ML_Project

Wei Chen

2025-06-19

Course Project Instructions

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. 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. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

The data for this project come from this source:

http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

What you should submit

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.

Peer Review Portion

Your submission for the Peer Review portion should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).

Course Project Prediction Quiz Portion

Apply your machine learning algorithm to the 20 test cases available in the test data above and submit your predictions in appropriate format to the Course Project Prediction Quiz for automated grading.

Reproducibility

Due to security concerns with the exchange of R code, your code will not be run during the evaluation by your classmates. Please be sure that if they download the repo, they will be able to view the compiled HTML version of your analysis.

Loading Data

```
training <- read.csv("pml-training.csv", na.strings = c("NA", "", "#DIV/0!"))
testing <- read.csv("pml-testing.csv", na.strings = c("NA", "", "#DIV/0!"))
```

Preprocessing Data

First convert the outcome variable classe to be a factor variable. Then remove columns with too many NAs and remove the variables that are not preictive variables, such as user_name and timestamp varaibles. The same preprocessing was done for the testing data

```
training$classe <- factor(training$classe)
# Remove columns with too many NAs
training_clean <- training[, colMeans(!is.na(training)) > 0.95]

# Remove non-predictive variables
training_clean <- training_clean %>% select(-c(X, user_name, raw_timestamp_part_1, raw_timest
amp_part_2, cvtd_timestamp, new_window, num_window))

# Ensure same for test set
testing_clean <- testing %>% select(names(training_clean)[-ncol(training_clean)])
```

Build Model

Split Dataset

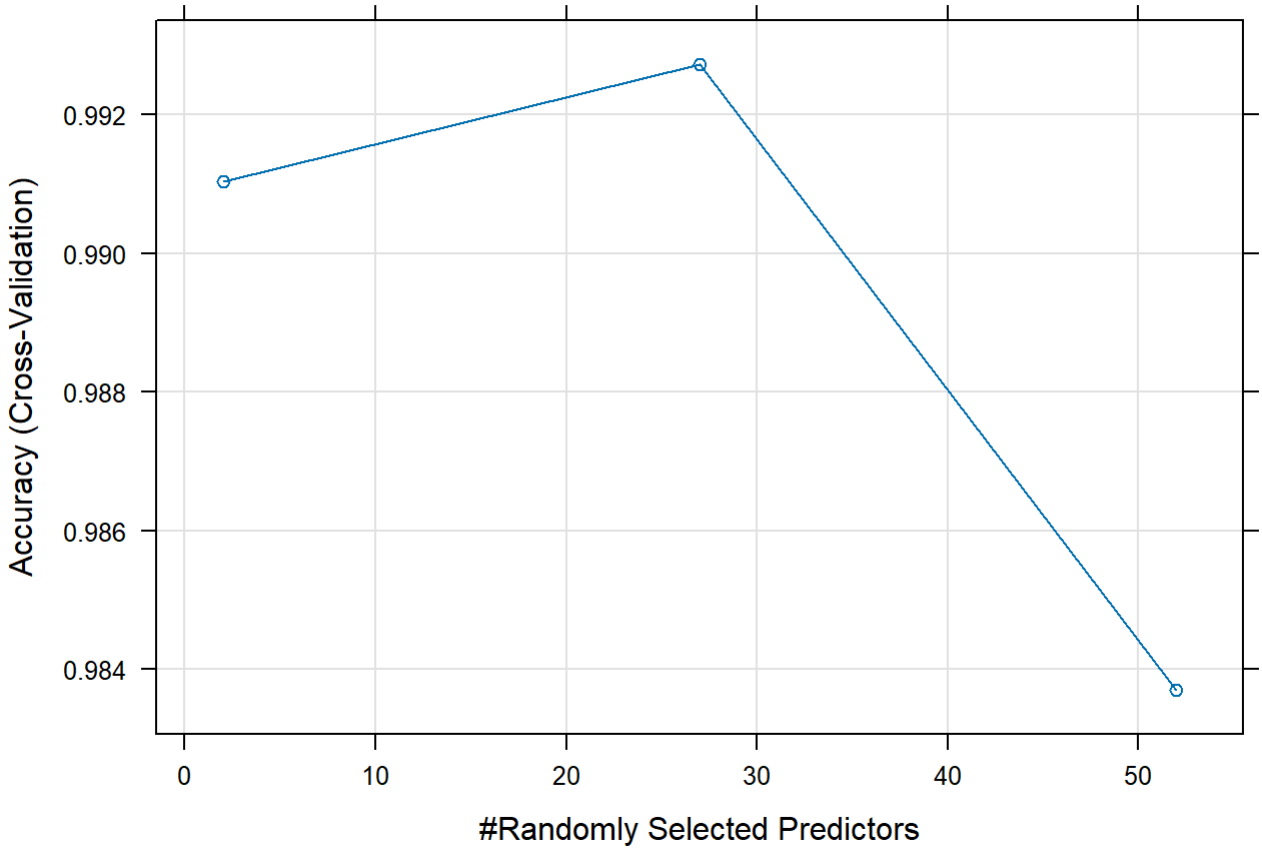
Split the training dataset to be 1/4 for validation data, and 3/4 for training data.

```
set.seed(1234)
inTrain = createDataPartition(training_clean$classe, p = 3/4)[[1]]
training_df = training_clean[inTrain,]
validation_df = training_clean[-inTrain,]
```

Train a Random Forest Model with Cross-Validation

Train a Random Forest Model using the training data with 5-fold cross-validation.
This is done using method = “cv” in the trainControl() function of the caret package.

```
set.seed(6888)
control <- trainControl(method = "cv", number = 5)
model_rf <- train(classe ~ ., data = training_df, method = "rf", trControl = control, importa
nce = TRUE)
plot(model_rf)
```



Out-Sample Error

The out-of-sample error is 0.0053.

```
# Predict on validation set
pred_rf <- predict(model_rf, validation_df)
cm <- confusionMatrix(pred_rf, validation_df$classe)
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394   10    0    0    0
##          B    0  938    5    0    0
##          C    1    1  848    6    0
##          D    0    0    2  798    2
##          E    0    0    0    0  899
##
## Overall Statistics
##
##                Accuracy : 0.9945
##                  95% CI : (0.992, 0.9964)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.993
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9884   0.9918   0.9925   0.9978
## Specificity            0.9972   0.9987   0.9980   0.9990   1.0000
## Pos Pred Value         0.9929   0.9947   0.9907   0.9950   1.0000
## Neg Pred Value         0.9997   0.9972   0.9983   0.9985   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1913   0.1729   0.1627   0.1833
## Detection Prevalence   0.2863   0.1923   0.1746   0.1635   0.1833
## Balanced Accuracy      0.9982   0.9936   0.9949   0.9958   0.9989
```

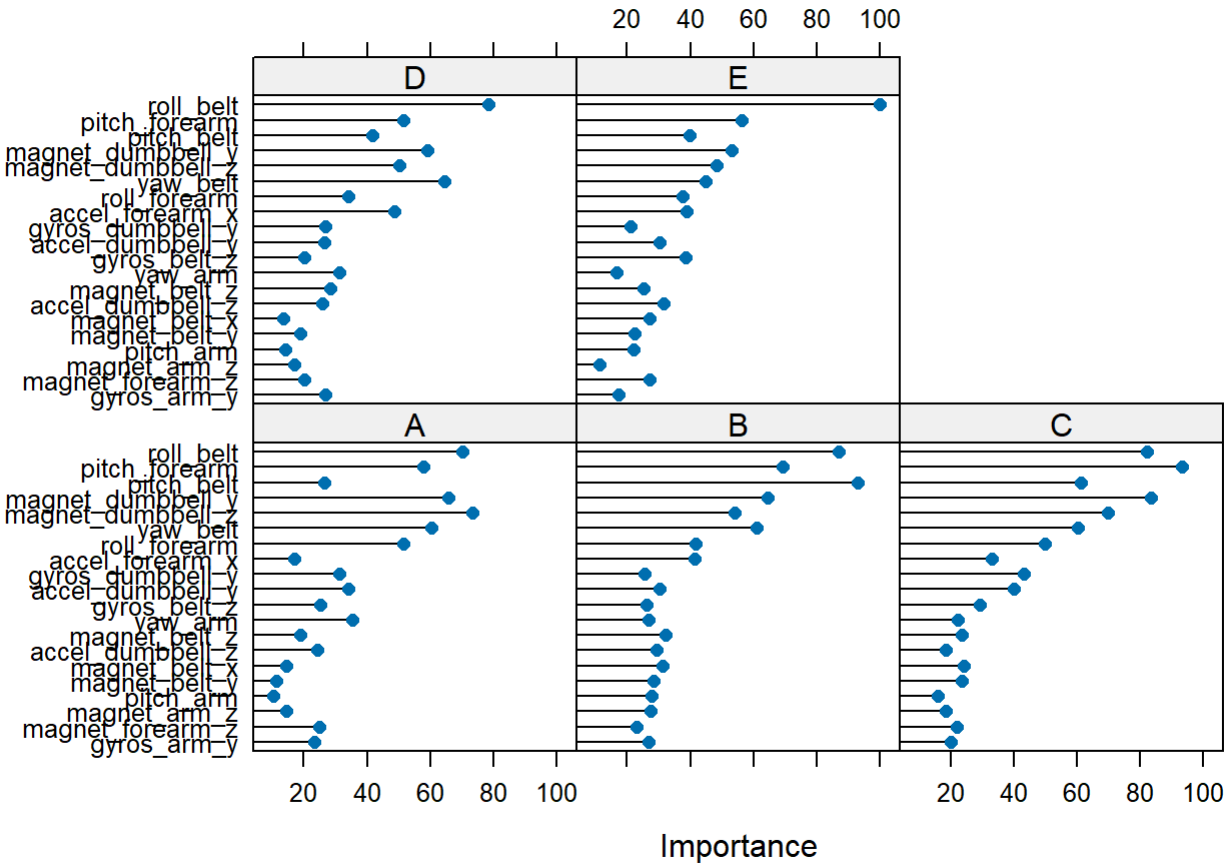
```
cm_tbl <- as.data.frame(cm$overall)
out_of_sample_error <- round((1 - cm_tbl[1,1])*10000)/10000
cat("\n")  # Adds a blank line
```

```
print(paste("out_of_sample_error is ",out_of_sample_error))
```

```
## [1] "out_of_sample_error is  0.0055"
```

Visualize the RF Model

```
importance <- varImp(model_rf)
plot(importance, top = 20)
```



Predict the Test Set

```
pred_final <- predict(model_rf, testing_clean)
print(pred_final)
```

```
##  [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
```