Fitness Acceleromter Analysis

Jillian Katz

2/15/2021

Intoduction

Project for Coursera (Hopkins) Practical Machine Learning

Jillian Katz

The purpose of this project is to analyze fitness accelerometer data to determine if accelerometer wearers are performing set weightlifting exercises correctly or incorrectly. The study data set includes 6 participants performing exercises in 5 different incorrect & correct ways (classified as A, B, C, D, & E).

Thank you to Groupware for supplying this data!

Assignment: "The goal of your project is to predict the manner in which they did the exercise. This is the"classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases. "

Data Exploration

Tidying data for analysis Upon viewing the pml_training dataset, I found that the first 7 columns had identification data (e.g., names of participant, timestamps)to exclude from model building. Following these were 152 columns of accelerometer data, and then finally the exercice form class ("classe"). Many of the columns contained primarily NA values, with summary information (e.g., standard deviations, averages) at a small number of rows. I considered making separate models for the different types of columns (all entries filled compared to only few); however, upon looking at the pml_testing data, I found that this 20-row dataset's columns were either all filled or all NA. Therefore, the NA-columns would not be useful in applying the predictive model to the test data.

Thus I used simplified training & validation (subsets from pml_training), and testing data frames containing only columns of accelerometer data (and results for training and validation) with all entries filled (no NAs).

```r
library(caret)
library(readr)
pml_training <- read_csv("C:/Users/jkjil/Downloads/pml-training.csv")
pml_testing <- read_csv("C:/Users/jkjil/Downloads/pml-testing.csv")

dim(pml_training) # 19622 x  160 : 160 columns
# many associated variables (accelerometers in different locations)
# requires classification, so cannot do regularized regression -
# except for glmnet
# other options: PCA,lda, rpart, rf

#View(pml_training)
# must handle NAs
# name and timestamps are not factors.  Neither is num_window (whatever that is)
# new_window appears to be the needed info to fill the previous chunk of NAs
# first 7 columns are not factors, but new window (6th?) is needed
# all following 7 until 160 (classe) appear to be relevant data

## also div/0 errors!

# yes for window: variance, standard devs, avgs, etc
# predict yes rows, and predict no rows with na columns removed

#View(pml_testing)

## pml_testing: only 20 rows.  no new_window == yes, many columns only NA
# no "classe" , 160 is "problem_id"

NAs <- sapply(pml_training, function(x){sum(is.na(x))})
sum(NAs != 0) # 100 columns do not have any NA entry
summary(NAs[NAs != 0]) # min: 19216, max: 19294 NAs

training <- pml_training[, NAs == 0] # remove columns with NAs (to match with test data limitations)
training <- training[, 8:60] # remove non-predictor columns: now only 51 possible predictors

set.seed(1)
trainIndex = createDataPartition(training$classe, p = 0.70, list = F)
training = training[trainIndex,]
val = training[-trainIndex,] # validation set

testing <- pml_testing[, NAs == 0] #remove columns with NAs (no information)
testing <- testing[, 8:59] # remove non-predictor columns
```

Model Building

For classification model building, I considered four methods in the caret package, along with principle component analysis (PCA). The four chosen methods included Linear Discriminant Analysis (lda), Decision Tree (rpart), Random Forest (rf), and Generalized Boosted Regression (gbm). After building each model, each was tested for accuracy with the validation set. They performed as follows:

Model	Accuracy
lda	0.71
tree	0.49
rf	0.998
gbm	0.976

As the decision tree performed under 50% accuracy, it was excluded from further consideration.

Preprocessing with PCA, with a threshold of 0.8, resulted in a reduction to 12 predictors. As the lda model had room for improvement, I created an lda model from the 12 PCs. This model performed with 47%

accuracy when applied to the training (not validation) data, and I concluded that PCA was not helpful in this problem.

```r
# training and testing must be data frames for caret

training <- data.frame(training)
testing <- data.frame(testing)
val <- data.frame(val)

val$classe <- as.factor(val$classe)
training$classe <- as.factor(training$classe)

tc <- trainControl(method = "cv", number = 5)

ldafit <- train(classe ~., data = training, method = "lda")
ldaConfusion <- confusionMatrix(val$classe,predict(ldafit,val))
# Accuracy 0.71

treefit <- train(classe~., data = training, method = "rpart", trControl = tc)
treeConfusion <- confusionMatrix(val$classe,predict(treefit,val))
# Accuracy 0.49, not good enough

rffit5 <- train(classe ~., data = training, ntree = 5, method = "rf",
                trControl = tc)
rfConfusion <- confusionMatrix(val$classe,predict(rffit5,val))
# Accuracy 0.998

gbmfit <- train(classe ~., data = training, method = "gbm",
                verbose = FALSE, trControl = tc)
gbmConfusion <- confusionMatrix(val$classe,predict(gbmfit,val))
# Accuracy 0.976


## Considering Preprocessing with PCA - I reject this
preobj <- preProcess(training,
                     method = c("pca"),
                     thresh = 0.8)
preobj$numComp #12

trainPC <- predict(preobj,training)
valPC <- predict(preobj, val)
testPC <- predict(preobj, testing)

ldaPC <- train(classe ~., method="lda",data=trainPC)

PCldaConfusion <- confusionMatrix(training$classe,predict(ldaPC,trainPC))
# Accuracy 0.47 -- not using PCA
```

Comparing Results

After applying the 3 successful models to the training set, I compared the results from each model to the other two. Unsurprisingly, rf and gbm predictions closely matched each other (agreeing on 19 of the 20 test cases). Additionally, there was one case in which the lda and gbm predictions matched and differed from the rf prediction.

I produced a random forest model to combine the predictors, and compared its results to the three original models. The combined predictor model matched the original rf predictions for all cases. However, it would have been better to choose a "majority vote" system of combining the predictors, as this model failed to predict the correct class for the single case in which the lda and gbm agreed and differed from the rf prediction.

```
plda <- predict(ldafit,testing)

prf <- predict(rffit5,testing)

pgbm <- predict(gbmfit,testing)

results <- data.frame(plda, prf, pgbm)
sum(prf == pgbm) # 19 / 20
sum(prf == plda) # 13
sum(pgbm == plda) # 14
sum(pgbm == plda & pgbm != prf) # 1
sum(prf == plda & pgbm != prf) # 0

# Combining Predictors
combotrain <- data.frame(training$classe,
                         predict(ldafit, training),
                         predict(rffit5, training),
                         predict(gbmfit, training))
names(combotrain) <- c("classe", "plda", "prf", "pgbm")
combofit <- train(classe ~ ., data = combotrain,
               ntree = 5, method = "rf")

combotest <- data.frame(plda, prf, pgbm)
pcombo <- predict(combofit, combotest)
results$pcombo <- pcombo ## apparently, this matches the rf prediction, rather than following the major
```

Conclusion

Linear Discriminant Analysis, Random Forest, and Generalized Boosted Regression models all performed admirably in predicting the classe of exercise performance. The lda model's lower accuracy was helpful to balance out the possible overfitting of the rf and gbm models. Combining the three predictive models, depending on how methodology, led to a prediction accuracy between 95% and 100% for the 20-case test set.

Allowing for lengthier computing times - such as to produce more trees for a random forest - and further fine-tuning the models could lead to even greater accuracy of models. However, these three models combined performed far better than the minimum acceptability.