Gov 2018: Lab 5 Random Forests

Your name:

February 22, 2022

This exercise is based off of Muchlinski, David, David Siroky, Jingrui He, and Matthew Kocher. 2016. "Comparing Random Forest with Logistic Regression for Predicting Class-Imbalanced Civil War Onset Data". *Political Analysis*.

Descriptions of the relevant variables in the data file `data_full.rds` are:

Name	Description
`warstds`	Factor, `peace` and `war`
`year`	Numeric for year of obs

And a list of 90 covariates from the Sambanis dataset: "ager", "agexp", "anoc", "army85", "autch98", "auto4", "autonomy", "avgnabo", "centpol3", "coldwar", "decade1", "decade2", "decade3", "decade4", "dem", "dem4", "demch98", "dlang", "drel", "durable", "ef", "ef2", "ehet", "elfo", "elfo2", "etdo4590", "expgdp", "exrec", "fedpol3", "fuelexp", "gdpgrowth", "geo1", "geo2", "geo34", "geo57", "geo69", "geo8", "illiteracy", "incumb", "infant", "inst", "inst3", "life", "lmtnest", "ln_gdpen", "lpopns", "major", "manuexp", "milper", "mirps0", "mirps1", "mirps2", "mirps3", "nat_war", "ncontig", "nmgdp", "nmdp4_alt", "numlang", "nwstate", "oil", "p4mchg", "parcomp", "parreg", "part", "partfree", "plural", "plurrel", "pol4", "pol4m", "pol4sq", "polch98", "polcomp", "popdense", "presi", "pri", "proxregc", "ptime", "reg", "regd4_alt", "relfrac", "seceduc", "second", "semipol3", "sip2", "sxpnew", "sxpsq", "tnatwar", "trade", "warhist", "xconst".

We're going to use the cross-validation function from the `caret` package. Set aside the years 1999 and 2000 for testing data.

```
# caret::trainControl() controls parameters for train
tc<-caret::trainControl(method="cv", # the resampling method
                        number=10, # the number of folds
                        summaryFunction=twoClassSummary, # a function to compute performance metrics ac
                                                         # twoClassSummary computes sensitivity, specif
                        classProb=T, # class probabilities be computed for classification models
                                     # (along with predicted values) in each resample
                        savePredictions = T)

# Set train data
data.train<-subset(data.full,year<1999)
```

Question 1

We're going to compare several model specifications using classic/penalized logistic regressions with a random forest model.

(a) The Fearon & Laitin model (2003) "FL" can be described as the following:

```
as.factor(warstds) ~ warhist + ln_gdpen + lpopns + lmtnest + ncontig + oil + nwstate +
```

`inst3 + pol4 + ef + relfrac`

Please run the `train` function in the `caret` library with metric as `ROC`, method as `glm`, family as `binomial` (FL used logistic), trControl as our set `tc`, and your training data, on the above specification. Do the same for a penalized logistic regression (`method="plr"`).

```
# Logistic regression
FL_mod <- train(as.factor(warstds) ~ warhist + ln_gdpen + lpopns + lmtnest + ncontig + oil + nwstate + i
      method = "glm", metric = "ROC", family = binomial(link = "logit"), trControl = tc)

# Penalized logistic regression
FL_mod_plr <- train(as.factor(warstds) ~ warhist + ln_gdpen + lpopns + lmtnest + ncontig + oil + nwstat
      method = "plr", metric = "ROC", trControl = tc)
```

(b) The Collier & Hoeffler model (2004) (CH) can be described as the following:

`as.factor(warstds) ~ sxpnew + sxpsq + ln_gdpen + gdpgrowth + warhist + lmtnest + ef + popdense + lpopns + coldwar + seceduc + ptime`

Please run the `train` function in the `caret` library with metric as `ROC`, method as `glm`, family as `binomial` (CH used logistic), trControl as our set `tc`, and your training data, on the above specification. Do the same for a penalized logistic regression (`method="plr"`).

```
# Logistic regression
CH_mod <- train(as.factor(warstds) ~ sxpnew + sxpsq + ln_gdpen + gdpgrowth + warhist + lmtnest + ef + po
      method = "glm", metric = "ROC", family = binomial(link = "logit"), trControl = tc)

# Penalized Logistic regression
CH_mod_plr <- train(as.factor(warstds) ~ sxpnew + sxpsq + ln_gdpen + gdpgrowth + warhist + lmtnest + ef
      method = "plr", metric = "ROC", trControl = tc)
```

```
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
```

```
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
```

(c) The Hegre & Sambanis model (2006) (HS) can be described as the following:

```
as.factor(warstds) ~ lpopns + ln_gdpen + inst3 + parreg + geo34 + proxregc + gdpgrowth +
anoc +  partfree + nat_war + lmtnest + decade1 + pol4sq + nwstate + regd4_alt + etdo4590
+ milper +  geo1 + tnatwar + presi
```

Please run the `train` function in the `caret` library with metric as `ROC`, method as `glm`, family as `binomial` (HS used logistic), trControl as our set `tc`, and your training data, on the above specification. Do the same for a penalized logistic regression (`method="plr"`).

```
# Logistic regression
HS_mod <- train(as.factor(warstds) ~ lpopns + ln_gdpen + inst3 + parreg + geo34 + proxregc + gdpgrowth +
      method = "glm", metric = "ROC", family = binomial(link = "logit"), trControl = tc)

# Penalized Logistic regression
HS_mod_plr <- train(as.factor(warstds) ~ lpopns + ln_gdpen + inst3 + parreg + geo34 + proxregc + gdpgrow
      method = "plr", metric = "ROC", trControl = tc)
```

```
##
```

```
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
```

```
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
##
## Convergence warning in plr: 2
```

(d) Finally, run a random forest model on the outcome `warstds` with all regressors (except `year`) using `train`, metric as `ROC`, sampsize as `c(30,90)`, importance as `TRUE`, proximity as `FALSE`, number of trees to `1000`, tcControl as our above specified `tc` on the training data. (This may take some time, so you might want to start from a small number of trees and proceed with it. Once the codes are done, return to this question and set `ntree=1000`.)

```
# Cleaning the data
x <- data.train %>% select(!year) %>% select(!warstds)
y <- as.factor(data.train$warstds)

# Creating random forest
tree_mod <- train(x, y, metric = "ROC", sampsize = c(30,90), importance = TRUE, proximity = FALSE, ntre
```

What are the types of variables that seem to feature most in each type of model as predictors?

```
# Calling a summary of all models to see variables
#summary(CH_mod)
#summary(CH_mod_plr)
#summary(FL_mod)
#summary(FL_mod_plr)
#summary(HS_mod)
#summary(HS_mod_plr)
#summary(tree_mod)
```

After looking at the summary output, there is a diverse range of variables used but predictors relating to GDP showed up across multiple models, both the ln(GDP) variable and the GDP growth variable. This speaks the strength of economic health as a predictor.

Save all models (total $3 \times 2 + 1 = 7$ models) with easy to read names.

```
# Defining list of the 7 models, leaving out tree model because it uses a different predict function
plr_models <- list(FL_mod_plr, HS_mod_plr, CH_mod_plr)
norm_models <- list(FL_mod, HS_mod, CH_mod)
```

Question 2

We will now create ROC plots for different models:

- Collect the predicted probabilities for the outcome from each of the above models (note these should be for the highest AUC score in the caret CV procedure, `your-logit-model$finalModel$fitted.values`). For the random forests model, requires a call from `predict()` with type set to "prob" (i.e., `predict(your-rf-model$finalModel, type="prob")`).
- Follow the sample code below to create a `prediction` object from which to calculate the performance of the classifier in terms of true positive and false positive rates.
- Plot the ROC curves of all the unpenalized models (= classic logistic regression) and the RF model.
- Then separate, plot the ROC curves of all penalized models and the RF model. How does the RF model compare?

Sample code:

```
## ROC plot: Example with a *classic logistic regression* model trained with caret CV procedure
library(ROCR) # We will use prediction() & performance() functions from this package


results <- c()

# Penalized models
for(model in plr_models) {

  ## 1. Collect the predicted probabilities
  predict_probas <- 1 - model$finalModel$fitted.values
  # The above line should be changed for *penalized logistic regression* and *RF* model

  ## 2. Using in-sample prediction, calculate true positive and false positive rates.
  predictions <- prediction(predict_probas, data.train$warstds)
  results <- c(results, performance(predictions,"tpr","fpr"))

}

# Different predict function for forest
tree_predicts <- predict(tree_mod$finalModel, type="prob")
predictions <- prediction(tree_predicts[,2], data.train$warstds)
results <- c(results, performance(predictions,"tpr","fpr"))

# Normal non-penalized models
for(model in norm_models) {

  ## 1. Collect the predicted probabilities
  predict_probas <- model$finalModel$fitted.values
  # The above line should be changed for *penalized logistic regression* and *RF* model

  ## 2. Using in-sample prediction, calculate true positive and false positive rates.
  predictions <- prediction(predict_probas, data.train$warstds)
  results <- c(results, performance(predictions,"tpr","fpr"))

}



# Plotting results
plot(results[[1]], main="Penalized Logits and Random Forest", col = cbp1[1], label = "Tree")
plot(results[[2]], add = TRUE, col = cbp1[2])
plot(results[[3]], add = TRUE, col =cbp1[3])
plot(results[[4]], add = TRUE, col =cbp1[4])


# Adding legend
legend("bottomright", c("CH Model Plr", "HS Model Plr", "FL model PLR", "Tree"), cex = .75, col = cbp1[
```

```
cols <- c("Tree", "CH Model", "HS Model", "FL Model")

plot(results[[4]], main="Logits and Random Forest", col = cbp1[4])
plot(results[[5]], add = TRUE, col =cbp1[5])
plot(results[[6]], add = TRUE, col = cbp1[6])
```

```
plot(results[[7]], add = TRUE, col = cbp1[7])


# Adding legend
legend("bottomright",c("Tree", "CH Model", "HS Model", "FL Model"), cex = .75, col = cbp1[4:7],pch = 20)
```

For both the normal logits and the penalized logits, the random forest analysis performs the best of the bunch, with a higher true positive rate across false positive rates.

Question 3

Finally, we will evaluate out of sample prediction of unpenalized models and the RF model.

Pull out the testing data (1999, 2000) and evaluate each of the model predictions on the testing data. Focus on true positives, or when `warstds` is (correctly) classified with high probability as a "war".

```
# Subsetting the data
dat <- subset(data.full, subset = data.full$year >= 1999)

# Showing the number of wars in dataset
table(dat$warstds)
```

```
##
## peace   war
##   334     2
```

```
# Generate out of sample predictions for Table 1
RF_pred<-as.data.frame(predict(tree_mod, newdata=dat, type="prob"))
FL_pred<-as.data.frame(predict(FL_mod, newdata=dat, type="prob"))
CH_pred<-as.data.frame(predict(CH_mod, newdata=dat, type="prob"))
HS_pred<-as.data.frame(predict(HS_mod, newdata=dat, type="prob"))

preds<-cbind(dat$year, dat$warstds, FL_pred[,2], CH_pred[,2], HS_pred[,2], RF_pred[,2])
colnames(preds)<-c("year", "CW_Onset",
                   "Fearon and Latin (2003)",
                   "Collier and Hoeffler (2004)",
                   "Hegre and Sambanis (2006)",
                   "Random Forest")
preds<-as.data.frame(preds)

# Table results
finalTable<-preds[order(-preds$CW_Onset, preds$year),]
#Filling in 0 or 1 with peace or war
finalTable$CW_Onset = c("peace", "war")[finalTable$CW_Onset]

head(finalTable, n=6)
```

```
##     year CW_Onset Fearon and Latin (2003) Collier and Hoeffler (2004)
## 138 1999      war              0.06759787                0.0164709598
## 308 1999      war              0.01634902                0.0078920572
## 1   1999    peace              0.01984194                0.0377360013
## 3   1999    peace              0.01989782                0.0009862537
## 5   1999    peace              0.01138666                0.0102527626
## 6   1999    peace              0.11454074                0.0722992909
##     Hegre and Sambanis (2006) Random Forest
## 138                0.11004861         0.926
```

```
## 308                0.01099924        0.828
## 1                  0.01599261        0.900
## 3                  0.02659714        0.052
## 5                  0.02027545        0.874
## 6                  0.07892516        0.990
```