**Programming for Data Analytics – Assignment 1 -Report**

Helmi K Abraham (K00302088)

**Introduction**

With this project, we investigated Python's potential for data analysis jobs by contrasting a pure Python code implementation along with more specialised libraries such as Pandas and NumPy. Among the 3 tasks, 1 and 2 were done together and third task is analysed individually. Here This gave us a better understanding of the trade-offs between usability, efficiency, and simplicity in data analysis using python.

**Understanding the Dataset**

Here, for task1 and task2, we worked with a dataset called ‘online\_shoppers\_intention’, which contains a mix of numerical and categorical variables and it required thoughtful handling of different data types. Even though the number of columns is large in number, all looks essential for the dataset. So, we chose all in order to find summary statistics. For numerical data, we calculated descriptive statistics to understand the distribution and spread of the data. Whereas, for categorical data, we looked at the relative frequency of each category to see how common they were in the dataset.

### **Pure Python Implementation**

For task1, we used the csv module to read the CSV file and load the data. Also checked for missing values using function and it seemed to be a clean dataset. We wrote the code to calculate summary statistics using pure Python, meaning we used only the built-in features of Python without relying on external libraries like Pandas or NumPy. This allowed us to manually calculate statistics from scratch. With the help of function, we calculated the mean, median, mode, variance, standard deviation, minimum value, maximum value, range, interquartile range for numerical columns, and for categorical data, we looked at the relative frequency of each category.

Usage of basic Python constructs like loops, lists, and conditionals helped us better understand how these calculations works. The code didn't rely on any external libraries, making it easily transferable to any environment without requiring additional installations.

#### ****Challenges with Pure Python****:

* **Higher execution time**: For larger datasets, this approach became noticeably slower.
* **High complexity**: Writing code for more advanced statistics like calculating IQR or handling missing value manually can lead to more complex code that's harder to maintain.

**Pandas and NumPy Implementation**

In task 2, we used Pandas and NumPy, the two python libraries for data analysis. These simplify many operations and enhance the performance of code. We used Pandas' read\_csv() function to load the dataset directly into a DataFrame.

Instead of manually writing code for each statistic, we used Pandas and NumPy’s built-in functions to calculate the summary statistics. Here, we can use describe() function in order to calculate all the summary statistics with a single line of code. But our aim was to replicate the task 1 activities using NumPy or pandas, that’s why we chose each built-in function separately.

**Strengths of Pandas and NumPy:**

* Efficiency: These libraries speed up the execution time, especially with large datasets.
* SimplerCode: Instead of writing loops and functions to calculate statistics, we could use simple commands.
* EasyHandling: Pandas automatically handles missing values, data type conversion and these type of summary statistics in a single line of code, which makes the process much easier and error free.

**Performance Comparison**

In order to compare the performance, we artificially enlarged the dataset into a larger number of rows, and did the same task1. Even though the pure Python approach worked fine for small datasets as well as large dataset, it was much slower when the dataset grew larger. In contrast, Pandas and NumPy performed much faster. Also, the line of code reduced when we used built-in functions. This showed how important it is to use the right tools for larger datasets to save time and effort.

**Reusability**

In task 3, I took a random dataset called ‘HR\_comma\_sep’, to check the code reusability. It executed well without any errors. Anyway, pure Python requires manual adjustments when applying the code to new datasets, like identifying column types and handling missing values and all. On the other hand, Pandas could automatically handle different datasets without needing many changes to the code. It makes Pandas much more flexible and reusable for different projects.

**Conclusion**

While pure Python is good for learning, libraries like Pandas and NumPy are much better for professional use as it can easily handle large datasets, and it makes the process easier and more readable.