TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

Báo cáo đồ án môn học

Project II

Áp dụng Machine learning và Deep learning trong một số bài toán

Sinh viên thực hiện : Vũ Minh Thanh

MSSV : 20183633

Giảng viên hướng dẫn : thầy Đỗ Phan Thuận

Hà Nội

7/2021

Mục lục

[Lời mở đầu 3](#_Toc77432513)

[Chương 1. Học máy (machine learning) và áp dụng 4](#_Toc77432514)

[1.1. Cơ sở lý thuyết 4](#_Toc77432515)

[1.1.1. Hồi quy tuyến tính (Linear Regression) 4](#_Toc77432516)

[1.1.2. Hồi quy logistic 5](#_Toc77432517)

[1.1.3. Mạng nơ-ron (Neural Networks) 7](#_Toc77432518)

[1.1.4. Máy vecto hỗ trợ (Support vector machines) 8](#_Toc77432519)

[1.2. Áp dụng vào bài toán Rain in Australia 14](#_Toc77432520)

[1.2.1. Phân tích, tiền xử lý dữ liệu 14](#_Toc77432521)

[1.2.2. Kết quả các thí nghiệm đánh giá hiệu năng hệ thống 30](#_Toc77432522)

[Chương 2. Thị giác máy tính (computer vision) và áp dụng 37](#_Toc77432523)

[2.1. Phân đoạn ảnh (image segmentation) 37](#_Toc77432524)

[2.1.1. Cơ sở lý thuyết 37](#_Toc77432525)

[2.1.2. Áp dụng vào bài toán đo kích thước vật thể 42](#_Toc77432526)

[2.2. Phát hiện vật thể (Object Detection) 43](#_Toc77432527)

[2.2.1. Cơ sở lý thuyết 43](#_Toc77432528)

[2.2.2. Áp dụng vào bài toán phát hiện xe trên cao tốc 50](#_Toc77432529)

[Chương 3. Khó khăn và hướng phát triển 57](#_Toc77432530)

[Tài liệu tham khảo 57](#_Toc77432531)

# Lời mở đầu

Trong những năm trở lại đây, AI – Artificial Intelligent (Trí tuệ nhân tạo) , cụ thể hơn là ML – Machine learning (Học máy hoặc máy học) đang phát triển mạnh mẽ và được sự quan tâm đặc biệt từ cộng đồng học thuật cũng như từ xã hội. Là trái tim của cuộc Cách mạng công nghiệp lần thứ 4, trí tuệ nhân tạo đang len lõi vào mọi lĩnh vực trong cuộc sống hằng ngày của chúng ta mà có thể chúng ta không nhận ra. Ví dụ như: hệ thống gợi ý sản phẩm của Amazon, hệ thống gợi ý quảng cáo từ Facebook, khả năng tự động tag khuôn mặt trong ảnh của Facebook, tính năng tự động phân loại ảnh theo Album của Google photo và nổi bật hơn cả không thể không kể đến xe tự lái của Google, Tesla; máy chơi cờ vây AlphaGo của Google DeepMind, . . .

Machine Learning là một lĩnh vực nhỏ của Khoa Học Máy Tính, nó có khả năng tự học hỏi dựa trên dữ liệu đưa vào mà không cần phải được lập trình cụ thể.

Những năm gần đây, khi mà khả năng tính toán của các máy tính được nâng lên một tầm cao mới và lượng dữ liệu khổng lồ được thu thập bởi các hãng công nghệ lớn, Machine Learning đã tiến thêm một bước dài và một lĩnh vực mới được ra đời gọi là Deep Learning. Deep Learning đã giúp máy tính thực thi những việc tưởng chừng như không thể vào 10 năm trước: phân loại cả ngàn vật thể khác nhau trong các bức ảnh, tự tạo chú thích cho ảnh, bắt chước giọng nói và chữ viết của con người, giao tiếp với con người, hay thậm chí cả sáng tác văn hay âm nhạc.

Trong môn học này, tôi được sự hướng dẫn của thầy Đỗ Phan Thuận, đã tìm hiểu được các kiến thức tổng quan về Machine learning và Deep learning thông qua khóa học Machine learning trên trang <https://www.coursera.org/> và khóa học Computer Vision trên <https://www.deeplearning.ai/> và đi sâu một số kiến trúc áp dụng trong các bài toán cụ thể.

Bài báo cáo của tôi bao gồm 2 chương :

Chương 1: Học máy (Machine learning) và áp dụng

Chương 2: Thị giác máy tính (Computer Vision) và áp dụng

Chương 3: Khó khăn và hướng phát triển

# Chương 1. Học máy (machine learning) và áp dụng

## Cơ sở lý thuyết

### Hồi quy tuyến tính (Linear Regression)

* Biểu diễn:

m : số lượng ví dụ học

n : số lượng đặc trưng của 1 ví dụ học

* Đầu ra dự đoán :
* Hàm mất mát (Loss function) :
* Gradient Descent:
* Phương trình chuẩn tắc:

### Hồi quy logistic

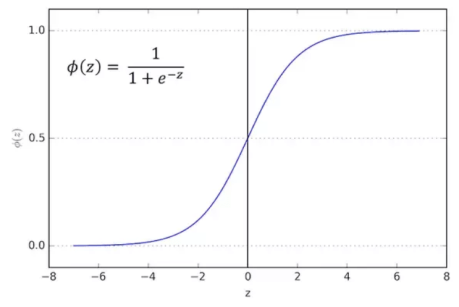
Đầu ra dự đoán của bài toán hồi quy logistic :

Trong đó : là hệ số cần học

x là giá trị đầu vào

g là hàm logistic, ở đây g là hàm sigmoid

a, Hàm Sigmoid



Nguồn : https://nttuan8.com/bai-2-logistic-regression/

* Hàm số liên tục, nhận giá trị thực trong khoảng (0,1).
* Hàm có đạo hàm tại mọi điểm (để áp dụng gradient descent)

g′(z) = = = g(z)(1 – g(z))

b, Mô hình

Xét bài toàn phân loại 2 nhãn lớp, giá trị dự đoán y nhận 2 giá trị 0 và 1

là xác suất của điểm dữ liệu x để output y = 1

c, Hàm mất mát

Hàm mất mát cho 1 điểm dữ liệu :

Ta có:

Do đó :

Biểu diễn dưới dạng vector:

Cập nhật hệ số theo Gradient Decent :

d, Ranh giới quyết định (Decision Boundary)

### Mạng nơ-ron (Neural Networks)

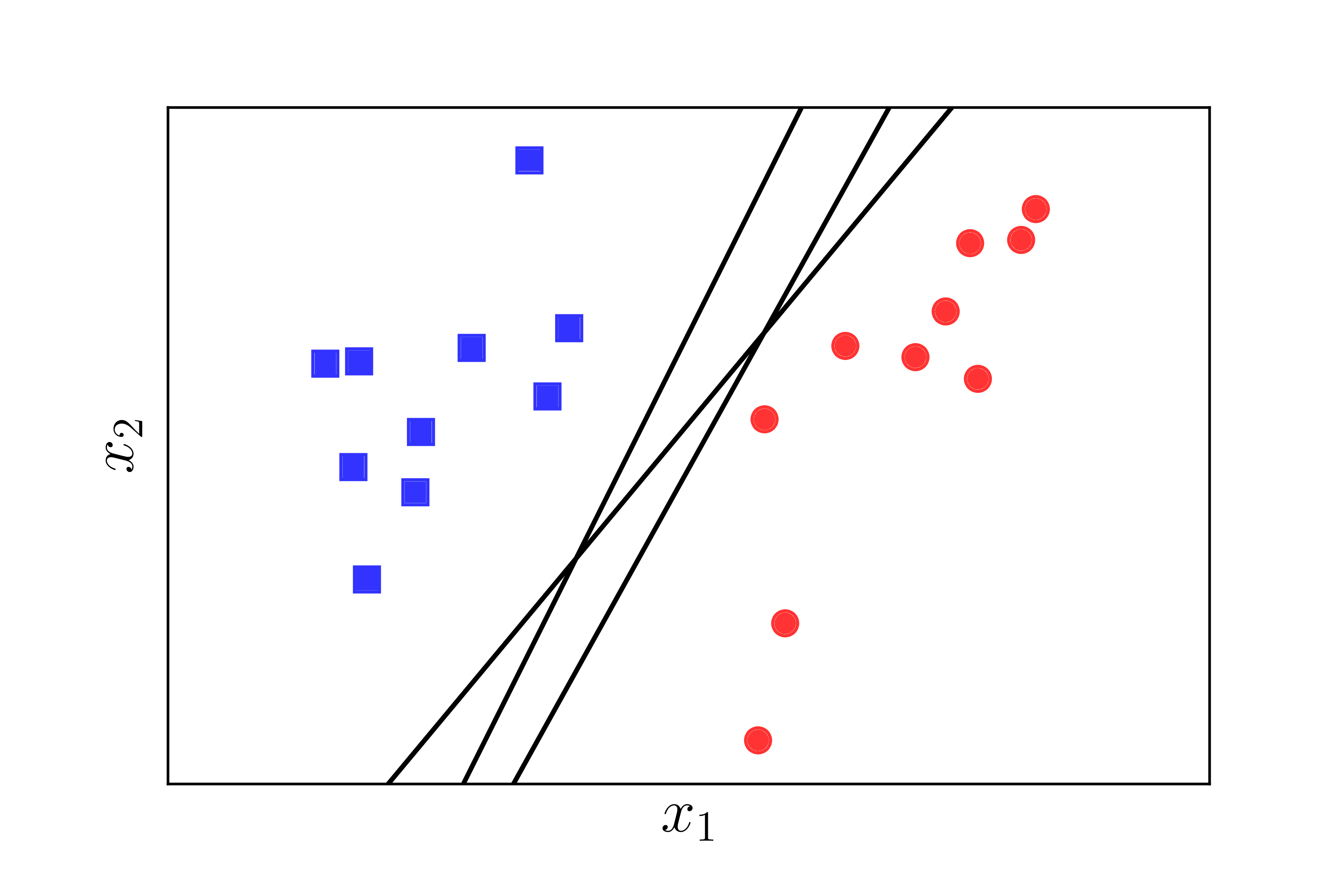
* Đầu vào : x1…xn, bias x0 = 0
* Đầu ra :
* : “activation” của nốt i trong layer j
* : ma trận trọng số từ lớp j đến lớp j+1, nếu lớp j có sj­ nốt và lớp j+1 có sj+1 nốt thì có chiều là sj+1 x (sj + 1)
* g là hàm kích hoạt (bẻ cong sự tuyến tính) : sigmoid, tanh, relu, leaky relu
* Biểu diễn vector:

Đặt

### Máy vecto hỗ trợ (Support vector machines)

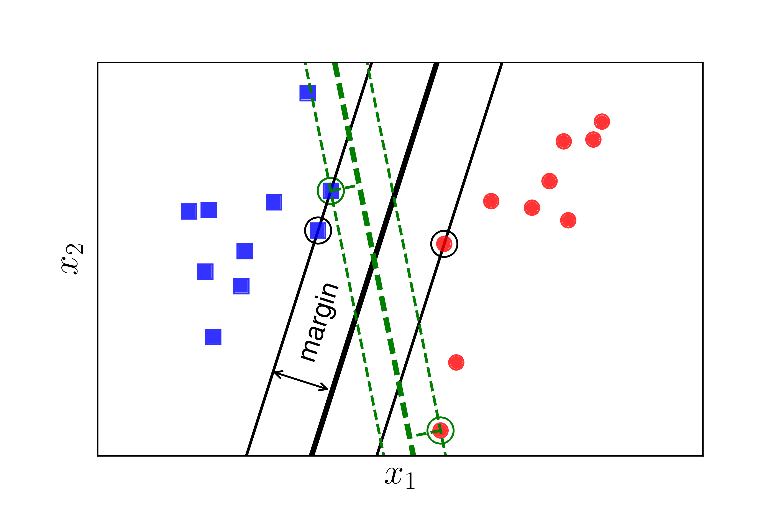
a, Linear SVM:

Trong bài toán phân loại 2 nhãn lớp, có thể tồn tại vô số siêu phẳng phân chia hai lớp, vậy câu hỏi đặt ra là siêu phẳng nào phân chia hai lớp tốt nhất theo một tiêu chuẩn nào đó



Nguồn : https://machinelearningcoban.com/2017/04/09/smv/

SVM được ra đời nhằm giải quyết câu hỏi trên, bằng cách xây dựng bài toán tối ưu sao cho tìm được đường phân chia có lề (margin) lớn nhất. Lề (margin) được định nghĩa là khoảng cách từ điểm gần nhất của mỗi lớp tới siêu phẳng phân chia và khoảng cách này là bằng nhau đối với 2 lớp



Nguồn : https://machinelearningcoban.com/2017/04/09/smv/

Bài toán tối ưu cho SVM được định nghĩa như sau:

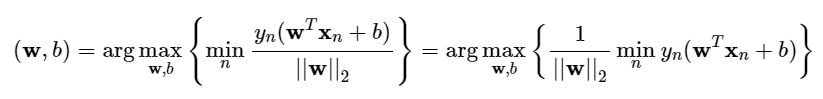
Giả sử rằng các cặp dữ liệu của tập huấn luyện là (x1, y1), (x2, y2), … , (xn, yn) với vector xi ∈ R­d là đầu vào của một điểm dữ liệu và yi là nhãn của điểm dữ liệu đó. d là số chiều của dữ liệu và N là số điểm dữ liệu. Giả sử rằng nhãn của mỗi điểm dữ liệu được xác định bởi yi = 1 (class 1) hoặc yi = -1 (class 0)

Xét siêu phẳng phân chia 2 lớp có dạng: wTx + b = 0 trong đó w ∈ Rd và b ∈ R là các hệ số của siêu phẳng chúng ta cần tìm. Với cặp dữ liệu (xn, yn) bất kỳ, khoảng cách từ điểm đó đến siêu phẳng được xác định bởi công thức:

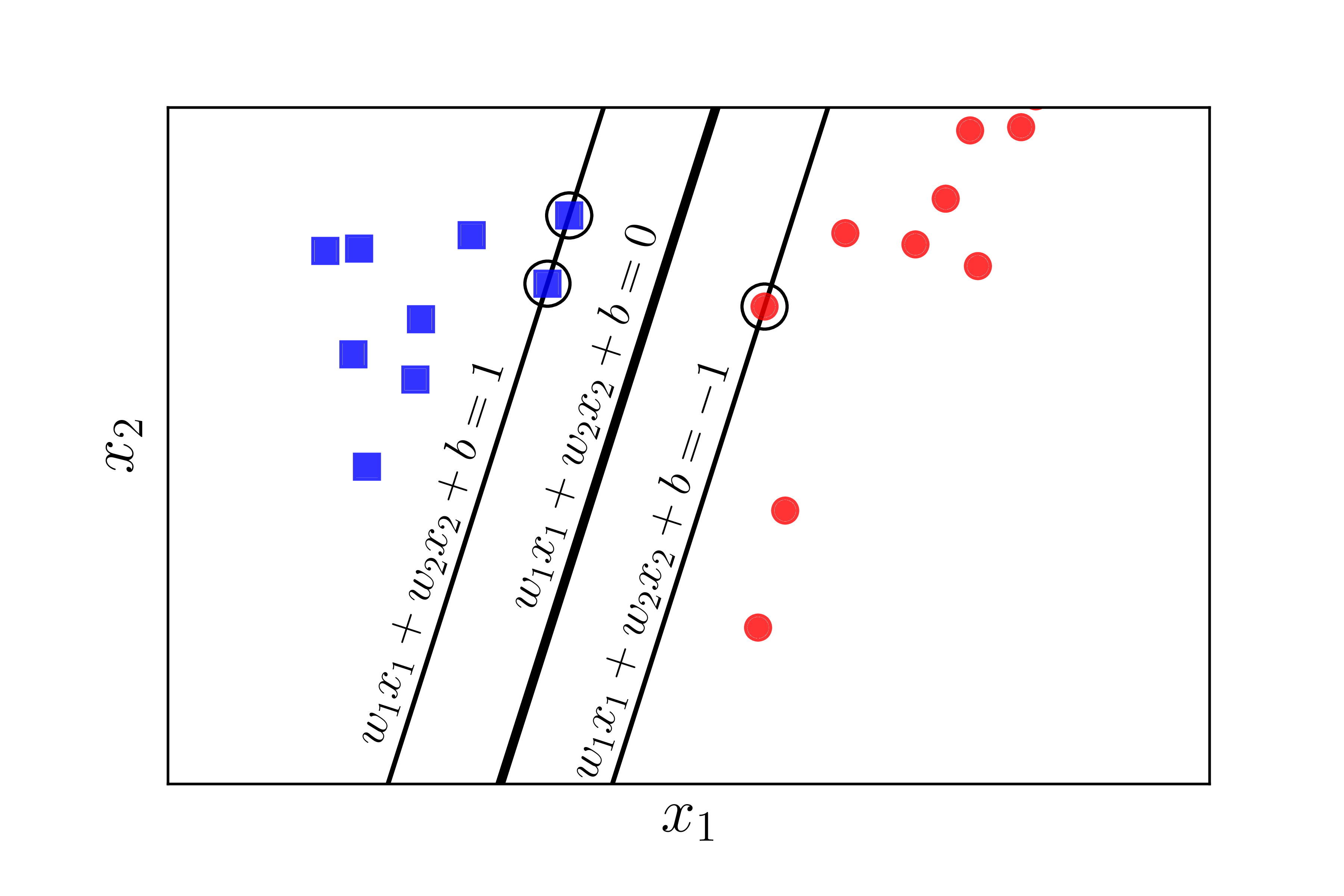
Với cách tính khoảng cách như trên, margin được tính là khoảng cách gần nhất từ 1 điểm tới siêu phẳng phân cách (bất kể điểm nào trong 2 lớp)

margin = min

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



Giả sử yn(wTxn + b) = 1 với những điểm nằm gần siêu phẳng phân chia nhất như hình vẽ trong không gian 2 chiều dưới đây

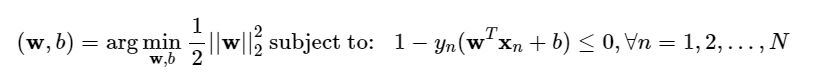


Nguồn : https://machinelearningcoban.com/2017/04/09/smv/

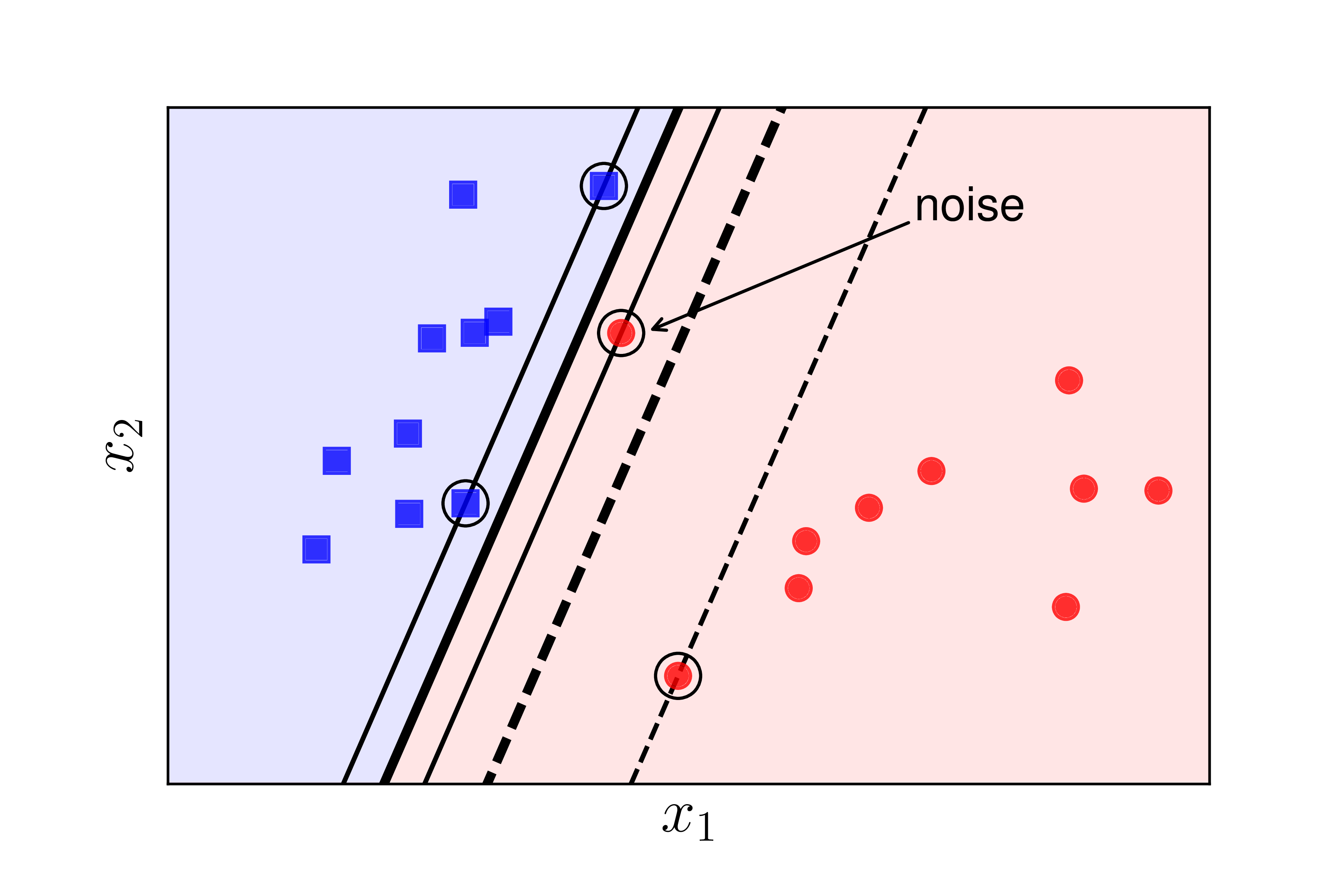
Như vậy với mọi n, ta có:

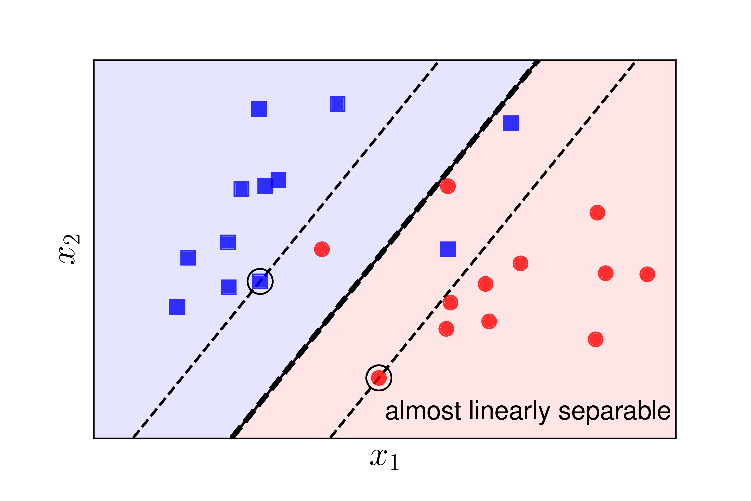
yn(wTxn + b) ≥ 1

Ta đưa bài toán tối ưu ban đầu về bài toán tối ưu có ràng buộc sau đây:



Đây là bài toán lề cứng (hard-margin) SVM. Tuy nhiên, trong thực tế, tập dữ liệu chúng ta sử dụng thường không phân biệt tuyến tính giữa 2 lớp hoặc có chứa các điểm dữ liệu nhiễu, khi đó bài toán tối ưu trên trở nên vô nghiệm. Nếu ta chịu hy sinh một số điểm dữ liệu nằm ở trong khoảng lề (margin), ta vẫn có thể tạo được một siêu phẳng phân chia khá tốt như ví dụ trong hình vẽ dưới đây





Nguồn : https://machinelearningcoban.com/2017/04/09/smv/

Dựa trên quan sát trên, ta thu được một biến thể của Hard Margin SVM có tên gọi là Soft Margin SVM.

Bài toán tối ưu cho Soft Margin SVM được xây dựng tương tự như bài toán tối ưu của Hard Margin SVM, với sự bổ sung của một đại lượng biểu diễn sự hy sinh ξn tương ứng với mỗi điểm dữ liệu xn trong toàn bộ dữ liệu huấn luyện. Với những điểm xn nằm trong vùng an toàn (nghĩa là nằm bên ngoài lề của hai lớp, ξn = 0. Ngược lại, với những điểm không nằm trong vùng an toàn , ta có ξn > 0.

Ta có bài toán tối ưu cho Soft Margin SVM như sau:

min

Với điều kiện:

Trong đó hằng số C được dùng để điều chỉnh tầm quan trọng giữa việc tối ưu margin và tối ưu sự hy sinh. Hằng số này được xác định từ trước hoặc có thể được xác định bằng việc sử dụng một tập tối ưu (validation set)

Việc giải bài toán tối ưu trên là không hề dễ dàng, do đó bài toán tối ưu trên thường được chuyển về bài toán đối ngẫu tương ứng để giải

Bài toán đối ngẫu (dual problem) có dạng:

max

Với điều kiện:

Bài toán đối ngẫu trên có thể được giải quyết bằng các phương pháp số học (để giải quyết bài toán tối ưu hàm lồi bậc hai có ràng buộc tuyến tính)

Sau khi giải được bài toán đối ngẫu tìm ra các , ta có thể tính ngược lại w và b của siêu phẳng cần tìm.

Quá trình phân loại một điểm dữ liệu mới được thực hiện bằng cách tính sign(wTx + b) và gán nhãn y= 1 nếu kết quả trả về lớn hớn 0 và y = -1 trong trường hợp ngược lại. Một đặc điểm quan trọng trong việc dự đoán nhãn cho một điểm dữ liệu mới là việc tính (wTx + b) chỉ phụ thuộc vào các điểm dữ liệu có hệ số tương ứng khác 0. Các ví dụ này được gọi là các **vector hỗ trợ (support vector)**. Do đó, ta có thể thấy rằng siêu phẳng phân tách được xác định dựa trên tập các **vector hỗ trợ**.

Ngoài góc nhìn bài toán tối ưu có ràng buộc, Soft Margin SVM có thể được đưa về bài toán tối ưu không ràng buộc dựa trên hinge loss:

min

b, Kernel SVM:

Đối với bài toán phân loại 2 lớp không thể phân tách bằng một siêu phẳng, Kernel SVM có thể được sử dụng để tìm một phép biến đổi sao cho dữ liệu ban đầu là không phân biệt tuyến tính được biến đổi sang không gian mới, với hy vọng dữ liệu sau khi được biến đổi sẽ trở nên phân biệt tuyến tính và từ đó có thể áp dụng các phương pháp Linear SVM như đã đề cập ở phần trước.

Như vậy, Kernel SVM tiến hành tìm một hàm biến đổi dữ liệu x từ không gian feature ban đầu thành dữ liệu trong một không gian mới bằng hàm số

Sau khi biến đổi x sang không gian mới, bài toán đối ngẫu trong Soft Margin SVM trở thành:

max

Với điều kiện:

Tuy nhiên, việc tính toán trực tiếp cho mỗi điểm dữ liệu có thể sẽ tốn rất nhiều bộ nhớ và thời gian vì số chiều của thường rất lớn, có thể là vô hạn. Thêm nữa, để tìm nhãn của một điểm dữ liệu mới x ta lại phải biến đổi x thông qua hàm rồi lấy tích vô hướng của nó với tất cả các với m trong tập hợp các support. Để tránh việc này, người ta áp dụng kĩ thuật **kernel trick** để tính dựa trên x và z mà không cần tính trực tiếp và

Trong bài toán này, chúng tôi thử nghiệm 3 hàm kernel để thử nghiệm cho bài toán Kernel SVM:

* Sigmoid kernel:

Trong đó và r là các hằng số

* Polynomial kernel:

Trong đó d là một hằng số dương để chỉ bậc của đa thức. d có thể không là số tự nhiên vì mục đích của chúng ta không phải bậc của đa thức mà là cách tính kernel. Polynomial kernel có thể dùng để mô tả hầu hết các đa thức có bậc không vượt quá d nếu d là một số tự nhiên

* Radial Basis Function (RBF):

## Áp dụng vào bài toán Rain in Australia

### Phân tích, tiền xử lý dữ liệu

#### Phân tích khám phá dữ liệu (EDA)

* ‘RainTomorrow’ là biến mục tiêu, có chứa những giá trị null cần loại bỏ
* Sau khi loại bỏ những mẫu có giá trị ‘RainTomorrow’ là null, dữ liệu còn 142193 mẫu, mỗi mẫu 23 thuộc tính. Từ đây, chúng em sử dụng dữ liệu này để xử lý tiếp
* Bảng thông tin tổng quan dữ liệu :

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Thuộc tính | Đếm các giá trị không là null | Kiểu dữ liệu |
| 1 | Date | 142193 | object |
| 2 | Location | 142193 | object |
| 3 | Mintemp | 141556 | float64 |
| 4 | MaxTemp | 141871 | float64 |
| 5 | Rainfall | 140787 | float64 |
| 6 | Evaporation | 81350 | float64 |
| 7 | Sunshine | 74377 | float64 |
| 8 | WindGustDir | 132863 | object |
| 9 | WindGustSpeed | 132923 | float64 |
| 10 | WindDir9am | 132180 | object |
| 11 | WindDir3pm | 138415 | object |
| 12 | WindSpeed9am | 140845 | float64 |
| 13 | WindSpeed3pm | 139563 | float64 |
| 14 | Humidity9am | 140419 | float64 |
| 15 | Humidity3pm | 138583 | float64 |
| 16 | Pressure9am | 128179 | float64 |
| 17 | Pressure3pm | 128212 | float64 |
| 18 | Cloud9am | 88563 | float64 |
| 19 | Cloud3pm | 85099 | float64 |
| 20 | Temp9am | 141289 | float64 |
| 21 | Temp3pm | 139467 | float64 |
| 22 | RainToday | 140787 | object |
| 23 | RainTomorrow | 142193 | object |

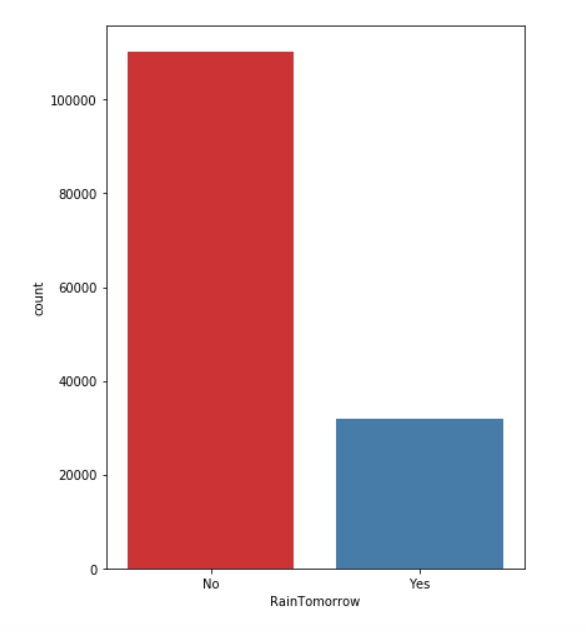
* Dữ liệu chứa cả biến phân loại và biến dạng số, một số thuộc tính có chứa giá trị thiếu

a, Phân tích đơn biến (RainTomorrow)

* Đã loại bỏ hết các giá trị bị thiếu
* Biến mục tiêu RainTomorrow nhận 2 giá trị : Yes và No

Yes : 31877

No : 110316



Phân phối biến RainTomorrow

b, Phân tích song biến

Khám phá các biến kiểu phân loại (Categorical Variable)

* Có 7 biến phân loại : Date (3436 nhãn), Location (49 nhãn), WindGustDir (17 nhãn), WindDir9am (17 nhãn), WinDir3pm (17 nhãn), RainToday (3 nhãn), RainTomorrow (2 nhãn)
* Có 2 biến phân loại nhị phân : RainToday và RainTomorrow
* RainTomorrow là biến mục tiêu
* Có 4 biến phân loại trong tập dữ liệu có chứa dữ liệu bị thiếu :

+ WindGustDir : 9330

+ WindDir9am : 10013

+ WindDir3pm : 3778

+ RainToday : 1406

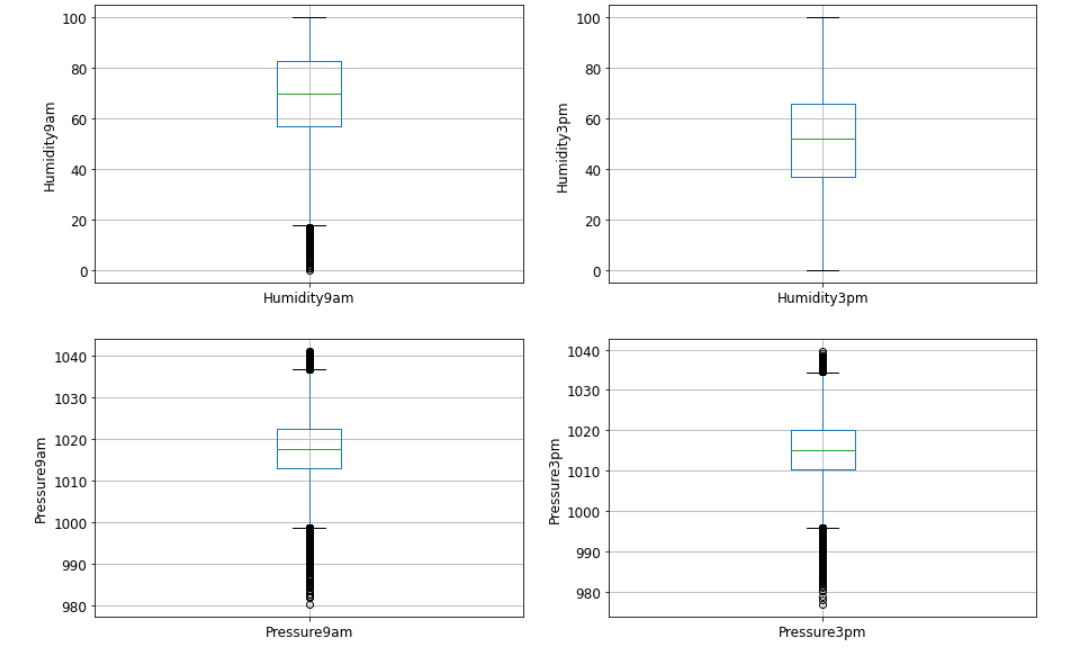
Khám phá các biến dạng số

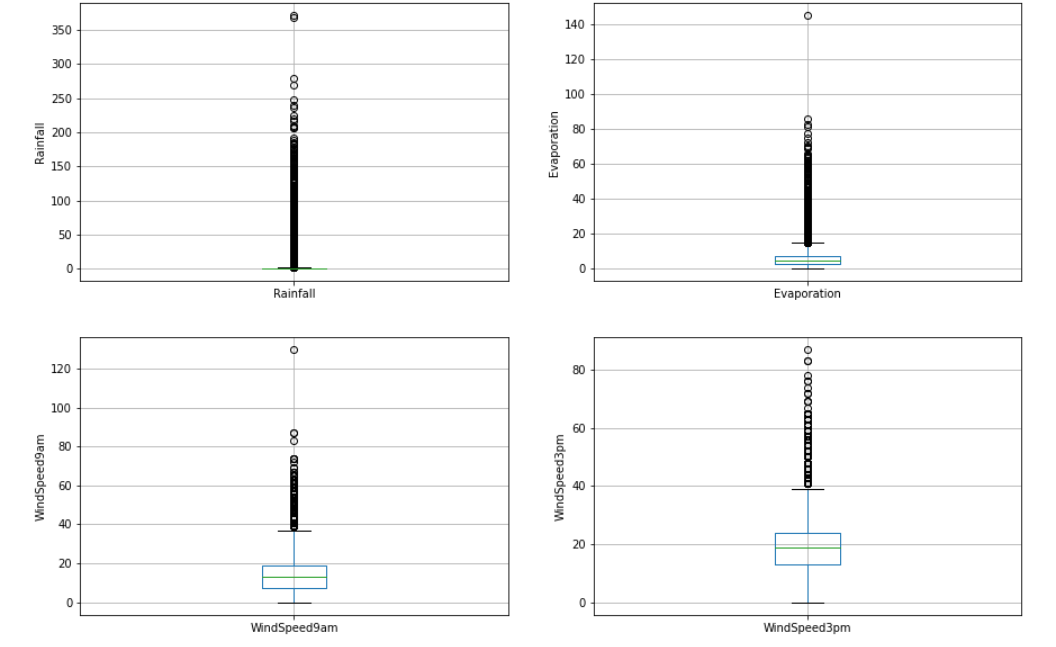
* Có 16 biến dạng số : MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity9am, Humidity3pm, Pressure9am, Pressure3pm, Cloud9am, Cloud3pm, Temp9am, Temp3pm
* Tất cả đều là biến liên tục
* Giá trị bị thiếu:

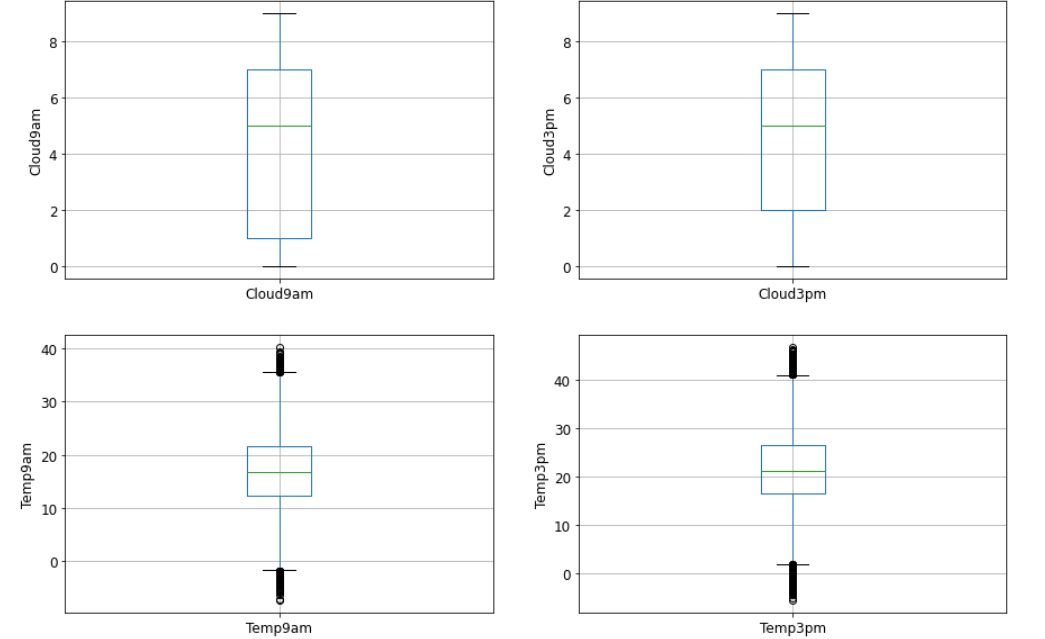
|  |  |  |
| --- | --- | --- |
| STT | Column | Số lượng missing values |
| 1 | MinTemp | 637 |
| 2 | MaxTemp | 322 |
| 3 | Rainfall | 1406 |
| 4 | Evaporation | 60843 |
| 5 | Sunshine | 67816 |
| 6 | WindGustSpeed | 9270 |
| 7 | WindSpeed9am | 1348 |
| 8 | WindSpeed3pm | 2630 |
| 9 | Humidity9am | 1774 |
| 10 | Humidity3pm | 3610 |
| 11 | Pressure9am | 14014 |
| 12 | Pressure3pm | 13981 |
| 13 | Cloud9am | 53657 |
| 14 | Cloud3pm | 57094 |
| 15 | Temp9am | 904 |
| 16 | Temp3pm | 2726 |

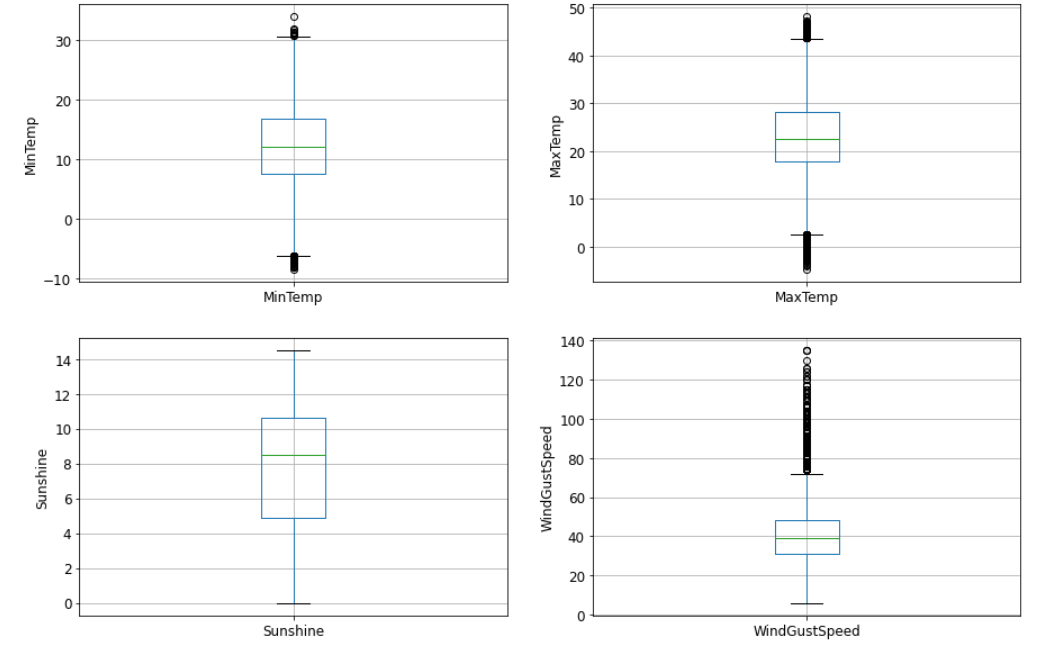
* Ngoại lệ:

Hiển thị ngoại lệ : Tôi dùng biểu đồ hộp (box plot) giúp biểu diễn các đại lượng quan trọng của dãy số như giá trị nhỏ nhất (min), giá trị lớn nhất (max), tứ phân vị (quartile), khoảng biến thiên tứ phân vị (Interquartile Range) một cách trực quan, dễ hiểu. Từ đó sẽ dự đoán các biến có thể chứa ngoại lệ



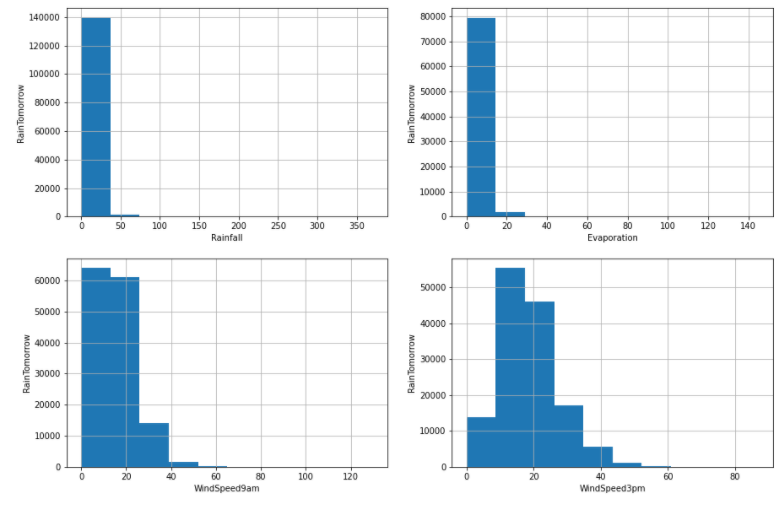


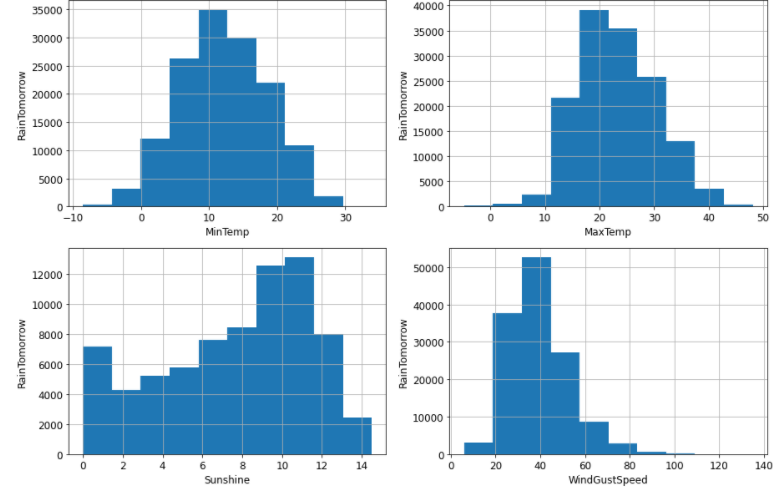


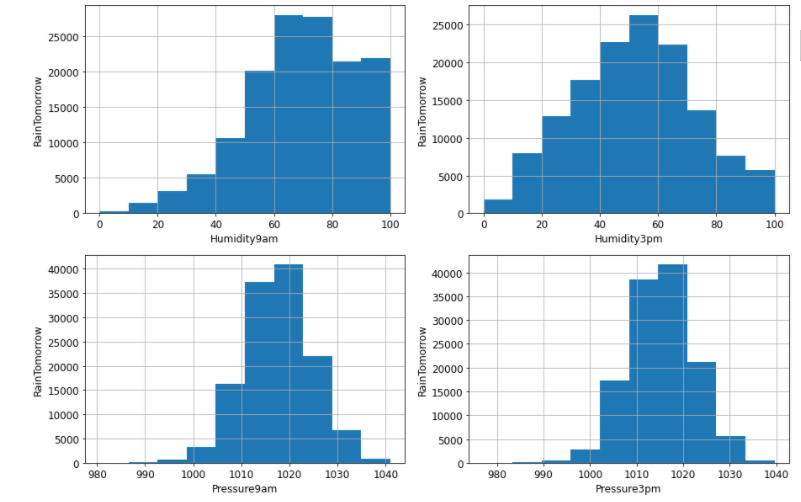


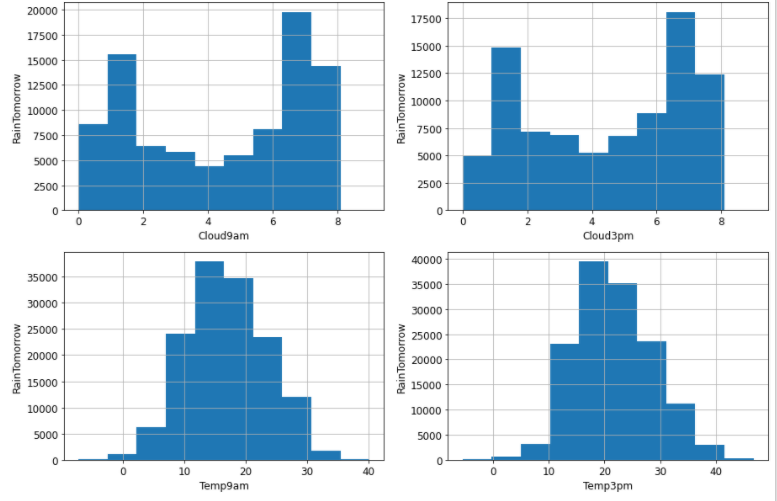
Biểu đồ hộp cho các biến

* Chúng ta có thể thấy :các cột Rainfall, Evaporation, WindSpeed9am, WindSpeed3pm có thể chứa các ngoại lệ
* Phân phối của các biến:







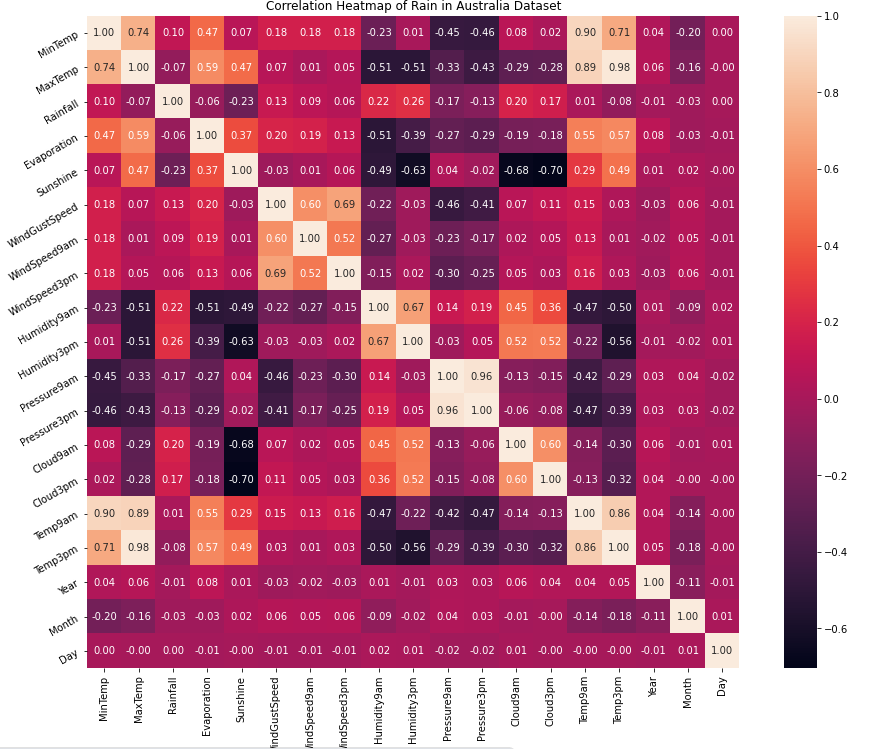


Phân phối các biến

* 4 biến : Rainfall, Evaporation, WindSpeed9am, WindSpeed3pm có phân phối lệch => sử dụng IQR (interquantile range) để tìm các ngoại lệ

c, Phân tích đa biến

* Bản đồ nhiệt (heatmap)



Bản đồ nhiệt

* Các cặp biến có hệ số tương quan cao :   
  + MinTemp và MaxTemp ( hệ số tương quan = 0.74)

+ MinTemp và Temp3pm ( hệ số tương quan = 0.71)

+ MinTemp và Temp9am ( hệ số tương quan = 0.90)

+ MaxTemp và Temp9am ( hệ số tương quan = 0.89)

+ MaxTemp và Temp3pm ( hệ số tương quan = 0.98)

+ WindGustSpeed và WindSpeed3pm ( hệ số tương quan = 0.69)

+ Pressure9am và Pressure3pm ( hệ số tương quan = 0.96)

+ Temp9am và Temp3pm ( hệ số tương quan = 0.86)

Các giả thiết: Phân phối từng nhãn của biến mục tiêu Rain Tomorrow theo giá trị của từng biến dạng số để dự đoán các biến quan trọng trong việc phân loại

|  |  |
| --- | --- |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |
| Chart, box and whisker chart  Description automatically generated | Chart, box and whisker chart  Description automatically generated |

* Các biến có thể quan trọng : Sunshine, Cloud9am, Cloud3pm, Humidity3pm, Humidity9am. Do median và Interquantile range của các biến này đối với 2 nhãn lớp có sự chênh lệch lớn

#### Feature Engineering

a, Biến ‘Date’

* Đây là thuộc tính dạng ngày, có chứa 3436 giá trị, rất lớn, cần xử lý
* Loại dữ liệu là object => chuyển về định dạng datetime để lấy ngày, tháng, năm
* Tạo ra 3 cột mới : Year, Month, Day tương ứng với năm, tháng, ngày được tách ra từ cột Date. Sau đó bỏ cột Date ra khỏi dữ liệu

b, Xử lý giá trị bị thiếu

* Đối với các biến dạng số:

+ Giả sử rằng dữ liệu bị thiếu hoàn toàn ngẫu nhiên.

+ Dữ liệu có những ngoại lệ , do đó chúng em sử dụng median impitation – thay thế các giá trị bị thiếu bằng giá trị trung vị của biến

+ Trung bị được tính trên tập đào tạo, sau đó thay thế các giá trị bị thiếu trên cả tập train và tập test bằng giá trị median để tránh overfitting

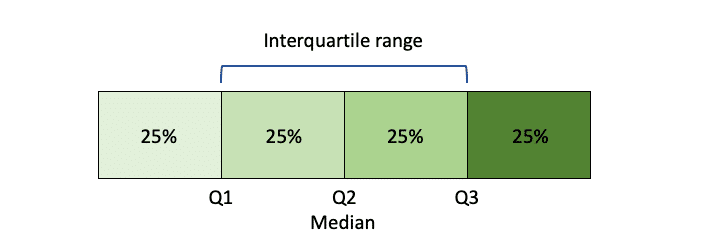
* Đối với các biến phân loại:

+ Đối với biến phân loại, chúng em sử dụng mode imputation, thay thế giá trị bị thiếu bằng giá trị có tần suất lớn nhất

+ Mode cũng được tính trên tập đào tạo, sau đó thay thế các giá trị bị thiếu trên cả tập đào tạo và tập test để tránh overfitting

c, Xử lý ngoại lệ đối với biến dạng số

* Theo EDA bên trên, có 4 biến dạng số có thể chứa ngoại lệ : Rainfall, Evaporation, WindSpeed9am, WindSpeed3pm
* Chúng em sử dụng interquantile range để tìm ngoại lệ
* Interquartile range (IQR) là thước đo sự phân tán của số liệu mẫu hoặc phân phối, thuật ngữ này được định nghãi như sự chênh lệch giữa giá trị cao nhất và thấp nhất của các tứ phân vị, và vì vậy nó chứa 50% giá trị trung tâm các quan sát của các biến số liên quan.



Nguồn : https://www.scribbr.com/statistics/interquartile-range/

IQR được tính bằng tứ phân vị thứ ba trừ tứ phân vị thứ nhất

IQR = Q3 – Q1

* Các điểm dữ liệu nằm ngoài vùng (Q1 – 3\*IQR, Q3 + 3\*IQR) là các ngoại lệ, bị loại bỏ khỏi dữ liệu

d, Mã hóa các biến phân loại

* Biến RainToday là biến nhận 2 giá trị Yes, No. Do đó chúng em dùng mã hóa nhị phân đối với biến này.
* Với các biến phân loại còn lại : Location, WindGustDir, WindDir9am, WindDir3pm chúng em dùng One Hot Encoding

e, Chuẩn hóa dữ liệu

* Sử dụng MinMaxScaler của thư viện sklearn.preprocessing để chuẩn hóa dữ liệu

### Kết quả các thí nghiệm đánh giá hiệu năng hệ thống

#### Kết quả phân loại

a, Hồi quy logistic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | support |
| 0 | 0.87 | 0.95 | 0.91 | 22067 |
| 1 | 0.74 | 0.52 | 0.61 | 6372 |
| accuracy |  |  | 0.85 | 28439 |
| Macro avg | 0.80 | 0.73 | 0.76 | 28439 |
| Weighted avg | 0.84 | 0.85 | 0.84 | 28439 |

b, SVM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | support |
| 0 | 0.87 | 0.96 | 0.91 | 22067 |
| 1 | 0.79 | 0.50 | 0.61 | 6372 |
| accuracy |  |  | 0.86 | 28439 |
| Macro avg | 0.83 | 0.73 | 0.76 | 28439 |
| Weighted avg | 0.85 | 0.86 | 0.85 | 28439 |

c, Cây quyết định

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | support |
| 0 | 0.86 | 0.95 | 0.90 | 22067 |
| 1 | 0.73 | 0.46 | 0.57 | 6372 |
| accuracy |  |  | 0.84 | 28439 |
| Macro avg | 0.79 | 0.71 | 0.73 | 28439 |
| Weighted avg | 0.83 | 0.84 | 0.83 | 28439 |

#### Kết quả gridsearch

Thực hiện chia tập dữ liệu thành 3 tập: training set, validation set và test set với tỉ lệ tương ứng 3:1:1

Hồi quy logistic:

* Tôi tìm kiếm giá trị tham số C cho kết quả tốt nhất trên tập validation set. Giá trị C được tìm kiếm ngẫu nhiên trong khoảng 10-4 đến 104 và tìm được giá trị C = 0.5 cho kết quả tốt nhất trên tập validation (Accuracy: 84.76%)
* Kết quả trên tập validation set của các giá trị C khác nhau được ghi lại trong bảng dưới đây

|  |  |  |
| --- | --- | --- |
| C | Training score | Validation score |
| 0.0001 | 77.537 | 77.746 |
| 0.0005 | 80.881 | 81.034 |
| 0.001 | 81.786 | 81.658 |
| 0.005 | 83.682 | 83.653 |
| 0.01 | 84.068 | 84.071 |
| 0.05 | 84.560 | 84.630 |
| 0.1 | 84.600 | 84.739 |
| **0.5** | **84.756** | **84.743** |
| 1 | 84.768 | 84.743 |
| 10 | 84.807 | 84.743 |
| 100 | 84.800 | 84.743 |
| 1000 | 84.803 | 84.743 |
| 10000 | 84.803 | 84.743 |



Từ hình vẽ trên ta có thể thấy với giá trị C quá nhỏ mô hình gặp phải hiện tượng overfit, sau khi tăng giá trị C lên ngưỡng nhất định (0.5) thì mô hình bắt đầu bão hòa và không có sự cải thiện rõ rệt về hiệu năng

SVM:

a, Linear SVM

* Đối với mô hình LinearSVM, chúng tôi tiến hành tìm kiếm các giá trị của tham số C ngẫu nhiên trong khoảng từ 10-4 đến 104 và tìm được giá trị C = 0.05 đạt kết quả tốt nhất trên tập validation (Accuracy: 84.68)
* Kết quả trên tập validation set của các giá trị C khác nhau:

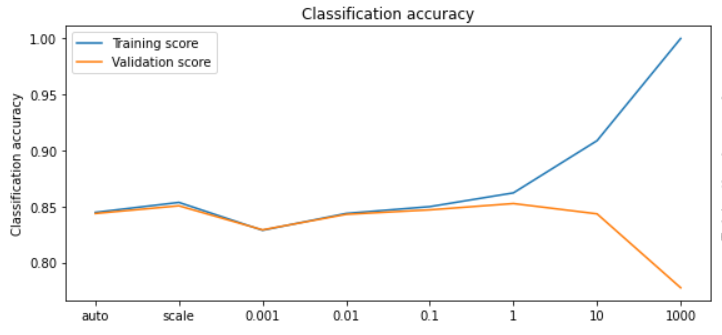
|  |  |  |
| --- | --- | --- |
| C | Training score | Validation score |
| 0.0001 | 81.487 | 81.399 |
| 0.0005 | 83.486 | 81.354 |
| 0.001 | 83.987 | 83.904 |
| 0.005 | 84.520 | 84.510 |
| 0.01 | 84.625 | 84.616 |
| **0.05** | **84.742** | **84.682** |
| 0.1 | 84.788 | 84.673 |
| 0.5 | 84.780 | 84.625 |
| 1 | 84.786 | 84.625 |
| 10 | 84.785 | 84.590 |
| 100 | 84.790 | 84.638 |
| 1000 | 78.468 | 78.314 |
| 10000 | 81.235 | 81.249 |

* Từ bảng kết quả trên ta có thể thấy với giá trị C nhỏ mô hình bị underfit, sau khi tăng giá trị C hiệu năng của mô hình được cải thiện đến 1 ngưỡng (C= 0.1) thì mô hình có dấu hiệu overfit do training score tiếp tục tăng nhưng validation score có dấu hiệu giảm

b, Kernel SVM:

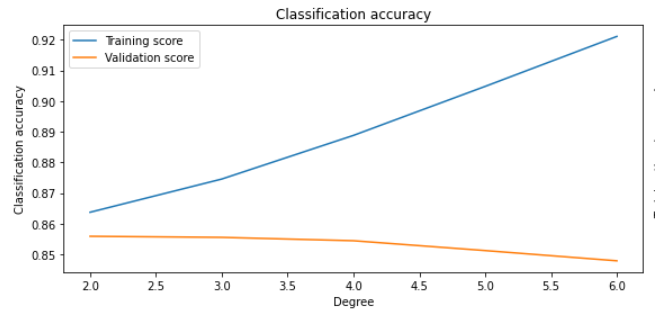
* Đối với Kernel SVM, chúng tôi thử nghiệm 3 hàm kernel sigmoid, polynomial và rbf với các tham số khác nhau để tìm ra tập tham số tốt nhất trên tập validation
* Mô hình cho kết quả tốt nhất trên tập validation là Polynomial Kernel SVM (Validation accuracy: 85.79) với tập tham số C = 1, gamma = ‘scale’ và degree = 2. Kernel SVM với hàm kernel rbf cho kết quả cạnh tranh với hàm kernel Polynomial với kết quả tốt nhất trên tập validatoin đạt 85.6. Trong 3 hàm kernel sigmoid cho kết quả thấp nhất với kết quả tốt nhất trên tập validation chỉ đạt 79.04
* Các tham số có ảnh hưởng đến hiệu năng của các mô hình kernel SVM bao gồm: giá trị điều chỉnh C và , bậc d đối với kernel polynomial
  + Kết quả tìm kiếm giá trị tham số đối với kernel rbf. Có thể thấy khi nhỏ mô hình ở trong trạng trái underfit, sau khi tăng giá trị hiệu quả mô hình được cải thiện tuy nhiên khi giá trị lớn mô hình bị overfit (Training score tại giá trị gần đạt ngưỡng tối đa)

|  |  |  |
| --- | --- | --- |
| **Gamma** | **Train\_score** | **Validation\_score** |
| auto | 0.844708 | 0.843699 |
| scale | 0.853587 | 0.850556 |
| 0.001 | 0.828731 | 0.829106 |
| 0.01 | 0.843763 | 0.842952 |
| 0.1 | 0.849818 | 0.846952 |
| 1 | 0.862092 | 0.852578 |
| 10 | 0.908651 | 0.843391 |
| 100 | 0.994725 | 0.779174 |
| 1000 | 0.999780 | 0.777460 |



* + Kết quả tìm kiếm giá trị tham số degree d đối với polynomial kernel

|  |  |  |
| --- | --- | --- |
| Degree | Training score | Validation score |
| 2 | 86.37 | 85.58 |
| 3 | 87.45 | 85.55 |
| 4 | 88.87 | 85.44 |
| 5 | 90.48 | 85.12 |
| 6 | 92.11 | 84.78 |



Decision Tree:

* Đối với Decision Tree, chúng tôi thử nghiệm các tham số: Max\_depth trên [2,10] để tìm ra độ sâu tốt nhất cho cây và tìm được 8 là độ sâu tốt nhất cho mô hình.Dưới đây là bảng kết quả:

|  |  |  |
| --- | --- | --- |
| Max depth | Train score | Validate score |
| 2 | 79.39 | 79.32 |
| 3 | 81.87 | 81.81 |
| 4 | 82.74 | 82.88 |
| 5 | 83.31 | 83.19 |
| 6 | 83.74 | 83.76 |
| 7 | 84.04 | 83.82 |
| **8** | **84.39** | **84.11** |
| 9 | 84.79 | 83.88 |
| 10 | 85.22 | 83.82 |



Như biểu đồ trên, chúng ta có thể dễ dàng nhận ra khi chiều cao cây >8, mô hình bị overfit.

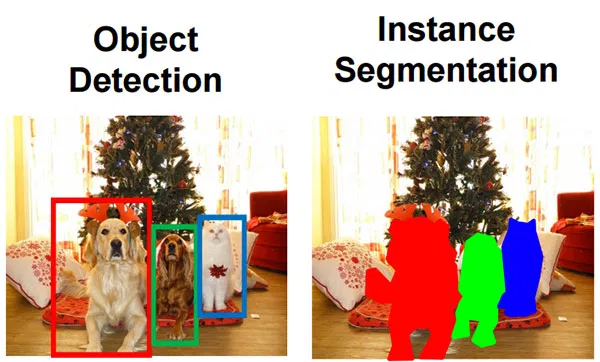
# Chương 2. Thị giác máy tính (computer vision) và áp dụng

## 2.1. Phân đoạn ảnh (image segmentation)

### 2.1.1. Cơ sở lý thuyết

a, Giới thiệu bài toán

* Tên của lớp bài toán là Image Segmentation có nghĩa là phân đoạn hình ảnh, hàm ý rằng bài toán sẽ phân chia một hình ảnh thành nhiều vùng ảnh khác nhau. Phân đoạn ảnh cũng có chung mục tiêu như bài toán phát hiện đối tượng là phát hiện ra vùng ảnh chứa vật thể và gán nhãn phù hợp cho chúng. Tuy nhiên tiêu chuẩn về độ chính xác của Phân đoạn ảnh ở mức cao hơn so với Phát hiện đối tượng khi nó yêu cầu nhãn dự báo đúng tới từng pixel.
* Mặc dù Phân đoạn ảnh yêu cầu về mức độ chi tiết cao hơn nhưng bù lại thuật toán giúp ta hiểu được nội dung của một bức ảnh ở mức độ sâu hơn khi chúng ta biết được đồng thời: **Vị trí** của vật thể trong ảnh, **hình dạng** của vật thể và **từng pixel nào thuộc về vật thể nào**.



Nguồn: cs231n.stanford.edu

b, Đầu vào và đầu ra của bài toán phân đoạn ảnh

* Phân đoạn ảnh nếu được huấn luyện theo bài toán học có giám sát trong thị giác máy tính thì sẽ yêu cầu gán nhãn cho ảnh. Đầu của bài toán là một bức ảnh và đầu ra là một ma trận mask mà giá trị của từng pixel đã được gãn nhãn trên đó.

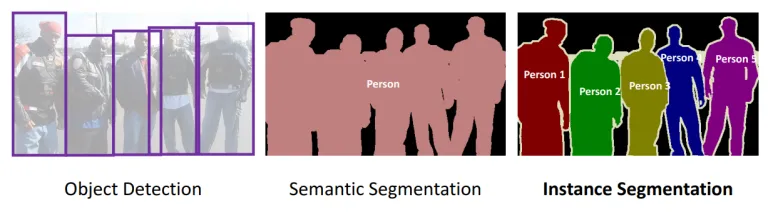


Nguồn : https://www.researchgate.net/

c, Phân loại bài toán phân đoạn ảnh

Bài toán phân đoạn ảnh được chia ra làm 2 loại:

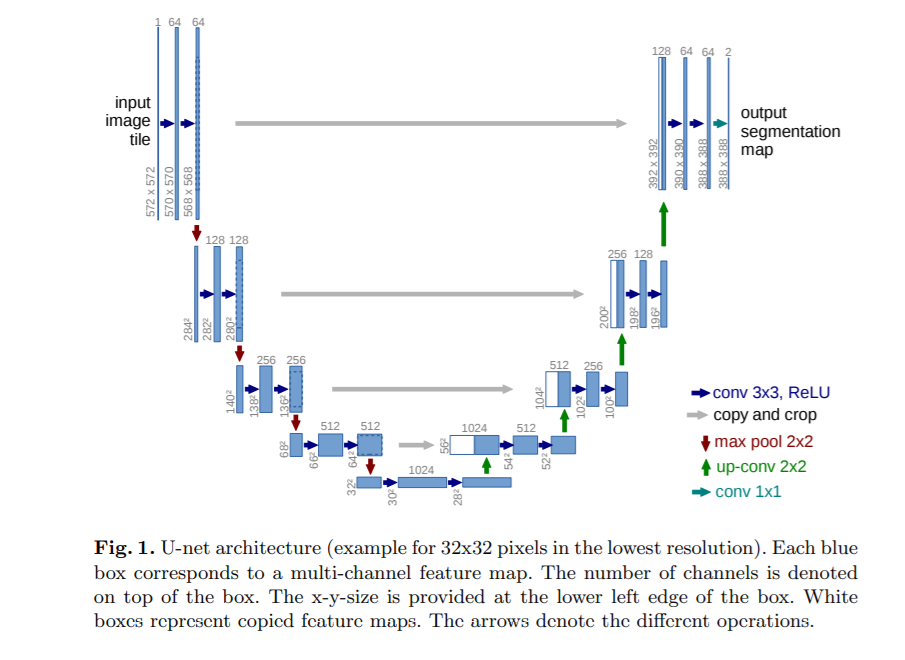
* **Semantic segmentation**: Thực hiện phân đoạn với từng lớp khác nhau, ví dụ: tất cả người là 1 lớp, tất cả ô tô là 1 lớp.
* **Instance segmentation**: Thực hiện phân đoạn với từng đối tượng trong một lớp. Ví dụ có 3 người trong ảnh thì sẽ có 3 vùng segment khác nhau cho mỗi người.



Nguồn : https://nttuan8.com/bai-12-image-segmentation-voi-u-net/

* Cần áp dụng kiểu segmentation nào thì phụ thuộc vào bài toán. Ví dụ: cần segment người trên đường cho ô tô tự lái, thì có thể dùng sematic segmentation vì không cần thiết phải phân biệt ai với ai, nhưng nếu cần theo dõi mọi hành vi của mọi người trên đường thì cần instance segmentation thì cần phân biệt mọi người với nhau.

d, Unet



* Kiến trúc bao gồm 2 phần :

+ bên trái gọi là contracting path. Nó bao gồm những sự áp dụng lặp lại của 2 lớp tích chập 3x3 (không padding), sau mỗi phần là 1 lớp max pooling 2x2 với stride 2 để downsampling. Vỡi mỗi bước downsampling, mạng nhân đôi feature chanels.

+ bên phải gọi là expansive path. Mỗi bước bao gồm việc upsampling feature map bởi 1 lớp up-convolution 2x2 làm giảm 1 nửa số lượng feature chanels, 1 sự kết nối với feature map đã được crop tương ứng bên phía contracting path, và 2 lớp tích chập 3x3, theo sau mỗi lớp là ReLU. Việc cropping là cần thiết vì sự mất mát pixel trong mỗi lớp tích chập.

Ở lớp cuối cùng, 1 lớp tích chập 1x1 được sử dụng để biến mỗi vector đặc trưng 64 chanels thành số lượng lớp mong muốn.

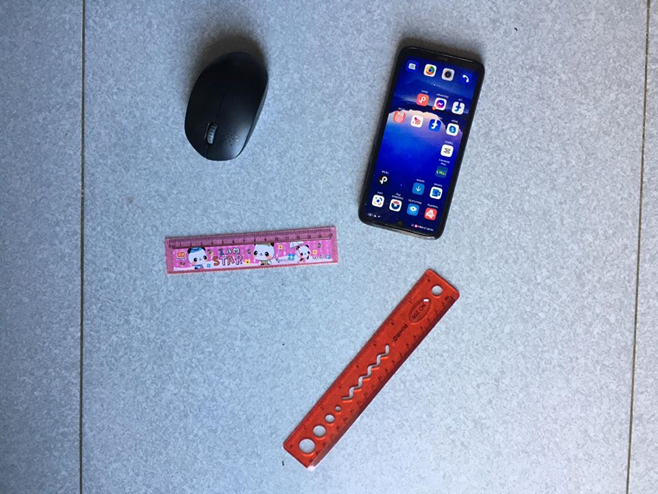
* Loss function : Vì bài toán là phân loại cho mỗi pixel nên loss function sẽ là tổng cross-entropy loss cho mỗi pixel trong toàn bộ bức ảnh.

### 2.1.2. Áp dụng vào bài toán đo kích thước vật thể

a, Dữ liệu:

* 5 lớp : chuột, điện thoại, thước, sách, nền
* Ảnh tự sinh : 200 ảnh



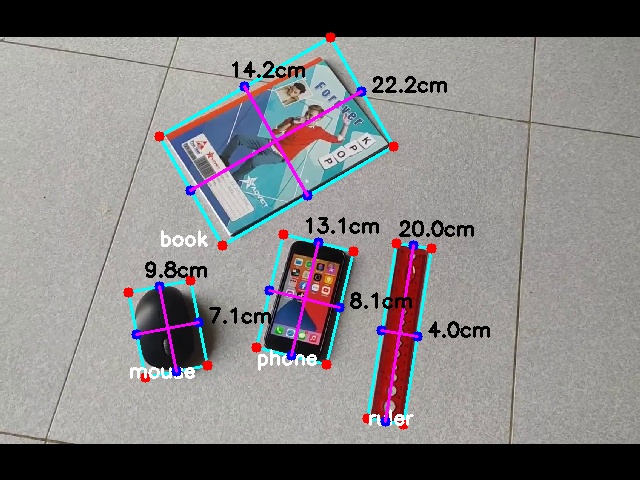
* Ảnh tự chụp : 70 ảnh



* Gán nhãn dữ liệu : tool labelme

b, Quá trình thực hiện

Sau khi đào tạo, đối với mỗi ảnh test, sẽ tạo ra được ảnh chứa các vùng ứng với các vật thể khác nhau, sẽ dụng xử lý ảnh trong thư viện của opencv của python để vẽ các box quanh các vật thể và từ đó dựa vào kích thước cố định của cái thước để tính toán kích thước của các vật thể còn lại trong ảnh



Kết quả

## 2.2. Phát hiện vật thể (Object Detection)

### 2.2.1. Cơ sở lý thuyết

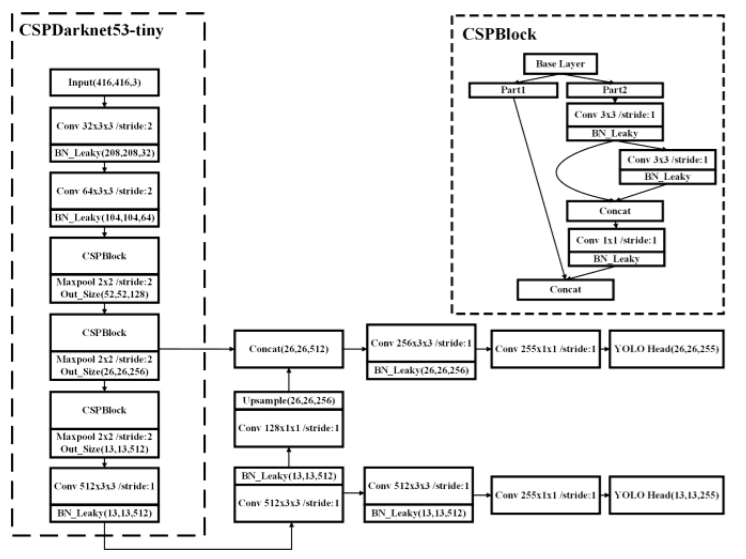
2.2.1.1. Giới thiệu bài toán phát hiện vật thể

* Object detection là bài toán phát hiện vật thể thuộc lớp nhất định trong ảnh.
* Đầu vào của bài toán phát hiện vật là một bức ảnh. Chúng ta không chỉ phải phân loại được object trên bức ảnh mà còn phải định vị được vị trí của đối tượng đó. Các phương pháp trước YOLOv1 thường sử dụng 2 bước: bước (1) thường sử dụng sliding window để lấy các vùng khác nhau của bức ảnh, hoặc sử dụng một thuật toán lựa chọn các vùng ứng viên (có thể chứa vật), tiếp theo đó, bước (2) sẽ phân loại các vị trí này xem vật đó thuộc lớp nào. Các cách tiếp cận này có nhược điểm là yêu cầu một lượng tính toán lớn, và bị phân nhỏ thành nhiều bước, khó có thể tối ưu về mặt tốc độ
* You only look once (YOLO) là một mô hình CNN để detect object mà một ưu điểm nổi trội là nhanh hơn nhiều so với những mô hình cũ. Một trong nhưng ưu điểm mà YOLO đem lại đó là chỉ sử dụng thông tin toàn bộ bức ảnh một lần và dự đoán toàn bộ object box chứa các đối tượng, mô hình được xây dựng theo kiểu end-to-end nên được huấn luyện hoàn toàn bằng gradient descent
* Dù đều được gọi là YOLO, các phiên bản của mô hình này đều có những cải tiến rất đáng kể sau mỗi phiên bản. Sau 3 phiên bản của tác giả chính [Joseph Redmon](https://pjreddie.com/) là YOLOv1 đến v3, tính đến thời điểm hiện tại có thêm một paper [YOLOv4](https://arxiv.org/pdf/2004.10934.pdf) của Alexey Bochkovskiy được dẫn link trực tiếp từ [repository gốc của Joseph Redmon](https://github.com/pjreddie/darknet) và [YOLOv5](https://github.com/ultralytics/yolov5) đang được phát triển

#### 2.2.1.2. YOLO

a, Yolotinyv4

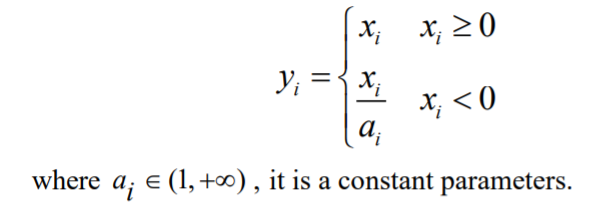
* YOLOv4-tiny được đề xuất dựa trên YOLOv4 để đơn giản cấu trúc mạng và giảm tham số, phù hợp phát triển trên các thiết bị mobile và nhúng.
* YOLO cho kết quả real-time khi chạy trên các nền tảng GPU cao cấp, cho nên không phù hợp để chạy real-time với các thiết bị mobiles và thiết bị nhúng do các thiết bị này bị hạn chế về khả năng tính toàn và bộ nhớ giới hạn. Để giải quyết vấn đề này, phương pháp lightweight object detection được đề xuất với kiến trúc mạng đơn giản hơn và ít tham số hơn. Do đó yêu cầu nguồn tính toán và bộ nhớ thấp hơn, tốc độ nhanh hơn tuy nhiên thì khả năng phát hiện đối tượng có độ chính xác thấp hơn.
* Một số phương pháp lightweight object detection được đề xuất như là MobileNet series, Squeezenet series, ShuffleNet series, lightweight YOLO series
* Các phương pháp lightweight YOLO được thiết kế dựa trên YOLO đầy đủ.
* YOLOv2 sử dụng mạng backbone là Darknet19, chứa 19 lớp tích chập và 6 lớp pooling. Phương pháp YOLOv2-tiny xóa các lớp tích chập còn 9 lớp để giảm độ phức tạp của mạng
* YOLOv3-tiny sử dụng 7 mạng tích chập và 6 lớp max pooling thay cho kiến trúc ResBlock trong mạng Darknet53 của YOLOv3. Nó cũng giảm đầu ra tự 3 dự đoán tỉ lệ 52x52, 26x26, 13x13 thành 2 dự đoán tỉ lệ 26x26, 13x13.
* YOLOv4-tiny cũng là 1 trong những phương pháp thuộc lightweight YOLO series. Nó sử dụng mạng backbone CSPDarknet53-tiny thay vì CSPDarknet53 của YOLOv4. The spatial pyramid pooling (SPP) và path aggregation network (PANet) cũng thay cho feature pyramid networks (FPN) để giảm thời gian detection. Bên cạnh đó, YOLOv4-tiny cũng sử dụng 2 dự đoán tỉ lệ 26x26 và 13x13.
* Kiến trúc mạng



Kiến trúc mạng Yolov4-tiny với ảnh đầu vào kích thước 416x416 và 80 classes.

Nguồn : https://arxiv.org/ftp/arxiv/papers/2011/2011.04244.pdf

* Backbone : CSPDarknet53-tiny
* CSPBlock module in cross stage partial network thay vì ResBlock module trong mạng phần dư. Nó có thể nâng cao khả năng học của mạng tích chập so với ResBlock module.
* Để quá trình tính toán đơn giản hơn, Yolov4-tiny sử dụng hàm kích hoạt LeakyReLU trong mạng CSPDarknet53-tiny mà không sử dụng Mish activation như ở Yolov4 :

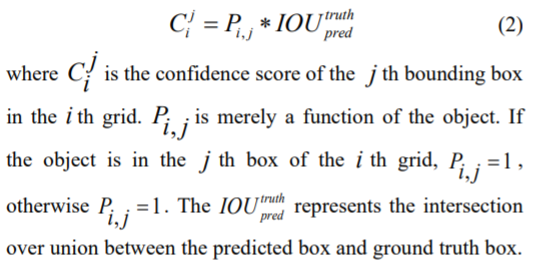


* Trong phần kết hợp đặc trưng, Yolov4-tiny sử dụng Feature pyramid network (FPN) để trích xuất đặc trưng với các tỉ lệ khác nhau để tăng tốc độ phát hiện đối tượng, mà không sử dụng the Spatial Pyramid Pooling (SPP) và Path Aggregation Network (PAN) trong phương pháp Yolov4. Yolov4-tiny sử dụng 2 bản đồ đặc trưng khác nhau là 13x13 và 26x26 để dự đoán kết quả phát hiện đối tượng.

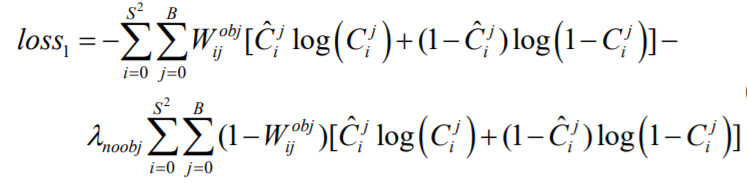
Quá trình dự đoán

* Quá trình dự đoán của phương pháp Yolov4-tiny giống phương pháp Yolov4. Đầu tiên nó cũng điều chỉnh kích thước tất cả ảnh đầu vào thành 1 kích thước cố định giống nhau. Thứ hai, ảnh đầu vào được chia thành các lưới có kích thước SxS, mỗi ô lưới sử dụng B bounding boxes để phát hiện đối tượng. Do đó, nó sinh ra ra SxSxB bounding boxes cho 1 ảnh đầu vào. Nếu trung tâm của vật thể nằm vào ô lưới nào thì những bounding boxes trong ô lưới đó sẽ dự đoán vật thể.
* Để giảm sự dư thừa của các bounding boxes trong quá trình dự đoán, ngưỡng tin cậy được đề xuất.

Độ tin cậy của bounding box được tính bằng :



* Loss function : giống Yolov4, gồm 3 phần:
  + - * + loss = loss1 + loss2 + loss3
    - The confidence loss function :

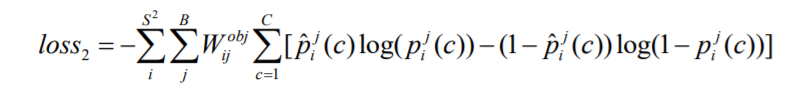


Trong đó : S2 là số lượng ô lưới trong ảnh đầu vào

* + - * + B là số lượng bounding box của 1 ô lưới
        + = 1 nếu bounding box thứ j của ô lưới thứ I chứa obj

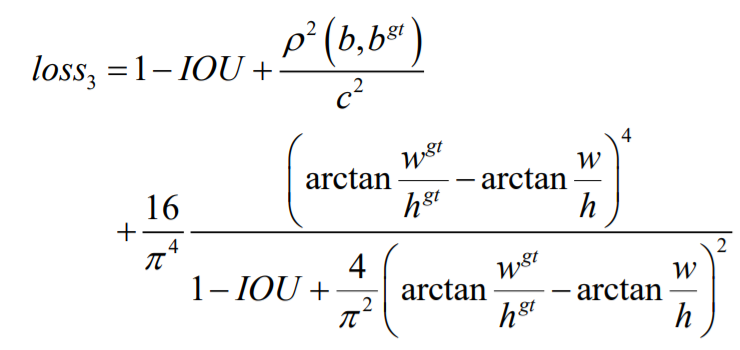
là độ tin cậy của box thật sự và box được dự đoán

* + - The classification loss function :



Trong đó là xác suất thực tế và xác suất dự đoán mà đối tượng thuộc về lớp phân loại c trong bounding box j của ô lưới i

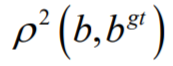
* + - The bounding box regression loss function :



Trong đó IOU là vùng chồng nhau giữa box dự đoán và box thực tế.

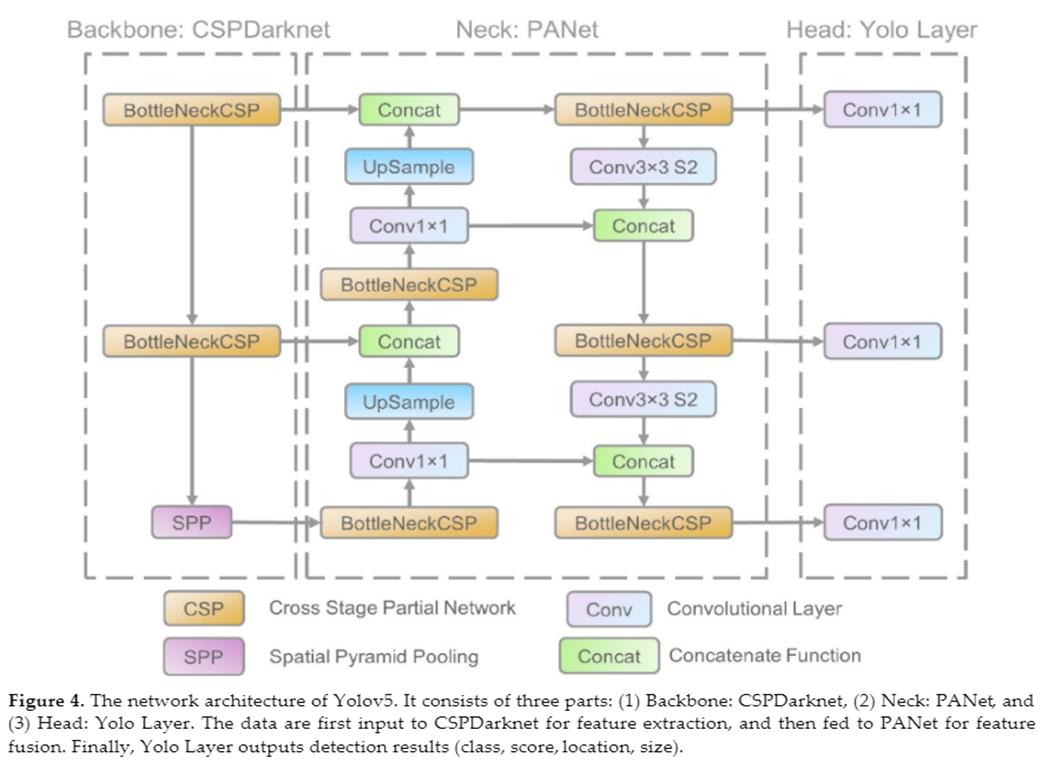
wgt  và hgt là chiều rộng và chiều cao thực tế của bounding box.

w và h là chiều rộng và chiều cao dự đoán của bounding box

 là khoảng cách Euclid giữa điểm trung tâm của bouding box dự đoán và bounding box thực tế.

c là khoảng cách đường chéo nhỏ nhất của box mà có thể chứa bounding box dự đoán và bounding box thực tế.

b, Yolov5



Nguồn : <https://machinelearningknowledge.ai/>

### 2.2.2. Áp dụng vào bài toán phát hiện xe trên cao tốc

a, Dữ liệu :

* images : gồm 8219 ảnh chụp các phương tiện giao thông đi lại trên cao tốc với 3 góc camera khác nhau (thu thập trên google)
* labels : được gán nhãn sẵn với format YOLO
* classes : auto, bus, car, lcv, motorcycle, multiaxle, tractor, truck

b, Kết quả

Yolov4-tiny

Chart

Description automatically generated

Kết quả train Yolotiny-v4

*Các điểm màu xanh biểu diễn loss qua mỗi vòng lặp*

*Các điểm màu đỏ biểu diễn giá trị mAP, sau 6000 batchs mAP đạt 79%*

Yolov5

A picture containing diagram

Description automatically generated

Hình 1 Confusion matrix

Chart, histogram

Description automatically generated

Hình 2 F1\_Curve

Chart, line chart

Description automatically generated

Hình 3 P\_Curve

Chart

Description automatically generated

Hình 4 PR\_Curve

Chart

Description automatically generated

Hình 5 R\_Curve

Graphical user interface, chart

Description automatically generated

Hình 6 Results

# Chương 3. Khó khăn và hướng phát triển

Trong quá trình thực hiện đồ án môn học, tôi gặp phải những khó khăn sau :

* Khó khăn trong việc tự học các kiến thức về Machine learning và Deep learning do đa số tài liệu là tiếng Anh, có những chỗ chưa được sáng tỏ và hiểu rõ
* Khó khăn trong việc chuẩn bị dữ liệu

Trong tương lai, tôi sẽ tìm hiểu các phương pháp, mô hình mới để bắt kịp với xu hướng để áp dụng vào các bài toán cụ thể

# Tài liệu tham khảo

1. <https://en.wikipedia.org/wiki/>
2. <https://www.coursera.org/learn/machine-learning>
3. <https://nttuan8.com/>
4. <https://arxiv.org/pdf/1505.04597v1.pdf>
5. <https://arxiv.org/ftp/arxiv/papers/2011/2011.04244.pdf>