STA 141A Project (ENR)

Seyoung Jung

12/15/2020

— Step 1: Data loading and procressing —

```
## --- Part a: Upload Metadata for samples ---
setwd("C:/Users/Martin/Desktop/Fall 2020/STA 141A")
path_data<-file.path(getwd(),"Project")
META_DATA<-as_tibble(read.csv(file.path(path_data,"IMPROVE_metadata.csv")))
## --- Filter samples from Korea and Canada ---
US_META<-META_DATA %>% filter(Country %nin% c("KR","CA"))
## --- Filter stats not in continental US ---
US_META<-META_DATA %>% filter(State %nin% c("HI","AK","VI"))
## --- Part b: Load samples data ---
DATA<-as_tibble(read.csv(file.path(path_data,"IMPROVE_2015_data_w_UNC_v2.csv")))
## --- Part c: Select samples from SW given site identifiers from SW_META table ("C
ode")
US_DATA_all<-as_tibble(DATA %>% filter(SiteCode %in% US_META$Code))
```

```
# Let's identify any samples that (grossly) violate PM2.5 mass balances
# PM2.5 (=Y) cannot be negative!
# Since there's some probability that PM2.5 is negative due to errors at low concen
tration, we may use PM2.5 uncertainties to remove samples that fall outside -3*PM2.
5_UNC.
# In this way, we don't risk censoring the data but do remove likely erroneous dat
a.
US_DATA_all<-US_DATA_all %>% dplyr::filter(PM2.5 > -3*PM2.5_UNC)
```

```
exclude<-c("fAbs","PM10","POC","ammNO3","ammSO4","SOIL","SeaSalt","OC1","OC2","OC
3","OC4","EC1","EC2","EC3","fAbs_MDL")
US_DATA_LRG<- US_DATA_all %>% dplyr::select(!contains(exclude) & !matches("_UNC") |
matches("PM2.5_UNC"))
any(is.na(US_DATA_LRG))
```

```
## [1] TRUE
```

```
US_DATA_LRG<-US_DATA_LRG[which(complete.cases(US_DATA_LRG)),]
any(is.na(US_DATA_LRG))
```

```
## [1] FALSE
```

```
## --- Instead of random partitioning, I will partition by first sorting samples by
SiteCode and DATE (already done) and place every other sample in the test set.
# --- This data has seasonality. Sorting by date therefore ensures seasonality is e
quivalent between datasets
n<-nrow(US_DATA_LRG)
ind_test<-seq(1,n,2)
US_DATA_LRG_test<-US_DATA_LRG[ind_test,]
US_DATA_LRG<-US_DATA_LRG[-ind_test,]
```

Categorical => dummy Test

Two of our predictor variables are categorical variables (SiteCode and Date). Hence, we need to convert the variables to dummy variables. Also, in order to use the cv.glmnet function to fit the Elastic Net Regression to the data, input data should be in a matrix format. Also, since the function does not accept formula notation, x and y must be passed in separately. So, we create two different sets for training set.

```
US_DATA_LRG_train_y <- US_DATA_LRG$PM2.5
x_train_cont <- US_DATA_LRG %>%
  select(-PM2.5, -SiteCode, -Date, -PM2.5_UNC) %>%
  as.matrix()
x_train_cat <- US_DATA_LRG %>%
  select(SiteCode, Date) %>%
  model.matrix( ~ .-1, .)
US_DATA_LRG_train_x <- cbind(x_train_cont, x_train_cat)


US_DATA_LRG_test_y <- US_DATA_LRG_test$PM2.5
x_test_cont <- US_DATA_LRG_test %>%
  select(-PM2.5, -SiteCode, -Date, -PM2.5_UNC) %>%
  as.matrix()
x_test_cat <- US_DATA_LRG_test %>%
  select(SiteCode, Date) %>%
  model.matrix( ~ .-1, .)
US_DATA_LRG_test_x <- cbind(x_test_cont, x_test_cat)
```

After converting the factor variables, the training set (US_DATA_LRG_train_x) will have 308 variables.

Now, we will fit models to the training data. By default, the cv.glmnet uses 10-fold cross validation to find the optimal values for lambda. Also, we will use the mean squared error for our evaluation metric. When alpha is 0 (or 1), this function fits Ridge Regression (or Lasso Regression). We will try 20 different values, between 0 and 1, for alpha to find a value that gives us the best result.

```r
set.seed(141)

fits_list <- list()
for (i in 0:20) {
  fits_name <- paste0("Alpha_", i/20)

  fits_list[[fits_name]] <- cv.glmnet(as.matrix(US_DATA_LRG_train_x), as.matrix(US_
DATA_LRG_train_y), type.measure="mse", alpha=i/20, family="gaussian")
}
```

lambda.1se is the value for lambda, stored in each fitted model, that resulted in the simplest model (i.e. the model with the least non-zero parameters) and was within 1 standard error of the lambda that had the smallest sum.

```r
fit_result <- data.frame()
for (i in 0:20) {
  fits_name <- paste0("Alpha_", i/20)

  predict_val <- predict(fits_list[[fits_name]], s=fits_list[[fits_name]]$lambda.1s
e, newx=as.matrix(US_DATA_LRG_test_x))

  mse <- mean((as.matrix(US_DATA_LRG_test_y) - predict_val)^2)

  temp_val <- data.frame(Alpha=i/20, MSE=mse, fits_name=fits_name)
  fit_result <- rbind(fit_result, temp_val)
  }

fit_result
```

```
##    Alpha       MSE  fits_name
## 1   0.00 0.7634257    Alpha_0
## 2   0.05 0.6054706 Alpha_0.05
## 3   0.10 0.6065652  Alpha_0.1
## 4   0.15 0.6123570 Alpha_0.15
## 5   0.20 0.6071487  Alpha_0.2
## 6   0.25 0.6166402 Alpha_0.25
## 7   0.30 0.6084149  Alpha_0.3
## 8   0.35 0.6113124 Alpha_0.35
## 9   0.40 0.6070638  Alpha_0.4
## 10  0.45 0.5977094 Alpha_0.45
## 11  0.50 0.6352535  Alpha_0.5
## 12  0.55 0.6141631 Alpha_0.55
## 13  0.60 0.6186419  Alpha_0.6
## 14  0.65 0.6227428 Alpha_0.65
## 15  0.70 0.6171354  Alpha_0.7
## 16  0.75 0.6373969 Alpha_0.75
## 17  0.80 0.6155697  Alpha_0.8
## 18  0.85 0.6355489 Alpha_0.85
## 19  0.90 0.6405099  Alpha_0.9
## 20  0.95 0.6545169 Alpha_0.95
## 21  1.00 0.6332386    Alpha_1
```

We can see that neither Ridge Regression nor Lasso Regression gives us the best result. Although it gives us a very similar MSE values when alpha is in between 0 and 1, it has the lowest MSE when alpha=0.45. Since we are using Elastic Net Regression, we can expect that this model has less predictor variables than the full model.

For the model with alpha=0.45, cross validation method chooses lambda=0.074. If we take a closer look at a model with alpha=0.45, we can observe that 52 of variables are nonzero. It means that the remaining variables are dropped when fitting this model to the training set.

```
# We can see that the mse is the lowest when alpha = 0.45.
fits_list$Alpha_0.45
```

```
##
## Call:  cv.glmnet(x = as.matrix(US_DATA_LRG_train_x), y = as.matrix(US_DATA_LRG_t
rain_y),      type.measure = "mse", alpha = i/20, family = "gaussian")
##
## Measure: Mean-Squared Error
##
##      Lambda Measure      SE Nonzero
## min 0.00723  0.5523 0.08607     261
## 1se 0.07397  0.6359 0.08232      52
```

```
fits_list$Alpha_0.45$lambda.1se # value for lambda
```

```
## [1] 0.07397486
```

```
coef(fits_list$Alpha_0.45)      # coefficients of this model
```

```
## 309 x 1 sparse Matrix of class "dgCMatrix"
##                               1
## (Intercept)      -0.114674053
## EC                0.426403466
## OC                1.711304246
## OP                0.807417478
## AL                0.971613825
## AS                .
## BR               12.595389511
## CA                1.756914687
## CL                2.745861899
## CR                .
## CU                .
## FE                3.510792738
## PB                8.392527873
## MG                1.062569693
## MN                .
## NI                .
## N2                .
## P                27.117590226
## K                 1.768529297
## RB               85.508506989
## SE              127.200123974
## SI                2.285695803
## NA.               0.362930418
## SR                .
## S                 3.102419694
## TI                9.062758625
## V                18.565709821
## ZN                .
## ZR                .
## NO3               1.178011209
## SO4               0.618401342
## SiteCodeACAD1     .
## SiteCodeAGTI1     .
## SiteCodeBADL1     .
## SiteCodeBALA1     .
## SiteCodeBALD1     .
## SiteCodeBAND1     .
## SiteCodeBIBE1     .
## SiteCodeBIRM1     .
## SiteCodeBLIS1     .
## SiteCodeBLMO1     .
## SiteCodeBOAP1     .
## SiteCodeBOLA1     .
## SiteCodeBOND1     .
## SiteCodeBOWA1     .
## SiteCodeBRCA1     .
## SiteCodeBRID1     .
## SiteCodeBRIG1     .
```

```
## SiteCodeBRIS1     .
## SiteCodeBRMA1     .
## SiteCodeBYIS1     .
## SiteCodeCABA1     .
## SiteCodeCABI1     .
## SiteCodeCACO1     .
## SiteCodeCACR1     .
## SiteCodeCANY1     .
## SiteCodeCAPI1     .
## SiteCodeCEBL1     .
## SiteCodeCHAS1     .
## SiteCodeCHIR1     .
## SiteCodeCLPE1     .
## SiteCodeCOHU1     .
## SiteCodeCORI1    -0.010450120
## SiteCodeCRES1     .
## SiteCodeCRLA1     .
## SiteCodeCRMO1     .
## SiteCodeDOME1     .
## SiteCodeDOSO1     .
## SiteCodeDOUG1     .
## SiteCodeEGBE1     .
## SiteCodeELDO1    -0.136389716
## SiteCodeELLI1     .
## SiteCodeEVER1     .
## SiteCodeFLAT1     .
## SiteCodeFLTO1     .
## SiteCodeFOPE1     .
## SiteCodeFRES1     .
## SiteCodeFRRE1     .
## SiteCodeGAMO1     .
## SiteCodeGICL1     .
## SiteCodeGLAC1     .
## SiteCodeGRBA1     .
## SiteCodeGRCA2     .
## SiteCodeGRGU1     .
## SiteCodeGRRI1     .
## SiteCodeGRSA1     .
## SiteCodeGRSM1     .
## SiteCodeGUMO1     .
## SiteCodeHECA1     .
## SiteCodeHEGL1     .
## SiteCodeHOOV1     .
## SiteCodeIKBA1     .
## SiteCodeISLE1     .
## SiteCodeJARB1     .
## SiteCodeJARI1     .
## SiteCodeJOSH1     .
## SiteCodeKAIS1     .
## SiteCodeKALM1     .
## SiteCodeLABE1     .
```

```
## SiteCodeLASU2    .
## SiteCodeLAVO1    .
## SiteCodeLIGO1    .
## SiteCodeLOND1    .
## SiteCodeLOST1    .
## SiteCodeLTCC1    .
## SiteCodeLYEB1    .
## SiteCodeMACA1    .
## SiteCodeMAKA2    0.002197642
## SiteCodeMAVI1    0.088369079
## SiteCodeMEAD1    .
## SiteCodeMELA1    .
## SiteCodeMEVE1    .
## SiteCodeMING1    .
## SiteCodeMOHO1    .
## SiteCodeMOMO1    .
## SiteCodeMONT1    .
## SiteCodeMOOS1    .
## SiteCodeMORA1    .
## SiteCodeMOZI1    .
## SiteCodeNEBR1    .
## SiteCodeNOAB1    .
## SiteCodeNOCA1    .
## SiteCodeNOCH1    .
## SiteCodeNOGA1    .
## SiteCodeOKEF1    .
## SiteCodeOLYM1    .
## SiteCodeORPI1    .
## SiteCodeOWVL1    .
## SiteCodePACK1    .
## SiteCodePASA1    .
## SiteCodePEFO1    .
## SiteCodePENO1    .
## SiteCodePHOE1   -0.127508792
## SiteCodePHOE5    .
## SiteCodePINN1    .
## SiteCodePMRF1    .
## SiteCodePORE1    0.660387523
## SiteCodePRIS1    .
## SiteCodePUSO1    .
## SiteCodeQUCI1    .
## SiteCodeQURE1    .
## SiteCodeQUVA1    .
## SiteCodeRAFA1    .
## SiteCodeREDW1    0.017723498
## SiteCodeROMA1    .
## SiteCodeROMO1    .
## SiteCodeSACR1    .
## SiteCodeSAGA1   -0.103433378
## SiteCodeSAGO1    .
## SiteCodeSAGU1    .
```

```
## SiteCodeSAMA1    0.074553728
## SiteCodeSAPE1    .
## SiteCodeSAWE1    .
## SiteCodeSAWT1    .
## SiteCodeSENE1    .
## SiteCodeSEQU1    .
## SiteCodeSHEN1    .
## SiteCodeSHMI1    .
## SiteCodeSHRO1    .
## SiteCodeSIAN1    .
## SiteCodeSIPS1    .
## SiteCodeSNPA1    .
## SiteCodeSTAR1    .
## SiteCodeSTIL1    .
## SiteCodeSULA1    .
## SiteCodeSWAN1    .
## SiteCodeSYCA1    .
## SiteCodeSYCA2    .
## SiteCodeTALL1    .
## SiteCodeTHBA1    .
## SiteCodeTHRO1    .
## SiteCodeTHSI1    .
## SiteCodeTONT1    .
## SiteCodeTRIN1    .
## SiteCodeULBE1    .
## SiteCodeUPBU1    .
## SiteCodeVILA1    .
## SiteCodeVOYA2    .
## SiteCodeWASH1    .
## SiteCodeWEMI1    .
## SiteCodeWHIT1    .
## SiteCodeWHPA1    .
## SiteCodeWHPE1    .
## SiteCodeWHRI1    .
## SiteCodeWICA1    .
## SiteCodeWIMO1    .
## SiteCodeYELL2    .
## SiteCodeYOSE1    .
## SiteCodeZICA1    .
## Date1/15/2015    .
## Date1/18/2015    .
## Date1/21/2015    .
## Date1/24/2015    .
## Date1/27/2015    .
## Date1/3/2015     .
## Date1/30/2015    .
## Date1/6/2015     .
## Date1/9/2015     .
## Date10/12/2015   .
## Date10/15/2015   .
## Date10/18/2015   .
```

```
## Date10/21/2015    .
## Date10/24/2015    .
## Date10/27/2015    .
## Date10/3/2015     .
## Date10/30/2015    .
## Date10/6/2015     .
## Date10/9/2015     .
## Date11/11/2015    .
## Date11/14/2015    .
## Date11/17/2015    .
## Date11/2/2015     .
## Date11/20/2015    .
## Date11/23/2015    .
## Date11/26/2015    .
## Date11/29/2015    .
## Date11/5/2015     .
## Date11/8/2015     .
## Date12/11/2015    .
## Date12/14/2015    .
## Date12/17/2015    .
## Date12/2/2015     .
## Date12/20/2015    .
## Date12/23/2015    .
## Date12/26/2015    .
## Date12/29/2015    .
## Date12/5/2015     .
## Date12/8/2015     .
## Date2/11/2015     .
## Date2/14/2015     .
## Date2/17/2015     .
## Date2/2/2015      .
## Date2/20/2015     .
## Date2/23/2015     .
## Date2/26/2015     .
## Date2/5/2015      .
## Date2/8/2015      .
## Date3/1/2015      .
## Date3/10/2015     .
## Date3/13/2015   -0.011986774
## Date3/16/2015     .
## Date3/19/2015   -0.026328360
## Date3/22/2015   -0.076507845
## Date3/25/2015     .
## Date3/28/2015     .
## Date3/31/2015     .
## Date3/4/2015    -0.009827235
## Date3/7/2015    -0.258708631
## Date4/12/2015     .
## Date4/15/2015     .
## Date4/18/2015     .
## Date4/21/2015     .
```

```
## Date4/24/2015    .
## Date4/27/2015    .
## Date4/3/2015     .
## Date4/30/2015    .
## Date4/6/2015     .
## Date4/9/2015     .
## Date5/12/2015    .
## Date5/15/2015    .
## Date5/18/2015    .
## Date5/21/2015    .
## Date5/24/2015    .
## Date5/27/2015    .
## Date5/3/2015     .
## Date5/30/2015    .
## Date5/6/2015     .
## Date5/9/2015     .
## Date6/11/2015    0.284272033
## Date6/14/2015    .
## Date6/17/2015    .
## Date6/2/2015     .
## Date6/20/2015    .
## Date6/23/2015    0.044890657
## Date6/26/2015    .
## Date6/29/2015    0.054805817
## Date6/5/2015     .
## Date6/8/2015     .
## Date7/11/2015    .
## Date7/14/2015    .
## Date7/17/2015    .
## Date7/2/2015     0.318254665
## Date7/20/2015    .
## Date7/23/2015    .
## Date7/26/2015    .
## Date7/29/2015    0.186479582
## Date7/5/2015     0.026638887
## Date7/8/2015     .
## Date8/1/2015     .
## Date8/10/2015    0.003149496
## Date8/13/2015    0.003635989
## Date8/16/2015    0.157029484
## Date8/19/2015    0.288633654
## Date8/22/2015    .
## Date8/25/2015    0.227724937
## Date8/28/2015    0.222549734
## Date8/31/2015    0.123963928
## Date8/4/2015     0.195171963
## Date8/7/2015     0.147065416
## Date9/12/2015    .
## Date9/15/2015    .
## Date9/18/2015    0.027475458
## Date9/21/2015    .
```

```
## Date9/24/2015    .
## Date9/27/2015    .
## Date9/3/2015     0.091789732
## Date9/30/2015    .
## Date9/6/2015     .
## Date9/9/2015     .
```