Commonly Used NumPy Statistical Functions

Sunil Kunwar

October 25, 2024

NumPy provides a set of functions to perform various statistical operations. Here are some commonly used NumPy statistical functions:

1 Basic Statistics

1.1 Mean (`numpy.mean`)

The mean (average) of the data can be calculated using:

$$\text{mean\_value} = \text{np.mean(data)}$$

```
import numpy as np

data = np.array([1, 2, 3, 4, 5])
mean_value = np.mean(data)
print(mean_value)  # Output: 3.0
```

1.2 Median (`numpy.median`)

The median (middle value) of the data can be calculated using:

$$\text{median\_value} = \text{np.median(data)}$$

```
median_value = np.median(data)
print(median_value)  # Output: 3.0
```

1.3 Standard Deviation (`numpy.std`)

The standard deviation (spread of data) can be calculated using:

$$\text{std\_value} = \text{np.std(data)}$$

```
std_value = np.std(data)
print(std_value)  # Output: 1.4142135623730951
```

1.4 Variance (`numpy.var`)

The variance (measure of data spread from the mean) can be calculated using:

$$\text{var\_value} = \text{np.var(data)}$$

```
var_value = np.var(data)
print(var_value)  # Output: 2.0
```

1.5 Minimum and Maximum (`numpy.min`, `numpy.max`)

The minimum and maximum values of the data can be calculated using:

$$\text{min\_value} = \text{np.min(data)}, \quad \text{max\_value} = \text{np.max(data)}$$

```
min_value = np.min(data)
max_value = np.max(data)
print(min_value, max_value)  # Output: 1 5
```

1.6 Percentiles (`numpy.percentile`)

The nth percentile of the data can be calculated using:

$$\text{percentile\_n} = \text{np.percentile(data, n)}$$

```
percentile_50 = np.percentile(data, 50)
print(percentile_50)  # Output: 3.0
```

1.7 Correlation Coefficient (`numpy.corrcoef`)

The correlation between two datasets can be calculated using:

$$\text{correlation} = \text{np.corrcoef(data1, data2)}$$

```
data1 = np.array([1, 2, 3, 4])
data2 = np.array([4, 3, 2, 1])
correlation = np.corrcoef(data1, data2)
print(correlation)
```

1.8 Covariance (`numpy.cov`)

The covariance between two datasets can be calculated using:

$$\text{covariance} = \text{np.cov(data1, data2)}$$

```
covariance = np.cov(data1, data2)
print(covariance)
```

1.9 Histogram (`numpy.histogram`)

To create a histogram and get bin edges, use:

$$\text{hist}, \text{bin\_edges} = \text{np.histogram}(\text{data}, \text{bins}=5)$$

```
hist, bin_edges = np.histogram(data, bins=5)
print(hist, bin_edges)
```

2 Advanced Statistics

2.1 Cumulative Statistics

2.1.1 Cumulative Sum

The cumulative sum of an array can be calculated using:

$$\text{cumulative\_sum} = \text{np.cumsum}(\text{data})$$

```
cumulative_sum = np.cumsum(data)
```

2.1.2 Cumulative Product

The cumulative product of an array can be calculated using:

$$\text{cumulative\_product} = \text{np.cumprod}(\text{data})$$

```
cumulative_product = np.cumprod(data)
```

2.2 Weighted Average

The weighted average of data with corresponding weights can be calculated using:

$$\text{weighted\_avg} = \text{np.average}(\text{data}, \text{weights}=\text{weights})$$

```
weights = np.array([0.1, 0.2, 0.3, 0.4, 0.5])
weighted_avg = np.average(data, weights=weights)
```

2.3 Running/Moving Average

A simple moving average can be computed using:

$$\text{moving\_avg} = \text{np.convolve}(\text{data}, \text{window}, \text{mode}=\text{'valid'})$$

```
window = np.ones(3) / 3
moving_avg = np.convolve(data, window, mode='valid')
```

3 Miscellaneous Statistics

3.1 Quartiles

Quartiles can be calculated using the 25th and 75th percentiles:

$$\text{q1} = \text{np.percentile(data, 25)}, \quad \text{q3} = \text{np.percentile(data, 75)}$$

```
q1 = np.percentile(data, 25)
q3 = np.percentile(data, 75)
```

3.2 Range

The range (difference between maximum and minimum values) can be calculated using:

$$\text{range\_value} = \text{np.ptp(data)}$$

```
range_value = np.ptp(data)
```