Overview

Introduction

Class Logistics

Set Up

Linear Regression

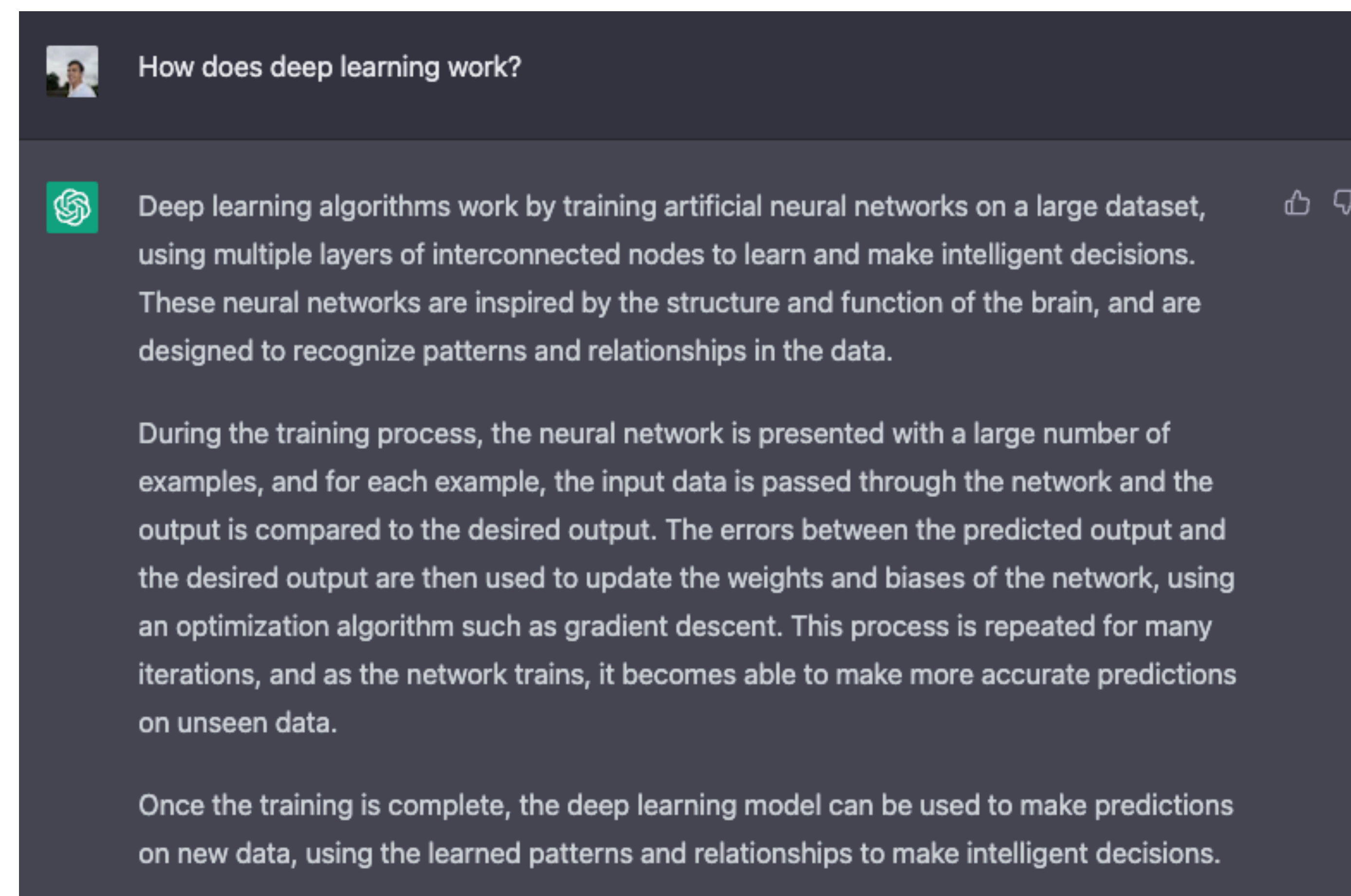
Classification

Introduction

Deep learning is amazing

→ stable diffusion

→ Chat GPT



How does deep learning work?

Deep learning algorithms work by training artificial neural networks on a large dataset, using multiple layers of interconnected nodes to learn and make intelligent decisions. These neural networks are inspired by the structure and function of the brain, and are designed to recognize patterns and relationships in the data.

During the training process, the neural network is presented with a large number of examples, and for each example, the input data is passed through the network and the output is compared to the desired output. The errors between the predicted output and the desired output are then used to update the weights and biases of the network, using an optimization algorithm such as gradient descent. This process is repeated for many iterations, and as the network trains, it becomes able to make more accurate predictions on unseen data.

Once the training is complete, the deep learning model can be used to make predictions on new data, using the learned patterns and relationships to make intelligent decisions.

Class Structure

Prerequisites

- linear algebra
- calculus
- python

lecture	10-noon	
demo	2-3	MTWR
office	3-	

Resources

- website go/cs1051/
- based on Chinmay Hegde's class at NYU
- post reading, notes (these), demo
- d2l.ai
- post questions on Canvas

Assignments

→ Google form after every class

→ Homework problem for each day of content

↳ self grade

↳ theory solution in LaTeX

→ Project (and proposal)

- code
- write up
- presentation

Machine Learning

ML (and AI and data science)

- "ML could create the perfect game bosses" (last week)
- "DL model can detect disease-causing mosaic mutations" (two days)
- "ML models can predict hypertension risk" (yesterday)

Why now?

- lots of compute
- lots of data

But often ML is a black box

↳ When do ML models work?

↳ How efficient are they?

↳ Why does work?

This class: <u>some</u> answers

What is machine learning?

- computers recognize patterns
- brain inspired learning
- answer questions without previous knowledge

Data → model → actionable information

What is your background and goal for this class?

☆ 3 Step Recipe ☆

Every ML model has

1. representation / architecture
2. measure of goodness / loss
3. optimizer / training algorithm

Vectors

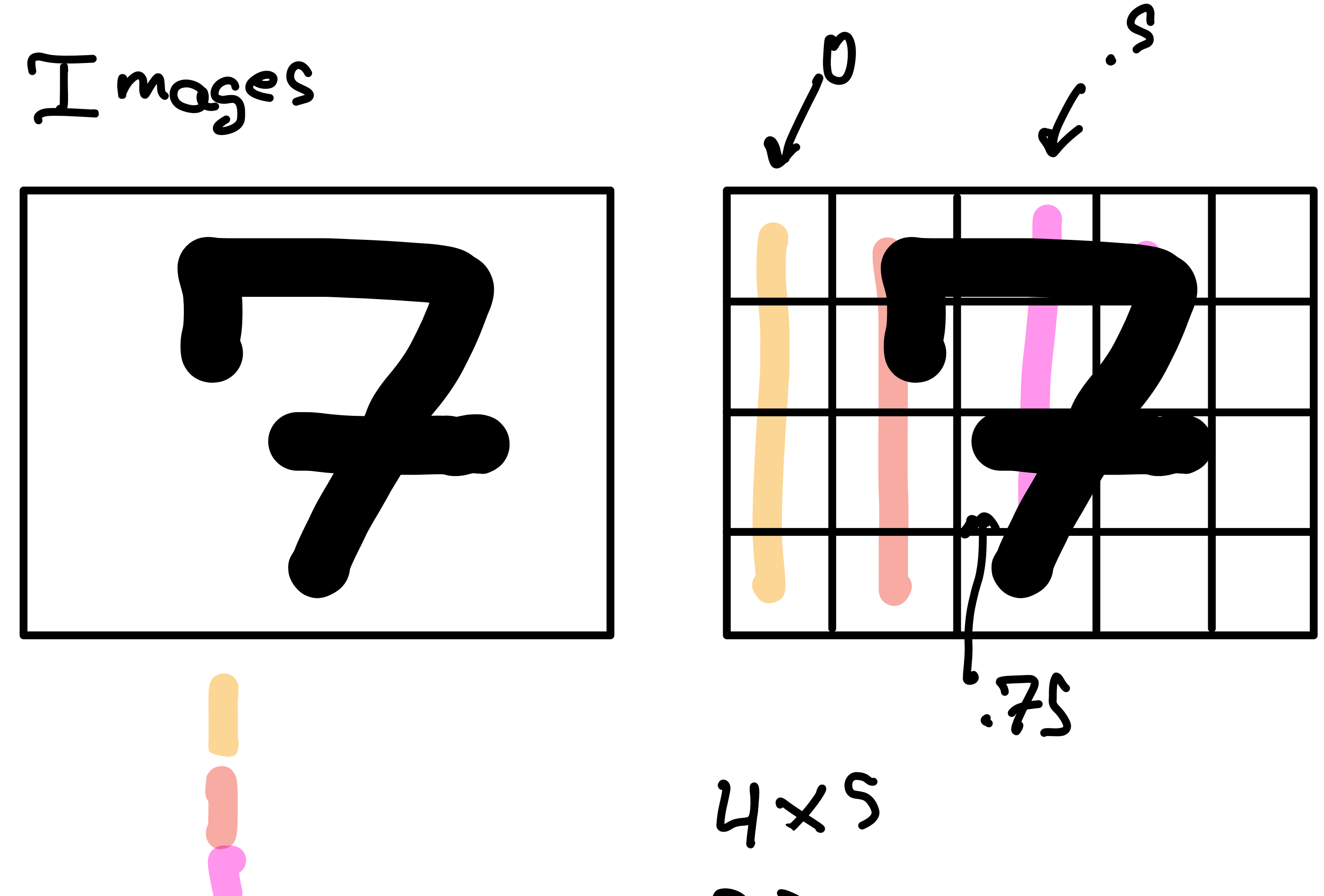
We represent data as vectors in high dimensional spaces

Weather: temp, wind, aqi, uv, humid, visib

$\begin{bmatrix} 43 \\ 8 \end{bmatrix}, \begin{bmatrix} 50 \\ 15 \end{bmatrix} \in \mathbb{R}^2$

$\begin{bmatrix} 5 \\ 1 \\ 90\% \\ 9 \end{bmatrix}, \begin{bmatrix} 5 \\ 1 \\ 90\% \\ 9 \end{bmatrix} \in \mathbb{R}^6 \quad \in \mathbb{R}^d$

Images



4×5

20

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \in \mathbb{R}^d$$

$$y = \begin{bmatrix} y_1 \\ y_d \end{bmatrix} \in \mathbb{R}^d$$

linearity

$$x + y = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_d + y_d \end{bmatrix}$$

scaling

$c \in \mathbb{R}$

$$c \cdot x = \begin{bmatrix} c x_1 \\ \vdots \\ c x_d \end{bmatrix}$$

$$\|x\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$$

$$\|x\|_1 = \sum_{i=1}^{d} |x_i|$$

$$\langle x, y \rangle = \sum_{i=1}^{d} x_i y_i$$

$$\langle x, x \rangle = \sum_{i=1}^{d} x_i x_i = \sum_{i=1}^{d} x_i^2 = \|x\|_2^2$$

$$\text{sim}(x, y) = \frac{\langle x, y \rangle}{\|x\|_2 \ \|y\|_2}$$

Linear Regression

Consider labelled data

$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \dots (x^{(n)}, y^{(n)})$

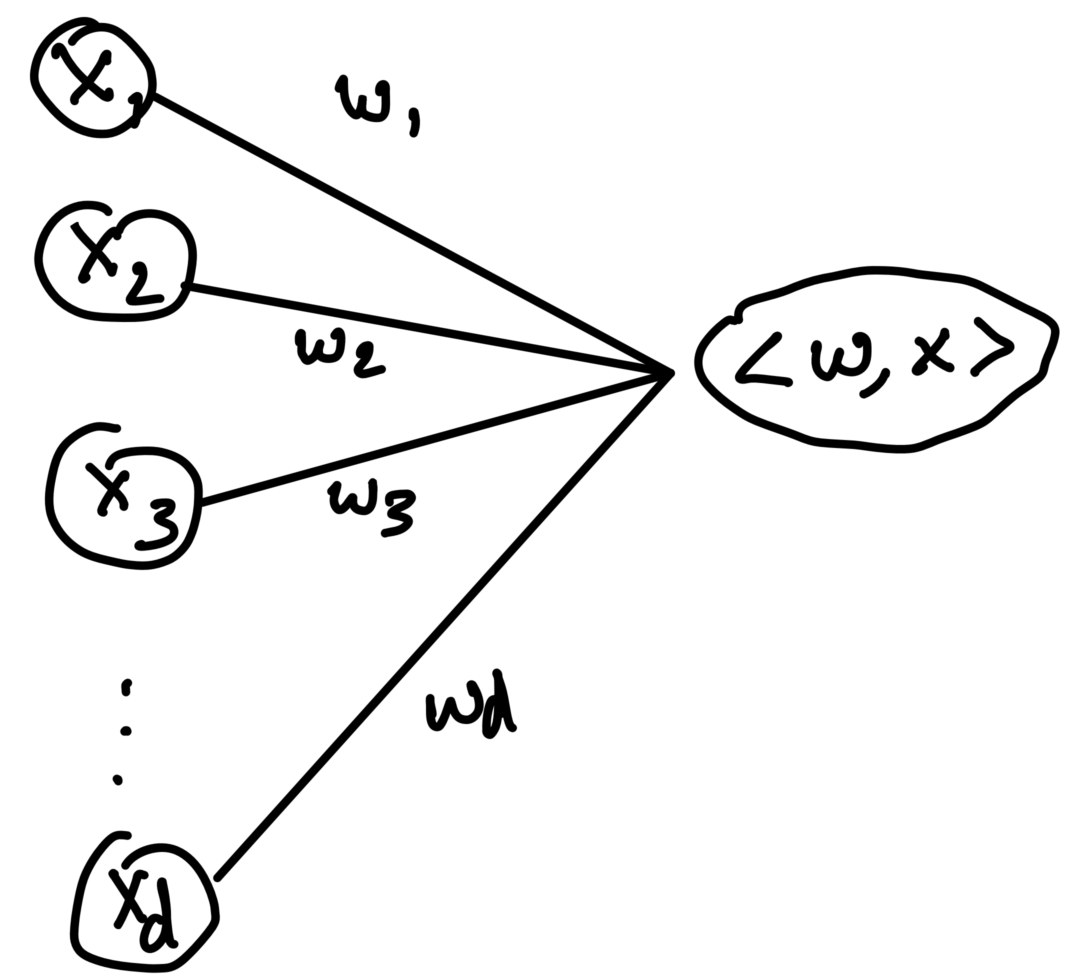
$x^{(i)} \in \mathbb{R}^d$ $\quad y^{(i)} \in \mathbb{R}$

We want $f: \mathbb{R}^d \to \mathbb{R}$

$f(x^{(i)}) \approx y^{(i)}$ for $i \in [n] = \{1, 2, \dots, n\}$

Let $w \in \mathbb{R}^d$ be the weights

① $f_w(x) = \langle w, x \rangle = \sum_{i=1}^{d} w_i x_i$



② loss:

$$\mathcal{L}(w) = \frac{1}{2}\sum_{i=1}^{n} \left(y^{(i)} - \langle w, x^{(i)}\rangle\right)^2$$

$$= \frac{1}{2}\|y - Xw\|_2^2$$

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ y^{(n)} \end{bmatrix}_{n} \qquad w = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}_{d}$$

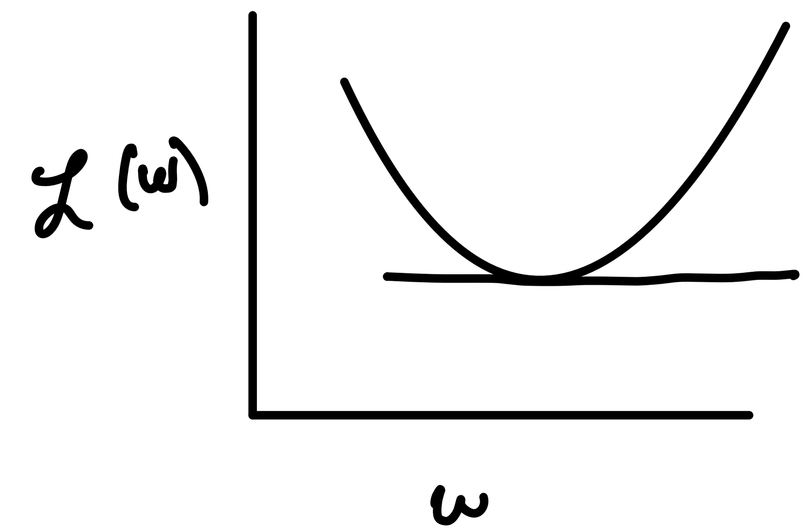
$$X = \begin{bmatrix} — x^{(1)T} — \\ — x^{(n)T} — \end{bmatrix}$$

$n \times d$

Xw

$n \times d \;\; d \times 1$

How do we find minimum of a function?



when derivative is 0

$$\nabla \mathcal{L}(w) = \begin{bmatrix} \frac{\partial \mathcal{L}(w)}{\partial w_1} \\ \frac{\partial \mathcal{L}(w)}{\partial w_2} \\ \vdots \\ \frac{\partial \mathcal{L}(w)}{\partial w_d} \end{bmatrix} \overset{\text{want}}{=} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

$$\mathcal{L}(w) = \frac{1}{2} \| y - Xw \|_2^2$$

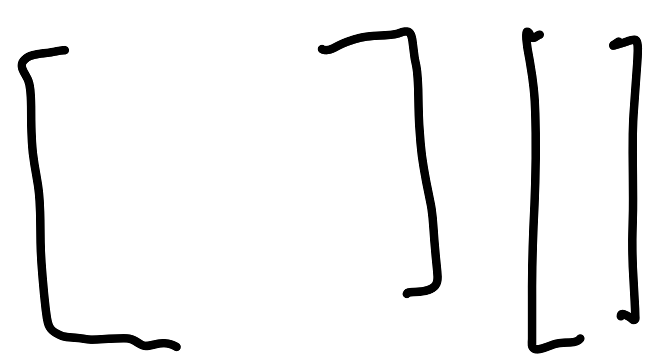
want $\nabla \mathcal{L}(w) = \underline{0}$

$$\nabla \mathcal{L}(w) = \underset{d \times n}{-X^T} \underset{n \times 1}{(y - Xw)}$$

$$\frac{d}{dw} \frac{1}{2} (y - xw)^2$$

$$= \frac{1}{2} \cdot 2 (y - xw) \frac{d (y - xw)}{dw}$$

$$= (y - xw) \cdot -x$$



$$\nabla \mathcal{L}(w) = \underline{0} = -X^T (y - Xw)$$

$$\underline{0} = -X^T y + X^T X w$$

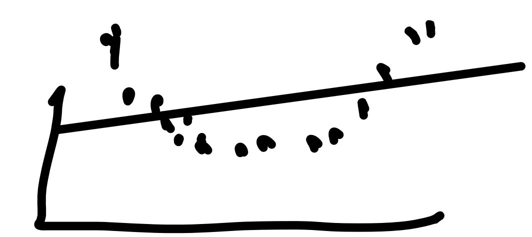
$$X^T y = X^T X w$$

③ $(X^T X)^{-1} X^T y = \underbrace{(X^T X)^{-1} (X^T X)}_{I} w$

$$= w$$

<u>Pros</u>: • linear is interpretable

<u>Cons</u>:



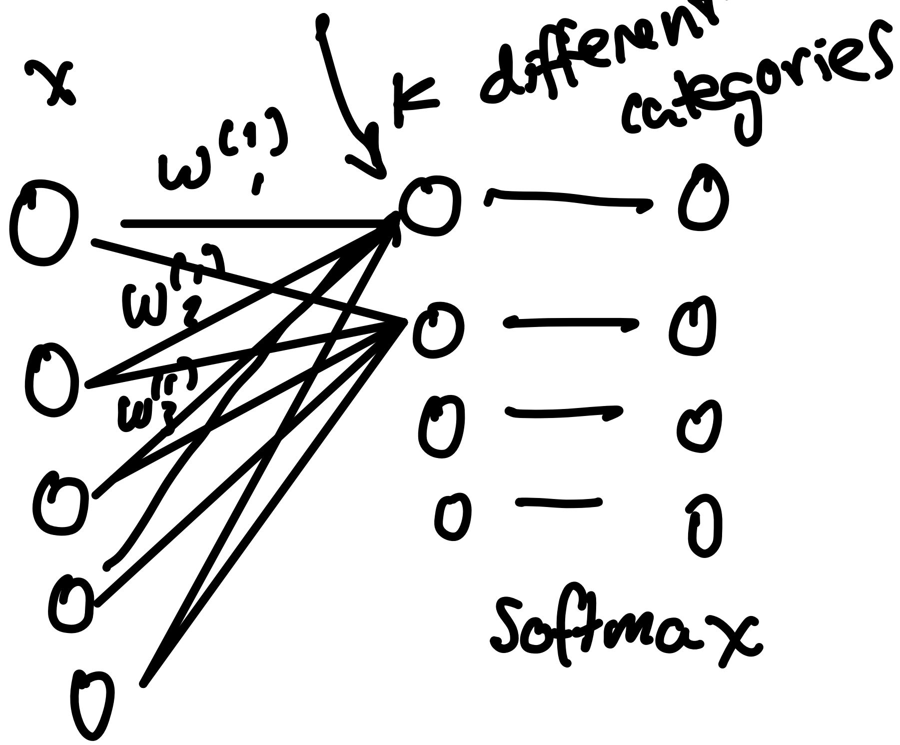
• not rich enough

• $\underset{d \times n}{X^T} \underset{n \times d}{X}$ $O(nd^2) + O(d^3)$

Logistic Regression

① output k values

$\langle w^{(1)}, x \rangle$



d

$$\text{softmax}(\langle w^{(1)}, x \rangle) = \frac{\exp(\langle w^{(i)}, x \rangle)}{\sum_{j=1}^{k} \exp(\langle w^{(j)}, x \rangle)}$$

probability after softmax :)

$$\sum_{i=1}^{k} \frac{\exp(\langle w^{(i)}, x \rangle)}{\sum_{j=1}^{k} \exp(\langle w^{(j)}, x \rangle)}$$

$$\frac{1}{\sum_{j=1}^{k} \exp(\langle w^{(j)}, x \rangle)} \sum_{i=1}^{k} \exp(\langle w^{(i)}, x \rangle)$$

$= 1$

② loss: cross entropy loss

$$\mathcal{L}(w) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \mathbb{1}[y^{(i)} = 1] \cdot -\log(f_w(x^{(i)}))$$

↑ indicator

1 if $y^{(i)} = 1$

0 else