Prediction Assignment Writeup

Sean

December 3, 2016

Summary

We'll use the pml-training dataset on predicting the manner that people exercise. The outcome virable is 'classe'. After training the dataset, we will apply the model on the test dataset and predict the 20 manners.

Import data

Several variables in the testing data shows NA, we will filter them on both training and testing dataset.

```
pml_training <- read.csv('pml-training.csv')
pml_testing <- read.csv('pml-testing.csv')
pml_trd <- pml_training %>%
    select(-starts_with('max_'), -starts_with('min_'), -starts_with('var_'),
           -starts_with('avg_'), -starts_with('stddev_'), -starts_with('amplitude_'), -skewness_yaw_belt
           -kurtosis_yaw_forearm, -skewness_yaw_forearm, -skewness_yaw_dumbbell, -kurtosis_yaw_dumbbell
           -kurtosis_yaw_belt, -X, -user_name, -raw_timestamp_part_1, -raw_timestamp_part_2, -cvtd_times
           -kurtosis_roll_belt, -kurtosis_picth_belt, -skewness_roll_belt, -skewness_roll_belt.1,
           -kurtosis_roll_arm, -kurtosis_picth_arm, -kurtosis_yaw_arm, -skewness_roll_arm, -skewness_pit
           -skewness_yaw_arm, -kurtosis_roll_dumbbell, -kurtosis_picth_dumbbell, -skewness_roll_dumbbell
           -skewness_pitch_dumbbell, -kurtosis_roll_forearm, -kurtosis_picth_forearm, -skewness_roll_for
           -skewness_pitch_forearm, -new_window)
```

Convert some factor variables to numeric.

```
asNumeric <- function(x) as.numeric(as.character(x))
factorsNumeric <- function(d) modifyList(d, lapply(d[, sapply(d, is.factor)],
                                                   asNumeric))

pmldata <- data.frame(lapply(pml_trd, function(x) as.numeric(x)))
pmldata <- pmldata %>% select(-classe)
pmldata <- cbind(pmldata, pml_trd$classe)
```

Build the model with train control set to cross validation

Use rpart

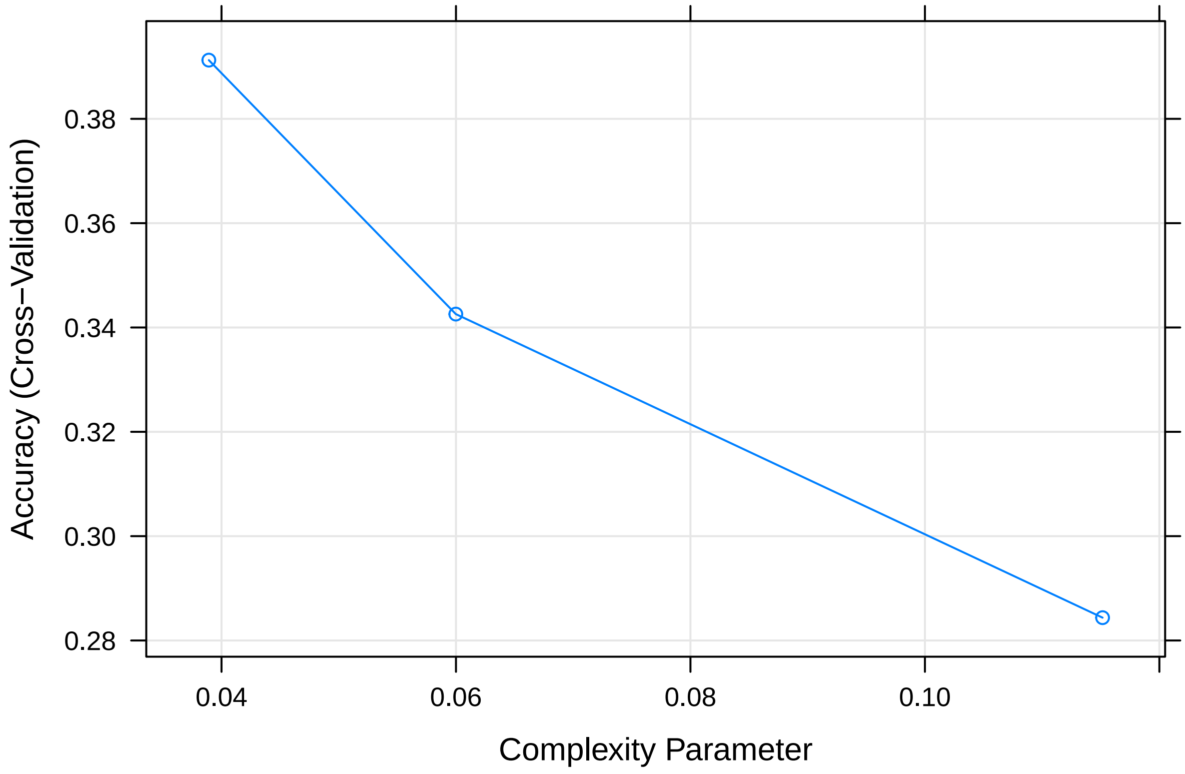
```
modelFit1 <- train(pmldata$`pml_trd$classe`~ ., method='rpart', preProcess='pca',
                   trControl = trainControl(method='cv'), data=pmldata)
```

```
## Loading required package: rpart
```

```
modelFit1$finalModel
```

```
## n= 19622
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 19622 14042 A (0.28 0.19 0.17 0.16 0.18)
##    2) PC8< 1.344481 16242 11052 A (0.32 0.18 0.2 0.12 0.19)
##      4) PC3>=-0.3868935 9968  5767 A (0.42 0.14 0.22 0.084 0.14) *
##      5) PC3< -0.3868935 6274  4605 E (0.16 0.23 0.17 0.17 0.27)
##       10) PC13< 0.2927563 4069  2770 B (0.16 0.32 0.2 0.16 0.16) *
##       11) PC13>=0.2927563 2205  1185 E (0.15 0.068 0.13 0.18 0.46) *
##    3) PC8>=1.344481 3380  2074 D (0.12 0.28 0.049 0.39 0.17) *
```

```
plot(modelFit1)
```



Obviously, the result is very poor with accuracy about 30%.

Use random forest

```
modelFit2 <- train(pmldata$`pml_trd$classe`~ ., method='rf', preProcess='pca',              trCon
```

```
modelFit2
```

Random Forest

19622 samples 53 predictor 5 classes: ‘A’, ‘B’, ‘C’, ‘D’, ‘E’

Pre-processing: principal component signal extraction (53), centered (53), scaled (53) Resampling: Bootstrapped (25 reps) Summary of sample sizes: 19622, 19622, 19622, 19622, 19622, 19622, … Resampling results across tuning parameters:

mtry Accuracy Kappa
2 0.9763976 0.9701402 27 0.9619002 0.9518036 53 0.9619973 0.9519278

Accuracy was used to select the optimal model using the largest value. The final value used for the model was mtry = 2.

- Note: since re-run the ramdon forest takes a lot time, I’ll quote the result above.

Convert the testing dataset

```
pml_testing1 <- pml_testing %>% select(-starts_with('max_'), -starts_with('min_'), -starts_with('var_')
                                       -starts_with('avg_'), -starts_with('stddev_'), -starts_with('ampl
                                       -kurtosis_yaw_forearm, -skewness_yaw_forearm, -skewness_yaw_dumb
                                       -kurtosis_yaw_belt, -X, -user_name, -raw_timestamp_part_1, -raw_
                                       -kurtosis_roll_belt, -kurtosis_picth_belt, -skewness_roll_belt,
                                       -kurtosis_roll_arm, -kurtosis_picth_arm, -kurtosis_yaw_arm, -ske
                                       -skewness_yaw_arm, -kurtosis_roll_dumbbell, -kurtosis_picth_dumb
                                       -skewness_pitch_dumbbell, -kurtosis_roll_forearm, -kurtosis_pict
                                       -skewness_pitch_forearm, -new_window)

pml_testing1 <- data.frame(lapply(pml_testing1, function(x) as.numeric(x)))
```

Use modelFit2 to predict

```
predict(modelFit2, newdata = pml_testing1)
```

[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