Random Forest

Calvin Seto

January 14, 2016

Random Forest

mtry = number of variables randomly sampled as candidates at each split Defaults: Classification sqrt(p) Regression p/3 where p = number of variables

```
library(caret)

setwd("~/Dropbox/jhudatascience/8_Practical_Machine_Learning/CourseProject")

# setwd("C:/Users/Calvin/Calvinsbiz/Dropbox/jhudatascience/8_Practical_Machine_Learning/CourseProject")

pmlTrain1 <- read.csv("data/pml-training.csv", stringsAsFactors = FALSE, na.strings = c("#DIV/0!","","NA

pmlTrain1MissingCounts <- sapply(pmlTrain1, function(x)sum(is.na(x)))
pmlTrain1Complete <- pmlTrain1MissingCounts[pmlTrain1MissingCounts==0]
pmlTrain2 <- pmlTrain1[,names(pmlTrain1Complete)]

inTrain <- createDataPartition(y=pmlTrain2$classe,
                               p=0.75, list=FALSE)
training <- pmlTrain2[inTrain,]
testing <- pmlTrain2[-inTrain,]

predictors <- training[,8:59]
outcome <- as.factor(training[,60])

# configure parallel
library(parallel)
library(doParallel)
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)


# seed
set.seed(168)

# default is bootstrap
fitRFControl <- trainControl(method="cv",
                              number=10
)

# fitRFGrid <- expand.grid(mtry=
# )

"Start Time "; Sys.time()
```

```
## [1] "Start Time "
## [1] "2016-01-21 10:49:32 EST"
```

```
fitRF <- train(x=predictors,
               y=outcome,
               data=training,
               method="rf",
               trControl=fitRFControl
)
"End Time "; Sys.time()
```

```
## [1] "End Time "
```

```
## [1] "2016-01-21 10:58:33 EST"
```

```
stopCluster(cluster)

# show model summary
fitRF
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 13246, 13246, 13247, 13247, 13246, 13246, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9932058  0.9914047  0.001838816  0.002327350
##   27    0.9930704  0.9912344  0.002390396  0.003024162
##   52    0.9856646  0.9818653  0.003331463  0.004215291
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
# What is this for?
# it seems to show all folds and accuracy, Kappa, and fold number for cross-validation process
fitRF$resample
```

```
##     Accuracy     Kappa Resample
## 1  0.9959239 0.9948449   Fold02
## 2  0.9938859 0.9922663   Fold01
## 3  0.9925221 0.9905388   Fold03
## 4  0.9932065 0.9914037   Fold06
## 5  0.9932065 0.9914061   Fold05
## 6  0.9911625 0.9888195   Fold04
## 7  0.9898167 0.9871136   Fold07
## 8  0.9952413 0.9939809   Fold10
## 9  0.9945652 0.9931242   Fold09
## 10 0.9925272 0.9905490   Fold08
```

```r
# Make predictions and make table
predictions <- predict(fitRF, newdata=testing)

# create confusiion matrix
confusionMatrix(predictions, testing$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    1    0    0    0
##          B    2  948    9    0    0
##          C    0    0  846   11    0
##          D    0    0    0  793    1
##          E    0    0    0    0  900
## 
## Overall Statistics
## 
##                Accuracy : 0.9951
##                  95% CI : (0.9927, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9986   0.9989   0.9895   0.9863   0.9989
## Specificity            0.9997   0.9972   0.9973   0.9998   1.0000
## Pos Pred Value         0.9993   0.9885   0.9872   0.9987   1.0000
## Neg Pred Value         0.9994   0.9997   0.9978   0.9973   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2841   0.1933   0.1725   0.1617   0.1835
## Detection Prevalence   0.2843   0.1956   0.1748   0.1619   0.1835
## Balanced Accuracy      0.9991   0.9981   0.9934   0.9930   0.9994
```

```r
# this creates confusion matrix also
testing$predRight <- predictions==testing$classe
table(predictions,testing$classe)
```

```
## 
## predictions    A    B    C    D    E
##           A 1393    1    0    0    0
##           B    2  948    9    0    0
##           C    0    0  846   11    0
##           D    0    0    0  793    1
##           E    0    0    0    0  900
```