codecademy

Statistics in NumPy

Conditions in Numpy.mean()

In Python, the function `numpy.mean()` can be used to calculate the percent of array elements that satisfies a certain condition.

```
import numpy as np
a = np.array([1,2,3,4])
np.mean(a)
# Output = 2.5

np.mean(a>2)
# The array now becomes array([False, False, True, True])
# True = 1.0,False = 0.0
# Output = 0.5
# 50% of array elements are greater than 2
```

NumPy's Mean and Axis

In a two-dimensional array, you may want the mean of just the rows or just the columns. In Python, the NumPy `.mean()` function can be used to find these values. To find the average of all rows, set the axis parameter to 1. To find the average of all columns, set the axis parameter to 0.

We will use the following 2-dimensional array for this example:

```
py
ring_toss = np.array([[1, 0, 0],
                      [0, 0, 1],
                      [1, 0, 1]])
```

The code below will calculate the average of each row.

```py
np.mean(ring_toss, axis=1)
# Output: array([ 0.33333333,  0.33333333,  0.66666667])
```

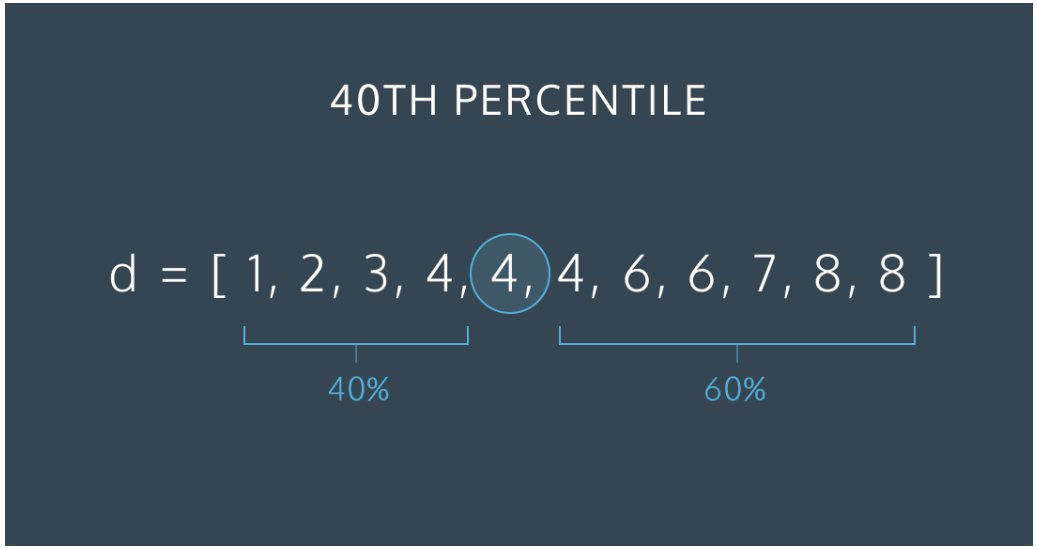
NumPy's Sort Function

In Python, the NumPy `.sort()` function takes a NumPy array and returns a different NumPy array, this one containing the same numbers in ascending order.

```
heights = np.array([49.7, 46.9, 62, 47.2, 47, 48.3, 48.7])
np.sort(heights)
# Output: array([ 46.9,  47. ,  47.2,  48.3,  48.7,  49.7,
62])
```

Definition of Percentile

In statistics, a data set's Nth percentile is the cutoff point demarcating the lower N% of samples.



NumPy Percentile Function

In Python, the NumPy `.percentile` function accepts a NumPy array and percentile value between 0 and 100. The function returns the value of the array element at the percentile specified.

```
d = np.array([1, 2, 3, 4, 4, 4, 6, 6, 7,  8, 8])
np.percentile(d, 40)
# Output: 4.00
```

NumPy's Percentile and Quartiles

In Python, the NumPy `.percentile()` function can calculate the first, second and third quartiles of an array. These three quartiles are simply the values at the 25th, 50th, and 75th percentiles, so those numbers would be the parameters, just as with any other percentile.

```
d = [1, 2, 3, 4, 4, 4, 6, 6, 7, 8, 8]
np.percentile(d, 25)
# Output: 3.5
np.percentile(d, 75)
#Output: 6.5
```

Histogram Visualization

A histogram is a plot that visualizes the distribution of samples in a dataset.
Histogram shows the frequency on the vertical axis and the horizontal axis is another dimension. Usually horizontal axis has bins, where every bin has a minimum and maximum value. Each bin also has a frequency between x and infinite.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(10,10))
plt.hist(x_axis,bins=10,color='blue')
plt.xlabel('Age bins')
plt.ylabel('Frequency')
plt.title('Age Distribution')
plt.show()
```

Datasets and their Histograms

When datasets are plotted as *histograms*, the way the data is distributed determines the distribution type of the data.
The number of peaks in the histogram determines the *modality* of the dataset. It can be *unimodal* (one peak), *bimodal* (two peaks), *multimodal* (more than two peaks) or *uniform* (no peaks).
Unimodal datasets can also be *symmetric*, *skew-left* or *skew-right* depending on where the peak is relative to the rest of the data.

Normal Distribution using Python Numpy module

Normal distribution in NumPy can be created using the below method.

`np.random.normal(loc, scale, size)`

Where `loc` is the mean for the normal distribution, `scale` is the standard deviation of the distribution, and `size` is the number of observations the distribution will have.

```
import numpy as np
mu = 0 #mean
sigma = 0.1 #standard deviation
np.random.normal(mu, sigma, 1000)
```

Standard deviation

The *standard deviation* of a *normal distribution* determines how spread out the data is from the mean.

- 68% of samples will fall between +/- 1 standard deviation of the mean.
- 95% of samples will fall between +/- 2 standard deviations of the mean.
- 99.7% of samples will fall between +/- 3 standard deviations of the mean.