КОМП'ЮТЕРНИЙ ПРАКТИКУМ 2.
АНАЛІЗ ТА ВІЗУАЛІЗАЦІЯ ДАНИХ У PYTHON

Мета роботи*:* продемонструвати свої знання про життєвий цикл аналізу даних, використовуючи заданий набір даних та вказані інструменти Python.

Теоретичні відомості

NumPy – це основний пакет для наукових обчислень з Python. Він містить потужний N-вимірний об'єкт масиву та складні (трансляційні) функції.

Pandas – це бібліотека з ліцензією BSD з відкритим кодом, що забезпечує високопродуктивні, прості у використанні структури даних та засоби аналізу даних для мови програмування Python.

Matplotlib – це бібліотека для побудови графіків для мови програмування Python та її числового математичного розширення NumPy.

Folim – це бібліотека для створення інтерактивної карти.

Завдання до комп'ютерного практикуму

Частина 1. Імпорт пакетів Python

Потрібно імпортувати пакети Python: pandas, numpy, matplotlib, folium, datetime та csv, необхідні для аналізу та візуалізації набору даних, що містить інформацію про злочини в Сан-Франциско (файл даних Map-Crime_Incidents-Previous_Three_Months.csv).

```
# Code cell 1
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import folium
```

Частина 2. Завантаження даних

Потрібно завантажити набір даних про злочини в Сан-Франциско (SF County). Завантажте дані про злочини Сан-Франциско у дата фрейм.

Імпортуйте дані про злочини в Сан-Франциско з файлу значень, відокремлених комами (comma-separated values, CSV), у фрейм даних.

```
# code cell 2
# This should be a local path
dataset_path = './Data/Map-Crime_Incidents-Previous_Three_Months.csv'
# read the original dataset (in comma separated values format) into a DataFrame
SF = pd.read_csv(dataset_path)
```

Потрібно використати засоби Python та Jupyter, щоб підготувати ці дані до аналізу, проаналізувати їх, побудувати графіки та повідомити про свої результати. Для перегляду перших п'яти рядків файлу csv використовується команда Linux **head**.

```
# code cell 3
!head -n 5 ./Data/Map-Crime_Incidents-Previous_Three_Months.csv
```

Перегляньте імпортовані дані. Набравши в клітинку ім'я змінної дата фрейму, ви можете структуровано візуалізувати верхні та нижні рядки.

```
# Code cell 4
pd.set_option('display.max_rows', 10) #Visualize 10 rows
SF
```

Використовуйте функцію **columns** для перегляду імені змінних у DataFrame.

```
# Code cell 5
SF.columns
```

Скільки змінних міститься у фреймі даних SF?

За допомогою функції **len** визначте кількість рядків у наборі даних.

```
# Code cell 6
len(SF)
```

Частина 3. Підготовка даних

Тепер, коли ви завантажили дані в робоче середовище, потрібно підготувати дані до аналізу.

а. Витягніть місяць і день із поля Дата.

Lambda – це ключове слово Python для визначення анонімних функцій. Lambda дозволяє вказати функцію в одному рядку коду, не використовуючи def та не визначаючи для неї конкретного імені. Синтаксис lambda виразу: **lambda parameters : expression**. Далі функція lambda

використовується для створення вбудованої функції, яка вибирає лише цифри місяця зі змінної **Date**, і **int** для перетворення рядкового подання у ціле число. Потім функція pandas **apply** використовується для застосування цієї функції до цілого стовпця (apply неявно визначає цикл for і передає один за одним рядки до lambda функції). Таку саму процедуру можна зробити для дня.

```
# Code cell 7
SF['Month'] = SF['Date'].apply(lambda row: int(row[0:2]))
SF['Day'] = SF['Date'].apply(lambda row: int(row[3:5]))
```

Щоб перевірити, чи ці дві змінні були додані до дата фрейму SF, використайте функцію **print**, щоб надрукувати деякі значення з цих стовпців і перевірити **type**, чи ці нові стовпці дійсно містять числові значення.

```
# Code cell 8
print(SF['Month'][0:2])
print(SF['Day'][0:2])
# Code cell 9
print(type(SF['Month'][0]))
```

b. Видаліть змінні з дата фрейму SF.

Стовпець IncidntNum містить багато комірок з NaN. У цьому випадку дані відсутні. Стовпець можна вилучити з дата фрейму. Одним із способів видалення небажаних змінних у фреймі даних є використання функції **del**.

```
# Code cell 10
del SF['IncidntNum']
```

Аналогічно, атрибут Location не буде використовуватися в цьому аналізі. Його можна викинути з дата фрейму. Можна використовувати функцію **drop** для дата фрейму, вказавши, що *вісь* дорівнює 1 (0 для рядків), і що команда не вимагає присвоєння іншому значенню для збереження результату (*inplace* = *True*).

```
# Code cell 11
SF.drop('Location', axis=1, inplace=True )
```

Переконайтеся, що стовпці видалено.

```
# Code cell 12
SF.columns
```

Частина 4. Аналіз даних

Тепер, коли дата фрейм підготовлений з даними, настав час проаналізувати дані.

a. Узагальніть змінні для отримання статистичної інформації.

Використовуйте функцію **value_counts** для підсумовування кількості злочинів, скоєних за типом, а потім **print** для відображення вмісту змінної *CountCategory*.

```
# Code cell 13
CountCategory = SF['Category'].value_counts()
print(CountCategory)
```

За замовчуванням підрахунок впорядковується за спаданням. Значення необов'язкового параметра за *зростанням* можна встановити на *True*, щоб змінити цю поведінку.

```
# Code cell 14
SF['Category'].value_counts(ascending=True)
```

Якого виду злочину було скоєно найбільше?

Вклавши дві функції в одну команду, ви можете досягти одного результату за допомогою одного рядка коду.

```
# Code cell 15
print(SF['Category'].value_counts(ascending=True))
```

У якому PdDistrict було найбільше випадків зареєстрованих злочинів?

Надайте команди Python для підтримки вашої відповіді.

```
# code cell 16
# Possible code for the challenge question
print(SF['PdDistrict'].value_counts(ascending=True))
```

b. Підгрупуйте дані у менші дата фрейми (кадри даних).

За допомогою логічного індексування можна вибрати лише ті рядки, для яких виконується дана умова. Витягніть лише ті злочини, скоєні в серпні, і збережіть результат у новому DataFrame.

```
# Code cell 17
AugustCrimes = SF[SF['Month'] == 8]
AugustCrimes
```

Скільки випадків злочинів було за серпень?

Скільки квартирних крадіжок було зареєстровано у серпні?

```
# code cell 18
# Possible code for the question: How many burglaries were reported in the
month  of August?
AugustCrimes = SF[SF['Month'] == 8]
AugustCrimesB = SF[SF['Category'] == 'BURGLARY']
len(AugustCrimesB)
```

Щоб створити підмножину кадру даних SF для певного дня, використайте операнд **query** функції для порівняння місяця та дня одночасно.

```
# Code cell 19
Crime0704 = SF.query('Month == 7 and Day == 4')
Crime0704
# Code cell 20
SF.columns
```

Частина 5. Представлення даних

Візуалізація та подання даних забезпечує миттєвий огляд, який може бути не очевидним, просто переглянувши вихідні дані. Дата фрейм SF містить координати довготи та широти, які можна використовувати для аналізу даних.

а. Побудуйте графік дата фрейму SF, використовуючи змінні X та Y.

Використайте функцію **plot()** для побудови кадру даних SF. Використовуйте необов'язковий параметр, щоб побудувати графік червоним кольором і встановити фігуру маркера в коло за допомогою *ro*.

```
# Code cell 21
plt.plot(SF['X'],SF['Y'], 'ro')
plt.show()
```

Визначте номери відділів поліції, складіть словник *pd_districts*, щоб зв'язати їх рядок із цілим числом.

```
# Code cell 22
pd_districts = np.unique(SF['PdDistrict'])
pd_districts_levels = dict(zip(pd_districts, range(len(pd_districts))))
pd_districts_levels
```

Використайте **apply** та **lambda**, щоб додати ціле число для поліцейського відділу до нового стовпця DataFrame.

```
# Code cell 23
SF['PdDistrictCode'] = SF['PdDistrict'].apply(lambda row:
pd_districts_levels[row])
```

Використайте щойно створений *PdDistrictCode* для автоматичної зміни кольору.

```
# Code cell 24
plt.scatter(SF['X'], SF['Y'], c=SF['PdDistrictCode'])
plt.show()
```

b. Додайте пакети для побудови карт, щоб покращити сюжет.

Ви створили простий сюжет, який показує, де мали місце злочини у окрузі SF. Цей графік корисний, але **folium** надає додаткові функції, які дозволять вам накласти цей графік на карту OpenStreet.

Folium вимагає вказувати колір маркера з використанням шістнадцяткового значення. З цієї причини використайте пакет *colors* і виберіть необхідні кольори.

```
# Code cell 25
from matplotlib import colors
districts = np.unique(SF['PdDistrict'])
print(list(colors.cnames.values())[0:len(districts)])
```

Створіть словник кольорів для кожного окружного відділу поліції.

```
# Code cell 26
color_dict = dict(zip(districts, list(colors.cnames.values())[0:-
1:len(districts)]))
color_dict
```

Створіть карту, використовуючи середні координати даних SF, щоб відцентрувати карту (за допомогою **mean**).

Щоб зменшити час обчислення, використайте *plotEvery* для обмеження кількості побудованих даних. Встановіть це значення в 1 для побудови всіх рядків (візуалізація карти може зайняти багато часу).

```
# Code cell 27
# Create map
map_osm = folium.Map(location=[SF['Y'].mean(), SF['X'].mean()], zoom_start =
12)
plotEvery = 50
obs = list(zip( SF['Y'], SF['X'], SF['PdDistrict']))
```

```
for el in obs[0:-1:plotEvery]:
        folium.CircleMarker(el[0:2], color=color_dict[el[2]],
fill_color=el[2],radius=10).add_to(map_osm)
```

```
# Code cell 28
map_osm
```

Вимоги до оформлення звіту

Звіт має включати:

1. Титульний аркуш.
2. Завдання на комп'ютерний практикум.
3. Хід роботи. Цей розділ складається з послідовного опису виконуваних кроків згідно інструкцій до комп'ютерного практикуму.
4. Висновки.

Питання для самоперевірки

1. Які пакети Python необхідні для аналізу та візуалізації набору даних?
2. Як відбувається завантаження даних у Python?
3. Яка команда використовується для перегляду перших п'яти рядків файлу?
4. Для чого використовуються пакети Python: pandas, numpy, matplotlib, folium, datetime та csv?
5. Які засоби Python використовуються для візуалізації даних?

Рекомендована література

1. IoT Fundamentals: Big Data & Analytics // Електронний ресурс. Режим доступу: https://www.netacad.com/courses/iot/big-data-analytics
2. Pandas // Електронний ресурс. Режим доступу: https://pandas.pydata.org/
3. NumPy // Електронний ресурс. Режим доступу: https://numpy.org/
4. Matplotlib // Електронний ресурс. Режим доступу: https://matplotlib.org/
5. Guide to Getting Started with Geospatial Analysis using Folium // Електронний ресурс. Режим доступу: https://www.analyticsvidhya.com/blog/2020/06/guide-geospatial-analysis-folium-python/