Full Stack Development with AI
Lab 7.1 – NumPy

Lab Overview

In this lab, you will learn how to perform some common numerical computing tasks using the NumPy library. Recall that NumPy provides an efficient multidimensional array object known as the `ndarray`. This object is much faster and more memory-efficient than standard Python built-in data structure such as `list` for numerical operations.

Instead of using a standard Python source file, you are strongly encouraged to use the Jupyter Notebook format. Jupyter Notebook offers several advantages such as cell-based execution and integrated output display. In Notebook, you can run individual cells without rerunning the whole Python script. Notebook makes it easier to perform exploratory coding and incremental development. Notebook also automatically displays output (tables, plots, images, etc.) below the code cell that produced it. This makes it easier to trace the output to the code and facilitates troubleshooting.

You can use the Jupyter editor, either the classic interface or the newer JupyterLab, or an integrated development environment that supports the Notebook format, e.g., Visual Studio Code.

Lab Scenario

This lab simulates the analysis of a typical customer purchase data. Suppose you work at an e-commerce company. You have a dataset of the amounts spent by 5,000 customers during a sale event. You want to quickly analyse this data to understand average spending, variability, identify big spenders, etc.

Recall that `ndarray` is a homogeneous data structure and all values in the object are of the same data type. In this case, all values will be floating-point numbers representing the purchase amounts. Also observe that we are only working with a one-dimensional data structure.

Exercise 1 – Synthetic Data Generation

`np.random` is a NumPy module that provides functions for generating random numbers or performing random sampling. Find out more about the `np.random` and then perform the following tasks:

1. Initialise the seed of the random number generator using the current time.

2. Generate 5,000 customers' purchase amounts in dollars. The data should follow a normal distribution with mean of 100 and standard deviation of 25.

<u>Exercise 2 – Basic Data Analysis</u>

Using the synthetic dataset that you have generated in Exercise 1, perform the following basis data analysis tasks:

1. Compute the average spending amount of customers.

2. Compute the standard deviation of customers' spending amount.

Explain your observations of the results for (1) and (2) with respect to the random number generation parameters that you have specified in Exercise 1.

3. Find high-value customers who spend more than $100. State the number of such customers and the percentage.

Did you observe anything interesting about the results? Note that the threshold to determine high-value customers is actually the average or mean spending amount.

4. Given the large number of high-value customers extracted using the mean spending amount, the results may not be useful for loyalty programmes.

   Instead, it is probably more useful to identify the top 5% spenders. State the number of top 5% spenders and the spending threshold.

Was the number of top 5% spenders within your expectation?

5. Calculate the total revenue contributed by these 5,000 customers from the sale event.

Was it easy to perform the calculation? How would you have performed the calculation <u>without</u> the use of NumPy? Note that NumPy is not just easier but also faster.

<u>Exercise 3 – Advanced Data Analysis</u>

Using the synthetic dataset that you have generated in Exercise 1, perform the following advanced data analysis tasks:

1. It is often useful to identify outliers in our data as they can provide useful insights. In the case of the customer purchase data, we could identify customers with extreme low or high spending amounts for further analysis.

   Sometime, these extreme data value might be caused by error. Other time, they might represent useful business patterns such as customers with very low purchase value that should be ignored or those with very high purchase value that should be accorded special benefits.

   A common heuristic for determining outliers is the 2 standard deviations rule. Any values that are below or above 2 standard deviations from the mean are considered as outliers.

Use the 2 standard deviations rule to identify outliers in the purchase amounts dataset and report the outlier threshold and number of outliers.

Were the outlier thresholds within your expectation? Why?

Compare the number of customers with extreme high purchase amount based on the 2 standard deviations rule with the number of top 5% customers obtained from Exercise 2. Which figure is smaller and which figure is more useful? Why?

2. In more advanced scenarios, you might want the ability to group customers into different spending tiers instead of just identifying the low and high value customers. This insight provides you with greater flexibility to design loyalty programmes that offer incremental and differentiated benefits for different groups of customers commensurate with their purchase amounts.

   Group the customers in the purchase amounts dataset into 5 groups based on the following thresholds:

   - Very Low Spender: 0 < $ <= 50
   - Low Spender: 50 < $ <= 100
   - Average Spender: 100 < $ <= 150
   - High Spender: 150 < $ <= 175
   - Very High Spender: 175 < $

   Report the number of customers in each of the 5 groups.

***Hint**: The `np.digitize` function can be used to assigns each value in an array to a bin (bucket), based on specified bin edges or thresholds. The function essentially tells you which bin each data values falls into. It is useful for grouping continuous numerical data into discrete categories or groups.*

3. Another useful marketing use case that you can perform on the purchase amounts dataset using NumPy is the what-if analysis. For example, you might want to simulate an increase in spending by offering a discount and then determine the increase in revenue.

   Suppose you would like to launch a marketing campaign offering a 10% discount voucher, which you estimate would then encourage people to spend 15% more. Calculate the projected revenue after the campaign and the projected revenue increase or lift.

Observe how easy and convenience it is using NumPy's vectorized operations to perform this task. You do not need to write any loops!

If you are deciding how much advertising budget to spend on this marketing campaign, would knowledge of the projected revenue increase be useful? Why?

-- End of Lab --