

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



BÁO CÁO

**LAB 04 – HỌC MÁY
TÌM HIỂU VỀ WEKA**

Ngô Minh Phát

Giáo viên hướng dẫn: Thầy Lê Ngọc Thành
Thành phố Hồ Chí Minh – 12/2022

A. MỤC LỤC

A. MỤC LỤC	1
B. Sơ lược về Weka	4
C. Tìm hiểu về Weka:	4
I. Bắt đầu với Weka:	4
1. Giới thiệu:	4
2. Chức năng Explorer	5
3. Dataset.....	6
4. Xây dựng classifier	6
5. Sử dụng filter	7
6. Visualize	8
II. Đánh giá kết quả	8
1. Training và testing	8
2. Repeated Holdout:	10
3. Baseline accuracy	11
4. Cross Validation:	12
5. So sánh Repeated Holdout và Stratified Cross-validation	13
III. Các mô hình phân lớp đơn giản	14
1. OneR - One attribute does all the work	14
2. Overfitting	16
3. Naïve Bayes - Attributes contribute equally and independently	16
4. Decision tree – use a few attributes	17
D. Lab04	18
I. Phân tích yêu cầu đề bài:	18
1. Tìm hiểu các độ đo (metric).....	18
2. Yêu cầu về thuật toán.....	18
3. Yêu cầu về dữ liệu:	18
4. Kiểm tra kết quả dự đoán các mẫu test.....	19
5. Tìm hiểu hiện tượng quá khớp (overfitting) trên cây Id3.....	19
6. So sánh 2 thuật toán Id3 và Naïve Bayes	19

II. Tìm hiểu về các độ đo	19
1. Confusion matrix	19
2. Accuracy	20
3. Precision.....	20
4. Recall	20
5. F1 Score	21
III. Dataset:	22
1. Thông tin thuộc tính.....	22
2. Phân tích dữ liệu thô	23
IV. Tạo file ARFF cho tập dữ liệu Zoo	26
1. Tạo phần mô tả	26
2. Tạo phần cấu trúc dữ liệu	27
3. Tạo phần dữ liệu	27
4. Tạo file arff cho training set của bài toán.....	29
5. Tạo file Arff cho file cần dự đoán	29
V. Tiền xử lý dữ liệu với Weka.....	30
VI. ZeroR - Zoo Data Set - Baseline accuracy.....	30
VII. Cách áp dụng Weka vào bài toán:.....	31
1. Sử dụng FilteredClassify	31
2. Cách mô hình Weka để dự đoán.....	32
VIII. Id3 – Zoo Data Set - Overfitting	34
1. Classify	34
2. Trực quan quá trình tiền xử lý bằng tab preprocess	34
3. Áp dụng Weka classifier Id3	35
IX. Id3 – Zoo Data Set – Sửa lỗi Overfitting	38
1. Classify	38
2. Trực quan quá trình tiền xử lý bằng tab preprocess	38
3. Áp dụng Weka classifier Id3	39
X. Naïve Bayes – Zoo Data Set.....	41
1. Classify	42
2. Trực quan quá trình tiền xử lý bằng tab preprocess	42

3.	Áp dụng Weka classifier NaiveBayes	42
XI.	So sánh Id3 và Naïve Bayes	43
1.	Một số qui ước:	43
2.	So sánh chung	44
3.	Dựa trên Zoo Data Set	44
4.	Dựa trên Statlog (Heart) Data Set.....	44
5.	Kết luận:	46
E. References		47

B. Sơ lược về Weka



Weka còn có tên đầy đủ là Waikato Environment for Knowledge Analysis. Đây chính là bộ phần mềm mã nguồn mở được sử dụng miễn phí để khai thác dữ liệu thuộc các dự án nghiên cứu của đại học Waikato, New Zealand. Weka hỗ trợ để tiên xử lý dữ liệu và chạy các thuật toán học máy kinh điển trên dữ liệu có cấu trúc.

C. Tìm hiểu về Weka:

Ở đây, chúng ta sẽ tìm hiểu về Weka bằng cách học theo khóa học *Data Mining with Weka MOOC - Material* (UCI Machine Learning, 2022)

Ở mức tìm hiểu cơ bản, ta chỉ tìm hiểu về tính năng Explorer.

I. Bắt đầu với Weka:

1. Giới thiệu:

Phiên bản đang học và sử dụng là Weka 3.8.6 – [Download link](#).

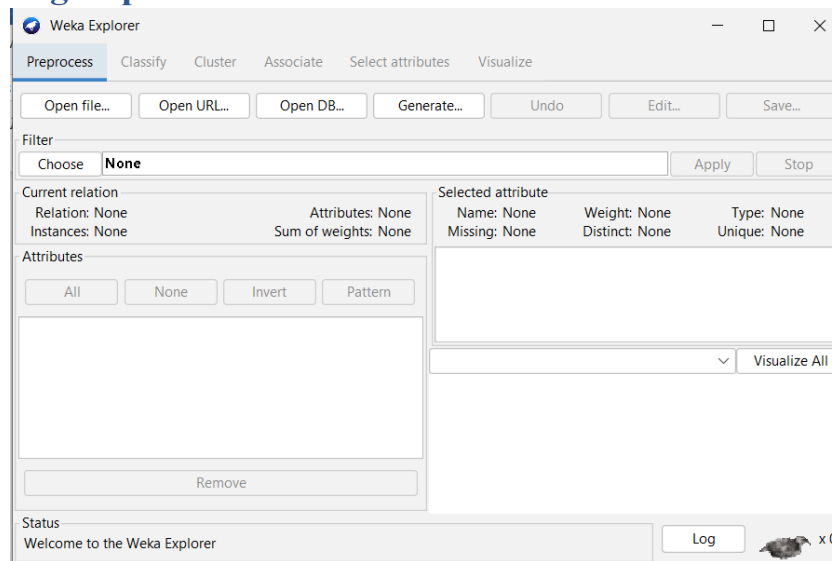


Ảnh: Giao diện khi khởi động Weka 3.8.6

Weka hiện hỗ trợ 5 tính năng:

- Explorer
- Experimenter
- KnowledgeFlow
- Workbench
- Simple CLI

2. Chức năng Explorer



Ảnh: Giao diện chức năng Explorer

Explorer hỗ trợ 6 tính năng trên 6 tab:

- Preprocess: tiền xử lý dữ liệu
- Classify: áp dụng mô hình phân lớp
- Cluster: áp dụng mô hình gom nhóm
- Associate
- Select attributes
- Visualize: trực quan hoá

Mở datasets:

Ta có datasets [weather.nominal.arff](#) mặc định của weka như sau:

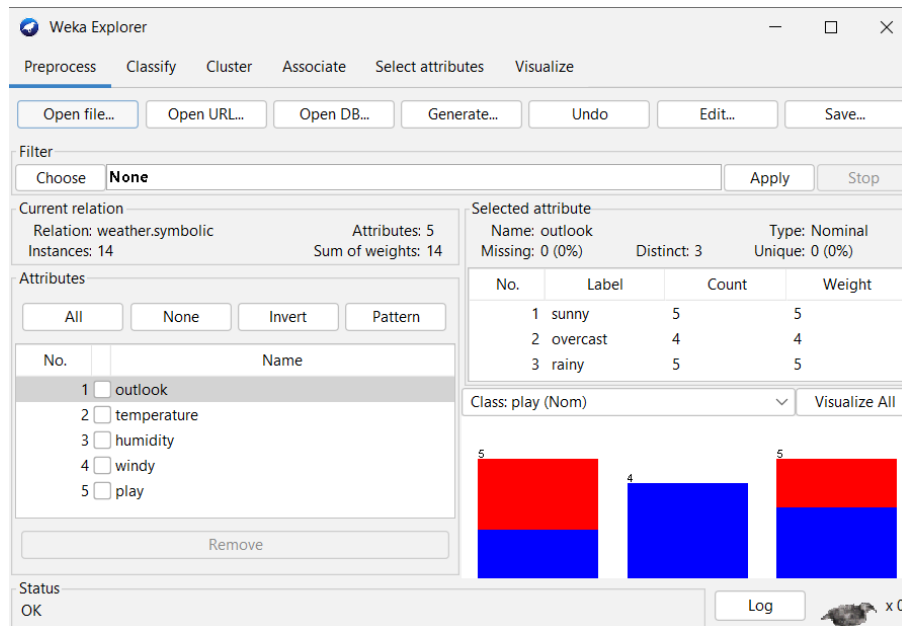
No.	1: outlook Nominal	2: temperature Nominal	3: humidity Nominal	4: windy Nominal	5: play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Ảnh: dataset weather.nominal.arff

Dữ liệu có chứa 14 instance¹. Mỗi instance có các attributes² bao gồm *outlook*, *temperature*, *humidity* and *windy*. *Play* là class³ cần dự đoán của bài toán.

Cách mở datasets trong Explorer:

Chọn “Open file”, chọn **weather.norminal.arff**



Ảnh: Sau khi mở file weather.norminal.arff

3. Dataset⁴

Có các loại dataset như sau: nominal và numeric.

Dữ liệu sử dụng trong Weka thường ở định dạng file ARFF, CSV,

4. Xây dựng classifier⁵

Tab classify, hỗ trợ nhiều loại thuật toán (Naïve Bayes, Id3, ...) để đánh giá và dự đoán class dựa trên dữ liệu hiện có.

VD:

Sử dụng ID3 để tạo mô hình phân lớp trên tập dữ liệu [weather.norminal.arff](#)

- Mở file weather.norminal.arff

¹ Instance: đối tượng (dòng) trên dataset

² Attributes: Thuộc tính

³ Class: lớp – giá trị cần dự đoán

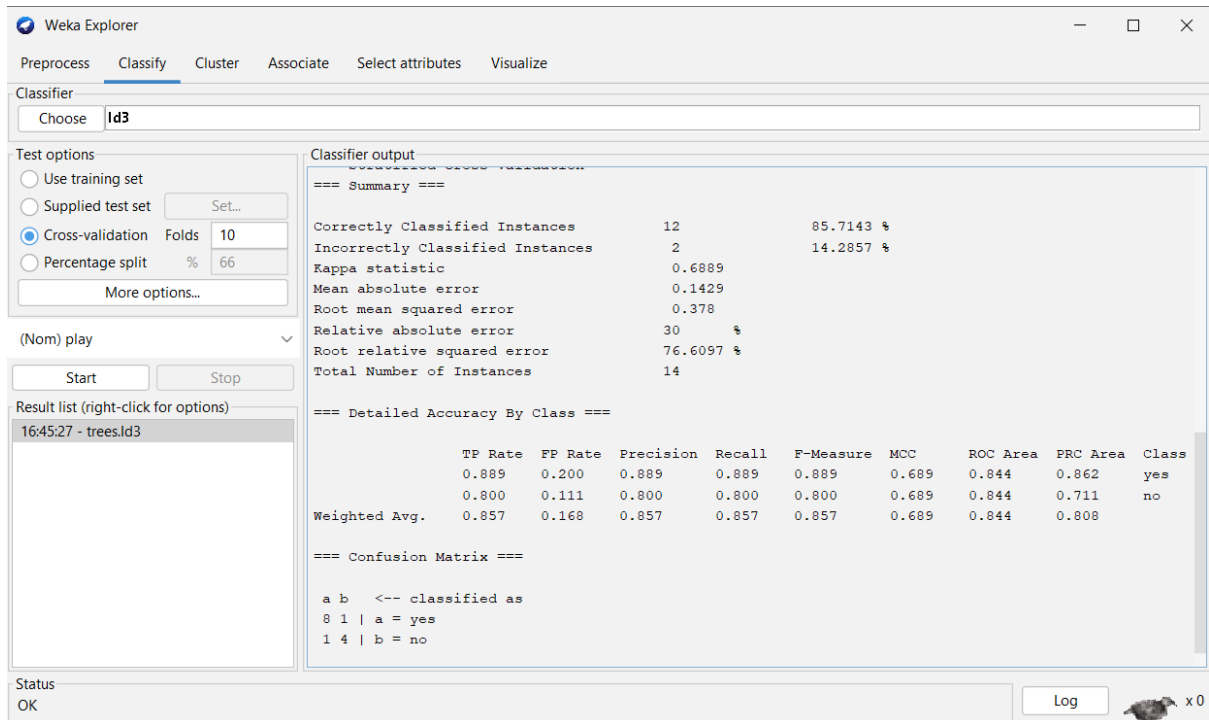
⁴ Dataset: tập dữ liệu

⁵ Classifier: Thuật toán/Công cụ hỗ trợ tạo mô hình phân lớp mà weka cung cấp.

Nhập môn trí tuệ nhân tạo

- Chọn tab classify
- Ở phần classifier, chọn **tree > Id3**
- Tùy chỉnh các option cho model ở Test options
- Start và đọc kết quả ở phần output

Kết quả



Weka Explorer

Preprocess **Classify** Cluster Associate Select attributes Visualize

Classifier: Choose **Id3**

Test options

- ☐ Use training set
- ☐ Supplied test set (Set...)
- ☒ Cross-validation (Folds: 10)
- ☐ Percentage split (%: 66)

More options...

(Nom) play

Start Stop

Result list (right-click for options)

16:45:27 - trees.Id3

Classifier output

```
==== Summary ====
```

Correctly Classified Instances	12	85.7143 %
Incorrectly Classified Instances	2	14.2857 %
Kappa statistic	0.6889	
Mean absolute error	0.1429	
Root mean squared error	0.378	
Relative absolute error	30 %	
Root relative squared error	76.6097 %	
Total Number of Instances	14	

```
==== Detailed Accuracy By Class ====
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.889	0.200	0.889	0.889	0.889	0.689	0.844	0.862	yes
	0.800	0.111	0.800	0.800	0.800	0.689	0.844	0.711	no
Weighted Avg.	0.857	0.168	0.857	0.857	0.857	0.689	0.844	0.808	

```
==== Confusion Matrix ====
```

```
a b  <-- classified as
8 1 | a = yes
1 4 | b = no
```

Status: OK

Log x 0

Ảnh: Giao diện của tab classify

5. Sử dụng filter

Filter hỗ trợ tiền xử lý dữ liệu trước khi áp dụng một classifier.

Cách sử dụng filter:

- Mở file weather.nominal.arff
- Chọn tab preprocess
- Chọn Filter và loại filter dữ liệu.
- Áp dụng classifier trên dữ liệu đã xử lý
- Có thể tùy ý sử dụng dữ liệu mới sau khi đã filter

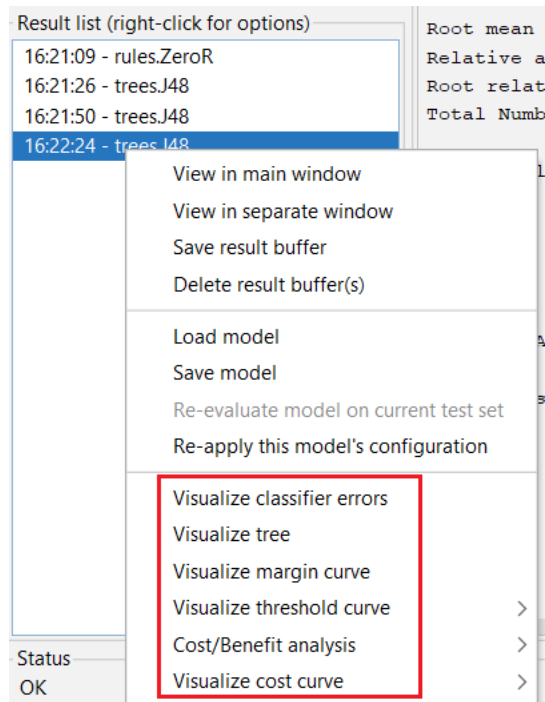
6. Visualize⁶

Trực quan hóa dữ liệu - Sử dụng tab Visualize

Sau khi mở dataset ở tab Preprocess, ta có thể chọn tab Visualize để quan sát sự trực quan của dữ liệu

Trực quan hóa mô hình - Sử dụng tab Classify:

Sau khi chọn một thuật toán và áp dụng vào để xử lý thành công dữ liệu. Ta có thể click chuột phải vào một kết quả ở Result list và chọn tùy chọn mong muốn.



Ảnh: Các tùy chọn để trực quan hóa đối với mô hình

II. Đánh giá kết quả

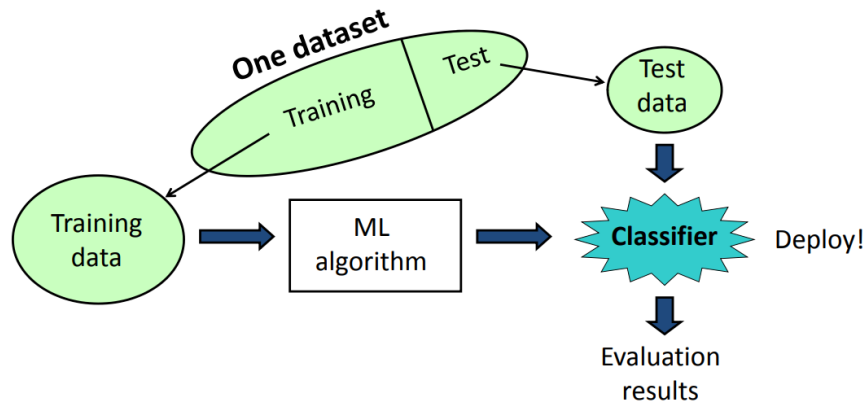
1. Training⁷ và testing⁸

Qui trình tạo nên một mô hình phân lớp:

⁶ Visualize: Trực quan hóa

⁷ Training: huấn luyện (mô hình)

⁸ Testing: kiểm tra (mô hình)

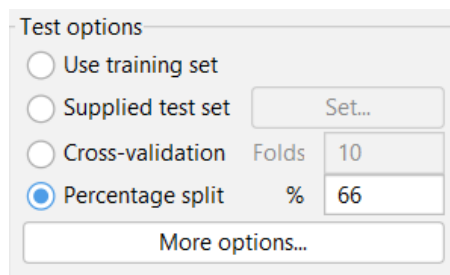


Giả định cơ bản: Training set⁹ và testing set¹⁰ được tạo bởi các mẫu độc lập từ một quần thể vô hạn.

Ảnh: Qui trình tạo nên một mô hình phân lớp

Kết quả của mô hình phân lớp chỉ đáng tin cậy khi và chỉ khi dữ liệu dùng để huấn luyện và kiểm tra là hoàn toàn khác nhau. Do đó, nếu chúng ta chỉ có một bộ dữ liệu thì chúng ta nên chia ngẫu nhiên nó thành 2 phần với tỉ lệ là 2/3 cho việc huấn luyện và 1/3 cho việc kiểm tra.

Weka hỗ trợ các tùy chọn khi huấn luyện đối với bộ dữ liệu như sử dụng test set¹¹, cross-validation¹², và percentage split¹³.



Ảnh: Các tùy chọn với dữ liệu ở tab Classify

⁹ Training set: Tập dữ liệu dùng để huấn luyện

¹⁰ Testing set: tập dữ liệu dùng để kiểm tra

¹¹ Test set: tùy chọn “using test set” khi chúng ta có 2 bộ dữ liệu training và testing độc lập

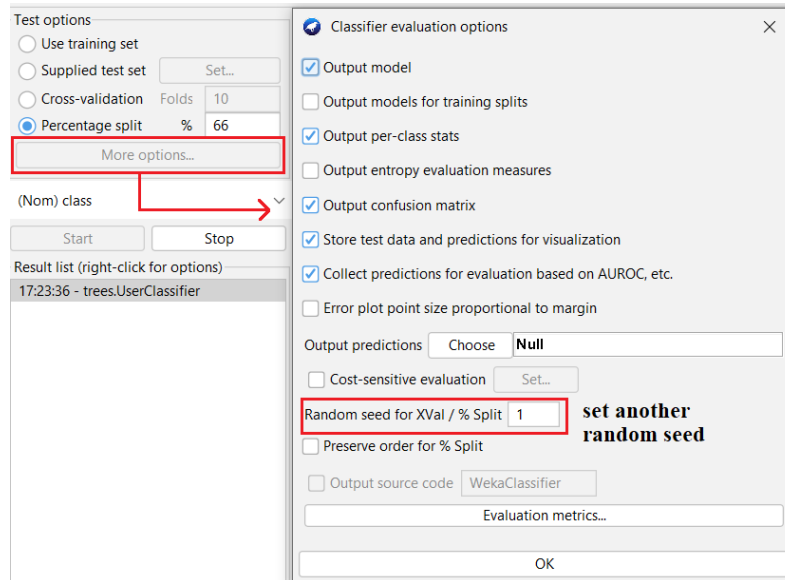
¹² Cross-validation: tùy chọn “cross-validation” – Xem chi tiết ở [C.II.2.4](#)

¹³ Percentage split: Chia ngẫu nhiên dữ liệu thành 2 phần với tỉ lệ training set là input được nhập vào. Lưu ý, weka luôn mặc định chọn một seed ngẫu nhiên cho mỗi mô hình, để đảm bảo việc chia ngẫu nhiên cho mỗi lần chạy đem lại kết quả giống nhau

2. Repeated Holdout¹⁴:

Repeated Holdout là việc lặp lại training và testing trên cùng một dataset với cách chia ngẫu nhiên khác nhau.

Do weka luôn mặc định một seed cho việc chia ngẫu nhiên dataset (tất cả các lần đều ra cùng một kết quả). Chúng ta có thể tùy chỉnh seed này thủ công ở tùy chọn more options.



Ảnh: Tùy chọn nâng cao đối với test option

VD:

Ta sẽ thực hiện việc tạo mô hình phân lớp với thuật toán J48 trên bộ dữ liệu của weka cung cấp sẵn segment-challenge.arff gồm 1500 instance với percentage split là 90% với seed từ 1 đến 10.

Độ chính xác của mô hình sẽ thể hiện trên output như sau:

Classifier output		
=== Summary ===		
Correctly Classified Instances	145	96.6667 %
Incorrectly Classified Instances	5	3.3333 %
Kappa statistic	0.961	
Mean absolute error	0.0117	
Root mean squared error	0.0971	
Relative absolute error	4.7637 %	
Root relative squared error	27.7573 %	
Total Number of Instances	150	

Kết quả nhận được sau 10 lần chạy như sau:

¹⁴ Repeated Holdout: Lặp lại training và testing

		0.967
		0.940
Sample mean	$\bar{x} = \frac{\sum x_i}{n}$	0.940
		0.967
Variance	$\sigma^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$	0.953
		0.967
Standard deviation	σ	0.920
		0.947
		0.933
		0.947

Ảnh: Kết cho 10 lần chạy với seed từ 1 đến 10

Dựa vào công thức ta tính được $\bar{x} = 0.949$ và $\sigma = 0.018$

⇒ Độ chính xác của mô hình xấp xỉ: $95\% \pm 1.8\%$

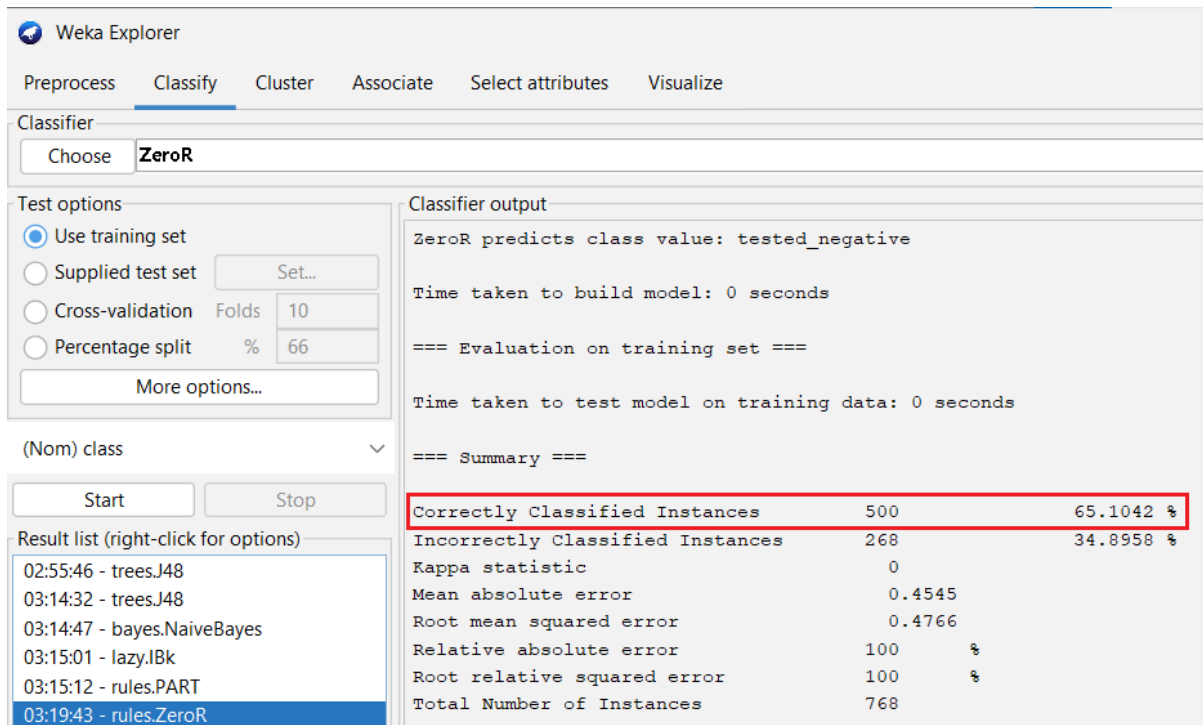
3. Baseline accuracy

Baseline accuracy thể hiện độ chính xác tối thiểu mà mô hình đạt được nếu dự đoán của ta luôn là lớp có tỉ lệ cao nhất trong datasets

Chúng ta sẽ xem xét qua 2 ví dụ sau

VD1:

- Sử dụng dataset [diabetes.arff](https://archive.ics.uci.edu/ml/diabetes.html)
- Thực hiện tạo mô hình phân lớp trên 4 thuật toán với percentage split là 66%:
 - J48: 76%
 - Naïve Bayes: 77%
 - Ibk: 73%
 - PART: 74%
- Tập dữ liệu bao gồm 768 instance, trong đó có 500/768 xấp xỉ 65% thuộc class “negative” còn lại là 268/768 thuộc class “positive”.
- Do đó, nếu ta luôn dự đoán một instance là “negative” thì xác suất dự đoán đúng luôn luôn là 65%. **Đây chính là baseline accuracy.**
- Weka hỗ trợ một baseline classifier – ZeroR làm điều này với option “using training set”. ZeroR sẽ luôn đưa ra dự đoán là class chiếm tỉ lệ cao nhất mà không quan tâm đến giá trị của thuộc tính của các instance.



Ảnh: độ chính xác của mô hình với thuật toán ZeroR bằng đúng 500/768

VD2:

- Sử dụng dataset [supermarket.arff](#)
- Thực hiện tạo mô hình phân lớp trên 4 thuật toán với percentage split là 80%
 - o ZeroR: 63.7% (use training set)
 - o J48: 63.35%
 - o Naïve Bayes: 63.35%
 - o IbK: 37.51% !!!
 - o PART: 63.35%
- Tập dữ liệu bao gồm 4627 instance, trong đó có 2948/4627 xấp xỉ 63.7% thuộc class “low” còn lại là 1679/4627 thuộc class “high”.
- Ở trường hợp với bộ dữ liệu này, ta thấy baseline accuracy là cao nhất.

⇒ Do đó, trước khi sử dụng Weka áp dụng bất kỳ một thuật toán nào vào mô hình phân lớp, ta cần sử dụng baseline classifier để đánh giá mô hình.

4. Cross Validation:

Chúng ta có một thuật toán tên là K-fold validation.

- Chia dữ liệu thành K phần (fold)
- Sử dụng từng phần làm test set và phần còn lại là training set. Thực hiện lặp lại K lần với K fold đã chia.

- Lấy kết quả trung bình
- Một fold được sử dụng 1 lần cho việc testing và K-1 lần cho việc training

Đối với Weka, sau khi thực hiện cross-validation, Weka thực hiện chạy một lần nữa trên toàn bộ dataset để tạo ra một classifier mà chúng ta sử dụng.

Cross-validation sẽ đem lại độ chính xác tốt hơn so với Repeated holdout¹⁵.

Tuy nhiên Stratified Cross-validation¹⁶ lại tốt hơn Cross-validation. Weka mặc định sử dụng Stratified Cross-validation.

Qui luật:

- Sử dụng percentage split khi chúng ta có nhiều dữ liệu
- Nếu không, sử dụng K-Fold cross-validation (K=10)

5. So sánh Repeated Holdout và Stratified Cross-validation

Thực hiện kiểm tra với dataset [diabetes.arff](#) gồm 768 instance.

Baseline accuracy (ZeroR): 65%

- Repeated Holdout: percentage split là 90% với seed 1-10
- Cross-validation: K=10 với seed 1-10

Ta có kết quả như sau:

Seed	Repeated Holdout (%)	Cross-validation (%)
1	75.3	73.8
2	77.9	75.0
3	80.5	75.5
4	74.0	75.5
5	71.4	74.4
6	70.1	75.6
7	79.2	73.6
8	71.4	74.0
9	80.5	74.5
10	67.5	73.0

Tính toán ta có:

- Repeated Holdout: $\bar{x} = 74.8 \%$ và $\sigma = 4.6\%$

¹⁵ Repeated holdout: lặp lại việc tạo mô hình phân lớp nhiều lần trên những tập dữ liệu ngẫu nhiên từ dataset

¹⁶ Stratified Cross-validation: là thuật toán phân chia các fold dựa trên kỹ thuật Stratified Sampling, một kỹ thuật lấy mẫu trong đó các mẫu được chọn theo cùng một tỷ lệ khi chúng xuất hiện trong tổng thể.

- Cross-validation: $\bar{x} = 74.5\%$ và $\sigma = 0.9\%$

Nhận xét:

- Ta thấy với độ chính xác trung bình của 2 phương pháp là xấp xỉ nhau.
 - Tuy nhiên độ lệch chuẩn mà phương pháp Repeated Holdout mang lại cao hơn nhiều so với Cross validation.
- ⇒ Cross validation tốt hơn so với Repeated Holdout

III. Các mô hình phân lớp đơn giản

1. OneR - One attribute does all the work

Chỉ duy nhất một thuộc tính tham gia vào việc dự đoán của mô hình.

Thuật toán:

```
For each attribute,  
  For each value of the attribute,  
    make a rule as follows:  
      count how often each class appears  
      find the most frequent class  
      make the rule assign that class  
      to this attribute-value  
Calculate the error rate of this attribute's rules  
Choose the attribute with the smallest error rate
```

Ảnh: thuật toán của mô hình phân lớp OneR

VD:

Đối với dataset weather.norminal.arff

No.	1: outlook Nominal	2: temperature Nominal	3: humidity Nominal	4: windy Nominal	5: play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Ảnh: dataset weather.norminal.arff

Thực hiện thuật toán:

Attribute	Rules	Errors	Total errors
Outlook	Sunny → No	2/5	4/14
	Overcast → Yes	0/4	
	Rainy → Yes	2/5	
Temp	Hot → No*	2/4	5/14
	Mild → Yes	2/6	
	Cool → Yes	1/4	
Humidity	High → No	3/7	4/14
	Normal → Yes	1/7	
Wind	False → Yes	2/8	5/14
	True → No*	3/6	

Ảnh: các tỉ lệ lỗi đối với từng thuộc tính – OneR

Dựa vào tỉ lệ lỗi, ta có thể chọn thuộc tính Outlook hoặc Humidity làm output cho mô hình.

Chạy lại với Weka

```

=== Classifier model (full training set) ===

outlook:
  sunny    -> no
  overcast -> yes
  rainy    -> yes
(10/14 instances correct)

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances      10           71.4286 %
Incorrectly Classified Instances     4           28.5714 %
Kappa statistic                     0.3778
Mean absolute error                  0.2857
Root mean squared error              0.5345
Relative absolute error              61.5385 %
Root relative squared error          111.4773 %
Total Number of Instances           14
  
```

Ảnh: output khi sử dụng thuật toán OneR để đánh giá mô hình

2. Overfitting¹⁷

Overfitting là hiện tượng chúng ta có được tỉ lệ chính xác cao đối với training set mà tỉ lệ chính xác thấp trên testing set.

3. Naïve Bayes - Attributes contribute equally and independently

Dựa vào xác suất của tất cả attributes đóng góp vào class mà tạo nên mô hình dự đoán.

Ta có công thức như sau:

$$\Pr[H | E] = \frac{\Pr[E_1 | H] \Pr[E_2 | H] \dots \Pr[E_n | H] \Pr[H]}{\Pr[E]}$$

Với:

H: giá trị lớp

E: Thuộc tính

$\Pr[H|E]$ là xác suất dự đoán

$\Pr[E_i|H]$ là xác suất biết được của thuộc tính so với lớp

Chạy lại với Weka, trên tập dữ liệu [weather.norminal.arff](#) và thuật toán Naïve Bayes, với 10-folds, ta được output như sau:

¹⁷ Overfitting: học thuộc lòng

Naive Bayes Classifier		
Attribute	Class	
	yes (0.63)	no (0.38)
=====		
outlook		
sunny	3.0	4.0
overcast	5.0	1.0
rainy	4.0	3.0
[total]	12.0	8.0
temperature		
hot	3.0	3.0
mild	5.0	3.0
cool	4.0	2.0
[total]	12.0	8.0
humidity		
high	4.0	5.0
normal	7.0	2.0
[total]	11.0	7.0
windy		
TRUE	4.0	4.0
FALSE	7.0	3.0
[total]	11.0	7.0

Ảnh: xác suất $Pr[H | E]$ của dataset

4. Decision tree – use a few attributtes

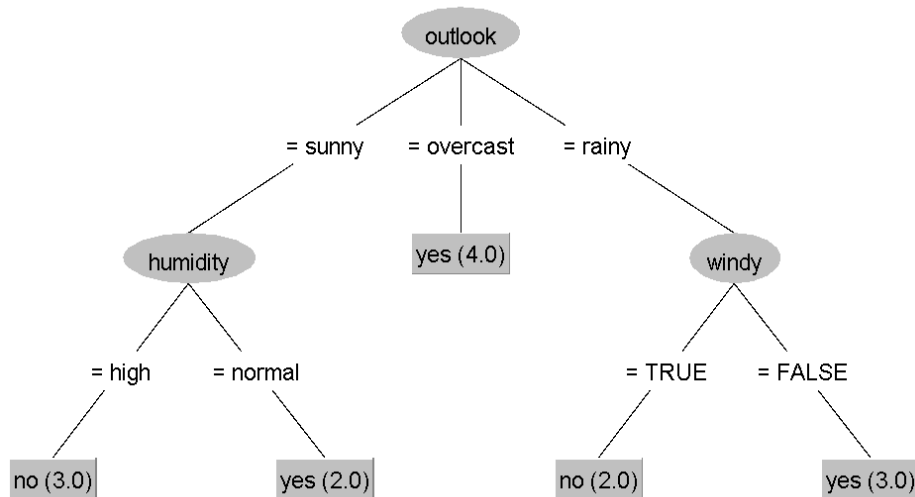
Một decision tree sẽ sử dụng một vài thuộc tính để phân loại các lớp của bài toán.

Thuật toán J48 trên Weka – dựa trên C4.5 một kế thừa của Id3. J48 hỗ trợ các phương thức cắt tía cây quyết định.

VD:

Đối với dataset weather.normal.arff , thuật toán J48, test option 10-folds, mô hình phân lớp cho độ chính xác 50%

Decision tree có dạng như sau:



Ảnh: Decision tree được tạo bởi thuật toán J48 trên dataset weather.norminal.arff

D. Lab04

Toàn bộ tư liệu của lab04 có thể tìm thấy tại github: <https://github.com/phanm-student-hcmus/ai-lab04>.

I. Phân tích yêu cầu đề bài:

1. Tìm hiểu các độ đo (metric)

Dựa vào metric ta có thể đánh giá mô hình là tốt hay không.

Ta cần tìm hiểu về Confusion matrix (tuy đây không phải một metric nhưng rất quan trọng) và một số metric như Accuracy, Precision, Recall, F1-score.

2. Yêu cầu về thuật toán

Chúng ta cần phân tích dữ liệu *Zoo Data Set* trên 2 thuật toán là Id3 và Naïve Bayes.

Id3

Thuật toán Id3 mà weka hỗ trợ chỉ có thể áp dụng trên các giá trị nominal và tập dữ liệu hoàn toàn không được chứa giá trị rỗng.

Naïve Bayes

Thuật toán Naïve Bayes có thể áp dụng cho nhiều kiểu dữ liệu của thuộc tính và tập dữ liệu có thể chứa giá trị rỗng.

3. Yêu cầu về dữ liệu:

Các phân lớp (class/type) cần có các tên mới đại diện cho lớp đó thay vì con số từ 1 đến 7

4. Kiểm tra kết quả dự đoán các mẫu test

NameIsSecret	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	1	?
NameIsSecret	0	1	1	0	1	0	0	0	1	1	0	0	2	1	1	0	?
NameIsSecret	0	0	1	0	0	0	1	1	1	1	1	0	0	1	0	0	?
NameIsSecret	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	?
NameIsSecret	0	0	1	0	0	1	1	1	1	1	0	0	4	1	0	0	?

5. Tìm hiểu hiện tượng quá khớp (overfitting) trên cây Id3

Tìm hiểu hiện tượng quá khớp (overfitting) dữ liệu và thực hiện các biện pháp để tránh quá khớp trên cây quyết định ID3.

6. So sánh 2 thuật toán Id3 và Naïve Bayes

Ngoài tập dữ liệu Zoo Data Set, ta cần so sánh và đánh giá 2 thuật toán trên một vài tập dữ liệu khác.

II. Tìm hiểu về các độ đo

1. Confusion matrix

Tuy **confusion matrix không phải là một metric**, nhưng nó rất quan trọng. Nó thể hiện có bao nhiêu điểm dữ liệu thật sự thuộc vào 1 class, và bao nhiêu điểm dữ liệu được dự đoán là rơi vào một class.

VD:

Đối với bài toán phân loại nhị phân (Binary Classification)

Giả sử chúng ta có bài toán phân loại các thư điện tử đến hộp thư là thư spam hay không spam.

Nhãn:

- 1 (positive): là thư spam, tương ứng với đây là nhãn xấu (bad label).
- 0 (negative): là thư không spam, tương ứng với đây là nhãn tốt (good label).

Quy ước các chỉ số như sau:

- TP (True Positive): Tổng số trường hợp dự đoán đúng mẫu dương.
- FP (False Positive): Tổng số trường hợp dự đoán sai mẫu dương.
- TN (True Negative): Tổng số trường hợp dự đoán đúng mẫu âm.
- FN (False Negative): Tổng số trường hợp dự đoán sai mẫu âm.

Giả sử ta có các số liệu như sau: tổng số mẫu khảo sát là 1000 mẫu trong đó có 900 mẫu là thư không spam và 100 mẫu là thư spam.

Kết quả của mô hình phân loại cho ra như sau:

	Real Values	
Predictions	Positive	Negative
Positive	TP = 50	FP = 50
Negative	FN = 25	TN = 875

Nhìn vào mô hình này Accuracy của mô hình sẽ là $(50+875)/1000 = 0.925$, tức là mô hình có độ chính xác đến 92.5%.

Tuy nhiên đối với trường hợp phát hiện thư spam, dự đoán có 100 thư spam nhưng chỉ có 50 dự đoán là đúng (chỉ đạt 50%). Tuy nhiên bài toán phân loại này, ta lại quan tâm đến độ chính xác của những dự đoán phân loại thư spam. Lúc này vai trò của F1 score, Precision và Recall sẽ được thể hiện trong việc đánh giá mô hình.

2. Accuracy

Accuracy thể hiện độ độ chính xác của mô hình dựa trên số dự đoán đúng trên tổng số mẫu cần dự đoán.

$$\text{Accuracy} = (TP + TN) / (TP + FP + TN + FN)$$

3. Precision

Precision là tỉ lệ của tổng số trường hợp dự đoán đúng mẫu dương trên tổng số trường hợp dự đoán là mẫu dương.

$$\text{Precision} = TP / (TP + FP).$$

Vậy độ precision càng cao thì mô hình sẽ dự đoán càng tốt các mẫu thuộc lớp dương (hay dự đoán càng tốt các trường hợp nhãn xấu).

4. Recall

Recall là tỉ lệ của tổng số trường hợp dự đoán đúng mẫu dương trên tổng số mẫu dương thật sự của dữ liệu đầu vào.

$$\text{Recall} = TP / (TP + FN)$$

Như tên gọi của nó, Recall cho biết mức độ bỏ sót các mẫu thuộc lớp positive của mô hình, nếu Recall càng cao chứng tỏ mô hình bỏ sót rất ít các mẫu thuộc lớp positive, chỉ khác với Precision là chỉ có sự tham gia của các mẫu đã được dự đoán là positive thì ở Recall có sự tham gia trên toàn bộ các mẫu của tập dữ liệu.

Trade off (đánh đổi) giữa Precision và Recall

Trong thực tế một mô hình binary classification lý tưởng phải đều thu được cả Precision và Recall cao, tuy nhiên điều này là rất khó xảy ra.

Xét ví dụ trên để tăng Precision thì có một cách đơn giản là tăng ‘ngưỡng’ để quyết định nhãn lên, khi đó sẽ đảm bảo việc dự đoán đúng các mẫu positive hơn nhưng sẽ làm giảm số lượng TP do đó làm giảm Recall.

Nếu nói lỏng ngưỡng để tăng Recall thì sẽ làm tăng nhanh số lượng mẫu được dự đoán là positive do đó lại làm giảm Precision.

Sự đánh đổi qua lại này thường xuyên diễn ra trong các bộ dữ liệu thực tế do đó cần một độ đo có thể kết hợp 2 độ đo trên, và đó chính là F1 Score.

5. F1 Score

Là trung bình điều hòa giữa recall và precision

$$F1 \text{ Score} = (2 * \text{precision} * \text{recall}) / (\text{precision} + \text{recall})$$

Điều đặc biệt là F1 score luôn nằm trong khoảng của Precision và Recall. Do đó đối với những trường hợp mà precision và recall quá chênh lệch thì F1 score sẽ cân bằng được cả hai giá trị này và giúp ta đưa ra một đánh giá khách quan hơn.

Bài toán phân loại đa lớp

Như đề bài của phần Decision Tree ở trên thì tập dữ liệu iris có 3 lớp hoa iris cần phân loại là Iris setosa, Iris versicolor và Iris virginica tương ứng với các giá trị lần lượt là 0, 1, 2 (Mỗi lớp có 50 mẫu).

Có một cách để tính các thông số trên gọi là Macro Average.

Giả sử ta có số liệu sau:

		Prediction			
Real value		Class 0	Class 1	Class 2	
	Class 0	50	0	0	$\text{recall}_0 = \frac{30}{30+0+0} = 1.000$
	Class 1	0	43	7	$\text{recall}_1 = \frac{43}{0+43+7} = 0.860$
	Class 2	0	5	45	$\text{recall}_2 = \frac{45}{0+5+45} = 0.900$
		$\text{precision}_0 = \frac{50}{50+0+0}$	$\text{precision}_1 = \frac{43}{0+43+5}$	$\text{precision}_2 = \frac{45}{0+7+45}$	

		=1.000	=0.896	=0.865	
--	--	--------	--------	--------	--

$$\text{MacroAveragePrecision} = \frac{\text{Precision}_0 + \text{Precision}_1 + \text{Precision}_2}{3} = 0.920$$

$$\text{MacroAverageRecall} = \frac{\text{Recall}_0 + \text{Recall}_1 + \text{Recall}_2}{3} = 0.920$$

$$\text{MacroAverageF1Score} = \frac{2 * \text{MacroAveragePrecision} * \text{MacroAverageRecall}}{\text{MacroAveragePrecision} + \text{MacroAverageRecall}} = 0.920$$

Ở mô hình phân loại đa lớp này, F1-score xấp xỉ 1, do đó, mô hình này có ý nghĩa phân loại rất tốt.

Dựa vào các metric trên, nếu F1-score có thể tính toán được, ta sẽ luôn luôn sử dụng F1-score để đánh giá mô hình.

III. Dataset:

Dataset được lấy từ (Forsyth, 1990) *CI Machine Learning Repository: Zoo Data Set*. Dataset chứa thông tin về động vật bao gồm 17 thuộc tính, 101 đối tượng được phân vào 7 lớp.

1. Thông tin thuộc tính

- | | |
|--|---|
| 1. animal name: Unique for each instance | 10. backbone: Boolean |
| 2. hair: Boolean | 11. breathes: Boolean |
| 3. feathers: Boolean | 12. venomous: Boolean |
| 4. eggs: Boolean | 13. fins: Boolean |
| 5. milk: Boolean | 14. legs: Numeric (set of values: {0,2,4,5,6,8}) |
| 6. airborne: Boolean | 15. tail: Boolean |
| 7. aquatic: Boolean | 16. domestic: Boolean |
| 8. predator: Boolean | 17. catsize: Boolean |
| 9. toothed: Boolean | 18. type: Numeric (integer values in range [1,7]) |

Ảnh: thông tin các thuộc tính của Zoo dataset, ảnh được chụp từ <http://archive.ics.uci.edu/ml/datasets/Zoo>

Giải thích ý nghĩa từng thuộc tính:

- | | |
|--|--|
| Animal name: tên động vật, có giá trị duy nhất | Backbone: có xương sống |
| Hair: có bờm | Breathe: hô hấp bằng mũi |
| Feather: Có lông vũ | Venomous: Có độc |
| Eggs: Có đẻ trứng | Fins: có vây |
| Milk: có sữa | Legs: số chân (chi) |
| Airborne: Trên trời | Tail: có đuôi |
| Aquatic: Dưới nước | Domestic: là vật nuôi |
| Predator: ăn thịt | Catsize: kích cỡ động vật (lớn hơn/nhỏ hơn so với mèo) |
| Toothed: Có răng | Type: phân lớp của động vật |

2. Phân tích dữ liệu thô

Ta sẽ dùng python để phân tích dữ liệu thô từ website (Tuy có thể dùng weka để làm điều này nhưng để tiện cho việc tạo file arff thì sẽ sử dụng python). Có thể xem source code phân tích dữ liệu tại ./src/code.ipynb hoặc tại github: <https://github.com/phatnm-student-hcmus/ai-lab04/blob/master/src/code.ipynb>

Từ mô tả của tập dữ liệu, tạo cols.csv chứa thông tin các cột của dữ liệu

```
cols = np.loadtxt("./arff_creator/zoo_cols.csv", delimiter=",", dtype=str)
cols
```

```
array(['animal-name', 'hair', 'feathers', 'eggs', 'milk', 'airborne',
      'aquatic', 'predator', 'toothed', 'backbone', 'breathes',
      'venomous', 'fins', 'legs', 'tail', 'domestic', 'catsize', 'class'],
      dtype='<U11')
```

Ảnh: Các cột của dữ liệu

Kiểm tra tập dữ liệu thô:

```
zoo_df = pd.read_csv('./arff_creator/zoo.data', names=cols)
zoo_df.head(3)
```

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	\
0	aardvark	1	0	0	1	0	0	1	
1	antelope	1	0	0	1	0	0	0	
2	bass	0	0	1	0	0	1	1	

	toothed	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	\
0	1	1	1	0	0	4	0	0	1	
1	1	1	1	0	0	4	1	0	1	
2	1	1	0	0	1	0	1	0	0	

	class
0	1
1	1
2	4

Ảnh: 3 dòng đầu tiên của dữ liệu

Ta thấy, 1 đại diện cho có hay giá trị True, 0 đại diện cho không hay hay trị False.

Hiển thị số dòng, cột của dữ liệu

Số dòng: số đối tượng nghiên cứu

Số cột: số thuộc tính + lớp (class)

```
num_rows, num_cols = zoo_df.shape
print(f'Số lượng dòng: {num_rows}\nSố lượng cột: {num_cols}')
```

Số lượng dòng: 101

Số lượng cột: 18

Ảnh: Số dòng và cột của dữ liệu

Kiểm tra các dòng dữ liệu có các dòng bị lặp không

```
have_duplicate_rows = zoo_df.duplicated().any()

print("Có dòng bị lặp: ", have_duplicate_rows)
```

Có dòng bị lặp: False

Ảnh: kết quả output kiểm tra dữ liệu có dòng bị lặp không

Từ output, ta thấy dữ liệu **không** chứa các dòng dữ liệu bị lặp.

Kiểm tra dữ liệu bất thường

Kiểm tra animal name (giá trị duy nhất) có bị lặp không

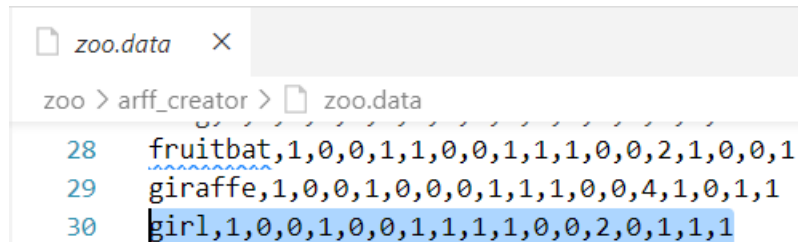
```
have_duplicate_animal_names = zoo_df[cols[0]].duplicated().any()
print("Có animal name bị lặp: ", have_duplicate_animal_names)
```

Có animal name bị lặp: True

Ảnh: Kết quả output kiểm tra thuộc tính animal-name có giá trị bị lặp không

Kiểm tra có chứa đối tượng bất thường hay không

Từ file dữ liệu zoo.data, ta phát hiện đối tượng girl không thuộc tập dữ liệu zoo.



```
zoo > arff_creator > zoo.data
28 fruitbat,1,0,0,1,1,0,0,1,1,1,0,0,2,1,0,0,1
29 giraffe,1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1,1
30 girl,1,0,0,1,0,0,1,1,1,1,0,0,2,0,1,1,1
```

Ảnh: Ảnh chụp màn hình dòng dữ liệu bất thường

Ta sẽ dùng weka để xử lý dòng các dòng dữ liệu này sau.

Tỉ lệ giá trị thiếu và thống kê mô tả của từng cột

```
missing_ratio = zoo_df.isnull().sum()  
missing_ratio = missing_ratio / num_rows  
missing_ratio
```

```
animal-name    0.0  
hair           0.0  
feathers       0.0  
eggs          0.0  
milk          0.0  
airborne      0.0  
aquatic       0.0  
predator      0.0  
toothed       0.0  
backbone      0.0  
breathes      0.0  
venomous      0.0  
fins          0.0  
legs          0.0  
tail          0.0  
domestic      0.0  
catsize       0.0  
class         0.0  
dtype: float64
```

Ảnh: output tỉ lệ giá trị thiếu của dataset

Ta thấy tập dữ liệu hoàn toàn **không chứa giá trị thiếu**.

IV. Tạo file ARFF cho tập dữ liệu Zoo

1. Tạo phần mô tả

Từ file zoo.names mà website cung cấp, ta thêm các kí tự % ở trước mỗi dòng và thay đổi một số thông tin cho phù hợp với các thuật toán yêu cầu. Sau đó lưu lại với tên zoo_arff_explanation.txt.

Bảng các thông tin đã thay đổi cho phù hợp với tập dữ liệu

STT	Zoo.names	zoo_arff_explanation.txt	Giải thích
1	6. Number of Attributes: 18 (animal name, 15 Boolean attributes, 2 numerics)	6. Number of Attributes: 17 + type = 18 (all Boolean valued but 'animal-name' as string and 'legs' attribute as set of values)	
2	1. animal name: Unique for each instance	1. animal name: string	Do mỗi động vật đều có tên khác nhau, và các động vật dùng để dự đoán cũng sẽ có tên khác nên không thể dùng giá trị nominal cho thuộc tính này
3	14. legs Numeric (set of values: {0,2,4,5,6,8})	14. legs {0,2,4,5,6,8}	Chuẩn hóa 'legs' về dạng nominal
4	Boolean	{yes, no}	Chuẩn hóa các giá trị Boolean (1/0) về nominal (yes/no)
5	18. Type Numeric (integer values in range [1,7])	18. Type {Mammal, Bird, Reptile, Fish, Amphibian, Bug, Invertebrate}	Chuẩn hóa tên mới đại diện cho lớp ¹⁸

Có thể xem file tại ./src/arff-creator/zoo_arff_explanation.txt hoặc tại [github](#)

¹⁸ Tên mới đại diện cho lớp: các tên được tham khảo từ UCI Machine Learning. (2022, 12 22). Zoo Animal Classification | Kaggle. Được truy lục từ Kaggle: <https://www.kaggle.com/datasets/uciml/zoo-animal-classification>

2. Tạo phần cấu trúc dữ liệu

Dựa vào mô tả ta có cấu trúc dữ liệu như sau, có thể xem tại ./src/arff_creator/zoo_arff_relation.txt hoặc tại github: [github](#)

```
@RELATION zoo

@ATTRIBUTE animal-name string
@ATTRIBUTE hair {yes, no}
@ATTRIBUTE feathers {yes, no}
@ATTRIBUTE eggs {yes, no}
@ATTRIBUTE milk {yes, no}
@ATTRIBUTE airborne {yes, no}
@ATTRIBUTE aquatic {yes, no}
@ATTRIBUTE predator {yes, no}
@ATTRIBUTE toothed {yes, no}
@ATTRIBUTE backbone {yes, no}
@ATTRIBUTE breathes {yes, no}
@ATTRIBUTE venomous {yes, no}
@ATTRIBUTE fins {yes, no}
@ATTRIBUTE legs {0, 2, 4, 5, 6, 8}
@ATTRIBUTE tail {yes, no}
@ATTRIBUTE domestic {yes, no}
@ATTRIBUTE catsize {yes, no}
@ATTRIBUTE class {Mammal, Bird, Reptile, Fish, Amphibian, Bug,
Invertebrate}

@DATA
```

3. Tạo phần dữ liệu

Sử dụng python để chuẩn hóa dữ liệu sau đó xuất dữ liệu đã chuẩn hóa ra file text.

Để chuẩn hóa dữ liệu thô, ta cần:

- Chuyển các giá trị 0 thành no và 1 thành yes ở các cột (không bao gồm legs)
- Chuyển đổi tên lớp từ số từ 1 đến 7 thành các giá trị có nghĩa

Cài đặt thuật toán chuẩn hóa:

Tạo một file zoo_classes.csv để lưu trữ các lớp

```
# load cols and classes
classes = np.loadtxt("./arff_creator/zoo_classes.csv", delimiter=",", dtype=str)
classes.tolist()

['Mammal', 'Bird', 'Reptile', 'Fish', 'Amphibian', 'Bug', 'Invertebrate']

# change 0 to "no" and 1 to "yes" and fix class names
def fix_zoo_df(zoo_df):
    # change no to test
    for col in [col for col in cols[1:-1] if col != 'legs']:
        col_to_replace = zoo_df[col]

        col_to_replace.replace(to_replace=0, value='no', inplace=True)
        col_to_replace.replace(to_replace=1, value='yes', inplace=True)

        zoo_df[col] = col_to_replace

    for idx, animal_class in enumerate(classes):
        col_to_replace = zoo_df['class']
        col_to_replace.replace(to_replace=idx + 1, value=animal_class,
                               inplace=True)

    return zoo_df
```

Ảnh: thuật toán chuẩn hóa dữ liệu thô ở dạng dataframe

Sau khi chuẩn hóa, ta được kết quả như sau:

```
fix_zoo_df(zoo_df).head(3)
```

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	aardvark	yes	no	no	yes	no	no	yes	yes	
1	antelope	yes	no	no	yes	no	no	no	yes	
2	bass	no	no	yes	no	no	yes	yes	yes	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	class
0	yes	yes	no	no	4	no	no	yes	Mammal
1	yes	yes	no	no	4	yes	no	yes	Mammal
2	yes	no	no	yes	0	yes	no	no	Fish

Ảnh: kết quả dataframe sau khi chuẩn hóa

Tạo phần dữ liệu cho bài toán:

```
fix_zoo_df(zoo_df).to_csv("./arff_creator/zoo_arff_data.txt", index=False,
                           header=False)
```

Ảnh: code tạo phần dữ liệu

Giải thích:

Do cấu trúc csv và cấu trúc data của arff là giống nhau (đều phân cách dữ liệu bằng dấu phẩy “,”) nên ta có thể sử dụng `pd.DataFrame.to_csv` như một mẹo để tạo phần dữ liệu cho bài toán.

Phần dữ liệu được tạo có thể xem tại `./src/arff_creator/zoo_arff_data.txt` hoặc tại github: [github](#)

4. Tạo file arff cho training set của bài toán

Sau khi tạo được 3 phần của file arff, ta chỉ cần nối chúng lại theo thứ tự: mô tả > cấu trúc dữ liệu > dữ liệu

```
def create_arff(explanation_file, relation_file, data_file, output_file):
    filenames = [explanation_file, relation_file, data_file]
    with open(output_file, 'w') as outfile:
        for fname in filenames:
            with open(fname) as infile:
                for line in infile:
                    outfile.write(line)

            outfile.write("\n\n")
        outfile.write("%\n%\n%\n")
```

Ảnh: hàm tạo file arff

File arff có thể xem trong thư mục nộp bài tại `./src/arff/zoo.arff` hoặc tại github: [github](#)

5. Tạo file Arff cho file cần dự đoán

File cần dự đoán sẽ có cấu trúc đơn giản hơn, nên không cần phần mô tả.

Lớp cần dự đoán sẽ ở dạng giá trị bị mất (missing value) – được thể hiện bằng dấu chấm hỏi “?”.

predict.arff

```
@RELATION zoo_prediction

@ATTRIBUTE animal-name string
@ATTRIBUTE hair {yes, no}
@ATTRIBUTE feathers {yes, no}
@ATTRIBUTE eggs {yes, no}
@ATTRIBUTE milk {yes, no}
@ATTRIBUTE airborne {yes, no}
@ATTRIBUTE aquatic {yes, no}
@ATTRIBUTE predator {yes, no}
@ATTRIBUTE toothed {yes, no}
@ATTRIBUTE backbone {yes, no}
@ATTRIBUTE breathes {yes, no}
@ATTRIBUTE venomous {yes, no}
@ATTRIBUTE fins {yes, no}
@ATTRIBUTE legs {0, 2, 4, 5, 6, 8}
```

```
@ATTRIBUTE tail {yes, no}
@ATTRIBUTE domestic {yes, no}
@ATTRIBUTE catsize {yes, no}
@ATTRIBUTE class {Mammal, Bird, Reptile, Fish, Amphibian, Bug,
Invertebrate}

@DATA

NameIsSecret, yes, no, no, yes, no, no, no, yes, yes, yes, no, no, 4, yes, no, yes, ?
NameIsSecret, no, yes, yes, no, yes, no, no, no, yes, yes, no, no, 2, yes, yes, no, ?
NameIsSecret, no, no, yes, no, no, no, yes, yes, yes, yes, yes, no, 0, yes, no, no, ?
NameIsSecret, no, no, yes, no, no, yes, yes, yes, yes, yes, no, no, yes, 0, yes, no, no, ?
NameIsSecret, no, no, yes, no, no, yes, yes, yes, yes, yes, yes, no, no, 4, yes, no, no, ?
```

File predict.arff có thể xem tại [./weka/zoo/dataset/predict.arff](#) hoặc tại [github](#)

Thể hiện dữ liệu của tập cần dự đoán dưới dạng bảng:

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	NameIsSecret	yes	no	no	yes	no	no	no	yes	
1	NameIsSecret	no	yes	yes	no	yes	no	no	no	
2	NameIsSecret	no	no	yes	no	no	no	yes	yes	
3	NameIsSecret	no	no	yes	no	no	yes	yes	yes	
4	NameIsSecret	no	no	yes	no	no	yes	yes	yes	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	class
0	yes	yes	no	no	4	yes	no	yes	?
1	yes	yes	no	no	2	yes	yes	no	?
2	yes	yes	yes	no	0	yes	no	no	?
3	yes	no	no	yes	0	yes	no	no	?
4	yes	yes	no	no	4	yes	no	no	?

Ảnh: dữ liệu cần dự đoán

V. Tiền xử lý dữ liệu với Weka

Như đã đề cập ở trên, bộ dữ liệu bao gồm các dữ liệu bất thường, cho nên ta cần tiền xử lý trước khi áp dụng một thuật toán (weka classifier) để xử lý.

Sử dụng tính năng edit dataset ở tab proprocess, ta xóa instance “girl” và đổi tên 2 “frog” bị trùng nhau thành “frog_1” và “frog_2”. Sau đó lưu lại với tên zoo_cleaned.arff. File zoo_cleaned.arff có thể xem tại [./weka/zoo/dataset/zoo_cleaned.arff](#) hoặc tại [github](#).

Từ bây giờ, mọi thuật toán sẽ được áp dụng trên bộ dữ liệu mới (zoo_cleaned.arff)

VI. ZeroR - Zoo Data Set - Baseline accuracy

Sử dụng classify ZeroR để tìm baseline accuracy cho mô hình.

Summary

=== Summary ===			
Correctly Classified Instances	40	40	%
Incorrectly Classified Instances	60	60	%
Kappa statistic	0		
Mean absolute error	0.2201		
Root mean squared error	0.3306		
Relative absolute error	100	%	
Root relative squared error	100	%	
Total Number of Instances	100		

Tập dữ liệu bao gồm 40/100 đối tượng thuộc phân lớp Mammal. Do đó baseline accuracy là 40%.

VII. Cách áp dụng Weka vào bài toán:

1. Sử dụng FilteredClassify

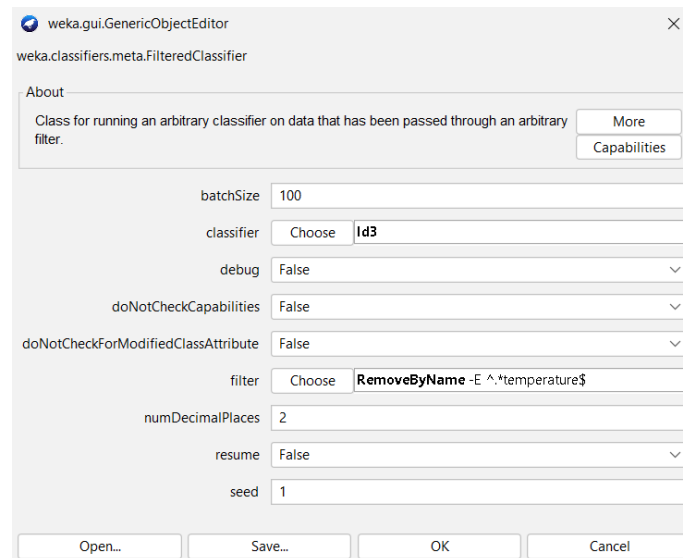
Do dữ liệu và file cần dự đoán chứa một thuộc tính kiểu string (animal name), thuật toán Id3 không thể áp dụng ngay được.

Để đảm bảo tính nhất quán của dữ liệu, ta sẽ không sử dụng các Weka filter ở tab Preprocess để xử lý mà sẽ sử dụng FilteredClassifier, một Weka classifier hỗ trợ filter trước khi áp dụng thuật toán.

VD:

Trên tập dữ liệu weather.nominal, ta cần áp dụng filter để xóa thuộc tính temperature và sau đó áp dụng classifier, ta sẽ làm như sau:

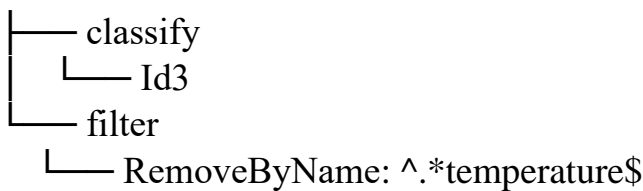
- Chọn FilteredClassifier ở tab Classify: meta > FilteredClassifier
- Click vào cài đặt của FilteredClassify
 - o Chọn classifier Id3: trees > Id3
 - o Chọn filter RemoveByName: unsupervised > attributes > RemoveByName. Tùy chỉnh lại cài đặt filter để xóa thuộc tính "temperature"
 - o Sau khi cài đặt:



Ảnh: các thông số của filteredClassifier

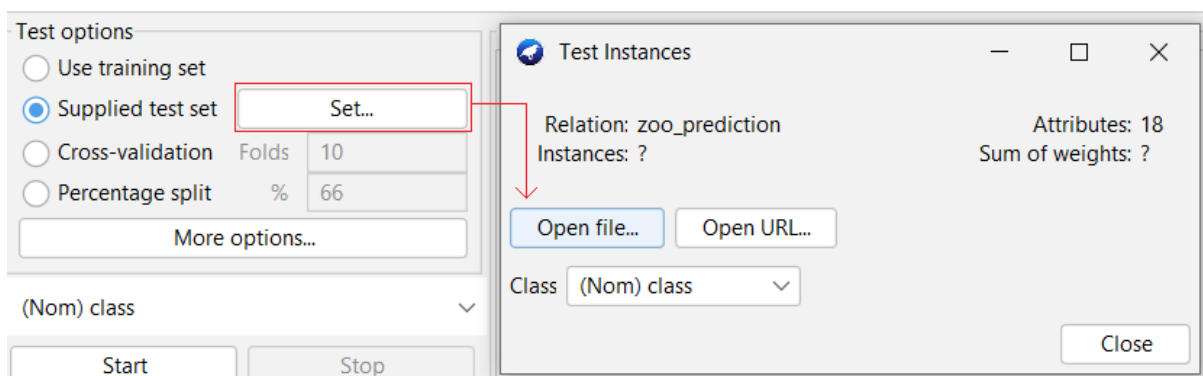
FilteredClassifier trên ví dụ trên có thể thể hiện cách cài đặt ở dạng cây như sau:

FilteredClassifier



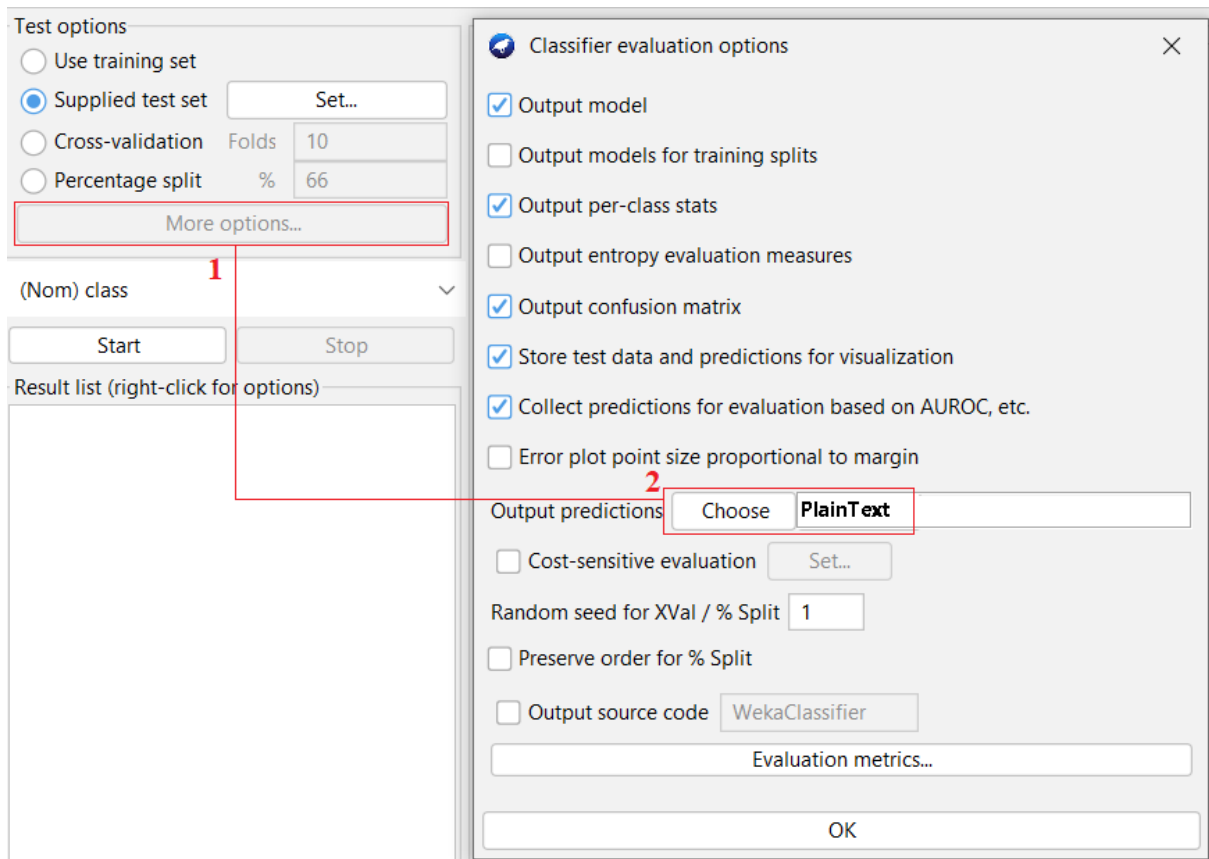
2. Cách mô hình Weka để dự đoán

- **Bước 1:** Ở phần test option, mở file cần mô hình đưa ra dự đoán như một test set



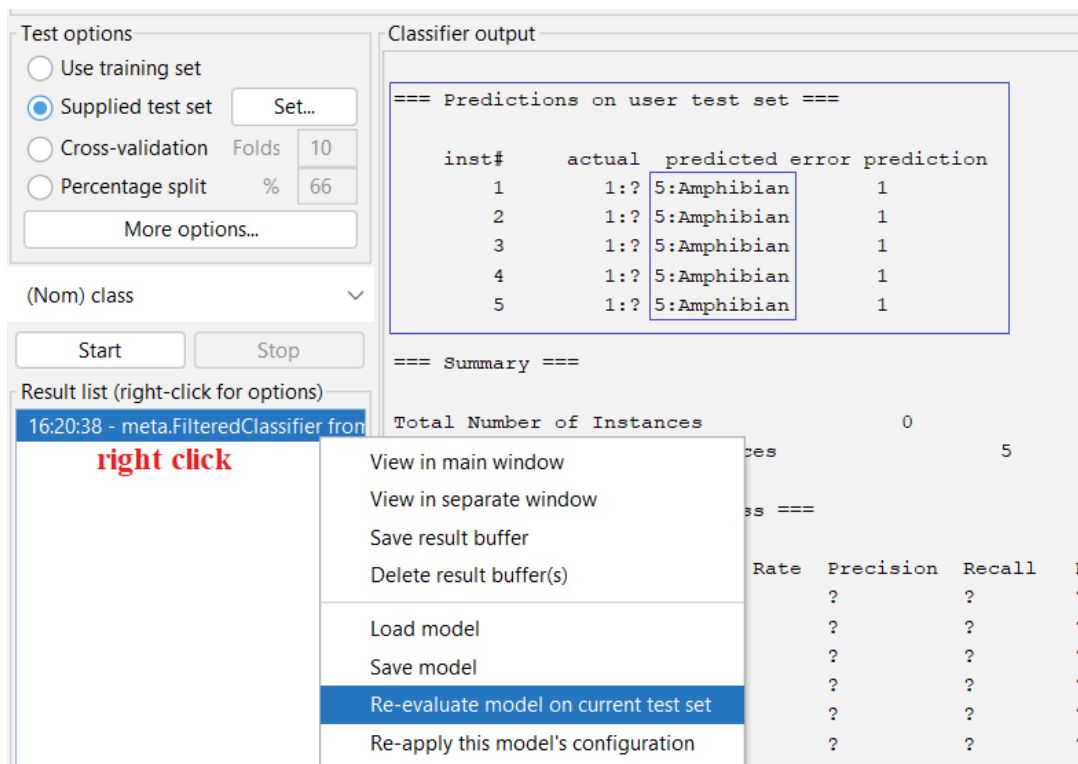
Ảnh: Cách chọn file chứa dữ liệu cần dự đoán

- **Bước 2:** Để hiển thị kết quả dự đoán ra màn hình output. Ở phần more option, thay đổi tùy chọn của predict output thành lựa chọn mong muốn, ở đây tôi sẽ chọn plain text (kết quả sẽ được in ra màn hình).



Ảnh: Hiện thị kết quả dự đoán

- **Bước 3:** Tiến hành dự đoán dựa trên mô hình ở result list



Ảnh: Các bước mô hình đưa ra dự đoán

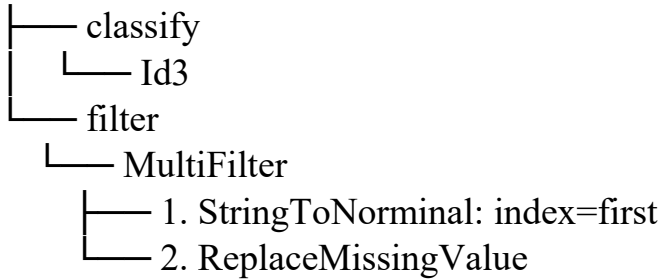
VIII. Id3 – Zoo Data Set - Overfitting

1. Classify

Ta sẽ áp dụng FilteredClassifier, để tiền xử lí và tạo mô hình phân lớp.

FilteredClassifier được cài đặt như sau:

FilteredClassifier/

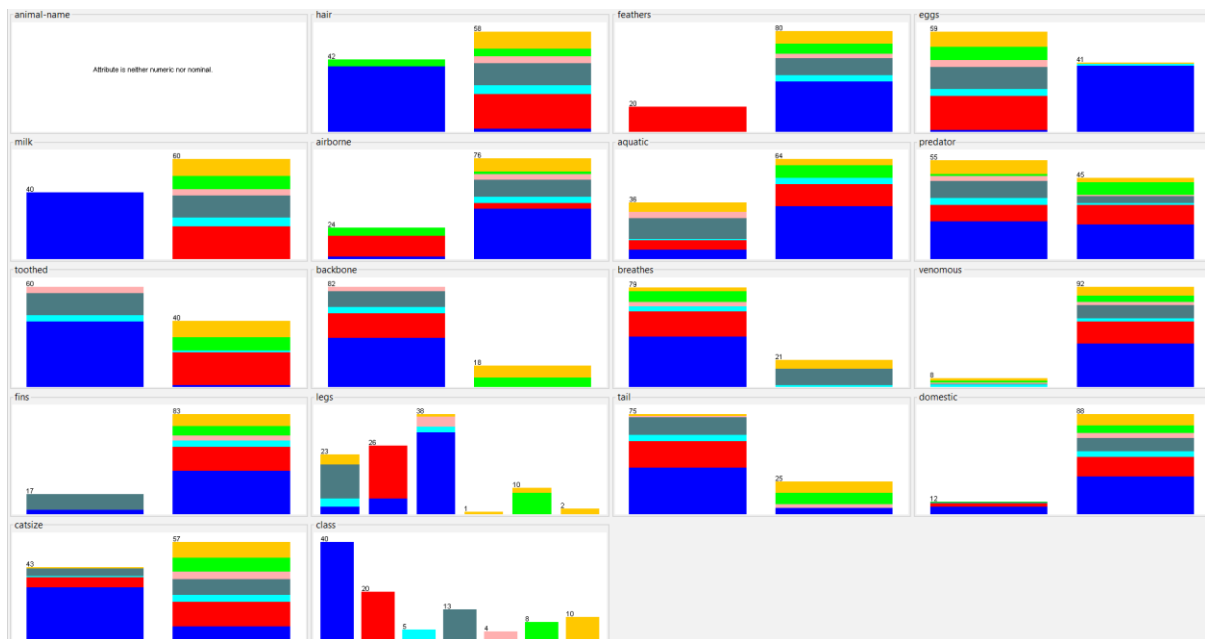


Do Id3 chỉ áp dụng được trên tập dữ liệu chứa toàn bộ các giá trị nominal, nên ta cần thay đổi kiểu dữ liệu của animal name từ string thành nominal (filter 1/Multifilter)

Dữ liệu testing/predict có thể bị thiếu các giá trị nominal của animal name (của dữ liệu training) do các animal name là độc lập => cần sử dụng thêm filter ReplaceMissingValue để xử lí điều này.

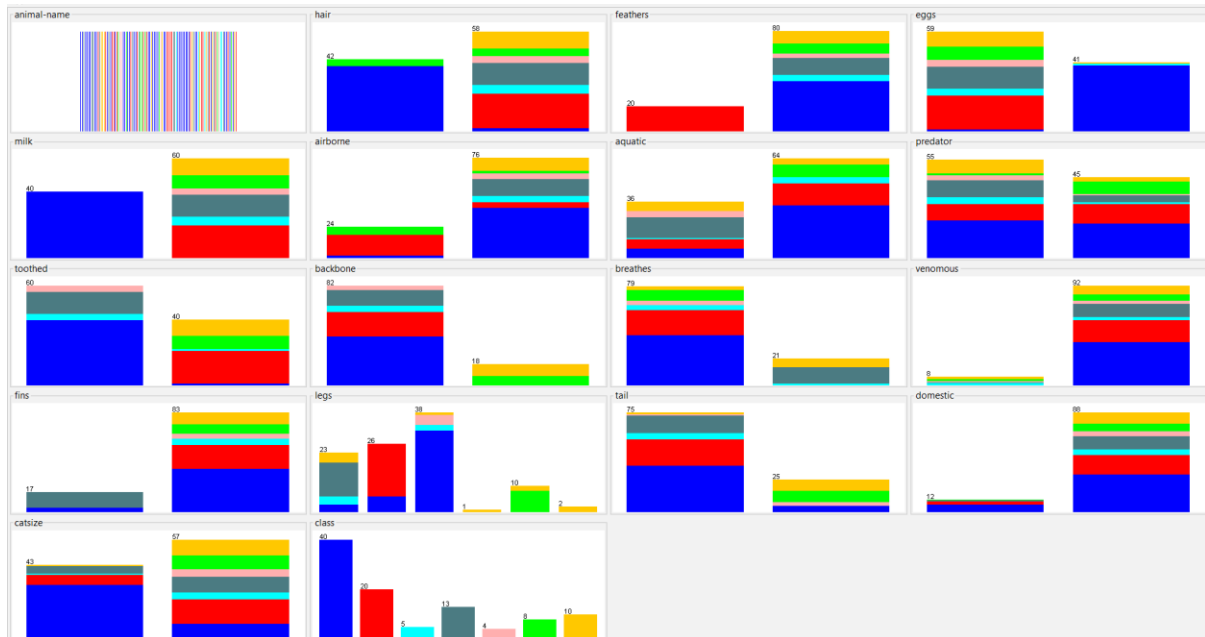
2. Trục quan quá trình tiền xử lí bằng tab preprocess

Dữ liệu trước khi được filter (các giá trị của thuộc tính animal name không được hiển thị)



Ảnh: Zoo Data Set trước tiền xử lí

Dữ liệu sau khi được filter (animal name đã được nhận diện và có số lượng bằng nhau và bằng 1):



Ảnh: Zoo Data Set sau tiền xử lí

3. Áp dụng Weka classifier Id3

Output của mô hình có thể xem tại `./weka/zoo/train_output/OUTPUT_Id3-overfitting.txt` hoặc tại [github](#).

Mô hình của thuật toán được xuất ra file lưu trữ tại `./models/ filteredClassifier-id3-multiFilter_stringToNominal_replaceMissingAttr-seed19126008.model`

Một số cài đặt bổ sung cho quá trình phân lớp:

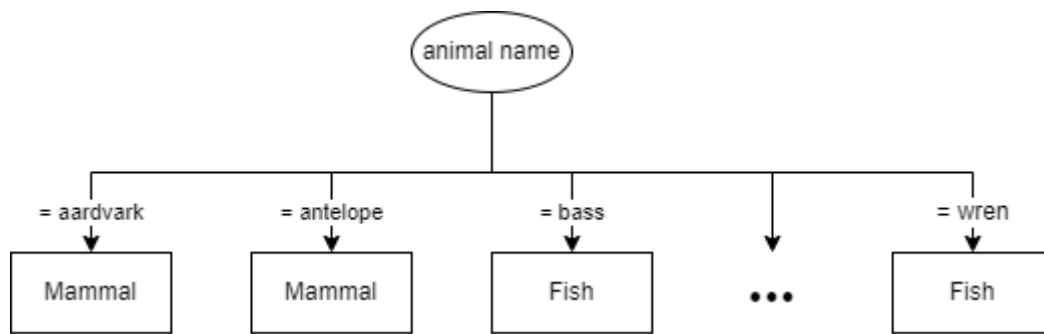
- Test option: vì có khá ít dữ liệu, nên ta sử dụng các chia dữ liệu 10-folds
- Seed: tất cả seed (nếu có) sử dụng cho bài toán sẽ là 19126008

Cây tạo bởi thuật toán

```
animal-name = aardvark: Mammal
animal-name = antelope: Mammal
animal-name = bass: Fish
.
.
.
animal-name = wren: Bird
```

Cây được tạo bởi thuật toán còn rất nhiều nhánh, có thể xem chi tiết cây tại output của mô hình.

Trực quan hóa cây được tạo bởi Id3



Ảnh: Cây quyết định của tập dữ liệu Zoo Data Set bằng thuật toán Id3

Summary của mô hình

=== Summary ===

Correctly Classified Instances	24	24	%
Incorrectly Classified Instances	76	76	%
Kappa statistic	-0.0066		
Mean absolute error	0.2171		
Root mean squared error	0.466		
Relative absolute error	98.6617	%	
Root relative squared error	140.9624	%	
Total Number of Instances	100		

Độ chính xác của mô hình: 24%

Độ đo

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.400	0.400	0.400	0.400	0.400	Mammal
0.300	0.300	0.200	0.300	0.240	Bird
0.000	0.000	?	0.000	?	Reptile
0.077	0.103	0.100	0.077	0.087	Fish
0.000	0.104	0.000	0.000	0.000	Amphibian
0.125	0.098	0.100	0.125	0.111	Bug
0.000	0.000	?	0.000	?	Invertebrate
0.240	0.245	?	0.240	?	Weighted Avg.

Một số giá trị không tính toán được => Một số giá trị độ đo bị thiếu.

Dự đoán dựa trên mô hình vừa tạo

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	NameIsSecret	yes	no	no	yes	no	no	no	yes	
1	NameIsSecret	no	yes	yes	no	yes	no	no	no	
2	NameIsSecret	no	no	yes	no	no	no	yes	yes	
3	NameIsSecret	no	no	yes	no	no	yes	yes	yes	
4	NameIsSecret	no	no	yes	no	no	yes	yes	yes	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	class
0	yes	yes	no	no	4	yes	no	yes	?
1	yes	yes	no	no	2	yes	yes	no	?
2	yes	yes	yes	no	0	yes	no	no	?
3	yes	no	no	yes	0	yes	no	no	?
4	yes	yes	no	no	4	yes	no	no	?

Ảnh: dữ liệu cần dự đoán

Kết quả dự đoán

```
=== Predictions on user test set ===
```

inst#	actual	predicted	error	prediction (animal-name)
1	1:?	1:Mammal	1	(NameIsSecret)
2	1:?	1:Mammal	1	(NameIsSecret)
3	1:?	1:Mammal	1	(NameIsSecret)
4	1:?	1:Mammal	1	(NameIsSecret)
5	1:?	1:Mammal	1	(NameIsSecret)

Nhận xét

Ở cây được tạo ra, mỗi animal name (id) sẽ luôn dự đoán được kết quả phân lớp.

Do đó, nếu động vật cần dự đoán không có tên nằm trong tập dữ liệu training thì kết quả trả về là là phân lớp chiếm đa số trong tập train (do filter ReplaceMissingValue)

Dự vào accuracy (0.24), f1-score (0.4, 0.24, 0.087, 0.0, 0.111) theo từng phân lớp (có phân lớp không tính được). Tất cả đều rất thấp so với 1. Vì đối với decision tree, các metric đều bằng 1. Nhưng trên tập test (cách chia dữ liệu 10-folds), các metric đều rất thấp. Do đó đã có hiện tượng **overfitting** xảy ra.

IX. Id3 – Zoo Data Set – Sửa lỗi Overfitting

Output của mô hình có thể xem tại ./weka/zoo/train_output/ OUTPUT_Id3_NO-overfitting.txt hoặc tại [github](#).

Để xử lý overfitting trên Decision tree, ta có thể thực hiện một trong các sau:

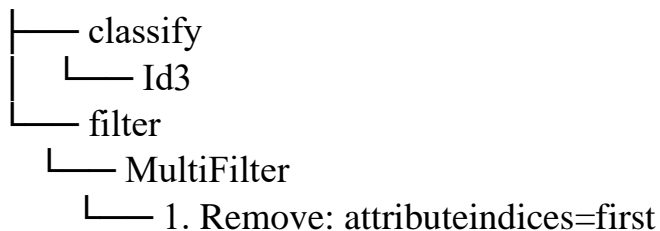
- Sử dụng kỹ thuật cắt tỉa (prunning decision tree) – không phân chia ra thành node con nếu số lượng đối tượng thuộc nốt con quá nhỏ.
- Gom thành những nhóm mới dựa vào sự liên quan với nhau (cùng là động vật ăn thịt, cùng đẻ trứng, ...)
- Loại bỏ một số thuộc tính không thiết, hoặc không tham gia vào việc đánh giá của mô hình.
- Đối với thuật toán Id3 trên Weka, chúng ta chỉ có thể áp dụng cách thứ 3, loại bỏ thuộc tính không ảnh hưởng. Thuộc tính “animal name” là duy nhất cho mỗi đối tượng, nên ta có thể loại bỏ nó để tránh sự bias trong mô hình.

1. Classify

Ta sẽ áp dụng FilteredClassifier, để tiền xử lý và tạo mô hình phân lớp

FilteredClassifier được cài đặt như sau:

FilteredClassifier/



Không cần filter ReplaceMissingValue vì dữ liệu đã hợp lệ.

2. Trục quan quá trình tiền xử lý bằng tab preprocess

Dữ liệu trước khi được filter: có 18 thuộc tính (đã bao gồm thuộc tính lớp)

Current relation
Relation: zoo
Instances: 100
Attributes: 18
Sum of weights: 100

Attributes

All None Invert Pattern

No.	Name
1	<input type="checkbox"/> animal-name
2	<input type="checkbox"/> hair
3	<input type="checkbox"/> feathers
4	<input type="checkbox"/> eggs
5	<input type="checkbox"/> milk
6	<input type="checkbox"/> airborne
7	<input type="checkbox"/> aquatic
8	<input type="checkbox"/> predator
9	<input type="checkbox"/> toothed
10	<input type="checkbox"/> backbone
11	<input type="checkbox"/> breathes
12	<input type="checkbox"/> venomous
13	<input type="checkbox"/> fins
14	<input type="checkbox"/> legs
15	<input type="checkbox"/> tail
16	<input type="checkbox"/> domestic
17	<input type="checkbox"/> catsize
18	<input type="checkbox"/> class

Ảnh: dữ liệu ban đầu

Dữ liệu sau khi được filter: có 17 thuộc tính (đã bao gồm thuộc tính lớp)

Current relation
Relation: zoo-weka.filters.unsupervised.attribute.Remove-R1
Instances: 100
Attributes: 17
Sum of weights: 100

Attributes

All None Invert Pattern

No.	Name
1	<input type="checkbox"/> hair
2	<input type="checkbox"/> feathers
3	<input type="checkbox"/> eggs
4	<input type="checkbox"/> milk
5	<input type="checkbox"/> airborne
6	<input type="checkbox"/> aquatic
7	<input type="checkbox"/> predator
8	<input type="checkbox"/> toothed
9	<input type="checkbox"/> backbone
10	<input type="checkbox"/> breathes
11	<input type="checkbox"/> venomous
12	<input type="checkbox"/> fins
13	<input type="checkbox"/> legs
14	<input type="checkbox"/> tail
15	<input type="checkbox"/> domestic
16	<input type="checkbox"/> catsize
17	<input type="checkbox"/> class

Ảnh: dữ liệu sau khi được filter

3. Áp dụng Weka classifier Id3

Mô hình của thuật toán được xuất ra file lưu trữ tại `./models/filteredClassifier-id3-multiFilter_removeAttr_animalName-seed19126008.model`

Một số cài đặt bổ sung cho quá trình phân lớp:

- Test option: vì có khá ít dữ liệu, nên ta sử dụng các chia dữ liệu 10-folds
- Seed: tất cả seed (nếu có) sử dụng cho bài toán sẽ là 19126008

Cây tạo bởi mô hình

```
legs = 0
| fins = yes
| | eggs = yes: Fish
| | eggs = no: Mammal
| fins = no
| | toothed = yes: Reptile
| | toothed = no: Invertebrate
legs = 2
| hair = yes: Mammal
| hair = no: Bird
legs = 4
| hair = yes: Mammal
| hair = no
| | aquatic = yes
| | | toothed = yes: Amphibian
| | | toothed = no: Invertebrate
| | aquatic = no: Reptile
legs = 5: Invertebrate
legs = 6
| aquatic = yes: Invertebrate
| aquatic = no: Bug
legs = 8: Invertebrate
```

Summary của mô hình

```
=== Summary ===

Correctly Classified Instances      97           97      %
Incorrectly Classified Instances    2            2      %
Kappa statistic                    0.9734
Mean absolute error                 0.0058
Root mean squared error             0.076
Relative absolute error             2.6535 %
Root relative squared error         23.1367 %
UnClassified Instances             1            1      %
Total Number of Instances          100
```

Độ chính xác của mô hình: 97%

Có một mẫu test không phân lớp được bởi vì mẫu này thuộc starfish – 5 chân (chỉ có duy nhất một đối tượng 5 chân trong dataset)

Độ đo

```
=== Detailed Accuracy By Class ===

TP Rate  FP Rate  Precision  Recall   F-Measure  Class
1.000    0.000    1.000     1.000    1.000      Mammal
1.000    0.000    1.000     1.000    1.000      Bird
0.800    0.000    1.000     0.800    0.889      Reptile
1.000    0.000    1.000     1.000    1.000      Fish
1.000    0.021    0.667     1.000    0.800      Amphibian
1.000    0.000    1.000     1.000    1.000      Bug
0.889    0.000    1.000     0.889    0.941      Invertebrate
0.980    0.001    0.987     0.980    0.981      Weighted Avg.
```

Output chỉ rõ (ở fold thứ 4):

1	1:Mammal	1:Mammal	1 (leopard)
---	----------	----------	-------------

Nhập môn trí tuệ nhân tạo

2	1:Mammal	1:Mammal	1	(seal)
3	1:Mammal	1:Mammal	1	(squirrel)
4	1:Mammal	1:Mammal	1	(buffalo)
5	3:Reptile	5:Amphibian	+	1 (tuatara)
6	7:Invertebrate		?	? (starfish)
7	2:Bird	2:Bird	1	(swan)
8	2:Bird	2:Bird	1	(ostrich)
9	4:Fish	4:Fish	1	(dogfish)
10	4:Fish	4:Fish	1	(bass)

Dự đoán dựa trên mô hình vừa tạo

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	NameIsSecret	yes	no	no	yes	no	no	no	yes	
1	NameIsSecret	no	yes	yes	no	yes	no	no	no	
2	NameIsSecret	no	no	yes	no	no	no	yes	yes	
3	NameIsSecret	no	no	yes	no	no	yes	yes	yes	
4	NameIsSecret	no	no	yes	no	no	yes	yes	yes	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	class
0	yes	yes	no	no	4	yes	no	yes	?
1	yes	yes	no	no	2	yes	yes	no	?
2	yes	yes	yes	no	0	yes	no	no	?
3	yes	no	no	yes	0	yes	no	no	?
4	yes	yes	no	no	4	yes	no	no	?

Ảnh: dữ liệu cần dự đoán

Kết quả dự đoán

```
=== Predictions on user test set ===

inst#      actual  predicted error prediction (animal-name)
  1         1:?    1:Mammal      1 (NameIsSecret)
  2         1:?    2:Bird      1 (NameIsSecret)
  3         1:?    3:Reptile     1 (NameIsSecret)
  4         1:?    4:Fish      1 (NameIsSecret)
  5         1:?    5:Amphibian    1 (NameIsSecret)
```

Nhận xét:

Sau khi bỏ đi sự ảnh hưởng của “animal name” trong mô hình, mô hình đã có các giá trị của độ đo tốt hơn rất nhiều so với mô hình cũ. Cụ thể, F1-score bằng 0.981 rất xấp xỉ với 1. Hơn nữa được độ chính xác 97% **cao hơn rất nhiều** so với mô hình cũ (24%) và baseline accuracy (40%)

X. Naïve Bayes – Zoo Data Set

Output của mô hình có thể xem tại `./weka/zoo/train_output/OUTPUT_NaiveBayes.txt` hoặc tại [github](#).

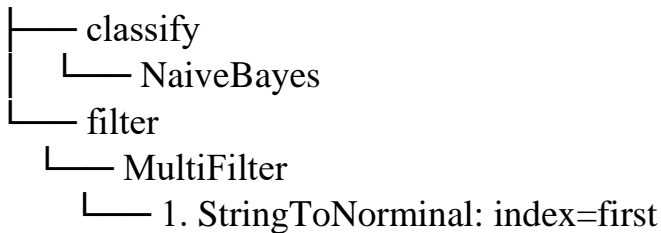
Naïve Bayes yêu cầu về kiểu dữ liệu của dataset ít khắt khe hơn với Id3, tuy nhiên, nó vẫn không chấp nhận kiểu dữ liệu string. Do đó, ta cần chuyển các giá trị “animal name” từ string thành nominal trước khi tiến hành phân lớp.

1. Classify

Ta sẽ áp dụng FilteredClassifier, để tiền xử lý và tạo mô hình phân lớp

FilteredClassifier được cài đặt như sau:

FilteredClassifier/



Không cần filter ReplaceMissingValue vì dữ liệu đã hợp lệ.

2. Trục quan quá trình tiền xử lý bằng tab preprocess

Dữ liệu sau khi tiền xử lý hoàn toàn giống với [D.III.2 Trục quan quá trình tiền xử lý bằng tab preprocess](#)

3. Áp dụng Weka classifier NaiveBayes

Mô hình của thuật toán được xuất ra file lưu trữ tại filteredClassifier-naiveBayes-multiFilter_stringToNominal-seed19126008.model

Một số cài đặt bổ sung cho quá trình phân lớp:

- Test option: vì có khá ít dữ liệu, nên ta sử dụng các chia dữ liệu 10-folds
- Seed: tất cả seed (nếu có) sử dụng cho bài toán sẽ là 19126008

Summary của mô hình

=== Summary ===			
Correctly Classified Instances	93	93	%
Incorrectly Classified Instances	7	7	%
Kappa statistic	0.9084		
Mean absolute error	0.0217		
Root mean squared error	0.1069		
Relative absolute error	9.8655	%	
Root relative squared error	32.3496	%	
Total Number of Instances	100		

Độ chính xác của mô hình: 93%

Độ đo

=== Detailed Accuracy By Class ===					
TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.975	0.000	1.000	0.975	0.987	Mammal
1.000	0.013	0.952	1.000	0.976	Bird
0.600	0.021	0.600	0.600	0.600	Reptile
1.000	0.023	0.867	1.000	0.929	Fish

0.750	0.000	1.000	0.750	0.857	Amphibian
1.000	0.022	0.800	1.000	0.889	Bug
0.700	0.000	1.000	0.700	0.824	Invertebrate
0.930	0.008	0.937	0.930	0.929	Weighted Avg.

Dự đoán dựa trên mô hình vừa tạo

	animal-name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	NameIsSecret	yes	no	no	yes	no	no	no	yes	
1	NameIsSecret	no	yes	yes	no	yes	no	no	no	
2	NameIsSecret	no	no	yes	no	no	no	yes	yes	
3	NameIsSecret	no	no	yes	no	no	yes	yes	yes	
4	NameIsSecret	no	no	yes	no	no	yes	yes	yes	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize	class
0	yes	yes	no	no	4	yes	no	yes	?
1	yes	yes	no	no	2	yes	yes	no	?
2	yes	yes	yes	no	0	yes	no	no	?
3	yes	no	no	yes	0	yes	no	no	?
4	yes	yes	no	no	4	yes	no	no	?

Ảnh: dữ liệu cần dự đoán

Kết quả dự đoán

=== Predictions on user test set ===				
inst#	actual	predicted	error	prediction (animal-name)
1	1:?	1:Mammal		1 (NameIsSecret)
2	1:?	2:Bird		1 (NameIsSecret)
3	1:?	3:Reptile	0.974	(NameIsSecret)
4	1:?	4:Fish	0.999	(NameIsSecret)
5	1:?	5:Amphibian	0.732	(NameIsSecret)

Nhận xét:

Thuật toán Naïve Bayes tạo mô hình dựa trên sự tham gia của tất cả các thuộc tính trong dataset. Do đó, thuộc tính “animal name” là duy nhất cho mỗi thuộc tính, dù có loại bỏ nó khỏi tập dữ liệu hay không cũng không ảnh hưởng đến mô hình.

Độ chính xác mà mô hình dự đoán rất tốt, F1-score bằng 0.929 cũng rất xấp xỉ với 1 (thấp hơn so với Id3 – No-Overfitting – 0.981)

XI. So sánh Id3 và Naïve Bayes

1. Một số qui ước:

- Id3 chỉ có thể xử lý các tập dữ liệu chỉ bao gồm thuộc tính nominal. Naïve Bayes lại có thể xử lý tập dữ liệu bao gồm các thuộc tính Nominal/Numeric.

- Do đó, dữ liệu trước khi đưa vào để tạo mô hình phân lớp sẽ được **xử lý tối thiểu** sao miễn sao có thể tiến hành quá trình phân lớp.
- Các tập dữ liệu đều không mang giá trị rỗng và đều lấy từ một không gian quần thể vô hạn. Bài toán tương tự poker hand¹⁹, balance scale²⁰ sẽ không được lựa chọn.

2. So sánh chung

Tiêu chí	Id3	Naïve Bayes
Số thuộc tính dùng trong mô hình phân loại	Một số	Tất cả
Kiểu dữ liệu của thuộc tính	Norminal	Norminal, Numeric, Date
Kiểu dữ liệu của lớp	Norminal	Norminal
Khả năng bị overfitting	Có	Không
Giá trị rỗng trong dataset	Không	Có
Khả năng giải thích	Cao	Thấp

3. Dựa trên Zoo Data Set

Ta sẽ không đưa mô hình Id3 bị overfitting ([D.VIII](#)) vào phần so sánh.

Mô tả tập dữ liệu

- Có 101 đối tượng, có 17 thuộc tính và 7 phân lớp
- Toàn bộ các thuộc tính đều mang giá trị nhị phân (trừ animal names là id)
- Không mang giá trị rỗng

Kết quả thực nghiệm

Tiêu chí	Id3	Naïve Bayes
Accuracy	0.97	0.94
F1 score	0.981	0.929

Nhận xét

Đối với Zoo Data Set, **Id3 tốt hơn Naïve Bayes** mặc dù cả 2 thuật toán đều đưa ra một mô hình rất tốt F1-score > 0.9

4. Dựa trên Statlog (Heart) Data Set

Dự đoán có (hoặc không) bệnh tim

Mô tả tập dữ liệu

- Có 270 đối tượng, có 13 thuộc tính và 2 phân lớp

¹⁹ Poker hand: Tập dữ liệu có thể lấy tại <https://archive.ics.uci.edu/ml/datasets/Poker+Hand>

²⁰ Balance scale: Tập dữ liệu có thể lấy tại <https://archive.ics.uci.edu/ml/datasets/Balance+Scale>

- Kiểu dữ liệu của thuộc tính: real (5 thuộc tính), ordered (1 thuộc tính), boolean (3 thuộc tính), nominal (3 thuộc tính)

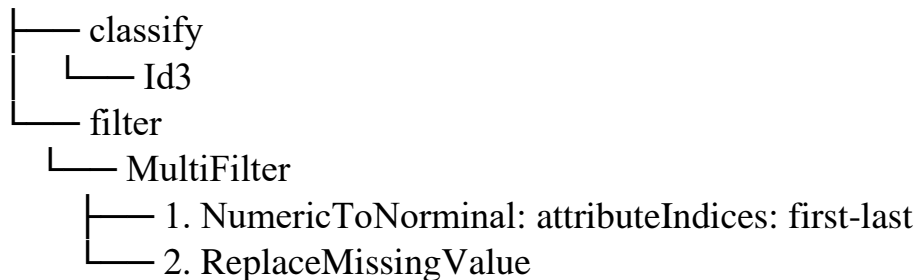
Cài đặt cho từng thuật toán

Test option:

- 10-folds
- Seed: 19126008

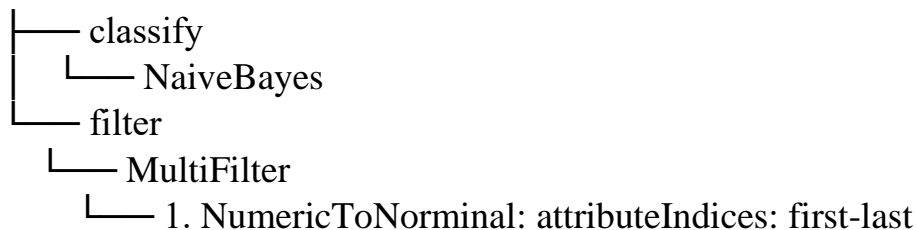
Id3:

FilteredClassifier



Naïve Bayes (toàn bộ các thuộc tính nominal):

FilteredClassifier



Naïve Bayes (các thuộc tính là numeric):

NaiveBayes

Kết quả thực nghiệm

Tiêu chí	Id3	Naïve Bayes Nominal	Naïve Bayes Numeric
Accuracy	35.5556 %	81.4815%	82.963%
F1 score	0.651	0.814	0.829

Nhận xét

Id3 đã có hiện tượng bias đối với những thuộc tính có kiểu dữ liệu numeric (đã được đổi thành nominal trong quá trình tiền xử lý) gây ra hiện tượng overfitting.

Naïve Bayes (F1-score=0.814) cho kết quả tốt hơn hẳn Id3 (F1-score=0.651). Việc giá trị thuộc tính là nominal hay numeric ảnh hưởng rất ít đến việc tạo mô hình phân lớp (0.814 với 0.829)

5. Kết luận:

Id3 tạo ra một mô hình phân lớp tốt hơn Naïve Bayes nếu các giá trị của thuộc tính là **nominal (Category)**

Naïve Bayes mang lại mô hình phân lớp tốt hơn Id3 nếu trong tập dữ liệu có chứa các thuộc tính kiểu **numeric/real**.

Id3 có thể bị overfitting do dữ liệu đầu vào kiểu numeric hay là id.

Naïve Bayes yêu cầu về việc tiền xử lí dữ liệu thấp hơn rất nhiều so với Id3.

Nên lựa chọn Id3 cho các mô hình cần giải thích cao và có tập dữ liệu dạng nominal.

Hiện nay, thay vì sử dụng Id3, ta có thể sử dụng J48²¹ mang lại hiệu quả tốt hơn Id3 rất nhiều, tránh được hiện tượng overfitting, có hỗ trợ cắt tỉa cây, mà có thể xử lí được trên tập dữ liệu có thuộc tính numeric.

²¹ J48: Trên Dataset Statlog (Heart) - numeric, J48 mang lại độ chính xác 78.9%, và F1-score = 0.789

E. References

- Antoniadis, P. (2022, November 6). *Decision Tree vs. Naive Bayes Classifier*. Được truy lục từ Baeldung: <https://www.baeldung.com/cs/decision-tree-vs-naive-bayes>
- Forsyth, R. (1990, 05 15). *Zoo Data Set*. Được truy lục từ UCI Machine Learning Repository: <http://archive.ics.uci.edu/ml/datasets/Zoo>
- Generated to model psychological experiments reported by Siegler, R. S.-5. (1994, 04 22). *Balance Scale Dataset*. Được truy lục từ UCI Machine Learning Repository: <https://archive.ics.uci.edu/ml/datasets/Balance+Scale>
- N/A. *Statlog (Heart) Dataset*. Được truy lục từ UCI Machine Learning Repository: <https://archive.ics.uci.edu/ml/datasets/Statlog+%28Heart%29>
- Tavish Srivastava — Published On August 6, 2019, 8 6). *Analytics Vidhya*. Được truy lục từ 11 Important Model Evaluation Metrics for Machine Learning Everyone should know: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/?utm_source=blog&utm_medium=decision-tree-weka-no-coding
- UCI Machine Learning. (2022, 12 22). *Zoo Animal Classification / Kaggle*. Được truy lục từ Kaggle: <https://www.kaggle.com/datasets/uciml/zoo-animal-classification>