Stratified Sampling Exercise

Researchers often take samples from a population and use the data from the sample to draw conclusions about the population as a whole.

One commonly used sampling method 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.

Let's learn two methods for performing stratified random sampling in Python.

Stratified Sampling Using Counts

Suppose we have the following pandas DataFrame that contains data about 8 basketball players on 2 different teams:

```
import pandas as pd

# create DataFrame
df = pd.DataFrame({'team': ['A', 'A', 'A', 'A', 'B', 'B', 'B', 'B'],
                   'position': ['G', 'G', 'F', 'G', 'F', 'F', 'C',
'C'],
                   'assists': [5, 7, 7, 8, 5, 7, 6, 9],
                   'rebounds': [11, 8, 10, 6, 6, 9, 6, 10]})

# view DataFrame
print(df)
```

```
  team position  assists  rebounds
0    A        G        5        11
1    A        G        7         8
2    A        F        7        10
3    A        G        8         6
4    B        F        5         6
5    B        F        7         9
6    B        C        6         6
7    B        C        9        10
```

Now you want to group them into two different groups, consisting of two players from each team.

The following code shows how to perform stratified random sampling by randomly selecting 2 players from each team to be included in the sample:

```
df.groupby('team', group_keys=False).apply(lambda x: x.sample(2))
```

	team	position	assists	rebounds
3	A	G	8	6
0	A	G	5	11
4	B	F	5	6
5	B	F	7	9

Notice that two players from each team are included in the stratified sample.

Stratified Sampling Using Proportions

Once again suppose we have the following pandas DataFrame that contains data about 8 basketball players on 2 different teams:

```
import pandas as pd

#create DataFrame
df = pd.DataFrame({'team': ['A', 'A', 'B', 'B', 'B', 'B', 'B', 'B'],
                   'position': ['G', 'G', 'F', 'G', 'F', 'F', 'C',
'C'],
                   'assists': [5, 7, 7, 8, 5, 7, 6, 9],
                   'rebounds': [11, 8, 10, 6, 6, 9, 6, 10]})

#view DataFrame
df
```

	team	position	assists	rebounds
0	A	G	5	11
1	A	G	7	8
2	B	F	7	10
3	B	G	8	6
4	B	F	5	6
5	B	F	7	9
6	B	C	6	6
7	B	C	9	10

Notice that 6 of the 8 players (75%) in the DataFrame are on team A and 2 out of the 8 players (25%) are on team B.

The following code shows how to perform stratified random sampling such that the proportion of players in the sample from each team matches the proportion of players from each team in the larger DataFrame:

```
import numpy as np
```

```
#define the total sample size desired
N = 4

#perform stratified random sampling
df.groupby('team',group_keys=False).apply(lambdax:x.sample(int
(np.rint(N*len(x)/len(df))))).sample(frac=1).reset_index(drop=True)
```

	team	position	assists	rebounds
0	B	C	6	6
1	A	G	5	11
2	B	F	5	6
3	B	F	7	9

Notice that the proportion of players from team A in the stratified sample (25%) matches the proportion of players from team A in the larger DataFrame.

Similarly, the proportion of players from team B in the stratified sample (75%) matches the proportion of players from team B in the larger DataFrame.