Finding Representative Basket for Chhattisgarh (Economic Statistics Mid Semester Project)

Srijan Chattopadhyay (BS2126)

11/10/2023

Introduction:

In the realm of economic statistics and policy formulation, the Consumer Survey serves as an invaluable tool to comprehend and assess the economic dynamics within a nation. The 68th round of the Consumer Survey in India, conducted with meticulous precision and expansive reach, presents a wealth of data that is crucial for understanding the consumption patterns, preferences, and economic well-being of Indian households. This comprehensive survey, undertaken by the Central Statistics Office (CSO) under the aegis of the Ministry of Statistics and Programme Implementation, serves as a cornerstone for empirical analysis and policy-making in the Indian economic landscape.

Goal:

In this project, we will analyze the consumer survey 68th round data for the state of Chhattisgarh and will provide a representative basket for the state with appropriate justification. A representative Basket means that the items in the Basket are consumed by most people with a greater chance than other items. As naturally rural and urban preferences differ, we will analyze those separately and will analyze the differences in those baskets also. We will analyze the results first district-wise and combine them to get for the whole state.

Data Description:

- For each of the households, for a particular item consumed by that household, there is a separate row.
- The columns "Sector", "Item Code", "Total Consumption Value", "HHID", "District Code", and "Combined Multiplier" are necessary for our project.
- In the sector column, 1 represents rural household, and 2 represents urban household.
- In the "District" column, the following is the code with the name of the district:

State Code	District Code	District Name
22	01	Koriya
22	02	Surguja
22	03	Jashpur
22	04	Raigarh
22	05	Korba
22	06	Janjgir-Champa
22	07	Bilaspur
22	08	Kawardha
22	09	Rajnandgaon
22	10	Durg
22	11	Raipur
22	12	Mahasamund
22	13	Dhamtari
22	14	Kanker
22	15	Bastar
22	16	Dantewada
22	17	Narayanpur
22	18	Bijapur

- Each entry of the "HHID" column represents a unique identifier given to each of the households.
- In the "Item Code" column, there are different codes for different items mentioned in the official documentation of the survey. We will not mention all of them here, but will mention the final basket along with their codes.

Some Theoretical Terms:

- **Combined Multiplier:** Combined multiplier is a characteristic of the household. It signifies on average, how many households in the whole area (where survey is done) are of the similar types of that household. This doesn't vary from item to item. ith household has combined multiplier M_i implies, on average there are M_i number of households in the whole survey region, i.e. the impact of the household on the whole survey is proportional to M_i.
- **Total Consumption Value and Expenditure Share:** Total Consumption value of an item j for ith household C_{ij} is the amount of money i used in the past few days for the jth item. Expenditure share is defined also for each item in each of the households. If R_{ij} is the expenditure share for jth item, in the ith household, then

$$R_{ij} = \frac{C_{ij}}{\sum_j C_{ij}} \tag{1}$$

i.e., out of the whole expenditure, how much does the ith household spend for the jth item. Note that,

$$\sum_j R_{ij} = 1 \ \ \forall i \tag{2}$$

- **Weight of an item:** This is a characteristic of items, which is based on the survey. This can be calculated for any collection of households, not necessarily for Let, $H = \{H_1, .., H_n\}$ be the set of households, and M_i be the combined multiplier of ith household for the set H. R_{ij} be the expenditure share for jth item in the ith household. Then, weight is the group expenditure share of all the households in the set H. So, if the weight of an item j is w_j, then

$$w_j = \frac{\sum_i M_i R_{ij}}{\sum_j \sum_i M_i R_{ij}} = \frac{\sum_i M_i R_{ij}}{\sum_i M_i} \tag{3}$$

note that,

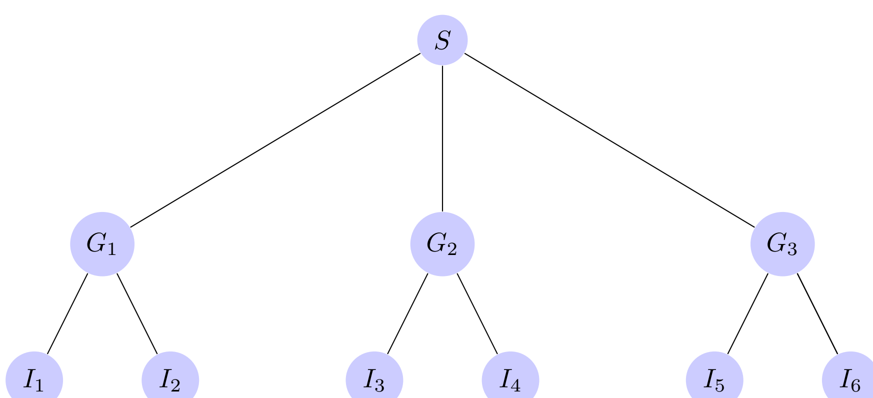
$$\sum_j w_j = \sum_j \frac{\sum_i M_i R_{ij}}{\sum_i M_i} = \frac{\sum_i M_i}{\sum_i M_i} = 1 \tag{4}$$

And, in that case, weight will only be calculated based on the union of items consumed by those households only. If we want weight of an item within a restricted set of items, then the formula is also similar just we have to take the sum over those sets of items only.

The Algorithm:

1. **Step 1:** Say, in the state S, there are n districts, call those $D_1, ..., D_n$. For each of the districts D_i, call the set of items as a group consumed by households of that district to be S_i, like a group of items may be Cereals, Medicine, Clothing, etc. For each of the groups in S_i, find the weight of each of the groups in each district.
2. **Step 2:** So, now for each district D_i, group w_j will be given some rank r_{ij} based on the weights for each of the district.
3. **Step 3:** So, then take the union of groups, i.e., $sort(S_1, ..., S_n)$. Then, for each of the states, if some group isn't consumed by that state, then assign 0 to it, else assign r_{ij} to it.
4. **Step 4:** So, now for each of the items, we have ranks assigned by each of the districts. So, each of the districts gives some preferential ordering to the group of items. Then, we combine them by taking $\beta * M + (1 - \beta) * m$, where M is the median of the ranks and m is the mean of the ranks. M gives a robust measure, and m gives a non robust measure. We want to account for both, because, there must be some amount of non-robustness to be captured as it is a district-wise setup. Also, to avoid creating more bias, robustness is also required. So, we took a convex combination of both. We used $\beta = 0.6$. Thus, we combined all the ranks to get a combined total rank. Based on that, we only take the upper sample quantile, greater than $(1 - \alpha)\%$, and can take $\alpha = 0.1$ for simplicity, i.e. only take those items which are consumed by more than 90% of the households.
5. **Step 5:** So, after these 4 steps, we are left with only the most preferential group of items. Call those $I_1, .., I_p$.
6. **Step 6:** Then, create p many data frames just subset with the group of items. Say, if the I_i contains $\{a_1, ...a_{k_i}\}$, then, subset with those items only. Apply the above 4 processes to all of the data frames. So, in the end, we will be left with a bunch of items for all of the groups.
7. **Step 7:** Combine them to get a total basket.

Say, S is the whole set of items, and, after running up to step 4 on the group of items, we are left with the groups $G_1, G_2, \& G_3$. And, in the groups, the most preferred items are $\{I_1, I_2\}, \{I_3, I_4\}, \{I_5, I_6\}$ So, the whole hierarchical structure for this example will be:



Implementation:

First, import some important libraries:

Importing Libraries

```
library(haven)
library(dplyr)
```

Data Processing

Now, I have already subset the whole data by only the state of Chhattisgarh (State code = "22") and exported it to the local machine. The following blocks of codes import those datasets and keep only the necessary columns for this analysis and remove the rest:

```
data1 = read_dta(file.choose())
data2 = read_dta(file.choose())
data3 = read_dta(file.choose())
data4 = read_dta(file.choose())
data5 = read_dta(file.choose())
data6 = read_dta(file.choose())

data1_sr1 = na.omit(data1[, c(6, 19, 23, 28, 30, 31)])
data2_sr1 = na.omit(data2[, c(6, 19, 21, 25, 27, 28)])
data3_sr1 = na.omit(data3[, c(6, 20, 28, 32, 34, 35)])
data4_sr1 = na.omit(data4[, c(6, 19, 20, 24, 26, 27)])
data5_sr1 = na.omit(data5[, c(6, 19, 20, 24, 26, 27)])
summary = na.omit(data6[, c(6, 19, 20, 24, 26, 27)])


colname = colnames(data1_sr1)
colname[3] = "TCV"
colnames(data1_sr1) = colnames(data2_sr1) = colnames(data3_sr1) = colnames(data4_sr1) =
colnames(data5_sr1) =
colnames(summary) = colname

head(data1_sr1)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      102         975 764861101 2201                         442.
## 2 1      103          12 764861101 2201                         442.
## 3 1      108          32 764861101 2201                         442.
## 4 1      110           7 764861101 2201                         442.
## 5 1      129        1026 764861101 2201                         442.
## 6 1      139          15 764861101 2201                         442.
```

```
head(data2_sr1)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      351         760 764861101 2201                         442.
## 2 1      352         252 764861101 2201                         442.
```

```
## 3 1      353         192 764861101 2201                         442.
## 4 1      356         480 764861101 2201                         442.
## 5 1      357         342 764861101 2201                         442.
## 6 1      363         280 764861101 2201                         442.
```

```r
head(data3_sr1)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      552         240 764861101 2201                         442.
## 2 1      559         240 764861101 2201                         442.
## 3 1      564          20 764861101 2201                         442.
## 4 1      569          20 764861101 2201                         442.
## 5 1      571         180 764861101 2201                         442.
## 6 1      579         180 764861101 2201                         442.
```

```r
head(data4_sr1)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      400         195 764861101 2201                         442.
## 2 1      404         295 764861101 2201                         442.
## 3 1      405          45 764861101 2201                         442.
## 4 1      409         535 764861101 2201                         442.
## 5 1      400         395 764861102 2201                         442.
## 6 1      401          25 764861102 2201                         442.
```

```r
head(data5_sr1)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      440         280 764861101 2201                         442.
## 2 1      441          30 764861101 2201                         442.
## 3 1      449         310 764861101 2201                         442.
## 4 1      450          38 764861101 2201                         442.
## 5 1      451          18 764861101 2201                         442.
## 6 1      452          28 764861101 2201                         442.
```

In the summary data, some items are given in subtotal format, but the subitems are already there. So, to avoid overcounting, we removed those subtotals. The following block of code does that and partitions the data into urban and rural.

```r
rep = c(6, 19, 20, 31, 38, 39, 40, 41, 42, 43)

cond <- !as.numeric(summary$Item_Code) %in% rep

sum_final = summary[cond,]

sum_rur = sum_final[which(as.numeric(sum_final$Sector) == 1),-1]
sum_urb = sum_final[which(as.numeric(sum_final$Sector) == 2),-1]


sum_rur = sum_rur %>% arrange(HHID)
```

```r
sum_urb = sum_urb %>% arrange(HHID)
```

Main Function 1

The following is the weight function. Whenever given a data frame in the same format as mentioned previously, the "weight" function will automatically give the weight of each of the items mentioned in that data frame. This function will return a data frame with the first column as items and the second column to be the corresponding weight of that.

```r
weight = function(final) {
  household = sort(as.numeric(unique(final$HHID)))
  item_list = sort(as.numeric(unique(final$Item_Code)))
  combmult = numeric(length(household))
  k = 0

  for (i in household) {
    k = k + 1
    combmult[k] = final$Combined_Multiplier[which(as.numeric(final$HHID) == i)[1]]
  }

  R = matrix(0,
             nrow = length(household),
             ncol = length(item_list))
  row = 0

  for (i in household) {
    col = 0
    row = row + 1
    h = final[which(as.numeric(final$HHID) == i),]
    for (j in item_list) {
      col = col + 1
      if (j %in% as.numeric(h$Item_Code) == TRUE) {
        R[row, col] = h$TCV[which(as.numeric(h$Item_Code) == j)] /
          sum(h$TCV)
      }
    }
  }

  weight = numeric(length(item_list))
  col = 0
  for (j in item_list) {
    col = col + 1
    weight[col] = sum(R[, col] * combmult) / sum(combmult)
  }
  return(cbind(item_list, weight))
}
```

Main Function 2

The following is the 2nd important function named "basket". It takes two inputs, one is a data frame and another is a probabilistic threshold value. There are multiple districts in the data frame. For each of the districts, it will find the weight of the items for that district only in some mentioned groups. Then, it will perform step 4 to step 6 of the algorithm. At the end, it will return a set of items, i.e. a vector.

```
basket = function(final, alpha){
  item_list = sort(as.numeric(unique(final$Item_Code)))
  district = split(final, final$District_Code)
  weight1 = vector("list", length(district))

  for (i in 1:length(district)) {
    weight1[[i]] = weight(district[[i]])
  }

  weight_rank = vector("list", length(weight1))

  for (i in 1:length(weight1)) {
    x = as.data.frame(weight1[[i]])
    weight_rank[[i]] = x[order(x$weight, decreasing = TRUE), ]
    y = as.data.frame(weight_rank[[i]])
    weight_rank[[i]] = cbind(weight_rank[[i]], rank(y$weight))
  }

  dist_rank = matrix(0,
                     nrow = length(item_list),
                     ncol = length(weight1) + 1)
  dist_rank[, 1] = item_list

  for (i in 1:nrow(dist_rank)) {
    for (j in 1:18) {
      h = as.data.frame(weight_rank[[j]])
      if (dist_rank[i, 1] %in% h$item_list == FALSE)
        dist_rank[i, j + 1] = 0
      else{
        index = which(h$item_list == dist_rank[i, 1])
        dist_rank[i, j + 1] = h[index, 3]
      }
    }
  }

  m1 = 0.4*rowSums(dist_rank[, c(2:19)]) / 18 + 0.6*apply(dist_rank[,c(2:19)],1,median)
  dist_rank = cbind(dist_rank, m1)
  dist_rank = as.data.frame(dist_rank)
  dist_rank = dist_rank[order(dist_rank$m1, decreasing = TRUE), ]
  s1 = quantile(dist_rank$m1, alpha)
  basket = dist_rank$V1[which(dist_rank$m1 > s1)]

  return(basket)

}
```

Now, in the "sum rur", we have the group of items for rural and the same for urban in "sum urb". Applying the basket function at 0.8 level, we get the most preferential set of items for both rural and urban.

```
group_rur = basket(sum_rur, 0.80)
group_urb = basket(sum_urb, 0.80)
```

The following is the result for that:

Rural	Urban
Cereals	Cereals
Pulses and Products	Fuel and light
Fuel and light	Clothing
Clothing	Footwear
Footwear	Education
Education	Medical (institutional)
Durable goods	Durable goods

Finding baskets groupwise

Now, we want to analyze the whole data together, i.e. gathering all the consumed items in a single data frame. The following code does that.

```
final_0 = bind_rows(data1_sr1, data2_sr1, data3_sr1, data4_sr1, data5_sr1)
```

But, there are repetitions of items in the actual data. For example, item number 129 represents "cereal: sub-total (101-122)", so it is already accounted for in item set 101-122. So, this kind of extra count should be removed to get the actual scenario.

```
repetition = c(
  129,159,169,179,189,199,219,239,249,269,279,
  289,299,309,319,329,349,379,389,399,409,419,
  429,439,449,459,479,499,519,529,549,559,569,
  579,599,609,619,629,639,649,659
)
condition <- !final_0$Item_Code %in% repetition
final = final_0[condition,]
head(final)
```

```
## # A tibble: 6 x 6
##   Sector Item_Code   TCV HHID      District_Code Combined_Multiplier
##   <chr>  <chr>     <dbl> <chr>     <chr>                       <dbl>
## 1 1      102         975 764861101 2201                         442.
## 2 1      103          12 764861101 2201                         442.
## 3 1      108          32 764861101 2201                         442.
## 4 1      110           7 764861101 2201                         442.
## 5 1      139          15 764861101 2201                         442.
## 6 1      140          45 764861101 2201                         442.
```

We will be analyzing the data for rural and urban sectors separately. In the sector column, "1" implies Rural and "2" implies Urban. Then we are sorting the data wrt household id, such that the household data is arranged in a proper order and it becomes easy to handle the data.

```
final_rur = final[which(as.numeric(final$Sector) == 1),-1]
final_urb = final[which(as.numeric(final$Sector) == 2),-1]
final_rur <- final_rur %>% arrange(HHID)
final_urb <- final_urb %>% arrange(HHID)
rur1 = urb1 = c(101:122)
rur2 = c(140:152)
rur3 = urb2 = c(330:345)
rur4 = urb3 = c(350:376)
rur5 = urb4 = c(390:395)
rur6 = urb5 = c(400:408)
```

```r
urb6 = c(410:414)
rur7 = urb7 = c(550:643)
cu1 <- as.numeric(final_rur$Item_Code) %in% rur1
dfrur1  = final_rur[cu1, ]
cu2 <- as.numeric(final_rur$Item_Code) %in% rur2
dfrur2  = final_rur[cu2, ]
cu3 <- as.numeric(final_rur$Item_Code) %in% rur3
dfrur3  = final_rur[cu3, ]
cu4 <- as.numeric(final_rur$Item_Code) %in% rur4
dfrur4  = final_rur[cu4, ]
cu5 <- as.numeric(final_rur$Item_Code) %in% rur5
dfrur5  = final_rur[cu5, ]
cu6 <- as.numeric(final_rur$Item_Code) %in% rur6
dfrur6  = final_rur[cu6, ]
cu7 <- as.numeric(final_rur$Item_Code) %in% rur7
dfrur7  = final_rur[cu7, ]
cu1 <- as.numeric(final_urb$Item_Code) %in% urb1
dfurb1  = final_urb[cu1, ]
cu2 <- as.numeric(final_urb$Item_Code) %in% urb2
dfurb2  = final_urb[cu2, ]
cu3 <- as.numeric(final_urb$Item_Code) %in% urb3
dfurb3  = final_urb[cu3, ]
cu4 <- as.numeric(final_urb$Item_Code) %in% urb4
dfurb4  = final_urb[cu4, ]
cu5 <- as.numeric(final_urb$Item_Code) %in% urb5
dfurb5  = final_urb[cu5, ]
cu6 <- as.numeric(final_urb$Item_Code) %in% urb6
dfurb6  = final_urb[cu6, ]
cu7 <- as.numeric(final_urb$Item_Code) %in% urb7
dfurb7  = final_urb[cu7, ]
basket_rur1 = basket(dfrur1,0.90)
basket_rur1
```

```
## [1] 102 101
```

```r
basket_rur2 = basket(dfrur2,0.90)
basket_rur2
```

```
## [1] 140 147
```

```r
basket_rur3 = basket(dfrur3,0.90)
basket_rur3
```

```
## [1] 331 332
```

```r
basket_rur4 = basket(dfrur4,0.90)
basket_rur4
```

```
## [1] 351 370 363
```

```r
basket_rur5 = basket(dfrur5,0.90)
basket_rur5
```

```
## [1] 393
```

```r
basket_rur6 = basket(dfrur6,0.90)
basket_rur6
```

```
## [1] 404
```

```r
basket_rur7 = basket(dfrur7,0.90)
basket_rur7
```

```
## [1] 632 600 603 570 601
```

```r
basket_urb1 = basket(dfurb1,0.90)
basket_urb1
```

```
## [1] 102 108
```

```r
basket_urb2 = basket(dfurb2,0.90)
basket_urb2
```

```
## [1] 332 331
```

```r
basket_urb3 = basket(dfurb3,0.90)
basket_urb3
```

```
## [1] 351 364 363
```

```r
basket_urb4 = basket(dfurb4,0.90)
basket_urb4
```

```
## [1] 393
```

```r
basket_urb5 = basket(dfurb5,0.90)
basket_urb5
```

```
## [1] 404
```

```r
basket_urb6 = basket(dfurb6,0.90)
basket_urb6
```

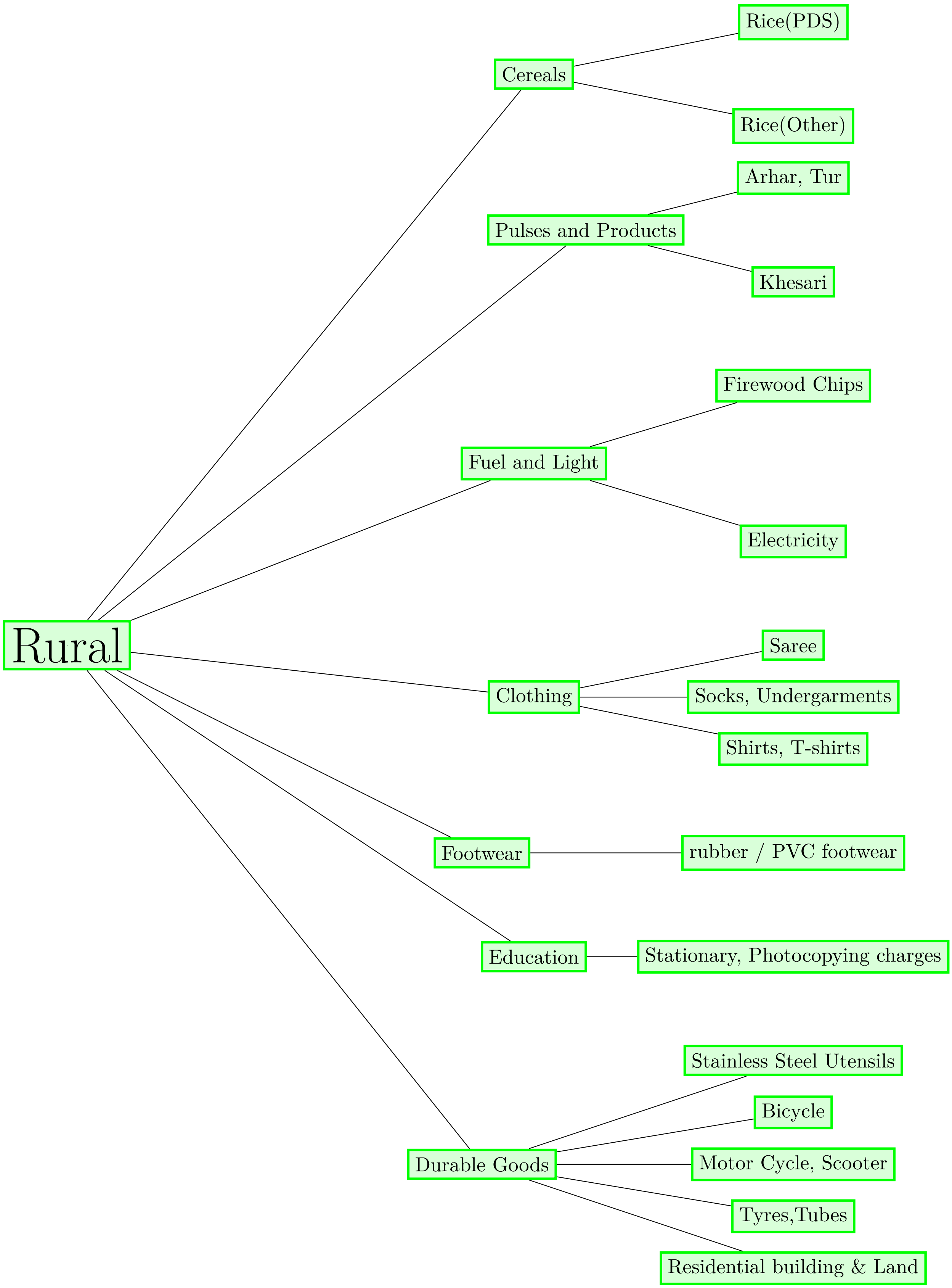
```
## [1] 410
```

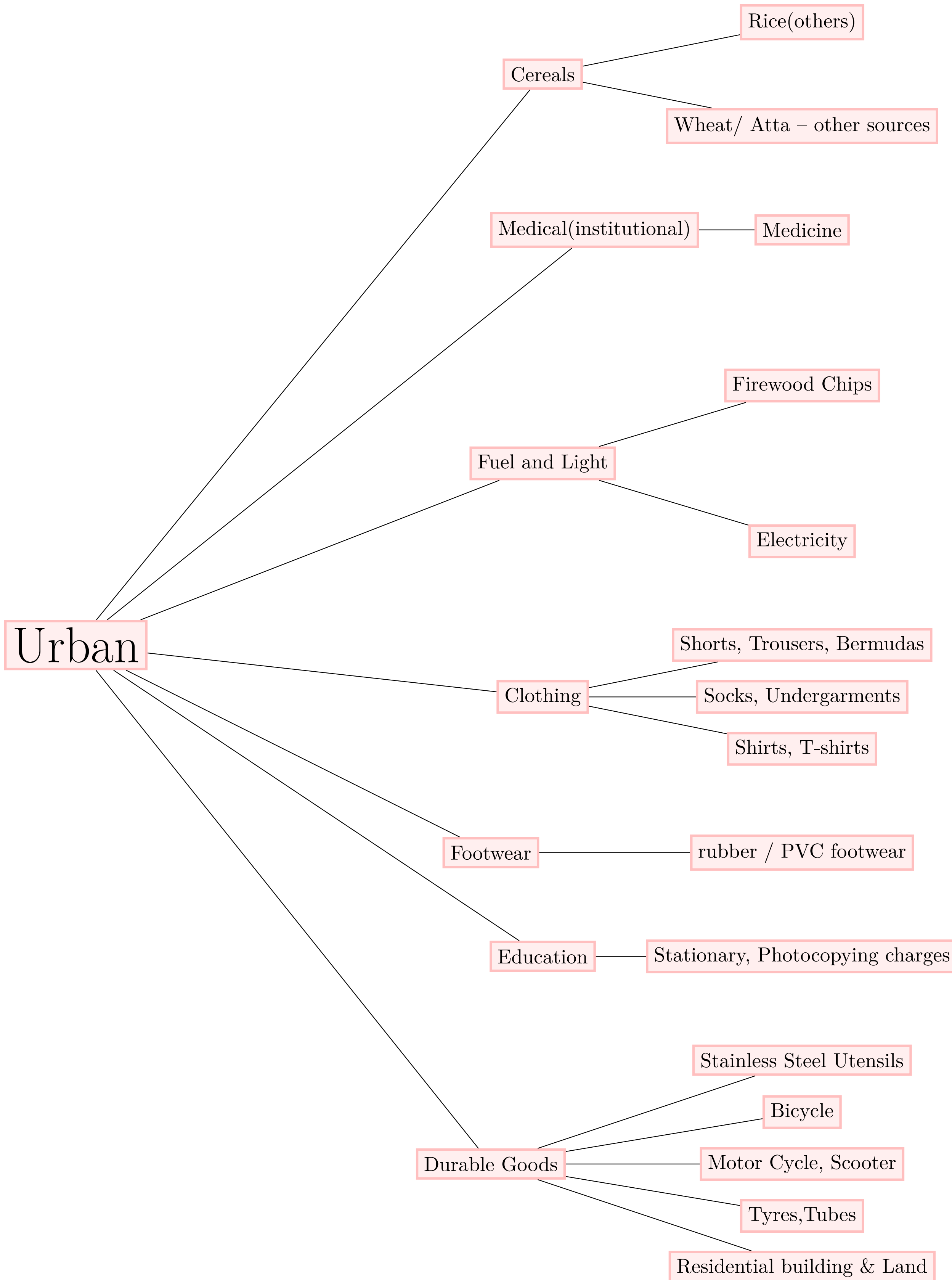
```r
basket_urb7 = basket(dfurb7,0.90)
basket_urb7
```

```
## [1] 632 600 601 570 603
```

Results and Discussion:

Below is the representative basket (GroupWise in the graph, and, as a whole in the table) for both rural and urban.


Rural
Cereals
Rice(PDS)
Rice(Other)
Pulses and Products
Arhar, Tur
Khesari
Fuel and Light
Firewood Chips
Electricity
Clothing
Saree
Socks, Undergarments
Shirts, T-shirts
Footwear
rubber / PVC footwear
Education
Stationary, Photocopying charges
Durable Goods
Stainless Steel Utensils
Bicycle
Motor Cycle, Scooter
Tyres,Tubes
Residential building & Land



Now, let's look at some interesting items. In the rural sector, most of the people used the Public Distribution

System or Ration for Rice, etc. That's why in the rural sector, that's an important item, whereas in the Urban sector, it is not. Let's look at the distribution of TCV for Rice-PDS at both rural and urban:

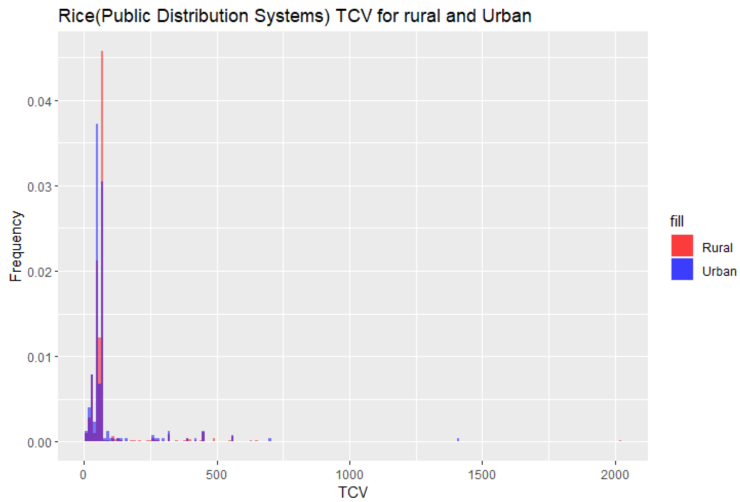


Figure 1: Rice(PDS) consumption rural vs urban

From fig. 1 We can clearly see that the mode of rural occurs at a higher value with a higher probability than urban. As, in most of the cases, relatively rich people live in urban places, most of them tend to buy rice from other sources rather than PDS or ration.

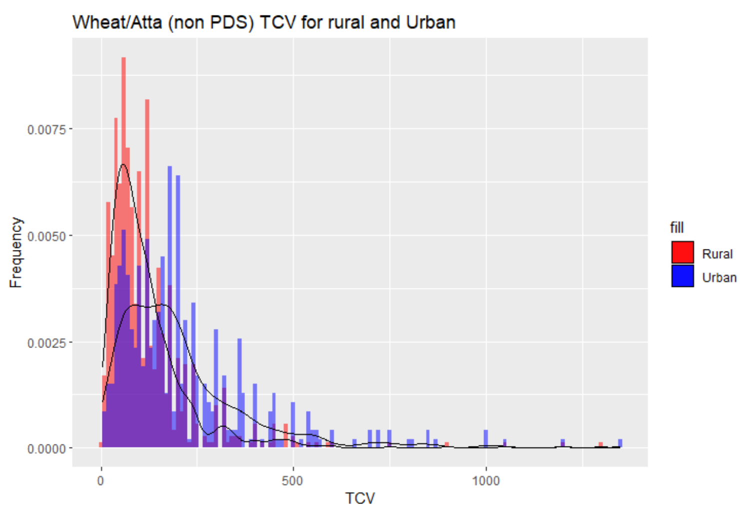


Figure 2: Wheat/Atta (non-PDS) consumption rural vs urban

From fig. 2, we can see that the urban consumption of wheat is stochastically larger than the rural consumption, i.e., it takes higher values with higher probabilities. So, it makes sense to keep wheat in the urban basket and not in the rural one. So, we can see that the role of wheat and Rice(PDS) kind of interchanged for rural and urban.

From fig. 3, we can see the peaks of rural in the upper values, hence it is almost stochastically larger than the urban. That's why it is in the rural basket, not in the urban one.

Most of the urban women don't wear Saree in general, whereas in the rural part, they wear Saree most of the time. That's why Saree is in the rural basket, not in the urban.

Also, naturally rural people use lungi more than trousers or Bermuda. That's why it is in the urban basket.

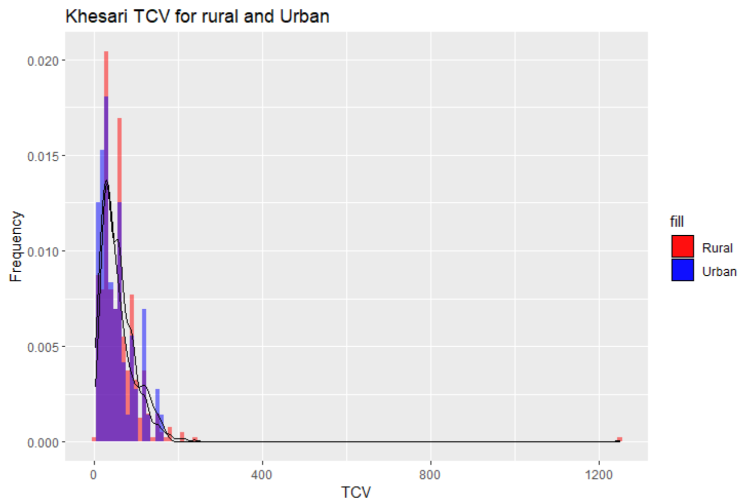


Figure 3: Khesari consumption rural vs urban

The data is from 2011, that time rural infrastructure was not that developed. There were not many hospitals in the rural sectors. So, the medical expenses of most rural families were much less. Naturally, they use natural remedies rather than artificial medical remedies. That is the reason why medicine was included in the urban basket and not in the rural basket. Hence, the total basket is:

Rural	Urban
Rice from Public Distribution System	Rice (Others)
Rice from other sources	Wheat/ Atta – other sources
Arhar, Tur	Medicine
Khesari	Firewood Chips
Firewood Chips	Electricity
Electricity	Shorts, Trousers, Bermudas
Saree	Socks, Undergarments
Socks, Undergarments	Shirts, T-shirts
Shirts, T-shirts	rubber/PVC footwear
rubber/PVC footwear	Stationary, Photocopying charges
Stationary, Photocopying charges	Stainless Steel Utensils
Stainless Steel Utensils	Bicycle
Bicycle	Motor Cycle, Scooter
Motor Cycle, Scooter	Tyres, Tubes
Tyres, Tubes	Residential building, Land
Residential building, Land	.