Statistics I

Statistics - Mathematics

GitHub

- Arithmetic mean
- Geometric mean
- Harmonic mean
- Mode
- Median
- Range
- Variance
- Standard deviation
- Ouput

Round up calculations via `round_n` parameter.

```
round_n = int(input("Round up [decimal places]: "))
```

```
Round up [decimal places]:  10
```

Data set

- Data to input via `load_data(file)` from `data_set1.txt`.
- Parameter `data_set` with all data elements.
- Visualize data elements via `pandas`.

```
def load_data(file):
    loaded_data = []
    with open(file, "r") as input_data:
        for line in input_data:
            loaded_data.append(int(line.strip()))
    return loaded_data


data_set = load_data("data_set1.txt")

import pandas as pd
df_data = {"x": data_set}
pd.DataFrame.from_dict(df_data)
```

	x
0	302
1	310
2	312
3	310
4	313
5	318
6	305
7	309
8	310
9	309

Arithmetic mean

`arithmetic_mean(arr)` returns `X_a`.

$$A = \frac{1}{n}\sum_{i-1}^{n} x_i = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

```
def arithmetic_mean(arr):
    x_a = int()
    for i in range(len(arr)):
        x_a += arr[i]
    return round(x_a / len(arr), round_n)


X_a = arithmetic_mean(data_set)
```

Geometric mean

`geometric_mean(arr)` returns `X_g`.

$$G = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 x_2 \cdots x_n}$$

```
def geometric_mean(arr):
    x_g = arr[0]
    for i in range(1, len(arr)):
        x_g *= arr[i]
    return round(x_g ** (len(arr) ** - 1), round_n)


X_g = geometric_mean(data_set)
```

Harmonic mean

`harmonic_mean(arr)` returns `X_h`.

$$H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}} = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} = \left(\frac{\sum_{i=1}^{n} x_i^{-1}}{n} \right)^{-1}$$

```
def harmonic_mean(arr):
    x_h = float()
    for i in range(len(arr)):
        x_h += (1 / arr[i])
    return round(len(arr) / x_h, round_n)


X_h = harmonic_mean(data_set)
```

Statistics II

Mode

`mode(arr)` returns `mod_x`.

```
def mode(arr):
    from collections import Counter
    frequency = []
    for key, value in Counter(arr).items():
        value_set = (value, key)
        frequency.append(value_set)

    max_frequency = max(frequency)
    return max_frequency[1]


mod_x = mode(data_set)
```

Median

`median(arr)` returns `med_x`.

```
def median(arr):
    r_arr = len(arr)
    ordered_arr = sorted(arr)
    index = int(r_arr / 2)

    if r_arr % 2 == 0:
        return round((ordered_arr[index - 1] + ordered_arr[index]) / 2, round_n)
    else:
        return round(index)


med_x = median(data_set)
```

Range

`range_function(arr)` returns `range_x`.

$$R = x_{\max} - x_{\min}$$

```
def range_function(arr):
    return min(arr), max(arr)


r = range_function(data_set)
range_x = f"<{r[0]};{r[1]}>"
```

Statistics III

Variance

`variance_function(arr, avg)` returns `v_x`.

$$s = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

```
def variance_function(arr, avg):
    v_sum = float()
    for i in range(len(arr)):
        v_sum += (arr[i] - avg) ** 2
    return round(v_sum / len(arr), round_n)


v_x = variance_function(data_set, X_a)
```

Standard deviation

`standard_deviation(v)` returns `s`.

$$s = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

```
def standard_deviation(v):
    return round(v ** (1 / 2), round_n)


s = standard_deviation(v_x)
```

Output

Output

Display calculations using `pandas`.

```
import pandas as pd

methods = ["Arithmetic mean", "Geometric mean", "Harmonic mean", "Mode", "Median", "Variance", "Stand
values = [X_a, X_g, X_h, mod_x, med_x, v_x, s]
data = {'Method': methods, 'Calculation': values}
pd.DataFrame.from_dict(data)
```

	Method	Calculation
0	Arithmetic mean	309.800000
1	Geometric mean	309.772947
2	Harmonic mean	309.745889
3	Mode	310.000000
4	Median	310.000000
5	Variance	16.760000
6	Standard deviation	4.093898