Problem Set 5

Kaleigh Strohl
ECON833: Computational Methods

Fall 2021

The objective for this assignment was to implement a maximum score estimator for a one-to-one matching market representing radio station mergers. My code was not successful - specifically for the objective function, there is an issue with the sizes of my dataframes. Hopefully I'll figure it out eventually. In the meantime, I can still explain the model I was trying to recreate in Python.

The objective function I was trying to define was ideally going to compare the payoffs between the observed matches and counterfactual matches. The counterfactual dataset was created in Python, and I had also initially added a 'payoff' variable in these datasets. After thinking more about the objective function, I started to think I needed to do the payoff calculation within the MSE, but I kept both codes anyway.

When the defined objective function compares these two payoffs for matches ii, jj, ij, and ji, if the summed payoffs for the matches of ii and jj are greater than the summed payoffs for ij and ji, the objective function would indicate a "successful prediction" by adding 1 to the score. In order to increase the efficiency of the objective function, I was trying to loop the over the number of buyers, the buyer's matches, and years (since we have both 2007 and 2008 data).

Once maximizing this objective function using a method such as Nelder-Meade or Differential Evolution, Python should return parameter estimates for alpha and beta, along with the "score" for the MSE. This score should be a number of the successfully predicted matches (out of 99).

This previous objective function described does not use prices. I would need to define a second objective function to incorporate the new payoff function and transfers for the second part of the assignment. In this case, our "net payoff function" that we would compare between observations and counterfactuals would be the difference between the payoff function and prices corresponding to a buyer and their match.

After maximizing this second objective function using a method such as Nelder-Meade or Differential Evolution, Python should return parameter estimates for alpha, beta, delta, and gamma, along with the "score" for the [second] MSE. This score should be a number of the successfully predicted matches (out of 99).

In each objective function, one of our parameters needs to be normalized to be able to interpret the other parameters as a relative estimate. Because my code wasn't working, I wasn't sure how to work with this. Were I to obtain parameter estimates, the interpretation would be a one-unit change in the associated variable would correspond with an increase or decrease in the "weight" on that predictor variable relative to the normalized predictor variable. The "weights" on each of the variables are the optimal estimates to maximize the number of successfully predicted outcomes.