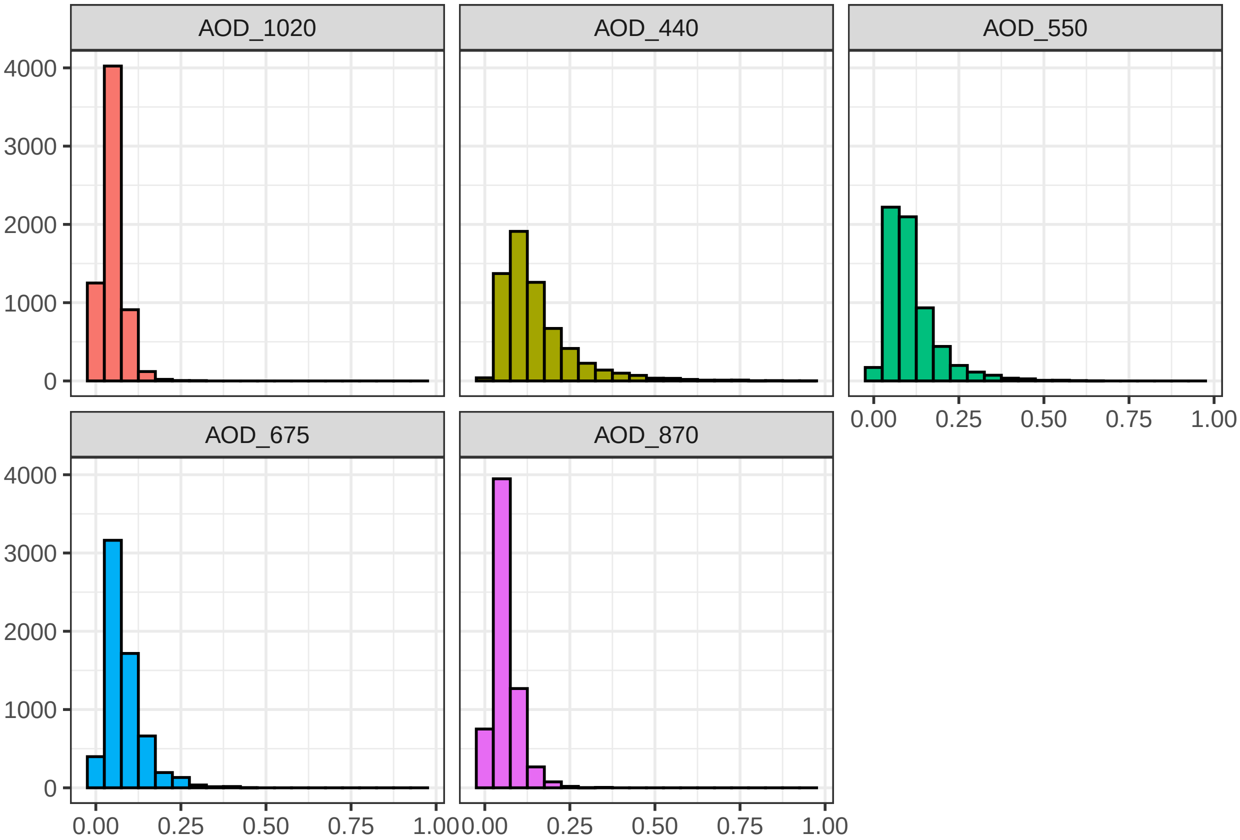
AERONET - Interpolate AOD 550nm & XGBoost by Regions

Meytar Sorek-Hamer, Meredith Franklin, Khang Chau

January 24, 2022

Regional Inversion Data

AOD Histograms (observed vs. interpolated at 550nm)

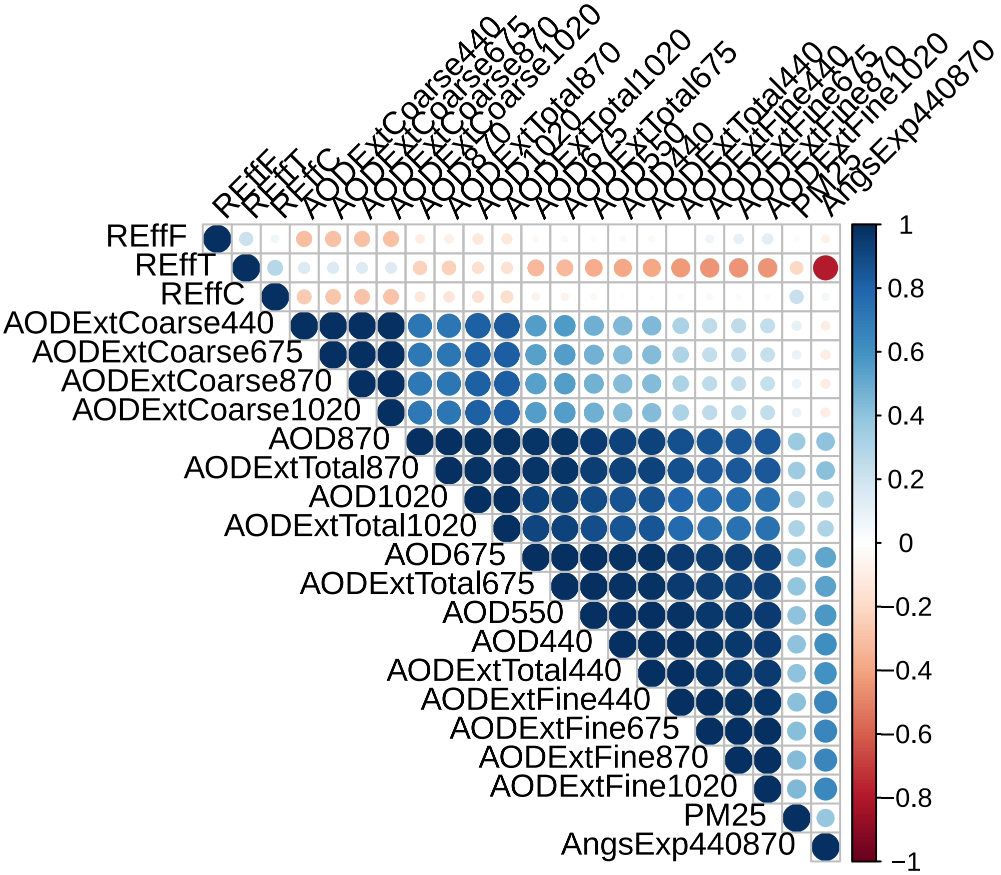


Summary of PM2.5 and AOD (550nm) relationship

Region	N	PM2.5 (SD)	AOD (SD)	R2	RMSE
Central	611	7.15 (3.42)	0.063 (0.048)	0.161	3.131
East	341	8.68 (5.57)	0.123 (0.091)	0.544	3.756
St. Louis	1161	10.52 (5.27)	0.121 (0.094)	0.268	4.509
West	3310	14.61 (12.49)	0.111 (0.066)	0.177	11.330

Region	Site
Central	Denver LaCasa
Central	DRAGON Chatfield Pk
Central	DRAGON DenverLaCasa
East	DRAGON LAREL
East	DRAGON PATUX
East	NASA LaRC
East	USDA-Howard
St. Louis	St Louis University
West	DRAGON Bakersfield
West	DRAGON Garland
West	DRAGON Madera City
West	DRAGON Tranquility
West	Fresno
West	Fresno 2

Correlations (possible multiple collinearity)

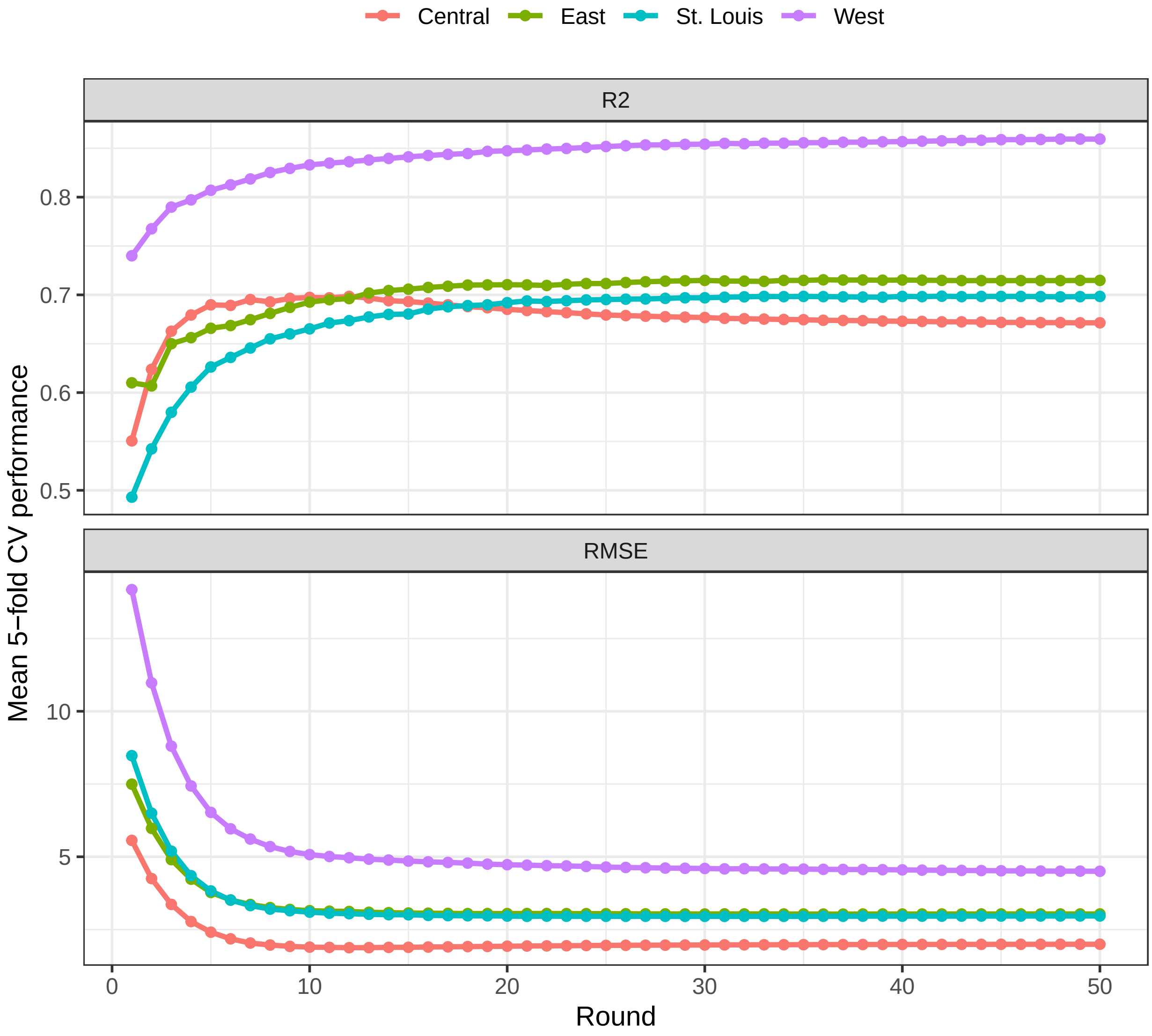


XGBoost Modeling

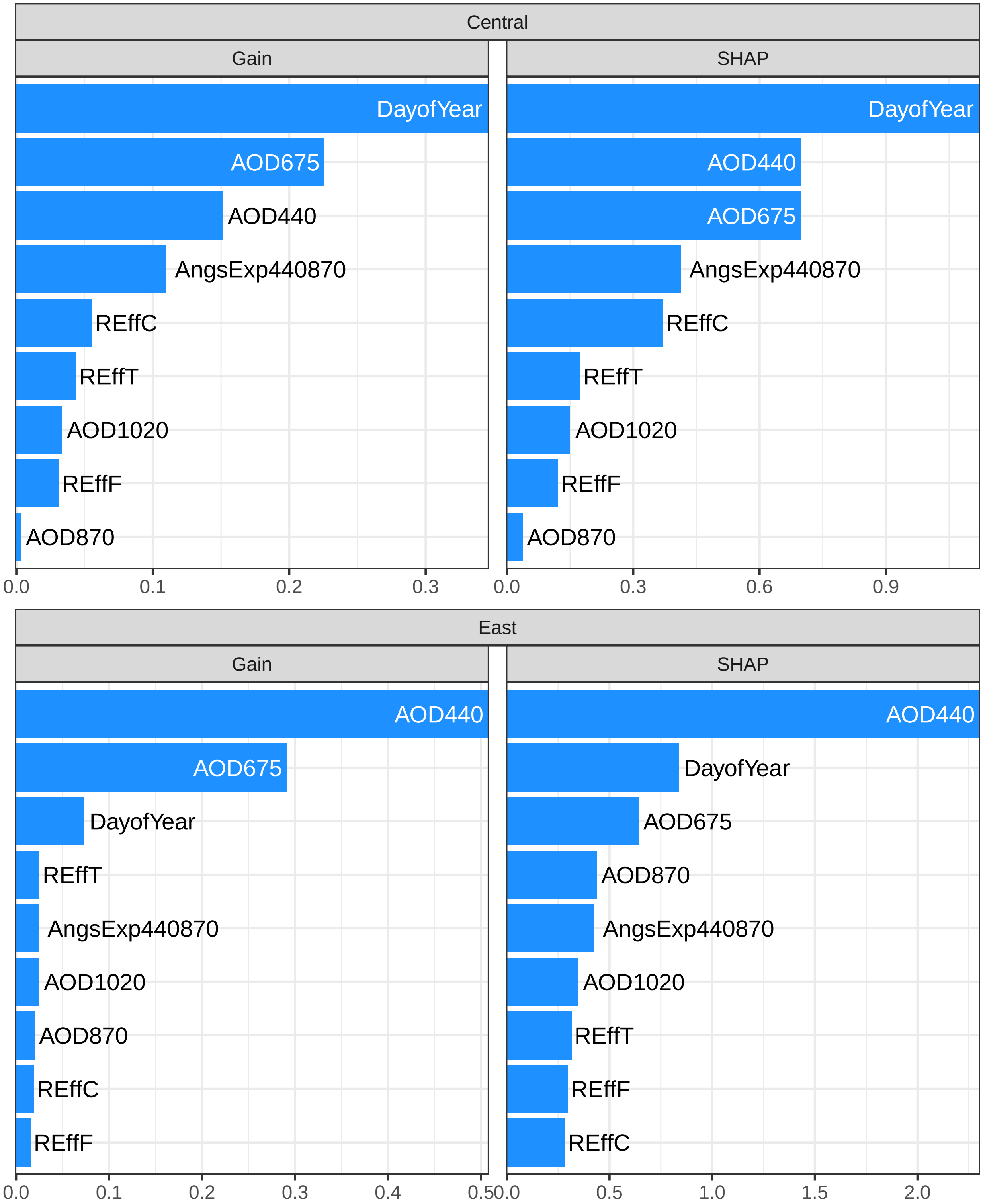
Performance

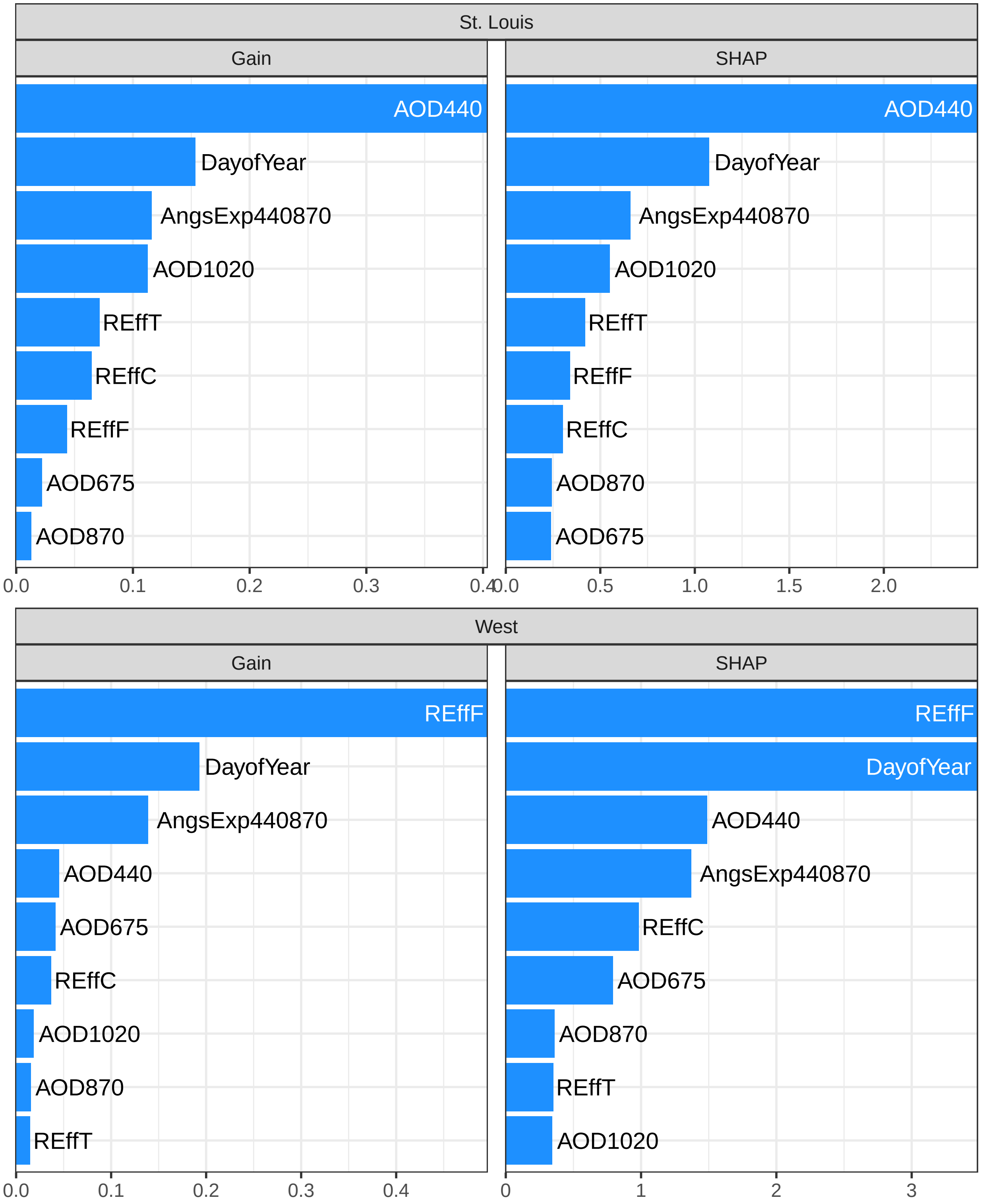
Region	Set	N	Round	R2	RMSE
Central	Train	430	12	0.870	1.250
Central	Test	181	12	**0.724**	**1.838**
East	Train	241	37	0.997	0.331
East	Test	100	37	**0.739**	**2.624**
St. Louis	Train	814	29	0.879	1.853
St. Louis	Test	347	29	**0.714**	**2.777**
West	Train	2318	50	0.982	1.701
West	Test	992	50	**0.919**	**3.440**

Cross-validation results



XGBoost feature importance




St. Louis
Gain
SHAP
AOD440
DayofYear
AngsExp440870
AOD1020
REffT
REffC
REffF
AOD675
AOD870
AOD440
DayofYear
AngsExp440870
AOD1020
REffT
REffF
REffC
AOD870
AOD675
0.0
0.1
0.2
0.3
0.4
0.0
0.5
1.0
1.5
2.0
West
Gain
SHAP
REffF
DayofYear
AngsExp440870
AOD440
AOD675
REffC
AOD1020
AOD870
REffT
REffF
DayofYear
AOD440
AngsExp440870
REffC
AOD675
AOD870
REffT
AOD1020
0.0
0.1
0.2
0.3
0.4
0
1
2
3

Source code: xgboost-pm25-regions.R

```r
# Packages and directories ------------------------------------------------
pkgs = c("car", "caret", "data.table", "ggvis", "Hmisc", "knitr", "leaps",
         "ModelMetrics", "pracma", "SHAPforxgboost", "tidyverse", "xgboost")
for(p in pkgs) require(p, character.only = TRUE)
rm(p, pkgs)
theme_set(theme_bw())

code_dir = ifelse(
  grepl("code/XGBOOST", getwd()),
  paste0(getwd(), "/"),
  paste0(getwd(), "/code/XGBOOST/"))
data_dir = gsub("code/XGBOOST", "data", code_dir)
res_dir = paste0(data_dir, "results/")

# SHAP functions
source(paste0(code_dir, "shap.R"), local = knitr::knit_global())
source(paste0(code_dir, "plot.shap.summary.R"),
       local = knitr::knit_global()) #Thanks to Y.LIU (MSSM)

# Load Working DataBase inversion dataset --------------------------------
wdb.inv = fread(paste0(data_dir, "inv.pm.data.csv")) %>%
  distinct() %>%
  as_tibble()

# Interpolate AOD 550 -----------------------------------------------------
wdb.inv = wdb.inv %>%
  # rename variables for convenience
  rename_with(.fn = ~gsub("-", "", .), .cols = everything()) %>%
  rename_with(.fn = ~gsub("\\[", "", .), .cols = everything()) %>%
  rename_with(.fn = ~gsub("nm]", "", .),
              .cols = contains("AOD_Coincident_Input")) %>%
  rename_with(.fn = ~gsub("Coincident_Input", "", .),
              .cols = contains("AOD_Coincident_Input")) %>%
  rename_with(.fn = ~gsub("AOD_Extinction", "AODExt", .),
              .cols = contains("AOD_Extinction")) %>%
  rename_with(.fn = ~gsub("nm]", "", .),
              .cols = contains("nm]")) %>%
  rename_with(.fn = ~gsub("Angstrom_Exponent", "AngsExp", .),
              .cols = contains("Angstrom_Exponent")) %>%
  rename(AngsExp440870 = `AngsExp_440870nm_from_Coincident_Input_AOD`)
log_waves = log(c(440, 675, 870, 1020))
wdb.inv = wdb.inv%>%
  mutate(AOD_550 = apply(
    X = wdb.inv %>% select(AOD_440:AOD_1020),
    MARGIN = 1,
    ## See aod_interpolation.R
    ## because AOD columns are complete, skipping the NA handling codes
    FUN = function(x) polyfit(log_waves, log(x), 2) %>%
      polyval(log(550)) %>% exp()))

# Define regions and petition data into regions --------------------------
sites.regions = list(
  data.table(Region = "West",
             Site_Name = c("Fresno_2", "Fresno", "DRAGON_Madera_City",
                           "DRAGON_Tranquility", "DRAGON_Bakersfield",
                           "DRAGON_Garland", "Modesto")),
  data.table(Region = "Central",
             Site_Name = c("Denver_LaCasa", "DRAGON_DenverLaCasa",
                           "DRAGON_Chatfield_Pk", "NEON_RNMP")),
  data.table(Region = "St. Louis",
             Site_Name = "St_Louis_University"),
  data.table(Region = "East",
             Site_Name = c("USDA-Howard", "DRAGON_LAREL", "DRAGON_PATUX",
                           "NASA_LaRC", "DRAGON_Essex", "Big_Meadows"))) %>%
  rbindlist()
wdb.inv = left_join(wdb.inv, sites.regions, by = "Site_Name")

# wdb.inv[, c(4:5, 7:19, 22, 73:74)] %>%
wdb.regions = wdb.inv %>%
  select(Region, PM25, Day_of_Year,
         starts_with("AOD_"), AOD_550, AngsExp440870,
         starts_with("AODExt"), starts_with("REff")) %>%
  rename_with(~ gsub("_", "", .x)) %>%
  rename_with(~ gsub("-", "", .x)) %>%
  rename_with(~ gsub("\\[", "", .x), fixed = TRUE) %>%
  rename_with(~ gsub("]", "", .x)) %>%
  filter(!is.na(Region)) %>%
  arrange(Region) %>%
  data.table() %>% split(by = "Region")

wdb.regions = lapply(wdb.regions, as_tibble)

# AOD Variable sets -------------------------------------------------------
res_set = "-v4" # "-v1" # "-nodoy" # "-simplified" # "-v4"

if(res_set == "-v1") {
  var_set =
    c('AngsExp440870', 'AOD1020', 'AOD440', 'AOD550', 'AOD675', 'AOD870',
      'AODExtCoarse1020', 'AODExtCoarse440', 'AODExtCoarse675', 'AODExtCoarse870',
      'AODExtFine1020', 'AODExtFine440', 'AODExtFine675', 'AODExtFine870',
      'AODExtTotal1020', 'AODExtTotal440', 'AODExtTotal675', 'AODExtTotal870',
      'DayofYear', 'REffC', 'REffF', 'REffT')
```

```r
} else if(res_set == "-nodoy") {
  var_set =
    c('AngsExp440870', 'AOD1020', 'AOD440', 'AOD550', 'AOD675', 'AOD870',
      'AODExtCoarse1020', 'AODExtCoarse440', 'AODExtCoarse675', 'AODExtCoarse870',
      'AODExtFine1020', 'AODExtFine440', 'AODExtFine675', 'AODExtFine870',
      'AODExtTotal1020', 'AODExtTotal440', 'AODExtTotal675', 'AODExtTotal870',
      'REffC', 'REffF', 'REffT')
} else if(res_set == "-simplified") {
  var_set =
    c('AngsExp440870', 'AOD1020', 'AOD440', 'AOD675', 'AOD870',
      'DayofYear', 'REffC', 'REffF', 'REffT')
} else if(res_set == "-v4") {
  var_set =
    c('AngsExp440870',
      'AODExtCoarse1020', 'AODExtCoarse440', 'AODExtCoarse675', 'AODExtCoarse870',
      'AODExtFine1020', 'AODExtFine440', 'AODExtFine675', 'AODExtFine870',
      'AODExtTotal1020', 'AODExtTotal440', 'AODExtTotal675', 'AODExtTotal870',
      'DayofYear', 'REffC', 'REffF')
} else{
  warning("UNCLEAR VARIABLE SET. PLEASE CHECK.")
}

# XGBoost modeling via 5-fold CV ------------------------------------------
split = 0.70
folds = 5
nrounds_max = 50

# xgb_params = list(eta = 0.05, gamma = 0.01, max_depth = 5,
#                   min_child_weight = 0, subsample = 0.5,
#                   colsample_bytree = 0.75)

xgb.regions = list()
xgb.cv.regions = list()
xgb.res.regions = list()
xgb.imp.regions = list()

set.seed(20200816)
for(r in seq_len(length(wdb.regions))) {
  message(names(wdb.regions[r]))

  # prep data for xgboost
  data_2 = wdb.regions[[r]] %>%
    select(all_of(var_set))

  data_dmy = dummyVars(" ~ .", data = data_2, fullRank = TRUE)
  data_x = predict(data_dmy, newdata = data_2)
  data_y = wdb.regions[[r]]$PM25
```

```r
# create train/split and CV partitions
train_idx = createDataPartition(data_y, p = split, list = FALSE)
data_x_train = data_x[train_idx, ]; data_y_train = data_y[train_idx]
data_x_test = data_x[-train_idx, ]; data_y_test = data_y[-train_idx]
train_cv_idx = createFolds(data_y_train, k = folds)

# run through nround
cv.res = list()
for(n in seq_len(nrounds_max)) {
  # for each nround, conduct k-fold cross-valiation
  kfolds.res = list()
  for(f in seq_len(folds)) {
    # find fold indices
    fold_idx = train_cv_idx[[f]]
    # fit model on k-1 folds
    model_xgb = xgboost(data = data_x_train[-fold_idx, ],
                        nrounds = n,
                        # params = xgb_params,
                        objective = "reg:squarederror",
                        label = data_y_train[-fold_idx],
                        verbose = 0)
    # extract OOS R2 and RMSE
    kfolds.res[[f]] = data.table(
      r2 = round(cor(predict(model_xgb, data_x_train[fold_idx, ]),
                     data_y_train[fold_idx]) ** 2, 3),
      rmse = round(sqrt(mean((predict(model_xgb,
                                      data_x_train[fold_idx, ]) -
                                data_y_train[fold_idx]) ** 2)), 3))
  }
  # get mean R2 and RMSE for current nround
  cv.res[[n]] = rbindlist(kfolds.res) %>%
    summarize(R2 = mean(r2), RMSE = mean(rmse)) %>%
    mutate(Round = n)
}
# aggregate mean R2 and RMSE
xgb.cv.regions[[r]] = rbindlist(cv.res) %>%
  mutate(Region = names(wdb.regions[r]))

# get best nround based on RMSE
best_nround = xgb.cv.regions[[r]] %>%
  arrange(RMSE) %>% slice_head(n = 1) %>%
  pull(Round)
# fit best model
xgb.regions[[r]] = xgboost(data = data_x_train,
                           label = data_y_train,
                           nrounds = best_nround,
                           objective = "reg:squarederror",
                             verbose = 0)
  # calculate SHAP values
  shap_result = shap.score.rank(xgb_model = xgb.regions[[r]],
                                X_train = data_x_train,
                                shap_approx = FALSE)
  # collect Gain and SHAP importance values
  xgb.imp.regions[[r]] = inner_join(
    xgb.importance(model = xgb.regions[[r]]),
    data.table(
      Feature = names(shap_result$mean_shap_score),
      SHAP = shap_result$mean_shap_score),
    by = "Feature") %>%
    select(Feature, Gain, SHAP) %>%
    mutate(Region = names(wdb.regions[r]),
           Feature = gsub("nmfromCoincidentInputAOD", "", Feature))

  # collect performance by train/test sets
  xgb.res.regions[[r]] = data.table(
    Set = c("Train", "Test"),
    N = c(nrow(data_x_train), nrow(data_x_test)),
    Round = rep(best_nround, 2),
    R2 = c(round(cor(predict(xgb.regions[[r]], data_x_train),
                     data_y_train) ** 2, 3),
           round(cor(predict(xgb.regions[[r]], data_x_test),
                     data_y_test) ** 2, 3)),
    RMSE = c(round(sqrt(mean((predict(xgb.regions[[r]], data_x_train) -
                                data_y_train) ** 2)), 3),
             round(sqrt(mean((predict(xgb.regions[[r]], data_x_test) -
                                data_y_test) ** 2)), 3)),
    Region = rep(names(wdb.regions[r]), 2))

  # clear memory
  rm(data_2, data_dmy,
     data_x, data_x_train, data_x_test,
     data_y, data_y_train, data_y_test,
     train_idx, train_cv_idx, cv.res, shap_result)
  gc(); gc(reset = TRUE)
}
names(xgb.regions) = names(xgb.cv.regions) = names(xgb.res.regions) =
  names(xgb.imp.regions) = names(wdb.regions)

# output modeling results for convenience
list(Models = xgb.regions,
     CV = reduce(xgb.cv.regions, rbind) %>% as_tibble(),
     Performance = reduce(xgb.res.regions, rbind) %>% as_tibble(),
     Importance = reduce(xgb.imp.regions, rbind) %>% as_tibble()
     ) %>%
  saveRDS(paste0(data_dir, "results/xgb-pm25-results", res_set, ".rds"))
```