Tasks for “So1 Challenge”

Task 1: Efficient Computation

Description

The task at hand is to develop a code for an implementation of a mathematical calculation. The equation to be calculated is described in the next section. We look for a

- fast,
- efficient, and
- scalable

solution. You can use any language and libraries you think are suitable (See output for more details).

Equation

The goal is to calculate probabilities for consumers buying certain consumer packaged goods. The Logit model is a frequently employed model to achieve this task. In the Logit model the probability that consumer n purchases product j at time t is given by

$$P_{njt} = \frac{\exp[U_{njt}]}{\sum_k \exp[U_{nkt}]}$$

U_{njt} is the “utility” product j offers to consumer n at time t. The purchase probability for one product j depends on the utility of all products k (including j) that are available to consumer n at time t (this set of available products is usually referred to as the “choice set”).

The utility is given by

$$U_{njt} = \sum_k \alpha_{nk} \cdot Dummy_k + \beta_n \cdot Price_{jt}$$

$Dummy_k$ is a dummy variable that is 1 for product k=j, and 0 for all other products. $Price_{jt}$ is product j’s price at time t. α_{nk} and β_n are the consumer specific preferences for different variables (these are estimated by the Logit model). $\alpha_{nj} \cdot Dummy_j$ is the time invariant utility that product j offers to consumer n. This part of the utility function is often referred to as the products’ alternative specific constant (ASC). $\beta_n \cdot Price_{jt}$ is the time variant (dis-)utility (“dis” because beta usually is negative) induced by the products’ prices.

Data

For calculating the choice probabilities you receive two files: “Coef.csv” and “Data.csv”.

“Coef.csv” includes the coefficients for the utility function. For each consumer n the consumer specific preferences for the variables in the utility function α_n and β_n are provided. The first column “n” in the user index n. The other columns (“beta_ASC_[number of alternative]” and “beta_Price”) are the coefficients.

“Data.csv” includes the data for the utility function. The first three colums are indexes for user (“n”), time (“t”) and product (“j”). Note that t is not a date but a counter over the consumers shopping trips that were observed in the data. Column “j” is a unique product identifier. The next columns are the product specific dummy variables $Dummy_j$ and the products’ prices $Price_{jt}$. A good starting point might be to get familiar with the indexes of the data.

Please note: The system of time invariant utilities is normalized by setting one dummy variable (in this case the dummy for alternative 9) to 0 - thus you will not find columns “beta_ASC_9” in Coef.csv or column “ASC_9” in the Data.csv.

Output

Please provide both your (executable) code and a result file that includes all combinations of the three indices n, t, and j you find in data.csv as well as the corresponding choice probabilities (preferably “.csv”-format).

Interfacing your solution with R project (http://www.r-project.org/) would be of great advantage for us (e.g. Rcpp for C++, rPython for Python, or other interfaces). If you cannot interface it with R, please provide a AMD64 Linux binary alongside your source code.

Literature

Train, K. (2009): Discrete Choice Methods with Simulation, 2nd Edition, Cambridge University Press, http://elsa.berkeley.edu/books/choice2.html

Task 2: Algorithm Development

Description

The task at hand is to summarize in a short text or graphic how you would approach the development and evaluation of the following algorithm: organize consumer packaged goods (CPG), identified by the European Article Number (EAN), into categories, which contain products which substitute each other directly. Examples for CPG categories are the shampoo category, the milk category, the color detergent category, the frozen pizza category, etc. Across categories products are not substitutes.

As of today we use a category definition provided by the GS1 (http://www.gs1-germany.de/). Our goal is to create an algorithm that derives a classification of products by only using transactional data (see section data). This data is already available to us and used for other algorithms in our statistical engine.

We would provide two datasets:

1. Transactional data including categories for all CPGs
2. Transactional data without categories for CPGs

None of the products in dataset 2 is in dataset 1 and vice versa.

We look for an algorithm that classifies the products in dataset 2 into the categories found in dataset 1. Dataset 1 includes the data you should use to develop, train and evaluate your algorithm. Then you predict the categories for the products in dataset 2. We would just use your results for dataset 2 to evaluate your algorithm based on the true categories of the products in dataset 2. The criterion for evaluating the quality of your algorithm is the percentage of wrongly classified products.

Please describe in a short text or graphics how you would approach this task, by describing the major steps and algorithms you would employ. We are explicitly not expecting code as a result, but rather a text, flow chart, decision diagramm or any other method you find suitable for explaining us, how you would structure and solve the task.

Data

Table 1 contains an excerpt from the transactional data. Besides various IDs (user, time, store, EAN), the transactional data includes unit sales (i. e. number of purchased products) and value sales (i. e. paid amount). The transactional data therefore is a database containing all information that you will find on the receipts consumers receive at the checkout.

User	Time	Store	EAN	Unit Sales	Value Ses	Category
1	28.01.2013	234	4001724821601	1	2,69	Frozen Pizza
1	13.03.2013	234	4009233012565	1	2,49	Frozen Pizza
1	23.03.2013	234	4009233012565	2	4,98	Frozen Pizza
2	21.12.2012	532	0062712000068	1	2,79	Frozen Pizza
...	...		...	...	...	...

Table 1: Transactional data (Category information only available in dataset 1!)