**Sampling of tax return data**

**Introduction:** in order to collect a representative sample for tax returns for microsimulation the following methodology may be used. See Appendix for detailed explanation of the data distribution issues.

**Note:** Tax form refers to the type of form, say Form 1A, 1B, 2, etc. (Tax on wages; Tax on self-employment income; Tax on royalties and related income; Tax on capital income; Tax on Capital Gains, Tax on Other Income; Annual Income Tax). Tax return refers to the tax return submitted by an individual or legal entity – payer of income (in case of withholding tax) which could be of any form type.

**Methodology:** For each tax form do the following:-

1. Where there are multiple tax return forms or data sources for each taxpayer **merge (link) all tax return data using the taxpayer ID**
2. We will now have for each individual or company, one row of data from all the tax returns.
3. For each category of tax forms **store the total tax collection from all the tax returns**. This information is used to check if the sample can replicate the total collection from all the tax returns.
4. Sort the tax return data on the **gross total income** or **accounting profit**;
5. Find the **lowest** and **highest gross total income/accounting profit**. Store these values;
6. Find the minimum, 10th to 100th percentiles and maximum of the gross total income– minimum, inc\_0, inc\_10, inc\_20, ..inc\_100, maximum, where inc\_0 is the lowest aggregate income; Note that inc\_0 should be a positive number i.e. inc\_0 = max(0, inc\_0);[[1]](#footnote-1)
7. If using groups of 20 (quintiles), then there would be group inc\_0, inc\_5, inc\_10, …inc\_100.
8. Store the number of tax returns in each 10-percentile block (or 5 percentile blocks if using qunitiles);
9. For the last block (inc\_90<= gross total income/accounting profit < inc\_100), **select all the tax returns** (i.e. 1:1 ratio) and **store the number of records in the block; The sample weight for this block is 1. Save these records along with the sample weight.**
10. For the second-to-last block (inc\_80<= gross total income/accounting profit < inc\_90), randomly **select one in two tax returns** (i.e. 1:2 ratio) and **store the number of records in the block**; **; The population weight for this block is 2. Save these records along with the sample weight.**
11. For the second-to-last block (inc\_70<= gross total income/accounting profit < inc\_80), randomly **select one in five tax returns** (i.e. 1:5 ratio) and **store the number of records in the block**; **The population weight for this block is 5. Save these records along with the sample weight.**
12. For returns in the block (0<= aggregate income < inc\_10):-
13. Note the number of tax returns in this bracket. Call this num\_tax\_returns\_10 and **store this information** (they will be different for each block);
14. Randomly select **1 in 10** tax returns. If the number of tax returns in this group is less than 10,000 select all returns in that group. **The population weight for this block is 10. Save these records along with the sample weight.**
15. **Repeat** for each decile (inc\_10<= aggregate income < inc\_20)…until (inc\_60<= aggregate income < inc\_70);
16. Find the minimum of the negative incomes (if any). **Store** this number;
    1. Randomly select **1 in 10** tax returns. **Store the number** of tax returns with negative incomes; **The population weight for this block is 10. Save these records along with the sample weight.**
17. Note that, along with the selected tax returns data it is necessary to provide the total number of tax returns in each bracket from which the sample was taken (including those negative incomes) so that the correct weights may be calculated;
18. The above methodology should give a sample of about 230 tax records for every 1000.
19. Test if the sample can replicate the population.
    1. Check the **total tax collection** from the sample by multiplying the data with the weight and adding up (weighted sum) - **B**. Compare this with the **total** from the full data - **A**. The error **(A-B)/B** should be less than 0.1%.
    2. Repeat the check with key columns like turnover and main deductions.
    3. For checking means calculate the sample mean and population mean for the tax, gross income, turnover, etc.
       1. sample\_mean = sample\_sum/total\_weight\_sample
       2. population\_mean = population\_sum/total\_weight\_population

and check if the error is less than 0.1%

**APPENDIX-1 Python Code Example to Sample a Dataset**

"""

This is a file that allows sampling of a large dataset.

"""

import pandas as pd

import numpy as np

pit\_df\_2021=pd.read\_csv('final\_ekamtajin\_2021.csv')

pit\_df\_2021['total\_income']=pit\_df\_2021['salary']+pit\_df\_2021['other\_income']+pit\_df\_2021['civil\_contract']

pit\_df\_2021=pit\_df\_2021.sort\_values(by=['total\_income'])

pit\_df\_2021=pit\_df\_2021.reset\_index()

# allocate the data into bins

pit\_df\_2021['bin'] = pd.qcut(pit\_df\_2021['total\_income'], 10, labels=False)

pit\_df\_2021['weight']=1

# bin\_ratio is the fraction of the number of records selected in each bin

# 1/10,...1/5, 1/1

bin\_ratio=[10,10,10,10,10,10,10,5,2,1]

frames=[]

df={}

for i in range(len(bin\_ratio)):

# find out the size of each bin

bin\_size=len(pit\_df\_2021[pit\_df\_2021['bin']==i])//bin\_ratio[i]

# draw a random sample from each bin

df[i]=pit\_df\_2021[pit\_df\_2021['bin']==i].sample(n=bin\_size)

df[i]['weight'] = bin\_ratio[i]

frames=frames+[df[i]]

pit\_sample\_2021= pd.concat(frames)

pit\_sample\_2021.to\_csv('ekamtajin\_sample\_2021.csv')

varlist = ['total\_income', 'amount\_rcvd','salary','other\_income','deduction',

'income\_tax','social\_fee', 'civil\_contract']

total\_weight\_sample = pit\_sample\_2021['weight'].sum()

total\_weight\_population = pit\_df\_2021['weight'].sum()

#comparing the statistic of the population and sample

for var in varlist:

pit\_sample\_2021['weighted\_'+var] = pit\_sample\_2021[var]\*pit\_sample\_2021['weight']

sample\_sum = pit\_sample\_2021['weighted\_'+var].sum()

population\_sum = pit\_df\_2021[var].sum()

print(" Sample Sum for ", var, " = ", sample\_sum)

print(" Population Sum for ", var, " = ", population\_sum)

print(" Sampling Error for Sum(%) ", var, " = ", "{:.2%}".format((sample\_sum- population\_sum)/sample\_sum))

sample\_mean = sample\_sum/total\_weight\_sample

population\_mean = population\_sum/total\_weight\_population

print(" Sample Mean for ", var, " = ", sample\_mean)

print(" Population Mean for ", var, " = ", population\_mean)

print("Sampling Error for Mean(%) ", var, " = ", "{:.2%}".format((sample\_mean-population\_mean)/sample\_mean))

**Appendix-2: Distribution of Upper Incomes**

Incomes at the upper end tends to follow a Pareto distribution which has a probability distribution functions

where is the Pareto parameter. Diamond and Saez (2011) estimate this parameter to be 1.5 for tax returns in the United States.

**Figure – 1: Distribution of Incomes - an illustration**

A close up of a map

Description automatically generated

In Figure-1, a typical income distribution is illustrated where the incomes are normally distributed below a certain level of income and above that level is Pareto distributed with the parameter of 1.5. In such a distribution the top 10% of individuals earn 28% of all the incomes of the population. The skewness in the distribution is also reflected in the fact that the median income is less than the mean income.

**Table 1: Income Distribution – illustration (9,491 taxpayers)**

|  |  |  |
| --- | --- | --- |
| **Income Decile** | **Cumulative Population** | **Decile income** |
| 10% | 949 | 220,000 |
| 20% | 1,898 | 400,000 |
| 30% | 2,847 | 550,000 |
| 40% | 3,796 | 690,000 |
| 50% | 4,745 | 840,000 |
| 60% | 5,694 | 990,000 |
| 70% | 6,643 | 1,050,000 |
| 80% | 7,592 | 1,400,000 |
| 90% | 8,542 | 1,860,000 |
| 100% | 9,491 | 5,000,000 |

An example of the distribution of 9,491 persons incomes in Table 1 with income levels from 0 to 5,000,000 shows that the first decile has incomes between 0 and 200,000 (a gap of 200,000), while the last decile has incomes from 1.86 million to 5 million (a gap of 3.14 million).

The distribution of tax contribution by incomes would be further skewed due to a progressive income tax. Figure 2 below shows that for a modestly progressive tax the contribution of tax could be highly skewed.

A screenshot of a cell phone

Description automatically generated

**Table 2: Distribution of Tax Revenue – an illustration (Total Revenue 978 million)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revenue Decile** | **Cumulative Revenue** | **Income level** | **Income Percentile** |
| 10% | 97,850,665 | 580,000 | 32% |
| 20% | 195,701,329 | 790,000 | 47% |
| 30% | 293,551,994 | 960,000 | 58% |
| 40% | 391,402,658 | 1,130,000 | 68% |
| 50% | 489,253,323 | 1,310,000 | 77% |
| 60% | 587,103,988 | 1,520,000 | 84% |
| 70% | 684,954,652 | 1,860,000 | 90% |
| 80% | 782,805,317 | 2,540,000 | 95% |
| 90% | 880,655,981 | 3,580,000 | 98% |
| 100% | 978,506,646 | 5,000,000 | 100% |

An example of the distribution of revenue of the same 9,491 persons of Table 1 is shown in Table 2 contributing to total revenue of 978 million with each decile contributing 97.8 million. The first decile of revenues is contributed by 32% of incomes (persons) between incomes 0 and 580,000 while the last decile is contributed by the top 2% of the incomes (between the 98% and the 100th percentile).

**Conclusion:** The discussion above indicates the importance of appropriately sampling the distribution of incomes in order tocapture the most information from the tax return for analysis. Ideally the goal is to obtain as much of a sample from the tax return population to equate the tax revenue from the samples across each income percentile group.

1. The 10th percentile refers to the 10% of taxpayers with the lowest gross total income/accounting profit. Inc\_10 should reflect the gross total income of the person with the highest income in the cohort of 10% of taxpayers with the lowest income. [↑](#footnote-ref-1)