Lab 8

November 4, 2021

1 Lab 8

In this lab we discuss simple random sampling, systematic sampling, and stratified random sampling.

1.1 Simple Random Sampling

random.sample: https://www.w3schools.com/python/ref_random_sample.asp

[1]:
```
import numpy as np
import pandas as pd
import random
```

[2]:
```
# Random sampling using random.sample()
# random.sample returns unique random elements from a sequence or set. This is␣
 ↪sampling without replacement.
random.seed(52) ## setting the seed so that we all get the same sample
names = ["Roger", "Jack", "John", "Jason", "Laura", "Mariya", "Martina",␣
 ↪"Lauren"]
sampled_list1 = random.sample(names, 3)
print(sampled_list1)
```

```
['Laura', 'Roger', 'Mariya']
```

[3]:
```
# if we change the seed another sample is obtained
random.seed(17)
sampled_list2 = random.sample(names, 3)
print(sampled_list2)
```

```
['Martina', 'Lauren', 'John']
```

[4]:
```
# an alternative way to sample is to assign a number to each name (or ID) in␣
 ↪the list and then sample the numbers
# generating 8 consecutive numbers corresponding to the positions of each name␣
 ↪in the list:
sequence_numbers = list(range(8))
print(sequence_numbers) # Python starts with zero
```

```
random.seed(17) # same seed as before to obtain the same sample of names as in␣
 ↪the code cell above
sample_list3 = random.sample(sequence_numbers,3) # randomly sampling 3 number␣
 ↪positions
print(sample_list3)
```

```
[0, 1, 2, 3, 4, 5, 6, 7]
[6, 7, 2]
```

[5]:
```
[names[i] for i in sample_list3 ] # getting the names corresponding to the␣
 ↪sampled number positions
```

[5]:
```
['Martina', 'Lauren', 'John']
```

[6]:
```
# Getting a sample array from a multidimensional array
array = np.array([[2, 5, 7], [5, 11, 16], [6, 13, 19], [7, 15, 22], [8, 17,␣
 ↪25]])
print("2D array \n", array)
```

```
2D array
 [[ 2  5  7]
 [ 5 11 16]
 [ 6 13 19]
 [ 7 15 22]
 [ 8 17 25]]
```

[7]:
```
random.seed(48)
random_rows = random.sample(range(5), 2) # randomly selecting two row indices,␣
 ↪without replacement
print(random_rows)
```

```
[4, 2]
```

[8]:
```
array[random_rows, :]
```

[8]:
```
array([[ 8, 17, 25],
       [ 6, 13, 19]])
```

1.2 Systematic Sampling

Systematic sampling is a type of sampling where we obtain a sample by going through a list of the population at fixed intervals from a randomly chosen starting point.

np.arange: https://numpy.org/doc/stable/reference/generated/numpy.arange.html

[9]:
```
# Let's assume we are interested in sampling from a population of 15 students␣
 ↪with the following ID list:
df = pd.DataFrame({'ID':np.arange(1, 16).tolist()})
df
```

[9]:
	ID
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10
10	11
11	12
12	13
13	14
14	15

[10]:
```
# Defining the function for systematic sampling
def systematic_sampling(df, starting_index, step):
    indices = np.arange(starting_index, len(df), step = step)
    systematic_sample = df.iloc[indices]
    return systematic_sample

# Obtaining a systematic sample of size 5
# Because 15/3=5, choose one of the first 3 IDs on the list at random and then␣
 ↪every 3rd ID after that.
random.seed(68)
random_start = random.randint(0,2)
print(random_start)
# another way
# random.seed(68)
# random_start = random.sample(range(3),1)
# print(random_start)
systematic_sample = systematic_sampling(df, random_start, 3)
systematic_sample # recall that Python starts at position 0, so position 2␣
 ↪corresponds to ID = 3
```

```
2
```

[10]:
	ID
2	3
5	6
8	9
11	12
14	15

1.3 Stratified Random Sampling

Another type of sampling is stratified random sampling, in which a population is split into groups and a certain number of members from each group are randomly selected to be included in the sample.

1.3.1 Stratified Random Sampling Using Counts

[11]:
```
# Suppose we have the following dataframe containing the ID of 8 students from␣
 ↪2 different undergrad programs.
# This is our population list.
df = pd.DataFrame({'ID':np.arange(1, 9).tolist(),
                   'program':['Stats']*4 + ['Math']*4}) # 4 students in Stats,␣
 ↪4 students in Math
df
```

[11]:

	ID	program
0	1	Stats
1	2	Stats
2	3	Stats
3	4	Stats
4	5	Math
5	6	Math
6	7	Math
7	8	Math

[12]:
```
# random sample of Stats students
df_Stats = df[df['program'] == 'Stats']
df_Stats
random_rows = random.sample(range(4), 2) #randomly selecting 2 students from␣
 ↪the 4 Stats students in the population
print(df_Stats.iloc[random_rows])
```

	ID	program
3	4	Stats
2	3	Stats

[13]:
```
# random sample of Math students
df_Math = df[df['program'] == 'Math']
df_Math
random_rows = random.sample(range(4), 2) #randomly selecting 2 students from␣
 ↪the 4 Math students in the population
print(df_Math.iloc[random_rows])
```

	ID	program
4	5	Math
6	7	Math

[14]:
```
# Alternative way using one line of code and additional Python functions
```

DataFrame.groupby: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
DataFrame.apply: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.apply.html
DataFrame.sample: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.sample.html

[15]:
```
# Stratified random sampling by randomly selecting 2 students from each program␣
 ↪to be included in the sample
df.groupby('program', group_keys = False).apply(lambda x:x.sample(2))
```

[15]:

	ID	program
6	7	Math
4	5	Math
1	2	Stats
2	3	Stats

1.3.2 Stratified Random Sampling Using Proportions

[16]:
```
# Now suppose we have the following population dataframe with 25% Stats and 75%␣
 ↪Math students.
df = pd.DataFrame({'ID':np.arange(1, 9).tolist(),
                   'program':['Stats']*2 + ['Math']*6}) # 2 students in Stats,␣
 ↪6 students in Math
df
```

[16]:

	ID	program
0	1	Stats
1	2	Stats
2	3	Math
3	4	Math
4	5	Math
5	6	Math
6	7	Math
7	8	Math

np.rint: https://numpy.org/doc/stable/reference/generated/numpy.rint.html

[17]:
```
# Stratified random sampling such that the proportion of students in each␣
 ↪program sample
# matches the proportion of students from each program in the population␣
 ↪dataframe
N = 4 # sample size
# So, the sample must contain 1 random student from Stats and 3 from Math to␣
 ↪maintain the population proportions
```

[18]:
```
# random sample of Stats students
df_Stats = df[df['program'] == 'Stats']
df_Stats
random_rows = random.sample(range(2), 1) #sampling 1 student from the 2 Stats␣
 ↪students in the population
```

```
print(df_Stats.iloc[random_rows])
```

```
   ID program
0   1   Stats
```

[19]:
```
# random sample of Math students
df_Math = df[df['program'] == 'Math']
df_Math
random_rows = random.sample(range(6), 3) #sampling 3 students from the 6 Math␣
 ↪students in the population
print(df_Math.iloc[random_rows])
```

```
   ID program
5   6    Math
7   8    Math
4   5    Math
```

Alternative way:
np.rint: https://numpy.org/doc/stable/reference/generated/numpy.rint.html

[20]:
```
df.groupby('program', group_keys = False).apply(lambda x:x.sample(int(np.rint(N␣
 ↪* len(x) / len(df)))))
```

[20]:
```
   ID program
3   4    Math
6   7    Math
7   8    Math
0   1   Stats
```

[ ]:
```

```