Laboratory Work 2: Data Exploration and Visualization with Python

Course: Python Data Processing

Student: Bahar Berra Uyar / KH-222ia.e

Instructor: Svitlana Mykolaivna Kovalenko

Date: 03/10/2024

Title: Data Exploration and Visualization with Python

Objective:

The objective of this laboratory work is to develop basic skills in using Python for data exploration and visualization. Specifically, analyze losses of military aircraft during the 2022 Ukraine-Russia war, leveraging libraries like NumPy for numerical computation, Pandas for data handling, and Matplotlib for visualization.

Task 1: Read CSV and Extract Column Values

In this task, I use the Pandas library to read data from a CSV file, which contains cumulative loss counts of various military equipment types (e.g., aircraft, tanks, etc.) during the war. I extract the column corresponding to aircraft losses and return it as a NumPy array. NumPy provides efficient handling and manipulation of large datasets through its array structure.

Code:

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Task 1
def get_column_values(filename, column_name):

    df = pd.read_csv(filename)
    return np.array(df[column_name], dtype=int)

# Example usage:
filename = 'C:\\Users\\Bahar\\Python3\\Lab2\\archive\\russia_losses_equipment.csv'
aircraft_array = get_column_values(filename, "aircraft")  # Correct variable name
```

Technical Details:

- pd.read_csv() reads the CSV file into a DataFrame object, a 2D data structure.
- np.array() converts the aircraft column from the DataFrame into a NumPy array with integer type.

Output:

Displays the extracted aircraft losses data as a NumPy array.

Task 2: Create Numpy Array from the Aircraft Column

In Task 1, I created a NumPy array from the aircraft column, which will be used in subsequent tasks for numerical operations like computing daily losses and analyzing trends. NumPy arrays are well-suited for vectorized operations, which make them more efficient than Python lists for numerical computation.

Output:

```
# Task 2: Create a numpy array from the "aircraft" column.
# The numpy array for "aircraft" column is created using Task 1
print("Aircraft data as numpy array:")
print(aircraft_array)
```

```
Aircraft data as numpy array:
[369 369 369 369 369 369 369 369 369 369 369 369 369 369 369 369 369 369
 368 368 368 368 368 368 368 368 368 368 368 368 368 368 368 367 367 367
 367 367 367 367 367 367 367 367 367 367 366 366 366 366 366 366 366 365
 365 365 363 363 363 363 363 363 363 363 363 363 363 363 362 362 362 362
 361 361 361 361 361 361 361 361 361 361 361 361 360 360 360 360 360 360
 360 360 360 360 359 359 359 359 359 359 359 359 359 359 359 359 359 359
 359 359 359 358 357 357 357 357 357 357 357 357 357 357 357 357 357 357
 357 356 356 355 354 354 354 354 354 353 351 351 351 350 350 349 349 349
 349 349 349 349 348 348 348 348 348 348 348 348 348 348 348 348 348 348
 348 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347
 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347 347
 347 347 347 347 347 347 347 347 347 347 347 347 347 346 345 342 342 340
 340 340 340 339 339 338 338 336 335 332 332 332 332 332 332 332 332 332
 332 332 332 332 332 332 332 332 332 332 331 331 331 331 331 331 331 331
 331 331 331 331 331 331 329 329 329 329 329 329 329 329 329 329 329 329
 329 329 329 329 329 329 329 329 329 329 327 327 324 324 324 324 324 324
 324 324 324 324 324 324 324 324 324 324 324 323 323 323 323 323 323 323
 323 323 323 323 323 323 323 323 323 323 323 323 323 323 322 322 322 322
 322 322 322 322 322 322 322 322 322 321 321 321 320 320 320 320 320 320
 320 320 320 320 320 319 318 318 317 317 316 316 316 315 315 315 315 315
 315 315 315 316 316 315 315 315 315 315 315 315 315 315 315 315 315 315
 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315
 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315
 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315 315
```

Task 3: Find Daily Aircraft Losses

This task involves calculating daily losses of aircraft. Since the CSV data contains cumulative totals, need to compute the difference between successive values to obtain daily losses. NumPy's np.diff() function is ideal for this, as it efficiently computes differences between adjacent elements in an array.

Code:

```
# Task 3: Find daily aircraft losses.
def daily_losses(losses_array):

    return np.diff(losses_array, prepend=losses_array[0])

daily_aircraft_losses = daily_losses(aircraft_array)
print("Daily aircraft losses:")
print(daily_aircraft_losses)
```

Technical Details:

- np.diff() calculates the difference between adjacent elements.
- prepend ensures that the resulting array has the same length as the original by inserting the first value at the beginning.

Output:

```
Daily aircraft losses:
[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  -1   0   0   0   0   0   0   0   0   0   0   0   0   0   0  -1   0   0
   0   0   0   0   0   0   0   0   0   0  -1   0   0   0   0   0   0  -1
   0   0  -2   0   0   0   0   0   0   0   0   0   0   0  -1   0   0   0
  -1   0   0   0   0   0   0   0   0   0   0   0  -1   0   0   0   0   0
   0   0   0   0  -1   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0  -1  -1   0   0   0   0   0   0   0   0   0   0   0   0   0
   0  -1   0  -1  -1   0   0   0   0  -1  -2   0   0  -1   0  -1   0   0
   0   0   0   0  -1   0   0   0   0   0   0   0   0   0   0   0   0   0
   0  -1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0  -1  -1  -3   0  -2
   0   0   0  -1   0  -1   0  -2  -1  -3   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0  -1   0   0   0   0   0   0   0
```

Task 4: Find Top 3 Greatest Daily Losses

I need to find the three largest daily losses of aircraft during the war. To achieve this, I use NumPy's np.sort() function to sort the daily losses in ascending order and extract the last three values, which represent the greatest losses.

Code:

```python
# Task 4: Find the 3 greatest daily losses of aircrafts.
def top_n_losses(losses_array, n=3):
    return np.sort(losses_array)[-n:]

top_3_losses = top_n_losses(daily_aircraft_losses)
print(f"Top 3 daily losses: {top_3_losses}")
```

```
Top 3 daily losses: [0 0 1]
```

Technical Details:

- np.sort() sorts the array in ascending order.
- Slicing ([-n:]) extracts the largest n values from the sorted array.

Task 5: Calculate Aircraft Losses in Summer 2024

To calculate the number of aircraft losses during summer 2024, I use Pandas for date filtering. Convert the date column to a datetime object and then filter the rows that correspond to June 1st, 2024, through August 31st, 2024. The difference between the first and last aircraft values in this period gives the total losses.

Code:

```
# Task 5: Determine how many aircrafts were shot down in the summer of 2024.
def losses_in_summer_2024(filename):

    df = pd.read_csv(filename)
    df['date'] = pd.to_datetime(df['date'])

    # Filter data for summer 2024
    summer_2024 = df[(df['date'] >= '2024-06-01') & (df['date'] <= '2024-08-31')]
    total_losses = summer_2024['aircraft'].iloc[-1] - summer_2024['aircraft'].iloc[0]

    return total_losses

summer_2024_losses = losses_in_summer_2024(filename)
print(f"Aircraft losses in the summer of 2024: {summer_2024_losses}")
```

```
Aircraft losses in the summer of 2024: -11
```

Technical Details:

- pd.to_datetime() converts the date column into a datetime object for easy date filtering.
- .iloc[] allows selecting rows by their integer position.

Task 6: Mean Aircraft Losses in the Last 300 Days

I calculate the mean daily aircraft losses over the last 300 days of the dataset. NumPy's np.mean() function provides a simple way to compute the average of the selected slice from the losses array.

Code:

```
# Task 6: Find the mean value of aircraft losses in the last 300 days of war.
def mean_last_300_days(losses_array):
    return np.mean(losses_array[-300:])

mean_losses_300_days = mean_last_300_days(daily_aircraft_losses)
print(f"Mean aircraft losses in the last 300 days: {mean_losses_300_days}")
```

```
Mean aircraft losses in the last 300 days: -0.91
```

Technical Details:

- np.mean() calculates the average value of the elements in the array slice.

Task 7: Plot Aircraft Losses in the First Year of War

This task requires us to plot the daily aircraft losses during the first year of the war using Matplotlib. Customizations such as gridlines, labels, legends, and line styles are applied to improve the readability of the plot.

Code:

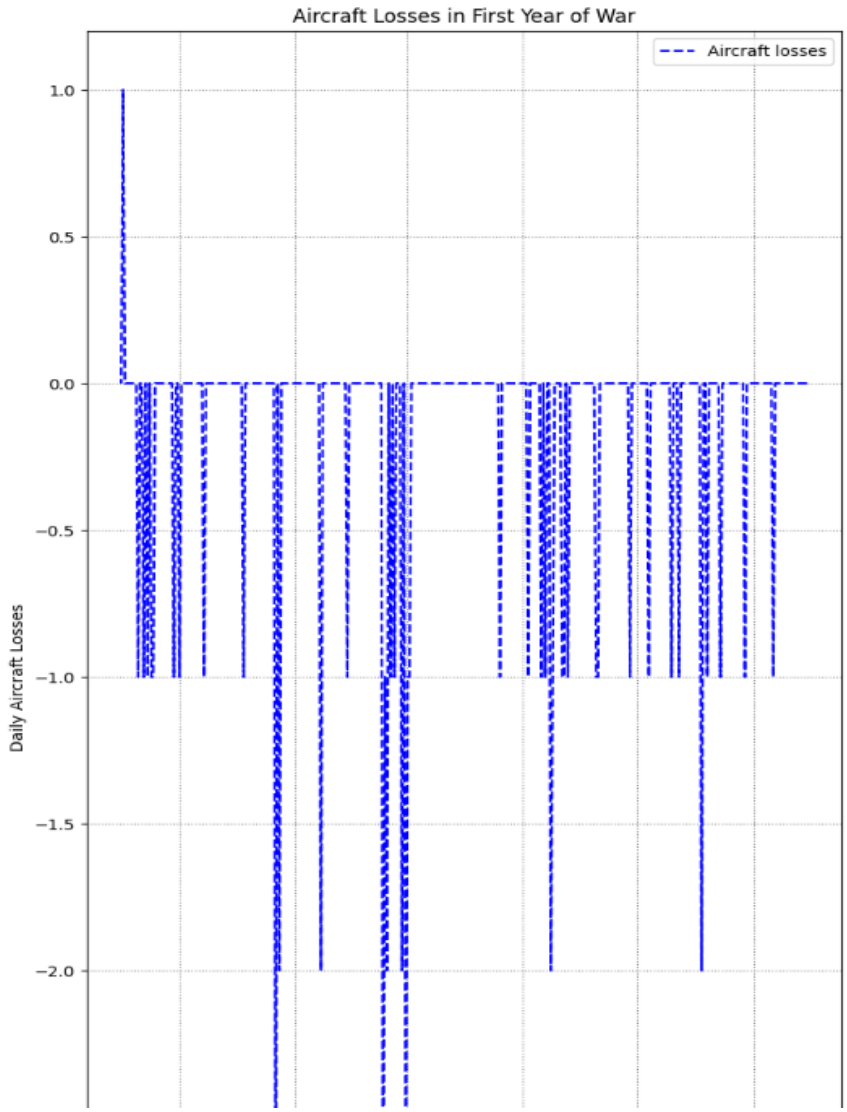
```
# Task 7: Plot aircraft losses for the first year of war.
def plot_aircraft_losses(dates, losses_array):
    plt.figure(figsize=(8, 16), dpi=100)
    plt.plot(dates[:365], losses_array[:365], linestyle='--', color='blue', label='Aircraft losses')

    plt.title("Aircraft Losses in First Year of War")
    plt.xlabel("Date")
    plt.ylabel("Daily Aircraft Losses")

    plt.grid(True, which='both', linestyle=':', color='gray')
    plt.legend()
    plt.xticks(rotation=45)

    # Save the plot
    plt.savefig("aircraft_losses_first_year.png")
    plt.show()

# Example usage for Task 7
df = pd.read_csv(filename)
df['date'] = pd.to_datetime(df['date'])
plot_aircraft_losses(df['date'], daily_aircraft_losses)
```

Technical Details:

- plt.plot() creates the line plot with customized line style.
- plt.grid() adds a grid with specified line style and color.

- plt.legend() adds a legend to the plot.
- plt.savefig() saves the plot as a PNG image file.

***Output**:*



Conclusion:

This report demonstrates the analysis of aircraft losses during the 2022 Ukraine-Russia war using Python. The tasks focus on reading and manipulating data using Pandas and NumPy, and visualizing the results using Matplotlib. Each function is designed to meet specific data exploration and visualization goals, providing insights into the trends and scale of military losses over time.
Github: https://github.com/BaharBerra/Python_lab2.git