Ensemble Methods Notebook

Francisco Trejo

Clean and Split Data

```
df <- read.csv("Training_set_advance.csv", header=TRUE)
df <- df[,c(2,5,6,8,17,18)]
df <- df[complete.cases(df[, 1:6]),]
df$Diagnosed_Condition <- factor(df$Diagnosed_Condition)
df$Survived_1_year <- factor(df$Survived_1_year)
df$Patient_Rural_Urban <- factor(df$Patient_Rural_Urban)
sapply(df, function(x) sum(is.na(x)==TRUE))
```

```
##     Diagnosed_Condition             Patient_Age Patient_Body_Mass_Index 
##                       0                       0                       0 
##     Patient_Rural_Urban     Number_of_prev_cond         Survived_1_year 
##                       0                       0                       0
```

```
str(df)
```

```
## 'data.frame':    23723 obs. of  6 variables:
##  $ Diagnosed_Condition    : Factor w/ 53 levels "0","1","2","3",..: 48 4 8 32 44 52 50 36 37
16 ...
##  $ Patient_Age            : int  60 2 20 8 53 20 5 45 43 60 ...
##  $ Patient_Body_Mass_Index: num  21.7 28.9 26.2 22.6 21.3 ...
##  $ Patient_Rural_Urban    : Factor w/ 2 levels "RURAL","URBAN": 2 1 1 1 1 1 1 2 1 1 ...
##  $ Number_of_prev_cond    : num  2 3 2 2 1 2 2 1 2 3 ...
##  $ Survived_1_year        : Factor w/ 2 levels "0","1": 2 1 2 1 2 2 2 2 1 1 ...
```

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Logistic Regression

```
library(mltools)
```

```
## Warning: package 'mltools' was built under R version 4.2.3
```

```
start_time <- Sys.time()
glm1 <- glm(Survived_1_year ~ Diagnosed_Condition, data=train, family="binomial")
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 1.862873 secs
```

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 1, 0)
acc_logreg <- mean(pred==as.integer(test$Survived_1_year))
mcc_logreg <- mcc(pred, as.integer(test$Survived_1_year))
print(paste("accuracy=", acc_logreg))
```

```
## [1] "accuracy= 0.19536354056902"
```

```
print(paste("mcc=", mcc_logreg))
```

```
## [1] "mcc= -0.126331273101353"
```

Random Forest

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.2.3
```

```
## randomForest 4.7-1.1
```

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

```
set.seed(1234)
start_time <- Sys.time()
rf <- randomForest(Survived_1_year~., data=train, importance=TRUE)
end_time <- Sys.time()
rf
```

```
##
## Call:
##  randomForest(formula = Survived_1_year ~ ., data = train, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 25.45%
## Confusion matrix:
##      0     1 class.error
## 0 3680  3440   0.4831461
## 1 1389 10469   0.1171361
```

```
pred <- predict(rf, newdata=test, type="response")
acc_rf <- mean(pred==test$Survived_1_year)
mcc_rf <- mcc(factor(pred), test$Survived_1_year)
end_time - start_time
```

```
## Time difference of 37.71599 secs
```

```
print(paste("accuracy=", acc_rf))
```

```
## [1] "accuracy= 0.747312961011591"
```

```
print(paste("mcc=", mcc_rf))
```

```
## [1] "mcc= 0.437222173552104"
```

AdaBoost

```
library(adabag)
```

```
## Warning: package 'adabag' was built under R version 4.2.3
```

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

```
## Loading required package: caret
```

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

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

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

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

```
## Loading required package: doParallel
```

```
## Warning: package 'doParallel' was built under R version 4.2.3
```

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

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

```
start_time <- Sys.time()
adab1 <- boosting(Survived_1_year~., data=train, boos=TRUE, mfinal=20, coeflearn='Breiman')
end_time <- Sys.time()
summary(adab1)
```

```
##            Length Class   Mode     
## formula        3  formula call     
## trees         20  -none-  list     
## weights       20  -none-  numeric  
## votes      37956  -none-  numeric  
## prob       37956  -none-  numeric  
## class      18978  -none-  character
## importance     5  -none-  numeric  
## terms          3  terms   call     
## call           6  -none-  call
```

```
pred <- predict(adab1, newdata=test, type="response")
acc_adabag <- mean(pred$class==test$Survived_1_year)
mcc_adabag <- mcc(factor(pred$class), test$Survived_1_year)
end_time - start_time
```

```
## Time difference of 18.99528 secs
```

```
print(paste("accuracy=", acc_adabag))
```

```
## [1] "accuracy= 0.726870389884088"
```

```
print(paste("mcc=", mcc_adabag))
```

```
## [1] "mcc= 0.388401422861786"
```

Summary

The data set used had to do with if someone survived one year of treatment and using predictors like their body max index, age, and number of previous conditions. I ran Logistic Regression on the data set to see how accurately it predicts someone surviving based on the predictors. I got a really low number of accuracy of .19 and a coefficient of -.12. The algorithm was fast and only took 2.49 seconds to complete. Now using the ensemble methods it had a significant change in both speed and accuracy. Random Forest was used first and got an accuracy .74 with a .43 coefficient. It was a significant change, but it took longer to run at 39 seconds. Then Adaboost was used and there wasn’t must change compared to Random Forest. It was faster at 24 seconds, but we lost some accuracy at .72 and the coefficient also decreased to .38. In all, it shows the ensemble methods do help with increasing accuracy, but there is a trade off when it comes to speed.