Coursera Machine Learning Project

Louis

11/15/2020

I loaded the necessary libraries to do the machine learning.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.3
```

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

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

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.3
```

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

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.0.3
```

```
## corrplot 0.84 loaded
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.3
```

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

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

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

```
library(latexpdf)
```

The test and training data was loaded and I created an data partition where I then made training and test validation data sets based on the data partition.

```
train_data <- read.csv("pml-training.csv")
test_data <- read.csv("pml-testing.csv")
intrain <- createDataPartition(train_data$classe, p=0.7, list = FALSE)
train_valid <- train_data[intrain,]
test_valid <- train_data[-intrain,]
```

Get rid of variables with zero variance then removed variable with over 95% NA values to make the training as efficient as possible.

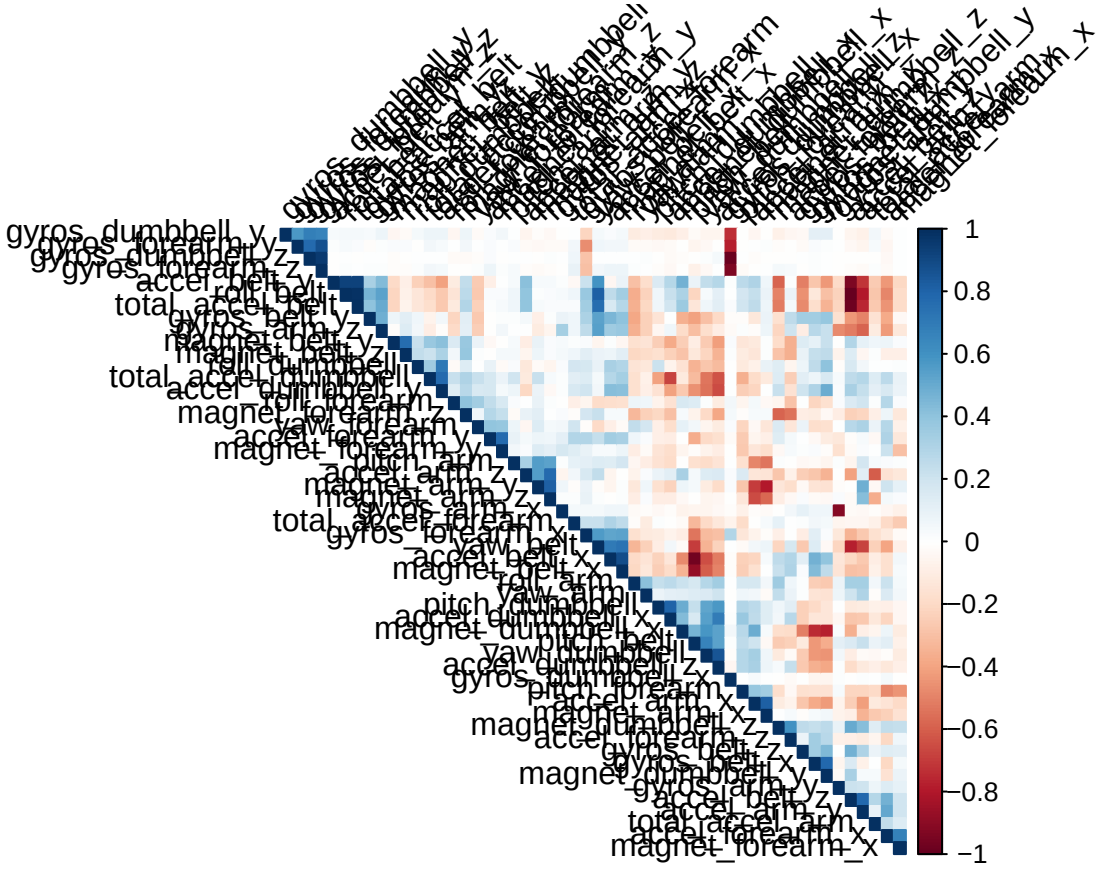
```
near_zero <- nearZeroVar(train_valid)
train_valid <- train_valid[,-near_zero]
test_valid <- test_valid[, -near_zero]

all_na <- sapply(train_valid,function(l) mean(is.na(l))) > 0.95
train_valid <- train_valid[, all_na == FALSE]
test_valid <- test_valid[, all_na == FALSE]
```

#Remove uncessary variables

```
train_valid <- train_valid[, -c(1:6)]
test_valid <- test_valid[, -c(1:6)]
```

Find what data is correlated with one another and used a cutoff of 0.7 to determine the best correlation

```
corr_train <- cor(train_valid[, -53], use = "pairwise.complete.obs")
corr_high <- findCorrelation(corr_train, cutoff = 0.7)
#corrplot doesn't help much
corrplot(corr_train,order = "hclust" , method = "color" ,type = "upper",tl.col="black", tl.srt=45)
```



Train the necessary prediction models and compare the accuracy of the seperate prediction models to find the best model to use.

```
control <- trainControl(method = "cv", number = 3,verboseIter=FALSE)
metric <- "Accuracy"
```

```r
#Prediction models
#Random Forest
set.seed(13)
fit.rf <- train(classe~., data=train_valid, method="rf", metric=metric, trControl=control)
#Classification Tree
set.seed(13)
fit.ct <- train(classe~., data=train_valid, method="rpart", trControl=control)
#Linear Algorithms
set.seed(13)
fit.la <- train(classe~., data=train_valid, method="lda", metric = metric, trControl=control)

#Rpart
set.seed(13)
fit.cart <- train(classe~., data=train_valid, method="rpart", metric=metric, trControl=control)

#Gradient Boosting method
set.seed(13)
fit.gbm <- train(classe~., data=train_valid, method="gbm", metric=metric, trControl=control)
```

```
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1292
##      2        1.5227             nan     0.1000    0.0864
##      3        1.4654             nan     0.1000    0.0680
##      4        1.4207             nan     0.1000    0.0524
##      5        1.3864             nan     0.1000    0.0495
##      6        1.3539             nan     0.1000    0.0386
##      7        1.3282             nan     0.1000    0.0428
##      8        1.3018             nan     0.1000    0.0366
##      9        1.2787             nan     0.1000    0.0264
##     10        1.2603             nan     0.1000    0.0312
##     20        1.1076             nan     0.1000    0.0150
##     40        0.9337             nan     0.1000    0.0084
##     60        0.8267             nan     0.1000    0.0066
##     80        0.7474             nan     0.1000    0.0040
##    100        0.6847             nan     0.1000    0.0035
##    120        0.6314             nan     0.1000    0.0035
##    140        0.5882             nan     0.1000    0.0021
##    150        0.5683             nan     0.1000    0.0021
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1879
##      2        1.4881             nan     0.1000    0.1258
##      3        1.4037             nan     0.1000    0.1038
##      4        1.3381             nan     0.1000    0.0850
##      5        1.2844             nan     0.1000    0.0697
##      6        1.2392             nan     0.1000    0.0657
##      7        1.1966             nan     0.1000    0.0587
##      8        1.1588             nan     0.1000    0.0516
##      9        1.1248             nan     0.1000    0.0421
##     10        1.0974             nan     0.1000    0.0419
##     20        0.8953             nan     0.1000    0.0251
##     40        0.6864             nan     0.1000    0.0113
##     60        0.5567             nan     0.1000    0.0067
```

```
##     80        0.4716             nan     0.1000    0.0049
##    100        0.4051             nan     0.1000    0.0034
##    120        0.3499             nan     0.1000    0.0041
##    140        0.3046             nan     0.1000    0.0027
##    150        0.2854             nan     0.1000    0.0028
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.2276
##      2        1.4611             nan     0.1000    0.1557
##      3        1.3624             nan     0.1000    0.1239
##      4        1.2823             nan     0.1000    0.1022
##      5        1.2174             nan     0.1000    0.0875
##      6        1.1605             nan     0.1000    0.0713
##      7        1.1139             nan     0.1000    0.0784
##      8        1.0639             nan     0.1000    0.0543
##      9        1.0282             nan     0.1000    0.0590
##     10        0.9918             nan     0.1000    0.0499
##     20        0.7561             nan     0.1000    0.0225
##     40        0.5349             nan     0.1000    0.0093
##     60        0.4070             nan     0.1000    0.0072
##     80        0.3249             nan     0.1000    0.0072
##    100        0.2685             nan     0.1000    0.0023
##    120        0.2245             nan     0.1000    0.0018
##    140        0.1890             nan     0.1000    0.0022
##    150        0.1734             nan     0.1000    0.0010
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1285
##      2        1.5233             nan     0.1000    0.0897
##      3        1.4637             nan     0.1000    0.0661
##      4        1.4203             nan     0.1000    0.0533
##      5        1.3851             nan     0.1000    0.0475
##      6        1.3531             nan     0.1000    0.0475
##      7        1.3236             nan     0.1000    0.0390
##      8        1.2996             nan     0.1000    0.0361
##      9        1.2758             nan     0.1000    0.0282
##     10        1.2576             nan     0.1000    0.0347
##     20        1.0989             nan     0.1000    0.0165
##     40        0.9252             nan     0.1000    0.0097
##     60        0.8170             nan     0.1000    0.0055
##     80        0.7354             nan     0.1000    0.0028
##    100        0.6711             nan     0.1000    0.0057
##    120        0.6174             nan     0.1000    0.0036
##    140        0.5719             nan     0.1000    0.0017
##    150        0.5532             nan     0.1000    0.0018
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1897
##      2        1.4857             nan     0.1000    0.1254
##      3        1.4043             nan     0.1000    0.1059
##      4        1.3360             nan     0.1000    0.0855
##      5        1.2798             nan     0.1000    0.0728
##      6        1.2333             nan     0.1000    0.0571
##      7        1.1938             nan     0.1000    0.0522
```

```
##      8        1.1596             nan     0.1000    0.0610
##      9        1.1227             nan     0.1000    0.0483
##     10        1.0925             nan     0.1000    0.0496
##     20        0.8860             nan     0.1000    0.0250
##     40        0.6699             nan     0.1000    0.0100
##     60        0.5399             nan     0.1000    0.0070
##     80        0.4547             nan     0.1000    0.0045
##    100        0.3881             nan     0.1000    0.0033
##    120        0.3352             nan     0.1000    0.0028
##    140        0.2962             nan     0.1000    0.0013
##    150        0.2785             nan     0.1000    0.0020
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.2391
##      2        1.4565             nan     0.1000    0.1601
##      3        1.3540             nan     0.1000    0.1192
##      4        1.2759             nan     0.1000    0.1102
##      5        1.2062             nan     0.1000    0.0893
##      6        1.1498             nan     0.1000    0.0791
##      7        1.0988             nan     0.1000    0.0615
##      8        1.0595             nan     0.1000    0.0601
##      9        1.0202             nan     0.1000    0.0614
##     10        0.9810             nan     0.1000    0.0549
##     20        0.7441             nan     0.1000    0.0226
##     40        0.5190             nan     0.1000    0.0081
##     60        0.3987             nan     0.1000    0.0077
##     80        0.3122             nan     0.1000    0.0043
##    100        0.2597             nan     0.1000    0.0026
##    120        0.2142             nan     0.1000    0.0017
##    140        0.1805             nan     0.1000    0.0015
##    150        0.1670             nan     0.1000    0.0004
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1294
##      2        1.5216             nan     0.1000    0.0864
##      3        1.4613             nan     0.1000    0.0650
##      4        1.4174             nan     0.1000    0.0563
##      5        1.3811             nan     0.1000    0.0461
##      6        1.3499             nan     0.1000    0.0440
##      7        1.3224             nan     0.1000    0.0361
##      8        1.2978             nan     0.1000    0.0324
##      9        1.2765             nan     0.1000    0.0375
##     10        1.2526             nan     0.1000    0.0289
##     20        1.0978             nan     0.1000    0.0162
##     40        0.9236             nan     0.1000    0.0094
##     60        0.8167             nan     0.1000    0.0053
##     80        0.7360             nan     0.1000    0.0034
##    100        0.6758             nan     0.1000    0.0029
##    120        0.6239             nan     0.1000    0.0028
##    140        0.5798             nan     0.1000    0.0017
##    150        0.5601             nan     0.1000    0.0031
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.1818
```

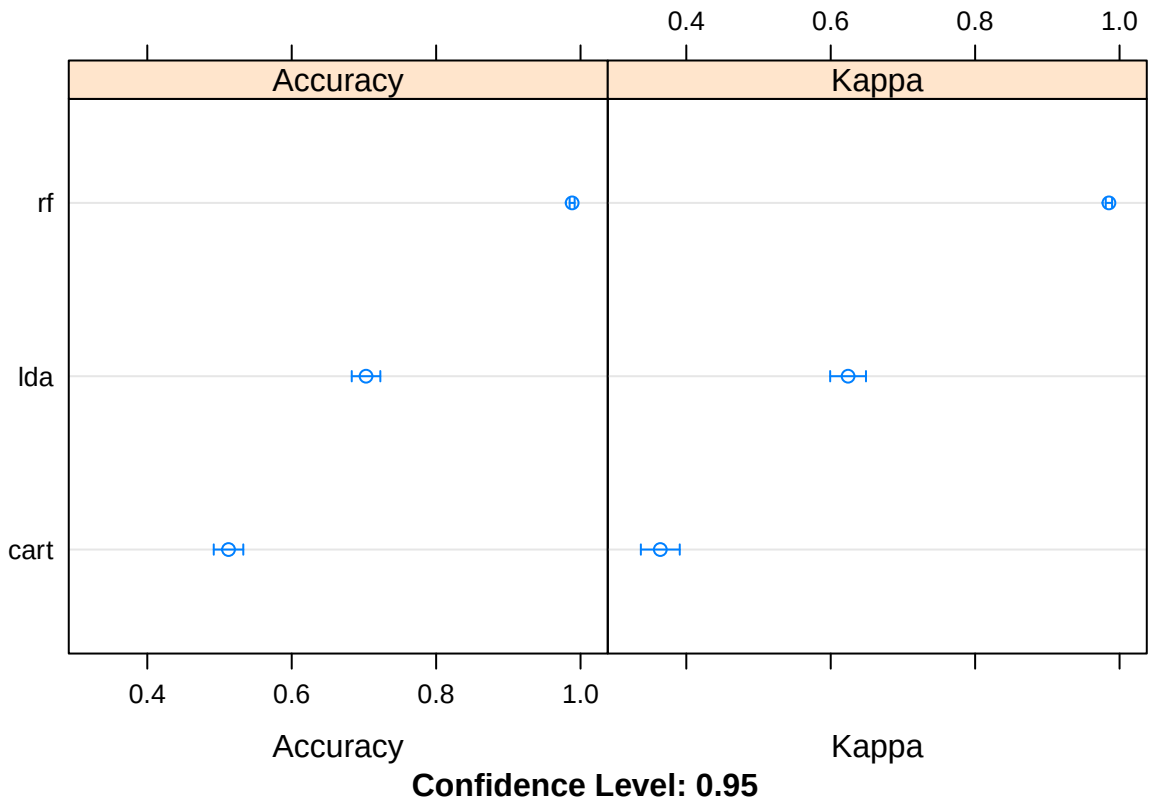
```
##      2        1.4862             nan     0.1000    0.1285
##      3        1.4021             nan     0.1000    0.1016
##      4        1.3354             nan     0.1000    0.0871
##      5        1.2811             nan     0.1000    0.0726
##      6        1.2349             nan     0.1000    0.0635
##      7        1.1940             nan     0.1000    0.0620
##      8        1.1543             nan     0.1000    0.0500
##      9        1.1219             nan     0.1000    0.0486
##     10        1.0898             nan     0.1000    0.0449
##     20        0.8901             nan     0.1000    0.0235
##     40        0.6715             nan     0.1000    0.0114
##     60        0.5503             nan     0.1000    0.0060
##     80        0.4632             nan     0.1000    0.0063
##    100        0.3979             nan     0.1000    0.0042
##    120        0.3440             nan     0.1000    0.0020
##    140        0.3019             nan     0.1000    0.0026
##    150        0.2824             nan     0.1000    0.0020
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.2309
##      2        1.4564             nan     0.1000    0.1648
##      3        1.3506             nan     0.1000    0.1196
##      4        1.2734             nan     0.1000    0.0985
##      5        1.2096             nan     0.1000    0.0826
##      6        1.1566             nan     0.1000    0.0858
##      7        1.1025             nan     0.1000    0.0656
##      8        1.0618             nan     0.1000    0.0682
##      9        1.0196             nan     0.1000    0.0604
##     10        0.9819             nan     0.1000    0.0480
##     20        0.7465             nan     0.1000    0.0241
##     40        0.5254             nan     0.1000    0.0084
##     60        0.4015             nan     0.1000    0.0088
##     80        0.3185             nan     0.1000    0.0029
##    100        0.2596             nan     0.1000    0.0029
##    120        0.2194             nan     0.1000    0.0028
##    140        0.1857             nan     0.1000    0.0010
##    150        0.1714             nan     0.1000    0.0021
##
## Iter   TrainDeviance   ValidDeviance   StepSize   Improve
##      1        1.6094             nan     0.1000    0.2308
##      2        1.4602             nan     0.1000    0.1601
##      3        1.3580             nan     0.1000    0.1278
##      4        1.2759             nan     0.1000    0.1007
##      5        1.2129             nan     0.1000    0.0920
##      6        1.1534             nan     0.1000    0.0801
##      7        1.1047             nan     0.1000    0.0736
##      8        1.0583             nan     0.1000    0.0608
##      9        1.0203             nan     0.1000    0.0614
##     10        0.9816             nan     0.1000    0.0487
##     20        0.7537             nan     0.1000    0.0240
##     40        0.5284             nan     0.1000    0.0143
##     60        0.4054             nan     0.1000    0.0074
##     80        0.3245             nan     0.1000    0.0050
##    100        0.2677             nan     0.1000    0.0023
```

```
##    120        0.2233             nan     0.1000    0.0023
##    140        0.1892             nan     0.1000    0.0017
##    150        0.1762             nan     0.1000    0.0016
```

```
results <- resamples(list(lda=fit.la, cart=fit.ct, rf=fit.rf))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, cart, rf
## Number of resamples: 3
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.6977506 0.6983294 0.6989083 0.7029201 0.7055048 0.7121014    0
## cart 0.5048056 0.5081355 0.5114654 0.5124833 0.5163222 0.5211790    0
## rf   0.9873362 0.9875533 0.9877703 0.9883528 0.9888611 0.9899519    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lda  0.6180592 0.6182738 0.6184884 0.6240335 0.6270206 0.6355529    0
## cart 0.3537108 0.3583166 0.3629224 0.3639809 0.3691160 0.3753096    0
## rf   0.9839781 0.9842527 0.9845274 0.9852643 0.9859074 0.9872874    0
```

```
dotplot(results)
```



We found that the random forest model is the most accurate according to the boxplot.

Use predict function to test the random forest model

```
end_result <- predict(fit.rf, newdata = test_data)
summary(end_result)
```

```
## A B C D E
## 7 8 1 1 3
```

```
end_result
```

```
##  [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
```