Prediction Assignment

TM

November 3, 2017

Introduction

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.

Executive Summary

This is a pure prediction task, with no need to create a simple parsimonious model and therefore perfectly suited to machine learning. I applied 2 classification models, a simpler decision tree and then also a random forest, which as expected gave the best prediction, achieving a accuracy of 99.61% (out of sample rate of 0.39%) and we can be 95% confident accuracy sits between 99.4% and 99.77%. Measurement of accuracy is based on taking a 75% / 25% split of the original testing data, training on the 75% and validating against the remaining 25%.

set up packages used

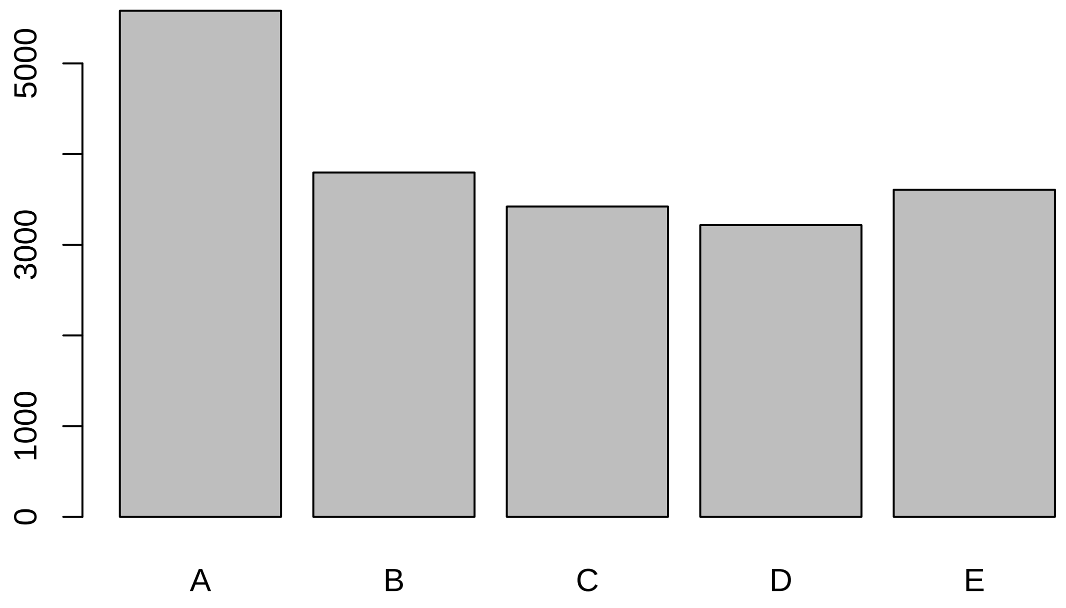
```r
IscaretInstalled <- require("caret")
if(!IscaretInstalled){
    install.packages("caret")
    library("caret")
}

IscaretInstalled <- require("rpart")
if(!IscaretInstalled){
    install.packages("rpart")
    library("rpart")
}

IscaretInstalled <- require("randomForest")
if(!IscaretInstalled){
    install.packages("randomForest")
    library("randomForest")
}
```

Set up seed for reproducability

```r
set.seed(111)
```

import data and prepare data

```r
training_file   <- 'D:/Training/Stats/Machine learning/data/pml-training.csv'
validation_file <- 'D:/Training/Stats/Machine learning/data/pml-testing.csv'

training_url <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'

validation_url  <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

#download.file(training_url, training_file)
#download.file(validation_url, validation_file)

training <- read.csv(training_file, na.strings=c("NA","#DIV/0!", ""))
validation <- read.csv(validation_file, na.strings=c("NA","#DIV/0!", ""))

#explore dependent variable
plot(training$classe)
```



```r
# some variables are all null so will remove
training <- training[, colSums(is.na(training))==0]
validation <- validation[, colSums(is.na(validation))==0]

# Other variables in columns 1-7 intuitively are not needed
training <- training[,-c(1:7)]
validation <- validation[,-c(1:7)]
```

```r
training$classe <- as.factor(training$classe)
```

Cross validation - splitting the training set and using original testing as validation

```r
inTrain <- createDataPartition(y = training$classe, p=0.75, list = FALSE)
training_sub <- training[inTrain,]
testing <- training[-inTrain,]
```

Model fitting

Start with a decision tree

```r
model_dt <- rpart(classe ~ ., method = "class", data = training_sub)
dt_predict <- predict(model_dt, newdata = testing, type = "class")
confusionMatrix(dt_predict,testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1286  163   16   59   36
##          B   39  531   68   62   88
##          C   33  128  693  115   99
##          D   16   82   54  518   43
##          E   21   45   24   50  635
##
## Overall Statistics
##
##                Accuracy : 0.7469
##                  95% CI : (0.7345, 0.7591)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6787
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9219   0.5595   0.8105   0.6443   0.7048
## Specificity            0.9219   0.9350   0.9074   0.9524   0.9650
## Pos Pred Value         0.8244   0.6739   0.6489   0.7265   0.8194
## Neg Pred Value         0.9674   0.8984   0.9578   0.9318   0.9356
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2622   0.1083   0.1413   0.1056   0.1295
## Detection Prevalence   0.3181   0.1607   0.2178   0.1454   0.1580
## Balanced Accuracy      0.9219   0.7473   0.8590   0.7984   0.8349
```

Results show that prediction accuracy is 74.69% for this model, and we can be 95% confident the real value sits between 73.45% and 75.91%. The out of sample error is 23.1%.

Next try a random forest

```
model_rf <- randomForest(classe ~., method = "class", data = training_sub)
rf_predict <- predict(model_rf,newdata = testing)
confusionMatrix(rf_predict,testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    5    0    0    0
##          B    1  943    2    0    0
##          C    0    1  852    7    0
##          D    0    0    1  795    0
##          E    0    0    0    2  901
##
## Overall Statistics
##
##                Accuracy : 0.9961
##                  95% CI : (0.994, 0.9977)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9951
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9937   0.9965   0.9888   1.0000
## Specificity            0.9986   0.9992   0.9980   0.9998   0.9995
## Pos Pred Value         0.9964   0.9968   0.9907   0.9987   0.9978
## Neg Pred Value         0.9997   0.9985   0.9993   0.9978   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1923   0.1737   0.1621   0.1837
## Detection Prevalence   0.2853   0.1929   0.1754   0.1623   0.1841
## Balanced Accuracy      0.9989   0.9965   0.9973   0.9943   0.9998
```

Random forest model generates a prediction accuracy of 99.61% (out of sample rate of 0.39%) and we can be 95% confident accuracy sits between 99.4% and 99.77%.

Conclusion

Random Forest is clearly the better predictor of the 2 models, and provides very strong results accuracy / out of sample error

Submission - appling to Validation data

```
rf_pred_val <- predict(model_rf,newdata = validation, type = "class")
print(rf_pred_val)
```

```
##  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
```