Python Data Preprocessing Workshop

Introduction

In this 1-hour workshop, you'll learn how to transition from Excel to Python for data preprocessing and analysis. We'll work with the same SuperStore Sales dataset from the Excel workshop, but implement the same operations using Python, pandas, and seaborn.

Workshop Objectives:

- Load, clean, and assess data using Python's pandas library.
- Create new columns, handle missing values, and transform data efficiently.
- Generate summary tables and visualizations using seaborn and matplotlib.
- Compare Python data analysis techniques with Excel workflows.
- Apply Python to automate and scale data analysis tasks.

Dataset Information

The SuperStore dataset is a retail sales dataset commonly used for data analysis and business intelligence exercises. It contains transaction-level data for a fictional retail store, offering insights into sales, profit, and customer behavior.

Key Features of the Dataset:

- **Order ID:** Unique identifier for each order placed by customers.
- **Product Name:** The name of the product sold in each order.
- **Category:** The category of the product (e.g., Furniture, Office Supplies, Technology).
- **Sales:** The sales revenue generated from the transaction.
- **Profit:** The profit made from the sale of the product.
- **Region:** The region where the order was placed.
- **Customer ID:** Unique identifier for customers.
- **Order Date:** The date when the order was placed.
- **Ship Date:** The date when the order was shipped to the customer.
- **Zip Code:** The postal code for the shipping address.

— Directions

Task 1: Data Loading and Initial Assessment

Exercise 1: Loading Data

1. Use pandas to load the SuperStore dataset
2. Examine the first few rows using head()
3. Check dataset dimensions with shape
4. Identify column names and data types using info()

Exercise 2: Initial Data Assessment

1. Check for missing values using isnull().sum()
2. Identify duplicate rows with duplicated().sum()
3. Fix date columns by converting string dates to datetime objects
4. Examine basic statistics with describe()

Compared to Excel: In Excel, we'd scan for blanks with conditional formatting and use "Remove Duplicates" from the Data tab. Here, we get the same information programmatically.

Task 2: Data Cleaning and Transformation

Exercise 3: Data Cleaning

1. Remove rows with missing values in the Region column
2. Drop duplicate rows
3. Format numeric columns (Sales, Profit) to two decimal places
4. Sort data by Region and Sales (descending)

Exercise 4: Data Transformation

1. Create calculated columns:
    - Profit Margin %: Profit divided by Sales
    - Days to Ship: Difference between Ship Date and Order Date

2. Join the zip_codes.csv file with the main dataset to add city and state information (Python's equivalent to VLOOKUP)
3. Create categorical columns based on numeric values:
    - Margin Category (High/Medium/Low)
    - Shipping Speed (Fast/Medium/Slow)
    - Sales Performance (Above/Below Average)

Compared to Excel: These operations replace Excel formulas, VLOOKUP/XLOOKUP functions, and conditional formatting. The difference is that in Python, we're creating new columns rather than just visual highlighting.

Task 3: Data Analysis and Visualization

Exercise 5: Summary Tables (Pivot Tables in Python)

1. Group data by Region to sum Sales
2. Create more complex groupings with multiple dimensions
3. Apply different aggregation functions to different columns
4. Create a pivot table of sales by Region and Month

Exercise 6: Visualization (Charts in Python)

1. Create a bar chart of sales by region
2. Plot monthly sales trends by region
3. Generate a heatmap of profit margins by region and category
4. Filter data and create visualizations for specific categories or time periods

Compare with Excel: These operations replace Excel's Pivot Tables, Pivot Charts, and filtering functionality. Seaborn and matplotlib give us more customization options than Excel charts.

Tips for Success

- Don't worry about memorizing syntax - focus on the concepts
- Use the provided notebook comments as guides
- Ask questions if you get stuck
- Remember that Python requires precision in syntax (spacing matters!)
- The power of Python becomes more apparent with larger datasets