TRƯỜNG ĐẠI HỌC THỦY LỢI

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

Description automatically generated with low confidence

**BÀI TẬP LỚN**

HỌC PHẦN: HỌC MÁY

**ĐỀ TÀI: Dự đoán bệnh ung thư vú**

Giáo viên hướng dẫn: Nguyễn Thị Kim Ngân

Nhóm sinh viên thực hiện:

1.Nguyễn Thị Giang 21551062758 63CNTT1

2. Phạm Thanh Lam 2151061166 63CNTT1

3. Nghiêm Đức Quân 2151060200 63CNTT1

4. Trần Đăng Trung 2151062885 63CNTT1

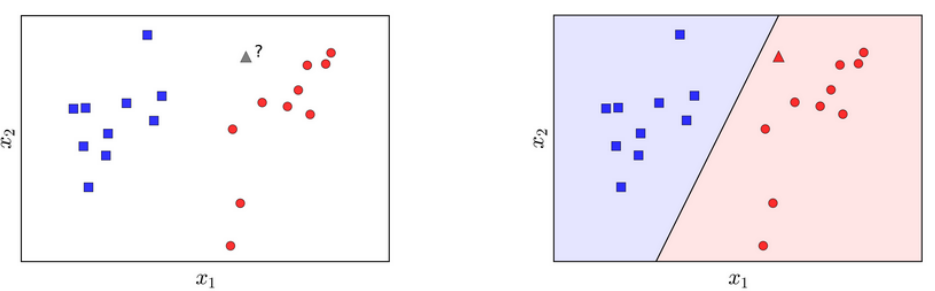
**Hà Nội, năm 2023**

**Phần 1: Lý thuyết**

**1. Perceptron:**

Perceptron là một thuật toán Classification cho trường hợp đơn giản nhất: chỉ có hai class (binary classification) và cũng chỉ hoạt động được trong một trường hợp rất cụ thể

Giả sử chúng ta có hai tập hợp dữ liệu đã được gán nhãn

****

Cho hai class được gán nhãn, hãy tìm một đường phẳng sao cho:

+toàn bộ các điểm thuộc class 1 nằm về 1 phía,

+ toàn bộ các điểm thuộc class 2 nằm về phía còn lại của đường phẳng đó.

Nếu tồn tại một đường phẳng phân chia hai class thì ta gọi hai class đó là linearly separable.

Các thuật toán classification tạo ra các boundary là các đường phẳng được gọi chung là Linear Classifier

ý tưởng cơ bản của PLA: xuất phát từ một nghiệm dự đoán nào đó, qua mỗi vòng lặp, nghiệm sẽ được cập nhật tới một ví trí tốt hơn cập nhật dựa trên việc giảm giá trị của một hàm mất mát nào đó.

G/S ma trận chứa các điểm dữ liệu:

X = [x1, x2, … , x𝑁] ∈ ℝ𝑑×𝑁

Mỗi cột x𝑖 ∈ ℝ𝑑×1 là một điểm dữ liệu trong không gian d chiều

G/S các nhãn của mỗi điểm dữ liệu được lưu trong một véc tơ hàng

y = [𝑦1, 𝑦2, … , 𝑦𝑁] ∈ ℝ1×𝑁

Với 𝑦𝑖 = 1 nếu x𝑖 thuộc class 1 và 𝑦𝑖 = −1 nếu x𝑖 thuộc class 2

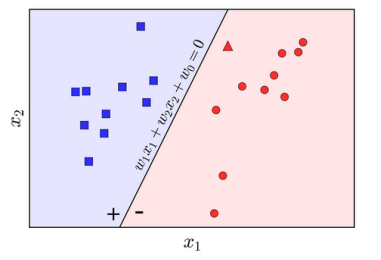
G/S, tại một thời điểm, ta tìm được đường boundary là đường phẳng có phương trình:

𝑓w x = 𝑤1𝑥1 + 𝑤2𝑥2 + ⋯ + 𝑤𝑑𝑥𝑑 + 𝑤0 = 0

Hay

w𝑇 x = 0 w=(w0 , w1 , w2 , …, wd ) x=(x1, x1 , x2 , …, xd )

Để đơn giả, ta làm việc với trường hợp mỗi điểm dữ liệu chỉ có hai chiều (d=2) G/S đường thẳng 𝑤1𝑥1 + 𝑤2𝑥2 + 𝑤0 = 0 là nghiệm cần tìm như hình:

****

Như vậy, các điểm nằm về cùng một phía so với đường thẳng làm cho 𝑓w (x) mang cùng dấu

Các dấu này tương ứng với nhãn 𝑦 của mỗi lớp

Vậy, nếu tìm được w (nghiệm của bài toán Perceptron) và một điểm x chưa có nhãn, ta có thể xác định class của nó bởi:

𝑙𝑎𝑏𝑒𝑙 x = 1 𝑛ế𝑢 w𝑇 x ≥ 0, 𝑛𝑔ượ𝑐 𝑙ạ𝑖 − 1

Hay:

𝑙𝑎𝑏𝑒𝑙 x = 𝑠𝑔𝑛( w𝑇 x)

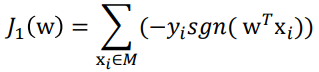
Xây dựng hàm mất mát:

Tiếp theo, ta cần xây dựng hàm mất mát với tham số w bất kỳ.

Vẫn trong không gian hai chiều, giả sử đường thẳng 𝑤1𝑥1 + 𝑤2𝑥2 + 𝑤0 = 0 được cho như:

Điều chúng ta mong muốn là không có điểm nào bị misclassified.

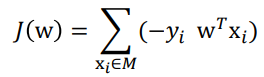
Hàm mất mát đơn giản nhất chúng ta nghĩ đến là hàm đếm số lượng các điểm bị misclassied và tìm cách tối thiểu hàm số này:

****

M: số điểm phân lớp lỗi

Hạn chế quan trọng: hàm số này là rời rạc, không tính được đạo hàm theo w nên rất khó tôi ưu

Xét hàm mất mát:



Khi một x𝑖 (bị phân lớp sai) nằm càng xa boundary thì giá trị −𝑦𝑖 w𝑇 x𝑖 càng lớn

Giá trị nhỏ nhất của hàm mất mát này bằng 0 nếu không có điểm nào bị phân lớp sai

Hàm này trừng phạt nặng những điểm lấn sâu sang lãnh thổ của lớp khác

Tại một thời điểm, nếu ta chỉ quan tâm đến điểm bị phân lớp sai, hàm 𝐽(w): tính được đạo hàm Ta có thể sử dụng giảm Gradient để tìm w Với một điểm x𝑖bị phân lớp sai, hàm mất mát trở thành:



Đạo hàm:



aw=> (aw)’=a

Qui tắc cập nhật:

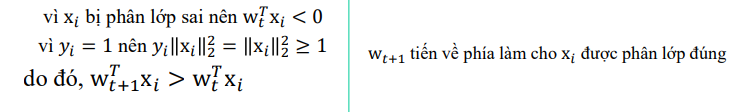
w = w + η𝑦𝑖 x𝑖

wt+1= wt + ηf’(w) = wt+η𝑦𝑖 x𝑖

Ta có một quan sát nhỏ

w𝑡+1 𝑇 x𝑖 = (w𝑡 + 𝑦𝑖 x𝑖) 𝑇 x𝑖 = w𝑡 𝑇 x𝑖 + 𝑦𝑖 x𝑖 2 2

Nếu 𝑦𝑖 = 1:



Đến đây, cảm nhận của chúng ta với thuật toán này là: cứ chọn đường boundary đi.

Xét từng điểm một, nếu điểm đó bị misclassified thì tiến đường boundary về phía làm cho điểm đó được classifed đúng.

Có thể thấy rằng, khi di chuyển đường boundary này:các điểm trước đó được classified đúng có thể lại bị misclassified.

Mặc dù vậy, ta sẽ tìm được đường phẳng phân chia hai lớp, miễn là hai lớp đó là linearly separable.

Kết luận:

1. Tại thời điểm t = 0, chọn ngẫu nhiên một vector hệ số w0

2. Tại thời điểm t, nếu không có điểm dữ liệu nào bị phân lớp lỗi, dừng thuật toán.

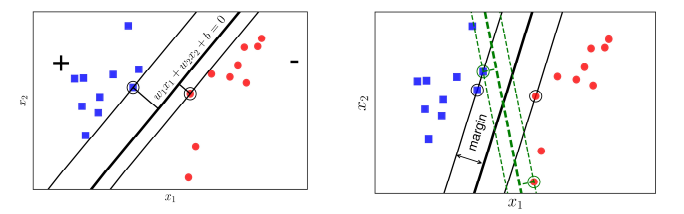
3. Giả sử xi là một điểm bị phân lớp lỗi. Cập nhật

wt+1 = wt + yi xi

4. Thay đổi t = t + 1 rồi quay lại Bước 2

**2. SVM:**

Cần tìm một tiêu chuẩn để đo sự công bằng của hai class

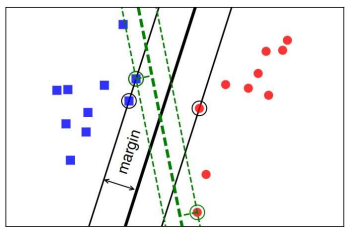
****

Ý tưởng của SVM: Margin của một siêu phẳng được định nghĩa là khoảng cách từ các điểm gần nhất của lớp đó tới mặt phân chia. Margin của hai lớp phải bằng nhau và lớn nhất có thể

Cần tìm một đường phân chia sao cho:

♣ Khoảng cách từ điểm gần nhất của mỗi lớp tới đường phân chia là như nhau (margin, lề).

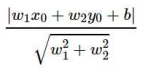
♣ Margin này phải là cực đại

****

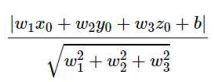
Bài toán tối ưu trong Support Vector Machine (SVM) là tìm đường phân chia sao cho margin là lớn nhất (Maximum Margin Classifier)

Độ đo khoảng cách:

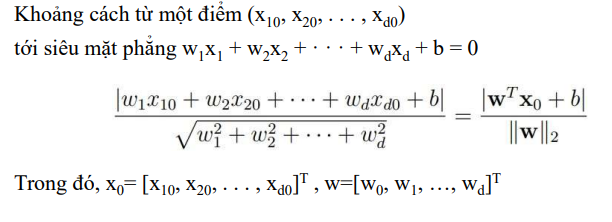
* Trong không gian 2D:



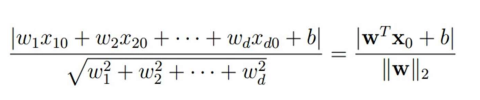
* Trong không gian 3D:



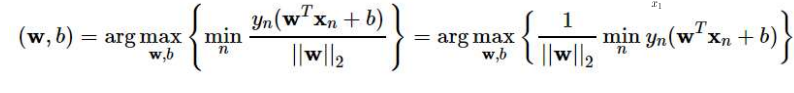
* Trong không gian d chiều:



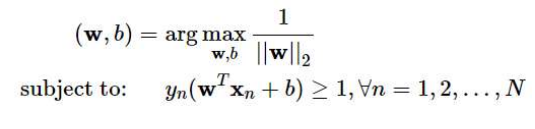
Dấu của biểu thức:

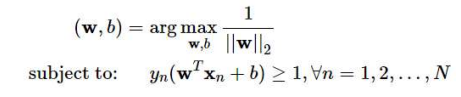


Bài toán tối ưu cho SVM:

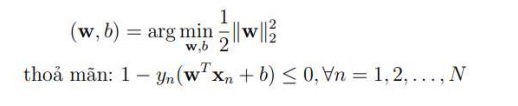


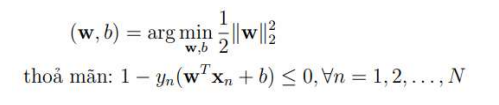
* Có thể đưa về bài toán tối ưu có ràng buộc sau đây:





* Bằng phép lấy nghịch đảo, bài toán trên chuyển thành:





* Bài toán tối ưu này là bài toán lồi, và là một quadratic programming (phương trình bậc 2)
* Suy ra nghiệm cho SVM là duy nhất
* Để giải bài toán này, người ta thường giải bài toán đối ngẫu Lagrange

Xác định lớp cho một điểm dữ liệu mới

♣ Sau khi đã tìm được mặt phân cách wTx + b = 0

♣ Nhãn của bất kỳ một điểm được xác định bằng class(x) = sgn(wTx + b)

Tóm tắt:

* Nếu bỏ dấu trị tuyệt đối ở tử số, ta biết được điểm đó nằm về phía nào của mặt phẳng đang xét:

♣ Những điểm mang dấu dương nằm về cùng 1 phía

♣ Những điểm mang dấu âm nằm về phía còn lại

♣ Những điểm nằm trên măt phẳng làm cho tử số có giá trị bằng 0, tức khoảng cách bằng

* Với bài toán binary classification mà 2 classes là linearly separable, có vô số các siêu mặt phẳng giúp phân biệt hai classes:

♣ Với mỗi mặt phân cách, ta có một classifier

♣ Khoảng cách gần nhất từ 1 điểm dữ liệu tới mặt phân cách ấy được gọi là margin của classifier đó.

* Support Vector Machine là bài toán đi tìm mặt phân cách sao cho margin tìm được là lớn nhất, đồng nghĩa với việc các điểm dữ liệu an toàn nhất so với mặt phân cách
* Bài toán tối ưu trong SVM là một bài toán hoàn toàn lồi (strictly convex) bậc 2. Nghiệm của bài toán này là duy nhất
* Mặc dù có thể trực tiếp giải SVM qua bài toán tối ưu gốc này, thông thường người ta thường giải bài toán đối ngẫu.
* Bài toán đối ngẫu cũng là một QP nhưng nghiệm là sparse nên có những phương pháp giải hiệu quả hơn.
* Với các bài toán mà dữ liệu gần linearly separable hoặc nonlinear separable, có những cải tiến khác cảu SVM để thích nghi với dữ liệu đó

**3. ID3:**

ID3(Iterative Dichotomiser 3) là một thuật toán trong cây quyết định để xây dựng mô hình dự đoán.

Nhiệm vụ của giải thuật ID3 là học cây quyết định từ một tập các ví dụ rèn luyện (training example) hay còn gọi là dữ liệu rèn luyện (training data)

Ý tưởng:

* chúng ta cần xác định thứ tự của thuộc tính cần được xem xét tại mỗi bước
* tại mỗi bước, một thuộc tính tốt nhất sẽ được chọn ra dựa trên một tiêu chuẩn nào đó
* Với mỗi thuộc tính được chọn, ta chia dữ liệu vào các nút con tương ứng với các giá trị của thuộc tính đó rồi tiếp tục áp dụng phương pháp này cho mỗi nút con
* Việc chọn ra thuộc tính tốt nhất ở mỗi bước như thế này được gọi là cách chọn tham lam (greedy). Cách chọn này có thể không phải là tối ưu, nhưng trực giác cho chúng ta thấy rằng cách làm này sẽ gần với cách làm tối ưu
* Tìm các cách phân chia hợp lý (thứ tự chọn thuộc tính hợp lý) sao cho hàm mất mát cuối cùng đạt giá trị càng nhỏ càng tốt.
* việc này đạt được bằng cách chọn ra thuộc tính sao cho nếu dùng thuộc tính đó để phân chia, entropy tại mỗi bước giảm đi một lượng lớn nhất

Thuật toán GenDecTree(Sample S, Attlist A)

1. Tạo một nút N

2. Nếu tất cả các mẫu thuộc cùng lớp C thì N được gán nhãn C; dừng thuật toán;

3. Nếu A là rỗng thì N được gán nhãn C là nhãn phổ biến nhất trong S; dừng thuật toán;

4. Chọn a thuộc A, có độ đo information gain cao nhất; Gán nhãn N theo a;

5. Với mỗi giá trị v của a:

a. Phát triển 1 nhánh từ N với điều kiện a=v;

b. Đặt Sv là tập con của S với a=v;

c. Nếu Sv là rỗng thì gắn một lá có nhãn phổ biến nhất trong S;

d. Ngược lại gắn một nút được tạo bởi GenDecTree(Sv , A-a)

Ta cần đi tìm một phép đo chất lượng của một cách phân chia:

* phép phân chia tốt
* Ta cần có một hàm số đo độ tinh khiết (purity), hoặc độ vẩn đục (impurity) của một phép phân chia
* Một hàm số có các đặc điểm này và được dùng nhiều trong lý thuyết thông tin là hàm entropy

Hàm số entropy:

-Cho một phân phối xác suất của một biến rời rạc x có thể nhận n giá trị khác nhau 𝑥1 , 𝑥2 , … , 𝑥𝑛.

+Giả sử rằng xác suất để x nhận các giá trị này là

Ký hiệu phân phối này là 𝑝 = (𝑝1 , 𝑝2 , … , 𝑝𝑛).

-Entropy của phân phối này được định nghĩa là

Xét một bài toán với C class khác nhau:

- GS ta đang làm việc với một non-leaf node với các điểm dữ liệu tạo thành một tập 𝑆 với số phần tử là |𝑆| = 𝑁.

- Giả sử thêm rằng trong số 𝑁 điểm dữ liệu này, 𝑁𝑐 , 𝑐 = 1,2, … , 𝐶 điểm thuộc vào class 𝑐. Xác suất để mỗi điểm dữ liệu rơi vào một class 𝑐 được xấp xỉ bằng 𝑁𝑐/ 𝑁 (maximum likelihood estimation).

- entropy tại n

GS thuộc tính được chọn là 𝑥. Dựa trên 𝑥, các điểm dữ liệu trong 𝑆 được phân ra thành 𝐾 child node 𝑆1 , 𝑆2 , … , 𝑆𝐾 với số điểm trong mỗi child node lần lượt là 𝑚1 , 𝑚2 , … , 𝑚𝐾. Ta định nghĩa

Ta định nghĩa information gain dựa trên thuộc tính x:

Tại mỗi node, thuộc tính được chọn được xác định dựa trên:

Điều kiện dừng:

* nếu ta tiếp tục phân chia các node chưa tinh khiết, ta sẽ thu được một tree mà mọi điểm trong tập huấn luyện đều được dự đoán đúng.
* Khi đó, tree có thể sẽ rất phức tạp (nhiều node) với nhiều leaf node chỉ có một vài điểm dữ liệu. Như vậy, nhiều khả năng overfitting sẽ xảy ra
* Để tránh overfitting, chúng ta phải có điều kiện dừng

Tại một node, nếu một trong số các điều kiện sau đây xảy ra, ta không tiếp tục phân chia node đó và coi nó là một leaf node:

1.Nếu node đó có entropy bằng 0, tức mọi điểm trong node đều thuộc một class.

2. Nếu node đó có số phần tử nhỏ hơn một ngưỡng nào đó. Trong trường hợp này, ta chấp nhận có một số điểm bị phân lớp sai để tránh overfitting. Class cho leaf node này có thể được xác định dựa trên class chiếm đa số trong node.

3. Nếu khoảng cách từ node đó đến root node đạt tới một giá trị nào đó. Việc hạn chế chiều sâu của tree này làm giảm độ phức tạp của tree và phần nào giúp tránh overfitting.

4. Nếu tổng số leaf node vượt quá một ngưỡng nào đó.

5. Nếu việc phân chia node đó không làm giảm entropy quá nhiều (information gain nhỏ hơn một ngưỡng nào đó)

Pruning:

-Pruning là một kỹ thuật regularization để tránh overfitting cho decision tree nói chung

-Trong pruning:

+Một decision tree sẽ được xây dựng tới khi mọi điểm trong training set đều được phân lớp đúng

+Sau đó, các leaf node có chung một non-leaf node sẽ được cắt tỉa và non-leaf node đó trở thành một leaf-node, với class tương ứng với class chiếm đa số trong số mọi điểm được phân vào node đó

Việc cắt tỉa cây quyết định này có thể được xác định dựa vào các cách sau:

* Dựa vào một validation set

+ Trước tiên, training set được tách ra thành một training set nhỏ hơn và một validation set.

+ Decision tree được xây dựng trên training set cho tới khi mọi điểm trong training set được phân lớp đúng.

+Sau đó, đi ngược từ các leaf node, cắt tỉa các sibling node của nó và giữ lại node bố mẹ nếu độ chính xác trên validation set được cải thiện.

+ Khi nào độ chính xác trên validation set không được cải thiện nữa, quá trình pruning dừng lại

* Dựa vào toàn bộ data set :

+Sử dụng toàn bộ dữ liệu trong tập này cho việc xây dựng decision tree.

+Một ví dụ cho việc này là cộng thêm một đại lượng regularization vào hàm mất mát.

+Đại lượng regularization sẽ lớn nếu số leaf node là lớn. Cụ thể:

* + Giả sử decision tree cuối cùng có K leaf node, tập hợp các điểm huấn luyện rơi vào mỗi leaf node lần lượt là 𝑆1,.… , 𝑆𝐾. Khi đó, regularized loss của ID3 có thể được tính tương tự như:

**4. CART**

Dựa vào đặc điểm của biến mục tiêu, có thể chia Decision Tree thành hai dạng:

♣ Classification Tree: nếu biến mục tiêu thuộc dạng categorical variable

♣ Regression Tree: nếu biến mục tiêu thuộc dạng continous variable

Sự khác nhau giữa Classification Tree và Regression Tree

♣ Regression Tree có biến mục tiêu là biến liên tục, trong khi Classification Tree có biến mục tiêu la biến phân loại.

♣ Trong Regression Tree, khi huấn luyện, giá trị tại nút lá bằng trung bình các giá trị biến mục tiêu của các điểm dữ liệu có trong nút đó. Nên khi đưa tập test vào, nếu các điểm dữ liệu rơi vào nút lá nào, kết quả trả ra sẽ là giá trị trung bình.

♣ Với Classification Tree, khi huấn luyện, giá trị tại nút lá(phân lớp) bằng giá trị có tần suất cao nhất(Mode) của các dữ liệu trong nút đó. Nên khi đưa tập test vào, nếu các điểm dữ liệu rơi vào nút lá nào, kết quả trả ra sẽ là Mode

Làm sao Decision Tree quyết định khi nào sẽ phân nhánh

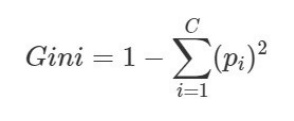
♣ Các quyết định phân nhánh sẽ ảnh hưởng đến độ chính xác của Cây.

♣ Cây hồi quy và cây phân lớp có các thuật toán phân nhánh khác nhau.

♣ Có nhiều thuật toán phân nhánh, tùy vào kiểu của biến mục tiêu mà sử dụng thuật toán như thế nào.

♣ Có thuật toán chính : Gini Index (CART), Reduction in Variance

Gini Index:



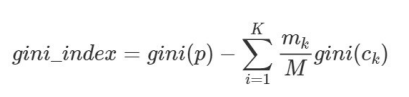
Trong đó,

♣ C: số lớp cần phân loại

♣ pi =ni /N,

♣ ni là số lượng phần tử ở lớp thứ i

♣ N là tổng số lượng phần tử ở node đó



Trong đó,

♣ gini(p): chỉ số gini ở node cha

♣ K: số node con được tách ra

♣ gini(ck ): chỉ số gini ở node con thứ k

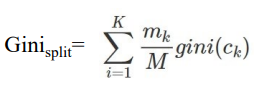
♣ M: số phần tử ở node p

♣ mi : là số phần tử ở node con thứ i



Gini split:

Chọn thuộc tính có hệ số Ginisplit nhỏ



**Phần 2: Ứng dụng trong thực tế**

***1. .Mô tả bài toán***

* Tên bài toán: dự đoán bệnh ung thư vú
* Mục đích của bài toán: dự đoán liệu khối u là lành tính hay ác tính dựa trên dữ liệu đặc điểm.
* Input: gồm 5 trường

+ mean\_radius :bán kính trung bình của khố u

+ mean\_texture: kết cấu trung bình của khối u

+ mean\_perimeter: chu vi trung bình của khối u

+ mean\_area: diện tích trung bình của khối u

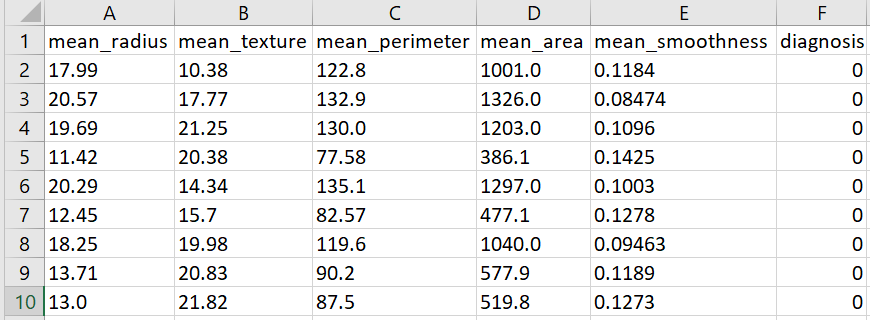
+ mean\_smoothness: độ mịn trung bình của khối u

* Ouput: diagnosis: chuẩn đoán

Trường này chứa nhãn nhị phân, trong đó:

+ 0 cho thấy một khối u lành tính (không ung thư).

+ 1 cho thấy một khối u ác tính (ung thư).



* Tóm tắt công việc thực hiện của bài toán:

1. Đọc dữ liệu
2. Chia tập dữ liệu (70% dùng để huấn luyện mô hình, 30% dùng để kiểm tra sự phù hợp của mô hình)

3. Dùng tập dữ liệu kiểm tra để so sánh các độ đo: tỷ lệ mẫu được dự đoán đúng (Accuracy), Precision, Recall, F1-score (chương 8 của tập slides bài giảng) của Perceptron, SVM, ID3, hoặc sử dụng các hàm có sẵn trong thư viện sklearn để tính các độ đo này

4. In ra kết quả

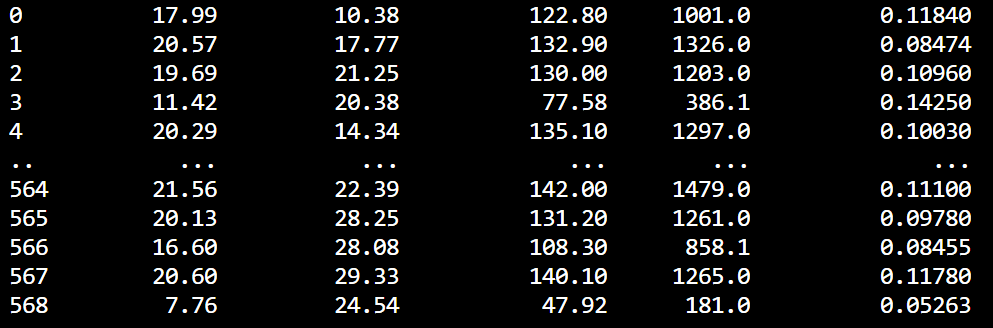
***2. Mô tả tập dữ liệu của bài toán***

* Dữ liệu gồm những chiều thông tin gì (mỗi mẫu (vertor) dữ liệu có những thông tin gì), có bao nhiêu mẫu dữ liệu (ít nhất là 100 vector dữ liệu). Mô tả nhãn lớp của dữ liệu.

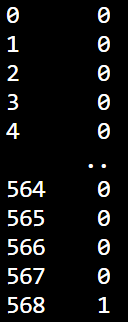
|  |  |
| --- | --- |
| Nhãn | Ý nghĩa |
| mean\_radius | bán kính trung bình của khố u |
| mean\_texture | kết cấu trung bình của khối u |
| mean\_perimeter | chu vi trung bình của khối u |
| mean\_area | diện tích trung bình của khối u |
| mean\_smoothness | độ mịn trung bình của khối u |
| diagnosis | Chuẩn đoán |

Có 569 mẫu dữ liệu

Tập dữ liệu X:



Tập nhãn lớp Y:



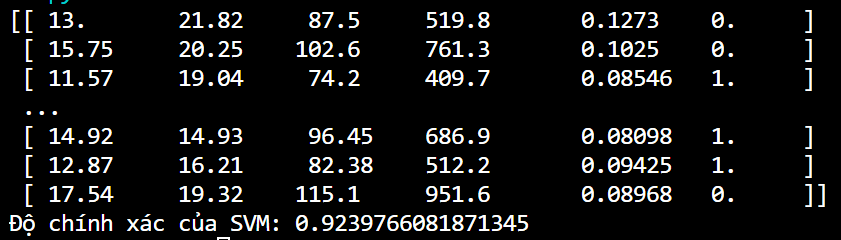
* Chia tập dữ liệu thành 2 phần: 70% dùng để huấn luyện mô hình, 30% dùng để kiểm tra sự phù hợp của mô hình.

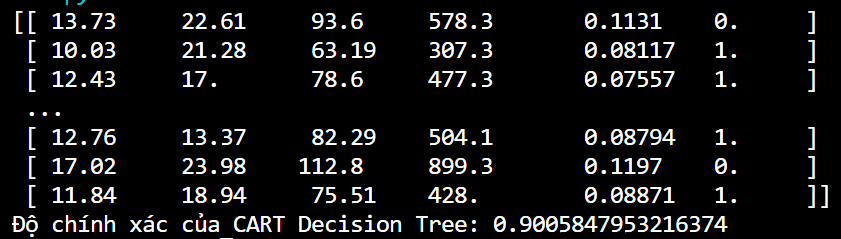
***3. Viết ứng dụng***

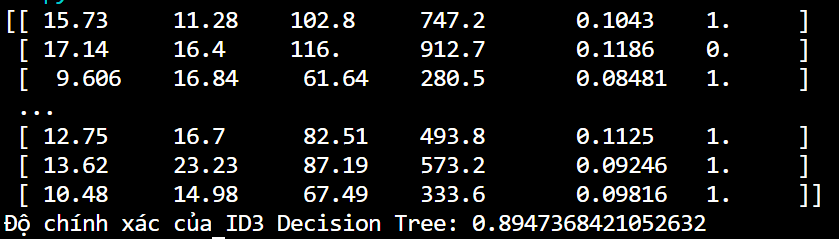
* Viết code của các thuật toán Perceptron, SVM, ID3 và CART để giải quyết bài toán.

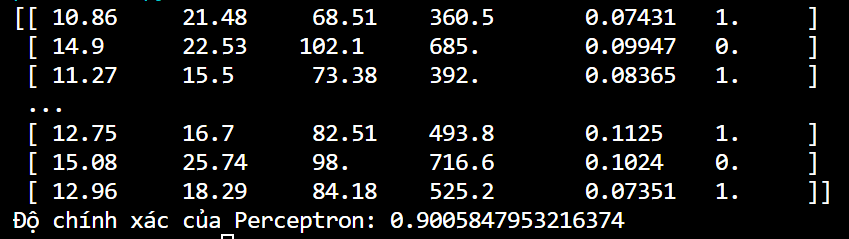
***4. Phân tích kết quả của chương trình***

* Tỷ lệ dự đoán đúng trên tập test









* So sánh tỷ lệ dự đoán đúng của 3 thuật toán. Từ đó, lựa chọn thuật toán SVM decisionTree là phù hợp nhất với bài toán.

**Kết luận**

* Sử dụng thuật toán Perceptron, ID3,CART decisionTree,SVM decisionTree cho bài toán chuẩn đoán ung thư vú
* Đánh giá chất lượng mô hình bằng độ đo Accuracy
* So sánh và tìm ra mô hình SVM decisionTree phù hợp với tập dữ liệu đã chọn

**Tài liệu tham khảo**

* File dữ liệu .csv trong Breast Cancer Dataset trong trang web kaggle: <https://www.kaggle.com/datasets/deeplearner09/breast-cancer-dataset>
* Slide “Học máy” trong “PDF bai giang” của cô Nguyễn Thị Kim Ngân
* Tài liệu tham khảo học máy cơ bản: <https://machinelearningcoban.com>
* SVM:<https://scikitlearn.org/stable/modules/generated/sklearn.svm.SVR.html>
* Perceptron: <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html>
* ID3 và CART: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html