Own_choice_wine_quality

yumeng

December 26, 2019

Summary

In this study, we will explore some machine learning methods on white wine quality data and try to predict wine quality. The quality score are treated as continuous data, and first we build a linear regression model as our baseline model, then several other methods such as logistic regression, K nearest neighbours etc. The RMSE is the loss function we used in this study, and based on the results we get, random forest has the least RMSE which is 0.64374. And the three most important factor to the final quality predict are volatile acidity, free sulfur dioxide and alcohol.

Step1, let's select the work directory and load all the packages

```r
my_dir <- choose.dir()  #if it doesn't work, please set your own work directory
setwd(my_dir)
print(paste("The director will be used is ", my_dir, sep = ":"))
```

```
## [1] "The director will be used is :C:\\001 files\\R related"
```

```r
options(digits = 5)

packages_used <- c("tidyverse", "scales", "data.table", "caret", "rpart", "rpart.plot",
    "corrplot", "rmarkdown", "knitr")

for (i in packages_used) {
    if (!requireNamespace(i)) {
        install.packages(i)
    }
}

library(tidyverse)
library(scales)
library(data.table)
library(caret)
library(rpart)
library(rpart.plot)
library(corrplot)
library(rmarkdown)
library(knitr)
```

Then download the white wine quality data We will use later

```
dl <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white
    destfile = dl, mode = "wb")

wine_white <- fread(dl, sep = ";")
setnames(wine_white, c("fixed_acidity", "volatile_acidity", "citric_acid", "residual_sugar",
    "chlorides", "free_sulfur_dioxide", "total_sulfur_dioxide", "density", "PH",
    "sulphates", "alcohol", "quality"))
save.image("wine_quality.rda")
```

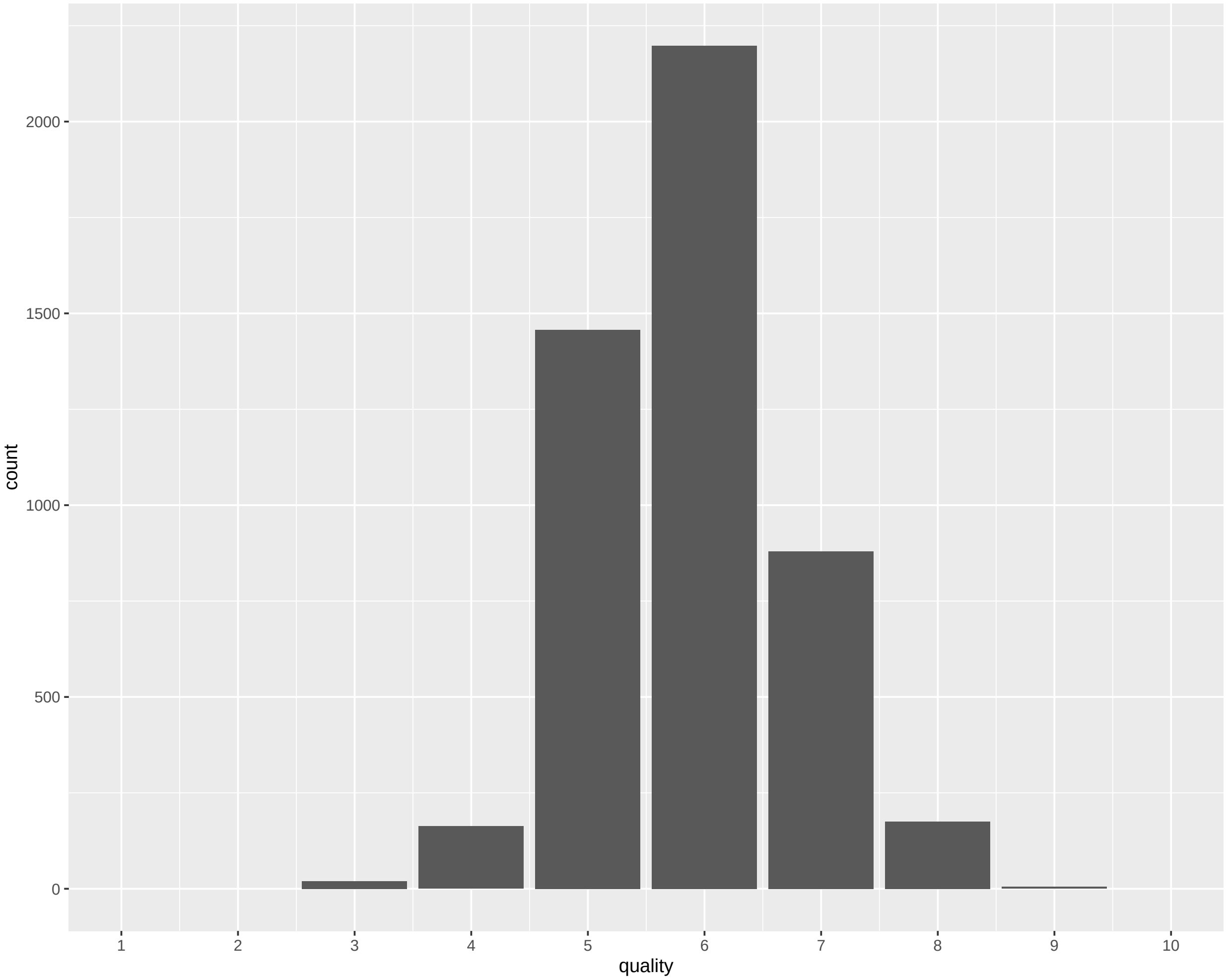
Load data and create train & test set, 10% of the data will be our test set

```
load("wine_quality.rda")
set.seed(7, sample.kind = "Rounding")  # set.seed(7) if R version is 3.5 or before
index <- createDataPartition(wine_white$quality, times = 1, p = 0.1, list = FALSE)
train_white <- wine_white %>% slice(-index)
test_white <- wine_white %>% slice(index)
```

Several algorithms will be introduced to see which one has better performance

Before modeling, let's check the distribution of our wine quality scores

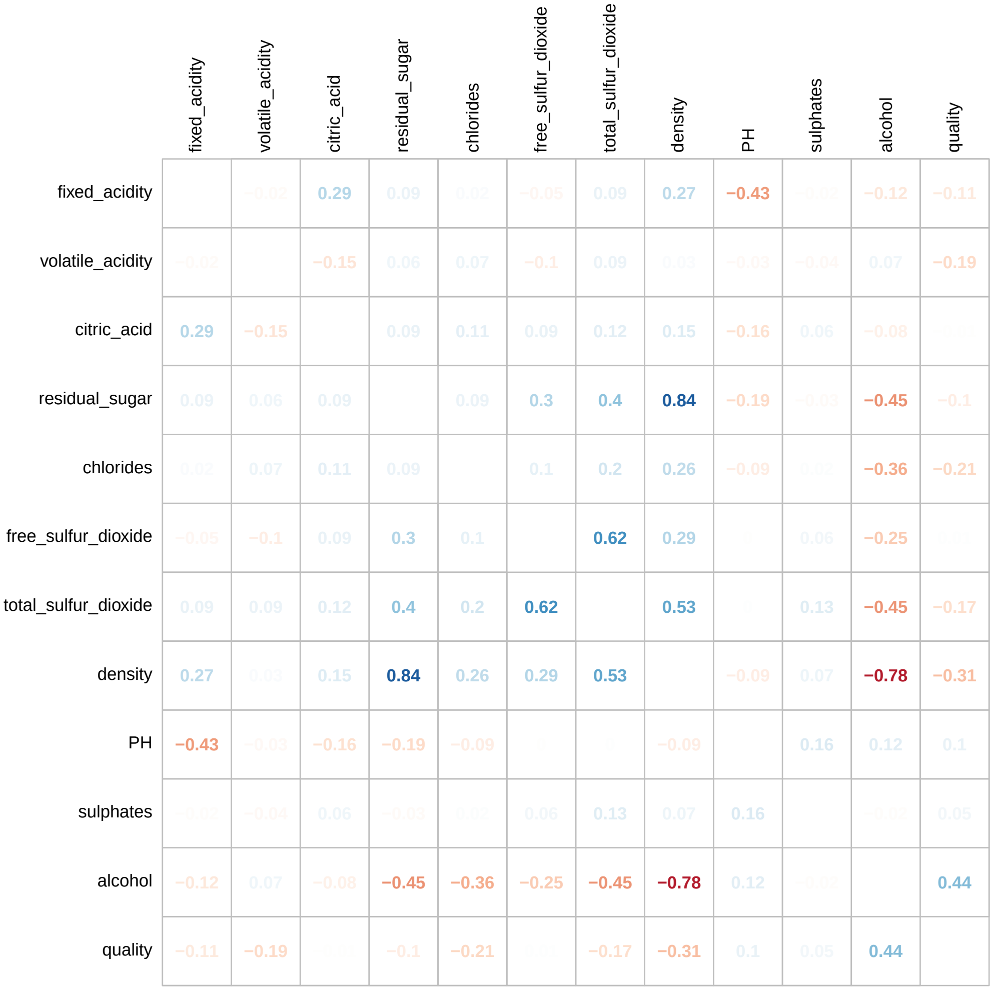
```
wine_white %>% ggplot(aes(quality)) + geom_bar(position = "dodge") + scale_x_continuous(limits = c(1,
    10), breaks = seq(1, 10, 1))
```



we can see most scores of white wine is 6

First a simple regression, let's have a quick look at the linear correlations between each variable

```
corrplot(cor(wine_white), addCoef.col = "grey", rect.col = "blue", method = "number",
    number.cex = 0.8, diag = FALSE, tl.pos = "lt", cl.pos = "n", tl.cex = 0.8, tl.col = "black")
```



Define the loss function

```
rmse <- function(x, y) {
    sqrt(mean((x - y)^2))
}
```

Based on the correlation matrix, "alcohol","density" and "chlorides" have higher linear correlation to the quality scores, let's find out if they play important roles in our linear models

First at first, we build a linear model that used all variables then use step regression to prun the model

```
white_lm_all <- lm(quality ~ ., data = train_white)
white_lm <- step(white_lm_all)
```

```
## Start:  AIC=-2526.5
## quality ~ fixed_acidity + volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     density + PH + sulphates + alcohol
##
##                        Df Sum of Sq  RSS   AIC
## - citric_acid           1       0.0 2471 -2528
## - total_sulfur_dioxide  1       0.2 2472 -2528
## - chlorides             1       0.5 2472 -2528
## <none>                              2471 -2526
## - fixed_acidity         1       4.3 2476 -2521
## - free_sulfur_dioxide   1       8.8 2480 -2513
## - sulphates             1      17.3 2489 -2498
## - PH                    1      21.1 2493 -2491
## - density               1      28.9 2500 -2477
## - alcohol               1      37.2 2509 -2463
## - residual_sugar        1      56.7 2528 -2428
## - volatile_acidity      1     140.8 2612 -2284
##
## Step:  AIC=-2528.5
## quality ~ fixed_acidity + volatile_acidity + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     density + PH + sulphates + alcohol
##
##                        Df Sum of Sq  RSS   AIC
## - total_sulfur_dioxide  1       0.2 2472 -2530
## - chlorides             1       0.5 2472 -2530
## <none>                              2471 -2528
## - fixed_acidity         1       4.4 2476 -2523
## - free_sulfur_dioxide   1       8.8 2480 -2515
## - sulphates             1      17.3 2489 -2500
## - PH                    1      21.2 2493 -2493
## - density               1      29.1 2501 -2479
## - alcohol               1      37.5 2509 -2464
## - residual_sugar        1      56.9 2528 -2430
## - volatile_acidity      1     145.2 2617 -2279
##
## Step:  AIC=-2530.1
## quality ~ fixed_acidity + volatile_acidity + residual_sugar +
##     chlorides + free_sulfur_dioxide + density + PH + sulphates +
##     alcohol
##
##                       Df Sum of Sq  RSS   AIC
## - chlorides            1       0.5 2472 -2531
## <none>                             2472 -2530
## - fixed_acidity        1       4.5 2476 -2524
## - free_sulfur_dioxide  1      11.3 2483 -2512
## - sulphates            1      17.1 2489 -2502
## - PH                   1      21.3 2493 -2494
## - density              1      31.2 2503 -2477
## - alcohol              1      37.4 2509 -2466
## - residual_sugar       1      58.7 2530 -2429
## - volatile_acidity     1     152.7 2624 -2268
##
```

```
## Step:  AIC=-2531.3
## quality ~ fixed_acidity + volatile_acidity + residual_sugar +
##     free_sulfur_dioxide + density + PH + sulphates + alcohol
##
##                       Df Sum of Sq  RSS   AIC
## <none>                             2472 -2531
## - fixed_acidity        1       5.1 2477 -2524
## - free_sulfur_dioxide  1      11.1 2483 -2514
## - sulphates            1      17.4 2490 -2502
## - PH                   1      23.1 2495 -2492
## - density              1      33.7 2506 -2474
## - alcohol              1      37.4 2510 -2467
## - residual_sugar       1      63.7 2536 -2421
## - volatile_acidity     1     155.4 2628 -2264
```

```
rmse_results <- data.frame(method = c("linear model selected variables", "linear model all variables"),
    RMSE = c(rmse(predict(white_lm, test_white), test_white$quality), rmse(predict(white_lm_all,
        test_white), test_white$quality)))
knitr::kable(rmse_results)
```

method	RMSE
linear model selected variables	0.76519
linear model all variables	0.76576

```
knitr::kable(broom::tidy(white_lm))
```

term	estimate	std.error	statistic	p.value
(Intercept)	146.39620	18.66454	7.8435	0.00000
fixed_acidity	0.06400	0.02130	3.0040	0.00268
volatile_acidity	-1.90176	0.11435	-16.6304	0.00000
residual_sugar	0.08056	0.00757	10.6460	0.00000
free_sulfur_dioxide	0.00314	0.00071	4.4386	0.00001
density	-146.54564	18.91822	-7.7463	0.00000
PH	0.69050	0.10779	6.4060	0.00000
sulphates	0.58189	0.10471	5.5570	0.00000
alcohol	0.20267	0.02486	8.1528	0.00000

we find after pruning, only density and alcohol are statistically significant in the linear model.

```
print(paste("The baseline RMSE of white wine is", format(RMSE(predict(white_lm, test_white),
    test_white$quality), digits = 5), sep = ":"))
```

```
## [1] "The baseline RMSE of white wine is:0.76519"
```

Now other methods will be tested

```
set.seed(1, sample.kind = "Rounding")
models <- c("glm", "svmLinear", "knn", "gamLoess", "rf", "rpart")
fits <- lapply(models, function(model) {
    print(model)
    train(quality ~ ., method = model, trControl = trainControl(method = "cv"), data = train_white)
})
```

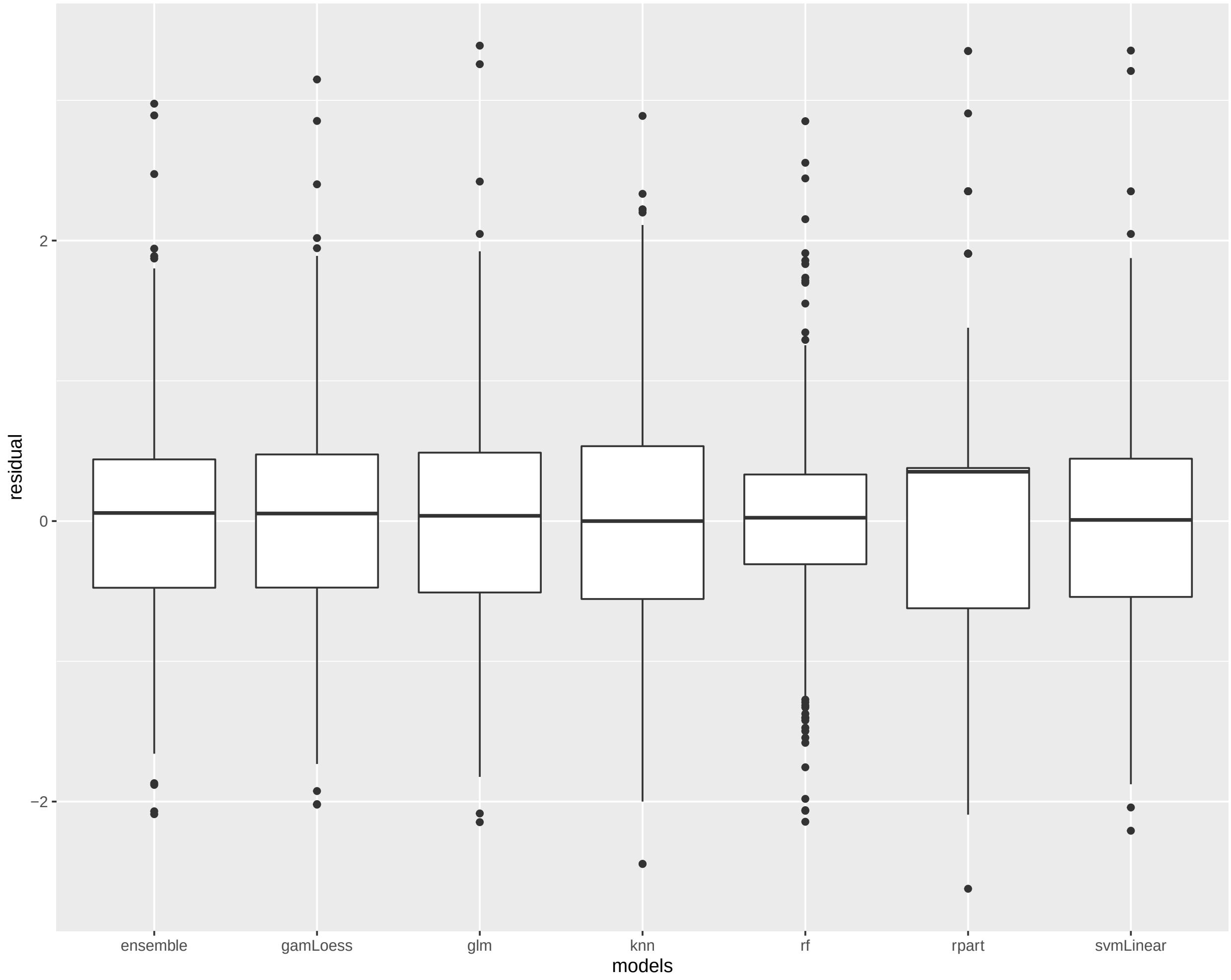
```
## [1] "glm"
## [1] "svmLinear"
## [1] "knn"
## [1] "gamLoess"
## [1] "rf"
## [1] "rpart"
```

```
names(fits) <- models
my_predict <- function(x) {
    predict(x, test_white)
}
predict <- sapply(fits, my_predict)
ensemble_pred <- rowMeans(predict)
predict <- as.data.frame(predict) %>% mutate(ensemble = ensemble_pred)
auu <- function(x) {
    RMSE(x, test_white$quality)
}
rmses <- apply(predict, 2, auu)
rmses <- as.data.frame(rmses)
rmses$method <- row.names(rmses)
names(rmses) <- c("RMSE", "method")
rmse_results <- bind_rows(rmse_results, rmses)
knitr::kable(rmses)
```

	RMSE	method
glm	0.76576	glm
svmLinear	0.76703	svmLinear
knn	0.81265	knn
gamLoess	0.73013	gamLoess
rf	0.64374	rf
rpart	0.83079	rpart
ensemble	0.72198	ensemble

we could see the prediction more clear in the image below

```
residual <- predict - test_white$quality
residual %>% pivot_longer(cols = 1:7, names_to = "models", values_to = "residual") %>%
    ggplot(aes(models, residual)) + geom_boxplot()
```



```
print(paste("The best performed algorithm is random forest", format(rmses[5, 1],
    5), sep = ":"))
```

```
## [1] "The best performed algorithm is random forest:0.64374"
```

Random forest not only has the least RMSE but also narrowest residual range

```
set.seed(3, sample.kind = "Rounding")
sqtmtry <- round(sqrt(ncol(train_white) - 1))
rfGrid <- expand.grid(mtry = c(round(sqtmtry/2), sqtmtry, 2 * sqtmtry))
# below code may take a while
rf_fit <- train(quality ~ ., method = "rf", data = train_white, tuneGrid = rfGrid,
    nodesize = 5, importance = T)

predict_rf <- predict(rf_fit, test_white)

imp <- varImp(rf_fit)
```

```r
imp$importance %>% as.data.frame() %>% mutate(var = rownames(imp$importance)) %>%
    arrange(desc(Overall)) %>% knitr::kable()
```

Overall	var
100.000	volatile_acidity
81.788	free_sulfur_dioxide
67.660	alcohol
34.152	PH
23.735	chlorides
20.277	sulphates
17.996	citric_acid
11.445	fixed_acidity
10.824	total_sulfur_dioxide
10.047	residual_sugar
0.000	density

```r
rmse_results <- bind_rows(rmse_results, data.frame(method = "tuned rf", RMSE = RMSE(predict_rf,
    test_white$quality)))
knitr::kable(rmse_results)
```

method	RMSE
linear model selected variables	0.76519
linear model all variables	0.76576
glm	0.76576
svmLinear	0.76703
knn	0.81265
gamLoess	0.73013
rf	0.64374
rpart	0.83079
ensemble	0.72198
tuned rf	0.64236

It’s slighly doing better than the aotmatically tuned rf, the final RMSE is 0.64236

Conclusion

In this study, several machine learning methods are applied on white wine quality data to predict wine quality. Random forest has the least RMSE which is 0.64236. And the three most important factor to the final quality predict are volatile acidity, free sulfur dioxide and alcohol.