Prediction of quality of wine in R

A Statistics and Machine Learning Challenge

Kamal Babaei Sonbolabadi

Fall 2018 and Winter 2019

Introduction: In this project, I have used Decision Tree and Random Forest in order to predict the quality of wine using red wine data set.

Context: This datasets is related to red variants of the Portuguese "Vinho Verde" wine. The datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones). This dataset is also available from the UCI machine learning repository.

Input variables: (based on physicochemical tests): 1 - fixed acidity 2 - volatile acidity 3 - citric acid 4 - residual sugar 5 - chlorides 6 - free sulfur dioxide 7 - total sulfur dioxide 8 - density 9 - pH 10 - sulphates 11 - alcohol Output variable (based on sensory data): 12 - quality (score between 0 and 10)

Data Description: Let us first get to know the data.

```
wine<-read.csv("../input/winequality-red.csv")

wine<-na.omit(wine)

str(wine)

wine$quality<-as.factor(wine$quality)
```

Output:

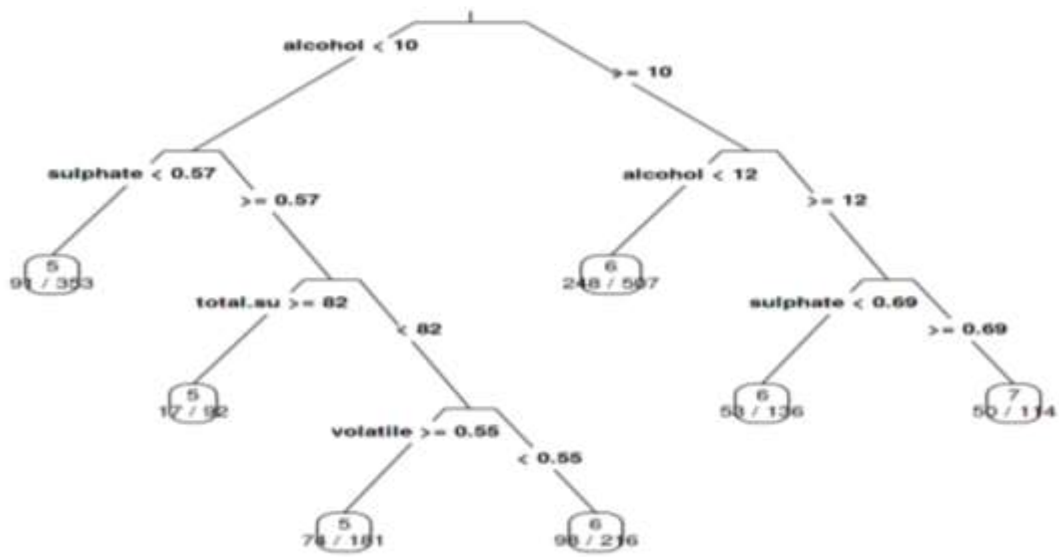
```
'data.frame':   1599 obs. of  12 variables:

 $ fixed.acidity      : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...

 $ volatile.acidity   : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...

 $ citric.acid        : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...

 $ residual.sugar     : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...

 $ chlorides          : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...

 $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
```

```
$ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
$ 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 3.35 ...
$ sulphates          : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
$ alcohol            : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
$ quality            : int  5 5 5 6 5 5 5 7 7 5 ...
```

Step by step explanation:

Here I am going to build a Decision Tree and Random Forest on this dataset in order to predict the quality of the wine based on other variables.

```
library(rpart)
library(rpart.plot)
library(caret)
library(ROCR)
library(randomForest)
library(rattle)
#Dividing the dataset into Training and Testing sets.
set.seed(1)
index<-createDataPartition(wine$quality, p= .8, list=FALSE)
Train<-wine[index,]
Test<-wine[-index,]
#Building the decision tree
set.seed(1)
tree<-rpart(wine$quality~., data=wine)
prp(tree, type=3, extra=3, tweak=0.8, main="The Quality of Wine", compress=TRUE )
```



<u>So, When the alcohol<10and sulphate <0.57 then the quality of the wine is predicted to be 5.</u>

#Making predictions

model1<-rpart(Train$quality~., data=Train)

pred<-predict(model1, Test, type="class")

confusionMatrix(pred, Test$quality)

Output:

Confusion Matrix and Statistics

```
          Reference
Prediction   3   4   5   6   7   8
         3   0   0   0   0   0   0
         4   0   0   0   0   0   0
         5   2   6 104  54   3   0
         6   0   4  30  68  24   3
         7   0   0   2   5  12   0
         8   0   0   0   0   0   0
```

Overall Statistics

Accuracy : 0.5804

```
95% CI : (0.524, 0.6354)
    No Information Rate : 0.429
    P-Value [Acc > NIR] : 4.273e-08
                  Kappa : 0.3018
 Mcnemar's Test P-Value : NA
Statistics by Class:
                     Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
Sensitivity          0.000000  0.00000   0.7647   0.5354  0.30769 0.000000
Specificity          1.000000  1.00000   0.6409   0.6789  0.97482 1.000000
Pos Pred Value            NaN      NaN   0.6154   0.5271  0.63158      NaN
Neg Pred Value       0.993691  0.96845   0.7838   0.6862  0.90940 0.990536
Prevalence           0.006309  0.03155   0.4290   0.4006  0.12303 0.009464
Detection Rate       0.000000  0.00000   0.3281   0.2145  0.03785 0.000000
Detection Prevalence 0.000000  0.00000   0.5331   0.4069  0.05994 0.000000
Balanced Accuracy    0.500000  0.50000   0.7028   0.6072  0.64126 0.500000
```

<u>So we see that the overall accuracy of the model is not very high , it is only 58,04%. Now it’s time to choose Random Forest.</u>

```
# Random Forest
set.seed(1)
model2<-randomForest(Train$quality~., data=Train, ntree=50, do.trace=T, importance=T)
```

Output:

```
ntree      OOB      1      2      3      4      5      6
    1:  46.57%100.00% 92.31% 39.09% 49.47% 47.54% 80.00%
    2:  45.87%100.00% 84.62% 36.76% 46.71% 60.22% 88.89%
    3:  43.34%100.00% 93.75% 32.32% 46.45% 52.25% 84.62%
    4:  41.54%100.00% 94.29% 30.04% 44.91% 50.00% 92.86%
    5:  43.79%100.00% 94.74% 34.06% 45.41% 51.39% 92.86%
    6:  40.74%100.00%100.00% 30.73% 42.62% 48.00% 78.57%
```

7: 39.76%100.00%100.00% 30.58% 37.91% 52.90% 93.33%

8: 40.43%100.00%100.00% 30.60% 40.72% 49.36% 86.67%

9: 40.05%100.00% 97.67% 30.80% 39.88% 49.04% 86.67%

10: 38.46%100.00%100.00% 29.10% 38.86% 44.94% 86.67%

11: 36.99%100.00%100.00% 27.94% 36.69% 44.03% 86.67%

12: 37.53%100.00% 97.67% 27.52% 38.11% 45.91% 86.67%

13: 37.69%100.00%100.00% 27.34% 38.31% 46.54% 86.67%

14: 36.02%100.00%100.00% 26.97% 35.49% 43.40% 86.67%

15: 36.92%100.00%100.00% 26.79% 36.47% 48.12% 86.67%

16: 36.07%100.00%100.00% 26.42% 35.10% 46.88% 86.67%

17: 34.19%100.00%100.00% 23.85% 34.31% 43.12% 86.67%

18: 35.75%100.00%100.00% 25.32% 34.71% 49.38% 86.67%

19: 35.05%100.00% 97.67% 24.40% 34.31% 48.75% 86.67%

20: 34.87%100.00%100.00% 24.59% 34.64% 45.00% 86.67%

21: 33.78%100.00%100.00% 23.12% 33.27% 45.62% 86.67%

22: 34.56%100.00%100.00% 24.22% 34.25% 45.00% 86.67%

23: 33.15%100.00% 97.67% 22.57% 32.88% 44.38% 86.67%

24: 33.00%100.00%100.00% 22.75% 33.07% 41.25% 86.67%

25: 33.31%100.00% 97.67% 23.67% 33.66% 39.38% 86.67%

26: 33.39%100.00%100.00% 23.49% 33.66% 40.00% 86.67%

27: 33.15%100.00%100.00% 23.12% 32.88% 41.88% 86.67%

28: 32.37%100.00%100.00% 23.30% 31.51% 39.38% 86.67%

29: 33.62%100.00%100.00% 24.04% 33.27% 41.25% 86.67%

30: 32.84%100.00%100.00% 23.30% 32.49% 40.00% 86.67%

31: 33.31%100.00%100.00% 23.30% 33.27% 41.25% 86.67%

32: 32.84%100.00%100.00% 23.67% 32.29% 39.38% 86.67%

33: 33.70%100.00%100.00% 24.04% 34.25% 38.75% 86.67%

34: 33.00%100.00%100.00% 22.57% 33.86% 39.38% 86.67%

35: 33.15%100.00%100.00% 22.94% 33.66% 40.00% 86.67%

36: 33.23%100.00%100.00% 22.94% 33.66% 40.62% 86.67%

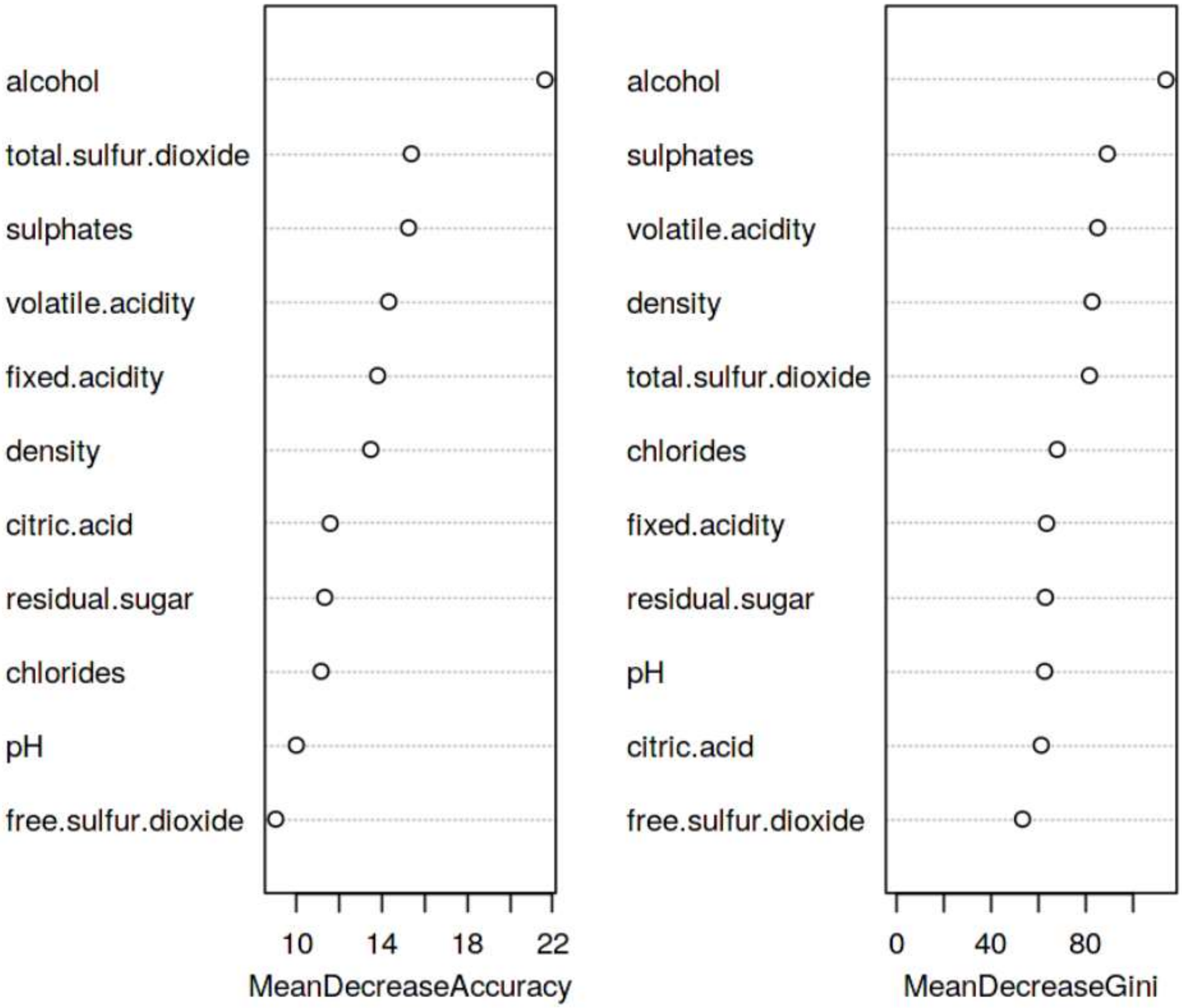
37: 33.39%100.00%100.00% 22.57% 33.86% 42.50% 86.67%

```
38:  33.15%100.00%100.00% 23.67% 33.07% 39.38% 86.67%
39:  32.92%100.00%100.00% 22.57% 32.68% 42.50% 86.67%
40:  33.23%100.00%100.00% 23.67% 32.88% 40.62% 86.67%
41:  32.92%100.00%100.00% 23.30% 33.07% 38.75% 86.67%
42:  32.92%100.00%100.00% 22.57% 32.68% 42.50% 86.67%
43:  32.45%100.00%100.00% 22.75% 31.70% 41.25% 86.67%
44:  32.14%100.00%100.00% 22.57% 31.51% 40.00% 86.67%
45:  32.29%100.00%100.00% 22.94% 31.51% 40.00% 86.67%
46:  33.15%100.00%100.00% 23.67% 32.09% 42.50% 86.67%
47:  32.45%100.00%100.00% 22.94% 31.70% 40.62% 86.67%
48:  32.84%100.00%100.00% 22.75% 31.70% 44.38% 86.67%
49:  32.45%100.00%100.00% 22.75% 31.12% 43.12% 86.67%
50:  32.68%100.00%100.00% 22.57% 31.90% 43.12% 86.67%
```

 # Let's look at the important variables

varImpPlot(model2)

We notice that. the variable “alcohol” is the most important variable for the overall accuracy of the model.




```
# making predictions
pred2<-predict(model2, newdata=Test, type="class")
confusionMatrix(pred2, Test$quality)
```

Output:

Confusion Matrix and Statistics

Reference

Prediction 3 4 5 6 7 8

3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	2	7	109	27	2	0
6	0	3	25	95	20	1
7	0	0	2	5	17	2
8	0	0	0	0	0	0

Overall Statistics

Accuracy : 0.6972

95% CI : (0.6433, 0.7473)

No Information Rate : 0.429

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.5027

Mcnemar's Test P-Value : NA

Statistics by Class:

	Class: 3	Class: 4	Class: 5	Class: 6	Class: 7	Class: 8
Sensitivity	0.000000	0.00000	0.8015	0.7480	0.43590	0.000000
Specificity	1.000000	1.00000	0.7901	0.7421	0.96763	1.000000
Pos Pred Value	NaN	NaN	0.7415	0.6597	0.65385	NaN
Neg Pred Value	0.993691	0.96845	0.8412	0.8150	0.92440	0.990536
Prevalence	0.006309	0.03155	0.4290	0.4006	0.12303	0.009464
Detection Rate	0.000000	0.00000	0.3438	0.2997	0.05363	0.000000
Detection Prevalence	0.000000	0.00000	0.4637	0.4543	0.08202	0.000000
Balanced Accuracy	0.500000	0.50000	0.7958	0.7451	0.70176	0.500000

Conclusion: The overall accuracy is 69.72% which is much better results compared to the Decision Tree. Therefore, it is better to use the Random Forest model while predicting the quality of the wine.