ML_Project

Dev Das

October 22, 2020

Data and Exploration

Downloading the data files and doing some basic exploration

```
train <- read.csv('pml-training.csv')
test <- read.csv('pml-testing.csv')
dim(train) # Looking at the rows and columns of the train dataset
```

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

```
dim(test) # Looking at the rows and columns of the test dataset
```

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

Train validation Split

We need to split the training data given to a training set and validation set so we can validate the model we create

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

```
## Warning: package 'lattice' was built under R version 3.6.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
set.seed(123)
trainset <- createDataPartition(train$classe, p = 0.6, list = FALSE)
training <- train[trainset, ]
validation <- train[-trainset, ]
```

Cleaning the data

Exclude columns with lots of missing values exclude descriptive columns like name etc

```
cnt <- sapply(training, function(x) {
    sum(!(is.na(x) | x == ""))
})
nullcol <- names(cnt[cnt < 0.5 * length(training$classe)])
descriptcol <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2",
    "cvtd_timestamp", "new_window", "num_window")
excludecols <- c(descriptcol, nullcol)
training <- training[, !names(training) %in% excludecols]
```

Training the model

We are going to use a Random Forest model to classify the data we are given to determine the prediction of the 'classe' variable. We use random forest since this will be a supervied learning model in the end. I set an arbitrary value of 8 trees so that the model could be run in a reasonable amount of time.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
## randomForest 4.6-14
```

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

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

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

```
rfModel <- randomForest(classe ~ ., data = training, importance = TRUE, ntrees = 8)
```

Predictions for the Model

Next we will test the model to see how it performs on the training set and validation set. The training set we are expecting very accurate predictions since that is the data we trained the model with. The validation set will give more interesting results

```
predict_training <- predict(rfModel, training)
print(confusionMatrix(predict_training, training$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
```

```
##          A 3348    0    0    0    0
##          B    0 2279    0    0    0
##          C    0    0 2054    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

As expected the training set had very high accuracy values, so we will look at the validation set next

```
predict_validation <- predict(rfModel, validation)
print(confusionMatrix(predict_validation, validation$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2228    9    0    0    0
##          B    3 1507   15    0    0
##          C    0    2 1352   16    1
##          D    0    0    1 1269    3
##          E    1    0    0    1 1438
##
## Overall Statistics
##
##                Accuracy : 0.9934
##                  95% CI : (0.9913, 0.995)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9916
##
```

```
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9928   0.9883   0.9868   0.9972
## Specificity            0.9984   0.9972   0.9971   0.9994   0.9997
## Pos Pred Value         0.9960   0.9882   0.9861   0.9969   0.9986
## Neg Pred Value         0.9993   0.9983   0.9975   0.9974   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2840   0.1921   0.1723   0.1617   0.1833
## Detection Prevalence   0.2851   0.1944   0.1747   0.1622   0.1835
## Balanced Accuracy      0.9983   0.9950   0.9927   0.9931   0.9985
```

We can see here that the accuracy decreased by a small amount here, however 99.34% is still a very good model. This also tells us that the out of sample error is extremely small, being less than 1%.

Testing set

These values will be entered into the quiz

```
prediction_test <- predict(rfModel, test)
prediction_test
```

```
##  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
```