Practical Machine Learning project for Coursera course

Jens

Sunday, March 22, 2015

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

Synopsis

In this project the weightlift data set from [1] is used to predict Which excersise the participants did. How the training set is constructed and how the random forest model is trained is described.

Data Processing

The data file was downloaded from https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv and loaded into R with the following command:

```
trainData <- read.csv("pml-training.csv")
```

As there are

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

varibles on the original data set, we will try and reduce this by a proper pre-processing/clearning of the data set. First we take away the near zero values:

```
nsv <- nearZeroVar(trainData, saveMetrics=TRUE)
trainData <- trainData[!nsv$nzv]
```

This us down to

```
length(names(trainData))
```

```
## [1] 100
```

variabels. We now remove variables where most values are NA:

```
nottoNA <- colSums(is.na(trainData))<0.8*nrow(trainData)
trainData <- trainData[nottoNA]
```

This leave us with

```
length(names(trainData))
```

```
## [1] 59
```

variables. (Some of these variables are highly correlated, but due to time constraint I have not looked into removing such variables. Instead PCA pre-processing will be used diretly in the training of the model.)

We split the training set into a training and a test set:

```r
inTrain <- createDataPartition(y=trainData$classe, p=0.60, list=FALSE)
train <- trainData[inTrain, ]
test <- trainData[-inTrain, ]
```

With the this, we will be able to evaluate the out of sample error on this test set.

We now train a random forest model where we use 3-fold cross validation. As cross validation is used directly in the model, we did not split original training set into both a training set, a cross validation set, and a test set, but only a training set and a test set.

```r
modelFit <- train(classe ~ ., method='rf', data=train, preProcess='pca',
                  trControl = trainControl(method = "repeatedcv", number=3, repeats=3),
                  ntree=100)
```

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

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range

## Warning in randomForest.default(x, y, mtry = param$mtry, ...): invalid
## mtry: reset to within valid range
```

Results

With the trained model, we estimate the out of simple error using our created test set:

```
confusionMatrix(test$classe, predict(modelFit, test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2214   12    6    0    0
##          B   23 1484   11    0    0
##          C    0   19 1338   11    0
##          D    0    0   32 1249    5
##          E    0    0    1   12 1429
##
## Overall Statistics
##
##                Accuracy : 0.9832
##                  95% CI : (0.9801, 0.9859)
##     No Information Rate : 0.2851
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9787
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9897   0.9795   0.9640   0.9819   0.9965
```

```
## Specificity            0.9968   0.9946   0.9954   0.9944   0.9980
## Pos Pred Value         0.9919   0.9776   0.9781   0.9712   0.9910
## Neg Pred Value         0.9959   0.9951   0.9923   0.9965   0.9992
## Prevalence             0.2851   0.1931   0.1769   0.1621   0.1828
## Detection Rate         0.2822   0.1891   0.1705   0.1592   0.1821
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9933   0.9871   0.9797   0.9881   0.9972
```

As can be seen the out of sampling error is fairly low (i.e. the accuracy is high).

Using the trained model, we can now finally predict the class of the 20 test cases provided for the project:

```
finaltest <- read.csv("pml-testing.csv")
finaltest <- finaltest[!nsv$nzv]
finaltest <- finaltest[nottoNA]
predictions <- predict(modelFit, finaltest)
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(predictions)
```

While the out of sample error seems low, the model only predicted 18 out 20 cases correct, so there is still room for improvement. However, this will be left for future studies.

References

[1] Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.