BÀI TẬP THỰC HÀNH MÔN NHẬP MÔN PHÂN TÍCH DỮ LIỆU VÀ HỌC SÂU

- Bài tập được thiết kế theo từng lab, mỗi lab là 3 tiết có sự hướng dẫn của GV.
- Cuối mỗi buổi thực hành, sinh viên nộp lại phần bài tập mình đã thực hiện cho GV hướng dẫn.
- Những câu hỏi mở rộng/khó giúp sinh viên trau dồi thêm kiến thức của môn học. Sinh viên phải có trách nhiệm nghiên cứu, tìm câu trả lời nếu chưa thực hiện xong trong giờ thực hành.

Nội dung

LAB 1:

BÀI THỰC HÀNH THAO TÁC DỮ LIỆU

Nội dung: Thao tác dữ liệu điểm thi đại học của học sinh được cho bởi bảng bên dưới

Mục tiêu: Sinh viên đạt được kiến thức sau

1. Tìm hiểu nghiệp vụ dữ liệu
2. Nhập liệu bằng công cụ từ file excel
3. Xác định dữ liệu định tính và định lượng
4. Hiệu chỉnh các thang đo phù hợp và kiểu giá trị dữ liệu cho từng biến số
5. Hiệu chỉnh dữ liệu và xử lý dữ liệu thiếu
6. Chuyển đổi (transformation) dữ liệu theo khoảng cho trước
7. Tạo biến số phụ thuộc theo biến độc lập
8. Tạo biến định tính phân loại

Dữ liệu **dulieuxettuyendaihoc.csv** được mô tả như sau

Dữ liệu lưu trữ điểm trung bình môn, khu vực, khối thi và điểm thi đại học của 100 học sinh.

- T1, L1, H1, S1, V1, X1, D1, N1 lần lượt là điểm trung bình các môn Toán, Lý ,Hóa, Sinh, Văn, Sử, Địa, Ngoại ngữ năm lớp 10
- T2, L2, H2, S2, V2, X2, D2, N2 lần lượt là điểm trung bình các môn Toán, Lý ,Hóa, Sinh, Văn, Sử, Địa, Ngoại ngữ năm lớp 11
- T6, L6, H6, S6, V6, X6, D6, N6 lần lượt là điểm trung bình các môn Toán, Lý ,Hóa, Sinh, Văn, Sử, Địa, Ngoại ngữ năm lớp 12
- GT: Giới tính
- DT: Dân tộc
- KV, KT lần lượt là khu vực thi và khối thi
- DH1, DH2, DH3 lần lượt là điểm thi đại học môn 1, môn 2, môn 3

Sử dụng Pandas để thực hiện các yêu cầu sau đây

1. Xác định và phân loại dữ liệu định tính và định lượng
2. Định nghĩa các thang đo phù hợp cho từng biến số
3. Sử dụng Python để tải dữ liệu lên chương trình và in ra màn hình 10 dòng đầu tiên và 10 dòng cuối cùng
4. Thống kê dữ liệu thiếu cho cột dân tộc và hiệu chỉnh dữ liệu thiếu như sau: Mặc định thiếu thì điền giá trị 0.

Hướng dẫn

1. Lập bảng tần số, tần suất để khảo sát dữ liệu thiếu, bao nhiêu dữ liệu riêng biệt (pandas unique)
2. Thực hiện thay thế dữ liệu thiếu bằng phương pháp điền dữ liệu 0

5. Thống kê dữ liệu thiếu cho biến T1 và hiệu chỉnh dữ liệu, lưu ý việc thay thế dữ liệu thiếu sử dụng phương pháp Mean.

Hướng dẫn

1. Lập bảng tần số, tần suất để khảo sát dữ liệu thiếu

2. Thực hiện thay thế dữ liệu thiếu bằng phương pháp Mean

6. Hãy thực hiện xử lý lần lượt tất cả dữ liệu thiếu cho các biến về điểm số còn lại.
7. Tạo các biến TBM1, TBM2, TBM3 tương ứng với trung bình môn của các năm lớp 10, 11 và 12.
    - Công thức tính: TBM = (T*2 + L + H + S + V*2 + X + D + N) / 10
8. Tạo các biến xếp loại XL1, XL2 và XL3 dựa trên TBM1,TBM2 và TBM3 cho từng năm lớp 10, 11, 12 như sau:
    - Nhỏ hơn 5.0 xếp loại: yếu (kí hiệu là Y)
    - Từ 5.0 đến dưới 6.5: trung bình (kí hiệu là TB)
    - Từ 6.5 đến dưới 8.0: khá (kí hiệu là K)
    - Từ 8.0 đến dưới 9.0: giỏi (kí hiệu là G)
    - Từ 9.0 trở lên: xuất sắc (kí hiệu là XS)
9. Tạo các biến US_TBM1, US_TBM2 và US_TBM3 để chuyển điểm trung bình các năm lớp 10, 11 và 12 từ thang điểm 10 của Việt Nam sang thang điểm 4 của Mỹ. Sử dụng phương pháp Min-Max Normalization
10. Tạo biến kết quả xét tuyển (kí hiệu là KQXT) nhằm xác định sinh viên đậu (giá trị "1") và rớt (giá trị "0") vào các khối dựa trên điểm DH1, DH2 và DH3 như sau
    - Với khối A, A1 nếu [(DH1*2 + DH2 + DH3)/4] lớn hơn hoặc bằng 5.0 thì đậu, ngược lại là rớt
    - Với khối B nếu [(DH1 + DH2*2 + DH3)/4] lớn hơn hoặc bằng 5.0 thì đậu, ngược lại là rớt
    - Với khối khác nếu [(DH1+ DH2 + DH3)/3] lớn hơn hoặc bằng 5.0 thì đậu, ngược lại là rớt
11. Lưu trữ dữ liệu xuống ổ đĩa thành file **processed_dulieuxettuyendaihoc.csv**

LAB 2:

BÀI THỰC HÀNH TRÌNH BÀY DỮ LIỆU

Nội dung: Trực quan hóa dữ liệu điểm thi đã được xử lý **processed_dulieuxettuyendaihoc.csv**
Mục tiêu: Sinh viên đạt được kiến thức sau.

1. Trình bày dữ liệu cơ bản
2. Trực quan hóa dữ liệu cơ bản

Phần 1: Thống kê dữ liệu

1. Hãy sắp xếp dữ liệu điểm DH1 theo thứ tự tăng dần
2. Hãy sắp xếp dữ liệu điểm DH2 tăng dần theo nhóm giới tính
3. Hãy tạo pivot-table để thống kê các giá trị count, sum, mean, median, min, max, std, Q1, Q2 và Q3 của DH1 theo KT
4. Hãy tạo pivot-table để thống kê các giá trị count, sum, mean, median, min, max, std, Q1, Q2 và Q3 của DH1 theo KT và KV
5. Hãy tạo pivot-table để thống kê các giá trị count, sum, mean, median, min, max, std, Q1, Q2 và Q3 của DH1 theo KT, KV và DT

Phần 2: Trình bày dữ liệu

1. Hãy trình bày dữ liệu biến: GT

Gợi ý

- Lập bảng tần số và tần suất
- Vẽ biểu đồ tần số (cột), biểu đồ tần suất (tròn).

2. Hãy trình bày dữ liệu lần lượt các biến: **US_TBM1, US_TBM2 và US_TBM3**
3. Hãy trình bày dữ liệu biến DT với các học sinh là nam
4. Hãy trình bày dữ liệu biến KV với các học sinh là nam thuộc dân tộc Kinh, có điểm thỏa mãn điều kiện (DH1 >= 5.0 và DH2 >= 4.0 và DH3 >= 4.0)
5. Hãy trình bày dữ liệu lần lượt các biến DH1, DH2, DH3 lớn hơn bằng 5.0 và thuộc khu vực 2NT

Phần 3: Trực quan hóa dữ liệu theo nhóm phân loại

1. Trực quan dữ liệu học sinh nữ trên các nhóm XL1, XL2, XL3 dạng unstacked

Gợi ý

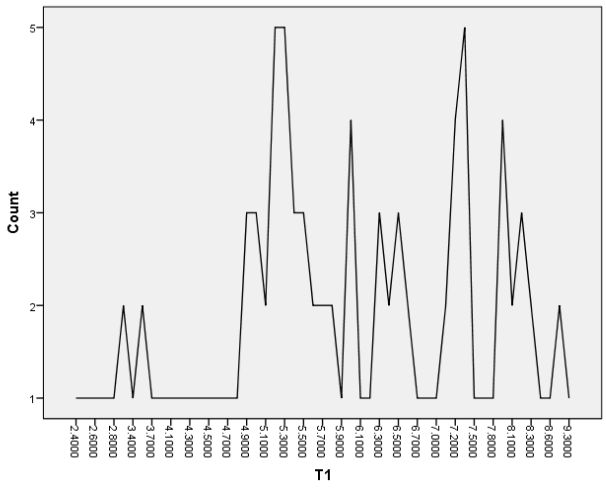
- Lọc dữ liệu giới tính là nữ
- Oy: Chiều cao biểu đồ cột thể hiển số lượng học sinh theo xếp loại
- Màu sắc thể hiện giá trị xếp loại: [Y, TB, K, G, XS]
- Ox: thể hiện nhóm XL1, XL2 và XL3

2. Trực quan dữ liệu KQXT trên nhóm học sinh có khối thi A, A1, B thuộc khu vực 1, 2
3. Trực quan dữ liệu số lượng thí sinh từng khu vực dựa trên từng nhóm khối thi
4. Trực quan dữ liệu số lượng thí sinh đậu, rớt trên từng nhóm khối thi
5. Trực quan dữ liệu số lượng thí sinh đậu rớt trên từng nhóm khu vực.
6. Trực quan dữ liệu số lượng thí sinh đậu rớt dựa trên từng nhóm dân tộc
7. Trực quan dữ liệu số lượng thí sinh đậu rớt dựa trên từng nhóm giới tính.

Phần 4: Trực quan hóa dữ liệu nâng cao

1. Vẽ biểu đồ đường Simple cho biến T1

Kết quả

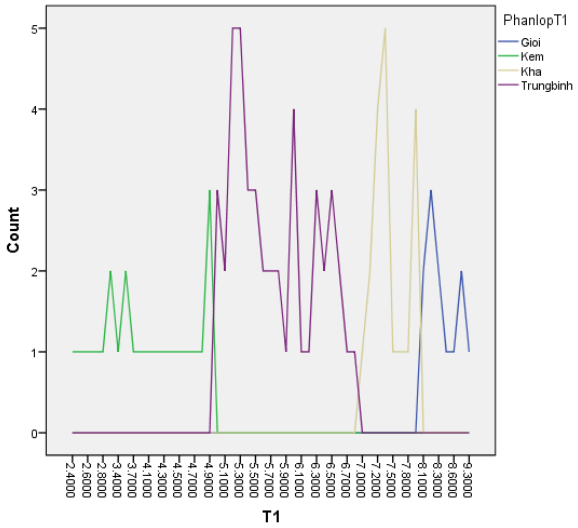


2. Hãy tạo biến phân loại (phanlopt1) cho môn toán (T1) như sau:
    a. Từ 0 đến dưới 5 = kém (ký hiệu "k")
    b. Từ 5 đến dưới 7 = trung bình (ký hiệu "tb")

c. Từ 7 đến dưới 8 = khá (ký hiệu "k")
d. Từ 8 trở lên = giỏi (ký hiệu "g")

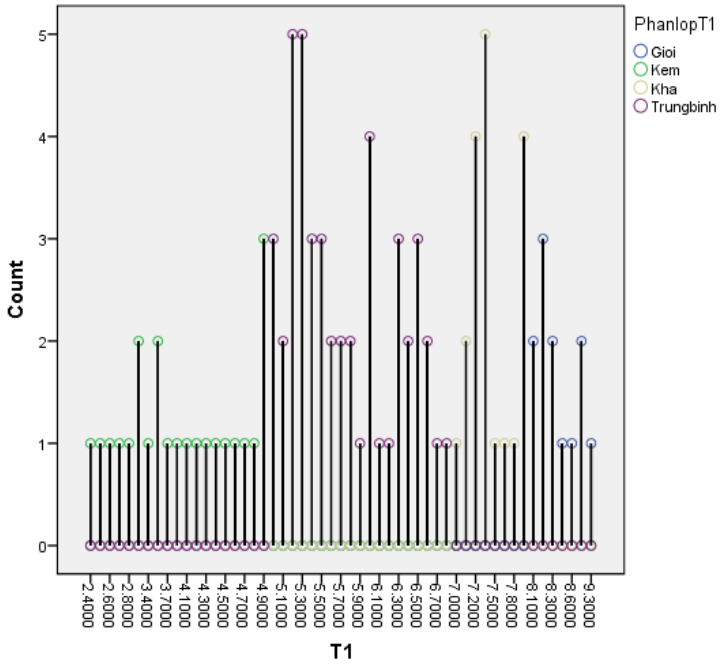
3. Lập bảng tần số cho biến phanloait1
4. Vẽ biểu đồ đường Multiple Line cho biến T1 được phân loại bởi biến phanlopt1

Kết quả



5. Vẽ biểu đồ Drop-line cho biến T1 được phân loại bởi biến phanlopt1

Kết quả



Phần 5: Mô tả dữ liệu và khảo sát dạng phân phối

1. Hãy mô tả và khảo sát phân phối cho biến T1

Gợi ý

- Mô tả độ tập trung và phân tán của dữ liệu T1
- Vẽ biểu đồ Box-Plot và xác định các 10 đại lượng trong biểu đồ đó

- Mô tả hình dáng lệch của phân phối T1 dựa vào các đại lượng hướng tâm
- Vẽ biểu đồ Histogram biểu thị hình dáng phân phối
- Mô tả các đặc trưng của phân phối, mức độ lệch và mức độ nhọn
- Kiểm chứng phân phối chuẩn QQ-Plot
- Nhận xét và đánh giá về phân phối của T1

2. Hãy mô tả và khảo sát phân phối cho biến T1 trên từng nhóm phân lớp (phanlopT1)

Gợi ý

- Trực quan hóa biểu đồ Box-plot, histogram và QQ-plot theo phân nhóm là giá trị của 'phanlopT1'.

3. Hãy khảo sát tương quan giữa biến DH1 theo biến T1

Gợi ý

- Nhận xét giá trị Covariance hoặc Correlation
- Vẽ biểu đồ Scatter thể hiện liên hệ của biến phụ thuộc DH1 theo biến độc lập T1

4. Hãy khảo sát tương quan giữa biến DH1 theo biến T1 trên từng nhóm khu vực
5. Hãy khảo sát tương quan giữa các biến DH1, DH2, DH3

Gợi ý

- Nhận xét ma trận hiệp phương sai hoặc ma trận tương quan
- Vẽ biểu đồ Scatter giữa các biến

LAB 3:

LÀM SẠCH DỮ LIỆU CƠ BẢN

Nội dung: Xử lý dữ liệu y khoa về huyết áp của bệnh nhân

Mục tiêu: Sinh viên biết cách sử dụng gói Pandas để xử lý dữ liệu

1. Tiến hành hiểu dữ liệu từ chuyên gia

   "*The data set has been kept small enough for you to be able to grok it all at once. The data is in csv format. Each row in the dataset has data about different individuals and their heart rate details for different time intervals. The columns contain information such as individual's Age, Weight, Sex and Heart Rates taken at different time intervals.*"

2. Thông thường ta thường xử lý các vấn đề sau về dữ liệu
   1. Thiếu dòng tiêu đề ở file csv
   2. Nhiều biến lưu ở một cột
   3. Dữ liệu cột chứa các giá trị đơn vị không nhất quán
   4. Dữ liệu có một dòng trống
   5. Dữ liệu có các dòng trùng lặp
   6. Các ký tự không phải ASCII
   7. Giá trị bị mất
   8. Tiêu đề cột là giá trị chứ không phải tên biến
3. ***Vấn đề 1:*** Tiến hành tải dữ liệu vào chương trình ứng dụng Python và giải quyết vấn đề "Missing header in the csv file"

```
#Problem 1
# Thêm header vào dataframe để diễn giải dữ liệu
column_names= ["Id", "Name", "Age", "Weight",'m0006','m0612','m1218','f0006','f0612','f1218']
# Đọc file dữ liệu
df = pd.read_csv("patient_heart_rate.csv", names = column_names)
#Hiển thị một vài dòng dữ liệu đầu tiên ra màn hình
print(df.head())
```

4. ***Vấn đề 2:*** Xử lý vấn đề một cột lưu hỗn hợp nhiều dữ liệu, ở đây là cột "Name" chứa bao gồm "Firstname" và "Lastname", giải pháp là ta sẽ tách ra làm 2 cột

```
#Problem 2
df[['Firstname','Lastname']] = df['Name'].str.split(expand=True)
df = df.drop('Name', axis=1)
print (df)
```

5. ***Vấn đề 3:*** Cột Weight có vấn đề về không thống nhất các đơn vị đo lường trong dữ liệu. Ta sẽ chuyển các đơn vị về thành đơn vị chuẩn "kg"

```
# Problem 3
#Get the Weight column
weight = df['Weight']

for i in range (0 ,len(weight)):
    x= str(weight[i])
    #Incase lbs is part of observation remove it
    if "lbs" in x[-3:]:
        #Remove the lbs from the value
        x = x[:-3:]
        #Convert string to float
        float_x = float(x)
        #Covert to kgs and store as int
        y =int(float_x/2.2)
        #Convert back to string
        y = str(y)+"kgs"
        weight[i]= y
print (df)
```

6. ***Vấn đề 4:*** Vấn đề về xuất hiện dòng dữ liệu rỗng (không có giá trị: NaN). Giải pháp có thể đưa ra là xóa bỏ

```
# Problem 4:
df.dropna(how="all", inplace=True)
print(df)
```

7. ***Vấn đề 5:*** Có nhiều dòng dữ liệu bị trùng lắp thông tin hoàn toàn[fullname, lastname, age, weight,....], giải pháp đưa ra là chỉ giữ lại một dòng dữ liệu, tuy nhiên giải pháp phải dựa trên nghiệp vụ của tập dữ liệu và quan sát của người xử lý.

```
df = df.drop_duplicates(subset=['Firstname','Lastname', 'Age', 'Weight'])
print (df)
```

8. ***Vấn đề 6:*** Xuất hiện dữ liệu bị ảnh hưởng bởi lỗi non-ASCII, không định dạng ASCII. Giải pháp: Tùy vào nghiệp vụ ta có thể: xóa dữ liệu tại đó, thay thế bằng dữ liệu khác hoặc thay bằng việc đánh dấu bằng một kí tự khác (ví dụ: 'warning')

```
#Problem 6:
df.Firstname.replace({r'[^\x00-\x7F]+':''}, regex=True, inplace=True)
df.Lastname.replace({r'[^\x00-\x7F]+':''}, regex=True, inplace=True)
print (df)
```

9. ***Vấn đề 7:*** "Missing values", vấn đề này xảy ra tại các cột "Age", "Weight" và "Heart Rate". Thiếu dữ liệu (dữ liệu không đầy đủ) là vấn đề xảy ra nhiều trong các nguồn dữ liệu do nhiều nguyên nhân chủ quan lẫn khách quan.Có một vài giải pháp để xử lý vấn đề này, chủ yếu dựa trên kinh nghiệm và nghiệp vụ về tập dữ liệu đó. Một số giải pháp đưa đề xuất từ chuyên gia như sau:
    a. **Deletion**: Remove records with missing values
    b. **Dummy substitution**: Replace missing values with a dummy but valid value: e.g.: 0 for numerical values.
    c. **Mean substitution**: Replace the missing values with the mean.
    d. **Frequent substitution**: Replace the missing values with the most frequent item.
    e. **Improve the data collector**: Your business folk will talk to the clients and inform them about why it is worth fixing the problem with the data collector.

    ……

    Yêu cầu:

    - Thống kê thông tin dữ liệu thiếu trên từng biến Age và Weight
    - Yêu cầu xử lý dữ liệu thiếu như sau: Nếu dòng nào có Age hoặc Weight có dữ liệu thì phần Age hoặc Weight được tính như bên dưới, nếu thiếu cả 2 thông tin thì xóa dòng
        - *Age*: Giá trị thay thế là mean của các giá trị trong cột Age

10. ***Vấn đề 8:*** "một cột chứa quá nhiều thông tin cần được phân rã", như trong bài toán này ta thấy header "m0006" chứa các nội dung bao gồm: m → male, 1218 ~ 12-18 (mm-dd). Còn giá trị thì là kết quả huyết áp.

3	4.0	NaN	78kgs	78	79	72	-	-	-	Scrooge	McDuck
4	5.0	54.0	90kgs	-	-	-	69	NaN	75	Pink	Panther
5	6.0	52.0	85kgs	-	-	-	68	75	72	Huey	McDuck
6	7.0	19.0	56kgs	-	-	-	71	78	75	Dewey	McDuck
7	8.0	32.0	78kgs	78	76	75	-	-	-	Scööpy	Doo

Chúng ta sẽ tách nội dung của cột này ra làm 3 cột sau: PulseRate : giá trị huyết áp, Sex: giới tính (m: male, f: female) và time: thời gian (tháng-ngày) như sau:

	Id	Age	Weight	Firstname	Lastname	PulseRate	Sex	Time
0	1.0	56.0	70kgs	Micky	Mous	72	m	00-06
9	1.0	56.0	70kgs	Micky	Mous	69	m	06-12
18	1.0	56.0	70kgs	Micky	Mous	71	m	12-18
27	1.0	56.0	70kgs	Micky	Mous	-	f	00-06
36	1.0	56.0	70kgs	Micky	Mous	-	f	06-12
45	1.0	56.0	70kgs	Micky	Mous	-	f	12-18

Gợi ý:

```
#Melt the Sex + time range columns in single column
df = pd.melt(df,id_vars=['Id','Age','Weight','Firstname','Lastname'], value_name="PulseRate",var_name="sex_and_time").sort_values(['Id','Age','Weight','Firstname','Lastname'])

# Extract Sex, Hour lower bound and Hour upper bound group
tmp_df = df["sex_and_time"].str.extract("(\D)(\d+)(\d{2})",expand=True)

# Name columns
tmp_df.columns = ["Sex", "hours_lower", "hours_upper"]

# Create Time column based on "hours_lower" and "hours_upper" columns
tmp_df["Time"] = tmp_df["hours_lower"] + "-" + tmp_df["hours_upper"]

# Merge
df = pd.concat([df, tmp_df], axis=1)

# Drop unnecessary columns and rows
df = df.drop(['sex_and_time','hours_lower','hours_upper'], axis=1)
df = df.dropna()
df.to_csv('outputcleanup.csv',index=False)
print (df)
```

11. Hãy khảo sát tỉ lệ dữ liệu thiếu trên biến huyết áp. Dữ liệu bị thiếu thì hãy xử lý bằng phương pháp sau
    - Thay thế bằng giá trị trung bình liền trước và liền sau của người đó. Nếu không được thì dùng 2)
    - Thay thế bằng giá trị trung bình 2 giá liền trước của người đó. Nếu không được thì dùng 3)
    - Thay thế bằng giá trị trung bình 2 giá liền sau của người đó. Nếu không được thì dùng 4)
    - Trung bình của các giá trị huyết áp của người đó. Nếu không được thì dùng 5).
    - Trung bình của các giá trị huyết áp của nhóm giới tính. Nếu không được thì dùng 6)
    - Trung bình của các giá trị dữ liệu. Nếu không được thì thay bằng mức ổn định trong y học.
12. Hãy rút gọn dữ liệu phù hợp và reindex lại dữ liệu. Sau đó, lưu trữ dữ liệu đã xử lý thành công với tên file ***patient_heart_rate_clean.csv***

Lưu ý: Ngoài ra còn rất nhiều vấn đề về mặt xử lý dữ liệu dựa trên nhiều khía cạnh khác nhau tùy vào sự am hiểu về dữ liệu của các chuyên gia như:

- Handling dates
- Correcting character encodings (a problem you hit when you scrape data off the web)

LAB 4:

BÀI THỰC HÀNH CHUẨN BỊ DỮ LIỆU

Nội dung: Chuẩn bị dữ liệu – Data Preparation
Tham khảo: seaborn: statistical data visualization — seaborn 0.11.1 documentation (pydata.org)
Mục tiêu: Sinh viên nắm được các kiến thức sau

1. Data Cleansing
2. Exploration Data Analysis (EDA)
3. Kỹ thuật function chain trong Pandas – pipe()
4. Feature Engineering
5. Data Wrangling

Mô tả dữ liệu: The sinking of the Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew. While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

Variable	Definition	Key
PassengerId	Identifier	
Survived	Survival	0 = No, 1 = Yes
Pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
Name	Passenger name	
Sex	Sex	
Age	Age in years	
SibSp	# of siblings / spouses aboard the Titanic	
Parch	# of parents / children aboard the Titanic	
Ticket	Ticket number	
Fare	Passenger fare	
Cabin	Cabin number	
Embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Yêu cầu: *Hãy chuẩn bị dữ liệu phục vụ cho bài toán: "Xây dựng mô hình dự báo nhóm hành khách có khả năng sống sót với các thông số đầu vào là các đặc trưng của hành khách (name, age, gender, socio-economic class, ...), trong sự kiện Titanic lịch sử"*

PHẦN 1: DATA CLEANSING & FEATURE ENGINEERING

Hướng dẫn

1. Viết hàm load_data() để tải dữ liệu lên ứng dụng. Sau đó, hiển thị ra màn hình 10 dòng đầu tiên.
2. Thống kê dữ liệu thiếu trên các biến số và trực quan hóa dữ liệu thiếu bằng biểu đồ (Heat map). Hãy cho nhận xét về tình trạng thiếu dữ liệu Age, Cabin và Embarked
3. Xử lý tên cột tên Name, tách ra làm 2 cột: firstName và secondName. Lưu ý: Sau khi tách cột xong thì xóa luôn cột Name
4. Xứ lý rút gọn kích thước dữ liệu trên cột Sex như sau: thay thế male → M và female → F
5. Xử lý dữ liệu thiếu trên biến Age bằng cách thay thế bằng giá trị trung bình tuổi: Hãy đưa ra quyết định dùng giá trị trung bình tuổi toàn bộ hành khách hay theo từng nhóm hạng vé (hạng hành khách: Pclass). Ta tiến hành làm các bước sau

a. Sử dụng Seaborn để vẽ biểu đồ (Box plot) trực quan dữ liệu để xác định phân phối tuổi trên từng hạng hành khách. Nhận xét về tuổi trung bình giữa các nhóm hành khách. Từ đó đưa ra quyết định cách thay thế giá trị tuổi bị thiếu.
b. Tiến hành thay thế giá trị Age bị thiếu. Sau đó, hiển thị kết quả dạng bảng và trực quan dữ liệu đã xử lý thiếu cho cột'Age' bằng biểu đồ Heat map.

6. Xây dựng biến số Agegroup có thang đo thứ tự được ánh xạ theo thang đo khoảng dựa trên độ tuổi của hành khách như sau: (age =< 12] → Kid; (12, 18]: Teen, (18, 60]: Adult và (age > 60): Older
7. Tiến hành thêm đặc trưng về danh xưng (namePrefix) trong xã hội bằng cách tách Mr, Mrs, Miss, Master ra khỏi "secondName"
8. Khai thác thêm thông tin số lượng thành viên đi theo nhóm thân quen (familySize) đối với mỗi hành khách trên chuyến hải trình; family size = 1+ SibSp + Parch
9. Tạo thêm đặc trưng 'Alone' để xác định hành khách đi theo nhóm hay cá nhân bằng cách dựa trên familySize như sau: Nếu familySize = 0 thì giá trị Alone = 1 và ngược lại là 0.
10. Tiến hành tách loại cabin (typeCabin) mà hành khách ở để lọc và phân tích đặc tính cabin. Loại cabin được kí hiệu bởi chữ cái đầu tiên. Lưu ý: Đối với dữ liệu cabin bị thiếu thì thay thế bằng "Unknown"
11. Loại bỏ dữ liệu thừa đối với các hành khách xuất hiện trong cả 2 tập dữ liệu huấn luyện (train.csv) và đánh giá (test.csv). Ưu tiên giữ lại dữ liệu trong tập huấn luyện.

PHẦN 2: KHAI THÁC THÔNG TIN HỮU ÍCH – EDA

Hướng dẫn: ***Sinh viên cần đưa ra nhận xét sau mỗi biểu đồ trực quan nhằm rút trích được thông tin có giá trị về hành khách sống sót dựa trên các đặc trưng bên trên***

12. Trực quan thông tin tương quan tỉ lệ sống sót và thiệt mạng trên từng nhóm giới tính.
13. Trực quan thông tin hành khách sống sót trên từng nhóm phân loại hành khách (Pclass).
14. Trực quan thông tin hành khách sống sót trên từng nhóm giới tính và thang đo tuổi tác
15. Trực quan xác suất hành khách sống sót dựa trên thông tin nhóm đi cùng
16. Trực quan xác suất hành khách sống sót dựa trên thông tin giá vé
17. Trực quan số lượng người thiệt mạng và sống sót theo phân lớp (Pclass) hành khách và cảng sẽ cập bến.

LAB 5:

BÀI TẬP TỔNG HỢP

Nội dung:

1. Học cách phân tích dữ liệu thông qua các giá trị tóm tắt dữ liệu và qua biểu diễn hình học của dữ liệu.
2. So sánh hai tập dữ liệu

Dữ liệu: Dữ liệu sử dụng trong lab này là tập dữ liệu về cân nặng của trẻ sơ sinh trong trường hợp bà mẹ hút thuốc lá khi mang thai và trong trường hợp bà mẹ không hút thuốc lá khi mang thai. (Dữ liệu được chuẩn bị sẵn trong tập tin: babies.txt).

Mô tả dữ liệu:

Tên cột	Ý nghĩa

bwt	Cân nặng của trẻ sơ sinh (baby weight), tính theo đơn vị ounce (100 ounce=2.83495kg)
smoke	Tình trạng hút thuốc của bà mẹ khi mang thai. 0= không hút, 1= có hút, 9=không biết

I. CÁC NỘI DUNG CẦN TÌM HIỂU:

Để thực hiện được lab này, sinh viên cần vận dụng các kiến thức ở các lab trên vào bài toán cụ thể:

1) Ước lượng độ biến động của dữ liệu:
   Hai yếu tố chính để ước lượng độ biến động của dữ liệu: tâm và đuôi dữ liệu. Qua đó, ta cần tìm hiểu: dữ liệu phân bố như thế nào ở trung tâm (center) và như thế nào ở hai bên đuôi (tail).
   Trong dữ liệu một chiều, để đo tính biến động của dữ liệu, ta có thể sử dụng các đại lượng: phương sai (Variance), độ lệch chuẩn (Standard deviation), khoảng cách giữa giá trị lớn nhất và nhỏ nhất (Range) và phần tư vị (IQR-InterQuantile Range). IQR cho phép khảo sát phần tâm dữ liệu trong khoảng từ ¼ cho đến ¾.
   Đôi khi, để dễ hình dung, người phân tích có thể biểu diễn dữ liệu theo boxplot hay histogram, sẽ minh họa sau.
2) Phân tích về hình dạng của phân phối dữ liệu:
   Để phân tích hình dạng phân phối dữ liệu, người phân tích cần tính giá trị **KURTOSIS**, là giá trị để đo độ "bè-nhọn" của đỉnh dữ liệu và giá trị **SKEWNESS** để đo độ "lệch (trái, phải)" của dữ liệu.
3) Phân tích tính chuẩn:
   Để phân tích xem dữ liệu có phân phối chuẩn hay không, một cách trực quan, ta biểu diễn theo đường cong chuẩn (normal curve) và đôi khi cần một số thao tác chuẩn hóa.

II.CÁC NỘI DUNG THỰC HIỆN:

Trong lab này, ta phân tích các dữ liệu quan sát được để trả lời câu hỏi: "Việc bà mẹ hút thuốc khi mang thai có ảnh hưởng đến cân nặng của trẻ sơ sinh hay không?"
Để trả lời câu hỏi trên, cần thực hiện so sánh cân nặng của trẻ sơ sinh trong hai trường hợp: trường hợp bà mẹ hút thuốc khi mang thai và trường hợp bà mẹ không hút thuốc khi mang thai. Sự khác biệt đó có ý nghĩa hay không?
Để so sánh cân nặng của trẻ sơ sinh trong 2 trường hợp, có thể dựa vào thống kê mô tả: thống kê mô tả bằng số (numerical summaries), thống kê mô tả bằng hình (graphical): histogram, boxplot, quantile plot. Do đó, các nội dung chi tiết cần thực hiện:

1) Tính các đại lượng thống kê mô tả từ đó rút ra nhận xét về từng tập dữ liệu (cân nặng của trẻ trong trường hợp bà mẹ hút thuốc và cân nặng của trẻ trong trường hợp bà mẹ không hút thuốc).
   Cụ thể, ta sẽ phân tích sự khác biệt giữa hai tập dữ liệu: cân nặng của trẻ trong trường hợp bà mẹ hút thuốc và cân nặng của trẻ trong trường hợp bà mẹ không hút thuốc dựa vào các đại lượng thống kê mô tả.
2) Biểu diễn dữ liệu dưới các dạng đồ thị từ đó rút ra nhận xét về từng tập dữ liệu (trường hợp bà mẹ hút thuốc và trường hợp bà mẹ không hút thuốc)

Cụ thể, ta sẽ sử dụng các dạng đồ thị: histogram, boxplot, quantile qua đó phân tích sự khác biệt giữa hai tập dữ liệu: cân nặng của trẻ trong trường hợp bà mẹ hút thuốc và cân nặng của trẻ trong trường hợp bà mẹ không hút thuốc dựa vào các đồ thị.

HƯỚNG DẪN THỰC HIỆN:

1. Mô tả dữ liệu bằng các giá trị số:

Bước 1: Tính các đại lượng thống kê cho hai tập dữ liệu:

(Cân nặng của trẻ trong trường hợp bà mẹ hút thuốc khi mang thai và cân nặng của trẻ trong trường hợp bà mẹ không hút thuốc khi mang thai).

Dùng python để thực hiện, kết quả được trình bày trong bảng sau:

	TH1: Bà mẹ hút thuốc	TH2: Bà mẹ không hút thuốc
Số lượng	484	742
Min	58	55
Max	163	176
Mean	114.10950413223141	123.04716981132076
Sd	18.09894568615237	17.39868877808027
Var	327.57183495029346	302.7143711964963
Median	115.0	123.0
Quantile 0%	58.0	55.0
Quanlite 25%	102.0	113.0
Quanlite 50%	115.0	123.0
Quantile 75%	126.0	134.0
IQR	24.0	21.0
Skewness	-0.03359497605204854	-0.18698408606617228
Kurtosis	2.988032478793404	4.037060312433822

Bước 2: Phân tích dữ liệu dựa trên các đại lượng vừa tính.

1. **Xét tập dữ liệu ứng với trường hợp bà mẹ có hút thuốc**

Vị trí tập trung của dữ liệu: khoảng giá trị: 114-115

Tính biến động của dữ liệu:

- **Phương sai (variance):** var= 327.57183495029346
- **Độ lệch chuẩn (standard deviation): sd=** 18.09894568615237
- **Khoảng giá trị:** min=58, max=163→ range=105
- **Khoảng cách giữa 2 phần tư vị:** IQR=Q3-Q1=126-102=24

Nhận xét: Như vậy dữ liệu phân bố gần nhau.

Hình dạng phân bố của dữ liệu:

- **Độ lệch:** Skewness=-0.03359497605204854
- **Độ bè nhọn của đỉnh dữ liệu:** Kurtosis=2.988032478793404

Nhận xét: Như vậy dữ liệu hơi lệch về phía trái, và đỉnh nhọn, hai bên giảm với tốc độ vừa phải.

2. **Xét tập dữ liệu ứng với trường hợp bà mẹ không hút thuốc**

Phần này sinh viên tự thực hiện.

Bước 3: So sánh các giá trị thống kê mô tả của hai tập dữ liệu.

Sự khác biệt về vị trí tập trung dữ liệu: chênh lệch khoảng 123 -115 = 8

Nhận xét: khác biệt không đáng kể.

Sự khác biệt về tính biến động của dữ liệu được thể hiện qua bảng sau:

	TH1: Bà mẹ hút thuốc	TH2: Bà mẹ không hút	Chênh lệch

		thuốc	(TH2-TH1)
Sd	18.09894568615237	17.39868877808027	-0.700256908
Var	327.57183495029346	302.7143711964963	-24.85746375
Range	163-58=105	176-55=121	16
IQR	126-102=24	134-113=21	-3

Dữ liệu trong trường hợp bà mẹ không hút thuốc có phân bố rộng hơn nhưng phần dữ liệu tập trung lại hẹp hơn so với trường hợp bà mẹ có hút thuốc. Sự biến động của dữ liệu trong hai trường hợp không khác biệt nhiều.

Sự khác biệt về hình dạng phân bố của dữ liệu: được thể hiện qua bảng sau:

	TH1: Bà mẹ hút thuốc	TH2: Bà mẹ không hút thuốc	Chênh lệch (TH2-TH1)
Skewness	-0.03359497605204854	-0.18698408606617228	-0.15338911
Kurtosis	2.988032478793404	4.037060312433822	1.049027834

- **Nhận xét:** trường hợp bà mẹ hút thuốc có phân bố dữ liệu nhọn hơn, đối xứng hơn so với trường hợp không hút thuốc. Cả 2 trường hợp đều hơi lệch về trái.

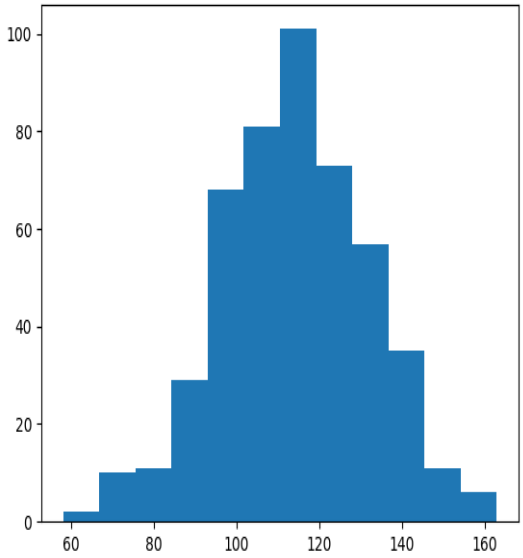
2. Biểu diễn hình học của dữ liệu

1. Dữ liệu cân nặng của trẻ trong trường hợp bà mẹ hút thuốc và bà mẹ không hút thuốc

Ta sẽ phân tích các biểu đồ:

- Histogram
- Boxplot

a) Histogram trong trường hợp bà mẹ có hút thuốc:



Vị trí tập trung dữ liệu: khoảng 110

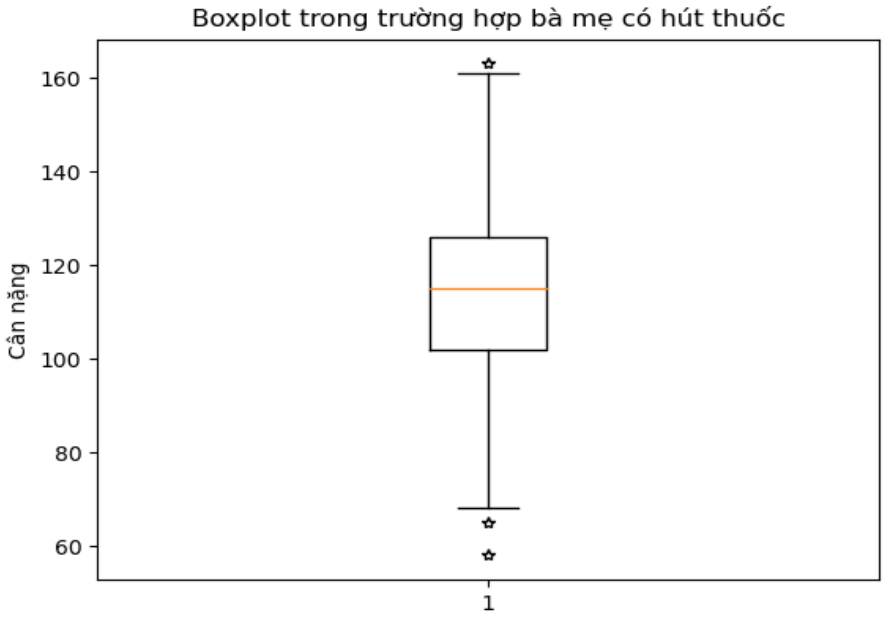
Tính biến động của dữ liệu: dữ liệu phân bố trong khoảng [50-170]

Tính đối xứng của phân bố dữ liệu: dữ liệu chỉ có 1 đỉnh. Bắt đầu từ đỉnh, hai bên giảm dần và tốc độ giảm vừa phải.

Dữ liệu phân bố gần đối xứng, hơi lệch về phía trái. Hai bên đuôi có độ dài vừa phải. Hai bên đỉnh dữ liệu cũng phân bố vừa phải.

Giá trị ngoại lệ: không thấy rõ có giá trị ngoại lệ nào đáng kể

b) Boxplot:



Tính biến động của dữ liệu: dữ liệu phân bố tập trung trong khoảng từ [102,126]
Giá trị ngoại lệ: có một số giá trị ngoại lệ (lớn hơn 162, nhỏ hơn 66) nhưng không nhiều.

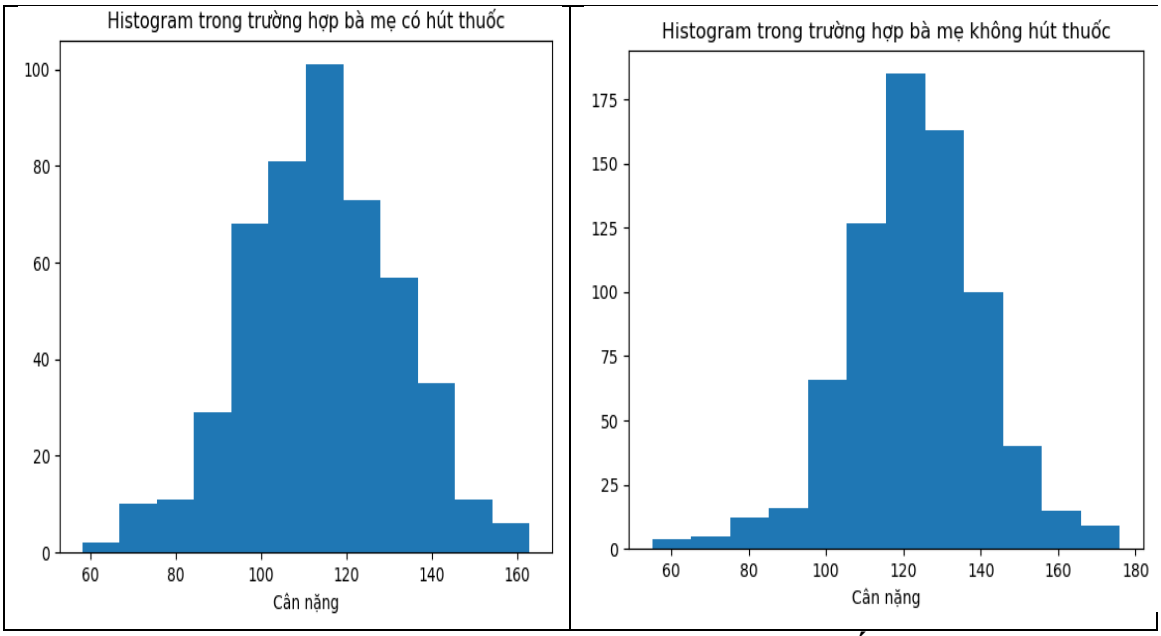
2. Dữ liệu cân nặng của trẻ trong trường hợp bà mẹ hút thuốc không hút thuốc

Phần này sinh viên tự thực hiện

So sánh hai tập dữ liệu dựa vào các biểu diễn hình học:

a) Histogram

Để so sánh, ta vẽ 2 histogram gần nhau:



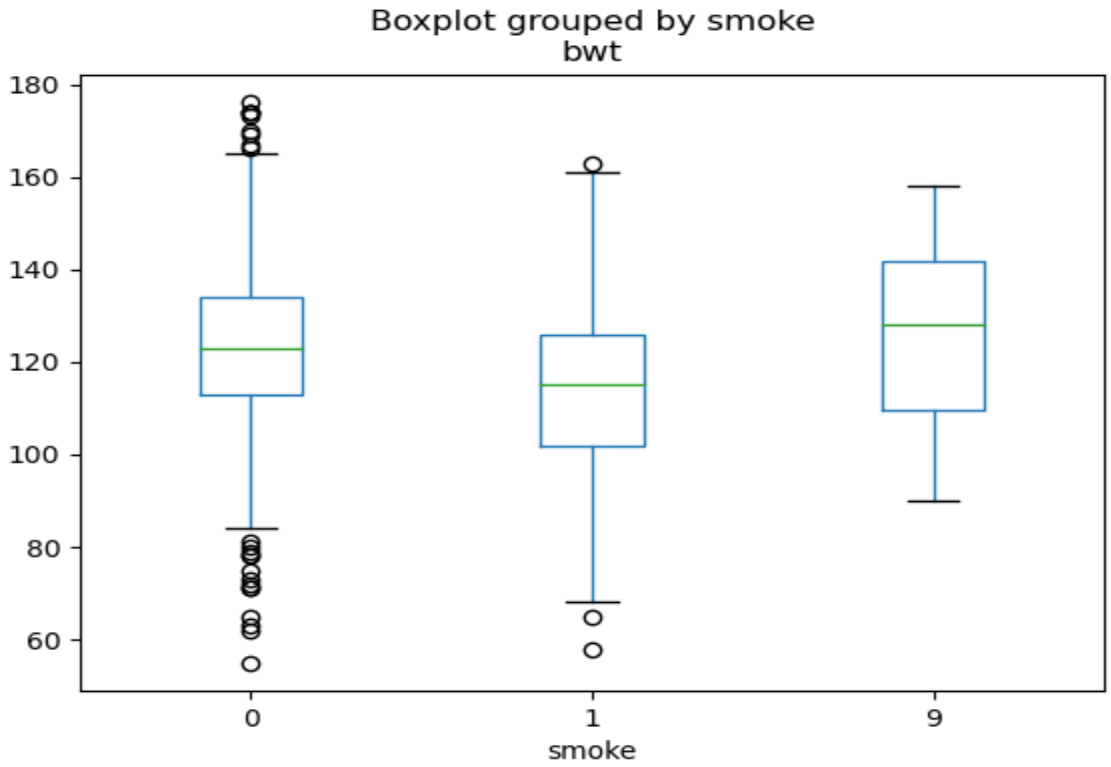
Cân nặng của trẻ trẻ trong trường hợp bà mẹ không hút thuốc cáo hơn so với trường hợp bà mẹ có hút thuốc

Tính biến thiên của 2 tập dữ liệu: tương tự nhau

Tính đối xứng của 2 tập dữ liệu: tương tự nhau

Giá trị ngoại lệ: cả 2 đều không có giá trị ngoại lệ đáng chú ý.

b) Boxplot



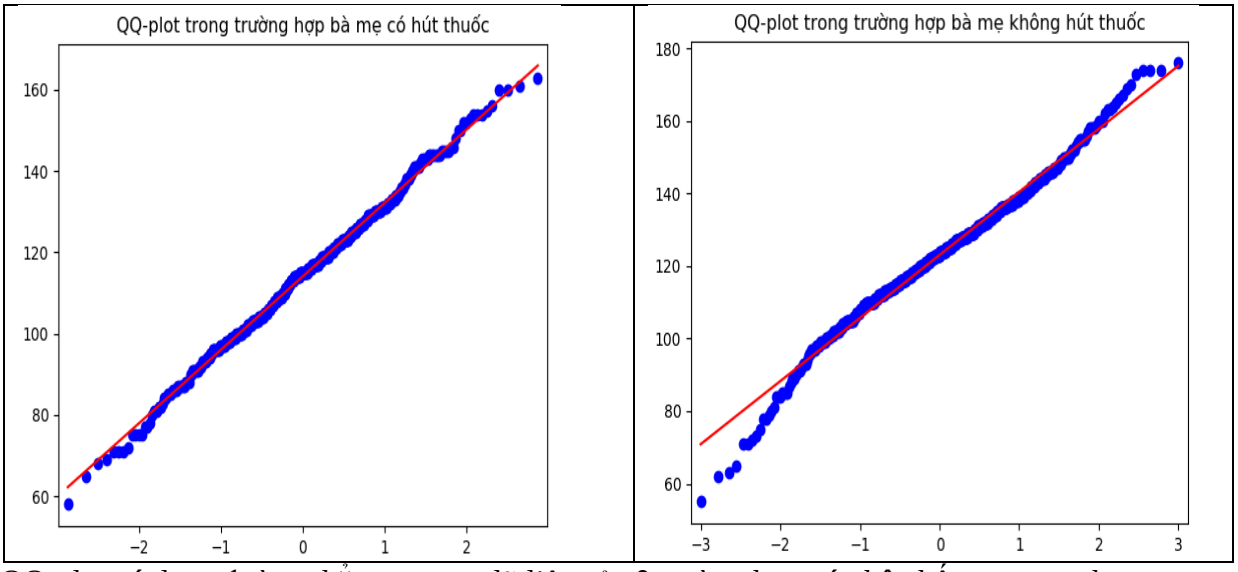
Khác biệt về vị trí: giá trị trung vị của trường hợp bà mẹ không hút thuốc lớn hơn trường hợp bà mẹ có hút thuốc (123 và 115). (Trường hợp smoke=9 là trường hợp không biết bà mẹ có hút thuốc hay không, trung vị trong trường hợp này cao hơn so với 2 trường hợp bà mẹ có hút thuốc và không hút thuốc).

Giá trị ngoại lệ: cả 2 trường hợp đều có giá trị ngoại lệ trên và dưới. Trường hợp không hút thuốc có nhiều giá trị ngoại lệ hơn.

Ta dùng thêm đồ thị QQ-plot để phân tích

So sánh QQ-plot phân bố của 2 tập dữ liệu:

QQ-plot	
Có hút thuốc	**Không hút thuốc**



QQ-plot có dạng đường thẳng, suy ra dữ liệu của 2 trường hợp có phân bố tương tự nhau.

LAB 6:

PHÂN LOẠI DỮ LIỆU VỚI SCIKIT-LEARN

Nội dung:

1. Mô tả dữ liệu IRIS cho bài toán phân loại
2. Kỹ thuật PCA Decomposition
3. Phân loại với thuật toán K-Nearest Neighbors
4. Bài tập thực hành

1. Mô tả dữ liệu IRIS cho bài toán phân loại:

Mô tả dữ liệu: tập dữ liệu về hoa Iris gồm 3 loại: Silky, Virginica và Versicolor. Dữ liệu này tương ứng với chiều dài và chiều rộng của đài hoa (sepal) và cánh hóa (petal). Dữ liệu gồm có 150 dòng dữ liệu, mỗi dòng gồm có 4 cột ứng với chiều dài và chiều rộng của đài hoa và cánh hoa.

Tập dữ liệu này hiện đang được sử dụng làm ví dụ điển hình cho nhiều loại phân tích, đặc biệt là đối với các vấn đề về phân loại, có thể được tiếp cận bằng cách phương pháp học máy.

Iris setosa | Iris versicolor | Iris virginica

Yêu cầu: xây dựng mô phân loại cho bài toán trên.

Hướng dẫn thực hiện:

Import thư viện và load tập dữ liệu vào biến iris:

```
from sklearn import datasets
iris = datasets.load_iris()
```

Xem dữ liệu của biến iris:

```
iris.data
```

Kết quả như sau:

```
array([[ 5.1,  3.5,  1.4,  0.2],
       [ 4.9,  3. ,  1.4,  0.2],
       [ 4.7,  3.2,  1.3,  0.2],
       [ 4.6,  3.1,  1.5,  0.2],
       [ 5. ,  3.6,  1.4,  0.2],
       [ 5.4,  3.9,  1.7,  0.4],
       [ 4.6,  3.4,  1.4,  0.3],
       [ 5. ,  3.4,  1.5,  0.2],
       [ 4.4,  2.9,  1.4,  0.2],
       [ 4.9,  3.1,  1.5,  0.1],
       [ 5.4,  3.7,  1.5,  0.2],
       [ 4.8,  3.4,  1.6,  0.2],
       [ 4.8,  3. ,  1.4,  0.1],
       [ 4.3,  3. ,  1.1,  0.1],
       [ 5.8,  4. ,  1.2,  0.2],
       [ 5.7,  4.4,  1.5,  0.4],
       [ 5.4,  3.9,  1.3,  0.4],
       [ 5.1,  3.5,  1.4,  0.3],
       [ 5.7,  3.8,  1.7,  0.3],
       [ 5.1,  3.8,  1.5,  0.3],
       [ 5.4,  3.4,  1.7,  0.2],
       [ 5.1,  3.7,  1.5,  0.4],
       [ 4.6,  3.6,  1. ,  0.2],
       [ 5.1,  3.3,  1.7,  0.5],
       [ 4.8,  3.4,  1.9,  0.2],
       [ 5. ,  3. ,  1.6,  0.2],
       [ 5. ,  3.4,  1.6,  0.4],
       [ 5.2,  3.5,  1.5,  0.2],
       [ 5.2,  3.4,  1.4,  0.2],
```

```
[ 4.7,  3.2,  1.6,  0.2],
[ 4.8,  3.1,  1.6,  0.2],
[ 5.4,  3.4,  1.5,  0.4],
[ 5.2,  4.1,  1.5,  0.1],
[ 5.5,  4.2,  1.4,  0.2],
[ 4.9,  3.1,  1.5,  0.1],
[ 5. ,  3.2,  1.2,  0.2],
[ 5.5,  3.5,  1.3,  0.2],
[ 4.9,  3.1,  1.5,  0.1],
[ 4.4,  3. ,  1.3,  0.2],
[ 5.1,  3.4,  1.5,  0.2],
[ 5. ,  3.5,  1.3,  0.3],
[ 4.5,  2.3,  1.3,  0.3],
[ 4.4,  3.2,  1.3,  0.2],
[ 5. ,  3.5,  1.6,  0.6],
[ 5.1,  3.8,  1.9,  0.4],
[ 4.8,  3. ,  1.4,  0.3],
[ 5.1,  3.8,  1.6,  0.2],
[ 4.6,  3.2,  1.4,  0.2],
[ 5.3,  3.7,  1.5,  0.2],
[ 5. ,  3.3,  1.4,  0.2],
[ 7. ,  3.2,  4.7,  1.4],
[ 6.4,  3.2,  4.5,  1.5],
[ 6.9,  3.1,  4.9,  1.5],
[ 5.5,  2.3,  4. ,  1.3],
[ 6.5,  2.8,  4.6,  1.5],
[ 5.7,  2.8,  4.5,  1.3],
[ 6.3,  3.3,  4.7,  1.6],
[ 4.9,  2.4,  3.3,  1. ],
[ 6.6,  2.9,  4.6,  1.3],
[ 5.2,  2.7,  3.9,  1.4],
[ 5. ,  2. ,  3.5,  1. ],
[ 5.9,  3. ,  4.2,  1.5],
[ 6. ,  2.2,  4. ,  1. ],
[ 6.1,  2.9,  4.7,  1.4],
[ 5.6,  2.9,  3.6,  1.3],
[ 6.7,  3.1,  4.4,  1.4],
[ 5.6,  3. ,  4.5,  1.5],
[ 5.8,  2.7,  4.1,  1. ],
[ 6.2,  2.2,  4.5,  1.5],
[ 5.6,  2.5,  3.9,  1.1],
[ 5.9,  3.2,  4.8,  1.8],
[ 6.1,  2.8,  4. ,  1.3],
[ 6.3,  2.5,  4.9,  1.5],
[ 6.1,  2.8,  4.7,  1.2],
[ 6.4,  2.9,  4.3,  1.3],
[ 6.6,  3. ,  4.4,  1.4],
[ 6.8,  2.8,  4.8,  1.4],
[ 6.7,  3. ,  5. ,  1.7],
[ 6. ,  2.9,  4.5,  1.5],
[ 5.7,  2.6,  3.5,  1. ],
[ 5.5,  2.4,  3.8,  1.1],
[ 5.5,  2.4,  3.7,  1. ],
[ 5.8,  2.7,  3.9,  1.2],
```

```
[ 6. ,  2.7,  5.1,  1.6],
[ 5.4,  3. ,  4.5,  1.5],
[ 6. ,  3.4,  4.5,  1.6],
[ 6.7,  3.1,  4.7,  1.5],
[ 6.3,  2.3,  4.4,  1.3],
[ 5.6,  3. ,  4.1,  1.3],
[ 5.5,  2.5,  4. ,  1.3],
[ 5.5,  2.6,  4.4,  1.2],
[ 6.1,  3. ,  4.6,  1.4],
[ 5.8,  2.6,  4. ,  1.2],
[ 5. ,  2.3,  3.3,  1. ],
[ 5.6,  2.7,  4.2,  1.3],
[ 5.7,  3. ,  4.2,  1.2],
[ 5.7,  2.9,  4.2,  1.3],
[ 6.2,  2.9,  4.3,  1.3],
[ 5.1,  2.5,  3. ,  1.1],
[ 5.7,  2.8,  4.1,  1.3],
[ 6.3,  3.3,  6. ,  2.5],
[ 5.8,  2.7,  5.1,  1.9],
[ 7.1,  3. ,  5.9,  2.1],
[ 6.3,  2.9,  5.6,  1.8],
[ 6.5,  3. ,  5.8,  2.2],
[ 7.6,  3. ,  6.6,  2.1],
[ 4.9,  2.5,  4.5,  1.7],
[ 7.3,  2.9,  6.3,  1.8],
[ 6.7,  2.5,  5.8,  1.8],
[ 7.2,  3.6,  6.1,  2.5],
[ 6.5,  3.2,  5.1,  2. ],
[ 6.4,  2.7,  5.3,  1.9],
[ 6.8,  3. ,  5.5,  2.1],
[ 5.7,  2.5,  5. ,  2. ],
[ 5.8,  2.8,  5.1,  2.4],
[ 6.4,  3.2,  5.3,  2.3],
[ 6.5,  3. ,  5.5,  1.8],
[ 7.7,  3.8,  6.7,  2.2],
[ 7.7,  2.6,  6.9,  2.3],
[ 6. ,  2.2,  5. ,  1.5],
[ 6.9,  3.2,  5.7,  2.3],
[ 5.6,  2.8,  4.9,  2. ],
[ 7.7,  2.8,  6.7,  2. ],
[ 6.3,  2.7,  4.9,  1.8],
[ 6.7,  3.3,  5.7,  2.1],
[ 7.2,  3.2,  6. ,  1.8],
[ 6.2,  2.8,  4.8,  1.8],
[ 6.1,  3. ,  4.9,  1.8],
[ 6.4,  2.8,  5.6,  2.1],
[ 7.2,  3. ,  5.8,  1.6],
[ 7.4,  2.8,  6.1,  1.9],
[ 7.9,  3.8,  6.4,  2. ],
[ 6.4,  2.8,  5.6,  2.2],
[ 6.3,  2.8,  5.1,  1.5],
[ 6.1,  2.6,  5.6,  1.4],
[ 7.7,  3. ,  6.1,  2.3],
[ 6.3,  3.4,  5.6,  2.4],
```

```
       [ 6.4,  3.1,  5.5,  1.8],
       [ 6. ,  3. ,  4.8,  1.8],
       [ 6.9,  3.1,  5.4,  2.1],
       [ 6.7,  3.1,  5.6,  2.4],
       [ 6.9,  3.1,  5.1,  2.3],
       [ 5.8,  2.7,  5.1,  1.9],
       [ 6.8,  3.2,  5.9,  2.3],
       [ 6.7,  3.3,  5.7,  2.5],
       [ 6.7,  3. ,  5.2,  2.3],
       [ 6.3,  2.5,  5. ,  1.9],
       [ 6.5,  3. ,  5.2,  2. ],
       [ 6.2,  3.4,  5.4,  2.3],
       [ 5.9,  3. ,  5.1,  1.8]])
```

Để biết loại hoa nào thuộc loại nào:

```
iris.target
```

Kết quả như sau:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

Để biết tên các loại hoa gồm những gì:

```
iris.target_names
```

Kết quả như sau:

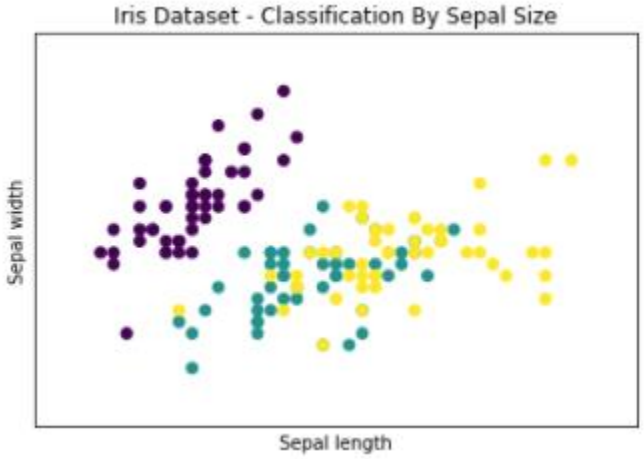
```
array(['setosa', 'versicolor', 'virginica'],
      dtype='<U10')
```

Vẽ đồ thị vẽ 3 loại hoa với 3 màu khác nhau, trục x là chiều dài của đài hoa và trục y là chiều rộng của đài hoa:

```
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
from sklearn import datasets

iris = datasets.load_iris()
x = iris.data[:,0] #X-Axis - sepal lenght
y = iris.data[:,1] #Y-Axis - sepal width
species = iris.target #species

x_min, x_max = x.min() - .5, x.max() + .5
y_min, y_max = y.min() - .5, y.max() + .5

#SCATTERPLOT
plt.figure()
plt.title('Iris Dataset - Classification By Sepal Size')
plt.scatter(x,y, c=species)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
```

```
plt.show()
```

Kết quả như sau:



Thực hiện tương tự với chiều dài và chiều rộng của cánh hoa ta có thêm 1 đồ thị:

```
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
from sklearn import datasets

iris = datasets.load_iris()
x = iris.data[:,2] #X-Axis - petal lenght
y = iris.data[:,3] #Y-Axis - petal width
species = iris.target #species

x_min, x_max = x.min() - .5, x.max() + .5
y_min, y_max = y.min() - .5, y.max() + .5

#SCATTERPLOT
plt.figure()
plt.title('Iris Dataset - Classification By Petal Sizes', size = 14)
plt.scatter(x,y, c=species)
plt.xlabel('Petal length')
plt.ylabel('Petal width')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Kết quả như sau:

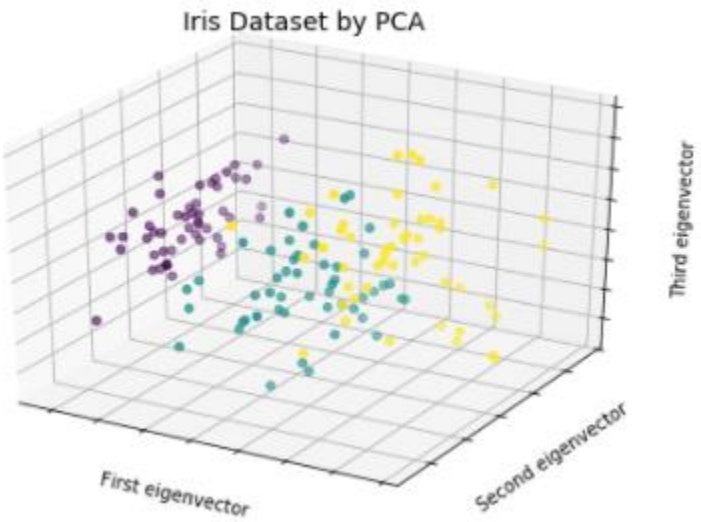
2. Kỹ thuật PCA Decomposition:

Chúng ta vẽ 2 đồ thị để thể hiện đặc trưng của 3 loại hoa trên. Làm cách nào để thể hiện đặc trưng trên chỉ với 1 đồ thị? Ta giảm chiều của các đặc trưng trên nhưng vẫn giữ nguyên thông tin, kỹ thuật này gọi là kỹ thuật PCA (*Principal Component Analysis*). Trong trường hợp này, ta giảm từ 4 chiều thành 3 chiều và sau đó trực quan hóa trên đồ thị 3D.

Trong thư viện scikit-learn ta dùng hàm fit_transform() để chuyển từ 4 chiều thành 3 chiều.

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA
iris = datasets.load_iris()
#x = iris.data[:,1] #X-Axis - sepal width
#y = iris.data[:,2] #Y-Axis - petal lenght
species = iris.target #species
x_reduced = PCA(n_components=3).fit_transform(iris.data)
#SCATTERPLOT 3D
fig = plt.figure()
ax = Axes3D(fig)
ax.set_title('Iris Dataset by PCA', size = 14)
ax.scatter(x_reduced[:,0],x_reduced[:,1],x_reduced[:,2], c=species)
ax.set_xlabel('First eigenvector')
ax.set_ylabel('Second eigenvector')
ax.set_zlabel('Third eigenvector')
ax.w_xaxis.set_ticklabels(())
ax.w_yaxis.set_ticklabels(())
ax.w_zaxis.set_ticklabels(())
plt.show()
```

Kết quả như sau:



3. Phân loại với thuật toán K-Nearest Neighbors:

Bây giờ, chúng ta sẽ thực hiện phân loại với thư viện scikit-learn.
Nhiệm vụ của mô hình phân loại là tìm ra một loại hoa ta đưa vào để phân loại xem nó thuộc về loài nào trong ba loại.
Ta phân chia tập ban đầu thành 2 tập dữ liệu, 1 tập là tập train và 1 tập là tập test.
Trong tập dữ liệu ban đầu, dữ liệu đã được sắp xếp theo loại. Do đó, trước khi phân chia tập dữ liệu này thành 2 tập, ta cần trộn dữ liệu lại. Dữ liệu sau khi trộn gồm 150 dòng, 140 dòng đầu cho tập train và 10 dòng sau cho tập test.

```
import numpy as np
from sklearn import datasets
np.random.seed(0)
iris = datasets.load_iris()
x = iris.data
y = iris.target
i = np.random.permutation(len(iris.data))
x_train = x[i[:-10]]
y_train = y[i[:-10]]
x_test = x[i[-10:]]
y_test = y[i[-10:]]
```

Bây giờ bạn có thể áp dụng thuật toán K-Nearest Neighbor để phân loại. Import thư viện KNeighborsClassifier, gọi hàm KNeighborsClassifier(), sau đó huấn luyện nó với hàm fit ().

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
```

Kết quả như sau

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=5, p=2,
           weights='uniform')
```

Thực hiện dự báo

```
knn.predict(x_test)
```

Kết quả:

```
array([1, 2, 1, 0, 0, 0, 2, 1, 2, 0])
Xem dữ liệu từ tập test:
```

```
y_test
```

Kết quả

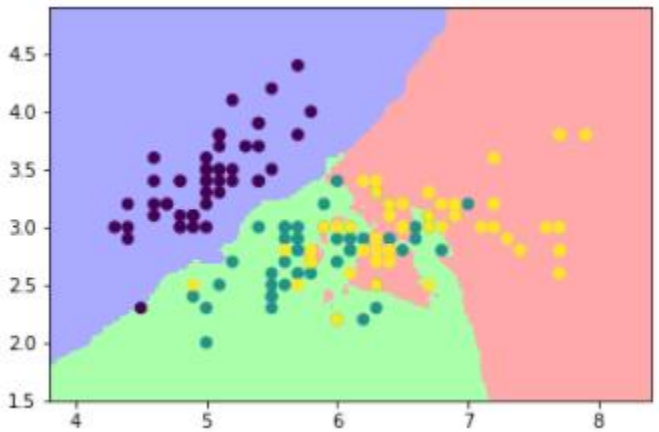
```
array([1, 1, 1, 0, 0, 0, 2, 1, 2, 0])
```

Chúng ta so sánh giá trị ta dự báo từ mô hình so với tập dữ liệu thực tế, ta thấy lỗi 10%.

Trực quan hóa dữ liệu của đài hoa:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import datasets
from sklearn.neighbors import KNeighborsClassifier

iris = datasets.load_iris()
x = iris.data[:,:2] #X-Axis - sepal lenght-width
y = iris.target     #Y-Axis - species

x_min, x_max = x[:,0].min() - .5, x[:,0].max() + .5
y_min, y_max = x[:,1].min() - .5, x[:,1].max() + .5

#MESH
cmap_light = ListedColormap(['#AAAAFF','#AAFFAA','#FFAAAA'])
h = .02
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max,
h))
knn = KNeighborsClassifier()
knn.fit(x,y)
Z = knn.predict(np.c_[xx.ravel(),yy.ravel()])
Z = Z.reshape(xx.shape)

plt.figure()
plt.pcolormesh(xx,yy,Z,cmap=cmap_light)

#Plot the training points
plt.scatter(x[:,0],x[:,1],c=y)
plt.xlim(xx.min(),xx.max())
plt.ylim(yy.min(),yy.max())
plt.show()
```

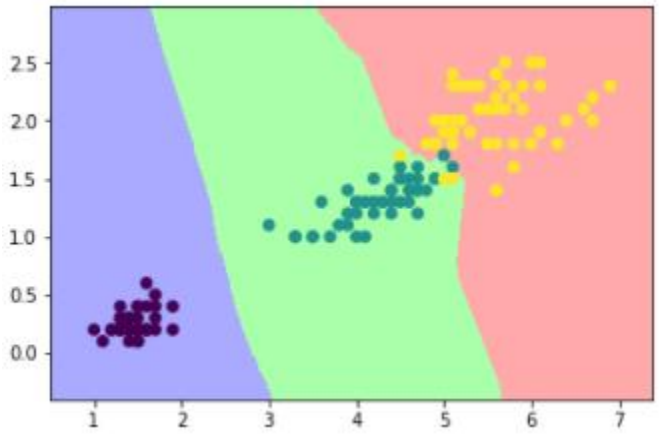
Kết quả:



Trực quan hóa dữ liệu của cánh hoa:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn import datasets
from sklearn.neighbors import KNeighborsClassifier

iris = datasets.load_iris()
x = iris.data[:,2:4] #X-Axis - petal lenght-width
y = iris.target      #Y-Axis - species

x_min, x_max = x[:,0].min() - .5, x[:,0].max() + .5
y_min, y_max = x[:,1].min() - .5, x[:,1].max() + .5

#MESH
cmap_light = ListedColormap(['#AAAAFF','#AAFFAA','#FFAAAA'])
h = .02
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max,
h))
knn = KNeighborsClassifier()
knn.fit(x,y)
Z = knn.predict(np.c_[xx.ravel(),yy.ravel()])
Z = Z.reshape(xx.shape)

plt.figure()
plt.pcolormesh(xx,yy,Z,cmap=cmap_light)

#Plot the training points
plt.scatter(x[:,0],x[:,1],c=y)
plt.xlim(xx.min(),xx.max())
plt.ylim(yy.min(),yy.max())
plt.show()
```

Kết quả như sau:



4. Bài tập thực hành

Dữ liệu: Dữ liệu sử dụng trong lab này là dữ liệu về kích thước giáp cua. (Dữ liệu được chuẩn bị sẵn trong tập tin: crabs.txt).

Mô tả dữ liệu:

Tên cột	Ý nghĩa
Premolt	Kích thước giáp cua trước khi lột vỏ (tính bằng mm)
Postmolt	Kích thước giáp cua sau khi lột vỏ (tính bằng mm)
Increment	Hiệu số giữa postmolt và premolt
Year	Năm (81: năm 1981, 82: năm 1982, 92: 1992).
Source	Nguồn gốc của cua: 1: lột vỏ trong phòng thí nghiệm; 0: lột vỏ trong tự nhiên.

Trong lab này, ta xem xét các vấn đề sau:

- **Tìm mối quan hệ giữa kích thước của giáp cua trước khi lột vỏ và sau khi lột vỏ.**
- **Dự đoán kích thước của giáp cua trước khi lột vỏ dựa vào thông tin về kích thước của giáp cua sau khi lột vỏ.**

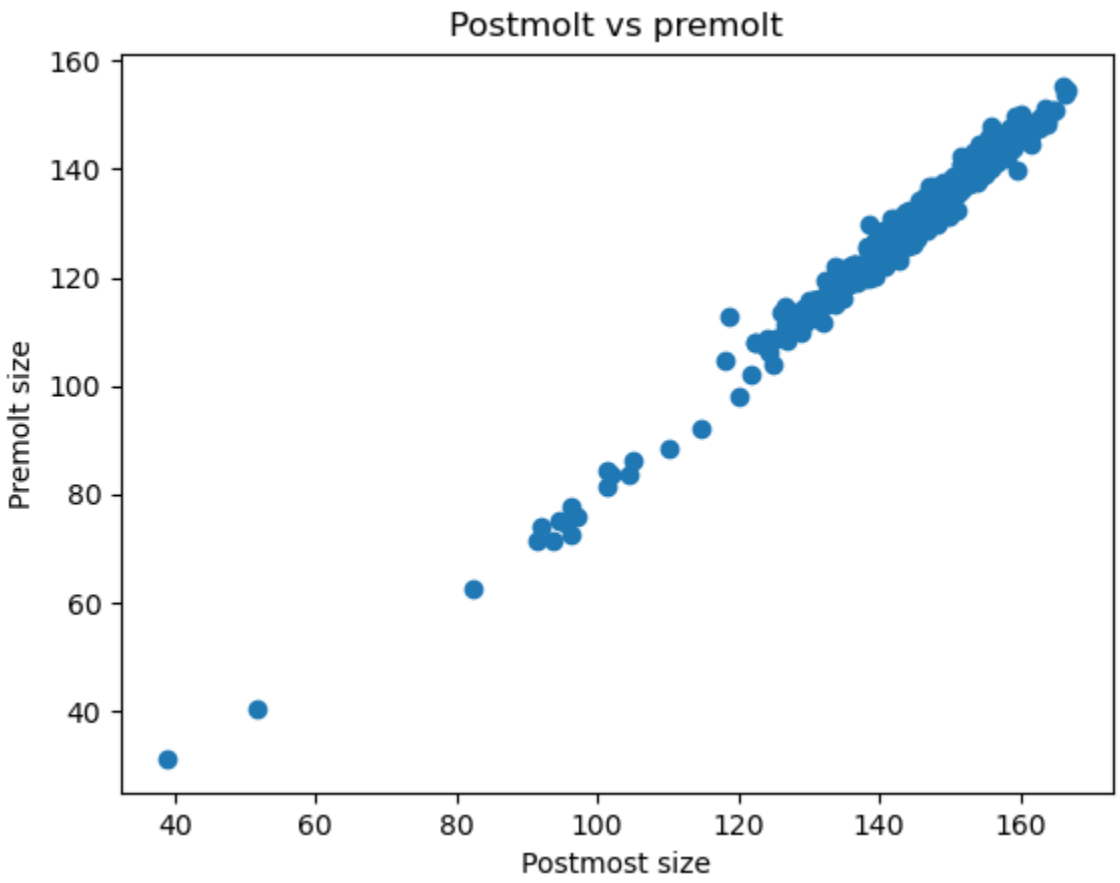
Trong lab này, ta thực hiện các nội dung sau:

- Vẽ đồ thị phân tán thể hiện mối tương quan giữa kích thước của giáp cua sau khi lột vỏ và trước khi lột vỏ (postmolt và premolt)
- Tính hệ số tương quan giữa kích thước của giáp cua sau khi lột vỏ và trước khi lột vỏ (postmolt và premolt)
- Xây dựng phương trình hồi quy
- Kiểm định xem phương trình hồi quy có khớp với dữ liệu không
- Tính khoảng sai số khi dự đoán giá trị premolt dựa vào postmolt
- Xác định và xử lý các giá trị có ảnh hưởng đến phương trình hồi quy
- Dựa vào phương trình hồi quy đã xây dựng để dự đoán

1. Vẽ đồ thị scatter plot thể hiện mối tương quan giữa postmolt và premolt

Dùng python để vẽ scatter plot thể hiện mối tương quan giữa postmolt và premolt

Kết quả:



Nhận xét: dữ liệu tập trung theo dạng đường thẳng.

2. Tính hệ số tương quan giữa postmolt và premolt

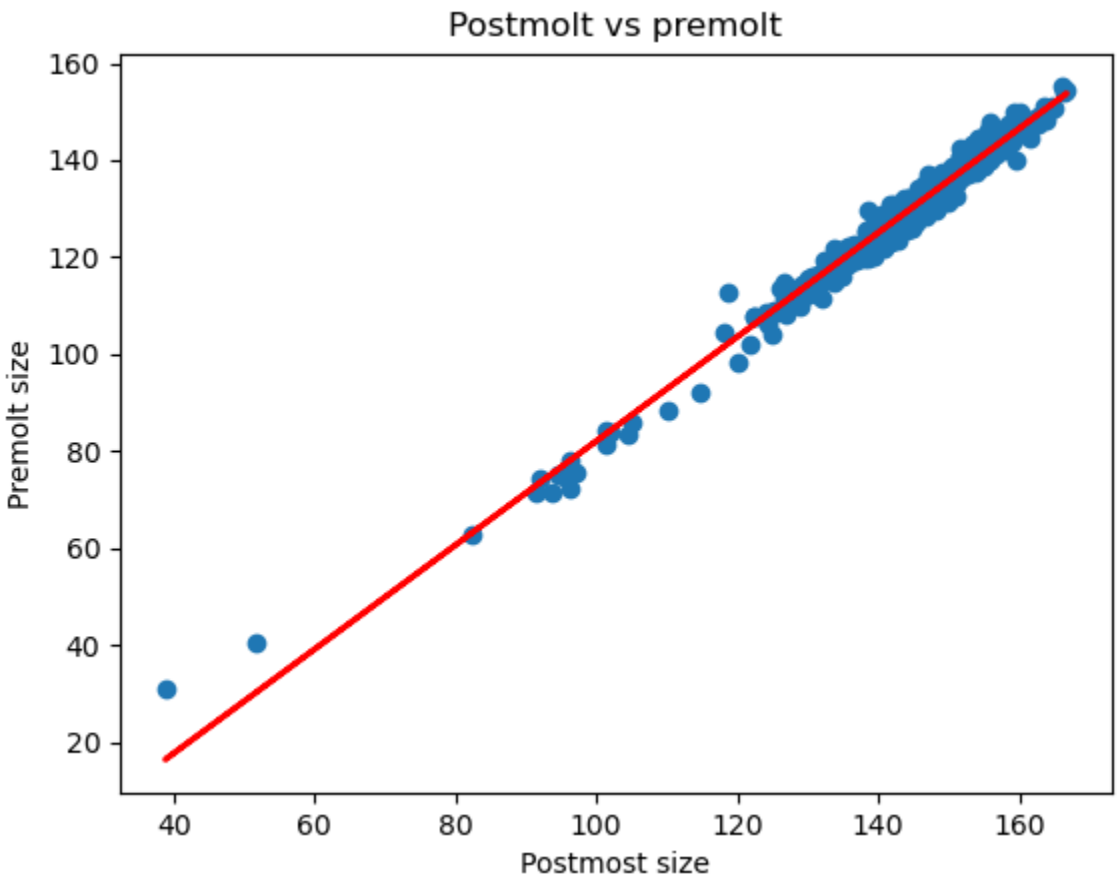
Dùng python tính hệ số tương quan giữa postmolt và premolt
Có nhận xét gì về hệ số tương quan đã tính được?
Kết quả:

```
He so tuong quan la:
(0.9903699282533851, 0.0)
```

Nhận xét: Hệ số tương quan là 0.9903699282533851, có giá trị gần với 1, P-value=0.0 < α (0.05) nghĩa là giữa 2 đại lượng Postmolt và Premolt có mối quan hệ tuyến tính mạnh, mối quan hệ này có ý nghĩa thống kê.

3. Xây dựng phương trình hồi quy tuyến tính

Dùng python để xây dựng phương trình hồi quy tuyến tính giữa postmolt và premolt.
Kết quả:



Kết quả:

OLS Regression Results

Dep. Variable:	PreMolt	R-squared:	0.981
Model:	OLS	Adj. R-squared:	0.981
Method:	Least Squares	F-statistic:	2.405e+04
Date:	Tue, 15 Sep 2020	Prob (F-statistic):	0.00
Time:	12:41:00	Log-Likelihood:	-1040.6
No. Observations:	472	AIC:	2085.
Df Residuals:	470	BIC:	2094.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-25.2137	1.001	-25.191	0.000	-27.180	-23.247
PostMolt	1.0732	0.007	155.083	0.000	1.060	1.087

Omnibus:	107.875	Durbin-Watson:	1.684
Prob(Omnibus):	0.000	Jarque-Bera (JB):	625.191
Skew:	0.845	Prob(JB):	1.74e-136
Kurtosis:	8.379	Cond. No.	1.43e+03

Giải thích:

No. Observations: số lượng đối tượng trong mẫu quan sát là n=472

Df Residuals: bậc tự do của phần dư = n- k (k: số lượng tham số trong phương trình hồi quy) = 472-2=470.

Df Model: bậc tự do của mô hình = k-1=2-1=1.

R-squared: có nghĩa là 98.1% kích thước giáp cua trước khi lột vỏ có thể được giải thích bởi biến dự báo.

Adj. R-squared: được sử dụng trong hồi quy đa biến. Trong hồi quy đơn biến thì **Adj. R-squared= R-squared.** Trong bài này **Adj. R-squared= R-squared=0.981.** Dùng **Adj. R-squared** để xác định phương trình hồi quy với số biến tham gia nào là tốt nhất. Chọn các phương trình hồi quy có giá trị **Adj. R-squared** cao và chỉ bao gồm một ít biến.

F-statistic: dùng trong hồi quy đa biến, ta kiểm định lại các hệ số $\beta_1, \beta_2,\ldots, \beta_n$ bằng 0 hay không bằng cách kiểm định giả thuyết sau:

- $H_0: \beta_1= \beta_2 =\ldots=\beta_n=0$
- $H_A: \beta_j \neq 0$

Prob (F-statistic): P-value (F-statistic). Nếu P-value (F-statistic) < α thì bác bỏ giả thuyết H_0.

AIC và BIC: là viết tắt của Akaike's Information Criteria và được sử dụng để lựa chọn mô hình **trong hồi quy đa biến**. AIC là phép tính để dung hòa tổng bình phương lỗi và số biến độc lập tham gia vào mô hình. AIC thấp hơn ngụ ý một mô hình tốt hơn. BIC là viết tắt Bayesian information criteria và là một biến thể của AIC. BIC thấp hơn ngụ ý một mô hình tốt hơn.

Log-Likelihood: một cách tiếp cận rất phổ biến trong thống kê là ý tưởng về ước tính khả năng xảy ra tối đa (maximum likelihood). Ý tưởng cơ bản hoàn toàn khác với cách tiếp cận OLS (bình phương nhỏ nhất): trong phương pháp tiếp cận bình phương nhỏ nhất, mô hình là không đổi, và sai số của biến phản hồi có thể thay đổi; ngược lại, trong phương pháp tiếp cận ước tính khả năng xảy ra tối đa, các giá trị phản hồi dữ liệu là được coi là không đổi, và khả năng của mô hình được tối đa hóa.

Regression coefficient (coef): hệ số hồi quy. Kết quả tính toán cho thấy β_0 (Intercept) =-25.2137 và β_1 (PostMolt) = 1.0732. Với 2 thông số này, chúng ta có thể ước tính của kích thước giáp cua trước khi lột cho bất cứ kích thước của giáp cua sau khi lột (trong khoảng kích thước giáp cua sau khi lột của mẫu) bằng phương trình tuyến tính:

$$\widehat{\text{Premolt}} = -25.2137 + 1.0732 * \text{PostMolt}.$$

Phương trình này có nghĩa là khi tăng kích thước giáp cua sau khi lột vỏ lên 1 đơn vị thì kích thước giáp cua trước khi lột vỏ tăng lên 1.0732 đơn vị.

Standard error: đo độ chính xác của hệ số β_1 (PostMolt) bằng cách ước tính sự biến thiên của hệ số nếu cùng 1 thử nghiệm chạy trên một mẫu khác nhau được lấy mẫu từ quần thể. Tương tự đối với hệ số hồi quy β_0 (Intercept).

t: ta kiểm định lại hệ số $\beta_0= 0, \beta_1= 0$ hay không bằng cách kiểm định giả thuyết sau:

Kiểm định hệ số β_0:

- $H_0: \beta_0=0$
- $H_A: \beta_0 \neq 0$

Và kiểm định hệ số β_1:

- $H_0: \beta_1= 0$
- $H_A: \beta_1 \neq 0$

Với $\widehat{\beta}_0$=-25.2137, $\widehat{\beta}_1$=1.0732

$t_0 = \frac{\hat{\beta}0 - \beta 0}{s.e.(\hat{\beta}0)} = \frac{\hat{\beta}0 - 0}{s.e.(\hat{\beta}0)} = -25.191$

$t_1 = \frac{\hat{\beta}1 - \beta 1}{s.e.(\hat{\beta}1)} = \frac{\hat{\beta}1 - 0}{s.e.(\hat{\beta}1)} = 155.083$

P-value (t_0) = P>|t_0| = 2*(1- $t_{(n-2,\ \alpha/2\ )}$(|t_0|)=0

P-value (t_1) = P>|t_1| = 2*(1- $t_{(n-2,\ \alpha/2\ )}$(|t_1|)=0

Vì P-value (t_0) = 0<α (0.05) nên bác bỏ giả thuyết H_0

Vì P-value (t_1) = 0<α (0.05) nên bác bỏ giả thuyết H_0

Chúng ta có bằng chứng để cho rằng có mối liên hệ giữa kích thước giáp cua trước khi lột và kích thước giáp cua sau khi lột, mối liên hệ này có ý nghĩa thống kê.

P>|t|: P-value (t) = Pr(T>|t|). Nếu P-value (t) <α thì bác bỏ giả thuyết H_0.

Confidence interval: phạm vi mà hệ số hồi quy dao động. Kết quả cho thấy rằng, chúng ta tin tưởng 95% rằng hệ số β_0 dao động từ -27.180 đến -23.247, hệ số β_1 dao động từ 1.060 đến 1.087. Kết quả trên được tính như sau:

Khoảng tin cậy cho hệ số $\boldsymbol{\beta_0}$ là từ $\hat{\beta}_0$ - $t_{(n-2,\alpha/2)}$ *s. e.($\hat{\beta}_0$) đến $\hat{\beta}_0$ + $t_{(n-2,\alpha/2)}$ *s. e.($\hat{\beta}_0$)

Tính $t_{(n-2,\ \alpha/2)} = t_{(470,0.025)}$ = stats.t(470).ppf(0.975) (from scipy import stats)

$t_{(n-2,\ \alpha/2)} = t_{(470,0.025)} = 1.965$

Thay $\hat{\beta}_0$= -25.2137 và $t_{(n-2,\ \alpha/2)}$ = 1.965 vào biểu thức trên ta được khoảng tin cậy cho hệ số $\boldsymbol{\beta_0}$ là:

-25.2137 – 1.965*1.001 **đến** -25.2137 + 1.965*1.001

- 27.180 **đến** -23.247

Khoảng tin cậy cho hệ số $\boldsymbol{\beta_1}$ là từ $\hat{\beta}_1$ - $t_{(n-2,\alpha/2)}$ *s. e.($\hat{\beta}_1$) đến $\hat{\beta}_1$ + $t_{(n-2,\alpha/2)}$ *s. e.($\hat{\beta}_1$)

Thay $\hat{\beta}_1$= 1.0732 và $t_{(n-2,\ \alpha/2)}$ = 1.965 vào biểu thức trên ta được khoảng tin cậy cho hệ số $\boldsymbol{\beta_1}$ là:

1.0732 – 1.965*0.007 **đến** 1.0732 + 1.965*0.007

1.060 **đến** 1.087

Skew và kurtosis: Skew và kurtosis đề cập đến hình dạng của một phân phối, giá trị **skew** để đo độ "lệch (trái, phải)" của dữ liệu (đối với dữ liệu được phân phối chuẩn, **skew** có giá trị khoảng bằng 0), **kurtosis**, là giá trị để đo độ "bè-nhọn" của đỉnh dữ liệu (đối với dữ liệu được phân phối chuẩn, **kurtosis** có giá trị khoảng bằng 3). Trong bài này, **skew=0.845 và kurtosis=8.379** nên phần dư không phân phối chuẩn.

Omnibus: kiểm định **Omnibus** sử dụng **skew** và **kurtosis** để kiểm tra giả thuyết Null: phân phối của phần dư là phân phối chuẩn. Nếu **P-value (Omnibus) < α**, thì phần dư không phân phối chuẩn, chúng ta cần xem xét lại mô hình.

Prob (Omnibus): P-value của Omnibus. Trong bài này, **Prob (Omnibus)=0.00< α** nên phần dư không phân phối chuẩn.

Durbin–Watson: kiểm định **Durbin – Watson** được sử dụng để phát hiện **sự hiện diện của sự tự tương quan** trong phần dư từ phân tích hồi quy. Giá trị thống kê **Durbin–Watson** sẽ luôn có giá trị từ 0 đến 4. Giá trị 2.0 có nghĩa là không có hiện tượng tự tương quan được phát hiện trong mẫu. Các giá trị từ 0 đến nhỏ hơn 2 cho biết tự tương quan dương và các giá trị từ 2 đến 4 cho biết tự tương quan âm.

Jarque–Bera: kiểm định **Jarque–Bera** là một dạng kiểm định khác xem xét độ lệch **skess và kurtosis**. Giả thuyết Null: phân phối của phần dư là phân phối chuẩn, hoặc nói một cách khác, **skew=0 và kurtosis=3**. Nếu **P-value (JB) < α**, thì phần dư không phân phối chuẩn, chúng ta cần xem xét lại mô hình.

Prob (JB): P-value của JB: Trong bài này, Prob (JB)=1.74e-136< **α** nên phần dư không phân phối chuẩn.

Cond. No.: (dùng trong hồi quy đa biến) đo lường độ nhạy của đầu ra của một hàm đối với đầu vào của hàm. Khi hai biến dự báo có tương quan cao, được gọi là **multicollinearity**, các hệ số hồi quy có thể dao động thất thường đối với những thay đổi nhỏ trong dữ liệu hoặc mô hình. **Multicollinearity** có thể gây ra kết quả không chính xác, cần xem lại mô hình. Nếu **Cond. No.** lớn hơn 30, thì hồi quy có thể gặp phải trường hợp **multicollinearity**.

Kết luận: qua các phân tích trên, phần dư không tuân theo phân phối chuẩn, do đó, mô hình hồi quy trên cần xây dựng lại bằng cách loại bỏ các giá trị có ảnh hưởng đến phương trình hồi quy.

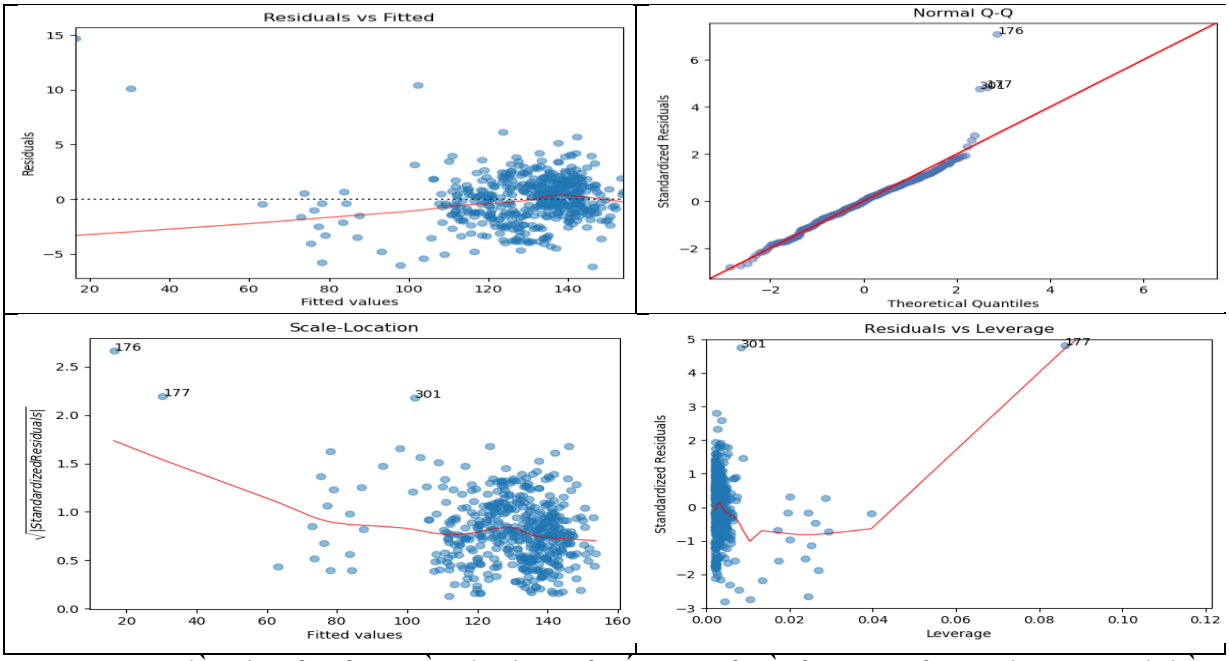
4. Kiểm định phương trình hồi quy tuyến tính

Dùng python để kiểm định lại phương trình hồi quy bằng các đồ thị sau:

- Dùng đồ thị Residual value vs Fitted value
- Dùng đồ thị Normal Q-Q
- Dùng đồ thị Scale-location
- Dùng đồ thị Residual vs Leverage

Hướng dẫn:

- Đồ thị Residual value vs Fitted value:
  Đồ thị vẽ phần dư e_i và giá trị dự đoán Premolt $\hat{y}_i$. Đồ thị này cho thấy các giá trị phần dư tập trung quanh đường y=0, tuy nhiên, có một vài điểm dữ liệu không tập trung quanh đường y=0 cho nên giả định ε_i có giá trị trung bình là 0 là không chấp nhận được.
- Đồ thị Normal Q-Q:
  Đồ thị vẽ giá trị phần dư và giá trị kỳ vọng dựa vào phân phối chuẩn. Chúng ta thấy các số phần dư tập trung rất gần các giá trị trên đường chuẩn, tuy nhiên có một số điểm bị lệch nhiều khỏi đường chuẩn, và do đó, giả định ε_i phân phối theo luật phân phối chuẩn không thể đáp ứng.
- Đồ thị Scale-location:
  Đồ thị vẽ căn bậc 2 của phần dư chuẩn (standardized residual) và giá trị dự đoán $\hat{y}_i$. Đồ thị này cho thấy không có gì khác giữa các số phần dư chuẩn cho các giá trị dự đoán $\hat{y}_i$, và do đó, giả định các ε_i có phương sai σ^2 cố định cho tất cả các x_i có thể áp dụng.
- Đồ thị Residual vs Leverage:
  Đồ thị này giúp xem các giá trị ngoại lệ trong mô hình hồi quy tuyến tính có ảnh hưởng đến việc phân tích hồi quy hay không. Nếu có thì cần loại bỏ các giá trị ngoại lệ khỏi tập dữ liệu.
  Dựa vào đồ thị này:
  - Các điểm có leverage cao khi $h_{ii} > \frac{3p}{n} = \frac{3*2}{472} = 0.0127$ (p: số các tham số cần ước lượng (trong bài này cần ước lượng 2 tham số β_0 và β_1 nên p=2), n: kích thước mẫu)
  - Các điểm là oulier khi Standard Residual>3
  - Các điểm có ảnh hưởng đến phương trình hồi quy (influence point) cần loại bỏ là các điểm oulier và có leverage cao



Dựa vào các đồ thị phân tích phần dư, bạn có kết luận gì về tính hợp lý của phương trình hồi quy đã xây dựng?

5. Tính khoảng sai số khi dự đoán

Dùng python để ước lượng các hệ số β_0, β_1 với độ tin cậy 1-α=0.95

Kết quả:

	0	1
Intercept	-27.180474	-23.246931
postmolt	1.059565	1.086760

Khoảng tin cậy cho hệ số $\boldsymbol{\beta_0}$ là từ - 27.180 **đến** -23.247

Khoảng tin cậy cho hệ số $\boldsymbol{\beta_1}$ là từ 1.060 **đến** 1.087

6. Xác định và xử lý các giá trị có ảnh hưởng đến phương trình hồi quy

- Sử dụng đồ thị đồ thị Residual vs Leverage (hoặc sử dụng khoảng cách Cook (đồ thị Cook's dist vs Leverage), khoảng cách Dffits (đồ thị Cook's dist vs Leverage)) để xác định các điểm có ảnh hưởng đến phương trình hồi quy.
- Xây dựng phương trình hồi quy nếu loại bỏ các giá trị có ảnh hưởng đến phương trình hồi quy (phần này sinh viên tự thực hiện).
- So sánh sự khác biệt giữa hai mô hình: đánh giá xem sự khác biệt có đáng kể không. Kết luận về sự quan trọng của các giá trị có ảnh hưởng đến phương trình hồi quy (phần này sinh viên tự thực hiện).
- Kết luận: bỏ hay giữ các giá trị có ảnh hưởng đến phương trình hồi quy (phần này sinh viên tự thực hiện).

7. Dựa vào phương trình hồi quy đã xây dựng để dự đoán

Giả sử giá trị postmolt size là: 85, dựa vào phương trình hồi quy đã xây dựng, với độ tin cậy là 1-α=0.95, bạn dự đoán giá trị premolt size nằm trong khoảng nào?

Bài tập làm thêm: hồi quy đa biến:

Bài 1:

Dữ liệu: Dữ liệu sử dụng trong lab này là tập dữ liệu về lượng nhựa, nicotine, CO trong thuốc lá cỡ king. (Dữ liệu được chuẩn bị sẵn trong tập tin: 04_CIGARET.xls).

Mô tả dữ liệu:

Tên cột	Ý nghĩa
KgTar	lượng nhựa trong 1 điếu thuốc lá
KgNic	lượng nicotine trong 1 điếu thuốc lá
KgCO	lượng CO trong 1 điếu thuốc lá

1. Xây dựng phương trình hồi quy thể hiện mối liên hệ giữa lượng nicotine trong thuốc lá và lượng nhựa, CO trong thuốc lá.
2. Bạn hãy xác định phương trình hồi quy trên có thể sử dụng để dự đoán lượng nicotine trong thuốc lá khi biết lượng nhựa và CO trong thuốc lá không? Vì sao có hoặc vì sao không?

Bài 2:

Dữ liệu: Dữ liệu sử dụng trong lab này là dữ liệu về giá bán nhà. (Dữ liệu được chuẩn bị sẵn trong tập tin: 23_HOMES.xls).

Mô tả dữ liệu:

Tên cột	Ý nghĩa
Selling_Price	giá bán
List_Price	giá niêm yết
Area	diện tích sử dụng của ngôi nhà
Acres	diện tích đất

1. Nếu chỉ sử dụng 1 biến x để dự đoán giá nhà, phương trình hồi quy 1 biến dự đoán (predictor) nào sau đây là tốt nhất? Tại sao?

Predictor (x) Variables	P-value	R^2	Adjusted R^2	Regression Equation
LP, LA, Lot	0.000	0.990	0.989	$\hat{y} = 1120 + 0.972$ LP $+ 0.281$ LA $+ 465$ LOT
LP, LA	0.000	0.990	0.989	$\hat{y} = -40.5 + 0.985$ LP $- 0.985$ LA
LP, LOT	0.000	0.990	0.989	$\hat{y} = 1004 + 0.974$ LP $+ 429$ LOT
LA, LOT	0.000	0.815	0.805	$\hat{y} = 111{,}309 + 98.2$ LA $+ 17{,}269$ LOT
LP	0.000	0.990	0.990	$\hat{y} = 99.2 + 0.979$ LP
LA	0.000	0.643	0.633	$\hat{y} = 133{,}936 + 101$ LA
LOT	0.003	0.215	0.194	$\hat{y} = 310{,}191 + 19{,}217$ LOT

2. Nếu sử dụng đúng 2 biến dự đoán để dự đoán giá nhà, phương trình hồi quy 2 biến dự đoán (predictor) nào ở trên là tốt nhất? Tại sao?
3. Phương trình hồi quy nào trong số các phương trình hồi quy trên là tốt nhất để dự đoán giá nhà? Tại sao?
4. Một ngôi nhà được rao bán với giá niêm yết 400.000 USD, nó có diện tích là 3000 feet vuông, và diện tích đất rộng 2 mẫu. Giá trị dự đoán tốt nhất của giá bán là bao nhiêu? Giá bán dự đoán có thể là ước lượng tốt hay không? Giá trị dự đoán đó có khả năng rất chính xác không?

LAB 7:

NHẬN DẠNG CHỮ VIẾT TAY VỚI TENSORFLOW

Nội dung:

1. Mô tả dữ liệu MNIST cho bài toán nhận dạng chữ viết tay
2. Quá trình học và dự báo
3. Bài tập áp dụng

1. Mô tả dữ liệu MNIST cho bài toán nhận dạng chữ viết tay:

Mô tả dữ liệu: tập dữ liệu về chữ số viết tay là MNIST. Dữ liệu chia thành 3 phần, 55 000 dòng cho tập train, 10 000 dòng cho tập test và 5 000 dòng cho tập validation.

Yêu cầu: xây dựng mô phân loại cho bài toán trên, sử dụng mạng neural.

Hướng dẫn thực hiện:

Import thư viện và load tập dữ liệu vào biến mnist

```
from tensorflow.examples.tutorials.mnist import input_data
import tensorflow as tf
import matplotlib.pyplot as plt
```

Vì tập dữ liệu được chứa trong các tệp nén, ta tạo một thư mục chứa các file tên là MNIST_data.

```
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
```

Kết quả như sau:

```
Extracting MNIST_data/train-images-idx3-ubyte.gz
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
```

Dữ liệu chia thành 3 phần, 55 000 dòng cho tập train, 10 000 dòng cho tập test và 5 000 dòng cho tập validation.

Với kích thước lớn của tập dữ liệu này, một phương pháp hay là chia nó thành các bó nhỏ hơn, đặc biệt là khi nó cần được phân tích như một tập huấn luyện. Để giúp bạn làm điều này, TensorFlow sử dụng hàm next_batch (n), cho phép bạn trích xuất n phần tử từ tập huấn luyện. Bất cứ khi nào hàm next_batch (n) được gọi, n phần tử sau sẽ được trích xuất, cho đến khi đạt được phần cuối của tập huấn luyện.

Để xem 10 phần tử đầu tiên của tập huấn luyện, ta thực hiện như sau:

```
pixels,real_values = mnist.train.next_batch(10)
print("list of values loaded",real_values)
```

Kết quả như sau:

```
list of values loaded [2 6 8 3 4 2 0 9 8 7]
```

Để xem tiếp 10 phần tử tiếp theo của tập huấn luyện, ta thực hiện như sau:

```
pixels,real_values = mnist.train.next_batch(10)
print("list of values loaded",real_values)
```

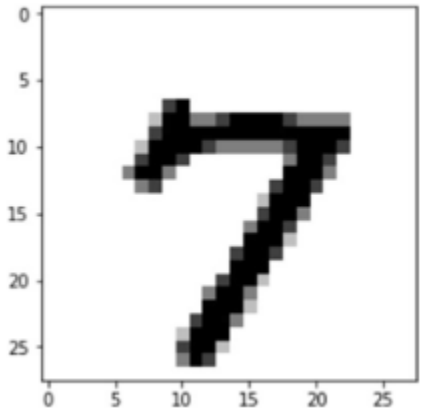
Kết quả như sau:

```
list of values loaded [6 1 8 5 0 1 8 4 7 3]
```

Nếu bạn muốn xem hình ảnh của chữ số viết tay, bạn có thể sử dụng matplotlib.

```
image = np.reshape(pixels[1,:],[28,28])
plt.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
plt.show()
```

Kết quả như sau:



2. Quá trình học và dự báo:

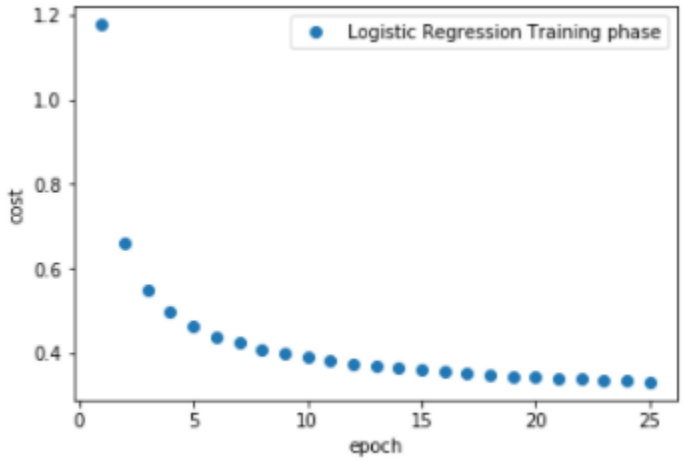
```
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

import tensorflow as tf
import matplotlib.pyplot as plt

# Parameters
learning_rate = 0.01
training_epochs = 25
batch_size = 100
display_step = 1

# tf Graph Input
x = tf.placeholder("float", [None, 784]) # mnist data image of shape 28*2
8=784
y = tf.placeholder("float", [None, 10]) # 0-
9 digits recognition => 10 classes

# Create model

# Set model weights
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))

evidence = tf.matmul(x, W) + b

# Construct model
activation = tf.nn.softmax(evidence) # Softmax

# Minimize error using cross entropy
```

```
cross_entropy = y*tf.log(activation)
cost = tf.reduce_mean(-tf.reduce_sum(cross_entropy,reduction_indices=1))

optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cos
t)

#Plot settings
avg_set = []
epoch_set=[]

# Initializing the variables
init = tf.global_variables_initializer()

# Launch the graph
with tf.Session() as sess:
    sess.run(init)

    # Training cycle
    for epoch in range(training_epochs):
        avg_cost = 0.
        total_batch = int(mnist.train.num_examples/batch_size)
        # Loop over all batches
        for i in range(total_batch):
            batch_xs, batch_ys = mnist.train.next_batch(batch_size)
            sess.run(optimizer, feed_dict={x: batch_xs, y: batch_ys})
            avg_cost += sess.run(cost, feed_dict={x: batch_xs, y: batch_y
s})/total_batch
        if epoch % display_step == 0:
            print("Epoch:", '%04d' % (epoch+1), "cost=", "{:.9f}".format(
avg_cost))
        avg_set.append(avg_cost)
        epoch_set.append(epoch+1)
    print("Training phase finished")

    plt.plot(epoch_set,avg_set, 'o', label='Logistic Regression Training
phase')
    plt.ylabel('cost')
    plt.xlabel('epoch')
    plt.legend()
    plt.show()

    # Test model
    correct_prediction = tf.equal(tf.argmax(activation, 1), tf.argmax(y,
1))
```

```
    # Calculate accuracy
    accuracy = tf.reduce_mean(tf.cast(correct_prediction, "float"))
    print("Model accuracy:", accuracy.eval({x: mnist.test.images, y: mnis
t.test.labels}))
```

Kết quả như sau:

```
Extracting MNIST_data/train-images-idx3-ubyte.gz
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
Epoch: 0001 cost= 1.176361134
Epoch: 0002 cost= 0.662538510
Epoch: 0003 cost= 0.550689667
Epoch: 0004 cost= 0.496738935
Epoch: 0005 cost= 0.463713668
Epoch: 0006 cost= 0.440845339
Epoch: 0007 cost= 0.423968329
Epoch: 0008 cost= 0.410662182
Epoch: 0009 cost= 0.399876185
Epoch: 0010 cost= 0.390923975
Epoch: 0011 cost= 0.383305770
Epoch: 0012 cost= 0.376747700
Epoch: 0013 cost= 0.371062683
Epoch: 0014 cost= 0.365925885
Epoch: 0015 cost= 0.361331244
Epoch: 0016 cost= 0.357197133
Epoch: 0017 cost= 0.353523670
Epoch: 0018 cost= 0.350157993
Epoch: 0019 cost= 0.347037680
Epoch: 0020 cost= 0.344143576
Epoch: 0021 cost= 0.341464736
Epoch: 0022 cost= 0.338996708
Epoch: 0023 cost= 0.336639690
Epoch: 0024 cost= 0.334515039
Epoch: 0025 cost= 0.332482831
Training phase finished
```




```
Model accuracy: 0.9143
```

3. Bài tập:

1. Thực hiện bài nhận dạng chữ viết tay với thư viện Keras.
2. Phân loại ảnh với dữ liệu CIFAR10 dataset bao gồm 50,000 training set và 10.000 test set ảnh màu kích thước 32x32 cho 10 thể loại khác nhau (máy bay, ô tô, thuyền, chim, chó, mèo, ngựa,...).

LAB 8:

PHÂN TÍCH DỮ LIỆU DẠNG VĂN BẢN VỚI NLTK

Nội dung:

1. Giới thiệu về thư viện NLTK
2. Tìm 1 từ với NLTK
3. Phân tích tần số của các từ
4. Lựa chọn các từ trong văn bản
5. Bigrams và collocations
6. Sử dụng văn bản trên mạng
7. Rút trích văn bản từ trang html
8. Phân tích cảm xúc người dùng
9. Bài tập áp dụng

1. Giới thiệu về thư viện NLTK:

Trong các phần trước, ta đã học cách phân tích dữ liệu dạng số hoặc dạng bảng thông qua các biểu thức toán học hoặc kỹ thuật thống kê. Nhưng hầu hết dữ liệu bao gồm dạng văn bản, quy tắc ngữ pháp khác nhau tùy vào ngôn ngữ nào đó. Trong một văn bản, các từ và nghĩa của từ có thể là nguồn thông tin rất hữu ích. Trong lab này, ta sẽ tìm hiểu một số kỹ thuật phân tích văn bản bằng thư viện NLTK cho phép bạn thực hiện các phép toán phức tạp. Hơn nữa giúp ta hiểu được một phần quan trọng trong phân tích dữ liệu.

Cài đặt thư viện NLTK:

Vào của sổ command line, gõ lệnh pip install NLTK

Import thư viện NLTK và download công cụng NLTK

```
import nltk
nltk.download_shell()
```

Kết quả:

```
NLTK Downloader
---------------------------------------------------------------------------
-
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
-
Downloader> l

Packages:
  [ ] abc..................... Australian Broadcasting Commission 2006
  [ ] alpino.................. Alpino Dutch Treebank
  [*] averaged_perceptron_tagger Averaged Perceptron Tagger
  [ ] averaged_perceptron_tagger_ru Averaged Perceptron Tagger (Russian)
  [ ] basque_grammars..... Grammars for Basque
```

```
  [ ] biocreative_ppi..... BioCreAtIvE (Critical Assessment of Information
                           Extraction Systems in Biology)
  [ ] bllip_wsj_no_aux.... BLLIP Parser: WSJ Model
  [ ] book_grammars....... Grammars from NLTK Book
  [ ] brown............... Brown Corpus
  [ ] brown_tei........... Brown Corpus (TEI XML Version)
  [ ] cess_cat............ CESS-CAT Treebank
  [ ] cess_esp............ CESS-ESP Treebank
  [ ] chat80.............. Chat-80 Data Files
  [ ] city_database....... City Database
  [ ] cmudict............. The Carnegie Mellon Pronouncing Dictionary
(0.6)
  [ ] comparative_sentences Comparative Sentence Dataset
  [ ] comtrans............ ComTrans Corpus Sample
  [ ] conll2000........... CONLL 2000 Chunking Corpus
  [ ] conll2002........... CONLL 2002 Named Entity Recognition Corpus
Hit Enter to continue:
  [ ] conll2007........... Dependency Treebanks from CoNLL 2007 (Catalan
                           and Basque Subset)
  [ ] crubadan............ Crubadan Corpus
  [ ] dependency_treebank. Dependency Parsed Treebank
  [ ] dolch............... Dolch Word List
  [ ] europarl_raw........ Sample European Parliament Proceedings Parallel
                           Corpus
  [ ] floresta............ Portuguese Treebank
  [ ] framenet_v15........ FrameNet 1.5
  [ ] framenet_v17........ FrameNet 1.7
  [ ] gazetteers.......... Gazeteer Lists
  [ ] genesis............. Genesis Corpus
  [*] gutenberg........... Project Gutenberg Selections
  [ ] ieer................ NIST IE-ER DATA SAMPLE
  [ ] inaugural........... C-Span Inaugural Address Corpus
  [ ] indian.............. Indian Language POS-Tagged Corpus
  [ ] jeita............... JEITA Public Morphologically Tagged Corpus (in
                           ChaSen format)
  [ ] kimmo............... PC-KIMMO Data Files
  [ ] knbc................ KNB Corpus (Annotated blog corpus)
  [ ] large_grammars...... Large context-free and feature-based grammars
                           for parser comparison
Hit Enter to continue:
  [ ] lin_thesaurus....... Lin's Dependency Thesaurus
  [ ] mac_morpho.......... MAC-MORPHO: Brazilian Portuguese news text with
                           part-of-speech tags
  [ ] machado............. Machado de Assis -- Obra Completa
  [ ] masc_tagged......... MASC Tagged Corpus
  [ ] maxent_ne_chunker... ACE Named Entity Chunker (Maximum entropy)
  [ ] maxent_treebank_pos_tagger Treebank Part of Speech Tagger (Maximum
entropy)
  [ ] moses_sample........ Moses Sample Models
  [*] movie_reviews....... Sentiment Polarity Dataset Version 2.0
  [ ] mte_teip5........... MULTEXT-East 1984 annotated corpus 4.0
  [ ] mwa_ppdb............ The monolingual word aligner (Sultan et al.
                           2015) subset of the Paraphrase Database.
  [*] names............... Names Corpus, Version 1.3 (1994-03-29)
```

```
  [ ] nombank.1.0......... NomBank Corpus 1.0
  [ ] nonbreaking_prefixes Non-Breaking Prefixes (Moses Decoder)
  [ ] nps_chat............ NPS Chat
  [ ] omw................. Open Multilingual Wordnet
  [ ] opinion_lexicon..... Opinion Lexicon
  [ ] panlex_swadesh...... PanLex Swadesh Corpora
  [ ] paradigms........... Paradigm Corpus
  [ ] pe08................ Cross-Framework and Cross-Domain Parser
                           Evaluation Shared Task
Hit Enter to continue:
  [ ] perluniprops........ perluniprops: Index of Unicode Version 7.0.0
                           character properties in Perl
  [ ] pil................. The Patient Information Leaflet (PIL) Corpus
  [ ] pl196x.............. Polish language of the XX century sixties
  [ ] porter_test......... Porter Stemmer Test Files
  [ ] ppattach............ Prepositional Phrase Attachment Corpus
  [ ] problem_reports..... Problem Report Corpus
  [ ] product_reviews_1... Product Reviews (5 Products)
  [ ] product_reviews_2... Product Reviews (9 Products)
  [ ] propbank............ Proposition Bank Corpus 1.0
  [ ] pros_cons........... Pros and Cons
  [ ] ptb................. Penn Treebank
  [*] punkt............... Punkt Tokenizer Models
  [ ] qc.................. Experimental Data for Question Classification
  [ ] reuters............. The Reuters-21578 benchmark corpus, ApteMod
                           version
  [ ] rslp................ RSLP Stemmer (Removedor de Sufixos da Lingua
                           Portuguesa)
  [ ] rte................. PASCAL RTE Challenges 1, 2, and 3
  [ ] sample_grammars..... Sample Grammars
  [ ] semcor.............. SemCor 3.0
Hit Enter to continue:
  [ ] senseval............ SENSEVAL 2 Corpus: Sense Tagged Text
  [ ] sentence_polarity... Sentence Polarity Dataset v1.0
  [ ] sentiwordnet........ SentiWordNet
  [ ] shakespeare......... Shakespeare XML Corpus Sample
  [ ] sinica_treebank..... Sinica Treebank Corpus Sample
  [ ] smultron............ SMULTRON Corpus Sample
  [ ] snowball_data....... Snowball Data
  [ ] spanish_grammars.... Grammars for Spanish
  [ ] state_union......... C-Span State of the Union Address Corpus
  [*] stopwords........... Stopwords Corpus
  [ ] subjectivity........ Subjectivity Dataset v1.0
  [ ] swadesh............. Swadesh Wordlists
  [ ] switchboard......... Switchboard Corpus Sample
  [ ] tagsets............. Help on Tagsets
  [ ] timit............... TIMIT Corpus Sample
  [ ] toolbox............. Toolbox Sample Files
  [ ] treebank............ Penn Treebank Sample
  [ ] twitter_samples..... Twitter Samples
  [ ] udhr2............... Universal Declaration of Human Rights Corpus
                           (Unicode Version)
  [ ] udhr................ Universal Declaration of Human Rights Corpus
Hit Enter to continue:
```

```
  [ ] unicode_samples..... Unicode Samples
  [ ] universal_tagset.... Mappings to the Universal Part-of-Speech Tagset
  [ ] universal_treebanks_v20 Universal Treebanks Version 2.0
  [ ] vader_lexicon....... VADER Sentiment Lexicon
  [ ] verbnet............. VerbNet Lexicon, Version 2.1
  [ ] webtext............. Web Text Corpus
  [ ] wmt15_eval.......... Evaluation data from WMT15
  [ ] word2vec_sample..... Word2Vec Sample
  [ ] wordnet............. WordNet
  [ ] wordnet_ic.......... WordNet-InfoContent
  [ ] words............... Word Lists
  [ ] ycoe................ York-Toronto-Helsinki Parsed Corpus of Old
                           English Prose

Collections:
  [P] all-corpora......... All the corpora
  [P] all-nltk............ All packages available on nltk_data gh-pages
                           branch
  [P] all................. All packages
  [P] book................ Everything used in the NLTK Book
  [P] popular............. Popular packages
  [P] tests............... Packages for running tests
Hit Enter to continue:
  [ ] third-party......... Third-party data packages

([*] marks installed packages; [P] marks partially installed collections)

---------------------------------------------------------------------------
-
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
-
Downloader> d

Download which package (l=list; x=cancel)?
  Identifier> gutenberg
    Downloading package gutenberg to
        C:\Users\nelli\AppData\Roaming\nltk_data...
      Package gutenberg is already up-to-date!

---------------------------------------------------------------------------
-
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
-
Downloader> q
```

Để có thể tạo một số các ví dụ để tìm hiểu về thư viện, bạn cần một số các văn bản để làm việc. Một nguồn văn bản tuyệt vời phù hợp cho mục đích này là kho ngữ liệu Gutenberg, có trong bộ sưu tập kho ngữ liệu. Kho ngữ liệu Gutenberg gồm các văn bản được trích xuất từ kho lưu trữ điện tử có tên là dự án Gutenberg (http: // www.gutenberg.org/). Có hơn 25.000 sách điện tử trong kho lưu trữ này. Để tải xuống gói này, trước tiên hãy nhập tùy chọn d để tải xuống. Công cụ sẽ hỏi bạn cho tên gói, vì vậy bạn nhập tên gutenberg.

```
---------------------------------------------------------------------------
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
Downloader> d

Download which package (l=list; x=cancel)?
  Identifier> gutenberg
```

Tại thời điểm này, các gói sẽ bắt đầu download.

Đối với những lần sau, nếu bạn đã biết tên của gói bạn muốn tải xuống, chỉ cần nhập lệnh nltk.download() với tên gói là tham số đầu vào. Thao tác này sẽ không mở công cụ NLTK Downloader, nhưng sẽ trực tiếp tải xuống gói yêu cầu. Vì vậy, thao tác trước đó tương đương với việc viết lệnh sau:

```
nltk.download('gutenberg')
```

Sau khi hoàn tất, bạn có xem tên các tập tin có trong gói 'gutenberg' bằng lệnh fileids().

```
gb = nltk.corpus.gutenberg
print("Gutenberg files : ", gb.fileids())
```

Kết quả:

```
Gutenberg files :  ['austen-emma.txt', 'austen-persuasion.txt', 'austen-
sense.txt', 'bible-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt',
'burgess-busterbrown.txt', 'carroll-alice.txt', 'chesterton-ball.txt',
'chesterton-brown.txt', 'chesterton-thursday.txt', 'edgeworth-
parents.txt', 'melville-moby_dick.txt', 'milton-paradise.txt',
'shakespeare-caesar.txt', 'shakespeare-hamlet.txt', 'shakespeare-
macbeth.txt', 'whitman-leaves.txt']
```

Để truy cập nội dung bên trong của một trong các tập tin này, trước tiên bạn chọn một tập tin, chẳng hạn Macbeth của Shakespeare (shakespeare-macbeth.txt), sử dụng hàm words (), rồi gán cho một biến nào đó.

```
macbeth = nltk.corpus.gutenberg.words('shakespeare-macbeth.txt')
```

Nếu bạn muốn biết chiều dài của văn bản trên (bao nhiêu từ), bạn dùng hàm len()

```
len(macbeth)
```

Kết quả:

```
23140
```

Nếu bạn muốn hiển thị 10 từ đầu tiên của tập tin

```
macbeth [:10]
```

Kết quả:

```
['[',
 'The',
 'Tragedie',
 'of',
 'Macbeth',
 'by',
 'William',
 'Shakespeare',
 '1603',
 ']']
```

Hàm trên trích ra 10 từ đầu tiên là tiêu đề của tập tin, dấu ngoặc vuông được tính là 1.

Ta muốn trích 5 câu đầu tiên của tập tin (một câu được kẹp trong cặp ngoặc vuông), ta dùng hàm sent()

```
macbeth_sents = nltk.corpus.gutenberg.sents('shakespeare-macbeth.txt')
macbeth_sents[:5]
```

Kết quả:

```
[['[',
  'The',
  'Tragedie',
  'of',
  'Macbeth',
  'by',
  'William',
  'Shakespeare',
  '1603',
  ']'],
 ['Actus', 'Primus', '.'],
 ['Scoena', 'Prima', '.'],
 ['Thunder', 'and', 'Lightning', '.'],
 ['Enter', 'three', 'Witches', '.']]
```

2. Tìm 1 từ với NLTK:

Tìm từ 'Stage' xuất hiện trong văn bản text

```
text = nltk.Text(macbeth)
text.concordance('Stage')
```

Kết quả:

```
Displaying 3 of 3 matches:
nts with Dishes and Seruice ouer the Stage . Then enter Macbeth Macb . If
it we with mans Act , Threatens his bloody Stage : byth ' Clock ' tis Day,
And yet d struts and frets his houre vpon the Stage , And then is heard no
more . It is
```

Tìm từ xuất hiện trước và sau từ 'Stage'

```
text.common_contexts(['Stage'])
```

Kết quả:

```
the then bloody byth the_and
```

Tìm từ tương tự từ 'Stage'

```
text.similar('Stage')
```

Kết quả:

```
fogge ayre bleeding reuolt good shew heeles skie other sea feare
consequence heart braine seruice herbenger lady round deed doore
```

3. Phân tích tần số của các từ

Phân tích số lần xuất hiện của từ:

Muốn xem 10 từ thông dụng nhất trong văn bản xuất hiện bao nhiêu lần, dùng lệnh most_common()

```
fd = nltk.FreqDist(macbeth)
fd.most_common(10)
```

Kết quả:

```
[(',', 1962),
 ('.', 1235),
 ("'", 637),
 ('the', 531),
 (':', 477),
```

```
 ('and', 376),
 ('I', 333),
 ('of', 315),
 ('to', 311),
 ('?', 241)]
```

Stopword là những từ thông dụng, thường ít có ý nghĩa trong quá trình phân tích văn bản và thường cần được loại bỏ.

Muốn download stopword

```
nltk.download('stopwords')
```

Kết quả:

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\nelli\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

Muốn xem các stopword trong tiếng Anh, dùng lệnh

```
sw = set(nltk.corpus.stopwords.words('english'))
print(len(sw))
list(sw)[:10]
```

Kết quả:

```
179
['once',
 'o',
 "hasn't",
 'i',
 "you'd",
 'because',
 'yourselves',
 'being',
 "wouldn't",
 'been']
```

Có 179 stopword trong từ vựng tiếng Anh. Ta sẽ loại bỏ các từ stopword trong biến macbeth

```
macbeth_filtered = [w for w in macbeth if w.lower() not in sw]
len(macbeth_filtered)
```

Kết quả:

```
14946
```

```
fd = nltk.FreqDist(macbeth_filtered)
fd.most_common(10)
```

Kết quả:

```
[(',', 1962),
 ('.', 1235),
 ("'", 637),
 (':', 477),
 ('?', 241),
 ('Macb', 137),
 ('haue', 117),
 ('-', 100),
 ('Enter', 80),
 ('thou', 63)]
```

Bây giờ, 10 từ phổ biến nhất đầu tiên được trả về, các stopword đã được loại bỏ nhưng vẫn còn các dấu câu, ta cần loại bỏ các dấu câu theo lệnh sau:

```
import string
punctuation = set(string.punctuation)
macbeth_filtered2 = [w.lower() for w in macbeth if w.lower() not in sw an
d w.lower() not in punctuation]
fd = nltk.FreqDist(macbeth_filtered2)
fd.most_common(10)
```

Kết quả:

```
[('macb', 137),
 ('haue', 122),
 ('thou', 90),
 ('enter', 81),
 ('shall', 68),
 ('macbeth', 62),
 ('vpon', 62),
 ('thee', 61),
 ('macd', 58),
 ('vs', 57)]
```

4. Lựa chọn các từ trong văn bản

Lựa chọn các từ trong văn bản:

Một dạng khác trong việc xử lý và phân tích dữ liệu là quá trình chọn một từ dựa vào 1 đặc điểm nào đó. Ví dụ như bạn cần quan tâm rút trích từ dựa vào độ dài của từ.

Rút trích các từ có độ dài lớn nhất, ví dụ các từ có độ dài lớn hơn 12 ký tự , dùng lệnh sau:

```
long_words = [w for w in macbeth if len(w)> 12]
sorted(long_words)
```

Kết quả:

['Assassination',
'Chamberlaines',
'Distinguishes',
'Gallowgrosses',
'Metaphysicall',
'Northumberland',
'Voluptuousnesse',
'commendations',
'multitudinous',
'supernaturall',
'vnaccompanied']

Rút trích các từ có chứa chuỗi ‘ious’

```
ious_words = [w for w in macbeth if 'ious' in w]
ious_words = set(ious_words)
sorted(ious_words)
```

Kết quả:

['Auaricious',
'Gracious',
'Industrious',
'Iudicious',
'Luxurious',
'Malicious',

'Obliuious',
'Pious',
'Rebellious',
'compunctious',
'furious',
'gracious',
'pernicious',
'pernitious',
'pious',
'precious',
'rebellious',
'sacrilegious',
'serious',
'spacious',
'tedious']

5. Bigrams và collocations

Một phần cơ bản trong phân tích văn bản là xem xét các cặp từ thay thế cho các từ đơn. Từ 'is' và 'yellow' là 1 ví dụ về bigram vì chúng có thể kết hợp lại và có ý nghĩa, vì thế "is yellow" có thể được tìm thấy trong văn bản. Ví dụ "fast food", "pay attention", "good morning",..,những bigram này gọi là collocation.

Lọc các bigram sau khi đã loại các stopword và các dấu câu, dùng lệnh sau:

```
bgrms = nltk.FreqDist(nltk.bigrams(macbeth_filtered2))
bgrms.most_common(15)
```

Kết quả:

[(('enter', 'macbeth'), 16),
(('exeunt', 'scena'), 15),
(('thane', 'cawdor'), 13),
(('knock', 'knock'), 10),
(('st', 'thou'), 9),
(('thou', 'art'), 9),
(('lord', 'macb'), 9),
(('haue', 'done'), 8),
(('macb', 'haue'), 8),
(('good', 'lord'), 8),
(('let', 'vs'), 7),
(('enter', 'lady'), 7),
(('wee', 'l'), 7),
(('would', 'st'), 6),
(('macbeth', 'macb'), 6)]

Ngoài bigram ra, còn có trigram, sự kết hợp của 3 từ, ta dùng lệnh trigrams()

```
tgrms = nltk.FreqDist(nltk.trigrams (macbeth_filtered2))
tgrms.most_common(10)
```

Kết quả:

[(('knock', 'knock', 'knock'), 6),
(('enter', 'macbeth', 'macb'), 5),
(('enter', 'three', 'witches'), 4),

(('exeunt', 'scena', 'secunda'), 4),
(('good', 'lord', 'macb'), 4),
(('three', 'witches', '1'), 3),
(('exeunt', 'scena', 'tertia'), 3),
(('thunder', 'enter', 'three'), 3),
(('exeunt', 'scena', 'quarta'), 3),
(('scena', 'prima', 'enter'), 3)]

6. Sử dụng văn bản trên mạng

Các ví dụ trước, chúng ta đã làm việc với văn bản sử dụng thư viện NLTK và gói 'gutenberg'. Nhưng trong thực tế, bạn sẽ cần phải truy cập Internet để trích xuất văn bản và thu thập nó như một kho tài liệu được sử dụng để phân tích với NLTK.

Import thư viện và mở url để đọc file

```
from urllib import request
url = "http://www.gutenberg.org/files/2554/2554-0.txt"
response = request.urlopen(url)
raw = response.read().decode('utf8')
raw[:75]
```

Kết quả:

'\ufeffThe Project Gutenberg EBook of Crime and Punishment, by Fyodor
Dostoevsky\r'

Thay bằng các lệnh sau:

```
from urllib import request
url = "http://www.gutenberg.org/files/2554/2554-0.txt"
response = request.urlopen(url)
raw = response.read().decode('utf8-sig')
raw[:75]
```

Kết quả:

'The Project Gutenberg EBook of Crime and Punishment, by Fyodor Dostoevsky\
r\n'

Thực hiện các lệnh sau:

```
tokens = nltk.word_tokenize (raw)
webtext = nltk.Text (tokens)
webtext[:12]
```

Kết quả:

['The',
'Project',
'Gutenberg',
'EBook',
'of',
'Crime',
'and',
'Punishment',
',',

'by',
'Fyodor',
'Dostoevsky']

7. Rút trích văn bản từ trang html

Trong ví dụ vừa rồi, ta tạo 1 copus NLTK từ 1 văn bản download từ Internet. Nhưng hầu như các văn bản trên Internet ở dạng trang html. Trong phần này, chúng ta sẽ tìm hiểu cách rút trích văn bản từ trang html.

```
url = "http://news.bbc.co.uk/2/hi/health/2284783.stm"
html = request.urlopen(url).read().decode('utf8')
html[:120]
```

Kết quả:

'<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/REC-html40/loose.dtd">\r\n<html>\r\n<hea'

Ta dùng thư viện bs4 (BeautifulSoup) cung cấp cho bạn các trình phân tích cú pháp phù hợp có thể nhận dạng HTML và trích xuất văn bản.

```
from bs4 import BeautifulSoup
raw = BeautifulSoup(html, "lxml").get_text()
tokens = nltk.word_tokenize(raw)
text = nltk.Text(tokens)
```

8. Phân tích cảm xúc người dùng

Phân tích cảm xúc là một lĩnh vực nghiên cứu gần đây để đánh giá quan điểm của người dùng về một chủ đề nào đó. Việc phân tích này dựa trên về các kỹ thuật khác nhau sử dụng phân tích văn bản, các lĩnh vực trên các mạng xã hội và các diễn đàn.

Nhờ nhận xét và đánh giá của người dùng, các thuật toán phân tích cảm xúc có thể đánh giá mức độ đánh giá hoặc đánh giá dựa trên các từ khóa nhất định. Mức độ đánh giá được gọi là ý kiến và có ba giá trị có thể có: **positive, neutral hoặc negative**. Việc đánh giá ý kiến này do đó trở thành một dạng phân loại.

Mục đích của ví dụ dưới đây là tìm các từ lặp lại nhiều nhất dạng positive và dạng negative hoặc các từ khóa liên quan đến một ý kiến nào đó, sử dụng thuật toán Naïve Bayes để phân loại cảm xúc người dùng dựa vào các đoạn văn bản movie_reviews của họ.

Trước tiên, download các movie review dùng lệnh sau:

```
nltk.download('movie_reviews')
```

Sau đó xây dựng tập train dựa vào copus trên. Tạo 1 mảng tên là documents. Mảng này chứa cột đầu tiên là nội dung review của người dùng, và cột thứ 2 là cột đánh giá **positive, neutral, or negative.** Sau đó trộn các dòng này ngẫn nhiên.

```
import random
reviews = nltk.corpus.movie_reviews
documents = [(list(reviews.words(fileid)), category)
for category in reviews.categories()
for fileid in reviews.fileids(category)]
random.shuffle(documents)
```

Xem nội dung review đầu tiên (dòng 0, cột 0)

```
first_review = ' '.join(documents[0][0])
print(first_review)
```

Kết quả:

topless women talk about their lives falls into that category that I mentioned in the devil ' s advocate : movies that have a brilliant beginning but don ' t know how to end . it begins by introducing us to a selection of characters who all know each other . there is liz , who oversleeps and so is running late for her appointment , prue who is getting married ,...

Xem kết quả review đầu tiên (dòng 0, cột 1)

```
documents[0][1]
```

Kết quả:

'neg'

Ta cần tạo bảng phân phối tần số các từ trong copus, bảng này cần chuyển sang dạng list, ta dùng hàm list()

```
all_words = nltk.FreqDist(w.lower() for w in reviews.words())
word_features = list(all_words)
```

Sau đó, bước tiếp theo là xác định một hàm để tính toán các đặc trưng, tức là những từ đủ quan trọng để thiết lập ý kiến của một review.

```
def document_features(document, word_features):
    document_words = set(document)
    features = {}
    for word in word_features:
        features['{}'.format(word)] = (word in document_words)
    return features
```

Khi bạn định nghĩa hàm document_features(), bạn tạo 1 tập các documents

```
featuresets = [(document_features(d,word_features), c) for (d,c) in docum
ents]
len(featuresets)
```

Kết quả:

```
2000
```

Tạo tập train và tập test: 1500 dòng đầu dùng cho tập train và 500 dòng còn lại dùng cho tập test để đánh giá độ chính xác của mô hình.

```
train_set, test_set = featuresets[1500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
```

Dùng thuật toán Naïve Bayes để phân loại, dùng thư viện NLTK. Sau đó tính toán độ chính xác của thuật toán

```
train_set, est_set = featuresets[1500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
print(nltk.classify.accuracy(classifier, test_set))
```

Kết quả

```
0.85
```

Chúng ta đã hoàn tất việc phân tích, dưới đây các từ với trọng số lớn nhất của các review được đánh giá là positive và negative

```
classifier.show_most_informative_features(10)
```

Kết quả:

```
Most Informative Features
                   blame = True              neg : pos    =     12.7 : 1.0
             wonderfully = True              pos : neg    =     12.4 : 1.0
                    era = True               pos : neg    =      9.5 : 1.0
                  inept = True               neg : pos    =      8.1 : 1.0
                  anger = True               pos : neg    =      7.8 : 1.0
                   porn = True               neg : pos    =      7.3 : 1.0
                 finest = True               pos : neg    =      7.2 : 1.0
             assistant = True                pos : neg    =      7.2 : 1.0
                   mass = True               pos : neg    =      6.6 : 1.0
               realizes = True               pos : neg    =      6.6 : 1.0
```

9. Bài tập áp dụng

1. Viết chương trình Python với thư viện NLTK để liệt kê các tên của copus.
2. Viết chương trình Python với thư viện NLTK để liệt kê danh sách các stopword bằng các ngôn ngữ khác nhau.
3. Viết chương trình Python với thư viện NLTK để kiểm tra danh sách các stopword bằng các ngôn ngữ khác nhau.
4. Viết chương trình Python với thư viện NLTK để loại bỏ các stopword từ một văn bản đã cho.
5. Viết chương trình Python với thư viện NLTK bỏ qua các stopword từ danh sách các stopword.
6. Viết một chương trình Python với thư viện NLTK để tìm định nghĩa và ví dụ của một từ đã cho bằng WordNet từ Wikipedia,
7. Viết chương trình Python với thư viện NLTK để tìm tập hợp các từ đồng nghĩa và trái nghĩa của một từ nào đó.
8. Viết chương trình Python với thư viện NLTK để có cái nhìn tổng quan về bộ tag, chi tiết của một tag cụ thể trong bộ tag và chi tiết về một số bộ tag liên quan, sử dụng biểu thức chính quy.
9. Viết chương trình Python với thư viện NLTK để so sánh sự giống nhau của hai danh từ đã cho.
10. Viết chương trình Python với thư viện NLTK để so sánh sự giống nhau của hai động từ đã cho.
11. Viết chương trình Python với thư viện NLTK để tìm số lượng tên nam và nữ trong các tên kho ngữ liệu. In tên 10 nam và nữ đầu tiên. Lưu ý: Kho văn bản tên chứa tổng cộng khoảng 2943 nam (male.txt) và 5001 nữ (Female.txt) tên. Kho được biên soạn bởi Kantrowitz, Ross.
12. Viết chương trình Python với thư viện NLTK để in 15 kết hợp ngẫu nhiên đầu tiên được gắn nhãn nam và được gắn nhãn tên nữ từ kho tên.
13. Viết chương trình Python với thư viện NLTK để trích xuất ký tự cuối cùng của tất cả các tên được gắn nhãn và tạo mảng mới với chữ cái cuối cùng của mỗi tên và nhãn được liên kết.