מערך שיעור: עבודה עם גוגל מפות בפייתון

קיימות בפייתון שתי ספריות המאפשרות עבודה עם מפות: **folium, gmplot**.

הבדלים בין folium ו-gmplot

1. מקור ובסיס:
   - folium: מבוסס על ספריית JavaScript בשם Leaflet.js
   - gmplot: משתמש ישירות ב-Google Maps JavaScript API

2. סוג המפות:
   - folium: יכול להשתמש במגוון מקורות מפות, כולל OpenStreetMap, Mapbox, ו-Stamen
   - gmplot: משתמש אך ורק במפות של Google

3. תלות ב-API:
   - folium: לא דורש מפתח API של Google (אלא אם כן משתמשים בשכבות של Google)
   - gmplot: דורש מפתח API של Google Maps

4. גמישות ואינטראקטיביות:
   - folium: מאפשר יצירת מפות אינטראקטיביות עם אפשרויות רבות כמו שכבות, בקרי זום, ועוד
   - gmplot: פחות גמיש, אך מספק גישה ישירה לתכונות ספציפיות של Google Maps

5. פלט:
   - folium: יוצר קבצי HTML עצמאיים שניתן לצפות בהם בכל דפדפן
   - gmplot: יוצר קבצי HTML שדורשים חיבור לאינטרנט לטעינת Google Maps API

6. עקומת למידה:
   - folium: נחשב לידידותי יותר למתחילים, עם API אינטואיטיבי יותר
   - gmplot: עשוי להיות מאתגר יותר ללמידה, אך מספק גישה ישירה לתכונות של Google Maps

7. ביצועים:
   - folium: יכול להיות מהיר יותר לטעינה כאשר משתמשים במקורות מפות פתוחים
   - gmplot: תלוי בזמני הטעינה של Google Maps API

8. תמיכה בתכונות מתקדמות:
   - folium: תומך במגוון רחב של תכונות מתקדמות כמו שכבות גיאו-JSON, מפות חום, וכו'
   - gmplot: מספק גישה לתכונות ספציפיות של Google Maps כמו מסלולים, אזורים גיאוגרפיים, וכו'

<u>לסיכום</u>:
- השתמשו ב-folium אם מחפשים גמישות, אינטראקטיביות, ואין צורך בתכונות ספציפיות של Google Maps.
- השתמשו ב-gmplot אם זקוקים לגישה ישירה לתכונות של Google Maps ומוכנים להשתמש במפתח API.

בשיעור זה, נלמד לעבוד עם שתי הספריות בצורה בסיסית.

דרישות טכניות מקדימות

- התקנה של פייתון 3.x
- רכישת API Key של גוגל מפות (נדרש כרטיס אשראי, אבל לא ייבה תשלום עבור 1000 לחיצות ראשונות).
- התקנה של ספריות folium, gmplot

עבודה עם ספריית folium

יצירת מפה בסיסית

```
import folium

# create map
map = folium.Map(location=[31.7767, 35.2345], zoom_start=7)

# save map as html file
map.save("my_map.html")
```

הוספת סמן על המפה

```
folium.Marker (
   location=[32.0853, 34.7818],
   popup="תל אביב",
   tooltip="לחץ כאן").add_to(map)

map.save("tel_aviv_map.html")
```

עבודה עם ספריית gmplot

יצירת מפה בסיסית

```
import gmplot

# Create a map object centered around a specific location
gmap = gmplot.GoogleMapPlotter(37.7749, -122.4194, 13)

# Draw the map
gmap.draw("my_map.html")
```

הוספת סמן על המפה

```
gmap.marker(37.7749, -122.4194, title="San Francisco")

# Draw the map
gmap.draw("my_map_with_markers.html")
```

הוספת שכבות מידע על המפות – Geocoding

גיאוקודינג הוא תהליך המרת תיאור טקסטואלי של מיקום, כגון כתובת או שם של מקום, לקואורדינטות גיאוגרפיות (בדרך כלל קו רוחב וקו אורך). זהו כלי חיוני במערכות מידע גיאוגרפיות (GIS) ובאפליקציות מבוססות מיקום.

כדי לעבוד עם גיאוקודינג, יש להתקין את הספרייה **googlemaps**. עבודה עם ספרייה זו מחייבת API Key של גוגל מפות (כדי לרכוש API Key, קראו את הקובץ שמסביר זאת).

נקודות עיקריות על גיאוקודינג:

1. **<u>סוגי גיאוקודינג</u>**:
   - גיאוקודינג ישיר: המרת כתובת לקואורדינטות
   - גיאוקודינג הפוך: המרת קואורדינטות לכתובת או תיאור מקום

2. **<u>שימושים נפוצים</u>**:
   - מיפוי נתונים
   - ניתוח מרחבי
   - ניווט ומציאת מסלולים
   - אופטימיזציה של שרשרת אספקה

- ניתוח דמוגרפי

3. **אתגרים בגיאוקודינג**:
- דיוק: לעתים קרובות תלוי באיכות הנתונים ובדיוק של מאגר המידע הגיאוגרפי
- אי-בהירות: כתובות או שמות מקומות יכולים להיות לא חד-משמעיים
- עדכניות: מידע גיאוגרפי עשוי להשתנות עם הזמן

4. **שירותי גיאוקודינג**:
- Google Maps Geocoding API
- OpenStreetMap Nominatim
- Bing Maps API
- HERE Geocoding API

5. דוגמת קוד מורחבת לגיאוקודינג בפייתון עם Google Maps API:

```
import googlemaps
from datetime import datetime

gmaps = googlemaps.Client(key='YOUR_API_KEY')

# גיאוקודינג ישיר
geocode_result = gmaps.geocode('1600 Amphitheatre Parkway, Mountain View, CA')

if geocode_result:
    location = geocode_result[0]['geometry']['location']
    print(f"קואורדינטות: {location['lat']}, {location['lng']}")

# גיאוקודינג הפוך
    reverse_geocode_result = gmaps.reverse_geocode((location['lat'], location['lng']))

    if reverse_geocode_result:
        address = reverse_geocode_result[0]['formatted_address']
        print(f"כתובת: {address}")
else:
    print("לא נמצאו תוצאות")

# חיפוש מקומות בקרבת מקום
places_result = gmaps.places_nearby(location=location, radius=1000, type='restaurant')
if places_result['results']:
    print("\n מסעדות בקרבת מקום:")
    for place in places_result['results'][:5]:  # הצגת 5 התוצאות הראשונות
        print(f"- {place['name']}")
```

6. **יישומים מתקדמים**:
- קלסטרינג: קיבוץ נקודות קרובות על המפה
- ניתוח צפיפות: יצירת מפות חום
- חישוב מרחקים ומסלולים

7. **שיקולים אתיים**:
- פרטיות: גיאוקודינג יכול לחשוף מידע אישי
- דיוק ואחריות: שימוש במידע לא מדויק עלול להוביל להחלטות שגויות

8. **מגמות עתידיות**:
- שילוב עם בינה מלאכותית לשיפור הדיוק
- גיאוקודינג בזמן אמת לאפליקציות IoT
- שילוב עם נתוני חישה מרחוק