CHƯƠNG 2. TRỰC QUAN HOÁ DỮ LIỆU

2.1. Giới thiệu matplotlib

2.2. Vẽ biểu đồ với matplotlib (line, curve, bar, scatter, pie, histogram, boxplot, …)

2.3. Trực quan hoá dùng seaborn

2.4. Trực quan hoá dùng các hàm trong pandas

2.5 Trực quan hoá với plotly và một số công cụ khác

2.6. Vẽ biểu đồ địa lý

2.7. Trực quan hoá các dữ liệu kinh doanh

CHƯƠNG 2. TRỰC QUAN HOÁ DỮ LIỆU

Trực quan hoá dữ liệu:

- Là quá trình biểu diễn thông tin, dữ liệu dưới dạng đồ họa (hình ảnh, biểu đồ, bản đồ, v.v.) để giúp dễ dàng nhận biết xu hướng, mẫu hình, hoặc điểm bất thường trong dữ liệu.
- Mục tiêu là chuyển đổi dữ liệu phức tạp thành hình ảnh trực quan, dễ hiểu, hỗ trợ ra quyết định hoặc truyền đạt thông tin hiệu quả.
- Công cụ trực quan hóa :
    - Lập trình Python (Matplotlib, Seaborn, Pandas, Plotly),
    - R (ggplot2).
    - Power BI

2.1. Giới thiệu matplotlib

- Matplotlib là một thư viện Python phổ biến được sử dụng để tạo các biểu đồ tĩnh, động và tương tác. Nó hỗ trợ nhiều loại biểu đồ như đường, cột, tán xạ, hình tròn,...

- Giúp trực quan hóa dữ liệu dưới dạng đồ họa, dễ dàng so sánh và nhận diện các xu hướng hay mẫu dữ liệu.

Ví dụ:

```
import matplotlib.pyplot as plt

x = [1,2, 3, 4] # Trục x

y = [10, 20, 25, 30] # Trục y

#    Vẽ biểu đồ đường cơ bản
plt.plot(x y)
plt.title("Biểu đồ đường cơ bản") # Tiêu đề
plt.xlabel("X-axis")     # Nhãn trục x
plt.ylabel("Y-axis")     # Nhãn trục y
plt.show()
```

2.2. Vẽ biểu đồ với matplotlib (line, curve, bar, scatter, pie, histogram, boxplot, ...)

Matplotlib hỗ trợ nhiều loại biểu đồ, mỗi loại biểu đồ phù hợp với các dạng dữ liệu khác nhau:

- **Line chart (Đường)**: Thường dùng để biểu diễn xu hướng.
- **Bar chart (Cột)**: Dùng để so sánh dữ liệu theo nhóm.
- **Scatter plot (Biểu đồ tán xạ)**: Dùng để quan sát mối quan hệ giữa hai biến.
- **Pie chart (Tròn)**: Thể hiện tỷ lệ các thành phần.
- **Histogram**: Phân phối tần suất của một tập dữ liệu.
- **Boxplot**: Phân tích phân vị, giá trị ngoại lai (outliers).

Các biểu đồ giúp trình bày và phân tích dữ liệu từ nhiều góc độ khác nhau.

Ví dụ (Line, Bar, Scatter, Pie, Histogram):

```
import matplotlib.pyplot as plt
#    Dữ liệu
x = [1,2, 3, 4]
y = [10, 20, 25, 30]
#    Line chart
plt.figure(figsize=(8, 5)) # Tạo vùng vẽ
plt.subplot(2, 3, 1)
plt.plot(x, y)
plt.title("Line Chart")
#    Bar chart
plt.subplot(2, 3, 2)
plt.bar(x, y)
plt.title("Bar Chart")
#    Scatter plot
plt.subplot(2, 3, 3)
plt.scatter(x, y)
plt.title("Scatter Chart")
#    Pie chart
plt.subplot(2, 3, 4)
plt.pie([10, 20, 30, 40], labels=['A',
'B', 'C', 'D'], autopct='%1.1f%%’)
plt.title("Pie Chart")
#    Histogram
plt.subplot(2, 3, 5)
data = [1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5]
plt.hist(data, bins=5)
plt.title("Histogram")
plt.tight_layout() # Sắp xếp đồ thị hợp
lý
save_path =
/Users/mrhai/Desktop/KHDL/bieudo.png"
plt.savefig(save_path, dpi=300)
plt.show()
```

2.3. Trực quan hoá dùng seaborn

- Seaborn là một thư viện Python, được xây dựng dựa trên Matplotlib, để tạo các biểu đồ trực quan phức tạp hơn (như nhiệt đồ - heatmaps, biểu đồ phân phối - distribution plots).
- Seaborn dễ sử dụng hơn Matplotlib trong việc trực quan hóa quan hệ, phân phối và phân tích dữ liệu từ góc độ thống kê.

Ví dụ:

```
import seaborn as sns

import matplotlib.pyplot as plt

#    Dữ liệu ví dụ
tips = sns.load_dataset("tips")

print(tips)

#    Vẽ heatmap để xem sự tương quan
sns.heatmap(tips.corr(numeric_only=True), annot=True,
cmap="coolwarm")
plt.title("Heatmap of Correlation")
plt.show()
```

2.4. Trực quan hoá dùng các hàm trong pandas

- Pandas là thư viện xử lý dữ liệu, và nó tích hợp khả năng trực quan hóa dữ liệu cơ bản thông qua các hàm như .plot().

- Dễ dàng vừa thao tác dữ liệu, vừa trực quan hóa mà không cần chuyển đổi thư viện.

Ví dụ:

```
import pandas as pd

import matplotlib.pyplot as plt

#    Dữ liệu
data = {"Year": [2019, 2020, 2021], "Sales": [200, 300, 400]}

df = pd.DataFrame(data)

#    Vẽ biểu đồ cột
df.plot(x="Year", y="Sales", kind="bar", title="Doanh số hàng năm")
plt.show()
```

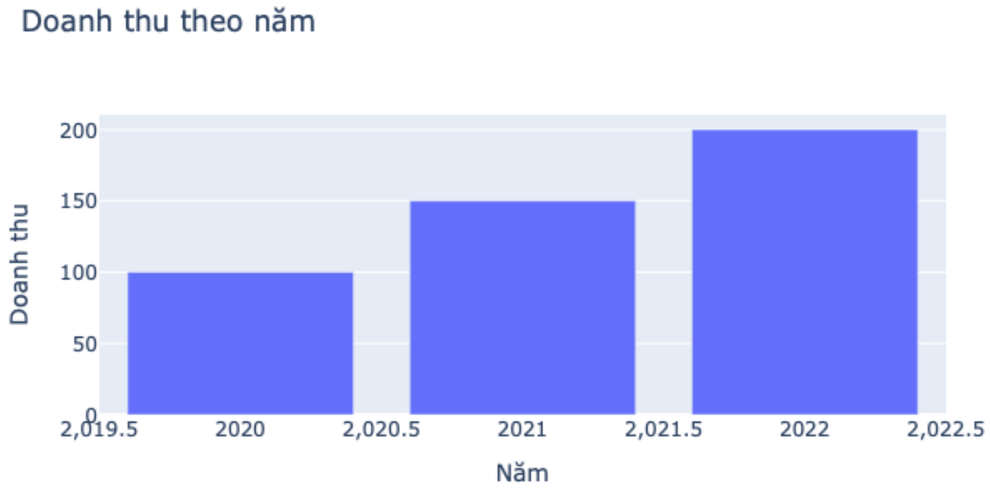
2.5. Trực quan hoá với Plotly và một số công cụ khác

- Plotly là thư viện tạo các biểu đồ tương tác đẹp, có thể dùng trực tuyến (đặc biệt trong báo cáo hoặc dashboard).

- Trình bày insight sâu sắc qua các biểu đồ tương tác, phù hợp cho thuyết trình và tạo dashboard.

Ví dụ với Plotly: cài đặt thu viện plotly (!pip install plotly)

```
import plotly.express as px
# Tạo một DataFrame đơn giản
data = {
    'Năm': [2020, 2021, 2022],
    'Doanh thu': [100, 150, 200]
}
# Vẽ biểu đồ
fig = px.bar(data, x='Năm', y='Doanh thu', title='Doanh thu theo năm')
fig.show()
```



2.5. Trực quan hoá với Plotly và một số công cụ khác

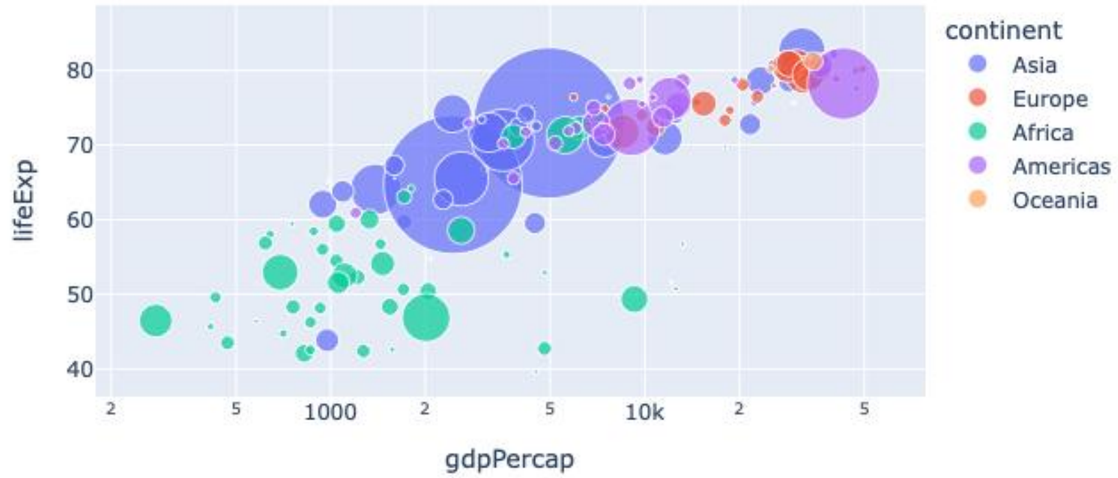
Ví dụ: Tập dữ liệu Gapminder được phát triển bởi giáo sư Hans Rosling và tổ chức Gapminder, nhằm mục đích theo dõi sự phát triển kinh tế và xã hội trên toàn cầu.

- **GDP per capita**: Tổng sản phẩm quốc nội (GDP) trên đầu người, thể hiện mức độ phát triển kinh tế của một quốc gia.
- **Life Expectancy**: Tuổi thọ trung bình, cho thấy sức khỏe và điều kiện sống của người dân.
- **Population**: Số lượng dân cư của mỗi quốc gia.
- **Continent**: Châu lục mà quốc gia đó thuộc về.
- **Year**: Năm mà các dữ liệu được thu thập

2.5. Trực quan hoá với Plotly và một số công cụ khác

Ví dụ: Tập dữ liệu Gapminder

```
import plotly.express as px
# Dữ liệu ví dụ
df = px.data.gapminder()
Print(df)
# Vẽ biểu đồ phân tán (scatter) tương tác
fig = px.scatter(
        df[df["year"] == 2007],
        x="gdpPercap",
        y="lifeExp",
        size="pop",
        color="continent",
        hover_name="country",
        log_x=True,
        size_max=60
)
fig.show()
```



Hover_name=country: biểu hiện di chuột đến sẽ hiện tên quốc gia
log_x=true : chuyển dổi trục x sang tỷ lệ logarit, Điều này rất hữu ích khi bạn có dữ liệu trải rộng từ số rất nhỏ đến rất lớn (như GDP)
Size_max=60: xác định kích thước tối đa của các điểm trong biểu đồ, ở đây là 60.

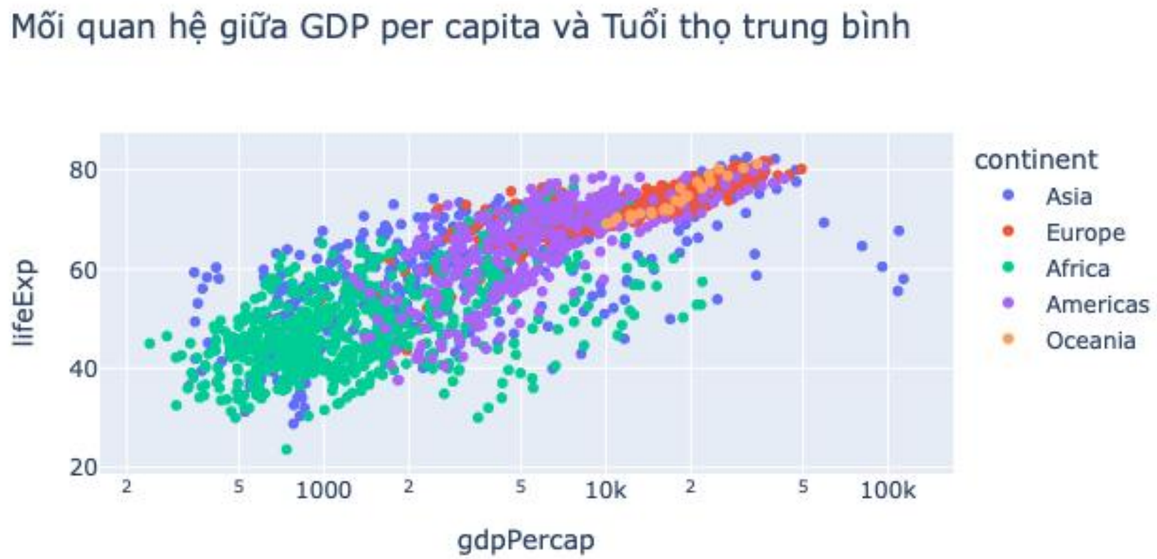
2.5. Trực quan hoá với Plotly và một số công cụ khác

Ví dụ: Tập dữ liệu Gapminder

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

# Tải tập dữ liệu gapminder từ thư viện plotly
df = px.data.gapminder()

# Hiển thị 5 dòng đầu tiên của tập dữ liệu
print(df.head())
# df.info()
# Vẽ biểu đồ phân tán
fig = px.scatter(df,
                 x='gdpPercap',
                 y='lifeExp',
                 color='continent',
                 hover_name='country',
                 title='Mối quan hệ giữa GDP per capita và Tuổi thọ trung bình',
                 log_x=True)  # Sử dụng log scale cho trục x
fig.show()
```



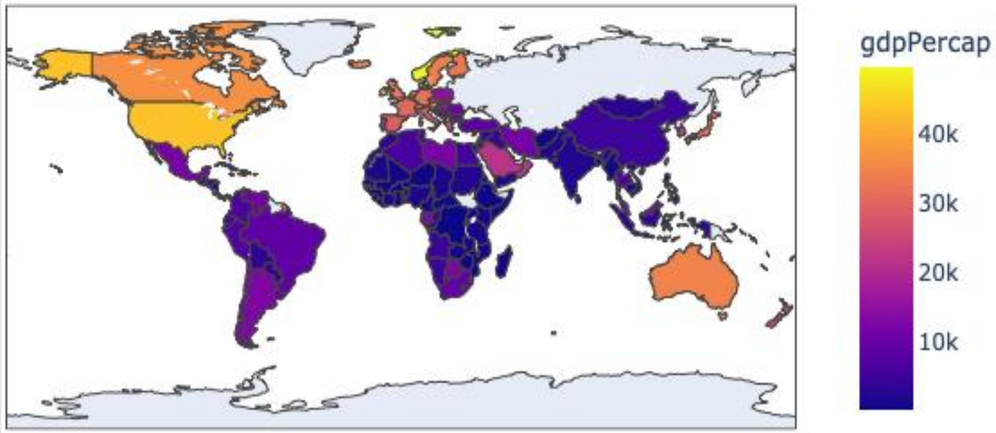
2.6. Vẽ biểu đồ địa lý

- Biểu đồ địa lý hiển thị dữ liệu trên bản đồ để minh họa trực quan các phân phối địa lý, chẳng hạn như Folium, Plotly Geo, hoặc Geopandas thường dùng để tạo biểu đồ địa lý.

- Hữu ích khi cần phân tích dữ liệu có liên quan đến địa điểm, vùng miền (vd: dân số theo quốc gia, doanh thu theo khu vực địa lý).

Ví dụ:

```
import plotly.express as px
#    Dữ liệu
df = px.data.gapminder()
# Biểu đồ bản đồ thể hiện GDP thế giới năm 2007
fig = px.choropleth(
        df[df["year"] == 2007],
        locations="iso_alpha",
        color="gdpPercap",
        hover_name="country",
        color_continuous_scale=px.colors.sequential.Plasma
)
fig.show()
```



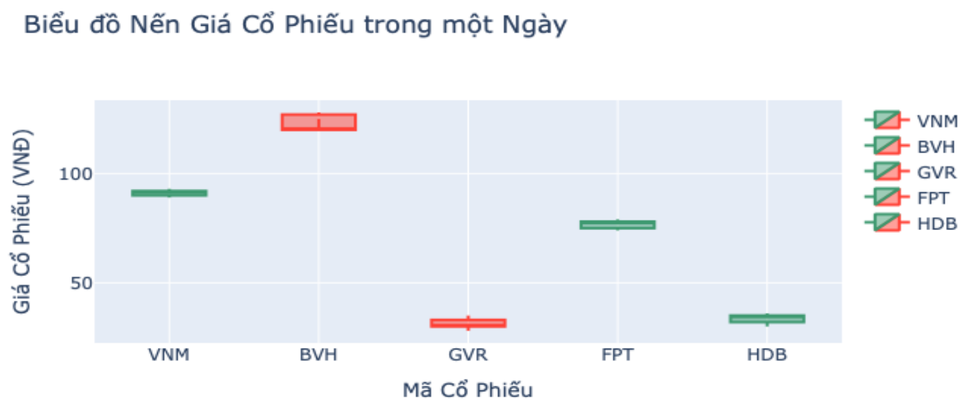
2.7. Trực quan hoá các dữ liệu kinh doanh

- Trong phân tích kinh doanh, trực quan hóa giúp theo dõi và so sánh các KPI (Key Performance Indicators), phân tích doanh số, lợi nhuận, chi phí, và nhận ra các yếu tố ảnh hưởng chính.

- Hỗ trợ các nhà phân tích ra quyết định nhanh chóng và chính xác từ các biểu đồ dễ hiểu.

Ví dụ về trực quan hóa dữ liệu chứng khoán:

```
import pandas as pd
import numpy as np
import plotly.graph_objects as go
# Tạo dữ liệu giả định cho 5 mã cổ phiếu trong một ngày
data = {
    'Stock': ['VNM', 'BVH', 'GVR', 'FPT', 'HDB'],
    'Open': [90, 127, 33, 75, 32],
    'Close': [92, 120, 30, 78, 35],
    'High': [93, 128, 35, 79, 36],
    'Low': [89, 125, 28, 74, 30],
}
# Chuyển dữ liệu thành DataFrame
df = pd.DataFrame(data)
```




```
# Tạo biểu đồ nến
fig = go.Figure()
for index, row in df.iterrows():
    fig.add_trace(go.Candlestick(
        x=[row['Stock']],
        open=[row['Open']],
        high=[row['High']],
        low=[row['Low']],
        close=[row['Close']],
        name=row['Stock']
    ))
# Cập nhật tiêu đề và các thông số khác
fig.update_layout(
    title='Biểu đồ Nến Giá Cổ Phiếu trong một Ngày',
    xaxis_title='Mã Cổ Phiếu',
    yaxis_title='Giá Cổ Phiếu (VNĐ)',
    xaxis_rangeslider_visible=False  # Ẩn thanh trượt
)
# Hiển thị biểu đồ
fig.show()
```

Kết luận:

Việc sử dụng các công cụ trực quan hóa như Matplotlib, Seaborn, Plotly, hoặc biểu đồ địa lý giúp phân tích và trình bày dữ liệu chính xác, dễ hiểu và hiệu quả.

Hy vọng những ví dụ trên giúp bạn nắm bắt được cách sử dụng chúng trong các dự án trực quan hóa dữ liệu.