Writeup for submission of Course Project

VG

Sunday, April 26, 2015

Load and examine the training data

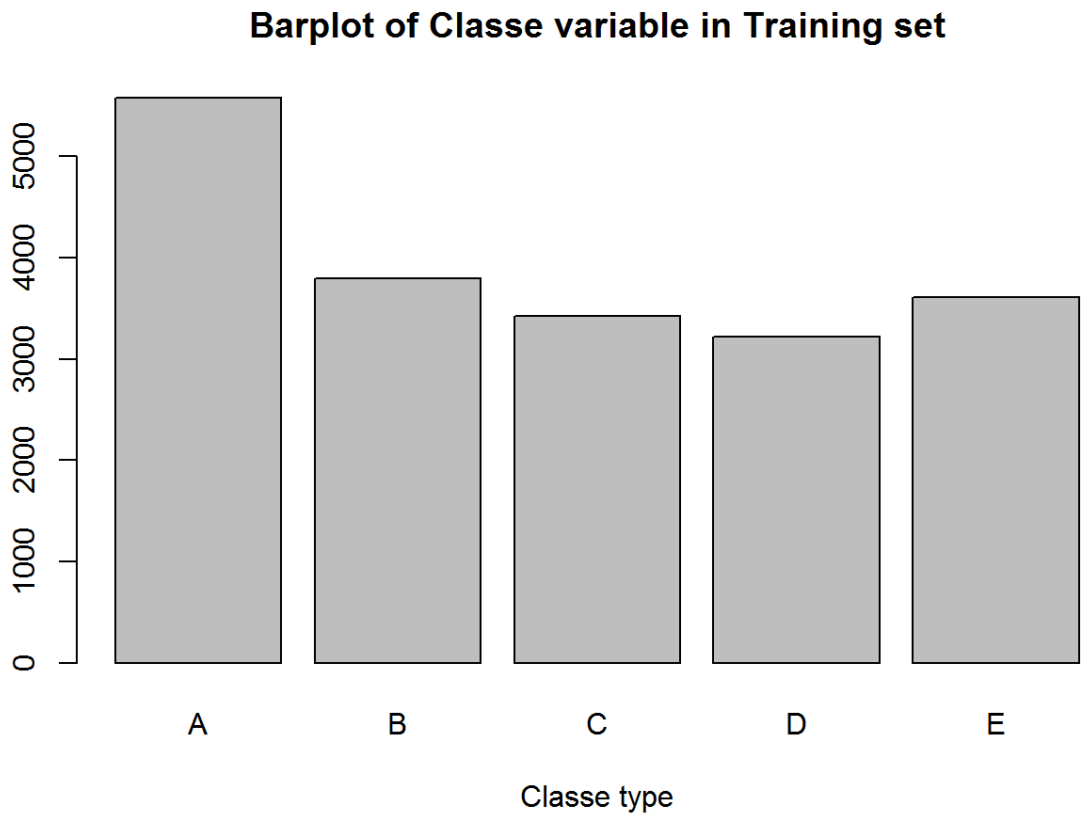
```
## Warning: package 'caret' was built under R version 3.1.3
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
## Warning: package 'randomForest' was built under R version 3.1.3
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

- There are 159 predictors and 19,622 observations in the test data provided.
- Some of the predictor variables are factor variables. Therefore, need to make sure that the factor variables have the same levels in the training and testing datasets.
- Only 406 rows of the 19,622 rows have complete observations. Therefore, cannot afford to lose as much data and only keep the 406 observations with complete data on all variables.
- All of the observations have data for the classe variable.
- Let’s plot the Classe variable to examine its distribution in the training data:



Load and examine the test data

```
## [1] 0
```

- There are 20 observations for testing.
- The predictor variables are the same as in the training data.
- There are factor variables.
- 100 of the predictor variables in the test set have all missing values, i.e., these variables have no predictive power. Therefore, we will exclude these variables from the training set in order to speed up the model. The number of predictors to be used goes down to 59.

Partition the training data

```
## [1] 14718    60
```

```
## [1] 4904   60
```

- Partition the training data into *train* and *test* sets — we will train the model on the *train* set and test it on the *test* set.

- We will use the *createDataPartition* command from the CARET package, which ensures a stratified sample with representative levels of response variable (a factor).
- 75% of the training data will be assigned to the *train* set and the remainder to the *test* set.
- We will use a Random Forest model (which as shown below performs very well) using the Random Forest package (which is much faster than the CARET package).
- Per one of the references provided during the course (http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr (http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr)): “In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error.” Therefore, we will not set aside a cross validation set.

Peform a Non-Zero Variance check on the *train* data.

```
##     freqRatio percentUnique       zeroVar           nzv
##      374.4368      736.3297        0.0000        1.0000
```

```
##            freqRatio percentUnique zeroVar  nzv
## new_window  46.63107     0.0135888   FALSE TRUE
```

- Variable new_window is identified as near-zero variance predictor. It is a factor variables with two levels - yes and no. Indeed, of the 19,266 observations only 406 have value no for this variable.
- We remove variable new_window from the training set. The number of predictors to be used goes down to 58.

Run exploratory Random Forest model

- We will use the default number of trees in the Random Forest package (500) plus 1 to avoid ties (ntree=501)

```
modelFitRFexpl <- randomForest(classe ~ ., data = train, ntree=501, na.rm=TRUE)
print(modelFitRFexpl)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = train, ntree = 501,      na.rm = TRU
E)
##                Type of random forest: classification
##                      Number of trees: 501
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4185    0    0    0    0           0
## B    0 2848    0    0    0           0
## C    0    0 2567    0    0           0
## D    0    0    0 2412    0           0
## E    0    0    0    0 2706           0
```

```
print(order(importance(modelFitRFexpl, type=2), decreasing=TRUE))
```

```
##  [1]  1  5  3  7  6  9 44 47  8 45 18 16 43 33 46 19 41 10 13 42 53 40 36
## [24] 35 17 20  2 30 38 58 55 27 34 31 15 56 22 57 14 12 28 48 32 21 37 25
## [47] 54 11 29 24 51 49 23 52 50 39 26  4
```

- The OOB estimate of the error rate is 0%, which suggests overfitting.
- When looking at the order of variable importance, we can see that the following variables are at the top of the list and they seem to be unique identifiers of each row and hence, each response variable: “X”, “cvtd_timestamp”, “raw_timestamp_part_1”, “roll_belt”
- We remove the first three of these predictors from the training data to avoid overfitting.

Re-fit the Random Forest model excluding the three predictors above and also excluding the new_window variable identified to have near-zero variance.

```
modelFitRF <- randomForest(classe ~ ., data = train[ ,-c(1, 3, 5, 6)], ntree=501, n
a.rm=TRUE)
print(modelFitRF)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = train[, -c(1, 3, 5,      6)], ntree =
501, na.rm = TRUE)
##                Type of random forest: classification
##                      Number of trees: 501
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.47%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4181    4    0    0    0 0.0009557945
## B   15 2829    4    0    0 0.0066713483
## C    0   14 2549    4    0 0.0070120764
## D    0    0   15 2395    2 0.0070480929
## E    0    0    4    7 2695 0.0040650407
```

```
print(order(importance(modelFitRF, type=2)))
```

```
##  [1]  2 22 46 35 48  7 19 45 10  8 11 47 33 20 25 21 50 24  1 44 17 30 28
## [24] 52 27 53  6 51 13 18 34 36 32 31 23 26 54  9 16 38 49 14 12 29 15 37
## [47] 39 42 40  4 41 43  5  3
```

- The OOB estimate of the error rate is now 0.47%, i.e., the expected number of misclassified observations on a test set of 4904 observations is approximately 23 misclassified cases.

Test the model using the *test* data

```
pred.FitRF <- predict(modelFitRF, test[ ,-c(60)], type="response")
table(observed = test$classe, predicted = pred.FitRF)
```

```
##          predicted
## observed    A    B    C    D    E
##        A 1394    1    0    0    0
##        B    2  945    2    0    0
##        C    0    9  845    1    0
##        D    0    0    7  797    0
##        E    0    0    0    0  901
```

- There are 22 misclassified cases out of 4904 observations, which closely mirrors the OOB estimate of the error rate (the latter is shown to be unbiased in a lot of situations).

Create predictions for the 20 sample test cases

```
vars.train <- names(train)
vars.test <- names(test_orig)
test_final <- test_orig[ ,names(test_orig) %in% vars.train]
levels(test_final$cvtd_timestamp) <- levels(train$cvtd_timestamp)
pred.FitRF2 <- predict(modelFitRF, test_final)
print(pred.FitRF2)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

- First, ensure that the test cases only keep the variables used in the training set and then ensure that the factor variables across the two sets use the same levels.
- The re-fit model above achieved 100% accuracy when run against the 20 sample test cases in Part 2 of the assignment.