Trường : Đại học Khoa học Tự nhiên Tp.HCM

20/12/2018

Khoa : Công nghệ thông tin

Môn : Cơ sở trí tuệ nhân tạo

GVLT : Thầy Lê Hoài Bắc

12/12/2018

Machine Learning

Tìm hiểu công cụ Weka và trải nghiệm các chức năng

Sinh viên:

ĐOÀN QUANG TUẤN 1612780

LÊ HOÀNG SANG 1612554

Mục lục

**[1.](#_Toc533080862)****[Mở đầu](#_Toc533080862)** [2](#_Toc533080862)

[1.1 Giới thiệu đồ án. 2](#_Toc533080863)

[1.2 Giới thiệu nhóm. 2](#_Toc533080864)

[1.3 Phân công công việc 2](#_Toc533080865)

[**2.** **Tìm hiểu về công cụ Weka** 2](#_Toc533080866)

[2.1. Giới thiệu về Weka 2](#_Toc533080867)

[2.2. Các chức năng và cách sử dụng 3](#_Toc533080868)

[1.1.1. Môi trường Explorer 3](#_Toc533080869)

[1.1.2. Môi trường Experimenter 13](#_Toc533080870)

[1.1.3. Môi trường KnowledgeFlow 15](#_Toc533080871)

[1.1.4. Môi trường Workbench 16](#_Toc533080872)

[1.1.5. Môi trường Simple CLI 16](#_Toc533080873)

[**2.** **Sử dụng Weka để chạy thuật toán ID3** 17](#_Toc533080874)

[2.1. Tạo tập tin Zoo.arff chứa dữ liệu Zoo 17](#_Toc533080875)

[2.2. Mô tả tổng quát về dữ liệu Zoo 17](#_Toc533080876)

[2.3. Sử dụng thuật toán ID3 để học ra cây quyết định từ dữ liệu Zoo 18](#_Toc533080877)

[2.4. Kết quả dự đoán của 5 mẫu đã cho 19](#_Toc533080878)

[**3.** **Chạy các thuật toán khác** 20](#_Toc533080879)

[3.1. Chương trình python cho giải thuật Naïve Bayes 20](#_Toc533080880)

[3.2. Dùng Weka để chạy thêm các thuật toán khác (Naïve Bayes, Localy weighted learning, Bagging) 21](#_Toc533080881)

[3.2.1. Naïve Bayes: 22](#_Toc533080882)

[3.2.2. Localy weighted learning (LWL): 24](#_Toc533080883)

[3.2.3. Bagging: 26](#_Toc533080884)

# **Mở đầu**

## Giới thiệu đồ án.

Tìm hiểu công cụ Weka và trải nghiệm các chức năng để chạy các thuật toán trong phần Máy Học.

## Giới thiệu nhóm.

Nhóm gồm 2 thành viên:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | MSSV | Tên | Email |
| 1 | 1612780 | Đoàn Quang Tuấn | 1612780@student.hcmus.edu.vn |
| 2 | 1612554 | Lê Hoàng Sang | 1612554@student.hcmus.edu.vn |

## Phân công công việc

|  |  |  |  |
| --- | --- | --- | --- |
| Yêu cầu | Công việc | Phụ trách | Mức độ hoàn thành |
| 2.1 | Tìm hiểu công cụ Weka | Lê Hoàng Sang | 100% |
| 2.2 | Sử dụng Weka để chạy thuật toán ID3 | Lê Hoàng Sang | 100% |
| 2.3.1 | Viết chương trình bằng python cho giải thuật Naïve Bayes | Đoàn Quang Tuấn | 100% |
| 2.3.2 | Sử dụng Weka chạy các thuật toán khác | Đoàn Quang Tuấn | 100% |
|  | Viết và chỉnh sửa báo cáo | Lê Hoàng Sang  Đoàn Quang Tuấn | 100% |

# **Tìm hiểu về công cụ Weka**

## Giới thiệu về Weka

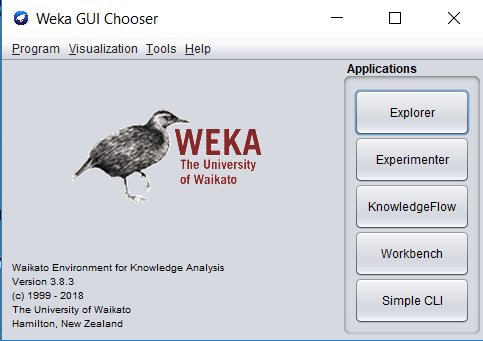
Weka (viết tắt của Waikato Environment for Knowledge Analysis) là một bộ phần mềm học máy được Đại học Waikato, New Zealand phát triển bằng Java. Weka là phần mềm tự do phát hành theo giấy phép công cộng GNU.

Mục đích nhằm xây dựng một công cụ hiện đại đẻ phát triển các kỹ thuật trong máy học và áp dụng chúng vào bài toán khai phá dữ liệu trong thực tế.

WEKA được xây dựng bằng ngôn ngữ Java, cấu trúc gồm hơn 600 lớp, tổ chức thành 10 packages.

Các chức năng chính của phần mềm:

* Khảo sát dữ liệu: tiền xử lí dữ liệu, phân lớp, gom nhóm dữ liệu, và khai thác luật kết hợp.
* Thực nghiệm mô hình: cung cấp phương tiện để kiểm chứng, đánh giá các mô hình học.
* Biểu diễn trực quan dữ liệu bằng nhiều dạng đồ thị khác nhau.

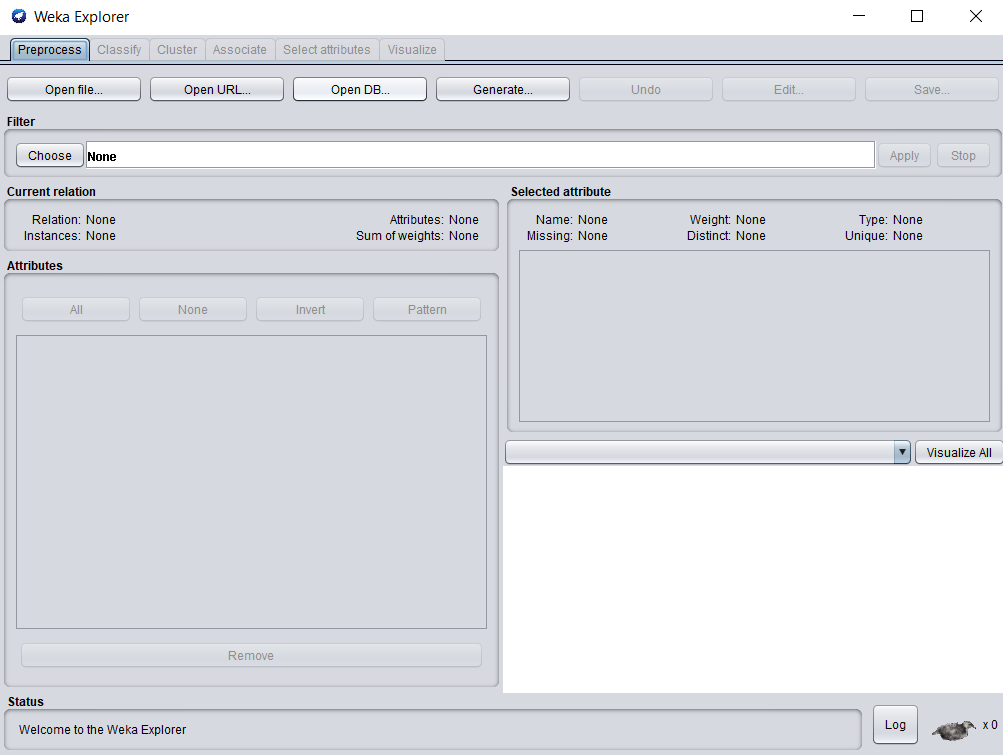


## Các chức năng và cách sử dụng

Phần mềm gồm 5 môi trường: Explorer, Experimenter, KnowledgeFlow, Workbench và Simple CLI. Báo cáo tập trung vào môi trường Explorer.

### Môi trường Explorer

Giao diện môi trường Explorer gồm: preprocess, classify, cluster, associate, select attributes, visualize.



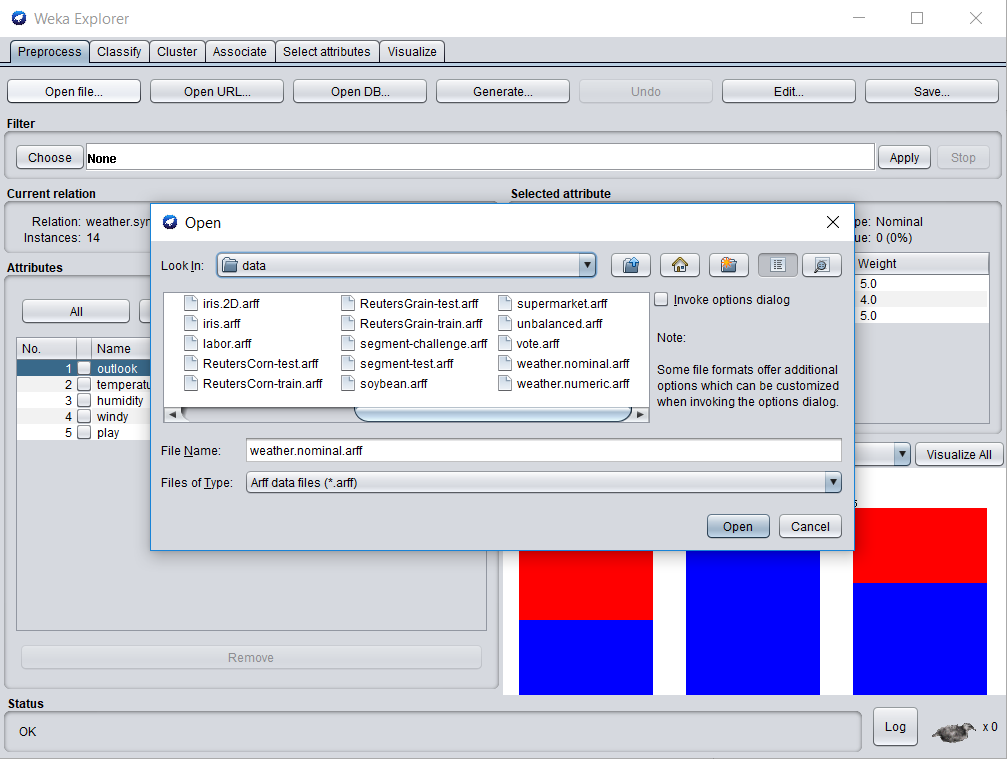
#### Preprocess – bước tiền xử lý dữ liệu

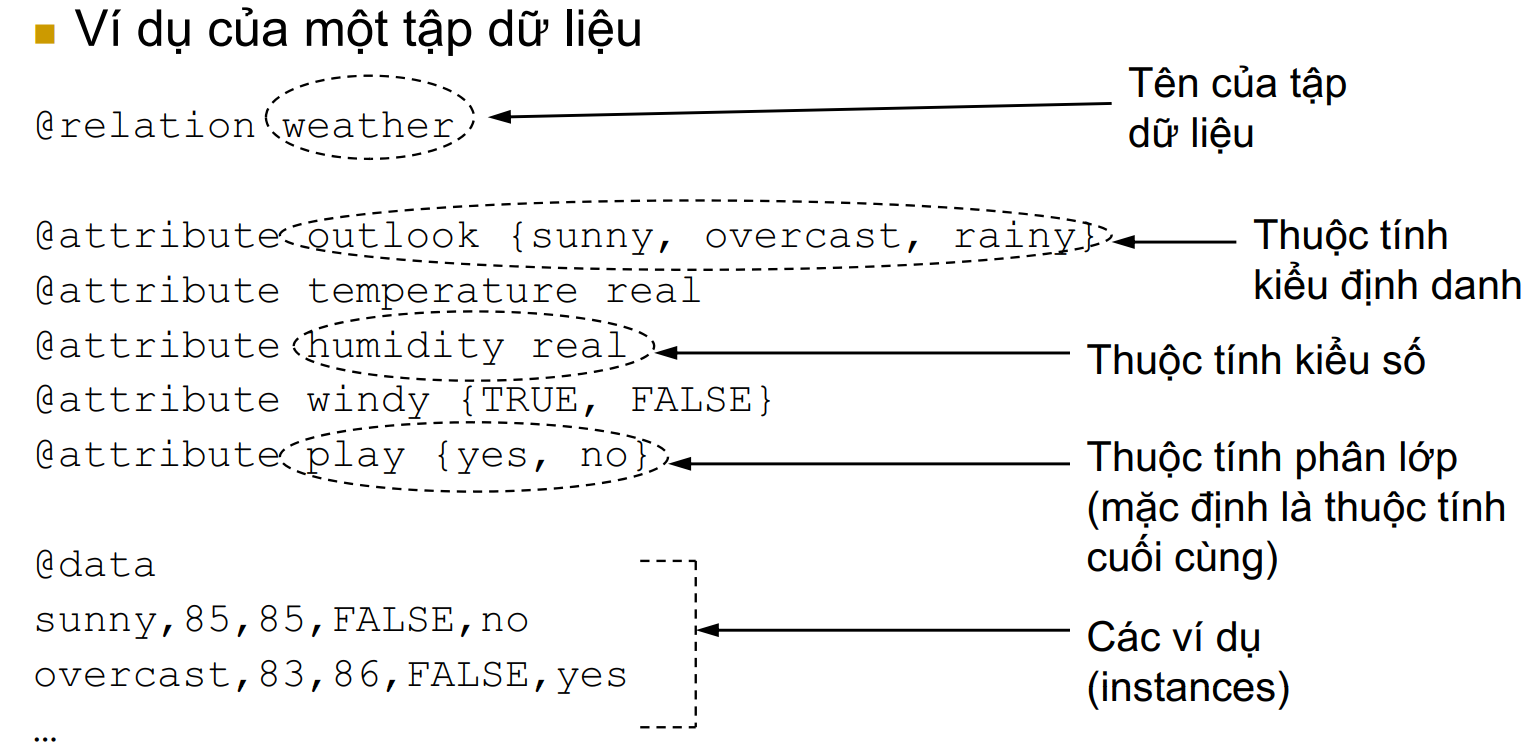
Trong qui trình khai phá dữ liệu, công việc xử lý dữ liệu trước khi đưa vào các mô hình là rất cần thiết, bước này làm cho dữ liệu có được ban đầu qua thu thập dữ liệu (gọi là dữ liệu gốc - ordinal data) có thể áp dụng được (thích hợp) với các mô hình khai phá dữ liệu (data mining model) cụ thể. Các công việc cụ thể của tiền xử lý dữ liệu bao gồm những công việc như:

* Filtering Attributes: chọn các thuộc tính phù hợp với mô hình.
* Filtering samples: lọc các mẫu (instances, patterns) dữ liệu cho mô hình.
* Clean data: làm sạch dữ liệu như xóa bỏ các dữ liệu bất thường (Outlier).
* Transformation: chuyển đổi dữ liệu cho phù hợp với các mô hình như chuyển đổi dữ liệu từ numeric qua nomial hay ordinal.
* Discetization (rời rạc hóa dữ liệu): nếu bạn có dữ liệu liên tục nhưng một vài mô hình chỉ áp dụng cho các dữ liệu rời rạc (luật kết hợp chẳng hạn) thì bạn phải thực hiện việc rời rạc hóa dữ liệu.

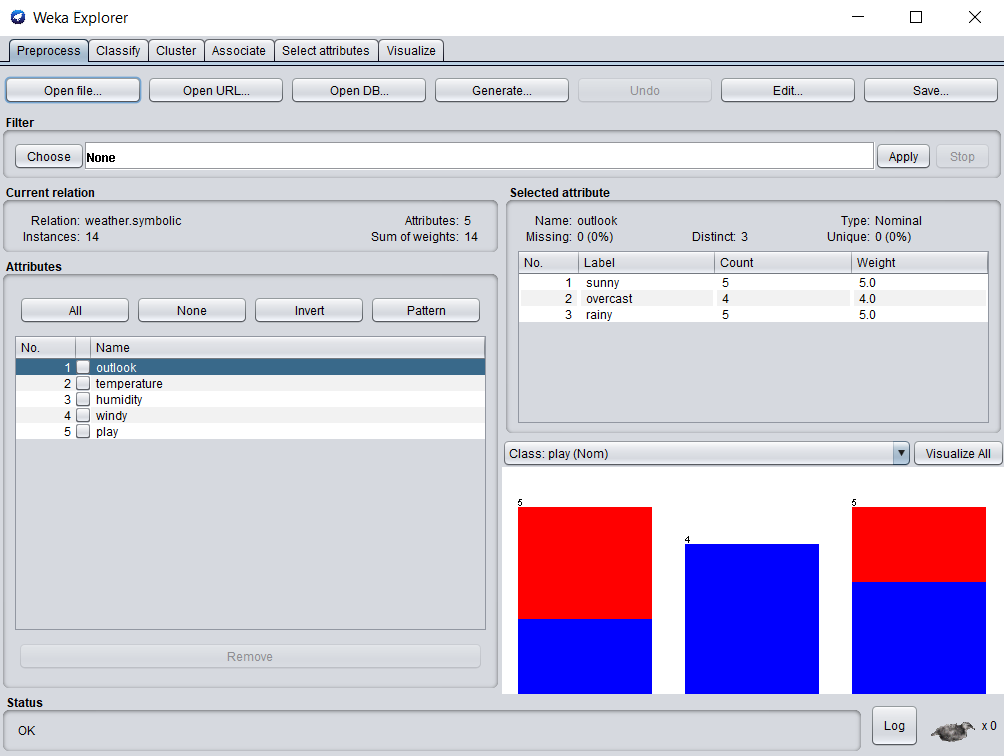
Dữ liệu có thể được nhập vào từ một tập tin có khuôn dạng ARFF, CSV. Dữ liệu có thể được đọc từ một địa chỉ URL, hoặc từ một cơ sở dữ liệu thông qua JDBC.

Chọn kiểu open ứng với nguồn dữ liệu để import một bộ dữ liệu vào weka.





Sau khi mở tập dữ liệu ta có giao diện như sau:

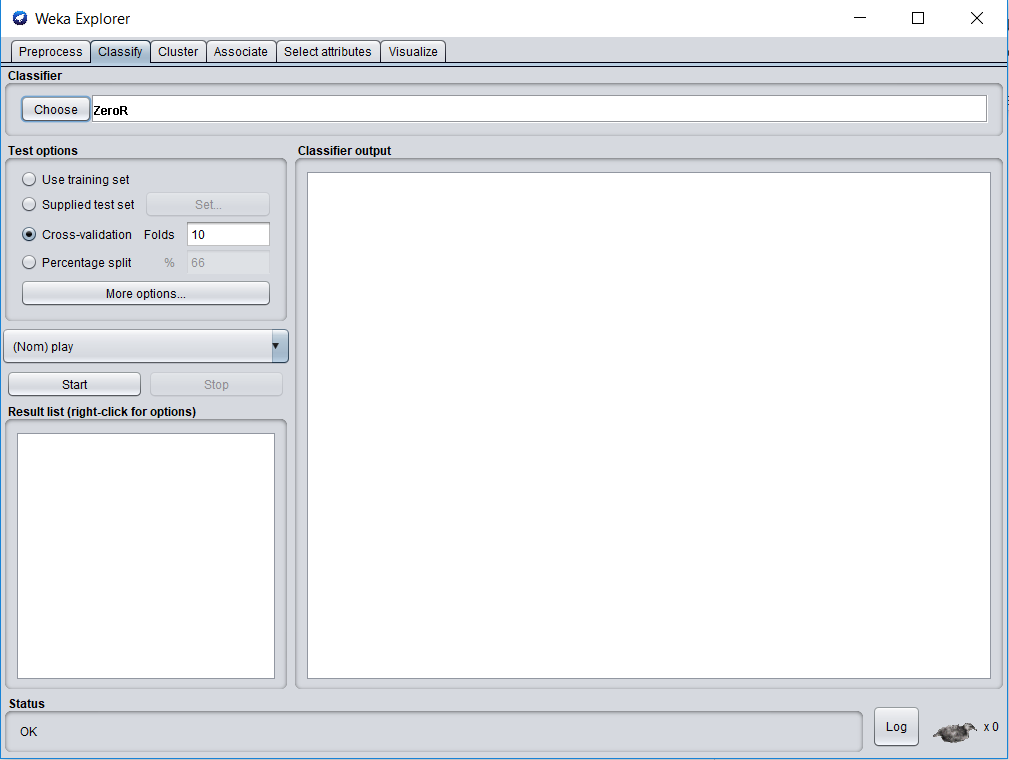


Các phần chính trong giao diện Preprocess:

* Phần **Filter** thiết lập kiểu lọc (choose) và apply.
* Phần **Current** relation nêu các mô tả cơ bản của tập dữ liệu.
* Phần **Attributes** chứa các thuộc tính của đối tượng trong dataset.
* Phần **Selected attribute** cho biết một số đặc tính cơ bản của thuộc tính đang được chọn.
* Phần **biểu đồ** mô tả phân bố của thuộc tính đang chọn. Mỗi màu trên biểu đồ đại diện cho 1 class.

#### Classify – phân lớp dữ liệu

Giao diện **Classify**:



Các bộ phân lớp (Classifiers) của Weka tương ứng với các mô hình dự đoán các đại lượng kiểu định danh (phân lớp) hoặc các đại lượng kiểu số (hồi quy/dự đoán).

Các kỹ thuật phân lớp được hỗ trợ bởi Weka:

* Naïve Bayes classifier and Bayesian networks.
* Decision trees.
* Instance-based classifiers.
* Support vector machines.
* Neural networks.
* …

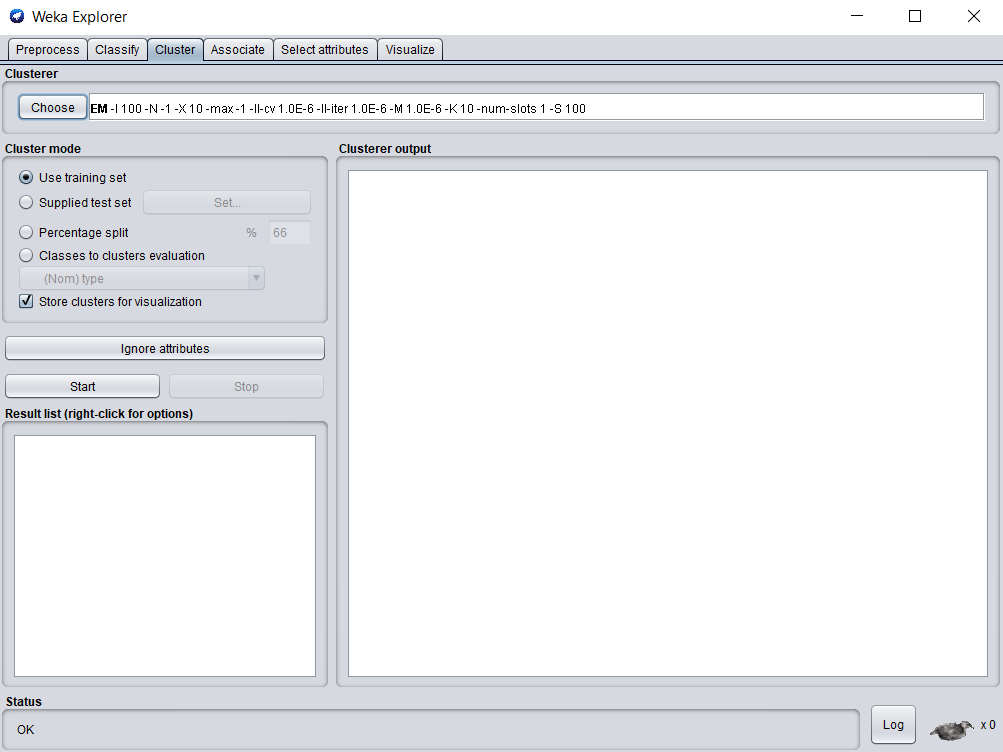
Mô tả về giao diện **Classify**:

* **Classifiers:** để chọn bộ phân lớp (gồm các bộ ở trên).
* **Test options:** lựa chọn các tùy chọn cho việc kiểm tra:
* **Use training set**: bộ phân loại học được sẽ được đánh giá trên tập học (training).
* **Supplied test set**: sử dụng một tập dữ liệu khác (với tập học) để cho việc đánh giá.
* **Cross-validation**: tập dữ liệu sẽ được chia đều thành k tập (folds) có kích thước xấp xỉ nhau, và bộ phân loại học được sẽ được đánh giá bởi phương pháp *cross-validation.*
* **Percentage split**: chỉ định tỷ lệ phân chia tập dữ liệu đối với việc đánh giá.
* **More options…:**
* **Output model:** hiển thị bộ phân lớp học được
* **Output per-class stats:** hiển thị các thông tin thống kê về precision/recall đối với mỗi lớp
* **Output entropy evaluation measures:** hiển thị đánh giá độ hỗn tạp (entropy) của tập dữ liệu.
* **Output confusion matrix:** hiển thị thông tin về ma trận lỗi phân lớp (confusion matrix) đối với phân lớp học được.
* **Store predictions for visualization:** các dự đoán của bộ phân lớp được lưu lại trong bộ nhớ, để có thể được hiển thị sau đó.
* **Output predictions:** hiển thị chi tiết các dự đoán đối với tập kiểm tra
* **Cost-sensitive evaluation:** các lỗi (của bộ phân lớp) được xác định dựa trên ma trận chi phí (cost matrix) chỉ định.
* **Random seed for XVal / % Split:** chỉ định giá trị random seed được sử dụng cho quá trình lựa chọn ngẫu nhiên các ví dụ cho tập kiểm tra.
* **Classifier output:** hiển thị các thông tin quan trọng
* **Run information**: các tùy chọn đối với mô hình học tên của tập dữ liệu, số lượng các ví dụ, các thuộc tính, và phương pháp thí nghiệm.
* **Classifier model (full training set)**: biểu diễn (dạng text) của bộ phân lớp học được.
* **Predictions on test data**: thông tin chi tiết về các dự đoán của bộ phân lớp đối với tập kiểm tra.
* **Summary**: các thống kê về mức độ chính xác của bộ phân lớp đối với phương pháp thí nghiệm đã chọn.
* **Detailed Accuracy By Class**: thông tin chi tiết về mức độ chính xác của bộ phân lớp đối với mỗi lớp.
* **Confusion Matrix**: các thành phần của ma trận này thể hiện số lượng các ví dụ kiểm tra (test instances) được phân lớp đúng và bị phân lớp sai.
* **Result list:** cung cấp một số chức năng hữu ích:
* **Save model**: lưu lại mô hình tương ứng với bộ phân lớp học được vào trong một tập tin nhị phân (binary file).
* **Load model**: đọc lại một mô hình đã được học trước đó từ một tập tin nhị phân.
* **Re-evaluate model on current test set**: đánh giá một mô hình (bộ phân lớp) học được trước đó đối với tập kiểm tra (test set) hiện tại.
* **Visualize classifier errors**: hiển thị cửa sổ biểu đồ thể hiện các kết quả của việc phân lớp.Các ví dụ được phân lớp chính xác sẽ được biểu diễn bằng ký hiệu bởi dấu chéo (x), còn các ví dụ bị phân lớp sai sẽ được biểu diễn bằng ký hiệu ô vuông.
* **…**

#### Cluster – phân cụm dữ liệu

Các bộ phân cụm (Cluster builders) của Weka tương ứng với các mô hình tìm các nhóm của các ví dụ tương tự đối với một tập dữ liệu.

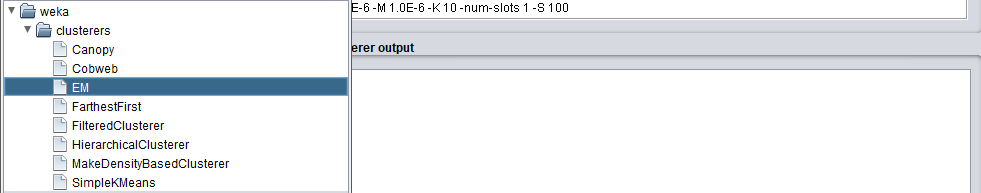
Giao diện **cluster**:



Mô tả giao diện **Cluster**:

* **Clusterers:** lựa chọn các kỹ thuật phân cụm.

Các kỹ thuật phân cụm hỗ trợ trong Weka:

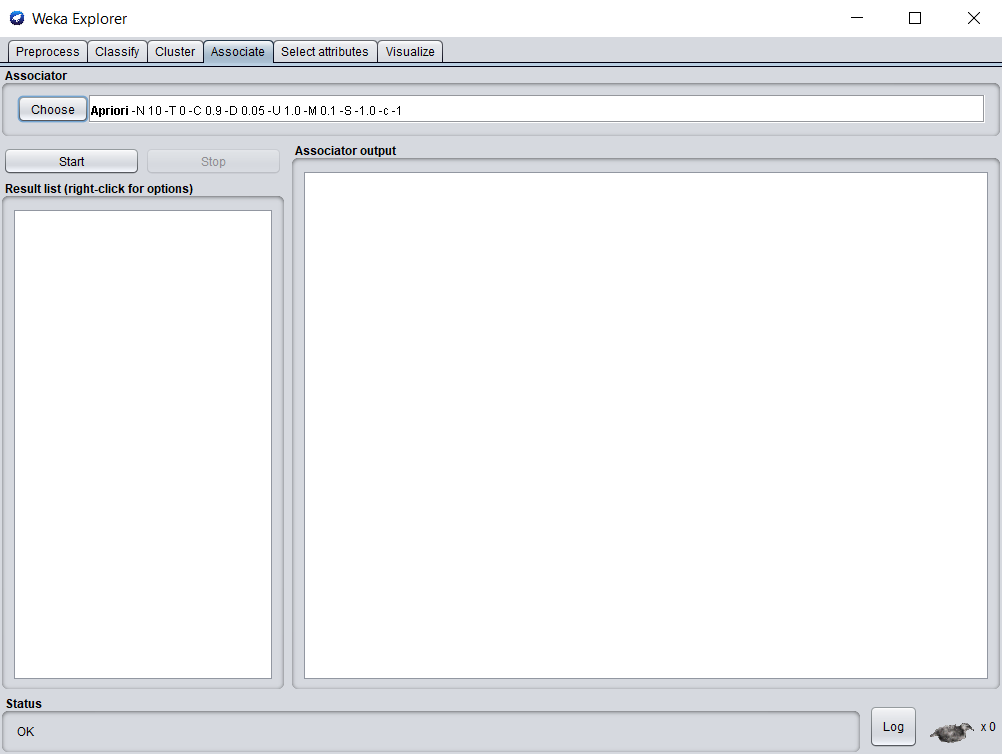


* **Cluster mode:** lựa chọn chế độ phân cụm.
* **Use training set:** các cụm học được sẽ được kiểm tra đối với tập học.
* **Supplied test set:** sử dụng một tập dữ liệu khác để kiểm tra các cụm học được.
* **Percentage split:** chỉ định tỷ lệ phân chia tập dữ liệu ban đầu cho việc xây dựng tập kiểm tra.
* **Classes to clusters evaluation:** so sánh độ chính xác của các cụm học được đối với các lớp được chỉ định.

Sau khi thiết lập các kỹ thuật phân cụm cũng như mode ta nhấn **start** để bắt đầu phân cụm.

#### Associate – luật kết hợp

Giao diện **associate:**

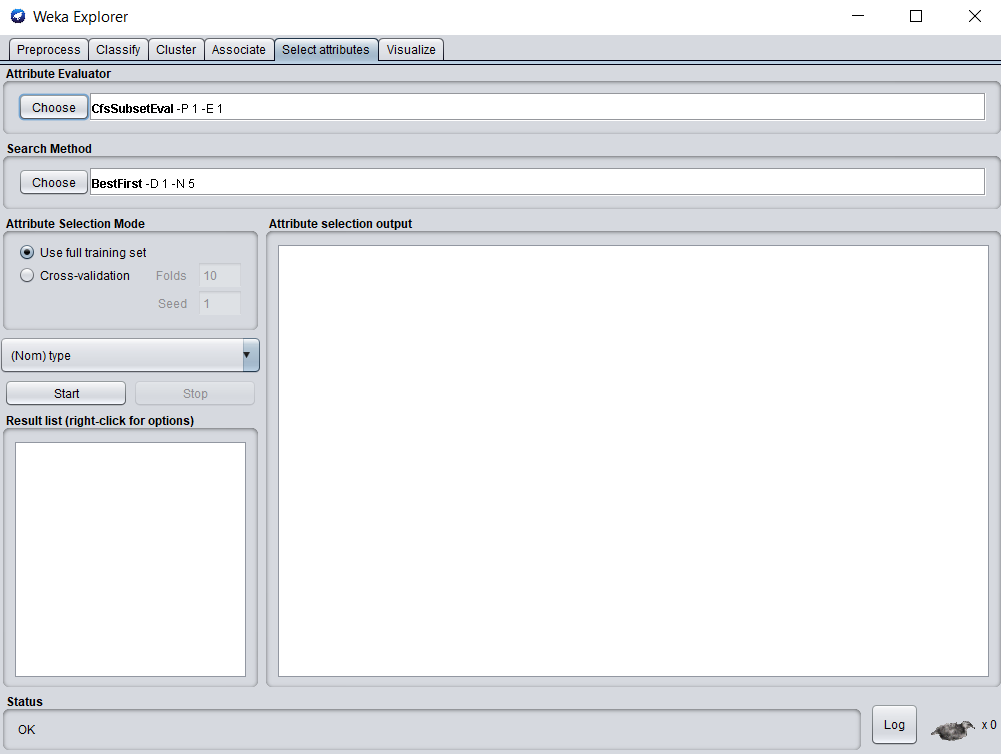


Mô tả giao diện **Associate**:

* **Associator**: lựa chọn một mô hình (giải thuật) phát hiện luật kết hợp.
* **Associator ouput**: hiển thị các thông tin quan trọng:
* **Run information**: Các tùy chọn đối với mô hình phát hiện kết hợp, tên tập dữ liệu, thuộc tính, số lượng các ví dụ.
* **Assciator mode**: Biểu diễn (dạng text) của tập các luật phát hiện kết hợp.
* Độ hỗ trợ tối thiểu.
* Độ tin cậy tối thiểu.
* Liệt kê các luật kết hợp tìm được.

#### Select attributes – lựa chọn thuộc tính

Giao diện **Select attributes**:



Mô tả giao diện **Select attributes:**

* **Attribute Evaluator**: Để xác định một phương pháp đánh giá mức độ phù hợp của các thuộc tính.

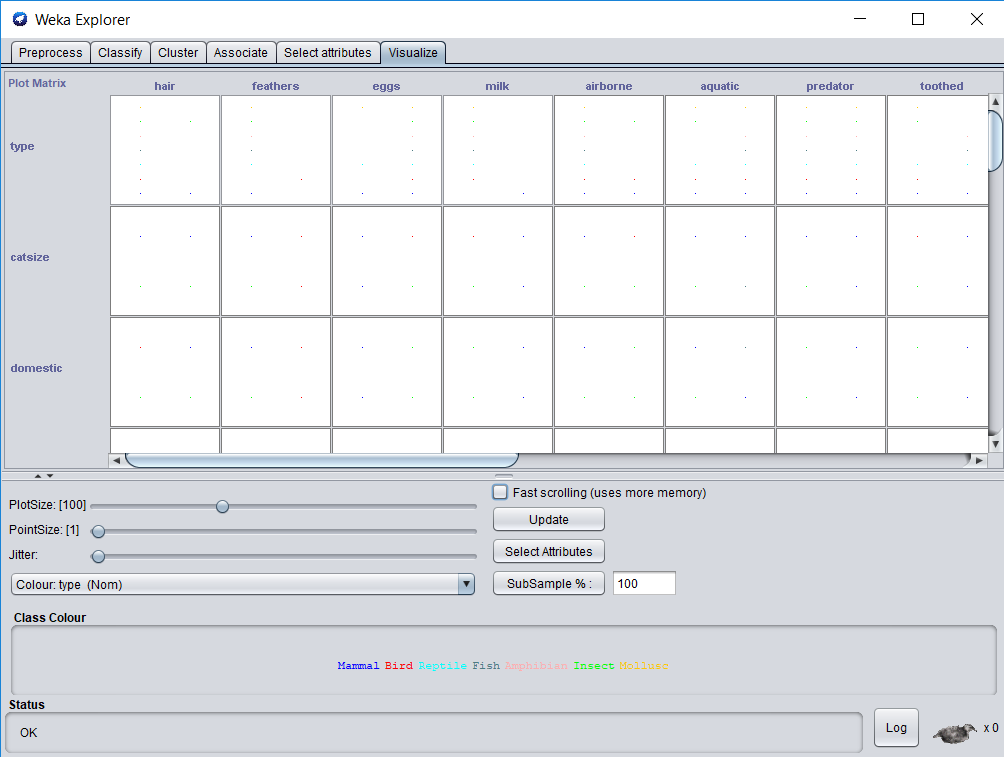
Ví dụ: correlation-based, wrapper, information gain, chisquared,…

* **Search Method**: Để xác định một phương pháp (thứ tự) xét các thuộc tính.

Ví dụ: best-first, random, exhaustive, ranking,…

#### Visualize – hiển thị dữ liệu

Visualize cung cấp các biểu đồ mô tả dữ liệu một cách trực quan.



Weka có thể hiển thị:

* Mỗi thuộc tính riêng lẻ (1-D visuallization)
* Một cặp thuộc tính (2-D visuallization)

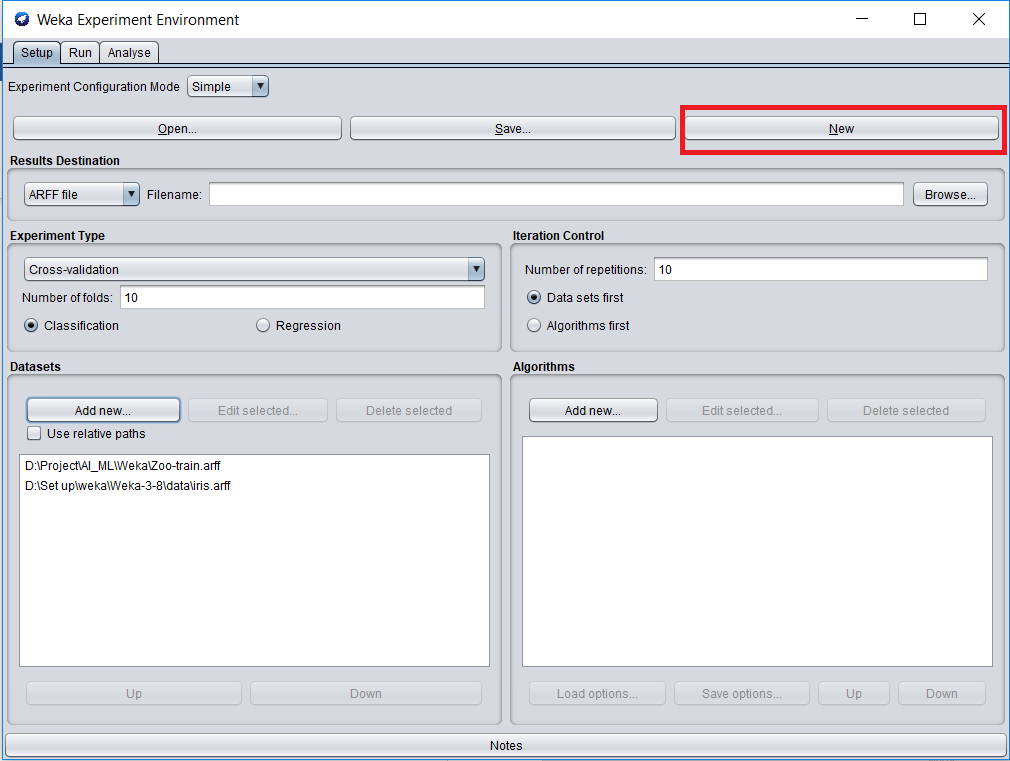
Các giá trị (các nhãn) lớp khác nhau sẽ được hiển thị bằng các màu khác nhau.

Thanh trượt **Jitter** hỗ trợ việc hiển thị rõ ràng hơn, khi có quá nhiều ví dụ (điểm) tập trung xung quanh một vị trí trên biểu đồ.

Tính năng phóng to/thu nhỏ (bằng cách tăng/giảm giá trị của **PlotSize** và **PointSize**).

### Môi trường Experimenter

Môi trường cho phép thực nghiệm (Setup, Run), so sánh, phân tích (Analyse) các mô hình học máy.

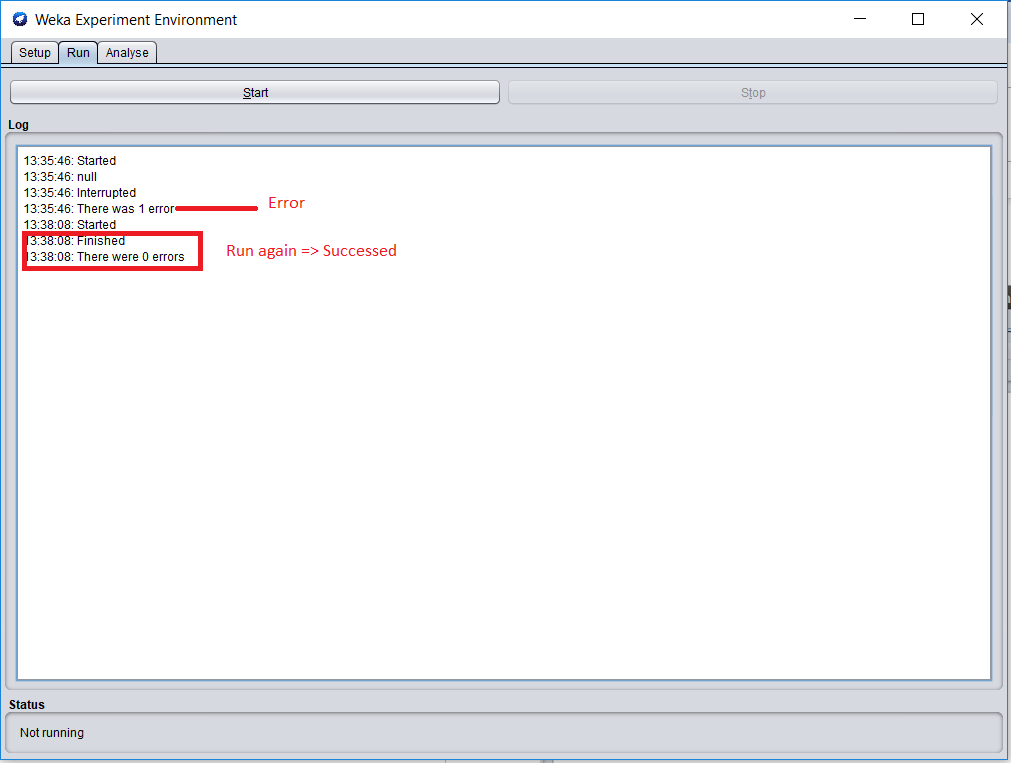
****

Tab **Set up:**

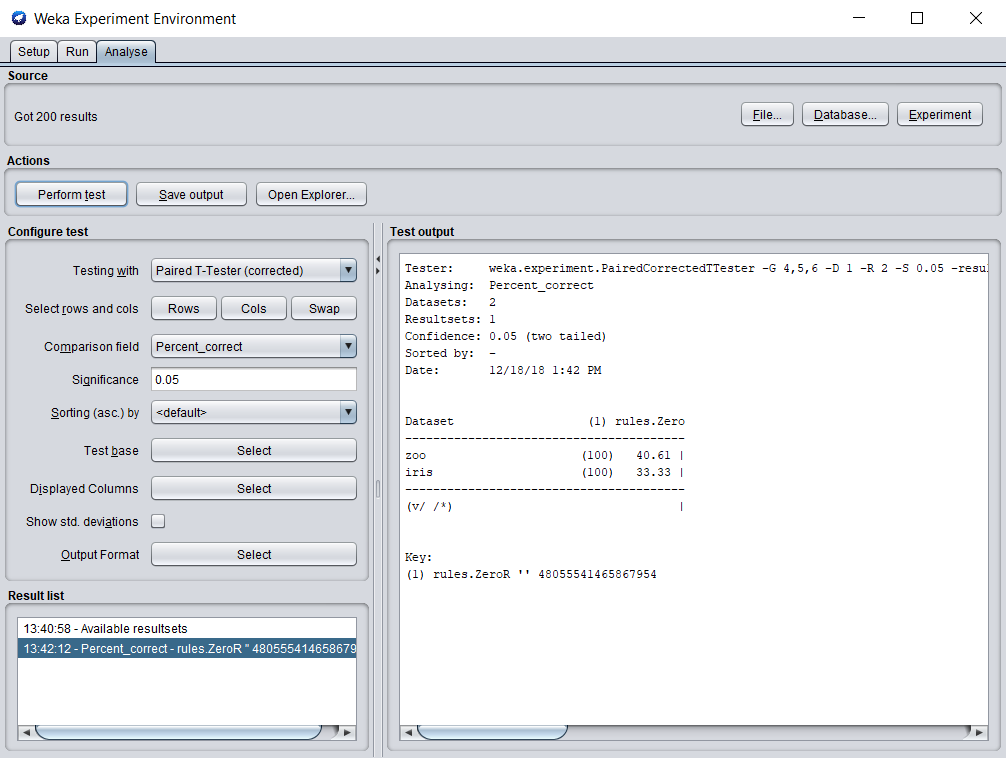
Nhấn **New** để tạo một thử nghiệm mới, hoặc có thể thử nghiệm đã lưu trước đó qua **Open**.

* **Result Destination**: lựa chọn format cũng như đường dẫn lưu file kết quả.
* **Experiment type**
* **Iteration Control**
* **Datasets**: thêm dữ liệu phục vụ hử nghiệm.
* **Algorithms**: lựa chọn thuật toán cho quá trình thử nghiệm.

Sau khi Set up ta chuyển qua tab **Run** để tiến hành chạy thử nghiệm. Nhấn Start để tiến hành chạy và xem log của chương trình đã chạy.



Nếu đã chạy thành công ta chuyển sang tab **Analyze**.

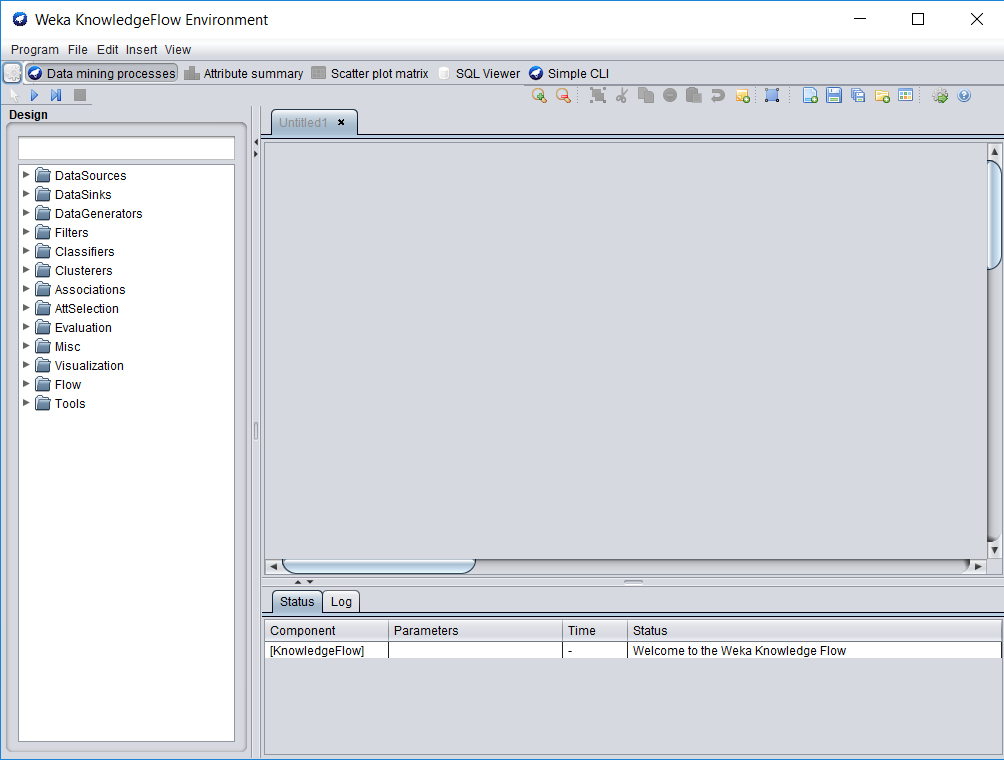


* **Configure test**: thiết lập tùy chọn muốn kiểm tra
* **Action**: nhấn Perform test để chạy theo config của mình.
* **Test ouput**: hiển thị kết quả.

### Môi trường KnowledgeFlow

Môi trường cho phép bạn tương tác đồ họa kiểu kéo/thả để thiết kế các bước (các thành phần) của một thí nghiệm.

Giao diện môi trường:

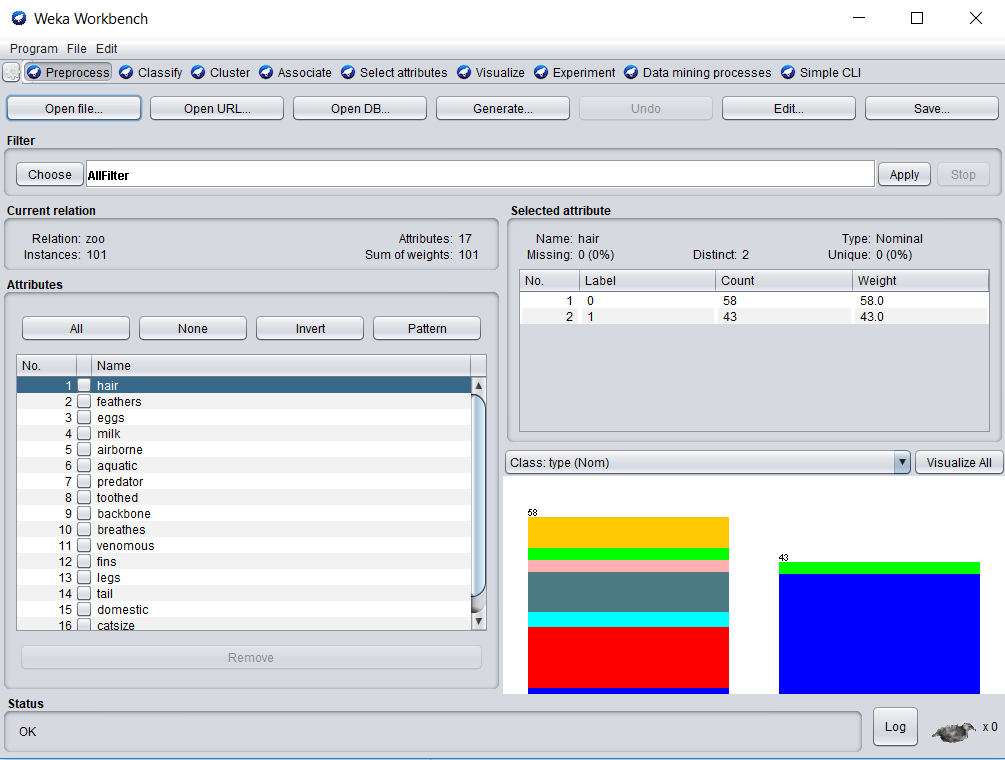
****

Chúng ta chỉ cần kéo các đối tượng, thuật toán từ khung **Design** và thả vào khung đang thiết kế.

Nhấn biểu tượng **start** để chạy chương trình.

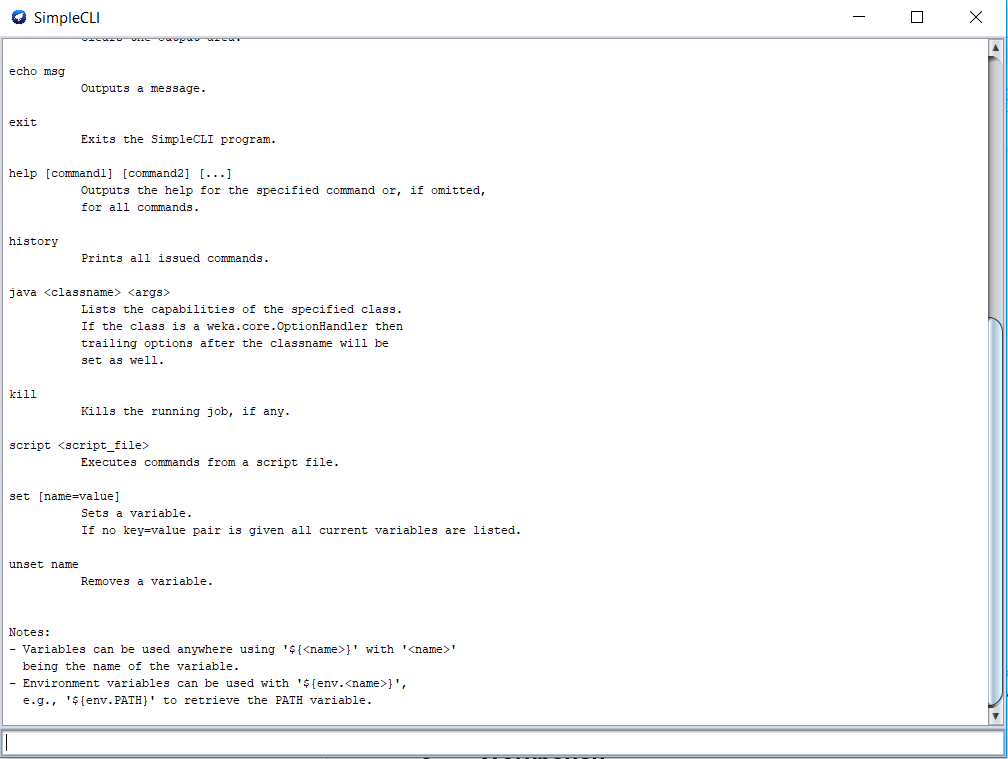
### Môi trường Workbench

Giao diện Workbench chỉ đơn giản là tổng hợp của tất cả các công cụ khác trong Explorer, Experimenter, Knowledge và Simple CLI.



### Môi trường Simple CLI

Giao diện đơn giản kiểu dòng lệnh.

****

# **Sử dụng Weka để chạy thuật toán ID3**

## Tạo tập tin Zoo.arff chứa dữ liệu Zoo

Xem file đính kèm (/source/Zoo.arff)

## Mô tả tổng quát về dữ liệu Zoo

Số mẫu trong tập dữ liệu là 101 loại động vật.

Tên và ý nghĩa các thuộc tính (18):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | Ý nghĩa | Miền giá trị |
| 1 | name | String | Tên động vật | Chuỗi |
| 2 | hair | Boolean | Tóc | 1,0 |
| 3 | feathers | Boolean | Lông | 1,0 |
| 4 | eggs | Boolean | Đẻ trứng | 1,0 |
| 5 | milk | Boolean | Có sữa | 1,0 |
| 6 | airbone | Boolean | Biết bay | 1,0 |
| 7 | aquatic | Boolean | Đông vật sống dưới nước | 1,0 |
| 8 | predator | Boolean | Động vật ăn thịt | 1,0 |
| 9 | toothed | Boolean | Có răng | 1,0 |
| 10 | backbone | Boolean | Có xương sống | 1,0 |
| 11 | breathes | Boolean | Có thở | 1,0 |
| 12 | venomous | Boolean | Có độc | 1,0 |
| 13 | fins | Boolean | Có vây | 1,0 |
| 14 | legs | Numeric | Số chân | 0,2,4,5,6,8 |
| 15 | tail | Boolean | Có đuôi | 1.0 |
| 16 | domestic | Boolean | Động vật được nuôi | 1,0 |
| 17 | type | Numeric | Lớp động vật | 1,2,3,4,5,6,7 |

Danh sách phân lớp, đặt tên cho mỗi phân lớp:

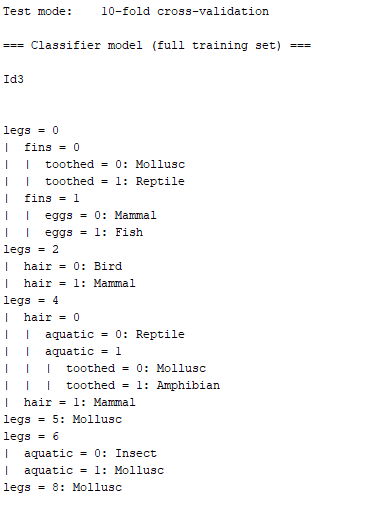
|  |  |  |  |
| --- | --- | --- | --- |
| Type | Đặt tên | Số lượng | Động vật thuộc lớp |
| 1 | Mammal | 44 | aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer, dolphin, elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, leopard, lion, lynx, mink, mole, mongoose, opossum, oryx, platypus, polecat, pony, porpoise, puma, pussycat, raccoon, reindeer, seal, sealion, squirrel, vampire, vole, wallaby,wolf |
| 2 | Bird | 20 | chicken, crow, dove, duck, flamingo, gull, hawk, kiwi, lark, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, vulture, wren. |
| 3 | Reptile | 5 | pitviper, seasnake, slowworm, tortoise, tuatara. |
| 4 | Fish | 13 | bass, carp, catfish, chub, dogfish, haddock, herring, pike, piranha, seahorse, sole, stingray, tuna. |
| 5 | Amphibian | 3 | frog, newt, toad |
| 6 | Insect | 8 | flea, gnat, honeybee, housefly, ladybird, moth, termite, wasp. |
| 7 | Mollusc | 10 | clam, crab, crayfish, lobster, octopus, scorpion, seawasp, slug, starfish, worm |

## Sử dụng thuật toán ID3 để học ra cây quyết định từ dữ liệu Zoo

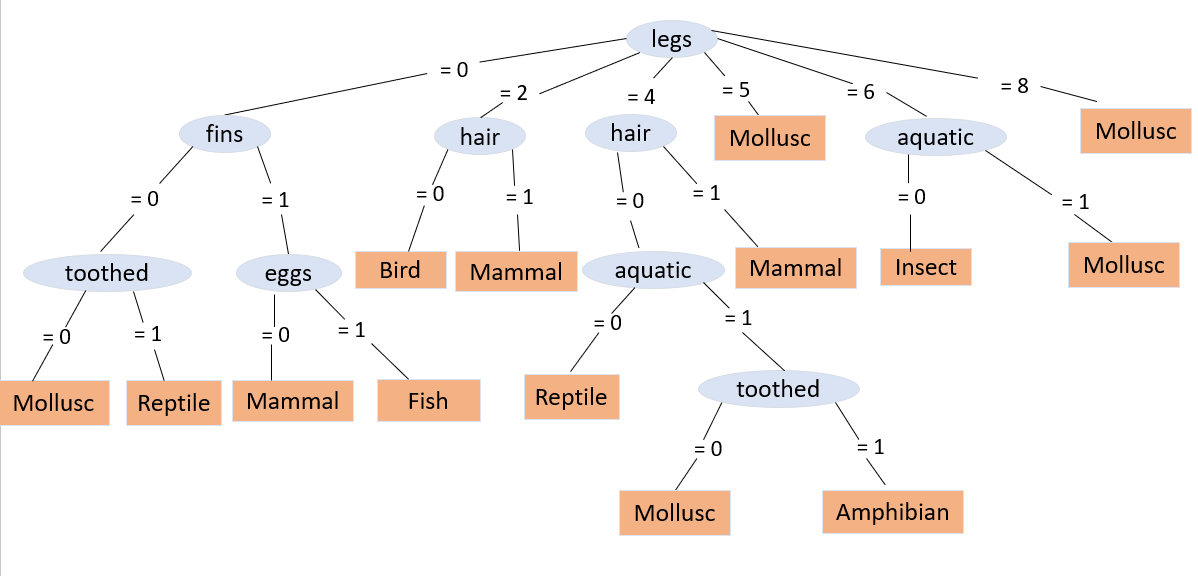
Sử dụng thuật toán ID3 cho thuộc tính type. Tên của động vật chúng ta không cần xét đến nên ta sử dụng bộ dữ liệu không có thuộc tính name: Zoo-train.arff

- Test option: 10-fold cross validation

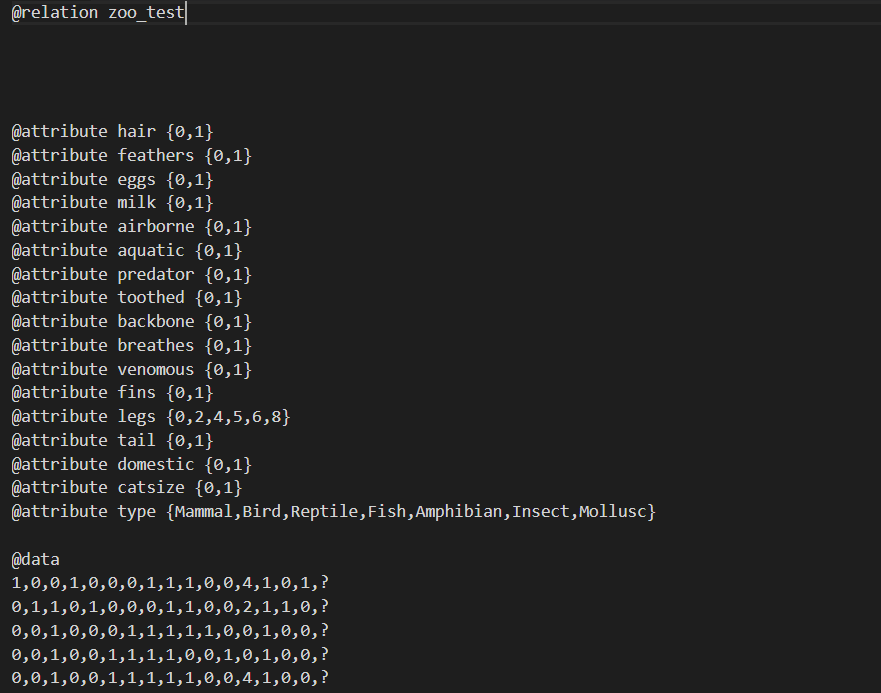
Ta được cây quyết định:

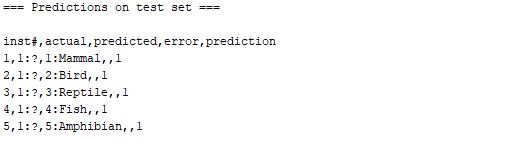


Cây được vẽ lại:



## Kết quả dự đoán của 5 mẫu đã cho





NameIsSecret,1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1, ? => Mamal

NameIsSecret,0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0, ? => Bird

NameIsSecret,0,0,1,0,0,0,1,1,1,1,1,0,0,1,0,0, ? => Reptile

NameIsSecret,0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0, ? => Fish

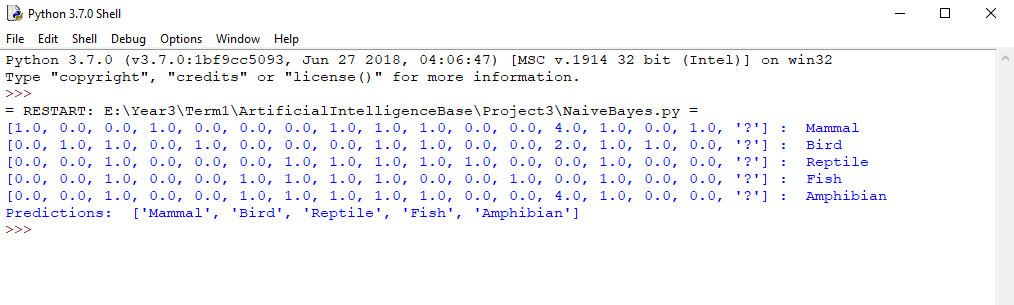
NameIsSecret,0,0,1,0,0,1,1,1,1,1,0,0,4,1,0,0, ? => Amphibian

# **Chạy các thuật toán khác**

## Chương trình python cho giải thuật Naïve Bayes

Tập tin chương trình: /source/NaiveBayes.py

Kết quả dự đoán 5 mẫu đã cho:



Hàm làm trơn:

**def** smooth(a, b):  
 **return** float((a+1)/(b+1))

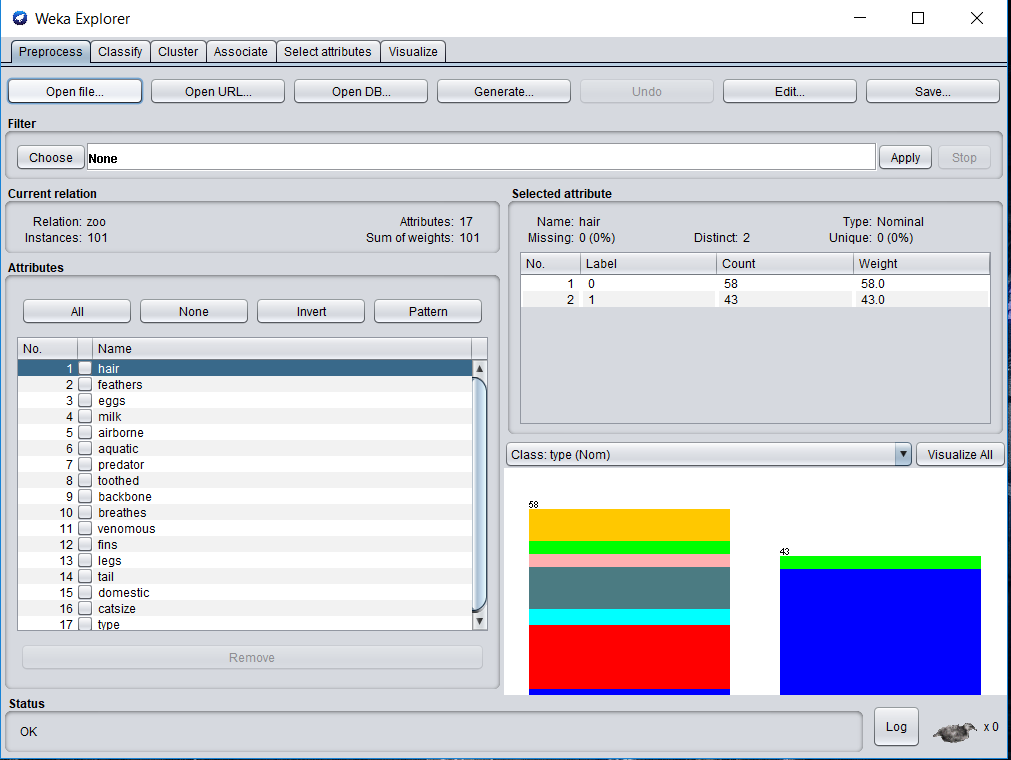
Mục đích là tránh cho việc tích một xác suất đang cao, bỗng nhiên gặp một xác suất điều kiện bằng 0, và kéo theo toàn bộ sẽ bằng 0 bất kể rất nhiều các điều kiện khác thế nào. Để giải quyết vấn đề này, ta sẽ áp dụng công thức sau:

P(xi​∣cj​)=(nc​+1)/(n+1)

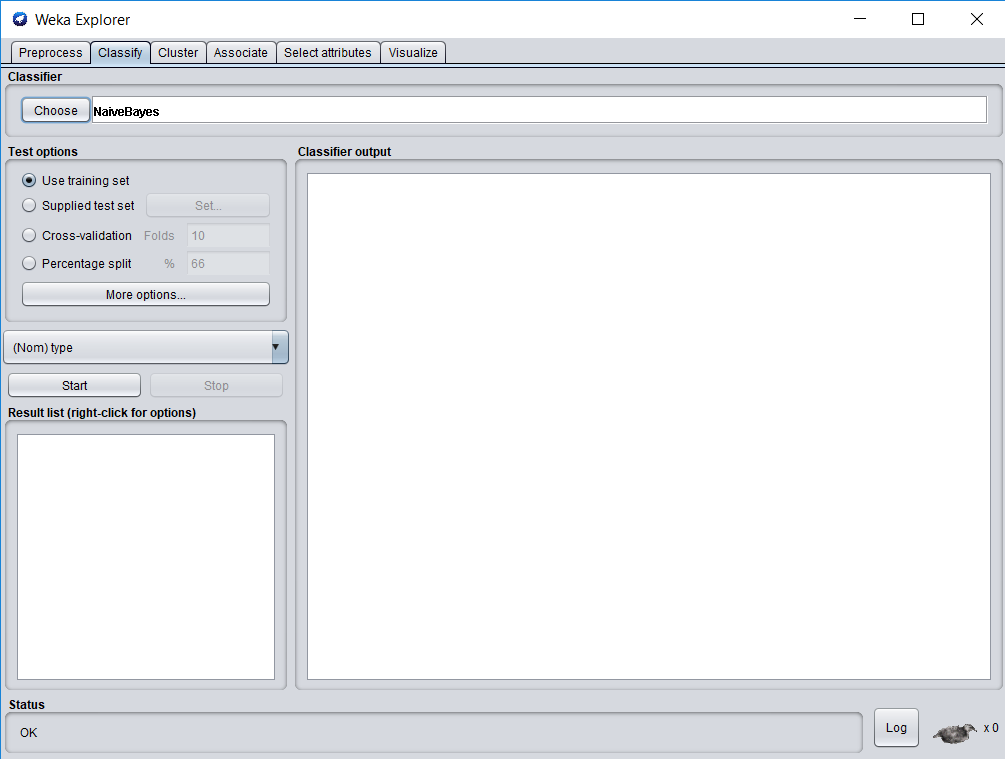
Việc cộng cả tử và mẫu với 1 sẽ giữ gần như nguyên vẹn và đảm bảo công bằng cho các xác suất, đồng thời tránh được việc một xác suất điều kiện bằng 0 kéo theo tích toàn bộ xác suất bằng 0.

## Dùng Weka để chạy thêm các thuật toán khác (Naïve Bayes, Localy weighted learning, Bagging)

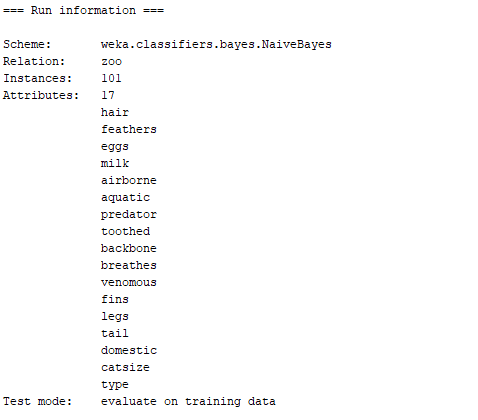
Vẫn sử dụng tập dữ liệu Zoo-train.arff (từ tập Zoo.arff nhưng loại bỏ thuộc tính name vì thuộc tính này không ảnh hưởng đến thí nghiệm).



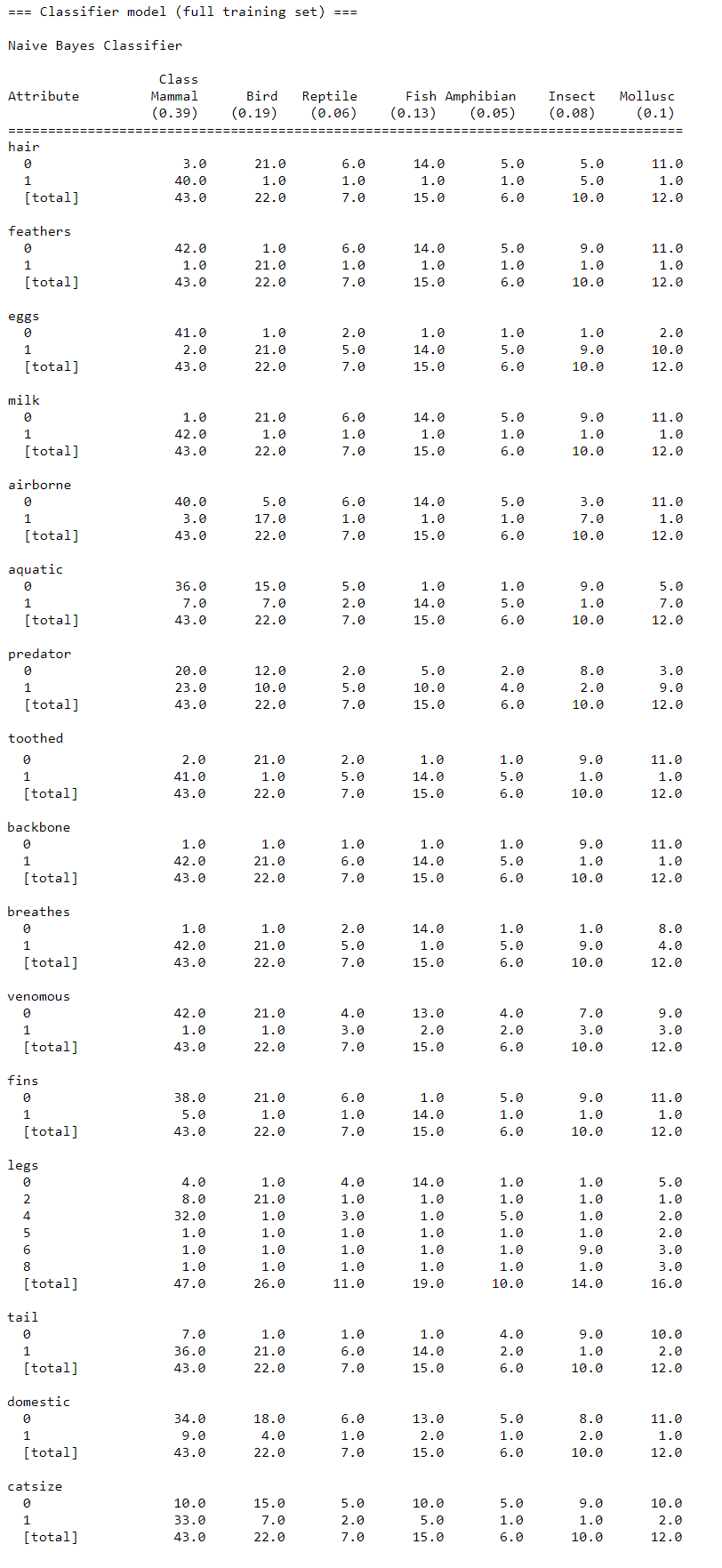
Ở **Test option** ta chọn dữ liệu test từ tập train (Use training set) và tiến hành chạy 3 thuật toán:



### Naïve Bayes:



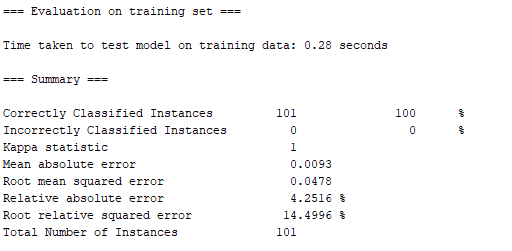
Thông tin chạy với thuật toán **Naïve Bayes** và test mode (use training set) như hình trên.



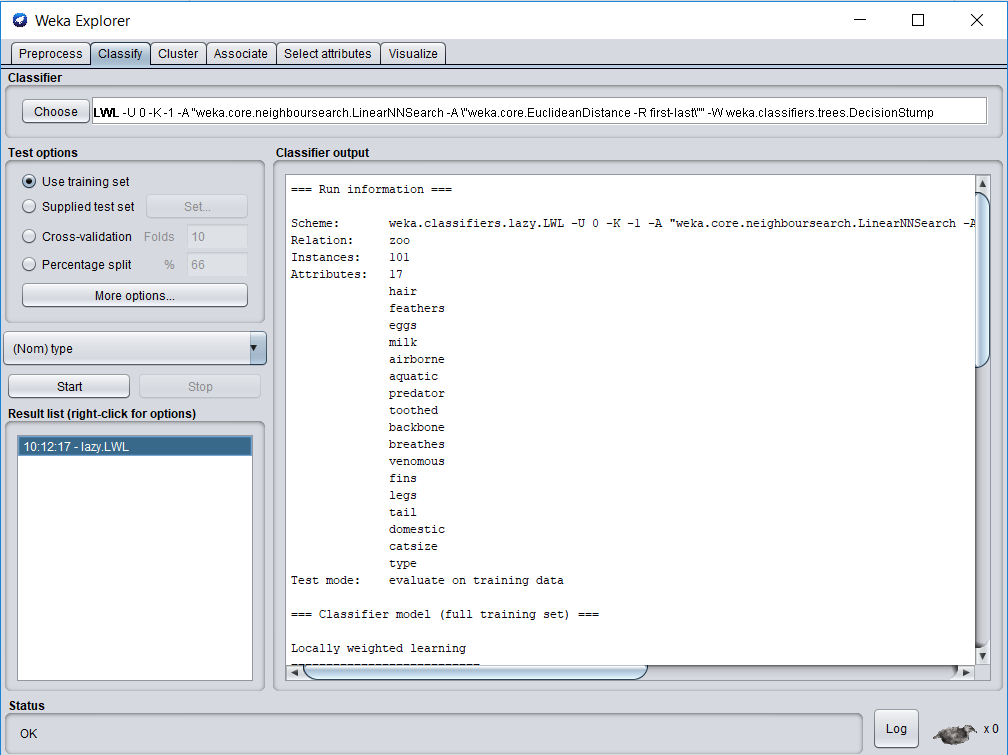
Kết quả tổng kết cho ta biết:

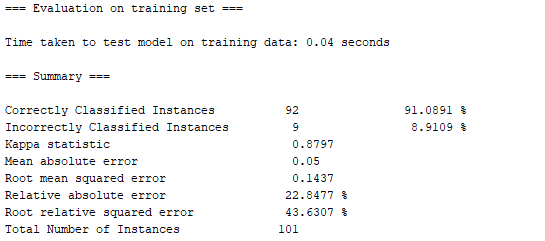
* Số mẫu phân lớp đúng là 101 tương ứng 100%
* Số mẫu phân lớp sai 0 tương ứng 0%
* Các giá trị về độ đo lỗi như hình.

=> Độ chính xác thuật toán Naïve Bayes trong trường hợp này là 100%, tuy nhiên với số lượng mẫu ít như vậy chúng ta chưa kết luận được thuật toán nào tốt hơn.



### Localy weighted learning (LWL):





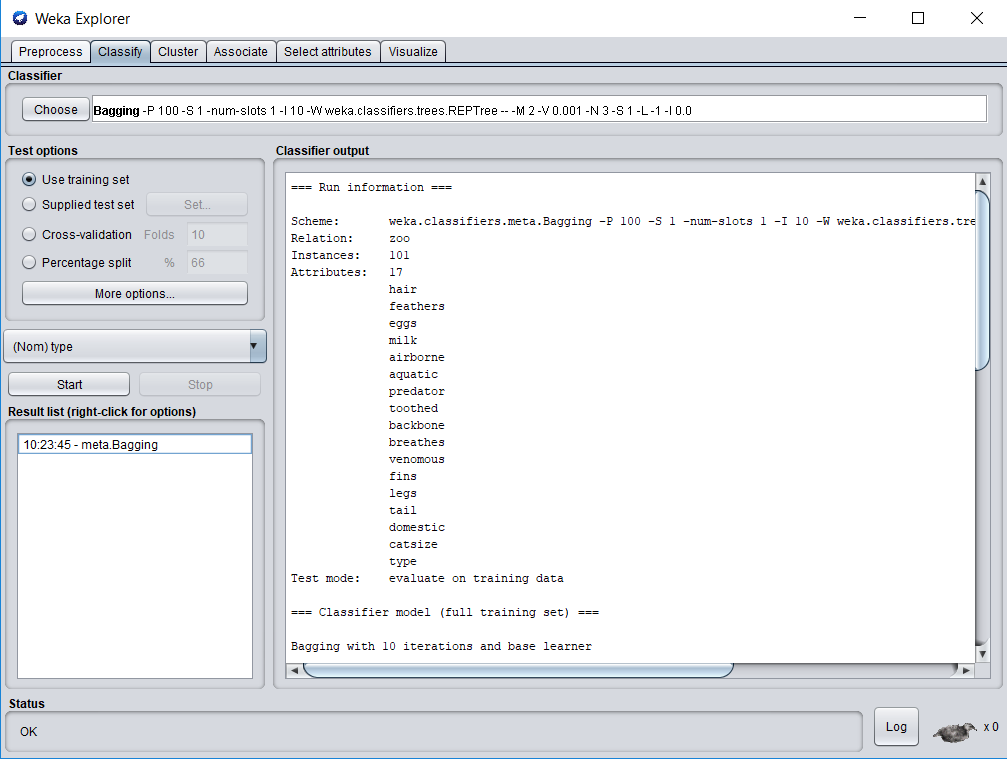
Kết quả chạy:

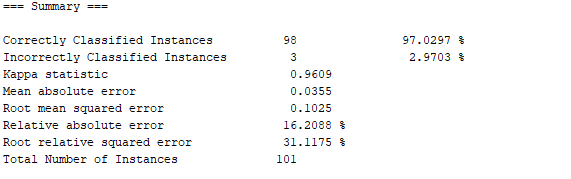
* Có 92 mẫu được phân lớp đúng (91,0891 %)
* Có 9 mẫu đưuọc phân lớp sai (8,9109 %)

Dựa vào confusion matrix ta thấy 9 loài bị phân lớp sai:

* Lớp Reptile (5 mẫu) đều phân lớp sai qua Mamal (1 mẫu), Bird (3 mẫu) và Fish (1 mẫu).
* Lớp Amphibian (4 mẫu) cũng đều bị phân lớp sai qua Mammal (2 mẫu), Bird (1 mẫu) và Fish (1 mẫu).

### Bagging:

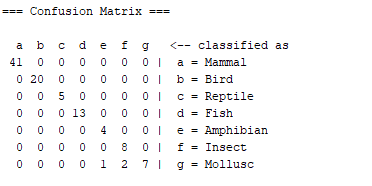




Phân lớp 101 mẫu với thuật toán **Bagging** (test option: **use training set**):

98 mẫu được phân lớp đúng, 3 mẫu bị phân lớp sai.

=> độ chính xác của thuộc tính trong trường hợp này là 97,0297 %



Dựa vào **Confusion Matrix** có 3 mẫu bị phân lớp sai như sau:

* 1 mẫu Moollusc (g) bị phân thành lớp Amphibian (e).
* 2 mẫu Moollusc (g) bị phân thành lớp Insect (f).

**Tài liệu tham khảo**

https://tailieu.vn/doc/khai-pha-du-lieu-gioi-thieu-ve-cong-cu-weka-1254352.html

https://vi.wikipedia.org/wiki/Weka\_(h%E1%BB%8Dc\_m%C3%A1y)

https://vi.scribd.com/presentation/214406821/Introduction-to-Weka