# Human Activity Recognition Data Analysis

***Durgesh Verma***

***Dec 19, 2015***

# Synopsis

The report study the personal excercise activity data captured by devices like Fitbit, Nike Fuelband. The devices quantify self movement performed by group of people on themselves regularly. The report aims to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants done in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har). The report will perform multi step process to come up with a statistical model that can predict the manner in which participants did the exercise. The report provides the details on how the model was built, how cross validations were performed, the expected out of sample error, and the reason of selecting the final model. At the end, report will use the prediction model to predict 20 different test cases to validate the accuracy of the model.

# Initial Setup

## Set current working directory

```
cur_dir <- "./"
setwd(cur_dir)
```

## Load required libraries

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.1.2
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.1.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.1.2
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.1.2
```

# Data Processing

- Download the data file and unzip the CSV file
- Read the CSV file into memory
- Clean up data - Removed all secondary columns calculated on raw data columns. Identified NA columns in training and test set, removed union of both sets from training and test set.
- Define predictors - As required by assignment, filtered the columns having arm,forearm,dumbell and belt related data.
- Fit and validate multiple models - Step 1 is preprocessing the clean data. This is done using PCA and center/scale method. GLM cannot be used as it only works upto two levels (classe), rpart was experimented but the accuracy was lower than top two, hence not shown in report. Step 2 is fitting model on training data set using random forest.
- Model selection - Compare the accuracy of models and selecting the one that has better accuracy. Verify the model on test data set to check which model better predict the test results. Cross-validation and out of sample error is discussed in the Result section.

## Download data file and load raw data

```
removeColsStartWith = function(cols, df) {
    for(col in cols) {
        df = df[,-grep(paste("^",col,sep=""), names(df))]
    }
    df
}
keepColsContains = function(cols, df) {
    col_indexes = -1
    for(col in cols) {
        col_indexes = c(col_indexes,grep(col, names(df)))
    }
    df[,col_indexes[! col_indexes %in% c(-1)]]
}

filename = 'pml-training.csv'
if (!file.exists(filename)) {
    message("downloading file...")
    download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv', f
ilename, method='curl')
}
rawdata = read.table(filename, sep=",", na.strings=c("NA","?"), header=TRUE)
```

```
filename = 'pml-testing.csv'
if (!file.exists(filename)) {
    message("downloading file...")
    download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv', fi
lename, method='curl')
}
rawdata2 = read.table(filename, sep=",", na.strings=c("NA","?"), header=TRUE)
```

Training Data file has tabular data that contains 19622 rows and 160 cols.

# Clean data and define predictors

```
set.seed(12345)
seeds <- vector(mode = "list", length = 51)
for(i in 1:50) seeds[[i]] <- sample.int(1000, 22)
## For the last model:
seeds[[51]] <- sample.int(1000, 1)

stats_cols = c("min_","max_","avg_","total_","var_","stddev_","skewness_","kurtosis_")
cols_to_keep_in_training = c("classe","belt","arm","dumbell")
cols_to_keep_in_testing = c("problem_id","belt","arm","dumbell")

##remove stats cols
rawdata = removeColsStartWith(stats_cols,rawdata)
rawdata2 = removeColsStartWith(stats_cols,rawdata2)

##keeps the possible predictors in the set
rawdata = keepColsContains(cols_to_keep_in_training,rawdata)
rawdata2 = keepColsContains(cols_to_keep_in_testing,rawdata2)

##remove NA - from training set perspective
newdata=rawdata[,colSums(is.na(rawdata)) == 0]
rm(rawdata)

newdata2 = rawdata2[,intersect(colnames(rawdata2),colnames(newdata))]
newdata2$problem_id = rawdata2$problem_id
rm(rawdata2)

##remove NA - from test set perspective
newdata2=newdata2[,colSums(is.na(newdata2)) == 0]
newdata = newdata[,c("classe",intersect(colnames(newdata),colnames(newdata2)))]
```

Training data after clean up step 19622 rows and 37 cols.

# Model 1 : Using Random Forest and pre process with method=PCA

```
tc = trainControl("repeatedcv",number=10,repeats=5,p=0.75,verboseIter=FALSE,classProbs=TRUE
,savePred=T,seeds=seeds)

preProc.pca = preProcess(newdata,method="pca")
train.pca = predict(preProc.pca, newdata)
test.pca = predict(preProc.pca, newdata2)

train.pca = select(train.pca,classe,starts_with("PC"))
test.pca = select(test.pca,problem_id,starts_with("PC"))

rf.pca = train(classe~.,data=train.pca,method="rf",trControl=tc)
```

```
## Loading required package: randomForest
```

```
## Warning: package 'randomForest' was built under R version 3.1.3
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
rf.pca
```

```
## Random Forest
##
## 19622 samples
##    19 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 5 times)
## Summary of sample sizes: 17659, 17660, 17662, 17659, 17659, 17659, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9720520  0.9646441  0.003578536  0.004528096
##   10    0.9688105  0.9605429  0.004144763  0.005244605
##   19    0.9609926  0.9506507  0.004670703  0.005910071
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
rf.pca$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: 2
##
##         OOB estimate of  error rate: 2.58%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 5527   33   14    4    2 0.009498208
## B   47 3680   49    3   18 0.030813800
## C    7  107 3262   36   10 0.046756283
## D    3    6  118 3081    8 0.041977612
## E    3   17    9   13 3565 0.011644026
```

# Model 2 : Using Random Forest and pre process with method=c(“center”,“scale”)

```
preProc.cs = preProcess(newdata,method=c("center","scale"))
train.cs = predict(preProc.cs, newdata)
test.cs = predict(preProc.cs, newdata2)

rf.cs = train(classe~.,data=train.cs,method="rf",trControl=tc)
rf.cs
```

```
## Random Forest
##
## 19622 samples
##    36 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 5 times)
## Summary of sample sizes: 17659, 17660, 17659, 17660, 17660, 17662, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9950771  0.9937728  0.001586569  0.002007016
##   19    0.9930181  0.9911684  0.002239078  0.002832505
##   36    0.9897870  0.9870813  0.002385456  0.003017875
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
rf.cs$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: 2
##
##         OOB estimate of  error rate: 0.4%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5578    1    0    1    0 0.0003584229
## B    7 3785    5    0    0 0.0031603898
## C    1   21 3390   10    0 0.0093512566
## D    0    2   25 3187    2 0.0090174129
## E    0    0    0    4 3603 0.0011089548
```

## `Test Data Prediction` - Model 1 (rf.pca)

```
predict(rf.pca$finalModel,test.pca)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  A  A  A  E  D  B  A  A  C  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

## `Test Data Prediction` - Model 1 (rf.cs)

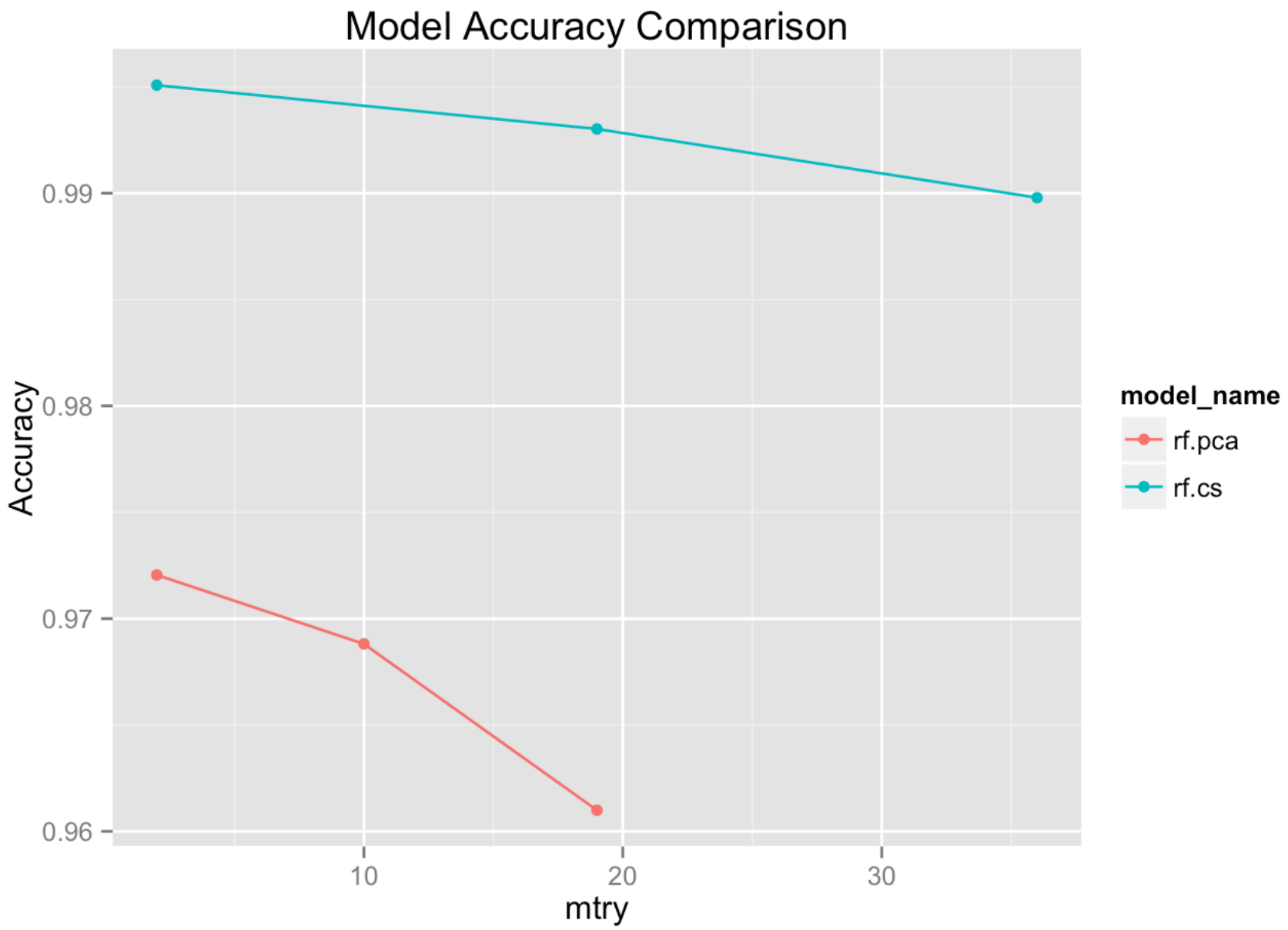
```
predict(rf.cs$finalModel,test.cs)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

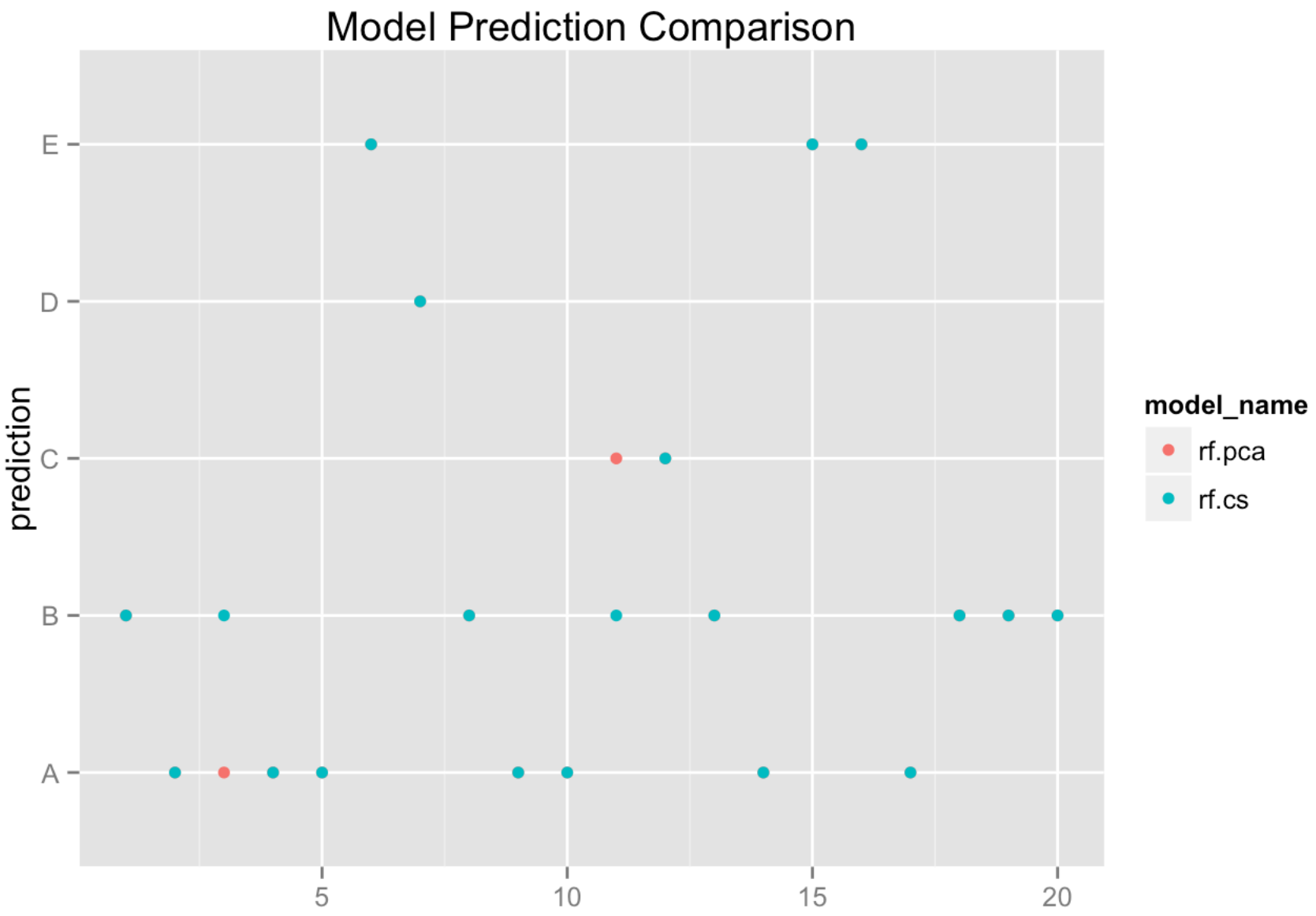
## `Results - Part 1` - Which model is better?

- Random forest with preprocess center/scale is better over PCA.
- Similarity - Both models used 10 folds and 5 repeatations.
- Comparison - PCA method used 19 predictors vs 36 in center/scale. Accuracy with PCA is 0.9721 vs 0.9951 with center/scale keeping mtry=2 which is final value used in both model.

```
model_sel_grid = cbind(select(rf.pca$results, mtry,Accuracy),model_name=rep("rf.pca",3))
model_sel_grid = rbind(model_sel_grid,cbind(select(rf.cs$results, mtry,Accuracy),model_name
=rep("rf.cs",3)))
#model accuracy comparison
qplot(mtry,Accuracy,data=model_sel_grid,colour=model_name) + geom_line() + ggtitle("Model A
ccuracy Comparison")
```




```
#prediction comparison
pt.pca = select(test.pca,problem_id)
pt.pca = cbind(pt.pca,data.frame(prediction=predict(rf.pca$finalModel,test.pca),model_name=
rep("rf.pca",nrow(pt.pca))))
pt.cs = select(test.cs,problem_id)
pt.cs = cbind(pt.cs,data.frame(prediction=predict(rf.cs$finalModel,test.cs),model_name=rep(
"rf.cs",nrow(pt.cs))))
predict_table = rbind(pt.pca,pt.cs)
rm(pt.pca,pt.cs)
qplot(problem_id,prediction,data=predict_table,colour=model_name) + ggtitle("Model Predicti
on Comparison")
```




## `Results - Part 2` - Out of sample error

- Looking at the finalModel parameter of each model fit, the Out of bag (OOB) estimate of error rate is 0.04% in model 2 (rf.cs) compared to 2.58% in model 1 (rf.pca).

## `Results - Part 3` - Estimate the error appropriately with cross-validation

- Cross validation is done using train method. trainControl is configured to create 10 folds and each repeat 5 times.
- For each fold, 75% was allocated to training set, and 25% was allocated to validation set.
- After all iterations, finalModel is the outcome of each model fit.
- Model 2 (rf.cs) is 100% accurate as checked from project submission. Model 1 (rf.pca) error rate is 0.1

## Clean up memory

```
#rm(list=ls())
```