Predicting Quality of Excercises

Mohit Kumar

21 January 2017

Pre-processing

```
#required libraries
library(caret)
library(corrplot)
library(ggplot2)
library(GGally)
library(Rtsne)
```

```
#saving the urls of train and test dataset

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

#saving the name of files

trainFile <- "./pml-training.csv"
testFile  <- "./pml-testing.csv"

#if files doesn't exist download them

if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="auto")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="auto")
}

set.seed(0)
```

Synopsis

For this project, we are given data from accelerometers on the belt, forearm, arm, and dumbell of 6 research study participants. Our training data consists of accelerometer data and a label identifying the quality of the activity the participant was doing. Our testing data consists of accelerometer data without the identifying label. Our goal is to predict the labels for the test set observations.

Part 1: Data

Reading the training dataset into `train` object.

```
#reading the train dataset

train<-read.csv(trainFile,na.strings = c(NA,"#DIV/0!"),header = TRUE)
```

```
head(train,n=1)
```

```
##   X user_name raw_timestamp_part_1 raw_timestamp_part_2   cvtd_timestamp
## 1 1  carlitos           1323084231               788290 05/12/2011 11:23
##   new_window num_window roll_belt pitch_belt yaw_belt total_accel_belt
## 1         no         11      1.41       8.07    -94.4                3
##   kurtosis_roll_belt kurtosis_picth_belt kurtosis_yaw_belt
## 1                 NA                  NA                NA
##   skewness_roll_belt skewness_roll_belt.1 skewness_yaw_belt max_roll_belt
## 1                 NA                   NA                NA            NA
##   max_picth_belt max_yaw_belt min_roll_belt min_pitch_belt min_yaw_belt
## 1             NA           NA            NA             NA           NA
##   amplitude_roll_belt amplitude_pitch_belt amplitude_yaw_belt
## 1                  NA                   NA                 NA
##   var_total_accel_belt avg_roll_belt stddev_roll_belt var_roll_belt
## 1                   NA            NA               NA           NA
##   avg_pitch_belt stddev_pitch_belt var_pitch_belt avg_yaw_belt
## 1             NA                NA             NA           NA
##   stddev_yaw_belt var_yaw_belt gyros_belt_x gyros_belt_y gyros_belt_z
## 1              NA           NA            0            0        -0.02
##   accel_belt_x accel_belt_y accel_belt_z magnet_belt_x magnet_belt_y
## 1          -21            4           22           -3          599
##   magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm var_accel_arm
## 1          -313     -128      22.5    -161              34            NA
##   avg_roll_arm stddev_roll_arm var_roll_arm avg_pitch_arm stddev_pitch_arm
## 1           NA              NA           NA            NA               NA
##   var_pitch_arm avg_yaw_arm stddev_yaw_arm var_yaw_arm gyros_arm_x
## 1            NA          NA             NA          NA           0
##   gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z magnet_arm_x
## 1           0       -0.02        -288         109        -123         -368
##   magnet_arm_y magnet_arm_z kurtosis_roll_arm kurtosis_picth_arm
## 1          337          516                NA                 NA
##   kurtosis_yaw_arm skewness_roll_arm skewness_pitch_arm skewness_yaw_arm
## 1               NA                NA                 NA               NA
##   max_roll_arm max_picth_arm max_yaw_arm min_roll_arm min_pitch_arm
## 1           NA            NA          NA           NA            NA
##   min_yaw_arm amplitude_roll_arm amplitude_pitch_arm amplitude_yaw_arm
## 1          NA                 NA                  NA               NA
##   roll_dumbbell pitch_dumbbell yaw_dumbbell kurtosis_roll_dumbbell
## 1      13.05217        -70.494    -84.87394                    NA
##   kurtosis_picth_dumbbell kurtosis_yaw_dumbbell skewness_roll_dumbbell
## 1                      NA                   NA                     NA
##   skewness_pitch_dumbbell skewness_yaw_dumbbell max_roll_dumbbell
## 1                      NA                   NA               NA
##   max_picth_dumbbell max_yaw_dumbbell min_roll_dumbbell min_pitch_dumbbell
## 1                 NA               NA                NA                 NA
##   min_yaw_dumbbell amplitude_roll_dumbbell amplitude_pitch_dumbbell
## 1               NA                       NA                       NA
##   amplitude_yaw_dumbbell total_accel_dumbbell var_accel_dumbbell
## 1                     NA                   37                 NA
##   avg_roll_dumbbell stddev_roll_dumbbell var_roll_dumbbell
## 1                NA                   NA                NA
##   avg_pitch_dumbbell stddev_pitch_dumbbell var_pitch_dumbbell
```

```
## 1                 NA                    NA                 NA
##   avg_yaw_dumbbell stddev_yaw_dumbbell var_yaw_dumbbell gyros_dumbbell_x
## 1               NA                   NA               NA                0
##   gyros_dumbbell_y gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y
## 1            -0.02                0            -234               47
##   accel_dumbbell_z magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z
## 1             -271              -559               293               -65
##   roll_forearm pitch_forearm yaw_forearm kurtosis_roll_forearm
## 1         28.4         -63.9       -153                    NA
##   kurtosis_picth_forearm kurtosis_yaw_forearm skewness_roll_forearm
## 1                     NA                   NA                    NA
##   skewness_pitch_forearm skewness_yaw_forearm max_roll_forearm
## 1                     NA                   NA               NA
##   max_picth_forearm max_yaw_forearm min_roll_forearm min_pitch_forearm
## 1                NA              NA               NA                NA
##   min_yaw_forearm amplitude_roll_forearm amplitude_pitch_forearm
## 1              NA                    NA                     NA
##   amplitude_yaw_forearm total_accel_forearm var_accel_forearm
## 1                    NA                 36                NA
##   avg_roll_forearm stddev_roll_forearm var_roll_forearm avg_pitch_forearm
## 1               NA                  NA               NA                NA
##   stddev_pitch_forearm var_pitch_forearm avg_yaw_forearm
## 1                   NA                NA              NA
##   stddev_yaw_forearm var_yaw_forearm gyros_forearm_x gyros_forearm_y
## 1                 NA              NA            0.03                0
##   gyros_forearm_z accel_forearm_x accel_forearm_y accel_forearm_z
## 1           -0.02             192             203            -215
##   magnet_forearm_x magnet_forearm_y magnet_forearm_z classe
## 1              -17              654              476      A
```

We can see that the first column is not useful along with the next 5 columns that are only for informative purposes.

So, we will clean our data so that it consists of significant predictors only.For this we are going to remove all the columns with near zero variability and columns with more than 80% values NA.

```
#removing the first column
train<-train[,-1]

#checking for variables that have zero variability
nzv<-nearZeroVar(train)

#removing variables that will not contribute towards model
train<- train[,-nzv]


#checking for columns that comprise of more than 80% NA values
mostlyNA <- sapply(train, function(x) mean(is.na(x))) > 0.80

#removing columns containing more than 80% NA values
train <- train[, !mostlyNA]

#removing the first five insignificant columns
train<-train[,-(1:5)]
```

Part 2: Exploratory data analysis

- Fig 1

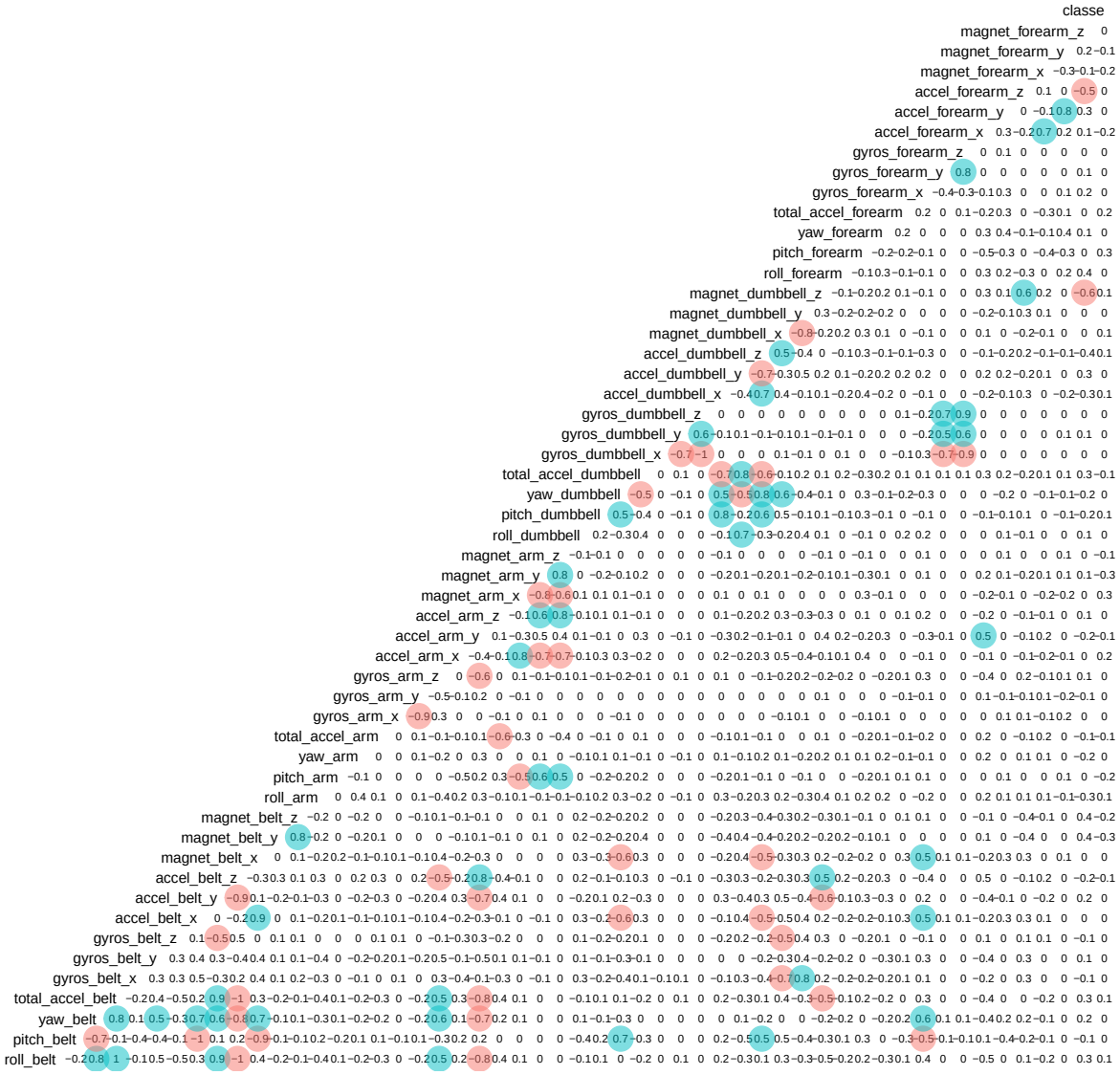
Plotting a correlation plot to check for redundant variables. Here I have colored the correlations greater than 0.5 as blue and less than -0.5 as red.

As `classe` is not numeric ,to use it in the `ggcorr` function we need to convert it to numeric data.

```
#temporarily saving the train data into 't' object
t<-train

#converting the `classe` column into 'numeric'
t$classe<-as.numeric(t$classe)

#plotting the correlation
ggcorr(t, geom = "blank", label = TRUE, hjust = 1,label_size = 2.2,layout.exp = 5,size=2.7,label_round =
    geom_point(size = 6, aes(color = coefficient > 0 , alpha = abs(coefficient) > 0.5)) +
    scale_alpha_manual(values = c("TRUE" = 0.5, "FALSE" = 0)) +
    guides(color = FALSE, alpha = FALSE)
```




```
#remove t object
rm(t)
```

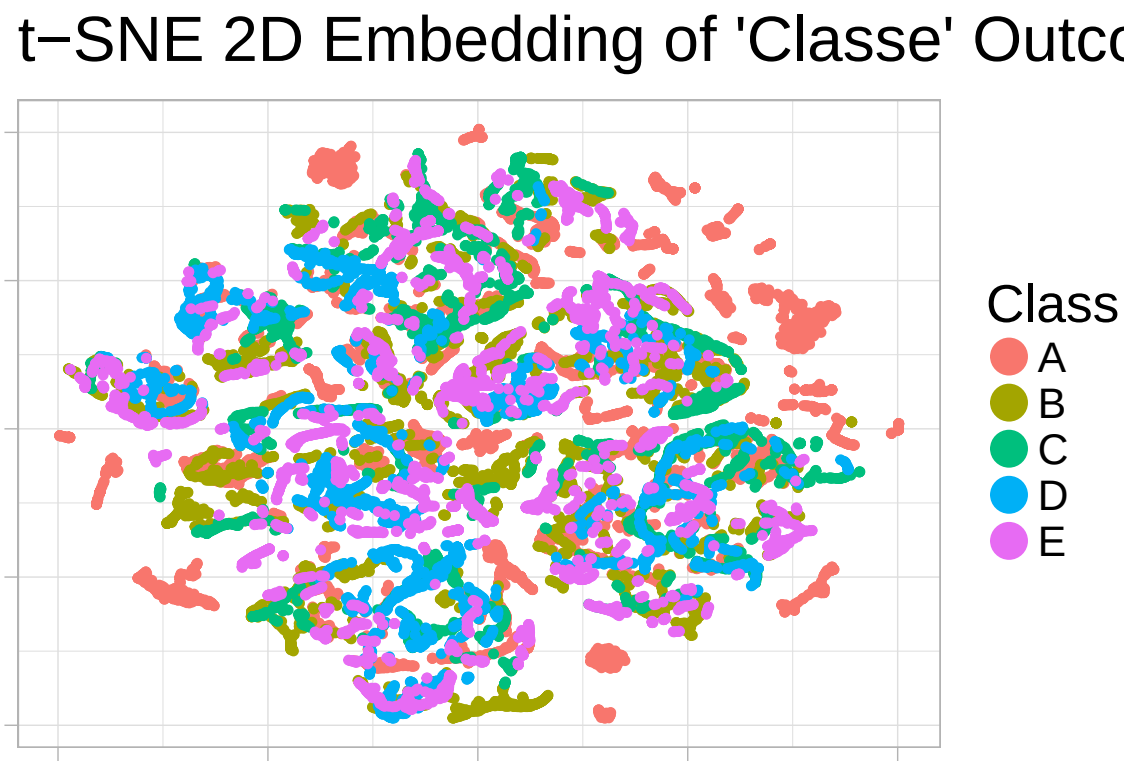
- Fig 2

t-distributed stochastic neighbor embedding (t-SNE) is a machine learning algorithm for dimensionality reduction developed by Geoffrey Hinton and Laurens van der Maaten.It is a nonlinear dimensionality reduction technique that is particularly well-suited for embedding high-dimensional data into a space of two or three dimensions, which can then be visualized in a scatter plot. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.

Here, I am providing the t-sne plot of the data.

```r
#creating a t-sne of train data
tsne = Rtsne(as.matrix(train[,-ncol(train)]), check_duplicates=FALSE, pca=TRUE,
             perplexity=30, theta=0.5, dims=2)
embedding = as.data.frame(tsne$Y)
embedding$Class = train$classe

#plotting the t-sne data
g = ggplot(embedding, aes(x=V1, y=V2, color=Class)) +
    geom_point(size=1.25) +
    guides(colour=guide_legend(override.aes=list(size=6))) +
    xlab("") + ylab("") +
    ggtitle("t-SNE 2D Embedding of 'Classe' Outcome") +
    theme_light(base_size=20) +
    theme(axis.text.x=element_blank(),
          axis.text.y=element_blank())
print(g)
```



We can see that there is no clear difference in the 5 classes of exercises. So, building a regression model or any manual modeling is not possible.

Next we will use machine learning algorithms to achieve the classification task.

Part 3: Modeling

I was considering to use random forest but it takes too much time to build the model using the following code.

```
modFit <- train(classe ~ ., method = "rf", data = train, importance = T, trControl = trainControl(metho
```

So, instead I will go for faster gradient boosting algorithm. It will be a bit less accurate but will be significantly faster.

We will use the `trControl` parameter to specify that we will use 10-fold cross validation to evaluate our model.

```
boostFit <- train(classe ~ ., method = "gbm", data = train, verbose = F, trControl = trainControl(metho
```

```
## Warning: package 'gbm' was built under R version 3.3.2
```

```
boostFit
```

```
## Stochastic Gradient Boosting
##
## 19622 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 17660, 17660, 17661, 17660, 17660, 17659, ...
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa
##   1                   50      0.7530830  0.6869218
##   1                  100      0.8220875  0.7748047
##   1                  150      0.8545510  0.8159158
##   2                   50      0.8552640  0.8166244
##   2                  100      0.9079607  0.8835398
##   2                  150      0.9331875  0.9154559
##   3                   50      0.8987873  0.8718923
##   3                  100      0.9434311  0.9284294
##   3                  150      0.9640203  0.9544821
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 150,
##  interaction.depth = 3, shrinkage = 0.1 and n.minobsinnode = 10.
```

`boostFit` model have an overall accuracy of 96% which is quite impressive .The random forest model is more accurate with overall accuracy 99% but it is computationally very expensive to fit.

Part 4: Prediction

First we will load the test data set and perform the tranformations that we have performed on the test dataset.

```r
#reading the test data
test<-read.csv(testFile,na.strings = c(NA,"#DIV/0!"),header = TRUE)

head(test,n=1)
```

```
##   X user_name raw_timestamp_part_1 raw_timestamp_part_2   cvtd_timestamp
## 1 1     pedro           1323095002               868349 05/12/2011 14:23
##   new_window num_window roll_belt pitch_belt yaw_belt total_accel_belt
## 1         no         74       123         27    -4.75               20
##   kurtosis_roll_belt kurtosis_picth_belt kurtosis_yaw_belt
## 1                 NA                  NA                NA
##   skewness_roll_belt skewness_roll_belt.1 skewness_yaw_belt max_roll_belt
## 1                 NA                   NA                NA            NA
##   max_picth_belt max_yaw_belt min_roll_belt min_pitch_belt min_yaw_belt
## 1             NA           NA            NA             NA           NA
##   amplitude_roll_belt amplitude_pitch_belt amplitude_yaw_belt
## 1                  NA                   NA                 NA
##   var_total_accel_belt avg_roll_belt stddev_roll_belt var_roll_belt
## 1                   NA           NA                NA            NA
##   avg_pitch_belt stddev_pitch_belt var_pitch_belt avg_yaw_belt
## 1             NA                NA             NA           NA
##   stddev_yaw_belt var_yaw_belt gyros_belt_x gyros_belt_y gyros_belt_z
## 1              NA           NA         -0.5        -0.02        -0.46
##   accel_belt_x accel_belt_y accel_belt_z magnet_belt_x magnet_belt_y
## 1          -38           69         -179           -13           581
##   magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm var_accel_arm
## 1          -382     40.7     -27.8     178              10            NA
##   avg_roll_arm stddev_roll_arm var_roll_arm avg_pitch_arm stddev_pitch_arm
## 1           NA              NA           NA            NA               NA
##   var_pitch_arm avg_yaw_arm stddev_yaw_arm var_yaw_arm gyros_arm_x
## 1            NA          NA             NA          NA       -1.65
##   gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z magnet_arm_x
## 1        0.48       -0.18          16          38          93         -326
##   magnet_arm_y magnet_arm_z kurtosis_roll_arm kurtosis_picth_arm
## 1          385          481                NA                 NA
##   kurtosis_yaw_arm skewness_roll_arm skewness_pitch_arm skewness_yaw_arm
## 1               NA                NA                 NA               NA
##   max_roll_arm max_picth_arm max_yaw_arm min_roll_arm min_pitch_arm
## 1           NA            NA          NA           NA            NA
##   min_yaw_arm amplitude_roll_arm amplitude_pitch_arm amplitude_yaw_arm
## 1          NA                 NA                  NA                NA
##   roll_dumbbell pitch_dumbbell yaw_dumbbell kurtosis_roll_dumbbell
## 1     -17.73748       24.96085      126.236                    NA
##   kurtosis_picth_dumbbell kurtosis_yaw_dumbbell skewness_roll_dumbbell
## 1                      NA                    NA                     NA
##   skewness_pitch_dumbbell skewness_yaw_dumbbell max_roll_dumbbell
## 1                      NA                    NA               NA
##   max_picth_dumbbell max_yaw_dumbbell min_roll_dumbbell min_pitch_dumbbell
## 1                 NA               NA                NA                 NA
##   min_yaw_dumbbell amplitude_roll_dumbbell amplitude_pitch_dumbbell
## 1               NA                      NA                       NA
##   amplitude_yaw_dumbbell total_accel_dumbbell var_accel_dumbbell
## 1                     NA                    9                 NA
```

```
##   avg_roll_dumbbell stddev_roll_dumbbell var_roll_dumbbell
## 1                NA                   NA                NA
##   avg_pitch_dumbbell stddev_pitch_dumbbell var_pitch_dumbbell
## 1                 NA                    NA                 NA
##   avg_yaw_dumbbell stddev_yaw_dumbbell var_yaw_dumbbell gyros_dumbbell_x
## 1               NA                  NA               NA            0.64
##   gyros_dumbbell_y gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y
## 1             0.06            -0.61               21              -15
##   accel_dumbbell_z magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z
## 1               81               523              -528               -56
##   roll_forearm pitch_forearm yaw_forearm kurtosis_roll_forearm
## 1          141          49.3         156                    NA
##   kurtosis_picth_forearm kurtosis_yaw_forearm skewness_roll_forearm
## 1                     NA                   NA                    NA
##   skewness_pitch_forearm skewness_yaw_forearm max_roll_forearm
## 1                     NA                   NA               NA
##   max_picth_forearm max_yaw_forearm min_roll_forearm min_pitch_forearm
## 1                NA              NA               NA                NA
##   min_yaw_forearm amplitude_roll_forearm amplitude_pitch_forearm
## 1              NA                    NA                     NA
##   amplitude_yaw_forearm total_accel_forearm var_accel_forearm
## 1                    NA                  33                NA
##   avg_roll_forearm stddev_roll_forearm var_roll_forearm avg_pitch_forearm
## 1               NA                  NA               NA                NA
##   stddev_pitch_forearm var_pitch_forearm avg_yaw_forearm
## 1                   NA                NA              NA
##   stddev_yaw_forearm var_yaw_forearm gyros_forearm_x gyros_forearm_y
## 1                 NA              NA            0.74           -3.34
##   gyros_forearm_z accel_forearm_x accel_forearm_y accel_forearm_z
## 1           -0.59            -110             267            -149
##   magnet_forearm_x magnet_forearm_y magnet_forearm_z problem_id
## 1             -714              419              617          1
```

We can see that `test` consisit of one extra last column that is not needed as it is informative only.

```
#applying the same transformation that was used to tranform train data
test<-test[,-1]      #removing the first column
test<- test[,-nzv]    #removing variables that are close to zero variability
test <- test[, !mostlyNA]       #removing variables that are mostly NA
test<-test[,-(1:5)]         #removing the fist five columns

test<-test[,-ncol(test)]   #removing the last column
```

Predictions are calculated on the test data and results are printed below.

```
#predicting the test data
preds<- predict(boostFit,newdata = test)

preds
```

```
##  [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
```

Part 5: Conclusion

We build a classifier to classify the 5 types of the acitivities. Overall accuracy of our model could be increased by using other more computational expensive models such as random forests.

(I am grateful to the providers of the dataset used in this assignment : http://groupware.les.inf.puc-rio.br/har .)