Practical Machine Learning Course Project

Author: Sachin Singh Date: 09/25/2015

Introduction

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. More information is available from the website here: http://groupware.les.inf.pucrio.br/har (http://groupware.les.inf.pucrio.br/har).

Data exploration

Closer examination of the variables reveals that many derived variables (such as skewness and kurtosis) are coded as factors while they should be numeric. When converting those to numeric variables many values are missing and NAs are introduced. Checking the precentage of NAs vs. real values for those variables reveals that most entries are NA; since they contribute little information to the dataset they can be removed. All columns that have more than 90% of NAs are removed from the training data frame.

Similar, the X, user_name, timestamp variables are removed since this is information specific to this dataset that should not have an influence on the classification of future sets.

After cleaning the data, we are left with 52 predictor variables.

Reproducibility

An overall pseudo-random number generator seed was set at 1234 for all code. In order to reproduce the results below, the same seed should be used. Different packages were downloaded and installed, such as caret and randomForest. These should also be installed in order to reproduce the results below (please see code below for ways and syntax to do so).

The outcome variable is classe, a factor variable with 5 levels. For this data set, â €œparticipants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions:

Prediction evaluations will be based on maximizing the accuracy and minimizing the out-of-sample error. All other available variables after cleaning will be used for prediction. Two models will be tested using decision tree and random forest algorithms. The model with the highest accuracy will be chosen as our final model.

The outcome variable classe is an unordered factor variable. Thus, we can choose our error type as 1accuracy. We have a large sample size with N= 19622 in the Training data set. This allow us to divide our Training sample into subTraining and subTesting to allow cross validation. Features with all missing values will be discarded as well as features that are irrelevant. All other features will be kept as relevant variables. Decision tree and random forest algorithms are known for their ability of detecting the features that are important for classification [2]. Feature selection is inherent, so it is not so necessary at the data preparation phase. Thus, there wonâ€™t be any feature selection section in this report.

Cross validation

Cross validation will be performed by sub-sampling our training data set randomly without replacement into 2 sub-samples: subTraining data (75% of the original Training data set) and subTesting data (25%). Our models will be fitted on the subTraining data set, and tested on the subTesting data. Once the most accurate model is choosen, it will be tested on the original Testing data set.

Out-of-sample error

The expected out-of-sample error will correspond to the quantity: 1accuracy in the cross validation data. Accuracy is the proportion of correct classified observation over the total sample in the subTesting data set. Expected accuracy is the expected accuracy in the outofsample data set (i.e. original testing data set). Thus, the expected value of the outofsample error will correspond to the expected number of missclassified observations/total observations in the Test data set, which is the quantity: 1accuracy found from the cross validation data set.

Results

```
library(caret)
df <- read.csv("pml-training.csv")
test <- read.csv("pml-testing.csv")
```

```
#Random forest for classification and regression
library(randomForest)
# Regressive Partitioning and Regression trees
library(rpart)
# Decision Tree plot
library(rpart.plot)
# setting the overall seed for reproducibility
set.seed(1234)
```

Data Loading

```
trainingset <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0
""))
# Loading the testing data set
testingset <- read.csv('pml-testing.csv', na.strings=c("NA","#DIV/0!"
# Check dimensions for number of variables and number of observations
dim(trainingset)
```

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

```
dim(testingset)
```

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

```
# Delete columns with all missing values
trainingset<- trainingset[,colSums(is.na(trainingset)) == 0]
testingset <- testingset[,colSums(is.na(testingset)) == 0]
```

```
# Removing user_name, raw_timestamp_part_1, raw_timestamp_part_,2 cvt
trainingset <- trainingset[,-c(1:7)]
testingset <- testingset[,-c(1:7)]
# and have a look at our new datasets:
# head(trainingset)
# head(testingset)
dim(trainingset)
```

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

```
dim(testingset)
```

```
## [1] 20 53
```

Partitioning the training data set to allow cross validation

The training data set contains 53 variables and 19622 obs. The testing data set contains 53 variables and 20 obs. In order to perform crossvalidation, the training data set is partionned into 2 sets: subTraining (75%) and subTest (25%). This will be performed using random sub-sampling without replacement.

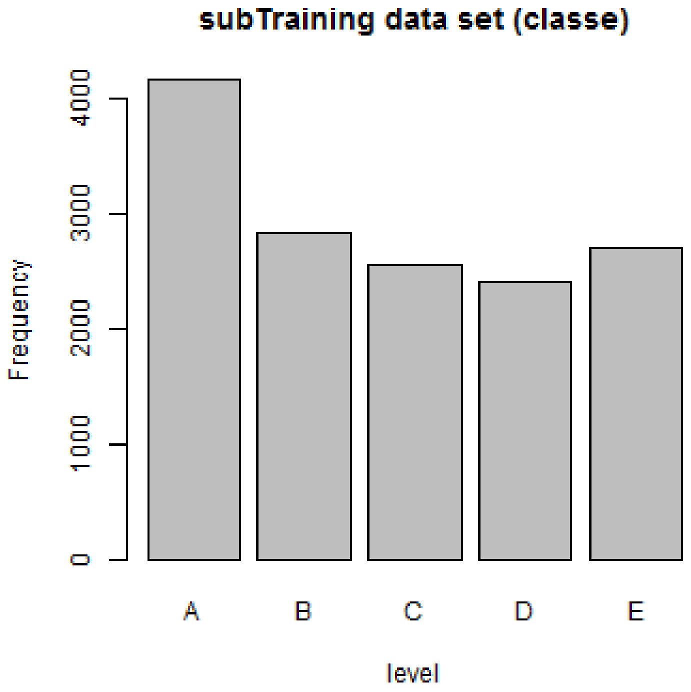
```
subsamples <- createDataPartition(y=trainingset$classe, p=0.75, list=
subTraining <- trainingset[subsamples, ]
subTesting <- trainingset[-subsamples, ]
# head(subTraining)
# head(subTesting)
dim(subTraining)
```

```
## [1] 14718    53
```

```
dim(subTesting)
```

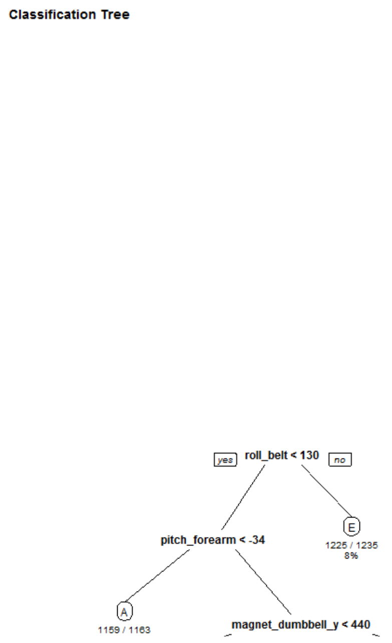
```
## [1] 4904   53
```

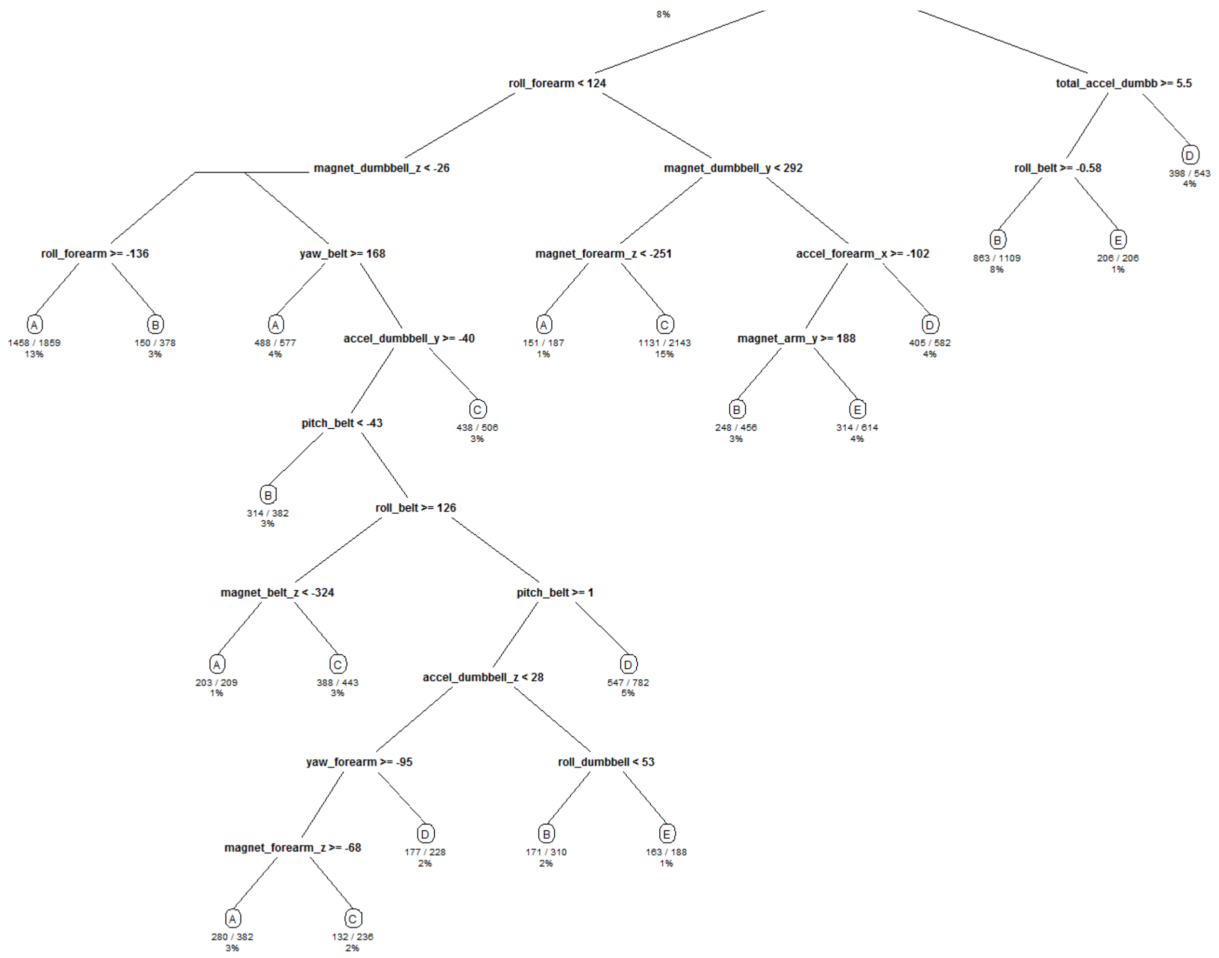
```
plot(subTraining$classe, col="grey", main="subTraining data set (clas
```



1st Prediction Model using decision tree

```
model1 <- rpart(classe ~ ., data=subTraining, method="class")
# Predicting:
prediction1 <- predict(model1, subTesting, type = "class")
# Plot of the Decision Tree
rpart.plot(model1, main="Classification Tree", extra=102, under=TRUE,
```





```
# Test results on subTesting data set
confusionMatrix(prediction1, subTesting$classe)
```

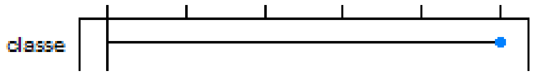
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1235  157   16   50   20
##          B   55  568   73   80  102
##          C   44  125  690  118  116
##          D   41   64   50  508   38
##          E   20   35   26   48  625
## 
## Overall Statistics
##                                           
##                Accuracy : 0.7394          
##                  95% CI : (0.7269, 0.7516)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.6697          
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8853   0.5985   0.8070   0.6318   0.6937
## Specificity            0.9307   0.9216   0.9005   0.9529   0.9678
## Pos Pred Value         0.8356   0.6469   0.6313   0.7247   0.8289
## Neg Pred Value         0.9533   0.9054   0.9567   0.9296   0.9335
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2518   0.1158   0.1407   0.1036   0.1274
## Detection Prevalence   0.3014   0.1790   0.2229   0.1429   0.1538
## Balanced Accuracy      0.9080   0.7601   0.8537   0.7924   0.8307
```

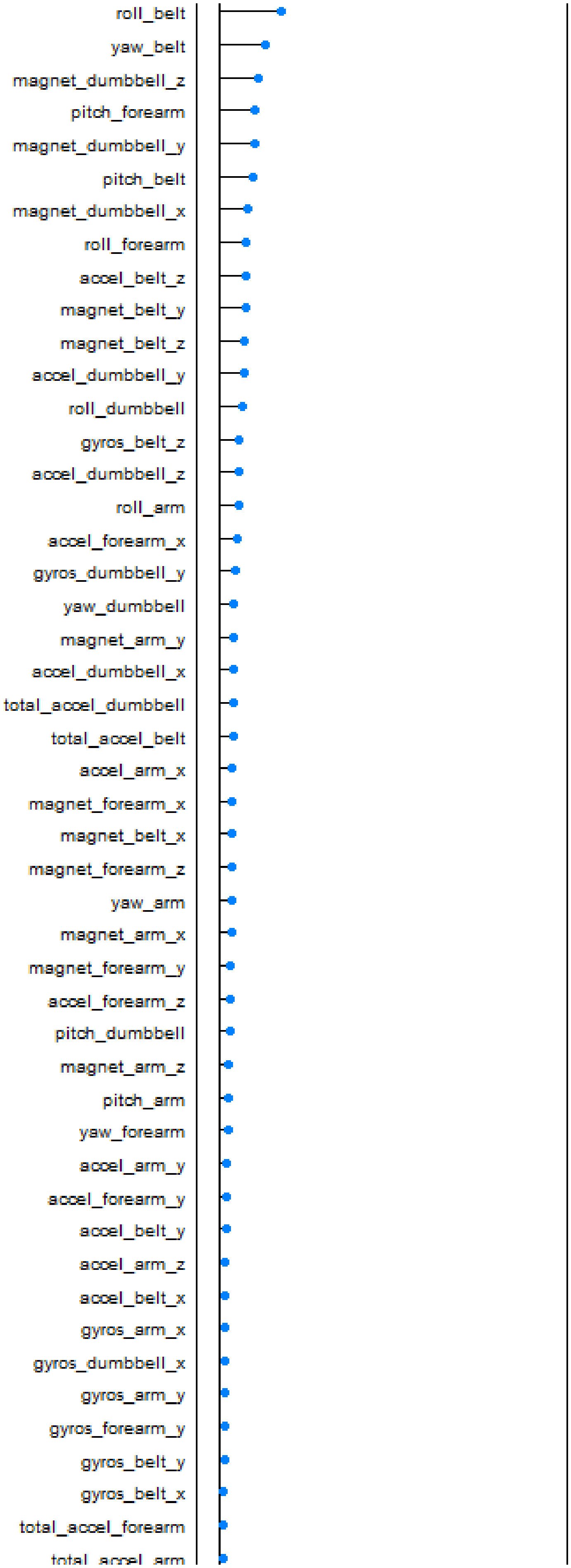
2nd Prediction Model using randomForest

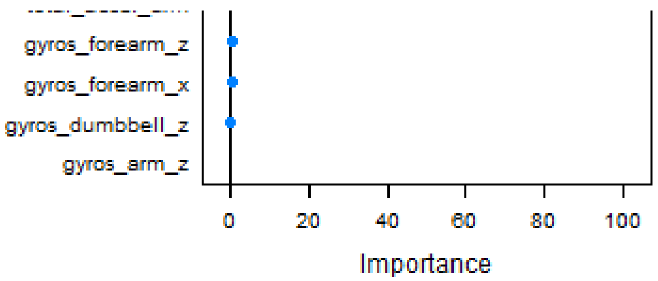
```
model2 <- randomForest(classe ~. , data=subTraining, method="class")
prediction2 <- predict(model2, subTesting, type = "class")
# Test results on subTesting data set
confusionMatrix(prediction2, subTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    3    0    0    0
##          B    1  944   10    0    0
##          C    0    2  843    6    0
##          D    0    0    2  798    0
##          E    0    0    0    0  901
##
## Overall Statistics
##
##                Accuracy : 0.9951
##                  95% CI : (0.9927, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9947   0.9860   0.9925   1.0000
## Specificity            0.9991   0.9972   0.9980   0.9995   1.0000
## Pos Pred Value         0.9979   0.9885   0.9906   0.9975   1.0000
## Neg Pred Value         0.9997   0.9987   0.9970   0.9985   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1925   0.1719   0.1627   0.1837
## Detection Prevalence   0.2849   0.1947   0.1735   0.1631   0.1837
## Balanced Accuracy      0.9992   0.9960   0.9920   0.9960   1.0000
```

```
# The Kappa statistic of 0.994 reflects the out-of-sample error.
randFor <- train(trainingset[,-57],
                       trainingset$classe,
                       tuneGrid=data.frame(mtry=3),
                       trControl=trainControl(method="none")
                       )
plot(varImp(randFor))
```




roll_belt
yaw_belt
magnet_dumbbell_z
pitch_forearm
magnet_dumbbell_y
pitch_belt
magnet_dumbbell_x
roll_forearm
accel_belt_z
magnet_belt_y
magnet_belt_z
accel_dumbbell_y
roll_dumbbell
gyros_belt_z
accel_dumbbell_z
roll_arm
accel_forearm_x
gyros_dumbbell_y
yaw_dumbbell
magnet_arm_y
accel_dumbbell_x
total_accel_dumbbell
total_accel_belt
accel_arm_x
magnet_forearm_x
magnet_belt_x
magnet_forearm_z
yaw_arm
magnet_arm_x
magnet_forearm_y
accel_forearm_z
pitch_dumbbell
magnet_arm_z
pitch_arm
yaw_forearm
accel_arm_y
accel_forearm_y
accel_belt_y
accel_arm_z
accel_belt_x
gyros_arm_x
gyros_dumbbell_x
gyros_arm_y
gyros_forearm_y
gyros_belt_y
gyros_belt_x
total_accel_forearm
total_accel_arm



Conclusion

As expected, Random Forest algorithm performed better than Decision Trees. Accuracy for Random Forest model was 0.995 (95% CI: (0.993, 0.997)) compared to 0.739 (95% CI: (0.727, 0.752)) for Decision Tree model. The random Forest model is choosen. The accuracy of the model is 0.995. The expected out-of-sample error is estimated at 0.005, or 0.5%. The expected out-of-sample error is calculated as 1 accuracy for predictions made against the cross validation set. Our Test data set comprises 20 cases. With an accuracy above 99% on our cross validation data, we can expect that very few, or none, of the test samples will be miss classified.

Course Submission

```
predictfinal <- predict(model2, testingset, type="class")
predictfinal
```

```
##  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
```

```
# Write files for submission
pml_write_files = function(x){
n = length(x)
for(i in 1:n){
filename = paste0("problem_id_",i,".txt")
write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=
}
}
pml_write_files(predictfinal)
```

References

[1] Velloso, E. Bulling, A. Gellersen, H. Ugulino, W. Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human â€™13) . Stuttgart, Germany: ACM SIGCHI, 2013.

[2]Breiman, Leo. 1996. Bagging Predictors. Machine Learning 24 (2): 123-140. doi:10.1007 FBF00058655.