***Day \_ 1* Chương 10 : MLPs**

1. **Perceptron**

* Là kiến trúc ANN đơn giản nhất ..nó dựa trên tế bào thần kinh TLU hoặc LTU
* Đầu vào là các số (**X**) liên kết với các trọng số (**w**)

Text

Description automatically generated

* Hàm **h(z)** được gọi là activation có thể là sigmoid,tanh,reuL,…..
* Hàm mất mát của **Perceptron**

J(**w**) = (tổng của các điểm bị lỗi)

Tối ưu hàm mất mát có thể sử dụng GD hoặc SGD :

* **W** = **w** + với ( là learning rate)
* Mỗi đầu ra là tuyến tính nên ko thể giải quyết các bài toán phức tạp nhưng với những bài toán tuyến tính thì nó chắc chắn hội tụ
* **Perceptron** ko xuất ra một lớp xác suất mà chỉ ra ngưỡng cứng nên ko linh hoạt như **Loggistic Regression** nên nó ko thể giải quyết được nhứng vấn đề nhỏ như **OR** hay **XOR**
* Diagram, line chart

  Description automatically generatedsử dụng **MLP ( multi-layer Percheptron)**

1. **Multi-layer Perceptron**

A picture containing scissors, tool

Description automatically generated Giả sử : X1 = 5

X2 = 6

X3 = 7

Layer 1:

z1 = 3x1 + 2x2 + x3 a1 =

z2 = -x1 + 4x2 - 3x3 a2 =

z3 = x1 + x2 + x3 a3 =

Layer 2:

……

Kết luận : mỗi lần train đưa ra dự đoán -> đo error -> backpropagation -> chỉnh sửa lại w , b

Quan trọng là khởi tạo w,b phải là ngẫu nhiên để mức học của mạng là lớn nhất , nếu khởi tạo toàn 0 thì có nhiều neuron đến mấy cũng như 0 .

* Ở đây sét activation ko phải hàm sgn vì đây là hàm phẳng mà GD thì ko đi được trên hàm phẳng nên phải chueyern sang hàm sigmoid …có một số hàm khác cũng có thể sử dụng (tanh, reLU,….)
* Vậy tại sao phải cần các hàm activation này ???? vì nếu ko có các hàm phi tuyến này thì 10 hàm tuyến tính tạo thành các mạng thì cx như 1 hàm mà thôi.

A/ **regression MLPS**

MLP có thể sử dụng cho regression nếu bạn muốn dự đoán 1 giá trị duy nhất

* Chỉ cần 1 output duy nhất .

Với multi regression -> cần nhiều neuron đầu ra hơn nữa.

Và thường chúng ta ko dùng hàm activation ở đầu ra nên có thể xuất ra bất kì giá trị nào. Còn nếu bạn muốn xuất ra giá trị dương thì có thể sử dụng hàm reLU

Về loss function thì thường là MSE nhưng bạn có thẻ sử dụng MAE hoặc Huber loss ( là kết hợp của MSE và MAE)

**Tổng kết 1 chút :**

*Input neuron :*  một cho mỗi input feature ( VD : 28x28 cho mnist )

*Hiden layer :* phụ thuộc vào từng vấn đề

*Neuron của mỗi layer :* cũng phụ thuộc vào từng vấn đề

*Output neuron :* 1 cho mỗi thứ cần dự đoán

*Hidden activation :* reLU

*Output activation :* ReLU cho dự đoán cần dương , loggistic/tanh cho dự đoán nhị phân

*Loss funtion :* MSE , MAE , Huber

B/ **Classification MLPs**

MLP cũng có thể sử dụng cho các bài toán phân loại

VD : với bài toán phân loại nhị phân chỉ cần 1 output neuron sử dụng activation là sigmoid hoặc tanh

Hoặc các bài multi- binary thì cần nhiều hơn 1 output

Hoặc các bài toán multi-class thì cần 1 activation như softmax

🡪 nên sử dụng loss funtion : **cross – entropy**