Private Label Demand: Data Preparation

Günter J. Hitsch

January 19, 2017

```
library(bit64)
library(data.table)
```

Overview

Private label products are an important component of retail strategy and a concern for national-brand manufacturers. A key question that is of importance to brand and retail managers is whether private label demand is sensitive to household income and wealth. We will study this question using data from the Nielsen Homescan panel, and we will also use publicly available local home value data from Zillow as a measure of wealth. An important feature of the data is that it includes the *Great Recession* in the U.S., which is the only recession associated with substantial income, wealth, and employment changes since high-quality household and consumer data became available in the 1980s.

The overall research strategy is as follows:

1. Construct household-level private label shares for all months between 2004-2014
2. Use household-level income, employment, and other information contained in the Nielsen data
3. Use local, 5-digit ZIP code Zillow home value data as a measure of local housing wealth
4. Summarize the data
5. Provide regression results of private label shares on income, employment, wealth (Zillow home value index), and other controls

Using local Zillow home values as a measure of wealth seems less than perfect. However:

- Housing wealth is a large component of overall household-level wealth.
- The ZIP code average home value is not the same as the exact home value of a household. However, *changes* in home values are locally correlated, and our strategy will ultimately rely on *differences*, not *levels* of housing wealth.
- During the Great Recession, there were large differences in the changes in housing values across regions. This is an import source of variation. Financial wealth changes, on the other hand, are largely uniform across the whole U.S.

In the first part of the assignment you will construct the private label share data and save them to file. In the second part you will finish the data preparation and then run the statistical analysis.

Data

We will use the Nielsen Homescan household panel data for the analysis. You are already familiar with many aspects of the data from the first assignment. The three main data sources are:

1. The purchase data contain information on product purchases at the `household_code`/`purchase_date`/product level. Recall that products are identified by a unique `upc`/`upc_ver_uc` combination. The overall data size is large: The largest sample that I made available represents 25 percent of all households in the original data and contains more than 156 million observations. To reduce the file size and RAM requirement, the data are the year-level, such as *purchases_2011.RData*. The purchase data span the years 2004-2014.
2. The product data, *Products.RData*, contains information (size, brand name, etc.) for each product.
3. The *panelists.RData* file contains the `panelists` table, which contains yearly household information. The data are at the `household_code`/`panel_year` level. The `panel_year` indicates the year when the household information was made available.

Household-level private label share creation

We will analyze the evolution of private label shares. Calculating the household-level private label shares is time-consuming (although even on a slow computer it will not take more than about 10 minutes, thanks to the incredible speed of data.table), and hence it's a good idea to pre-compute these shares and save them to file before performing the main analysis.

Tagging private label products

Load the product data and add a new variable, a *flag* that we call `is_PL`. `is_PL` should equal `TRUE` if a product is a private label product. In the Nielsen data, private label products include the string `"CTL BR"` in the `brand_descr` variable, such as `CTL BR` (the vast majority of cases) and `CTL BR AMERICAN INDIA PALE ALE` (extremely rare). `CTL BR` stands for *control brand*. Consult the data.table notes (*Additional Topics*) to learn about string pattern matching and selection.

Take a look at all the matches for `CTL BR` (use `table`).

Iteration and data reduction strategy

The purchase data are at the year-level. To process all of the data, we need to *iterate* (loop) over all the files and finally combine the calculations in one table.

Such a task—*iteration and data reduction*—is extremely frequent in data science. Let's discuss a *general strategy* to solve this or a similar task.

To illustrate, we create a very simple data.table with 5 groups (identified by `id`) and 4 observations of `x` in each group.

```
set.seed(963)
DT = data.table(id = rep(1:5, each = 4),
                x  = round(rnorm(20, mean = 10, sd = 3)))
setkey(DT, id)
```

Now we perform the iteration task. We first create an empty list, named `container`, and initialize an `index` variable, which serves as a counter. Then we iterate/loop over all unique `id` values and create a data.table that contains the results of the calculations performed on the data corresponding to a specific `id` value (here we calculate the mean of `x` for the corresponding data). At the end of each iteration, we store the output data (`id_DT`) in the list (`container`) and increment the counter.

```
container = list()
index = 1

for (id_value in unique(DT$id)) {
   id_DT = DT[id == id_value, .(id     = id_value,
                                mean_x = mean(x))]
   container[[index]] = id_DT
   index = index + 1
}
```

Please be sure to use a double bracket, `[[ ]]`, to index the individual elements in the `container`. It's one of those R quirks (for some good bedtime reading, see https://www.r-bloggers.com/selection-in-r/).

Then we combine the id-level output values using the `rbindlist` (row bind list) function:

```
reduced_DT = rbindlist(container)
reduced_DT
```

```
   id mean_x
1:  1  11.50
2:  2  13.00
3:  3  10.75
4:  4  10.25
5:  5   9.50
```

Alternatively, we could iterate directly over all five index values:

```
container = list()
id_values = unique(DT$id)

for (index in 1:5) {
   id_value = id_values[index]
   id_DT = DT[id == id_value, .(id     = id_value,
                                mean_x = mean(x))]
   container[[index]] = id_DT
}

reduced_DT_a = rbindlist(container)
identical(reduced_DT, reduced_DT_a)
```

```
[1] TRUE
```

Of course, in this simple example we could have calculated the summary output much more efficiently using the `by` syntax in data.table.

```
reduced_DT_b = DT[, .(mean_x = mean(x)), by = id]
setkey(reduced_DT, id)
identical(reduced_DT, reduced_DT_b)
```

```
[1] TRUE
```

However, to calculate the private label shares we need to iterate over all year-level data files, and hence we cannot directly use data.table and `by`.

Household-level private label share computation

Goal: For each household, calculate private label shares at the year/month level. The private label share is defined as the percentage of total monthly dollar spending on private label products.

Strategy: Iterate over all purchase files. A straightforward way of doing this is to recognize the commonality in the files names, `puchases_<year>.RData`. Iterate over all year values (`yr in 2004:2014`), and load the corresponding file names that we can construct using the `paste0` function:

```
load(paste0(data_folder, "/purchases_", yr, ".RData"))
```

Within each iteration we need to perform the following tasks:

1. Load the purchase data
2. Create a year and month variable
3. Merge the `is_PL` flag (and department code—see discussion below)
4. Create total dollar spending at the household/year/month level, separately for private label and other (national brand) products
5. Convert to percentage shares
6. Keep data on household/year/month private label shares only
7. Insert the year-level data into a list

And finally combine all data into one single data.table.

Before you save the data, remove any missing values. Missing values can occur in extremely rare instances where total household spending at the year/month level is 0. The private label share is then not defined (check the result of `0/0` and `is.na(0/0)` in R). Recall how to remove missing values from a data.table:

```
shares_DT = shares_DT[complete.cases(shares_DT)]
```

Key the data (household/year/month) and save to file.

One more detail: Take a look at the the 12 department codes and descriptions that you can find in the products data. I recommend to exclude products in the General Merchandise department, which includes all kind of categories where the comparison of national brand versus private label products is much less clear than in other categories (breakfast cereal, household cleaners, over-the-counter drugs, ...). Also exclude the Magnet Data, which includes products with non-standard UPC's, including vegetables, fruits, and in-store-baked products (see the official Homescan documentation), and exclude products with an unknown department code.

When performing somewhat lengthy iterations, you may want to include a timer in the loop:

```
start_time   = proc.time()                  # Initial time
elapsed_time = proc.time() - start_time     # Elapsed time since start time was taken
elapsed_time[3]                             # To print the elapsed seconds
```