INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Brasília
Campus Brasília

TECNOLOGIA EM SISTEMAS PARA INTERNET

Rosana da Silva Soares
Kennedy Viana Aguiar
Aurélio Vinícius França dos Santos

RELATÓRIO 4 DE PRÁTICA INTEGRADA DE CIÊNCIA DE DADOS E INTELIGÊNCIA ARTIFICIAL

Brasília - DF

25 de Março de 2021

Sumário

1. Objetivos

Analisar dados de um espaço de tempo e local determinados.

Nesta etapa do projeto realizamos uma predição dos dados obtidos até este momento, podendo assim trabalhar em cima de possíveis referências futuras de dados.

2. Descrição do problema

Nos dias de hoje, temos vários dados entrando nos data frames e data lakes, diariamente. No caso do objeto desta pesquisa temos os ufo reports, mas seria possível fazer uma antecipação e ter uma previsão de quando ou quantos dados irão entrar? com os últimos 20 anos de dados de coletas podemos ter uma noção de como funcionam os relatos e podemos assim trabalhar em cima dos dados que já temos e fazer indicadores de como ficarão as curvas de crescimento.

Além do que foi descrito acima também foi proposto analisar dados sobre a cidade de phoenix Nos eua, puxando de anos diferentes os relatos durante os meses.

3. Desenvolvimento

Para organizar as colunas, utilizamos algumas bibliotecas como: *datetime, pandasql* e *pandas*

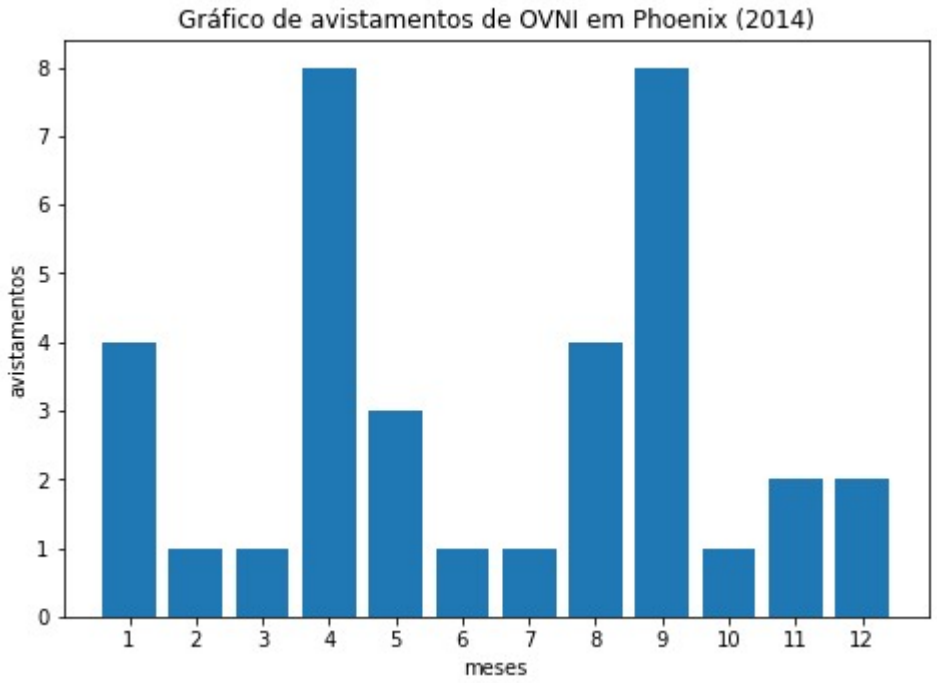
3.1 Código implementado

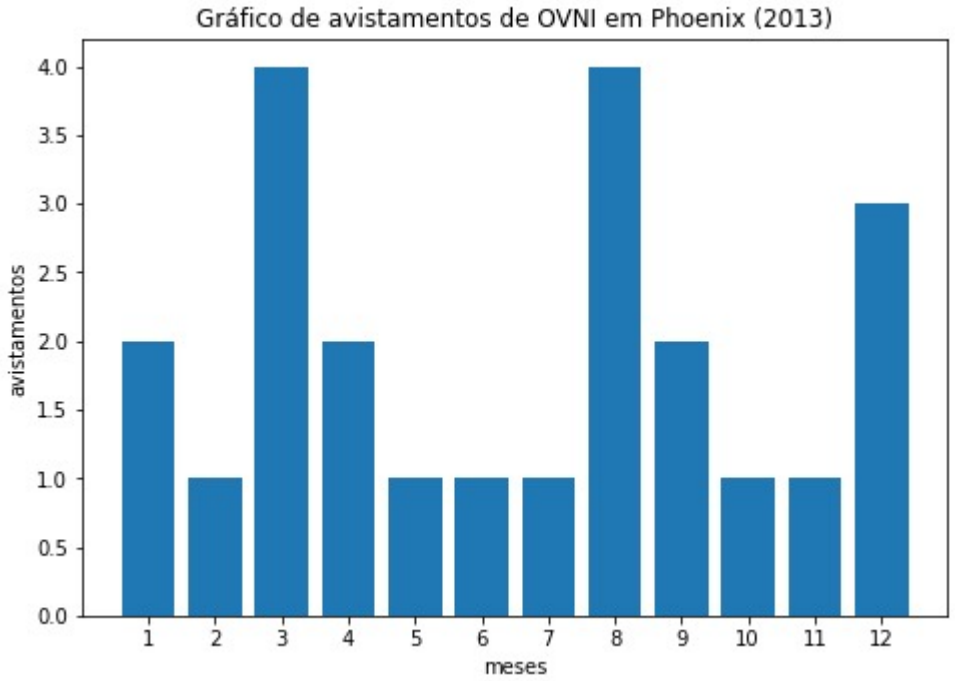
Importando as bibliotecas e módulos necessários:

```
#pip install zipcodes
#!pip install -U pandasql
from folium.plugins import HeatMap
import folium
import zipcodes
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from collections import Counter
import pandasql
from datetime import datetime
import statsmodels.api as sm
```

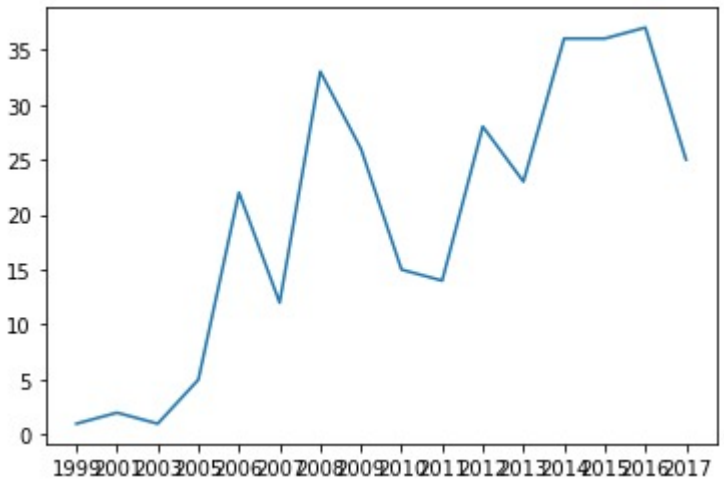
```
    # Lendo e ajustando o df_OVNI_preparado.csv
    df_preparado = pd.read_csv('df_OVNI_preparado.csv')
    df_preparado.drop(columns='Unnamed: 0', inplace=True)
    df_preparado['data'] = pd.to_datetime(df_preparado['data'],
format='%m/%d/%y')

    # Consulta o dataframe para obter a formação desejada
    ano = 2016
    qr = f"""
    SELECT data, mes, COUNT(*) AS views FROM df_preparado
    WHERE cidade='Phoenix' and ano={ano} GROUP BY data"""
    df_by_date = pd.DataFrame(pandasql.sqldf(qr, locals()))
```

```
      # Gera o figure do gráfico, as informações do eixo x e as seta o
eixo y
      fig = plt.figure()
      ax = fig.add_axes([0,0,1,1])
      eixo_mes = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11',
'12']
      eixo_views = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

      # Atualiza as informações do eixo y de acordo com o que vem do
dataframe
      for i in range(0, len(df_by_date['mes'])):
        eixo_views[df_by_date['mes'][i]-1] += df_by_date['views'][i]

      # Seta as informações do gráfico e exibe
      ax.bar(eixo_mes, eixo_views)
      ax.set_xlabel('meses')
      ax.set_ylabel('avistamentos')
      ax.set_title(f'Gráfico de avistamentos de OVNI em Phoenix ({ano})')
      plt.show()
```






```
# Faz uma busca pelas views por data da cidade de Phoenix
qr = """
SELECT ano, COUNT(*) AS views FROM df_preparado
WHERE cidade='Phoenix' GROUP BY ano"""
df_by_year = pd.DataFrame(pandasql.sqldf(qr, locals()))

# Gera os eixos x e y do gráfico
ypoints = df_by_year['views']
xpoints = list(map(str, df_by_year['ano']))

# Plota o gráfico (linhas)
plt.plot(xpoints, ypoints)
plt.show()
```




```
# Faz uma consulta pelas visualizacoes por data (novamente)
qr = f"""
SELECT data, COUNT(*) AS views FROM df_preparado
WHERE cidade='Phoenix' GROUP BY data"""
d = pd.DataFrame(pandasql.sqldf(qr, locals()))
```

```
# Divide o dataframe resultante da consulta em treinamento (70%) e
teste (30%)
treinamento, teste = d[0:198], d[198:]
modelo = SARIMAX(d['views'])
```

```
1 treinamento.head(10)
```

	data	views
0	1999-06-12 00:00:00.000000	1
1	2001-04-23 00:00:00.000000	1
2	2001-11-12 00:00:00.000000	1
3	2003-05-31 00:00:00.000000	1
4	2005-02-21 00:00:00.000000	1
5	2005-03-20 00:00:00.000000	1
6	2005-06-08 00:00:00.000000	1
7	2005-10-15 00:00:00.000000	1
8	2005-12-10 00:00:00.000000	1
9	2006-04-30 00:00:00.000000	1

Primeiras 10 linhas do Dataframe separado para treinamento

```
1 teste.tail(10)
```

	data	views
273	2017-05-04 00:00:00.000000	1
274	2017-05-06 00:00:00.000000	1
275	2017-05-19 00:00:00.000000	1
276	2017-05-29 00:00:00.000000	1
277	2017-06-07 00:00:00.000000	1
278	2017-06-15 00:00:00.000000	2
279	2017-07-06 00:00:00.000000	1
280	2017-07-26 00:00:00.000000	1
281	2017-08-04 00:00:00.000000	1
282	2017-08-14 00:00:00.000000	1

Últimas 10 linhas do Dataframe separado para teste

Com relação à parte de criação de um modelo de aprendizagem e às previsões, os resultados obtidos foram o valor de qualidade AIC e uma previsão gerada de acordo com o dataframe de treinamento (70%). A figura abaixo representa o código implementado e seu respectivo resultado:

```
1 modelo = SARIMAX(treinamento['views'], order=(1, 0, 0), trend='c')
2 res = modelo.fit()
3 print(f"A qualidade do modelo estimada pelo AIC é: {res.aic}\n")
4 print(f"Previsão: \n{res.forecast()}")
```

```
A qualidade do modelo estimada pelo AIC é: 282.35126963667506

Previsão:
198    1.087409
dtype: float64
```

4. Considerações Finais

Com a mudança do Script implementado, o código rodou de forma mais eficiente, nos foi poupado tempo e muitas linhas de código, a compreensão de cada bloco de código também foi um ponto crucial para ajudar a alinhar o raciocínio.

Diversos obstáculos foram encontrados na resolução desta proposta de prática integrada, mas ainda assim todas foram superadas.

Referências

The National UFO Reporting Center. Nuforc, 2021. Disponível em: <http://www.nuforc.org/>. Acesso em: 20 de Março. 2021.

Python 3.9.2 documentation. Python,2021. Disponível em: <https://docs.python.org/pt-br/3/library/datetime.html>. Acesso em: 20 de Março.2021.

Python package index. Python, 2021. Disponível em: <https://pypi.org/project/pandasql/>. Acesso em: 20 de Março.2021.