Untitled

Ansh Tiwari

10/21/2020

Precis

Using devices such as Jawwbone Upp, Nikee FuelBandd, & Fitbitt, itt iss now easier to get big data about regular daily activity comparatively cheap. These devices are a part of the appraised self-movement – a group of motivated who periodically take precautions about themselves to make their health, find similarities in their behavior, or because they are techies. People regularly tell a particular activity they do, but they rarely tell how well they do it. In this assignment, our motto is to use data from accelerometers on the belt, forearm, arm, and dumbbell of six volunteers. They were asked to perform barbell lifts correctly and incorrectly in five different types.

The motto of this assignment is to guess the way in which they did the exercise. This is the `classe` variable in the training set.

Abour Data

The resulting variable is `classe`, a factor variable with five layers. For this data set, volunteers were asked to do one set of 10 reiteration of the Unilateral Dumbbell Biceps Curl in five different fashions:

- In accordance to specs (Class A)
- Pulling the elbows in the front (Class B)
- Lifting the dumbbell only midway (Class C)
- Lowering the dumbbell only midway (Class D)
- Pulling the hips in the front (Class E)

Configure

In the beginning configuration is of loading some required packages and advancing some variables.

```
training_ansh251099.file   <- './data/pml-training.csv'
test_ansh251099.cases.file <- './data/pml-testing.csv'
training_ansh251099.url    <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
test_ansh251099.cases.url  <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

if (!file.exists("data")){
  dir.create("data")
}
if (!file.exists("data/submission")){
  dir.create("data/submission")
}

IscaretInstalled_ansh251099 <- require("caret")
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

```
if(!IscaretInstalled_ansh251099){
    install.packages("caret")
    library("caret")
    }
IsrandomForestInstalled_ansh251099 <- require("randomForest")
```

```
## Loading required package: randomForest
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
## 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
```

```
if(!IsrandomForestInstalled_ansh251099){
    install.packages("randomForest")
    library("randomForest")
    }
IsRpartInstalled_ansh251099 <- require("rpart")
```

```
## Loading required package: rpart
```

```
if(!IsRpartInstalled_ansh251099){
    install.packages("rpart")
    library("rpart")
    }
IsRpartPlotInstalled_ansh251099 <- require("rpart.plot")
```

```
## Loading required package: rpart.plot
```

```
## Warning: package 'rpart.plot' was built under R version 3.6.3
```

```
if(!IsRpartPlotInstalled_ansh251099){
    install.packages("rpart.plot")
    library("rpart.plot")
    }

set.seed(9999)
```

Data preprossesing stage

We are removing redundant datas, NA/NULL values, and blank spaces this process is also called data munging/cleaning. Irrespective columns such as `user_name`, `raw_timestamp_part_1`, `raw_timestamp_part_2`, `cvtd_timestamp`, `new_window`, and `num_window` (columns 1 to 7) will be omitted in the subset.

The `pml-training.csv` data is used to devide train and test data sets. The `pml-test.csv` data is used to guess & answer the twenty questions in accordance to trained algorithm.

```
download.file(training_ansh251099.url, training_ansh251099.file)
download.file(test_ansh251099.cases.url,test_ansh251099.cases.file )

training_ansh251099 <-read.csv(training_ansh251099.file, na.strings=c("NA","#DIV/0!", ""))
testing_ansh251099 <-read.csv(test_ansh251099.cases.file , na.strings=c("NA", "#DIV/0!", ""))
training_ansh251099<-training_ansh251099[,colSums(is.na(training_ansh251099)) == 0]
testing_ansh251099 <-testing_ansh251099[,colSums(is.na(testing_ansh251099)) == 0]

training_ansh251099   <-training_ansh251099[,-c(1:7)]
testing_ansh251099 <-testing_ansh251099[,-c(1:7)]
```

Cross-validating stage

Here cross-validating will be done by keeping the training data in training (75%) and testing (25%) data.

```
subSamples_ansh251099 <- createDataPartition(y=training_ansh251099$classe, p=0.75, list=FALSE)
subTraining_ansh251099 <- training_ansh251099[subSamples_ansh251099, ]
subTesting_ansh251099 <- training_ansh251099[-subSamples_ansh251099, ]
```

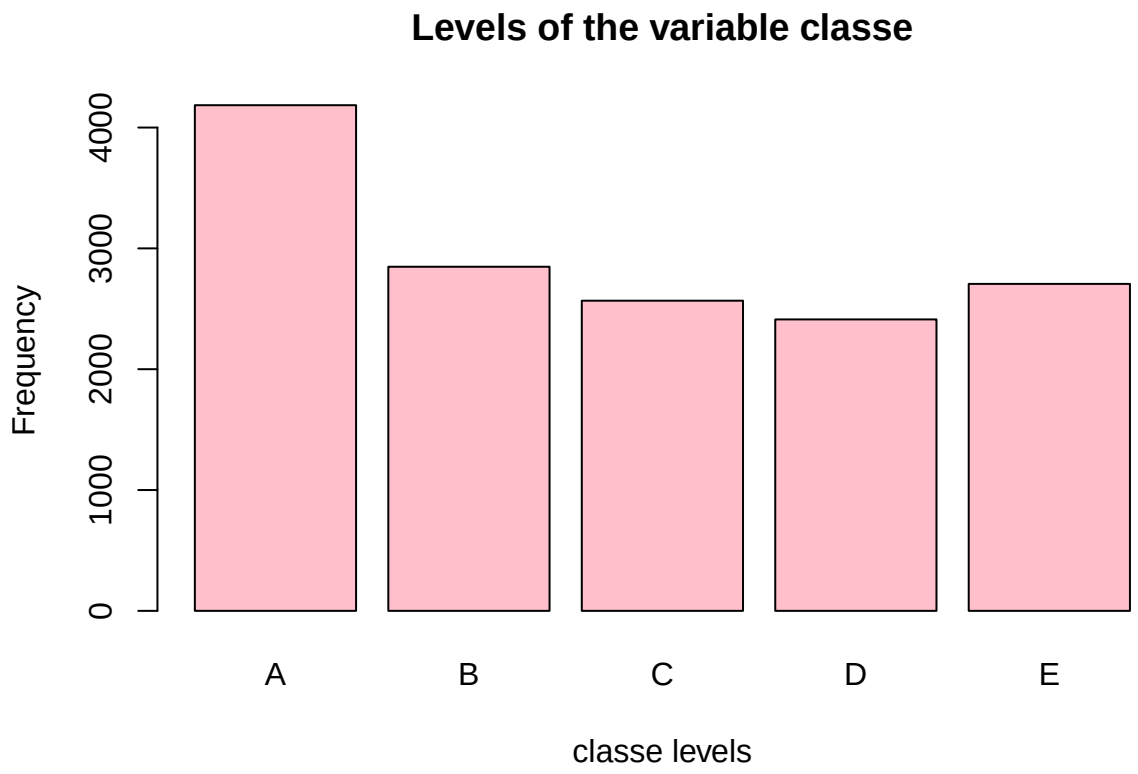
Expected out_of_sample errror

The expected out-of-sample errror will corrrespond to the quantity: 1-accuracyy in the crosss-vallidation data, Accuracy is the directly proportional of right classified observations over the tottal sample in thee subTesting dataset. Expected acccuracy is the expected accuracy in the out_of_sample dataset (i.e. original test data set). Thus, the expected value of the out_of_sample error will corrrespond to the predicted numberr of missclassified total observations or observation in the test data set, which is the quantity: 1-accuracy found from the cross-validation data set.

Performing EDA

The variable `classe` contains five layers. The plot of the result variable shows the frequency of each layers in the subTraining data.

```
plot(subTraining_ansh251099$classe, col="pink", main="Levels of the variable classe", xlab="classe level
```



The plot above shows that Level A is the most frequent classe. D appears to be the least frequent one.

Predicted models

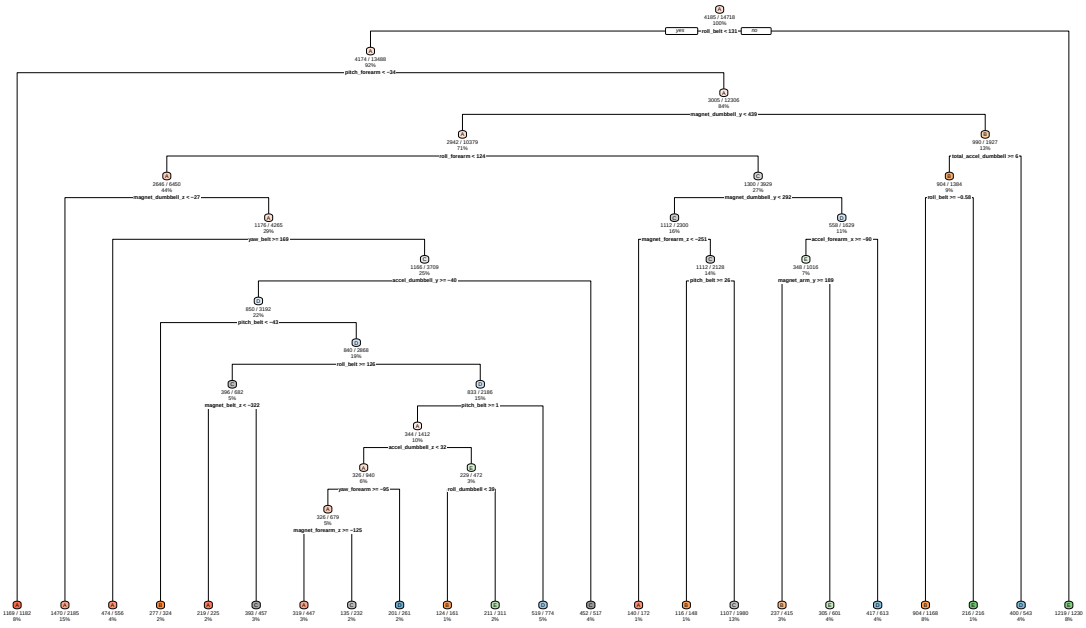
In this section a random forest & decision tree will be apply to the dataset.

Decision Tree

```
modFitDT_ansh251099 <- rpart(classe ~ ., data=subTraining_ansh251099, method="class")

predictDT_ansh251099 <- predict(modFitDT_ansh251099, subTesting_ansh251099, type = "class")

rpart.plot(modFitDT_ansh251099, main="Classification Tree", extra=102, under=TRUE, faclen=0)
```



Following confusion matrix shows the errors of the prediction algorithm.

```
confusionMatrix(predictDT_ansh251099, subTesting_ansh251099$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1247  212   23   83   30
##          B   32  530   73   23   73
##          C   35   96  695  112  121
##          D   60   66   46  532   46
##          E   21   45   18   54  631
##
## Overall Statistics
##
##                Accuracy : 0.7412
##                  95% CI : (0.7287, 0.7534)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6712
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8939   0.5585   0.8129   0.6617   0.7003
## Specificity            0.9008   0.9492   0.9101   0.9468   0.9655
## Pos Pred Value         0.7818   0.7250   0.6563   0.7093   0.8205
## Neg Pred Value         0.9553   0.8996   0.9584   0.9345   0.9347
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2543   0.1081   0.1417   0.1085   0.1287
## Detection Prevalence   0.3252   0.1491   0.2159   0.1529   0.1568
## Balanced Accuracy      0.8974   0.7538   0.8615   0.8043   0.8329
```

Random forest

```
modFitRF_ansh251099 <- randomForest(classe ~ ., data=subTraining_ansh251099, method="class")

predictRF_ansh251099 <- predict(modFitRF_ansh251099, subTesting_ansh251099, type = "class")
```

Following confusion matrix shows the errors of the prediction model.

```
confusionMatrix(predictRF_ansh251099, subTesting_ansh251099$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    5    0    0    0
##          B    2  943    2    0    0
##          C    0    1  853    8    0
##          D    0    0    0  795    2
##          E    0    0    0    1  899
##
## 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            0.9986   0.9937   0.9977   0.9888   0.9978
## Specificity            0.9986   0.9990   0.9978   0.9995   0.9998
## Pos Pred Value         0.9964   0.9958   0.9896   0.9975   0.9989
## Neg Pred Value         0.9994   0.9985   0.9995   0.9978   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2841   0.1923   0.1739   0.1621   0.1833
## Detection Prevalence   0.2851   0.1931   0.1758   0.1625   0.1835
## Balanced Accuracy      0.9986   0.9963   0.9977   0.9942   0.9988
```

Conclusion

Result

The confusionn matrices shows dat d Random Forest algorithm performs bettter thann deccision trees. Thhe acccuracy for the Random Forest model was 0.995 (95% CI: (0.993, 0.997)) compared to 0.739 (95% CI: (0.727, 0.752)) for Deccision Tree model. The random Forest model is selected.

Expected out-of-sample error

Thee expectedd out_of_samplee errorr is esstimated @ 0.005 or else 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 misclassifiedd.

Submission

In thiss sectionn, the projectt submissionn files r generatedd usingg the random forest algorithms on the testing datasets.

```
predictSubmission_ansh251099 <- predict(modFitRF_ansh251099, testing_ansh251099, type="class")
predictSubmission_ansh251099
```

```
##  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
```

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("./data/submission/problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
pml_write_files(predictSubmission_ansh251099)
```