ST. XAVIER'S COLLEGE (AUTONOMOUS), KOLKATA
Department of Information Technology
B. Com Semester VI
Business Data Analytics Practical under NEP

Unit 1: Introduction to Python for Business Data Analytics

B. Python for Data Analysis

Assignment 3: Importing Python Libraries NumPy, SciPy for Business Data Analytics

Topics Covered: NumPy for Numerical Computation, SciPy for Scientific and Technical Computing

This assignment aims to familiarize with key Python libraries essential for business data analytics, including NumPy and SciPy. The tasks will involve data manipulation and statistical analysis on business-related datasets using these libraries.

NumPy

Array Creation: Create NumPy arrays and demonstrate basic array operations (creation, indexing, slicing) on business-related data (e.g., sales figures).

Numerical Operations: Apply NumPy for numerical computations (e.g., mean, median, standard deviation) on business data.

SciPy

Statistical Analysis: Utilize SciPy for statistical analysis (e.g., hypothesis testing, probability distributions) on business data (e.g., sales forecasting).

Integration with NumPy: Demonstrate the integration between SciPy and NumPy for advanced numerical computations.

Students are required to create a folder as FourDigitRollno_Roomno_DeptCode_BDA (For Example, 0012_14_COMA_BDA) on their desktop. All the files related to the assignment should be saved within this folder. Students are required to use either Spyder or Jupyter Notebook to complete the assignment. The final working code should be saved as AssignmentNo_Rollno.py

NumPy for Numerical Computation

1. **Progressive Sales Target Planning and Performance Analysis:** A retail organization defines a progressive monthly sales target plan for its sales team. The targets begin at 1,00,000 INR and increase by 5,000 INR each month until they reach 10,00,000 INR.
    a. Using NumPy's arange() function, generate a one-dimensional array representing these monthly sales targets.
    b. Display the generated array of monthly sales targets.
    c. Calculate and display:
        i. The total cumulative sales target.
        ii. The average monthly sales target.
        iii. The maximum and minimum sales targets.
        iv. All sales targets exceeding 3,00,000 INR.
2. **Multi-Product Weekly Sales Analysis:** A company tracks its weekly sales (in dollars) for five products over three weeks as shown below:
    - Week 1: [1200, 1500, 1000, 1700, 1300]
    - Week 2: [1400, 1600, 1100, 1800, 1250]
    - Week 3: [1350, 1550, 1050, 1750, 1450]

    a. Create a 2D NumPy array from this data.
    b. Display the sales data for Week 2.
    c. Slice and display the sales data for the first two products across all weeks.
    d. Calculate and print the total sales for each product over the three weeks.
3. **Annual Revenue Statistical Analysis:** A company records its monthly revenue figures (in USD) for the previous financial year, represented by the

following dataset:

4500,5200,4800,5800,6100,4900,4700,5300,5700,6500,6200,4900

a. Create a NumPy array from the given revenue data.
b. Calculate and display the following statistical measures:
   i. Mean revenue
   ii. Median revenue
   iii. Standard deviation of the revenue

SciPy for Scientific and Technical Computing

4. **Sparse Sales Data Optimization for Retail Inventory Analysis:** A bookstore records its daily book sales for one week, from Monday to Sunday. Sales occurred only on Tuesday (40 units), Thursday (10 units), and Sunday (2 units), while no books were sold on the remaining days. Using SciPy's csr_matrix (Compressed Sparse Row) representation, store this weekly sales data and display only the non-zero sales values to demonstrate efficient handling of sparse business data.
5. **Supply Chain Optimization through Matrix Inversion:** Compute the Inverse of the Supply-Demand Matrix to give the company the optimal order quantities from each supplier.

$$\begin{matrix} 1 & 1 & 1 \\ 2 & 4 & 5 \\ 3 & 4 & 3 \end{matrix}$$

   Here rows represent the suppliers, columns represent the products and numbers in the matrix represent the quantity of each product required from each supplier. Use SciPy's linalg.inv() function to calculate the inverse of the supply-demand matrix.
6. **Financial Portfolio Risk–Diversification Assessment:** A company is managing a financial portfolio consisting of two types of assets: Asset X and Asset Y. The company wants to analyze the risk and return relationship between these two assets. The risk-return model is represented by a matrix that describes the covariance between the two assets. The company's financial analyst has created the following covariance matrix that represents the relationship between Asset X and Asset Y.

$$\begin{matrix} 9 & 6 \\ 4 & 5 \end{matrix}$$

   Determine the determinant of this covariance matrix, which is a key metric in portfolio optimization. A positive determinant indicates that the portfolio is well-diversified, while a determinant of zero suggests that the portfolio is not diversified, potentially exposing the company to higher risk.

****