# Practical ML

Arian A

2022-09-04

## Objective

In this project, we will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the manner in which they did the exercise. This is the "classe" variable in the training set. We train different machine learning models and evaluate their performance using a validation set randomly selected from the training csv data. Then we try the best model on the test data to evaluate the out of sample error.

## Background

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 type of 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. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

## Importing packages

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
## randomForest 4.7-1.1
```

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

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
## corrplot 0.92 loaded
```

### Download the Data

```
trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

```
trainFile <- "./data/pml-training.csv"
testFile  <- "./data/pml-testing.csv"
if (!file.exists("./data")) {
  dir.create("./data")
}
if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="curl")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="curl")
}
```

### Read the Data

After downloading the data from the data source, we can read the two csv files into two data frames.

```
traincsv <- read.csv("./data/pml-training.csv")
testcsv <- read.csv("./data/pml-testing.csv")
dim(traincsv)
```

```
## [1] 19622   160
```

```
dim(testcsv)
```

```
## [1]  20 160
```

The training data set contains 19622 observations and 160 variables, while the testing data set contains 20 observations and 160 variables. The "classe" variable in the training set is the outcome to predict.

### Clean the data

First, we remove the unnecessary data as well as the data with near zero vairance.

```
traincsv <- traincsv[,colMeans(is.na(traincsv)) < .9] #removing mostly na columns
traincsv <- traincsv[,-c(1:7)] #removing metadata which is irrelevant to the outcome
nvz <- nearZeroVar(traincsv)
traincsv <- traincsv[,-nvz]
dim(traincsv)
```

```
## [1] 19622    53
```

### Slice the data

Then, we can split the cleaned training set into a pure training data set (70%) and a validation data set (30%). We will use the validation data set to conduct cross validation in future steps.

```
set.seed(22519) # For reproducibile purpose
inTrain <- createDataPartition(y=traincsv$classe, p=0.7, list=F)
train <- traincsv[inTrain,]
valid <- traincsv[-inTrain,]
```

## Data Modeling

We fit a predictive model for activity recognition using **Random Forest and Support Vector Machine** algorithms. We will use **5-fold cross validation** when applying the algorithms.

### Random Forrest

```r
control <- trainControl(method="cv", number=5, verboseIter=F)
mod_rf <- train(classe~., data=train, method="rf", trControl = control, tuneLength = 5)
pred_rf <- predict(mod_rf, valid)
cmrf <- confusionMatrix(pred_rf, factor(valid$classe))
cmrf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1671    3    0    0    0
##          B    1 1133    4    0    0
##          C    2    3 1022    8    4
##          D    0    0    0  956    0
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9958
##                  95% CI : (0.9937, 0.9972)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9946
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9947   0.9961   0.9917   0.9963
## Specificity            0.9993   0.9989   0.9965   1.0000   1.0000
## Pos Pred Value         0.9982   0.9956   0.9836   1.0000   1.0000
## Neg Pred Value         0.9993   0.9987   0.9992   0.9984   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2839   0.1925   0.1737   0.1624   0.1832
## Detection Prevalence   0.2845   0.1934   0.1766   0.1624   0.1832
## Balanced Accuracy      0.9987   0.9968   0.9963   0.9959   0.9982
```

So, the estimated accuracy of Random Forrest model is 99%.

### Support Vector Machine

```r
mod_svm <- train(classe~., data=train, method="svmLinear", trControl = control, tuneLength = 5, verbose

pred_svm <- predict(mod_svm, valid)
cmsvm <- confusionMatrix(pred_svm, factor(valid$classe))
cmsvm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1553  153   96   58   62
##          B   27  819   83   32  140
##          C   42   54  806  107   69
##          D   48   26   25  719   54
##          E    4   87   16   48  757
##
## Overall Statistics
##
##                Accuracy : 0.7908
##                  95% CI : (0.7802, 0.8012)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7339
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9277   0.7191   0.7856   0.7459   0.6996
## Specificity            0.9124   0.9406   0.9440   0.9689   0.9677
## Pos Pred Value         0.8080   0.7439   0.7477   0.8245   0.8300
## Neg Pred Value         0.9695   0.9331   0.9542   0.9511   0.9346
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2639   0.1392   0.1370   0.1222   0.1286
## Detection Prevalence   0.3266   0.1871   0.1832   0.1482   0.1550
## Balanced Accuracy      0.9200   0.8298   0.8648   0.8574   0.8337
```

So, the estimated accuracy of SVM model is 79%.

## Predicting for Test Data Set

Now, we apply the Random Forrest model to the original testing data set downloaded from the data source.

```
pred <- predict(mod_rf, testcsv)
print(pred)
```

```
##  [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
```

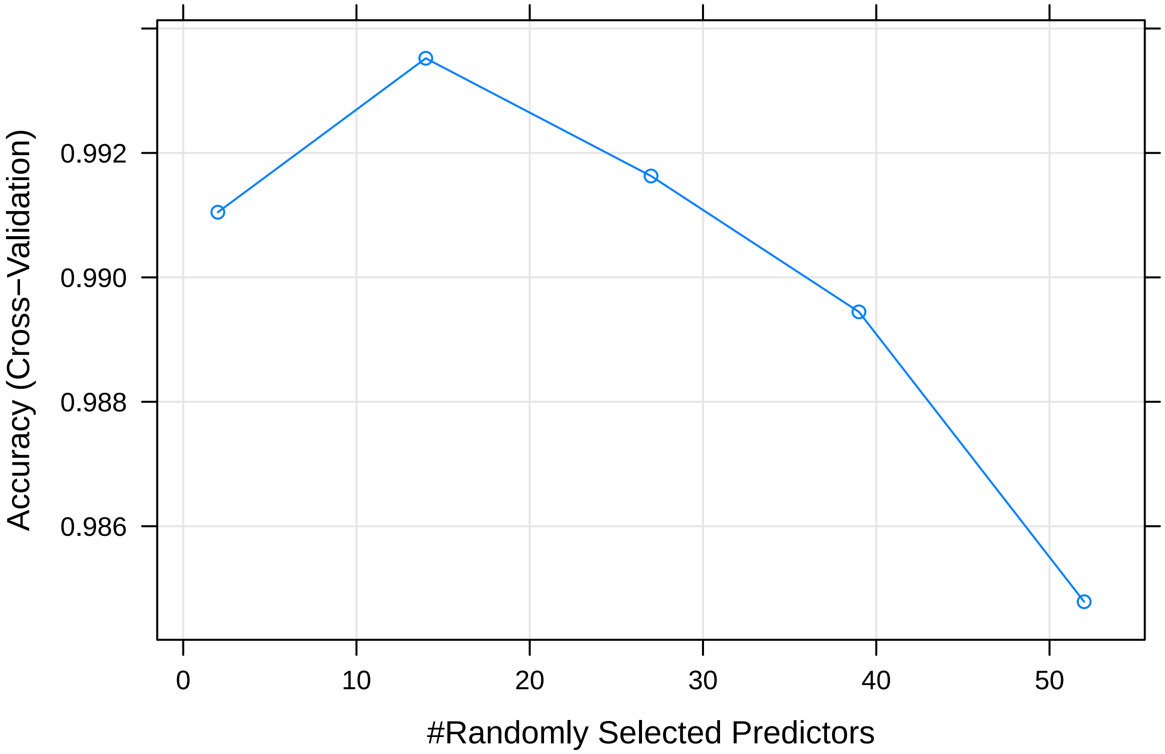
## Appendix: Figures

1. Correlation Matrix Visualization

```
#corrPlot <- cor(trainData[, -length(names(trainData))])
#corrplot(corrPlot, method="color")
```

2. Random Forrest Plot

```
plot(mod_rf)
```



Accuracy (Cross−Validation)
0.992
0.990
0.988
0.986
0
10
20
30
40
50
#Randomly Selected Predictors