

KEY_Practice16_Basic_Stats_II_Percents

June 4, 2020

For this practice, let's use the California housing dataset.

```
[0]: # Import the fetch_california_housing method & load the data
from sklearn.datasets import fetch_california_housing
california = fetch_california_housing()
```

```
[0]: # Import pandas, so that we can work with the data frame version of the
      ↪ California housing dataset
import pandas as pd
```

```
[3]: # Print the characteristics of the California housing dataset
print(california.DESCR)
```

```
.. _california_housing_dataset:
```

California Housing dataset

****Data Set Characteristics:****

:Number of Instances: 20640

:Number of Attributes: 8 numeric, predictive attributes and the target

:Attribute Information:

- MedInc median income in block
- HouseAge median house age in block
- AveRooms average number of rooms
- AveBedrms average number of bedrooms
- Population block population
- AveOccup average house occupancy
- Latitude house block latitude
- Longitude house block longitude

:Missing Attribute Values: None

This dataset was obtained from the StatLib repository.

<http://lib.stat.cmu.edu/datasets/>

The target variable is the median house value for California districts.

This dataset was derived from the 1990 U.S. census, using one row per census block group. A block group is the smallest geographical unit for which the U.S. Census Bureau publishes sample data (a block group typically has a population of 600 to 3,000 people).

It can be downloaded/loaded using the
:func:`sklearn.datasets.fetch_california_housing` function.

.. topic:: References

- Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregressions, Statistics and Probability Letters, 33 (1997) 291-297

```
[22]: # Convert the California housing data to a dataframe format so it's easier to
      ↪view and process
california_df = pd.DataFrame(california['data'], columns =
      ↪california['feature_names'])
california_df['HouseValue'] = california['target']
california_df
```

```
[22]:
```

	MedInc	HouseAge	AveRooms	...	Latitude	Longitude	HouseValue
0	8.3252	41.0	6.984127	...	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	...	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	...	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	...	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	...	37.85	-122.25	3.422
...
20635	1.5603	25.0	5.045455	...	39.48	-121.09	0.781
20636	2.5568	18.0	6.114035	...	39.49	-121.21	0.771
20637	1.7000	17.0	5.205543	...	39.43	-121.22	0.923
20638	1.8672	18.0	5.329513	...	39.43	-121.32	0.847
20639	2.3886	16.0	5.254717	...	39.37	-121.24	0.894

[20640 rows x 9 columns]

Determine the percentage of recently built houses (i.e. houses with an age less than 10 years).

```
[23]: # Using the boolean array method,
      # get the number of houses less than 10 years old
num_new_houses = sum(california_df['HouseAge'] < 10)

      # Determine the total number of houses in the dataset
total_num = len(california_df['HouseAge'])
```

```
# Calculate the percentage of recently built houses.
num_new_houses/total_num*100
```

[23]: 6.3226744186046515

What is the easiest way to calculate the percentage of houses that are 10 years or older? Try to do this in one line of code.

```
[24]: 100 - num_new_houses/total_num*100
```

[24]: 93.67732558139535

That's right! Just take the difference from 100%.

Now, let's double check this by calculating the percentage using comparison operators (<, >, ==, >=, !=, ==).

```
[25]: # Determine number of houses with an age of 10 years or greater.
num_old_houses = sum(california_df['HouseAge'] >= 10)

# Calculate the percentage of older houses.
num_old_houses/total_num*100
```

[25]: 93.67732558139535

Nicely done!

Let's do another problem. Determine the percentages of houses that are **less than** 20 years old **AND** have an average value of **greater than or equal to** \$80,000 (which is 0.8 in this data, HouseValue is in units of \$100,000).

You'll be using logical operators (&, |) to solve this problem. The "&" operator signifies all conditions must be true, while "|" only requires one of the conditions to be true.

```
[26]: # Determine number of houses with an age less than 20 years AND valued at
      ↳ $80,000 or more
num_both = sum((california_df['HouseAge'] < 20) & (california_df['HouseValue']
      ↳ >= 0.8))

# Calculate the percentage of these houses meeting both conditions from the
      ↳ total.
num_both/total_num*100
```

[26]: 26.148255813953487

Now we'll calculate the percentages of houses that are **less than** 20 years old **OR** have an average value of **greater than or equal to** 0.8.

```
[27]: # Determine number of houses with an age less than 20 years OR cost $80,000 or
      ↪ more
      num_either = sum((california_df['HouseAge'] < 20) |
      ↪ (california_df['HouseValue'] >= 0.8))

      # Calculate the percentage of these houses meeting either condition from the
      ↪ total.
      num_either/total_num*100
```

[27]: 92.84399224806201

Why are these two results different?

Nice work learning how to calculate percentages!