Solution Review: Using Numpy and Scipy

This lesson provides the solutions to the previous challenges.

WE'LL COVER THE FOLLOWING

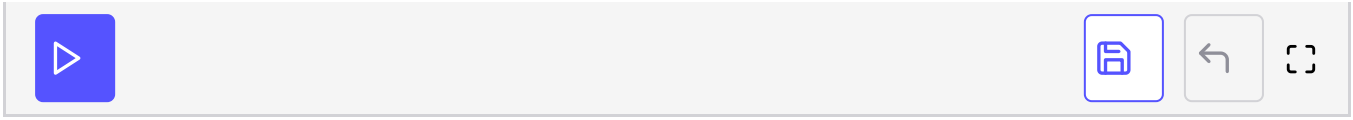
Numpy

```
import numpy as np  # importing numpy module

def perform_calculations(array):

    # returning max, std, sum, and dot product
    return np.max(array), np.std(array), np.sum(array), np.dot(array, array)

# calling the function and printing result
print(perform_calculations(np.random.rand(5)))
```



According to the problem statement, we needed these *four* values using *one* `numpy` 1-D array as an output: **max**, **std**, **sum**, and **dot product**. In the code above, at **line 1** we imported the `numpy` module for this purpose. Next, we implemented the function `perform_calculations()`.

Look at its header at **line 3**. It takes one argument, `array` as an input. At **line 6**, we called *four* built-in functions for required outputs:

- `numpy.max(array)` : Returns the *maximum* value from the `array`
- `numpy.std(array)` : Returns the *standard deviation* among the values of `array`.
- `numpy.sum(array)` : Returns the *sum* of all the values of `array`.

- `numpy.dot(array)` : Returns the *dot product* of `array` with itself.

Now, look at **line 9** where we are calling the function `perform_calculations(array)`. We are generating an array of **five** random values, using `np.random.rand(5)`. It returns an array which is passed to the function `perform_calculations(array)`, and **max**, **std**, **sum**, and **dot product** are printed at the end.

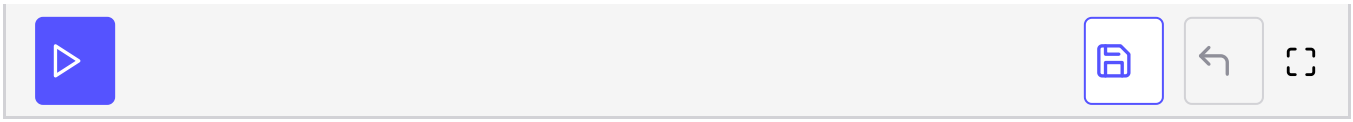
Scipy

```
from scipy import stats  #importing scipy module
import numpy as np       #importing numpy module

def correlation(array1, array2):

  # returning correlation and p-value as a tuple
  return stats.pearsonr(array1, array2)

# calling the function and printing result
print(correlation(np.random.rand(5),np.random.rand(5)))
```



According to the problem statement, we needed these *two* values using *two* `numpy` 1-D array as an output: **correlation**, and **p-value**. In the code above, at **line 1** we imported the `stats` module from `scipy` library for this purpose. Next, we implemented the function `correlation()`.

Look at its header at **line 4**. It takes two argument, `array1` and `array2` as an input. At **line** 7, we called *one* built-in function for the required output:

- `stats.pearsonr(array1, array2)` : Returns the *correlation* and *p-value* value between `array1` and `array2`.

Now, look at **line 10** where we are calling the function `correlation(array1, array2)`. We are generating two arrays each of **five** random values, using `np.random.rand(5)` which are then passed to the function `correlation(array1, array2)`, and **correlation** and **p-value** are printed at the end in the form of a tuple.

That's it about the basics of Python and how to use Python to calculate and gather the main statistics of data. The next chapter explains how to read the

data that is to be analyzed and visualized in the future.