Machine Learning Script for the Prediction of Stability Score of a country

Adrien Ratsimbaharison

July 16, 2019

## Introduction

In this machine learning implementation, we follow the guidelines suggested by different data scientists who specialize in the use *r* statistical and programming language and particularly the Caret package, created and maintained by Max Kuhn. Among these guidelines, we found particularly useful Saurav Kaushik’s “Practical guide to implement machine learning with CARET in R” and Brett Lanz’s “Machine Learning with R.” After the initial step of installing the Caret package and loading the dataset into r, this machine learning implementation includes the following:

* defining the problem,
* preprocessing the data,
* spliting the data into training and test sets,
* feature selection using the “recursive feature elimination”" or “rfe”" function,
* traning models on the training set,
* generating variable importance,
* making predictions on the test set and assessing the accuracy of the predictions.

## 1. Getting started with loading the package, looking at the data, and defining the problem

Installing and loading the Caret package and its dependencies:

Reading the data in R and looking at its structure:

# Reading the data  
  
stabilityFullDataset <- read.csv("WGI2popDevIneqPovRegimeConflict2.csv", header = TRUE)  
  
stabilityFullDataset <- as.data.frame(stabilityFullDataset)  
  
# Selecting the variables of interest  
  
stabilityFullDataset <- select(stabilityFullDataset, stability, corruptionControl, governmentEffectiveness, regulatoryQuality, ruleOfLaw, voiceAndAccountability, population, GNIperCapita, GDPannualGrowthRate, HDI, GINI, povertyHeadCount, status, inverse\_pr, inverse\_cl, inverse\_mean, politicalChangeFH, conflictHistory, region)  
  
# Correcting the types of some variables  
stabilityFullDataset$population <- as.numeric(stabilityFullDataset$population)  
stabilityFullDataset$inverse\_pr <- as.numeric(stabilityFullDataset$inverse\_pr)  
stabilityFullDataset$inverse\_cl <- as.numeric(stabilityFullDataset$inverse\_cl)  
stabilityFullDataset$inverse\_mean <- as.numeric(stabilityFullDataset$inverse\_mean)  
stabilityFullDataset$conflictHistory <- as.factor(stabilityFullDataset$conflictHistory)  
  
# Looking at its structure of the full dataset  
str(stabilityFullDataset)

## 'data.frame': 3953 obs. of 19 variables:  
## $ stability : num 1.02 0.99 0.99 0.99 1.17 ...  
## $ corruptionControl : num 1.543 1.596 1.658 1.228 0.162 ...  
## $ governmentEffectiveness: num 1.77177 1.97887 2.04215 1.99657 0.00406 ...  
## $ regulatoryQuality : num 1.783 1.778 1.849 1.639 0.244 ...  
## $ ruleOfLaw : num 0.809 0.909 0.862 0.859 0.862 ...  
## $ voiceAndAccountability : num 0.227 1.006 0.945 0.946 1.115 ...  
## $ population : num 83200 87277 90853 94992 97017 ...  
## $ GNIperCapita : num 3114 3106 18286 3331 3395 ...  
## $ GDPannualGrowthRate : num 1.19 1.99 7.62 -3.27 1.98 ...  
## $ HDI : num 0.671 0.676 0.756 0.664 0.673 0.789 0.695 0.777 0.707 0.71 ...  
## $ GINI : num 53.6 42.3 37.3 45.2 40.9 42.2 44.4 38.1 37.3 45.6 ...  
## $ povertyHeadCount : num 14.1 14.6 5.9 2.8 10.7 2.5 3.1 1.5 1.3 6.7 ...  
## $ status : Factor w/ 3 levels "Free","Not Free",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ inverse\_pr : num 7 7 7 7 7 7 7 7 7 7 ...  
## $ inverse\_cl : num 7 7 7 6 7 7 7 7 7 7 ...  
## $ inverse\_mean : num 7 6.5 6.5 6.5 7 7 7 7 7 7 ...  
## $ politicalChangeFH : Factor w/ 3 levels "autocratization",..: 3 1 3 3 2 3 3 3 3 3 ...  
## $ conflictHistory : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ region : Factor w/ 5 levels "Africa","Americas",..: 2 2 2 2 2 2 2 2 2 2 ...

Defining the problem:

The problem in this machine learning is to predict the stability score of a country. In other words, we are dealing here with a machine learning regression on the variable “stability”.

Feature engineering:

# summary statistics of the stability scores  
summary(stabilityFullDataset$stability)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -3.31494 -0.65253 0.14080 0.01739 0.90506 1.96506

boxplot(stabilityFullDataset$stability, ylab = "Political Stability Estimate")  
abline(h = 0, lwd = 2, col = "red")

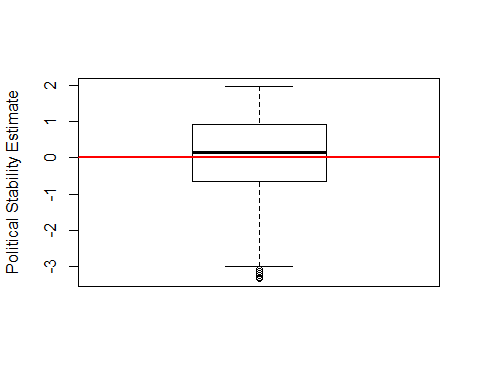


Fig. 1 - Boxplot of the Political Stability Estimate of the World for the Period 1996-2017

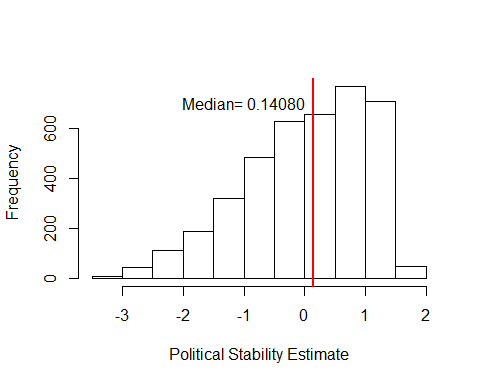


Fig. 2 - Frequency Distribution of Political Stability Estimate of the World for the Period 1996-2017

As shown in these figures, the stability score around the world during the period of 1996-2017 was slightly positive and skewed to the left. This outcome variable needs to be centered and scaled to get meaningful statistical results.

## 2. Pre-processing the data using Caret

In this pre-processing step, we first check for the missing values and remove them.

# Checking for missing values  
sum(is.na(stabilityFullDataset))

## [1] 171

# Removing NAs  
stabilityFullDataset <- na.omit(stabilityFullDataset)  
sum(is.na(stabilityFullDataset))

## [1] 0

Next, we are centering and scaling the numerical values:

# centering and scaling the numerical variable educationLevel  
preProcValues <- preProcess(stabilityFullDataset, method = c("center","scale"))  
  
stabilityFullDataset\_processed <- predict(preProcValues, stabilityFullDataset)  
sum(is.na(stabilityFullDataset\_processed))

## [1] 0

Then, we create “one hot encoding” for the factor variables:

#Converting every categorical variable to numerical using dummy variables  
dmy <- dummyVars(" ~ .", data = stabilityFullDataset\_processed,fullRank = T)  
stabilityFullDataset\_processed <- data.frame(predict(dmy, newdata = stabilityFullDataset\_processed))  
  
#Checking the structure of transformed train file  
str(stabilityFullDataset\_processed)

## 'data.frame': 3782 obs. of 24 variables:  
## $ stability : num 1.016 0.982 0.982 0.982 1.157 ...  
## $ corruptionControl : num 1.482 1.534 1.594 1.175 0.133 ...  
## $ governmentEffectiveness : num 1.677 1.8778 1.9392 1.895 -0.0372 ...  
## $ regulatoryQuality : num 1.727 1.721 1.792 1.584 0.204 ...  
## $ ruleOfLaw : num 0.8 0.899 0.853 0.849 0.853 ...  
## $ voiceAndAccountability : num 0.228 1.009 0.948 0.949 1.119 ...  
## $ population : num -0.258 -0.258 -0.258 -0.258 -0.258 ...  
## $ GNIperCapita : num -0.508 -0.509 0.369 -0.496 -0.492 ...  
## $ GDPannualGrowthRate : num -0.476 -0.336 0.646 -1.255 -0.339 ...  
## $ HDI : num -0.0433 -0.0113 0.5007 -0.0881 -0.0305 ...  
## $ GINI : num 1.8118 0.1794 -0.5429 0.5983 -0.0228 ...  
## $ povertyHeadCount : num 0.0388 0.0669 -0.4213 -0.5952 -0.152 ...  
## $ status.Not.Free : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ status.Partly.Free : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ inverse\_pr : num 1.03 1.03 1.03 1.03 1.03 ...  
## $ inverse\_cl : num 1.166 1.166 1.166 0.625 1.166 ...  
## $ inverse\_mean : num 1.109 0.855 0.855 0.855 1.109 ...  
## $ politicalChangeFH.democratization: num 0 0 0 0 1 0 0 0 0 0 ...  
## $ politicalChangeFH.no.change : num 1 0 1 1 0 1 1 1 1 1 ...  
## $ conflictHistory.1 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ region.Americas : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ region.Asia : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ region.Europe : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ region.Oceania : num 0 0 0 0 0 0 0 0 0 0 ...

## 3. Splitting the data using Caret

In this step, we splitt the dataset into trainSet and testSet based on outcome with a ratio of 75% and 25%, using createDataPartition in Caret.

# Randomizing the dataset before spliting  
set.seed(123)  
rows <- sample(nrow(stabilityFullDataset\_processed))  
stabilityFullDataset\_processed <- stabilityFullDataset\_processed[rows,]  
  
#Spliting dataset into trainSet and testSet  
set.seed(1234)  
index <- createDataPartition(stabilityFullDataset\_processed$stability, p=0.75, list=FALSE)  
stabilityTrainSet <- stabilityFullDataset\_processed[index,]  
stabilityTestSet <- stabilityFullDataset\_processed[-index,]  
  
#Checking the structure of approvalTrainSet  
str(stabilityTrainSet)

## 'data.frame': 2838 obs. of 24 variables:  
## $ stability : num -1.442 1.067 1.237 0.873 0.923 ...  
## $ corruptionControl : num -1.217 -0.207 1.66 1.88 2.04 ...  
## $ governmentEffectiveness : num -0.976 0.348 1.385 1.741 2.043 ...  
## $ regulatoryQuality : num -0.777 0.758 0.603 1.553 2.169 ...  
## $ ruleOfLaw : num -1.099 -0.147 1.071 1.867 1.263 ...  
## $ voiceAndAccountability : num -0.482 -0.841 1.117 1.446 0.113 ...  
## $ population : num 0.684 -0.256 -0.256 -0.105 -0.228 ...  
## $ GNIperCapita : num -0.614 -0.583 0.275 1.936 1.049 ...  
## $ GDPannualGrowthRate : num 0.192 0.586 -0.569 -0.162 -1.072 ...  
## $ HDI : num -0.0561 -0.4016 0.6991 1.4479 0.7695 ...  
## $ GINI : num -0.7162 0.0349 0.3383 -1.0918 0.1794 ...  
## $ povertyHeadCount : num 0.336 -0.191 -0.702 -0.696 -0.494 ...  
## $ status.Not.Free : num 0 1 0 0 0 1 0 0 0 0 ...  
## $ status.Partly.Free : num 1 0 0 0 1 0 1 1 0 0 ...  
## $ inverse\_pr : num -0.368 -1.297 1.026 1.026 -0.833 ...  
## $ inverse\_cl : num -0.457 -0.998 1.166 1.166 -0.998 ...  
## $ inverse\_mean : num -0.414 -1.176 1.109 1.109 -0.922 ...  
## $ politicalChangeFH.democratization: num 1 0 0 0 0 0 0 0 0 0 ...  
## $ politicalChangeFH.no.change : num 0 1 1 1 0 1 1 1 1 1 ...  
## $ conflictHistory.1 : num 1 0 0 0 0 1 0 1 1 0 ...  
## $ region.Americas : num 0 0 1 0 0 0 0 0 0 0 ...  
## $ region.Asia : num 0 1 0 0 1 1 0 1 0 0 ...  
## $ region.Europe : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ region.Oceania : num 0 0 0 1 0 0 1 0 0 1 ...

## 4. Feature selection using Caret

In this step, we use the “recursive feature elimination” or “rfe” function in Caret to identify the best subset of features to be included in the models.

# Feature selection using rfe in caret  
ctrl <- rfeControl(functions = rfFuncs,  
 method = "repeatedcv",  
 repeats = 3,  
 verbose = FALSE)  
  
y <- stabilityTrainSet$stability  
x <- select(stabilityTrainSet, - stability)  
  
stabilityProfile <- rfe(x, y,  
 rfeControl = ctrl)  
  
stabilityProfile

##   
## Recursive feature selection  
##   
## Outer resampling method: Cross-Validated (10 fold, repeated 3 times)   
##   
## Resampling performance over subset size:  
##   
## Variables RMSE Rsquared MAE RMSESD RsquaredSD MAESD Selected  
## 4 0.4673 0.8017 0.3550 0.02333 0.024244 0.020685   
## 8 0.2922 0.9187 0.2158 0.01685 0.010315 0.010779   
## 16 0.2915 0.9191 0.2133 0.01306 0.008342 0.008442 \*  
## 23 0.2952 0.9170 0.2163 0.01421 0.009460 0.008935   
##   
## The top 5 variables (out of 16):  
## population, conflictHistory.1, ruleOfLaw, HDI, GNIperCapita

## 5. Training models on training set using Caret

In this step, we train the generalized linear model (glm) on the train set:

stabilityModel\_glm <- train(stability ~ ., data = stabilityTrainSet,   
 method = "glm")

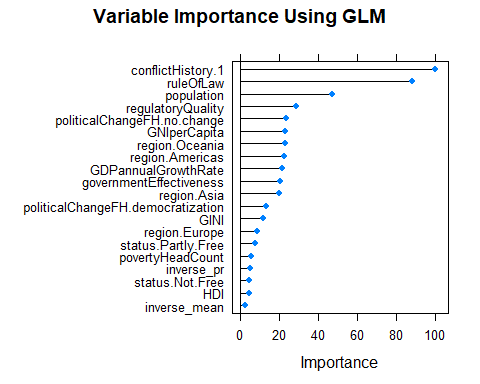
## 6. Variable importance estimation on training set using Caret

In this step, we check the variable importance estimates in Caret by using the “varImp” function" for the glm model.

# Checking variable importance with glm  
# Variable Importance  
varImp(object=stabilityModel\_glm)

## glm variable importance  
##   
## only 20 most important variables shown (out of 23)  
##   
## Overall  
## conflictHistory.1 100.000  
## ruleOfLaw 88.064  
## population 47.263  
## regulatoryQuality 28.836  
## politicalChangeFH.no.change 23.978  
## GNIperCapita 23.488  
## region.Oceania 23.294  
## region.Americas 22.862  
## GDPannualGrowthRate 21.902  
## governmentEffectiveness 20.733  
## region.Asia 19.970  
## politicalChangeFH.democratization 13.584  
## GINI 11.982  
## region.Europe 8.973  
## status.Partly.Free 8.041  
## povertyHeadCount 5.958  
## inverse\_pr 5.412  
## status.Not.Free 4.953  
## HDI 4.789  
## inverse\_mean 2.739

#Plotting Variable importance for GBM model  
plot(varImp(object=stabilityModel\_glm),main="Variable Importance Using GLM", top = 20)



## 7. Making predictions on test set using Caret

#Predictions with glm  
stabilityPrediction\_glm <- predict.train(object=stabilityModel\_glm,stabilityTestSet,type="raw")  
  
summary(stabilityPrediction\_glm)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -1.89951 -0.70039 -0.10612 0.01179 0.75937 1.88204

stabilityModel\_glm

## Generalized Linear Model   
##   
## 2838 samples  
## 23 predictor  
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 2838, 2838, 2838, 2838, 2838, 2838, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 0.5388817 0.7146584 0.4161206

# Assessing the accuracy of the prediction  
  
postResample(pred = stabilityPrediction\_glm, obs = stabilityTestSet$stability)

## RMSE Rsquared MAE   
## 0.5167297 0.7211697 0.3970843

## Conclusion