Tổng Quan

Ý tưởng của neuron network và supervised learning

là tìm được hàm maping input X và output Y

với các nút dần được tối ưu nhờ gradient

ứng dụng

dự đoán giá nhà, click quảng cáo standard neuron network

hình ảnh, photo tagging CNN

Audio (sequence data) RNN

dịch, (sequence data) RNNs

xe tự lái Custom, Hybrid

dữ liệu có cấu trúc vs không cấu trúc

có cấu trúc database SQL, mỗi feature đều có giá trị

phi cấu trúc database noSQL, hình ảnh, âm thanh. text

Binary Classification

n = nx số chiều input x

(x, y) x thuộc Rnx (không gian nx = n chiều)

y thuộc {0,1}

m (traning example)

m = {(x(1),y(1)), (x(2),y(2)), ... (x(m),y(m))}

mtrain dùng để train

mtest dùng để test

Logistic Regression

dùng khi output của bài toán chỉ là 0 và 1

VD: nhận diện con mèo

xác suất = y^ = P(y=1 | x) xác xuất để y=1 với điều kiện x

x thuộc Rnx (x xếp theo chiều dọc, vector)

w thuộc Rnx  (w xếp theo chiều dọc, vector)

b thuộc R

xác xuất = y^ = sig(wTx + b)

với sig là hàm sigmoid

1 / (1 + e-1)

loss function (error function)

L(y^, y) tính độ lệch giữa train và mong muốn

tìm w, b mà y^ càng gần y càng tốt ( y^ ≈ y )

tức L càng nhỏ càng tốt

tức tìm cực tiểu của L

L(y^, y) = 1 \ 2 .(y^ - y)2 (origin)

hàm này tìm cực tiểu k hiệu quả

L(y^, y) = - (y.log(y^) + (1 - y).log(1 - y^)) (other)

hàm này tìm cực tiểu hiệu quả hơn

... (còn rất nhiều hàm khác)

=> đánh giá hiệu quả (chính xác)

của 1 đơn vị trong tập dataset

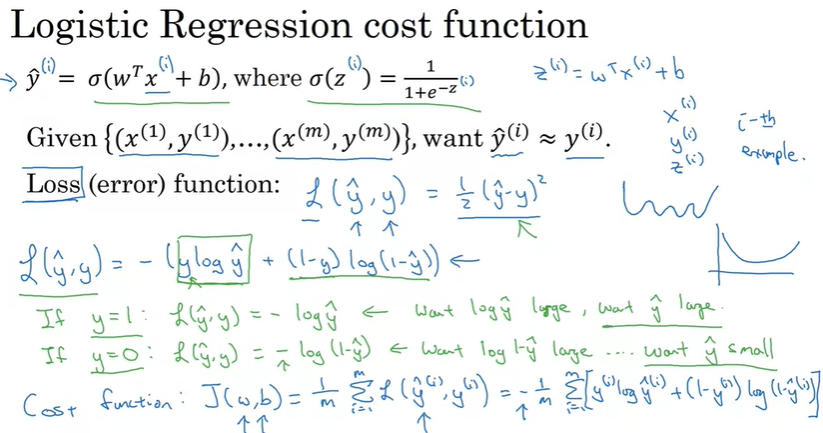
cost function

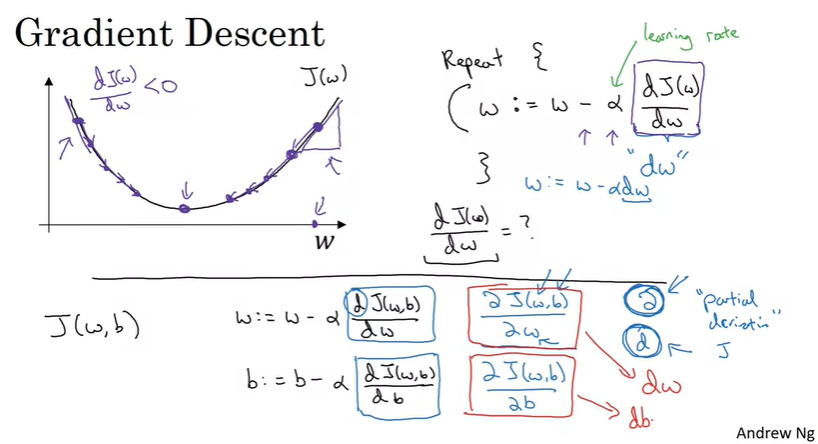
J(w,b) = 1 / m \* ∑ L(y^(i), y(i)) vs i= {1:m}



=> đánh giá hiệu quả (chính xác) trung bình

của của toàn tập dataset





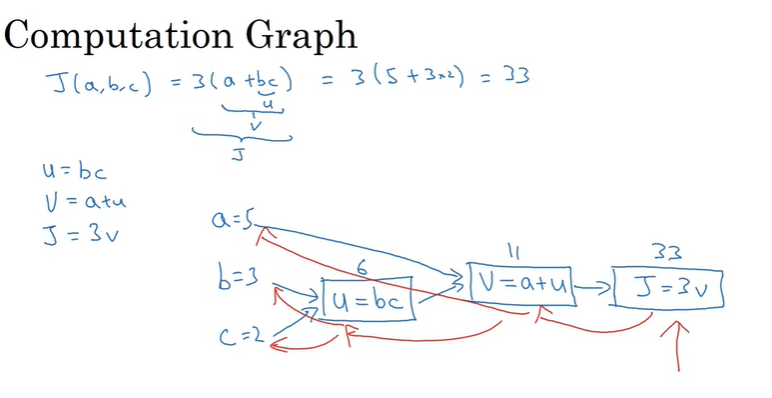
đồ thị tính toán

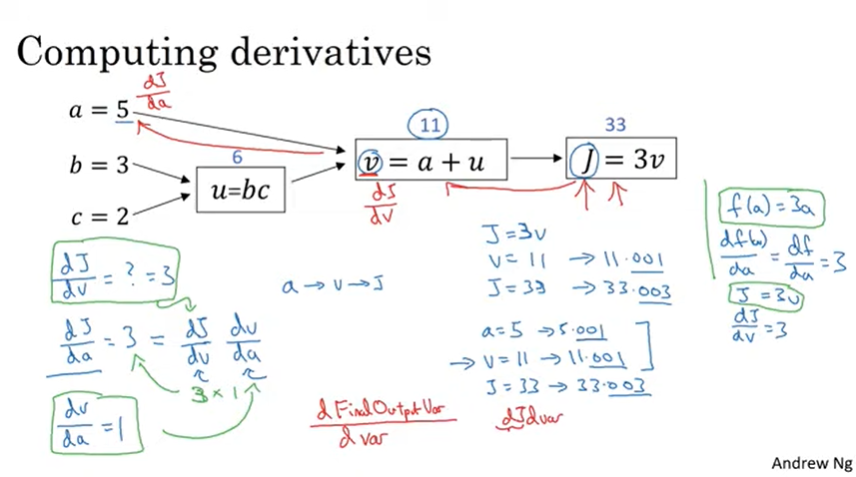
mũi tên xanh là chiều xuôi (ford propagation)

mà yêu cầu giảm chi phí J

mũi tên đỏ là đi người lại (back propagation)

có dùng đạo hàm (derivative)

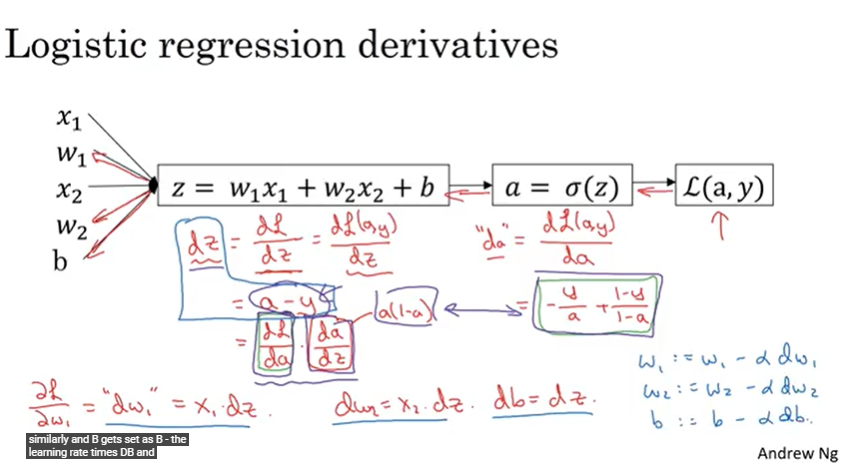




đạo hàm với một đồ thị tính toán.

thực ra nó chính là đạo hàm theo từng biến đã học trong giải tích

dJ / da = 3 dJ / db = 3c dJ / dc = 3b



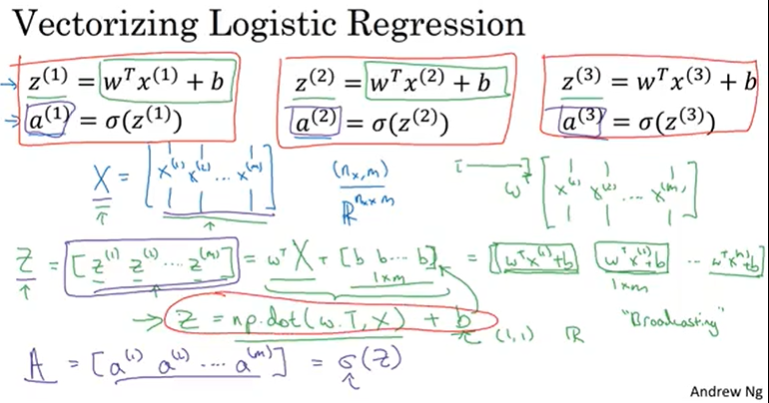
tương quan cũng như đạo hàm theo các biến và gradient descent cần tính

giống như linear mình có thể tình được đạo hàm của từng thằng

để ý tất cả đều có chung dz

thì dz là cái mình tính được tại mỗi trạng thái của w

đạo hàm của toàn bộ tập trainning chính là tổng của các đạo hàm thành phần



x hàng dọc

tập data train sẽ stack x theo chiều ngang X (1 -> m)

w cũng theo chiều dọc

nên khi nhân cần w.T trước rồi nhân vs X (1 -> m)

b thực ra là chiều dọc

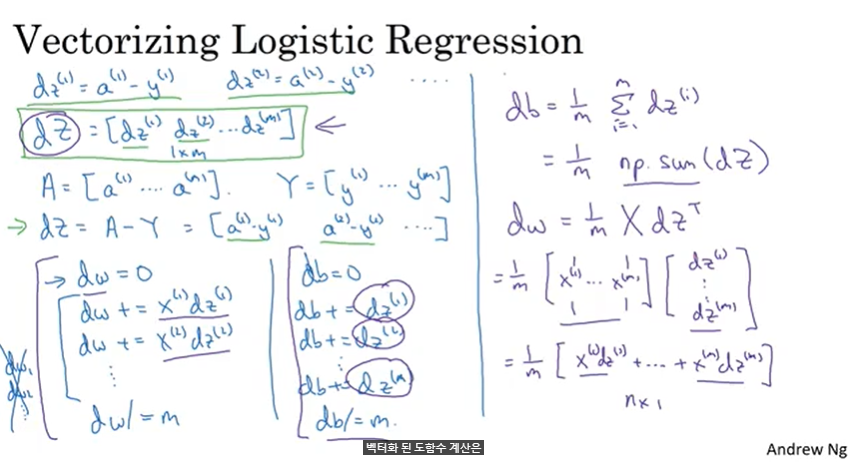
do trung gian chỉ có 1 giá trị nên b trông giống ngang (broadcast)

thực ra là b giống nhau ngang (1 -> m)

VD:

về sau mạng có 2 nút trung gian

có b1, b2 xếp hàng dọc, stack hàng ngang (1->m)

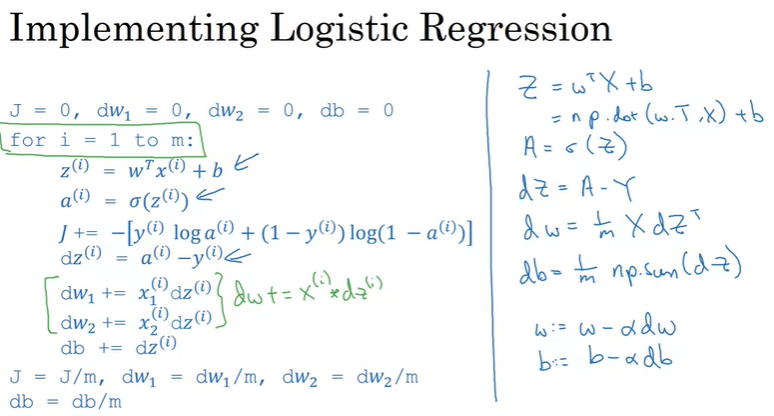


db xếp thành hàng ngang từ 1 -> m

dz xếp thành hàng ngang từ 1 -> m

dw xếp thành hàng ngang từ 1 -> m

//chiều dọc của tất cả những cái này sẽ giống nhau là số chiều của input



tổng hợp lại các bước cần tính

Z = WT.X + B

A = sig(Z)

dZ = A - Y //Y là tập giá trị train

dB = 1/m \* np.sum(dZ)

dW = 1/m \* X.dot(dZT)