Self Movement Prediction

Ingrida

27 November 2016

```
library(caret)
```

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

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

```
library(ggplot2)
library(survival)
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
library(randomForest)
```

```
## randomForest 4.6-12
```

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

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

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

```
library(rpart)
library(rpart.plot)
library(gbm)
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
library(plyr)
library(splines)
library(parallel)
```

Executive Summary

The goal of this project is to predict the manner in which 6 participants did the exercise.The data were colected using accelerometers on the belt, forearm, arm, and dumbell. The participant were asked to perform barbell lifts correctly and incorrectly in 5 different ways:

- exactly according to the specification (Class A)
- throwing the elbows to the front (Class B)
- lifting the dumbbell only halfway (Class C)
- lowering the dumbbell only halfway (Class D)
- throwing the hips to the front (Class E)

The data for this project come from http://groupware.les.inf.puc-rio.br/har. First, data are collected, cleaned and preprocessed. Then the training data are splitted into training (70%) and validation (30%) sets where the former is used to fit prediction model and the latter to validate the fitted model. In this study, we use three different methods, namely GBM, LDA and RF. The model with the highest accuracy is selcted as our final model to make prediction for 20 samples.

Data preparation

While reading the data, cells with empty, NA or #DIV/0! are replaced with NA values. Then columns with missing values are removed.

```
# Read training and testing data
trainD <- read.csv("~/datasciencecoursera/Practical Machine Learning/pml-training.csv", head=TRUE, sep=
testD <- read.csv("~/datasciencecoursera/Practical Machine Learning/pml-testing.csv", head=TRUE, sep=",
# Check dimensions for training and testing data
dim(trainD)
```

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

```
dim(testD)
```

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

```
# Delete columns with all missing values
trainD<-trainD[,colSums(is.na(trainD)) == 0]
testD <-testD[,colSums(is.na(testD)) == 0]
# Check if we removed columnss with missing data values
sum(is.na(trainD) == TRUE)
```

```
## [1] 0
```

```
sum(is.na(testD) == TRUE)
```

```
## [1] 0
```

Next, we identify variables whose variance is nearly 0 using function nearZeroVar and remove them from our data as well.

```
nzColumns <- nearZeroVar(trainD, saveMetrics = TRUE)
tidy_train <- trainD[, nzColumns$nzv==FALSE]
tidy_train$classe <- as.factor(tidy_train$classe)
tidy_test <- testD[, nzColumns$nzv==FALSE]
```

The last step in cleaning data is to remove variables which are irrelevant to our prediction. These are X, timestamp, user_name and problem_id.

```
id1 <- grepl("X|timestamp|user_name", names(tidy_train))
tidy_train <- tidy_train[, which(id1 ==FALSE)]
id2 <- grepl("X|timestamp|user_name|problem_id", names(tidy_test))
tidy_test <- tidy_test[, which(id2 ==FALSE)]
```

Model fitting

Partitioning of a dataset

We split the tidy training data into training set of 70% of all data which is used for fitting the prediction model and test set of 30% of all data which is used for validating the fitted model.

```
set.seed(12345)
partT <- createDataPartition (tidy_train$classe, p=0.7, list=FALSE)
train <- tidy_train[partT,]
test <- tidy_train[-partT,]
```

Our aim of the model is to predict the variable "classe" which contains 5 levels: A, B, C, D and E.

```
table(train$classe)
```

```
##
##    A    B    C    D    E
## 3906 2658 2396 2252 2525
```

Training different models

We train three different models: Gradient Boosted Machine (GBM), Linear Discriminant Analysis (LDA) and Random Forest (RF). We use 10-folds crossvalidation to find the best model.

```
control <- trainControl(method="cv", number=10)
#training
model1 <- train(classe ~ ., data = train, method = "gbm",trControl=control, verbose=FALSE)
model2 <- train(classe ~ ., data = train, method = "lda",trControl=control)
```

```
## Loading required package: MASS
```

```
model3 <- randomForest(classe ~., data=train, method="class",trControl=control)
# prediction
pred1 <- predict(model1, test)
pred2 <- predict(model2, test)
pred3 <- predict(model3, test)
# GBM model
confusionMatrix(pred1, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1665   14    0    2    0
##          B    4 1111   17    3    3
##          C    0   14 1004   21    2
##          D    5    0    5  938   13
##          E    0    0    0    0 1064
##
## Overall Statistics
##
##                Accuracy : 0.9825
##                  95% CI : (0.9788, 0.9857)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9779
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9946   0.9754   0.9786   0.9730   0.9834
## Specificity            0.9962   0.9943   0.9924   0.9953   1.0000
## Pos Pred Value         0.9905   0.9763   0.9645   0.9761   1.0000
## Neg Pred Value         0.9979   0.9941   0.9955   0.9947   0.9963
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2829   0.1888   0.1706   0.1594   0.1808
## Detection Prevalence   0.2856   0.1934   0.1769   0.1633   0.1808
## Balanced Accuracy      0.9954   0.9849   0.9855   0.9842   0.9917
```

```
# LDA model
confusionMatrix(pred2, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1404  168   92   52   44
##          B   39  728   99   47  166
##          C  110  146  655  133   89
##          D  112   45  144  694  103
##          E    9   52   36   38  680
```

```
##
## Overall Statistics
##
##                Accuracy : 0.7071
##                  95% CI : (0.6952, 0.7187)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6291
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8387   0.6392   0.6384   0.7199   0.6285
## Specificity            0.9155   0.9260   0.9016   0.9179   0.9719
## Pos Pred Value         0.7977   0.6747   0.5781   0.6321   0.8344
## Neg Pred Value         0.9345   0.9145   0.9219   0.9436   0.9207
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2386   0.1237   0.1113   0.1179   0.1155
## Detection Prevalence   0.2991   0.1833   0.1925   0.1866   0.1385
## Balanced Accuracy      0.8771   0.7826   0.7700   0.8189   0.8002
```

```
# RF model
confusionMatrix(pred3, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    6    0    0
##          C    0    1 1020   13    0
##          D    0    0    0  951    2
##          E    0    0    0    0 1080
##
## Overall Statistics
##
##                Accuracy : 0.9954
##                  95% CI : (0.9933, 0.997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9942
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9942   0.9865   0.9982
## Specificity            0.9988   0.9987   0.9971   0.9996   1.0000
## Pos Pred Value         0.9970   0.9947   0.9865   0.9979   1.0000
## Neg Pred Value         1.0000   0.9987   0.9988   0.9974   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
```

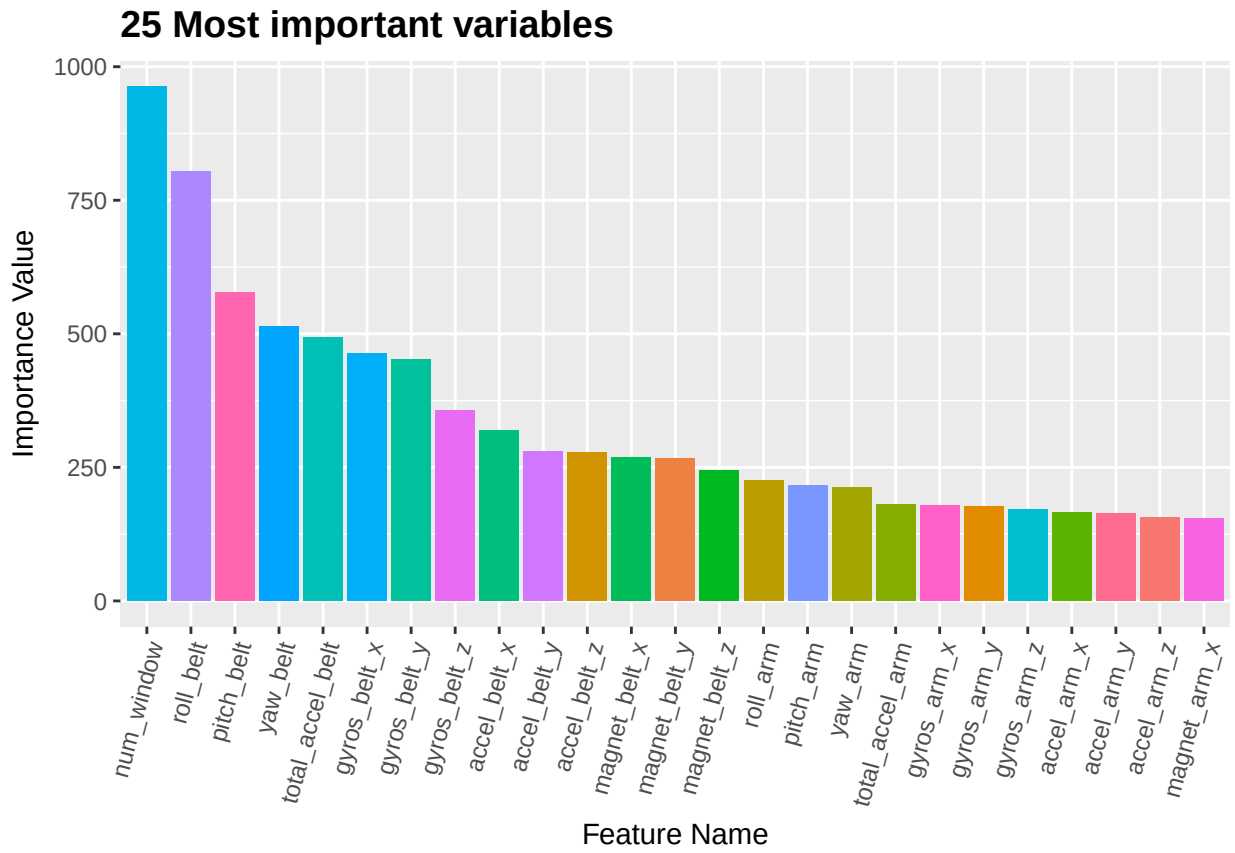
```
## Detection Rate         0.2845   0.1925   0.1733   0.1616   0.1835
## Detection Prevalence   0.2853   0.1935   0.1757   0.1619   0.1835
## Balanced Accuracy      0.9994   0.9967   0.9956   0.9931   0.9991
```

Selecting prediction model

Accuracy and expected out-of-sample error are important while selecting the final model. As expected, RF performed better than GBM and LDA. Accuracy of RF model is 0.995 (95% CI: (0.993, 0.997)) compared to 0.983 (95% CI : (0.9788, 0.9857)) GBM and 0.707 (95% CI : (0.6952, 0.7187)) LDA. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set and for RF model it is very small, i.e. equal to 0.0046.

The following figure demonstrates first 25 most important predictors. As you can observe the num_window is the most important.

```
imp <- varImp(model3)
impN<-rownames(imp)
order<-data.frame(varnames=impN,imp)
order<-arrange(order,desc(Overall))[1:25,]
ggplot(order, aes(x=reorder(impN[1:25],desc(Overall))[1:25],y=Overall,fill=varnames)) + geom_bar(stat="
  xlab("Feature Name")+guides(fill=FALSE)+
  ylab("Importance Value")+ggtitle("25 Most important variables")+
  theme(axis.text.x=element_text(angle=75,hjust=1)) +
  theme(plot.title = element_text(size=14, face="bold"))
```



Predicting on test data

Test data set consists of 20 samples. With an accuracy above 99% on our validation data, we can expect that very few, or none, of the test samples will be missclassified.

```
predictfinal <- predict(model3, tidy_test, 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
```

Conclusions

It is not a surprise that RF model was the most accurate among GBM and LDA and provided the best results. Random forests are able to handle a large number of variables including categorical ones, especially when the interactions between variables are unknown. The obtained prediction model is accurate (higher than 99% accuracy) to predict the manner in which the exercise was performed.