EPFL
ENAC TRANSP-OR
Prof. M. Bierlaire

Mathematical Modeling of Behavior
Fall 2017

NETHERLANDS MODE CHOICE

The objective of this lab is to become familiar with the usage of choice models in order to predict the aggregated market shares in some hypothetical situations, compute relevant indicators for policy analysis and forecast the prices that maximize the expected revenue. To this end, we will use the simulation instructions that are available in Biogeme.

With respect to the market shares, the idea is to predict them with a method called *sample enumeration.* In order to be able to use the Netherlands dataset for aggregation and prediction, we have to restrict it to contain only the revealed preference (RP) data (i.e., the actual choices). From now on, when we refer to the Netherlands dataset, we refer to its restricted version containing RP data only. This dataset is called `netherlandsRP.dat` and is provided.

Consider a choice model that has been estimated for the Netherlands case study (included in the file `Netherlands_Base_Model.py`). The deterministic terms of the utility function are the following:

$$\begin{aligned} V_{CAR} &= \text{ASC}_{\text{CAR}} + \beta_{\text{Cost\_Age1}} \cdot \text{Cost}_{\text{CAR}} \cdot \text{Age}_1 + \beta_{\text{Cost\_Age2}} \cdot \text{Cost}_{\text{CAR}} \cdot \text{Age}_2 \\ &\quad + \beta_{\text{Time\_CAR}} \cdot \text{TravelTime}_{\text{CAR}}, \\ V_{RAIL} &= \beta_{\text{Cost\_Age1}} \cdot \text{Cost}_{\text{RAIL}} \cdot \text{Age}_1 + \beta_{\text{Cost\_Age2}} \cdot \text{Cost}_{\text{RAIL}} \cdot \text{Age}_2 \\ &\quad + \beta_{\text{Time\_RAIL}} \cdot \text{TravelTime}_{\text{RAIL}} + \beta_{\text{Female}} \cdot \text{Female} + \beta_{\text{ArrivalTime}} \cdot \text{ArrivalTime}, \end{aligned}$$

where Cost_{CAR} and $\text{Cost}_{\text{RAIL}}$ are the cost of car and rail in euros, respectively, $\text{TravelTime}_{\text{CAR}}$ and $\text{TravelTime}_{\text{RAIL}}$ are the total travel time of car and rail in hours, respectively, Age_1 is 1 if the individual is 40 years or younger, Age_2 is 1 if the individual is 41 years or older (i.e., $\text{Age}_2 = 1 - \text{Age}_1$), Female is 1 if the reported gender is female, and ArrivalTime is 1 if the individual must arrive at a given time.

1 Aggregation

Assume that the procedure to collect the sample is *stratified random sampling.* Thus, it is necessary to associate a weight with each group or stratum. In our case, and given the available data and the relevant socioeconomic characteristics included in the choice model, we consider 4 strata:

1. Male and Age_1;
2. Male and Age_2;

3. Female and Age_1;
4. Female and Age_2.

In order to compute the weights associated with each group g, we need the total number of individuals in the population (N) and the number of individuals in the population within each group (N_g). Even if the population should only contain the individuals in the Netherlands commuting between Nijmegen and the Randstad in 1987 (scope of the case study), due to the unavailability of the data for this population, we consider the above mentioned strata for the whole country in 2011 for the sake of illustration.

	Age_1	Age_2
Male	4,092,390	4,151,092
Female	3,984,028	4,428,289

Table 1: Dutch Census 2011

Table 1 shows the size of the 4 groups in the population. Now, it is necessary to identify the size of these groups within the sample. To do so, we recommend you to use a statistical software like Excel or R. You should obtain the following results:

1. $S_1 = 89$,
2. $S_2 = 37$,
3. $S_3 = 65$, and
4. $S_4 = 37$.

The weight of group g is then calculated as follows:

$$\omega_g = \frac{N_g}{N} \cdot \frac{S}{S_g}, \tag{1}$$

where $N = \sum_{g=1}^{4} N_g$ and $S = \sum_{g=1}^{4} S_g$. You should obtain the following weights:

1. $\omega_1 = 0.63$,
2. $\omega_2 = 1.54$,
3. $\omega_3 = 0.84$, and
4. $\omega_4 = 1.64$.

Finally, we need to associate a weight with each individual. The general idea is to split the weight of each group among its individuals, i.e., we need to define ω_n such that $\sum_{n=1}^{S_g} \omega_n = \omega_g$. The individual weights are calculated as follows:

$$\omega_n = \frac{1}{S_g}\omega_g, \tag{2}$$

where g is the group individual n belongs to. Now, we just need to normalize the weights, i.e., $\sum_{n=1}^{S} \omega_n = 1$. We divide the individual weights by the sum of group weights, i.e.,

$$\overline{\omega}_n = \omega_n \cdot \frac{1}{\sum_{g=1}^{4} \omega_g}. \tag{3}$$

The column containing the weights for each individual needs to be included in the Netherlands dataset (for instance, it can be called `Weights`) so that Biogeme can use this information for aggregation and prediction. You have to store this dataset and use it from now on.

Once the weights have been calculated and included in the dataset, we might want to exclude later on some observations satisfying some criteria, and therefore the weights will not sum up to 1 anymore. In this case, we need to renormalize the weights so that their sum is equal to 1. In general, in order to account for the weights in Biogeme, you should follow these steps:

1. **Normalization**: as long as some observations are excluded, the sum of weights is no longer 1. To normalize it, the following instruction has to be included in the model file `Netherlands_Base_Model.py` (assume the column containing the weight is labeled as `Weights`):
   `BIOGEME_OBJECT.STATISTICS['Sum of weights'] = Sum(Weights,'obsIter').`
   In this way, the sum of weights will be shown on the `.html` file generated when running `Netherlands_Base_Model.py`, and will be used in the following step. In our case, the sum should be already one since we are not excluding any observation.

2. **Sample size**: Check the sample size on the file `Netherlands_Base_Model.html` that is generated. Note that it is not equivalent to the number of entries in the dataset if we exclude observations. In our case, the sample size is the same since we are not excluding any observation.

3. **Weight in Biogeme**: A new variable (`theWeight`) needs to be defined in order to normalize the existing weights (`Weights`). The normalized weights are calculated by multiplying the original weights (`Weights`) by the sample size (x) and dividing them by the total sum of weights (y), both obtained in the previous steps:
   ```
   theWeight = Weights * x/y
   BIOGEME_OBJECT.WEIGHT = theWeight
   ```
   In our case, since our weights sum up to 1, we need to define the weights for Biogeme in this way, where $x = 228$ and $y = 1$.

2 Simulation file

Up to this point, you should have added the column containing the weights to the Netherlands dataset. In order to construct the simulation file, follow the instructions provided in the document called *Simulation file*. You can call this file `Netherlands_Base_Simul.py`. We ask you to use the `simulate` variable to answer the following questions:

1. *Compute the predicted market shares for car and rail with stratified random sampling.*
   The predicted market shares are obtained with the following instructions:
   `prob_CAR = bioLogit(V,av,0)`
   `prob_RAIL = bioLogit(V,av,1)`
   `simulate = {'Prob.  CAR': prob_CAR, 'Prob.  RAIL': prob_RAIL}`
   The aggregated market shares can be found in the row "Weighted average" from the resulting `.html` file:

   - car: 63.65%, and
   - rail: 36.35%.

2. *Compare the predicted market shares with the actual choices. More precisely calculate the following shares:*

   - *share of users choosing car with a higher probability for rail, and*
   - *share of users choosing rail with a higher probability for car.*

   Try to find the possible causes.
   You can easily drag the `.html` into a spreadsheet (e.g., Excel) to calculate the mentioned shares. You should obtain:

   - share of users choosing car with a higher probability for rail: 9.21% ,and
   - share of users choosing rail with a higher probability for car: 14.91%.

3 Indicators

In this section, we ask you to compute the different indicators you have seen in the lecture.

1. *Value of time.* The value of time is calculated as

$$\text{VOT}_{in} = \frac{(\partial V_{in}/\partial t_{in})(c_{in}, t_{in})}{(\partial V_{in}/\partial c_{in})(c_{in}, t_{in})}, \tag{4}$$

   where c_{in} and t_{in} are the cost and travel time of alternative i and individual n, respectively. In Biogeme, the calculation of derivates is written as follows (example for car):
   `VOT_CAR = Derive(V_CAR,'TravelTimeCar')/Derive(V_CAR,'CostCarEuro')`
   Then, we just need to add this instruction in the `simulation` variable to be reported by Biogeme (example for car):
   `simulate = {'VOT CAR': VOT_CAR}`

   Remark: Notice that the `DefineVariable` operator is designed to preprocess the data file, and can be seen as a way to add another column in the data file, defining a new variable. However, the relationship between the new variable and the original one is lost. Therefore, Biogeme is not able to properly calculate the derivatives. For instance, if you have scaled one variable, your variable of interest is the original variable and not the scaled one. Thus, if your variable of interest is `Time`, and not `Time_Scaled`, its definition must be explicitly known to correctly calculate the derivatives. Consequently, all statements such as `Time_Scaled = DefineVariable('Time_Scaled', Time/100)` should be replaced by statements such as

```
Time_Scaled = Time/100
```

in order to maintain the analytical structure of the formula to be derived. In our case this step is not necessary because our variable of interest is the one we are defining.

(a) *Provide an estimate of the average value of time in the population in eur/h for each alternative.*
The estimate of the average value of time in the population is obtained from the weighted average of the sample. The aggregate values are found in the "Weighted average" row of the `.html` file. The estimates of the value of time are equal to:

- car: 16.72 eur/h, and
- rail: 3.76 eur/h.

(b) *Analyze the distribution of the value of time for each alternative in the sample by identifying the socioeconomic characteristic(s) that play(s) a role in the calculation of the value of time and report its value together with the 90% confidence interval.*
By looking at the deterministic utilities and the `.html` file, one can easily identify two groups of the population with different values of time for both rail and car: individuals within Age1 and individuals within Age2. To easily identify which individuals belong to each group, we can include in the `simulate` variable one of the two variables (`Age1` or `Age2`) to be reported. The average values, together with their confidence intervals, are provided in Table 1. Note that the lower values of the intervals for rail are negative. This is due to the method that Biogeme uses to calculate these values.
Remark: In order to obtain the 90% confidence interval we need to uncomment/include the following statement:

```
BIOGEME_OBJECT.VARCOVAR = vc
```

	Age_1	Age_2
Car	21.48 [13.68, 40.14]	14.51 [8.51,27.75]
Rail	4.83 [-1.70, 13.32]	3.26 [-0.97, 9.06]

Table 2: Average value of time and 90% confidence interval for car and rail

2. *Aggregate elasticities.* Since the aggregate point elasticities are obtained by aggregating the disaggregate elasticities, we need to calculate the normalization factors. To do so, we include the following statements in the simulation file and run the resulting file:

```
BIOGEME_OBJECT.STATISTICS['Normalization for elasticities CAR'] = Sum(theWeight
* prob_CAR ,'obsIter')
BIOGEME_OBJECT.STATISTICS['Normalization for elasticities RAIL'] = Sum(theWeight
* prob_RAIL ,'obsIter')
```

Then, a simulation file with the statements for the aggregate elasticities of interest has to be created. For instance, if we want to calculate the aggregate elasticity of the choice of car with respect to its travel time, the following instructions need to be included:

```
normalization_car = ...
elas_car_time = Derive(prob_CAR,'TravelTimeCar') * TravelTimeCar / prob_CAR
'Agg.  Elast.  PT - Time':  elas_car_time * prob_CAR / normalization_car
```

where the first statement corresponds to the normalization factor (calculated when running the simulation file with the additional statements for the normalization factors), the second statement calculates the disaggregate elasticity of the choice of car with respect to its travel time and the third statement is the entry to the simulation dictionary (`simulate` variable) that is designed to calculate the aggregate elasticities.

The normalization factors are the following:

- car: 145.116, and
- rail: 82.884.

Now we can compute all the aggregate elasticities by including all the corresponding instructions in the simulation file (i.e., the two normalization factors, all the elasticities expressions and we have to include them in the `simulate` variable). The aggregated values can be found in the columns "Weighted total":

(a) *Elasticity of the share of car with respect to travel time by car:* -0.97
(b) *Elasticity of the share of car with respect to the cost of car:* -0.25
(c) *Elasticity of the share of rail with respect to travel time by rail:* -0.45
(d) *Elasticity of the share of rail with respect to the cost of rail:* -0.84
(e) *Elasticity of the share of car with respect to travel time by rail:* 0.26
(f) *Elasticity of the share of car with respect to the cost of rail:* 0.48
(g) *Elasticity of the share of rail with respect to travel time by car:* 1.69
(h) *Elasticity of the share of rail with respect to the cost of car:* 0.43

4 Forecasting

Now we ask you to forecast the market shares for car and rail under a certain increase of the latter, as well as to decide on the price that maximizes the generated revenue from a given set of price levels.
Compute the predicted market shares for an increase of the cost of rail of 10%.
We need to define a new variable capturing the increase in the cost of rail (before the utility statements):

```
CostRailEuro_inc = DefineVariable('CostRailEuro_inc', CostRailEuro*1.10)
```

Then, we replace the original cost variable by the new variable wherever it appears in the utility statements, and run the file as usual. The obtained market shares are the following:

- car: 66.60%, and
- rail: 33.40%.

Note that this technique can also be used to test different price levels and to decide on the one maximizing the generated revenue. Indeed, we can increase/decrease the price of an alternative in the same way as described before and then include in the simulation the following statement to report the revenue generated by the alternative. For instance, for the car alternative with its original cost we will write:
`'Revenue CAR': prob_CAR * CostCarEuro`
Then, the total generated revenue is included in the row "Weighted total".

If we want to divide the population into some socioeconomic groups of interest, and then compute the revenue generated by each group, the quantities of interest are calculated as follows:

- Probability of individual n within group k to choose alternative i:

  $$P_k(i|p_i, p_j) = \frac{e^{\beta_{pk}p_i - 0.5}}{e^{\beta_{pk}p_i - 0.5} + e^{\beta_{pk}p_j}},$$

  where β_{pk} is the coefficient of the cost associated with group k;

- Market share of alternative i within group k:

  $$\frac{1}{N_k}\sum_{n=1}^{N_k} P_k(i|p_i, p_j) = P_k(i|p_i, p_j),$$

  as the choice model is the same for all individuals in a group;

- Total market share of alternative i:

  $$\frac{1}{\sum_{k=1}^{3} N_k}\sum_{k=1}^{3} N_k P_k(i|p_i, p_j);$$

- Revenue generated by group k:

  $$p_i N_k P_k(i|p_i, p_j);$$

- Total revenue:

  $$p_i \sum_{k=1}^{3} N_k P_k(i|p_i, p_j).$$