Machine Learning Course Project

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://web.archive.org/web/20161224072740/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

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 :-). also Apply your machine learning algorithm to the 20 test cases available in the test data above and submit your predictions in appropriate format to the Course Project Prediction Quiz for automated grading.

Approach:

Our outcome variable is classe, a factor variable. For this data set, “participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions: - 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)

Two models will be tested using decision tree and random forest. The model with the highest accuracy will be chosen as our final model.

Cross-validation

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

Expected out-of-sample error

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

Our outcome variable “classe” is a factor variable. We split the Training dataset into TrainTrainingSet and TestTrainingSet datasets.

Install packages and load the required libraries

```
library(lattice); library(ggplot2); library(caret); library(randomForest); library(rpart); library(rpart.plot);
```

```
## randomForest 4.6-14
```

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

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

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

```
set.seed(1234)
```

load data

```
trainingset <- read.csv("C:/Users/szhang/Desktop/class materials/machine _learning/pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
testingset <- read.csv("C:/Users/szhang/Desktop/class materials/machine _learning/pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))
dim(trainingset); dim(testingset);
```

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

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

delete columns with all missing values

```
trainingset<-trainingset[,colSums(is.na(trainingset))==0]
testingset<-testingset[,colSums(is.na(testingset))==0]
```

remove some variables that we don’t need user_name, raw_timestamp_part_1, raw_timestamp_part_,2 cvtd_timestamp, new_window, and num_window (columns 1 to 7).

```
trainingset<-trainingset[,-c(1:7)]
testingset<-testingset[,-c(1:7)]
```

check the new dataset:

```
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 cross-validation, the training data set is partionned into 2 sets: subTraining (75%) and subTest (25%). This will be performed using random subsampling without replacement.

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

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

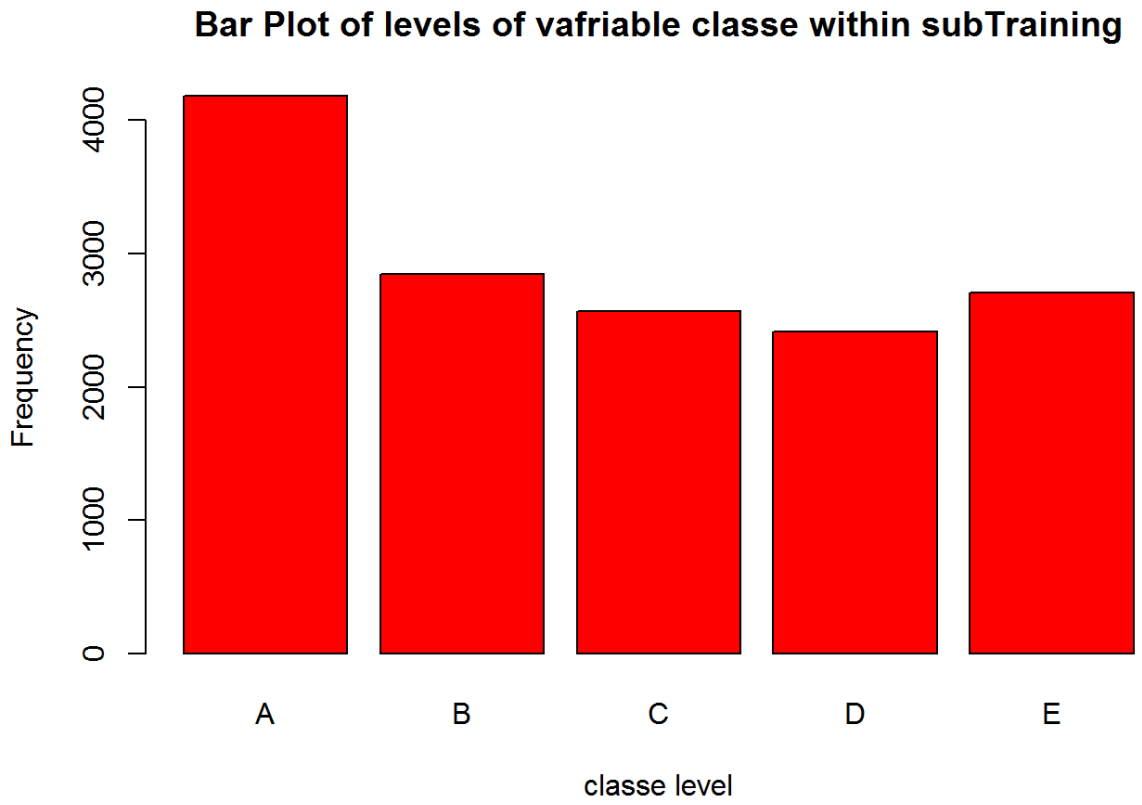
```
dim(subTesting)
```

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

check the Data

The variable “classe” contains 5 levels: A, B, C, D and E. A plot of the outcome variable will allow us to see the frequency of each levels in the subTraining data set and compare one another.

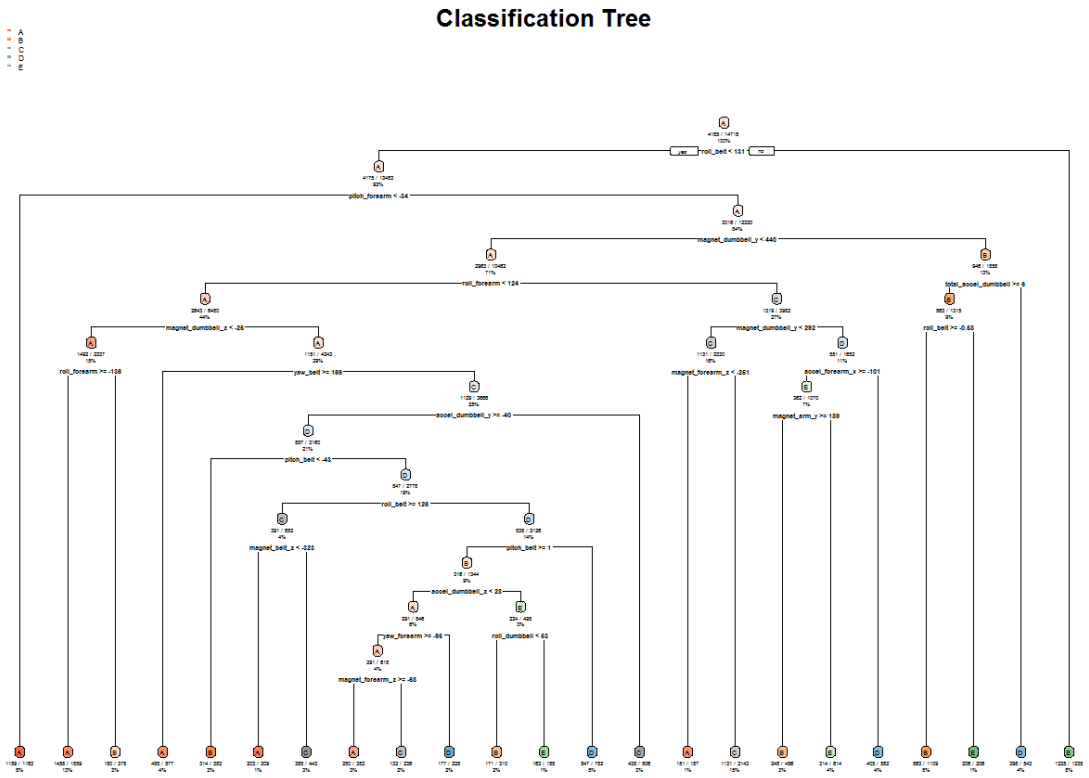
```
plot(subTraining$classe, col="red",main="Bar Plot of levels of vafriable classe within subTraini
ng", xlab="classe level",ylab="Frequency")
```



From the graph above, we can see that each level frequency is within the same order of magnitude of each other. Level A is the most frequent with more than 4000 occurrences while level D is the least frequent with about 2500 occurrences.

First prediction model: Decision Tree

```
model1<-rpart(classe~.,data=subTraining,method="class")
prediction1<-predict(model1,subTesting,type="class")
rpart.plot(model1,main="Classification Tree",extra=102, under=TRUE,faclen=0)
```



test results on our 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
```

Second prediction model: Random Forest

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    3    0    0    0
##          B    0  943   10    0    0
##          C    0    3  844    5    0
##          D    0    0    1  799    0
##          E    0    0    0    0  901
##
## Overall Statistics
##
##                Accuracy : 0.9955
##                  95% CI : (0.9932, 0.9972)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9943
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9937   0.9871   0.9938   1.0000
## Specificity            0.9991   0.9975   0.9980   0.9998   1.0000
## Pos Pred Value         0.9979   0.9895   0.9906   0.9988   1.0000
## Neg Pred Value         1.0000   0.9985   0.9973   0.9988   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1923   0.1721   0.1629   0.1837
## Detection Prevalence   0.2851   0.1943   0.1737   0.1631   0.1837
## Balanced Accuracy      0.9996   0.9956   0.9926   0.9968   1.0000
```

Conclusion

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 missclassified.

submission

predict outcome levels on the original Testing data set using Random Forest algorithm

```
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

```
write_file=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 = FALSE)

  }
}
write_file(predictfinal)
```