Statistics

Learn how to apply statistical metrics to NumPy data.

Chapter Goals:

- Learn about basic statistical analysis in NumPy
- Write code to obtain statistics for NumPy arrays

A. Analysis

It is often useful to analyze data for its main characteristics and interesting trends. Though we will go more in-depth on data analysis in the section of this course titled Data Preprocessing with scikit-learn, there are still a few techniques in NumPy that allow us to quickly inspect data arrays.

For example, we can obtain minimum and maximum values of a NumPy array using its inherent `min` and `max` functions. This gives us an initial sense of the data's range, and can alert us to extreme outliers in the data.

The code below shows example usages of the `min` and `max` functions.

```
arr = np.array([[0, 72, 3],
                [1, 3, -60],
                [-3, -2, 4]])
print(arr.min())
print(arr.max())

print(repr(arr.min(axis=0)))
print(repr(arr.max(axis=-1)))
```

The `axis` keyword argument is identical to how it was used in `np.argmin` and `np.argmax` from the chapter on Indexing. In our example, we use `axis=0` to find an array of the minimum values in each column of `arr` and `axis=1` to find an array of the maximum values in each row of `arr`.

B. Statistical metrics

NumPy also provides basic statistical functions such as `np.mean`, `np.var`, and `np.median`, to calculate the mean, variance, and median of the data, respectively.

The code below shows how to obtain basic statistics with NumPy. Note that `np.median` applied without `axis` takes the median of the flattened array.

```
arr = np.array([[0, 72, 3],
                [1, 3, -60],
                [-3, -2, 4]])
print(np.mean(arr))
print(np.var(arr))
print(np.median(arr))
print(repr(np.median(arr, axis=-1)))
```

Each of these functions takes in the data array as a required argument and `axis` as a keyword argument. For a more comprehensive list of statistical functions (e.g. calculating percentiles, creating histograms, etc.), check out the NumPy statistics page.

Time to Code!

Each coding exercise in this chapter will be to complete a small function that takes in a 2-D NumPy matrix (`data`) as input. The first function to complete is `get_min_max`, which returns the overall minimum and maximum element in `data`.

Set `overall_min` equal to `data.min` applied with no arguments.

Set `overall_max` equal to `data.max` applied with no arguments.

Return a tuple of `overall_min` and `overall_max`, in that order.

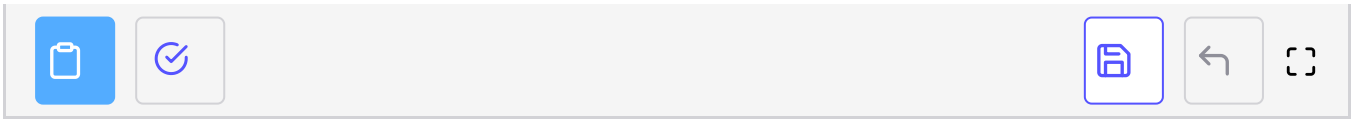
```
def get_min_max(data):
  # CODE HERE
  pass
```

Next, we'll complete `col_min`, which returns the minimums across each column of `data`.

Set `min0` equal to `data.min` with the `axis` keyword argument set to `0`.

Then return `min0`.

```
def col_min(data):
  # CODE HERE
  pass
```



The final function to complete is `basic_stats`, which returns the mean, median, and variance of `data`.

Set `mean` equal to `np.mean` applied to `data`.

Set `median` equal to `np.median` applied to `data`.

Set `var` equal to `np.var` applied to `data`.

Return a tuple containing `mean`, `median`, and `var`, in that order.

```
def basic_stats(data):
  # CODE HERE
  pass
```