PracticalMachineLearningFinalProject

Wanhao Chi

October 13, 2016

Data Process

1. Read the datasets and do data cleaning

```
### read data
training <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"),na.strings = c("NA","#div/0!",""))
testing <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"),na.strings = c("NA","#div/0!",""))
### take a first look at the data
# dimersions of the dataset
dim(training);dim(testing)
```

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

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

```
#list types for each attribute
#sapply(training,class)
# how many types in total
#table(sapply(training,class));table(sapply(testing,class))
# take a look at the first 2 rows of the data
#head(training, n = 2);head(testing, n =2)
# summarize attribute distributions
#summary(training); summary(testing)
### data cleaning
# remove the first 7 columns that have nothing with the predictions
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
# remove the columns that contain missing values
training <- training[,colSums(is.na(training)) == 0]
testing <- testing[,colSums(is.na(testing)) == 0]

# take a look at the data again
# dimersions of the dataset
dim(training);dim(testing)
```

```
## [1] 19622    53
```

```
## [1] 20 53
```

2. Create a validation dataset from the Training dataset

```
set.seed(123)
library(caret)
```

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

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

```
validation_index <- createDataPartition(training$classe, p =0.8, list = FALSE)
validation <- training[-validation_index,]
training <- training[validation_index,]
```

Model selection

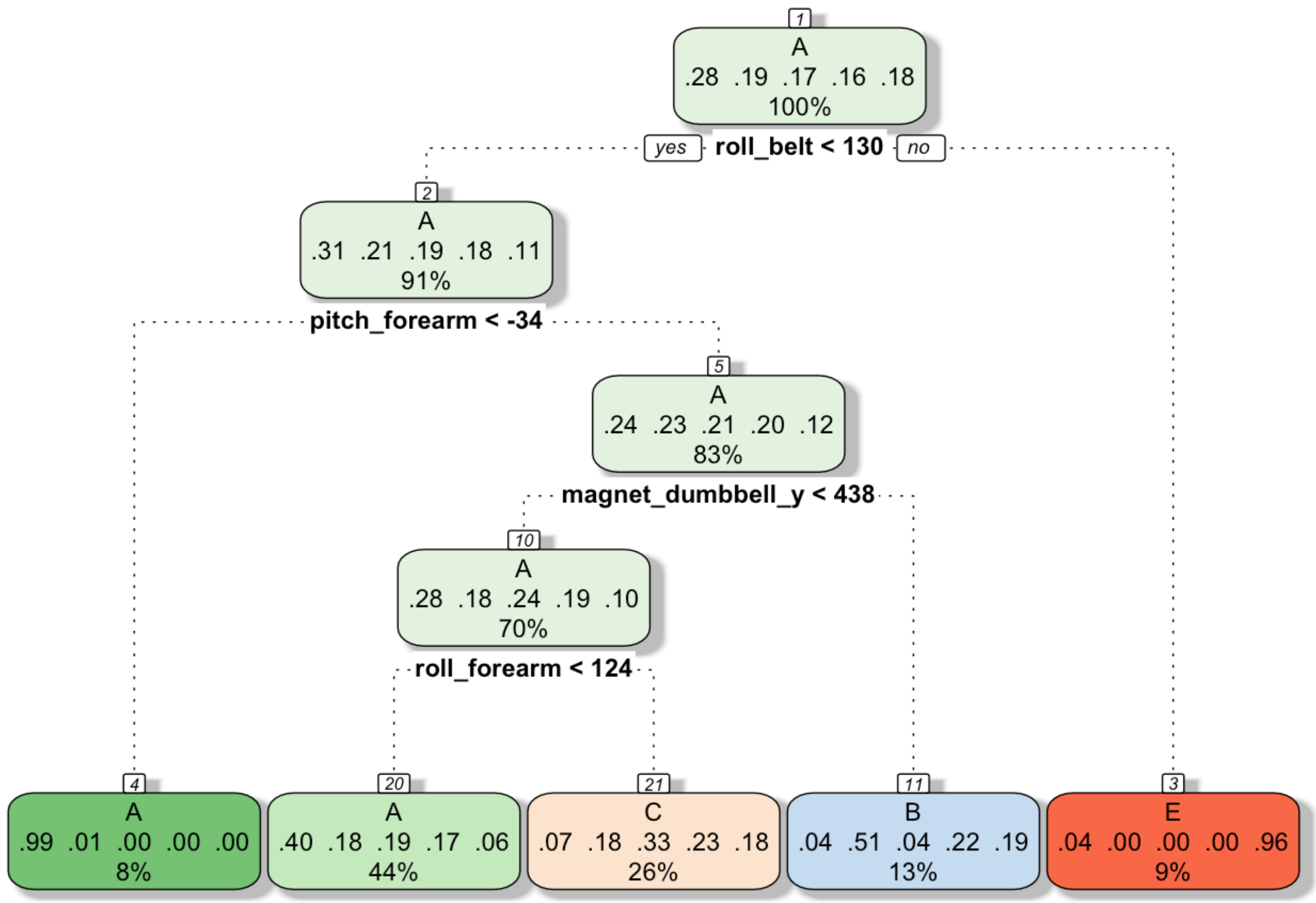
Since the outcome is categorical, we will use classification trees and random forests to predict it.

1. Classification trees

```
set.seed(125)
ctrl <- trainControl(method = "cv", number = 10) # 10 fold cross-validation is used i
n general
modFit_rpart <- train(classe ~ ., data= training, method = "rpart",trControl = ctrl,n
a.action = na.omit)
print(modFit_rpart,digits = 3)
```

```
## CART
##
## 15699 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 14129, 14127, 14129, 14130, 14129, 14130, ...
## Resampling results across tuning parameters:
##
##   cp      Accuracy  Kappa
##   0.0368  0.502     0.3489
##   0.0601  0.414     0.2063
##   0.1170  0.332     0.0734
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.0368.
```

```
library("rattle")
fancyRpartPlot(modFit_rpart$finalModel)
```



Rattle 2016-Oct-13 20:58:31 wanhaochi

```
# predict outcomes using validation dataset
predict_rpart <- predict(modFit_rpart,validation)
confusion_rpart <- confusionMatrix(validation$classe, predict_rpart)
confusion_rpart
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 987  14  93   0  22
##          B 312 271 176   0   0
##          C 317  29 338   0   0
##          D 296 125 222   0   0
##          E 108  80 200   0 333
##
## Overall Statistics
##
##                Accuracy : 0.4917
##                  95% CI : (0.476, 0.5075)
##     No Information Rate : 0.5149
##     P-Value [Acc > NIR] : 0.9983
##
##                   Kappa : 0.3361
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.4886  0.52216  0.32847       NA  0.93803
## Specificity            0.9322  0.85664  0.88044   0.8361  0.89126
## Pos Pred Value         0.8844  0.35705  0.49415       NA  0.46186
## Neg Pred Value         0.6320  0.92162  0.78666       NA  0.99313
## Prevalence             0.5149  0.13230  0.26230   0.0000  0.09049
## Detection Rate         0.2516  0.06908  0.08616   0.0000  0.08488
## Detection Prevalence   0.2845  0.19347  0.17436   0.1639  0.18379
## Balanced Accuracy      0.7104  0.68940  0.60446       NA  0.91464
```

```
accuracy_rpart <- confusion_rpart$overall[1]
accuracy_rpart
```

```
##  Accuracy
## 0.4917155
```

Therefore, the accuracy using classification tree is 0.492. The out-of-sample error rate is 0.508.

2. random forest

```
set.seed(124)
#library("randomForest")
ctrl <- trainControl(method = "cv", number = 5) # 5 fold cross-validation is used
modFit_rf <- train(classe ~ ., data= training, method = "rf",trControl = ctrl,na.acti
on = na.omit)
#modFit_rf <- randomForest(classe ~ ., data = training)
print(modFit_rf, digits = 3)
```

```
## Random Forest
##
## 15699 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 12558, 12560, 12562, 12558, 12558
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    0.991     0.989
##   27    0.992     0.990
##   52    0.985     0.981
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
# predict outcomes using validation dataset
predict_rf <- predict(modFit_rf, validation)
confusion_rf <- confusionMatrix(validation$classe, predict_rf)
confusion_rf
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    1    0    0    0
##          B    0  758    1    0    0
##          C    0    1  683    0    0
##          D    0    0    7  636    0
##          E    0    0    0    0  721
## 
## Overall Statistics
## 
##                Accuracy : 0.9975
##                  95% CI : (0.9953, 0.9988)
##     No Information Rate : 0.2842
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9968
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   0.9884   1.0000   1.0000
## Specificity            0.9996   0.9997   0.9997   0.9979   1.0000
## Pos Pred Value         0.9991   0.9987   0.9985   0.9891   1.0000
## Neg Pred Value         1.0000   0.9994   0.9975   1.0000   1.0000
## Prevalence             0.2842   0.1937   0.1761   0.1621   0.1838
## Detection Rate         0.2842   0.1932   0.1741   0.1621   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9998   0.9985   0.9941   0.9989   1.0000
```

```
accuracy_rf <- confusion_rf$overall[1]
accuracy_rf
```

```
##  Accuracy 
## 0.9974509
```

Therefore, the accuracy using random forest is 0.997, which is much better than what we got from classification tree method. The out-of-sample error for random forest method is 0.003. However, the random forest method is computationally inefficient.

Prediction on testing dataset

```
prediction_testing <- predict(modFit_rf,testing)
prediction_testing
```

```
##  [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
```