Practical Machine Learning Project

Sid M

Prerequisites

- First, download the training data and test data and have placed them in the same directory as the R files.
- Next, set the working directory to the place you have downloaded it to.
- Ensure you have the following R packages installed: **caret**, **randomForest**

Preprocessing

In this step, we:
1. Read the raw csv from files we downloaded in prerequisites
2. Remove features that have zero variance
3. Remove the descriptive columns [1 - 7]
4. Remove all columns that are NA

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.1.3
```

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

```
loadAndCleanCSV <- function(file) {
  raw <- read.csv(file)
  zeroVCols <- nearZeroVar(raw)
  raw <- raw[, -zeroVCols]
  raw <- raw[-(1:7)]
  naCols <- apply(raw, 2, function(x) { sum(is.na(x)) })
  return(raw[, which(naCols == 0)])
}
trainingData <- loadAndCleanCSV("pml-training.csv")
testData <- loadAndCleanCSV("pml-testing.csv")
```

We then partition the trainingData into a training and cross validation set. We set the seed to a constant number for the purposes of reproducibility.

```
set.seed(1337)
trainingIndices <- createDataPartition(trainingData$classe, p = 0.8, list = FALSE)
trainingSet <- trainingData[trainingIndices, ]
crossValidationSet <- trainingData[-trainingIndices, ]
```

Training the Model

We use the randomForest package for classification and regression.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.1.3
```

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

```
model <- randomForest(classe ~ ., data = trainingSet)
```

Now that our model has been created, we use it to see how well we perform with cross validation.

Cross Validation accuracy (Out of Sample)

```
OOSample <- predict(model, crossValidationSet)
confusionMatrix(OOSample, crossValidationSet$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    2    0    0    0
##          B    0  757    6    0    0
##          C    0    0  678    5    2
##          D    0    0    0  636    1
##          E    0    0    0    2  718
## 
## Overall Statistics
## 
##                Accuracy : 0.9954
##                  95% CI : (0.9928, 0.9973)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9942
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   0.9912   0.9891   0.9958
## Specificity            0.9993   0.9981   0.9978   0.9997   0.9994
## Pos Pred Value         0.9982   0.9921   0.9898   0.9984   0.9972
## Neg Pred Value         1.0000   0.9994   0.9981   0.9979   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1930   0.1728   0.1621   0.1830
## Detection Prevalence   0.2850   0.1945   0.1746   0.1624   0.1835
## Balanced Accuracy      0.9996   0.9977   0.9945   0.9944   0.9976
```

As can be seen, our accuracy with the Cross Validation is: **99.54%**.
Thus, our out of sample error is **0.46%**.
So far, our model proves that it is good, so we now run it on the test data.

Test Set Prediction

```
testPrediction <- predict(model, testData)
testPrediction
```

```
##  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
```

Finally, to conclude this assignment, we turn the prediction to a format that the submission page can accept given the *pml_write_files* function:

```
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(as.vector(testPrediction))
```