Practical Machine Learning - Data Science Specialization

335emily

2019-08-04

Overview and Executive Summary

The goal of this assisngment is to predict how well 20 barbell lifts were performed, using data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. Participants were asked to perform movements correctly (in which case, the classe vairable is A) and incorrectly in 4 different ways (classe variables are B, C, D and E).

Training and validation data were provided. More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Loading Data

First, load the training and test sets from the online sources:

```
rawTrainingData <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
rawValidationData <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
```

Then, set a seed for reproducibility:

```
set.seed(1245)
```

Load the required libaries

```
library(caret)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
library(corrplot)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(rattle)
```

Exploratory Data Analysis

First, review the dimensions of the data to understand the size of the dataset.

```
dim(rawTrainingData)
```

```
## [1] 19622   160
```

Then, look use the head functions to review some of the data and the column names. The results are excluded from the report to conserve space.

```
head(rawTrainingData)
```

Cleaning Data

Removing Variables with many NA or Blank Values

Looking at the results from the head function result, there are many NAs in the data. Therefore, it would be useful to understand how many rows are complete.

```
sum(complete.cases(rawTrainingData))
```

```
## [1] 406
```

See that only 406 rows of the 19622 rows in the training data are complete. Therefore, it is likely that many of the columnns will not be useful for prediction, as they contain NA values or missing observations that could create errors during training. These can be removed.

```
maxNAPerVariable = 20
maxNACount <- nrow(rawTrainingData) / 100 * maxNAPerVariable
removeColumns <- which(colSums(is.na(rawTrainingData) | rawTrainingData=="") > maxNAC
ount)
clean01TrainingData <- rawTrainingData[,-removeColumns]
clean01ValidationData <- rawValidationData[,-removeColumns]
```

By removing variables with more than 20 NA or blank results, there are 60 variables remaining.

Removing Near Zero Variance Variables

Next, there are some variables with near zero variance, which can also be excluded.

```
lowVarVarible <-nearZeroVar(clean01TrainingData)
clean02TrainingData <- clean01TrainingData[,-lowVarVarible]
clean02ValidationData <- clean01ValidationData[,-lowVarVarible]
```

By further removing variables with low variance, there are 59 variables remaining.

Removing Identifiers and Timestamps

The final cleaning will remove variables used for identification or timestamps, assuming that neither will impact the `classe` variable,

```
cleanTrainingData <- clean02TrainingData[,-(1:5)]
cleanValidationData <- clean02ValidationData[,-(1:5)]
```

After all cleaning, there are 54 variables variables remaining of the original 160 variables.

Cross Validation

In order to perform cross validation on the features and the model, the training set will be split into a training set and a test set.

```
inTrain <- createDataPartition(y=cleanTrainingData$classe,p=0.75, list=FALSE)
training <- cleanTrainingData[inTrain,]
test <- cleanTrainingData[-inTrain,]
```

The `training` dataset will be used to create the model. The `test` dataset will be used for cross-validation of variable and model selection. Finally, the `cleanValidationData` dataset set will be used as a final validation set in the assignment.

More Exploratory Data Analysis

Correlation Review

Now that the data is clean and split, and the unhelpful variables are removed, the next step is to review if the variables are correlated to each other.

```
corrMat <- cor(cleanTrainingData[,-(ncol(cleanTrainingData))])
```

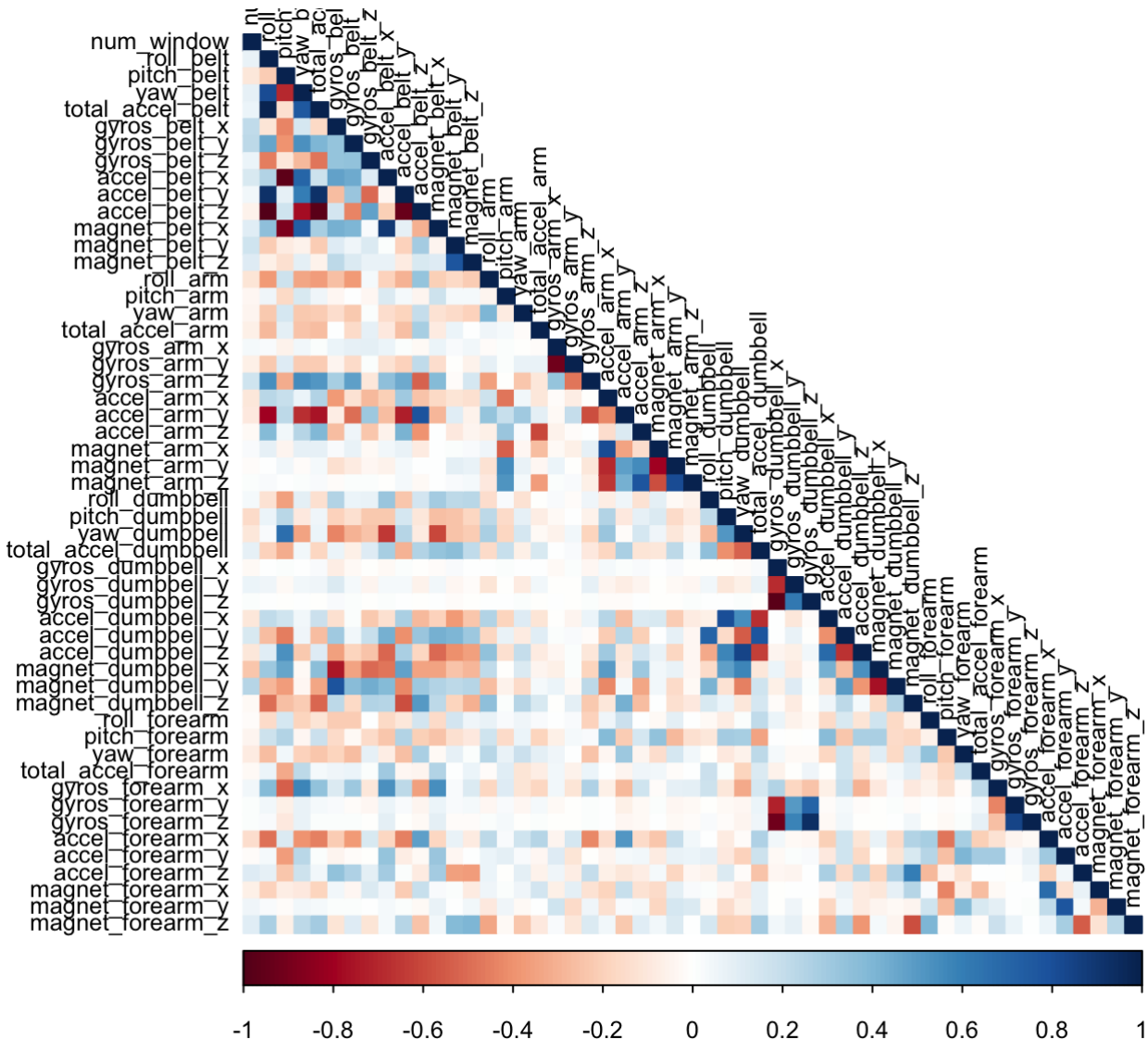


Fig 1 correlation

In the plot, darker colours demonstrate a higher correlation.

There are very few correlated variables, so a Principal Component Analysis is not productive, and therefore will not be performed.

Model Selection

Four methods were considered for creating the prediction model: Linear Regression Model with Multiple Covariates, Decision Tree, Random Forest and a Generalized Boosted Model.

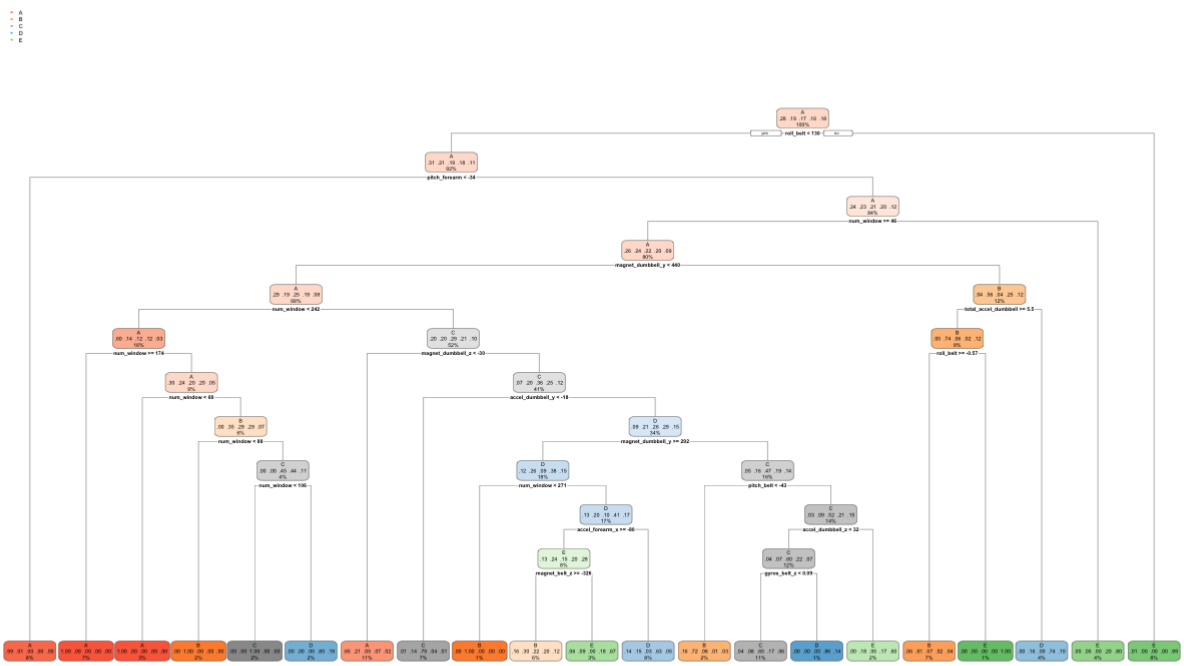
The model with the best accuracy found in during cross-validation will be selected for the final validation. Specifically, models will be built using the training data, and cross-validation will be performed using the test data. The model with the highest accuracy will be used to predict the 20 new test cases.

1. Linear Regression Model with Multiple Covariates

There is not sufficient information to determine which variables to include in the regression, therefore this model will not be tested. Including them all would lead to overfitting.

2. Decision Tree

```
modelDT <- rpart(classe ~ ., data = training, method = "class")
rpart.plot(modelDT)
```



```
predictDT <- predict(modelDT, test, type = "class")
confMatDT <- confusionMatrix(predictDT, test$classe)
accuracyDT <- confMatDT$overall['Accuracy']
```

3. Random Forest

```
control <- trainControl(method = "cv", number = 3, verboseIter=FALSE)
modelRF <- train(classe ~ ., data = training, method = "rf", trControl = control)
modelRF$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.22%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4183    1    0    0    1 0.0004778973
## B    8 2836    3    1    0 0.0042134831
## C    0    6 2561    0    0 0.0023373588
## D    0    0    9 2403    0 0.0037313433
## E    0    1    0    2 2703 0.0011086475
```

```
predictRF <- predict(modelRF, test)
confMatRF <- confusionMatrix(predictRF, test$classe)
accuracyRF <- confMatRF$overall['Accuracy']
```

4. Generalized Boosted Model

```
control <- trainControl(method = "repeatedcv", number = 5, repeats = 1, verboseIter =
FALSE)
modelGBM <- train(classe ~ ., data = training, trControl = control, method = "gbm", v
erbose = FALSE)
modelGBM$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 43 had non-zero influence.
```

```
predictGBM <- predict(modelGBM, test)
confMatGBM <- confusionMatrix(predictGBM, test$classe)
accuracyGBM <- confMatGBM$overall['Accuracy']
```

Comparing Accuracy

Of the three models created, the accuracies are: * Decision Tree: 0.7701876 * Random Forest: 0.9985726 * Generalized Boosted Mode: 0.9902121

The Random Forest model has the highest accuracy, and will be used to predict the class value of the validation data

Predicting Class Variable in the Validation Set

```
predictRF <- predict(modelRF, cleanValidationData)
predictRF
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```