05

Construindo camadas em um mapa

A classe `folium.GeoJson()` cria um objeto no formato GeoJSON para ser *plotado* como camada em um mapa.

Uma funcionalidade que não cobrimos em nossas últimas aulas é a possibilidade de formatarmos estas camadas. O parâmetro `style_function`, da classe `GeoJson()`, permite criar uma formatação para as camadas a partir de um dicionário ou de uma função *lambda*, que preencha este dicionário de acordo com determinados critérios. Utilizando um trecho do código implementado na última aula, segue um exemplo da utilização desta funcionalidade:

```
for i in range(len(bairro)):
    geo = folium.GeoJson(
        bairro[i:i+1],
        name=bairro['NM_BAIRRO'][i],
        style_function=lambda feature: {
            'fillColor': 'green' if feature['properties']['V002'] > 50000 else 'yellow',
            'color': 'black',
            'weight': 1
        }
    )
    label = '{} - {} habitantes'.format(bairro['NM_BAIRRO'][i], bairro['V002'][i])
    folium.Popup(label).add_to(geo)
    geo.add_to(base)
```

A parte do código destacada mostra a utilização do parâmetro de formatação, utilizando uma função *lambda* que atribui a cor verde na camada, quando a população daquela área for maior que 50.000, e a cor amarela, quando for menor ou igual a 50.000. Também são configuradas a cor (`'color': 'black'`) e a largura (`'weight': 1`) das bordas de cada camada (mais opções podem ser encontradas aqui (https://leafletjs.com/reference-1.3.4.html#path).

Dadas as informações acima, e considerando o código gerado em nossa último vídeo, assinale a opção que configura o parâmetro `style_function`, seguindo os seguintes critérios:

1. Preenchimento da camada em verde para os bairros que apresentam população acima ou igual à média do município;
2. Preenchimento da camada em amarelo para os demais municípios;
3. Opacidade do preenchimento da camada em 30%;
4. Cor das bordas de cada camada seguindo o mesmo critério dos itens 1 e 2;
5. Largura de todas as bordas de 1 pixel;
6. Opacidade das bordas de todas as camadas em 10%.

Selecione uma alternativa

A

```
style_function=lambda feature: {
    'fillColor': 'green' if feature['V002'] >= bairro['V002'].mean() else 'yellow',
    'fillOpacity': 0.3,
    'color': 'green' if feature['V002'] >= bairro['V002'].mean() else 'yellow',
```

```
    'weight': 1,
    'opacity': 0.1
}
```

B

```
style_function=lambda feature: {
    'fillColor': 'green' if feature['properties']['V002'] >= bairro['V002'].mean() el
    'fillOpacity': 0.3,
    'color': 'green' if feature['properties']['V002'] >= bairro['V002'].mean() else '
    'weight': 1,
    'opacity': 0.1
}
```

C

```
style_function=lambda feature: {
    'fillColor': 'green' if feature['properties']['V002'] >= bairro['V002'].mean() el
    'fillOpacity': 30%,
    'color': 'green' if feature['properties']['V002'] >= bairro['V002'].mean() else '
    'weight': 1,
    'opacity': 30%
}
```

D

```
style_function=lambda feature: {
    'fillColor': 'green' if feature['properties']['V002'] >= mean(bairro['V002']) els
    'fillOpacity': 0.3,
    'color': 'green' if feature['properties']['V002'] >= mean(bairro['V002']) else 'y
    'weight': 1,
    'opacity': 0.1
}
```