Practical Machine Learning final submission

MeWe

20 April 2019

Practical Machine Learning - Final Course Project

This is the result of the final course project for the coursera course on Practical Machine Learning. The assignement consists of the following backgrounf information:

Project Question

One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants.

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.

Solution Approach

1. The goal is to find an algorithm that allows to predict the outcome of the class variable based on the availabe variables in the test set:

Approach:

a) Select only variables that are available in the test set, have no near zero variance and are non missing.

b) Prepare training set and split it up in training and validation set and do some descriptive analysis in R.

c) Estimate three different ml algorithms and use confusion matrix to show accuracy.

e) Show Importance of variable where appropriate.

d) Predict classe variables for the 20 cases in the test data data set.

```
suppressMessages(library(caret))
suppressMessages(library(dplyr))
suppressMessages(library(corrplot))
```

```
## Warning: package 'corrplot' was built under R version 3.5.3
```

Load data

```
train <- read.csv("C:/Users/Workstation/Documents/marimachine/coursera/Practical Machine Learning - Final Project/
pml-training.csv")
test <- read.csv("C:/Users/Workstation/Documents/marimachine/coursera/Practical Machine Learning - Final Project/p
ml-testing.csv")
```

a) Variable preprocessing

Checking dimensions of classe variable

```
unique(train$classe)
```

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

First 7 variables to be dropped out, since they are not meaningful.

```
train <- train[-c(1:7)]
test <- test[-c(1:7)]
```

Check for availability of remaining variables test set in train set. Here, only the first 10 rows will be displayed but one could compare the whole train and test data set. All variables available in train and test set apart from classe variable.

```
ntest<-names(test)
ntrain<-names(train)
result <- cbind(ntest, ntrain)
head(result, 10)
```

```
##       ntest                  ntrain
##  [1,] "roll_belt"            "roll_belt"
##  [2,] "pitch_belt"           "pitch_belt"
##  [3,] "yaw_belt"             "yaw_belt"
##  [4,] "total_accel_belt"     "total_accel_belt"
##  [5,] "kurtosis_roll_belt"   "kurtosis_roll_belt"
##  [6,] "kurtosis_picth_belt"  "kurtosis_picth_belt"
##  [7,] "kurtosis_yaw_belt"    "kurtosis_yaw_belt"
##  [8,] "skewness_roll_belt"   "skewness_roll_belt"
##  [9,] "skewness_roll_belt.1" "skewness_roll_belt.1"
## [10,] "skewness_yaw_belt"    "skewness_yaw_belt"
```

As a next step we want to identify variables with a near zero variance and remove them from test and trainig set since they would be irrelevant or would not bring additional explanatory power to the model.

```
nzv_train <- nearZeroVar(train)
train <- train[,-nzv_train]
test <- test[,-nzv_train]
```

Check for NA NaN -Inf variables in the training data set and remove them from both the training and test set as they cannot be used lateron to predict on classe variable in the test set.

```
namesNA <- names(test)[seq_along(names(test))[sapply(test, function(x)all(is.na(x)))]]
```

```
train <- train[,-which(names(train) %in% namesNA)]
test <- test[,-which(names(test) %in% namesNA)]
```

b) Prepare training and validation set from train dataset.

Use a split of 0.7 training and 0.3 validation.

```
set.seed(1981)
inTrain = createDataPartition(train$classe, p = .7)[[1]]
training = train[inTrain,]
validation = train[-inTrain,]
dim(training)
```

```
## [1] 13737    53
```

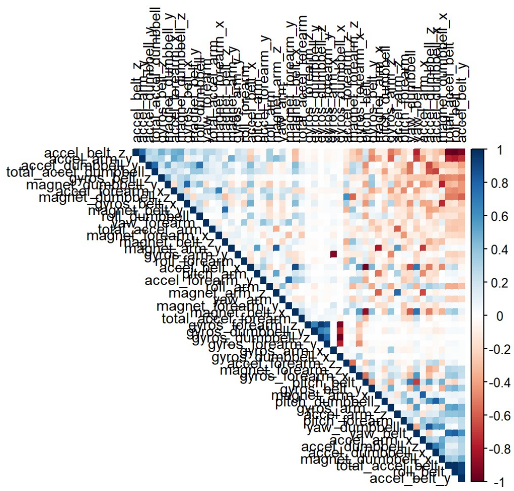
```
dim(validation)
```

```
## [1] 5885   53
```

Data visualisation -

 # 1 - Find highly correlated data

```
cor_mat <- cor(training[, -53]) # all corrs without classe var (53)
corrplot(cor_mat, order = "FPC", method = "color", type = "upper",
         tl.cex = 0.9, tl.col = rgb(0, 0, 0))
```



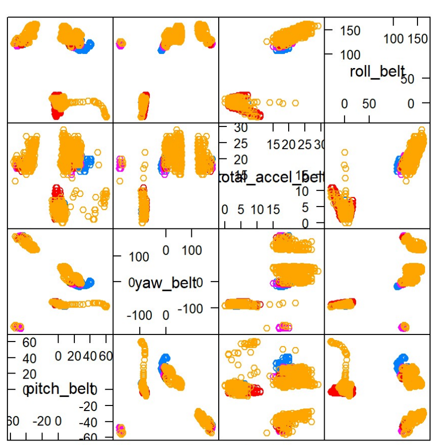

```
highlyCorrelated = findCorrelation(cor_mat, cutoff=0.70)
names(training)[highlyCorrelated]
```

```
##  [1] "accel_belt_z"      "roll_belt"         "accel_belt_y"
##  [4] "total_accel_belt"  "yaw_belt"          "accel_dumbbell_z"
##  [7] "accel_belt_x"      "pitch_belt"        "magnet_dumbbell_x"
## [10] "accel_dumbbell_y"  "magnet_dumbbell_y" "accel_arm_x"
## [13] "accel_dumbbell_x"  "accel_arm_z"       "magnet_arm_y"
## [16] "magnet_belt_z"     "accel_forearm_y"   "gyros_forearm_y"
## [19] "gyros_dumbbell_x"  "gyros_dumbbell_z"  "gyros_arm_x"
```

 # 2 - Some q plots qplot(pitch_belt,roll_belt,colour=classe,data=training) qplot(yaw_belt,roll_belt,colour=classe,data=training) qplot(total_accel_belt,roll_belt,colour=classe,data=training)

 # 3 - Feature plot

```
featurePlot(x=training [,c("pitch_belt","yaw_belt", "total_accel_belt","roll_belt")],
            y= training$classe,
            plot="pairs")
```



Scatter Plot Matrix

c) Training the models

I am training four models - one random forests, one gbm and one lda. I then perform predictions on the validation set and perform and analyse performance based on the accuracy indicator. Finally, I am performing a model ensemble based on the three models.

```
traindata = training

model_rf <- train(classe ~., method="rf", data = traindata)
model_gbm <- train(classe ~., method = "gbm", data = traindata, verbose = F )
model_lda <- train(classe ~., method = "lda", data = traindata, verbose = F )
```

Single prediction on validation set

```
rf_pred <- predict(model_rf, validation)
gbm_pred <- predict(model_gbm, validation)
lda_pred <- predict(model_lda, validation)
```

Combine predictions of single predictions

```
predDF <- data.frame(rf_pred, gbm_pred, lda_pred, classe = validation$classe, stringsAsFactors = F)
modelStack <- train(classe ~ ., data = predDF, method = "rf")
modelStack_pred <- predict(modelStack, validation)
```

Show confusion matrices for the models, incl. model stack.

```
random_forest_accuracy <- confusionMatrix(rf_pred, validation$class)$overall['Accuracy']
confusionMatrix(gbm_pred, validation$class)$overall['Accuracy']
```

```
##  Accuracy
## 0.9639762
```

```
confusionMatrix(lda_pred, validation$class)$overall['Accuracy']
```

```
##  Accuracy
## 0.7079014
```

```
confusionMatrix(modelStack_pred, validation$class)$overall['Accuracy']
```

```
##  Accuracy
## 0.9938828
```

Although the ensemble model (model stack) shows the best accuracy, I also want to interpret the used predictors. The variable importance for the random forest can help.

```
varImp(model_rf)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 52)
##
##                      Overall
## roll_belt            100.000
## pitch_forearm         58.573
## yaw_belt              54.975
## magnet_dumbbell_z     45.120
## pitch_belt            44.533
## magnet_dumbbell_y     44.504
## roll_forearm          43.316
## accel_dumbbell_y      22.510
## accel_forearm_x       18.513
## magnet_dumbbell_x     16.941
## roll_dumbbell         16.303
## magnet_belt_z         15.558
## accel_dumbbell_z      15.024
## magnet_forearm_z      14.373
## accel_belt_z          12.935
## magnet_belt_y         12.359
## total_accel_dumbbell  12.255
## gyros_belt_z          11.210
## yaw_arm               11.090
## magnet_belt_x          9.904
```

I finally decide for the random forest model to perform the prediction for the 20 test cases provided in the test data set.

```
final_pred_rf <- predict(model_rf, newdata=test)
final_pred_rf
```

```
##  [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
```