Practical Machine Learning: Course Project

Prediction of Exercise Types in six Participants

Mohsen Soltanifar

July 30, 2020

Introduction

In this project we explore data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to investigate how well people do excercise activities. The appendix includes the R code for the presented analysis.

Data Preparation

The data preparation is performed in three steps: (1) Loading the training and the test data; (2) Cleaning the loaded datasets from three types of variables: Near zero variance variables, those with minimum 90% missing values, and those intutively useless characters; (3) Partitioning the train data to 60% train data and 40% cross validation data.

Data Analysis

We consider three machine learning algorithms: the Decision Tree Model, the Random Forest Model and the knn. We fit each of them on the 60% training data above and predict over the leftover 40% cross validation data to see its acuracy. The acuracy statistics were 0.459, 0.990 and 0.953 respectively.

The Decision Tree Model

Confusion Matrix and Statistics

```
      Reference
```

Prediction A B C D E A 2021 51 121 37 2 B 633 493 174 217 1 C 653 48 563 104 0 D 570 246 195 275 0 E 322 321 197 192 410

Overall Statistics

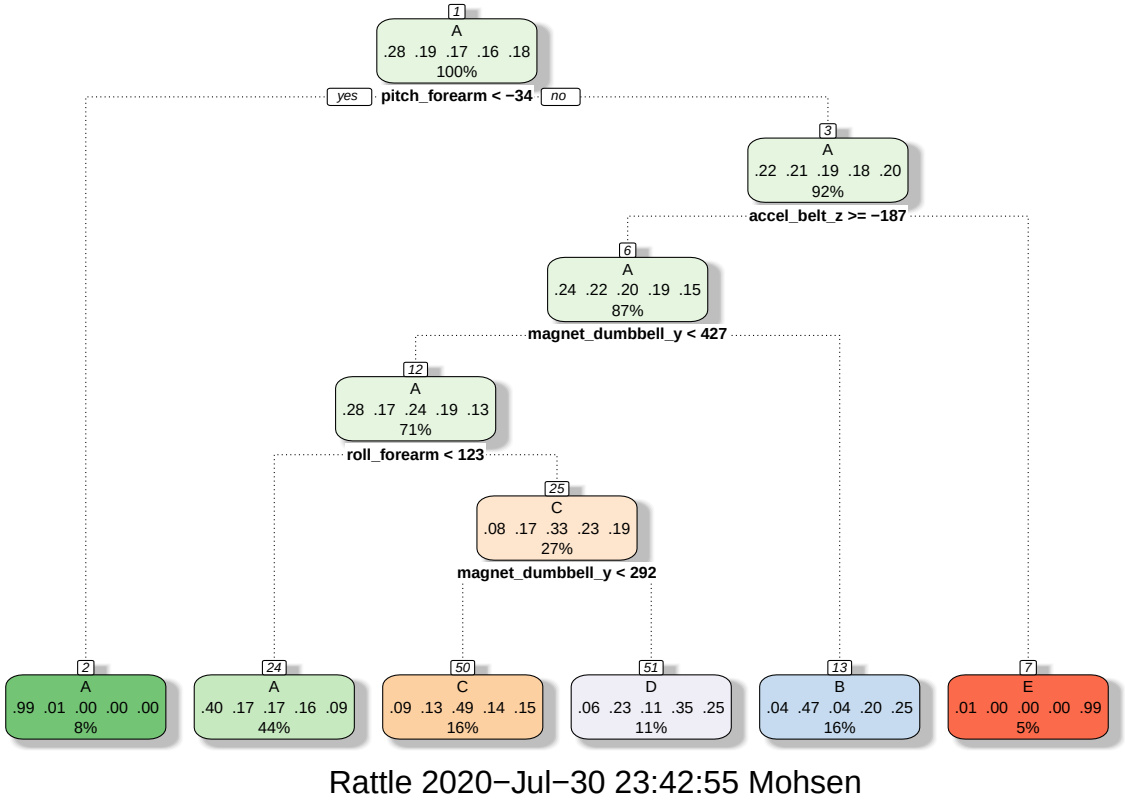
```
          Accuracy : 0.4795
            95% CI : (0.4684, 0.4906)
No Information Rate : 0.5352
P-Value [Acc > NIR] : 1

             Kappa : 0.3191
```

Mcnemar's Test P-Value : <2e-16

Statistics by Class:

```
                    Class: A Class: B Class: C Class: D Class: E
```

Sensitivity 0.4813 0.42537 0.45040 0.33333 0.99274 Specificity 0.9421 0.84672 0.87796 0.85600 0.86116 Pos Pred Value 0.9055 0.32477 0.41155 0.21384 0.28433 Neg Pred Value 0.6120 0.89475 0.89395 0.91616 0.99953 Prevalence 0.5352 0.14772 0.15932 0.10515 0.05264 Detection Rate 0.2576 0.06283 0.07176 0.03505 0.05226 Detection Prevalence 0.2845 0.19347 0.17436 0.16391 0.18379 Balanced Accuracy 0.7117 0.63604 0.66418 0.59467 0.92695



The Random Forest Model

Confusion Matrix and Statistics

```
          Reference
```

Prediction A B C D E A 2229 3 0 0 0 B 17 1489 11 0 1 C 0 16 1343 9 0 D 0 0 22 1261 3 E 0 2 4 7 1429

Overall Statistics

```
               Accuracy : 0.9879
                 95% CI : (0.9852, 0.9902)
    No Information Rate : 0.2863
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9847
```

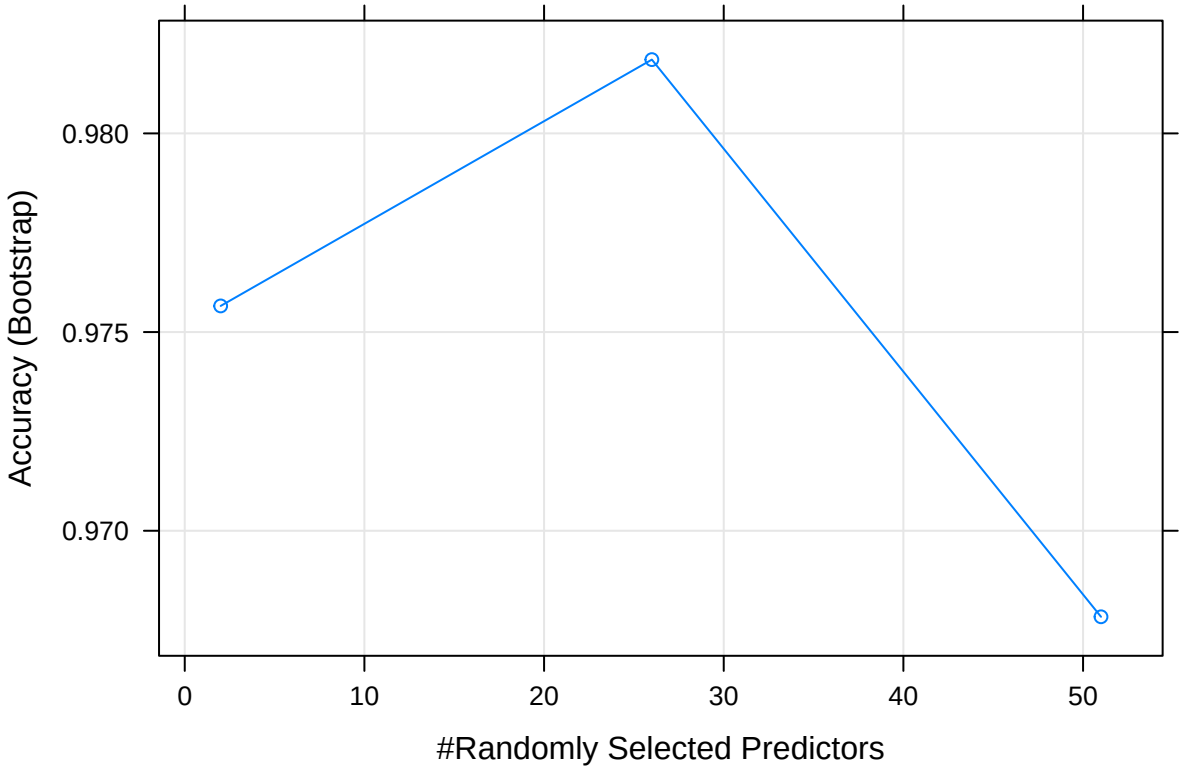
Mcnemar's Test P-Value : NA

Statistics by Class:

```
                     Class: A Class: B Class: C Class: D Class: E
```

Sensitivity 0.9924 0.9861 0.9732 0.9875 0.9972 Specificity 0.9995 0.9954 0.9961 0.9962 0.9980 Pos Pred Value 0.9987 0.9809 0.9817 0.9806 0.9910 Neg Pred Value 0.9970 0.9967 0.9943 0.9976 0.9994 Preva-

lence 0.2863 0.1925 0.1759 0.1628 0.1826 Detection Rate 0.2841 0.1898 0.1712 0.1607 0.1821 Detection Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838 Balanced Accuracy 0.9959 0.9908 0.9847 0.9918 0.9976



The KNN Model

Confusion Matrix and Statistics

```
          Reference
```

Prediction A B C D E A 2193 12 16 9 2 B 52 1412 44 8 2 C 8 34 1291 25 10 D 5 5 57 1211 8 E 5 18 25 16 1378

Overall Statistics

```
               Accuracy : 0.954
                 95% CI : (0.9491, 0.9585)
    No Information Rate : 0.2884
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9418
```

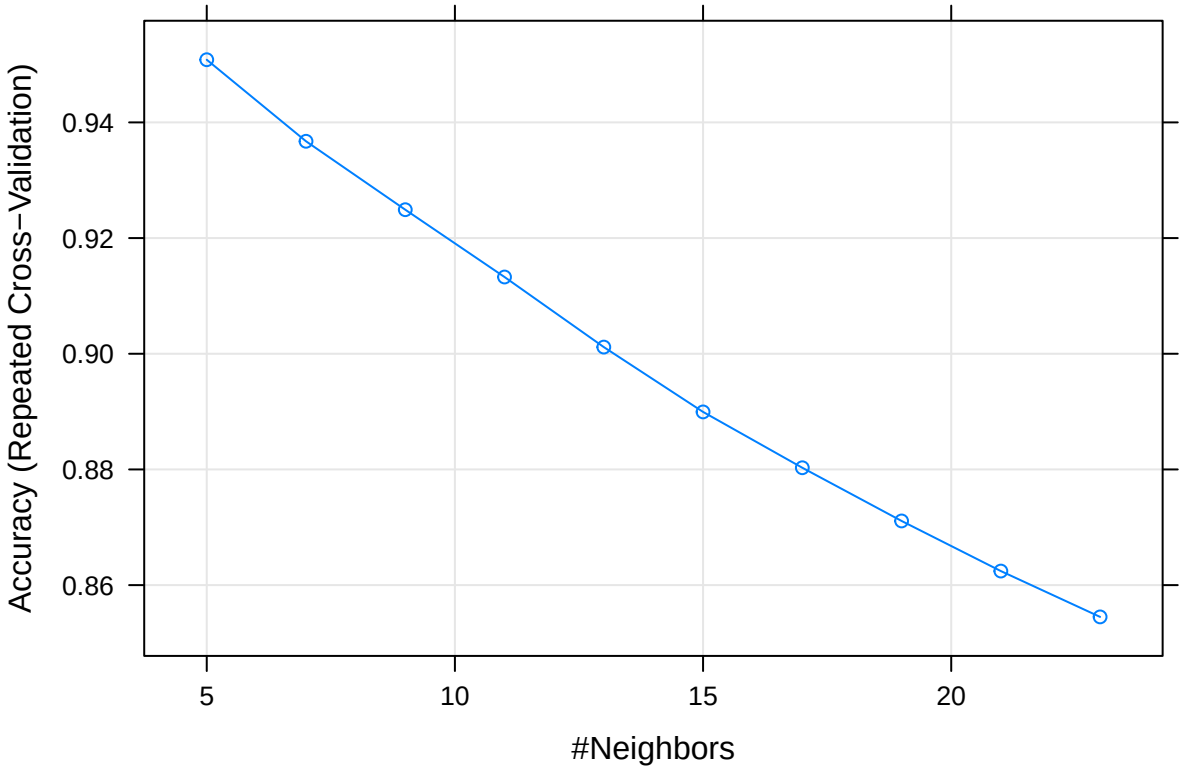
Mcnemar's Test P-Value : 2.136e-10

Statistics by Class:

```
                     Class: A Class: B Class: C Class: D Class: E
```

Sensitivity 0.9691 0.9534 0.9009 0.9543 0.9843 Specificity 0.9930 0.9833 0.9880 0.9886 0.9901 Pos Pred Value 0.9825 0.9302 0.9437 0.9417 0.9556 Neg Pred Value 0.9875 0.9891 0.9781 0.9912 0.9966 Prevalence 0.2884 0.1888 0.1826 0.1617 0.1784 Detection Rate 0.2795 0.1800 0.1645 0.1543 0.1756 Detection

Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838 Balanced Accuracy 0.9810 0.9684 0.9445 0.9714 0.9872



Conclusion.

Comparing three methods of Decision Tree Model, Random Forest Model and the KNN method, the best method is the Random Forest Model with maximum Accuracy of 0.99.

Prediction.

We predict the first 20 cases from the test using the Random Forest Model.

Appendix

R Code for the Analysis

R Code for Processing Data

```
defaultW <- getOption("warn")
options(warn = -1)

library(caret)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(RGtk2)
```

```r
library(rattle)
library(randomForest)

setwd("C:/Users/Mohsen/Desktop/Data.Science.Certificate.Coursera2020/Course 8. Practical Machine Learni

#1. Data Loading

trainingdata <- read.csv('pml-training.csv', na.strings = c("NA", "#DIV/0!", ""))
testdata <- read.csv('pml-testing.csv', na.strings = c("NA", "#DIV/0!", ""))

#2. Data Cleaning
# Remove variables with Nearzero variances

nearzerovarindex <- nearZeroVar(trainingdata)
trainingdata<-trainingdata[,-nearzerovarindex]
testdata<-testdata[,-nearzerovarindex]

#Remove variables more than 90% of the observation to be NA.

clnColumnIndex <- colSums(is.na(trainingdata))/nrow(trainingdata) < 0.90
cleantrainingdata <- trainingdata[,clnColumnIndex]
cleantestdata <- testdata[,clnColumnIndex]

#Removing variables which are non-numeric. Intuitively, these are useles variables.

cleantrainingdata<-cleantrainingdata[,-c(1:7)]
cleantestdata<-cleantestdata[,-c(1:7)]


#3. Data Partitioning
#Partition training data to 60% training and 40% corss vadiatin test data.

Index <- createDataPartition(cleantrainingdata$classe, p=0.60)[[1]]
trainingtrainingdata <- cleantrainingdata[Index,]
trainingcrossvaldata <- cleantrainingdata[-Index,]

options(warn = defaultW)
```

R Code for 3 Machine Learning Algorithms

```r
defaultW <- getOption("warn")
options(warn = -1)

#1. Decision Tree Model
DTmodel <- train(classe ~ ., data = trainingtrainingdata, method="rpart")
DTPrediction <- predict(DTmodel, trainingcrossvaldata)
CM1=confusionMatrix(as.factor(trainingcrossvaldata$classe), DTPrediction)
CM1$overall
#0.4598522
#fancyRpartPlot(DTmodel$finalModel)

#2. Random Forest Model
set.seed(12345)
```

```
RFmodel<- train(classe ~ ., data = trainingtrainingdata, method = "rf", ntree = 20)
RFPrediction <- predict(RFmodel, trainingcrossvaldata)
CM2=confusionMatrix(as.factor(trainingcrossvaldata$classe), RFPrediction)
CM2$overall
#0.9886566
#plot(RFmodel)
#Maximum Acuracy by number of selected predictors: almost 27 predictor for AC=0.982

#3. KNN method
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
set.seed(12345)
KNNmodel=train(classe ~ ., data = trainingtrainingdata,
               method="knn", trControl=trctrl,
               preProcess = c("center", "scale"),
               tuneLength = 10)

KNNPrediction <- predict(KNNmodel, trainingcrossvaldata)
CM3=confusionMatrix(as.factor(trainingcrossvaldata$classe), KNNPrediction)
CM3$overall
#0.95398
#plot(KNNmodel)

options(warn = defaultW)
```

R Code for Prediction for th first 20 test cases

```
defaultW <- getOption("warn")
options(warn = -1)

Testprediction <- predict(RFmodel, cleantestdata )
Testprediction

options(warn = defaultW)
```