ML-markdown

diakt

3/9/2021

Description of Goals

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

"Executive" conclusion (what a pretentious name)

I built a random forest model on the data, after pruning down to 53 relevant features through removing NA data and irrelevant, misleading timestamp data. The main reason I used an ootb random forest model was the base bagging that removed the need for cross vaalidation to prevent over-fitting. I did prune the training m2m comparisons to five, which was about the only way I could employ a random forest on this absolute dinosaur of a machine I have. My expected OOSE was around 0.68%, which was validated by my observed prediction accuracy. I detail my choices pretty step-by-step, but I nixed a lot of features to end up with 53 ones for the model. I ended up with around 99 low accuracy on 99 kappa.

```
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(caret)
```

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

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

```
library(stringr)
library(parallel)
library(doParallel)
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

Data import

I import the data, and go 70/30 on a training and validation split. Standard caret things.

```
data <- tibble::as_tibble(read.csv("pml-training.csv"))

inTrain <- createDataPartition(data$classe, p=.7, list=FALSE)
training <- data[inTrain, ]
```

```
## Warning: The `i` argument of ``[`()` can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
validation <- data[-inTrain, ]

#maybe elim down to all that have no missing vals
# function(x) sum(is.na(x))/nrow(data)
#
```

Data analysis for inclusion

Not trying to feed garbage in. I list my adaptations:

1. Change all "character" quantitatives to numerics to recognize na's later
2. Eliminates all columns with less than 99% real values, though most NA columns have less than 2% data
3. Eliminate first seven columns that deal with time stamps (can massively bias model and I didn't want to take a chance on this being nonreproducible on the quiz)

I am left with 53 features.

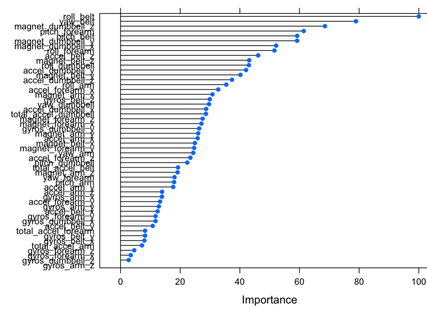
```
#removes any non-quantitative character vars
quant2Num <- function(x) {class(x) <- "numeric";x }
suppressWarnings(final <- training %>% mutate_if(sapply(., is.character) & !(str_detect(colnames(.), "user_name|classe|cvtd_timestamp|new_window")), quant2Num) %>% select(which(colMeans(!is.na(.)) > 0.99)) %>% select(!(1:7)))
```

Model training after setting up parallel cores

```
#testing phase
set.seed(42069)

#parallel setup
cluster <- makeCluster(detectCores() - 2) #call me a baby, this thing is ancient, wanted to be safe
registerDoParallel(cluster)
fitRepetitions <- trainControl(method = "cv", number = 5, allowParallel = TRUE)

#fitting model
model <- train(classe~., data=final, method="rf", trainControl=fitRepetitions)

# 17 minutes. ending core regulation
stopCluster(cluster)

#Analysis
model$finalModel[5]
```

```
## $confusion
##      A    B    C    D    E  class.error
## A 3903    2    0    0    1 0.0007680492
## B   12 2641    5    0    0 0.0063957863
## C    0   21 2374    1    0 0.0091819699
## D    0    0   35 2215    2 0.0164298401
## E    0    0    1    7 2517 0.0031683168
```

```
modelFactors <- varImp(model)
plot(modelFactors)
```



Took a decently long time, around 17 minutes.

Predictions on validation set

```
#Now we predict our validation data

prediction <- predict(model, validation)
confusionMatrix(prediction, as.factor(validation$classe))$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    2    0    0    0
##          B    1 1135    2    0    0
##          C    0    2 1024    5    0
##          D    0    0    0  959    0
##          E    0    0    0    0 1082
```

```
confusionMatrix(prediction, as.factor(validation$classe))$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9979609      0.9974288      0.9964408      0.9989459      0.2844520
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

Ending conclusions

I was pretty happy with the outcome. Onto the testing set. Wish me luck.