Code ▾

Advanced Predict Assignment Writeup

Wendi YANG_67609_EDHEC Business School

1.Overview

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it.

The objective of this report is to demonstrate the process employed to arrive at a prediction algorithm, which aims to classify the manner in which the participants employed certain exercises. The data comes from accelerometers attached on the belt, forearm and dumbells.

2.Introduction

The goal of Predict Assignment Writeup is to predict the manner in which 6 participants did a weight lifting exercise. For this, I downloaded a training dataset and a test dataset and then created a model, used cross validation, calculated an expected out of sample error. In this write-up, I would also describe how I built the model, and why I made the choices that I did. I also used the model to predict the 20 test cases.

3.Dataset Overview

3.1 Dataset Loading

From the URL provided from the Coursera, I download from the link and then get the original training dataset and test dataset.

Hide

```
#Environment uploading R libraries
install.packages(c("lattice","ggplot2","dplyr","randomForest","gridExtra","rattle","t
ibble","bitops"))
```

```
Error in install.packages : Updating loaded packages
```

Hide

```
library(ggplot2)
library(rattle)
library(dplyr)
library(randomForest)
library(corrplot)
library(rpart)
library(rpart.plot)
library(knitr)
library(caret)
library(corrplot)
library(lattice)
library(gridExtra)
library(grid)

#data loading
urltrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
urltest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(urltrain), na.strings = c("NA","#DIV/0!",""))
install.packages(c("lattice", "ggplot2", "dplyr", "randomForest", "gridExtra", "rattl
e", "tibble", "bitops"))
```

```
  There is a binary version available but the source version is later:
```

	binary	source	needs_compilation
	<fctr>	<fctr>	<lgl>
ggplot2	3.3.0	3.3.1	FALSE

1 row

```
试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/lattice_0.20-41.tg
z'
Content type 'application/x-gzip' length 1155029 bytes (1.1 MB)
==================================================
downloaded 1.1 MB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/dplyr_0.8.5.tgz'
Content type 'application/x-gzip' length 6859111 bytes (6.5 MB)
==================================================
downloaded 6.5 MB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/randomForest_4.6-1
4.tgz'
Content type 'application/x-gzip' length 253893 bytes (247 KB)
==================================================
downloaded 247 KB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/gridExtra_2.3.tgz'
Content type 'application/x-gzip' length 1103470 bytes (1.1 MB)
==================================================
downloaded 1.1 MB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/rattle_5.4.0.tgz'
Content type 'application/x-gzip' length 5479591 bytes (5.2 MB)
==================================================
downloaded 5.2 MB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/tibble_3.0.1.tgz'
Content type 'application/x-gzip' length 389871 bytes (380 KB)
==================================================
downloaded 380 KB

试开URL’https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.6/bitops_1.0-6.tgz'
Content type 'application/x-gzip' length 25079 bytes (24 KB)
==================================================
downloaded 24 KB
```

```


The downloaded binary packages are in
	/var/folders/m7/6jxb2sbx3sq25ps_z60wgqp80000gn/T//RtmpWonVln/downloaded_packages
```

```
installing the source package ‘ggplot2’

试开URL’https://cran.rstudio.com/src/contrib/ggplot2_3.3.1.tar.gz'
Content type 'application/x-gzip' length 3035612 bytes (2.9 MB)
==================================================
downloaded 2.9 MB

* installing *source* package ‘ggplot2’ ...
** 成功将‘ggplot2’程序包解包并MD5和检查
** using staged installation
** R
** data
*** moving datasets to lazyload DB
** inst
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
*** copying figures
** building package indices
** installing vignettes
** testing if installed package can be loaded from temporary location
** testing if installed package can be loaded from final location
** testing if installed package keeps a record of temporary installation path
* DONE (ggplot2)

The downloaded source packages are in
    ‘/private/var/folders/m7/6jxb2sbx3sq25ps_z60wgqp80000gn/T/RtmpWonVln/downloaded_p
ackages’
```

Hide

```
testing <- read.csv(url(urltest), na.strings = c("NA","#DIV/0!",""))

dim(training)
```

```
[1] 19622   160
```

Hide

```
dim(testing)
```

```
[1]  20 160
```

Hide

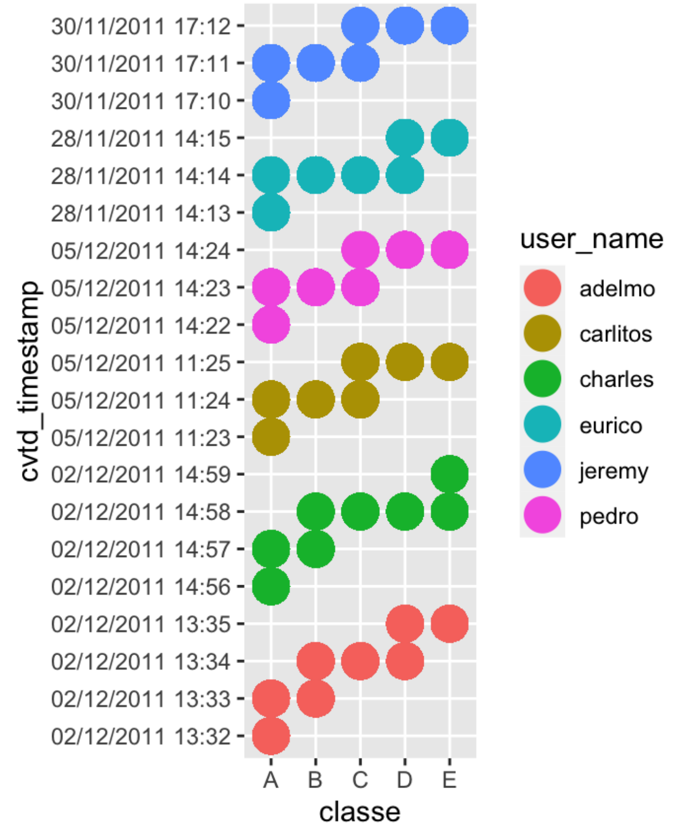
```
head(sapply(training, function(x) sum(is.na(x))), n=30)
```

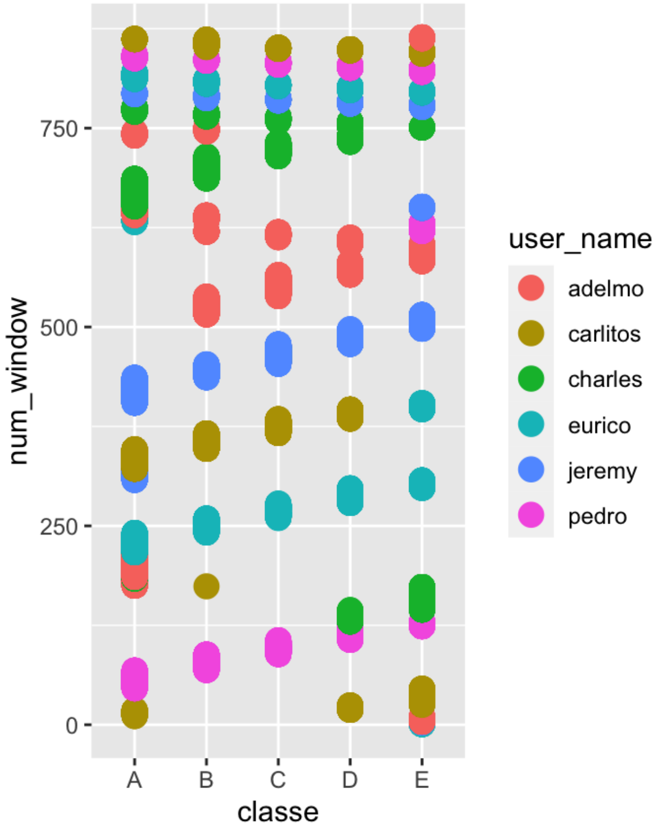
```
                   X           user_name raw_timestamp_part_1 raw_timestamp_part_2
cvtd_timestamp
                   0                   0                    0                    0
0
          new_window          num_window            roll_belt           pitch_belt
yaw_belt
                   0                   0                    0                    0
0
    total_accel_belt   kurtosis_roll_belt  kurtosis_picth_belt    kurtosis_yaw_belt
skewness_roll_belt
                   0               19226                19248                19622
19225
skewness_roll_belt.1    skewness_yaw_belt        max_roll_belt       max_picth_belt
max_yaw_belt
               19248               19622                19216                19216
19226
       min_roll_belt       min_pitch_belt         min_yaw_belt  amplitude_roll_belt a
mplitude_pitch_belt
               19216               19216                19226                19216
19216
  amplitude_yaw_belt var_total_accel_belt         avg_roll_belt     stddev_roll_belt
var_roll_belt
               19226               19216                19216                19216
19216
```

The dataset consists 160 variables in all, excluding the outcome variable “Classe”, there are 159 candidates to be included as predictors. However, a close examination of the data would indicate that some variables might not be usefull in this model. As we can see, some variables have plenty of missing values NA.

Hide

```
qplot1 <- qplot(classe, cvtd_timestamp, data = training, color=user_name,size=I(6))
qplot2 <-qplot(classe, num_window, data = training, color=user_name, size=I(4))
grid.arrange(qplot1, qplot2,ncol=2)
```





3.2 Exploratory Analysis

As there are many dummy variables in these data, several columns do not measurements for each observations, but are rather summary statistics for one sliding window. In the validation data which we would like to predict are just random draws of one observation at a particular time point. In this case, it's easy to omit many of the summary variables in the dataset as they are not usefull for particular prediction problem, and also seem to be mislabeled in some cases, such as the summary statistics in the wrong columns. The inappropriate structure of the variables is borne out by the near zero variances.

Hide

```
#create a partition with the training set
set.seed(12345)
inTrain <- createDataPartition(training$classe, p=0.8, list=FALSE)
trainset <- training[inTrain,]
testset <- training[-inTrain,]
dim(trainset)
```

```
[1] 15699   160
```

Hide

```
dim(testset)
```

```
[1] 3923  160
```

From the exploratory plots, the participants in this case all performed these trials in temporal order. They all started doing biceps curls the proper way in Class A, then proceeded with Class B then C etc. The relation is an artifact of the case design which would allow to predict the validation data with great accuracy, but may fall if I try to accurately predict new data given only accelerometer measurements. Therefor, I would exclude all of the near zero variables and ID variables, as it is based solely onb accelerometer measurements.

3.3 Data splitting

The original training dataset is partinioned in 2 to create a training set, which is 80% of the data as suggested by course for modeling process and a test set with the remaining 20% for validations. The test dataset keep as original and is only used for evaluate the final generation test.

Hide

```
#remove NA variables
na <- sapply(trainset, function(x) mean(is.na(x))) > 0.95
trainset <- trainset[, na==FALSE]
testset <- testset[, na==FALSE]
#remove near zero variance variables
nearzerovar <- nearZeroVar(trainset)
trainset <- trainset[,-nearzerovar]
testset <- testset[,-nearzerovar]
#remove columns 1 to 7 identification only variables
trainset <- trainset[,-(1:7)]
testset <- testset[,-(1:7)]
dim(trainset)
```

```
[1] 15699    52
```

Hide

```
dim(testset)
```

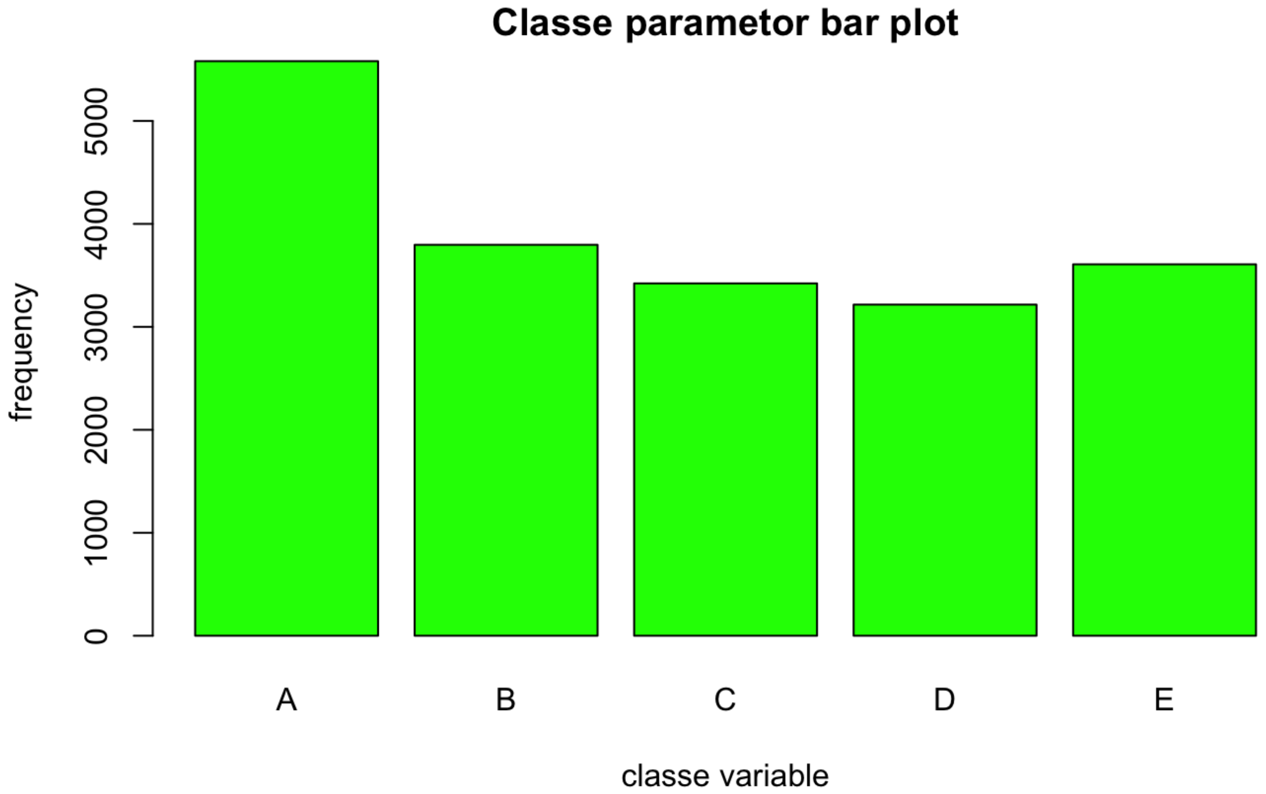
```
[1] 3923   52
```

3.4 Data cleaning

As resulted, the trainset and testset both have 160 variables, and to prevent the varibles contain NA values, then I remove all NA and Near Zero variance variables and ID varibles with cleaning procedures as below.

Hide

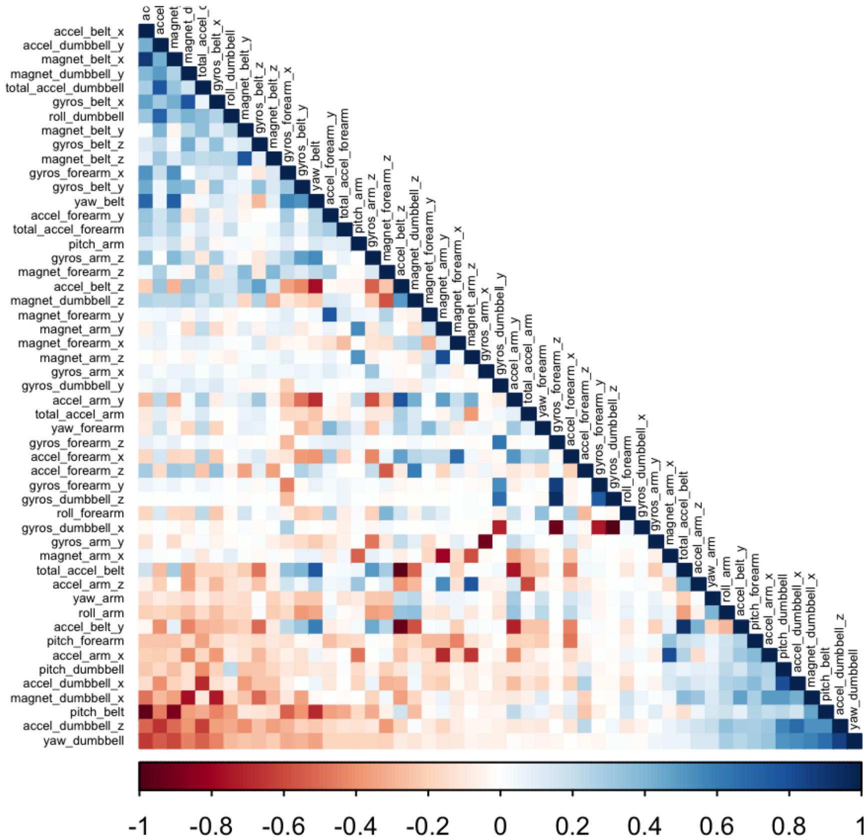
```
#plot the classe
plot(training$classe, col="green", main="Classe parametor bar plot", xlab="classe var
iable", ylab="frequency")
```



After clearning process, there remain 52 variables.

Hide

```
cormatrix <- cor(trainset[,-52])
corrplot(cormatrix, order="FPC", method="color", type="lower", tl.cex = 0.4, tl.col=
 rgb(0,0,0))
```



Hide

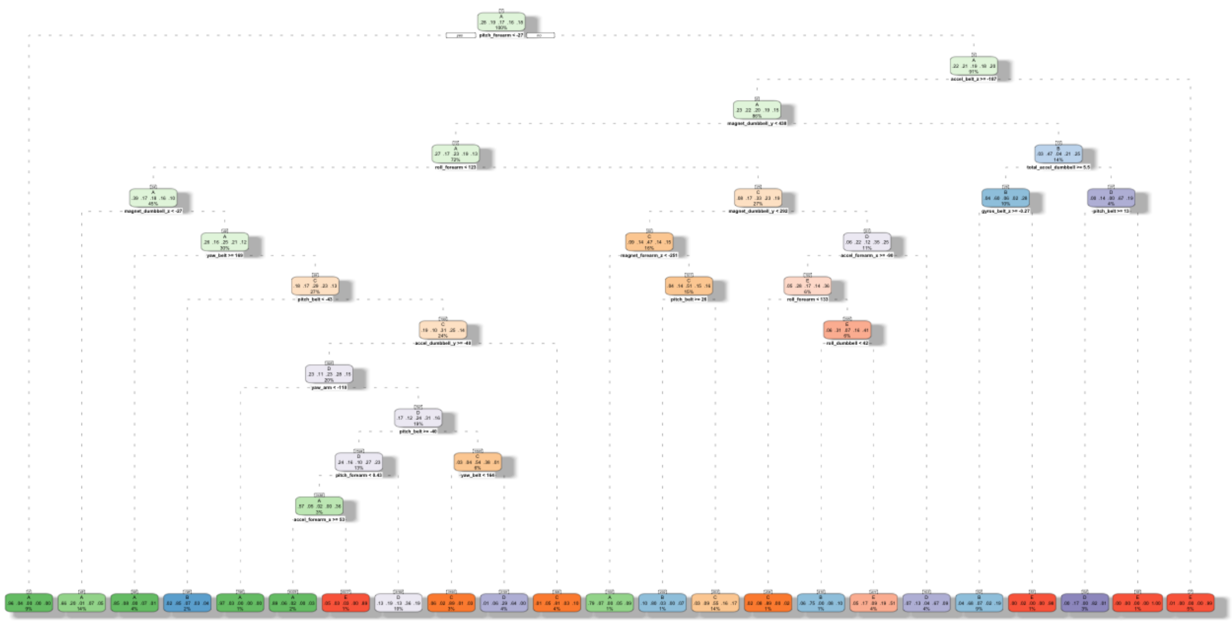
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     2.8908 2.8404 2.15722 2.06310 1.91698 1.73606
## Proportion of Variance 0.1607 0.1552 0.08949 0.08185 0.07067 0.05796
## Cumulative Proportion  0.1607 0.3159 0.40535 0.48721 0.55788 0.61584
##                           PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     1.4970 1.44260 1.31145 1.22700 1.18091 1.05869
## Proportion of Variance 0.0431 0.04002 0.03308 0.02895 0.02682 0.02155
## Cumulative Proportion  0.6589 0.69895 0.73203 0.76098 0.78780 0.80935
##                           PC13    PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.99735 0.93894 0.90818 0.88628 0.82585 0.76246
## Proportion of Variance 0.01913 0.01695 0.01586 0.01511 0.01312 0.01118
## Cumulative Proportion  0.82848 0.84544 0.86130 0.87640 0.88952 0.90070
##                           PC19    PC20    PC21    PC22    PC23    PC24
## Standard deviation     0.72281 0.69490 0.64577 0.63079 0.61252 0.58068
## Proportion of Variance 0.01005 0.00929 0.00802 0.00765 0.00722 0.00648
## Cumulative Proportion  0.91075 0.92003 0.92805 0.93570 0.94292 0.94940
##                           PC25    PC26    PC27   PC28    PC29    PC30
## Standard deviation     0.55190 0.54020 0.50381 0.4838 0.44827 0.42168
## Proportion of Variance 0.00586 0.00561 0.00488 0.0045 0.00386 0.00342
## Cumulative Proportion  0.95526 0.96087 0.96575 0.9703 0.97412 0.97754
##                           PC31    PC32    PC33    PC34    PC35    PC36
## Standard deviation     0.39737 0.36458 0.34743 0.33281 0.30361 0.28094
## Proportion of Variance 0.00304 0.00256 0.00232 0.00213 0.00177 0.00152
## Cumulative Proportion  0.98058 0.98313 0.98545 0.98758 0.98936 0.99087
##                           PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     0.25247 0.23698 0.23329 0.19946 0.19353 0.18415
## Proportion of Variance 0.00123 0.00108 0.00105 0.00077 0.00072 0.00065
## Cumulative Proportion  0.99210 0.99318 0.99423 0.99499 0.99571 0.99636
##                           PC43    PC44    PC45    PC46    PC47    PC48
## Standard deviation     0.17967 0.17258 0.16761 0.16217 0.14641 0.14235
## Proportion of Variance 0.00062 0.00057 0.00054 0.00051 0.00041 0.00039
## Cumulative Proportion  0.99699 0.99756 0.99810 0.99860 0.99902 0.99941
##                           PC49   PC50    PC51    PC52
## Standard deviation     0.11222 0.1012 0.07688 0.04626
## Proportion of Variance 0.00024 0.0002 0.00011 0.00004
## Cumulative Proportion  0.99965 0.9999 0.99996 1.00000
```

3.5 Correlation Analysis

Before proceeding the model procedures, we use the correlation analysis to see the correlation among variables analyzed.

Hide

```
#modelling fit
set.seed(12345)
ModfitDectree <- rpart(classe ~ ., data = trainset, method="class")
fancyRpartPlot(ModfitDectree)
```



Rattle 2020- 5-29 10:36:41 yangwendi

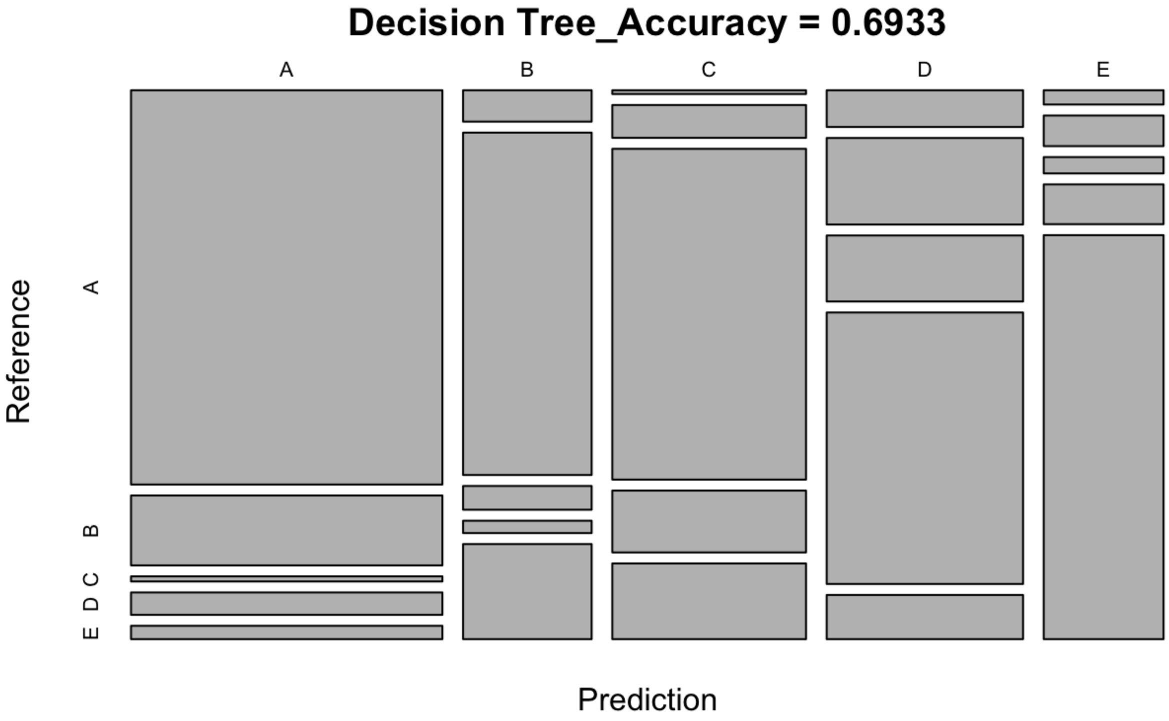
Hide

```
#prediction on testset
predictDectree <- predict(ModfitDectree, newdata=testset, type="class" )
confmatDectree <- confusionMatrix(predictDectree, testset$classe)
confmatDectree
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1004  178   13   57   34
         B   33  360   25   13  100
         C    6   52  524   98  120
         D   59  139  106  436   71
         E   14   30   16   39  396

Overall Statistics

               Accuracy : 0.6933
                 95% CI : (0.6787, 0.7078)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6108

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.8996  0.47431   0.7661   0.6781   0.5492
Specificity            0.8995  0.94595   0.9148   0.8857   0.9691
Pos Pred Value         0.7807  0.67797   0.6550   0.5376   0.8000
Neg Pred Value         0.9575  0.88237   0.9488   0.9335   0.9052
Prevalence             0.2845  0.19347   0.1744   0.1639   0.1838
Detection Rate         0.2559  0.09177   0.1336   0.1111   0.1009
Detection Prevalence   0.3278  0.13536   0.2039   0.2067   0.1262
Balanced Accuracy      0.8996  0.71013   0.8404   0.7819   0.7592
```

Hide

```
#plot matrix result
plot(confmatDectree$table, col=confmatDectree$byclass, main=paste("Decision Tree_Accu
racy =", round(confmatDectree$overall['Accuracy'], 4)))
```



The dark colors in the graph above are highly correlated variables. To make a more compact analysis, below I would also do a Principla Components Analysis to perform as pre-processing step to the datasets.

3.6 Principal Components Analysis(PCA)

As there are 52 candidate predictor for the model, it makes sense to employ a dimension reduction technique to manage this large number of predictors. PCA is used on the training set to determine key components among the predictors.

Hide

```
#modeling fit
controlRf <- trainControl(method = "cv", number=10, verboseIter = FALSE)
ModfitRandforest <- train(classe ~ ., data = trainset, method="rf", trControl=control
Rf)
ModfitRandforest$finalModel
```

```


Call:
 randomForest(x = x, y = y, mtry = param$mtry)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 26

        OOB estimate of  error rate: 0.55%
Confusion matrix:
     A    B    C    D    E  class.error
A 4460    3    0    0    1 0.0008960573
B   23 3007    6    1    1 0.0102040816
C    0   14 2719    5    0 0.0069393718
D    0    0   24 2547    2 0.0101049359
E    0    0    4    3 2879 0.0024255024
```

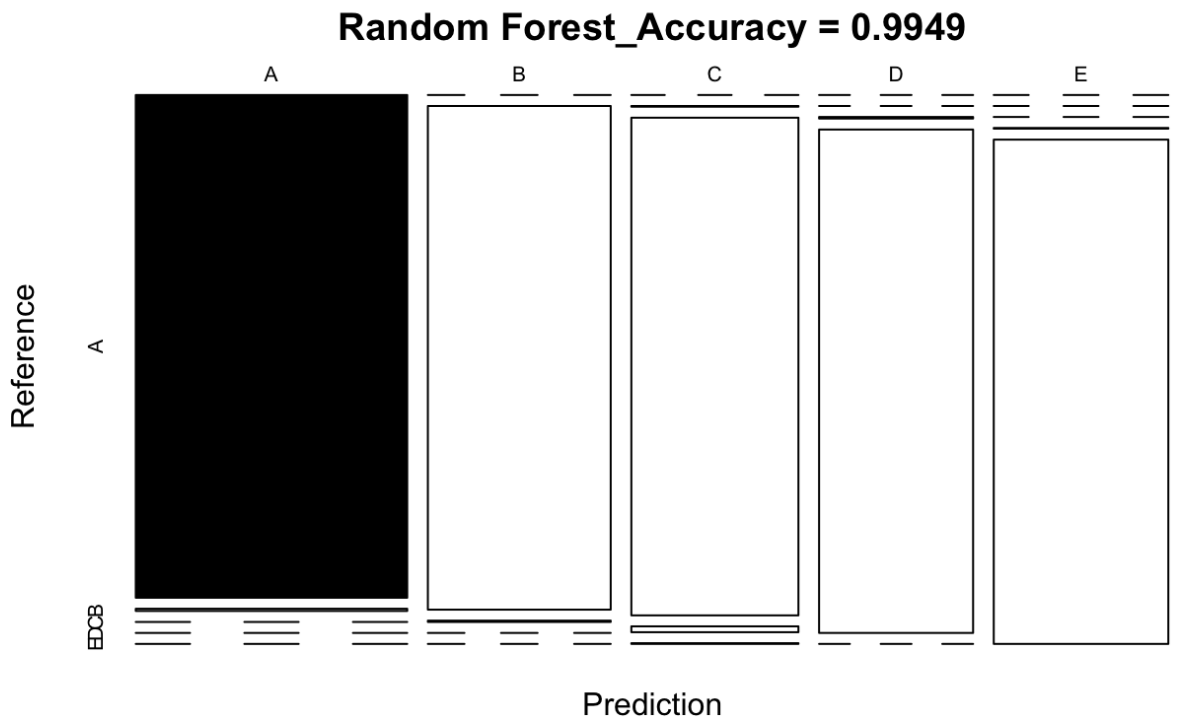
Hide

```
#plot
ModFitRF <- randomForest(classe ~ ., data = trainset, method = "rf", importance = T,
 trControl = trainControl(method = "cv", classProbs=TRUE,savePredictions=TRUE,allowPa
rallel=TRUE, number = 10))

#predition on testset
predictRanforest <- predict(ModfitRandforest, newdata=testset)
confmatRanforest <- confusionMatrix(predictRanforest, testset$classe)
confmatRanforest
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1116    5    0    0    0
         B    0  753    2    0    0
         C    0    1  680    8    1
         D    0    0    2  634    0
         E    0    0    0    1  720

Overall Statistics

               Accuracy : 0.9949
                 95% CI : (0.9921, 0.9969)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9936

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9921   0.9942   0.9860   0.9986
Specificity            0.9982   0.9994   0.9969   0.9994   0.9997
Pos Pred Value         0.9955   0.9974   0.9855   0.9969   0.9986
Neg Pred Value         1.0000   0.9981   0.9988   0.9973   0.9997
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2845   0.1919   0.1733   0.1616   0.1835
Detection Prevalence   0.2858   0.1925   0.1759   0.1621   0.1838
Balanced Accuracy      0.9991   0.9957   0.9955   0.9927   0.9992
```

Hide

```
#plot matrix result
plot(confmatRanforest$table, col=confmatRanforest$byClass, main = paste("Random Fores
t_Accuracy =", round(confmatRanforest$overall['Accuracy'], 4)))
```



4.Prediction Model Building

Four methods are applied to model the regression in the train dataset and the best one, which has highest accuracy when applied to the test dataset , would be used for the final predictions. The method used are: Decision Trees, Random Forests, Bagging and Boosting. And I also apply a matrix to present the accuracy below each model in order to select best model by comparing, described as following.

4.1 Decision Trees Method

In fact, it isn't expected the accuracy to be high under decision tree model, because anything around 80% would be acceptable.

Hide

```
#modeling fit
set.seed(12345)
controlbag <- trainControl(method="repeatedcv", number=10, repeats= 2)
modfitBag <- train(classe ~., data = trainset, method="gbm", trControl = controlbag,
 verbose=FALSE)
modfitBag$finalModel
```

```
A gradient boosted model with multinomial loss function.
150 iterations were performed.
There were 51 predictors of which 51 had non-zero influence.
```

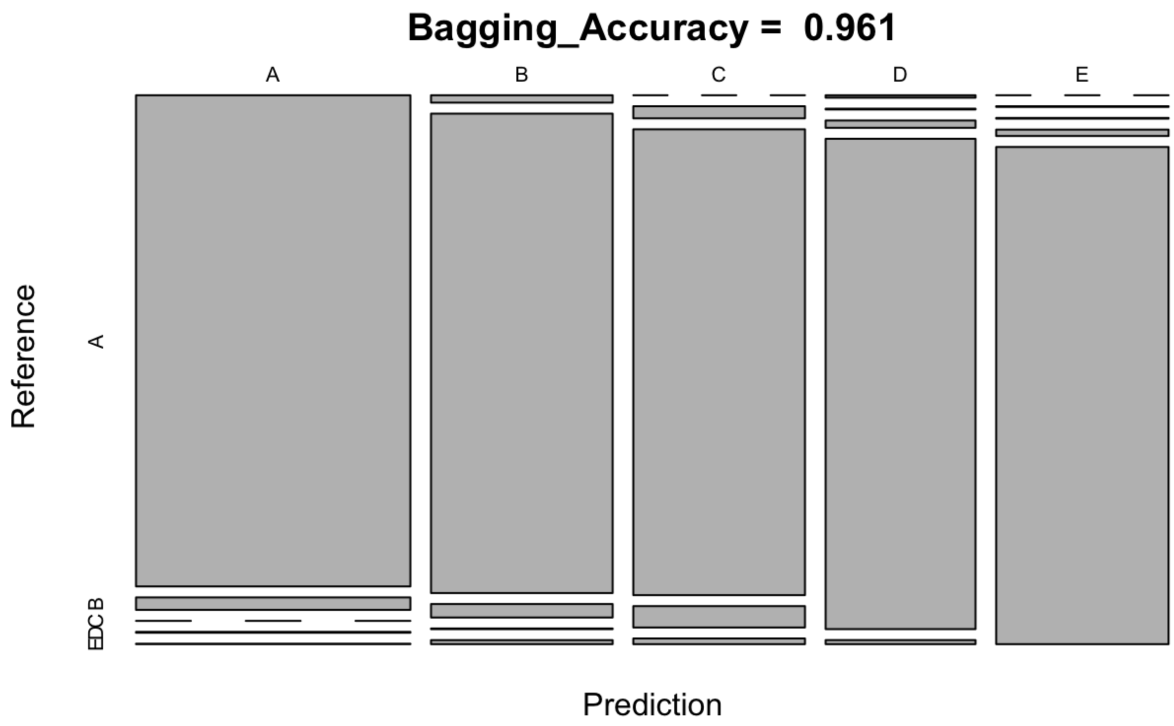
Hide

```
#prediction on testset
predictBag <- predict(modfitBag, newdata=testset)
confmatBag <- confusionMatrix(predictBag, testset$classe)
confmatBag
```

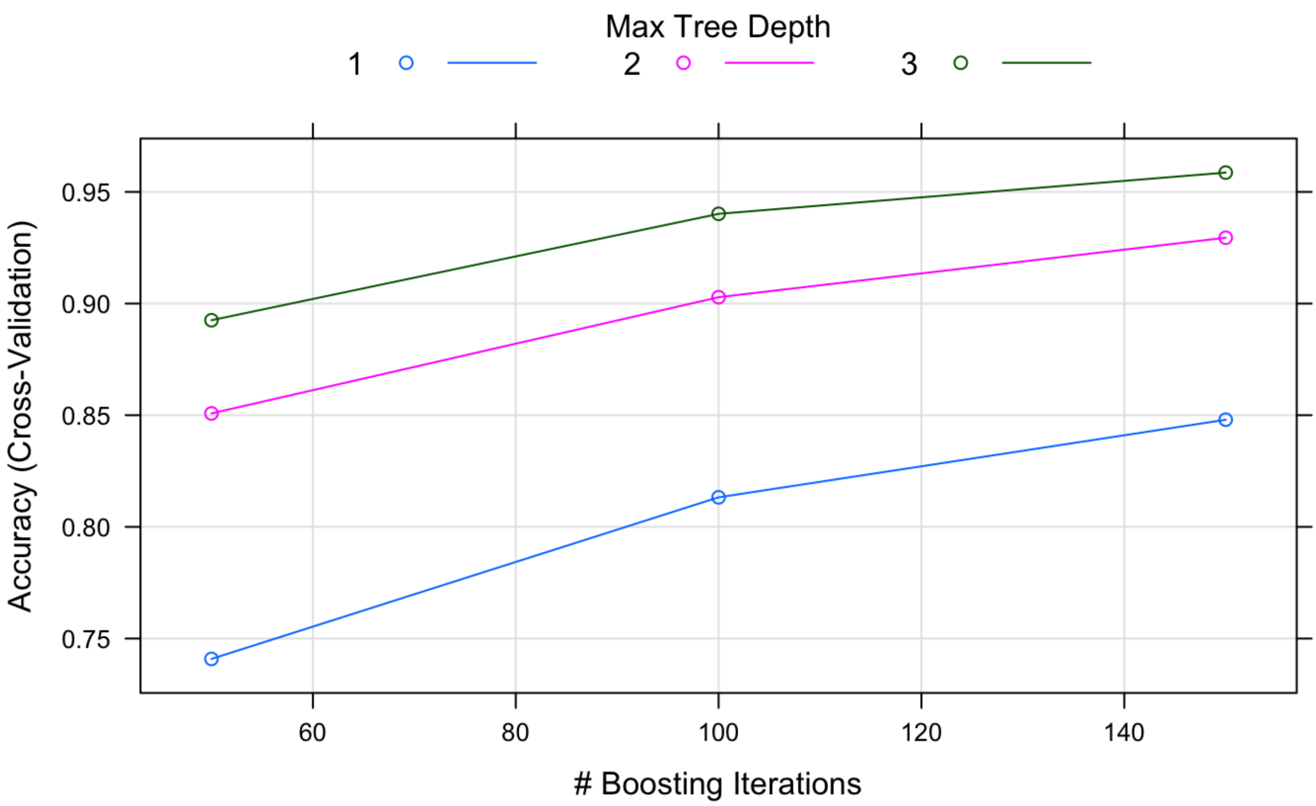
```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1102   28    0    2    1
         B   11  712   20    1    6
         C    0   17  654   30    8
         D    3    1    9  601    5
         E    0    1    1    9  701

Overall Statistics

               Accuracy : 0.961
                 95% CI : (0.9545, 0.9668)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9506

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9875   0.9381   0.9561   0.9347   0.9723
Specificity            0.9890   0.9880   0.9830   0.9945   0.9966
Pos Pred Value         0.9726   0.9493   0.9224   0.9709   0.9846
Neg Pred Value         0.9950   0.9852   0.9907   0.9873   0.9938
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2809   0.1815   0.1667   0.1532   0.1787
Detection Prevalence   0.2888   0.1912   0.1807   0.1578   0.1815
Balanced Accuracy      0.9882   0.9630   0.9696   0.9646   0.9844
```

Hide

```
#plot matrix results
plot(confmatBag$table, col=confmatBag$byclass, main = paste("Bagging_Accuracy = ", ro
und(confmatBag$overall['Accuracy'],4)))
```



4.2 Random Forest Method



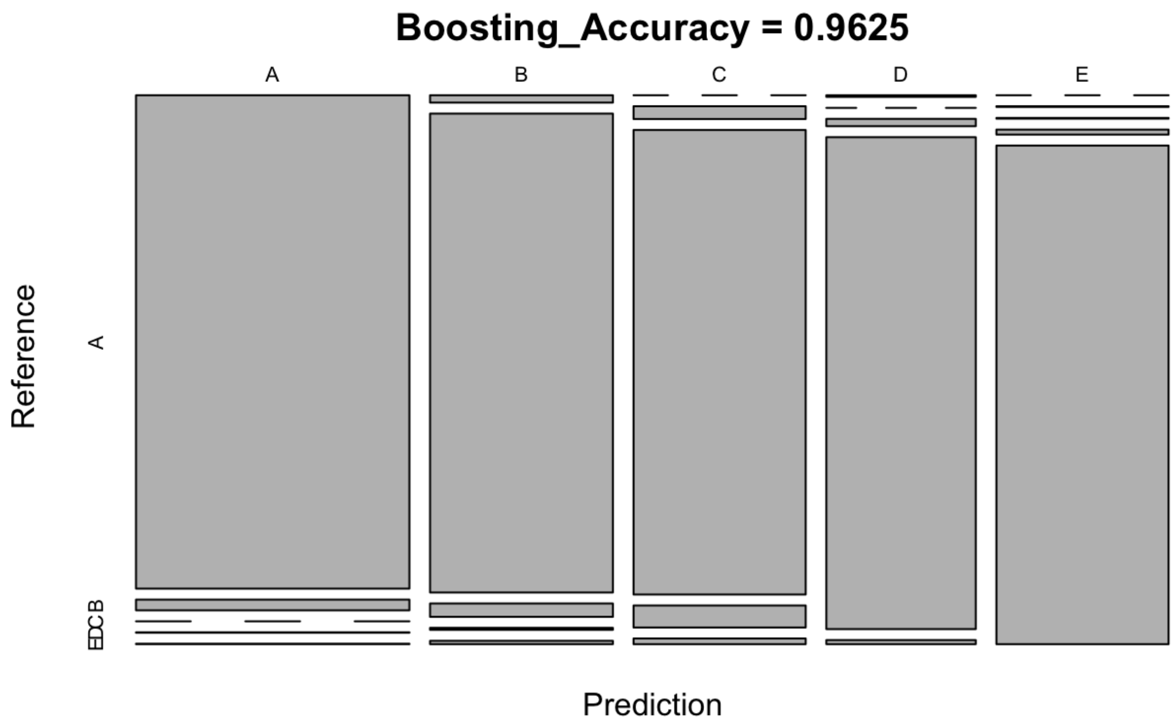
Hide

```
plot(modfitBoost) #plot the modelling fit

#prediction on testset
predictBoost <- predict(modfitBoost, testset)
confmatBoost <- confusionMatrix(predictBoost, testset$classe)
confmatBoost
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1103   24    0    1    1
         B   11  716   20    3    5
         C    0   18  654   31    8
         D    2    0    9  601    5
         E    0    1    1    7  702

Overall Statistics

               Accuracy : 0.9625
                 95% CI : (0.9561, 0.9683)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9526

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9884   0.9433   0.9561   0.9347   0.9736
Specificity            0.9907   0.9877   0.9824   0.9951   0.9972
Pos Pred Value         0.9770   0.9483   0.9198   0.9741   0.9873
Neg Pred Value         0.9953   0.9864   0.9907   0.9873   0.9941
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2812   0.1825   0.1667   0.1532   0.1789
Detection Prevalence   0.2878   0.1925   0.1812   0.1573   0.1812
Balanced Accuracy      0.9895   0.9655   0.9693   0.9649   0.9854
```

Hide

```
#plot matrix results
plot(confmatBoost$table, col=confmatBoost$byclass, main=paste("Boosting_Accuracy =",
 round(confmatBoost$overall['Accuracy'], 4)))
```



4.3 Bagging/Broost Aggreating Method

Hide

```
predictTest <- predict(ModfitRandforest, newdata=testing)
predictTest
```

```
 [1] B A B A A E D B A A B C B A E E A B B B
Levels: A B C D E
```

4.4 Boosting Method

Hide

```
#modelling fit
modfitBoost <- train(classe ~., method="gbm", data = trainset, verbose=FALSE, trContr
ol=trainControl(method="cv", number=10))
modfitBoost
plot(modfitBoost) #plot the modelling fit

#prediction on testset
predictBoost <- predict(modfitBoost, testset)
confmatBoost <- confusionMatrix(predictBoost, testset$classe)
confmatBoost

#plot matrix results
plot(confmatBoost$table, col=confmatBoost$byclass, main=paste("Boosting_Accuracy =",
 round(confmatBoost$overall['Accuracy'], 4)))
```

4.5 Best Model to Select

The accuracy of the 4 regression methods above are: - Decision Tree: 0.6933 - Random Forest: 0.9949 - Bagging: 0.9610 - Boosting: 0.9625

In this case, the Random Forest model has the highest accuracy compared to other models, so I applied it to predict the 20 testing dataset as shown below:

Hide

```
predictTest <- predict(ModfitRandforest, newdata=testing)
predictTest
```

5.Discussion in Financial Application

5.1

Decision Trees for real option analysis; Pricing of interest rate instruments with binominal trees