1. Neural Networks and Deep Learning

AI is the new Electricity

Electricity had once transformed countless industries:

AI will now bring about an equally big transformation.

+dự đoán giá nhà là model cơ bản của neural network với các yếu tố như sze,bedroom …..

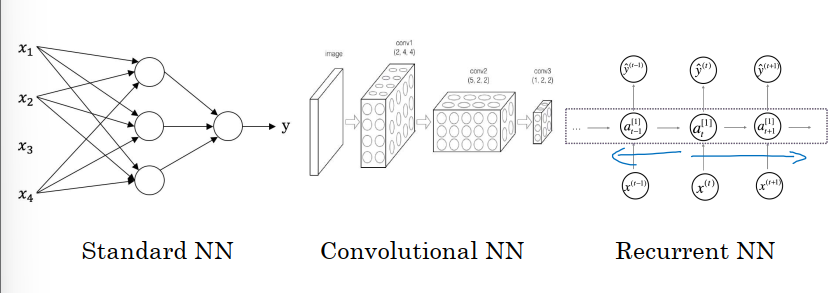
+supervised learning gồm xác định dữ liệu int và out :

+standal NN :real estate và qcao online

+CNN :photo

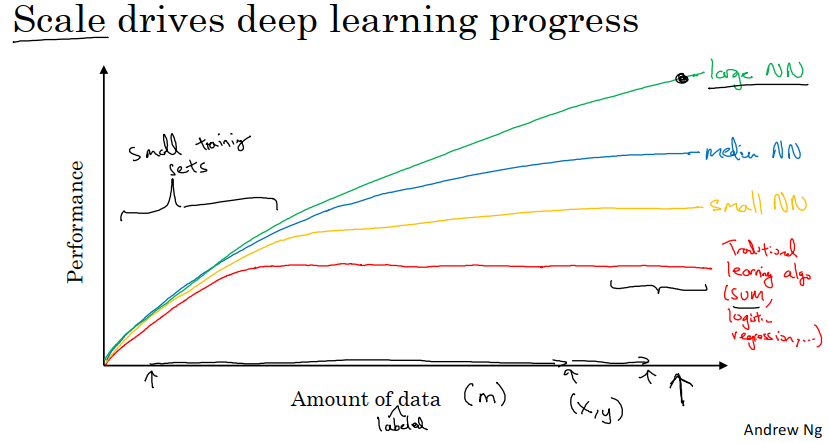
+RNN :nhận dạng speech và translation

+customer hybrid là tự động hóa



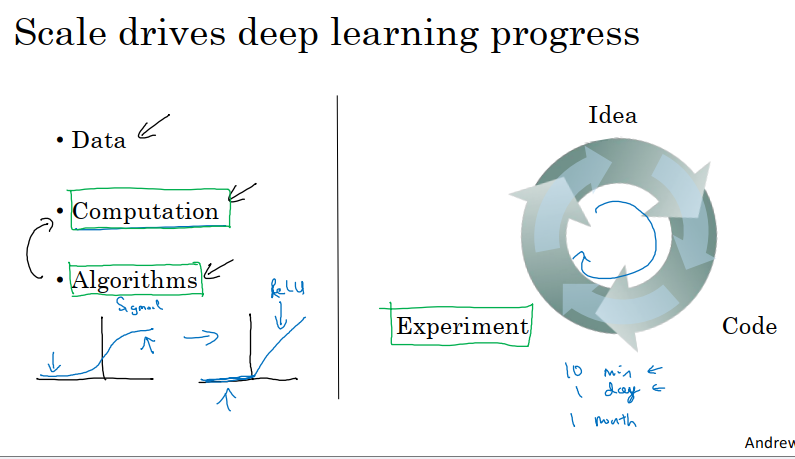
+data có cấu trúc là các thông số của 1 vật thể =>xử lý với thuật toán truyền thống

+data không cấu trúc là audio ,image,text => xử lý với deep



🡺deep hoạt động tốt với bộ dữ liệu càng lớn

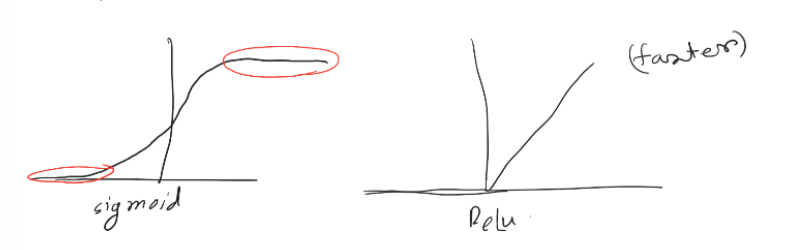
=>vòng lặp deep :data =>tính toán => thuật toán sigmoid,relu=>experiment



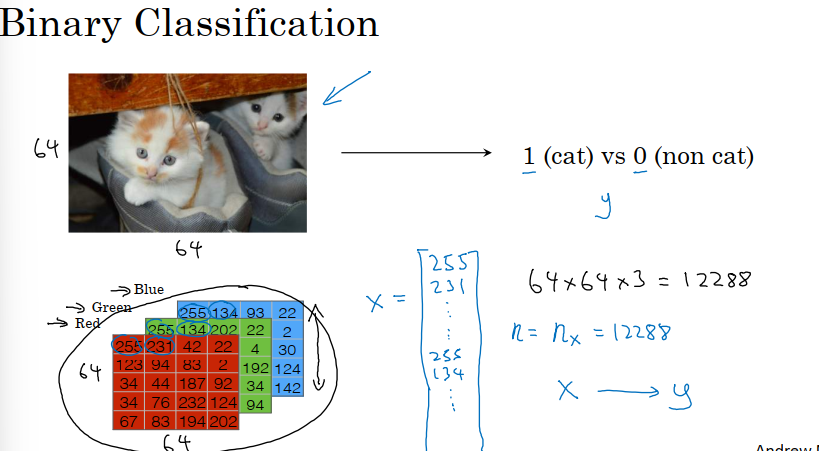
Deeplearning refer to train neural network

+mỗi neural hiểu hết các features từ data đầu vào chứ hong phải là 1 features riêng lẻ

+hàm sigmoid và relu :



Binary Classification



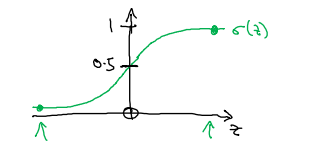
Nhận dạng data thuộc 1 trong 2 lớp mèo và không mèo :

+ảnh vào là matrix và qua mạng dự đoán ra kết quả

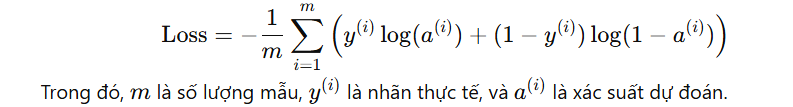
+shape của int và out lần lượt là (n,m) và (1,m)

Logistic Regression

Hàm sigmoid : chuyển đầu ra về giữa 0 và 1



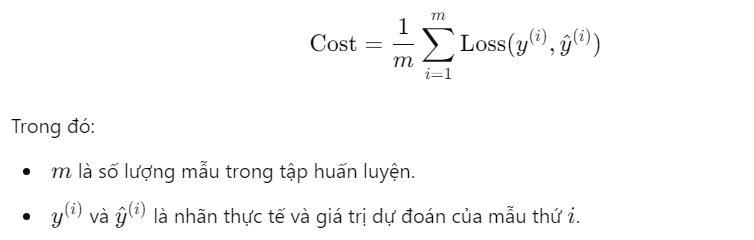
Hàm loss là đo lường sự khác biệt giữa dự đoán và thực tế



🡺-(thực x log ảo + (1-thực)x log(1-ảo)

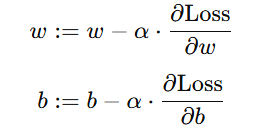
Tương tự thì y(i) = y và a(i) là y^

Hàm cost sẽ là tổng trung bình các hàm loss



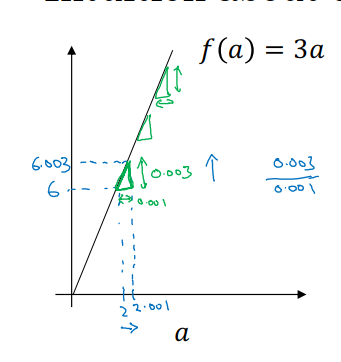
Gradient Descent

tìm các tham số w và b tối ưu bằng giảm dần hàm chi phí



learning rate :tốc độ học  α để điều chỉnh bước đi thuạt toán :w = w+ α

Derivatives

cho biết độ dốc của hàm số tại 1 điểm cụ thể ,thay đổi tốc độ theo biến thay đổi

**Computation Graph, Logistic Regression Gradient descent và m examples :**sẽ được cho các công thức và từ đó tính toán theo công thức đó

Vectorization :sử dụng matrix và vector để tăng tính toán neural ,sử dụng GPU

=>sử dụng để tối ưu hóa hơn là loop

+ Logistic regression is a binary classification algorithm ,let we have a image .

+overview

Mạng neural một lớp ẩn one hidden layer

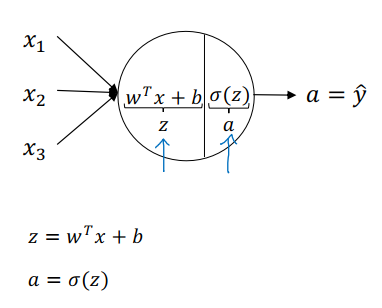
🡺gồm các lớp kết nối với nhau ,mỗi neuron nhận tín hiệu lớp trước ,xử lý và truyền tới lớp tiếp

=>lớp ẩn nằm giữa int và out giúp học đặc trưng phức tạp

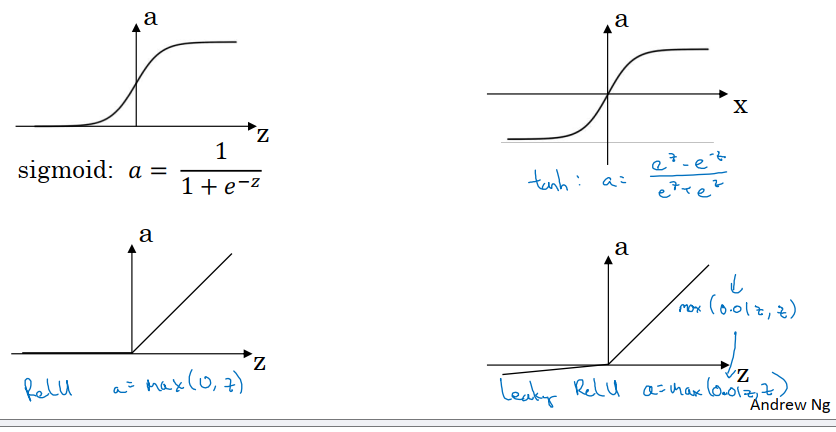
Tham số gồm w và b :tính bằng w = wx+b

+hàm kích hoạt activation function



🡺 

Hàm các hàm activation



+Hàm tanh (hyperbolic tangent) # sigmoid là nó đi từ -1 tới 1 ,đồ thị lại đi qua điểm 0 ,độ dốc lơn hơn sigmoid trong khoảng gần 0 học nhanh và tránh mất độ dốc

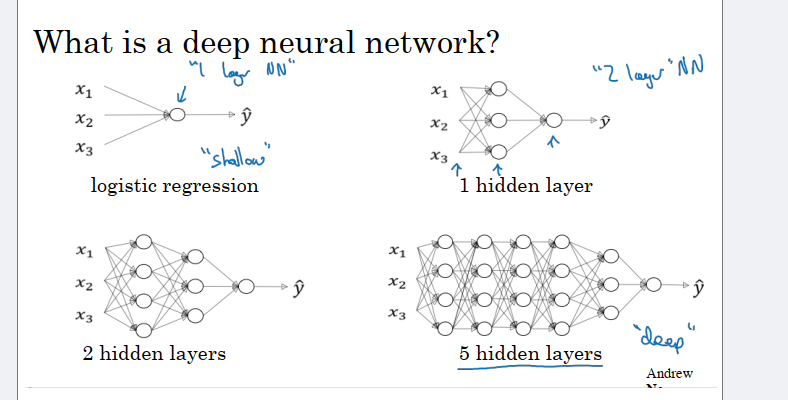
+hàm relu (rectifled linear unit ) giá trị = 0 khi z <0 và =1 khi z >0 ,không ảnh hưởng bởi gradient

* Sigmoid và tanh dùng truyền thống còn 2 còn lại sài tốc độ nhanh

Backpropagation:tính lan truyền ngược từ đầu ra về lại lớp ẩn

Random initialization:giúp các neuron học được các đặc trưng khác nhau thay vì khởi tạo bằng 0

Deep L-layer Neural network



Deep neural network là mạng sâu và có hơn 1 lớp ẩn ,1 lớp thì là shallow neurau network

+n là lớp

+ a là vector kích hoạt

+w:matrix trọng số

+b là vector bias

Tính bằng z = wa+b

Forward Propagation sẽ thông qua hàm z và a để tính ra đầu ra y

+khi mặt đưa vào tính toán đơn giản rồi đến phức tạp trong neural và đưa ra đầu ra

+ audio => low level features =>phonames => từ => câu

🡺mạng showllower network chỉ làm các bài nhỏ và ẩn nhân theo cấp số nhanh còn neuron học sau thì tính toán phức tạp và ẩn tăng theo logarit

Improving Deep Neural Networks\_ Hyperparameter tuning, Regularization and Optimization

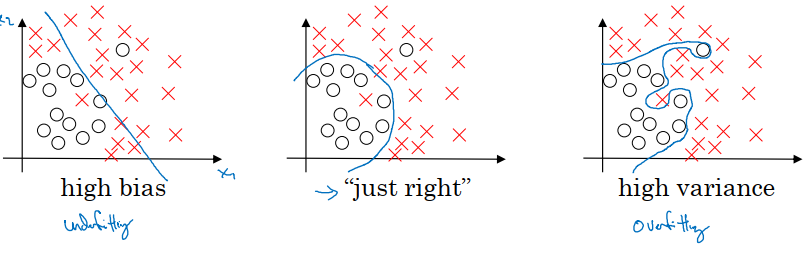
Training set là chia ra theo tập để traing

Dev

Test

Nếu train và test Mismatched chỉ cần có train và dev ,không có test cũng kg sao

Bias/Variance



Bias:độ sai lệch trung bình của mô hình dự đoán so với thực tế

Var:đô lệch chuẩn :nó sẽ nhạy cảm thích nghi tốt với tập dữ liệu mới

Underfiting :model đơn giản quá nên hong thể dự đoán chính xác

Overfitting :phức tạp quá mức làm cho model ,học tốt ở tập train nhưng kém ở tập test

Just right cân bằng để thỏa mãn hai over và under

Nhìn vào đường phân chia x và o

+Regularization giúp tránh overfitting và giúp cho machine học tốt ở tập data mới ,cách làm là thêm hệ số phạt lambda (λ),đưa các w kg cần về 0

+drop out là tắt đi 1 số neural giúp cho model bớt phụ thuộc vào 1 neural mà phân rải đều ra ,tương tự như regulaztion

+data augmentation là xoay lật,… để tăng cường data giúp model học được các data chưa có

+early stopping ;giúp cho model giảm quá khớp ,khi chạy 1 time thì lỗi trên train giảm đi còn trên tập dev thì tăng lên và nó dừng lại để giúp kg tăng nữa

+ normalize inputs là đưa tất cả các data quy về 1 data gốc để giúp dễ học hơn :ví dụ đưa km,mét quy về cm

+vanishing /exploding gradients

Van là khi w tăng lên nhanh ,gradient tăng dẫn đến exploding ,không converging hội tụ

Explo khi w giảm thì gradient giảm làm model hong thể học nhanh

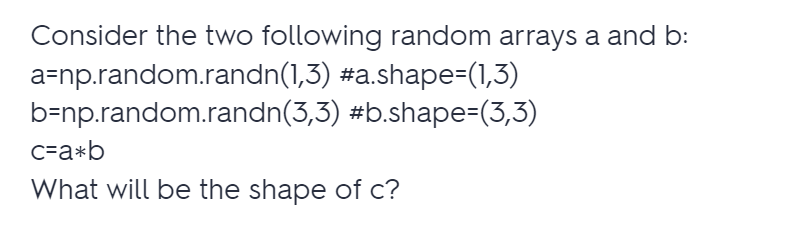
+gradient checking là kiểm tra xem gradient của loss có đúng hong ,tính tay và tnsh bằng analytical

* Các lưu ý :không dùng trong training ,xác định lỗi ,remember regulazation,không có dropout ,check random initialization
* Tính toán

Tính đưa array về a column vector bằng cách thêm 1 vào sau abc

+tính reshape của array thì làm sau cho các số trong reshape đó tích lại bằng với tích của các số trong mảng

* Cách tính reshape :

Nếu cho 2 mảng

Quy luật tính :mỗi 1 cột ứng với nhau là 1 chiều

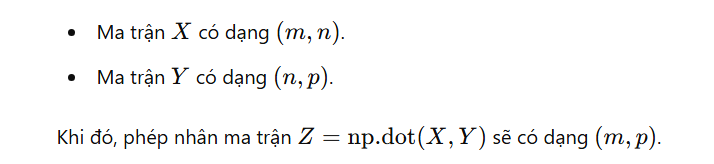
Nếu có 1 chiều có 1 thì được quyền nâng lên để bằng số còn lại của chiều đó

Nếu 2 mảng có cùng số chiều nhưng khác nhau về kích thước trừ số 1 thì loại

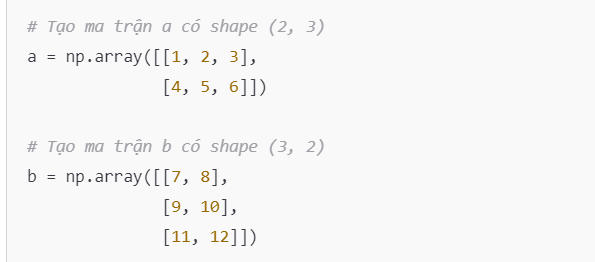
Trong bài trên chìu đầu là 1 3 và chiều 2 là 33

+quy tắc hàng ,cột

+Còn tính np.dot thì là số cột b2 bằng số hàng a1 thì sẽ lấy số hàng b1 ,cột a2



+còn a\*b là



Thì 2 cái buộc y chang nhau mới nhân được

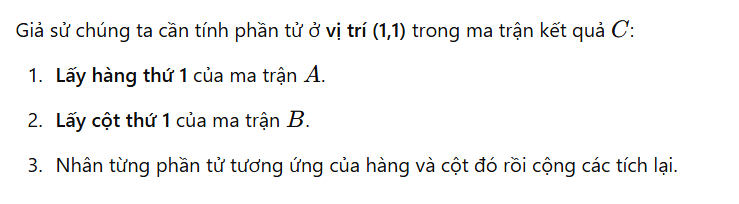
+Dimention of X :

M hàng và n cột nếu nó đứng và ngược lại nếu ngang

* Còn nếu tính dim với ảnh chuyển sang grayscale thì sẽ từ vector hàng chuyển sang cột đứng

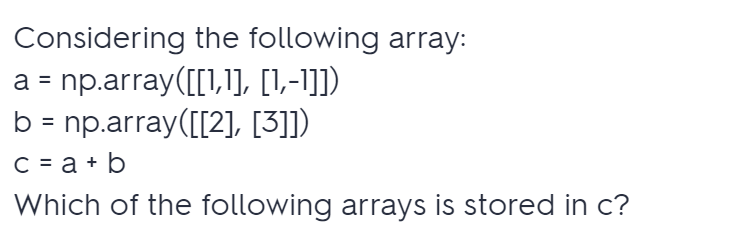
+mỗi phần tử sẽ nhân với nhau nếu a\*a

+quy tắc nhân a,b và b,c thì mới nhân đc



+tính vectorize thì là làm sao cho nó về tính như chiều shape

+có keepdims = true phải giữ 1 lại



Câu này ta lấy 1 ,1 +2 và 1,-1 +3

+tính equivalent output for c? thì cứ theo công thức C mà triển khai còn range là bằng shape

Trong logistic thì đầu ra tính bằng 

+-W\_1 là phải có rows transpose và –W[4]\_3 là 4 layer và 3 neuron

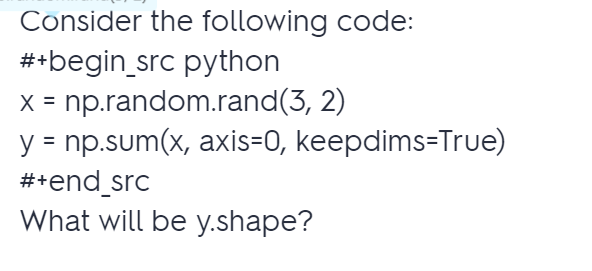
+tanh tập trung vào data ,giúp simple learn next step ,có slope ~ 0 và là version của sigmoid

+a[3](4)\_2 là neuron 2 của lớp 3 áp cho next 4

🡺quy tắc:[] là layer ,() next,\_số là neuron

+hidden thì sài tanh còn đầu ra làm sigmoid vì tanh ngoại lệ kg dùng với đầu ra

+hàm kích hoạt sử dụng trong layer của 1 layer và nhiều layer đều như nhau

+

Áp dụng dim đứng mn và ngang nm và đi tìm n ,trong đó axis 0 ngang 1 dọc ,keepdims thì có 1

+Relu được chọn khi bạn không biết chọn hàm nào cho ẩn

+cache được dùng trong lan ngược thuận

+hypermeter là size of ẩn ,số ẩn trong network,số interation và learning rate

+trong backpropagation thì dùng gradient để compute new w và b, (không có change)

+trong back thì dùng

+vector dùng để tính da trong back

+trong L layer thì mặc dù vector tính hết nhưng cũng cần vòng lặp trong tính lan truyền thuận propagation

+vector có thể tính training example same time mà kg cần loop

+cho w,b,I,g tính propagation của L layere bằng cthuc chạy 1 tới L+1

Z= w.a(i-1) +b

A = G