CS-GY 6923 Machine Learning

Professor: Dr. Raman Kannan

HW3: Combining Individual Classifiers

Anil Poonai

Due 5th December 2021

Contents

For homework assignment three, I used ensemble methods: cross validation of a logistic regression algorithm, bagging, and boosting This was to compare them to single individual classifiers used in the previous homework assignment.

1 Cross Validation

Cross-validation is used to make the model independent of the training data, it divides the data into a certain number of folds and keeps on sampling the data, with one group being left out as a test group within the model generation, not the end results. The methods cross-validation used was logit boost, which is a boosting algorithm with a logistic regression applied on a linear model.

The formula for logit boost is:

$$\sum_i \log\left(1 + e^{-y_i f(x_i)}\right)$$

The code used is below:

```
#CrossValidation
library(caTools)
set.seed(223)
train.control <- trainControl(method = "cv", number = 10)
CV <- train(Dependent ~., data = train, method = "LogitBoost", trControl = train.control)
CV
cvpred = predict(CV, test, decision.values = TRUE, type ='raw')
confusionMatrix(cvpred, test$Dependent)
#Bagging
set.seed(224)
library(ipred)
bag = bagging(Dependent ~ ., data = train,nbagg = 150,coob = TRUE)
bagpred = predict(bag, test, decision.values = TRUE, type ='class')
confusionMatrix(as.factor(bagpred$class), test$Dependent)
confusionMatrix(bagpred, test$Dependent)
```

The results are:

```
> CV <- train(Dependent ~., data = train, method = "LogitBoost", trControl = tra
in.control)
> CV
Boosted Logistic Regression

51701 samples
   17 predictor
    3 classes: 'FELONY', 'MISDEMEANOR', 'VIOLATION'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 46532, 46531, 46531, 46531, 46530, 46531, ...
Resampling results across tuning parameters:

  nIter  Accuracy   Kappa
  11     0.5717458  0.2601585
  21     0.5632962  0.2616334
  31     0.5620960  0.2347768

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was nIter = 11.
```

```
> confusionMatrix(cvpred, test$Dependent)
Confusion Matrix and Statistics

             Reference
Prediction     FELONY MISDEMEANOR VIOLATION
  FELONY         146          24        17
  MISDEMEANOR   1199        1978       229
  VIOLATION     2816        2861      7272

Overall Statistics

               Accuracy : 0.568
                 95% CI : (0.5604, 0.5756)
    No Information Rate : 0.4545
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2563

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: FELONY Class: MISDEMEANOR Class: VIOLATION
Sensitivity               0.035088             0.4067           0.9673
Specificity               0.996688             0.8777           0.3709
Pos Pred Value            0.780749             0.5807           0.5616
Neg Pred Value            0.754509             0.7804           0.9315
Prevalence                0.251542             0.2940           0.4545
Detection Rate            0.008826             0.1196           0.4396
Detection Prevalence      0.011305             0.2059           0.7828
Balanced Accuracy         0.515888             0.6422           0.6691
```

2 Bagging

Bagging is an ensemble method also knows as bootstrap aggregating, weak learners are trained in parallel and typically have high variance and low bias.

The code used is below:

```r
library(ipred)
bag = bagging(Dependent ~ ., data = train,nbagg = 150,coob = TRUE)
bagpred = predict(bag, test, decision.values = TRUE, type ='class')
confusionMatrix(as.factor(bagpred$class), test$Dependent)
confusionMatrix(bagpred, test$Dependent)
```

The results are:

```
> confusionMatrix(bagpred, test$Dependent)
Confusion Matrix and Statistics

             Reference
Prediction    FELONY MISDEMEANOR VIOLATION
  FELONY        2713        1701      1185
  MISDEMEANOR   1601        2509      1053
  VIOLATION     2217        2490      6727

Overall Statistics

               Accuracy : 0.5383
                 95% CI : (0.5318, 0.5449)
    No Information Rate : 0.4039
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.287

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: FELONY Class: MISDEMEANOR Class: VIOLATION
Sensitivity                 0.4154             0.3745           0.7504
Specificity                 0.8158             0.8287           0.6442
Pos Pred Value              0.4846             0.4860           0.5883
Neg Pred Value              0.7700             0.7539           0.7920
Prevalence                  0.2942             0.3019           0.4039
Detection Rate              0.1222             0.1130           0.3031
Detection Prevalence        0.2523             0.2326           0.5151
Balanced Accuracy           0.6156             0.6016           0.6973
```

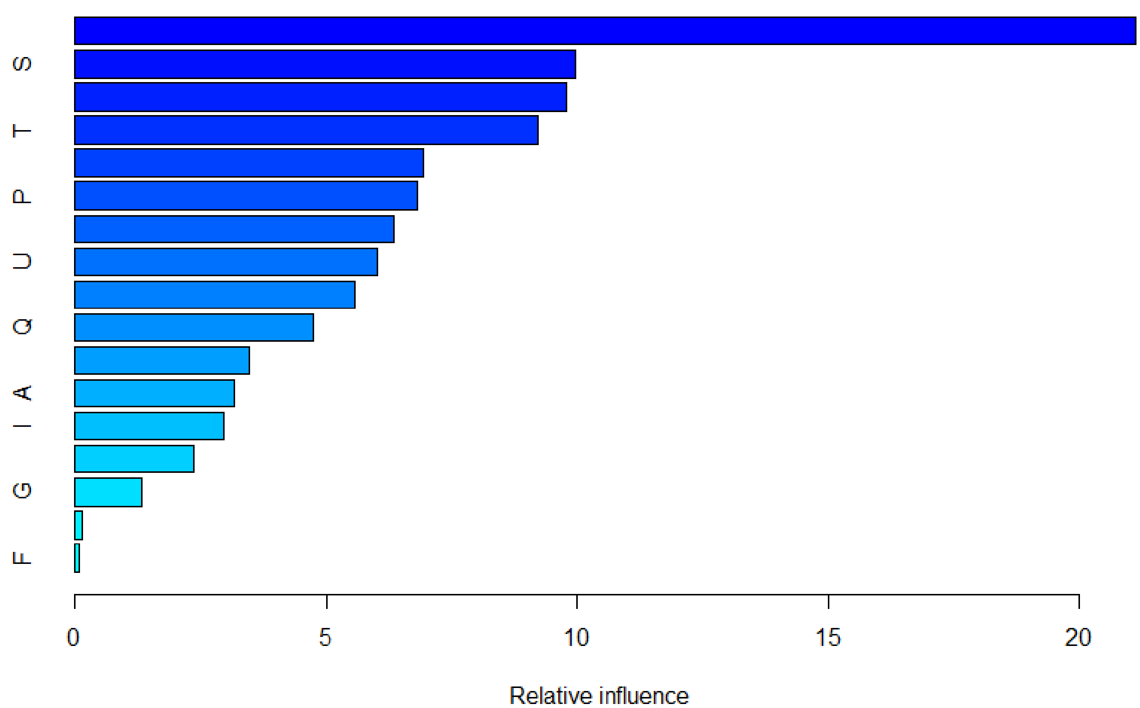
3 Boosting

Boosting is an ensemble method also knows as bootstrap aggregating, weak learners are trained sequentially, this incorporates the weighted values of other learners into the next learner. They typically have low variance and high bias.

The code used is below:

```
#Boosting
set.seed(225)
library(gbm)
boost=gbm(Dependent ~ ., data = train,distribution = "multinomial",n.trees = 10000, shrinkage = 0.01, interaction.
summary(boost)
boostpred = predict(boost, test, decision.values = TRUE, type ='response')
boostpred[1:6,,]
p.boostpred = apply(boostpred,1,which.max)
# 1 = Felony, 2 = Misdemeanor, 3 = Violation
head(p.boostpred)
str(p.boostpred)
boostdf = data.frame(p.boostpred)
boostdf$predictions = as.factor(ifelse(boostdf$p.boostpred == 1, "FELONY",
                                ifelse(boostdf$p.boostpred == 2, "MISDEMEANOR","VIOLATION")))
confusionMatrix(boostdf$predictions, test$Dependent)
```

The graph or variable importance given was:



The results are:

```
> confusionMatrix(boostdf$predictions, test$Dependent)
Confusion Matrix and Statistics

             Reference
Prediction    FELONY MISDEMEANOR VIOLATION
  FELONY        1435         753       252
  MISDEMEANOR   1259        2050       129
  VIOLATION     3837        3897      8584

Overall Statistics

               Accuracy : 0.5437
                 95% CI : (0.5372, 0.5503)
    No Information Rate : 0.4039
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2688

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: FELONY Class: MISDEMEANOR Class: VIOLATION
Sensitivity                0.21972            0.30597           0.9575
Specificity                0.93584            0.91043           0.4155
Pos Pred Value             0.58811            0.59628           0.5260
Neg Pred Value             0.74205            0.75211           0.9352
Prevalence                 0.29424            0.30186           0.4039
Detection Rate             0.06465            0.09236           0.3867
Detection Prevalence       0.10993            0.15489           0.7352
Balanced Accuracy          0.57778            0.60820           0.6865
```

4 Results/Recommendations

In general, the ensemble methods have a higher accuracy compared to the single individual classifiers, with the lowest ensemble method being more accurate than the best single classifier, in general ensemble methods ae probably more accurate except for when data follows one classifier model heavily. Cross validation makes the data have low bias but does make the variance go up as the training sets aren't going to be stable and the same. I would pick the cross-validation model using logit boost in the end as it has the highest accuracy with 56.8%.

5 Code

```
#Load the data

library(readr)

library(plyr)

library(dplyr)

library(tidyverse)

library(usethis)

library(devtools)

#install_github("vqv/ggbiplot", force=TRUE)

library(grid)

library(ggbiplot)

library(ggplot2)

library(lattice)

library(caret)

library(pROC)

#From IBM Terminal

df <-
read_csv("/home/2021/nyu/fall/ap5254/hw01/NYPD_Complaint_Data_Current__Year_To_Date_.csv")

#If you have the file

#df <- read_csv("C:/Users/poona/Downloads/NYPD_Complaint_Data_Current__Year_To_Date_.csv")

#If you have the link

#df = read.csv(url("https://data.cityofnewyork.us/api/views/5uac-
w243/rows.csv?accessType=DOWNLOAD"))

#Check dimensions

dim(df)


#Change names to numbers to help reduce bias

names(df) = c(1:36)

names(df)
```

```
#Label the dependent variable
names(df)[14] = 'Dependent'
names(df)

head(df)

ggplot(data = df) +
  geom_bar(mapping = aes(Dependent))

#Get rid of Identifier
df = df[-c(1)]

#Data Types of each column
str(df)

#Removing columns that have missing values summing at least half of the total amount of observations
colSums(is.na(df))
nrow(df)/2
df = df[-c(5,6,8,9,16,22,26)]

#Removing columns that are a description of another column

df = df[-c(8,11,14)]
str(df)

#Imbalance check
sum(df$Dependent=='FELONY')/nrow(df)
sum(df$Dependent=='MISDEMEANOR')/nrow(df)
```

```
sum(df$Dependent=='VIOLATION')/nrow(df)
#Imbalance check
sum(df$Dependent=='FELONY')
sum(df$Dependent=='MISDEMEANOR')
sum(df$Dependent=='VIOLATION')
dff = df[df[, "Dependent"] == 'FELONY',]
dfm = df[df[, "Dependent"] == 'MISDEMEANOR',]
dfv = df[df[, "Dependent"] == 'VIOLATION',]
dff = dff[sample(nrow(dff), 34552), ]
dfm = dfm[sample(nrow(dfm), 34552), ]
df = rbind(dff,dfm,dfv)

ggplot(data = df) +
  geom_bar(mapping = aes(Dependent))

df = df[-c(24,25)]

#randomForest and Variable Importance
library(randomForest)
df$Dependent = as.factor(df$Dependent)
table(df$Dependent)
colnames(df) = c('A','B','C','D','E','F','G','Dependent','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V')

df$B = as.factor(df$B)
df$C = as.factor(df$C)
df$E = as.factor(df$E)
df$G = as.factor(df$G)
df$H = as.factor(df$H)
df$I = as.factor(df$I)
```

```
df$K = as.factor(df$K)
df$L = as.factor(df$L)
df$M = as.factor(df$M)
df$N = as.factor(df$N)
df$O = as.factor(df$O)
df$P = as.factor(df$P)
df$Q = as.factor(df$Q)
df$R = as.factor(df$R)

df$C = NULL
df$D = NULL
df$K = NULL
df$L = NULL

sum(complete.cases(df)=='TRUE')
df=df[complete.cases(df),]

set.seed(222)
index = sample(2, nrow(df), replace = TRUE, prob = c(0.7,0.3))
train = df[index==1,]
test = df[index==2,]
str(train)
summary(train)

train$J = NULL
test$J = NULL

str(df)
#VIF
```

```
library(car)
m = lm(A~F+S+T+U+V, data=df)
vif(m)

library(doParallel)

ncores = detectCores(logical = TRUE)
ncores
cl = makePSOCKcluster(5)
registerDoParallel(cl)

start.time = proc.time()
#CrossValidation
library(caTools)
set.seed(223)
train.control <- trainControl(method = "cv", number = 10)
CV <- train(Dependent ~., data = train, method = "LogitBoost", trControl = train.control)
CV
cvpred = predict(CV, test, decision.values = TRUE, type ='raw')
confusionMatrix(cvpred, test$Dependent)
#Bagging
set.seed(224)
library(ipred)
bag = bagging(Dependent ~ ., data = train,nbagg = 150,coob = TRUE)
bagpred = predict(bag, test, decision.values = TRUE, type ='class')
confusionMatrix(as.factor(bagpred$class), test$Dependent)
confusionMatrix(bagpred, test$Dependent)
#Boosting
set.seed(225)
```

```
library(gbm)

boost=gbm(Dependent ~ ., data = train,distribution = "multinomial",n.trees = 10000, shrinkage = 0.01,
interaction.depth = 4)

summary(boost)

boostpred = predict(boost, test, decision.values = TRUE, type ='response')

boostpred[1:6,,]

p.boostpred = apply(boostpred,1,which.max)

# 1 = Felony, 2 = Misdemeanor, 3 = Violation

head(p.boostpred)

str(p.boostpred)

boostdf = data.frame(p.boostpred)

boostdf$predictions = as.factor(ifelse(boostdf$p.boostpred == 1, "FELONY",

                        ifelse(boostdf$p.boostpred == 2, "MISDEMEANOR","VIOLATION")))

confusionMatrix(boostdf$predictions, test$Dependent)

stopCluster(cl)

stop.time = proc.time()

run.time = start.time = stop.time

run.time
```