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"`).

```
# model for the logistic regression
mod_fl_1 <- caret::train(as.factor(warstds) ~ warhist + ln_gdpen + lpopns + lmtnest + ncontig + oil + nw
             method="glm",
             metric="ROC",
             family = "binomial",
             trControl = tc,
             data=data.train
             )

summary(mod_fl_1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0079  -0.1907  -0.1363  -0.1019   3.2525
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.170176   1.116431  -7.318 2.51e-13 ***
## warhist      0.000317   0.258559   0.001  0.99902
## ln_gdpen    -0.362901   0.118119  -3.072  0.00212 **
## lpopns       0.174164   0.068327   2.549  0.01080 *
## lmtnest      0.187061   0.080235   2.331  0.01973 *
## ncontig      0.167825   0.289625   0.579  0.56228
## oil          0.337220   0.309738   1.089  0.27627
## nwstate      1.820748   0.319323   5.702 1.18e-08 ***
## inst3        1.336491   0.208496   6.410 1.45e-10 ***
## pol4        -0.020348   0.017992  -1.131  0.25808
## ef           0.408354   0.425961   0.959  0.33773
## relfrac      0.621164   0.509993   1.218  0.22323
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1158.4  on 6803  degrees of freedom
## Residual deviance: 1036.2  on 6792  degrees of freedom
## AIC: 1060.2
##
## Number of Fisher Scoring iterations: 7
```

```
# model for the penalized logistic regression
mod_fl_2 <- caret::train(as.factor(warstds) ~ warhist + ln_gdpen + lpopns + lmtnest + ncontig + oil + nw
             method="plr",
             metric="ROC",
             # family = "binomial",
```

```
            trControl = tc,
            data=data.train
            )

summary(mod_fl_2)
```

```
##
## Call:
## stepPlr::plr(x = x, y = y, weights = if (!is.null(wts)) wts else rep(1,
##     length(y)), lambda = param$lambda, cp = as.character(param$cp))
##
## Coefficients:
##           Estimate Std.Error z value Pr(>|z|)
## Intercept  8.13042   1.10937   7.329    0.000
## warhist    0.00334   0.25486   0.013    0.990
## ln_gdpen   0.36656   0.11725   3.126    0.002
## lpopns    -0.17341   0.06800  -2.550    0.011
## lmtnest   -0.18624   0.07999  -2.328    0.020
## ncontig   -0.16809   0.28456  -0.591    0.555
## oil       -0.33120   0.30327  -1.092    0.275
## nwstate   -1.78237   0.31520  -5.655    0.000
## inst3     -1.32119   0.20645  -6.400    0.000
## pol4       0.02001   0.01791   1.117    0.264
## ef        -0.40496   0.40782  -0.993    0.321
## relfrac   -0.59585   0.48222  -1.236    0.216
##
##     Null deviance: 1158.39 on 6803 degrees of freedom
## Residual deviance: 1036.21 on 6792.17 degrees of freedom
##             Score: deviance + 8.8 * df = 1140.64
```

(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"`).

```
# model for the logistic regression
mod_ch_1 <- caret::train(as.factor(warstds) ~ sxpnew + sxpsq + ln_gdpen + gdpgrowth + warhist + lmtnest
            method="glm",
            metric="ROC",
            family = "binomial",
            trControl = tc,
            data=data.train
            )
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(mod_ch_1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1292  -0.1889  -0.1184  -0.0802   3.6134
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.612e+00  1.255e+00  -5.267 1.38e-07 ***
## sxpnew       1.749e+01  4.327e+00   4.042 5.30e-05 ***
## sxpsq       -7.662e+01  1.141e+01  -6.718 1.84e-11 ***
## ln_gdpen    -4.682e-01  1.424e-01  -3.288  0.00101 **
## gdpgrowth   -6.753e+00  1.233e+00  -5.475 4.36e-08 ***
## warhist     -5.455e-01  2.662e-01  -2.049  0.04045 *
## lmtnest      2.385e-01  8.259e-02   2.888  0.00388 **
## ef          -7.986e-03  4.478e-01  -0.018  0.98577
## popdense    -1.554e-04  9.070e-04  -0.171  0.86395
## lpopns       2.040e-01  6.936e-02   2.941  0.00327 **
## coldwar     -7.653e-02  2.779e-01  -0.275  0.78305
## seceduc     -4.353e-03  5.666e-03  -0.768  0.44234
## ptime       -2.384e-03  7.763e-04  -3.071  0.00213 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1158.4  on 6803  degrees of freedom
## Residual deviance:  985.7  on 6791  degrees of freedom
## AIC: 1011.7
##
## Number of Fisher Scoring iterations: 9
```

```
# model for the penalized logistic regression
mod_ch_2 <- caret::train(as.factor(warstds) ~ sxpnew + sxpsq + ln_gdpen + gdpgrowth + warhist + lmtnest
             method="plr",
             metric="ROC",
             # family = "binomial",
```

```
             trControl = tc,
             data=data.train
             )
```

```
##
## 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
```

```
summary(mod_ch_2)
```

```
##
## Call:
## stepPlr::plr(x = x, y = y, weights = if (!is.null(wts)) wts else rep(1,
##     length(y)), lambda = param$lambda, cp = as.character(param$cp))
##
## Coefficients:
##            Estimate Std.Error z value Pr(>|z|)
## Intercept   5.06077   1.20132   4.213    0.000
## sxpnew      7.48970   4.91897   1.523    0.128
## sxpsq       6.29554  11.54802   0.545    0.586
## ln_gdpen    0.43291   0.13062   3.314    0.001
## gdpgrowth   7.51143   1.18292   6.350    0.000
## warhist     0.20617   0.26574   0.776    0.438
## lmtnest    -0.17427   0.07889  -2.209    0.027
## ef         -0.32092   0.44204  -0.726    0.468
## popdense    0.00104   0.00112   0.929    0.353
## lpopns     -0.15139   0.06776  -2.234    0.025
## coldwar    -0.03981   0.27576  -0.144    0.886
## seceduc     0.00698   0.00559   1.249    0.212
## ptime       0.00126   0.00075   1.680    0.093
##
##     Null deviance: 1158.39 on 6803 degrees of freedom
## Residual deviance: 1024.65 on 6791.03 degrees of freedom
##             Score: deviance + 8.8 * df = 1139.09
```

(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"`).

```
# model for the logistic regression
mod_hs_1 <- caret::train(as.factor(warstds) ~ lpopns + ln_gdpen + inst3 + parreg + geo34 + proxregc + g
             method="glm",
             metric="ROC",
             family = "binomial",
             trControl = tc,
             data=data.train
```

```
            )
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(mod_hs_1)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2940  -0.1921  -0.1237  -0.0742   3.7350
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.348e+00  1.347e+00  -5.454 4.92e-08 ***
## lpopns       2.000e-01  7.430e-02   2.691  0.00712 **
## ln_gdpen    -3.228e-01  1.324e-01  -2.438  0.01476 *
## inst3        1.401e+00  2.505e-01   5.592 2.25e-08 ***
## parreg      -2.915e-01  1.378e-01  -2.116  0.03435 *
## geo34       -6.690e-02  3.155e-01  -0.212  0.83206
## proxregc    -8.856e-01  3.134e-01  -2.826  0.00471 **
## gdpgrowth   -6.193e+00  1.211e+00  -5.113 3.17e-07 ***
## anoc         3.456e-01  3.091e-01   1.118  0.26358
## partfree     1.678e-01  2.757e-01   0.609  0.54274
## nat_war      9.994e-01  3.145e-01   3.178  0.00148 **
## lmtnest      1.668e-01  8.030e-02   2.077  0.03781 *
## decade1      1.835e-01  2.568e-01   0.714  0.47495
## pol4sq       1.663e-03  5.725e-03   0.291  0.77140
## nwstate      1.865e+00  3.524e-01   5.291 1.22e-07 ***
## regd4_alt   -5.950e-02  2.970e-02  -2.003  0.04513 *
## etdo4590     2.410e-02  2.294e-01   0.105  0.91632
## milper      -2.545e-05  2.743e-04  -0.093  0.92607
## geo1        -8.188e-01  1.114e+00  -0.735  0.46242
## tnatwar     -3.408e-01  1.758e-01  -1.939  0.05253 .
## presi       -1.284e+00  6.180e-01  -2.077  0.03778 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1158.39  on 6803  degrees of freedom
## Residual deviance:  974.44  on 6783  degrees of freedom
## AIC: 1016.4
##
## Number of Fisher Scoring iterations: 9
```

```
# model for the penalized logistic regression
mod_hs_2 <- caret::train(as.factor(warstds) ~ lpopns + ln_gdpen + inst3 + parreg + geo34 + proxregc + g
             method="plr",
             metric="ROC",
             # family = "binomial",
             trControl = tc,
             data=data.train
```

```
              )
```

```
##
## 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
```

```
summary(mod_hs_2)
```

```
##
## Call:
## stepPlr::plr(x = x, y = y, weights = if (!is.null(wts)) wts else rep(1,
##     length(y)), lambda = param$lambda, cp = as.character(param$cp))
##
## Coefficients:
##            Estimate Std.Error z value Pr(>|z|)
## Intercept  5.26489   1.30087   4.047    0.000
## lpopns    -0.09644   0.07309  -1.319    0.187
## ln_gdpen   0.32887   0.13076   2.515    0.012
## inst3     -1.41541   0.24809  -5.705    0.000
## parreg     0.36563   0.13314   2.746    0.006
## geo34      0.08000   0.30935   0.259    0.796
## proxregc   0.88110   0.30862   2.855    0.004
## gdpgrowth  4.43942   1.29968   3.416    0.001
## anoc      -0.22852   0.30607  -0.747    0.455
## partfree  -0.12516   0.27280  -0.459    0.646
## nat_war   -1.02032   0.31254  -3.265    0.001
## lmtnest   -0.16780   0.07895  -2.125    0.034
## decade1   -0.13752   0.25565  -0.538    0.591
## pol4sq     0.00007   0.00569   0.012    0.990
## nwstate   -1.85050   0.34754  -5.325    0.000
## regd4_alt  0.05279   0.02899   1.821    0.069
## etdo4590   0.03529   0.22757   0.155    0.877
## milper    -0.00008   0.00026  -0.308    0.758
## geo1       0.75236   1.07903   0.697    0.486
## tnatwar    0.30841   0.17517   1.761    0.078
## presi      1.38651   0.62122   2.232    0.026
##
##     Null deviance: 1158.39 on 6803 degrees of freedom
## Residual deviance: 978.54 on 6783 degrees of freedom
##             Score: deviance + 8.8 * df = 1163.87
```

(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`.)

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

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

```
model.rf <- caret::train(as.factor(warstds)~. -year,
                         metric="ROC", method="rf",
                         samplesize=c(30,90),
                         importance=T,
                         proximity=F,
                         ntree=10,
                         trControl=tc,
                         data=data.train)
model.rf
```

```
## Random Forest
##
## 6804 samples
##   91 predictor
##    2 classes: 'peace', 'war'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6123, 6123, 6124, 6124, 6124, 6123, ...
## Resampling results across tuning parameters:
##
##   mtry  ROC        Sens       Spec
##    2    0.7455734  0.9994021  0.03560606
##   46    0.7917510  0.9983558  0.11363636
##   90    0.7591430  0.9976084  0.11136364
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 46.
```

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:

```
library(tidyverse)
## 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

## 1. Collect the predicted probabilities
pred.fl.war <- mod_fl_1$finalModel$fitted.values
pred.ch.war <- mod_ch_1$finalModel$fitted.values
pred.hs.war <- mod_hs_1$finalModel$fitted.values
rf.1.pred<-predict(model.rf$finalModel, type="prob")
```

```
rf.1.pred<-as.data.frame(rf.1.pred)

## 2. Using in-sample prediction, calculate true positive and false positive rates.
pred.fl <- prediction(pred.fl.war, data.train$warstds)
perf.(your-mod-name) <- performance(pred.(your-mod-name),"tpr","fpr")

## 2. Using in-sample prediction, calculate true positive and false positive rates.
pred.fl <- prediction(pred.fl.war, data.train$warstds)
perf.fl <- performance(pred.fl,"tpr","fpr")
pred.ch <- prediction(pred.ch.war, data.train$warstds)
perf.ch <- performance(pred.ch,"tpr","fpr")
pred.hs <-prediction(pred.hs.war, data.train$warstds)
perf.hs <-performance(pred.hs, "tpr", "fpr")
# pred.rf.1<-prediction(rf.1.pred$war, data.train$warstds)
# perf.rf.1<-performance(pred.rf.1, "tpr", "fpr")

## 3. Plot the ROC curves
plot(perf.fl, main="Logits and Random Forests", col=cbp1[1])
plot(perf.ch, add=T, lty=2, col=cbp1[2])
plot(perf.hs, add=T, lty=3, col=cbp1[3])
# plot(perf.rf.1, add=T, lty=4, col=cbp1[4])
legend(0.32, 0.25, c("Fearon and Laitin (2003)", "Collier and Hoeffler (2004)", "Hegre and Sambanis (20


#### DO AGAIN FOR LOGITS AND Random FOrest

### ROC Plots for Penalized Logits and RF###
FL.2.pred<-1-mod_fl_2$finalModel$fitted.values # 1 - prob(peace)
CH.2.pred<-1-mod_ch_2$finalModel$fitted.values
HS.2.pred<-1-mod_hs_2$finalModel$fitted.values

pred.FL.2 <- prediction(FL.2.pred, data.train$warstds)
perf.FL.2 <- performance(pred.FL.2,"tpr","fpr")
pred.CH.2<- prediction(CH.2.pred, data.train$warstds)
perf.CH.2 <- performance(pred.CH.2,"tpr","fpr")
pred.HS.2<- prediction(HS.2.pred, data.train$warstds)
perf.HS.2 <- performance(pred.HS.2,"tpr","fpr")

##### Plot ROC Curves for penalized logistic regression models.
plot(perf.FL.2, main="Penalized Logits and Random Forests", col=cbp1[1])
plot(perf.CH.2, add=T, lty=2, col=cbp1[2])
plot(perf.HS.2, add=T, lty=3, col=cbp1[3])
# plot(perf.RF.1, add=T, lty=4, col=cbp1[4])
legend(0.32, 0.25, c("Fearon and Laitin (2003)", "Collier and Hoeffler (2004)", "Hegre and Sambanis (20
```

For some reason I keep getting "Error: 'predictions' contains NA." for the random forrest prediction. Because of that I was unable to include the RF ROC curve.

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".

```
mena<-subset(data.full, data.full$year >= 1999)
table(mena$warstds)
```

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

```
### Generate out of sample predictions for Table 1
fl.pred<-predict(mod_fl_1, newdata=mena, type="prob")
fl.pred<-as.data.frame(fl.pred)
ch.pred<-predict(mod_ch_1, newdata=mena, type="prob")
ch.pred<-as.data.frame(ch.pred)
hs.pred<-predict(mod_hs_1, newdata=mena, type="prob")
hs.pred<-as.data.frame(hs.pred)
rf.pred<-predict(model.rf, newdata=mena, type="prob")
rf.pred<-as.data.frame(rf.pred)

predictions<-cbind(mena$year, mena$warstds, fl.pred[,2], ch.pred[,2], hs.pred[,2], rf.pred[,2])
### Write column headings for the out of sample data. ###
colnames(predictions)<-c("year", "CW_Onset", "Fearon and Latin (2003)", "Collier and Hoeffler (2004)",
"Random Forest")
### Save predictions as data frame for ordering the columns.
predictions<-as.data.frame(predictions)
### Table 1 Results, ordered by Onset (decreasing), and year (increasing) in R
Onset_table<-predictions[order(-predictions$CW_Onset, predictions$year),]
Onset_table$CW_Onset = c("peace", "war")[Onset_table$CW_Onset] # change it to character
### Rows 1-5 of the above ###
head(Onset_table, n=5)
```

```
##     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
##     Hegre and Sambanis (2006) Random Forest
## 138                0.11004861           0.2
## 308                0.01099924           0.1
## 1                  0.01599261           0.0
## 3                  0.02659714           0.0
## 5                  0.02027545           0.1
```