PracticalMachineLearning_Project

wenlarry

2/2/2017

Executive Summary The goal is to predict 5 diffferent types of exercise from accelerometers on the belt, forearm, arm and dumbell of 6 paricipants. More information is available from the website here:

http://groupware.les.inf.puc-rio.br/har

After exploring the data, we decide to use random forest to build a predictive model. The model was then cross validated with 99.3% accuracy. Then the model was used to predict the validation data with a 99.1% accuracy. Finally, the model was applied to the original dataset.

Load Libraries

```
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

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(corrplot)
library(rpart)
library(rpart.plot)
```

Download Data

```
trainUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-tr
aining.csv"

testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-tes
ting.csv"
```

```
trainFile <-"./data/pml-training.csv"
testFile <-"./data/pml-testing.csv"
if (!file.exists("./data")){dir.create("./data")
}
if(!file.exists(trainFile)) {download.file(trainUrl, destfile=trainFile
, method="curl")
}
if(!file.exists(testFile)) {download.file(testUrl,
destfile=testFile, method="curl")
}
```

Place files in dataframes

```
trainDF <-read.csv("./data/pml-training.csv")
testDF <-read.csv("./data/pml-testing.csv")

dim(trainDF)

## [1] 19622   160

## [1] 19622 obs 160 variables
dim(testDF)

## [1]  20 160

## [1] 20 obs 160 variables
```

The classe variable in the training set is the prediction outcome.

Data Cleaning

Using the (str) command on the training dataframe, we identify that there are missing values and information that are not needed for prediction.

1) Remove missing values in columns.

```
trainDF <-trainDF[,colSums(is.na(trainDF))==0]
testDF <-testDF[,colSums(is.na(testDF))==0]
```

2) The data set contains three types of metrics.
a) Raw metrics from the sensors
b) Derived metrics for roll, pitch and yaw
c) Summaries (min, max. etc) of the derived metrics

The summaries are not reported for every observation. As such, these 'summaries' metrics shall be excluded from this analysis, leaving just the raw and derived metrics.

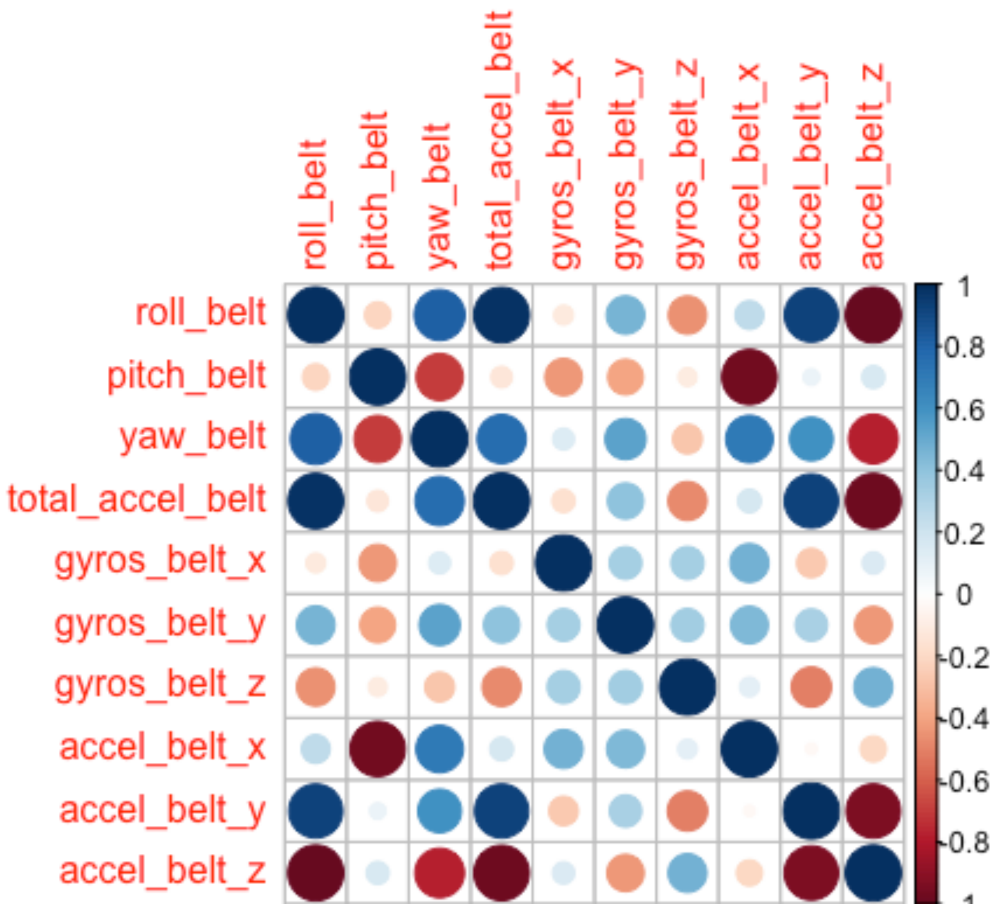
In addition, the identifier fields are removed. These include row numbers, window ids and time stamps.

```r
classe <-trainDF$classe
trainClean <- grepl("^X|timestamp|window", names(trainDF))
trainDF <-trainDF[, !trainClean]
trainClean2<-trainDF[, sapply(trainDF, is.numeric)]
trainClean2$classe<-classe
testClean <-grepl("^X|timestamp|window", names(testDF))
testDF <-testDF[, !testClean]
testClean2 <- testDF[, sapply(testDF,is.numeric)]

dim(trainClean2)

## [1] 19622    53

# [1] 19622 obs 53 variables
dim(testClean2)

## [1] 20 53

# [1] 20 obs 53 variables
```

** Data Exploration**

Explore correlation of variables and plotting few variables, we observe that there are variables with high correlation.

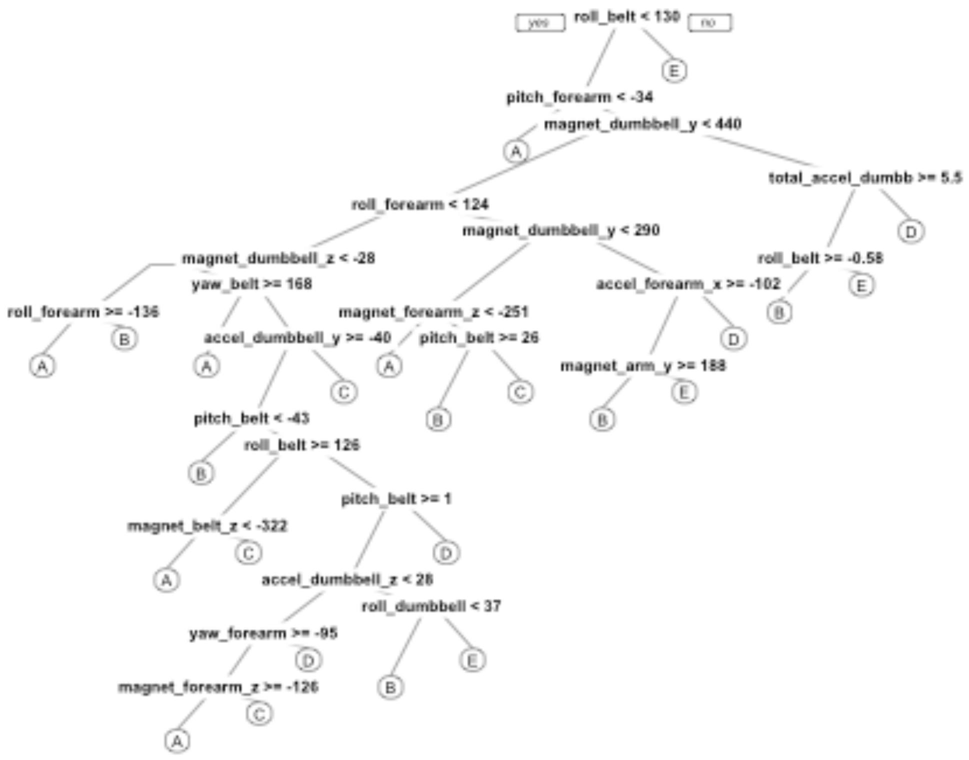
```r
corVar <- cor(trainClean2[, -53], use="pair")
```



Although there are high correlation of variables, we will not remove them, rather we use Random Forest as our machine learning algorithm. Random Forest is robust in dealing with correlated covariates.

Decision Tree

```
tree<-rpart(classe~.,data=trainClean2, method="class")
```



Model Building

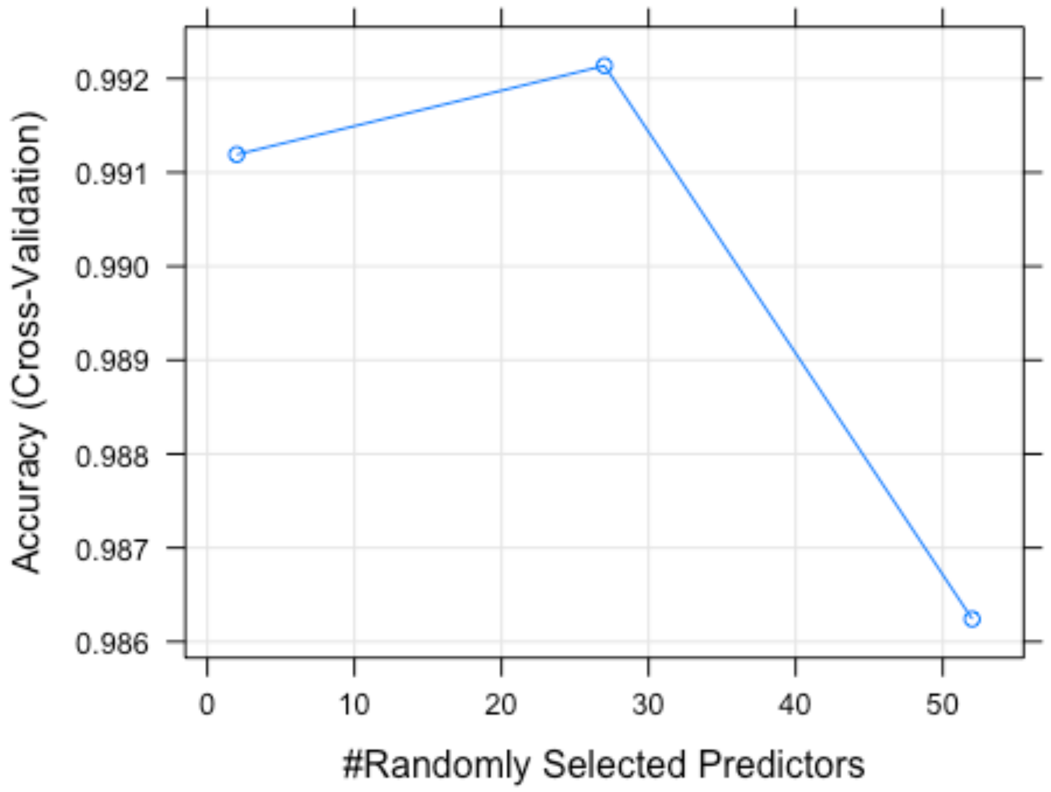
1) Split the cleaned training set into training data (70%) and validation data (30%)

```
set.seed (2317)
inTrain<-createDataPartition(trainClean2$classe,p=0.70,list=F)
trainDat <- trainClean2[inTrain,]
testDat <-trainClean2[-inTrain,]
```

2) Use Random Forest to fit a predictive model with a conventional 10 fold cross validation. Our data exploration revealed that accuracy converged after about 100 trees.

```
rfControl <-trainControl(method="cv", 10)
rfModel <- train(classe~., data=trainDat,method="rf",
trControl=rfControl,ntree=100)
rfModel

## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
```

```
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12365, 12363, 12364, 12364, 12361, 12362, .
..
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9911908  0.9888562
##   27    0.9921370  0.9900535
##   52    0.9862409  0.9825927
##
## Accuracy was used to select the optimal model using  the largest val
ue.
## The final value used for the model was mtry = 27.
```



The final value used by the model was mtry = 2 with 99.1 accuracy. It has 52 randomly selected predictors from a 10 fold cross validation. The out of sample error would be 0.9% (i.e. 1 - 0.991) .

3) Predictions on the validation data

```
rfPredict<-predict(rfModel,testDat)
confusionMatrix(testDat$classe, rfPredict)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1671    3    0    0    0
##          B    8 1126    5    0    0
##          C    0    8 1013    5    0
##          D    0    0    9  954    1
##          E    0    0    1    2 1079
##
## Overall Statistics
##
##                Accuracy : 0.9929
##                  95% CI : (0.9904, 0.9949)
##     No Information Rate : 0.2853
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.991
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9952   0.9903   0.9854   0.9927   0.9991
## Specificity            0.9993   0.9973   0.9973   0.9980   0.9994
## Pos Pred Value         0.9982   0.9886   0.9873   0.9896   0.9972
## Neg Pred Value         0.9981   0.9977   0.9969   0.9986   0.9998
## Prevalence             0.2853   0.1932   0.1747   0.1633   0.1835
## Detection Rate         0.2839   0.1913   0.1721   0.1621   0.1833
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9973   0.9938   0.9914   0.9953   0.9992
```

```
accuracy <- postResample( rfPredict,testDat$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9928632 0.9909719
```

```
outSam <- 1-(confusionMatrix(testDat$classe,rfPredict)$overall[1])
outSam
```

```
##    Accuracy
## 0.007136788
```

The estimated accuracy of the model is 99.3% the estmated out-of sample error is 0.7%

Test Dataset Prediction

Apply the model to the original testing data after removing the column 'problem_id'

```
Testdata <- predict(rfModel, testClean2[,-length(names(testClean2))] )
Testdata

##  [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
```