Practical Machine Learning Final 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 (A,B,C,D,E) - (Class A) exactly according to the specification , (Class B) throwing the elbows to the front , (Class C) lifting the dumbbell only halfway, (Class D) lowering the dumbbell only halfway and (Class E)throwing the hips to the front .

My goal of this project is to predict the manner in which they did the exercise, describe how I built the model and cross validate, and how I select the best model for prediction. Last, I will use the model I select to predict the test data in the project.

Data Clearning

There is 19,622 observations 160 variables in the raw training dataset (download from here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv).Since there are many variables that have missing values (N/A), we need to clean up the raw dataset first.

Load packages

```
library(Hmisc)
library(caret)
library(randomForest)
library(foreach)
library(rattle)
set.seed(62339)
options(warn=-1)
```

Reading Data into R

```
traindata_raw<-read.csv('pml-
training.csv',header=TRUE,sep=",")
questiondata_raw<-read.csv('pml-
testing.csv',header=TRUE,sep=",")
dim(traindata_raw)
dim(questiondata_raw)

for(i in c(8:ncol(traindata_raw)-1)) {traindata_raw[,i]
= as.numeric(as.character(traindata_raw[,i]))}
for(i in c(8:ncol(questiondata_raw)-1))
{questiondata_raw[,i] =
as.numeric(as.character(questiondata_raw[,i]))}

head(traindata_raw)
head(is.na(traindata_raw))

keep_set1 <-
colnames(traindata_raw[colSums(is.na(traindata_raw)) ==
0])[-(1:7)]
traindata <- traindata_raw[keep_set1]
keep_set1
```

```
keep_set2 <-
colnames(questiondata_raw[colSums(is.na(questiondata_ra
w)) == 0])[-(1:7)]
questiondata <- questiondata_raw[keep_set2]
keep_set2

dim(traindata)
dim(questiondata)
```

```
> keep_set1
 [1] "roll_belt"            "pitch_belt"           "yaw_belt"             "total_accel_belt"
 [5] "gyros_belt_x"         "gyros_belt_y"         "gyros_belt_z"
"accel_belt_x"
 [9] "accel_belt_y"         "accel_belt_z"         "magnet_belt_x"
"magnet_belt_y"
[13] "magnet_belt_z"        "roll_arm"             "pitch_arm"            "yaw_arm"
[17] "total_accel_arm"      "gyros_arm_x"          "gyros_arm_y"
"gyros_arm_z"
[21] "accel_arm_x"          "accel_arm_y"          "accel_arm_z"
"magnet_arm_x"
[25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
"pitch_dumbbell"
[29] "yaw_dumbbell"         "total_accel_dumbbell" "gyros_dumbbell_x"
"gyros_dumbbell_y"
[33] "gyros_dumbbell_z"     "accel_dumbbell_x"     "accel_dumbbell_y"
"accel_dumbbell_z"
[37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
"roll_forearm"
```

[41] "pitch_forearm" "yaw_forearm" "total_accel_forearm" "gyros_forearm_x"
[45] "gyros_forearm_y" "gyros_forearm_z" "accel_forearm_x" "accel_forearm_y"
[49] "accel_forearm_z" "magnet_forearm_x" "magnet_forearm_y" "magnet_forearm_z"
[53] "classe"

53 variables are remaining at the end, for both raw training dataset and raw question data set (20 observations), which we are asked to make prediction (download from here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

```
> dim(traindata)
```
[1] 19622 53
```
> dim(questiondata)
```
[1] 20 53

Machine Learning

Data Partitioning, we splitting original *traindata* set into training and testing datasets, the testing dataset would be used to cross validation.

```
>inTrain <- createDataPartition(traindata$classe, p=0.75, list=FALSE)
>training_set <- traindata[inTrain,]
>testing_set <- traindata[-inTrain,]
```

I will starting with a decision tree model

```
>rpmodel<- train(classe ~ ., method="rpart",
```

```
data=training_set)
>rpmodel$finalMode
>fancyRpartPlot(rpmodel$finalModel, sub='')     ######
plotting the decision tree (Plot 1 in Appendix)
```

Using testing_set data to predict “classe”, this is cross validation step to see how well the model is from the testing data we split from the original training dataset.

```
>rpcv <- predict(rpmodel, newdata=testing_set)
> confusionMatrix(rpcv, testing_set$classe)
```

Confusion Matrix and Statistics

	Reference				
Prediction	A	B	C	D	E
A	1275	415	372	339	134
B	20	305	24	150	112
C	99	229	459	315	233
D	0	0	0	0	0
E	1	0	0	0	422

Overall Statistics

Accuracy : 0.5018
95% CI : (0.4877, 0.5159)
No Information Rate : 0.2845
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.3493

Mcnemar's Test P-Value : NA

Statistics by Class:

	Class: A	Class: B	Class: C	Class: D	Class: E
Sensitivity	0.9140	0.32139	0.5368	0.0000	0.46837
Specificity	0.6409	0.92263	0.7837	1.0000	0.99975
Pos Pred Value	0.5030	0.49918	0.3438	NaN	0.99764
Neg Pred Value	0.9493	0.84999	0.8890	0.8361	0.89310
Prevalence	0.2845	0.19352	0.1743	0.1639	0.18373
Detection Rate	0.2600	0.06219	0.0936	0.0000	0.08605
Detection Prevalence	0.5169	0.12459	0.2722	0.0000	0.08626
Balanced Accuracy	0.7775	0.62201	0.6602	0.5000	0.73406

Only 50.2% of observations are predicted correctly in the testing dataset, the estimated out of sample error with the cross validation dataset for this model is 49.8%, which is very high, we would try a different model - random forest.

```
> rfcontrol <- trainControl(method="cv", 5)
> rfmodel <- train(classe ~ ., data=training_set,
method="rf", trControl=rfcontrol, ntree=250)
```

Using *testing_set* data to validate the model *rfmodel*

```
> predictRF <- predict(rfmodel, testing_set)
> confusionMatrix(testing_set$classe, predictRF)
```

Confusion Matrix and Statistics

Reference

```
Prediction    A    B    C    D    E
         A 1393    1    0    0    1
         B    1  946    2    0    0
         C    0    1  854    0    0
         D    0    0   21  782    1
         E    0    0    0    1  900

Overall Statistics

               Accuracy : 0.9941
                 95% CI : (0.9915, 0.996)
    No Information Rate : 0.2843
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9925
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9993   0.9979   0.9738   0.9987   0.9978
Specificity            0.9994   0.9992   0.9998   0.9947   0.9998
Pos Pred Value         0.9986   0.9968   0.9988   0.9726   0.9989
Neg Pred Value         0.9997   0.9995   0.9943   0.9998   0.9995
Prevalence             0.2843   0.1933   0.1788   0.1597   0.1839
Detection Rate         0.2841   0.1929   0.1741   0.1595   0.1835
Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
Balanced Accuracy      0.9994   0.9986   0.9868   0.9967   0.9988
```

We have 99.4% accuracy of this model, thus the out-of-sample error with the cross validation dataset for this model is 0.6%, which is a great result. I think we can use this model to predict the original testing set for this project.

Making Test Set Predictions

To answer the question of this project, we need to use the final model we choose to predict the manner in which they did the exercise for the original testing dataset.

```
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=FA
LSE,col.names=FALSE)
  }
}

x<-questiondata
result <- predict(rfmodel, x[, -length(names(x))])
result
```

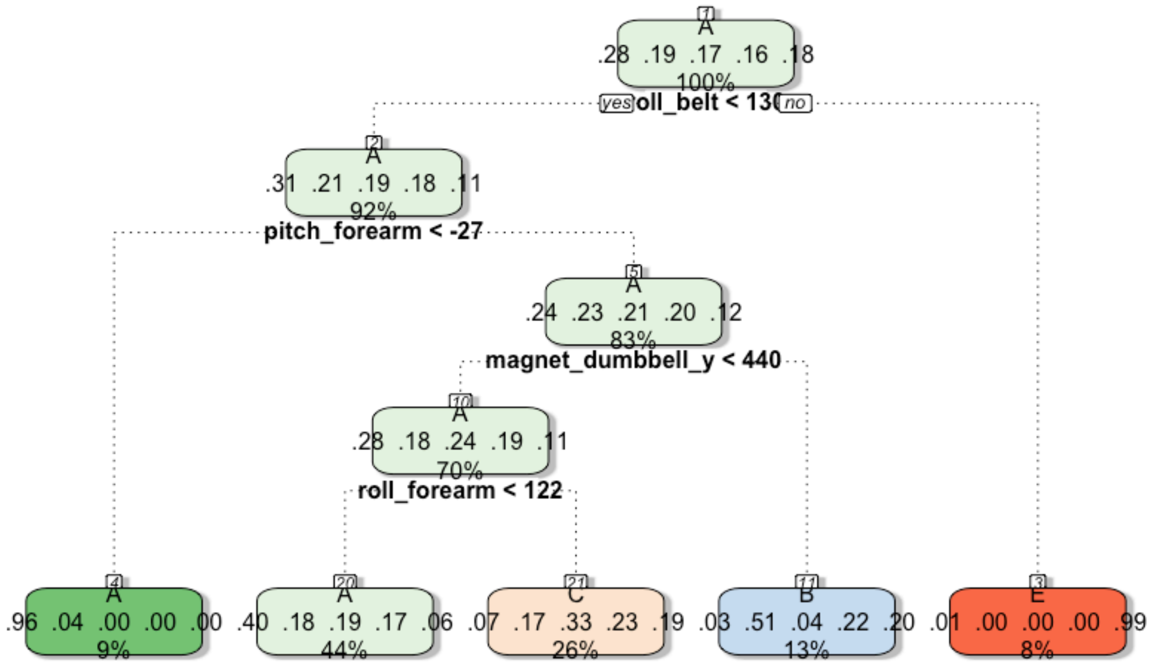
> result

[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

Appendix:

plot 1



1
A
.28 .19 .17 .16 .18
100%
yes
roll_belt < 130
no
2
A
.31 .21 .19 .18 .11
92%
pitch_forearm < -27
5
A
.24 .23 .21 .20 .12
83%
magnet_dumbbell_y < 440
10
A
.28 .18 .24 .19 .11
70%
roll_forearm < 122
4
A
.96 .04 .00 .00 .00
9%
20
A
.40 .18 .19 .17 .06
44%
21
C
.07 .17 .33 .23 .19
26%
11
B
.03 .51 .04 .22 .20
13%
3
E
.01 .00 .00 .00 .99
8%