KEY_Lesson15_Basic_Stats_II

August 14, 2019

1 Introduction to Statistics Part II

Now that we have learned how to use the mean and median, we'll talk about some more advanced statistics.

[1]:
```
# import pandas and numpy
import numpy as np
import pandas as pd
# mount Google Drive
from google.colab import drive
drive.mount('/content/gdrive')
path = '/content/gdrive/My Drive/SummerExperience-master/'
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call
drive.mount("/content/gdrive", force_remount=True).
```

1.1 Count Statistics

Count variables are variables which represent the number of events that occur of a specific category. This can be anything, like the number of dogs in a park or how many people went to a concert. For both of these examples, each of the counts must be *whole numbers*, i.e. `int` data type.

Run the cell below to load a listing of the weather in Detroit for every day since 1950:

[0]:
```
data_table = pd.read_csv(path + 'SampleData/detroit_weather.csv') # Data from␣
↪Mathematica WeatherData, 2019
```

Take a look at the contents of `data_table`:

[3]:
```
# Run head on data_table to look at its contents
data_table.head(10)
```

[3]:

	Unnamed: 0	YEAR	MONTH	DAY	Rain	Snow
0	0	1950	1	1	True	False
1	1	1950	1	2	True	False
2	2	1950	1	3	True	False
3	3	1950	1	4	True	True
4	4	1950	1	5	False	False

```
5           5  1950      1    6  False   True
6           6  1950      1    7  False   True
7           7  1950      1    8  False   True
8           8  1950      1    9  False  False
9           9  1950      1   10   True  False
```

[4]:
```
# Run tail on data_table to look at its contents
data_table.tail(10)
```

[4]:
```
       Unnamed: 0  YEAR  MONTH  DAY   Rain   Snow
25304       25304  2019      4   22  False  False
25305       25305  2019      4   23  False  False
25306       25306  2019      4   24  False  False
25307       25307  2019      4   25  False  False
25308       25308  2019      4   26   True  False
25309       25309  2019      4   27  False  False
25310       25310  2019      4   28   True  False
25311       25311  2019      4   29  False  False
25312       25312  2019      4   30   True  False
25313       25313  2019      5    1   True  False
```

This table contains if it was snowing and if it was raining for each day in Detroit since 1950. We will use this as an example dataset.

[5]:
```
# Lookup the weather for May 1, 2019:
data_table.query('YEAR == 2019 and MONTH == 5 and DAY == 1')
```

[5]:
```
       Unnamed: 0  YEAR  MONTH  DAY  Rain   Snow
25313       25313  2019      5    1  True  False
```

[6]:
```
# another way to do the same thing is chain together multiple calls to query
data_table.query('YEAR == 2019').query('MONTH == 5').query('DAY == 1')
```

[6]:
```
       Unnamed: 0  YEAR  MONTH  DAY  Rain   Snow
25313       25313  2019      5    1  True  False
```

As we can see, it was raining, but not snowing that day!

Now, let's create some count statistics! To do this, we will use the `Counter` **module** from the `collections` **package**. Let's import it!

[0]:
```
# Import the Counter class from collections to help us do the counting

from collections import Counter
```

`Counter` summarizes any `list` with the counts of all its unique variables:

[8]:
```
# Create a list and count it using Counter

Counter([1,1,1,1,1,2,2,2,2,2])
```

[8]:
```
Counter({1: 5, 2: 5})
```

Now, let's count the weather data!

[9]:
```
# Count how many days it has snowed in Detroit since 1950:

snow_days = Counter(data_table["Snow"])
snow_days
```

[9]:
```
Counter({False: 21079, True: 4235})
```

It looks like it has snowed 4,235 days in that time period, that is a lot!

This `Counter` variable functions a lot like a dictionary object - we haven't talked about this data type in this course, but essentially its a way of mapping **keys** to **values**. We can access the **values** associated with each **key** in a similar way that we index lists. For example, if we wanted to get the total number of snow days in our data set:

[10]:
```
snow_days[True]
```

[10]:
```
4235
```

Let's break this down a little more granularly. How are these 4235 total snow days distributed across our 12 months?

[11]:
```
# Count how many days *per month* it has snowed since 1950:

snow_days_by_month = Counter(data_table.query('Snow')["MONTH"])
print(snow_days_by_month)
```

```
Counter({1: 1110, 12: 933, 2: 903, 3: 648, 11: 369, 4: 227, 10: 34, 5: 10, 8:
1})
```

What about the days that is *has not* snowed per month?

[12]:
```
# How many days *per month* has it NOT snowed since 1950?

not_snow_days_by_month = Counter(data_table.query('not Snow')["MONTH"])
print(not_snow_days_by_month)
```

```
Counter({7: 2139, 8: 2137, 5: 2130, 10: 2105, 6: 2070, 9: 2069, 4: 1873, 11:
1701, 3: 1522, 12: 1199, 2: 1074, 1: 1060})
```

1.2 Percentages

A *percentage* is a number between 0 and 1 which represents the fraction of a given variable that meets a given condition. i.e. if there are 28 dogs and 45 cats at the humane society, the percentage of adoptable animals that are dogs is:

[13]:
```
28/(28 + 45)
```

[13]:
```
0.3835616438356164
```

First, let's calculate the percentage of all days since 1950 that have been snow days using the variable `snow_days` from above.

[14]:
```
snow_days[True]/(snow_days[True]+snow_days[False])
```

[14]:
```
0.16729872797661374
```

Now, let's calculate the percent of January days that have had snow since 1950. To do this, we first need the total number of January days since 1950.

[15]:
```
# How many days TOTAL have there been in each month since 1950?

days_by_month = Counter(data_table["MONTH"])
print(days_by_month)
```

```
Counter({1: 2170, 3: 2170, 5: 2140, 7: 2139, 10: 2139, 8: 2138, 12: 2132, 4:
2100, 6: 2070, 11: 2070, 9: 2069, 2: 1977})
```

Now let's use the `snow_days_by_month` and `days_by_month` variables to isolate the **values** associated with the **key** for January to calculate the percentage.

[16]:
```
# Find the percentage of days in January where it snowed:

snow_days_by_month[1] / days_by_month[1]
```

[16]:
```
0.511520737327189
```

A percentage of 51% means that half the January days since 1950 have seen snowfall.

Now let's do the same for June.

[17]:
```
# Now do the same for June:

snow_days_by_month[6] / days_by_month[6]
```

[17]:
```
0.0
```

It shouldn't come as much suprise that it doesn't snow much in summer!

In this lesson you learned how to:

- Calculate count statistics using data from `pandas`
- Calculate percentages from count statistics

Now, lets continue to practice with your partner!