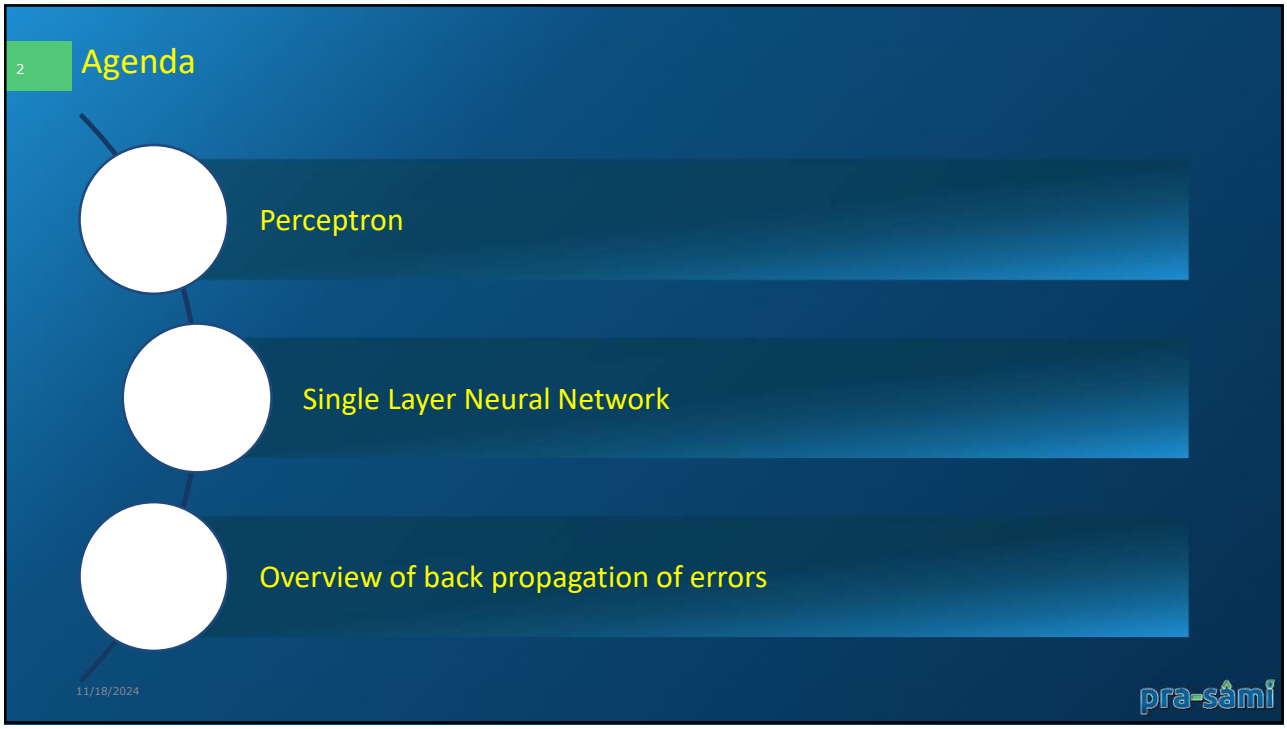
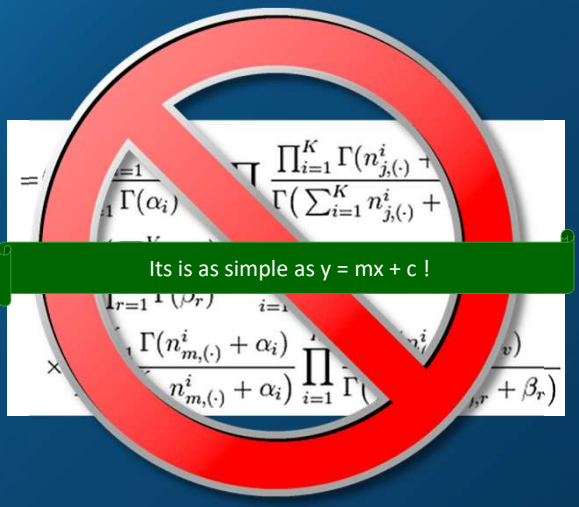

Perceptron, ANN, BackProp
Deep Neural Networks
Session 02
Pramod Sharma
pramod.sharma@prasami.com


2
Agenda
Perceptron
Single Layer Neural Network
Overview of back propagation of errors
11/18/2024
pra-sami

3 Solution to Equation of Perceptron



Its is as simple as y = mx + c !

Frank Rosenblatt

pra-sâmi

4 To play or not to play…

id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0	38	1	15	0	600	1
2	0	25	1	15	1	800	1
3	0	26	1	15	1	1000	1
4	5	27	1	10	1	600	0
5	20	23	0	8	1	1800	0
6	30	22	0	6	0	600	0

- Features:
  - Rains in millimeter
  - Temperature in ° C
  - Homework completed? – 0 : No; 1: Yes
  - Team members : How many team members are ready to play?
  - Is cricket equipment available?
  - Ground: per hour rent in Rupees/hour

pra-sâmi

5

Weights

- Each of the feature has different importance
- To assign importance to each of the feature, we use weights!
- Values of each features are in different order of magnitude
  - Summation is not going to work
  - Scale the features between 0 and 1

id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0	38	1	15	0	600	1
2	0	25	1	15	1	800	1
3	0	26	1	15	1	1000	1
4	5	27	1	10	1	600	0
5	20	23	0	8	1	1800	0
6	30	22	0	6	0	600	0

- Note:
  - Variation in features have different bearing on the results
  - Team members ➔ higher the better
  - Ground cost ➔ lower the better

pra-sami

6

Perceptron

- In MP Neuron Model,
  - All inputs had same weights
  - Threshold 'w_0' could take limited values
  - Every feature needed to be [0,1]
- Perceptron model introduced different weights to different inputs features
- Real values are also accepted
  - Temperatures are in tens and ground rent is in hundreds.
  - Min – Max – Scaler to compensate for huge difference is values
- Threshold 'w_0' can take any value
- Outputs are still [0, 1]

pra-sami

7

Perceptron

- Loss Function:
  - A correction is applied on the outputs
  - To adjust values of 'w_i' to reach right results
  - It would also give us indications of what weights to be fixed to arrive at the solution

- Activation function g(x) is applied as follows:
  - If $\sum x_i . w_i \geq w_0$ => $\hat{y} = 1$
  - If $\sum x_i . w_i < w_0$ => $\hat{y} = 0$

pra-sâmi

8

Perceptron – Data Preprocessing

- Lets consider "Ground" and "Team Members" as features and its associated weights to arrive at the solution.

id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0	38	1	15	0	600	1
2	0	25	1	15	1	800	1
3	0	26	1	15	1	1000	1
4	5	27	1	10	1	600	0
5	20	23	0	8	1	1800	0
6	30	22	0	6	0	600	0

pra-sâmi

9

Perceptron – Data Preprocessing

- Scaled Data (all columns to be between 0 and 1)

id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0.00	0.00	1.00	1.00	0.00	1.00	1
2	0.00	0.81	1.00	1.00	1.00	0.83	1
3	0.00	0.75	1.00	1.00	1.00	0.67	1
4	-0.17	0.69	1.00	0.44	1.00	1.00	0
5	-0.67	0.94	0.00	0.22	1.00	0.00	0
6	-1.00	1.00	0.00	0.00	0.00	1.00	0

- What about reverse correlation
- Two option to address reverse correlation
  - Take negative of values
  - Use negative weight

pra-sami

10

Perceptron – Weights

- Weights – consider importance of each of the feature

id	Threshold	Team Members		Ground		Calculations	Likely	Played	Loss
	w0	x1	w1	x2	w2	w0+x1*w1+x2*w2	(y_hat)	(y)	(y-y_hat)^2
1	-1.00	1.00	1.10	1.00	1.00	1.10	1	1	0
2	-1.00	1.00	1.10	0.83	1.00	0.93	1	1	0
3	-1.00	1.00	1.10	0.67	1.00	0.77	1	1	0
4	-1.00	0.44	1.10	1.00	1.00	0.49	1	0	1
5	-1.00	0.22	1.10	0.00	1.00	-0.76	0	0	0
6	-1.00	0.00	1.10	1.00	1.00	0.00	1	0	1

pra-sami

11

Perceptron – Weights and Loss

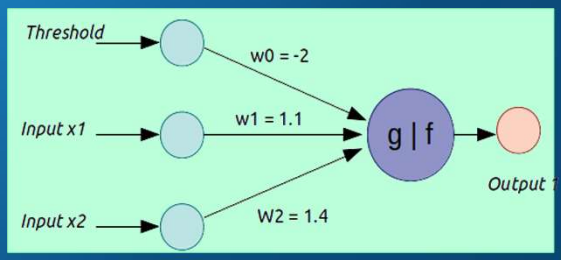
- Our best solution would be where ground truth and predicted values are same
- Loss is some function of ground truth and predicted values
- And we want it to be cumulative, Square of difference looks promising
  - $\ell(\hat{y}, y) = (y - \hat{y})^2$
  - Our overall loss was 2.
- By adjusting weights (w_1, w_2) and threshold (w_0) we can bring the loss to minimum (zero in this case)

id	Threshold	Team Members		Ground		Calculations	Likely	Played	Loss
	w0	x1	w1	x2	w2	w0+x1*w1+x2*w2	(y_hat)	(y)	(y-y_hat)^2
1	-2.00	1.00	1.10	1.00	1.40	0.50	1	1	0
2	-2.00	1.00	1.10	0.83	1.40	0.27	1	1	0
3	-2.00	1.00	1.10	0.67	1.40	0.03	1	1	0
4	-2.00	0.44	1.10	1.00	1.40	-0.11	0	0	0
5	-2.00	0.22	1.10	0.00	1.40	-1.76	0	0	0
6	-2.00	0.00	1.10	1.00	1.40	-0.60	0	0	0

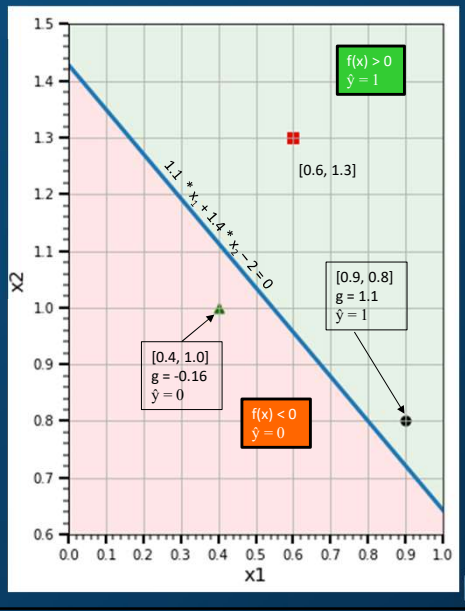
pra-sami

13

Perceptron



- We can represent : g = $w_0 + x_1 * w_1 + x_2 * w_2$
  - As g = $[x_1, x_2].\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + w_0$
- Given: W = $\begin{bmatrix} 1.1 \\ 1.4 \end{bmatrix}$ and w_0 = -2
  - g = $[x_1, x_2].\begin{bmatrix} 1.1 \\ 1.4 \end{bmatrix}$ - 2
  - g = $1.1 * x_1, +1.4 * x_2$ - 2

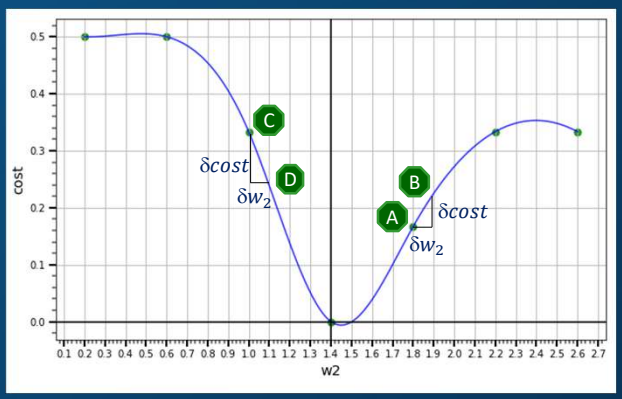


pra-sami

14

Perceptron – Gradient Descent

- w_0, w_1, w_2 need to be adjusted to arrive at most optimal solution i.e. lowest point on the graph.
- Assume that w_0 is fixed at -2, and w_1 at 1.1 and w_2 varies from 0 to 3 (only one variable considered to make plotting simple)
- From point A to B, slope is positive hence w_2 value needs to be decreased
- From point C to D slope is negative hence w_2 needs to be increased.



pra-sami

15

Perceptron – Activation Function

- So we based our entire calculations on:

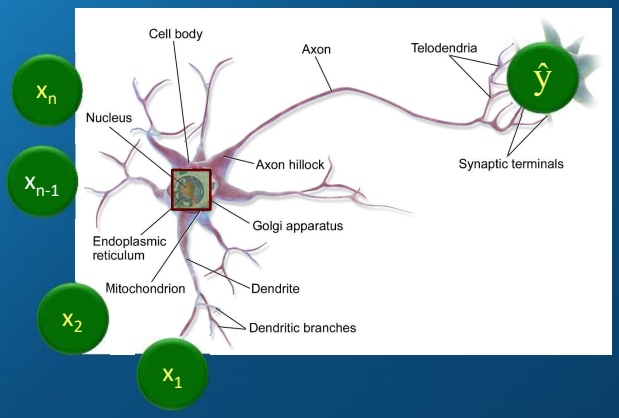
$$z = w_0 + x_1 * w_1 + x_2 * w_2$$

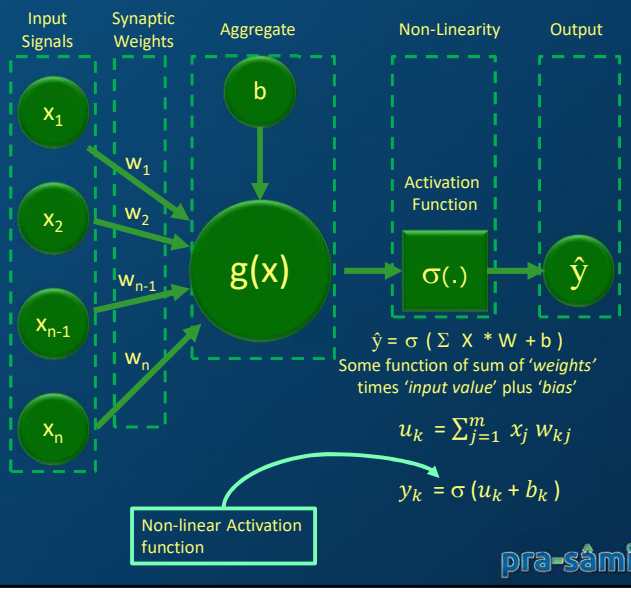
But that's an equation of straight line! ☺
What happened to all those 'inhibitory' features?

pra-sami

16

Non Linear Activation function

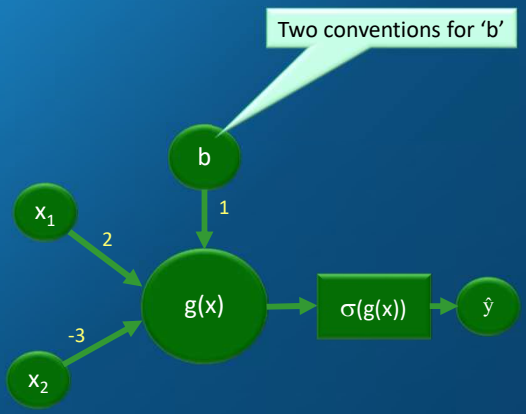




pra-sâmi

17

Perceptron with non-linear activation function



- Given:
  - W = $\begin{bmatrix} 2 \\ -3 \end{bmatrix}$ and b = 1
  - $\hat{y}$ = σ ([x1, x2].$\begin{bmatrix} 2 \\ -3 \end{bmatrix}$ + 1)
  - $\hat{y}$ = σ (1 + 2 * x1 -3 * x2)

z

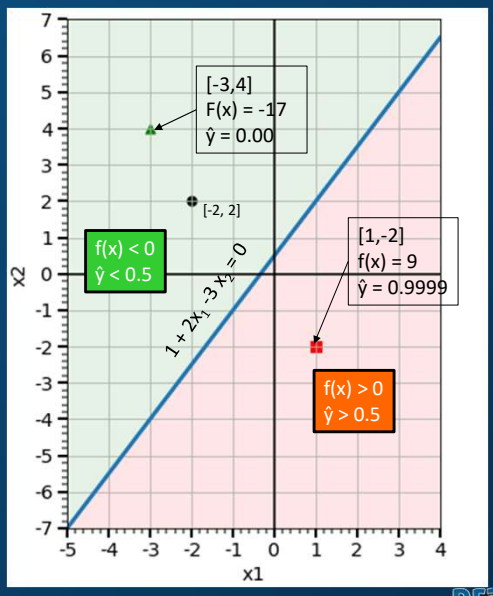
- $\hat{y}$ = σ (z) ;
- Lets use sigmoid function for σ.
  - $\hat{y} = \frac{1}{(1+e^{-z})}$

pra-sâmi

18

Perceptron with non-linear activation function

- $\hat{y} = \sigma\ (1 + 2 * x_1 - 3 * x_2)$
- For X = [-3, 4]
  - $\hat{y} = \sigma\ (1 + 2 * (-3) - 3 * 4)$
  - $\hat{y} = \sigma\ (1 - 6 - 12)$
  - $\hat{y} = \sigma\ (-17)$
  - $\hat{y} = 0.0$
- Similarly, for X = [1, -2]
  - $\hat{y} = \sigma\ (1 + 2 * 1 - 3 * (-2))$
  - $\hat{y} = \sigma\ (1 + 2 - 6)$
  - $\hat{y} = \sigma\ (9)$
  - $\hat{y} = 1.0$



11/18/2024

pra-sami

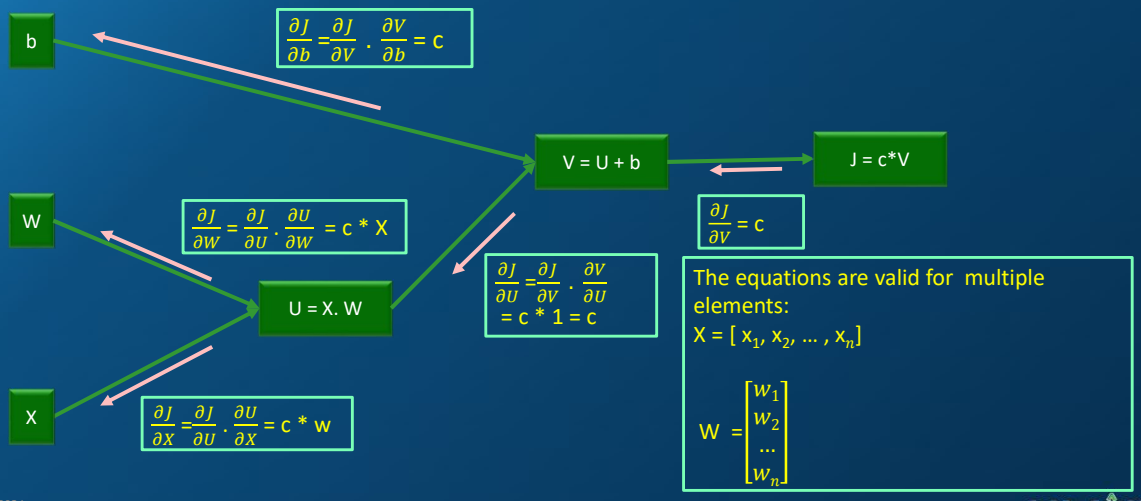
19

Perce ith non-linear activation function

20

Computational Graph

- Consider following hypothetical case, basic equation for single neuron :
  - $\hat{y}$ = X. W + b and Cost is some constant times $\hat{y}$; J = c * $\hat{y}$



The equations are valid for multiple elements:

X = [x_1, x_2, … , x_n]

$$W = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{bmatrix}$$

pra-sami

21

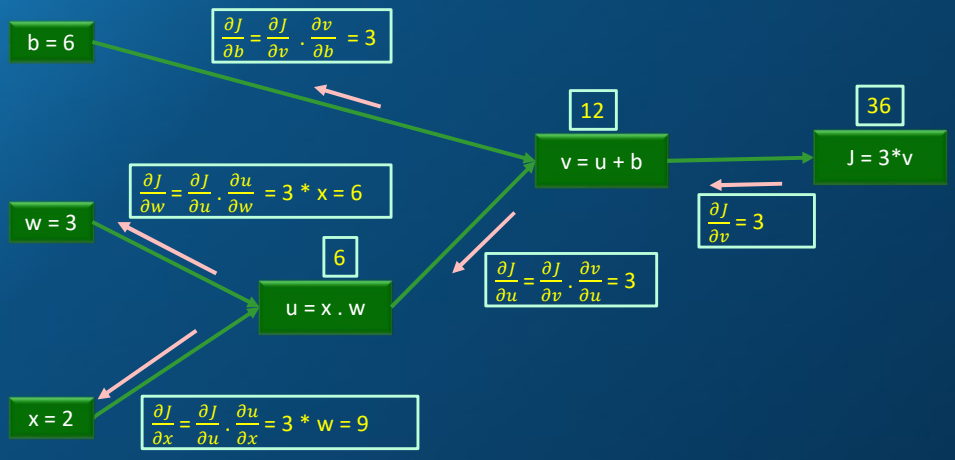
Exercise 2 : Computational Graph

- Given a Cost Function J
  - J (w, x, b) = 3 *(b + x * w)
- Calculate $\frac{\partial J}{\partial w}$, $\frac{\partial J}{\partial x}$ and $\frac{\partial J}{\partial b}$
- Calculate slope at point :
  - b = 6
  - w = 3
  - x = 2

pra-sami

22 Exercise - Solution

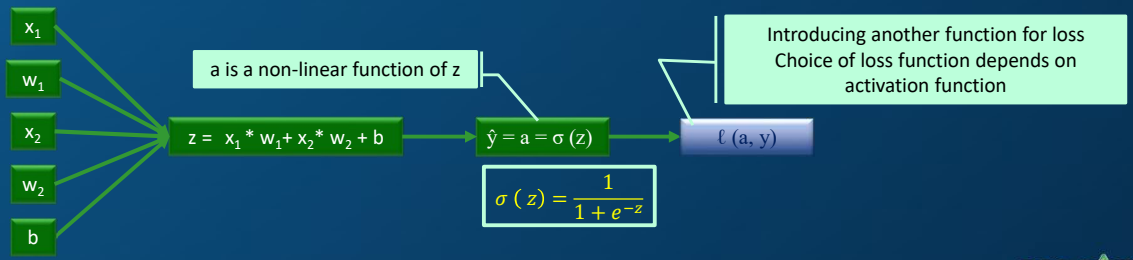
- Given a Cost Function J (w, x, b) = 3 *(b + w*x)



pra-sami

23 Consider Single Path… MLE

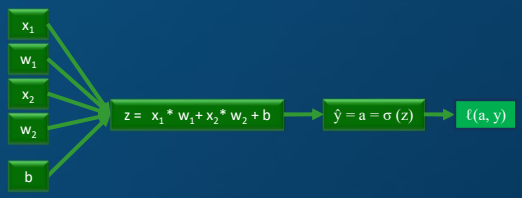
- Maximum likelihood estimation, or MLE, is a framework for inference for finding the best statistical estimates of parameters from historical training data
  - Exactly what we are trying to do with the neural network
- In Classification, output is probability of it belonging to a class
  - Maximum likelihood estimation, seeks a set of model weights that minimize the difference between the predicted probability distribution and the Ground Truth [cross-entropy]
- In Regression problems:
  - Use the mean squared error (MSE) loss function or equivalent.



pra-sami

24 Consider Single Path… Loss Function

- A function used to evaluate a candidate solution
- Helps to maximize or minimize the objective function



- Estimates how closely the distribution of predictions made by a model matches the ground truth (maximum likelihood)
- Under maximum likelihood framework , the error between two probability distributions is measured using cross-entropy
  - Hence $\ell(\hat{y}, y) = -[y * \log(\hat{y}) + (1 - y) * \log(1 - \hat{y})]$

pra-sami

25 Cost Function

- $\hat{y} = \sigma (\Sigma W * X + b)$
- Where $\sigma (z) = \frac{1}{1+e^{-z}}$
- Loss function:
  - A parameter which defines how good our outputs are i.e.
  - How far our predicted values ‘ŷ’ (y hat) were from ground truth ‘y’
- For logistic regression
  - $Loss(\hat{y}, y) = - (y . \log \hat{y} + (1 – y) . \log (1 - \hat{y})$
  - Loss function is for an instance
  - In case of binary classification, $Loss(\hat{y}, y) = - y . \log \hat{y}$
- Cost Function: Its a sum of losses for all instances
  - $J (W, b) = \frac{1}{m} (\Sigma Loss(\hat{y}, y))$
  - $= - \frac{1}{m} (\Sigma (y . \log \hat{y} + (1 – y) . \log (1 - \hat{y}))$
- For binary classification:
  - $J (W, b) = \frac{1}{m} (\Sigma Loss(\hat{y}, y))$
  - $= - \frac{1}{m} (\Sigma (y . \log \hat{y})$

pra-sami

26

Forward and Back Propagation

$z = x_1 * w_1 + x_2 * w_2 + b$ → $\hat{y} = a = \sigma(z)$ → $\ell(a, y)$

$$\frac{\partial \ell}{\partial z} = \frac{\partial \ell}{\partial a} \cdot \frac{\partial a}{\partial z}$$
$$= \{ -\frac{y}{a} + \frac{1-y}{1-a} \} * a * (1-a)$$
$$= a - y$$

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z) \cdot (1 - \sigma(z))$$
$$\frac{\partial a}{\partial z} = a \cdot (1-a)$$

$$\frac{\partial \ell}{\partial a} = -\frac{y}{a} + \frac{1-y}{1-a}$$

$z = X * W + b$

$\hat{y} = a = \sigma(z)$

$\sigma(z) = \frac{1}{1+e^{-z}}$

$\ell(a, y) = -[y * \log(a) + (1-y) * \log(1-a)]$

For binary classification:

$\ell(a, y) = -y * \log(a)$

$\Rightarrow$

$$\frac{\partial \ell}{\partial w1} = x_1 \cdot \frac{\partial \ell}{\partial z} = x_1 \cdot (a-y)$$
$$\frac{\partial \ell}{\partial w2} = x_2 \cdot \frac{\partial \ell}{\partial z} = x_2 \cdot (a-y)$$
$$\frac{\partial \ell}{\partial b} = \frac{\partial \ell}{\partial z} = (a-y)$$

$\Rightarrow$

$w_1 = w_1 - \alpha * \frac{\partial \ell}{\partial w_1} = w_1 - \alpha * x_1 * (a-y)$

$w_2 = w_2 - \alpha * \frac{\partial \ell}{\partial w_2} = w_2 - \alpha * x_2 * (a-y)$

$b = b - \alpha * \frac{\partial \ell}{\partial b} = b - \alpha * (a-y)$

Where α is learning rate. The cost function is

$J(W, b) = \frac{1}{m} * (\Sigma \ell(a, y))$

Hence $\frac{\partial J}{\partial w_1} = \frac{1}{m} * (\Sigma \frac{\partial \ell(a, y)}{\partial w_1})$

pra-sami

27

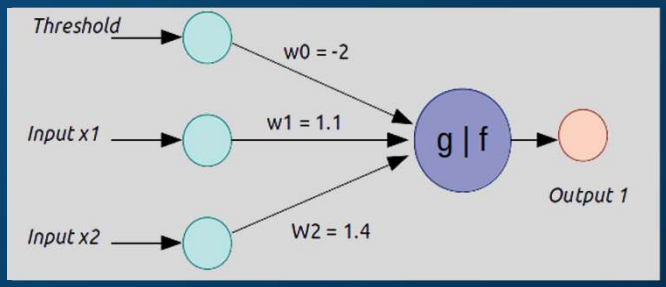
So where are the hidden layers!!!

pra-sami

28

Hidden Layers

id	Threshold	Team Members		Ground	
	x0	x1	w1	x2	w2
1	-2.00	1.00	1.10	1.00	1.40
2	-2.00	1.00	1.10	0.83	1.40
3	-2.00	1.00	1.10	0.67	1.40
4	-2.00	0.44	1.10	1.00	1.40
5	-2.00	0.22	1.10	0.00	1.40
6	-2.00	0.00	1.10	1.00	1.40

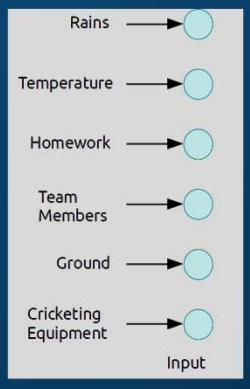


pra-sâmi

29

Hidden Layers

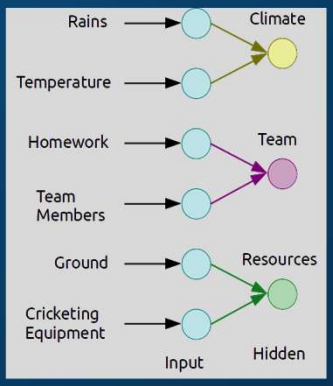
id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0.00	0.00	1.00	1.00	0.00	1.00	1
2	0.00	0.81	1.00	1.00	1.00	0.83	1
3	0.00	0.75	1.00	1.00	1.00	0.67	1
4	-0.17	0.69	1.00	0.44	1.00	1.00	0
5	-0.67	0.94	0.00	0.22	1.00	0.00	0
6	-1.00	1.00	0.00	0.00	0.00	1.00	0



pra-sâmi

30 Hidden Layers

id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0.00	0.00	1.00	1.00	0.00	1.00	1
2	0.00	0.81	1.00	1.00	1.00	0.83	1
3	0.00	0.75	1.00	1.00	1.00	0.67	1
4	-0.17	0.69	1.00	0.44	1.00	1.00	0
5	-0.67	0.94	0.00	0.22	1.00	0.00	0
6	-1.00	1.00	0.00	0.00	0.00	1.00	0

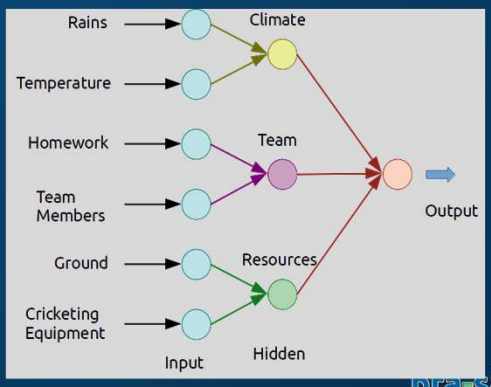


11/18/2024

pra-sâmi

31 Hidden Layers

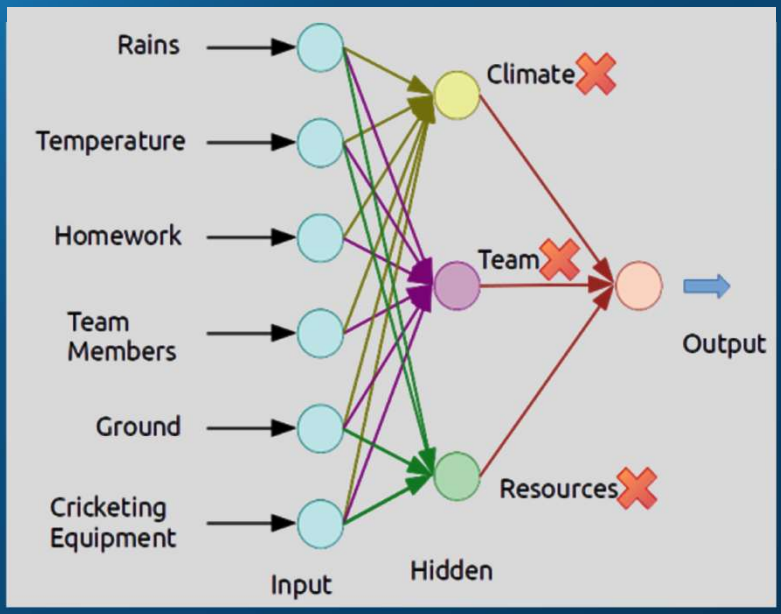
id	Rains	Temp	Homework	Team Members	Equipment	Ground	Played
1	0.00	0.00	1.00	1.00	0.00	1.00	1
2	0.00	0.81	1.00	1.00	1.00	0.83	1
3	0.00	0.75	1.00	1.00	1.00	0.67	1
4	-0.17	0.69	1.00	0.44	1.00	1.00	0
5	-0.67	0.94	0.00	0.22	1.00	0.00	0
6	-1.00	1.00	0.00	0.00	0.00	1.00	0



11/18/2024

pra-sâmi

32 Hidden Layers

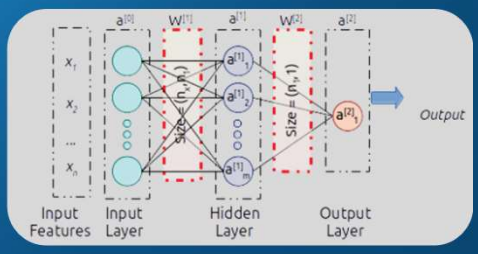


pra-sâmi

33 Hidden Layers

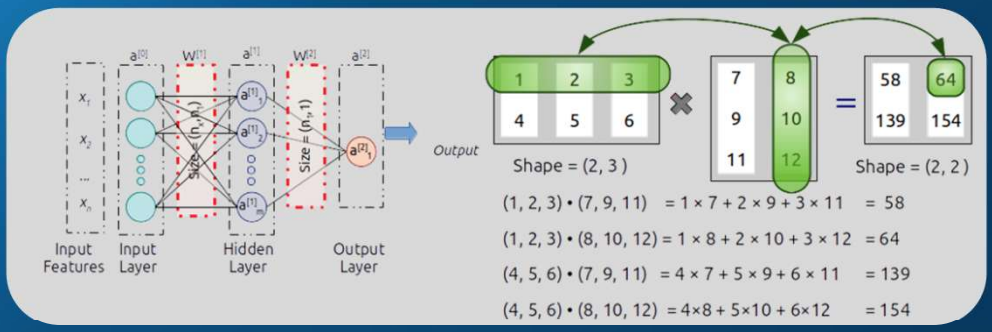
pra-sâmi

34 Two Major Conventions



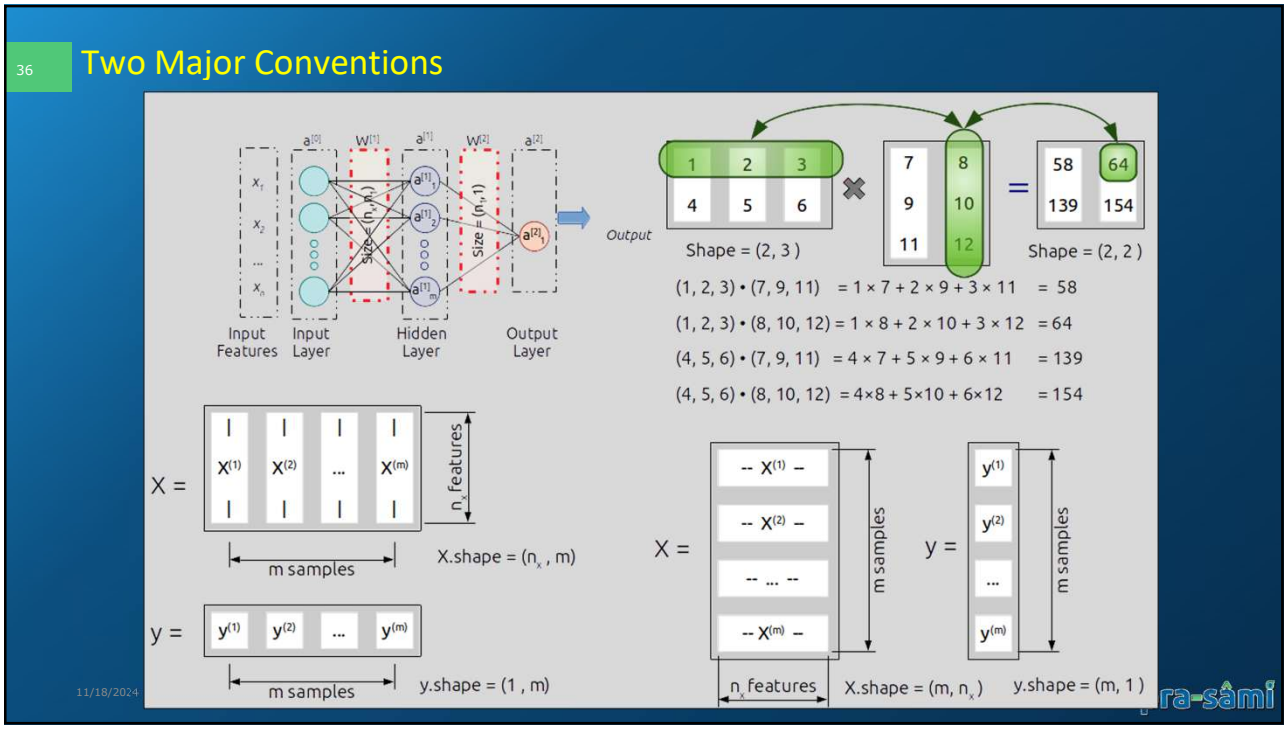
pra-sâmi

35 Two Major Conventions



pra-sâmi

36 Two Major Conventions



37 Two M

38

Next Session - Coding Perceptron Model in Python

pra-sâmi

39

pra-sâmi

ADDITIONAL MATERIAL

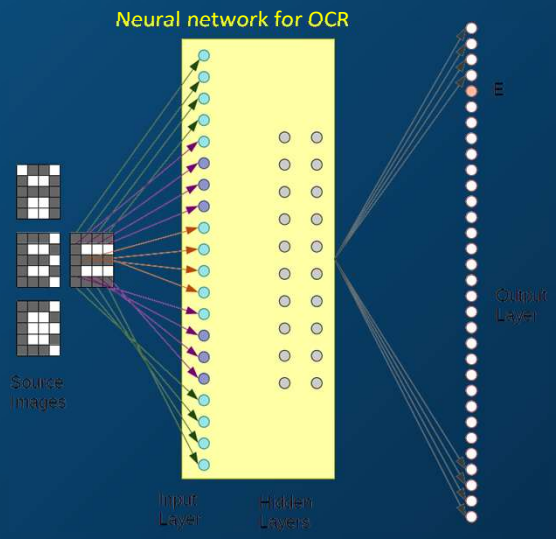
pra-sâmi

Applications

41

Applications

- The properties of neural networks define where they are useful
- Typical Network
  - Can learn complex mappings from inputs to outputs, based solely on samples
  - Difficult to analyse
  - Firm predictions about neural network behaviour difficult;
    - Unsuitable for safety-critical applications.
  - Require limited understanding from trainer, who can be guided by heuristics

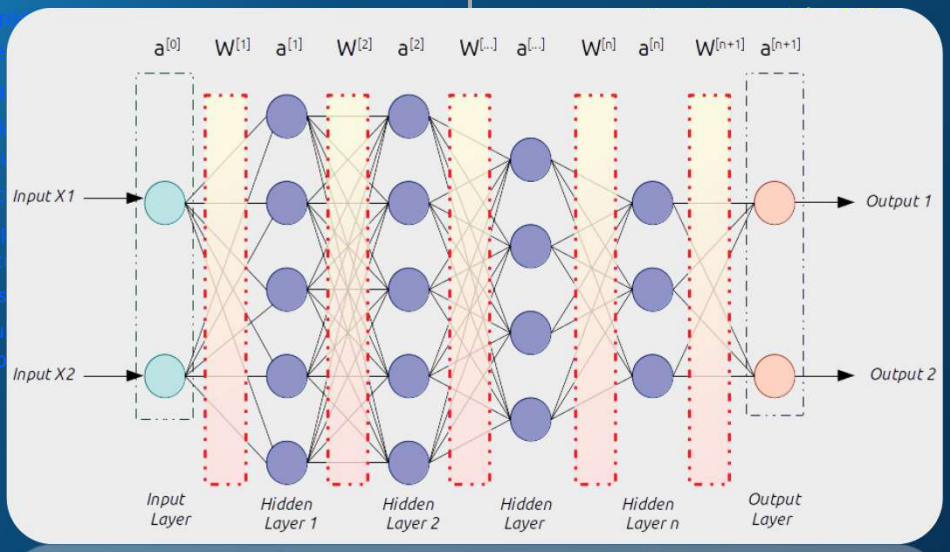


pra-sâmi

42

Applications

- The pro
  where
- Typica
  - Can l
    outpu
  - Diffic
  - firm p
    diffic
    - Uns
  - Requ
    can b

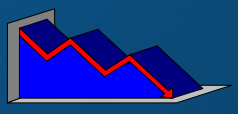


pra-sâmi

43

Applications

- Stock market prediction
  - "Technical trading" refers to trading based solely on known statistical parameters; e.g. previous price
  - Neural networks have been used to attempt to predict changes in prices.
  - Difficult to assess success or otherwise
    - Since companies using these techniques are reluctant to disclose information.

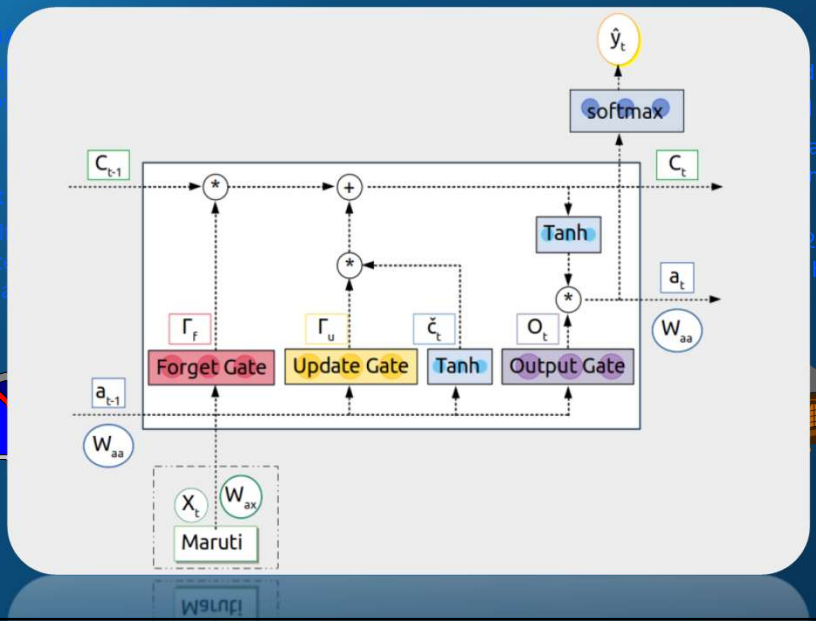


- Mortgage assessment
  - Assess risk of lending to an individual
  - Difficult to decide on marginal cases
  - Neural networks have been trained to make decisions, based upon the opinions of expert underwriters
  - Neural network produced a 12% reduction in delinquencies compared with human experts

pra-sâmi

44 Applications

- Stock m
  - "Techni on knowl price
  - Neural predict
  - Difficul these t informa



ividual.
cases.
ained to make
nions of expert
2% reduction in
human experts.

pra-sâmi

45 Applications

- ALVINN: Autonomous Land Vehicle In a Neural Network

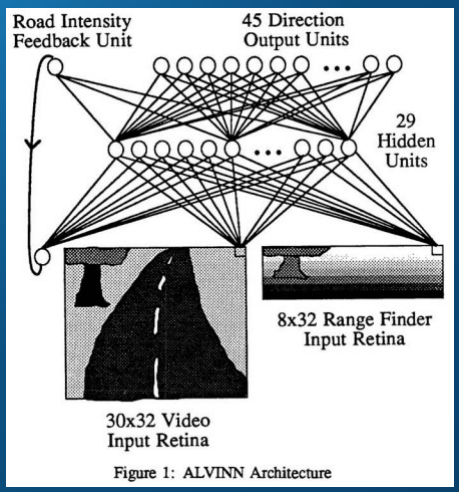
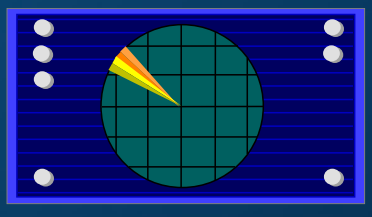


Figure 1: ALVINN Architecture

- Sonar target recognition
  - Distinguish mines from rocks on sea-bed
  - The neural network is provided with a large number of parameters which are extracted from the sonar signal.
  - The training set consists of sets of signals from rocks and mines.

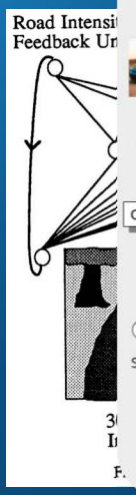


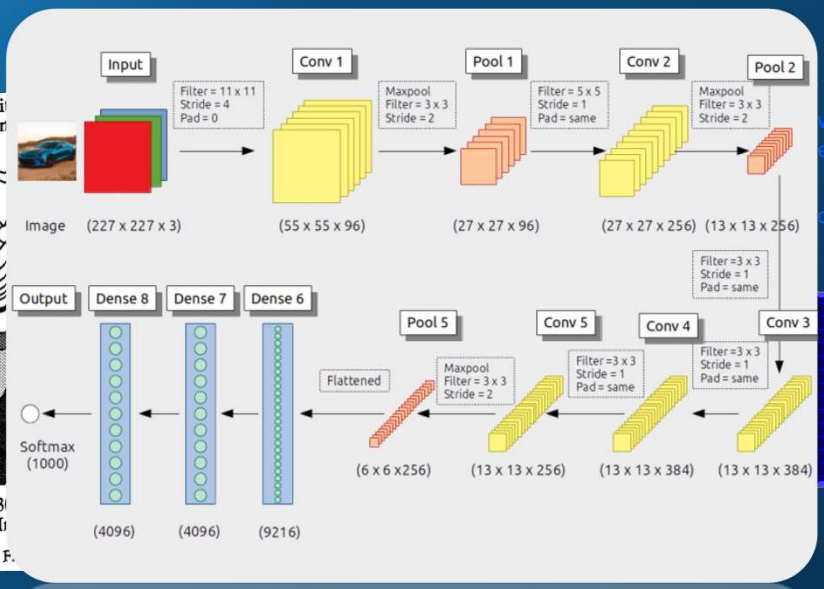
pra-sâmi

46

Applications

- ALVINN:
  Network





sea-bed
with a large
e extracted from

of signals from

pra-sami

47

Applications

- Engine management
  - The behavior of a car engine is influenced by a large number of parameters
    - temperature at various points
    - fuel/air mixture
    - lubricant viscosity.
  - Major companies have used neural networks to dynamically tune an engine depending on current settings

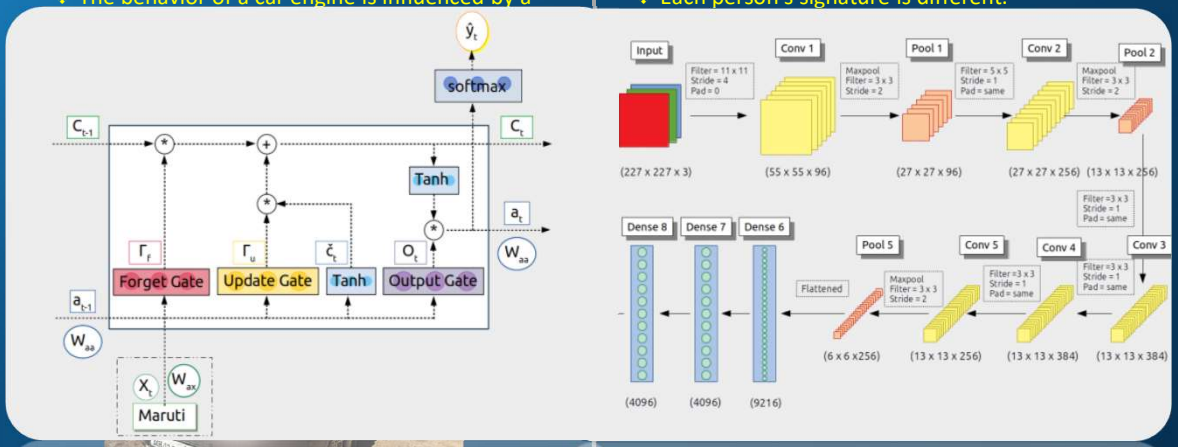
- Signature recognition
  - Each person's signature is different.
  - There are structural similarities which are difficult to quantify.
  - Recognizes signatures to a high level of accuracy.
  - Considers speed in addition to gross shape
  - Makes forgery even more difficult.

pra-sami

48

Applications

- Engine management
  - The behavior of a car engine is influenced by a
- Signature recognition
  - Each person's signature is different.



pra-sâmi

49

Derivation of Sigmoid

$$
\begin{aligned}
\partial a &= \partial\sigma(z) \\
&= \frac{\partial}{\partial z}\left[\frac{1}{1+e^{-z}}\right] \\
&= \frac{\partial}{\partial z}\left(1+e^{-z}\right)^{-1} \\
&= -(1+e^{-z})^{-2}(-e^{-z}) \\
&= \frac{e^{-z}}{(1+e^{-z})^2} \\
&= \frac{1}{1+e^{-z}} \circ \frac{e^{-z}}{1+e^{-z}} \\
&= \frac{1}{1+e^{-z}} \circ \frac{(1+e^{-z})-1}{1+e^{-z}} \\
&= \frac{1}{1+e^{-z}} \circ \left[\frac{1+e^{-z}}{1+e^{-z}} - \frac{1}{1+e^{-z}}\right] \\
&= \frac{1}{1+e^{-z}} \circ \left[1 - \frac{1}{1+e^{-z}}\right] \\
&= \sigma(z) \circ (1-\sigma(z)) \\
&= a \circ (1-a)
\end{aligned}
$$

pra-sâmi