Competition

uông Thị Thanh Thủy - Trần Gia Bảo - Hoàng Thị Cẩm Tú - Lê Kha

2021/01/23

load packages

```
library(ggthemes)
library(ggplot2)
library(caret)
```

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

```
library(randomForest)
```

```
## 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
```

```
library(e1071)
library(grid)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

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

```
library(mice)
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(dplyr)
```

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

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:randomForest':
##
##     combine

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

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

```
library(pscl)
```

```
## Classes and Methods for R developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University
## Simon Jackman
## hurdle and zeroinfl functions by Achim Zeileis
```

```
library(caretEnsemble)
```

```
##
## Attaching package: 'caretEnsemble'

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

```
#load data

train <- read.csv('C:/Users/uongt/OneDrive/Desktop/Competition/train.csv',sep="|",stringsAsFactors = F)
test  <- read.csv('C:/Users/uongt/OneDrive/Desktop/Competition/test.csv',sep="|",stringsAsFactors = F)
```

set random seed for model reproducibility make fraud a factor number 0 is trusty and number 1 is fraud

```
set.seed(100)
train$fraud <- as.factor(train$fraud)
levels(train$fraud) <- c("trusty","fraud")
```

The function trainControl can be used to specifiy the type of resampling In the code above, 10-fold CV mean dividing your training dataset randomly into 10 parts and then using each of 10 parts as testing dataset for the model trained on other 9. We take the average of the 10 error terms thus obtained.

In 3 repeats of 10 fold CV, we'll perform the average of 3 error terms obtained by performing 10 fold CV five times. Important thing to note is that 3 repeats of 10 fold CV is not same as 30 fold CV.

Caret Ensemble allows the user to train multiple models by using the caret List function. The only drawback is the computing time this might take. Models can also be combined to utilize the caret stack function to make better predictions.

```
# prepare training scheme
#Stacking Algorithms - Run multiple algos in one call
trainControl <- trainControl(method="repeatedcv",
                             number=10,
                             repeats=3,
                             savePredictions=TRUE,
                             classProbs=TRUE)
```

I start by parallelizing to decrease the speed it takes to train multiple models. I also created a train control using repeatedcv. The models being fitted were Random Forest,xgbDart, and svmRadial. The caretlist function is similar to the train function in the caret package.

```
algorithmList <- c('rf','xgbDART', 'svmRadial')
```

```
set.seed(100)
models <- caretList(fraud~., data=train, trControl=trainControl, methodList=algorithmList)
```

```
## Warning in trControlCheck(x = trControl, y = target): x$savePredictions == TRUE
## is depreciated. Setting to 'final' instead.

## Warning in trControlCheck(x = trControl, y = target): indexes not defined in
## trControl. Attempting to set them ourselves, so each model in the ensemble will
## have the same resampling indexes.
```

```
results <- resamples(models)
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: rf, xgbDART, svmRadial
## Number of resamples: 30
##
## Accuracy
```

```
##                Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf        0.9411765 0.9679571 0.9786665 0.9756937 0.9840426 0.9946524    0
## xgbDART   0.9625668 0.9734394 0.9839572 0.9815496 0.9880530 0.9947090    0
## svmRadial 0.9255319 0.9415668 0.9468085 0.9473207 0.9521277 0.9734043    0
##
## Kappa
##                  Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## rf         0.13243357 0.6502169 0.7868336 0.7321899 0.8339678 0.9445597    0
## xgbDART    0.60933148 0.7671973 0.8245621 0.8078138 0.8747209 0.9495866    0
## svmRadial -0.03134796 0.2632302 0.3646090 0.3530717 0.4177959 0.6924084    0
```

```
save(models,file="models_final.RData")
```

Combine the predictions of models to form final prediction Create the trainControl

```
set.seed(101)
stackControl <- trainControl(method="repeatedcv",
                             number=10,
                             repeats=3,
                             savePredictions=TRUE,
                             classProbs=TRUE)
```

Ensemble the predictions of `models` to form a new combined prediction based on glm

```
stack.glm <- caretStack(models, method="glm", metric="Accuracy", trControl=stackControl)
save(stack.glm,file="combined_predictions.RData")
print(stack.glm)
```

```
## A glm ensemble of 3 base models: rf, xgbDART, svmRadial
##
## Ensemble results:
## Generalized Linear Model
##
## 5637 samples
##    3 predictor
##    2 classes: 'trusty', 'fraud'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 5073, 5073, 5074, 5073, 5074, 5073, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9795408  0.7906959
```

The three models gives an accuracy of 98%.

Predict on testData

```
stack_predicteds <- predict(stack.glm, newdata=test)
head(stack_predicteds)
```

```
## [1] trusty trusty trusty trusty trusty trusty
## Levels: trusty fraud
```

```
save.image("script.RData")
```

trusty is number 0 and fraud is number 1 make stack_predicteds a data frame call name col is fraud

```
levels(stack_predicteds) <- c(0,1)
stack_predicteds <- data.frame(stack_predicteds)
names(stack_predicteds)<-"fraud"
write.csv(stack_predicteds,file="C:/Users/uongt/OneDrive/Desktop/Competition/stack_predicteds.csv",row.
```