  0.6604830
##   11  0.8410704  0.1348595  0.6607732
##   13  0.8362286  0.1351371  0.6592506
##   15  0.8319334  0.1366840  0.6572607
##   17  0.8291867  0.1370971  0.6565183
##   19  0.8268021  0.1380980  0.6551296
##   21  0.8261150  0.1369565  0.6548615
##   23  0.8256074  0.1360509  0.6550781
##   25  0.8246953  0.1360959  0.6545299
##   27  0.8242342  0.1356681  0.6542548
##   29  0.8235525  0.1358078  0.6536381
##   31  0.8230953  0.1357791  0.6533001
##   33  0.8228200  0.1356248  0.6533383
##   35  0.8225708  0.1355713  0.6531418
##   37  0.8221830  0.1358565  0.6531999
##   39  0.8218666  0.1360292  0.6528212
##   41  0.8219524  0.1356191  0.6527735
##   43  0.8213937  0.1364609  0.6520878
##   45  0.8214510  0.1361700  0.6521349
##   47  0.8215777  0.1357727  0.6520413
##   49  0.8213909  0.1359466  0.6518711
##   51  0.8214740  0.1356203  0.6517481
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 49.
```

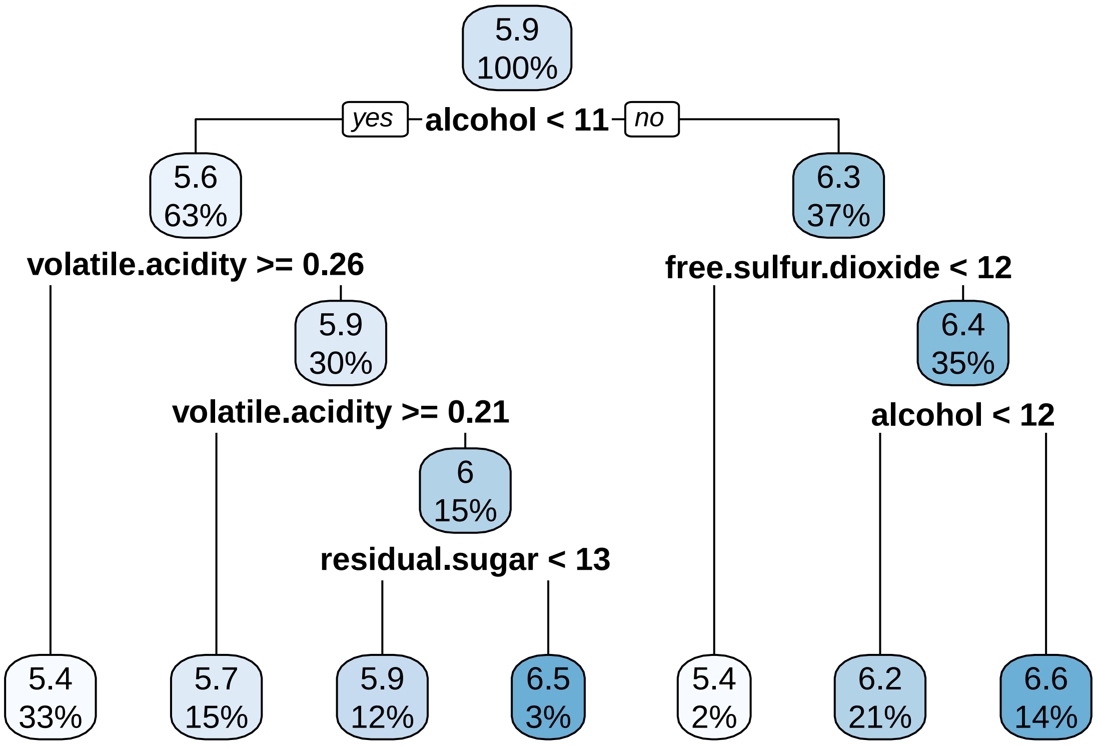
```
X_tr_knn=train[,1:11]
Y_tr_knn=train$quality
X_ts_knn=test[,1:11]
Y_ts_knn=test$quality
ts_knn=test[,1:11]
mod_2 <- knn(train=scale(X_tr_knn), test=scale(X_ts_knn), cl=Y_tr_knn, k=c(train_knn$bestTune), prob=TRU
RMSE_2 <- RMSE(test$quality, c(mod_2)) #2.154
rmse_table <- bind_rows(rmse_table,
                        data_frame(Method="Knn",
                                   Type="Classification",
                                   RMSE = RMSE_2))
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524
Knn	Classification	2.1806196

3.2.3 Model 3: Regression Tree

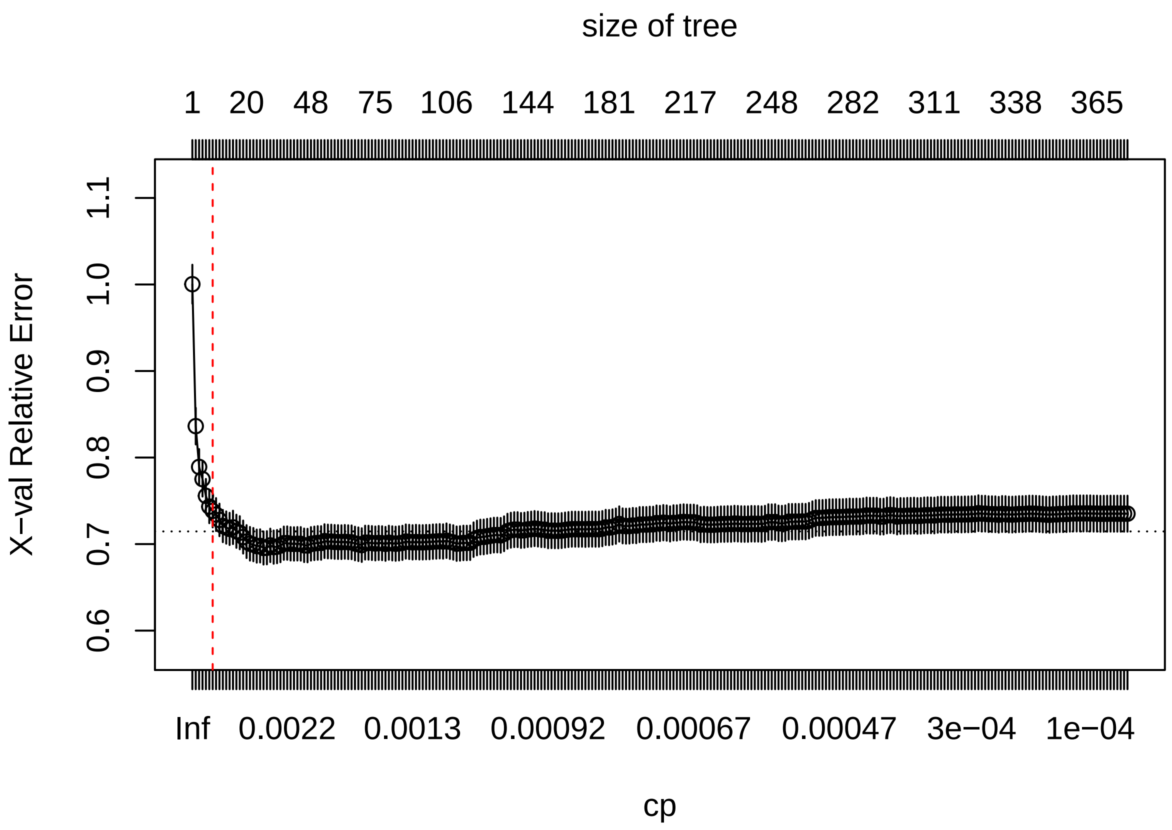
The third model we consider is **regression tree**, which is a classification method where target variables can be continuous. This method would be an improvement from Knn due to its property of being able to classify continuous values. First, the predictions regression tree is calcualted without cost penalty. Then, generated with appropriate penalty using 10-fold cross validation. The terminal nodes obtained from the initial regression tree is 7. We use this number in the model with cross validation, the next plot shows the plot of cost complexity parameter and red dotted line at size of tree being 7. With size of tree over 7, we can see that the rate of change in X-val error slows down. Therefore, this is an appropriate regression tree model with a RMSE of 0.7525, much better than the naive model.

```
# Model 3: Regression Tree (RMSE=0.7525170)
par(mfrow=c(1,1))
fit_a<- rpart(quality ~ ., data = train,
              method="anova")
rpart.plot(fit_a)
```



```r
predict_a <-predict(fit_a, newdata=test)

fit_b<- rpart(quality ~ ., data = train,
              method="anova",
              control=list(cp=0, xval=10)) #10-fold cross validation
predict_b <-predict(fit_b, newdata=test)
plotcp(fit_b)
abline(v=7, lty="dashed",col = "red")
```



```
RMSE_3 <- RMSE(test$quality, predict_b)
rmse_table <- bind_rows(rmse_table,
                        data_frame(Method="Regression Tree",
                                   Type="Classification",
                                   RMSE = RMSE_3))
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524
Knn	Classification	2.1806196
Regression Tree	Classification	0.7525170

3.2.4 Model 4: Linear Regression (1 attribute)

After examining classification methods, we move on to regression models. The fourth model to examine is the simplest **linear regression with one attribute**. As previously found in the correlation table, the relationship between the attribute "alcohol" and "quality" is the most vivid amongst all. Therefore, we use the simplest linear regression model in generating our RMSE of 0.8245, which is an improvement from the naive model but worse than regression tree. This model is too simple in determining a precise prediction.

```
# Model 4: Linear Regression (Alcohol) (RMSE=0.8244660)
fit <- lm(quality~alcohol, data=train)
mod_4<- fit$coef[1]+fit$coef[2]*test$alcohol
RMSE_4<- RMSE(test$quality, mod_4) #0.82447
rmse_table <- bind_rows(rmse_table,
                        data_frame(Method="LM_Alcohol",
                                   Type="Regression",
                                            RMSE = RMSE_4))
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524
Knn	Classification	2.1806196
Regression Tree	Classification	0.7525170
LM_Alcohol	Regression	0.8244660

3.2.5 Model 5: Linear Regression (2 attributes)

Since the model previously created is too simple, the fifth model is a linear regression with two main attributes alcohol and density. The results show RMSE of 0.8247, meaning that the relations between quality and the two attributes are not direct contributions that can be modeled using simple linear regression. The attributes have correlations with each other, that requires the use of higher order and complex regression methods.

```
# Model 5: Linear Regression (Alcohol+Density) (RMSE=0.8247146)
fit_2 <- lm(quality~alcohol+density, data=train)
mod_5 <-fit_2$coef[1]+fit_2$coef[2]*test$alcohol+fit_2$coef[3]*test$density
RMSE_5 <- RMSE(test$quality, mod_5) #0.82471
rmse_table <- bind_rows(rmse_table,
                        data_frame(Method="LM_Alcohol+Density",
                                   Type="Regression",
                                   RMSE = RMSE_5))
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524
Knn	Classification	2.1806196
Regression Tree	Classification	0.7525170
LM_Alcohol	Regression	0.8244660
LM_Alcohol+Density	Regression	0.8247146

3.2.6 Model 6: Support Vector Regression (SVR)

The sixth model we used is the **Support Vector Regression (SVR)**. The more well known method is the Support Vector Machine (SVM), a classification method. SVR is an altered regression version of SVM. SVR is an effective algorithm in estimating real-values. It benefits in estimation through strongly penalizing misestimates. In our SVR model, all 11 other attributes are used in estimating the quality attribute. Through adjusting tune parameters cost and width of error tolerance (epsilon), the resulting RMSE is the best result of 0.704.

```
# Model 6: SVR (RMSE=0.7042508)
mod_6 <- svm(quality~fixed.acidity+volatile.acidity+citric.acid+residual.sugar+chlorides+free.sulfur.di
predict_6<- predict(mod_6, newdata=test)
RMSE_6 <- RMSE(test$quality, predict_6) #0.70425
rmse_table <- bind_rows(rmse_table,
                        data_frame(Method="SVR",
                                   Type="Regression",
                                   RMSE = RMSE_6))
rmse_table %>%knitr::kable()
```

Method	Type	RMSE
Naive	Mean	0.9045524
Knn	Classification	2.1806196
Regression Tree	Classification	0.7525170
LM_Alcohol	Regression	0.8244660
LM_Alcohol+Density	Regression	0.8247146
SVR	Regression	0.7042508

4. Conclusion

Through analyzing data and generating multiple machine learning models to predict wine quality, we have acheived a RMSE value of 0.70425 using the SVR model. In the beginning, we cleaned and partitioned the dataset. Then, use statistical methods and visualization to examine the dataset. The histograms, correlation table, and PCA are all methods that help us in determining the types of models to use later. After analyzing dataset, we have concluded that both classification and regression methods could be used for modeling, with a preference of regression due to dataset properties. In the modeling process, some models (especially Knn) suffers from its classification characteristic of not being able to successfully group models of too many attributes. Other models such as linear regression and naive method fail to predict well enough since they are too simple or they only consider relationship between the attributes and quality instead of interactions between all attributes. Whereas the two better models are regression tree and SVR since they both succeed in the consideration of multiple attributes and able to deal with continuous attributes rather than categorical attributes.

The limitations of this dataset include the lack of categorical value or descriptors. This dataset is mainly based on chemicals and their values. If there were other characteristics such as grape variety and region, these categorical attributes may help in modeling. Whereas the limitations of this work is that we only take white wine dataset into consideration. For future work, this work can be expanded into combining white and red dataset to generate a wholesome prediction model.

Appendix

Environment

```
## [1] "Operating System:"
```

```
##                _
## platform       x86_64-apple-darwin15.6.0
## arch           x86_64
## os             darwin15.6.0
## system         x86_64, darwin15.6.0
## status
## major          3
## minor          6.2
## year           2019
## month          12
## day            12
## svn rev        77560
## language       R
## version.string R version 3.6.2 (2019-12-12)
## nickname       Dark and Stormy Night
```