Untitled

Zhengyu Lu

2022-12-03

## Importing libraries  
library(MASS)  
library(randomForest)

## randomForest 4.7-1.1

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

library(ggplot2)

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:randomForest':  
##   
## margin

library(corrplot)

## corrplot 0.92 loaded

library(caret)

## Loading required package: lattice

library(gbm)

## Loaded gbm 2.1.8.1

library(e1071)  
library(tree)  
whitequality <- read.csv("winequality-white.csv", sep = ";")  
head(whitequality)

## 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

sum(is.na(whitequality))

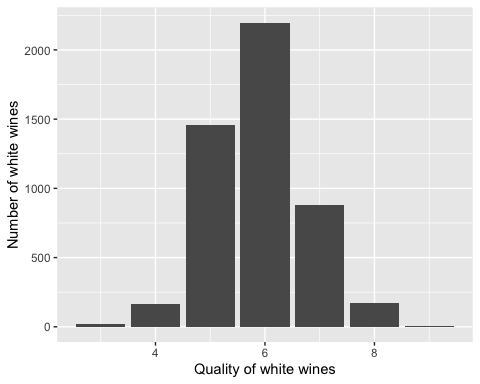
## [1] 0

summary(whitequality)

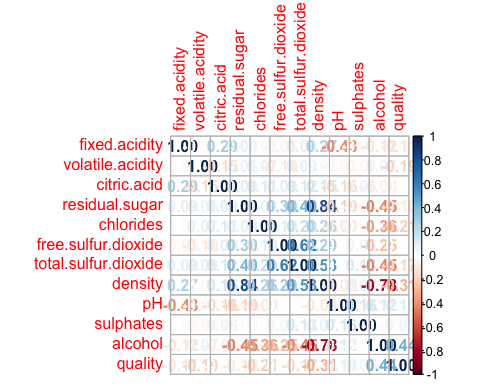
## fixed.acidity volatile.acidity citric.acid residual.sugar   
## Min. : 3.800 Min. :0.0800 Min. :0.0000 Min. : 0.600   
## 1st Qu.: 6.300 1st Qu.:0.2100 1st Qu.:0.2700 1st Qu.: 1.700   
## Median : 6.800 Median :0.2600 Median :0.3200 Median : 5.200   
## Mean : 6.855 Mean :0.2782 Mean :0.3342 Mean : 6.391   
## 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 :0.9937   
## Mean :0.04577 Mean : 35.31 Mean :138.4 Mean :0.9940   
## 3rd Qu.:0.05000 3rd Qu.: 46.00 3rd Qu.:167.0 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 :0.4898 Mean :10.51 Mean :5.878   
## 3rd Qu.:3.280 3rd Qu.:0.5500 3rd Qu.:11.40 3rd Qu.:6.000   
## Max. :3.820 Max. :1.0800 Max. :14.20 Max. :9.000

##Quality frequency  
ggplot(whitequality,aes(quality)) + geom\_histogram(stat="count") +  
 xlab("Quality of white wines") + ylab("Number of white wines")

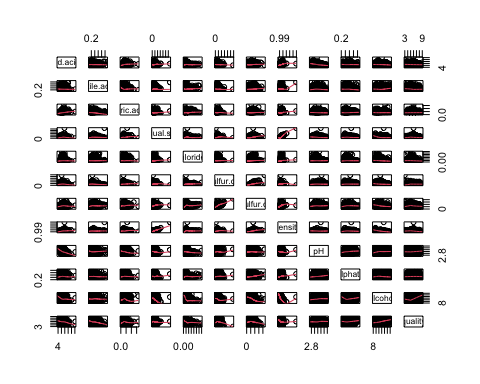
## Warning in geom\_histogram(stat = "count"): Ignoring unknown parameters:  
## `binwidth`, `bins`, and `pad`



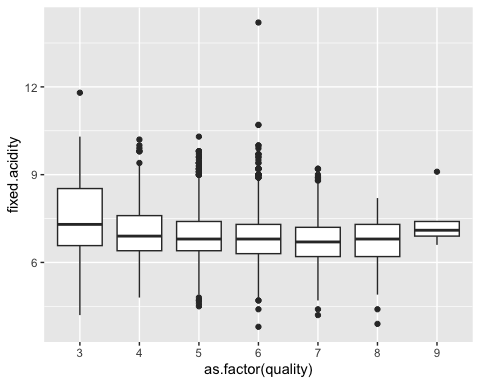
##correlation  
M <- cor(whitequality)  
corrplot(M, method = "number")



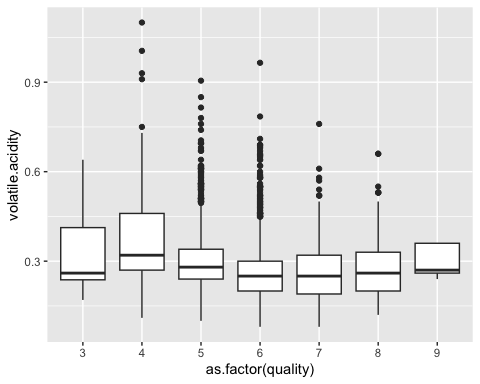
##scatter plot  
pairs(whitequality, panel=panel.smooth)



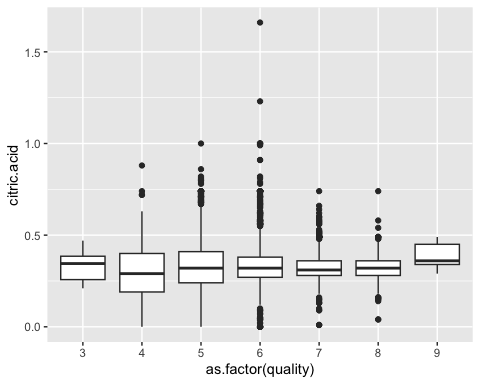
##box plot  
ggplot(whitequality, aes(as.factor(quality),fixed.acidity))+ geom\_boxplot()



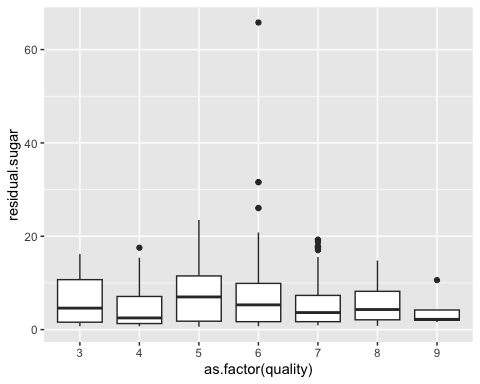
ggplot(whitequality, aes(as.factor(quality),volatile.acidity))+ geom\_boxplot()



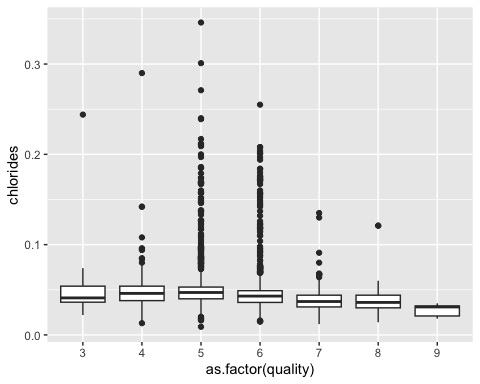
ggplot(whitequality, aes(as.factor(quality),citric.acid))+ geom\_boxplot()



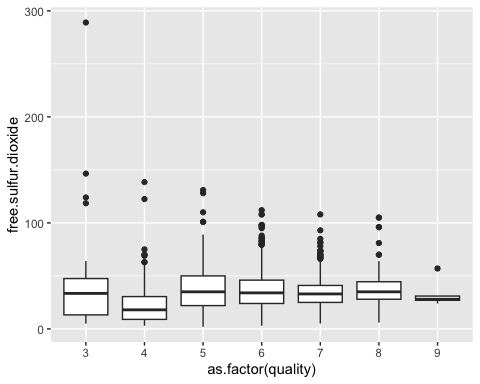
ggplot(whitequality, aes(as.factor(quality),residual.sugar))+ geom\_boxplot()



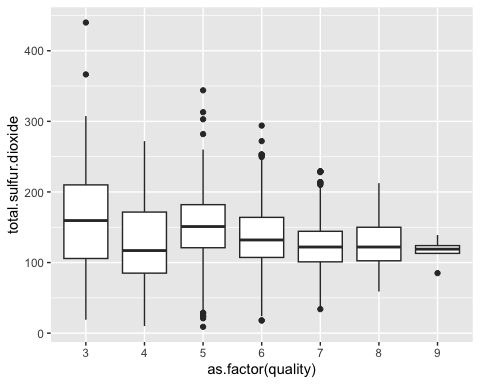
ggplot(whitequality, aes(as.factor(quality),chlorides))+ geom\_boxplot()



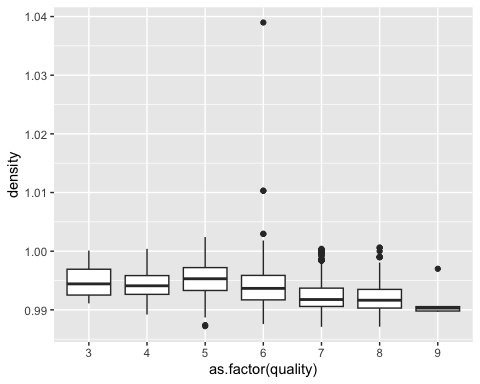
ggplot(whitequality, aes(as.factor(quality),free.sulfur.dioxide))+ geom\_boxplot()



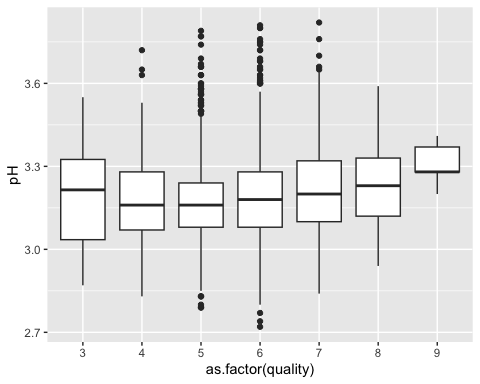
ggplot(whitequality, aes(as.factor(quality),total.sulfur.dioxide))+ geom\_boxplot()



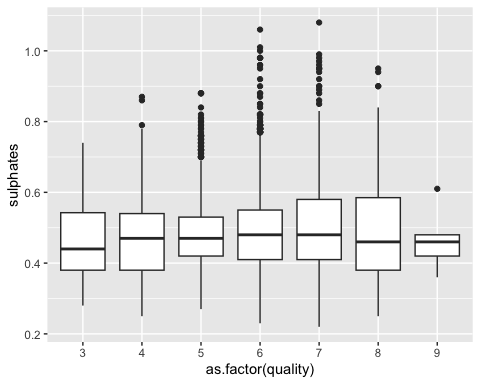
ggplot(whitequality, aes(as.factor(quality),density))+ geom\_boxplot()



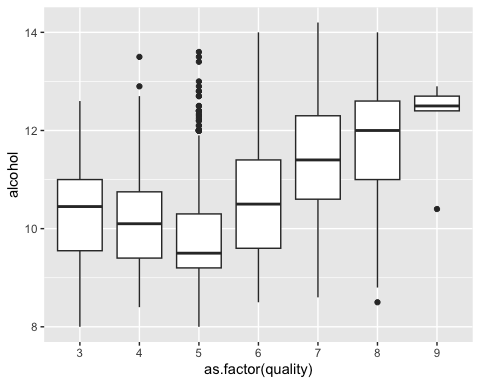
ggplot(whitequality, aes(as.factor(quality),pH))+ geom\_boxplot()



ggplot(whitequality, aes(as.factor(quality),sulphates))+ geom\_boxplot()



ggplot(whitequality, aes(as.factor(quality),alcohol))+ geom\_boxplot()



##Create training and testing dataset)  
set.seed(1)  
index <- sort(sample(nrow(whitequality),nrow(whitequality)\*0.85))  
train <- whitequality[index,]  
test <- whitequality[-index,]  
  
##linear regression  
lm.fit <- lm(quality~., data = train)  
summary(lm.fit)

##   
## Call:  
## lm(formula = quality ~ ., data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.8123 -0.4922 -0.0339 0.4669 3.1278   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.559e+02 2.003e+01 7.784 8.79e-15 \*\*\*  
## fixed.acidity 6.482e-02 2.249e-02 2.882 0.00397 \*\*   
## volatile.acidity -1.807e+00 1.238e-01 -14.602 < 2e-16 \*\*\*  
## citric.acid 2.074e-02 1.048e-01 0.198 0.84318   
## residual.sugar 8.314e-02 8.086e-03 10.282 < 2e-16 \*\*\*  
## chlorides -1.915e-01 5.972e-01 -0.321 0.74852   
## free.sulfur.dioxide 3.820e-03 9.086e-04 4.204 2.68e-05 \*\*\*  
## total.sulfur.dioxide -4.659e-04 4.126e-04 -1.129 0.25892   
## density -1.560e+02 2.032e+01 -7.677 2.02e-14 \*\*\*  
## pH 6.942e-01 1.141e-01 6.083 1.29e-09 \*\*\*  
## sulphates 7.012e-01 1.095e-01 6.404 1.68e-10 \*\*\*  
## alcohol 1.839e-01 2.582e-02 7.124 1.23e-12 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.7586 on 4151 degrees of freedom  
## Multiple R-squared: 0.2804, Adjusted R-squared: 0.2785   
## F-statistic: 147.1 on 11 and 4151 DF, p-value: < 2.2e-16

pre.lm <- predict(lm.fit, test)  
mean((pre.lm-test$quality)^2)

## [1] 0.5038929

train\_control <- trainControl(method = "cv",  
 number = 10)  
model <- train(quality ~., data = test,   
 method = "lm",  
 trControl = train\_control)  
print(model)

## Linear Regression   
##   
## 735 samples  
## 11 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 662, 660, 661, 662, 661, 661, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 0.7162831 0.2894722 0.5693872  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE

#glm  
glm.fit <- glm(quality~., data=train)  
summary(glm.fit)

##   
## Call:  
## glm(formula = quality ~ ., data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.8123 -0.4922 -0.0339 0.4669 3.1278   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.559e+02 2.003e+01 7.784 8.79e-15 \*\*\*  
## fixed.acidity 6.482e-02 2.249e-02 2.882 0.00397 \*\*   
## volatile.acidity -1.807e+00 1.238e-01 -14.602 < 2e-16 \*\*\*  
## citric.acid 2.074e-02 1.048e-01 0.198 0.84318   
## residual.sugar 8.314e-02 8.086e-03 10.282 < 2e-16 \*\*\*  
## chlorides -1.915e-01 5.972e-01 -0.321 0.74852   
## free.sulfur.dioxide 3.820e-03 9.086e-04 4.204 2.68e-05 \*\*\*  
## total.sulfur.dioxide -4.659e-04 4.126e-04 -1.129 0.25892   
## density -1.560e+02 2.032e+01 -7.677 2.02e-14 \*\*\*  
## pH 6.942e-01 1.141e-01 6.083 1.29e-09 \*\*\*  
## sulphates 7.012e-01 1.095e-01 6.404 1.68e-10 \*\*\*  
## alcohol 1.839e-01 2.582e-02 7.124 1.23e-12 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for gaussian family taken to be 0.575438)  
##   
## Null deviance: 3319.5 on 4162 degrees of freedom  
## Residual deviance: 2388.6 on 4151 degrees of freedom  
## AIC: 9527.5  
##   
## Number of Fisher Scoring iterations: 2

pre.glm <- predict(glm.fit, test)  
mean((pre.glm-test$quality)^2)

## [1] 0.5038929

train\_control <- trainControl(method = "cv",  
 number = 10)  
model <- train(quality ~., data = test,   
 method = "glm",  
 trControl = train\_control)  
print(model)

## Generalized Linear Model   
##   
## 735 samples  
## 11 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 662, 661, 662, 661, 660, 661, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 0.7145056 0.2856299 0.567801

#LDA  
lda.fit <- lda(quality~., train)  
lda.predict <- predict(lda.fit, test)  
table(lda.predict$class, test$quality)

##   
## 4 5 6 7 8 9  
## 3 0 0 0 0 0 0  
## 4 2 10 1 0 0 0  
## 5 6 114 69 5 1 0  
## 6 6 106 244 84 15 0  
## 7 1 1 22 37 10 1  
## 8 0 0 0 0 0 0  
## 9 0 0 0 0 0 0

mean(lda.predict$class !=test$quality)

## [1] 0.4598639

summary(lda.fit)

## Length Class Mode   
## prior 7 -none- numeric   
## counts 7 -none- numeric   
## means 77 -none- numeric   
## scaling 66 -none- numeric   
## lev 7 -none- character  
## svd 6 -none- numeric   
## N 1 -none- numeric   
## call 3 -none- call   
## terms 3 terms call   
## xlevels 0 -none- list

##Naive Bayes  
nb.fit <- naiveBayes(quality~., data = train)  
nb.predict <- predict(nb.fit, test)  
table(nb.predict,test$quality)

##   
## nb.predict 4 5 6 7 8 9  
## 3 0 3 2 0 0 0  
## 4 3 12 2 0 0 0  
## 5 4 122 97 16 1 0  
## 6 5 70 128 21 8 0  
## 7 3 24 105 87 17 1  
## 8 0 0 2 2 0 0  
## 9 0 0 0 0 0 0

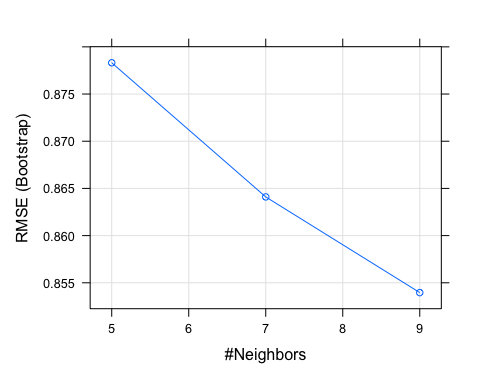
mean(nb.predict!=test$quality)

## [1] 0.537415

#knn  
model\_knn <- train(  
 quality ~.,  
 data = train,  
 method = 'knn'  
)  
model\_knn

## k-Nearest Neighbors   
##   
## 4163 samples  
## 11 predictor  
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 4163, 4163, 4163, 4163, 4163, 4163, ...   
## Resampling results across tuning parameters:  
##   
## k RMSE Rsquared MAE   
## 5 0.8783131 0.1432980 0.6699360  
## 7 0.8640992 0.1406604 0.6724278  
## 9 0.8539588 0.1402448 0.6708039  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was k = 9.

plot(model\_knn)



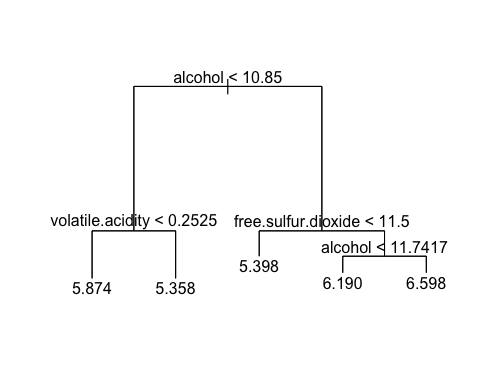
pre.knn <- predict(model\_knn, test)  
mean((pre.knn-test$quality)^2)

## [1] 0.5933796

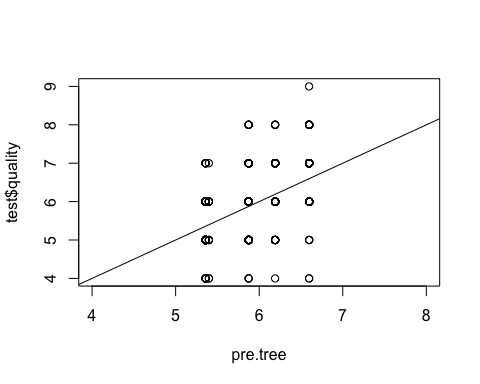
## Regression Tree  
set.seed(2)  
tree.fit=tree(quality~.,train)  
summary(tree.fit)

##   
## Regression tree:  
## tree(formula = quality ~ ., data = train)  
## Variables actually used in tree construction:  
## [1] "alcohol" "volatile.acidity" "free.sulfur.dioxide"  
## Number of terminal nodes: 5   
## Residual mean deviance: 0.5938 = 2469 / 4158   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -3.5980 -0.3585 0.1263 0.0000 0.6415 3.6420

plot(tree.fit)  
text(tree.fit,pretty=0)



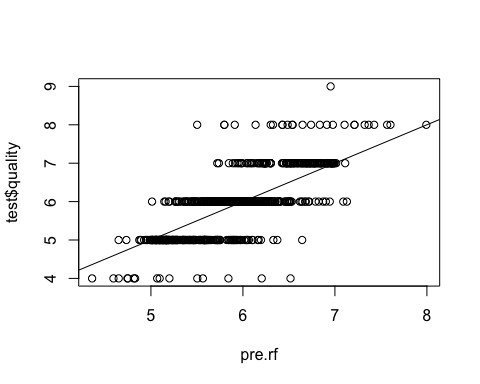
pre.tree <- predict(tree.fit,test)  
plot(pre.tree,test$quality,xlim = c(4,8))  
abline(0,1)



mean((pre.tree-test$quality)^2)

## [1] 0.5156528

## Random Forest  
rf.fit<- randomForest(quality ~.,data=train, importance=TRUE)  
pre.rf <- predict(rf.fit, test)  
plot(pre.rf, test$quality)  
abline(0,1)



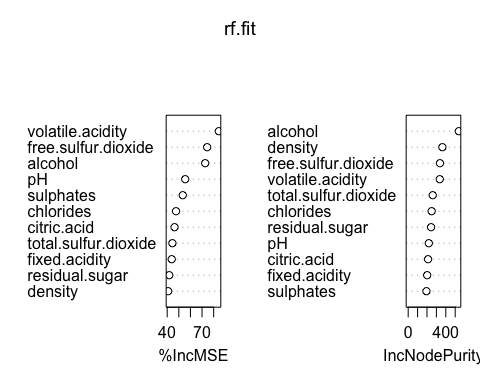
mean((pre.rf-test$quality)^2)

## [1] 0.3056975

importance(rf.fit)

## %IncMSE IncNodePurity  
## fixed.acidity 43.89827 202.0124  
## volatile.acidity 84.40867 336.0000  
## citric.acid 46.31903 211.4250  
## residual.sugar 41.77202 243.4484  
## chlorides 47.60513 249.7338  
## free.sulfur.dioxide 74.38997 338.1172  
## total.sulfur.dioxide 44.47585 261.2226  
## density 40.85550 363.7510  
## pH 55.51724 219.4972  
## sulphates 53.41391 194.0878  
## alcohol 72.71656 538.1510

varImpPlot(rf.fit)



train\_control <- trainControl(method = "cv",  
 number = 10)  
model <- train(quality ~., data = test,   
 method = "rf",  
 trControl = train\_control)

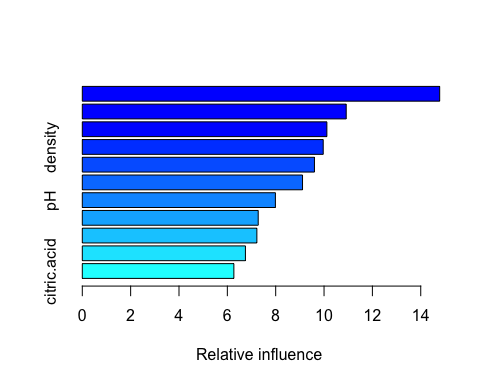
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has  
## five or fewer unique values. Are you sure you want to do regression?

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has  
## five or fewer unique values. Are you sure you want to do regression?  
  
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has  
## five or fewer unique values. Are you sure you want to do regression?

print(model)

## Random Forest   
##   
## 735 samples  
## 11 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 663, 660, 661, 662, 662, 662, ...   
## Resampling results across tuning parameters:  
##   
## mtry RMSE Rsquared MAE   
## 2 0.6596069 0.3961041 0.5222419  
## 6 0.6633301 0.3820868 0.5180283  
## 11 0.6678715 0.3741224 0.5201518  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was mtry = 2.

## Boosting  
set.seed(1)  
boost.fit=gbm(quality~.,data=train,distribution="gaussian",n.trees=5000,interaction.depth=4)  
summary(boost.fit)



## var rel.inf  
## alcohol alcohol 14.783529  
## free.sulfur.dioxide free.sulfur.dioxide 10.916901  
## volatile.acidity volatile.acidity 10.117072  
## density density 9.965432  
## total.sulfur.dioxide total.sulfur.dioxide 9.603342  
## residual.sugar residual.sugar 9.109811  
## pH pH 7.987842  
## fixed.acidity fixed.acidity 7.276761  
## chlorides chlorides 7.221477  
## sulphates sulphates 6.748020  
## citric.acid citric.acid 6.269813

pre.boost=predict(boost.fit,newdata=test,n.trees=5000)  
mean((pre.boost-test$quality)^2)

## [1] 0.3946781