Tutorial da videoaula 12 - Semana 5: Visualização de dados com Python

Parte 1: Plotly

Vamos criar um gráfico de dispersão 3D com dados sobre avaliações e características de vinhos. Para isso, vamos usar as bibliotecas Plotly, para a criação do gráfico de dispersão dentro Google Colab, e Pandas. Sobre o Google Colab, recomendamos que, se necessário, reveja a videoaula Jupyter Notebook e Colab Google, videoaula 4 do curso COM350 - Introdução à Ciência de Dados (https://youtu.be/ZC8bfSZLl80) ou acesse a ferramenta no site https://colab.research.google.com/. Caso não tenha uma conta Google ou não queira usar, pode fazer também no Jupyter Notebook. A base de dados com as avaliações de vinhos é frequentemente utilizada em mineração de dados.

Qualidade de vinhos tintos

URL original do conjunto de dados

https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv

Fonte: UCI Machine Learning Repository, Centro para Aprendizado de Máquina e Sistemas Inteligentes, Universidade da Califórnia, Irvine.

Descrição dos atributos da base de dados:

- **fixed acidity** (acidez fixa)
- **volatile acidity** (acidez volátil)
- **citric acid** (acidez cítrica)
- **residual sugar** (açúcar residual)
- **chlorides** (cloretos)
- **free sulfur dioxide** (dióxido de enxofre livre)
- **total sulfur dioxide** (dióxido de enxofre total)
- **density** (densidade)
- **pH**
- **sulphates** (sulfatos)
- **alcohol** (álcool) Atributo classe:
- **quality** (score between 0 and 10) (qualidade - pontuação de 0 a 10)

1. Nesta atividade, usaremos as bibliotecas **plotly** (módulo **express**) e **pandas**. Importe as bibliotecas.

```
import plotly.express as px
import pandas as pd
```

```
from google.colab import files
```

2. Importe a base de dados direto da URL e verifique as primeiras linhas. O arquivo contém 4898 registros.

```
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv'
vinhos = pd.read_csv(url, sep=';')
vinhos.head(10)
```

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

3. Vamos ver a distribuição das notas.

```
notas = px.histogram(vinhos,x='quality')

notas.show()
```

4. As notas variam de 3 a 8, com grandes concentrações em notas médias. Vamos diminuir o número de classes para três, cosiderando notas baixas (menores que 5), médias (5 a 6) e altas (maiores ou iguais a 7). Assim, podemos ver melhor as diferenças entre os atributos.

```
vinhos['quality'] = vinhos['quality'].map({3:'baixa',4:'baixa',5:'media',6:'media',7:'alta',8:'alta'})

vinhos.head()
```

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	media
7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	media
7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	media
11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	media
7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	media

5. Vamos escolher três atributos para visualizá-los no gráfico de dispersão (**sulphates**, **volatile acidity**, **alcohol**). O atributo-alvo (classe) será usado para diferenciar os objetos na visualização, tanto em cor quanto em forma, através dos parâmetros **color** e **symbol**, associados ao atributo de classe (**quality**).

```
dispersao = px.scatter_3d(vinhos, x='sulphates', y='volatile acidity', z='alcohol', color='quality', symbol='quality')

dispersao.show()
```

O resultado mostra que há uma separação clara entre os vinhos avaliados como de alta e baixa qualidade. Já os vinhos de média qualidade estão bem misturados entre os de alta e baixa qualidade considerando os atributos usados.

Experimente ver os resultados usando outros atributos do conjunto de dados.

Versões das bibliotecas

Este tutorial foi feito usando as seguintes versões de bibliotecas:

```
plotly==5.5.0
pandas==1.3.5
```

Parte 2: Folium

Agora, vamos criar um mapa coroplético com a densidade demográfica dos municípios do estado de São Paulo. Para isso, vamos usar a biblioteca Folium para a criação do mapa dentro Google Colab. Sobre o Google Colab, recomendamos que, se necessário, reveja a videoaula Jupyter Notebook e Colab Google, videoaula 4 do curso COM350 - Introdução à Ciência de Dados (https://youtu.be/ZC8bfSZLl80) ou acesse a ferramenta no site https://colab.research.google.com/. Caso não tenha uma conta Google ou não queira usar, pode fazer também no Jupyter Notebook.

A base de dados com as informações das áreas dos municípios e suas densidades demográficas de 2021 está disponível no site DataGEO (https://datageo.ambiente.sp.gov.br/app/#). Após descompactar, faça o upload dos arquivos da pasta criada (**DensidadeDemografica2021**) no Colab para usá-lo.

A URL do arquivo é:
https://datageo.ambiente.sp.gov.br/geoserver/datageo/DensidadeDemografica2021/wfs?version=1.0.0&request=GetFeature&outputFormat=SHAPE-ZIP&typeName=DensidadeDemografica2021

Descrição dos atributos da base de dados:

- **Codigo**: código do município
- **Nome**: nome do município
- **Area_km**: área do município em quilômetros quadrados
- **Densidadem**: densidade demográfica
- **Populacao**: quantidade de habitantes do município em 2021
- **geometry**: polígono com as coordenadas da área do município

1. Crie um novo notebook e inclua uma descrição para ele. Nesta atividade, usaremos as bibliotecas **folium**, **google.colab** e **geopandas.** É necessário instalar o geopandas. A seguir, importe as bibliotecas.

```
pip install -U geopandas

import folium

import geopandas as gpd

from google.colab import files
```

2. Faça upload dos arquivos da pasta descompactada no Google Colab.

```
arquivos = files.upload()
```

3. Carregue os arquivos m um DataFrame.

```
municipios = gpd.read_file('DensidadeDemografica2021Polygon.shp')
municipios.head()
```

	Codigo	Nome	Area_km	Densidadem	Populacao	geometry
0	3550308	São Paulo	1521.10	7833.00	11914851.0	POLYGON ((-46.63082 -23.6...
1	3500105	Adamantina	411.39	82.21	33869.0	POLYGON ((-51.17735 -21.6...
2	3500204	Adolfo	211.08	16.28	3436.0	POLYGON ((-49.74715 -21.29...
3	3500303	Aguaí	474.74	75.62	35885.0	POLYGON ((-47.23298 -22.05...
4	3500501	Águas de Lindóia	60.13	306.64	18438.0	POLYGON ((-46.66020 -22.4...

4. Vamos ver a distribuição dos dados.

```
municipios.describe()
```

	Codigo	Area_km	Densidadem	Populacao
count	6.450000e+02	645.000000	645.000000	6.450000e+02
mean	3.528698e+06	384.842186	334.361504	6.960141e+04
std	1.670033e+04	319.979242	1305.418065	4.820846e+05
min	3.500105e+06	5.540000	3.840000	8.110000e+02
25%	3.514601e+06	157.900000	20.950000	5.561000e+03
50%	3.528700e+06	280.670000	40.580000	1.350300e+04
75%	3.543204e+06	511.620000	122.900000	4.140500e+04
max	3.557303e+06	1977.950000	14083.130000	1.191485e+07

5. A média do atributo de densidade demográfica é muito inferior em relação aos municípios com maior densidade. Nesse caso, o mapa vai ficar praticamente todo com a cor de baixa densidade e apenas os poucos municípios com alta densidade terão cor. Para ver melhor as diferenças entre os municípios com baixa densidade em relação

àqueles com densidades médias, vamos mudar a escala padrão, retornando uma lista com subdivisões da escala para distinguir melhor os valores de densidade distintos.

```
escala = (municipios['Densidadem'].quantile((0,0.5,0.75,0.8,0.9,0.95,1.0))).tolist()
```

```
[3.84, 40.58, 122.9, 160.84, 420.98, 1154.31, 14083.13]
```

6. Em seguida, vamos gerar o mapa a partir de coordenadas do centro do estado (**SP_LAT, SP_LON**). O objeto **folium.Choropleth** recebe os dados de geolocalização com as áreas dos municípios e os valores de densidade demográfica. Incluímos também as colunas que serão usadas e exibidas quando passar o mouse sobre o mapa (**Nome e Densidadem**), o nome da propriedade chave (**key_on**), a escala de cores (**YlOrRd** - amarelo-laranja-vermelho, crescendo conforme o valor de densidade). Em seguida, passamos a escala modificada (**threshold_scale**).

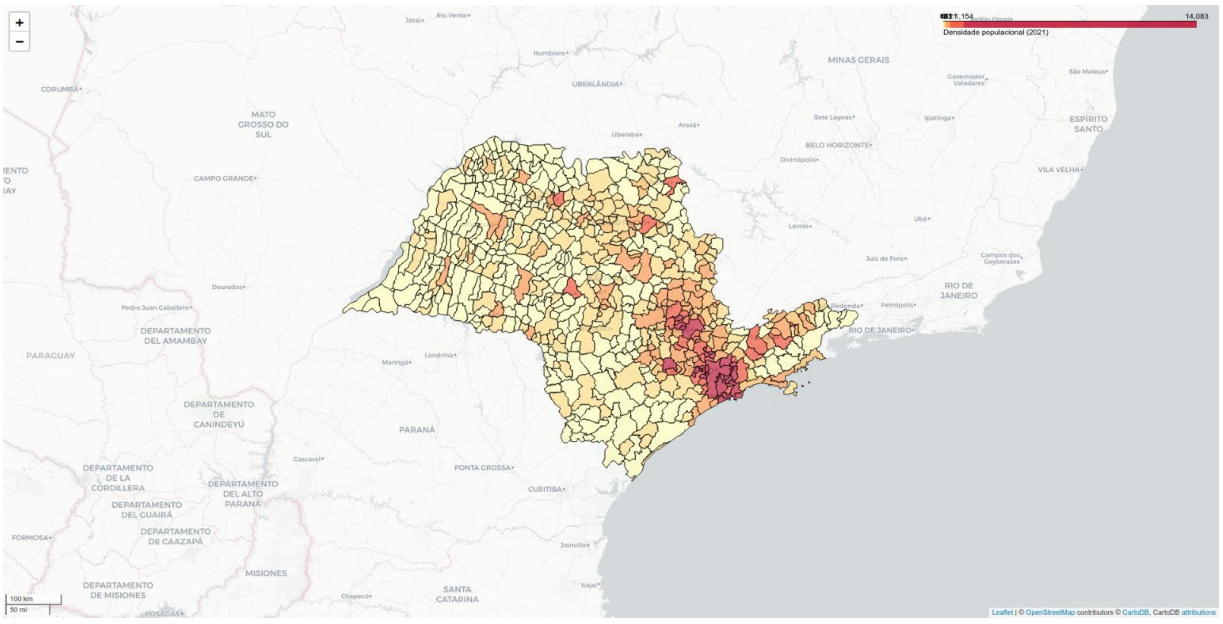
```
SP_LAT = -21.2922
SP_LON = -50.3428

mapa = folium.Map(location=[SP_LAT,SP_LON], control_scale = True, zoom_start=7,
tiles='cartodbpositron')


coropletico = folium.Choropleth(
geo_data=municipios,
data=municipios,
columns=['Nome','Densidadem'],
key_on='feature.properties.Nome',
legend_name='Densidade populacional (2021)',
fill_color = 'YlOrRd',
threshold_scale=escala
).add_to(mapa)
```

```
coropletico.geojson.add_child(

folium.features.GeoJsonTooltip(['Nome','Densidadem'])

)

mapa.save('densidade-demografica-sp-2021.html')

display(mapa)
```

7. Como o mapa resultante é grande, salvamos o arquivo localmente. Para ver o mapa, basta abrir o arquivo em um navegador.

```
files.download('densidade-demografica-sp-2021.html')
```

O resultado pode ser visto a seguir. Analise outras bases de dados e veja os resultados. Tente usar também outros valores de parâmetros para ver as diferenças entre métodos estatísticos e algorítmicos, com um ou mais atributos etc.



Versões das bibliotecas

Este tutorial foi feito usando as seguintes versões de bibliotecas:

```
folium==0.12.1.post1
```

geopandas==0.10.2