Statistics and Artificial Intelligence

Lecture 3: Logistic Regression as a Neural Network

Yixin Wang

Goals for Today

Logistic Regression = handles binary classification (0, 1)

- Suggestions and Questions from JiTTs
- Logistic regression as a neural network
- Loss functions
- Gradient descent core algorithm to train neural net
- Derivatives with a computational graph
- Logistic regression with gradient descent

Preliminary Draft.
Please do not distribute.

Suggestions and Questions from JiTTs

Readings, JiTTs, Lectures, Quizzes, and Homework Assignments

- Relationship between lectures, readings, quizzes, homework assignments, and projects
  - **Lectures** offer a road map that highlights important concepts and methods that are essential for understanding and implementing deep learning algorithms.
  - **Readings** offer more extensive descriptions, examples, and details about algorithms.

    ↳ Good preview material from lecture
  - **Quizzes** are mostly about basic concepts.

    ↳ Quiz 1 is ungraded but must submit!
  - **Labs** and **homework** support your learning of algorithmic implementations through examples and exercises.

    ↳ Focus on implementation
  - In **projects**, you find your own dataset and implement your own deep learning algorithms.

Quiz: 5Q, 20 minute, open course material

Suggestions and Questions from JiTTs

Lectures

- Speak slower, write bigger, repeat the question.
  - Gotcha, I’ll try to adjust! Thanks!
  - Please remind me if you couldn’t hear or if it is too fast.
  - Please continue to offer me feedback to improve!
  - We very much appreciate it.

Suggestions and Questions from JiTTs

JiTTs

yay!

- We have updated the syllabus to drop the lowest four scores for JiTTs.
- Please feel free to use JiTTs to guide your reading, e.g. what are important concepts to get out of the readings.
- It mainly serves as a quick check-in before class, so it is not meant to take you more than 10mins to complete. Please feel free to use online resources too.

JiTT = graded for effort, not accuracy

- guide reading, highlight most important concepts
- open to all resources

Suggestions and Questions from JiTTs

Homework assignments and Programming

- Homework assignments offers tutorial and hands on experience on building different parts of deep learning models.
- If you are not familiar with programming or python, it could feel very intense.
- We will release homework 2 very soon. It is typical of all the weekly homework assignments.
- If it feels too hard, then maybe it is worth waiting until another semester to take this class. (This class will be offered every semester.)

HW 2 - TensorFlow, covered in lab this week

· HW 2 released on Canvas

Suggestions and Questions from JiTTs

Workload

- This is a heavy workload class. But it is not because we intentionally made it so :-)
- Let me explain why.
  - It is heavy workload mainly because of the subject matter.
  - Deep learning is fundamentally computational (as opposed to theoretical), so it has a very different nature than other classes you have taken.
  - **Its computational nature makes it very heavy workload if you are not familiar with programming.**

Suggestions and Questions from JiTTs

Linear algebra and multivariable calculus

- We will have a linear algebra bootcamp and multivariable calculus bootcamp to help you ramp up the knowledge.
- I'd encourage you to start engaging with these linear algebra and multivariable calculus ideas early. It takes a while to feel comfortable with these ideas.
- Some good resources:
  - The linear algebra bootcamp and the multivariable calculus bootcamp from the previous offering of the course: https://ambujtewari.github.io/stats315-winter2022/
  - The relevant sections in the text book (e.g. D2L, Sec. 19.1.1-9 for linear algebra and D2L, Sec. 19.4 for multivariable calculus)

Suggestions and Questions from JiTTs

Projects

- To look for project datasets, I'd suggest you look at kaggle.com (demo)
- Please read the newly updated **final project guidelines** on the course website for detailed requirements.
- Please feel free to talk to the teaching team about project-related questions during office hours.
- Some of you asked for a "database" of the projects. Your homework assignments, which teach you how to implement models and perform analysis. Kaggle can help as such a database too.

Suggestions and Questions from JiTTs

Other questions from JiTTs

- We will discuss the most common questions in class.
- If your questions are not addressed in class within a week, please make a piazza post about it!

Questions?

Training Neural Networks

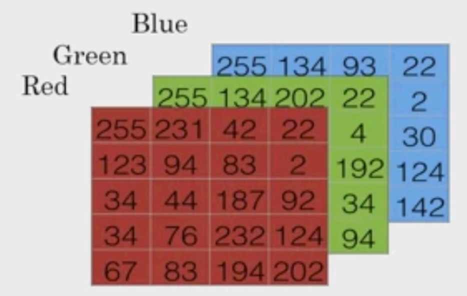
Logistic Regression as a Toy Neural Network

- Idea 1 handle whole dataset as a unit to feed into net → don't loop over every example
  - process your entire training set without using an explicit for loop to loop over your entire training set
- Idea 2
  - why the computations in learning a neural network can be organized in a forward propagation and a separate backward propagation
- etc.
- We'll convey these ideas using logistic regression to make them easier to understand.

Logistic Regression: statistical model for binary classification

Binary Classification

Logistic regression is for binary classification



255	231	42	22
123	94	83	2
34	44	187	92
34	76	232	124
67	83	194	202

- Input: Image X (Unroll pixel values into feature vector)
- Output: Label Y takes value 1 (cat) or 0 (non-cat)

Take in images → decide if there's a cat in them

- get values for each pixel
- Output label of 0 or 1

n_x = dimensionality of input vector

Image to Input Feature Vector

convert image to an input feature vector

$x^i = [n, n, n \ldots]$

each training sample is a vector of numbers

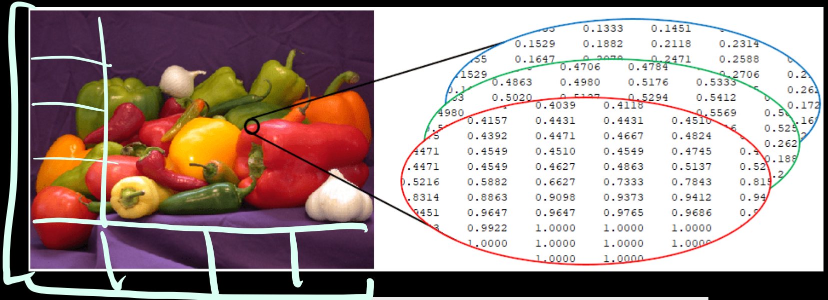
<u>image</u>

5 x 4 pixels

$\Rightarrow \underbrace{3}_{\text{channel}} \cdot 5 \cdot \underbrace{4}_{\text{strength}}$ numbers

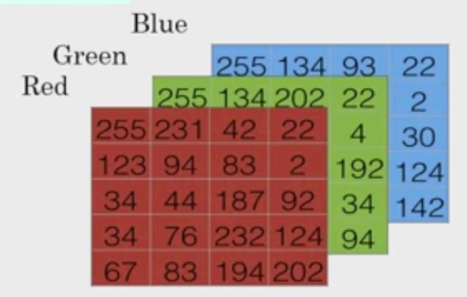
$$x = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ 255 \\ 134 \\ \vdots \\ 255 \end{bmatrix} \begin{matrix} \text{red } 5\times4 \\ \\ \\ \text{green } 5\times4 \\ \text{blue } 5\times4 \end{matrix}$$

Total 60 entries



Cell = how red pixel is

Larger # = stronger color



Red

255	231	42	22
123	94	83	2
34	44	187	92
34	76	232	124
67	83	194	202

Green

255	134	202	22
			4
			192
			34
			94

Blue

255	134	93	22
			2
			30
			124
			142

3 color channels red, blue, green

Some Notations We'll Use

what does the whole dataset look like?

Datapoint: (x, y)

x ↓ input vector

y ↓ label

$x \to \mathbb{R}^{n_x}$

$y \to \{0, 1\}$

Dataset has m training examples

- m or m_{train}

m_{test} = # test examples

$$\{(x^1 y^1), (x^2, y^2), \ldots, (x^m, y^m)\}$$

→ whole data set

organize into 2 matrix

$$Y = \begin{bmatrix} y^1 & y^2 & \cdots & y^m \end{bmatrix} \quad 1 \times m \text{ dimension}$$

$$y \in \{0, 1\}^m$$

Shape $(1, m)$

$$X = \underbrace{\begin{bmatrix} | & | & & | \\ x^1 & x^2 & \cdots & x^m \\ | & | & & | \end{bmatrix}}_{m \text{ dimension}} \Bigg\} \; n_x \text{ dimension}$$

shape (n_x, m)

each image is a column

Logistic Regression

Scalar and matrix version

$$X = \underbrace{\begin{bmatrix} | & | & & | \\ x^1 & x^2 & \cdots & x^m \\ | & | & & | \end{bmatrix}}_{\leftarrow \; m \; \rightarrow} \begin{matrix} \uparrow \\ n_x \\ \downarrow \end{matrix}$$

$X.\text{shape} = (n_x, m)$

$$\underbrace{\begin{bmatrix} - & x^1 & - \\ - & x^2 & - \\ & \vdots & \\ - & x^m & - \end{bmatrix}}_{\leftarrow \; n_x \; \rightarrow} \begin{matrix} \uparrow \\ m \\ \downarrow \end{matrix}$$

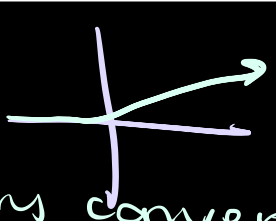
Each example is a row

we don't use this notation

Logistic Regression

Alternative notation (that we will not use)

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



slowly converts large # to 1

$$\hat{y} = \underbrace{\sigma}_{\text{sigmoid}}\Big(\underbrace{\sum_{j=1}^{n} w_j x_j}_{\text{weighted combo}} + \underbrace{b}_{\text{bias}}\Big)$$

$$Z = \begin{bmatrix} z_1^i \\ z_2^i \\ \vdots \\ z_{n_x}^i \end{bmatrix}$$

Each x_j is an input neuron from a previous layer

Goal

Given the dataset, want to find parameters so

$\hat{y}_i \approx y_i$

parameters:

$w_1, \dots, w_{n_x}$ + b

Poll

- What are the parameters of logistic regression?
  - W, an n_x dimensional vector, and b, a real number.
  - W, an identity vector, and b, a real number.
  - W and b, both real numbers.
  - W and b, both n_x dimensional vectors.

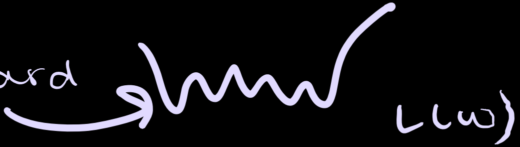
Logistic Regression Cost Function

Cost function / Loss function

$\hat{y}_i$ y_i

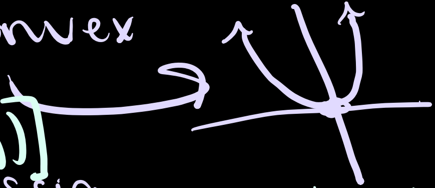
- Loss function measures how good the prediction is, relative to the true label.
- Training: Make the cost function as small as possible
- Why this loss function?

$L(\hat{y},y) = \Sigma(\hat{y}-y)^2$ non-convex, hard to optimize



nonconvex

convex



$L(\hat{y},y) = -[y \cdot \log(\hat{y}) + (1-y) \cdot \log(1-\hat{y})]$

loss function we use in logistic regression

convex

→ cross entropy loss

$y=1$ → $L(\hat{y},y) = -\log(\hat{y})$ → minimize when $\hat{y}$ is large

Logistic Regression Cost Function

Cost function / Loss function

- Loss function measures how good the prediction is, relative to the true label.
- Training: Make the cost function as small as possible
- Why this loss function?

$y=0 \Rightarrow L(\hat{y},y) = -\log(1-\hat{y})$

- to minimize this, we want $\hat{y}$ to be very small

Logistic Regression Cost Function

Cost function / Loss function

- Loss function (defined for a single training example);
- Cost function (defined for the whole training set): Cost for parameters

Cost $J(w,b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^i, y^i)$

Average loss for each sample over dataset

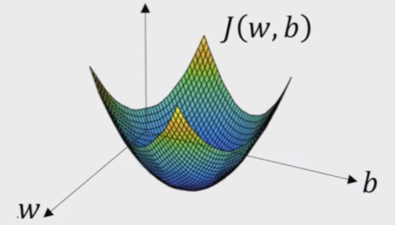
Logistic Regression Cost Function

Cost function / Loss function

- Loss function measures how good the prediction is, relative to the true label.
- Training: Make the cost function as small as possible
- Why this loss function?

Gradient Descent for Training Neural Network

- Gradient Descent:
  - Initialize;
  - Take a step in the steepest downhill direction at each iteration
  - Repeat until the algorithm converges



https://math.stackexchange.com/questions/1582452/logistic-regression-prove-that-the-cost-function-is-convex

Gradient Descent

- Learning rate: Control how big a step we take at each iteration
- Derivative: Slope of the function; the direction to go downhill
- Code: Often use ‘dw’ to represent the derivative dJ(w)/dw
    - The amount you want to update for w

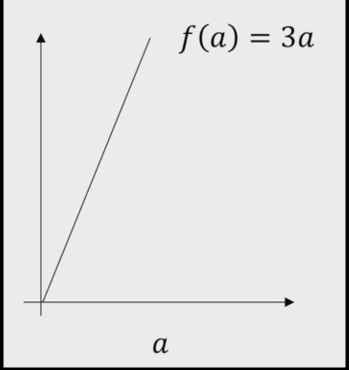
Question

- True or false. A convex function always has multiple local optima.

Intuition about Derivatives

$f(a) = 3a$

- Slope / Derivative: If I nudge a by a tiny little bit, I expect $f(a)$ to move three times as large as the nudge I gave a.

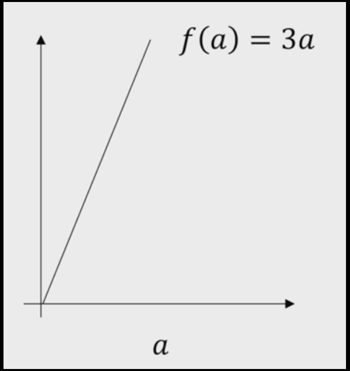


Intuition about Derivatives

$f(a) = 3a$

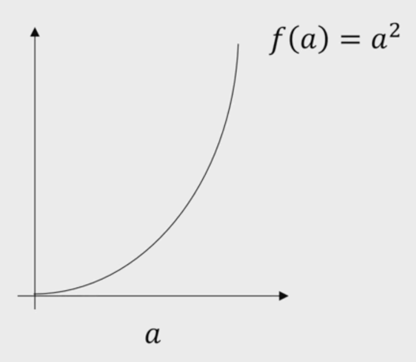
- Formal definition through infinitesimal nudge.

- This function $f(a) = 3a$ has constant slope.



Intuition about Derivatives

$f(a) = a^2$



Preliminary Draft.
Please do not distribute.

More Derivative Examples

Preliminary Draft.
Please do not distribute.

Computation Graph

- The computations of a neural network are organized in terms of
  - a forward pass or a forward propagation step, in which we compute the output of the neural network,
  - followed by a backward pass or back propagation step, which we use to compute gradients or compute derivatives.
- The computation graph explains why it is organized this way.

Computational Graph

Left-to-right pass

Computing Derivatives with a Computation Graph

Right-to-left pass (Chain rule)

- Backpropagation

Logistic Regression Gradient Descent

Logistic regression recap

Logistic Regression Gradient Descent

Logistic regression derivatives

Logistic Regression Gradient Descent

Logistic regression on m examples

Logistic Regression Gradient Descent

Logistic regression on m examples

The materials in this course are adapted from materials created by Alexander Amini, Alfredo Canziani, Justin Johnson, Andrew Ng, Bhiksha Raj, Grant Sanderson and the 3blue1brown channel, Rita Singh, Ava Soleimany, and Ambuj Tewari.