Classification of wine: Supervised Machine Learning

Nigel K. Gondo

Brief Description

The purpose of this project is quite simple, it is to conduct a classification exercise to put wine into different classes of quality based on multiple variables such as acidity, sugar, alcohol etc.

Setting the directory with the data required

```
setwd('C://Users//Nigel Gondo//Documents//Portfolio
Projects//Classification')
```

Importing the relevant libraries

```
#Importing libraries
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(caTools)
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

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

library(rpart)
```

Importing and Exploring the dataset

```
df_wine <- read.csv('WineQT.csv')
head(df_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 Id
## 1       5  0
## 2       5  1
## 3       5  2
## 4       6  3
## 5       5  4
## 6       5  5

#checking on the structure of the data frame
str(df_wine)

## 'data.frame':    1143 obs. of  13 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 6.7 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58
0.58 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.08 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 1.8 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069
0.065 0.073 0.097 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 15 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 65 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36
3.28 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57
0.54 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 9.2 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
##  $ Id                  : int  0 1 2 3 4 5 6 7 8 10 ...

#checking the dimensions of the data
dim(df_wine)
```

```
## [1] 1143   13

#quality needs to be converted into a categorical variable as this will the
variable put into classes
df_wine$quality <- as.factor(df_wine$quality)
str(df_wine$quality)

##  Factor w/ 6 levels "3","4","5","6",..: 3 3 3 4 3 3 3 5 5 3 ...

#checking if there are any null values in the data set
sum(is.na(df_wine))

## [1] 0

#Summary statistics
summary(df_wine)

##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 4.600   Min.   :0.1200   Min.   :0.0000   Min.   : 0.900
##  1st Qu.: 7.100   1st Qu.:0.3925   1st Qu.:0.0900   1st Qu.: 1.900
##  Median : 7.900   Median :0.5200   Median :0.2500   Median : 2.200
##  Mean   : 8.311   Mean   :0.5313   Mean   :0.2684   Mean   : 2.532
##  3rd Qu.: 9.100   3rd Qu.:0.6400   3rd Qu.:0.4200   3rd Qu.: 2.600
##  Max.   :15.900   Max.   :1.5800   Max.   :1.0000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 21.00       1st Qu.:0.9956
##  Median :0.07900   Median :13.00       Median : 37.00       Median :0.9967
##  Mean   :0.08693   Mean   :15.62       Mean   : 45.91       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 61.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :68.00       Max.   :289.00       Max.   :1.0037
##        pH           sulphates         alcohol      quality       Id
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   3:  6   Min.   :   0
##  1st Qu.:3.205   1st Qu.:0.5500   1st Qu.: 9.50   4: 33   1st Qu.: 411
##  Median :3.310   Median :0.6200   Median :10.20   5:483   Median : 794
##  Mean   :3.311   Mean   :0.6577   Mean   :10.44   6:462   Mean   : 805
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   7:143   3rd Qu.:1210
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   8: 16   Max.   :1597
```

Spliting the dataset into train and test to train and predict the model

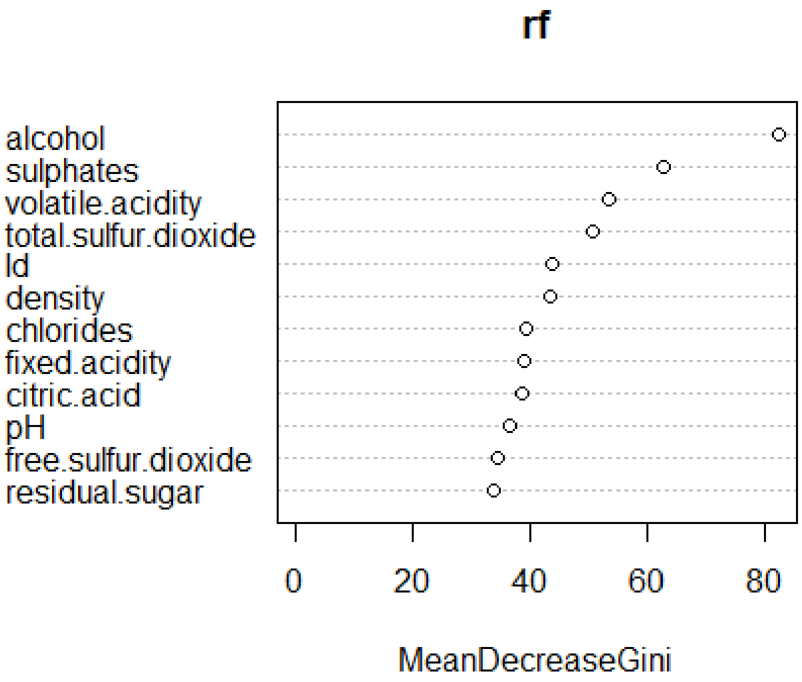
```
set.seed(42)
split_df_wine <- sample.split(df_wine, SplitRatio = 0.8)
training_set <- subset(df_wine, split_df_wine == TRUE)
testing_set <- subset(df_wine, split_df_wine == FALSE)

#Creating the random forest model and displaying the metrics
rf <- randomForest(quality~.,data=training_set)
rf
```

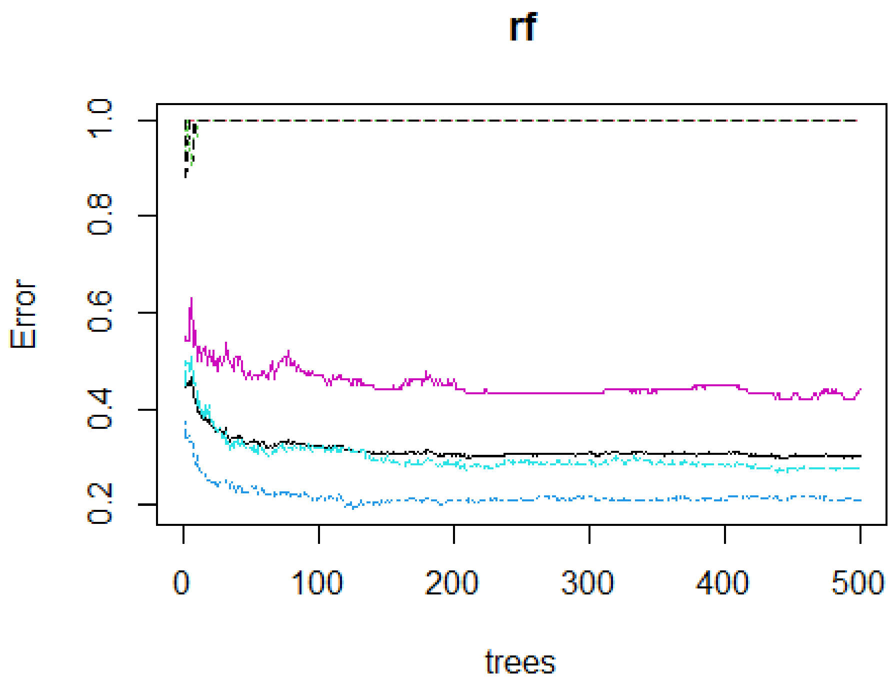
```
##
## Call:
##  randomForest(formula = quality ~ ., data = training_set)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 30.11%
## Confusion matrix:
##   3 4   5   6  7 8 class.error
## 3 0 0   4   0  0 0   1.0000000
## 4 0 0  20   5  0 0   1.0000000
## 5 0 0 303  78  2 0   0.2088773
## 6 0 0  82 255 15 1   0.2776204
## 7 0 0   2  41 57 2   0.4411765
## 8 0 0   0   7  6 0   1.0000000
```

The error rate of the model is 30.11%, meaning that the accuracy of the model is 69.89%

```
#Variable importance
varImpPlot(rf)
```



```
plot(rf)
```



The error rate is stabilising as more trees are added