Replication Report of Muchlinski et al. 2016. Some parts are not replicable

Introduction

For our letter *"Is Random Forest really better in predicting civil war onset than logistic regression?"* we use the paper by Muchlinski et al. (2016) *"Comparing Random Forest with Logistic Regression for Predicting Class-Imbalanced Civil War Onset Data"* as an application. Replicating the paper we encountered replication problems that go beyond the problematic parts we state in our letter. In this file we comment the replication code by Muchlinski et al. (2016) and highlight the problems. Overall, we find three major problems:

- Problem 1: No proper seeds. The original results are not replicable.
- Problem 2: Problematic implementation of Random Forests
- Problem 3: Severe problems with the out-of-sample prediction:

At least some conclusions of the paper (in particular the out-of-sample prediction using the updated Civil War Data 2001-2014) are not replicable.

The titles of the code chunks specify the lines in the original replication file. If the chunk title states *"New Code:"* we added some code to make our point.

A first look at the data

```r
data=read.csv(file="SambnisImp.csv") # data for prediction
data2<-read.csv(file="Amelia.Imp3.csv") # data for causal machanisms
```

Here the authors read in their dataset. In the paper it sounds like they base their predictions (and other inferences) on the original Civil War Data by Hegre and Sambanis (2006). Only in the "Read Me-Comparing Random Forest with Logistic Regression.docx" accompanying the replications files they specify: "There are two data files. The first, *SambanisImp.csv* contains the fully imputed dataset taken from Hegre and Sambanis (2006). This first dataset was used to explore the predictive accuracy of Random Forest compared to logistic regression. Figures 1,2, and 4 were constructed from this data. The second data file, Amelia.Imp3.csv was used to explore causal mechanisms related to civil war onset. Amelia.Imp3 is a smaller dataset where only variables theorized to be most relevant to the onset of civil war were imputed. Variables removed from the Amelia dataset were chosen based on theory and regression models included in Fearon and Laitin (2003), Collier and Hoeffler (2004), and Hegre and Sambanis (2006)."

Here we find the first unclarity. The original Hegre and Sambanis (2006) data set contains 9,691 observations (with a considerable amount of missing values), whereas their data set only contains 7,140 observations. From their comments and the replication code it is not clear how exactly the authors imputed missing values (they write that they imputed based on Random Forest and that other imputation methods do not affect their predictions, yet they provide no code for this) and more importantly, why their dataset only contains 7140 observations as compared to the 9,691 observations in the original dataset. It is also unclear why they need two different imputed datasets for their predictions and the causal mechanisms. More problematic in our view is that they never tell the reader of the paper that their inferences are based on different datasets. In the paper it sounds like all their models use the same data.

Imputation of missing values certainly can be useful and justified. However, it should be at least specified in the paper or replication code what data we are actually looking at.

```r
library(randomForest) #for random forests
library(caret) # for CV folds and data splitting
library(ROCR) # for diagnostics and ROC plots/stats
library(pROC) # same as ROCR
library(stepPlr) # Firth;s logit implemented thru caret library
library(doMC) # for using multipe processor cores
```

```r
###Using only the 88 variables specified in Sambanis (2006) Appendix###
data.full<-data[,c("warstds", "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")]
```

Muchlinski et al. write of 88 variables (the same as in Hegre & Sambanis) however, they did only include 87 variables (`drace` is missing) (all `geo` variables are dummies for world regions, hence not counted into the 88 variables). This however should not affect the results. More importantly, Muchlinski et al. chose a different dependent variable (`warstds`) than Hegre & Sambanis did (`warstns`) even though both variables are in the original dataset. Furthermore, the `ptime` variable is not the peacetime variable used by Hegre & Sambanis.

```r
###Converting DV into Factor with names for Caret Library###
data.full$warstds<-factor(
  data.full$warstds,
  levels=c(0,1),
  labels=c("peace", "war"))

registerDoMC(cores=7) # distributing workload over multiple cores for faster computaiton
```

This is fine.

Problem 1: No proper seeds. The original results are not replicable.

```r
set.seed(666) #the most metal seed for CV
```

This is fine. Setting a seed for data ensures that if the code is run again, the exact same slices of data are reproduced which were used for the cross-validation. However, Muchlinski et al. only set their seed once. Thus, the seed only applies to the first function call where a random number generator is needed. The seed does not apply to function calls after. This means that the original results of the paper are unfortunately not exactly reproducible.

Best Practice: Always specify set.seed() before a function call, or use the *caret* `createDataPartition()` function to create partitions manually. These partitions can then be used for each model.

The logit and penalized logit models

```
#This method of data slicing - or CV - will be used for all logit models
# - uncorrected and corrected
tc<-trainControl(method="cv",
                 number=10,#creates CV folds - 10 for this data
                 summaryFunction=twoClassSummary,
                 # provides ROC summary stats in call to model
                 classProb=T)

#Fearon and Laitin Model Specification###
model.fl.1<-train(as.factor(warstds)~warhist+ln_gdpen+lpopns+lmtnest+ncontig+oil+nwstate
             +inst3+pol4+ef+relfrac, #FL 2003 model spec
             metric="ROC", method="glm", family="binomial", #uncorrected logistic model
             trControl=tc, data=data.full)

summary(model.fl.1) #provides coefficients & traditional R model output
model.fl.1 # provides CV summary stats # keep in mind caret takes first class
# (here that's 0)
# as refernce class so sens and spec are backwards

#confusionMatrix(model.fl.1, norm="average") ### comfusion matrix for predicted classes
### not used in paper

###Now doing Fearon and Laitin (2003) penalized logistic regression
model.fl.2<-train(as.factor(warstds)~warhist+ln_gdpen+lpopns+lmtnest+ncontig+oil+nwstate
             +inst3+pol4+ef+relfrac, #FL 2003 model spec
             metric="ROC", method="plr", # Firth's penalized logistic regression
             trControl=tc, data=data.full)
summary(model.fl.2)
model.fl.2

#confusionMatrix(model.fl.2, norm="average")
```

Here the authors run the logit model and penalized logit with the Fearon and Laitin specification. They use 10-fold cross-validation for all of their models. Setting the seed only once is problematic here as outlined before. This essentially means, that the two (and all subsequent models) are trained on different slices of the dataset.

Apparently Muchlinski et al. looked at confusion matrices/classification tables in the process of building their models. It remains unclear why they do not present them in the paper or the online appendix.

```
###Now doing RF on FL (2003) Specification### RF for each model independently
### not used in paper
#model.fl.rf<-train(as.factor(warstds)~warhist+ln_gdpen+lpopns+lmtnest+ncontig+oil+nwstate
 #                  +inst3+pol4+ef+relfrac, #FL 2003 model spec
  #                 metric="ROC", method="rf",
   #                sampsize=c(30,90), #downsampling the DV
    #               trControl=tc, data=data.full)
#model.fl.rf

#confusionMatrix(model.fl.rf, norm="average")
```

This would have been an almost fair test of random forest against the two previous logit models. (Set aside that the authors use up-sampling here, and they did not resample for the logit models.) Running this shows that using the same variables as Fearon and Laitin, random forest hardly does better than the logit specifications when looking at the respective ROC-AUC values of both models.

```r
###Now doing Collier and Hoeffler (2004) uncorrected logistic specification###

model.ch.1<-train(as.factor(warstds)~sxpnew+sxpsq+ln_gdpen+gdpgrowth+warhist+lmtnest+ef+
                    popdense+lpopns+coldwar+seceduc+ptime, #CH 2004 model spec
metric="ROC", method="glm", family="binomial",
trControl=tc, data=data.full)
model.ch.1

#confusionMatrix(model.ch.1, norm="average") # again, confusion matrix not used in paper

###Now Collier and Hoeffler with penalized logistic regression###

model.ch.2<-train(as.factor(warstds)~sxpnew+sxpsq+ln_gdpen+gdpgrowth+warhist+lmtnest+ef
                  +popdense+lpopns+coldwar+seceduc+ptime, #CH 2004 model spec
                  metric="ROC", method="plr", #penalized logistic regression
                  trControl=tc, data=data.full)
model.ch.2
#confusionMatrix(model.ch.2, norm="average")

#model.ch.rf<-train(as.factor(warstds)~sxpnew+sxpsq+ln_gdpen+gdpgrowth+warhist+lmtnest
#          +ef+popdense +lpopns+coldwar+seceduc+ptime, #CH 2004 model spec
  #                metric="ROC", method="rf", #random forest
   #               sampsize=c(30, 90),
    #              trControl=tc, data=data.full)
#model.ch.rf

#confusionMatrix(model.ch.rf, norm="average")

###Now the Hegre and Sambanis Model Specification###

model.hs.1<-train(warstds~lpopns+ln_gdpen+inst3+parreg+geo34+proxregc+gdpgrowth+anoc+
                    partfree+nat_war+lmtnest+decade1+pol4sq+nwstate+regd4_alt+etdo4590
                  +milper+ geo1+tnatwar+presi,
                  metric="ROC", method="glm", family="binomial",
                  trControl=tc, data=data.full)
model.hs.1

# confusionMatrix(model.hs.1, norm="average") not used

model.hs.2<-train(warstds~lpopns+ln_gdpen+inst3+parreg+geo34+proxregc+gdpgrowth+anoc+
                    partfree+nat_war+lmtnest+decade1+pol4sq+nwstate+regd4_alt+etdo4590
                  +milper+geo1+tnatwar+presi,
                  metric="ROC", method="plr", #penalized logit
                  trControl=tc, data=data.full)
model.hs.2
#confusionMatrix(model.hs.2, norm="average")

#model.hs.rf<-train(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,
   #              metric="ROC", method="rf",
    #              sampsize=c(30,90),
     #             trControl=tc, data=data.full)
#model.hs.rf

#confusionMatrix(model.hs.rf, norm="average")
```

Here the authors set up the two other civil war model specifications. Again, since the seed was never set again all of these models run on different partitions of data (folds of cross-validation).

The Random Forest Model

```
##Implementing RF (with CV) on entirety of data###

model.rf<-train(as.factor(warstds)~.,
                  metric="ROC", method="rf",
                  sampsize=c(30,90), #Downsampling the class-imbalanced DV
                    importance=T, # Variable importance measures retained
                   proximity=F, ntree=1000, # number of trees grown
                   trControl=tc, data=data.full)
model.rf

#confusionMatrix(model.rf, norm="average")
```

Problem 2: Problems with the Random Forest implementation

This is the random forest specification (that supposedly outperforms the logistic regression models) of Muchlinski et al. However, as we already outline in the paper, this comparison is problematic:

1. No Seed, meaning that they train the models on different data.
2. Furthermore, they use **all** variables (90) in the dataset as compared to 11/12/20 in the other logistic regression models with the respective model specifications.
3. They up-sample the minority class (commonly applied to class-imbalanced data) for random forest, but not for the other models. More specifically, the sample from the majority class using 30 peaces and sample 90 wars.
4. Unfortunately, they resample outside the cross-validation sampling. This leads to heavy overfitting of the data and impressive in-sample performance measures in course of the cross-validation. The performance on out-of-sample data/hold-out data is not nearly as impressive (see https://topepo.github.io/caret/subsampling-for-class-imbalances.html#resampling). More on out-of-sample prediction below.

Example code for a fair comparison

A fair comparison should use the same data for all models. This includes the same folds for cross-validation, the same variables and the same resampling of the data (ie. oversampling). In the following chunk we present a fair comparison of the models and the results.

```
# Caret makes it easy to resample inside the cross-validation and for all models!

tcFair <-trainControl(method="cv",
                 number=10,#creates CV folds - 10 for this data
                 summaryFunction=twoClassSummary,
                 # provides ROC summary stats in call to model
```

```
                  classProb=T,
                  sampling = "up" # That's the only new line of code here.
                  )

set.seed(07132017) # set.seed() for every function call.
model.logit <-train(as.factor(warstds)~., # Instead of selecting variables we
                                        #also train the model on all variables.
metric="ROC", method="glm", family="binomial",
trControl=tcFair, data=data.full)
model.logit

# Same for plr model

set.seed(07132017) # set.seed() for every function call.
model.plr<-train(as.factor(warstds)~.,
                  metric="ROC", method="plr", #penalized logistic regression
                  trControl=tcFair, data=data.full)
model.plr

# And a fair random forest.

set.seed(07132017) # set.seed() for every function call.
model.rfFair<-train(as.factor(warstds)~.,
                  metric="ROC", method="rf",
                    importance=T, # Variable importance measures retained
                   proximity=F, ntree=1000, # number of trees grown
                   trControl=tcFair, data=data.full)
```

```
###Running Random Forest without CV to get OOB Error Rate - results not shown in paper###

RF.out<-randomForest(as.factor(warstds)~., sampsize=c(30, 90),
       importance=T, proximity=F, ntree=1000, confusion=T, err.rate=T, data=data.full)
print(RF.out)


###Random Forests on Amelia Imputed Data for Variable Importance Plot###
###Data Imputed only for Theoretically Important Variables###
#Done to analyze causal mechanisms

myvars <- names(data2) %in% c("X", "country", "year", "atwards")
newdata <- data2[!myvars]

RF.out.am<-randomForest(as.factor(warstds)~.,sampsize=c(30, 90),
          importance=T, proximity=F, ntree=1000, confusion=T, err.rate=T, data=newdata)
```

Here the authors run two **new** random forest models, using the original `RandomForest` package to get the OOB (out of bag) error rate. Note that this model is not exactly the same as the one created with the `caret` package before, and used to create the ROC-curves later. The latter was find using 10-fold cross-validation, and the final model had a *mtry* value (mtry defines the number of variables tried at each split) of 46, whereas the Random Forest obect `rf.out` only uses a *mtry* of 9. In terms of a strict model comparison, these two Random Forests are thus not exactly the same model. This is not best practice. Even more so since in the paper it reads like all results come from the **same** model.

```
varImpPlot(RF.out.am, sort=T, type=2,
main="Variables Contributing Most to Predictive Accuracy of
 Random Forests",n.var=20)

###Creating dotplot for Variable Importance Plot for RF###
importance(RF.out.am)

x<-c(2.3246380, 2.3055470, 1.8544127, 1.7447636, 1.6848230, 1.6094923,
     1.5564487, 1.4832437, 1.4100489, 1.3116247, 1.2875924, 1.1799487,
     1.1034743, 1.0983414, 1.0689367, 1.0663479, 1.0123892, 0.9961138,
     0.9961138, 0.9922545)
g<-c("GDP Growth", "GDP per Capita",
     "Life Expectancy", "Western Europe and US Dummy", "Infant Mortality",
     "Trade as Percent of GDP", "Mountainous Terrain", "Illiteracy Rate",
     "Population (logged)", "Linguistic Hetrogeneity", "Anocracy",
     "Median Regional Polity Score", "Primary Commodity Exports (Squared)",
     "Democracy", "Military Power", "Population Density",
     "Political Instability", "Ethnic Fractionalization", "Secondary Education",
     "Primary Commodity Exports")
dotchart(rev(x), rev(g), cex=0.75, xlab="Mean Decrease in Gini Score (OOB Estimates)",
         main="Variable Importance for Random Forests",
         xlim=c(1, 2.5))
```

Here the authors grow a new Random Forest, this time using the Amelia imputed data set, to obtain variable importance measures. This is again a different Random Fores, and different to the one obtained from the cross-validation for the ROC-curves later. Again, this is not best practice because in the paper it reads like all results come from the **same** model.

Performance Measures

```
###ROC Plots for Different Models###

library(ROCR)
attach(data.full) #have to attach the data to get probs for some reason

###Gathering info for ROC Plots: Uncorrected Logists###
FL.1.pred<-predict(model.fl.1, data.full$warstds, type="prob")
CH.1.pred<-predict(model.ch.1, data.full$warstds, type="prob")
HS.1.pred<-predict(model.hs.1, data.full$warstds, type="prob")
RF.1.pred<-predict(model.rf, data.full$warstds, type="prob")
RF.Fair.pred<-predict(model.rfFair, data.full$warstds, type="prob")

pred.FL <- prediction(FL.1.pred$war, data$warstds)
perf.FL <- performance(pred.FL,"tpr","fpr")
pred.CH <- prediction(CH.1.pred$war, data$warstds)
perf.CH <- performance(pred.CH,"tpr","fpr")
pred.HS<-prediction(HS.1.pred$war, data$warstds)
perf.HS<-performance(pred.HS, "tpr", "fpr")
pred.RF.1<-prediction(RF.1.pred$war, data$warstds)
perf.RF.1<-performance(pred.RF.1, "tpr", "fpr")
```

```r
plot(perf.FL, main="Uncorrected Logits and Random Forests")
plot(perf.CH, add=T, lty=2)
plot(perf.HS, add=T, lty=3)
plot(perf.RF.1, add=T, lty=4)
legend(0.32, 0.25, c("Fearon and Laitin (2003) 0.77",
                     "Collier and Hoeffler (2004) 0.82",
                     "Hegre and Sambanis (2006) 0.80",
                     "Random Forest 0.91" ),
       lty=c(1,2,3,4), bty="n",
       cex = .75)

###Now ROC Plots for Penalized Logits and RF###
FL.2.pred<-predict(model.fl.2, data.full$warstds, type="prob")
CH.2.pred<-predict(model.ch.2, data.full$warstds, type="prob")
HS.2.pred<-predict(model.hs.2, data.full$warstds, type="prob")


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

plot(perf.FL.2, main="Penalized Logits and Random Forests")
plot(perf.CH.2, add=T, lty=2)
plot(perf.HS.2, add=T, lty=3)
plot(perf.RF.1, add=T, lty=4)
legend(0.32, 0.25, c("Fearon and Laitin (2003) 0.77",
                     "Collier and Hoeffler (2004) 0.77",
                     "Hegre and Sambanis (2006) 0.80",
                     "Random Forest 0.91" ),
       lty=c(1,2,3,4), bty="n",
       cex = .75)

###Separation Plots###
library(separationplot)

##Have to transform DV back to 0,1 values for sep plots
data.full$warstds<-factor(
  data.full$warstds,
  levels=c("peace","war"),
  labels=c(0, 1))

Warstds<-as.vector(data.full$warstds) #transforming actual obs into vector

separationplot(FL.1.pred$war, Warstds, type = "line", line = T, lwd2=1,
  show.expected=T, heading="Fearon and Laitin (2003)",
  height=1.5, col0="white", col1="black", newplot = F)
separationplot(CH.1.pred$war, Warstds, type = "line", line = T, lwd2=1,show.expected=T,
         heading="Collier and Hoeffler (2004)",
         height=2.5, col0="white", col1="black", newplot = F)
separationplot(HS.1.pred$war, Warstds, type = "line", line = T, lwd2=1, show.expected=T,
         heading="Hegre and Sambanis (2006)",
         height=2.5, col0="white", col1="black", newplot = F)
separationplot(RF.1.pred$war, Warstds, type = "line", line = T, lwd2=1, show.expected=T,
        heading="Random Forests",
        height=2.5, col0="white", col1="black", newplot = F)
```

The code to produce the ROC-curves and separation plots is fine. However, these plots are missleading since the impressive ROC-AUC values of the cross-validated Random Forest are due to the heavy overfitting because of resampling outside the cross-validation. These plots should be presented using out-of-sample/hold out data. Especially when the goal is prediction.

```
par(mfrow=c(3,3))
partialPlot(RF.out.am, data2, gdpgrowth, which.class="1", xlab="GDP Growth Rate", main="",
            ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, ln_gdpen, ylim=c(-0.15, 0.15), which.class="1",
            xlab="GDP per Capita (log)",
            main="", ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, life, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Life Expectancy",
            main="", ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, infant, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Infant Mortality Rate",
            main="", ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, lmtnest, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Mountainous Terrain (log)",
            main="",  ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, pol4sq, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Polity IV Sq",
            main="", ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, lpopns, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Population", main="",
            ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, trade, ylim=c(-0.15, 0.15), which.class="1",
            xlab="Trade", main="",
            xlim=c(0,200), ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
partialPlot(RF.out.am, data2, geo1, ylim=c(-0.15, 0.15), which.class="1",
            xlab="W. Europe and U.S.",
            main="",ylab=expression(paste(Delta, "Fraction of Votes p(Y=1)")))
```

This is fine.

Problem 3: Severe problems with the out-of-sample prediction:

The gold standard of a testing the predictive power of a model is out-of-sample performance. Muchlinski et at. claim to use an updated version of the CWD ranging from 2001-2014 to perform an out-of-sample prediction. Unfortunately, we encounter several problems using their code that was supposedly used to produce Table 1 in the paper:

1. We could not reproduce Table 1.
   1. There are no ID variables (neither country nor year IDs) in the out-of-sample dataset. This makes it impossible to assign a certain prediction to a certain country/year as it is done in Table 1.
   2. Again, the seed was only set once so we do not get the same numbers.

2. What is more problematic: The authors randomly draw civil war onset probabilities from their cross-validated prediction and merge them to the out-of-sample observations of civil war onset. This means the predictions provided by the authors are in fact random probabilities
3. With the provided out-of-sample data it is not possible to correct this. None of the variables used to train the models are in the provided out-of-sample dataset. It is just not possible to make predictions based on the models and this dataset. Therefore, it is questionable whether the authors in fact "[...] updated the CWD for all countries in Africa and the Middle East from 2001 to 2014" (96).
4. The authors only present the true positive outcomes of their prediction. On this they base their main finding (which is already prominently cited) that "Random Forest correctly predicts nine out of twenty civil war onsets in this out-of-sample data when the threshold for positive prediction is 0.50" (96). Set aside that the predictions are in fact just random numbers only, focusing on the true positive outcomes is only telling part of the story. With their random numbers the authors also would predict 221 civil war onsets where there were none (false positives).

```r
####Analysis for Out of Sample Africa Data###
data3<-read.csv(file="AfricaImp.csv") # Reading in the Africa Data from 2001-2014

model.fl.africa<-glm(as.factor(warstds)~warhist+ln_gdpen+lpopns+lmtnest+ncontig+oil+
                       nwstate+inst3+pol4+ef+relfrac,
                     family="binomial", data=data.full)
model.ch.africa<-glm(as.factor(warstds)~sxpnew+sxpsq+ln_gdpen+gdpgrowth+warhist+lmtnest+
                       ef+popdense+lpopns+coldwar+seceduc+ptime,
                     family="binomial", data=data.full)
model.hs.africa<-glm(warstds~lpopns+ln_gdpen+inst3+parreg+geo34+proxregc+gdpgrowth+anoc+
              partfree+nat_war+lmtnest+decade1+pol4sq+nwstate+regd4_alt+etdo4590+milper+
               geo1+tnatwar+presi,
              family="binomial", data=data.full)



yhat.rf<-predict(RF.out, type="prob") #taken from RF on whole data
###We used original CW data for training data here for all models/algorithms###
Yhat.rf<-as.data.frame(yhat.rf[,2])
yhat.fl.africa<-predict(model.fl.africa, type="response")
Yhat.fl.africa<-as.data.frame(yhat.fl.africa)
yhat.ch.africa<-predict(model.ch.africa, type="response")
Yhat.ch.africa<-as.data.frame(yhat.ch.africa)
yhat.hs.africa<-predict(model.hs.africa, type="response")
Yhat.hs.africa<-as.data.frame(yhat.hs.africa)
```

These are in-sample predictions. The models are all trained with the training data `data.full` in the predict command no new data is specified. Therefore, this returns 7140 in-sample predictions.

```r
###Selecting random samples to make pred and actual lengths equal###
set.seed(100)
predictors.rf<-Yhat.rf[sample(nrow(Yhat.rf), 737),]
predictors.fl<-Yhat.fl.africa[sample(nrow(Yhat.fl.africa), 737),]
predictors.ch<-Yhat.ch.africa[sample(nrow(Yhat.ch.africa), 737),]
predictors.hs<-Yhat.hs.africa[sample(nrow(Yhat.hs.africa), 737),]

library(SDMTools)


confusion.matrix(data3$warstds, predictors.rf, threshold=.5)
confusion.matrix(data3$warstds, predictors.fl, threshold=.5)
confusion.matrix(data3$warstds, predictors.ch, threshold=.5)
confusion.matrix(data3$warstds, predictors.hs, threshold=.5)
```

This is where the code becomes really problematic! `Yhat.rf` and the other `Yhat.` are the in-sample predictions from above. Now the authors **randomly sample** from those predictions 737 times (the number of observations in the new dataset). In the next step they merge those random predictions with the out-of-sample dependent variable to get the confusion matrices and performance measures. Civil war onset prediction in this case appears to be a matter of guessing, not proper prediction.

```
###ROC and AUC scores for out of sample data###
library(ROCR)
pred.fl.africa <- prediction(predictors.fl, data3$warstds)
perf.fl.africa<- performance(pred.fl.africa,"tpr","fpr")
pred.ch.africa<-prediction(predictors.ch, data3$warstds)
perf.ch.africa<-performance(pred.ch.africa, "tpr", "fpr")
pred.hs.africa<-prediction(predictors.hs, data3$warstds)
perf.hs.africa<-performance(pred.hs.africa, "tpr", "fpr")
pred.rf.africa<-prediction(predictors.rf, data3$warstds)
perf.rf.africa<-performance(pred.rf.africa, "tpr", "fpr")
auc.fl.africa<-performance(pred.fl.africa, "auc")
auc.fl.africa
auc.ch.africa<-performance(pred.ch.africa, "auc")
auc.ch.africa
auc.hs.africa<-performance(pred.hs.africa, "auc")
auc.hs.africa
auc.rf.africa<-performance(pred.rf.africa, "auc")
auc.rf.africa

###Plot not included in paper###
plot(perf.fl.africa, main="Out of Sample Predictive Accuracy: Africa Data 2001-2014")
plot(perf.ch.africa, lty=2, add=T)
plot(perf.hs.africa, lty=3, add=T)
plot(perf.rf.africa, lty=4, add=T)
legend(0.55, 0.20, c("Fearon and Laitin (2003) 0.43",
                     "Collier and Hoeffler (2004) 0.55",
                      "Hegre and Sambanis (2006) 0.40",
                     "Random Forest 0.60" ),
       lty=c(1,2,3,4), bty="n",
       cex = .75)
```

The code for these plots and scores is fine. However, the scores are meaningless since the numbers are random draws.

```
###csv file for Table 1###
d<-data.frame(data3$warstds, predictors.fl, predictors.ch, predictors.hs, predictors.rf)
write.csv(d, file="CompareCW_dat.csv")
```

This last bit of code unfortunately does not produce Table 1 for the many reasons lined out above. Table 1 on which the authors base their main findings is not reproducible.

Futher comments:

1. The code to produce Figure 3 (Comparison of F_1 Score with varying training set ratio) of the paper is not included in the replication materials.