ML Project

konstantin mingoulin

April 23, 2016

Background:

How well the participants perform weight-lifting exercises.Six participnts performed a varity of exaercise. Partcipants were supervised by an experienced trainer.Five classes (A - "correct" and 4 other) were identified based on common mistakes.

Objective:

Use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants and predict the correct the class of how they performed the exercise

Data:

- Data for the project consisted of 2 files:
  - Main dataset (refered as training) will be used for training, validation and testing of the models
  - Another dataset refered as "testing" and is used to evaluate the performance of the final model

Distribution of classes (classe variable) in the main dataset:

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

- Data clean-up
  - Check for NAs, compare Main dataset to Final testing
  - It appears that the same features have NAs. Too many to correct for outliers
  - Exclude from the list of variables in the model
  - Further reduce the list of features to exclude non-informative columns like timestamps and user names

List of features used in the model:

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
```

```
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"
```

Partition the original Main dataset into training, validation and testing:

```
##      [,1]       [,2]      [,3]
## [1,] "Training" "Testing" "Validation"
## [2,] "9619"     "4118"    "5885"
```

Principle component analysis was performed on Training dataset. The preprocessing model was then applied to both Tesing and Validation datasets. The main reasoning behind the application of PCA is feature count reduction.

Model selection approach:

- Create a vector of "methods" that were covered in the course lectures
- Loop through the list of "methods" on Training set:
  - collect Accuracy for Training, Testing and Validation sets at each step
  - predict outcome at each step for Training and Validation
  - select top 3 models based on Accuracy for the Testing data set
  - create ensemble model using predictions for Testing dataset. Use "glmnet" as the combining algorithm

List of models:

```
## [1] "rf"        "rpart"     "nb"        "lda"       "gbm"       "svmRadial"
```

Accuracy Report:

```
##      Method AccuracyTrain AccuracyTest AccuracyValidation
## 1        rf     1.0000000    0.9674599          0.9578590
## 2     rpart     0.3867346    0.3851384          0.3875956
## 3        nb     0.6497557    0.6197183          0.6285472
## 4       lda     0.5229234    0.5128703          0.5238743
## 5       gbm     0.8685934    0.8278290          0.8224299
## 6 svmRadial     0.9073708    0.8875668          0.8858114
```

Top 3 Algorithms

```
## [1] rf        svmRadial gbm
## Levels: gbm lda nb rf rpart svmRadial
```

Top 3 algorithms are: Random Forest, SVM and Boosting

Final Steps: Ensemble model based on "glmnet" as a combiner

Accuracy on the Validation set:

```
## Accuracy
## 0.957859
```

Final words:

```
* Performance of the combined model is only marginally better than Random Forest approach
* The final model achieved 20 out of 20 on the final testing dataset
```

Appendix: Code

```
library(caret)


#get the data into R
fileUrltrain<-"http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(url=fileUrltrain,destfile="train_raw.csv")

fileUrltest<-"http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(url=fileUrltest,destfile="test_raw.csv")


mainData<-read.csv("train_raw.csv")
finalTesting<-read.csv("test_raw.csv")


#Explore

table(mainData$classe)

set.seed(12345)

#Check for NAs, compare test and train. Same features have NAs. Too many to correct for outliers. Exclu
EmptyVar<-as.data.frame(cbind(apply(mainData,2,function(x) {sum(is.na(x))}),apply(finalTesting,2,functi
names(EmptyVar)<-c("NATrain","NATest","FeatureName")

ValidVars<-as.character(EmptyVar$FeatureName[EmptyVar$NATest==0])

#further reduce the list of vars to exclude non-informative columns like timestamps and user names

ValidVars<-ValidVars[8:59]

mainData<-data.frame(mainData[,ValidVars],classe=mainData$classe)
finalTesting<-finalTesting[,ValidVars]
```

```r
#partition the original trianing into training validation and testing
inBuild<-createDataPartition(y=mainData$classe, p=0.7, list=FALSE)

#split data in to validation and build
validation<-mainData[-inBuild,]
buildData<-mainData[inBuild,]

#split build into train and test

inTrain<-createDataPartition(y=buildData$classe,p=0.7, list=FALSE)

training<-buildData[inTrain,]
testing<-buildData[-inTrain,]

#preprocess the data by using PCA

preProc<-preProcess(training[,-53],method="pca")

trainPC<-predict(preProc,training)
testPC<-predict(preProc,testing)
validationPC<-predict(preProc,validation)
finalTestingPC<-predict(preProc,finalTesting)


#Models to consider
methods<-c("rf","rpart","nb","lda","gbm","svmRadial")



AccuracyListTest<-NULL
AccuracyListValidation<-NULL
AccuracyList<-NULL
modList<-list()
PredDF<-testPC$classe
PredVDF<-validation$classe

for (i in 1:length(methods)) {
    print(methods[i])

    if (i==5) {
    modTemp<-train(classe~.,method=methods[i],data=trainPC,verbose = FALSE)
    }
    else {
    modTemp<-train(classe~.,method=methods[i],data=trainPC)
    }
    #AccuracyTemp<-confusionMatrix(trainPC$classe,predict(modTemp,trainPC))$overall[1]
    #assign(prCounter,predict(modTemp,testPC))
    PredDF<-data.frame(PredDF,predict(modTemp,testPC))
    PredVDF<-data.frame(PredVDF,predict(modTemp,validationPC))

    modList[[i]]<-modTemp
```

```
    Accuracy<-confusionMatrix(trainPC$classe,predict(modTemp,trainPC))$overall[1]
    AccuracyTest<-confusionMatrix(testPC$classe,predict(modTemp,testPC))$overall[1]
    AccuracyValidation<-confusionMatrix(validationPC$classe,predict(modTemp,validationPC))$overall[1]

    AccuracyList<-c(AccuracyList,Accuracy)
    AccuracyListTest<-c(AccuracyListTest,AccuracyTest)
    AccuracyListValidation<-c(AccuracyListValidation,AccuracyValidation)
}

names(PredDF)<-c("classe",methods)
names(PredVDF)<-c("classe",methods)

AccuracyReport<-data.frame(Method=methods,AccuracyTrain=AccuracyList,AccuracyTest=AccuracyListTest,Accu

#select top 3 models based on Accuracy
top3Index<-order(-AccuracyListTest)[1:3]

#PredDF and PredVDF contain estimates by method but need to mo move column index by one to alighn with
combModFit<-train(classe~.,method="glmnet",data=PredDF[,c(1,top3Index+1)])

stackedAccuracyTest<-confusionMatrix(testPC$classe,predict(combModFit,PredDF))$overall[1]
stackedAccuracyValidation<-confusionMatrix(validationPC$classe,predict(combModFit,PredVDF))$overall[1]

print(stackedAccuracyTest);print(stackedAccuracyValidation)

#Predict Project Test

finalTestingPCDF<-data.frame(predict(modList[top3Index[1]],finalTestingPC),
                             predict(modList[top3Index[2]],finalTestingPC),
                             predict(modList[top3Index[3]],finalTestingPC))


names(finalTestingPCDF)<-methods[top3Index]

quizOutput<-data.frame(obs=(1:20),classe=predict(combModFit,finalTestingPCDF))
write.csv(quizOutput,"quizOutput.csv")
```