Google

Google Search Analysis with Python

Full Project

Data Science & Data Analytics

Google

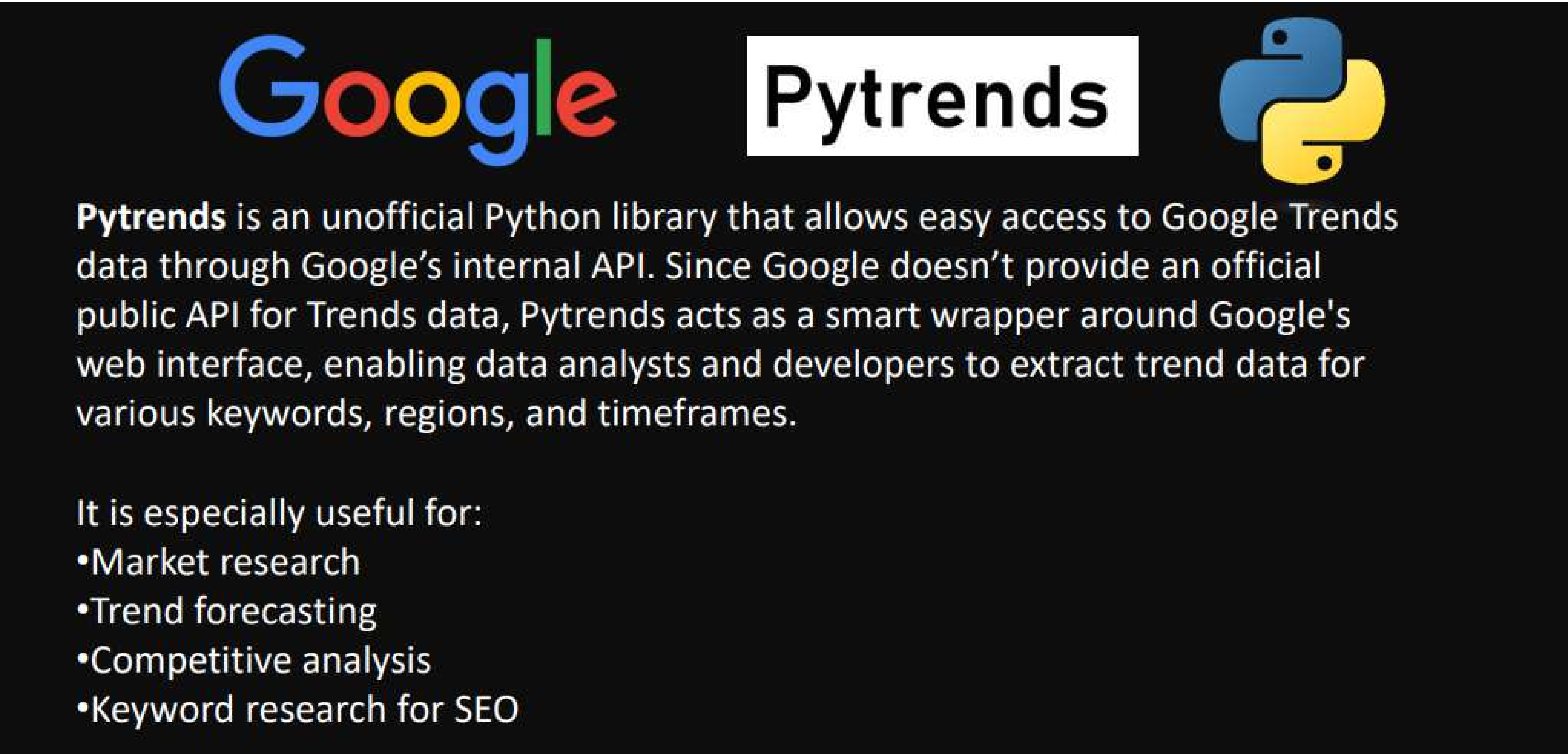
As of recent estimates (2024–2025), Google processes over 8.5 billion searches per day

Google handles more than 99,000 searches every second, which equals:

- ~5.9 million searches per minute
- ~8.5 billion searches per day
- ~2.6 trillion searches per year

Google Trends data represents a very reliable proxy for public interest, making it ideal for:

- Market research
- Political forecasting
- News and media analysis
- Product demand analysis

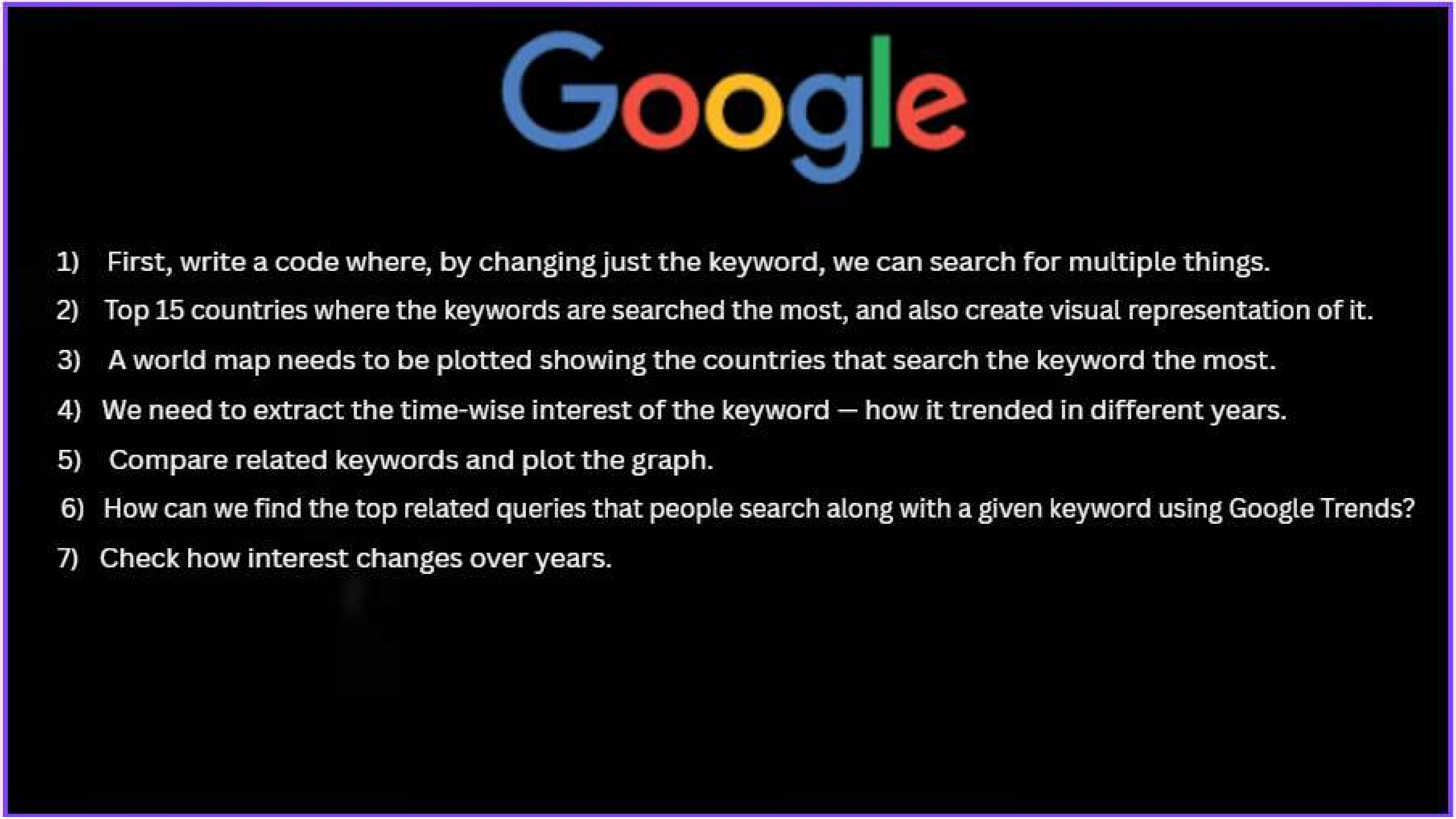


Pytrends

Pytrends is an unofficial Python library that allows easy access to Google Trends data through Google’s internal API. Since Google doesn’t provide an official public API for Trends data, Pytrends acts as a smart wrapper around Google's web interface, enabling data analysts and developers to extract trend data for various keywords, regions, and timeframes.

It is especially useful for:

- Market research
- Trend forecasting
- Competitive analysis
- Keyword research for SEO



Google

1) First, write a code where, by changing just the keyword, we can search for multiple things.

2) Top 15 countries where the keywords are searched the most, and also create visual representation of it.

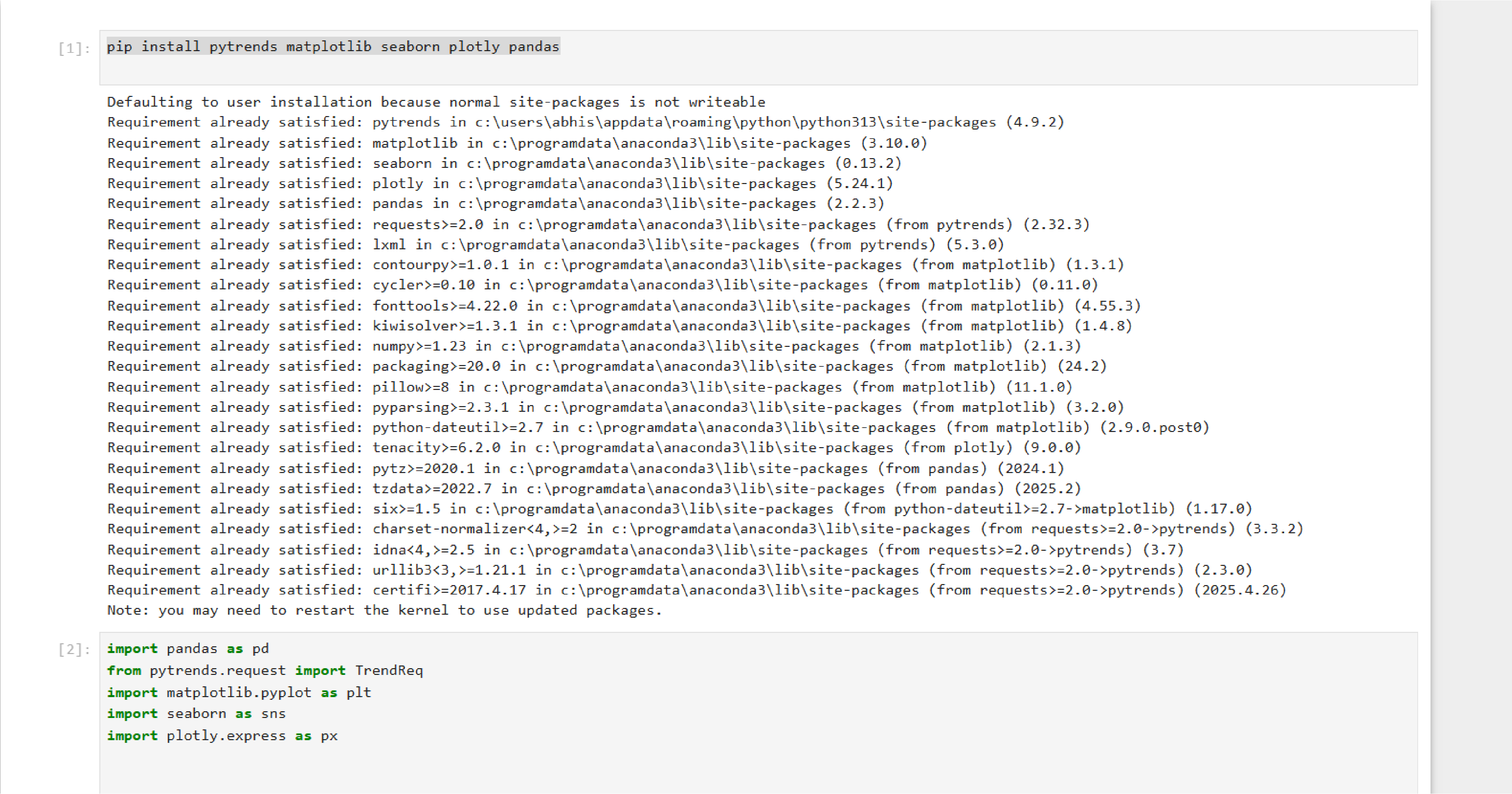
3) A world map needs to be plotted showing the countries that search the keyword the most.

4) We need to extract the time-wise interest of the keyword — how it trended in different years.

5) Compare related keywords and plot the graph.

6) How can we find the top related queries that people search along with a given keyword using Google Trends?

7) Check how interest changes over years.



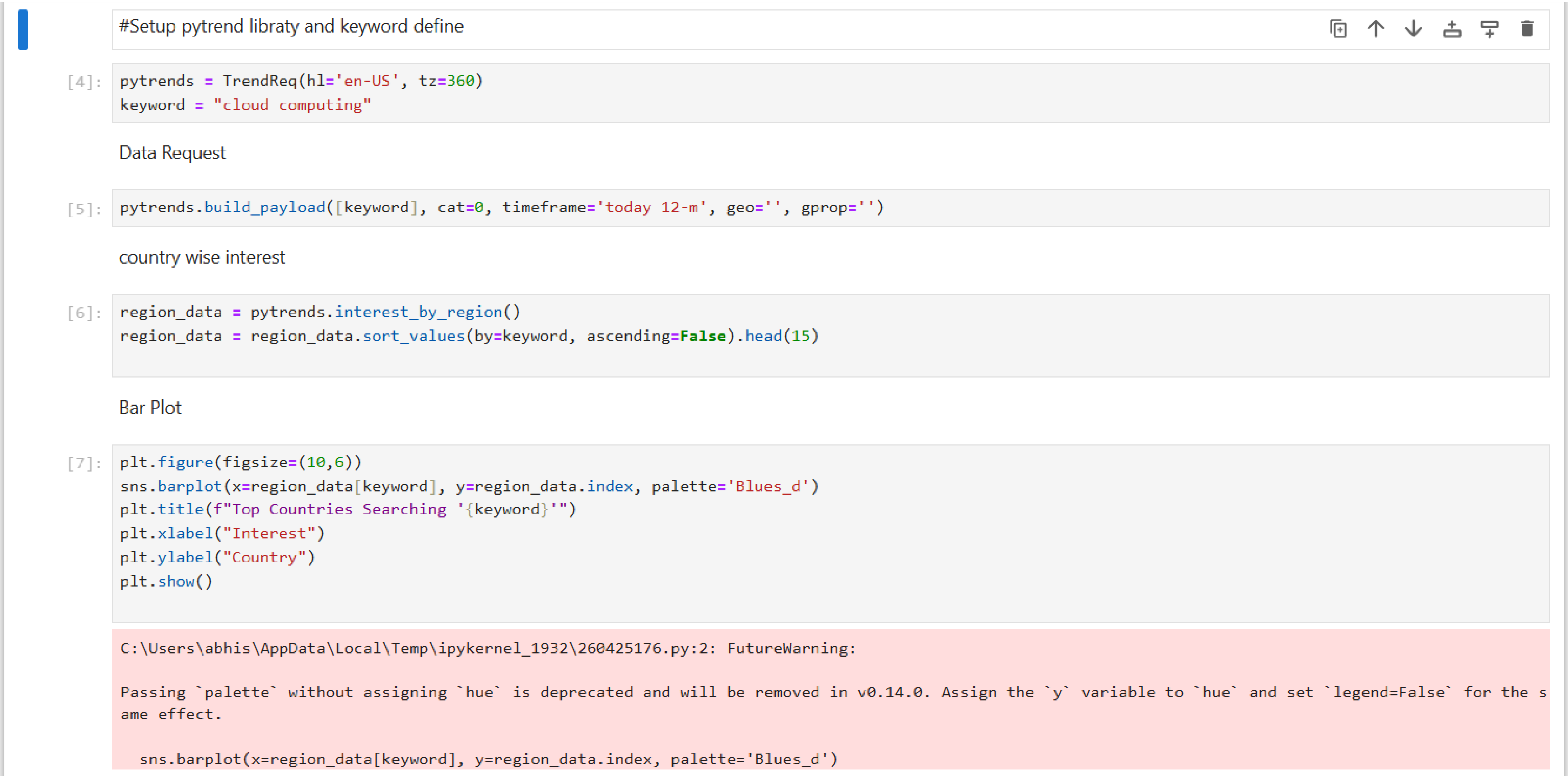
[1]:

```
pip install pytrends matplotlib seaborn plotly pandas
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pytrends in c:\users\abhis\appdata\roaming\python\python313\site-packages (4.9.2)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: seaborn in c:\programdata\anaconda3\lib\site-packages (0.13.2)
Requirement already satisfied: plotly in c:\programdata\anaconda3\lib\site-packages (5.24.1)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (2.2.3)
Requirement already satisfied: requests>=2.0 in c:\programdata\anaconda3\lib\site-packages (from pytrends) (2.32.3)
Requirement already satisfied: lxml in c:\programdata\anaconda3\lib\site-packages (from pytrends) (5.3.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (2.1.3)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: tenacity>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from plotly) (9.0.0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.0->pytrends) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.0->pytrends) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.0->pytrends) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests>=2.0->pytrends) (2025.4.26)
Note: you may need to restart the kernel to use updated packages.
```

[2]:

```python
import pandas as pd
from pytrends.request import TrendReq
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```



#Setup pytrend libraty and keyword define

[4]:
```
pytrends = TrendReq(hl='en-US', tz=360)
keyword = "cloud computing"
```

Data Request

[5]:
```
pytrends.build_payload([keyword], cat=0, timeframe='today 12-m', geo='', gprop='')
```

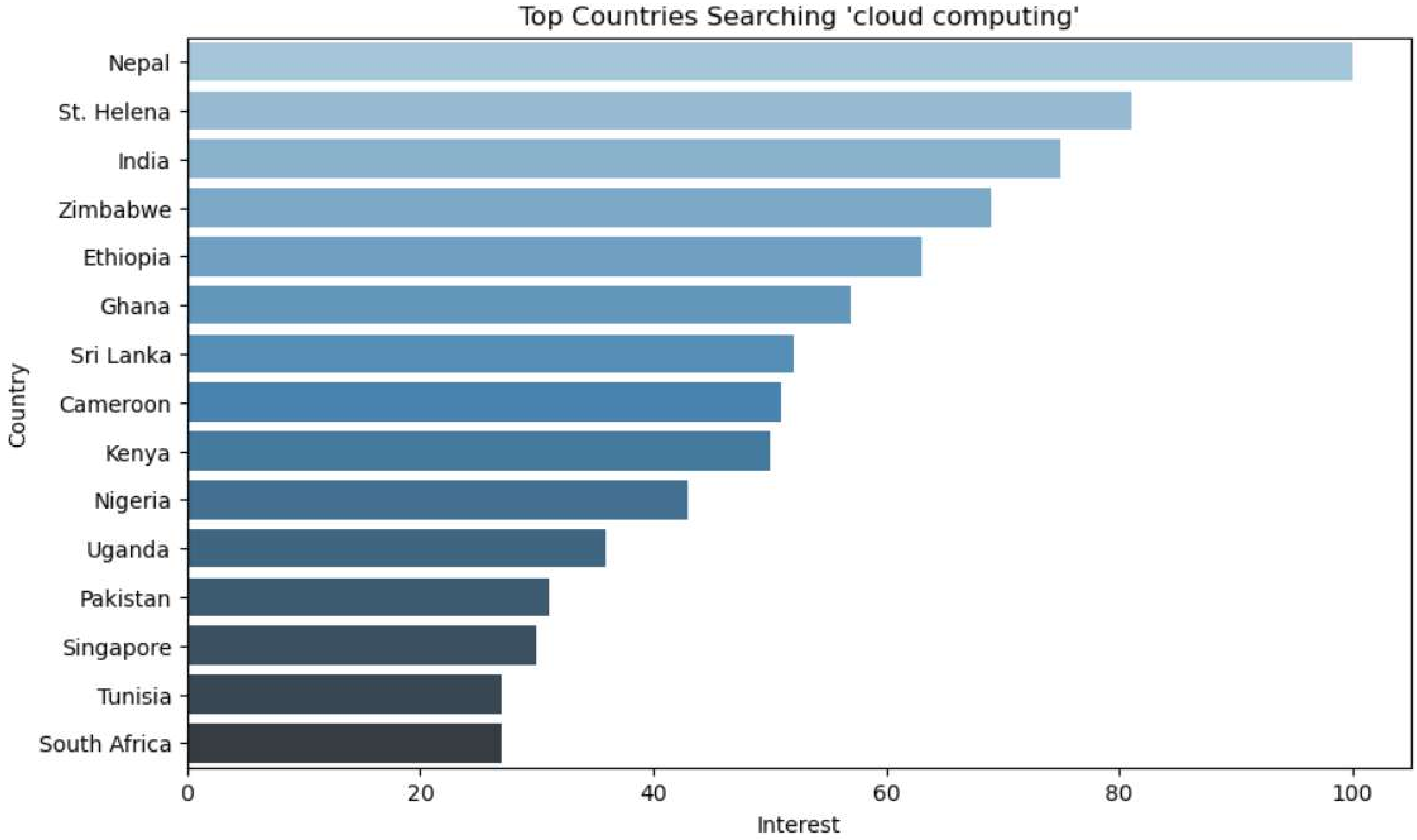
country wise interest

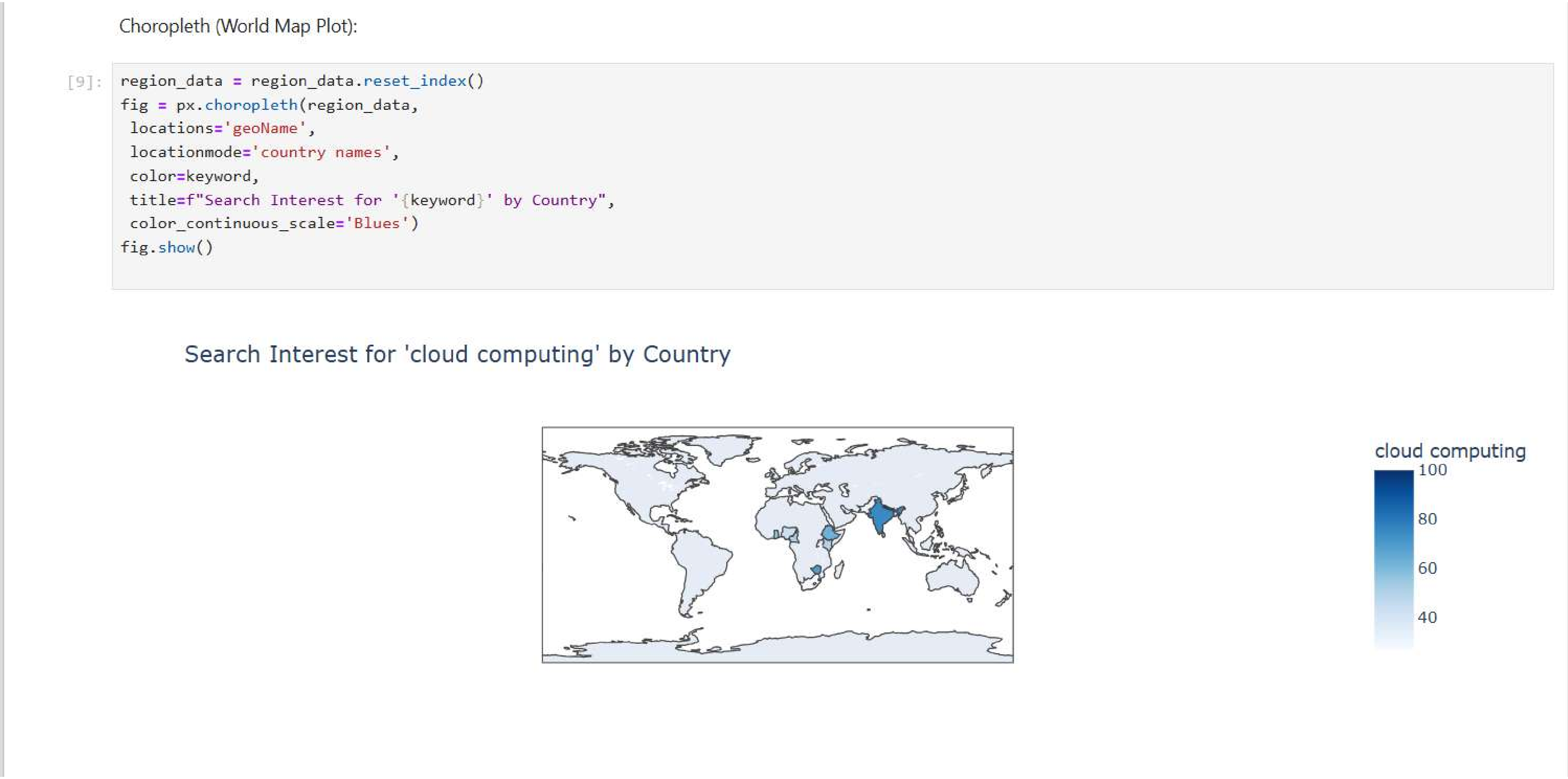
[6]:
```
region_data = pytrends.interest_by_region()
region_data = region_data.sort_values(by=keyword, ascending=False).head(15)
```

Bar Plot

[7]:
```
plt.figure(figsize=(10,6))
sns.barplot(x=region_data[keyword], y=region_data.index, palette='Blues_d')
plt.title(f"Top Countries Searching '{keyword}'")
plt.xlabel("Interest")
plt.ylabel("Country")
plt.show()
```

```
C:\Users\abhis\AppData\Local\Temp\ipykernel_1932\260425176.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the s
ame effect.

  sns.barplot(x=region_data[keyword], y=region_data.index, palette='Blues_d')
```

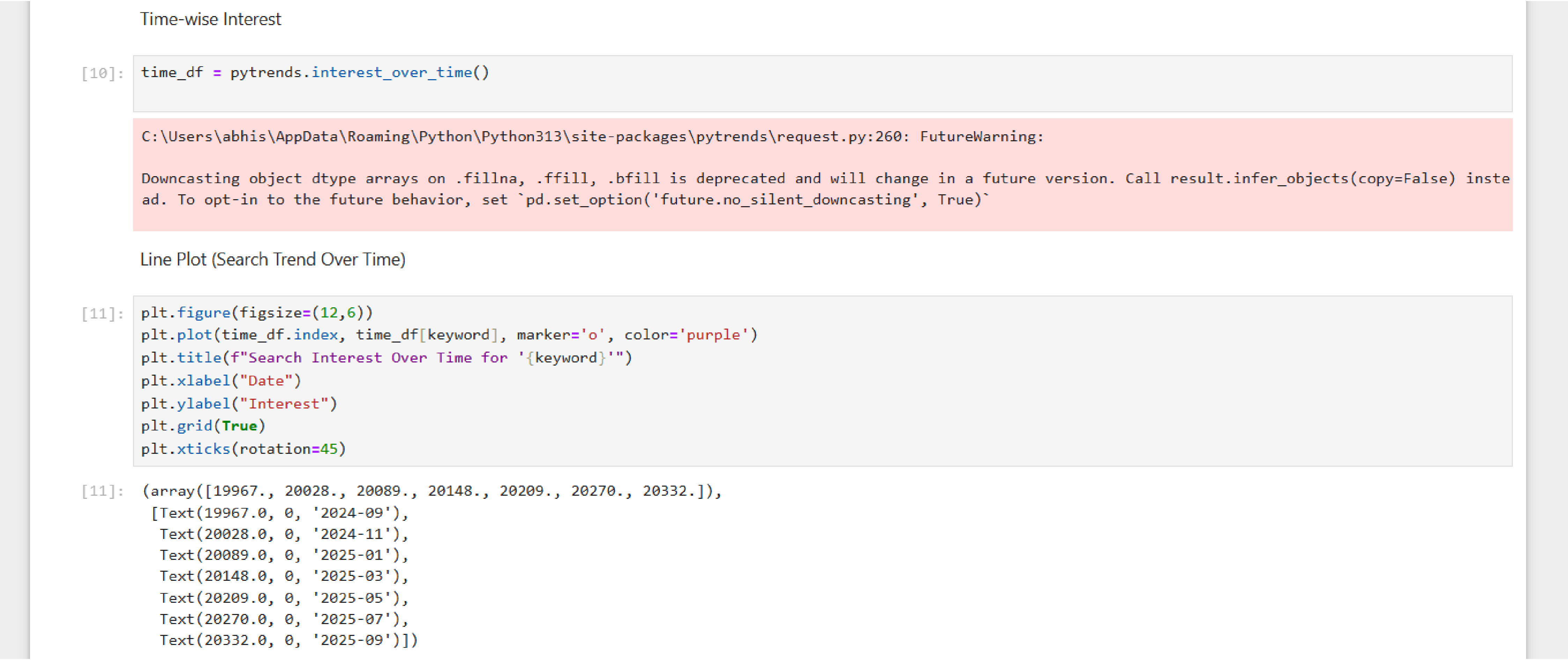

Top Countries Searching 'cloud computing'
Nepal
St. Helena
India
Zimbabwe
Ethiopia
Ghana
Sri Lanka
Cameroon
Kenya
Nigeria
Uganda
Pakistan
Singapore
Tunisia
South Africa
Country
0
20
40
60
80
100
Interest



Choropleth (World Map Plot):

```
[9]: region_data = region_data.reset_index()
     fig = px.choropleth(region_data,
      locations='geoName',
      locationmode='country names',
      color=keyword,
      title=f"Search Interest for '{keyword}' by Country",
      color_continuous_scale='Blues')
     fig.show()
```

Search Interest for 'cloud computing' by Country



Time-wise Interest

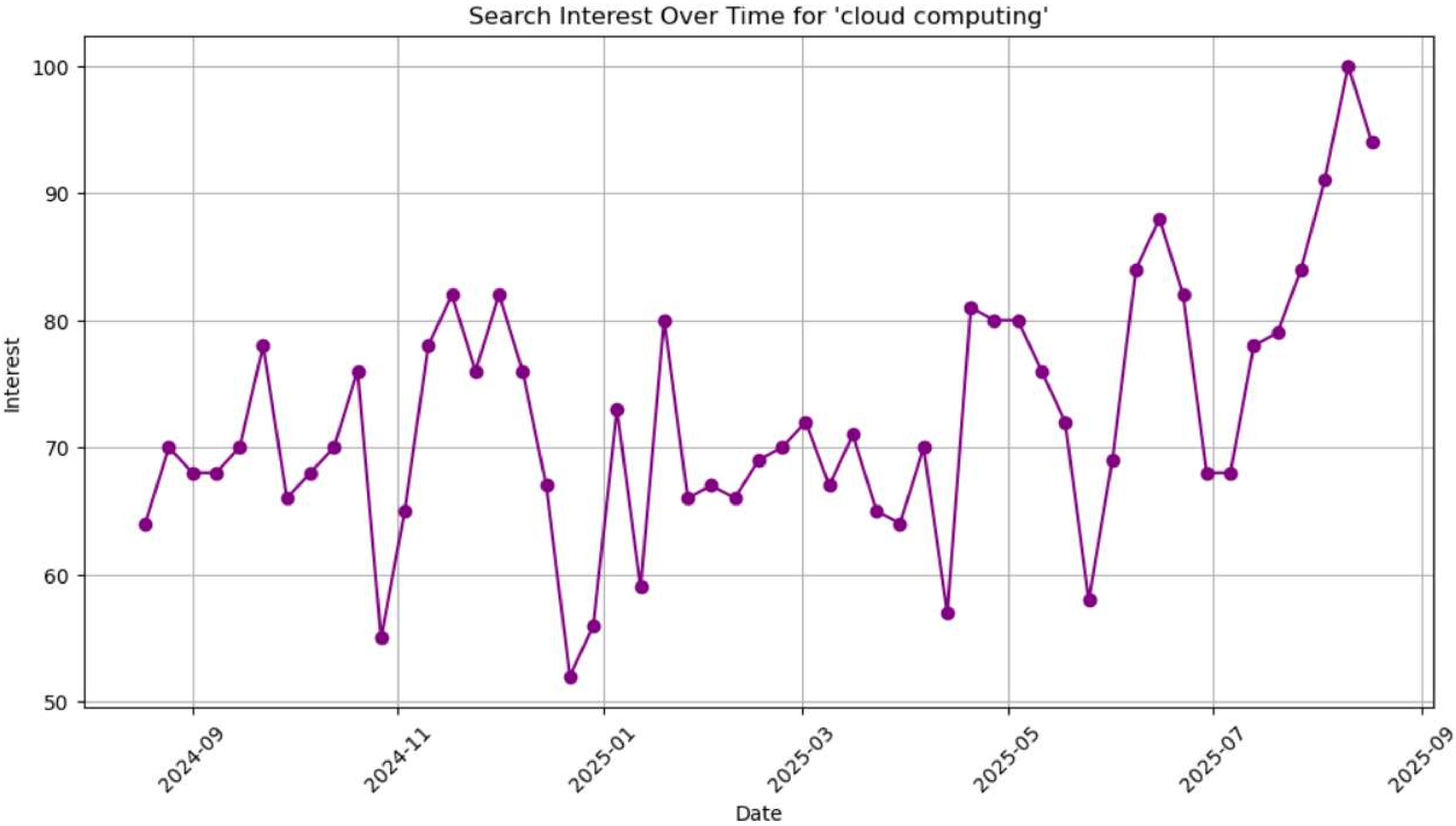
```
[10]: time_df = pytrends.interest_over_time()
```

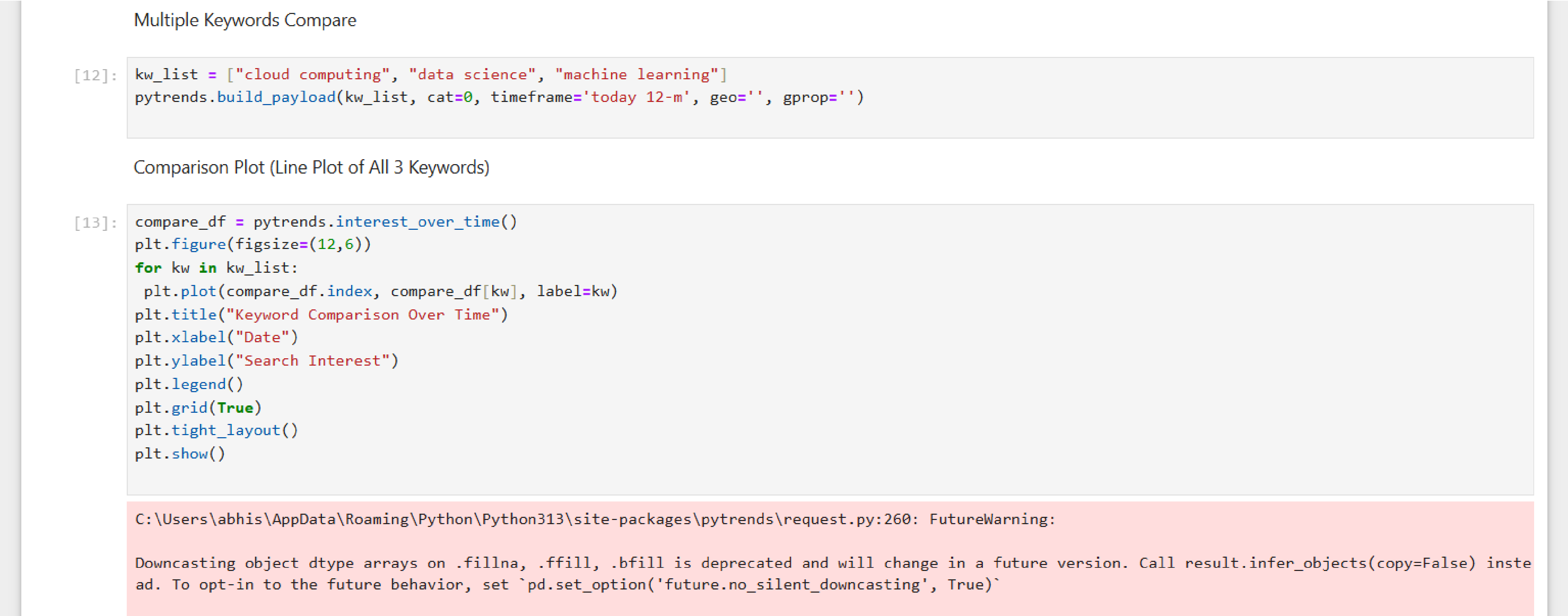
```
C:\Users\abhis\AppData\Roaming\Python\Python313\site-packages\pytrends\request.py:260: FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and will change in a future version. Call result.infer_objects(copy=False) inste
ad. To opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
```

Line Plot (Search Trend Over Time)

```
[11]: plt.figure(figsize=(12,6))
      plt.plot(time_df.index, time_df[keyword], marker='o', color='purple')
      plt.title(f"Search Interest Over Time for '{keyword}'")
      plt.xlabel("Date")
      plt.ylabel("Interest")
      plt.grid(True)
      plt.xticks(rotation=45)
```

```
[11]: (array([19967., 20028., 20089., 20148., 20209., 20270., 20332.]),
       [Text(19967.0, 0, '2024-09'),
        Text(20028.0, 0, '2024-11'),
        Text(20089.0, 0, '2025-01'),
        Text(20148.0, 0, '2025-03'),
        Text(20209.0, 0, '2025-05'),
        Text(20270.0, 0, '2025-07'),
        Text(20332.0, 0, '2025-09')])
```


Search Interest Over Time for 'cloud computing'
Interest
100
90
80
70
60
50
Date
2024-09
2024-11
2025-01
2025-03
2025-05
2025-07
2025-09



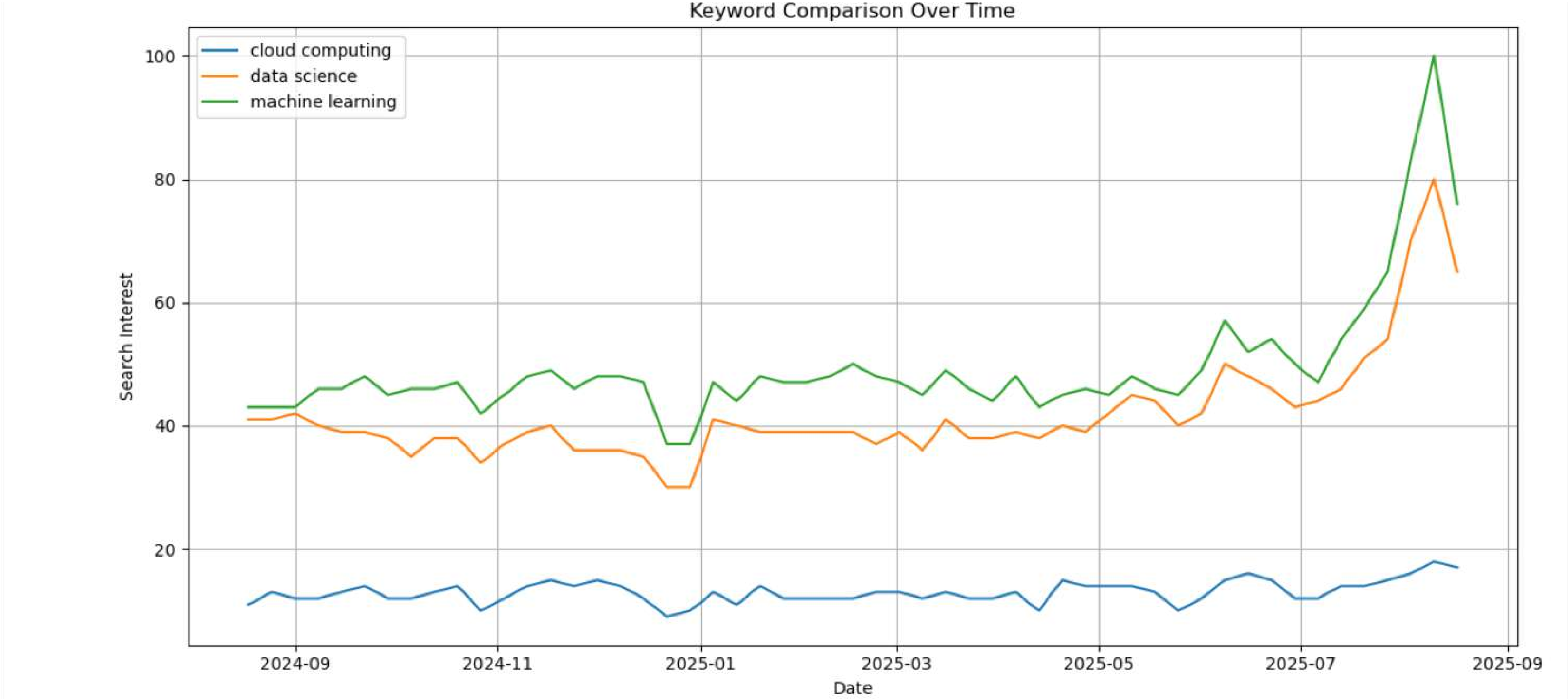
Multiple Keywords Compare

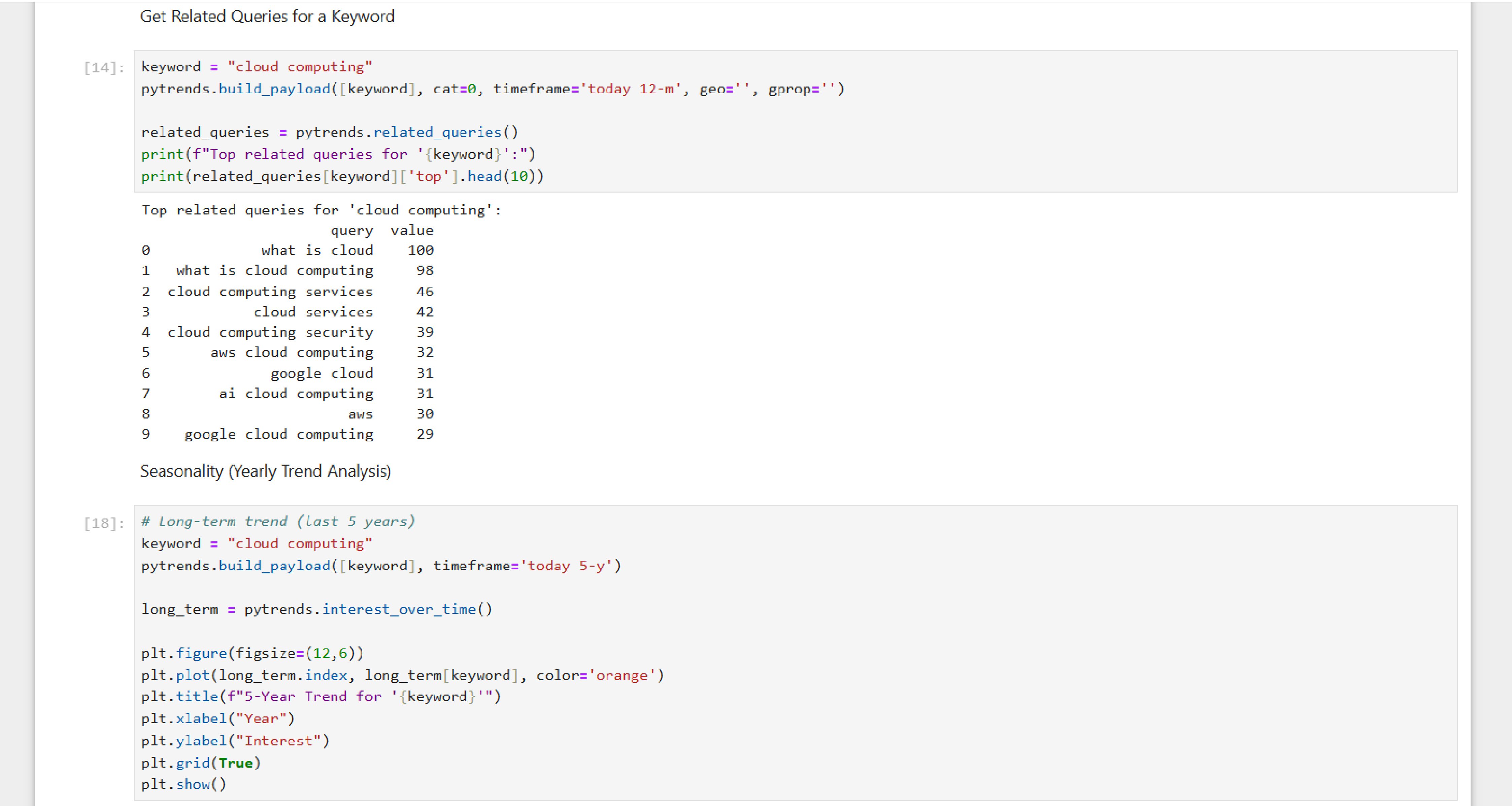
```
[12]: kw_list = ["cloud computing", "data science", "machine learning"]
      pytrends.build_payload(kw_list, cat=0, timeframe='today 12-m', geo='', gprop='')
```

Comparison Plot (Line Plot of All 3 Keywords)

```
[13]: compare_df = pytrends.interest_over_time()
      plt.figure(figsize=(12,6))
      for kw in kw_list:
       plt.plot(compare_df.index, compare_df[kw], label=kw)
      plt.title("Keyword Comparison Over Time")
      plt.xlabel("Date")
      plt.ylabel("Search Interest")
      plt.legend()
      plt.grid(True)
      plt.tight_layout()
      plt.show()
```

```
C:\Users\abhis\AppData\Roaming\Python\Python313\site-packages\pytrends\request.py:260: FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and will change in a future version. Call result.infer_objects(copy=False) inste
ad. To opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
```


Keyword Comparison Over Time
cloud computing
data science
machine learning
Search Interest
100
80
60
40
20
Date
2024-09
2024-11
2025-01
2025-03
2025-05
2025-07
2025-09



Get Related Queries for a Keyword

[14]:

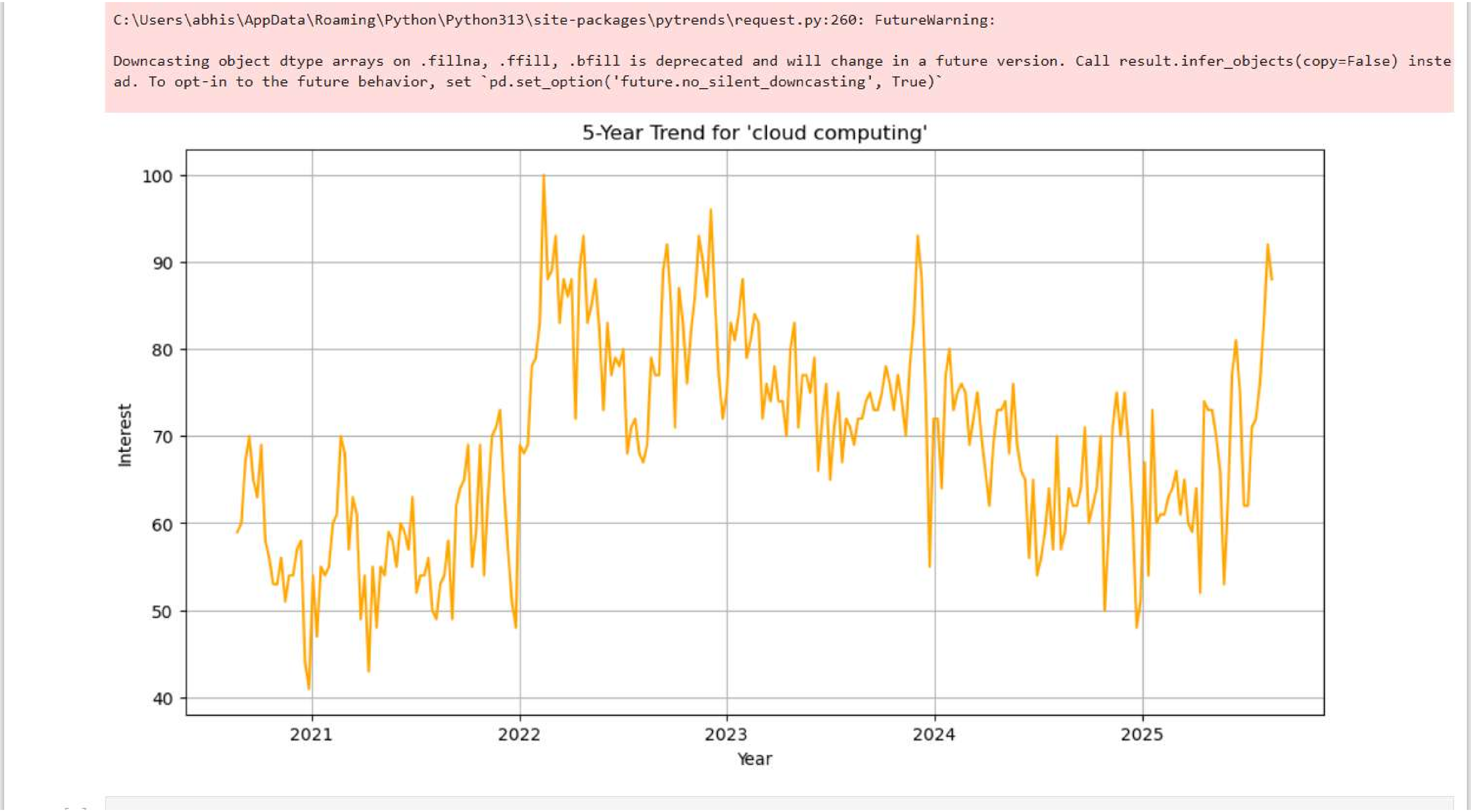
```
keyword = "cloud computing"
pytrends.build_payload([keyword], cat=0, timeframe='today 12-m', geo='', gprop='')

related_queries = pytrends.related_queries()
print(f"Top related queries for '{keyword}':")
print(related_queries[keyword]['top'].head(10))
```

```
Top related queries for 'cloud computing':
                      query  value
0             what is cloud    100
1  what is cloud computing     98
2 cloud computing services     46
3            cloud services     42
4 cloud computing security     39
5      aws cloud computing     32
6              google cloud     31
7       ai cloud computing     31
8                       aws     30
9    google cloud computing     29
```

Seasonality (Yearly Trend Analysis)

[18]:

```
# Long-term trend (last 5 years)
keyword = "cloud computing"
pytrends.build_payload([keyword], timeframe='today 5-y')

long_term = pytrends.interest_over_time()

plt.figure(figsize=(12,6))
plt.plot(long_term.index, long_term[keyword], color='orange')
plt.title(f"5-Year Trend for '{keyword}'")
plt.xlabel("Year")
plt.ylabel("Interest")
plt.grid(True)
plt.show()
```



```
C:\Users\abhis\AppData\Roaming\Python\Python313\site-packages\pytrends\request.py:260: FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and will change in a future version. Call result.infer_objects(copy=False) inste
ad. To opt-in to the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
```