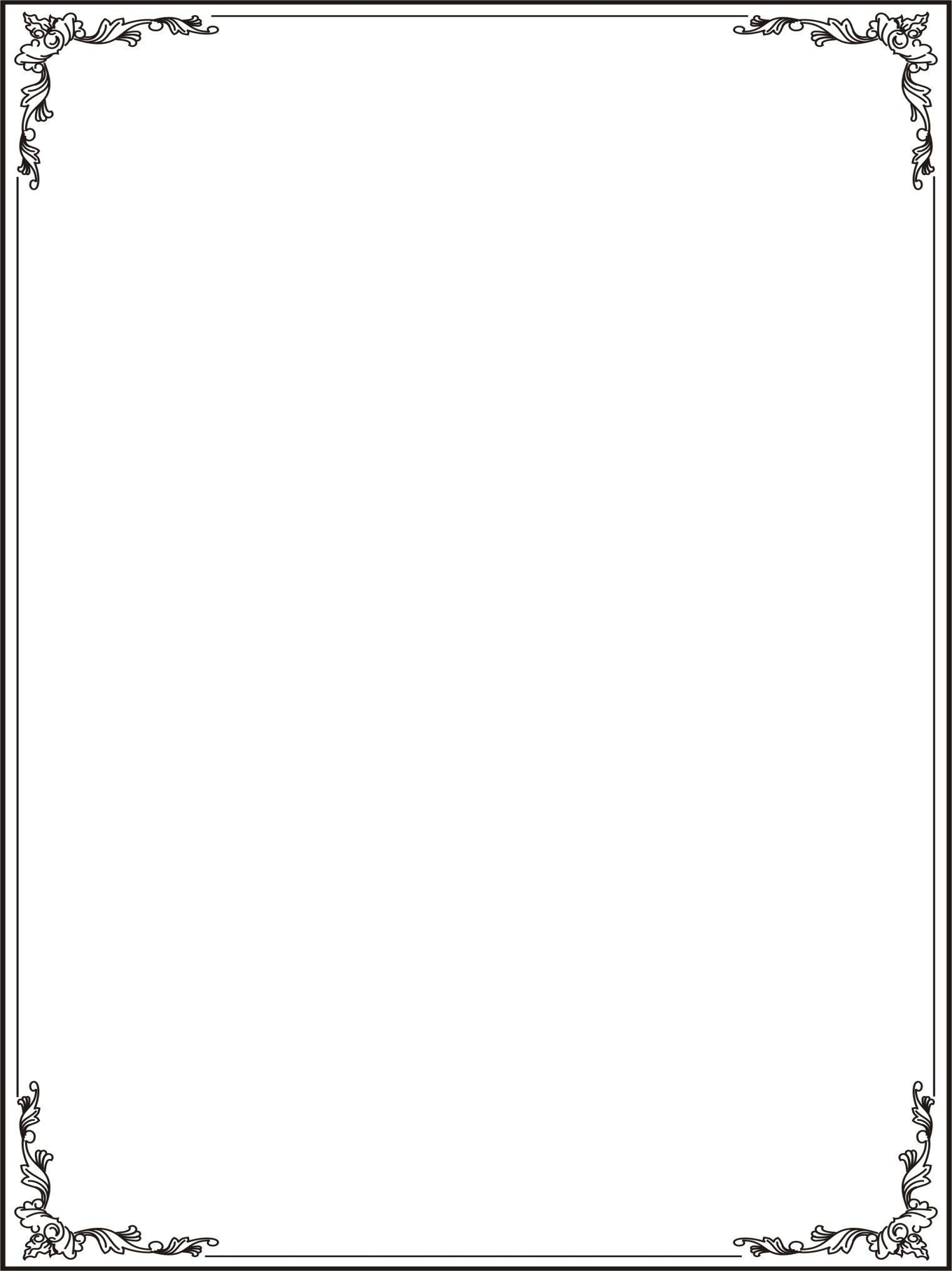
**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**



**BÁO CÁO BÀI TẬP LỚN**

MÔN : XỬ LÝ SONG SONG

**HIỆN THỰC GIẢI THUẬT DIJKSTRA VÀ**

**Giảng viên bộ môn :** Đặng Trần Trí **Giảng viên hướng dẫn :** Lê Thị Kim Tuyến  
 **Lớp** : A02  
 **Nhóm** : 5

***Đà nẵng, ngày 15 tháng 02 năm 2015 NĂMnăm2015221212202015***

***Hồ Chí Minh, 23/10/2016 NĂMnăm2015221212202015***

**DANH SÁCH THÀNH VIÊN**

* LÊ MẠNH QUÂN………………………………51303210
* NGUYỄN NGỌC THẬT…………......................
* TRẦN QUỐC TRÍ…………………………........
* NGÔ ANH VŨ………………………………….

1. **Tiền xử lý:**
2. **Xử lý dữ bị liệu thiếu và lọc nhiễu:**

Với file dữ liệu ban đầu ta thấy có nhiều ô bị NULL. Ta tiến hành thay những ô dữ liệu NULL đó bằng giá trị trung bình của cột chứa nó. Điều đó sẽ đảm bảo dữ liệu không bị lệch do những giá trị NULL đó.

Lọc nhiễu là quá trính thực hiện bằng kinh nghiệm ví dụ: ở cột nhiệt độ, ta thấy có những giá trị vô lý như 100, 5, 7.5,.. Nhiệt độ cơ thể sống không thể ở nhiệt độ này được nên đây có thể là những giá trị bị nhập sai. Hoặc ở cột bạch cầu ta thấy có giá trị 3, điều này cũng rất vô lý. Đối với những giá trị vô lý đó, ta tiến hành loại bỏ những dòng dữ liệu đó.

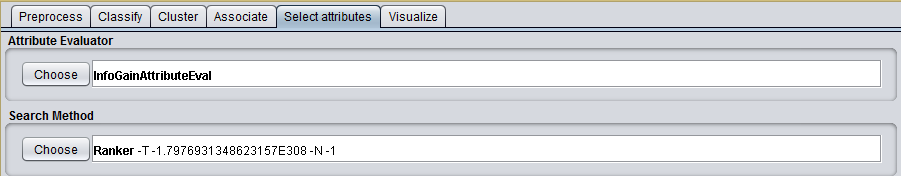
1. **Chọn thuộc tính:**

Trong giải thuật mạng neural thì việc chọn ra những thuộc tính có ảnh hưởng lên kết quả phân loại là điều hết sức cần thiết. Nó giúp mạng chạy chính xác và nhanh hơn.

Vì mạng chỉ xử lý các giá trị kiểu số (numberic) nên ta loại bỏ các trường : họ tên, ngày vào, chuẩn đoán ra viện. Và một số trường chắc chắn không gây ảnh hưởng đến kết quả như: STT, tuổi, khoảng tuổi, giới, nghề, nơi ở, cn(cân nặng), cc(chiều cao) và bmi(tỉ số diễn tả thể trạng người).

Xử lý kết quả phân loại: nếu giá trị ở cột ChuanDoanID lớn hơn 4 thì ta gán nó bằng 4. Trong đó, ChuanDoanID =1,2,3 tương đương với xuất huyết loại 1, 2, 3, ChuanDoanID=4 tương đương với sốt loại khác.

Ta dùng Weka để chọn thuộc tính. Ta chọn mục Select attributes, chọn Attribute Evaluator là: InfoGainAttributeEval, Search Method là: Ranker

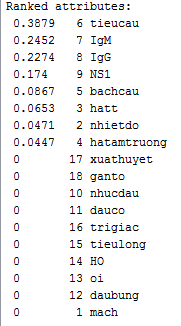


InfoGainAttributeEval: đánh giá các giá trị của thuộc tính bằng cách đo độ lợi thông tin liên quan đến các lớp học với:

InfoGain(Class,Attribute) = H(Class) - H(Class | Attribute).

Ranker: các thuộc tính được sắp xếp và đánh giá riêng rẽ. kết hợp với các thuộc tính đánh giá(ReliefF, GainRatio, Entropy, ..).  
P (tập bắt đầu) : bắt đầu bởi một tập các thuộc tính, bất kỳ thuộc tính bắt đầu nào được mô tả đều bị bỏ qua trong quá trình xếp hạnh  
T (ngưỡng): mô tả một ngưỡng mà thuộc tính có thể bị bỏ qua trong quá trình xếp hành  
N (số lượng lựa chon): mô tả số lượng thuộc tính được lựa chọn

Ta được kết quả sau:



Kết quả thu được là bảng xếp hạng độ ảnh hưởng của các thuộc tính lên kết quả phân loại được sắp xếp theo thứ tự từ cao đến thấp tương ứng với chiều từ trên xuống. Ta chọn 8 thuộc tính gây ảnh hưởng gồm: tieucau, IgM, IgG, NS1, bachcau, hatt, nhietdo, hatamtruong. Ngoài ra ta còn chọn 2 thuộc tính là xuathuyet và ganto vì theo một số tài liệu về bệnh sốt xuất huyết thì xuất huyết và gan to cũng là 2 biểu hiện rõ rệt của bệnh.

1. **Xử lý Outliers:**

Những phần tử biên (outliers) là những phần tử không có biểu diễn đặc trưng của thuộc tính.

Ta sử dụng công thức Interquartile Range (IQR):

Outlier là những phần tử nằm ở trên Q3 hoặc dưới Q1 một khoảng 1.5\*IQR với IQR = Q3-Q1, Q3 là percentile, Q1 là percentile.

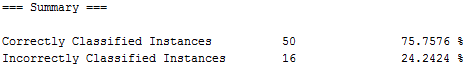
1. **Chuẩn hóa:**

Trong bài toán này, ta thử 2 cách chuẩn hóa là chuẩn hóa từ 0-1 và cách thứ 2 là chuẩn hóa theo z-score. Và ta sẽ so sánh kết quả ban đầu để chọn được cách chuẩn hóa hợp lý.

1. **Chuẩn hóa về khoảng [-1 1]:**

Chuẩn hóa về khoảng [-1 1] là dạng mặc định của weka. Công thức chuẩn hóa :

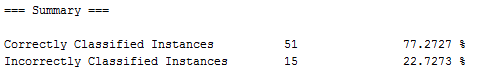
Sau khi chạy thử ta được kết quả độ chính xác là 75.7576%



1. **Chuẩn hóa theo z-score:**

Công thức chuẩn hóa:

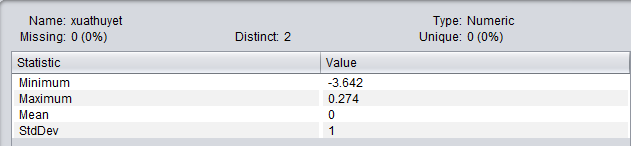
Chuẩn hóa theo dạng z-score cho kết quả là 77.2727%.



Đây là một trong những trường hợp cho kết quả là z-score trong bài toán này cho kết quả cao hơn từ 2-3%.

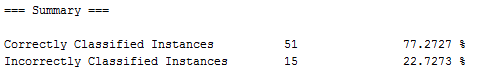
Trong weka có hỗ trợ việc chuẩn hóa z-scrore. Cụ thể như sau:

Ở bước tiền xử lý ta chọn filters 🡪 unsupervised 🡪 attribute 🡪 Standardize sau đó nhấn Apply, thì weka sẽ chuẩn hóa tất cả các thuộc tính về dạng z-score cho ta. Tuy nhiên ở những thuộc tính chỉ có giá trị luận lý (0 hoặc 1) thì weka cũng chuẩn hóa.

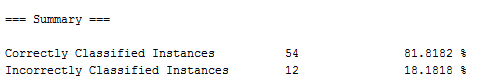


Như vậy có vẻ không được hợp lý nên ta sẽ so sánh giữa việc có chuẩn hóa z-score ở những thuộc tính luận lý và không chuẩn hóa ở những cột đó.

Với trường hợp có chuẩn hóa thì giải thuật cho kết quả là:



Còn với trường hợp không chuẩn hóa cho các thuộc tính đó thì kết quả là:



Như vậy ta thấy được rằng việc không chuẩn hóa z-score toàn bộ cho kết quả cao hơn.

Vậy tóm lại, để chuẩn bị dữ liệu một cách tốt nhất ta cần: xử lý dữ liệu bị thiếu, chọn thuộc tính, loại bỏ nhiễu, loại bỏ các phần tử ngoại biên, chuẩn hóa.

1. **Chạy giải thuật:**
2. **Các bước chạy:**

Chuyển file dữ liệu từ định dạng xlsx sáng csv(comma delimited). Import file vào weka.

Ta chọn mục classify 🡪 function 🡪 multilayerPerceptron. Ở đây ta sẽ có các test options sau:

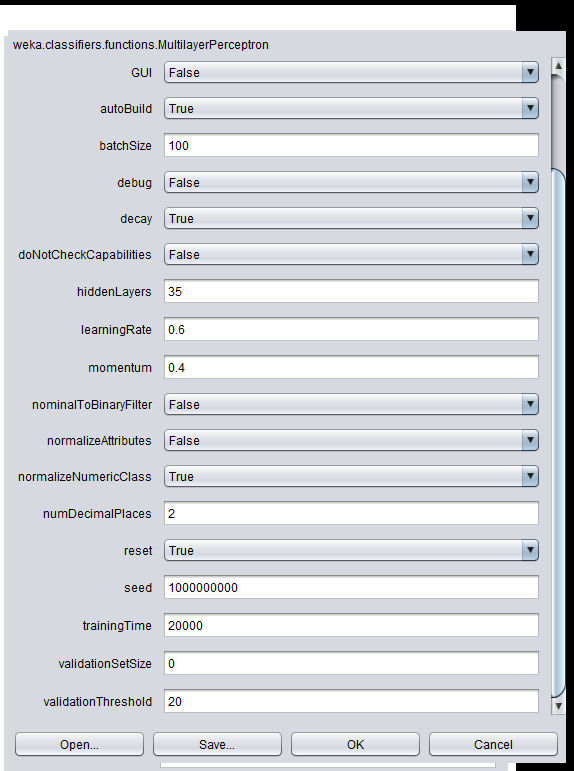
Use trainning set: test tri thức học được ngay trên chính tập trainning. Theo như khuyến cáo thì không nên dùng như vậy, vì chương trình chỉ cần xây dựng để lưu trữ tập trainning thì lúc đó độ chính xác sẽ đạt rất cao mà chẳng cần làm gì cả. Lý do thứ hai được đưa ra là tập trainning và tập test không được trùng nhau, như vậy kết quả đo được sẽ khách quan hơn.

Cross-validation(CV): giả sử bạn chọn 10 folds thì cứ 10 dòng weka sẽ chọn trong đó 9 dòng để train và 1 dòng để test và có sự xáo trộn giữa các lần train và test. Phương pháp này được dùng khá nhiều trong các nghiên cứu khoa học và được khuyên là nên sử dụng trên 5 folds.

Percentage split: chia dữ liệu thành 2 phần, x% để học và (100-x)% để test.

Trong bài toán này ta chọn percentage split.

1. **Tham số giải thuật:**



Trên là bảng điều chỉnh thông số của giải thuật.

Ý nghĩa của các thông số như sau:

GUI: có hiển thị mô hình mạng bằng đồ họa hay không? Mặc định là False.

AutoBuild: tự động connect các node giữa các lớp. Mặc định là True.

Debug: nếu là True thì quá trình classify sẽ thêm các thông tin khác vào console.

Decay: đây là yếu tố suy biến gây ra sự giảm learning rate. Nó sẽ làm cho learning rate giảm dần nếu khoảng cách giữa các node càng lớn, giúp mạng hoạt động chính xác hơn và cho hiệu năng chung tốt hơn.

Hidden layer: tại đây điều chỉnh số layer và số node trên các layer. Với các giá trị được định nghĩa sẵn cho mạng chỉ gồm 1 hidden layer:   
\* a: số node = (số thuộc tính + số classes)/2  
\* i: số node = số thuộc tính  
\* o: số node = số classes  
\* t: số node = số thuộc tính + số classes   
Ngoài ra ta có thể nhập số node và số node của các hidden layer cách nhau bằng dấu phẩy, nếu điền 0 thì nghĩa là mạng không có hidden layer.

Learning rate và momentum là 2 thông số rất quan trọng của giải thuật, đã được đề cập ở cơ sở lý thuyết.

normalizeAttributes: chuẩn hóa các attributes theo khoảng [-1 1].

numDecimalPlaces: sử dụng số thập phân cho kết quả thu được.

normalizeNumericClass: chuẩn hóa class nếu nó ở dạng số.

reset: cho phép mạng được đặt lại với learning rate nhỏ hơn.

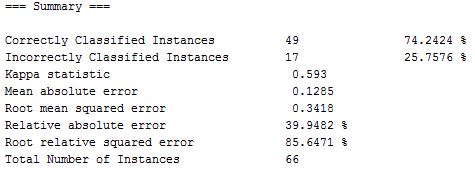
Seed: tham số của hàm random, nhằm tạo ra giá trị khởi tạo của mạng một cách ngẫu nhiên. Đây là 1 số kiểu long và nên đặt lớn hơn 0.

TrainningTime: số lần học.

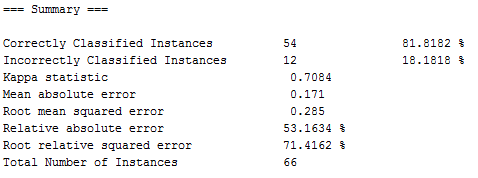
validationSetSize: độ lớn (%) của tập validation, mạng sẽ tiếp tục quá trình trainning cho tới khi nhận thấy sai số liên tục tăng hoặc đã đạt tới lượng trainning time.

1. **Điều chỉnh thông số:**
2. **Điều chỉnh decay:**

Trường hợp decay = False:



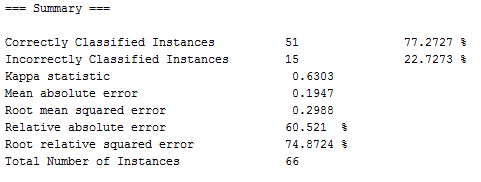
Trường hợp decay = True:



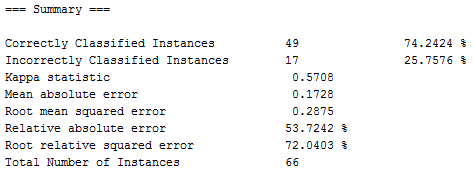
Kết quả đúng như đã nêu ở phần lý thuyết, nếu có decay thì mạng sẽ học hiệu quả hơn.

1. **Điều chỉnh hidden layer:**

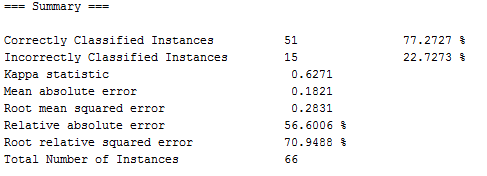
Với hidden layer = o:



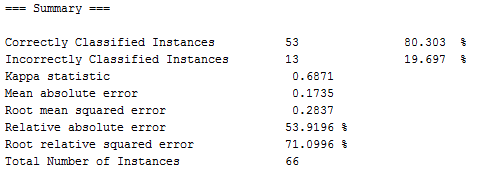
Với hidden layer = i:



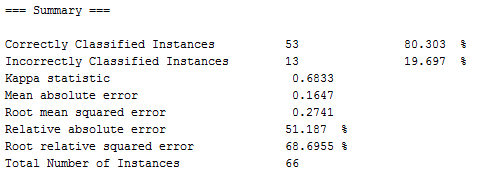
Với hidden layer = a:



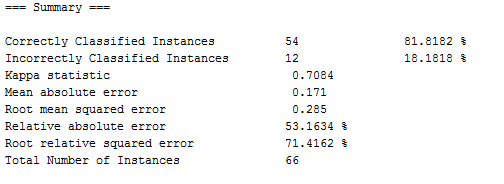
Với hidden layer = t:



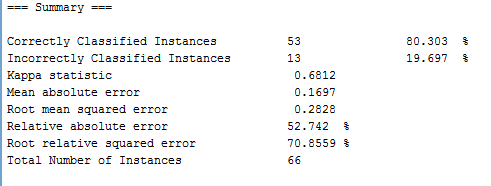
Với hidden layer = 20:



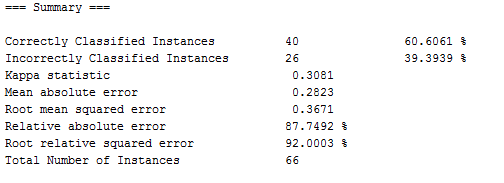
Với hidden layer = 35:



Với hidden layer =40:



Với hidden layer = 20,20:

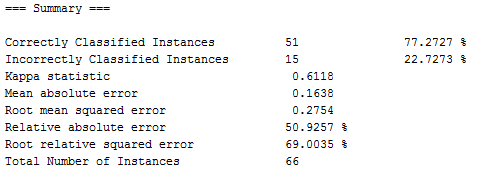


Ta thấy không phải lúc nào hidden layer nhiều node cũng tốt. Quan trọng nhất là tìm được một hidden layer với số node phù hợp và có được learning rate và momentum phù hợp.

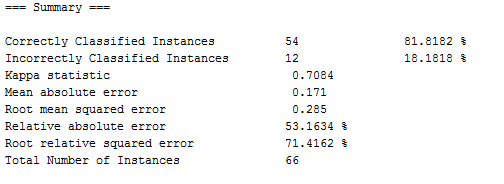
Theo nghiên cứu thì đối với các bài toán thông thường chỉ cần dùng 1 hidden layer và chỉ nên dùng 2 hidden layer trở lên với các bài toán về mô phỏng não người và động vật.

1. **Điều chỉnh seed:**

Với seed =0:



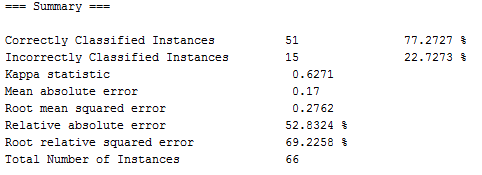
Với seed = 1.000.000.000:



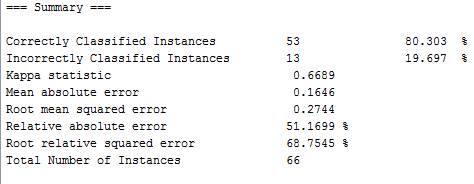
Vậy ta thấy được rằng nên đặt seed để bước khởi tạo diễn ra tốt hơn.

1. **Điều chỉnh trainning time:**

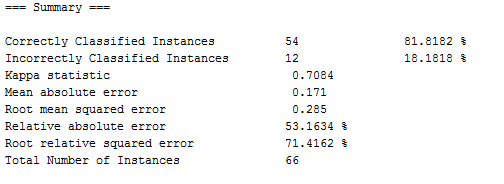
Trainning time = 1000:



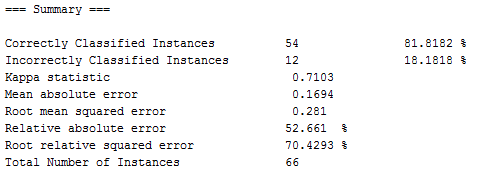
Trainning time = 10.000:



Trainning time =20.000:



Trainning time = 40.000: kết quả tốt hơn không đáng kể nhưng thời gian chạy là quá lâu.



Như ta đã thấy, không phải lúc nào tăng trainning time cũng tốt. Đôi khi tăng trainning time chỉ làm tốn thời gian mà kết quả có lúc còn tệ hơn. Nên chúng ta không nên nhanh chóng quyết định chọn trainning time thật lớn với hy vọng giải thuật sẽ tốt tuyệt đối.

1. **Các thông số kết quả:**

Correctly Classified Instances: số phân loại đúng.

Incorrectly Classified Instances: số phân loại sai.

Kappa statistic (hệ số kappa): là 1 thống kê sử dụng để kiểm tra độ tin cậy của dữ liệu.  
<0.4 : rất yếu  
0.4-0.6 : Trung bình  
0.61-0.8: Tốt  
0.81-1: Rất tốt

Mean absolute error(MAE) (gọi là sai số tuyệt đối trung bình) là 1 đại lượng để đo cường đọ trung bình của các sai sót trong 1 tập hợp dữ liệu.   
Chú ý: RMSE sẽ lun lớn hơn hoặc bằng MAE.

Root mean squared error (RMSE): độ lệch chuẩn.

TP rate: tỉ lệ phân loại đúng 1 class trên tổng số phân loại thuộc class đó.

FP rate: số lượng phân loại nhầm là đúng / tổng số thuộc class khác (Negative). Tỉ lệ này càng nhỏ càng tốt.

Precision: độ chính xác, trong tập tìm được bao nhiêu phân loại đúng.

Recall: độ bao phủ, trong số các tồn tại, tìm được bao nhiêu phân loại

F-Measure: được tính theo công thức 2\* Precision\* Recall/ (Recall + Precision)

MCC:

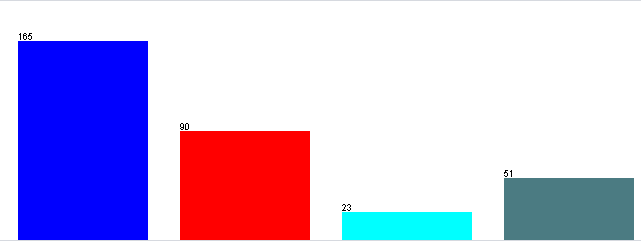
ROC Area:

PRC Area:

1. **Nhận xét về kết quả thu được:**

Kết quả lần chạy tốt nhất thì độ chính xác đánh giá được là 81.8182%. Đây không phải là độ chính xác quá cao. Giải thích cho việc này ta có thể đưa ra một số nguyên nhân sau:

* Bộ dữ liệu đầu vào khá nhiều nhiễu và dữ liệu thiếu khá nhiều.
* Dữ liệu đầu vào quá ít (sau khi làm sạch chỉ còn 329 mẫu). Với dữ liệu ít xỏi đó thì không thể nào biểu diễn cho một bài toán được. Chính vì vậy việc yêu cầu giải thuật đạt độ chính xác cao là rất khó.
* Dữ liệu bị lệch: ta thấy rằng số dữ liệu giữa các class là không đều, vậy nên nếu mạng neural dự đoán 100% là xuất huyết loại 1 thì tỉ lệ đúng cũng đã hơn 50%. Và xuất huyết loại 3 là quá ít.



* Cũng có thể do mạng neural được thiết kế chưa chính xác.

1. **Tri thức rút ra:**
2. **Tri thức về bệnh sốt xuất huyết:**

Theo như các attributes đã được chọn để dự đoán bệnh sốt xuất huyết thì ta thấy chỉ số về tiểu cầu là quan trọng nhất. Thật vậy, khi bệnh nhân có đồng thời hai dấu hiệu là giảm tiểu cầu và cô đặc máu thì được chuẩn đoán là sốt xuất huyết.

* Độ I: giảm tiểu cầu kèm cô đặc máu nhưng không có chảy máu tự phát.
* Độ II: giảm tiểu cầu và cô đặc máu kèm theo chảy máu tự phát.
* Độ III: giảm tiểu cầu và cô đặc máu, huyết động không ổn định: mạch lăn tăn, huyết áp kẹp (hiệu số huyết áp tâm thu – huyết áp tâm trương < 20 mm Hg)

Qua đó ta thấy được rằng xuất huyết cũng là một trong những triệu chứng để nhận biết bệnh.

Sốt là biểu hiện đầu tiên của bệnh. Người bệnh sốt cao từ 39-40 độ kéo dài từ 5 đến 7 ngày, đi kèm là các triệu chứng như mệt mỏi, chán ăn,đau bụng, nhức đàu,…Ngoài ra, gan to là biểu hiện của bệnh nặng.

1. **Tri thức về cách xử lý số liệu:**

Bước tiền xử lý là bước rất quan trọng. Kết quả thu được phụ thuộc rất lớn vào việc tiền xử lý thế nào. Việc tiền xử lý cần phải chú trọng ở các khâu sau: đầu tiên phải xử lý những dữ liệu bị thiếu, ta nên ưu tiên chọn cách thay dữ liệu bị thiếu bằng giá trị mean của toàn bộ thuộc tính, điều đó có thể giúp dữ liệu đỡ bị lệch hơn, tuy nhiên cần phải cần thận vì có những thuộc tính không thể thay bằng mean ví dụ như những thuộc tính có giá trị kiểu boolean chẳng hạn.

Chúng ta cần chuẩn bị sẵn những kiến thức liên quan đến bài toán để phục vụ cho việc loại bỏ nhiễu và các thuộc tính không gây ảnh hưởng đến kết quả đầu ra.

Nên sử dụng các giải thuật select attributes kèm với những kiến thức thực tế để chọn ra những attributes phù hợp nhất. Việc này giúp tiết kiệm thời gian chạy và đảm bảo hiệu quả của giải thuật.

Dữ liệu trước khi được đưa vào giải thuật cần chuẩn hóa để tránh gặp những sai số do sự chênh lệch về miền dữ liệu. Ta cần chọn cách chuẩn hóa phù hợp với từng bài toán.

Ngoài ra, tùy trường hợp mà ta còn cần phải tích hợp và thu giảm dữ liệu.

1. **Tri thức về mạng neural:**

Có 4 tham số rất quan trọng trong mạng neural mà cần phải thử nhiều lần để chọn ra những bộ tham số tốt nhất: learning rate, momentum, hiddenlayer và trainning time.

Theo nghiệm ta nên chọn 2 tham số learning rate và momentum trước, không nên đặt 2 tham số này quá lớn. Tiếp theo ta chọn hidden layer theo thứ tự từ ít đến nhiều và khi đã chọn được hiddenlayer hợp lý thì ta bắt đầu tăng trainning time.

Khi tăng số hidden layer và trainning time thì thời gian chạy của giải thuật sẽ tăng đáng kể(trong lần chạy tốt nhất, hiddenlayer =35 và trainningtime= 20.000, chạy trong vòng 20 phút). Nên phải có sự cân bằng giữa thời gian chạy giải thuật và độ chính xác sao cho hợp lý. Trên thực tế mạng neural là một giải thuật cho kết quả khá tốt nhưng không phải lúc nào nó cũng cho kết quả tốt nhất (kết quả ở đây có xét tới độ chính xác và thời gian chạy).

Ngoài ra nên đặt các tham số giúp tăng độ chính xác cho mạng như decay, seed,…

https://sites.google.com/site/diepnn80/datamininginfo/cacdodohaydungchobaitoanphanloai