Problem Set 4

Jonathan Tregde
ECON815: Topics in Microeconomics

Fall 2019

Problem 1. Maximum Score Estimator

For this problem set, we used radio station merger data to estimate the parameters of the payoff function of the merger. We began by estimating the function,

$$f_m(b,t) = x_{1bm}y_{1tm} + \alpha x_{2bm}y_{1tm} + \beta distance_{btm} + \epsilon_{btm}$$

After estimating α and β for that equation, we then estimate a payoff function which includes target characteristics

$$f_m(b,t) = \delta x_{1bm}y_{1tm} + \alpha x_{2bm}y_{1tm} + \gamma HHI_{tm} + \beta distance_{btm} + \epsilon_{btm}$$

and we again use maximum score estimation to estimate δ, α, γ, and β for this function.

The MSE looks something like

$$\hat{\beta} = argmaxQ(\beta) = \sum_{y=1}^{Y}\sum_{b=1}^{M_y-1}\sum_{b'=b+1}^{M_y} \mathbb{1}[f(b,t \mid \beta) + f(b^{prime}, t^{prime} \mid \beta) \geq f(b^{prime}, t \mid \beta) + f(b, t^{prime} \mid \beta)]$$

Problem 2. Procedure

First, after reading in our data, we calculated the distances and put them in a new column for the data frame of the observed matches.

```
from geopy import distance as dist
rad['d'] = rad.apply(lambda row: dist.geodesic((row.buyer_lat, row.buyer_long)
                                        , (row.target_lat, row.target_long)),
                                        axis = 1)
```

Next, we split the data by market in order to allow proper calculation of the payoff function. After this, we created a matrix of counterfactuals to be used in the objective function for the maximum score estimator.

```
rad_CF = [cf[B].iloc[b].values.tolist() + cf[T].iloc[t].values.tolist() for cf
                                         in years for b in range(len(cf) - 1)
                                        for t in range(b + 1, len(cf))]
rad_CF = pd.DataFrame(rad_CF, columns = B + T)
```

At this point, we were then ready to create the distance column for the counterfactuals and then write a function to calculate the payoffs as follows and then run the optimization routine:

```
# Create the distance column
rad_CF['d'] = rad_CF.apply(lambda row: dist.geodesic((row.buyer_lat, row.
                                        buyer_long), (row.target_lat, row.
                                        target_long)), axis = 1)

def payoffs(data, i, params):
    '''
    A function to calculate the payoffs of the mergers.

    Args:
        data: the data to be used
        i: used for iterating through the rows
        params: a list of length (2) of initial parameter estimates
    Returns:
        f: the payoff
    '''
    alpha = params[0]
    beta = params[1]

    f = data['num_stations_buyer'].iloc[i] * data['logk_pop'].iloc[i] + alpha
                                        * data['corp_owner_buyer'].iloc[i]
                                        * data['logk_pop'].iloc[i] + beta *
                                         data['d'].iloc[i]


    return(f)
```

Next, we write the objective function to be maximized:

```
def objective(freal7, freal8, fCF7, fCF8, Params):
    '''
    A function to calculate the value of the objective function to be
                                        maximized

    Args:
```

```
        data:

    Returns:
        score: the score
    '''
    score = 0
    for i in [[freal7, fCF7], [freal8, fCF8]]:
        for j in range(len(i[0])):
            for k in range(len(i[0])):
                if i[0][j] + i[0][k] >= i[1][k: j] + i[1][k: (j - 1)]:
                    score = score - 1

    return(score)

results = opt.minimize(objective, Params, args = (freal7, freal8, fCF7, fCF8),
                                        method = 'Nelder-Mead', options = {'
                                       maxiter': 5000})
print("1st Part Results:", results)
```

The output from this should give us our estimates for α and β. However, it just returned my "guesses" for the values (i.e. whatever I put in Params, so clearly something was wrong. These coefficients could be interpreted as follows: a positive value for α would indicate that radio stations owned by corporation tend to match with stations in an area with higher population. A negative value for β would indicate that buyer stations tend to match with target stations that are closer to them (i.e. with a smaller distance between them).

To calculate the model that includes the transfer data, note much code had to be changed. The main change was in the objective function which looked something like this:

```
def objective2(freal72, freal82, fCF72, fCF82, Params):
    '''
    A function to calculate the value of the objective function to be
                                            maximized

    Args:
        data:

    Returns:
        score: the score
    '''
    score = 0
    for i in [[freal72, fCF72], [freal82, fCF82]]:
        for j in range(len(i[0])):
            for k in range(len(i[0])):
                if (i[0][j] - i[1][k: j] >= rad.logk_price[j] - rad_CF.
                                                        logk_price[k]) & (i[0][
                                                        k] - i[1][k: (j - 1)] >
                                                        = rad.logk_price[k] -
                                                        rad_CF.logk_price[j]):
                    score = score - 1

    return(score)
```

To extend the interpretations to the model with transfer data, α and β retain their same interpretation, but now we have δ and γ to consider. δ is the coefficient on the interaction of the number of stations owned by the parent company of the buyer and the population in range of the target. A positive value for δ would indicate that buyer stations with parent companies that more stations tend to match with targets with a higher population in range of the station. A positive value for γ would mean that buyer stations tend to match with targets with a higher HHI.