MachineLearning\_Week4\_Project

setwd("~/Desktop")  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

# Download Datasets

training<-read.csv(file="pml-training.csv")  
testing<-read.csv(file="pml-testing.csv")

#Remove irrelevant variables for prediction

training<-training[,-(1:7)]  
testing<-testing[,-(1:7)]

#Remove variables with too many missing datapoints

training[training==""]<- NA  
naFraction<-colSums(is.na(training))/(dim(training)[1])  
training<-training[,(naFraction<0.95)]

#Create a validation set

inBuild<-createDataPartition(y=training$classe, p=0.7, list=FALSE)  
validation<-training[-inBuild,]  
buildData<-training[inBuild,]

#Use PCA to collapse number of variables to only the ones that are needed to differentiate between outcomes

preProc<-preProcess(buildData,method="pca",thresh=.95)  
trainPC<-predict(preProc,buildData)

#Use random forest method to create a model

mod1<-randomForest(buildData$classe ~ .,data=trainPC,do.trace=F)  
print(mod1)

##   
## Call:  
## randomForest(formula = buildData$classe ~ ., data = trainPC, do.trace = F)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 5  
##   
## OOB estimate of error rate: 2.5%  
## Confusion matrix:  
## A B C D E class.error  
## A 3871 11 13 7 4 0.008960573  
## B 45 2566 39 1 7 0.034612491  
## C 7 39 2319 26 5 0.032136895  
## D 4 1 89 2155 3 0.043072824  
## E 2 17 15 8 2483 0.016633663

importance(mod1)

## MeanDecreaseGini  
## PC1 585.9720  
## PC2 449.6343  
## PC3 505.6156  
## PC4 376.5724  
## PC5 557.4779  
## PC6 446.8919  
## PC7 419.8595  
## PC8 650.6054  
## PC9 514.4113  
## PC10 419.8321  
## PC11 345.8684  
## PC12 628.1784  
## PC13 373.7049  
## PC14 634.2082  
## PC15 488.4701  
## PC16 437.1889  
## PC17 404.9278  
## PC18 300.0855  
## PC19 352.1992  
## PC20 359.0612  
## PC21 383.8327  
## PC22 422.0030  
## PC23 239.4950  
## PC24 269.5516  
## PC25 293.9819

#Predict on the validation data

validation[validation==""]<- NA  
naFractionTest<-colSums(is.na(validation))/(dim(validation)[1])  
validation<-validation[,(naFractionTest<0.95)]  
  
validPC<-predict(preProc,validation)  
confusionMatrix(as.factor(validation$classe), predict(mod1,validPC))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1663 4 5 2 0  
## B 26 1099 14 0 0  
## C 0 16 997 13 0  
## D 2 1 28 931 2  
## E 0 5 10 7 1060  
##   
## Overall Statistics  
##   
## Accuracy : 0.9771   
## 95% CI : (0.9729, 0.9807)  
## No Information Rate : 0.2873   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.971   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9834 0.9769 0.9459 0.9769 0.9981  
## Specificity 0.9974 0.9916 0.9940 0.9933 0.9954  
## Pos Pred Value 0.9934 0.9649 0.9717 0.9658 0.9797  
## Neg Pred Value 0.9934 0.9945 0.9883 0.9955 0.9996  
## Prevalence 0.2873 0.1912 0.1791 0.1619 0.1805  
## Detection Rate 0.2826 0.1867 0.1694 0.1582 0.1801  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.9904 0.9842 0.9700 0.9851 0.9968

#Predict classe of testing dataset using model created from training dataset

testing[testing==""]<- NA  
naFractionFinal<-colSums(is.na(testing))/(dim(testing)[1])  
testing<-testing[,(naFractionFinal<0.95)]  
  
testPC <- predict(preProc,testing)  
testing$classe <- predict(mod1,testPC)

#Conclusion ###The goal of this course project was to quantify how well participants completed a weight-lifting exercise. Data was collected from accelerometers on the belt, forearm, arm, and dumbell from 6 parcipants. Variables with too many missing values (over 95% missing) were not used to build the prediction model. 70% of the training dataset was used to build the model using random forest and the remaining observations (30%) were used for validation of the model. PCA was used to reduce the predictor set into a smaller set of variables that would differentiate between outcomes. When the random forest model was used on the validation set, there was 97.3% accuracy, sensitivity level between 92.4% - 99.1%, and specificity between 99%-99.8%. The model was used to predict the performance of 20 different test cases. Potential limitations of this study is that there was data from only 6 participants, with limited variety in background, meaning that it may not be a good predicting model for participants who are very different in terms of age and fitness levels.