XGBoost

Using the library xgboost and the method as “xgbTree”, I ran the XGBoost model on the training data, calculated the importance of the independent variables involved and used the fitted model to make predictions.

##Libraries--------------------------------------------  
library(tree)  
library(ISLR)  
library(boot)  
library(xgboost)  
library(tidyverse)

## ── Attaching packages ───────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.1.0 ✔ purrr 0.3.0  
## ✔ tibble 2.0.1 ✔ dplyr 0.7.8  
## ✔ tidyr 0.8.2 ✔ stringr 1.4.0  
## ✔ readr 1.3.1 ✔ forcats 0.3.0

## ── Conflicts ──────────────────────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ dplyr::slice() masks xgboost::slice()

library(leaflet)  
library(stringr)  
library(rgdal)

## Loading required package: sp

## rgdal: version: 1.4-3, (SVN revision 828)  
## Geospatial Data Abstraction Library extensions to R successfully loaded  
## Loaded GDAL runtime: GDAL 2.1.3, released 2017/20/01  
## Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/3.5/Resources/library/rgdal/gdal  
## GDAL binary built with GEOS: FALSE   
## Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ\_VERSION: 493]  
## Path to PROJ.4 shared files: /Library/Frameworks/R.framework/Versions/3.5/Resources/library/rgdal/proj  
## Linking to sp version: 1.3-1

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

library(forecast)  
library(DT)  
library(prophet)

## Loading required package: Rcpp

## Loading required package: rlang

##   
## Attaching package: 'rlang'

## The following objects are masked from 'package:purrr':  
##   
## %@%, as\_function, flatten, flatten\_chr, flatten\_dbl,  
## flatten\_int, flatten\_lgl, flatten\_raw, invoke, list\_along,  
## modify, prepend, splice

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'lattice'

## The following object is masked from 'package:boot':  
##   
## melanoma

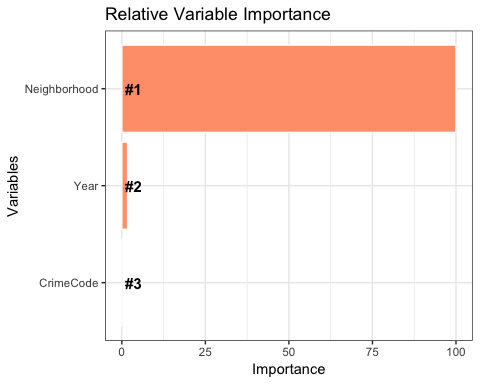
##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

##Read the data  
mergedf <- read.csv("~/Desktop/GitAdd/Data\_Mining/Files/Outputs/CrimeRentData.csv")  
  
set.seed(12345)  
LACrimeDataSet = mergedf %>% select(Year, CrimeCode, Amount,Neighborhood)  
  
LACrimeDataSet = na.omit(LACrimeDataSet)  
  
mergedf<-LACrimeDataSet  
  
##Create Amount categories  
mergedf$Amount <- as.numeric(mergedf$Amount)  
  
ra <- range(mergedf$Amount)  
div <- (ra[2]-ra[1])/10  
ini <- ra[1]  
br <- rep(0,11)  
br[1]<-ra[1]  
for(i in 2:11){  
 ini<-ini+div  
 br[i]<-ini  
}  
  
mergedf$Renth <- cut(mergedf$Amount,   
 breaks=br,   
 labels=c("1","2","3","4","5","6","7","8","9","10"))  
  
mergedf <- within(mergedf,rm("Amount"))  
  
LACrimeDataSet2 = mergedf %>% select(-Renth)  
  
features <- colnames(LACrimeDataSet2)  
  
for (f in features) {  
 if ((class(LACrimeDataSet2[[f]])=="factor") || (class(LACrimeDataSet2[[f]])=="character")) {  
 levels <- unique(LACrimeDataSet2[[f]])  
 LACrimeDataSet2[[f]] <- as.numeric(factor(LACrimeDataSet2[[f]], levels=levels))  
 }  
}  
  
  
LACrimeDataSet2$Renth = as.factor(mergedf$Renth)  
LACrimeDataSet2 <- na.omit(LACrimeDataSet2)  
levels(LACrimeDataSet2$Renth) = make.names(unique(LACrimeDataSet2$Renth))  
  
  
  
library(caret)  
  
attach(LACrimeDataSet2)  
  
#Sample  
train <- sample(nrow(LACrimeDataSet2),0.7\*nrow(LACrimeDataSet2))  
traindata <- data.frame(LACrimeDataSet2[train,])  
testdata <- data.frame(LACrimeDataSet2[-train,])  
  
formula = Renth ~ .  
  
fitControl <- trainControl(method="cv",number = 3,classProbs = TRUE,summaryFunction=mnLogLoss)  
  
xgbGrid <- expand.grid(nrounds = 3,  
 max\_depth = 3,  
 eta = .05,  
 gamma = 0,  
 colsample\_bytree = .8,  
 min\_child\_weight = 1,  
 subsample = 1)  
  
set.seed(13)  
  
traindata$Renth<- factor(traindata$Renth)  
  
LACrimeXGB = train(data = traindata, formula,  
 method = "xgbTree",trControl = fitControl,  
 tuneGrid = xgbGrid,na.action = na.omit,  
 objective='multi:softprob',metric = "logLoss",  
 num\_class=length(unique(traindata$Renth)))

## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:  
## num\_class, objective  
## Only the last value for each of them will be used.  
  
## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:  
## num\_class, objective  
## Only the last value for each of them will be used.  
  
## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:  
## num\_class, objective  
## Only the last value for each of them will be used.  
  
## Warning in check.booster.params(params, ...): The following parameters were provided multiple times:  
## num\_class, objective  
## Only the last value for each of them will be used.

importance = varImp(LACrimeXGB)  
  
varImportance <- data.frame(Variables = row.names(importance[[1]]),   
 Importance = round(importance[[1]]$Overall,2))  
  
  
# Create a rank variable based on importance  
rankImportance <- varImportance %>%  
 mutate(Rank = paste0('#',dense\_rank(desc(Importance)))) %>%  
 head(10)  
  
rankImportancefull = rankImportance  
  
library(ggplot2)  
fillColor = "#FFA07A"  
ggplot(rankImportance, aes(x = reorder(Variables, Importance),   
 y = Importance)) +  
 geom\_bar(stat='identity',colour="white", fill = fillColor) +  
 geom\_text(aes(x = Variables, y = 1, label = Rank),  
 hjust=0, vjust=.5, size = 4, colour = 'black',  
 fontface = 'bold') +  
 labs(x = 'Variables', title = 'Relative Variable Importance') +  
 coord\_flip() +   
 theme\_bw()



##Testing-------------------------------  
pred <- predict(LACrimeXGB, newdata=testdata)  
  
table1<-table(testdata$Renth,pred)  
  
##Accuracy  
sum=0  
for(i in 1:nrow(table1)){  
   
 sum=sum+table1[i,i]  
}  
  
(acc= sum/nrow(testdata))

## [1] 0.5325255

No\_of\_Records<-rep(0,10)  
for(i in 1:10){  
 No\_of\_Records[i]<- table1[i,i]  
}  
  
plot(1:10,No\_of\_Records)

