Practical-Machine-Learning-project.R

Mahesha

Sat Nov 04 19:09:03 2017

```
install.packages("caret")
```

```
## Error in install.packages : Updating loaded packages
```

```
library(caret)
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
## Import files
setwd("C:\\Users\\Mahesha\\Desktop\\Desktop\\Data_Science\\Courseera\\Practica
l Machine learning\\Project")

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

dim(trainds)
```

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

```
dim(testds)
```

```
## [1]  20 160
```

```
## Data explore
#str(trainds)
#summary(trainds)

## Data cleaning

# remove variables with nearly zero variance
nzv <- nearZeroVar(trainds)
trainds <- trainds[, -nzv]

# remove variables that are mostly NA
mostlyNA <- sapply(trainds, function(x) mean(is.na(x))) > 0.75
trainds <- trainds[, mostlyNA==F]

# remove variables which won't contribute much for prediction, in this case 1:
5 variables are of no use.
trainds <- trainds[, -(1:5)]

dim(trainds)
```

```
## [1] 19622    54
```

```
# Data Split
set.seed(123)
traindssplit <- createDataPartition(y=trainds$classe, p=0.6, list=F)
trainds1 <- trainds[traindssplit, ]
trainds2 <- trainds[-traindssplit, ]

dim(trainds1)
```

```
## [1] 11776    54
```

```
dim(trainds2)
```

```
## [1] 7846   54
```

```
##Build model , i am using Random forest, Decision Trees and Boosting
## Model with Random forest
install.packages("randomForest")
```

```
## Installing package into 'C:/Users/Mahesha/Documents/R/win-library/3.4'
## (as 'lib' is unspecified)
```

```
## package 'randomForest' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Mahesha\AppData\Local\Temp\RtmpIRzxwj\downloaded_packages
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.2
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
#  use 3-fold CV to select optimal tuning parameters
fitControl <- trainControl(method="cv", number=3, verboseIter=F)

#  model on trainds1
install.packages("e1071")
```

```
## Installing package into 'C:/Users/Mahesha/Documents/R/win-library/3.4'
## (as 'lib' is unspecified)
```

```
## package 'e1071' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Mahesha\AppData\Local\Temp\RtmpIRzxwj\downloaded_packages
```

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.4.2
```

```
model1 <- train(classe ~ ., data=trainds1, method="rf", trControl=fitControl)

# print final model to see tuning parameters it chose
model1$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: 0.35%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3346    1    0    0    1 0.0005973716
## B    7 2270    1    1    0 0.0039491005
## C    0    7 2045    2    0 0.0043816943
## D    0    0   13 1916    1 0.0072538860
## E    0    1    0    6 2158 0.0032332564
```

```
# use model1 to predict classe in validation set (trainds2)
preds1 <- predict(model1, newdata=trainds2)

# show confusion matrix to get estimate of out-of-sample error
confusionMatrix(trainds2$classe, preds1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    0    0    0    0
##          B    1 1514    3    0    0
##          C    0    7 1358    3    0
##          D    0    0    7 1279    0
##          E    0    0    0    0 1442
##
## Overall Statistics
##
##                Accuracy : 0.9973
##                  95% CI : (0.9959, 0.9983)
##     No Information Rate : 0.2846
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9966
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9954   0.9927   0.9977   1.0000
## Specificity            1.0000   0.9994   0.9985   0.9989   1.0000
## Pos Pred Value         1.0000   0.9974   0.9927   0.9946   1.0000
## Neg Pred Value         0.9998   0.9989   0.9985   0.9995   1.0000
## Prevalence             0.2846   0.1939   0.1744   0.1634   0.1838
## Detection Rate         0.2845   0.1930   0.1731   0.1630   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9998   0.9974   0.9956   0.9983   1.0000
```

```
#################################
## Model with Decision Trees
library(rpart)
set.seed(123)
model2 <- rpart(classe ~ ., data=trainds1, method="class")
preds2 <- predict(model2, trainds2, type = "class")
confusionMatrix(trainds2$classe, preds2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2031   90    0   86   25
##          B  243 1044   76   60   95
##          C   43  135 1043  129   18
##          D   62  110   42  993   79
##          E   36   62    5  121 1218
##
## Overall Statistics
##
##                Accuracy : 0.8067
##                  95% CI : (0.7977, 0.8153)
##     No Information Rate : 0.3078
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7547
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8410   0.7245   0.8945   0.7149   0.8488
## Specificity            0.9630   0.9260   0.9513   0.9546   0.9651
## Pos Pred Value         0.9099   0.6877   0.7624   0.7722   0.8447
## Neg Pred Value         0.9316   0.9373   0.9810   0.9396   0.9661
## Prevalence             0.3078   0.1837   0.1486   0.1770   0.1829
## Detection Rate         0.2589   0.1331   0.1329   0.1266   0.1552
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9020   0.8252   0.9229   0.8348   0.9069
```

```
#####################################
## Model with GBM (Generalised Boosting)
install.packages("gbm")
```

```
## Installing package into 'C:/Users/Mahesha/Documents/R/win-library/3.4'
## (as 'lib' is unspecified)
```

```
## package 'gbm' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Mahesha\AppData\Local\Temp\RtmpIRzxwj\downloaded_packages
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.4.2
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.3
```

```
set.seed(123)
fitControl <- trainControl(method = "repeatedcv",
                           number = 3,
                           repeats = 1)

model3 <- train(classe ~ ., data=trainds1, method = "gbm",
                trControl = fitControl,
                verbose = FALSE)


gbmFinMod3 <- model3$finalModel

preds3 <- predict(model3, newdata=trainds2)
confusionMatrix(trainds2$classe, preds3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2228    3    0    1    0
##          B   10 1494   12    2    0
##          C    0   17 1347    3    1
##          D    0    7   16 1261    2
##          E    1    3    4    9 1425
##
## Overall Statistics
##
##                Accuracy : 0.9884
##                  95% CI : (0.9858, 0.9907)
##     No Information Rate : 0.2854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9853
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9951   0.9803   0.9768   0.9882   0.9979
## Specificity            0.9993   0.9962   0.9968   0.9962   0.9974
## Pos Pred Value         0.9982   0.9842   0.9846   0.9806   0.9882
## Neg Pred Value         0.9980   0.9953   0.9951   0.9977   0.9995
## Prevalence             0.2854   0.1942   0.1758   0.1626   0.1820
## Detection Rate         0.2840   0.1904   0.1717   0.1607   0.1816
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9972   0.9883   0.9868   0.9922   0.9976
```

```
## Models Comparison
## Considering the Accuracy , model1 ( Random Forest) looks better compared to
model2 and model3.

## Prediction for test dataset using model1

## Data cleaning for test dataset just like training dataset

# remove variables with nearly zero variance
nzv2 <- nearZeroVar(testds)
testds <- testds[, -nzv2]

# remove variables that are mostly NA
mostlyNA2 <- sapply(testds, function(x) mean(is.na(x))) > 0.75
testds <- testds[, mostlyNA2==F]

# remove variables which won't contribute much for prediction, in this case 1:
5 variables are of no use.
testds <- testds[, -(1:5)]

dim(testds)
```

```
## [1] 20 54
```

```
## Final prediction for test dataset
predsfinal <- predict(model1, newdata=testds)
predsfinal
```

```
##  [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
```

```
## writing to a file
write.table(predsfinal,"Outputfinal1.txt")
write.table(predsfinal,"Outputfinal2.csv")

## writing to individual files
# convert predictions to character vector
predsfinal <- as.character(predsfinal)

# create function to write predictions to files
pml_write_files <- function(x) {
  n <- length(x)
  for(i in 1:n) {
    filename <- paste0("problem_id_", i, ".txt")
    write.table(x[i], file=filename, quote=F, row.names=F, col.names=F)
  }
}

# create prediction files to submit
#pml_write_files(predsfinal)
```