Practical Machine Learning Writeup

ramgovin

Tuesday, August 18, 2015

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har

Data Cleaning and Preparation

```
set.seed(9)
library(gbm)
library(randomForest)
library(caret)
library(ggplot2)


#Data loading
pmlTrain<-read.csv("pml-training.csv", header=T, na.strings=c("NA",
"#DIV/0!"))
pmlTest<-read.csv("pml-testing.csv", header=T, na.string=c("NA", "#DIV/0!"))
```

Training data was partitioned and preprocessed using the code described below. In brief, all variables with at least one "NA" were excluded from the analysis. Variables related to time and user information were excluded for a total of 51 variables and 19622 class measurements. Same variables were mainteined in the test data set (Validation dataset) to be used for predicting the 20 test cases provided.

```
## NA exclusion for all available variables
noNApmlTrain<-pmlTrain[, apply(pmlTrain, 2, function(x) !any(is.na(x)))]
dim(noNApmlTrain)

## [1] 19622    60

## variables with user information, time and undefined
cleanpmlTrain<-noNApmlTrain[,-c(1:8)]
dim(cleanpmlTrain)

## [1] 19622    52

## 20 test cases provided clean info - Validation data set
cleanpmltest<-pmlTest[,names(cleanpmlTrain[,-52])]
dim(cleanpmltest)

## [1] 20 51
```

Data Partitioning and Prediction Process

The cleaned downloaded data set was subset in order to generate a test set independent from the 20 cases provided set. Partitioning was performed to obtain a 75% training set and a 25% test set.

```
#data cleaning
inTrain<-createDataPartition(y=cleanpmlTrain$classe, p=0.75,list=F)
training<-cleanpmlTrain[inTrain,]
test<-cleanpmlTrain[-inTrain,]
#Training and test set dimensions
dim(training)

## [1] 14718    52

dim(test)

## [1] 4904   52
```

Results

Random forest trees were generated for the training dataset using cross-validation. Then the generated algorithm was examnined under the partitioned training set to examine the accuracy and estimated error of prediction.

By using 51 predictors for five classes using cross-validation at a 5-fold an accuracy of 99.2% with a 95% CI [0.989-0.994] was achieved accompanied by a Kappa value of 0.99.

```
fitControl2<-trainControl(method="cv", number=5, allowParallel=T, verbose=T)
rffit<-train(classe~.,data=training, method="rf", trControl=fitControl2,
verbose=F)

## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=26
## - Fold1: mtry=26
## + Fold1: mtry=51
## - Fold1: mtry=51
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=26
## - Fold2: mtry=26
## + Fold2: mtry=51
## - Fold2: mtry=51
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=26
## - Fold3: mtry=26
## + Fold3: mtry=51
## - Fold3: mtry=51
## + Fold4: mtry= 2
## - Fold4: mtry= 2
## + Fold4: mtry=26
## - Fold4: mtry=26
## + Fold4: mtry=51
## - Fold4: mtry=51
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry=26
## - Fold5: mtry=26
## + Fold5: mtry=51
## - Fold5: mtry=51
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 26 on full training set
```

```
predrf<-predict(rffit, newdata=test)
confusionMatrix(predrf, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    7    0    0    0
##          B    0  940    4    0    0
##          C    0    2  851    3    1
##          D    0    0    0  801    3
##          E    0    0    0    0  897
##
## Overall Statistics
##
##                Accuracy : 0.9959
##                  95% CI : (0.9937, 0.9975)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9948
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9905   0.9953   0.9963   0.9956
## Specificity            0.9980   0.9990   0.9985   0.9993   1.0000
## Pos Pred Value         0.9950   0.9958   0.9930   0.9963   1.0000
## Neg Pred Value         1.0000   0.9977   0.9990   0.9993   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1917   0.1735   0.1633   0.1829
## Detection Prevalence   0.2859   0.1925   0.1748   0.1639   0.1829
## Balanced Accuracy      0.9990   0.9948   0.9969   0.9978   0.9978
```

```
pred20<-predict(rffit, newdata=cleanpmltest)
# Output for the prediction of the 20 cases provided
pred20
```

```
##  [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
```

Alternative Algorithm: A boosting algorithm (gbm) was also tested to confirm and compare predictions. Though data is not shown, but the boosting approach presented a less accuracy (96%) compared to the one mentioned (99.59%) here. However, the predictions for the 20 test cases are matched for both algorithms.

Once, the predictions were obtained for the 20 test cases provided, the below shown script was used to obtain single text files to be uploaded to the courses web site to comply with the submission assigment.

```
#getwd()
#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(pred20)
```