CS60010: Deep Learning

Spring 2023

Sudeshna Sarkar

Module 2 Part A

Sudeshna Sarkar

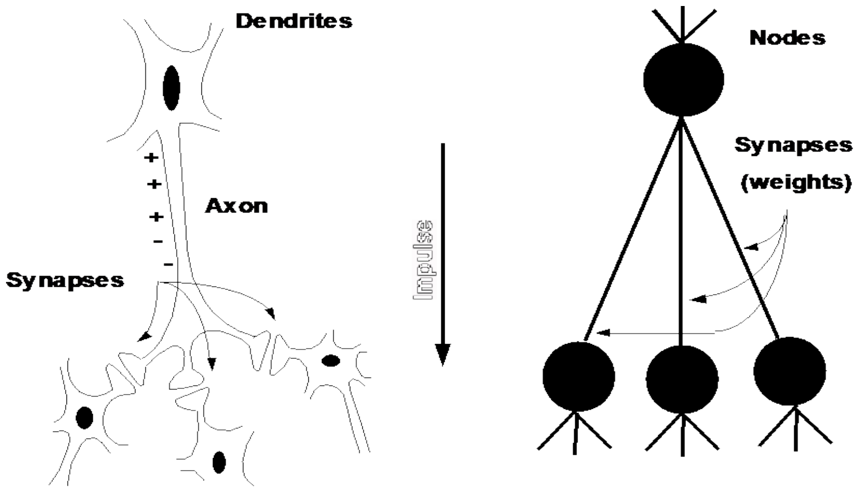
18 Jan 2023

Biological Inspiration

- Deep learning is inspired by neural networks of the brain to build learning machines which discover rich and useful internal representations, computed as a composition of learned features and functions
- Obtains its power by a nested hierarchy of concepts
  - each concept defined by relationship to simpler concepts
    - More abstract representations computed in terms of less abstract ones

Connectionist Models

- Consider humans:
  - Neuron switching time
    ~ 0.001 second
  - Number of neurons
    ~ 10^{10}
  - Connections per neuron
    ~ 10^{4-5}
  - Scene recognition time
    ~ 0.1 second
  - 100 inference steps doesn't seem like enough
    → much parallel computation
- Properties of artificial neural nets (ANN)
  - Many neuron-like threshold switching units
  - Many weighted interconnections among units
  - Highly parallel, distributed processes



Explain backpropagation through a poem.

Explain backpropagation through a poem.

Backpropagation is like a journey
Through the depths of a neural net
We start with the output, so hearty
And work our way to the first set

We calculate the error, so crucial
Then propagate it backwards with care
Adjusting weights, so dutiful
To minimize the error, our dear

We repeat this process, so steady
Til the error is at a minimum
Then our network is ready
For the tasks, it will soon come to consummation

Backpropagation, a technique so grand
Helps improve a neural net's command.

Explain backpropagation through a poem.

Backpropagation is like a journey
Through the depths of a neural net
We start with the output, so hearty
And work our way to the first set

We calculate the error, so crucial
Then propagate it backwards with care
Adjusting weights, so dutiful
To minimize the error, our dear

We repeat this process, so steady
Til the error is at a minimum
Then our network is ready
For the tasks, it will soon come to consummation

Backpropagation, a technique so grand
Helps improve a neural net's command.

GW

Write a poem about online convex optimization

Online convex optimization,
A powerful tool for computation,
It helps us find solutions,
To problems with changing conditions.

With its mathematical prowess,
It can tackle any problem we toss,
From machine learning to control systems,
It's a versatile tool, no loss.

It adapts to changing data,
In real-time it finds the best route,
No matter how complex the problem,
Online convex optimization knows how to sort it ou

So if you're facing a difficult task,
And you're not sure how to proceed,
Consider using online convex optimization,
It's a powerful tool indeed.

Stable Diffusion

DL in Science, Engineering, Games

- Genomics, Chemistry, Physics, Astronomy

Machine Learning

$\mathcal{X}$: A space of "observations" (Instance space)

$\mathcal{Y}$: space of "targets" or "labels"

How the observations determine the targets?

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

2. Choose each of these:

– Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

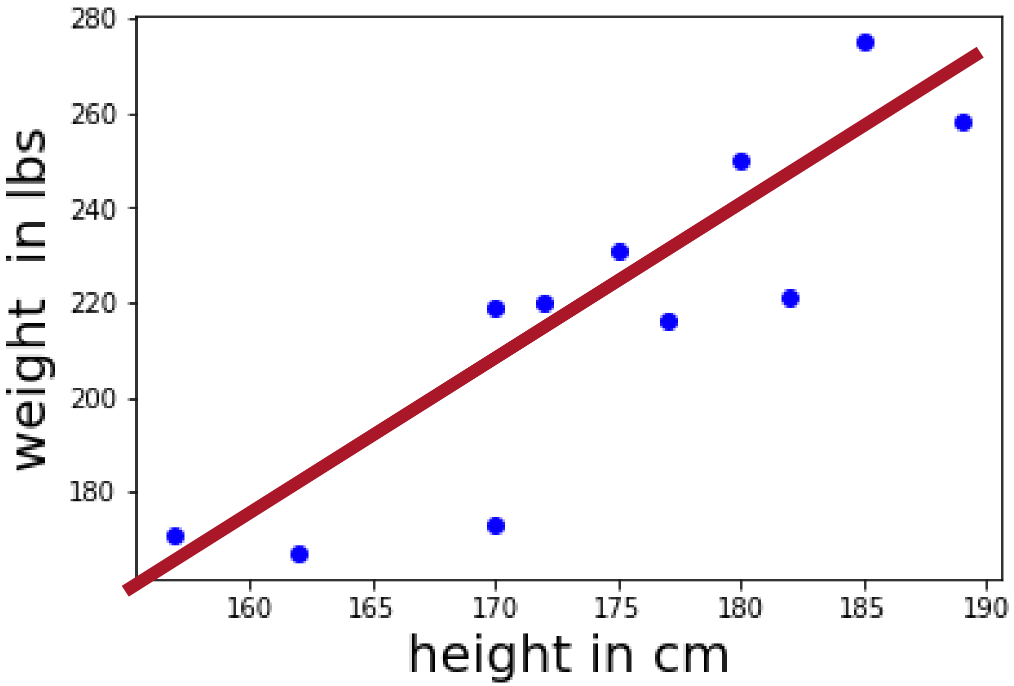
– Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Examples: Linear regression, Logistic regression, Neural Network

Examples: Mean-squared error, Cross Entropy

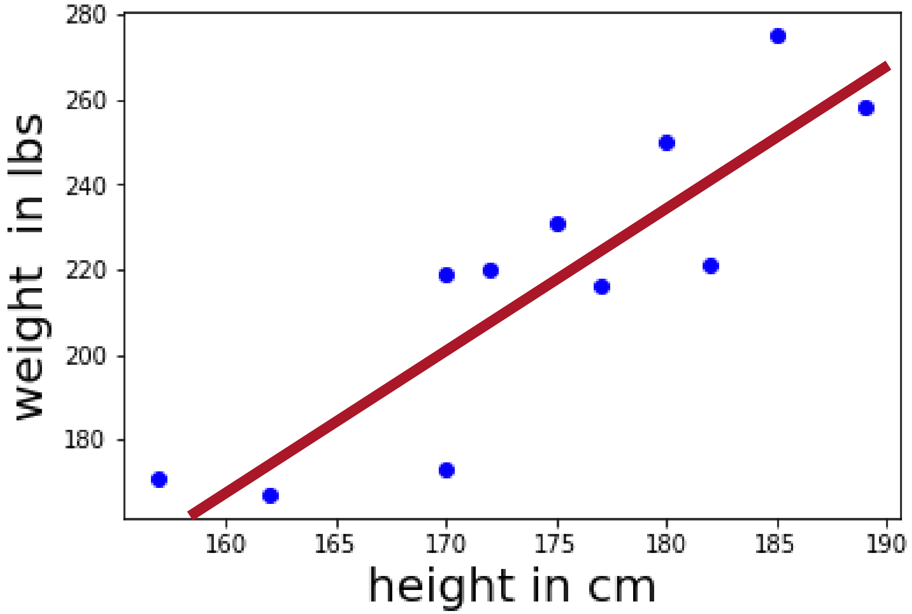
The machine learning method



1. Datapoints
2. Define your model class
   - Learnable parameters
3. Define your loss function
4. Pick your optimizer
5. Test “goodness” of learnt model on new (i.e., previously unseen) data

Least squares fit (Gauss ~1800)

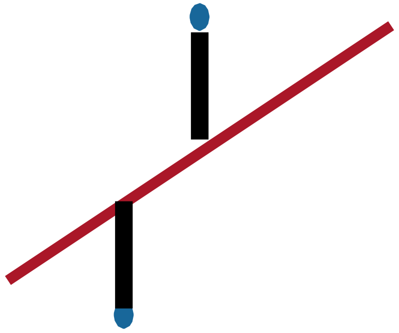
“Response variable”



“Predictor variable”

Class of models

$$\hat{y} = \theta_0 + \theta_1 x$$



(h, w)

“residual” = difference between actual value and model prediction.

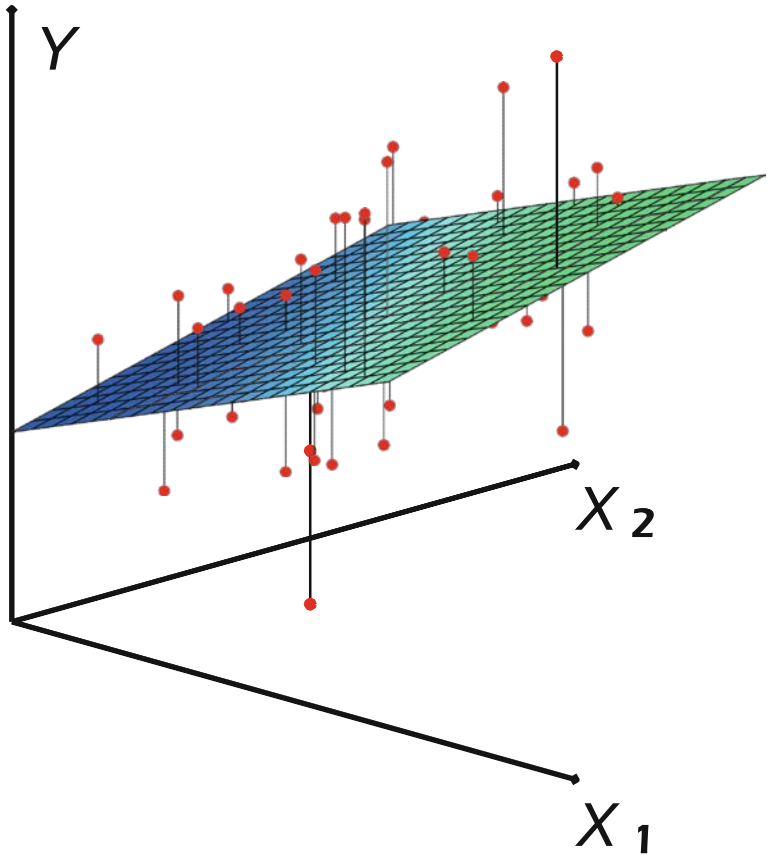
Minimize sum of squared residuals for given datapoints

$$\min_{\theta_0, \theta_1} (y - \theta_0 - \theta_1 x)^2$$

Test “goodness” of learnt model on new data.

Goodness of Model = Average Squared Residual of the model on male population

Linear models with 2 predictor variables



- Class of models:

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Optimization?

How is the dataset generated?

$\sim p(x)$: probability distribution over photos

"tree" $\sim p(y|x)$: conditional probability distribution over labels

Result: $(x, y) \sim p(x, y)$

How is the dataset generated?

- $(x, y) \sim p(x, y)$
- Training Set: $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
- $p(\mathcal{D})$?

every (x_i, y_i) independent of each (x_j, y_j)

key assumption: *independent* and *identically distributed* (i.i.d.)

$$\text{when i.i.d.: } p(\mathcal{D}) = \prod_i p(x_i, y_i)$$

$$= \prod_i p(x_i) p(y_i | x_i)$$

Generating the Dataset

$$p(\mathcal{D}) = \prod_i p(x_i) p_\theta(y_i|x_i)$$

multiplying together many numbers ≤ 1

$$\log p(\mathcal{D}) = \sum_i \log p(x_i) + \log p_\theta(y_i|x_i) = \sum_i \log p_\theta(y_i|x_i) + \cancel{\text{const}}$$

$$\theta^\star \leftarrow \arg\max_\theta \sum_i \log p_\theta(y_i|x_i)$$

maximum likelihood estimation (MLE)

$$\theta^\star \leftarrow \arg\min_\theta - \sum_i \log p_\theta(y_i|x_i)$$

negative log-likelihood (NLL)

this is our **loss function**!

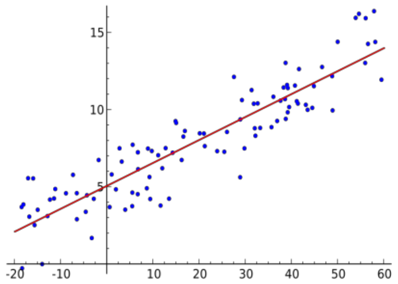
Loss functions

- The loss function quantifies how bad θ is.

negative log-likelihood: $-\sum_i \log p_\theta(y_i|x_i)$

zero-one loss: $\sum_i \delta(f_\theta(x_i) \neq y_i)$

mean squared error: $\sum_i \frac{1}{2}||f_\theta(x_i) - y_i||^2$



Prediction Functions

Inputs often referred to as predictors and features;

Outputs are known as targets and labels.

- The input and output variables are assumed to be related via a relation, known as hypothesis, $\hat{y} = h_\theta(x)$ or $\hat{y} = f(x;\ \theta)$

θ is the parameter vector.

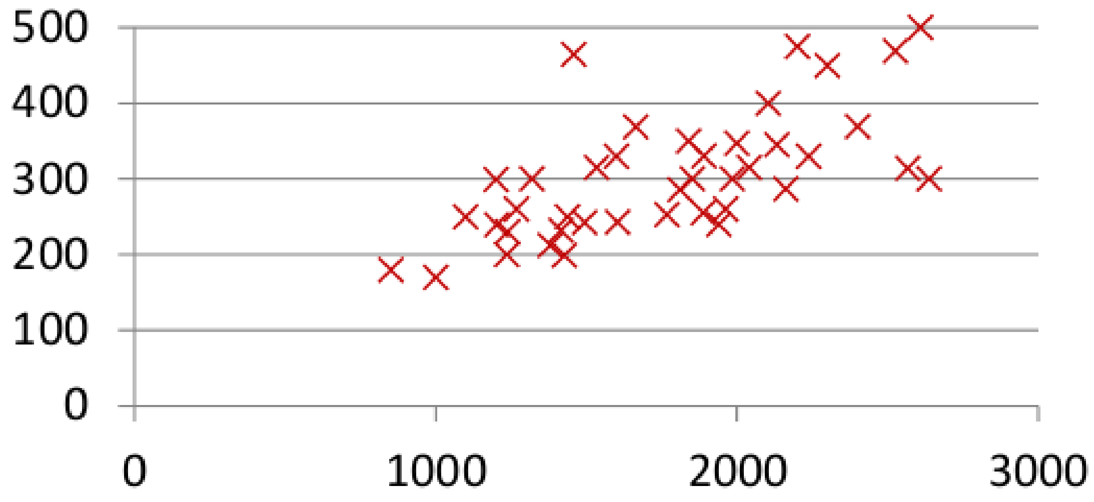
1. **Regression**: $y \in \mathcal{R}$ is a real value.
2. **Classifier**: y is the predicted class of x,
   and $y \in \{1, \dots, k\}$ is the class number.

Regression

Regression Examples:

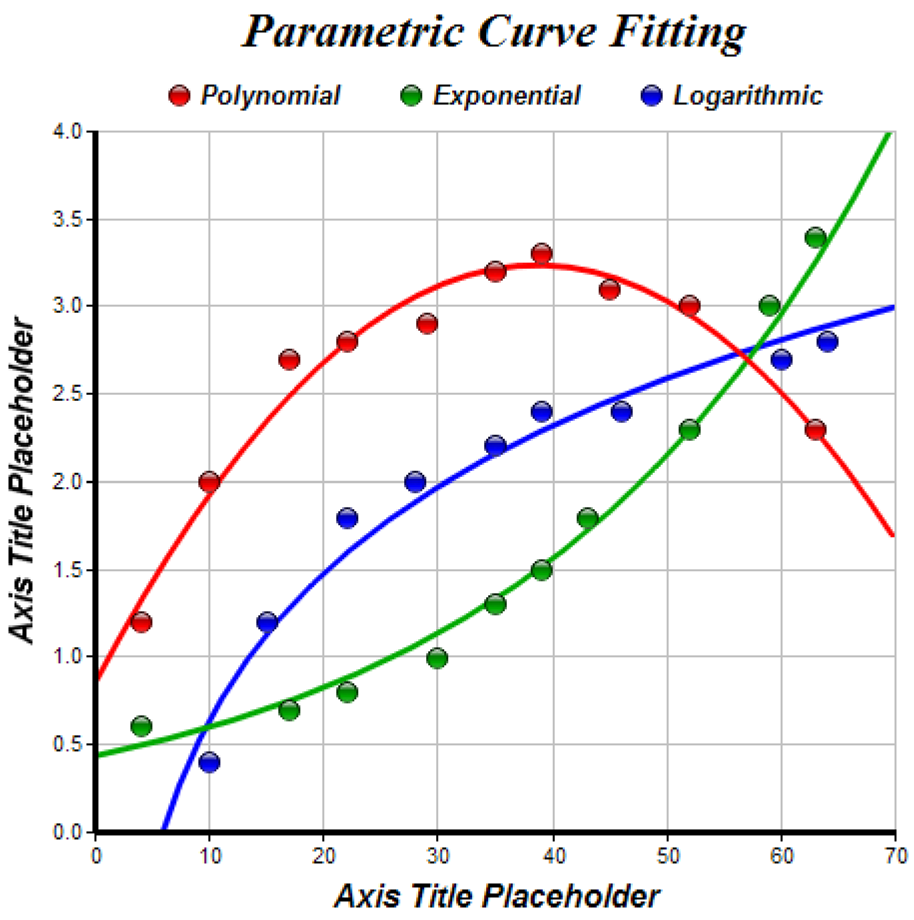
- (Outside temperature, People inside classroom, target room temperature | Energy requirement)
- (Size, Number of Bedrooms, Number of Floors, Age of the Home | Price)

A set of N observations of y as $\{y^{(1)}, \dots, y^{(m)}\}$ and the corresponding inputs $\{x^{(1)}, \dots, x^{(m)}\}$



Regression Problems

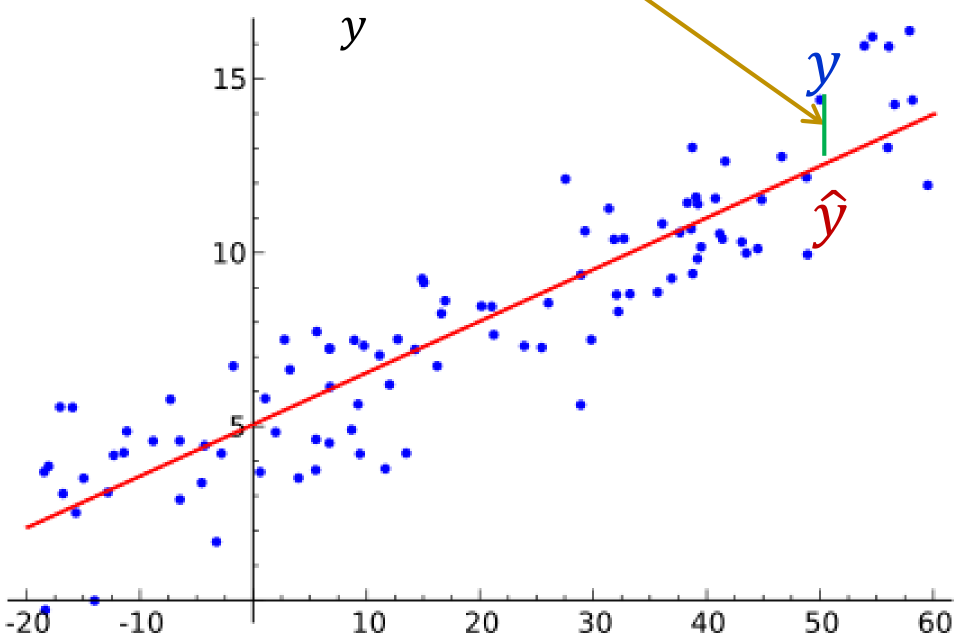
- Curve Fitting



Linear Regression

$$\hat{y} = \theta_0 + \theta_1 x$$

The loss is the squared loss $L_2(\hat{y}, y) = (\hat{y} - y)^2$



Data (x, y) pairs are the blue points.
The model is the red line.

x

Optimization objective: Find model parameters $\boldsymbol{\theta}$ that will minimize the loss.

Linear Regression

The total loss across all points is

$$L = \sum_{i=1}^{m} \left(\widehat{y^{(i)}} - y^{(i)}\right)^2$$

$$= \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

$$J(\theta_0, \theta_1) = \frac{1}{N} \sum_{i=1m} \left(f\left(x^{(i)};\ \theta\right) - y^{(i)}\right)^2$$

We want the optimum values of θ_0, θ_1 that will minimize the sum of squared errors. Two approaches:

1. Analytical solution via mean squared error
2. Iterative solution via MLE and gradient ascent

Linear Regression: Analytic Solution

Since the loss is differentiable, we set

$$\frac{dL}{d\theta_0} = 0 \quad \text{and} \quad \frac{dL}{d\theta_1} = 0$$

We want the loss-minimizing values of $\boldsymbol{\theta}$, so we set

$$\frac{dL}{d\theta_1} = 0 = 2\theta_1 \sum_{i=1}^{N} \left(x^{(i)}\right)^2 + 2\theta_0 \sum_{i=1}^{N} x^{(i)} - 2 \sum_{i=1}^{N} x^{(i)} y^{(i)}$$

$$\frac{dL}{d\theta_0} = 0 = 2\theta_1 \sum_{i=1}^{N} x^{(i)} + 2\theta_0 N - 2 \sum_{i=1}^{N} y^{(i)}$$

These being linear equations of θ, have a unique closed form solution

$$\theta_1 = \frac{m \sum_{i=1}^{m} y^{(i)} x^{(i)} - \left(\sum_{i=1}^{m} x^{(i)}\right)\left(\sum_{i=1}^{m} y^{(i)}\right)}{m \sum_{i=1}^{m} (x^{(i)})^2 - \left(\sum_{i=1}^{m} x^{(i)}\right)^2}$$

$$\theta_0 = \frac{1}{m}\left(\sum_{i=1}^{m} y^{(i)} - \theta_1 \sum_{i=1}^{m} x^{(i)}\right)$$

Risk Minimization

We found θ_0, θ_1 which minimize the squared loss on data ***we already have***. What we actually minimized was an averaged loss across a finite number of data points. This averaged loss is called **empirical risk.**

What we really want to do is predict the y values for points x ***we haven't seen yet***. i.e. minimize the expected loss on some new data:

$$E[(\hat{y} - y)^2]$$

The expected loss is called **risk**.

Machine learning approximates risk-minimizing models with empirical-risk minimizing ones.

Risk Minimization

Generally minimizing empirical risk (loss on the data) instead of true risk works fine, but it can fail if:

- The **data sample is biased**. e.g. you cant build a (good) classifier with observations of only one class.
- There is **not enough data** to accurately estimate the parameters of the model. Depends on the complexity (number of parameters, variation in gradients, complexity of the loss function, generative vs. discriminative etc.).