МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем та мереж

Лабораторна робота №8

з курсу

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

Виконав студент

групи ІТ-21сп

**Петровський Т.О.**

Прийняв

**Щербак С.С.**

Львів - 2023

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

**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)

from .main import execute\_lab8

import logging

from Lab8.financial\_data\_visualizer import MonthlyFinancialAnalyzer

from utils import UserInputHelper

# Represents financial data visualizer interface

class FinancialDataInterface:

def \_\_init\_\_(self):

logging.info('Initialize financial data interface')

self.\_\_financial\_data\_visualizer = MonthlyFinancialAnalyzer('files/data.csv')

def run(self):

logging.info('Executing financial data visualizer')

while True:

option = UserInputHelper.get\_limited\_user\_input(

"1-Explore data\n2-Show total income\n3-Show expenses\n4-Show profit\n5-Show all plots\n0-exit\n",

['1', '2', '3', '4', '5', '0'])

if option == '1':

self.\_\_financial\_data\_visualizer.explore\_monthly\_financial\_data()

elif option == "2":

self.\_\_financial\_data\_visualizer.plot\_total\_income\_distribution()

elif option == "3":

self.\_\_financial\_data\_visualizer.plot\_expenses\_distribution()

elif option == "4":

self.\_\_financial\_data\_visualizer.plot\_profit\_distribution()

elif option == "5":

self.\_\_financial\_data\_visualizer.show\_all\_monthly\_plots()

else:

break

import logging

import pandas as pd

import matplotlib.pyplot as plt

import plotly.express as px

# Analyze financial data from file

class MonthlyFinancialAnalyzer:

# Constructor

def \_\_init\_\_(self, csv\_file\_path):

logging.info('Initialize finance analyzer')

self.df = pd.read\_csv(csv\_file\_path)

# Explores financial data

def explore\_monthly\_financial\_data(self):

logging.info('Exploring financial data')

min\_values = self.df.min()

max\_values = self.df.max()

print("Minimum values:")

print(min\_values)

print("\nMaximum values:")

print(max\_values)

# Shows total income distribution

def plot\_total\_income\_distribution(self):

logging.info('Showing total income distribution')

total\_income\_distribution = self.df

total\_income\_distribution.plot(kind='pie', y='Total\_Income', autopct='%1.1f%%',

title='Total Income Distribution')

plt.show()

# Shows expenses distribution

def plot\_expenses\_distribution(self):

logging.info('Showing expenses distribution')

expenses\_distribution = self.df

expenses\_distribution.plot(kind='bar', x='Month', y='Expenses',

xlabel='Month', ylabel='Expenses (Dollars)', title='Monthly Expenses Distribution')

plt.show()

# Shows profit distribution

def plot\_profit\_distribution(self):

logging.info('Showing profit distribution')

profit\_distribution = self.df

profit\_distribution.plot(kind='bar', x='Month', y='Profit',

xlabel='Month', ylabel='Profit (Dollars)', title='Monthly Profit Distribution')

plt.show()

# Shows all monthly plots

def show\_all\_monthly\_plots(self):

logging.info('Showing all monthly plot')

fig, axes = plt.subplots(2, 2, figsize=(12, 10))

self.plot\_total\_income\_distribution\_subplot(axes[0, 0])

self.plot\_expenses\_distribution\_subplot(axes[0, 1])

self.plot\_profit\_distribution\_subplot(axes[1, 0])

plt.savefig('files/monthly\_plot.png')

self.plot\_expenses\_distribution\_plotly()

plt.show()

# Sets total income distribution

def plot\_total\_income\_distribution\_subplot(self, ax):

logging.info('Setting total income distribution')

total\_income\_distribution = self.df

ax.pie(total\_income\_distribution['Total\_Income'], autopct='%1.1f%%',

labels=total\_income\_distribution['Month'], startangle=90)

ax.set\_title('Total Income Distribution')

# Sets expenses distribution

def plot\_expenses\_distribution\_subplot(self, ax):

logging.info('Setting expenses distribution')

expenses\_distribution = self.df

ax.bar(expenses\_distribution['Month'], expenses\_distribution['Expenses'])

ax.set\_xlabel('Month')

ax.set\_ylabel('Expenses (Dollars)')

ax.set\_title('Monthly Expenses Distribution')

# Setts profit distribution

def plot\_profit\_distribution\_subplot(self, ax):

logging.info('Setting profit distribution')

profit\_distribution = self.df

ax.bar(profit\_distribution['Month'], profit\_distribution['Profit'])

ax.set\_xlabel('Month')

ax.set\_ylabel('Profit (Dollars)')

ax.set\_title('Monthly Profit Distribution')

# Sets expenses distribution

def plot\_expenses\_distribution\_plotly(self):

logging.info('Setting expenses distribution')

fig = px.bar(self.df, x='Month', y='Expenses', title='Monthly Expenses Distribution')

fig.write\_html('files/monthly\_plot.png')

from Lab8.financial\_data\_interface import FinancialDataInterface

def execute\_lab8():

FinancialDataInterface().run()

if \_\_name\_\_ == '\_\_main\_\_':

execute\_lab8()

**Висновок:** Виконавши ці завдання, я створив багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib. Цей проект покращив мої навички візуалізації даних, дозволяючи досліджувати результати з різноманітними наборами даних