Appendix

Charlotte Abrams, Laura Cosgrove, Alyssa Vanderbeek

7 April 2019

Data prep

```
## Data import
predictors <- read_csv("./data/Training_values.csv")
response <- read_csv("./data/Training_labels.csv")


## Manipulation
data <- response %>%
  full_join(predictors, by = "row_id") %>%
  separate(col = area__rucc, into = c('metro', 'population'), sep = ' - ') %>%
  rename(urban_influence = area__urban_influence,
         economic_typology = econ__economic_typology) %>%
  mutate(pure_population = fct_collapse(as.factor(population),
                                        "more_than_1mil" = "Counties in metro areas of 1 million popula
                                        "250k_to_1mil" = "Counties in metro areas of 250,000 to 1 milli
                                        "less_than_250k" = "Counties in metro areas of fewer than 250,0
                                        "more_than_20k" = c("Urban population of 20,000 or more, adjace
                                                            "Urban population of 20,000 or more, not
                                        "2500_to_20k" = c("Urban population of 2,500 to 19,999, adjacen
                                                          "Urban population of 2,500 to 19,999, not a
                                        "less_than_2500" = c("Completely rural or less than 2,500 urban
                                                             "Completely rural or less than 2,500 urban popul
         economic_typology = as.factor(recode(economic_typology,
                                              "Nonspecialized" = "Nonspecialized",
                                              "Manufacturing-dependent" = "Manufacturing",
                                              "Farm-dependent" = "Farming",
                                              "Federal/State government-dependent" = "Government",
                                              "Mining-dependent" = "Mining",
                                              "Recreation" = "Recreation")),
         metro = factor(metro,
                        levels = c("Metro", "Nonmetro")),
         urban_influence = str_replace_all(urban_influence, " |/|-", "_"), # replace problematic charact
         urban_influence = str_replace_all(urban_influence, ",", ""), # replace problematic characters
         demo__pct_nonwhite = demo__pct_hispanic + demo__pct_asian + demo__pct_american_indian_or_alaska
         urban_influence = fct_rev(urban_influence),
         metro_adjacency = fct_collapse(population,
                                        metro = c("Counties in metro areas of 1 million population or mo
                                                  "Counties in metro areas of 250,000 to 1 million popul
                                                  "Counties in metro areas of fewer than 250,000 populat
                                        adjacent = c("Urban population of 20,000 or more, adjacent to a
                                                     "Urban population of 2,500 to 19,999, adjacent to a
                                                     "Completely rural or less than 2,500 urban populati
                                        nonadjacent = c("Urban population of 20,000 or more, not adjacen
                                                        "Urban population of 2,500 to 19,999, not adjace
                                                        "Completely rural or less than 2,500 urban popul
```

```
  dplyr::select(-demo__pct_hispanic,
                -demo__pct_asian,
                -demo__pct_american_indian_or_alaskan_native,
                -demo__pct_non_hispanic_african_american,
                -demo__pct_non_hispanic_white,
                -health__homicides_per_100k, # >90% missing
                -health__pct_excessive_drinking, # >90% missing
                -yr,
                -population,
                -row_id)
```

Training and testing data split. Imputation on missing data.

```
## training/test data
set.seed(1)
train_ind <- sample(seq_len(nrow(data)), size = 2/3*nrow(data)) # select rows in 2:1 ratio

train <- data[train_ind, ] # training dataset
test <- data[-train_ind, ] # testing dataset

# Imputation for missing values with caret, based on training data
training_preproc = caret::preProcess(train[,-1],
                                     method = "knnImpute", # automatically centers and scales data
                                     pcaComp = 10,
                                     na.remove = TRUE,
                                     k = 5,
                                     knnSummary = mean,
                                     outcome = NULL,
                                     fudge = .2,
                                     numUnique = 3,
                                     verbose = TRUE)

# Impute training imputation on both training and testing datasets
train_imputed = predict(training_preproc, train)
test_imputed = predict(training_preproc, test)

#save files to Rdata: was not saving the factor structure in read from csv
saveRDS(train_imputed, file = './data/train_imputed.Rdata')
saveRDS(train_imputed, file = './data/test_imputed.Rdata')
```

Linear Models

Set up `caret` training control. We will use this for all models.

```
set.seed(100)
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 5)
```

Stepwise regression:

```
set.seed(2)
step.fit = caret::train(x, y,
                        method = 'glmStepAIC',
                        metric = "RMSE",
                        trControl = ctrl)
saveRDS(step.fit, "lm_step_imputed.rds")
```

Lasso:

```
set.seed(2)
lasso_fit <- caret::train(x, y,
                          method = "glmnet",
                          metric = "RMSE",
                          tuneGrid = expand.grid(alpha = 1,
                                                 lambda = exp(seq(-2, 4, length = 200))),
                          trControl = ctrl)
plot(lasso_fit, xTrans = function(x) log(x)) #in correct range

saveRDS(lasso_fit, "lasso_imputed.rds")
```

Ridge:

```
set.seed(100)

ridge_fit <- caret::train(x, y,
                      method = "glmnet",
                      tuneGrid = expand.grid(alpha = 0,
                                             lambda = exp(seq(-2, 10, length = 200))),
                     trControl = ctrl1)


plot(ridge_fit, xTrans = function(x) log(x)) #in correct range

best_lambda_ridge = ridge_fit$bestTune$lambda

saveRDS(ridge_fit, "ridge.rds")
```

PCR:

```
set.seed(2)
pcr_fit <- caret::train(x, y,
                        method = "pcr",
                        trControl = ctrl,
                        metric = "RMSE",
                        tuneLength = 200)
saveRDS(pcr_fit, "pcr_imputed.rds")
```

Non-linear models

GAM:

```
set.seed(2)
gam_fit <- caret::train(x, y,
                        method = "gam",
                        metric = 'RMSE',
                        tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE, FALSE)),
                        trControl = ctrl)
summary(gam_fit)
saveRDS(gam_fit, "gam_fit_imputed.rds")
```

MARS:

```
mars_grid <- expand.grid(degree = 1:3, # degree: 1 vs 2 vs 3, no interaction vs. interaction;
                         nprune = 10:40) # nprune is number of coef
```

```r
set.seed(2)

mars_fit <- caret::train(x, y,
                         method = "earth",
                         tuneGrid = mars_grid,
                         trControl = ctrl)

saveRDS(mars_fit, "mars.rds")

#based on initial results, choose parsimonious version
mars_grid_refined <- expand.grid(degree = 2, nprune = 25:40)

mars_fit_refined <- caret::train(x, y,
                                 method = "earth",
                                 tuneGrid = mars_grid_refined,
                                 trControl = ctrl)
saveRDS(mars_fit_refined, "mars2.rds")
```