ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

LÊ VĂN ĐÔNG – NGUYỄN VĂN MINH TRIẾT

Lab01

Thành phố Hồ Chí Minh – 2021

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

LÊ VĂN ĐÔNG – NGUYỄN VĂN MINH TRIẾT

Lab01

|Giáo viên hướng dẫn|

Thầy Lê Hoài Bắc

Thầy Nguyễn Khánh Toàn

Thành phố Hồ Chí Minh – 2021

Họ và tên thành viên:

Lê Văn Đông – 19127363

Nguyễn Văn Minh Triết – 19127599

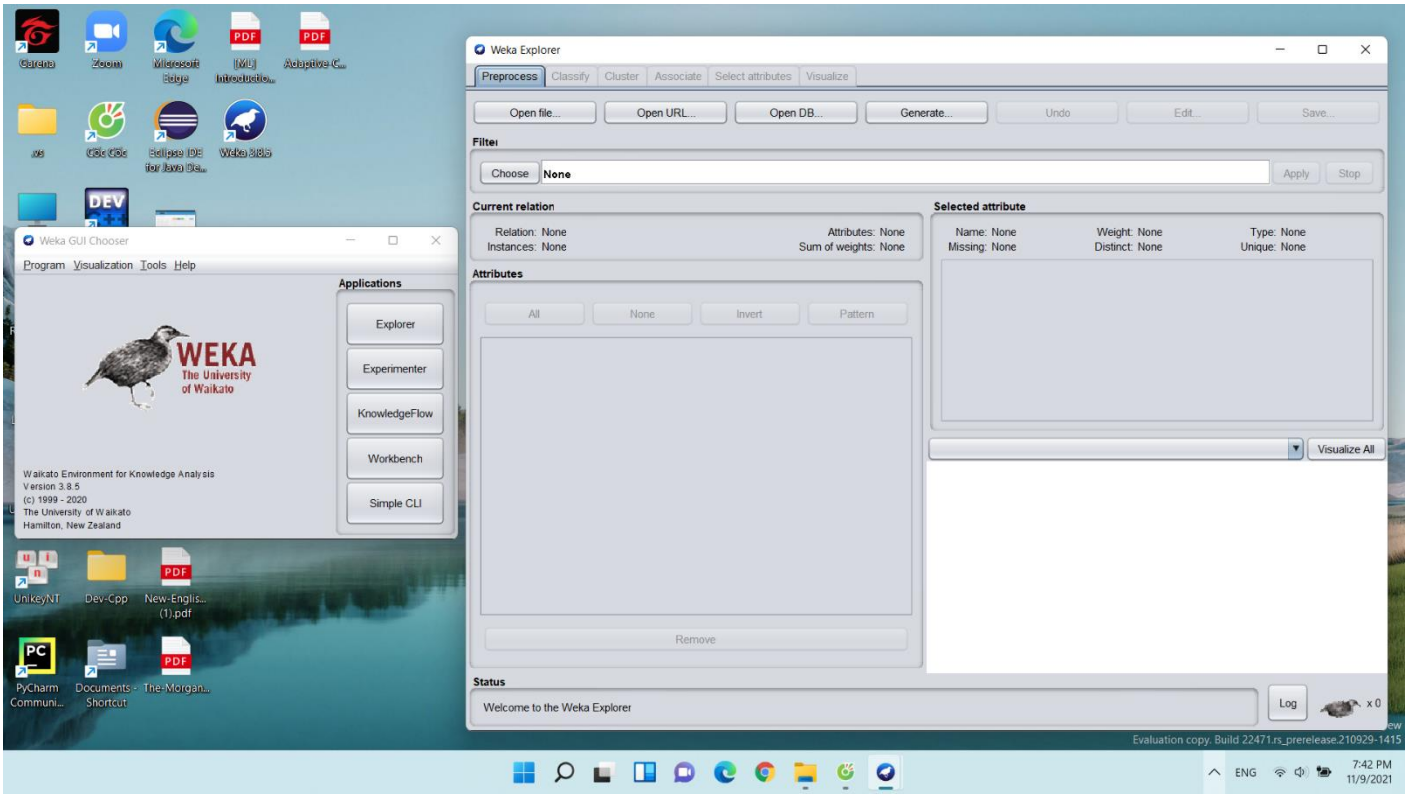
Công việc	Hoàn thành	Tên
Đọc dữ liệu vào Waka.	100%	Nguyễn Văn Minh Triết
Khám phá tập dữ liệu Weather.	100%	Nguyễn Văn Minh Triết
Khám phá tập dữ liệu Tín dụng Đức.	100%	Lê Văn Đông
Liệt kê các cột thiếu dữ liệu.	100%	Nguyễn Văn Minh Triết
Đếm số dòng bị thiếu dữ liệu.	100%	Lê Văn Đông
Điền giá trị bị thiếu bằng phương pháp mean, median.	100%	Nguyễn Văn Minh Triết
Xóa các dòng bị thiếu dữ liệu với ngưỡng tỉ lệ thiếu cho trước.	100%	Lê Văn Đông
Xóa các cột bị thiếu dữ liệu với ngưỡng tỉ lệ thiếu cho trước.	100%	Nguyễn Văn Minh Triết
Xóa các mẫu bị trùng lặp.	100%	Nguyễn Văn Minh Triết
Chuẩn hóa một thuộc tính numeric bằng phương pháp min-max và Z-score.	100%	Lê Văn Đông
Tính giá trị biểu thức thuộc tính.	100%	Lê Văn Đông

Mục lục

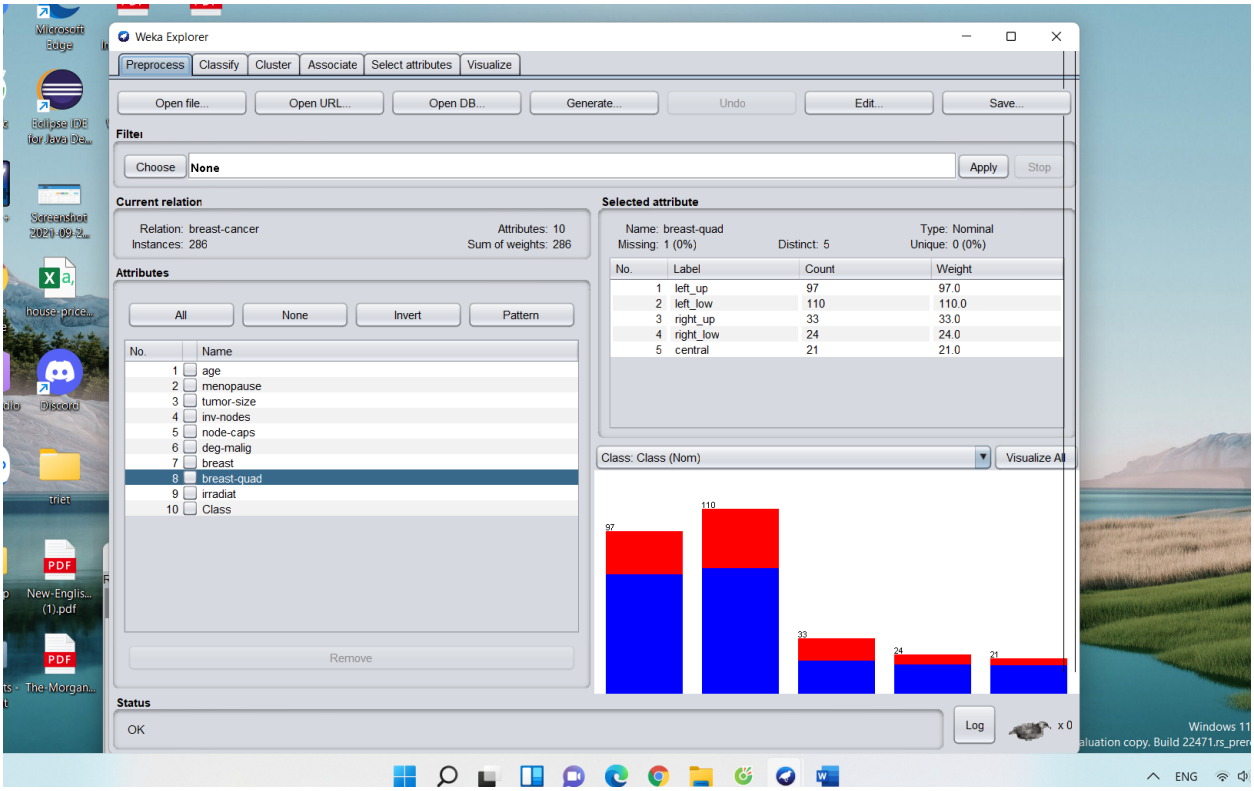
REPORT

1. Cài đặt Weka

- Màng hình giao diện cài đặt Weka – Explorer:



- ý nghĩa của các nhóm điều khiển trong tab Preprocess:

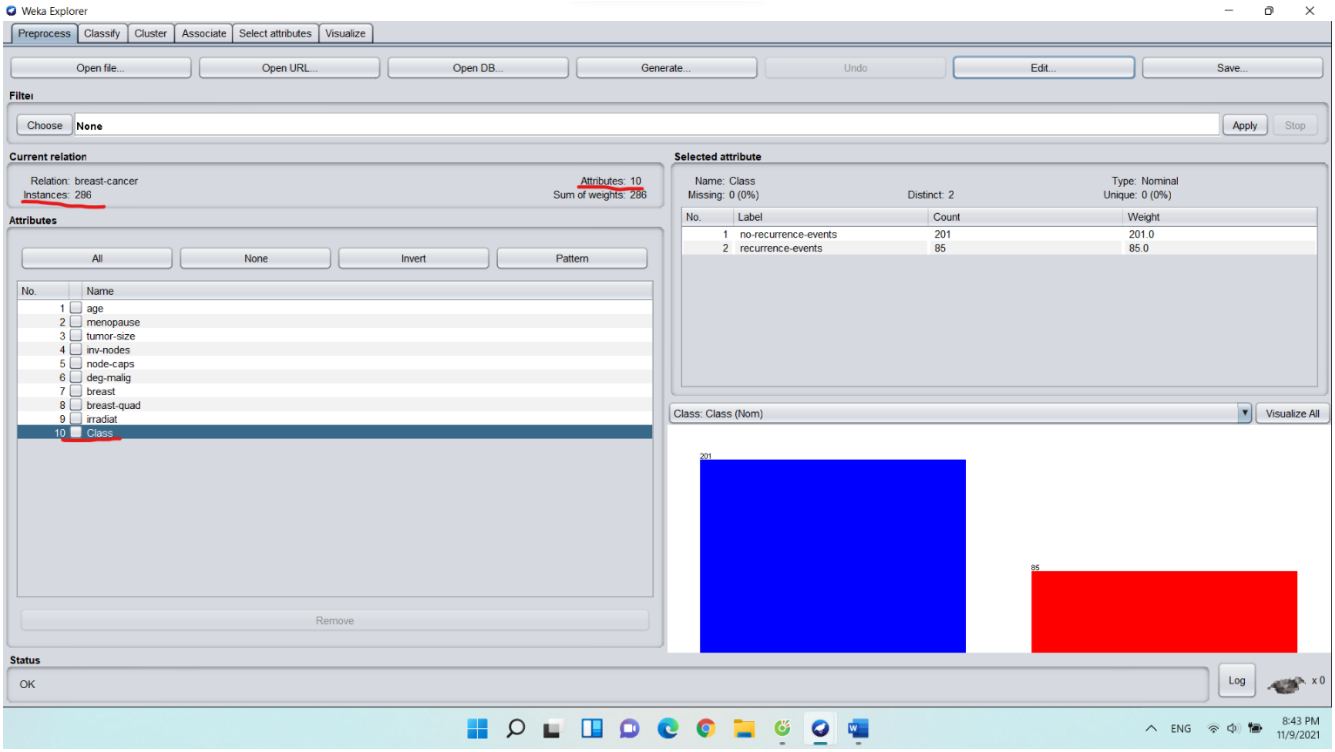


- Current relation:
  + Relation: tên của thư mục data đã được chọn.
  + Attributes: hiển thị tổng số thuộc tính có trong dữ liệu.
  + Instances: hiển thị tổng số các trường hợp có trong dữ liệu.
  + Sum of weights: tổng trường hợp có trong data.
- Attributes: là nơi hiển thị tên tất cả các thuộc tính của data và là nơi để chúng ta thao tác chọn phần dữ liệu được hiển thị trong phần đồ thị ở góc trái dưới của giao diện.
- Selected attributes: gồm các phần cơ bảng:
  + Name: tên của thuộc tính được chọn để hiển thị.
  + Missing: số các giá trị bị mất của thuộc tính đang được chọn.
  + Distinct: số lượng các giá trị của thuộc tính (thuộc tính gồm các giá trị nào).
  + 1 cái bảng: dùng để cho người dùng thấy tên của các giá trị thuộc thuộc tính, số lượng cũng như khối lượng của chúng (thường thì 2 cái này bằng nhau).
- Classify: dùng để chọn bộ phân loại, tùy chọn kiểm tra, thuộc tính lớp, đào tạo trình phân loại, đầu ra của phân loại.

- Cluster: chọn trình phân nhóm, chọn chế độ cụm, làm việc với bộ lọc, phân nhóm học tập.
- Associate: thiết lập và liên kết học tập.
- Select attributes: tìm kiếm và đánh giá thuộc tính, thực hiện lựa chọn thuộc tính.
- Visualize: hình ảnh hóa các thuộc tính, đưa ra ma trận biểu đồ phân tán.

2. Làm quen với Weka

2.1 Đọc dữ liệu vào Weka



a) Tập dữ liệu có bao nhiêu mẫu (instances)

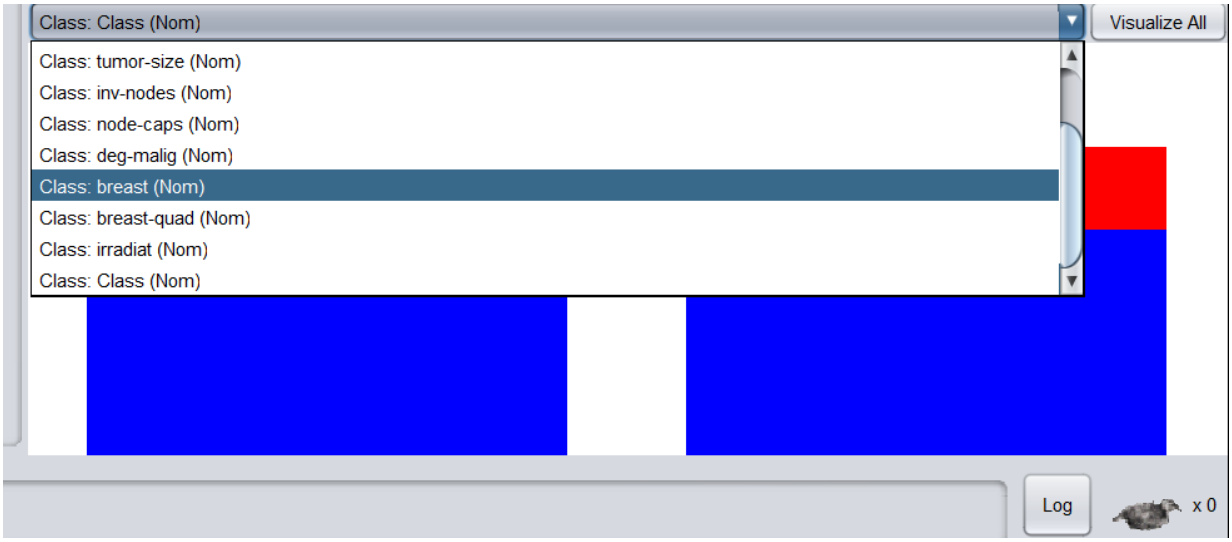
- Tập dữ liệu có 286 mẫu (instances).

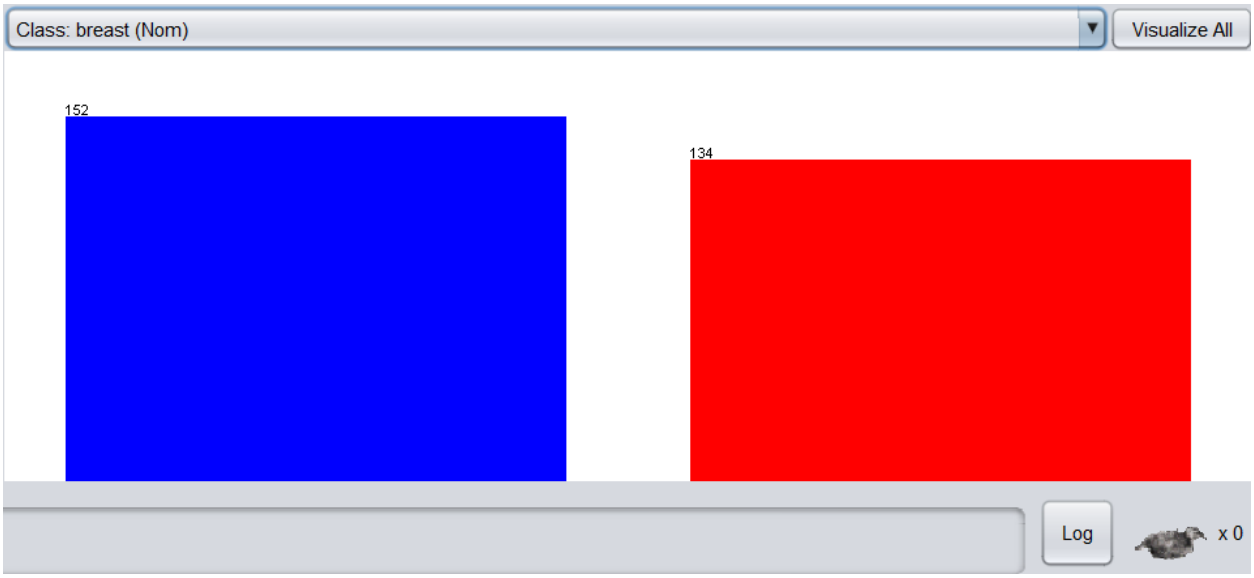
b) Tập dữ liệu có bao nhiêu thuộc tính (attributes)

- Tập dữ liệu có 10 thuộc tính (attributes).

c) Thuộc tính nào được dùng làm lớp (class)? Có thể thay đổi thuộc tính dùng làm lớp hay không? Nếu có thì bằng cách nào?

- Thuộc tính được dùng làm lớp class là thuộc tính Class. Có thể thay đổi thuộc tính bằng các lớp khác.

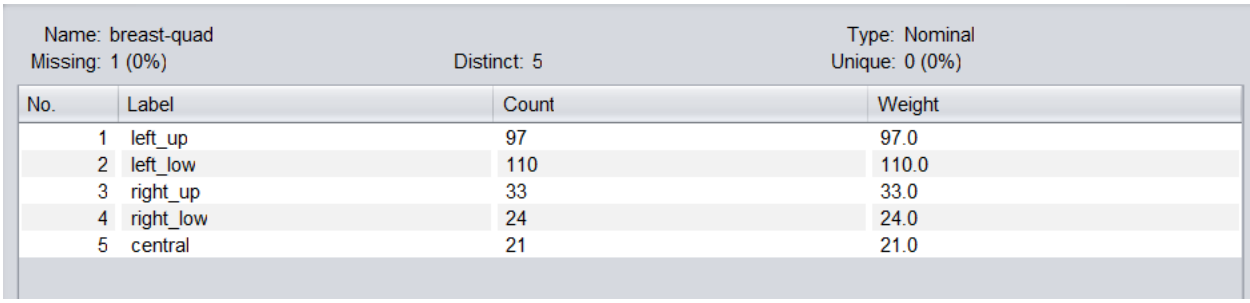




- Nhấp chuột vào phần Class -> xuất hiện 1 list các thuộc tính -> nếu bạn muốn thay đổi thuộc tính nào dùng làm Class thì nhấm chọn thuộc tính đó. Ở trong ví dụ là thuộc tính breast được chọn làm class mới.

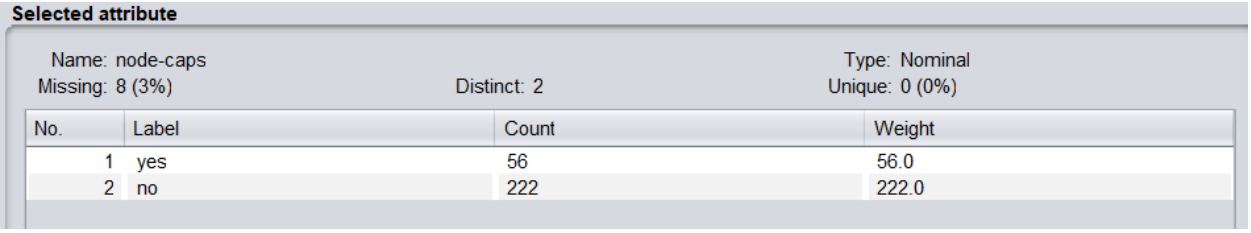
d) Tìm hiểu chi tiết từng thuộc tính trong khung Attributes và cho biết: có bao nhiêu thuộc tính bị thiếu dữ liệu (missing values)? Thuộc tính nào thiếu dữ liệu ít nhất/ nhiều nhất? Trình bày tổng quát các cách để giải quyết vấn đề missing values.

- Có 2 thuộc tính bị thiếu dữ liệu đó là: node-caps và breast-quad.



Name: breast-quad
Missing: 1 (0%)
Distinct: 5
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	left_up	97	97.0
2	left_low	110	110.0
3	right_up	33	33.0
4	right_low	24	24.0
5	central	21	21.0



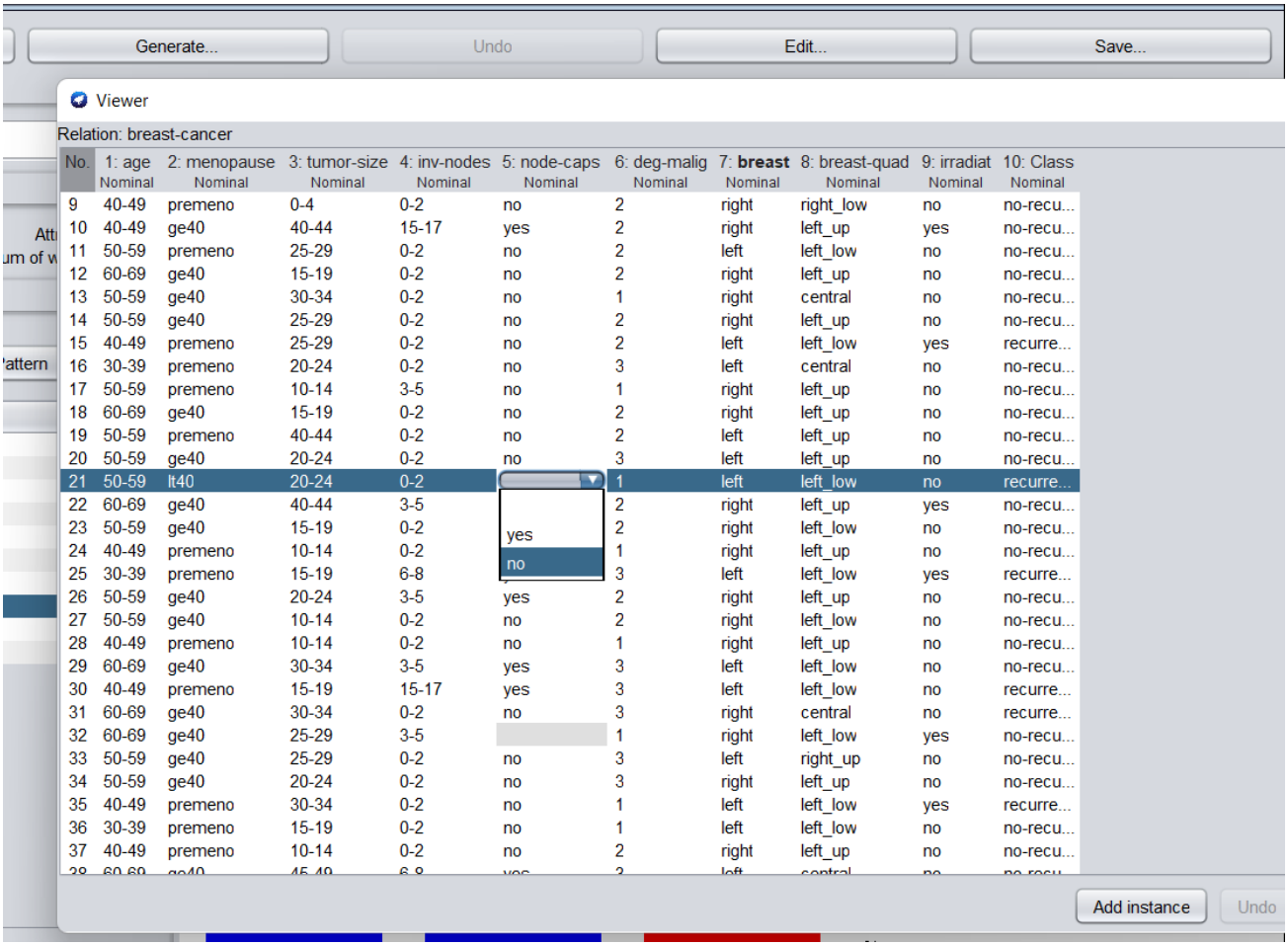
Selected attribute

Name: node-caps
Missing: 8 (3%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	yes	56	56.0
2	no	222	222.0

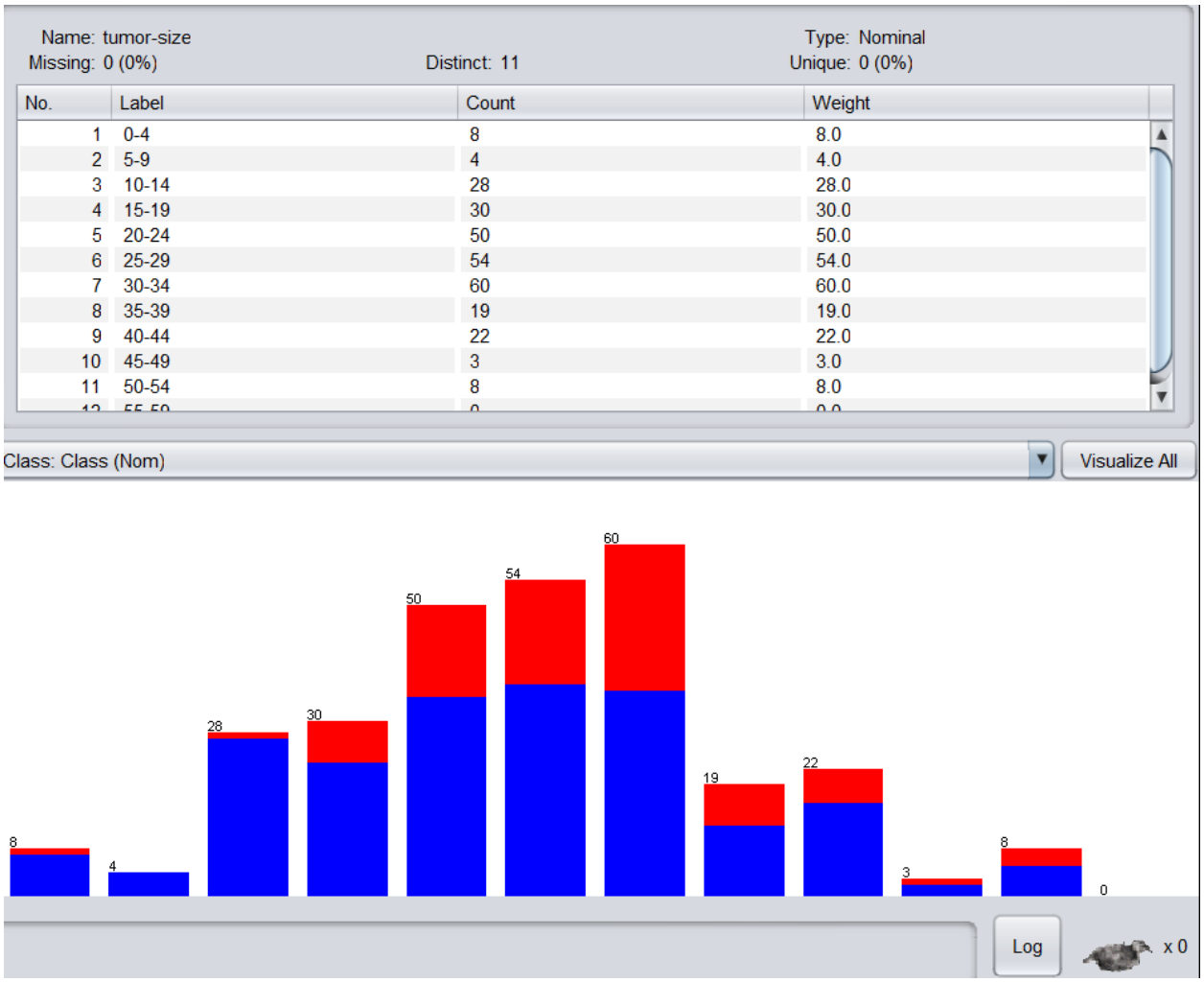
- Thuộc tính bị thiếu dữ liệu nhiều hơn là node-caps với 8 dữ liệu bị mất, trong khi đó breast-quad chỉ bị mất 1 dữ liệu.

- Cách để giải quyết vấn đề này nhanh nhất đó chính là vào phần Edit sau đó nó sẽ hiển thị cho chúng ta 1 bảng dữ liệu tổng quát của database này. Sau đó với những ô dữ liệu bị mất, người dùng có thể tự chọn kiểu dữ liệu có sẵng để thêm vào, sau đó ấn Add instance. Cách làm này chỉ hiệu quả với các thuộc tính mất ít dữ liệu (tỉ lệ mất mát vô cùng nhỏ không làm ảnh hưởng đến quá trình train tập dữ liệu về sau này) đối với lượng dữ liệu bị mất mát quá nhiều thì không nên làm theo cách này.



e) Giải thích ý nghĩa của đồ thị trong cửa sổ Explorer. Bạn đặt tên cho đồ thị này là gì? Màu xanh và màu đỏ ý nghĩa là gì? Đồ thị này biểu diễn cho cái gì?

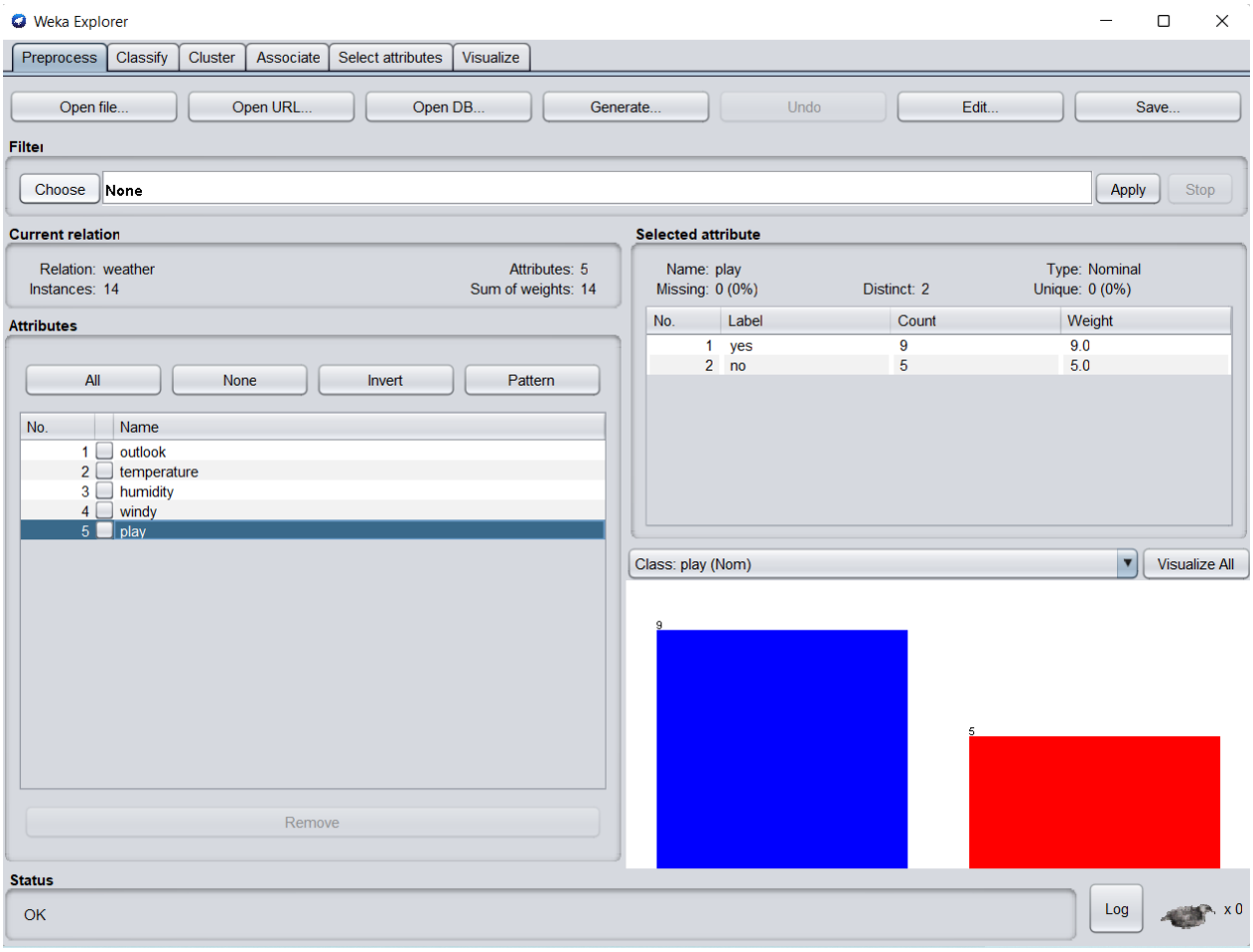
- Ý nghĩa của đồ thị trong cửa số Explorer: là nơi để hiển thị trực quang hóa tỉ trọng, số lượng và tỉ lệ của các thể hiện của thuộc tính. Nếu được đặt tên cho đồ thị này thì nó sẽ có tên là Đồ thị tỉ trọng. Và màu xanh là đại diện cho no-recurrence-events với database này, và màu đỏ đại diện cho recurrence-envent của thuộc tính dùng làm lớp chính, ở trên mỗi cột có thể thấy rõ tỉ trọng của các label này.



- Ở các thuộc tính khác: thì biểu đồ sẽ cho ra hiển thị các label theo đúng tỉ lệ của attribute được dùng làm class chính.

2.2 Khám phá tập dữ liệu Weather

a) Tập dữ liệu có bao nhiêu thuộc tính? Bao nhiêu mẫu? Phân loại các thuộc tính theo kiểu dữ liệu (categorical/numeric). Thuộc tính nào là lớp?



- Tập dữ liệu có 5 thuộc tính (attributes); có tất cả 14 mẫu (instances).

- Phân loại thuộc tính theo kiểu dữ liệu:

+ Outlook: nominal

+ Temperature: numeric

+ Humidity: numeric

+ Windy: nominal

+ Play: nominal

- Thuộc tính được dùng làm lớp chính là Play.

b) Liệt kê five-number summary của thuộc tính temperature và humidity. Weka có cung cấp những giá trị này không?

Selected attribute

Name: temperature Type: Numeric
Missing: 0 (0%) Distinct: 12 Unique: 10 (71%)

Statistic	Value
Minimum	64
Maximum	85
Mean	73.571
StdDev	6.572

Selected attribute

Name: humidity Type: Numeric
Missing: 0 (0%) Distinct: 10 Unique: 7 (50%)

Statistic	Value
Minimum	65
Maximum	96
Mean	81.643
StdDev	10.285

- Temperature:

+ Minimum: 64 + Maximum: 85

+ Mean: 73.571 + StdDev: 6.572

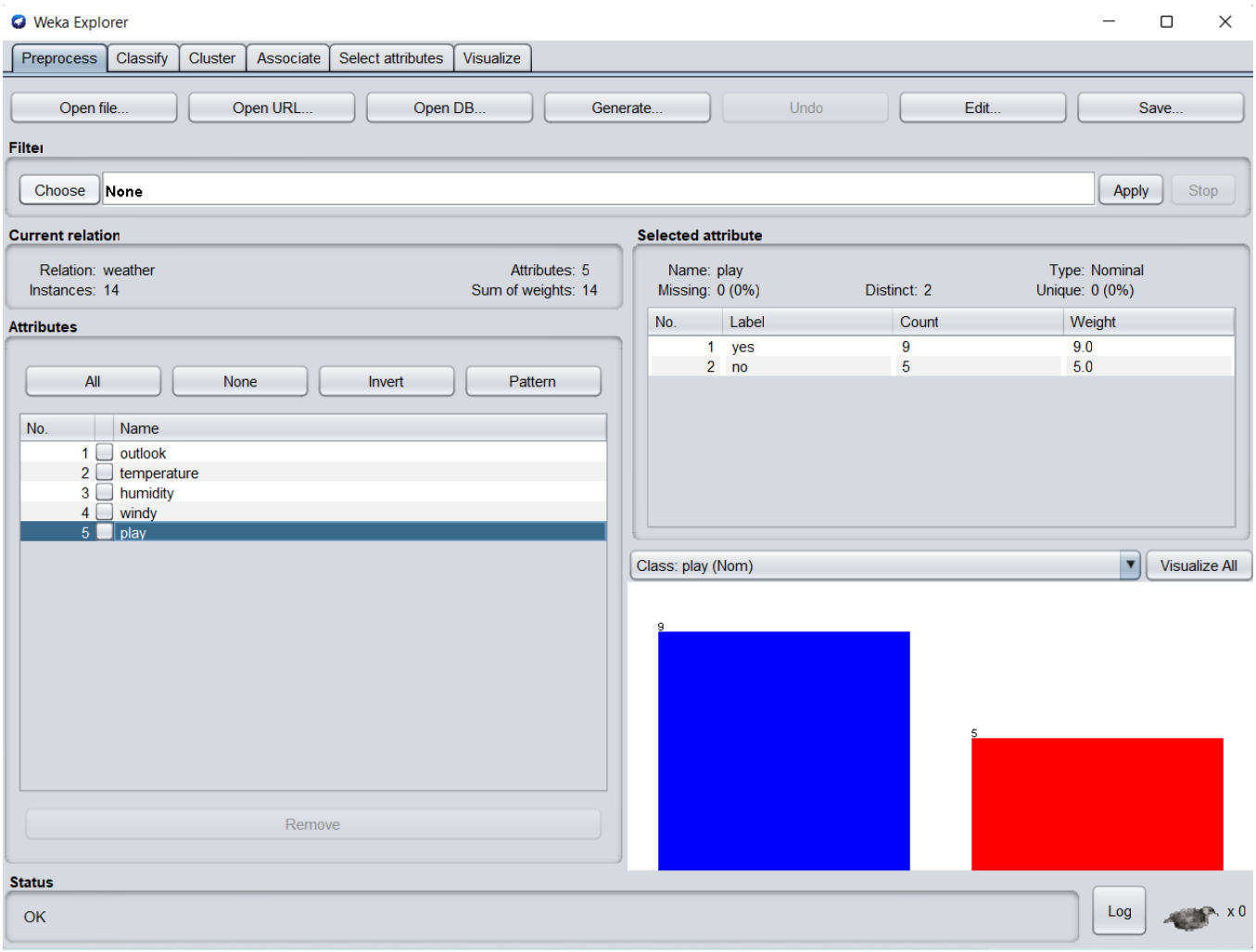
- Humidity:

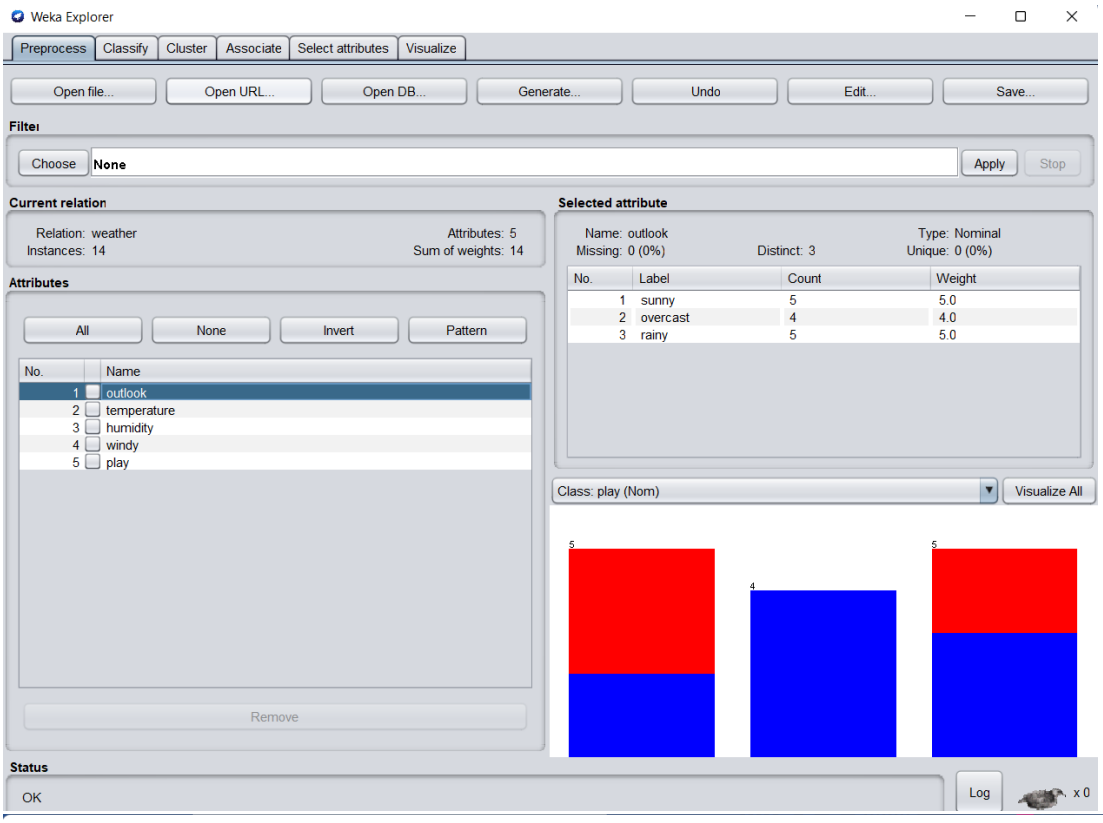
+ Minimum: 65 + Maximum: 96

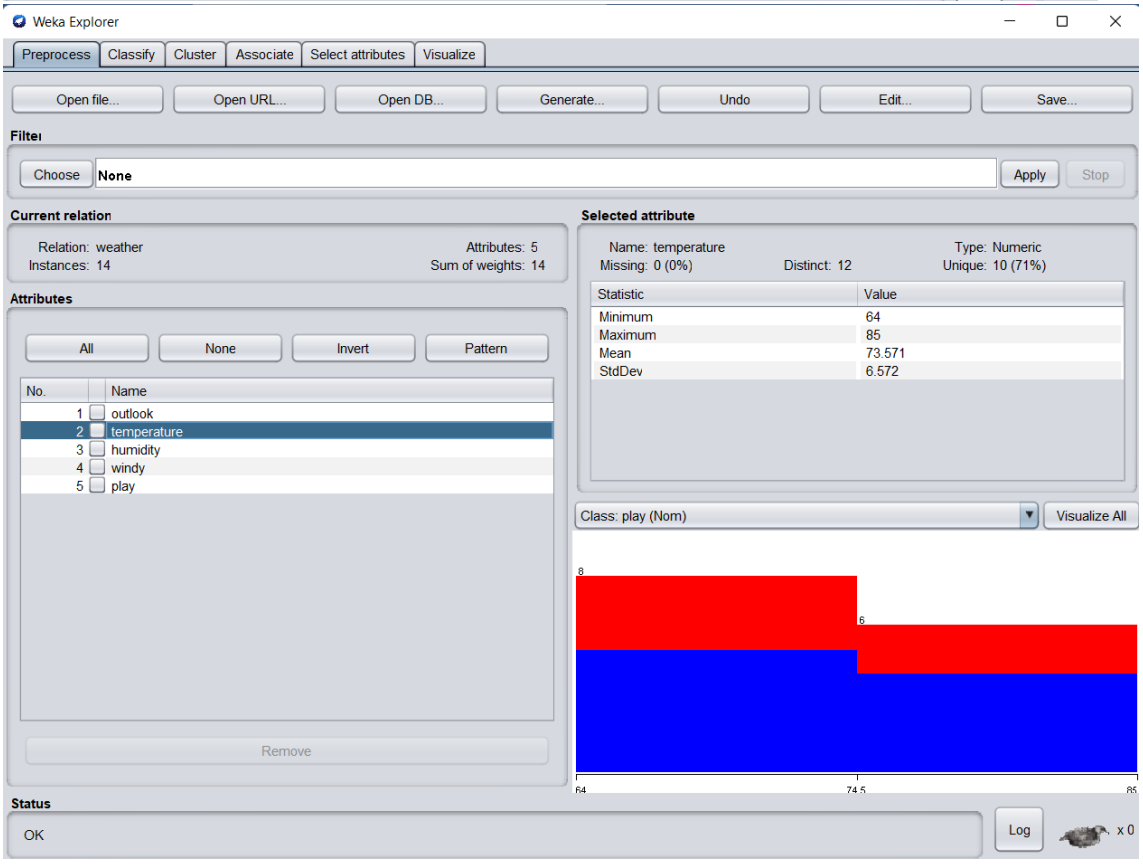
+ Mean: 81.643 + StdDev: 10.285

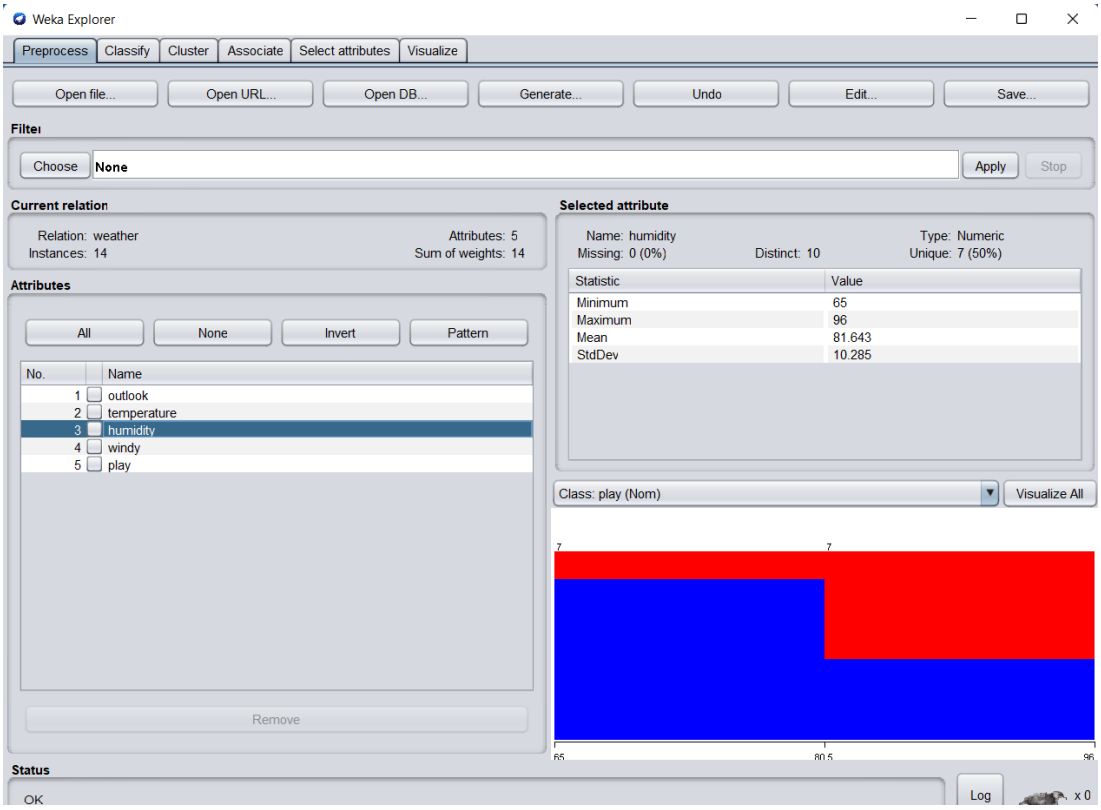
- Weka có cung cấp những giá trị này cho người dùng.

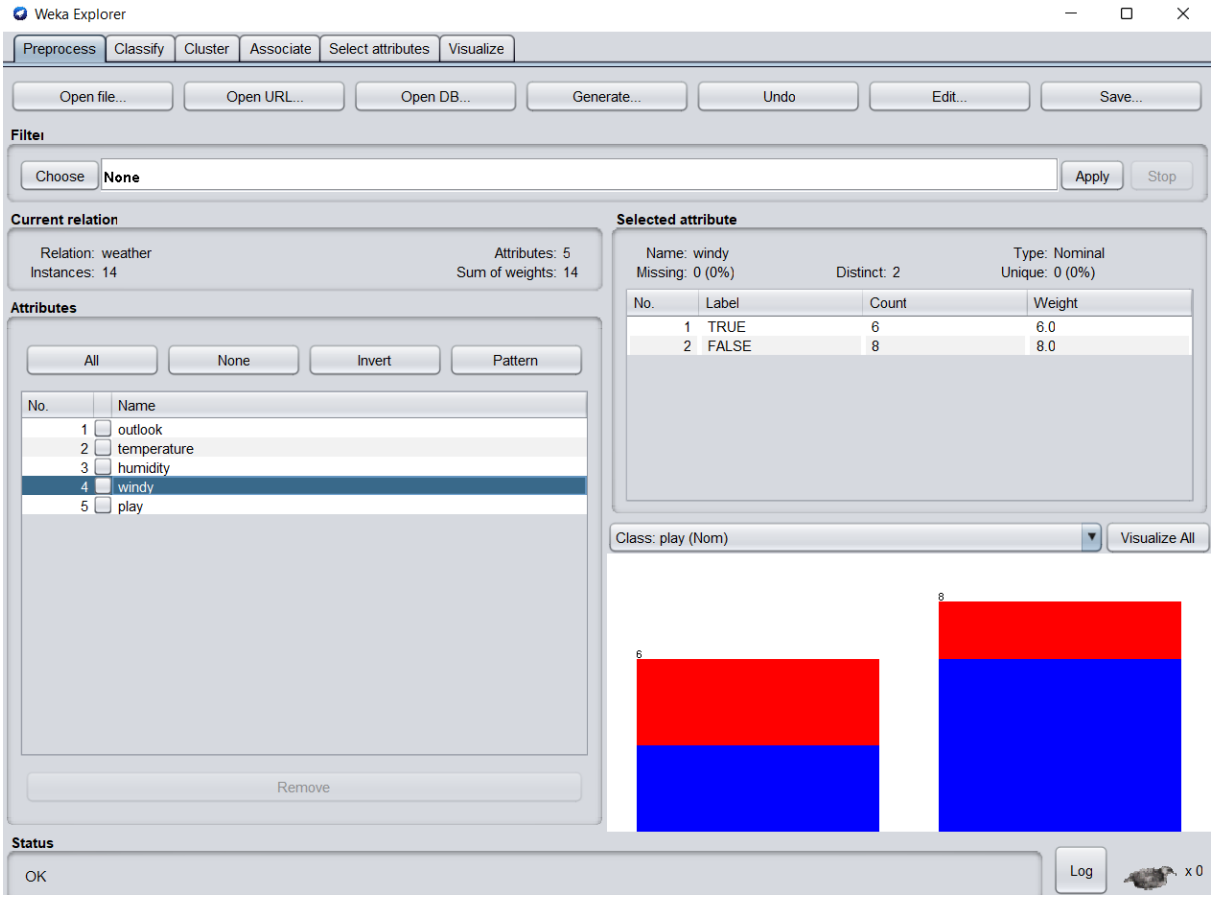
c) Lần lượt xem xét các thuộc tính khác của dataset dưới dạng đồ thị. Dán các ảnh chụp màn hình vào bài làm.


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: weather
Instances: 14
Attributes: 5
Sum of weights: 14
Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Selected attribute
Name: play
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)
No.
Label
Count
Weight
1 yes 9 9.0
2 no 5 5.0
Class: play (Nom)
Visualize All
9
5
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: weather
Instances: 14
Attributes: 5
Sum of weights: 14
Selected attribute
Name: outlook
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)
No. Label Count Weight
1 sunny 5 5.0
2 overcast 4 4.0
3 rainy 5 5.0
Attributes
All
None
Invert
Pattern
No. Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Class: play (Nom)
Visualize All
Status
OK
Log
x 0

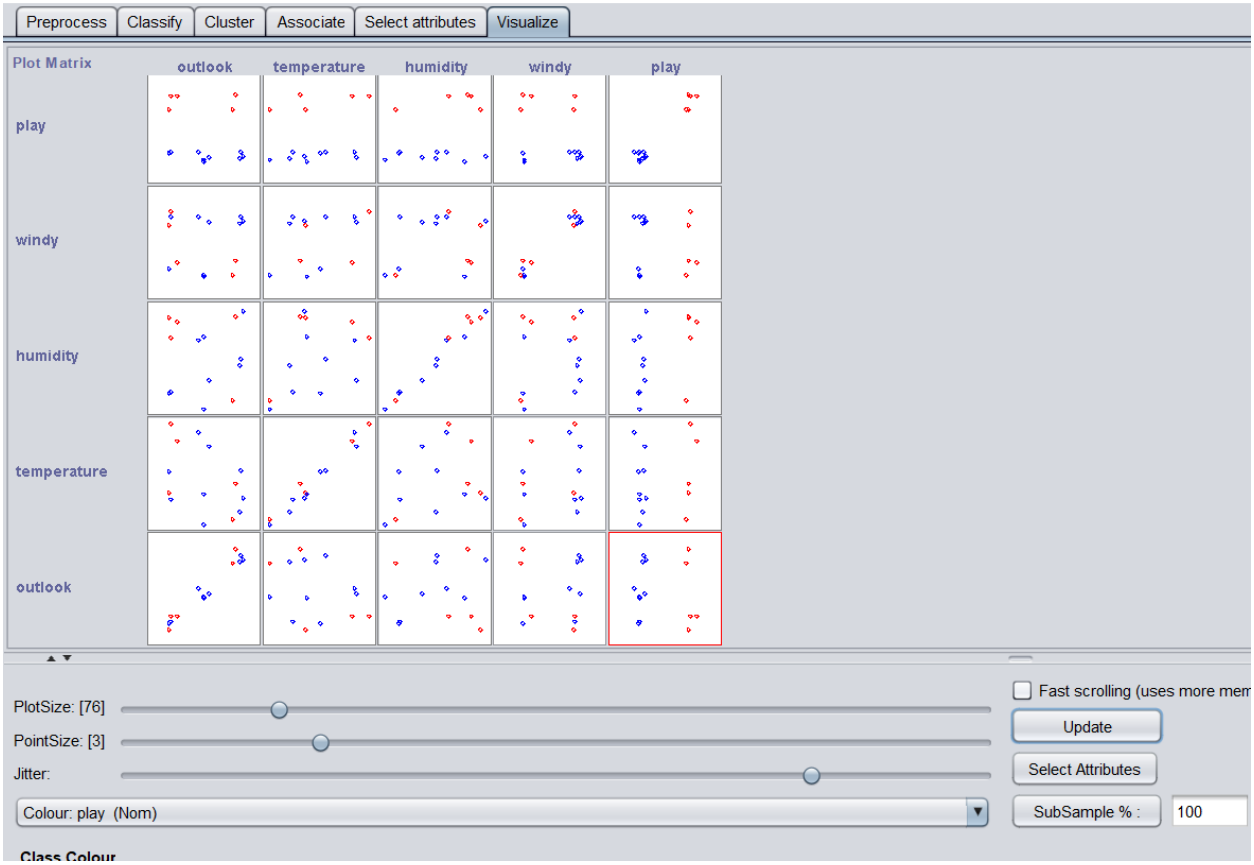

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: weather
Instances: 14
Attributes: 5
Sum of weights: 14
Selected attribute
Name: temperature
Missing: 0 (0%)
Distinct: 12
Type: Numeric
Unique: 10 (71%)
Statistic Value
Minimum 64
Maximum 85
Mean 73.571
StdDev 6.572
Attributes
All
None
Invert
Pattern
No. Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Class: play (Nom)
Visualize All
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: weather
Instances: 14
Attributes: 5
Sum of weights: 14
Selected attribute
Name: humidity
Missing: 0 (0%)
Distinct: 10
Type: Numeric
Unique: 7 (50%)
Statistic
Value
Minimum
65
Maximum
96
Mean
81.643
StdDev
10.285
Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Class: play (Nom)
Visualize All
7
7
65
80.5
96
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Stop
Current relation
Relation: weather
Instances: 14
Attributes: 5
Sum of weights: 14
Selected attribute
Name: windy
Missing: 0 (0%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)
No.
Label
Count
Weight
1 TRUE 6 6.0
2 FALSE 8 8.0
Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Class: play (Nom)
Visualize All
6
8
Status
OK
Log
x 0

d) Chuyển sang tab visualize. Thuật ngữ sử dụng trong textbook để đặt tên cho các đồ thị ở đây là gì? Chọn jitter tối đa để thấy tổng quan hơn về phân bố dữ liệu. Theo bạn có những cặp thuộc tính khác nhau nào có vẻ như tương quan với nhau không?

- Thuật ngữ để đặt tên cho các đồ thị ở đây là sự kết hợp giữa tên 2 attributes lại, ví dụ attri1-attri2 sẽ là tên 1 đồ thị ngược lại attri2-attri1 lại là tên 1 đồ thị khác, chúng khác nhau là trục hoành đồ thị này là trụ tung đồ thị kia. Với sự kết hợp như vậy chúng ta sẽ được 1 ma trận đồ thị nxn (với n là số attributes của tập dữ liệu).



- Theo tôi thì nhìn tổng quát trên 1 hàng ngang theo 1 attribute thì các đồ thị trông có vẻ tương quan với nhau. Và nhìn đồ thị theo hướng đường chéo phụ thì chúng ta thấy có sự đối xứng dữ liệu đồ thị với nhau thông qua đường chéo phụ của ma trận.
- 2 màu xanh đỏ đại diện cho 2 giá trị của attributes được dùng làm class chính.

- Đối với nhiệt độ (temperature) và độ ẩm (humidity), khi nhiệt độ và độ ẩm cùng tăng, việc chơi thể thao sẽ giảm. Nghĩa là khi nhiệt độ và độ ẩm thấp, người ta thường có xu hướng chơi thể thao nhiều hơn.

2.3 Khám phá tập dữ liệu Tín dụng Đức

a) Nội dung của phần ghi chú (comment) trong credit-g.arff (khi mở bằng 1 text editor bất kì) nói về điều gì? Tập dữ liệu có bao nhiêu mẫu? Bao nhiêu thuộc tính? Mô tả 5 thuộc tính bất kì (phải vừa có cả thuộc tính rời rạc và thuộc tính liên tục).

Các dòng có dấu % là comment. Và có rất nhiều thông tin bao gồm:

- Tiêu đề: German Credit data.
- Thông tin của source.
- Số lượng Instances.
- Mô tả về dữ liệu.
- Mô tả về số lượng các thuộc tính bao gồm: số lượng thuộc tính kiểu số và số lượng thuộc tính kiểu nomial.
- Danh sách các thuộc tính.
- Chi phí ma trận.

Tập dữ liệu có 1000 mẫu dựa vào thông tin Number of Instances.

Tập dữ liệu có 20 thuộc tính dựa vào thông tin Number of Attributes german.

Mô tả 5 thuộc tính, bao gồm thuộc tính rời rạc và thuộc tính liên tục.

- Thuộc tính rời rạc checking_status: nói về tình trạng lương tài khoản trong vòng ít nhất một năm. Miền giá trị: “< 0 DM”, “0 DM <= … < 200 DM ”, “>= 200 DM”, “Không kiểm tra tài khoản”.
- Thuộc tính liên tục duration: nói về thời gian tính theo tháng. Miền giá trị trong dữ liệu đang có: từ 4 đến 72 tháng.
- Thuộc tính credit_history: nói về lịch sử tín dụng. Miền giá trị: no credits/all paid - “Không có tín dụng/Tất cả tín dụng ở tất cả ngân

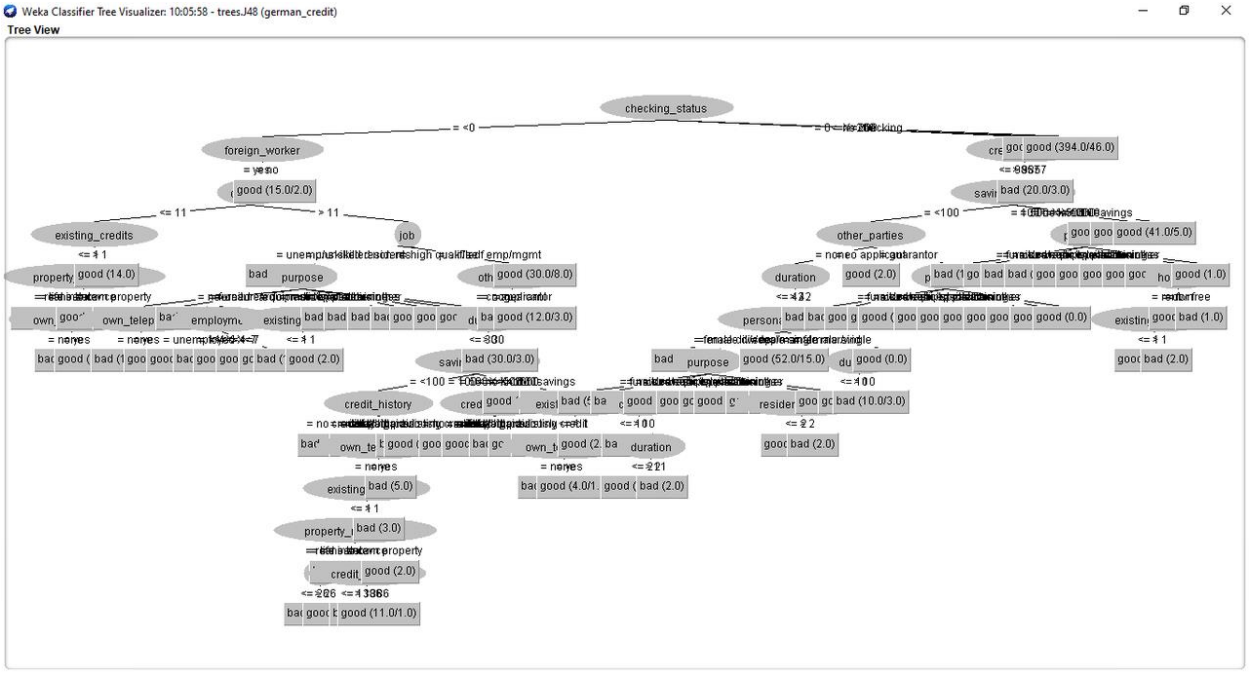
hàng đã được hoàn trả”, all paid - “Tất cả tín dụng ở ngân hàng này đã được hoàn trả”, existing paid - “Các tín dụng hiện tại cho đến nay đã hoàn trả”, delayed previously - “Trong quá khứ đã có tình trạng chậm trễ”, critical/other existing credit - “Tài khoản quan trọng/có các tín dụng khác mà không phải của ngân hàng này”.

- Thuộc tính credit_amount: mô tả số tiền tín dụng.
- Thuộc tính installment_commitment : Tỷ lệ trả góp theo phần trăm thu nhập.

---

b) Tên của thuộc tính lớp là gì? Đánh giá phân bố của các lớp, tức là cân bằng hay lệch về một lớp?

Tên của thuộc tính lớp là class. Khi vẽ cây quyết định



Ta có thể thấy cây bị lệch một lớp.

c) Sử dụng tab Select attributes. Liệt kê những lựa chọn khác nhau của Weka để chọn lọc thuộc tính, giải thích ngắn gọn từng phương pháp.

CfsSubsetEval: là một scheme-independent subset eval. Theo CfsSubsetEval thì một tập thuộc tính tốt phải thoả mãn: Có tương quan cao với thuộc tính phân lớp và tương quan yếu với các thuộc tính trong cùng tập.

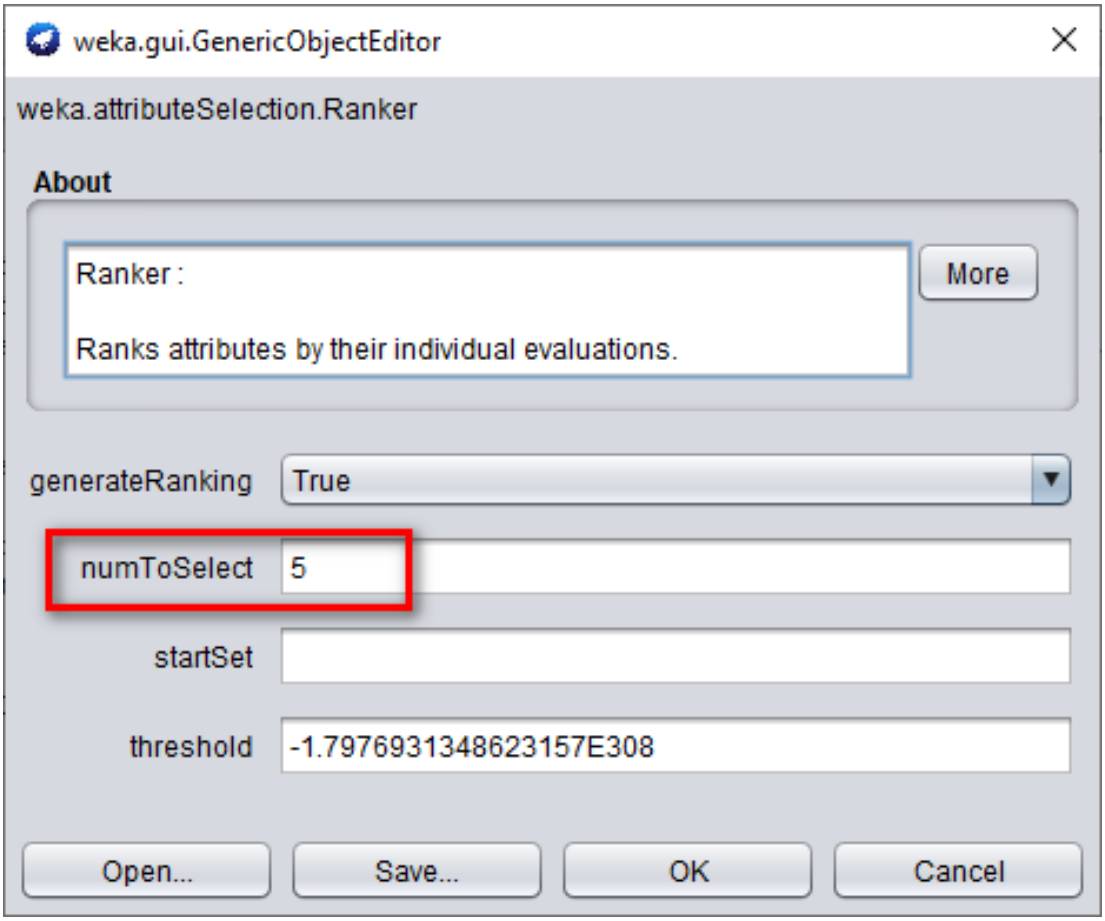
ClassifierAttributeEval : Đánh giá giá trị của một thuộc tính bằng cách sử dụng bộ phân lớp của người dùng.
ClassifierSubsetEval: Đánh giá thuộc tính dựa trên subset của tập train hoặc các bộ test độc lập không liên quan với nhau. Sử dụng bộ phân lớp để ước tính ”giá trị” của bộ thuộc tính.
CorrelationAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách đo lường mối tương quan giữa nó và lớp.
GainRatioAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách đo gain ratio liên quan đến lớp.
InfoGainAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách đo lường thông tin infomation gain liên quan đến lớp.
OneRAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách sử dụng bộ phân lớp OneR.
PrincipalComponents: Thực hiện phân tích và chuyển đổi các thành phần chính của dữ liệu. Sử dụng kết hợp với tìm kiếm Rank. Việc giảm kích thước được thực hiện bằng cách chọn đủ các yếu tố đặc trưng để chiếm một số phần trăm phương sai trong dữ liệu gốc. Có thể lọc nhiễu thuộc tính bằng cách chuyển đổi sang không gian PC, loại bỏ một số đặc điểm xấu nhất, sau đó chuyển đổi trở lại không gian ban đầu.
ReliefFAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách liên tục lấy mẫu một instance và xem xét giá trị của thuộc tính cho instance gần nhất của cùng một lớp và khác lớp.
SymmetricalUncertAttributeEval: Đánh giá giá trị của một thuộc tính bằng cách đo độ bất đối xứng với lớp.
WrapperSubsetEval: Đánh giá giá trị của một thuộc tính bằng một giải thuật học. Độ chính xác của giải thuật học trên tập thuộc tính này được xấp xỉ nhờ cross-validation.

---

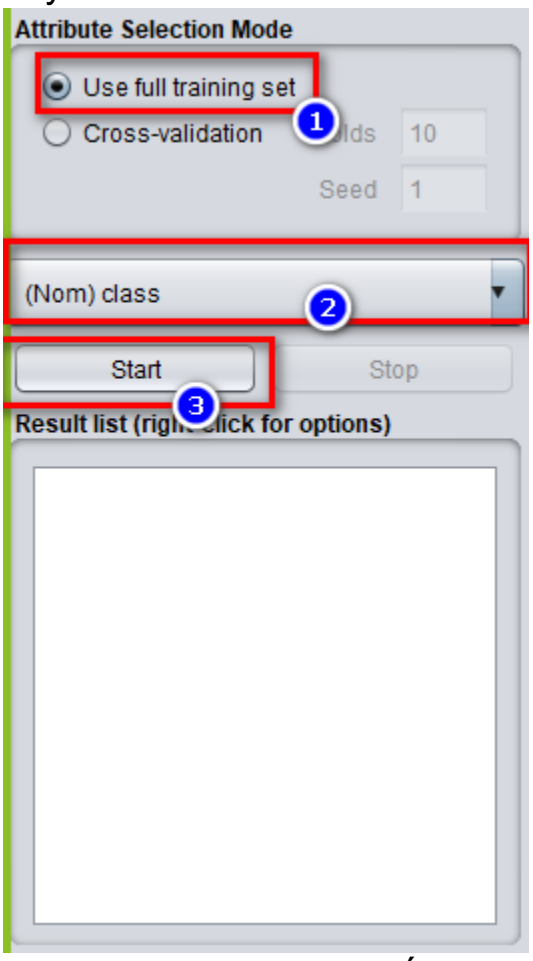
d) Cần sử dụng bộ lọc nào để chọn ra 5 thuộc tính có tương quan cao nhất với thuộc tính lớp? Mô tả các bước làm, kèm theo hình chụp từng bước và kết quả cuối cùng.

Vì đề bài yêu cầu dùng bộ lọc nào tìm kiếm 5 thuộc tính có độ tương quan cao nhất với thuộc tính lớp. Theo như tôi tìm hiểu ở trên, tôi sẽ chọn CorrelationAttributeEval vì nó đánh giá giá trị một thuộc tính bằng cách đo mối tương quan giữa thuộc tính đang xét với lớp.
Hình ảnh các bước thực hiện:
Bước 1: Ở tab Select Attribute. Ta chọn CorrelationAttributeEval ở Attribute Evalutor, Ranker ở Search Method.

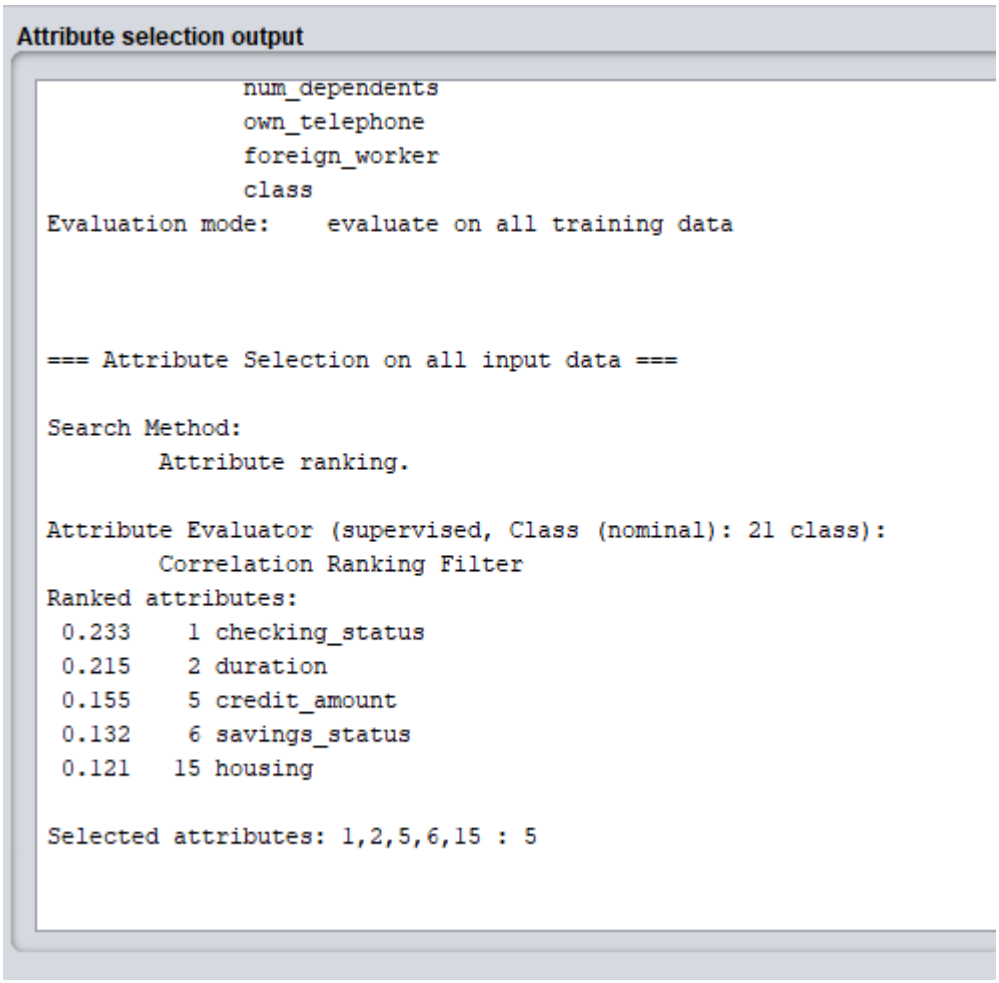
Bước 2: Ta nhấp chuột vào Ranker và chọn numToSelect = 5 và nhấn OK.



Bước 3: Ta chọn mode Use full training set. Chọn thuộc tính phân lớp, ở đây là class. Rồi bấm Start.



Vậy chúng ta có được kết quả cuối cùng, 5 thuộc tính tương quan với class nhất là: checking_status, duration, credit_amount, savings_status, housing.



3. Cài đặt tiền xử lý dữ liệu

Các thử viện sử dụng

- **Pandas:** dùng để đọc, ghi file và truy xuất dòng cột.
- **Sys, getopt**: để lấy tham số dòng lệnh.
- **Math**: để tính toán căn bậc 2 ở bài chuẩn hoá z-score, phần tìm độ lệch chuẩn. Mặc dù có thể không sử dụng hàm này bằng cách **(1/2). Tuy nhiên, kết quả giữa 2 cách này có sai số khá cao, nên em quyết định dùng thư viện.
- **Copy**: để deep copy list. Vì python theo kiểu tham chiếu, nên dùng hàm copy để dùng theo kiểu tham trị.

Các hàm tự defind và thường xuyên sử dụng

Hàm kiểm tra giá trị NaN

- Input: value
- Output: bool
- Ý tưởng: Một số mà không bằng chính nó thì nó là NaN. Hoặc NaN là một số không nằm trong khoảng âm vô cùng đến dương vô cùng. Ta có thể dùng cách này để kiểm tra xem giá trị truyền vào có phải là NaN không. Ở đây, để ngắn gọn ta dùng cách so sánh với chính nó.

```
def isNan(num):
    return num!= num
```

Hàm trả về kiểu dữ liệu của list

- Input: list
- Output: kiểu dữ liệu trả về.
- Ý tưởng: Ta duyệt qua hết tất cả các phần tử trong List. Nếu gặp NaN, ta bỏ qua cho đến khi gặp giá trị khác NaN thì ra trả về kiểu dữ liệu của nó. Nếu list rỗng, nó sẽ trả về None.

```
def dtype(l):
    for i in l:
        if(isNan(i)):
            continue
        else:
            return type(i)
    return None
```

1. **Liệt kê các cột bị thiếu dữ liệu:**

a) Chương trình:

- Input: file csv
- Output: chuỗi (tên các attribute bị thiếu dữ liệu)
- Tham số: file csv

b) Giải thích code:

- Trước hết chúng ta sẽ đi vào giải thích đề và tìm ra cách làm. Vậy liệt kê các cột bị thiếu dữ liệu thì chúng ta sẽ có 1 hàm để kiểm tra việc 1 ô trong data có bị NaN hay không. Sau đó ta tiến hành duyệt theo thứ tự từng cột, đến dòng nếu 1 cột nào đó có 1 ô trả về NaN thì chúng ta lập tức add tên của attribute đó vào 1 mảng, sau đó break

qua xét cột tiếp theo. Vậy thì sau khi xét hết mảng dữ liệu chúng ta sẽ được 1 list gồm tên các thuộc tính bị thiếu dữ liệu trong bảng.

```
def isNan(num):
    return num != num

def count_column_miss_value(df):
    head = df.columns
    count = []

    for i in range(1, 81):  #cot
        for j in range(df.shape[0]):  #dong
            if isNan(df.iat[i,j]):
                count.append(head[i])
                break
    return count

def main():
    argv = sys.argv[1]
    src = argv

    df = pd.read_csv(src)
    print(count_column_miss_value(df))
```

- Hàm isNan() sẽ trả về true nếu ô đó là ô trống
- Hàm count_column_miss_value(): là hàm để duyệt hế tất cả mảng, đầu tiên ta tạo 1 mảng head để chưa tên của các attribute, và 1 mảng rỗng count. Sau đó chúng ta tiến hành duyệt mảng. Và giá trị trả về của hàm này là mảng tên các attribute bị thiếu dữ liệu count.
- Hàm main(): src là biến chưa tên của file data input đầu vào, df dùng để chứa dữ liệu đầu vào của src.

c) Chạy demo:

- Phương thức gọi hàm rất đơn giản:

```
(tf) C:\Users\Admin\PycharmProjects\DataMining\Data-Mining-Lab01>python 1_column_mission_value.py house-prices.csv
```

- Và kết quả trả về:

```
['MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1', 'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating', 'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual', 'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual', 'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC', 'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType', 'SaleCondition', 'SalePrice']
```

- Và thế là chúng ta đã biết được những cột nào bị thiếu dữ liệu.

2. Đếm số dòng bị thiếu dữ liệu:

a) Chương trình:

- Input: file csv
- Output: int (số dòng bị thiếu dữ liệu).
- Tham số: file csv

b) Giải thích code:

- Ý tưởng: Tạo biến count = 0. Ta sẽ duyệt từng dòng, sau đó duyệt từng cột. Chỉ cần gặp 1 cột bị thiếu dữ liệu à dòng đó bị thiếu dữ liệu, count tăng lên một đơn vị. Khi kết thúc vòng lặp từng dòng, ta sẽ có biến count chứa số dòng bị thiếu dữ liệu.

```python
def count_row_miss_value(df):
    count = 0
    for row in range(df.shape[0]):
        for col in range(df.shape[1]):
            if isNan(df.iat[row,col]):
                count +=1
                break
    return count
```

c) Chạy demo

- Kết quả chạy demo:

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 2_count_mission_value.py house-prices.csv
1000
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

3. Điềm giá trị bị thiếu bằng phương pháp mean, median (cho thuộc tính numeric) và mode (cho thuộc tính nominal).

a) Chương trình:

- Input: file csv
- Output: file csv
- Tham số: file csv nguồn, file csv output, tên các cột bạn muốn điền khuyết, các phương thức tính toán (mean, median, mode),

b) Giải thích code:

- Trước hết chúng ta phải hiểu sơ lượt về mean, median, mode là gì:

+ Mean: là lấy giá trị trung bình của bộ dữ liệu (sử dụng cho bộ dữ liệu số).
+ Median: là lấy giá trị chính giữa của dãy (sử dụng cho bộ dữ liệu số).
+ Mode: là tìm giá trị xuất hiện nhiều nhất trong mảng (sử dụng cho bộ sữ liệu categorical).

- Nếu đã hiểu được như vậy thì việc viết hàm cho mỗi phương thức đã đơn giản hơn. Chúng ta chỉ cần xét các cột đầu vào và bắt đầu tính các giá trị trong những ô không NaN.

```
def is_not_Nan(num):  #tra ve true neu no nan
    return num == num

def isNan(num):
    return num != num

def dtype(l):
    for i in l:
        if(isNan(i)):
            continue
        else:
            return type(i)

def get_mean(df, attribute):
    data = df[attribute].values.tolist()
    count = 0
    sum = 0
    for i in data:
        if is_not_Nan(i):
            sum += i
            count += 1
    return sum/count
```

- Hàm is_not_Nan(): sẽ trả về true nếu tham số đầu vào không phải là Nan.
- Dtype(l): trả về kiểu dữ liệu của cột l đưa vào.
- Hàm get_mean(): sẽ trả về giá trị trung bình cộng của cột attribute.

```python
def get_median(df, attribute):
    data = df[attribute].to_numpy().tolist()
    Array = []
    for i in data:
        if is_not_Nan(i):
            Array.append(i)
    Array.sort()
    if len(Array)%2 == 1:
        return Array[len(Array)//2]
    else:
        return (Array[len(Array)//2] + Array[len(Array)//2 + 1])/2

def get_mode(df, attribute):
    data = df[attribute].to_numpy().tolist()
    dmax = -99

    for i in range(len(data)):
        if data.count(data[i]) > dmax and is_not_Nan(data[i]):
            dmax = data.count(data[i])
            x = data[i]
    return x
```

- Hàm get_madian(): trả về giá trị giữa mảng sau khi sắp xếp mảng đầu vào. Nếu mảng có tổng phần tử là lẻ thì giá trị được trả về giá trị chính giữa mảng, ngược lại sẽ trả về vị trí giữa mảng + 1.
- Hàm get_mode(): sẽ đếm các giá trị của attribute và bỏ qua các giá trị bị rỗng. Và hàm sẽ trả về giá trị có số lần xuất hiện nhiều nhất dmax.

```
def main():
    argv = sys.argv[1:]
    src = argv[0]  #doc ten file csv
    des = argv[1]  #doc ten file output
    expression = argv[2]
    type = argv[3]

    df = pd.read_csv(src)



    columns = expression.split(' ')
    if any(x not in df.columns for x in columns):
        print('Columns is not exist. Please check again!')
        return
    print(columns)

    if type == 'mean':
        for col in columns:
            data = df[col].to_numpy().tolist()

            if (dtype(data) is str):
                print('Your column is not numeric. Please check again!: ', col)
                break

            res = get_mean(df, col)   #tim so trung binh cong
            for i in range(len(data)):
                if isNan(data[i]):    #gan so res vao vi tri rong
                    data[i] = res
```

- Hàm main(): tương tự các câu trên, và chúng ta có 1 mảng các attribute người dùng nhập vào và muốn điền khuyết (columns). Trước tiên chúng ta sẽ check tên các attribute nhập vào có thực sự có trong df hay không, nếu không có thì return để người dùng nhập lại. Nếu kiểu mà người dùng muốn sử dụng để điền khuyết là ‘mean’ thì chúng ta phải kiểm tra những kiểu dữ liệu của các attribute có đúng là kiểu numeric hay không, nếu không thì thoát khoải chương trình. Còn nếu đúng thì tiếp tục gọi hàm get_mean(). Làm tương tự với 2 trường hợp còn lại.

c) Chạy demo:

- Phương thức gọi hàm:

```
Data-Mining-Lab01>python 3_mean_median_mode.py house-prices.csv output.csv LotFrontage,MSSubClass median
```

- Ở đây tôi muốn sử dụng phương thức median để điền khuyết cho 2 cột Lotfrontage và MSSubClass.
- Và đây là kết quả trả về trong file ouput.csv:

B	C	D
MSSubCla	MSZoning	LotFronta
20	RL	83
90	RL	70
50	RM	50
30	RL	52
20	RL	68
90	RL	65
20	RL	80
120	RM	32
60	RL	71
30	RM	52
20	RL	70
20	RL	71
20	RL	60
20	RL	70
20	RL	68
20	RL	36
70	RM	34
160	FV	35
50	RM	51
120	RM	44
60	RL	108
20	RL	71
60	RL	80
120	RM	37
30	RL	56

- Và 1 số lần chạy của các method khác:

BV	BW
Fence	MiscFeatu
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
GdWo	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
GdWo	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed
MnPrv	Shed

- Trong phương thức mode có 1 cột PoolQC là không có dữ liệu cho trước nên khi gọi hàm mode cho cột này nó sẽ báo lỗi không có giá trị trả về.
- Tôi sẽ thử nhập sai 1 cột cho 1 phương thức khác:

```
tf) C:\Users\Admin\PycharmProjects\DataMining\Data-Mining-Lab01>python 3_mean_median_mode.py house-prices.csv output.csv Fence,MiscFeature,LotFrontage mean
'Fence', 'MiscFeature', 'LotFrontage']
our column is not numeric. Please check again!:  Fence
```

- Kiểu method là mean, nhưng 2 cột đầu lại là kiểu categorical thì nó sẽ báo lỗi.

4. <u>Xóa các dòng bị thiếu dữ liệu với ngưỡng tỉ lệ cho trước.</u>

a) Chương trình:

- Input: file csv
- Output: file csv

- Tham số: file csv nguồn (-s hoặc --source), file csv output (-d hoặc --destination), tỉ lệ thiếu tính theo tỉ lệ phần trăm (-r hoặc --rate).

b) Giải thích code:

- Input: dataframe, rate
- Output: dataframe
- Ý tưởng: Ta chuyển rate thành số lượng cột lưu vào num_col, con số này là khi dòng nào bị thiếu dữ liệu bằng hoặc hơn num_col thì ta sẽ xoá dòng đó.
- Index_delete là một List. Chứa các index cần xoá.
- Sau khi duyệt qua các dòng và kiểm tra thì ta có list index_delete cần xoá. Bắt đầu gọi hàm delete_row_by_list_index.
- Tiếp theo ta trả về DataFrame đã xoá các dòng theo yêu cầu bài toán.

```
def delete_row_missing_value_rate(df,rate):
    head = df.columns
    #Convert pandas to list by row
    l_df = df.values.tolist()

    num_col = round(rate / 100 * len(head))
    index_delete = []
    for row in range(len(l_df)):
        count = 0
        for col in range(len(head)):
            if isNan(l_df[row][col]):
                count+=1
                if count == num_col:
                    index_delete.append(row)
                    break

    l_df = delete_row_by_list_index(l_df,index_delete)
    #Convert list to dataframe
    df = pd.DataFrame(l_df, columns=head)
    return df
```

Xóa các dòng tại index

- Input: List 2D, List (list index cần xoá)
- Output: List 2D
- Ý tưởng: Ta sẽ duyệt qua từng dòng và xoá các dòng có index có trong index_delete. Vì khi xoá, index sẽ bị thay đổi, cho nên ta dùng biến count nhằm xoá chính xác tại các vị trí muốn xoá.

```
def delete_row_by_list_index(l,index):
    count = 0
    for i in index:
        l.pop(i-count)
        count+=1
    return l
```

c) Chạy demo:

- Với rate = 50%:

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 4_delete_row.py -s house-prices.csv -d output.csv -r 50
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

1001	862	190	RL

- Ta sẽ không có dòng nào bị xoá, vì không có dòng nào mất 50% giá trị thuộc tính cả.
- Với rate = 5%

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 4_delete_row.py -s house-prices.csv -d output.csv -r 5
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
11	1084	20	RL	80	8800	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm	Norm	1Fam	1Story	6	6	1964	
12	1436	20	RL	80	8400	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm	Norm	1Fam	1Story	6	9	1962	
13	298	60	FV	66	7399	Pave	Pave	IR1	Lvl	AllPub	Inside	Gtl	Somerst	Norm	Norm	1Fam	2Story	7	5	1997	
14	993	60	RL	80	9760	Pave		Reg	Lvl	AllPub	Inside	Mod	mes	Norm	Norm	1Fam	2Story	6	8	1964	
15	1329	50	RM	60	10440	Pave	Grvl	Reg	Lvl	AllPub	Corner	Gtl	OldTown	Norm	Norm	1Fam	1.5Fin	6	7	1920	
16	796	60	RL	70	8400	Pave		Reg	Lvl	AllPub	Inside	Gtl	SawyerW	Norm	Norm	1Fam	2Story	6	6	1980	
17	323	60	RL	86	10380	Pave		IR1	Lvl	AllPub	Inside	Gtl	SawyerW	Norm	Norm	1Fam	2Story	7	5	1986	
18	52	50	RM	52	6240	Pave		Reg	Lvl	AllPub	Inside	Gtl	BrkSide	Norm	Norm	1Fam	1.5Fin	6	6	1934	
19	1458	70	RL	66	9042	Pave		Reg	Lvl	AllPub	Inside	Gtl	Crawfor	Norm	Norm	1Fam	2Story	7	9	1941	
20	317	60	RL	94	13005	Pave		IR1	Lvl	AllPub	Corner	Gtl	NWAmes	Norm	Norm	1Fam	2Story	7	7	1980	
21	891	50	RL	60	8064	Pave		Reg	Lvl	AllPub	Corner	Gtl	mes	Artery	Norm	1Fam	1.5Fin	5	7	1949	
22	1274	80	RL	124	11512	Pave		IR1	Lvl	AllPub	Corner	Gtl	Edwards	Norm	Norm	1Fam	SLvl	6	7	1959	
23	540	20	RL		11423	Pave		Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	1Story	8	5	2001	
24	1457	20	RL	85	13175	Pave		Reg	Lvl	AllPub	Inside	Gtl	NWAmes	Norm	Norm	1Fam	1Story	6	6	1978	
25	1436	20	RL	80	8400	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm	Norm	1Fam	1Story	6	9	1962	
26	891	50	RL	60	8064	Pave		Reg	Lvl	AllPub	Corner	Gtl	mes	Artery	Norm	1Fam	1.5Fin	5	7	1949	
27	1077	50	RL	60	10800	Pave	Grvl	Reg	Lvl	AllPub	Inside	Gtl	OldTown	Norm	Norm	1Fam	1.5Fin	5	8	1936	
28	41	20	RL	84	8658	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm	Norm	1Fam	1Story	6	5	1965	
29	870	60	RL	80	9938	Pave		Reg	Lvl	AllPub	Inside	Gtl	SawyerW	Norm	Norm	1Fam	2Story	7	5	1993	
30	540	20	RL		11423	Pave		Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	1Story	8	5	2001	
31	993	60	RL	80	9760	Pave		Reg	Lvl	AllPub	Inside	Mod	mes	Norm	Norm	1Fam	2Story	6	8	1964	
32	1329	50	RM	60	10440	Pave	Grvl	Reg	Lvl	AllPub	Corner	Gtl	OldTown	Norm	Norm	1Fam	1.5Fin	6	7	1920	
33	993	60	RL	80	9760	Pave		Reg	Lvl	AllPub	Inside	Mod	mes	Norm	Norm	1Fam	2Story	6	8	1964	
34	41	20	RL	84	8658	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm	Norm	1Fam	1Story	6	5	1965	
35	1053	60	RL	100	9500	Pave		Reg	Lvl	AllPub	Corner	Gtl	mes	Artery	Norm	1Fam	2Story	6	6	1964	

output

- Ta sẽ chỉ còn 35 dòng thôi, vì có 965 dòng mất đi 5% giá trị thuộc tính.

5. Xóa các cột bị thiếu dữ liệu với ngưỡng tỉ lệ thiếu cho trước.

a) Chương trình:

- Input: file csv
- Output: file csv

- Tham số: file csv nguồn, file csv output, tỉ lệ thiếu tính theo tỉ lệ phần trăm.

b) Giải thích code:

- Ý tưởng: chúng ta sẽ có 1 hàm dùng để xóa các cột với các vị trí index cho trước.

```
def delete_col_index(df, index):
    #matran = zip(*df)
    matran = [[df[j][i] for j in range(len(df))] for i in range(len(df[0]))]
    count = 0
    for i in index:
        matran.pop(i - count)
        count += 1

    l = [[matran[j][i] for j in range(len(matran))] for i in range(len(matran[0]))]
    return l
```

- Sẽ có 1 hàm duyệt qua tất cả các phần tử của mảng, của cột và đếm xem số lượng miss value của cột này, sau đó so sánh với tỉ lệ thiếu người dùng đã nhập trước. Nếu số ô bị thiếu dữ liệu bằng với ngưỡng cho trước thì ta sẽ thêm chỉ số chỉ vị trí của cột này vào 1 chuỗi index, và xóa attribute này ra khỏi chuỗi chứa các attribute. Sau đó gọi hàm ở trên với chỉ số index đã có được.

```python
def delete_col_missing_value_rate(df, rate):
    head = df.columns
    head = head.tolist()
    list = df.to_numpy().tolist()

    num = round(int(rate)/100 * 1000)

    index = []
    for i in range(len(head)):  #cot
        count = 0
        for j in range(len(list)):   #dong
            if isNan(list[j][i]):
                count += 1
                if count == num:
                    index.append(i)
                    del head[i]
                    break
    list = delete_col_index(list, index)
    l = pd.DataFrame(list, columns=head)
    return l
```

c) Chạy demo:

- Ví dụ với tỉ lệ miss_values là 30%:

```
(tf) C:\Users\Admin\PycharmProjects\DataMining\Data-Mining-Lab01>python 5_delete_col.py house-prices.csv output.csv 30
(1000, 75)
```

- Có thể thấy rằng đã bị xóa 6 cột (ban đầu là 1000x81).

6. Xóa các mẫu bị trùng lặp.

a) Chương trình:

- Input: file csv
- Output: file csv
- Tham số: file csv nguồn, file csv output

b) Giải thích code:

- Ý tưởng: khá đơn giản là duyệt qua toàn bộ dữ liệu, ở mỗi dòng chúng ta sẽ tạo ra 1 mảng A[] dùng để chứa toàn bộ dữ liệu của 1 dòng đó.
- Sau đó chúng ta sẽ có 1 vòng lặp bắt đầu từ dòng dòng hiện thời (A). chúng ta cũng sẽ tạo 1 mảng B[] để chứa dữ liệu, sau đó nếu A == B

thì chúng ta sẽ xóa ngay dòng đó (B) trong bộ dự liệu, và giảm số lượng dòng đi 1 đơn vị (row). Nếu A != B thì tăng biến đếm trong vòng lặp sau này lên 1 rồi tiếp tục chương trình.

```
def delete_row_duplicate(df):
    head = df.columns
    l_df = df.to_numpy().tolist()
    row = len(l_df)
    col = len(l_df[0])
    i = 0
    while (i < row):
        A = []
        for j in range(col):
            A.append(l_df[i][j])
        # A.to_numpy().tolist()
        r = i + 1
        while (r < row):
            B = []
            for c in range(col):
                B.append(l_df[r][c])
            # B.to_numpy().tolist()
            if check_is_same(A, B):
                l_df.pop(r)
                row -= 1
            else:
                r += 1
        i += 1

    l_df = pd.DataFrame(l_df, columns=head)
    return l_df
```

```
def check_is_same(l1, l2):
    for i in range(len(l1)):
        if (l1[i] == l2[i] or (isNan(l1[i]) and isNan(l1[i]) == isNan(l2[i]))):
            continue
        else:
            return False
    return True
```

c) Chạy demo:

- Kết quả trả về:

```
(tf) C:\Users\Admin\PycharmProjects\DataMining\Data-Mining-Lab01>python 6_delete_row_duplicate.py house-prices.csv output.csv
(717, 81)
```

- Và đây là kết quả khi xóa trên excel:

706	254	80	RL	85	9350	Pave		Reg	Lvl	AllPub	Inside	Gtl	mes	Norm
707	314	20	RL	150	215245	Pave		IR3	Low	AllPub	Inside	Sev	Timber	Norm
708	174	20	RL	80	10197	Pave		IR1	Lvl	AllPub	Inside	Gtl	mes	Norm
709	213	60	FV	72	8640	Pave		Reg	Lvl	AllPub	Inside	Gtl	Somerst	Norm
710	458	20	RL		53227	Pave		IR1	Low	AllPub	CulDSac	Mod	ClearCr	Norm
711	62	75	RM	60	7200	Pave		Reg	Lvl	AllPub	Inside	Gtl	IDOTRR	Norm
712	826	20	RL	114	14803	Pave		Reg	Lvl	AllPub	Inside	Gtl	NridgHt	PosN
713	985	90	RL	75	10125	Pave		Reg	Lvl	AllPub	Inside	Gtl	Mitchel	Norm
714	582	20	RL	98	12704	Pave		Reg	Lvl	AllPub	Inside	Gtl	NridgHt	Norm
715	668	20	RL	65	8125	Pave		Reg	Lvl	AllPub	Inside	Gtl	SawyerW	Norm
716	1190	60	RL	60	7500	Pave		Reg	Lvl	AllPub	Inside	Gtl	Gilbert	Norm
717	192	60	RL		7472	Pave		IR1	Lvl	AllPub	CulDSac	Gtl	mes	Norm
718														
719														
720														
721														
722														
723														
724														
725														
726														
727														

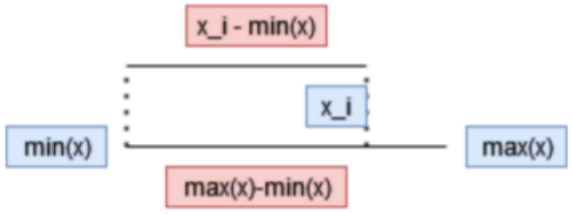
7. Chuẩn hóa một thuộc tính numeric bằng phương pháp min-max và Z-score.

a) Chương trình:

- Min-max
- Chuẩn hoá min-max là ta đang muốn co giãn giá trị dữ liệu về [0,1].
- Công thức:

$$X_i = \frac{xi - \min(x)}{\max(x) - \min(x)}$$

- Tức là xi càng lớn thì càng gần 1, càng nhỏ thì càng gần 0.



- Tuy nhiên, sẽ có trường hợp max(x) = min(x) và sẽ xảy ra lỗi chia cho 0. Tôi đã tìm hiểu nhiều nguồn khác nhau. Thường thì họ sẽ loại

bỏ cột đó. Một số người họ sẽ dùng cách 1/$len(x)$. Ở đây, tôi sẽ dùng cách loại bỏ cột đó, không tính cột đó.

b) Giải thích code:

- Input: list (cần chuẩn hóa)
- Output: list (đã chuẩn hóa)

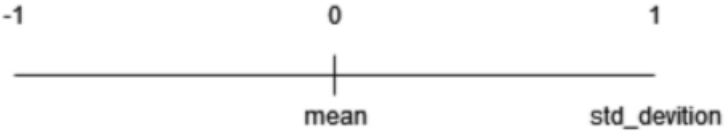
```
def min_max(l_col):
    #Convert NaN to zero
    tmp = copy.deepcopy(l_col)
    for i in range(len(l_col)):
        l_col[i] = 0 if (isNan(l_col[i])) else l_col[i]

    if max(l_col) == min(l_col):
        return tmp
    return [((x - min(l_col)) / ((max(l_col) - min(l_col)) if (max(l_col) -
min(l_col)) > 0 else len(l_col))) for x in l_col]
```

Z-code:

- Chính qui: Chính quy hóa dữ liệu giúp cho giá trị của mỗi đặc trưng có trung bình bằng 0 và phương sai bằng 1. Tức là chuẩn hoá nó khi giá trị trung bình có mốc là 0 và phương sai có mốc là 1.
- Công thức:

$$xi = \frac{xi - mean_x}{\sigma_x}$$

- Ta có hiểu, dấu tượng trưng cho $xi>mean$ | $xi<mean$. Chuẩn hoá z-score sẽ co giãn giá trị về [-1,1].



- Ở trường hợp này, nếu như $standard\ devition$=0 thì ta sẽ bị lỗi chia cho 0. $Standard\ devition$=0 khi giá trị của tất cả các thuộc tính đều bằng nhau. Cách giải quyết là ta sẽ không tính các cột như thế và bỏ qua.
- Input: list (cần chuẩn hóa)
- Output: list (đã chuẩn hóa)

```python
def z_score(l_col):
    #Convert NaN to zero
    tmp = copy.deepcopy(l_col)
    for i in range(len(l_col)):
        l_col[i] = 0 if (isNan(l_col[i])) else l_col[i]

    mean = sum(l_col) / len(l_col)
    std_devition = math.sqrt(sum([(abs(x-mean)**2) for x in l_col])/len(l_col))
    if(std_devition == 0):
        return tmp
    return [((x - mean)/std_devition) for x in l_col]
```

- Chương trình: file csv nguồn (-s, --source), file csv output (-d, -destination), chế độ min-max hay z-score (-m, --mode), cột cần chuẩn hoá (-c, --columns) – một hay nhiều cột đều được.
- Output: File csv đã chuẩn hoá.
- Ý tưởng: Ta sẽ đi qua từng cột. Chuyển cột thành list. Loại bỏ cột nào không phải kiểu dữ liệu số và bỏ cột Id. Sau đó, dựa vào mode người dùng nhập, ta cho tính toán tương ứng. Cuối cùng, gán giá trị đã được chuẩn hoá vào cột tương ứng. Và xuất ra file.

c) Chạy demo:

- Ví dụ: Ở ví dụ này, tôi chỉ lấy vài cột tượng trưng, ẩn gần như toàn bộ cột nào không phải numberic, chỉ chừa 1 cột categories để tham khảo thuật toán chạy có đúng không. Và sẽ hiển thị 2 cột đặc biệt là PoolArea với tất cả giá trị bằng 0 và cột PoolQC không chứa giá trị, 2 cột này là trường hợp khiến chuẩn hoá chia cho 0: min(x)=max(x)| σx=0.
- Chuẩn hoá min-max với tất cả các cột

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 7_numeric.py -s house-prices.csv -d output.csv -m min-max -c all
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	A	B	C	D	E	R	S	T	U	AA	AI	BT	BU	BX	BY	BZ	CC
1	Id	MSSubCla	MSZoning	LotFronta	LotArea	OverallQu	OverallCo	YearBuilt	YearRemc	MasVnrAr	BsmtFinS	PoolArea	PoolQC	MiscVal	MoSold	YrSold	SalePrice
2	1242	0	RL	0.542484	0.039164	0.666667	0.625	0.976923	0.95	0	0	0		0	0.454545	0.25	0.369599
3	1233	0.411765	RL	0.457516	0.039131	0.333333	0.5	0.630769	0.2	0	0	0		0	0.181818	0.25	0.115363
4	1401	0.176471	RM	0.326797	0.021158	0.555556	0.75	0.376923	0	0	0	0		0	0.545455	0.5	0.146941
5	1377	0.058824	RL	0.339869	0.022524	0.555556	0.5	0.384615	0	0	0.169912	0		0	0.272727	0.5	0.096624
6	208	0	RL	0	0.051533	0.333333	0.5	0.615385	0.166667	0	0.185398	0		0	0.272727	0.5	0.183378
7	1392	0.411765	RL	0.424837	0.03493	0.444444	0.5	0.669231	0.283333	0	0	0		0	0.272727	0.75	0.153882
8	980	0	RL	0.522876	0.034332	0.444444	0.625	0.638462	0.216667	0	0.288053	0		0	0.454545	0.75	0.179908
9	484	0.588235	RM	0.20915	0.014141	0.555556	0.5	0.907692	0.8	0.0725	0.396903	0		0	0.363636	0	0.223284
10	392	0.235294	RL	0.464052	0.050204	0.555556	0.5	0.930769	0.866667	0	0.30531	0		0	0.454545	0.75	0.311773
11	730	0.058824	RM	0.339869	0.022281	0.333333	0.5	0.346154	0	0	0.067257	0		0	0	0.75	0.117445

- Chuẩn hoá z-score với tất cả các cột

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 7_numeric.py -s house-prices.csv -d output.csv -m z-score -c all
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	A	B	D	E	R	S	T	U	AA	BT	BU	BV	BX	BY	BZ	CC
1	Id	MSSubCla	LotFronta	LotArea	OverallQu	OverallCo	YearBuilt	YearRemc	MasVnrAr	PoolArea	PoolQC	Fence	MiscVal	MoSold	YrSold	SalePrice
2	1242	-0.84868	0.788585	-0.03953	0.679642	0.416532	1.191954	1.095081	-0.5683	0			-0.15301	-0.05964	-0.66337	0.876183
3	1233	0.785639	0.389472	-0.04029	-1.40728	-0.4423	-0.25214	-1.02197	-0.5683	0			-0.15301	-1.19199	-0.66337	-0.95236
4	1401	-0.14826	-0.22455	-0.45877	-0.016	1.27536	-1.31114	-1.58652	-0.5683	0			-0.15301	0.317814	0.067214	-0.72524
5	1377	-0.6152	-0.16314	-0.42697	-0.016	-0.4423	-1.27905	-1.58652	-0.5683	0			-0.15301	-0.81454	0.067214	-1.08713
6	208	-0.84868	-1.75959	0.248468	-1.40728	-0.4423	-0.31632	-1.11606	-0.5683	0		GdWo	-0.15301	-0.81454	0.067214	-0.46318
7	1392	0.785639	0.235968	-0.1381	-0.71164	-0.4423	-0.09168	-0.78674	-0.5683	0			-0.15301	-0.81454	0.797796	-0.67532
8	980	-0.84868	0.696482	-0.15205	-0.71164	0.416532	-0.22005	-0.97493	-0.5683	0		MnPrv	-0.15301	-0.05964	0.797796	-0.48813
9	484	1.48606	-0.77716	-0.62216	-0.016	-0.4423	0.903136	0.67167	0.044941	0			-0.15301	-0.43709	-1.39395	-0.17616
10	392	0.085218	0.420173	0.217534	-0.016	-0.4423	0.999409	0.859853	-0.5683	0			-0.15301	-0.05964	0.797796	0.460279
11	730	-0.6152	-0.16314	-0.43263	-1.40728	-0.4423	-1.4395	-1.58652	-0.5683	0			-0.15301	-1.9469	0.797796	-0.93738

- Chuẩn hoá min-max ở các cột được chỉ định (MSSubClass, PoolArea, PoolQC)

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 7_numeric.py -s house-prices.csv -d output.csv -m min-max -c MSSubClass,PoolArea,PoolQC
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	A	B	D	BT	BU	BV
1	Id	MSSubCla	LotFronta	PoolArea	PoolQC	Fence
2	1242	0	83	0		
3	1233	0.411765	70	0		
4	1401	0.176471	50	0		
5	1377	0.058824	52	0		
6	208	0		0		GdWo
7	1392	0.411765	65	0		
8	980	0	80	0		MnPrv
9	484	0.588235	32	0		
10	392	0.235294	71	0		
11	730	0.058824	52	0		
12	255	0	70	0		
13	1094	0	71	0		MnPrv
14	1021	0	60	0		

- Chuẩn hoá z-score ở các cột được chỉ định (MSSubClass, PoolArea, PoolQC)

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 7_numeric.py -s house-prices.csv -d output.csv -m z-score -c MSSubClass,PoolArea,PoolQC
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	A	B	D	BT	BU	BV
1	Id	MSSubCla	LotFronta	PoolArea	PoolQC	Fence
2	1242	-0.84868	83	0		
3	1233	0.785639	70	0		
4	1401	-0.14826	50	0		
5	1377	-0.6152	52	0		
6	208	-0.84868		0		GdWo
7	1392	0.785639	65	0		
8	980	-0.84868	80	0		MnPrv
9	484	1.48606	32	0		
10	392	0.085218	71	0		
11	730	-0.6152	52	0		
12	255	-0.84868	70	0		

- Chuẩn hoá nhập sai tên cột.

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 7_numeric.py -s house-prices.csv -d output.csv -m z-score -c MSSubClass,PoolArea,PolC
Columns is not exist. Please check again!
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

8. Tính giá trị biểu thức thuộc tính:

a) Chương trình:

- Input: file csv nguồn, file csv output, biểu thức.
- Output: File csv output kèm cột đã tính toán biểu thức.

b) Giải thích code:

- Intput: DataFrame, chuỗi biểu thức
- Output: DataFrame
- Ý tưởng: Ta sử dụng hàm eval() để tính toán. Ở bài toán này ta sử dụng 2 tham số của hàm eval(): biểu thức ở kiểu string và giá trị biến ở kiểu dictionary.

- Tạo dictionary rỗng.
- Đầu tiên, ta duyệt qua từng dòng giá trị. Thêm vào dict với key là tên cột và value là giá trị ở dòng đó.
- Sau đó, ta cho vào hàm eval để tính toán và nhận được kết quả. Lưu lại ở list ret.
- Sau khi duyệt qua tất cả các dòng. Ta thêm cột Result với giá trị của list ret ở trên.

```python
def cal(df,expression):
    head = df.columns
    ret = []
    for i in range(df.shape[0]):
        dictx={}
        for j in head:
            s = df[j].values.tolist()
            dictx[j]=s[i]
        ret.append(eval(expression,dictx))
    df[expression] = ret
    return df
```

c) Chạy demo:

- Tính toán bình thường.
- Biểu thức: MSSubClass*OverallQual

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 8_function.py house-prices.csv output.csv MSSubClass*OverallQual
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	B	R	CD
1	MSSubClass	OverallQual	MSSubClass*OverallQual
2	20	7	140
3	90	4	360
4	50	6	300
5	30	6	180
6	20	4	80
7	90	5	450
8	20	5	100
9	120	6	720
10	60	6	360
11	30	4	120

- Tính toán có toán tử ưu tiên và có dữ liệu bị thiếu
- Biểu thức: MSSubClass-(OverallQual+LotFrontage)/OverallCond

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 8_function.py house-prices.csv output.csv MSSubClass-\(OverallQual+LotFrontage\)/Overal
lCond
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	B	D	R	S	CD
1	MSSubClass	LotFrontage	OverallQual	OverallCond	MSSubClass-(OverallQual+LotFrontage)/OverallCond
2	20	83	7	6	5
3	90	70	4	5	75.2
4	50	50	6	7	42
5	30	52	6	5	18.4
6	20		4	5	
7	90	65	5	5	76
8	20	80	5	6	5.833333333
9	120	32	6	5	112.4
10	60	71	6	5	44.6
11	30	52	4	5	18.8
12	20	70	5	6	7.5
13	20	71	5	8	10.5
14	20	60	4	5	7.2
15	20	70	4	5	5.2
16	20		8	6	
17	20	36	5	6	13.16666667

- Tính toán khi có cột không phải là numberic

```
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$ python3 8_function.py house-prices.csv output.csv MSSubClass-MSZoning
Something went wrong. Please notify me :)
(min_ds-env) itlvd@lvd:~/MyCode/Data_mining/Data-Mining-Lab01$
```

	B	C
1	MSSubCla	MSZoning
2	20	RL
3	90	RL
4	50	RM
5	30	RL
6	20	RL