Practical Machine Learning Project

Predict the “classe” in which 20 test subject did an exercise, correctly (“A”), or 4 incorrect methods (“B” to “E”) To create the prediction model, cleaned up data provided (“make tidy”), split the training data into data for the model (70%) and cross-validation data (30%), then used RandomForest algorithm on the training data set.

Prediction of model on portiion of training data (not used in generating the model) had Accuracy : 0.9995 As expected the prediction error on the training data set used to create the model was smaller (no error) than the cross-valiation set.

Prep work

Install libraries

```
require(randomForest)
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
require(caret)
```

```
## Loading required package: caret
## Loading required package: lattice
## Loading required package: ggplot2
```

```
require(ggplot2)
```

Read csv files

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

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

```
dim(testing)
```

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

Make the data tidy, using same approach for testing and training data

Remove columns that that are NA or blank in testing in both files (calculated data by researchers)

Remove index, new_window and cvtd_timestamp columns in both files

Replace ‘problem_id’ with ‘classe’ column in test data - “tidytesting”

```
tidytraining<- subset(training,select =grep("min",invert=TRUE,names(training)))
tidytraining<- subset(tidytraining,select =grep("max",invert=TRUE,names(tidytraining)))
tidytraining<- subset(tidytraining,select =grep("kurtosis",invert=TRUE,names(tidytraining
)))
tidytraining<- subset(tidytraining,select =grep("avg",invert=TRUE,names(tidytraining)))
tidytraining<- subset(tidytraining,select =grep("var",invert=TRUE,names(tidytraining)))
tidytraining<- subset(tidytraining,select =grep("stddev",invert=TRUE,names(tidytraining))
)
tidytraining<- subset(tidytraining,select =grep("skewness",invert=TRUE,names(tidytraining
)))
tidytraining<- subset(tidytraining,select =grep("amplitude",invert=TRUE,names(tidytrainin
g)))

tidytraining<-tidytraining[,-1]
tidytraining<-tidytraining[,-5]
tidytraining<-tidytraining[,-4]

dim(tidytraining)
```

```
## [1] 19622    57
```

```
tidytesting<- subset(testing,select =grep("min",invert=TRUE,names(testing)))
tidytesting<- subset(tidytesting,select =grep("max",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("kurtosis",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("avg",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("var",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("stddev",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("skewness",invert=TRUE,names(tidytesting)))
tidytesting<- subset(tidytesting,select =grep("amplitude",invert=TRUE,names(tidytesting))
)

tidytesting <-tidytesting [,-1]
tidytesting <-tidytesting [,-5]
tidytesting <-tidytesting [,-58]
tidytesting$classe <- factor(c("A", "B", "C", "D", "E"))
tidytesting <-tidytesting [,-4]

dim(tidytesting)
```

```
## [1] 20 57
```

Generating model

Parition training set into subset of training and testing for model checking

```
intrain <-createDataPartition(y=tidytraining$classe,p=0.7,list=FALSE)
subtraining <- tidytraining[intrain,]
subcrossvald <- tidytraining[-intrain,]
```

Use RandomForest algorithm (as also done by the researchers)

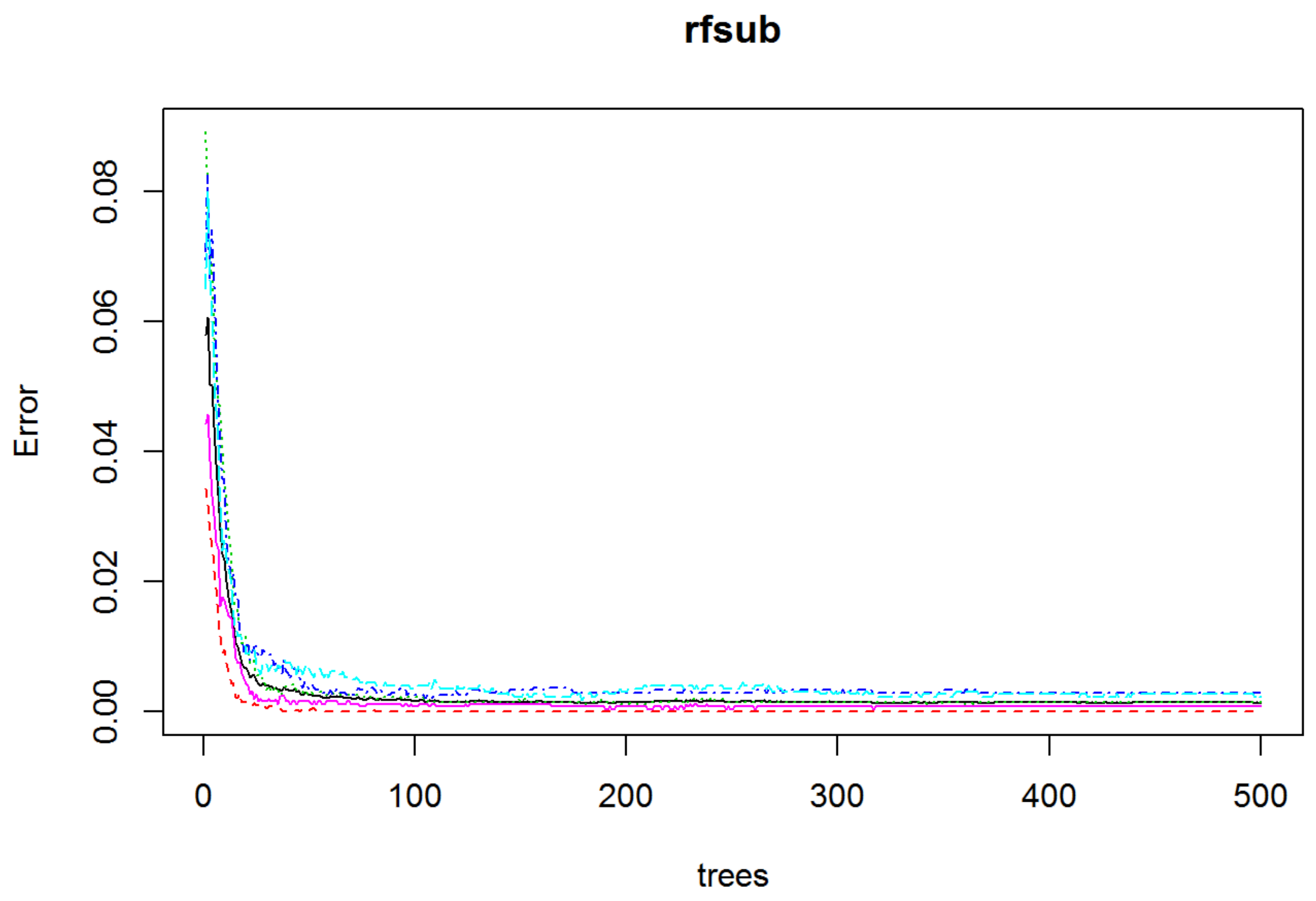
```
rfsub <- randomForest(classe ~.,data=subtraining)
```

View output of model

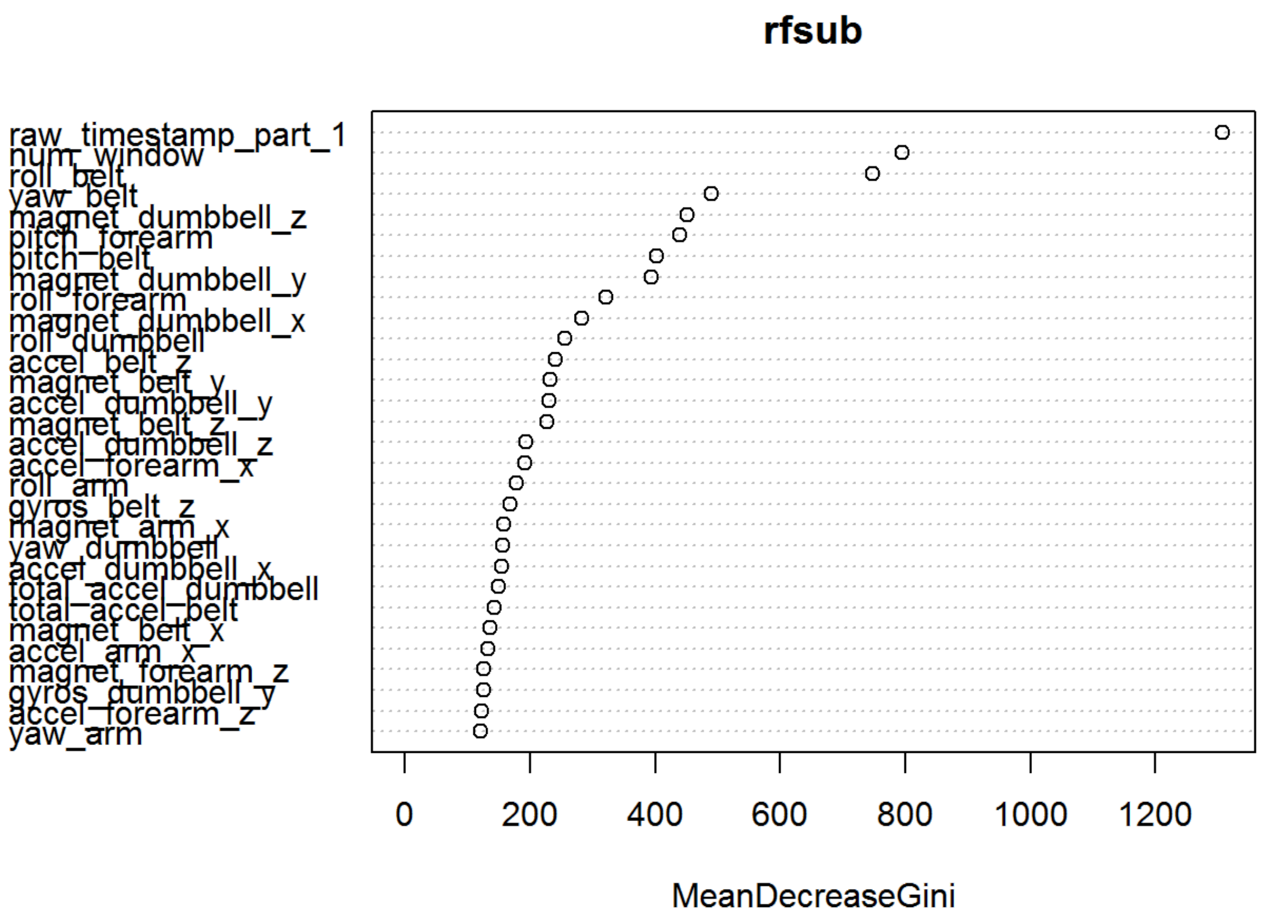
```
rfsub
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = subtraining)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.13%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3906    0    0    0    0 0.0000000000
## B    4 2654    0    0    0 0.0015048909
## C    0    7 2389    0    0 0.0029215359
## D    0    0    5 2247    0 0.0022202487
## E    0    0    0    2 2523 0.0007920792
```

```
plot(rfsub)
```



```
varImpPlot(rfsub)
```



Predict outcomes using model from RandomForest

Predict values on each training data subset

```
predsubtrain <- predict(rfsub,newdata=subtraining)
predcrossvald <- predict(rfsub,newdata=subcrossvald)
```

Compare results of model fit on both against subtraining, and subtesting data

```
confusionMatrix(predsubtrain,subtraining$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
## 
## Overall Statistics
## 
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
confusionMatrix(predcrossvald,subcrossvald$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    0    0    0
##          C    0    0 1026    3    0
##          D    0    0    0  960    1
##          E    0    0    0    1 1081
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9992          
##                  95% CI : (0.998, 0.9997) 
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9989          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   0.9959   0.9991
## Specificity            1.0000   1.0000   0.9994   0.9998   0.9998
## Pos Pred Value         1.0000   1.0000   0.9971   0.9990   0.9991
## Neg Pred Value         1.0000   1.0000   1.0000   0.9992   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1935   0.1743   0.1631   0.1837
## Detection Prevalence   0.2845   0.1935   0.1749   0.1633   0.1839
## Balanced Accuracy      1.0000   1.0000   0.9997   0.9978   0.9994
```

No error on subset of training data used to create model

Very small error on the cross validation data using the model built from the training data

Predict 20 values on tidytesting data

```
pred <- predict(rfsub,newdata=tidytesting)
pred
```

```
##  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
```

generate text files for submission

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

pml_write_files(pred)
```