Title: Human Physical Activity Prediction Using Wearable Accelerometer Data

Executive Summary: The goal of this project is to built a model to predict the manner in which human physical activity or movement occurs using the data collected from wearable accelerometer devices such as Jawbone, and Fit bit. The training data for this project is available at [1] and the test data is available at [2]. The data was collected as part of the research cited in the publication [3]. First we inspected and cleansed the data starting with 160 columns, the data set was finally reduced to 60 columns. We then performed exploratory data analysis and ran correlation on the reduced data set. We also performed principal component analysis (PCA) by using singular values decomposition (SVD). These analyses helped understand independent variables in the data set that explains most of the variation in the outcome or dependent variable. Finally, we divided the data into training and cross validation (CV) sets. Then using Random Forest in caret package we built the predictive model and evaluated it against the cross validation data set to check its accuracy and out of sample error rate. Our final model exhibits an accuracy of 99.8% against the CV data. We applied our final model to the test data with 20 samples to predict the outcome with 100% correct predictions.

Data Collection: For our analysis we used the data file 'pml-testing.csv' which was downloaded on 09/01/2014 at 12:47 pm. The R programming language (v3.1.0) was used to perform the analysis [5]. To help further understand the meaning of data columns, also called data features, we reference detail information included in [4].

Data Cleaning and Reduction: A data frame was created by reading the file 'pml-testing.csv' using R. Exploration and preliminary analysis of the data was then performed, first by examining the size of the data frame, missing values, column names (features), data type etc.

```
# Read the data
setwd('C:\\Coursera_New_Home\\8 - Practical Machine Learning\\Projects')
pmlTrain <- read.csv("pml-training.csv")
pmlTest <- read.csv("pml-testing.csv")

# Number of rows (records) and columns in the data set
mRows <- nrow(pmlTrain); mCols <- ncol(pmlTrain)

# Report the count of NA's in the seb data set
naCount <- length(which(is.na(pmlTrain) == TRUE))

# Report all the column anmes and duplicate columns names in the data set
colNames <- names(pmlTrain)
nDup = length(colNames) - length(unique(colNames))

# Report data type of the features in the data set
dataType <- sapply(pmlTrain[1,],class)
lvl_classe <- levels(pmlTrain$classe)
```

Here are some of the observations made.

1. Number of features (columns): 160
2. Number of samples/observations (rows): 19622
3. Missing data count: 1287472

4. Count of duplicate column names: 0

5. Dependent variable (outcome): A, B, C, D, E

As reported in [3] The outcome or the independent variable is of factor type consisting of 5 levels namely sitting-down, standing-up, standing, walking, and sitting. Since we discovered a large numbers of missing data (1287472) and empty cells, we decided to check out missing data per column to see if there is a pattern.

```
# Populate empty cells with NA's then count non NA's in each column
pmlTrain[pmlTrain == ""] <- NA
naCount1 <- length(which(is.na(pmlTrain) == TRUE))
len <- character(0)
for (i in 1:mCols) {
   len <- c(len, length(which(!is.na(pmlTrain[i]))))
}
```

After running the above script and then inspecting the object 'len', we discovered that a large number of columns are populated with only 406 values, while the other have no missing values with a length of 19622. We also found that the total number of missing values including empty cells are 1921600.

```
# Select columns with 19622 rows and with 406 rows
j = 0; k = 0; nam1 <- ""; nam2 <- "";
for (i in 1:mCols) {
   len1 <- length(which(!is.na(pmlTrain[i])))
   if (len1 > 406) {
      if (j == 0) {
         nam1 <- c(names(pmlTrain[i]))
      } else {
         nam1 <- c(nam1, names(pmlTrain[i]))
      }
      j <- j + 1
   } else {
      if (k == 0) {
         nam2 <- c(names(pmlTrain[i]))
      } else {
         nam2 <- c(nam2, names(pmlTrain[i]))
      }
      k <- k + 1
   }
}
n1 <- length(nam1); n2 <- length(nam2)
```

It was found out that there are 60 columns with no missing values, and 100 columns with mostly missing values. The data set was then reduced by excluding the columns with mostly missing values.

```
# Reduce the data set with populated columns
pmlTrain1 <- pmlTrain[ , names(pmlTrain) %in% c(nam1)]
pmlTest1 <- pmlTest[ , names(pmlTrain) %in% c(nam1)]

# Report data type of the features in the reduced data set
dataType1 <- sapply(pmlTrain1[1,],class)
```

We also noted that 7 columns "1. X, 2.user_name, 3. raw_timestamp_part_1, 4. raw_timestamp_part_2, 5. cvtd_timestamp, 6. new_window, 60. classe" contains date and time data, activity type etc. Rest of the 53 columns contains measured accelerometer data.

```r
# user names
usrNames <- levels(pmlTrain1$user_name)

# Activity Type
actvType <- levels(pmlTrain1$classe)
```

We also noted that
User Names: adelmo, carlitos, charles, eurico, jeremy, pedro
Activity Type: A, B, C, D, E

Exploratory Data Analysis and Features Selection: As part of the exploratory data analysis, Principal Component Analysis (PCA) and Correlation Analysis was performed to evaluate the key contributors towards activity. PCA was performed using singular value decomposition (SVD) to explore the data set. SVD can identify the important dimensions of a data set. Selected features with the higher singular values can then be included in the final model. This also helps not to over fit the predictive model.
A correlation matrix for the reduced data set was also created to quantify the linear association between the dependent or outcome variable 'classe' and the other features (dependent variable) in the pmlTrain1 data set.

```r
# Perform SVD on pmlTrain1 data set excluding the columns 1,2,3,4,5,6,60
pmlTrain2 <- pmlTrain1[-c(1,2,3,4,5,6,60)]
mCol2 <- ncol(pmlTrain2)
svdTot = svd(scale(pmlTrain2))

# Sort contributers in decreasing order then accumulate feature names
sContrib <- sort(svdTot$v[,2],decreasing = TRUE, index.return = TRUE)

# Sort contributers in decreasing order then accumulate feature names
sPCA <- character(0)
for (i in 1:mCol2) {
   n1 <- as.integer( lapply(sContrib, "[[", i)[2] )
   sPCA <- c( sPCA, paste( (names(pmlTrain2)[n1]) , sep='') )
}

# Perform correlation analysis and extract the row for the
# feature classe (exclude columns 1,2,3,4,5,6)
pmlTrain3 <- pmlTrain1[-c(1,2,3,4,5,6)]
pmlTrain3$classe <- as.numeric(pmlTrain3$classe)
sCor <- sort(cor(pmlTrain3)["classe",])
sCor <- sort(abs(cor(pmlTrain3)["classe",]), decreasing = TRUE)
#pmlTrain3 <- NULL

# Check dependent variables with high correlation coefficients
pmlTrain4 <- pmlTrain1[-c(1,2,3,4,5,6,60)]
M <- abs(cor(pmlTrain4))
diag(M) <- 0
xCorr <- which(M > 0.8, arr.ind = T)
pmlTrain4 <- NULL
```

```r
# Create plots
numClasse <- as.numeric(as.factor(pmlTrain1$classe))
par(mfrow=c(1,2))
plot(svdTot$u[,1],col = numClasse, pch = 19)
plot(svdTot$u[,2],col = numClasse, pch = 19)
```

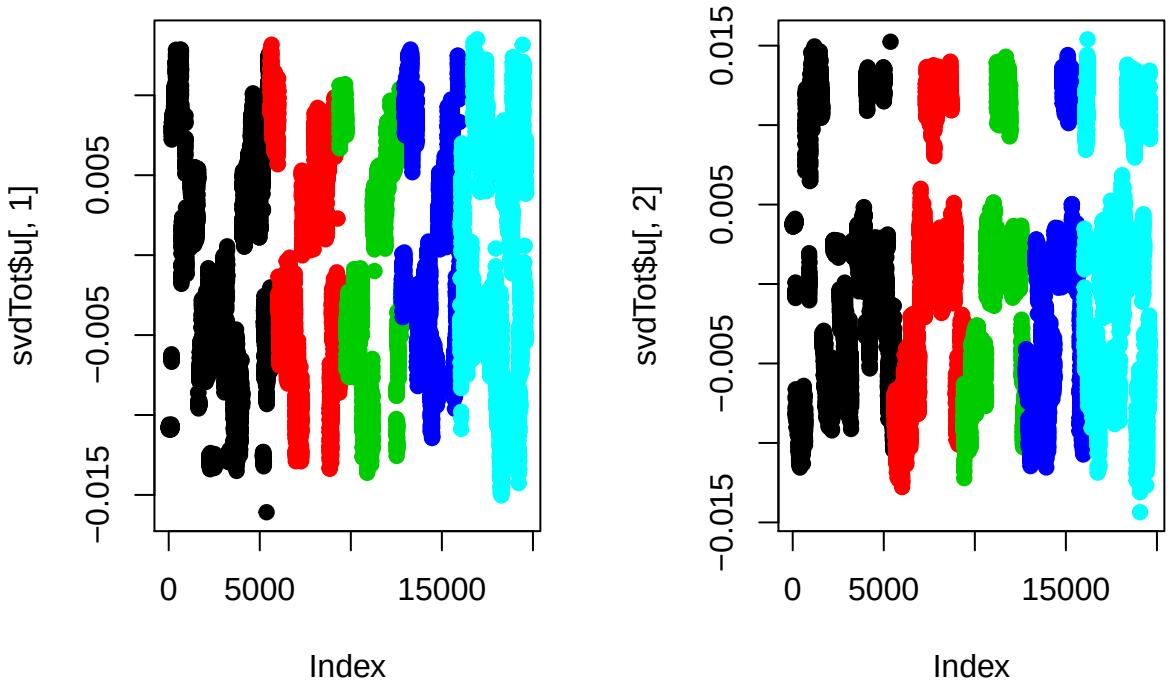

Figure 1: Exploratory plot showings SVD u and v vectors for 5 activities for all the 6 users

```
# Create plot
suppressWarnings(library(gclus))
```

```
## Loading required package: cluster
```

```
par(mfrow=c(1,1))
plot(sort(svdTot$v[,2]),col = "#3366FF", pch = 19)
```

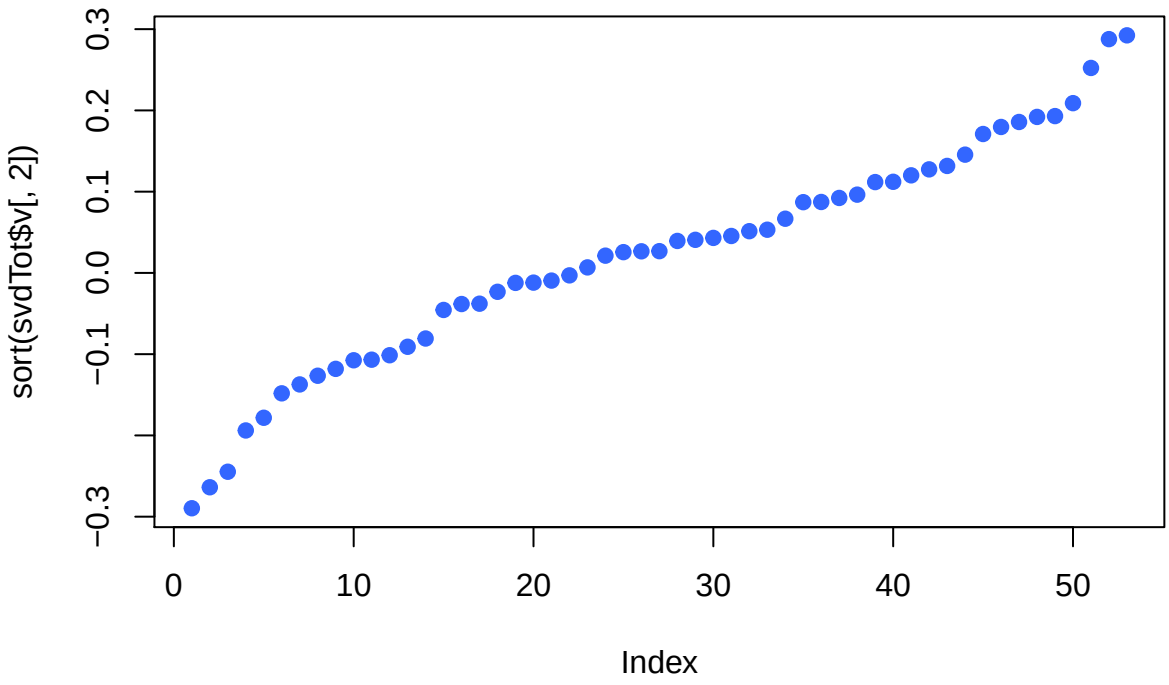

Figure 2: Plot showing singular values of 53 features of the data in descending order

The results of performed SVD analysis are shown in Figure-1. This figure shows the singular values of all 6 user without any exception. Figure-2 plot includes the sorted singular values in descending order, with highest to the lowest contributing features towards human physical activity variation. This plot accounts for all 6 users. The top 10 contributing features are listed in the table below in descending order.

```
# Reference the libraries
suppressWarnings(library(gclus))

# Get a subset of the data
dSub <- pmlTrain3[c("pitch_forearm","classe","magnet_belt_y","magnet_arm_y","accel_arm_x")]

# Get correlations of the subset
dCorr <- abs(cor(dSub))

# Get colors for the scatter plots
dColr <- dmat.color(dCorr)

# Reorder the features - highest correlation are closest to the diagonal
dOrder <- order.single(dCorr)
cpairs(dSub, dOrder, panel.colors=dColr, gap=.5, main="Scatter plots of the top contributors" )
```

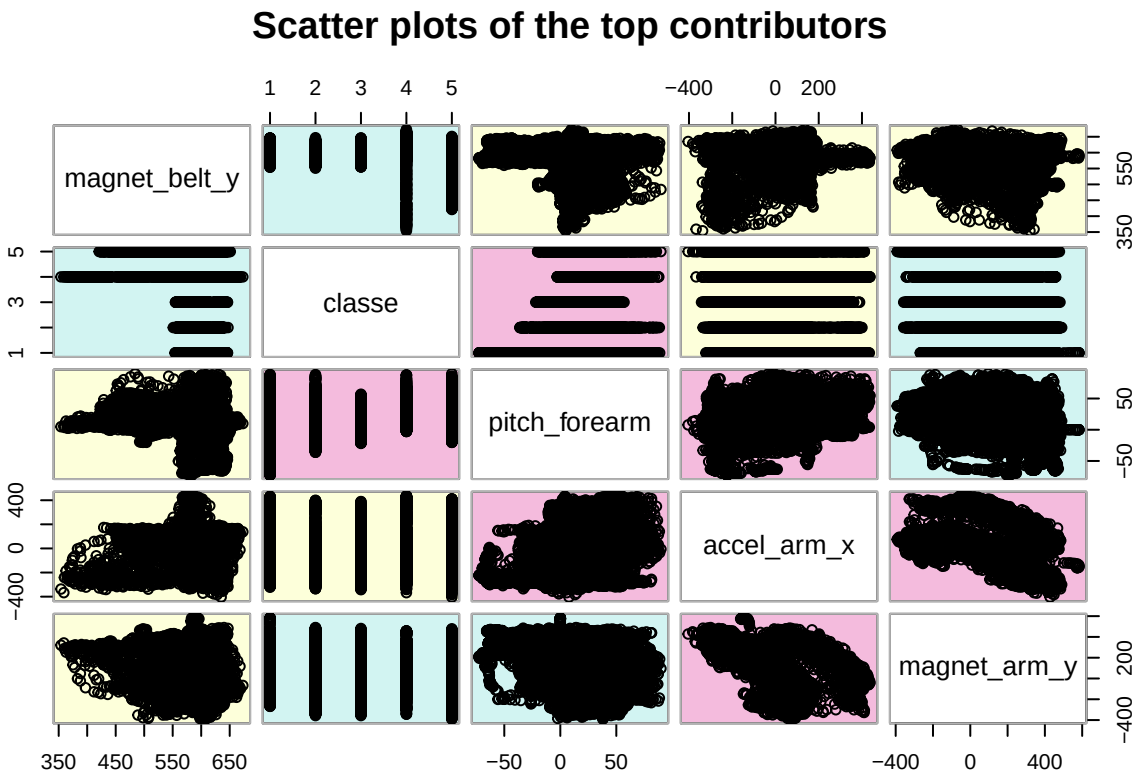

Figure-3: Scatter plots showing association between classe and 4 top contributors

The results of SVD analysis are shown below with 10 features in descending order.

```
##  [1] "accel_belt_x"          "magnet_belt_x"         "yaw_belt"
##  [4] "gyros_belt_y"          "gyros_forearm_x"       "gyros_belt_x"
##  [7] "gyros_arm_z"           "accel_dumbbell_y"      "magnet_dumbbell_y"
## [10] "total_accel_dumbbell"
```

From the correlation analysis, top ten correlation coefficients extracted for the outcome variable 'classe' are as follows.

```
sCor[2:11]
```

```
##       pitch_forearm        magnet_arm_x        magnet_belt_y
##              0.3438              0.2960              0.2903
##        magnet_arm_y         accel_arm_x     accel_forearm_x
##              0.2567              0.2426              0.1887
##    magnet_forearm_x       magnet_belt_z           pitch_arm
##              0.1821              0.1800              0.1777
## total_accel_forearm
##              0.1545
```

Predictive Model: After exploratory analysis was completed; we identified a select set of top contributor towards physical activity variation. We are ready to build a predictive model based on the knowledge we

acquired. we selected 'Random Forest' technique used for building our predictive model. Before we started building the model with Random Forest, we divided our data into two parts, namely train and cross validation (CV) data sets. The training set data included 70% and CV data consisted of 30% of the original data set. First 6 variables that included date and time are excluded from both training and CV data. We decided not to specify any particular dependent variable for model building, rather let Random Forest select the best features for modeling the outcome.

```
library(caret)

# Select 70% from the cleaned data set 'pmlTrain1'
sel70 <- createDataPartition(y = pmlTrain1$classe, p = 0.7, list = FALSE)

# Partiton the data "pmlTrain1", 70% for model training and 30% from CV
pmlTrn1 <- pmlTrain1[sel70, ]
pmlCV1 <- pmlTrain1[-sel70, ]

# Exclude date and time columns
pmlTrn1 <- pmlTrn1[-c(1,2,3,4,5,6)]
pmlCV1 <- pmlCV1[-c(1,2,3,4,5,6)]
pmlTst1 <- pmlTest1[-c(1,2,3,4,5,6)]

# Model the data 'pmlTrn1' using Random Forest using 50 trees
rfModel <- train(classe ~ ., data = pmlTrn1, method = "rf", ntree = 50, verbose=FALSE)

# Predict using the CV data set
rfPred <- predict(rfModel, pmlCV1)

# Predict physical activity using the 20 sample test data set
rfPred2 <- predict(rfModel, pmlTst1)

# Check the model accurany using the CV data
confusionMatrix(rfPred, pmlCV1$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    3    0    0    0
##          B    0 1135    1    0    1
##          C    0    1 1025    4    0
##          D    0    0    0  960    2
##          E    0    0    0    0 1079
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.996, 0.999)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.997
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    0.996    0.999    0.996    0.997
## Specificity             0.999    1.000    0.999    1.000    1.000
## Pos Pred Value          0.998    0.998    0.995    0.998    1.000
## Neg Pred Value          1.000    0.999    1.000    0.999    0.999
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.193    0.174    0.163    0.183
## Detection Prevalence    0.285    0.193    0.175    0.163    0.183
## Balanced Accuracy       1.000    0.998    0.999    0.998    0.999
```

```
# List top variables explaining most of the variation in the outcome
varImp(rfModel)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 53)
##
##                      Overall
## num_window            100.00
## roll_belt             60.13
## pitch_forearm         38.05
## yaw_belt              31.59
## magnet_dumbbell_z     30.15
## pitch_belt            27.94
## magnet_dumbbell_y     25.77
## roll_forearm          15.82
## roll_dumbbell         12.58
## magnet_dumbbell_x     11.99
## accel_dumbbell_y      10.84
## magnet_belt_y          9.82
## accel_belt_z           8.97
## accel_forearm_x        8.54
## accel_dumbbell_z       7.86
## magnet_belt_z          6.69
## total_accel_dumbbell   6.10
## accel_forearm_z        5.70
## roll_arm               5.61
## magnet_belt_x          5.18
```

Results and Conclusion: For out of sample error estimate we used cross validation (CV) method by evaluating the model on the CV or the holdout set.
Random Forest model predict the outcome using the holdout data set with an accuracy of 99.8%.
Our model exhibited 100% correct outcome on the 20 sample test data: B, A, B, A, A, E, D, B, A, A, B, C, B, A, E, E, A, B, B, B

Results of our analysis suggest that the Random Forest exhibited very high performance.
Even with a specified 50 trees limit to build the model, its accuracy was very high.
Random tree technique took care of potential confounders in our data set quite successfully.

We made no attempt to use "cross terms" (multiplying the top contributing features to generate new features), and "power terms" (e.g. square of features) in building our predictive model. Similarly, other predictive modeling techniques such as support vector machines and neural network were not tried for this analysis.

References:

1. URL: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
2. URL: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
3. Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6. Cited by 2 (Google Scholar)
4. Qualitative Activity Recognition of Weight Lifting Exercises; Eduardo Velloso, Andreas Bulling, Hans Gellersen, Wallace Ugulino, Hugo Fuks; Proceedings of 4th International Conference in Cooperation with SIGCHI.
5. R session information


```
sessionInfo()
```

```
## R version 3.1.0 (2014-04-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] randomForest_4.6-10 caret_6.0-35        ggplot2_0.9.3.1
## [4] lattice_0.20-29     gclus_1.3.1         cluster_1.15.2
##
## loaded via a namespace (and not attached):
##  [1] BradleyTerry2_1.0-5 brglm_0.5-9         car_2.0-21
##  [4] class_7.3-10        codetools_0.2-8     colorspace_1.2-4
##  [7] compiler_3.1.0      digest_0.6.4        e1071_1.6-4
## [10] evaluate_0.5.5      foreach_1.4.2       formatR_0.10
## [13] grid_3.1.0          gtable_0.1.2        gtools_3.4.1
## [16] htmltools_0.2.4     httr_0.3            iterators_1.0.7
## [19] knitr_1.6           lme4_1.1-7          MASS_7.3-31
## [22] Matrix_1.1-3        minqa_1.2.3         munsell_0.4.2
## [25] nlme_3.1-117        nloptr_1.0.4        nnet_7.3-8
## [28] plyr_1.8.1          proto_0.3-10        Rcpp_0.11.1
## [31] RCurl_1.95-4.1      reshape2_1.4        rmarkdown_0.2.64
## [34] rpart_4.1-8         scales_0.2.4        splines_3.1.0
## [37] stringr_0.6.2       swirl_2.2.14        testthat_0.8.1
## [40] tools_3.1.0         yaml_2.1.13
```