Law_of_Small_Numbers

January 9, 2021

[1]:
```
import random
from pandas import Series, DataFrame
```

1 The Dangerous Law of Small Numbers

Based on the story at the beginning of Chapter 10 in Kahneman's "Thinking, Fast and Slow".

The law of large numbers and the central limit theorem both tell us that, when we have a big sample, the mean of the sample will be pretty close to the mean for the overall population. If we act as if the same is true for small samples, we will make some wrong inferences.

Here we simulate large and small counties. For each, we simulate the observed kidney cancer rate in a given year. Because small counties are effectively small samples, there is higher variance in the observed kidney rates, even if the underlying true rates are exactly the same. Higher variance means that both the lowest **and** highest observed cancer rates will come from small counties. We shouldn't be fooled into thinking there's actually a difference in the cancer rates.

1.0.1 Simulating a County

We will determine an observed cancer rate by simulating a random outcome for each of the people.

To make things simple, we will: - assume *all* counties have the same true cancer rate, 4% - simulate 10 counties each of size 100, 2000, 30000, and 400000

Of course, the real kidney cancer rate is much lower and counties are much bigger. But these assumptions make it easier to read off the observed cancer rates, without having to squint or count the number of zeros after the decimal point.

(Note: because all students should have seen python list comprehensions before, but not everyone taking this course will have yet learned how to use numpy or pandas, I've chosen to run these simulations without using numpy or pandas. They would be a little simpler using them.)

[1]:
```
def simulate_one_person(true_cancer_rate):
    # return True if a person is simulated to have cancer
    # random.random() gives a value between 0 and 1
    return random.random() < true_cancer_rate
```

[3]:
```
# Generate one simulated observed cancer rate for a county; express as␣
↪percentage between 0 and 100
def observed_rate(pop_size, true_cancer_rate):
    # returns the observed cancer rate for a simulated county with pop_size␣
↪people
```

```
    simulated_outcomes = [simulate_one_person(true_cancer_rate) for _ in␣
 ↪range(pop_size)]
    # multiply by 100 to express as a percentage
    return 100 * sum(simulated_outcomes) / len(simulated_outcomes)

# sample run
observed_rate(1000, .04)
```

[3]:
```
3.5
```

[4]:
```
# Repeatedly generate an observed cancer rate for a county of a given size
def simulate_counties(n_counties, pop_size, true_cancer_rate):
    return [observed_rate(pop_size, true_cancer_rate) for _ in␣
 ↪range(n_counties)]

# sample run
simulate_counties(10, 1000, .04)
```

[4]:
```
[4.4, 4.5, 2.9, 4.2, 4.4, 4.0, 4.3, 3.8, 3.5, 3.7]
```

[5]:
```
population_sizes = [100, 2000, 30000, 400000]
counties_of_each_size = 10
true_cancer_rate = .04

data = [simulate_counties(counties_of_each_size, pop_size, true_cancer_rate)␣
 ↪for pop_size in population_sizes]

# Put it in pandas data frame for pretty printing
DataFrame(data, index=['Pop=%d' % pop_size for pop_size in population_sizes])
```

[5]:

	0	1	2	3	4	5	6	\
Pop=100	4.00000	4.000000	5.000000	3.000000	6.0000	2.000000	6.000000	
Pop=2000	3.65000	4.450000	3.150000	4.300000	4.3000	4.100000	4.150000	
Pop=30000	4.16000	4.086667	4.033333	3.833333	3.9500	3.966667	4.176667	
Pop=400000	3.96475	4.055750	3.997750	3.999750	3.9615	4.066500	3.979500	

	7	8	9
Pop=100	4.000000	2.00000	3.000000
Pop=2000	4.150000	3.95000	3.150000
Pop=30000	3.956667	4.09000	3.983333
Pop=400000	4.018000	4.05025	3.983500

Notice that: - as the simulated population is larger, the counties all have observed cancer rates close to the true rate, 4%. - the smallest and largest observed cancer rates are in the low population counties