TRƯỜNG ĐẠI HỌC NGÂN HÀNG
THÀNH PHỐ HỒ CHÍ MINH

KHOA HỌC DỮ LIỆU CHO TÀI CHÍNH
DATA SCIENCE FOR FINANCE

GV: TS. Đặng Đình Thuận
EMAIL: thuandd@hub.edu.vn

TRƯỜNG ĐẠI HỌC NGÂN HÀNG
THÀNH PHỐ HỒ CHÍ MINH

NỘI DUNG MÔN HỌC

- **CHƯƠNG 1. XỬ LÝ DỮ LIỆU VỚI PANDAS**
- **CHƯƠNG 2. TRỰC QUAN HOÁ DỮ LIỆU (Data Visualization)**
- **CHƯƠNG 3. HỌC MÁY CƠ BẢN (MACHINE LEARNING)**
- **CHƯƠNG 4. XỬ LÝ DỮ LIỆU CHUỖI THỜI GIAN, DỮ LIỆU BẢNG**

Tài Liệu Tham Khảo

[1] Ramesh Sharda, Dursun Delen, Efraim Turban (2020). Analytics, data science, & artificial intelligence: Systems for decision support, Pearson

[2] Page, Scott E., (2021). The model thinker: what you need to know to make data work for you, Basic Books.

[3] Cady, Field, (2017). The data science handbook. Wiley

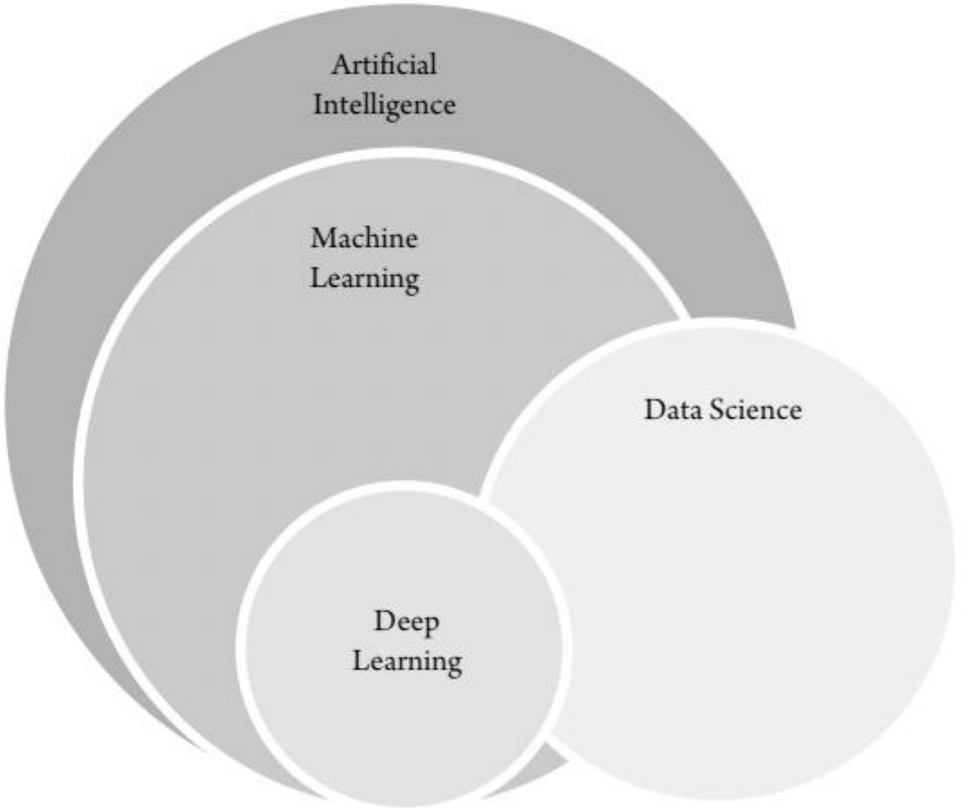
HÌNH THỨC KIỂM TRA ĐÁNH GIÁ

- Đánh giá tính chuyên cần: điểm danh, phát biểu,... **10%**
- Đánh giá quá trình 1(học xong chương 3): Bài kiểm tra cá nhân (thực hành lập trình, or trực tuyến) **20%**
- Đánh giá quá trình 2: Bài tập lớn theo nhóm (bài báo cáo và chương trình máy tính) **20%**
- Bài đánh giá cuối kỳ (Tiểu luận/đồ án/bài tập lớn theo nhóm): dự án phân tích dữ liệu trong kinh doanh ứng dụng các kiến thức đã học **50%**

Giới thiệu

❖ **Trí tuệ nhân tạo (AI)**

- AI là một công nghệ có thể sử dụng để tạo ra các hệ thống thông minh mô phỏng trí thông minh của con người.
- Nói một cách đơn giản, AI là một mô hình hoặc thuật toán có thể cung cấp cho máy móc khả năng bắt chước, học hỏi và minh họa hành vi của con người.
- Hệ thống AI có thể thực hiện các kỹ năng như học tập, lập kế hoạch, đặt mục tiêu, ra quyết định, lập luận và giải quyết vấn đề trong thế giới kinh doanh.
- AI bao hàm khái niệm rộng hơn về việc phát triển các máy móc thông minh mô phỏng khả năng nhận thức và hành động của con người.
- ML là một ứng dụng hoặc tập hợp con cụ thể, cho phép máy móc học hỏi từ dữ liệu mà không cần lập trình rõ ràng.



Giới thiệu

❖ **Học máy**

Học máy là về việc đào tạo máy móc trên dữ liệu lịch sử để xử lý các đầu vào mới dựa trên các mẫu đã học mà không cần lập trình rõ ràng.

Nó bao gồm hồi quy, phân loại, phân cụm và khai thác liên kết.

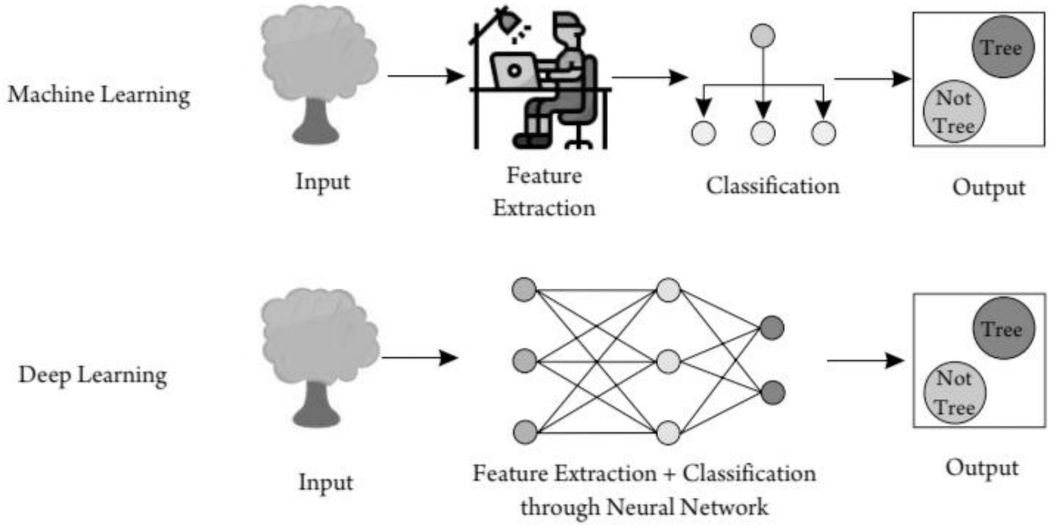
❖ **Học sâu**

Học sâu là một nhánh của ML sử dụng các thuật toán phức tạp của mạng nơ-ron sâu lấy cảm hứng từ hoạt động của não bộ con người.

Cần một lượng dữ liệu khổng lồ để huấn luyện, và nó có thể rút ra những hiểu biết sâu sắc từ dữ liệu đầu vào mà không cần được cho biết về các đặc điểm của dữ liệu.

Tự động trích xuất các đặc điểm mong muốn từ dữ liệu đầu vào để đưa ra dự đoán chính xác. Đồng thời, trong ML truyền thống, nhà thiết kế phải chỉ định các đặc điểm dữ liệu.

các thuật toán ML truyền thống, kỹ sư ML phải thực hiện trích xuất đặc điểm theo cách thủ công, trong khi trong DL, việc thiết kế đặc điểm được thực hiện tự động thông qua mạng nơ-ron.



Giới thiệu

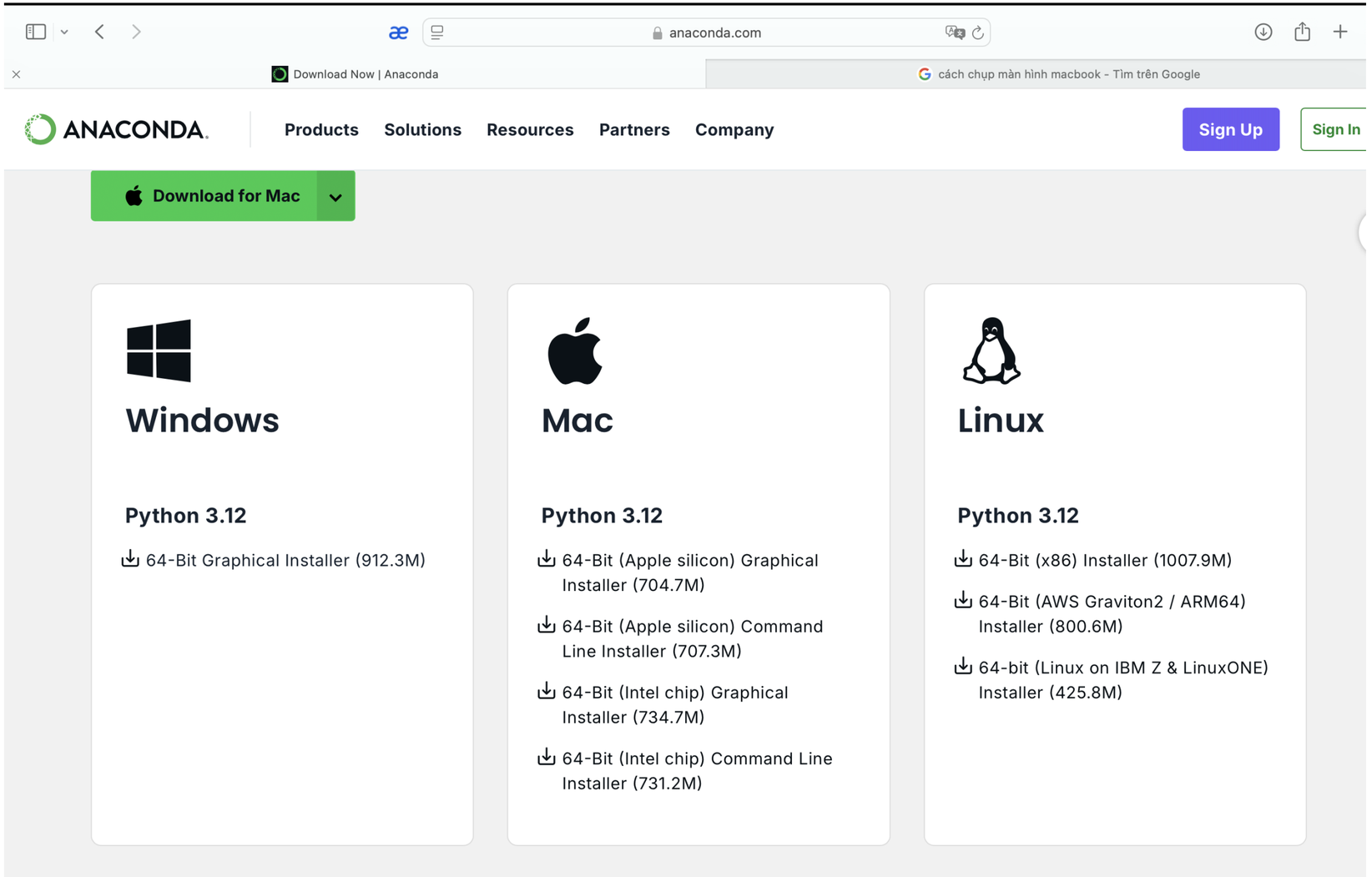
- **Khoa học Dữ liệu**

Khoa học dữ liệu là một ngành khoa học tập trung vào việc khai thác dữ liệu, dựa trên việc khám phá thông tin từ dữ liệu, và liên quan đến việc thu thập, chuẩn bị và phân tích dữ liệu để tạo ra nhiều thông tin chi tiết khác nhau từ dữ liệu.

Mục tiêu chính của khoa học dữ liệu là tìm kiếm ý nghĩa từ dữ liệu, khám phá những vấn đề chưa từng biết đến và giải quyết các vấn đề phức tạp.

Cài đặt Python sử dụng Anaconda

Vào trang chủ: https://www.anaconda.com/download/success



CHƯƠNG 1: XỬ LÝ DỮ LIỆU VỚI PANDAS

1.1 Giới thiệu

1.2. Cấu trúc dữ liệu trong pandas

1.3. Dữ liệu chuỗi

1.4. Làm việc với khung dữ liệu

1.5. Các hàm cơ bản trong pandas

1.6. Thống kê dùng pandas

1.7. Xử lý dữ liệu khuyết

1.8. Đánh chỉ mục

1.9. Thao tác dữ liệu

1.10. Trộn và gôp dữ liệu

1.11. Làm sạch và chuyển đổi dữ liệu

1.12. Thu gom và nhóm dữ liệu

1.13. Một số tác vụ trong pandas

1.14 Pandas ứng dụng cho phân tích dữ liệu kinh doanh

CHƯƠNG 1: XỬ LÝ DỮ LIỆU VỚI PANDAS

1.1. Giới thiệu

- Pandas là một thư viện Python mạnh mẽ, hỗ trợ việc thao tác và phân tích dữ liệu dạng bảng (Panel data), xử lý tập dữ liệu lớn và phức tạp.
- Pandas phổ biến trong khoa học dữ liệu và phân tích dữ liệu nhờ vào tính dễ sử dụng, khả năng xử lý dữ liệu tốt và hiệu suất cao.
- Cài đặt thư viện Pandas: *!pip install pandas*
- Cài đặt thu viện Matplotlib: *!pip install Matplotlib*

1.2. Cấu trúc dữ liệu trong Pandas

- Pandas cung cấp hai cấu trúc chính: Series (một chiều, giống danh sách) và DataFrame (hai chiều, giống bảng Excel).
- Hai cấu trúc này giúp dễ dàng lưu và thao tác dữ liệu.

Ví dụ:

```
#!pip install pandas
import pandas as pd
#    Series
s = pd.Series([1,2, 3, 4])
print("Series:")
print(s)

#    DataFrame
df = pd.DataFrame({'A': [1,2], 'B': [3, 4]})
print("DataFrame:")
print(df)
```

Kết quả:

```
Series:
0   1
1   2
2   3
3   4
DataFrame:
   A B
0  1 3
1  2 4
```

1.3. Dữ liệu chuỗi

- Pandas cung cấp các phương thức để thao tác và xử lý dữ liệu chuỗi (string) trong DataFrame hoặc Series.
- Rất hữu ích khi xử lý dữ liệu, ví dụ như làm sạch, tách chuỗi, thay thế hoặc chuyển đổi chuỗi.

Ví dụ:

```
#     Xử lý chuỗi
data = {'Name': ['Alice', 'Bob', 'Charlie’]}

df = pd.DataFrame(data)

#     Chuyển tất cả tên sang chữ hoa
df['Name_UPPER'] = df['Name'].str.upper()
print(df)
```

Kết quả:

	Name	Name_UPPER
0	Alice	ALICE
1	Bob	BOB
2	Charlie	CHARLIE

1.4. Làm việc với khung dữ liệu

- Khung dữ liệu (DataFrame) là cấu trúc chính trong pandas, dùng để lưu trữ và thao tác dữ liệu hai chiều.
- DataFrame tổ chức dữ liệu dạng bảng, giúp tương tác với dữ liệu một cách hiệu quả.

Ví dụ: Chúng ta có dữ liệu về doanh số bán hàng của một cửa hàng trong một tháng, bao gồm các thông tin như mã sản phẩm, tên sản phẩm, số lượng bán ra và doanh thu.

CHƯƠNG 1: XỬ LÝ DỮ LIỆU VỚI PANDAS

- **Tạo Khung Dữ liệu**

Chúng ta sẽ tạo một DataFrame chứa dữ liệu này.

```
import pandas as pd
#    Dữ liệu bán hàng
data = {
'Product_ID': [101, 102, 103, 104],
'Product_Name': ['Laptop', 'Smartphone', 'Tablet', 'Headphones’],
'Quantity_Sold': [10, 50, 30, 20],
'Revenue': [10000, 25000, 9000, 4000]}
#    Tạo DataFrame
sales_df = pd.DataFrame(data)

#    Hiển thị DataFrame
print(sales_df)
```

Kết quả: Khi bạn chạy đoạn mã trên, bạn sẽ nhận được kết quả như sau:

	Product_ID	Product_Name	Quantity_Sold	Revenue
0	101	Laptop	10	10000
1	102	Smartphone	50	25000
2	103	Tablet	30	9000
3	104	Headphones	20	4000

- **Thao tác trên Khung Dữ liệu**

Bây giờ, thực hiện một số thao tác phổ biến với khung dữ liệu để phân tích dữ liệu bán hàng.

```
#Tính tổng doanh thu
total_revenue = sales_df['Revenue'].sum()
print("Tổng doanh thu:", total_revenue)
#Tính số lượng sản phẩm bán ra
total_quantity_sold = sales_df['Quantity_Sold'].sum()
print("Tổng số lượng sản phẩm bán ra:", total_quantity_sold)
#Lọc các sản phẩm có doanh thu lớn hơn 10.000
high_revenue_products = sales_df[sales_df['Revenue'] > 10000]
print("Sản phẩm có doanh thu lớn hơn 10,000:")
print(high_revenue_products)
```

Kết quả:

```
Tổng doanh thu: 48000
Tổng số lượng sản phẩm bán ra: 110
Sản phẩm có doanh thu lớn hơn 10,000:
   Product_ID Product_Name  Quantity_Sold  Revenue
1         102    Smartphone             50    25000
```

1.5. Các hàm cơ bản trong pandas

- Pandas cung cấp các hàm cơ bản để thao tác dữ liệu, như
  - head(): Xem 5 hàng đầu tiên của data
  - tail(): xem 5 dòng cuối
  - info(): xem thông tin data frame
  - describe(): xem thống kê mô tả của data
- Các hàm này giúp hiểu nhanh dữ liệu trong DataFrame trước khi phân tích.

Ví dụ:

Bước 1: Tạo Khung Dữ liệu

```
import pandas as pd

#    Dữ liệu bán hàng
data = {
'Product_ID': [101, 102, 103, 104, 105, 106],
'Product_Name': ['Laptop', 'Smartphone', 'Tablet', 'Headphones', 'Smartwatch', 'Desktop'],
'Quantity_Sold': [10, 50, 30, 20, 15, 8],
'Revenue': [10000, 25000, 9000, 4000, 5000, 12000]
}

#    Tạo DataFrame
sales_df = pd.DataFrame(data)

#    Hiển thị DataFrame

print(sales_df)
```

CHƯƠNG 1: XỬ LÝ DỮ LIỆU VỚI PANDAS

- **Các Hàm Cơ Bản**

1. head(a)

Hàm này cho phép bạn xem a hàng đầu tiên trong DataFrame. Mặc định () nó hiển thị 5 hàng.

```
print("a hàng đầu tiên:")
print(sales_df.head())
```

Kết quả:

5 hàng đầu tiên:

	Product_ID	Product_Name	Quantity_Sold	Revenue
0	101	Laptop	10	10000
1	102	Smartphone	50	25000
2	103	Tablet	30	9000
3	104	Headphones	20	4000
4	105	Smartwatch	15	5000

2. tail(b)

Hàm này cho phép bạn xem b hàng cuối cùng trong DataFrame. Cũng giống như head(), nó mặc định hiển thị 5 hàng cuối.

```
print("5 hàng cuối cùng:")
print(sales_df.tail())
```

Kết quả: 5 hàng cuối cùng

	Product_ID	Product_Name	Quantity_Sold	Revenue
1	102	Smartphone	50.	25000
2	103	Tablet	30.	9000
3	104	Headphones	20.	4000
4	105	Smartwatch	15.	5000
5	106	Desktop	8	12000

3. info()

Hàm này cung cấp thông tin tổng quan về DataFrame, bao gồm số lượng không giống nhau, kiểu dữ liệu của các cột và bộ nhớ sử dụng.

```
print("Thông tin về DataFrame:")
print(sales_df.info())
```

KQ:

Thông tin về DataFrame:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5 Data' columns (total 4 columns):
 # Column Non-Null Count Dtype

0 Product_ID 6 non-null int64
1 Product_Name 6 non-null object
2 Quantity_Sold 6 non-null int64
3 Revenue 6 non-null int64
dtypes: int64(3), object(1) memory usage: 200.0+ bytes None

4. Describe()

Hàm này cung cấp thống kê mô tả cho các cột số trong DataFrame, bao gồm trung bình, độ lệch chuẩn, giá trị tối đa và tối thiểu.

```
print("Thống kê mô tả:")
print(sales_df.describe())
```

Kết quả: Thống kê mô tả:

	Product_ID	Quantity_Sold	
count	6.000000	6.000000	6.000000
mean	103.500000	25.833333	11833.333333
std	1.870829	12.820335	8827.184724
min	101.000000	8.000000	4000.000000
25%	102.250000	12.500000	5450.000000
50%	103.500000	20.000000	10000.000000
75%	104.750000	25.000000	14500.000000
max	106.000000	50.000000	25000.000000

1.6. Thống kê dùng pandas

Các phương pháp để thực hiện các phép toán thống kê trên dữ liệu:

- mean() tính giá trị trung bình của mỗi cột.
- median() tính giá trị trung vị của mỗi cột.
- std() tính độ lệch chuẩn của mỗi cột.
- var() tính phương sai của mỗi cột.
- corr() tính ma trận tương quan giữa các cột

....

1.6. Thống kê dùng pandas

Ví dụ: (python)

```
import pandas as pd

 # Tạo một DataFrame

data = {

'Doanh_thu': [100, 150, 200, 250, 300],

'Chi_phi': [80, 30, 45, 30, 100],

'Loi_nhuan': [20, 30, 40, 50, 60]

}

df = pd.DataFrame(data)
```

```
# Tính toán các giá trị thống kê cơ bản
 mean_values = df.mean() # Giá trị trung
bình
median_values = df.median() # Giá trị
trung vị s
td_values = df.std() # Độ lệch chuẩn
var_values = df.var() # Phương sai
corr_values = df.corr() # Ma trận tương
quan

# In kết quả
print("Giá trị trung bình:\n",
mean_values)
print("\nGiá trị trung vị:\n",
median_values)
print("\nĐộ lệch chuẩn:\n", std_values)
print("\nPhương sai:\n", var_values)
print("\nMa trận tương quan:\n",
corr_values)
```

1.7. Xử lý dữ liệu khuyết

- Dữ liệu khuyết (NaN) xuất hiện khi thông tin bị thiếu,

-> Mô hình AR, MA, trung bình trượt Holt-Winter, Smooth,…

- Pandas giúp phát hiện và xử lý dữ liệu này.

Chú ý:

- isna() được sử dụng để phát hiện các giá trị khuyết trong DataFrame.
- fillna() được sử dụng để điền các giá trị khuyết bằng giá trị trung bình của cột hoặc một giá trị cụ thể.
- dropna() được sử dụng để loại bỏ các hàng chứa giá trị khuyết.

1.7. Xử lý dữ liệu khuyết

Ví dụ:

```
import pandas as pd

import numpy as np

# Tạo một DataFrame

data = {

'Doanh_thu': [100, 150, np.nan,
250, 300], 'Chi_phi': [80,
np.nan, 160, 200, 240],
'Loi_nhuan': [20, 30, 40,
np.nan, 60] }

df = pd.DataFrame(data)
```

```
# Hiển thị DataFrame ban đầu
print("DataFrame ban đầu:\n", df)
# Phát hiện các giá trị khuyết
missing_values = df.isna()
print("\nCác giá trị khuyết:\n",
missing_values)
# Điền các giá trị khuyết bằng giá trị trung
bình của cột
df_filled = df.fillna(df.mean())
print("\nDataFrame sau khi điền giá trị khuyết
bằng giá trị trung bình:\n", df_filled)
# Loại bỏ các hàng chứa giá trị khuyết
df_dropped = df.dropna()
print("\nDataFrame sau khi loại bỏ các hàng
chứa giá trị khuyết:\n", df_dropped)
# Điền các giá trị khuyết bằng giá trị cụ thể
(ví dụ: 0) df_filled_specific = df.fillna(0)
print("\nDataFrame sau khi điền giá trị khuyết
bằng giá trị cụ thể (0):\n",
df_filled_specific)
```

1.8. Đánh chỉ mục

Pandas dùng các chỉ mục để truy cập, tìm kiếm và nhóm dữ liệu.

Hữu ích khi cần xử lý dữ liệu lớn, gộp nhóm hoặc tìm kiếm nhanh.

Chú ý:

- `set_index('Ngày', inplace=True)` được sử dụng để đặt cột 'Ngày' làm chỉ mục của DataFrame.
- `loc[]` được sử dụng để truy cập dữ liệu bằng chỉ mục.
- `df[df['Doanh_thu'] > 200]` được sử dụng để tìm kiếm các hàng có doanh thu lớn hơn 200.
- `groupby()` và `sum()` được sử dụng để nhóm dữ liệu theo tháng và tính toán tổng doanh thu, chi phí và lợi nhuận.

1.8. Đánh chỉ mục

Ví dụ:

```
import pandas as pd

# Tạo một DataFrame mẫu với dữ liệu kinh tế tài chính
data = {
    'Ngày': ['2025-01-01', '2025-01-02', '2025-01-03', '2025-01-04', '2025-
01-05'],
    'Doanh_thu': [100, 150, 200, 250, 300],
    'Chi_phi': [80, 120, 160, 200, 240],
    'Loi_nhuan': [20, 30, 40, 50, 60]
}

df = pd.DataFrame(data)

# Đặt cột 'Ngày' làm chỉ mục
df.set_index('Ngày', inplace=True)

# Hiển thị DataFrame sau khi đặt chỉ mục
print("DataFrame sau khi đặt chỉ mục:\n", df)

# Truy cập dữ liệu bằng chỉ mục
print("\nDữ liệu ngày 2025-01-03:\n", df.loc['2025-01-03'])

# Tìm kiếm dữ liệu theo điều kiện
print("\nCác ngày có doanh thu lớn hơn 200:\n", df[df['Doanh_thu'] > 200])

# Nhóm dữ liệu theo một tiêu chí và tính toán tổng
grouped = df.groupby(df.index.str[:7]).sum()
print("\nTổng doanh thu, chi phí và lợi nhuận theo tháng:\n", grouped)
```

1.9. Thao tác dữ liệu

Ví dụ thao tác chèn:

```
import pandas as pd

# Tạo một DataFrame mẫu với dữ liệu kinh tế tài chính
data = {
    'Ngày': ['2025-01-01', '2025-01-02', '2025-01-03'],
    'Doanh_thu': [100, 150, 200],
    'Chi_phi': [80, 120, 160],
    'Loi_nhuan': [20, 30, 40]
}

df = pd.DataFrame(data)

# Hiển thị DataFrame ban đầu
print("DataFrame ban đầu:\n", df)

# Chèn một hàng dữ liệu mới
new_data = {'Ngày': '2025-01-04', 'Doanh_thu': 250, 'Chi_phi': 200,
'Loi_nhuan': 50}
df = df.append(new_data, ignore_index=True)

# Hiển thị DataFrame sau khi chèn dữ liệu
print("\nDataFrame sau khi chèn dữ liệu:\n", df)
```

1.9. Thao tác dữ liệu

Ví dụ thao tác xoá dữ liệu:

```
# Xóa một hàng dữ liệu theo chỉ mục
df = df.drop(1)

# Hiển thị DataFrame sau khi xóa dữ liệu
print("\nDataFrame sau khi xóa dữ liệu:\n", df)

# Xóa các hàng dữ liệu theo điều kiện
df = df[df['Doanh_thu'] > 100]

# Hiển thị DataFrame sau khi xóa các hàng theo điều kiện
print("\nDataFrame sau khi xóa các hàng có doanh thu <= 100:\n", df)
```

Trong ví dụ này:

- `append()` được sử dụng để chèn một hàng dữ liệu mới vào DataFrame.
- `drop()` được sử dụng để xóa một hàng dữ liệu theo chỉ mục.
- `df[df['Doanh_thu'] > 100]` được sử dụng để xóa các hàng dữ liệu theo điều kiện (trong trường hợp này là doanh thu lớn hơn 100).

1.10. Trộn và gộp dữ liệu

Ví dụ Trộn dữ liệu (Merge):

```
import pandas as pd

# Tạo hai DataFrame mẫu với dữ liệu kinh tế tài chính
data1 = {
    'Ngày': ['2025-01-01', '2025-01-02', '2025-01-03'],
    'Doanh_thu': [100, 150, 200]
}

data2 = {
    'Ngày': ['2025-01-01', '2025-01-02', '2025-01-04'],
    'Chi_phi': [80, 120, 160]
}

df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)

# Trộn hai DataFrame theo cột 'Ngày'
merged_df = pd.merge(df1, df2, on='Ngày', how='outer')

# Hiển thị DataFrame sau khi trộn
print("DataFrame sau khi trộn:\n", merged_df)
```

1.10. Trộn và gộp dữ liệu

Ví dụ Gộp dữ liệu (Concat):

```
# Tạo hai DataFrame mẫu với dữ liệu kinh tế tài chính
data3 = {
    'Ngày': ['2025-01-01', '2025-01-02', '2025-01-03'],
    'Doanh_thu': [100, 150, 200]
}

data4 = {
    'Ngày': ['2025-01-04', '2025-01-05', '2025-01-06'],
    'Doanh_thu': [250, 300, 350]
}

df3 = pd.DataFrame(data3)
df4 = pd.DataFrame(data4)

# Gộp hai DataFrame theo hàng
concat_df = pd.concat([df3, df4], ignore_index=True)

# Hiển thị DataFrame sau khi gộp
print("\nDataFrame sau khi gộp:\n", concat_df)
```

1.11. Làm sạch và chuyển đổi dữ liệu

- Làm sạch dữ liệu (Data Cleaning) là quá trình loại bỏ hoặc sửa chữa các dữ liệu không chính xác, sai định dạng, trùng lặp, không liên quan hoặc không đầy đủ trong tập dữ liệu.

➢ Mục tiêu là đảm bảo dữ liệu nhất quán và chính xác để phân tích và ra quyết định

- Chuyển đổi dữ liệu (Data Transformation) là quá trình biến đổi dữ liệu từ một định dạng hoặc cấu trúc này sang một định dạng hoặc cấu trúc khác để đáp ứng nhu cầu sử dụng, như logarit,..

➢ Quá trình này có thể bao gồm nhiều thao tác như chuẩn hóa, tổng hợp, lọc và tích hợp dữ liệu

1.11. Làm sạch và chuyển đổi dữ liệu

• **Ví dụ về làm sạch dữ liệu:** Giả sử có một DataFrame chứa thông tin về giao dịch tài chính của khách hàng, nhưng dữ liệu có một số lỗi như giá trị thiếu, trùng lặp và không chính xác.

- **Làm sạch dữ liệu**: Trong ví dụ đầu tiên, chúng ta loại bỏ các hàng có giá trị thiếu (`dropna()`) và các hàng trùng lặp (`drop_duplicates()`). Điều này giúp đảm bảo dữ liệu chính xác và nhất quán.

1.11. Làm sạch và chuyển đổi dữ liệu

```
import pandas as pd

# Tạo DataFrame mẫu
data = {
    'Customer_ID': [1, 2, 2, 3, 4, None],
    'Transaction_Amount': [100, 200, 200, None, 500, 600],
    'Transaction_Date': ['2025-01-01', '2025-01-02', '2025-01-02', '2025-01-
03', '2025-01-04', '2025-01-05']
}

df = pd.DataFrame(data)

# Hiển thị DataFrame ban đầu
print("DataFrame ban đầu:")
print(df)

# Loại bỏ các hàng có giá trị thiếu
df_cleaned = df.dropna()

# Loại bỏ các hàng trùng lặp
df_cleaned = df_cleaned.drop_duplicates()

# Hiển thị DataFrame sau khi làm sạch
print("\nDataFrame sau khi làm sạch:")
print(df_cleaned)
```

1.11. Làm sạch và chuyển đổi dữ liệu

- **Ví dụ về chuyển đổi dữ liệu:** Giả sử muốn chuyển đổi dữ liệu về doanh số bán hàng từ nhiều nguồn khác nhau thành một định dạng thống nhất để phân tích .

  - **Chuyển đổi dữ liệu**: Trong ví dụ thứ hai, chúng ta chuyển đổi kiểu dữ liệu của cột `Sales_Amount` từ chuỗi sang số nguyên (`astype(int)`) và chuyển đổi định dạng ngày tháng của cột `Sales_Date` sang định dạng datetime (`pd.to_datetime()`).

1.11. Làm sạch và chuyển đổi dữ liệu

```
# Tạo DataFrame mẫu
sales_data = {
    'Product_ID': [101, 102, 103, 104],
    'Sales_Amount': ['1000', '2000', '3000', '4000'],
    'Sales_Date': ['2025-01-01', '2025-01-02', '2025-01-03', '2025-01-04']
}

df_sales = pd.DataFrame(sales_data)

# Hiển thị DataFrame ban đầu
print("DataFrame ban đầu:")
print(df_sales)

# Chuyển đổi kiểu dữ liệu của cột Sales_Amount từ chuỗi sang số nguyên
df_sales['Sales_Amount'] = df_sales['Sales_Amount'].astype(int)

# Chuyển đổi định dạng ngày tháng
df_sales['Sales_Date'] = pd.to_datetime(df_sales['Sales_Date'])

# Hiển thị DataFrame sau khi chuyển đổi
print("\nDataFrame sau khi chuyển đổi:")
print(df_sales)
```

1.12. Thu thập và nhóm dữ liệu

• **Thu thập dữ liệu (Data Collection)** là quá trình thu thập và lưu trữ dữ liệu từ nhiều nguồn khác nhau để phục vụ cho việc phân tích và ra quyết định

• Các PP thu thập:

-> Khảo sát, phỏng vấn; quan sát; thí nghiệm, … gọi là dữ liệu sơ cấp.

-> Thu thập từ bên thứ 3 gọi là dữ liệu thứ cấp. Ví dụ chỉ số chứng khoán, GDP, CPI,…

1.12. Thu thập và nhóm dữ liệu

• **Nhóm dữ liệu (Data Grouping)** là quá trình sắp xếp và phân loại dữ liệu vào các nhóm hoặc cụm dựa trên các tiêu chí nhất định.

•**->** Mục tiêu là để dễ dàng phân tích và tìm ra các mẫu hoặc xu hướng trong dữ liệu

•Các PP nhóm:

-> Phân cụm (Clustering): các thuật toán như K-means, Gaussian Mixture Model (GMM) sử dụng mô hình xác suất để xác định mật độ phân phối

-> Phân loại (Classification): Sử dụng các thuật toán học máy để phân loại dữ liệu vào các nhóm đã biết trước

-> Phân tích thành phần chính (PCA): Giảm số chiều của dữ liệu và nhóm các đối tượng dữ liệu dựa trên các thành phần chính ….

1.13. Ví dụ tổng hợp trong phân tích dữ liệu kinh doanh

Ví dụ 1: Dự đoán doanh thu trung bình theo các sản phẩm.

```
import pandas as pd
#    Tạo dữ liệu bán hàng
data = {
'Product': ['Laptop', 'Laptop', 'Phone', 'Phone', 'Tablet'], 'Price':
[1200, 1300, 700, 750, 400],
'Units': [3, 4, 5, 6, 8]}
df = pd.DataFrame(data)
#    Tính doanh thu từng hàng
df['Revenue'] = df['Price'] * df['Units']
#    Thống kê doanh thu trung bình theo sản phẩm
summary = df.groupby('Product')['Revenue'].mean()
print(summary)
```

Kết quả:

Doanh thu trung bình mỗi sản phẩm:

```
Product
Laptop    5000.0
Phone     6975.0
Tablet    3200.0
```

Ví dụ 2: Phân tích doanh thu bán hàng

Giả sử bạn có một bảng dữ liệu về doanh thu bán hàng của một công ty trong năm qua. Bạn muốn phân tích doanh thu theo từng tháng và tìm ra tháng có doanh thu cao nhất.

```
import pandas as pd

#    Tạo DataFrame từ dữ liệu doanh thu
data = {
'Tháng': ['Tháng 1', 'Tháng 2', 'Tháng 3', 'Tháng 4', 'Tháng 5',
'Tháng 6', 'Tháng 7', 'Tháng 8', 'Tháng 9', 'Tháng 10', 'Tháng 11',
'Tháng 12'], 'Doanh thu (triệu đồng)': [120, 150, 130, 160, 170, 180,
190, 200, 210, 220, 230, 240]}
df = pd.DataFrame(data)

# Hiển thị thông tin cơ bản về DataFrame

print(df.info())
```

```
#    Tính tổng doanh thu trong năm

total_revenue = df['Doanh thu (triệu đồng)'].sum()

print(f'Tổng doanh thu trong năm: {total_revenue} triệu đồng')

#    Tìm tháng có doanh thu cao nhất
max_revenue_month = df.loc[df['Doanh thu (triệu đồng)'].idxmax()]

print(f'Tháng có doanh thu cao nhất: {max_revenue_month["Tháng"]} với
{max_revenue_month["Doanh thu (triệu đồng)"]} triệu đồng’)


#    Tính doanh thu trung bình mỗi tháng

average_revenue = df['Doanh thu (triệu đồng)'].mean()

print(f'Doanh thu trung bình mỗi tháng: {average_revenue} triệu đồng')
```

```
#  Vẽ biểu đồ doanh thu theo tháng

import matplotlib.pyplot as plt

df.plot(x='Tháng', y='Doanh thu (triệu đồng)', kind='bar',
legend=False)
plt.title('Doanh thu bán hàng theo tháng')
plt.xlabel('Tháng')
plt.ylabel('Doanh thu (triệu đồng)')
plt.show()
```

Kết quả:

- Tổng doanh thu trong năm: 2,100 triệu đồng
- Tháng có doanh thu cao nhất: Tháng 12 với 240 triệu đồng
- Doanh thu trung bình mỗi tháng: 175 triệu đồng

BẢNG HỎI KHẢO SÁT

Nghiên cứu: Các yếu tố ảnh hưởng đến ý định sử dụng giao thông công cộng của người dùng phương tiện cá nhân tại TP. Hồ Chí Minh

Phần A – Thông tin chung

Mã	Câu hỏi	Lựa chọn	Mã hóa biến
A1	Giới tính	☐ Nam ☐ Nữ	Nam=1, Nữ=0
A2	Độ tuổi	☐ <18 ☐ 18–25 ☐ 26–35 ☐ 36–45 ☐ >45	1,2,3,4,5
A3	Thu nhập hàng tháng	☐ <5tr ☐ 5–10tr ☐ 10–15tr ☐ >15tr	1,2,3,4
A4	Nghề nghiệp	☐ HSSV ☐ NVVP ☐ KD ☐ Khác	1,2,3,4
A5	Tần suất sử dụng PTCC	☐ Hằng ngày ☐ Hàng tuần ☐ Ít hơn	1,2,3

BẢNG HỎI KHẢO SÁT

Nghiên cứu: Các yếu tố ảnh hưởng đến ý định sử dụng giao thông công cộng của người dùng phương tiện cá nhân tại TP. Hồ Chí Minh

Phần B – Đánh giá chất lượng dịch vụ PTCC

(Thang đo Likert 5 mức: 1 = Hoàn toàn không đồng ý … 5 = Hoàn toàn đồng ý)

Mã	Câu hỏi	Mã hóa biến
SQ1	Thông tin lộ trình và thời gian đến rõ ràng	1–5
SQ2	Phương tiện sạch sẽ, bảo dưỡng tốt	1–5
SQ3	Nhân viên phục vụ thân thiện	1–5
SQ4	Thời gian chờ đợi hợp lý	1–5
SQ5	Dịch vụ đúng giờ, đáng tin cậy	1–5
SQ6	Giá vé hợp lý so với thu nhập	1–5
SQ7	Kết nối thuận tiện giữa các tuyến và phương tiện khác	1–5
SQ8	Tình trạng giao thông không ảnh hưởng nhiều đến thời gian	1–5

BẢNG HỎI KHẢO SÁT

Nghiên cứu: Các yếu tố ảnh hưởng đến ý định sử dụng giao thông công cộng của người dùng phương tiện cá nhân tại TP. Hồ Chí Minh

Phần C – Mức độ hài lòng tổng thể

Mã	Câu hỏi	Mã hóa biến
SAT1	Tôi hài lòng với chất lượng dịch vụ PTCC hiện tại	1–5
SAT2	Trải nghiệm PTCC đáp ứng mong đợi	1–5

Phần D – Thái độ và ý định sử dụng

Mã	Câu hỏi	Mã hóa biến
ATT1	Tôi tin rằng PTCC là phương án di chuyển tốt cho thành phố	1–5
ATT2	Tôi có cái nhìn tích cực về PTCC	1–5
BI1	Tôi dự định sử dụng PTCC thường xuyên hơn	1–5 (biến phụ thuộc)
BI2	Tôi sẵn sàng giới thiệu PTCC cho người khác	1–5