Coursera - Practical Machine Learning - Project Write-Up

Alex Erasso Sunday, October 25, 2015

Summary

This machine learning project aims to predict the quality of a specific type of gym exercise based on measurements of acceleration of six individuals. It uses data from accelerometers on the belt, forearm, arm, and dumbell of six participants performing barbell lifts correctly and incorrectly in five different ways. More information is available at: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Below we present the steps we have followed to develop a classifier for the quality of the exercise. This is the “classe” variable in the training set of about 20,000 readings of 160 variables. We have used cross-validation by splitting the training set and using 30% of its records for model validation. Results after pre-processing the training set, finding its principal components, and training a random forest algorithm indicated accuracy of 97% (or out of sample error expected at about 3%).

A final test of the classifier, using 20 records provided by Coursera, was executed with 100% accuracy.

Pre Processing Phase

Reading files with training and testing data sets

```
workdir<-getwd()

fpath1 = file.path(workdir, "pml-training.csv")
if (!file.exists(fpath1)){
    fileUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
    download.file(fileUrl, "./pml-training.csv")
}
training <- read.csv("pml-training.csv", header=TRUE, sep=",")

fpath2 = file.path(workdir, "pml-testing.csv")
if (!file.exists(fpath2)){
    fileUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
    download.file(fileUrl, "./pml-testing.csv")
}
testing <- read.csv("pml-testing.csv", header=TRUE, sep=",")

library(caret)
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
set.seed(3433)
```

Removing columns with NAs or “” in the samples

```
noNATrain <- sapply(training, function(x) any(is.na(x) || x=="" ))

training1.3 <- training[ ,!(noNATrain)]
testing1.3  <- testing[ ,!noNATrain]
```

Removing predictors that have constant values. The data set must be a numeric vector or matrix, or a data frame with all numeric data.

```
nzv <- nearZeroVar(training1.3, saveMetrics=TRUE)
nzv
```

```
##                       freqRatio percentUnique zeroVar   nzv
## X                      1.000000  100.00000000   FALSE FALSE
## user_name              1.100679    0.03057792   FALSE FALSE
## raw_timestamp_part_1   1.000000    4.26562022   FALSE FALSE
## raw_timestamp_part_2   1.000000   85.53154622   FALSE FALSE
## cvtd_timestamp         1.000668    0.10192641   FALSE FALSE
## new_window            47.330049    0.01019264   FALSE  TRUE
## num_window             1.000000    4.37264295   FALSE FALSE
## roll_belt              1.101904    6.77810621   FALSE FALSE
## pitch_belt             1.036082    9.37722964   FALSE FALSE
## yaw_belt               1.058480    9.97349913   FALSE FALSE
## total_accel_belt       1.063160    0.14779329   FALSE FALSE
## gyros_belt_x           1.058651    0.71348486   FALSE FALSE
## gyros_belt_y           1.144000    0.35164611   FALSE FALSE
## gyros_belt_z           1.066214    0.86127816   FALSE FALSE
## accel_belt_x           1.055412    0.83579655   FALSE FALSE
## accel_belt_y           1.113725    0.72877383   FALSE FALSE
## accel_belt_z           1.078767    1.52379982   FALSE FALSE
## magnet_belt_x          1.090141    1.66649679   FALSE FALSE
## magnet_belt_y          1.099688    1.51870350   FALSE FALSE
## magnet_belt_z          1.006369    2.32901845   FALSE FALSE
## roll_arm              52.338462   13.52563449   FALSE FALSE
## pitch_arm             87.256410   15.73234125   FALSE FALSE
## yaw_arm               33.029126   14.65701763   FALSE FALSE
## total_accel_arm        1.024526    0.33635715   FALSE FALSE
## gyros_arm_x            1.015504    3.27693405   FALSE FALSE
## gyros_arm_y            1.454369    1.91621649   FALSE FALSE
## gyros_arm_z            1.110687    1.26388747   FALSE FALSE
## accel_arm_x            1.017341    3.95984099   FALSE FALSE
## accel_arm_y            1.140187    2.73672409   FALSE FALSE
## accel_arm_z            1.128000    4.03628580   FALSE FALSE
## magnet_arm_x           1.000000    6.82397309   FALSE FALSE
## magnet_arm_y           1.056818    4.44399144   FALSE FALSE
## magnet_arm_z           1.036364    6.44684538   FALSE FALSE
## roll_dumbbell          1.022388   84.20650290   FALSE FALSE
## pitch_dumbbell         2.277372   81.74498012   FALSE FALSE
## yaw_dumbbell           1.132231   83.48282540   FALSE FALSE
## total_accel_dumbbell   1.072634    0.21914178   FALSE FALSE
## gyros_dumbbell_x       1.003268    1.22821323   FALSE FALSE
## gyros_dumbbell_y       1.264957    1.41677709   FALSE FALSE
## gyros_dumbbell_z       1.060100    1.04984201   FALSE FALSE
## accel_dumbbell_x       1.018018    2.16593619   FALSE FALSE
## accel_dumbbell_y       1.053061    2.37488533   FALSE FALSE
## accel_dumbbell_z       1.133333    2.08949139   FALSE FALSE
## magnet_dumbbell_x      1.098266    5.74864948   FALSE FALSE
## magnet_dumbbell_y      1.197740    4.30129447   FALSE FALSE
## magnet_dumbbell_z      1.020833    3.44511263   FALSE FALSE
## roll_forearm          11.589286   11.08959331   FALSE FALSE
## pitch_forearm         65.983051   14.85577413   FALSE FALSE
## yaw_forearm           15.322835   10.14677403   FALSE FALSE
## total_accel_forearm    1.128928    0.35674243   FALSE FALSE
## gyros_forearm_x        1.059273    1.51870350   FALSE FALSE
## gyros_forearm_y        1.036554    3.77637346   FALSE FALSE
## gyros_forearm_z        1.122917    1.56457038   FALSE FALSE
## accel_forearm_x        1.126437    4.04647844   FALSE FALSE
## accel_forearm_y        1.059406    5.11160942   FALSE FALSE
## accel_forearm_z        1.006250    2.95586586   FALSE FALSE
## magnet_forearm_x       1.012346    7.76679238   FALSE FALSE
## magnet_forearm_y       1.246914    9.54031189   FALSE FALSE
## magnet_forearm_z       1.000000    8.57710733   FALSE FALSE
## classe                 1.469581    0.02548160   FALSE FALSE
```

```
nzv <- nearZeroVar(training1.3, saveMetrics=FALSE)
nzv
```

```
## [1] 6
```

```
training2 <-training1.3[,-nzv]
testing2  <-testing1.3[,-nzv]
```

Splitting the training set into a training and a cross-validation subsets

```
inTrain <- createDataPartition(y=training2$classe, p =0.7, list=FALSE)
trainSet <- training2[inTrain, ]
crossVal <- training2[-inTrain, ]
```

Finding principal components for the training subset to reduce the number of predictors. Notice that predictors with large number of NA were excluded since they would obtain variance. Also individuals' names and outcome were excluded.

```
preProc <- preProcess(trainSet[ , 2:58], method=c("center", "scale", "pca"), thresh = 0.80)
preProc
```

```
## Created from 13737 samples and 57 variables
##
## Pre-processing:
##   - centered (55)
##   - principal component signal extraction (55)
##   - scaled (55)
##
## PCA needed 13 components to capture 80 percent of the variance
```

```
preProc$std
```

```
## raw_timestamp_part_1 raw_timestamp_part_2          num_window
##         2.049173e+05         2.885593e+05        2.481511e+02
##            roll_belt           pitch_belt            yaw_belt
##         6.277123e+01         2.234521e+01        9.521789e+01
##     total_accel_belt         gyros_belt_x         gyros_belt_y
##         7.735089e+00         2.058961e-01         7.836158e-02
##         gyros_belt_z         accel_belt_x         accel_belt_y
##         2.424817e-01         2.971358e+01         2.856659e+01
##         accel_belt_z        magnet_belt_x        magnet_belt_y
##         1.004529e+02         6.423863e+01         3.600123e+01
##        magnet_belt_z             roll_arm            pitch_arm
##         6.557215e+01         7.288283e+01         3.057300e+01
##              yaw_arm      total_accel_arm          gyros_arm_x
##         7.155175e+01         1.052034e+01         1.993867e+00
##          gyros_arm_y          gyros_arm_z          accel_arm_x
##         8.483135e-01         5.535272e-01         1.814657e+02
##          accel_arm_y          accel_arm_z         magnet_arm_x
##         1.101311e+02         1.350458e+02         4.426197e+02
##         magnet_arm_y         magnet_arm_z        roll_dumbbell
##         2.021676e+02         3.282373e+02         7.008627e+01
##       pitch_dumbbell         yaw_dumbbell total_accel_dumbbell
##         3.700222e+01         8.248114e+01         1.021985e+01
##     gyros_dumbbell_x     gyros_dumbbell_y     gyros_dumbbell_z
##         1.784288e+00         6.547376e-01         2.724547e+00
##     accel_dumbbell_x     accel_dumbbell_y     accel_dumbbell_z
##         6.730590e+01         8.091674e+01         1.095413e+02
##    magnet_dumbbell_x    magnet_dumbbell_y    magnet_dumbbell_z
##         3.395963e+02         3.259586e+02         1.401035e+02
##         roll_forearm        pitch_forearm          yaw_forearm
##         1.080868e+02         2.813158e+01         1.031690e+02
##  total_accel_forearm      gyros_forearm_x      gyros_forearm_y
##         1.007936e+01         6.589580e-01         3.425306e+00
##      gyros_forearm_z      accel_forearm_x      accel_forearm_y
##         2.061887e+00         1.803379e+02         2.001915e+02
##      accel_forearm_z     magnet_forearm_x     magnet_forearm_y
##         1.385057e+02         3.462015e+02         5.081644e+02
##     magnet_forearm_z
##         3.723551e+02
```

```
preProc$thresh
```

```
## [1] 0.8
```

```
preProc$numComp
```

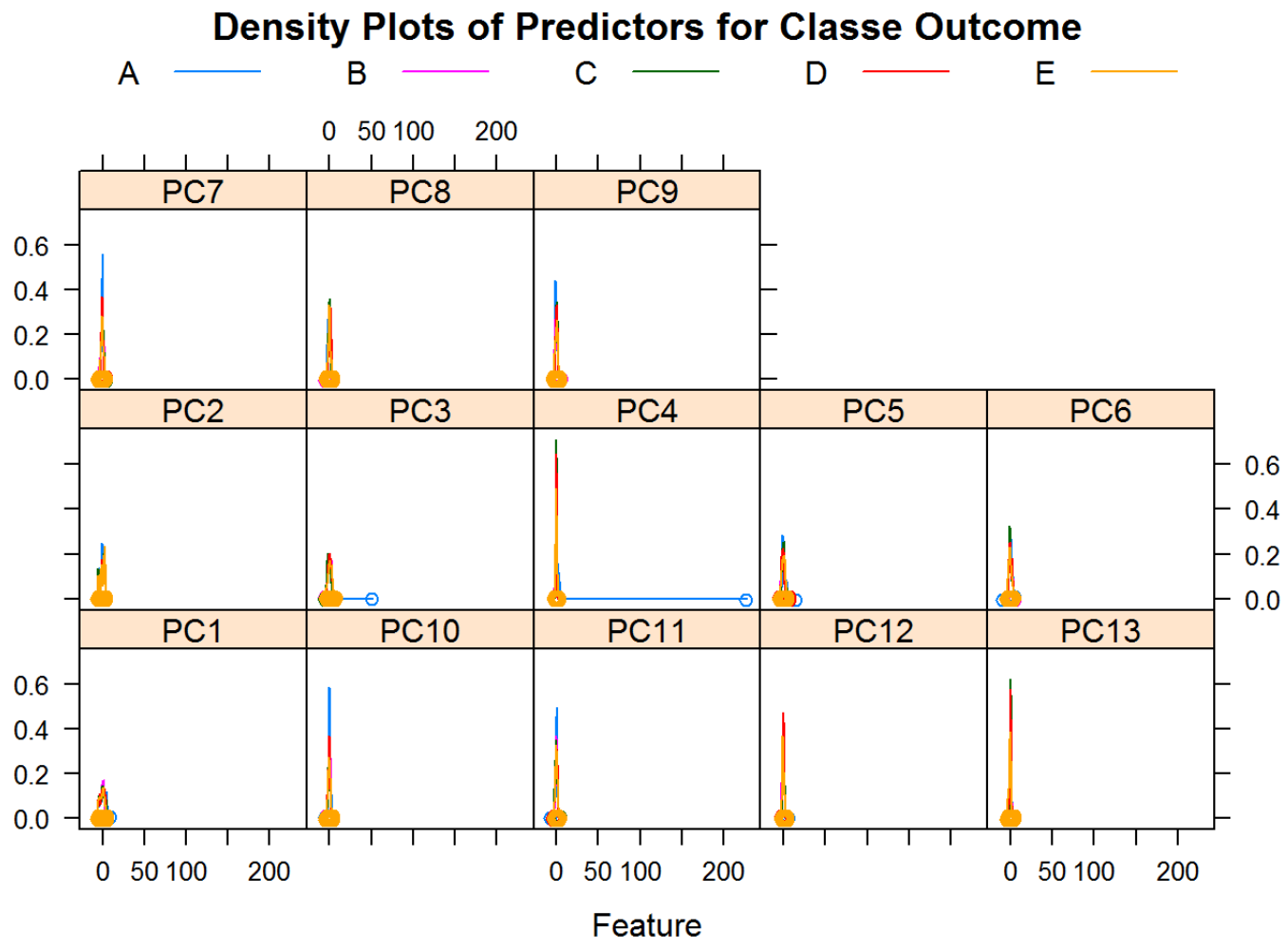
```
## [1] 13
```

Reduced training set with fewer predictors using principal components

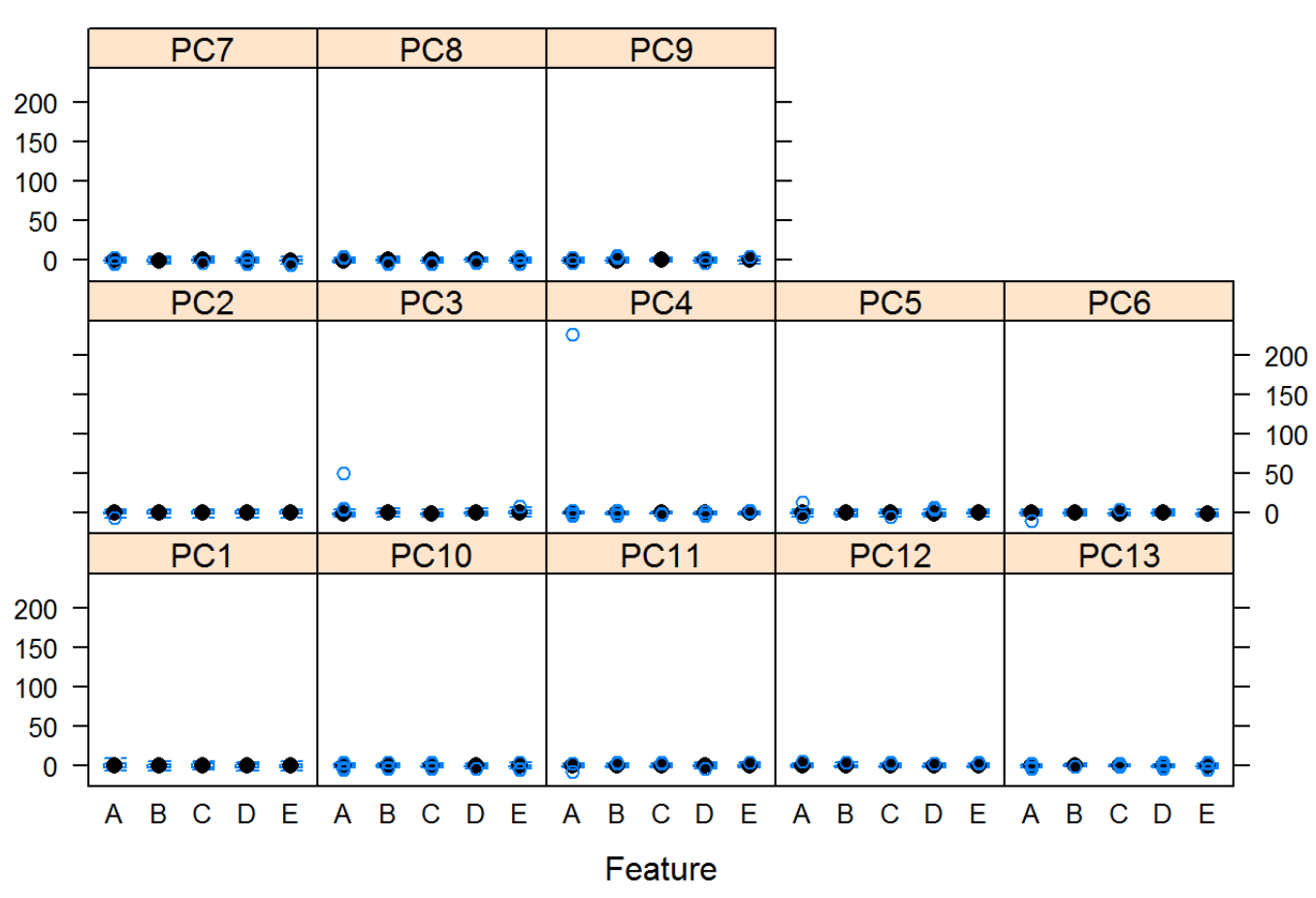
```
trainPC <- predict(preProc, trainSet[ , 2:58])
crossPC <- predict(preProc, crossVal[ , 2:58])
```

Exploration of reduced list of predictors

```
featurePlot(x = trainPC[,3:15],
            y = trainSet$classe,
            plot = "density",
            main = "Density Plots of Predictors for Classe Outcome",
            auto.key = list(columns = 5))
```




```
featurePlot(x = trainPC[,3:15],
            y = trainSet$classe,
            plot = "box",
            main = "Boxplots of Predictors for Classe Outcome",
            ## Add a key at the top
            auto.key = list(columns = 5))
```



Training Phase

Parallel computing to accelerate calculations

```
require(parallel)
```

```
## Loading required package: parallel
```

```
require(doParallel)
```

```
## Loading required package: doParallel
```

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

```
## Loading required package: foreach
```

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

```
## Loading required package: iterators
```

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

```
cl <- makeCluster(detectCores()- 1)
registerDoParallel(cl)
```

Seting the control parameters and fitting the model

```
controlParam <- trainControl(classProbs=TRUE,savePredictions=TRUE,allowParallel=TRUE, search = "random")
trainingModel <- train(trainSet$classe ~ ., data=trainPC, method="rf")
```

```
## Loading required package: randomForest
```

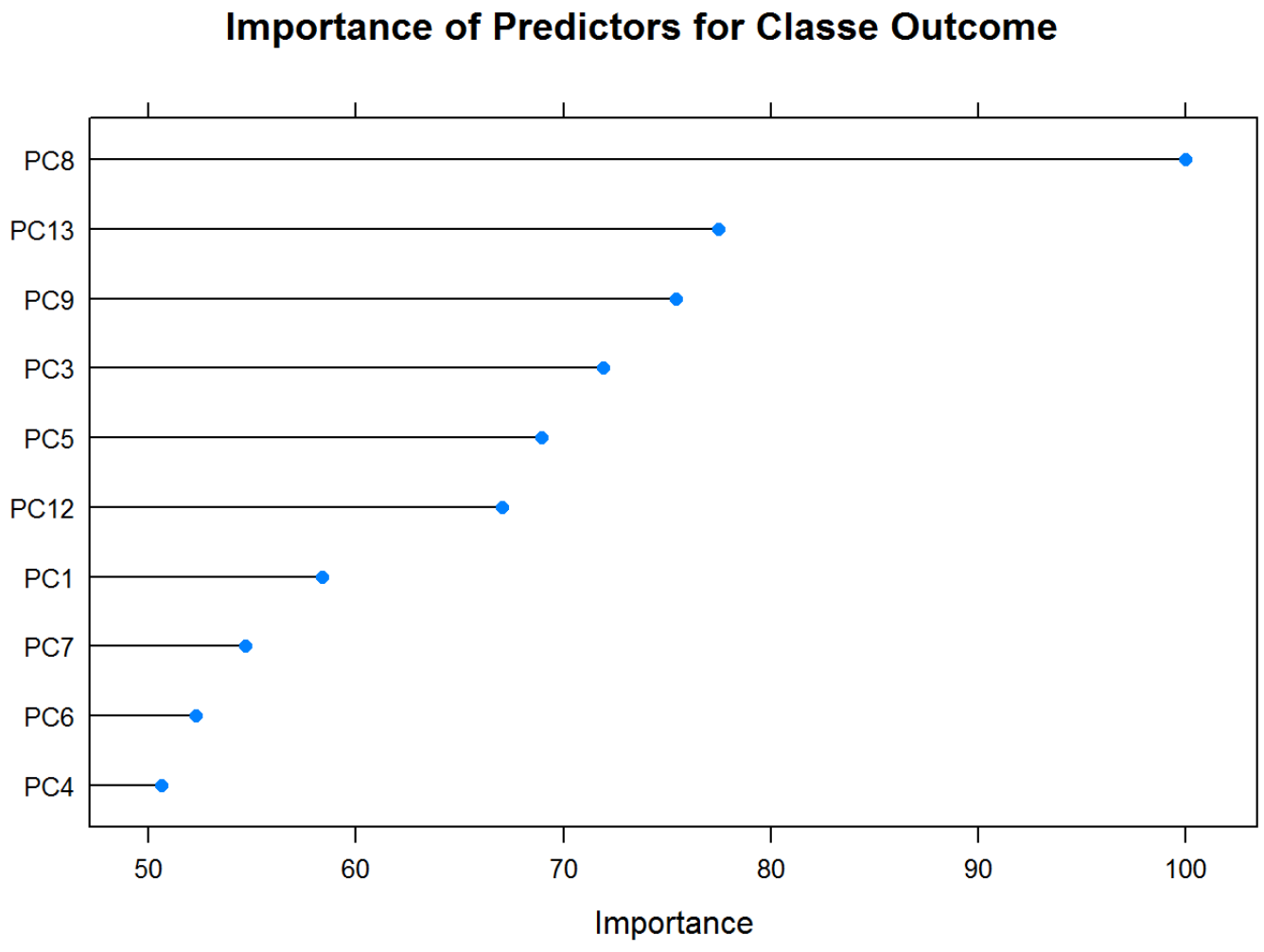
```
## Warning: package 'randomForest' was built under R version 3.1.3
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
trainingModel
```

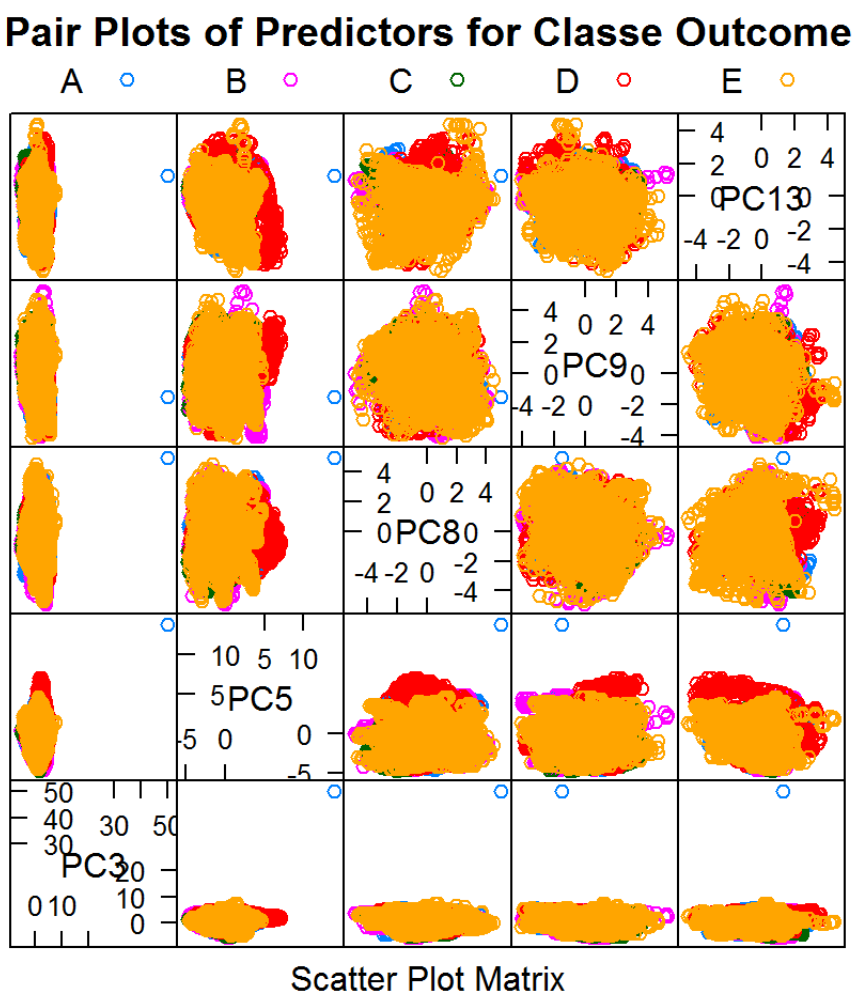
```
## Random Forest
##
## 13737 samples
##    14 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 13737, 13737, 13737, 13737, 13737, 13737, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.8335922  0.7897173  0.008513457  0.010729314
##   19    0.9624009  0.9524544  0.003520211  0.004449602
##   37    0.9528126  0.9403314  0.004698227  0.005944440
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 19.
```

```
varImpTrain <- varImp(trainingModel)
plot(varImpTrain, top = 10, main="Importance of Predictors for Classe Outcome")
```



Exploration of relationships among the principal components with importance over 70/100

```
pcover70 <-c(FALSE,FALSE,FALSE,FALSE,TRUE,FALSE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,FALSE,FALSE,TRUE)
featurePlot(x = trainPC[,pcover70],
            y = trainSet$classe,
            plot = "pairs",
            main = "Pair Plots of Predictors for Classe Outcome",
            auto.key = list(columns = 5))
```



Model Evaluation

Model assessment on training data set

```
est1 <- predict(trainingModel, trainPC)
```

```
## Loading required package: randomForest
```

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

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
confusionMatrix(est1, trainSet$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
```

Model assessment of cross-validation set

```
est2 <-predict(trainingModel, crossPC)
confusionMatrix(est2, crossVal$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1634   12    1    0    0
##          B   32 1110   25    0    0
##          C    8   17  991   24    1
##          D    0    0    9  925   13
##          E    0    0    0   15 1068
##
## Overall Statistics
##
##                Accuracy : 0.9733
##                  95% CI : (0.9689, 0.9773)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9663
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9761   0.9745   0.9659   0.9595   0.9871
## Specificity            0.9969   0.9880   0.9897   0.9955   0.9969
## Pos Pred Value         0.9921   0.9512   0.9520   0.9768   0.9861
## Neg Pred Value         0.9906   0.9939   0.9928   0.9921   0.9971
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2777   0.1886   0.1684   0.1572   0.1815
## Detection Prevalence   0.2799   0.1983   0.1769   0.1609   0.1840
## Balanced Accuracy      0.9865   0.9813   0.9778   0.9775   0.9920
```

Preparing the Testing Data Set and the Submission Files

```
testPC <- predict(preProc, testing2[ , 2:58])
predictions <-predict(trainingModel, testPC)
predictions2 <- as.array(predictions)

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(predictions2)
```

Stopping the cluster for parallel computing

```
stopCluster(cl)
```

```
library(knitr)
knit('knitr-minimal.Rmd')

rmarkdown::render("analysis.R")
```