Leyla Samiee Practical Machine Learning Course Handout

GitHub Documents

Background

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.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:
[https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv]

The test data are available here:
[https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv]

The data for this project come from this source: [http://groupware.les.inf.puc-rio.br/har]. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

What you should submit

The goal of your project is to predict the manner in which they did the exercise. This is the “classe” variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross

validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-). You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

```
library("readr", lib.loc="/Library/Frameworks/R.framework/Versions/3.5/Resourc
library("rpart", lib.loc="/Library/Frameworks/R.framework/Versions/3.5/Resourc
library("rpart.plot", lib.loc="/Library/Frameworks/R.framework/Versions/3.5/Re
library("randomForest", lib.loc="/Library/Frameworks/R.framework/Versions/3.5/
```

```
## randomForest 4.6-14

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

```
library("caret", lib.loc="/Library/Frameworks/R.framework/Versions/3.5/Resourc
```

```
## Loading required package: lattice

## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

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

Reading input files

```
## [1] 19622   160

## [1]  20 160
```

Looking at the data set, it have some missing data, some data cleansing and preporcessiong is needed on the data.

cleaning up the data

```
ls_training_data <- ls_training_data[,colSums(is.na(ls_training_data)) == 0]
ls_testing_data <- ls_testing_data[,colSums(is.na(ls_testing_data)) == 0]

# remove column 1 - 7:  user_name, raw_timestamp_part_1, raw_timestamp_part_,2
# these are not useful for the analysis

ls_testing_data <- ls_testing_data[,-c(1:7)]
ls_training_data <- ls_training_data[,-c(1:7)]

# head(ls_testing_data)
# head(ls_training_data)
```

using 75 - 25% rules we will break the dataset to two group of training and testing

```
ls_partition <- createDataPartition(ls_training_data$classe, times = 1, p=0.75

ls_subtraining <- ls_training_data[ls_partition, ]
ls_subtesting <- ls_training_data[-ls_partition, ]
dim(ls_subtraining)
```

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

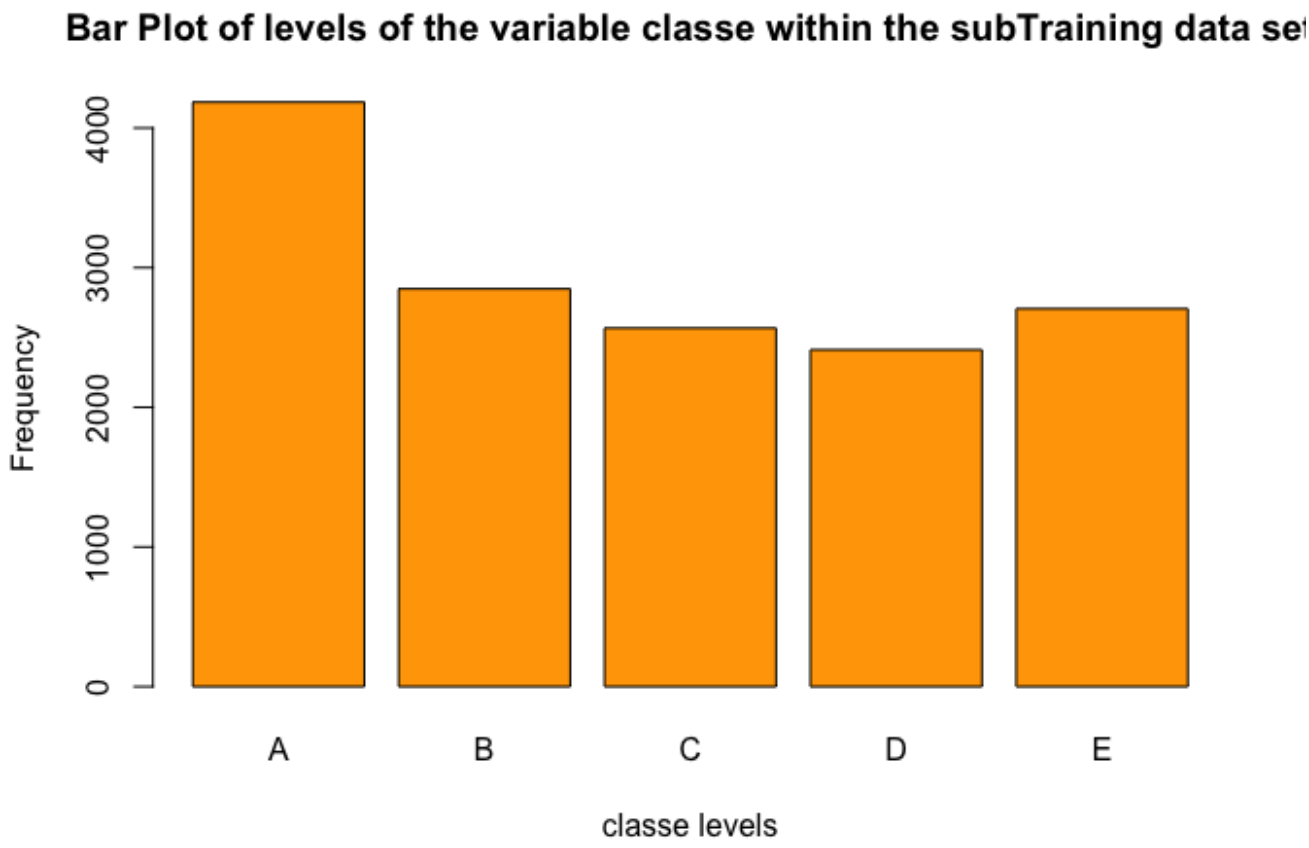
```
dim(ls_subtesting)
```

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

```
# head(ls_subtraining)
# head(ls_subtesting)
```

Quick look at the classe data in a graph. To observe each level frequency in comparison with eachother.

```
plot(ls_subtraining$classe, col="orange", main="Bar Plot of levels of the vari
```



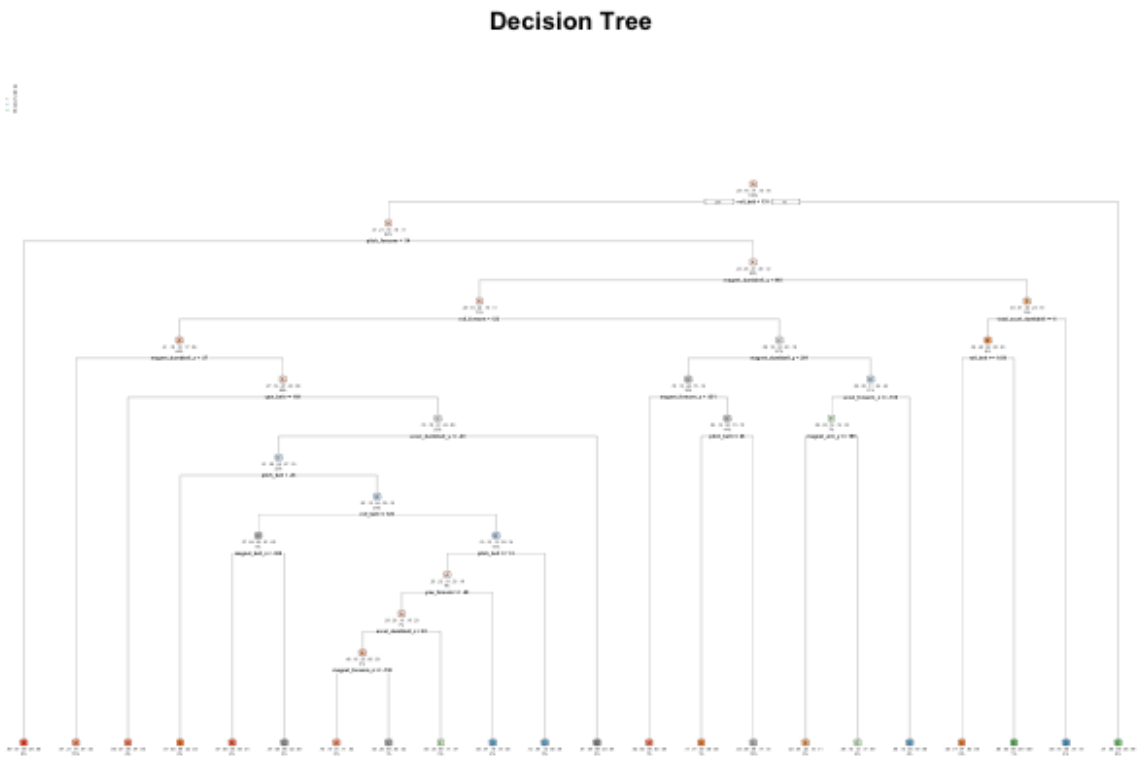
Level A with 4000 event of count, has the highest frequency. level D with more than 2600 time is the least frequent event

First prediction model: Decision Tree

```
# Model
ls_predict_M1 <- rpart(classe ~ ., data=ls_subtraining, method="class")

# Predict:
ls_prediction1 <- predict(ls_predict_M1, ls_subtesting, type = "class")

# Plot our decision tree
rpart.plot(ls_predict_M1, main="Decision Tree", extra="auto", under=TRUE, facl
```




```
# Look at the results on the data set:
confusionMatrix(ls_prediction1, ls_subtesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1244  194   16   95   29
##          B   31  516   60   25   55
##          C   37   93  693  134  106
##          D   55   66   66  491   46
##          E   28   80   20   59  665
##
## Overall Statistics
##
##                Accuracy : 0.7359
##                  95% CI : (0.7234, 0.7482)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6646
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8918   0.5437   0.8105   0.6107   0.7381
## Specificity            0.9048   0.9568   0.9086   0.9432   0.9533
## Pos Pred Value         0.7883   0.7511   0.6519   0.6782   0.7805
## Neg Pred Value         0.9546   0.8973   0.9578   0.9251   0.9418
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2537   0.1052   0.1413   0.1001   0.1356
## Detection Prevalence   0.3218   0.1401   0.2168   0.1476   0.1737
## Balanced Accuracy      0.8983   0.7502   0.8596   0.7769   0.8457
```

Decision Trees algorithm has accuracy of 0.739 (95% CI: (0.727, 0.752))

Second prediction model: Random Forest

```
ls_predict_M2 <- randomForest(classe ~. , data=ls_subtraining, method="class")

# Predicting:
ls_predict2 <- predict(ls_predict_M2, ls_subtesting, type = "class")

# Test results on subTesting data set:
confusionMatrix(ls_predict2, ls_subtesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    5    0    0    0
##          B    1  943    4    0    0
##          C    0    1  851    9    0
##          D    0    0    0  794    0
##          E    0    0    0    1  901
##
## Overall Statistics
##
##                Accuracy : 0.9957
##                  95% CI : (0.9935, 0.9973)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9946
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
```

```
## Sensitivity            0.9993   0.9937   0.9953   0.9876   1.0000
## Specificity            0.9986   0.9987   0.9975   1.0000   0.9998
## Pos Pred Value         0.9964   0.9947   0.9884   1.0000   0.9989
## Neg Pred Value         0.9997   0.9985   0.9990   0.9976   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1923   0.1735   0.1619   0.1837
## Detection Prevalence   0.2853   0.1933   0.1756   0.1619   0.1839
## Balanced Accuracy      0.9989   0.9962   0.9964   0.9938   0.9999
```

Conclusion:

Accuracy for Random Forest model was 0.995 (95% CI: (0.993, 0.997)) as expected better preformed in comparison to Decision Tree.

The random Forest model would be a better logical method to select. We can calculate the out-of-sample error is at 0.5% based on 1 - accuracy.

```
predictQuiz <- predict(ls_predict_M2, ls_testing_data, type="class")
predictQuiz
```

```
##  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
```

```
ls_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    fname = paste0("problem_id_",i,".txt")
    write.table(x[i],file=fname,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}

ls_write_files(predictQuiz)
```