Economics 1152/SUP 135
Spring 2019

Professor Raj Chetty and Dr. Gregory Bruich
Department of Economics, Harvard University

Empirical Project 4
Using Google DataCommons to Predict Social Mobility

Posted on Thursday, April 18, 2019
Due at 11:59 p.m. on Thursday, May 2, 2019
2 extra credit points if Part 1 submitted by 11:59 p.m. on Thursday, April 25

In this Empirical Project, you will use variables from Google DataCommons to predict intergenerational mobility using machine learning methods. The measure of intergenerational mobility that we will focus on is the mean rank of a child whose parents were at the 25th percentile of the national income distribution in each county (`kfr_pooled_p25`). Your goal is to construct the best predictions of this outcome using other variables, an important step in creating forecasts of upward mobility that could be used for future generations before data on their outcomes become available.

The "training" dataset is a 50% random sample of all counties with at least 10,000 residents available from the Opportunity Atlas. You will use predictors from Google DataCommons to predict the variable `kfr_pooled_p25` in the *other* half of these data. There are about 5,000 possible covariates available from Google DataCommons! We have included 121 predictors in these data already. Part of the assignment is to carefully select at least 10 more predictors from Google DataCommons to use in your prediction algorithm.

The assignment has three parts:

1. *Data set up*. In the first part, you are asked to select at least 10 predictors from DataCommons. Download these data for all counties using the Bulk Downloads link. Merge these data with the atlas_training.dta data file. Produce descriptive statistics and run a simple linear regression using these combined data.

   In an effort to encourage you to start this project early, **we will award you two extra credit points if Part 1 is submitted on Canvas by 11:59 p.m. on April 25.**

2. *Prediction challenge*. The second section is the main part. Using the training data, you will construct a prediction algorithm that produces good out-of-sample predictions of `krf_pooled_p25`.

3. *Out-of-sample validation*. After completing Part 2, you will evaluate your predictions in the test data, which consists of the *other* half of the data. For this part, you will use the atlas_test.dta data file. You will merge your predictions from part 2 with these data, and assess the performance of your prediction algorithm.

Instructions

Please submit your Empirical Project on Canvas. Your submission should include three files:

1. A word or pdf document with output described below, and responses to the questions asked below, which will be easily 4-6 pages but there is no maximum page limit.

2. A do-file with your STATA code or an .R script file with your R code

3. A log file of your STATA or R output

Part 1: Data set up

1. Go to Google DataCommons and select at least 10 county-level variables that you think might be useful in predicting the statistic that we are using to describe intergenerational mobility which is the variable `kfr_pooled_p25`.

2. Select and download at least 10 predictors in DataCommons for all counties in the United States. First, select a geography and choose predictors. Next, click "Get Code/Data". Then, click "Bulk Download data." Picking a particular year will generate a .csv file that contains the data for all counties. (Note that some data are only available in certain years, so you should pick a year where the variables you want to use are available).

3. Merge these data with the atlas_training.dta data file.

4. Many of the Google DataCommons variables are counts (e.g., total number of female residents of a county or owner-occupied housing units). Replace these counts with rates (e.g., percent female or fraction of owner-occupied housing units) by dividing by the population and housing variables given to you in atlas_training.dta. (Note that Google DataCommons is still under development; although you can draw graphs with per capita figures, only the counts can be downloaded via the Bulk Downloads).

5. Produce simple summary statistics for the 10 predictors you selected from DataCommons and `krf_pooled_p25` in the combined data set for observations that exist in both data sets.

6. Run a linear regression of `krf_pooled_p25` on the 10 predictors (converted to rates when appropriate) from Google DataCommons, inspect the results, and comment on what you find.

7. How well does your linear regression predict `krf_pooled_p25` in-sample? **Submit your answer to this question, the summary statistics table, and regression output by 11:59 p.m. on April 25 to receive 2 extra credit points.**

(continued on next page)

Part 2: Prediction Challenge

8. Run a linear regression of `krf_pooled_p25` on the full predictor set (consisting of the 10 predictors you chose from DataCommons and the 121 predictors included in the training data). Interpret one of the coefficients. Obtain predictions of `kfr_pooled_p25`.

9. Implement a decision tree on the full predictor set using 10 fold cross-validation to select the optimal tree size. What is the first split? Discuss why the first split is often an important predictor or correlate of the outcome.

   Note: If you are using Stata, we suggest that you save the data set at this point and switch over to R Studio Desktop or Python to implement the decision tree. More details and fully worked out examples are provided with the assignment data.

10. You could have created a larger tree that would have had lower prediction error in this training data. Why do we use cross-validation to select a smaller tree instead of just using as many splits as possible?

11. Implement a random forest with at least 1000 bootstrap samples and obtain predictions.

    Note: Similarly, here we suggest that you switch over to R Studio Desktop (or Python) to implement random forests using the `randomForest` package.

12. Calculate and compare the mean squared error for your results on 8, 9, 11 **in -sample**.

13. Briefly comment on whether or not you think your regression from question 8, question 9 or from question 11 will predict `krf_pooled_p25` better **out-of-sample.**

(continued on next page)

Part 3: Out-of-sample validation

14. Now turn to the test data set. Calculate the mean squared error for your results from 8, 9, and 11 out-of-sample.

15. Which model did the best? Write a one page summary of your analysis with a nicely formatted table showing the in-sample and out-of-sample mean squared error for your models estimated in questions 8, 9, and 11.

16. Draw some graphs or maps to visualize your predictions.

Extra Credit - 2 points

When beginning any prediction problem, researchers often have many different algorithms at their disposal, such as the ones we have considered in this assignment. It is often difficult to know beforehand which algorithm will perform best and it is common practice to try several. State of the art machine learning methods fit a library of algorithms and use a predetermined metric to choose an optimally weighted combination of algorithms.

For example, the Super Learner, a commonly used algorithm in biostatistics, is an ensembling machine learning approach that combines multiple algorithms into a single algorithm and returns a prediction function with the best cross-validated mean squared error.

For two points extra credit, use the p4_SL.R file (posted on the website) to implement to Super Learner algorithm and obtain predictions.

Add these new predictions to your answers to questions 12, 14, 15, and 16.

DATA DESCRIPTION, FILE: atlas_training.dta

The data consist of all 2,518 counties with at least 10,000 residents available from the Opportunity Atlas. For *n* = 1,259 counties in the "test" portion of the data, the outcome variable is set to missing. These observations are a 50% random sample of all counties with at least 10,000 residents available from the Opportunity Atlas. For more details on the construction of the variables included in this data set, please see Chetty, Raj, John Friedman, Nathaniel Hendren, Maggie R. Jones, and Sonya R. Porter. 2018. "The Opportunity Atlas: Mapping the Childhood Roots of Social Mobility." NBER Working Paper No. 25147.

Variable	**Definition**	**Obs.**
(1)	(2)	(3)
geoid	County FIPS code	2,518
pop	County Population from DataCommons	2,518
housing	Total number of housing units from Census	2,518
kfr_pooled_p25	Mean percentile rank in the national distribution of household income in 2014-2015 for children with parents at the 25th percentile of the national income distribution (missing for *n* = 1,259 counties in the test data, non-missing for the other *n* = 1,259 counties)	1,259
test	1 = Observation is in test data set (outcome variable is missing) 0 = Observation is in training data (outcome variable is non-missing)	2,518
training	1 = Observation is in training data set (outcome variable is non-missing) 0 = Observation is in the test data (outcome variable is missing)	2,518
P_1 through *P_121*	Predictors taken from the Opportunity Insights' county characteristics file and various other sources	2,518

Note: Full list of definitions of *P_1* through *P_121* is posted on the class website.

DATA DESCRIPTION, FILE: atlas_test.dta

Variable	**Definition**	**Obs.**
(1)	(2)	(3)
kfr_actual	Actual value for *kfr_pooled_p25* for all 2,518 counties with at least 10,000 residents	2,518
geoid	County FIPS code	2,518

Table 2a
Stata Commands

Commands	Description
**clear the workspace* *clear all* *set more off* *cap log close* **change working directory and open data set* *cd "C:\Users\gbruich\Ec1152\Projects\"*	This code shows how to clear the workspace, change the working directory, and open a Stata data file. To change directories on either a mac or windows PC, you can use the drop down menu in Stata. Go to file -> change working directory -> navigate to the folder where your data is located. The command to change directories will appear; it can then be copied and pasted into your .do file.
import delimited "export.csv", clear	This commands show how to import a .csv file into stata. You can also use the drop down menu.
rename county **	These commands show how to rename variables by removing the county prefix from any variable starting with county.
merge 1:1 geoid using atlas_training.dta, gen(mtrain)	These commands show how to merge the data in current working memory with the training data. The key that connects them is *geoid*. The option *gen(mtrain)* will generate a new variable *mtrain* that marks indicates which observations matched up across the two data sets. See this tutorial for more details.
replace xvar = xvar/pop	This command shows how to replace the variable *xvar* with a rate per person instead of a count.
replace xvar = xvar/housing	This command shows how to replace the variable *xvar* with a rate (fraction of housing units) instead of a count.
save project4.dta, replace	This command saves the data that is currently in the working memory. It will be saved to the working directory (which can be changed as shown above).
gen pred_error = kfr_actual - predictions_forest *gen mse_forest = pred_error^2* *sum mse_forest if test == 1*	This command shows how to report the mean squared prediction error for the test sample. First, we generate prediction errors and squared prediction errors. Then, we summarize this variable for observations in the test sample.

Table 2b
R Commands

Commands	Description

#clear the workspace
rm(list=ls())
 	
#Install and load haven package
install.packages("haven")
library(haven)
 	
#Change working directory and load stata data set
setwd("C:/Users/gbruich/Ec1152/Projects")
atlas <- read_dta("atlas_training.dta")
	This sequence of commands shows how to open Stata datasets in R. The first block of code clears the work space. The second block of code installs and loads the "haven" package. The third block of code changes the working directory to the location of the data and loads in atlas_training.dta.

gdc<- read.csv("export.csv")	This commands show how to import a .csv file into R
colnames(gdc) <- gsub('County.', '', colnames(gdc))	These commands show how to rename variables by removing the county prefix from any variable starting with county.
gdc <- merge(gdc,atlas,by = "geoid")	These commands show how to merge the gdc data frame with the training data. The key that connects them is *geoid*.
gdc$xvar <- gdc$xvar/gdc$pop	This command shows how to replace the variable *xvar* with a rate per person instead of a count.
gdc$xvar <- gdc$xvar/gdc$housing	This command shows how to replace the variable *xvar* with a rate (fraction of housing units) instead of a count.
gdc$pred_error = gdc$kfr_actual - gdc$predictions *gdc$mse_forest = gdc$pred_error^2* *mse_test <- subset(gdc, test==1, select = c(mse_forest,mse_trees,mse_ols))* *summary(mse_test)*	These command show one way to report the mean squared prediction error for the test sample. First, we generate prediction errors and squared prediction errors. Then, we summarize this variable for observations in the test sample.