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 NỒNG ĐỘ CỒN CỦA RƯỢU**

Giáo viên hướng dẫn: Nguyễn Thị Kim Ngân

Sinh viên/nhóm sinh viên thực hiện:

1. Lương Anh Tú, lớp 63CNTT.02

2. Nguyễn Quốc Huy, lớp 63CNTT.02

3. Dương Ngô Quyền, lớp 63CNTT.02

**Hà Nội, năm 2023**

**Phần 1: Lý thuyết**

* Trình bày lý thuyết của các thuật toán Perceptron, SVM, ID3 và CART.

**I.Thuật toán Perceptron(PLA)**

Trình bày lý thuyết của các thuật toán Perceptron

* Perceptron là thuật toán đơn giản giúp tìm một ranh giới siêu
* Perceptron là thuật toán đơn giản giúp tìm một ranh giới siêu phẳng cho bài toán phân lớp nhị phân
* Phương trình đường siêu phẳng
* Giả sử X = [x1,x2,…,xn] là tập dữ liệu huấn luyện. Y =
  + X=[x1,x2,..,xn] là tập dữ liệu huấn luyện. Y=[y1,y2,…,yn] là nhãn tương ứng với từng điểm dữ liệu trong tập X. Với yi = 1 nếu xi thuộc lớp 1 và yi = -1 nếu xi thuộc lớp 2



* + Với w[w1,w2,…,wn] là vector hệ số và w0 là hệ số tự do
* Xây dựng hàm mất mát
* thẳng w1x1 + w2x2 + w0 = 0 là nghiệm cần
  + Giả sử đường thẳng w1x1+w2x2+w0 = 0 là nghiệm cần tìm. Các điểm nằm cùng một phía sẽ làm cho hàm fw(x) mang cùng dấu. Vậy nếu w là một nghiệm của bài toán perceptron với một dữ liệu X chưa được gán nhãn, ta có thể xác định nhãn của nó bằng phép toán



* Xét một điểm xi bất kì với nhãn yi. Nếu bị phân lớp lỗi, ta
  + Xét một điểm xi bất kỳ với nhãn yi. Nếu bị phân lớp lỗi ta phải có sgn(w^Tx) ≠ yi. Vì hai giá trị này là khác nhau nên yi \* sgn(w^Tx) = -1. Như vậy, hàm số đếm số lượng điểm bị phân lớp lỗi có dạng

A white background with black text

Description automatically generated

* + Tìm w cho không có điểm phân lỗi ⌠ J1(w) = 0 nhưng đây là hàm số rời rạc nên rất khó tối ưu nên ta sẽ xét hàm mất mát
  + A black and white symbol

    Description automatically generated
  + Với một điểm dữ liệu Xi bị phân lỗi, hàm mất mát sẽ trở thành A math equation with a symbol

    Description automatically generated with medium confidence
  + Đạo hàm

A black text on a white background

Description automatically generated

* + Cập nhật lại theo công thức



* + Lặp lại tới khi tất cả các điểm được phân lớp đúng

**II.Thuật toán ID3**

* **Ý tưởng**
  + - Chúng ta cần xác định thứ tự của thuộc tính cần được xem xét tại mỗi bước
    - Tạo mỗi bước, một thuộc tính tốt nhất sẽ được chọn ra dự 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
* **Thuật toán ID3**
  + - Algorithm 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∈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)
* **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
    - pi = p(x=xi) , với 0 ≤ pi ≤ 1, A black text with a white background

      Description automatically generated
    - Ký hiệu phân phối này là p = (p1 , p2 , … , pn)
    - Entropy của phân phối này được định nghĩa là A black and white math equation

      Description automatically generated

**III.Thuật toán CART**

* Classification and Regression Tree(CART) là một thuật toán dự đoán được sử dụng trong học máy. Nó giải thích cách dự đoán giá trị của biến mục tiêu dựa trên các giá trị khác. Nó là một cây quyết định trong đó mỗi ngã ba được chia thành một biến dự đoán và mỗi nút ở cuối có một dự đoán cho biến mục tiêu
* CART là một loại thuật toán phân loại được yêu cầu để xây dựngcây quyết định trên cơ sở chỉ số Gini's impurity index.
* Gini Index
  + - Công thức tính chỉ số Gini A math equation with numbers and symbols

      Description automatically generated
    - Trong đó,
      * - C: số lớp cần phân loại
      * - pi=ni/N,
      * - N
      * - ni là số lượng phần tử ở lớp thứ i
      * - N là tổng số lượng phần tử ở node đó
* Công thức tính chỉ số Gini index A black text on a white background

  Description automatically generated
  + - 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 A black text on a white background

        Description automatically generated
* Gini split
  + - Công thức tính Gini split A mathematical equations and formulas

      Description automatically generated with medium confidence

oPerceptron là thuật toán đơn giản giúp tìm một ranh giới siêu

oPhương trình đường siêu phẳng

**IV.Thuật toán SVM**

Giống như [Perceptron Learning Algorithm (PLA)](https://machinelearningcoban.com/2017/01/21/perceptron/), [Support Vector Machine (SVM) thuần](https://machinelearningcoban.com/2017/04/09/smv/) chỉ làm việc khi dữ liệu của 2 classes là [linearly separable](https://machinelearningcoban.com/2017/01/21/perceptron/#bai-toan-perceptron). Một cách tự nhiên, chúng ta cũng mong muốn rằng SVM có thể làm việc với dữ liệu gần linearly separable giống như [Logistic Regression](https://machinelearningcoban.com/2017/01/27/logisticregression/) đã làm được.

Có hai trường hợp dễ nhận thấy SVM làm việc không hiệu quả hoặc thậm chí không làm việc:

* Trường hợp 1: Dữ liệu vẫn linearly separable như Hình 1a) nhưng có một điểm nhiễu của lớp tròn đỏ ở quá gần so với lớp vuông xanh. Trong trường hợp này, nếu ta sử dụng SVM thuần thì sẽ tạo ra một margin rất nhỏ. Ngoài ra, đường phân lớp nằm quá gần lớp vuông xanh và xa lớp tròn đỏ. Trong khi đó, nếu ta hy sinh điểm nhiễu này thì ta được một margin tốt hơn rất nhiều được mô tả bởi các đường nét đứt. SVM thuần vì vậy còn được coi là nhạy cảm với nhiễu (sensitive to noise).
* Trường hợp 2: Dữ liệu không linearly separable nhưng gần linearly separable như Hình 1b). Trong trường hợp này, nếu ta sử dụng SVM thuần thì rõ ràng bài toán tối ưu là infeasible, tức feasible set là một tập rỗng, vì vậy bài toán tối ưu SVM trở nên vô nghiệm. Tuy nhiên, nếu ta lại chịu hy sinh một chút những điểm ở gần biên giữa hai classes, ta vẫn có thể tạo được một đường phân chia khá tốt như đường nét đứt đậm. Các đường support đường nét đứt mảnh vẫn giúp tạo được một margin lớn cho bộ phân lớp này. Với mỗi điểm nằm lần sang phía bên kia của các đường suport (hay đường margin, hoặc đường biên) tương ứng, ta gọi điểm đó rơi vào vùng không an toàn. Chú ý rằng vùng an toàn của hai classes là khác nhau, giao nhau ở phần nằm giữa hai đường support.

-**Xây dựng bài toán tối ưu cho SVM*:***

+ Giả sử rằng các cặp dữ liệu của training set là(x1,y1),(x2,y2),…,(xn,yn) với :

* vector  thể hiện đầu vào của một điểm dữ liệu
* yi là nhãn của điểm dữ liệu đó
* d là số chiều của dữ liệu
* N là số điểm dữ liệu

Với cặp dữ liệu (xn ,yn ) bất kỳ, khoảng cách từ điểm đó tới mặt phân chia là:

A black text on a white background

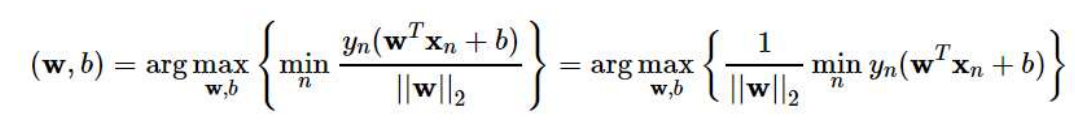
Description automatically generated

* Vì y𝑛 luôn cùng dấu với phía của x𝑛 , nên y𝑛 cùng dấu với  , tử số luôn là 1 số không âm.
* Với mặt phân chia như trên, margin được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó: (x1,y1),(x2,y2),…,(xn,yn)

A black text with a black line

Description automatically generated

Bài toán tối ưu trong SVM chính là bài toán tìm và b sao cho margin này đạt giá trị lớn nhất:



* Việc giải trực tiếp bài toán này sẽ rất phức tạp, nhưng ta có cách để đưa nó về bài toán đơn giản hơn
* Nhận xét quan trọng nhất là nếu ta thay vector hệ số bởi và bởi trong đó là một hằng số dương thì mặt phân chia không thay đổi, tức khoảng cách từ từng điểm đến mặt phân chia không đổi, tức margin không đổi

\* **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) = 

**Phần 2: Ứng dụng trong thực tế**

**1. .Mô tả bài toán**

* Tên bài toán: Dự đoán nồng độ cồn trong rượu
* Mục đích của bài toán: dựa trên các thông số của các thành phần có trong rượu để dự đoán nồng độ rượu. Các dữ liệu thu thập được tập hợp thành 1 tập huấn luyện
* Input:
  + fixed acidity: độ axit cố định
  + volatile acidity: độ axit dễ bay hơi
  + citric acid: nồng độ axit hữu cơ
  + residual sugar: lượng đường
  + chlorides: lượng clorua
  + free sulfur dioxide: lượng lưu huỳnh dioxit tự do trong rượu
  + total sulfur dioxide: tổng lượng lưu huỳnh dioxit trong rượu
  + density: mật độ khối lượng của một đơn vị thể tích
  + pH: độ pH
  + sulphates: lượng muối sunfat có trong rượu
  + alcohol: nồng độ rượu
* Ouput:
  + quality: đánh giá nồng độ rượu
* Tóm tắt công việc thực hiện của bài toán: dự đoán nồng độ rượu là nặng hay nhẹ thông qua các thành phần có trong rượu

**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.
* Mô tả ma trận dữ liệu (X), 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.

\*Dữ liệu của bài toán

A table of chemical data

Description automatically generated with medium confidence

* Thu thập tổng cộng 1597 mẫu
* Ma trận dữ liệu X gồm các cột:
  1. fixed acidity: độ axit cố định
  2. volatile acidity: độ axit dễ bay hơi
  3. citric acid: nồng độ axit hữu cơ
  4. residual sugar: lượng đường
  5. chlorides: lượng clorua
  6. free sulfur dioxide: lượng lưu huỳnh dioxit tự do trong rượu
  7. total sulfur dioxide: tổng lượng lưu huỳnh dioxit trong rượu
  8. density: mật độ khối lượng của một đơn vị thể tích
  9. pH: độ pH
  10. sulphates: lượng muối sunfat có trong rượu
  11. alcohol: nồng độ rượu
* Nhãn lớp Y bao gồm cột quality
* 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**

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

\*Đánh giá chất lượng mô hình

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Perceptron | ID3 | CART | SVM |
| Accuracy | 0.9561586638830898 | 0.9582463465553236 | 0.941544885177453 | 0.9498956158663883 |
| Unaccuracy | 0.04384133611691021 | 0.04175365344467641 | 0.05845511482254695 | 0.050104384133611735 |
| Precision\_micro | 0.9561586638830898 | 0.9582463465553236 | 0.941544885177453 | 0.9498956158663883 |
| Precision\_macro | 0.4780793319415449 | 0.48214285714285715 | 0.5425403711737768 | 0.47494780793319413 |
| Recall\_micro | 0.9561586638830898 | 0.9582463465553236 | 0.941544885177453 | 0.9498956158663883 |
| Recall\_macro | 0.4780793319415449 | 0.48214285714285715 | 0.5425403711737768 | 0.47494780793319413 |
| F1\_score\_micro | 0.9561586638830898 | 0.9582463465553236 | 0.941544885177453 | 0.9498956158663882 |
| F1\_score\_macro | 0.48879402347918893 | 0.48933901918976547 | 0.5473812095032398 | 0.487152034261242 |

=> Dựa vào kết quả của các độ đo, thuật toán Perceptron phù hợp nhất với bài toán

**\*Kết luận**

* Sử dụng thuật toán Perceptron,ID3,CART cho bài toán dự báo nồng độ cồn của rượu
* Đánh giá chất lượng mô hình bằng các độ đo Accuracy,Precision,Recall,F1

**Tài liệu tham khảo**

-Slide bài giảng :Chương 3.2,chương 4.1,chương 4.2,chương 5.2

- <https://machinelearningcoban.com/2017/01/21/perceptron/>

- <https://machinelearningcoban.com/2018/01/14/id3/>

- dataset: https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009