МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №8

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Візуалізація та обробка даних за допомогою

спеціалізованих бібліотек Python

Виконала:

ст. гр. РІ-31

Оксана ЛЕСЮК

Прийняв:

Сергій Сергійович ЩЕРБАК

Львів-2024

**Мета роботи:** Розробка додатка для візуалізації CSV-наборів даних за допомогою Matplotlib та базових принципів ООП (наслідування, інкапсуляція, поліморфізм)

**Завдання:**

План роботи

Завдання 1: Вибір CSV-набору даних Оберіть CSV-набір даних, який ви хочете візуалізувати. Переконайтеся, що він містить відповідні дані для створення змістовних візуалізацій.

Завдання 2: Завантаження даних з CSV Напишіть код для завантаження даних з CSV-файлу в ваш додаток Python. Використовуйте бібліотеки, такі як Pandas, для спрощення обробки даних.

Завдання 3: Дослідження даних Визначте екстремальні значення по стовцям

Завдання 4: Вибір типів візуалізацій Визначте, які типи візуалізацій підходять для представлення вибраних наборів даних. Зазвичай це може бути лінійні графіки, стовпчикові діаграми, діаграми розсіювання, гістограми та секторні діаграми.

Завдання 5: Підготовка даних Попередньо обробіть набір даних за необхідністю для візуалізації. Це може включати виправлення даних, фільтрацію, агрегацію або трансформацію.

Завдання 6: Базова візуалізація Створіть базову візуалізацію набору даних, щоб переконатися, що ви можете відображати дані правильно за допомогою Matplotlib. Розпочніть з простої діаграми для візуалізації однієї змінної.

Завдання 7: Розширені візуалізації Реалізуйте більш складні візуалізації, виходячи з характеристик набору. Поекспериментуйте з різними функціями Matplotlib та налаштуваннями.

Завдання 8: Декілька піддіаграм Навчіться створювати кілька піддіаграм в межах одного малюнка для відображення декількох візуалізацій поруч для кращого порівняння.

Завдання 9: Експорт і обмін Реалізуйте функціональність для експорту візуалізацій як зображень (наприклад, PNG, SVG) або інтерактивних веб-додатків (наприклад, HTML)

Виконавши ці завдання, ви створите багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib. Цей проект покращить ваші навички візуалізації даних, дозволяючи досліджувати результати з різноманітними наборами даних

**Хід роботи**

**DataProcessor.py:**

class DataProcessor:

def \_\_init\_\_(self, csv\_file, plot\_dir, data\_dir):

if not os.path.isfile(csv\_file):

raise FileNotFoundError(f"CSV file '{csv\_file}' does not exist.")

self.data = pd.read\_csv(csv\_file)

self.csv\_file\_name = os.path.splitext(os.path.basename(csv\_file))[0]

self.plot\_dir = plot\_dir

self.data\_dir = data\_dir

def process(self):

raise NotImplementedError("Subclasses should implement this!")

def save\_plot(self, plt, filename):

path = os.path.join(self.plot\_dir, f"{filename}.png")

plt.savefig(path, bbox\_inches='tight')

plt.close()

print(f"Saved plot to {path}")

**DataCleaning.py:**

class DataCleaning(DataProcessor):

def process(self):

print("\nData Cleaning:")

if self.data.isnull().values.any():

print("Missing values found. Cleaning data.")

self.data.dropna(inplace=True)

cleaned\_path = os.path.join(self.data\_dir, f"{self.csv\_file\_name}\_cleaned.csv")

self.data.to\_csv(cleaned\_path, index=False)

print(f"Cleaned data saved to {cleaned\_path}")

else:

print("No missing values found. Cleaning not required.")

**DataExploration.py**:

class DataExploration(DataProcessor):

def process(self):

print("\nData Exploration:")

print("First 5 rows:\n", self.data.head())

print("Last 5 rows:\n", self.data.tail())

print("Columns:\n", self.data.columns)

print("Shape:\n", self.data.shape)

self.get\_extreme\_values()

def get\_extreme\_values(self):

print("\nExtreme values of data:")

for col in self.data.select\_dtypes(include=[np.number]).columns:

print(f"{col} - min: {self.data[col].min()}, max: {self.data[col].max()}, median: {self.data[col].median()}")

**DataPipeline.py:**

class DataPipeline:

def \_\_init\_\_(self, csv\_file, plot\_dir, data\_dir):

self.csv\_file = csv\_file

self.plot\_dir = plot\_dir

self.data\_dir = data\_dir

def run\_pipeline(self):

for processor\_type in ["exploration", "cleaning", "visualization"]:

processor = DataProcessorFactory.create\_processor(processor\_type, self.csv\_file, self.plot\_dir, self.data\_dir)

processor.process()

**DataProcessorFactory.py:**

class DataProcessorFactory:

@staticmethod

def create\_processor(processor\_type, \*args):

if processor\_type == "exploration":

return DataExploration(\*args)

elif processor\_type == "cleaning":

return DataCleaning(\*args)

elif processor\_type == "visualization":

return DataVisualization(\*args)

else:

raise ValueError("Unknown processor type")

**DataVisualization.py:**

class DataVisualization(DataProcessor):

def process(self):

print("\nData Visualization:")

self.plot\_pairplot()

self.plot\_heatmap()

def plot\_pairplot(self):

sns.pairplot(self.data)

self.save\_plot(plt, f"{self.csv\_file\_name}\_pairplot")

def plot\_heatmap(self):

numeric\_df = self.data.select\_dtypes(include=[np.number])

if not numeric\_df.empty:

plt.figure(figsize=(10, 8))

sns.heatmap(numeric\_df.corr(), annot=True, cmap='coolwarm', fmt=".2f")

plt.title("Correlation Heatmap")

self.save\_plot(plt, f"{self.csv\_file\_name}\_heatmap")

**Посилання на гіт-репозиторій:**

https://github.com/lesiukoksana/SMP-labs

**Висновок:**

У процесі виконання лабораторної роботи було розроблено програмно-інформаційний продукт для візуалізації CSV-наборів даних за допомогою бібліотеки Matplotlib із застосуванням основних принципів об’єктно-орієнтованого програмування.