МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ



Лабораторна робота №8

з дисципліни “Спеціалізовані мови програмування”

Виконав:

студент групи ІТ-31

Роман ХРАПЧУН

Прийняв:

доцент кафедри ІСМ

Сергій ЩЕРБАК

Львів-2023

**Тема роботи:** Візуалізація та обробка даних за допомогою спеціалізованих бібліотек Python

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

Хід 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)

**Код програми:**

import pandas as pd

import matplotlib.pyplot as plt

import json

import csv

from prettytable import PrettyTable

import os

class Loader:

"""Клас для завантаження даних з CSV-файлу."""

def \_\_init\_\_(self, file\_path):

"""Ініціалізація об'єкта класу Loader.

Args:

file\_path (str): Шлях до CSV-файлу.

"""

self.file\_path = file\_path

def load\_csv(self):

"""Завантажує дані з CSV-файлу.

Returns:

pd.DataFrame: Завантажений DataFrame.

"""

if not os.path.exists(self.file\_path):

raise FileNotFoundError(f"Файл за шляхом {self.file\_path} не знайдено.")

if os.stat(self.file\_path).st\_size == 0:

raise ValueError(f"Файл за шляхом {self.file\_path} є порожнім.")

try:

dataframe = pd.read\_csv(self.file\_path)

except pd.errors.EmptyDataError:

raise ValueError(f"Файл за шляхом {self.file\_path} не містить даних.")

if dataframe.empty:

raise ValueError(f"Файл за шляхом {self.file\_path} не містить стовпців.")

return dataframe

class Analyzer:

"""Клас для аналізу даних."""

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

"""Ініціалізує об'єкт класу Analyzer.

Args:

dataframe (pd.DataFrame): DataFrame для аналізу.

"""

self.dataframe = dataframe

def find\_extremes(self, column):

"""Знайти максимальне та мінімальне значення в стовпці.

Аргументи:

column (str): Назва стовпця.

Повертає:

tuple: Максимальне та мінімальне значення.

"""

if column not in self.dataframe.columns:

raise ValueError(f"Колонка {column} не існує в DataDrame.")

return self.dataframe[column].max(), self.dataframe[column].min()

class Visualizer:

"""Клас для візуалізації даних."""

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

"""Ініціалізує об'єкт класу Visualizer.

Args:

dataframe (pd.DataFrame): DataFrame для візуалізації.

"""

self.dataframe = dataframe

self.last\_figure = None

def plot\_basic(self, column, title, xlabel, ylabel):

"""Базова візуалізація даних.

Args:

column (str): Назва колонки для візуалізації.

title (str): Заголовок графіку.

xlabel (str): Підпис осі X.

ylabel (str): Підпис осі Y.

"""

if self.last\_figure is not None:

plt.close(self.last\_figure)

self.last\_figure, (ax1, ax2) = plt.subplots(nrows=2, ncols=1, figsize=(10, 8))

ax1.plot(self.dataframe.index, self.dataframe[column])

ax1.set\_title(title)

ax1.set\_xlabel(xlabel)

ax1.set\_ylabel(ylabel)

value\_counts = self.dataframe[column].value\_counts()

ax2.bar(value\_counts.index, value\_counts.values, color='skyblue', edgecolor='black')

ax2.set\_xlabel(xlabel)

ax2.set\_ylabel('Count')

plt.tight\_layout()

plt.show()

def plot\_bar(self, categories, values, title='Bar Chart', xlabel='Category', ylabel='Value'):

"""Створює стовпчату діаграму.

Args:

categories (list): Список категорій.

values (list): Список значень.

title (str, optional): Заголовок графіку. Defaults to 'Bar Chart'.

xlabel (str, optional): Підпис осі X. Defaults to 'Category'.

ylabel (str, optional): Підпис осі Y. Defaults to 'Value'.

"""

self.last\_figure, ax = plt.subplots()

ax.bar(categories, values)

ax.set\_title(title)

ax.set\_xlabel(xlabel)

ax.set\_ylabel(ylabel)

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

def plot\_histogram(self, column, bins=10):

"""Створює гістограму.

Args:

column (str): Назва колонки для гістограми.

bins (int, optional): Кількість бінів. Defaults to 10.

"""

self.last\_figure, ax = plt.subplots()

ax.hist(self.dataframe[column], bins=bins)

plt.show()

def plot\_scatter(self, column\_x, column\_y):

"""Створює діаграму розсіювання.

Args:

column\_x (str): Назва колонки для осі X.

column\_y (str): Назва колонки для осі Y.

"""

self.last\_figure, ax = plt.subplots()

ax.scatter(self.dataframe[column\_x], self.dataframe[column\_y])

plt.show()

def plot\_count\_by\_category(self, category):

"""Створює стовпчату діаграму для підрахунку кількості елементів за категорією.

Args:

category (str): Назва колонки категорій.

"""

counts = self.dataframe[category].value\_counts()

self.plot\_bar(counts.index, counts.values, title='Count of items by category', xlabel='Category', ylabel='Count')

def get\_figure(self):

"""Отримує поточну фігуру.

Returns:

plt.Figure: Об'єкт поточної фігури.

"""

return self.last\_figure

def save\_figure(self, filename):

"""Зберігає поточну фігуру у вказаній папці 'Data'."""

if self.last\_figure:

current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

base\_dir = os.path.dirname(current\_dir)

data\_dir = os.path.join(base\_dir, 'Data')

file\_path = os.path.join(data\_dir, filename)

self.last\_figure.savefig(file\_path)

class Exporter:

def export\_plot(self, figure, filename):

"""Експортує фігуру в файл у вказаній папці 'Data'."""

current\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

base\_dir = os.path.dirname(current\_dir)

data\_dir = os.path.join(base\_dir, 'Data')

file\_path = os.path.join(data\_dir, filename)

figure.savefig(file\_path)

class FileStorage:

@staticmethod

def store\_data(data, file\_format):

"""Зберігає дані у вказаному форматі.

Args:

data (pd.DataFrame): Дані для збереження.

file\_format (str): Формат збереження ('json', 'csv', 'txt').

"""

if file\_format == 'json':

with open('current\_pro\_meta.json', 'w') as file:

data.to\_json(file, orient='records', lines=True)

elif file\_format == 'csv':

data.to\_csv('current\_pro\_meta.csv', index=False)

elif file\_format == 'txt':

with open('current\_pro\_meta.txt', 'w') as file:

file.write(data.to\_string())

def run\_lab8():

file\_path = "current\_pro\_meta.csv"

loader = Loader(file\_path)

try:

dataframe = loader.load\_csv()

except ValueError as e:

print(e)

return

if 'Win Rate' not in dataframe.columns:

print("У даних немає колонки 'Win Rate'. Перевірте CSV-файл.")

return

analyzer = Analyzer(dataframe)

max\_value, min\_value = analyzer.find\_extremes('Win Rate')

print(f"Max Value: {max\_value}\nMin Value: {min\_value}")

visualizer = Visualizer(dataframe)

visualizer.plot\_basic('Win Rate', 'Completion Status', 'Index', 'Completion Status')

exporter = Exporter()

figure = visualizer.get\_figure()

visualizer.save\_figure('plot.png')

exporter.export\_plot(figure, 'plot.png')

file\_storage = FileStorage()

file\_storage.store\_data(dataframe, 'csv')

if \_\_name\_\_ == "\_\_main\_\_":

run\_lab8()

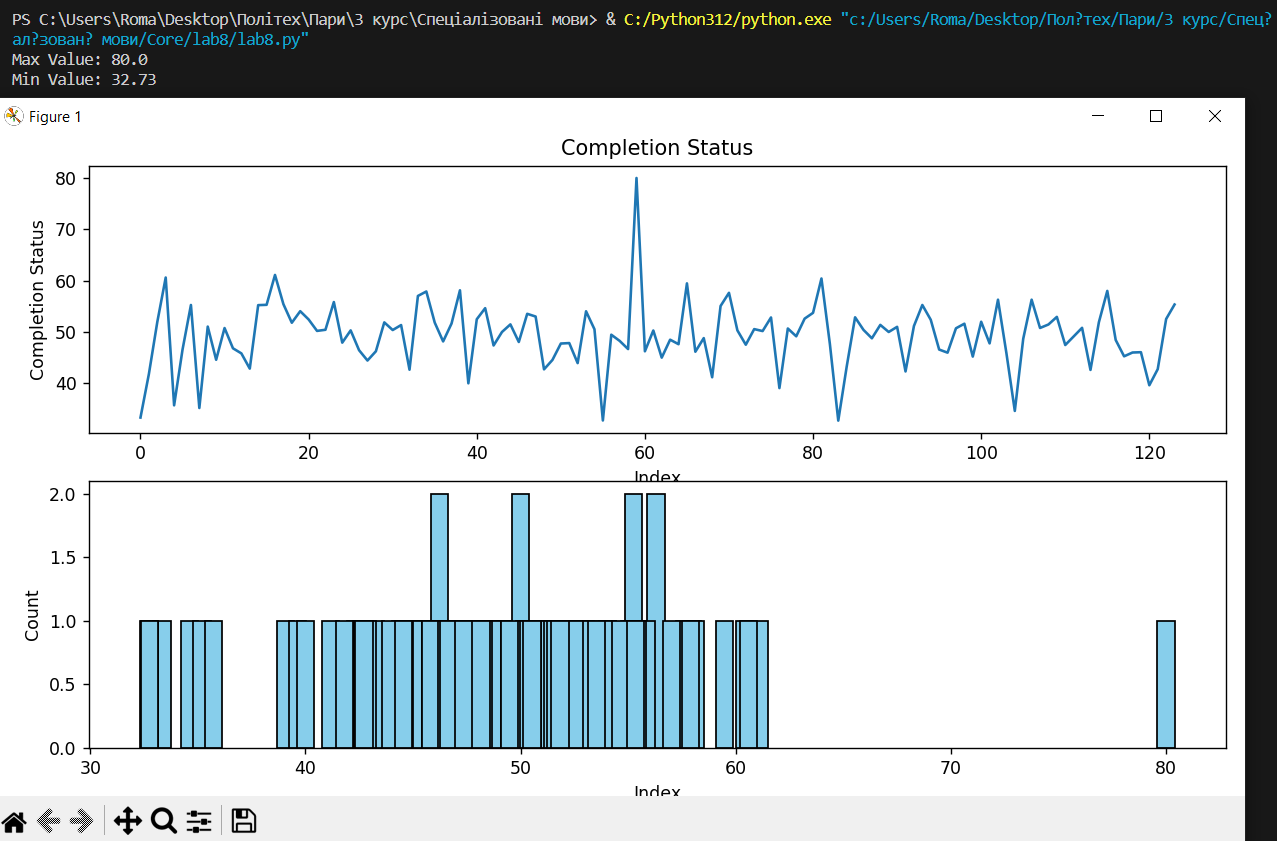


Рис.1 Результат виконання роботи

**Висновок**: Виконуючи лабораторну роботу було створено багатофункціональний додаток для візуалізації CSV-наборів даних за допомогою Matplotlib. Цей проект покращив мої навички візуалізації даних, дозволяючи досліджувати результати з різноманітними наборами даних