CS 260: Foundations of Data Science

Prof. Sara Mathieson

Fall 2023

HAVERFORD
COLLEGE

Admin

- In lab today:
  - Project meetings with all groups
  - Try to come to the same lab as your partner

Admin

- In lab today:
  - Project meetings with all groups
  - Try to come to the same lab as your partner

- Flexibility for this class:
  - I will drop the lowest lab grade
  - Final project can be simplified as needed
  - Accommodations or flexibility beyond this: collaboration with class deans

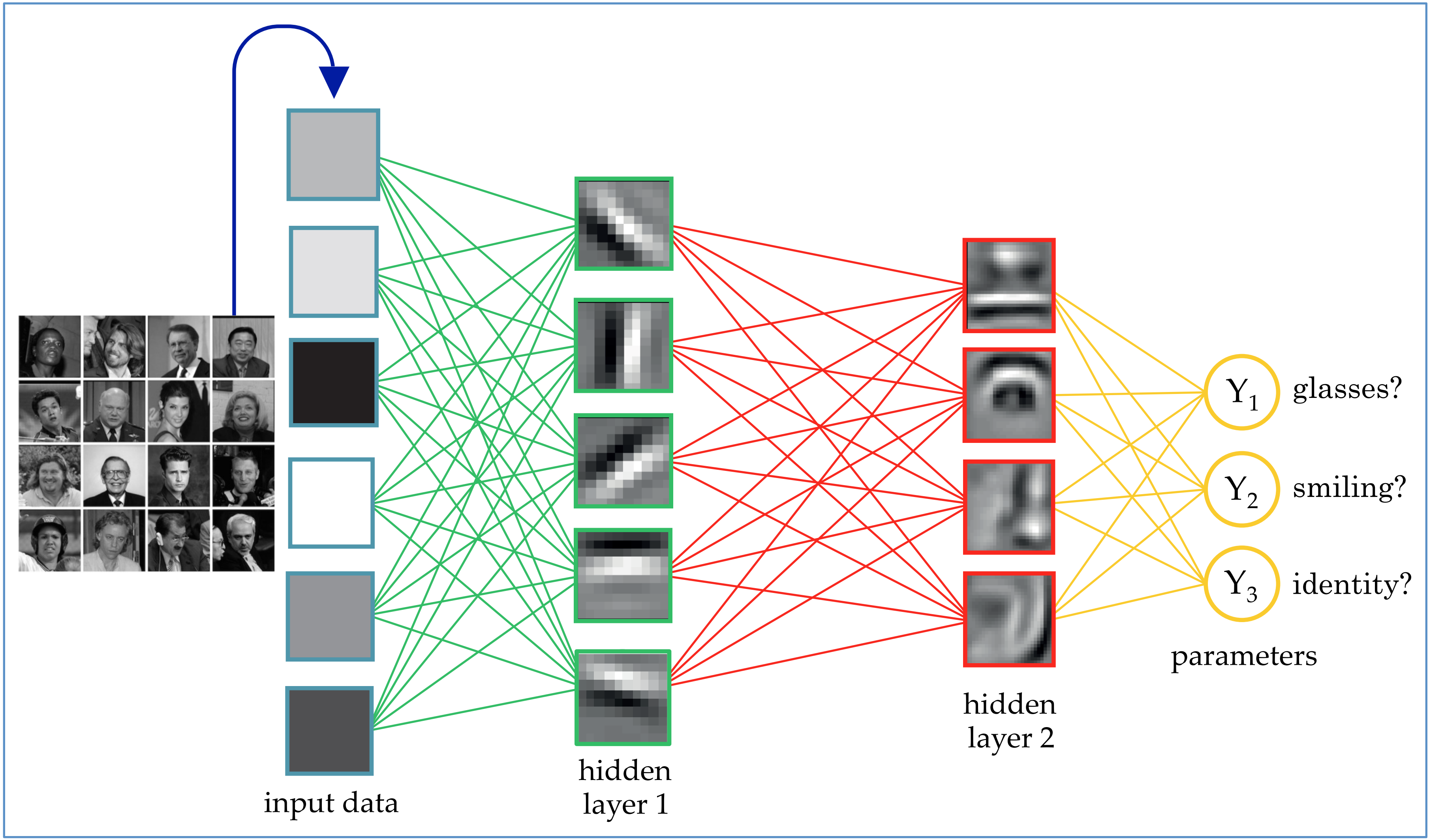
Outline for December 5

- Finish: neural networks
- Advice about git for final project
- Go over Midterm 2

Outline for December 5

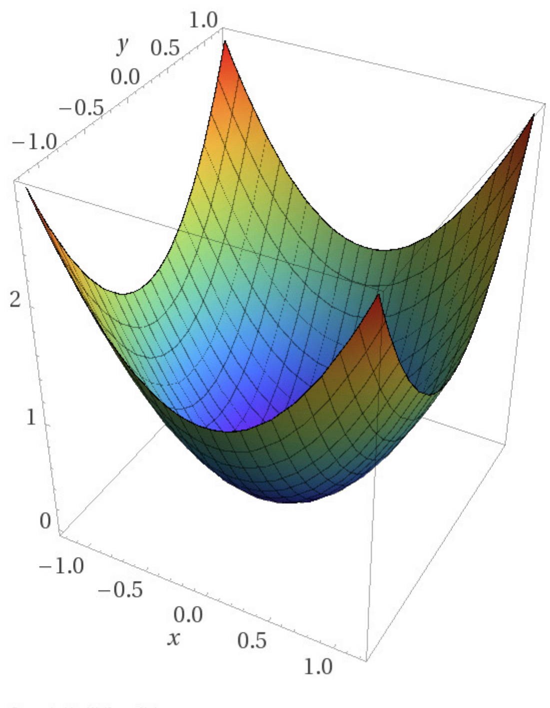
- Finish: neural networks
- Advice about git for final project
- Go over Midterm 2

First fully connected neural networks for images

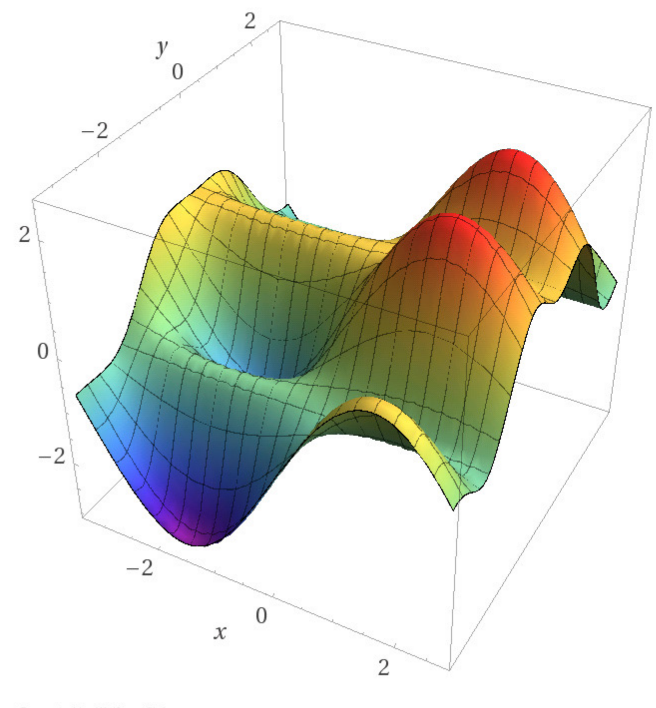


Takeaways from last time

- As the number of parameters grows, a non-convex function often has more and more local minima
- Starting at a “good” point is crucial!



Convex



Non-convex

Image: O’Reilly Media

Takeaways from last time

- Unsupervised pre-training uses latent structure in the data as a starting point for weight initialization
- After this process, the network is “fine-tuned”
- In practice this has been found to increase accuracy on specific tasks (which could be specified after feature learning)

Weight initialization

- We still have to initialize the pre-training
- All 0's initialization is bad! Causes nodes to compute the same outputs, so then the weights go through the same updates during gradient descent
- Need asymmetry! => usually use small random values

Mini-batches

- So far in this class, we have considered *stochastic gradient descent*, where one data point is used to compute the gradient and update the weights
- On the flipside is *batch gradient descent*, where we compute the gradient with respect to all the data, and then update the weights
- A middle ground uses *mini-batches* of examples before updating the weights

Notes about scores and softmax

- The output of the final fully connected layer is a vector of length K (number of classes)

Notes about scores and softmax

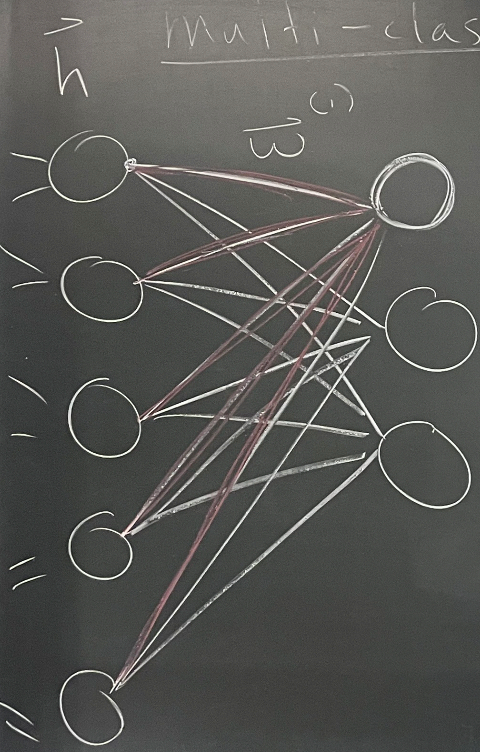
- The output of the final fully connected layer is a vector of length *K* (number of classes)

- The raw scores are transformed into probabilities using the *softmax function*: (let s_k be the score for class *k*)

$$\hat{y}_k = \frac{e^{s_k}}{\sum_{j=1}^{K} e^{s_j}}$$

- Then we apply *cross-entropy loss* to these probabilities

multi-class classification



$y=1$ (dog)

$y=2$ (cat)

$y=3$ (bird)

(score)

$$y = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\left.\begin{aligned} S_1 &= \vec{w}^{(1)} \cdot \vec{h} \\ S_2 &= \vec{w}^{(2)} \cdot \vec{h} \\ S_3 &= \vec{w}^{(3)} \cdot \vec{h} \end{aligned}\right\} \text{linear functions}$$

$$\begin{bmatrix} S_1 \\ S_2 \\ S_3 \end{bmatrix} = \begin{bmatrix} -7.2 \\ 10.9 \\ 0.71 \end{bmatrix} \leftarrow \boxed{\hat{y} = 2} \text{ (cat)}$$

argmax

softmax

"soft" (probability)

prob?

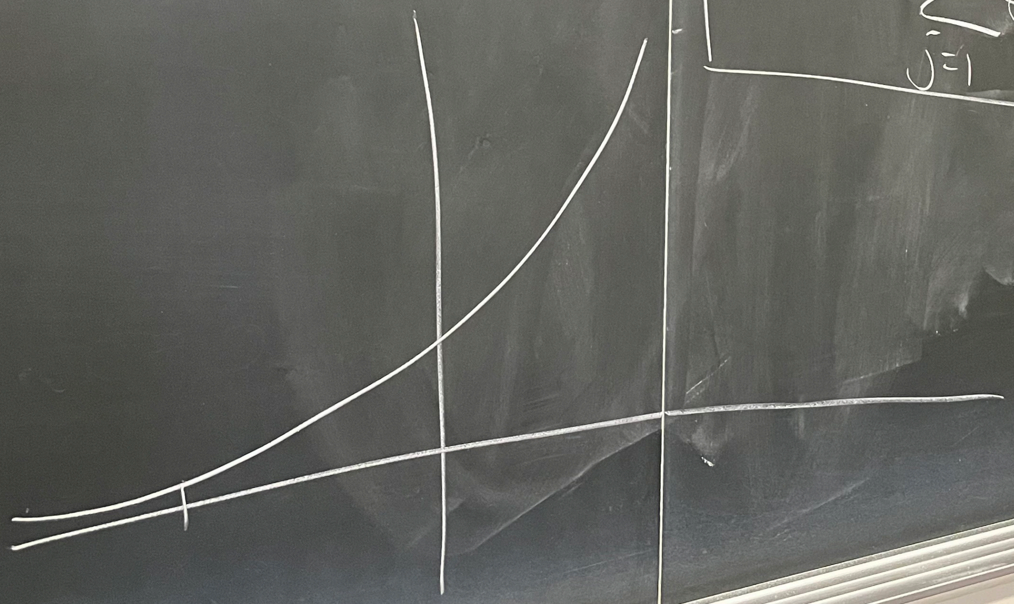
$$P_1 = \cancel{\frac{-7.2}{-7.2+10.9+0.71}}$$

negative prob!

softmax:

$$\boxed{\hat{y}_k = \frac{e^{s_k}}{\sum_{j=1}^{k} e^{s_j}}}$$

$$\frac{e^{-7.2}}{e^{-7.2}+e^{10.9}+e^{0.71}}$$



$$\frac{1}{1+e^{-x}}$$

Motivation for moving away from FC architectures

- For a 32x32x3 image (very small!) we have p=3072 features in the input layer

- For a 200x200x3 image, we would have p=120,000! *doesn't scale*

Motivation for moving away from FC architectures

- For a 32x32x3 image (very small!) we have p=3072 features in the input layer
- For a 200x200x3 image, we would have p=120,000! *doesn't scale*
- FC networks do not explicitly account for the structure of an image and the correlations/ relationships between nearby pixels

Idea: 3D volumes of neurons

- Do not “flatten” image, keep it as a volume with *width*, *height*, and *depth*

Idea: 3D volumes of neurons

- Do not "flatten" image, keep it as a volume with *width*, *height*, and *depth*
- For **CIFAR-10**, we would have:
  - Width=32, Height=32, Depth=3

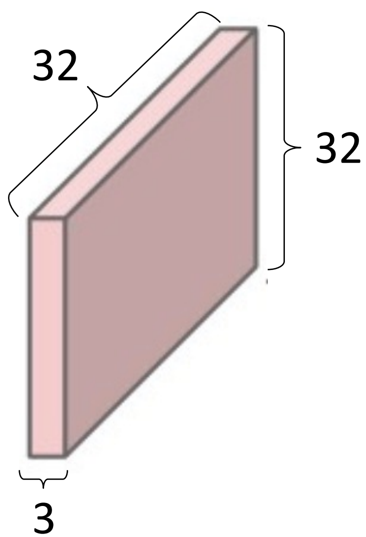


Image: modified from Stanford Course CS231n: http://cs231n.github.io/convolutional-networks/

Idea: 3D volumes of neurons

- Do not "flatten" image, keep it as a volume with *width*, *height*, and *depth*
- For **CIFAR-10**, we would have:
  - Width=32, Height=32, Depth=3
- Each layer is also a 3 dimensional volume

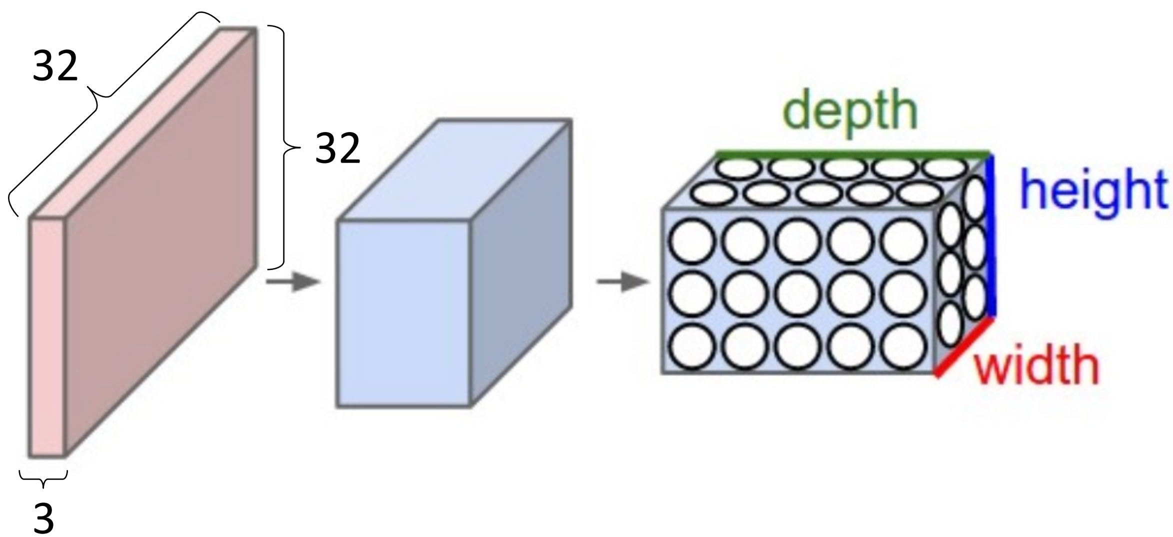


Image: modified from Stanford Course CS231n: http://cs231n.github.io/convolutional-networks/

Idea: 3D volumes of neurons

- Do not "flatten" image, keep it as a volume with *width*, *height*, and *depth*
- For **CIFAR-10**, we would have:
  - Width=32, Height=32, Depth=3
- Each layer is also a 3 dimensional volume
- The output layer is 1x1xC, where C is the number of classes (10 for CIFAR-10)

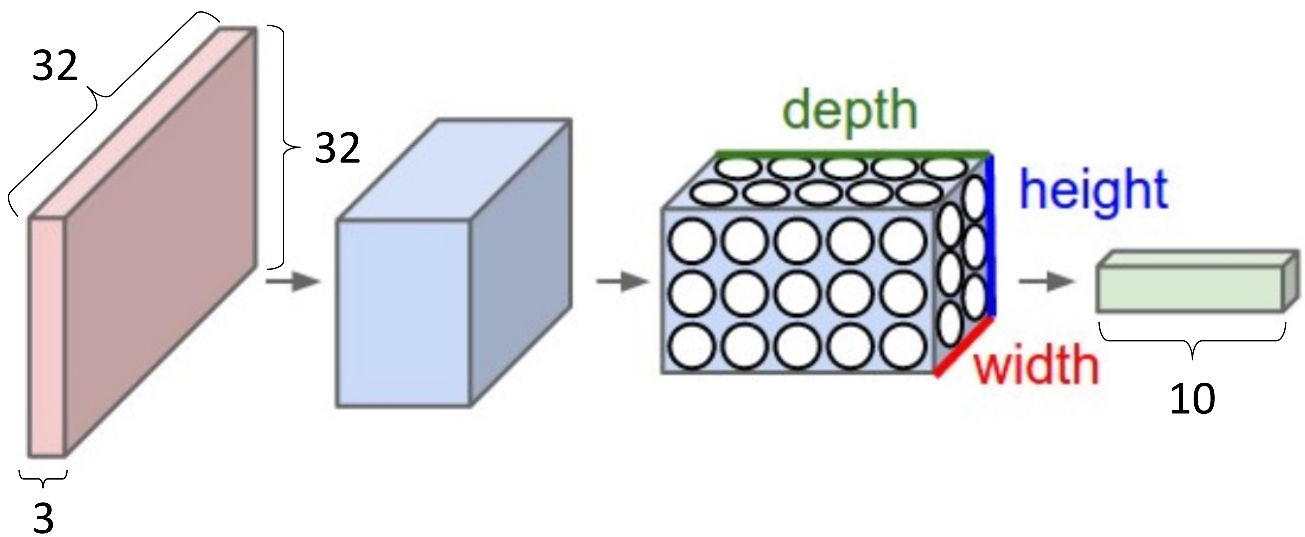


Image: modified from Stanford Course CS231n: http://cs231n.github.io/convolutional-networks/

Outline for December 5

- Finish: neural networks
- Advice about git for final project
- Go over Midterm 2

Github workflow with your partner

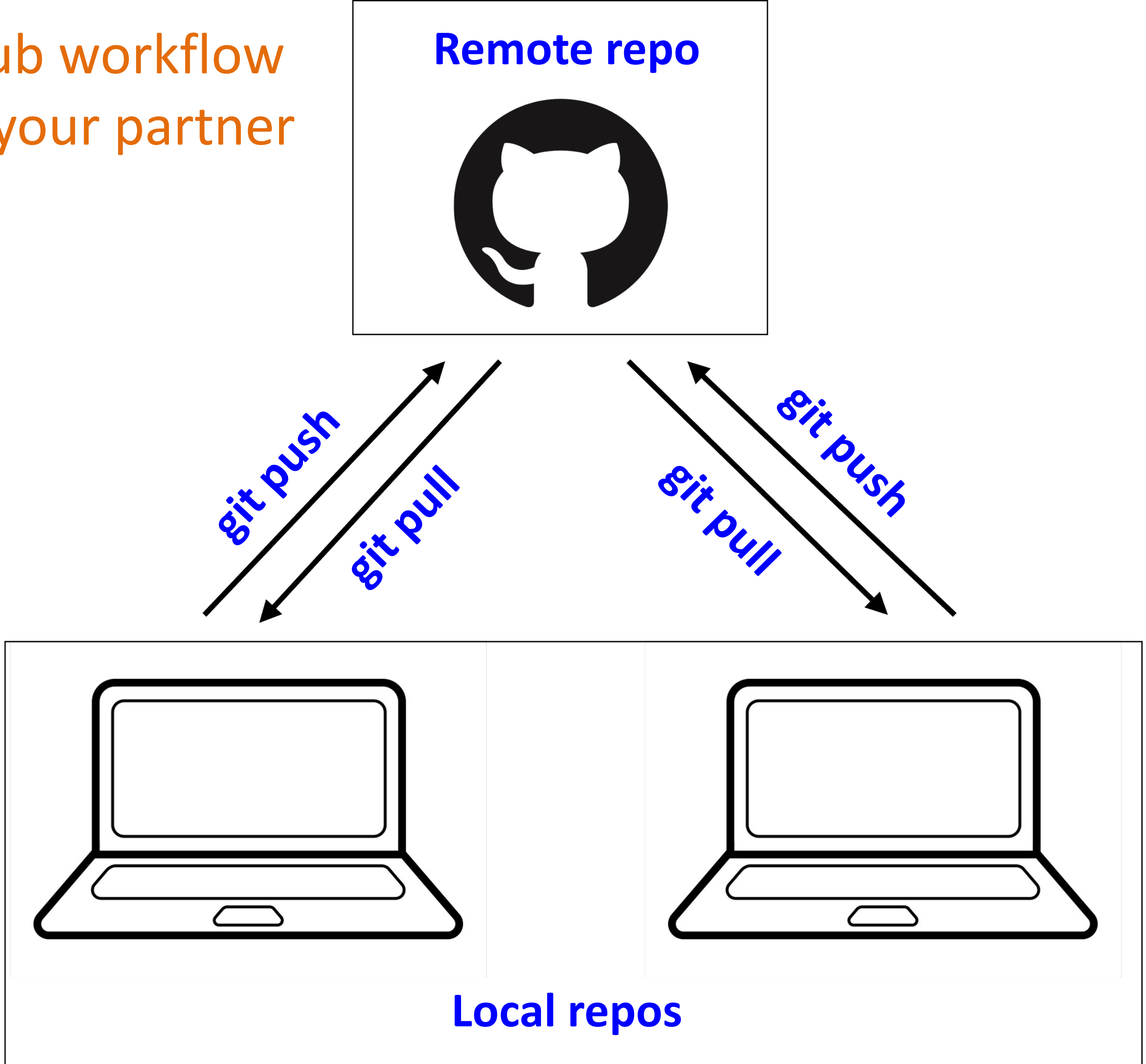


Image: modified from https://gcapes.github.io/

Outline for December 5

- Finish: neural networks
- Advice about git for final project
- Go over Midterm 2

1e $A = \left[\boxed{2} \right]_{p \times p}$

$$\rightarrow \frac{1}{n-1} \sum_{i=1}^{n} (f_1 - \bar{f}_1)(f_2 - \bar{f}_2)$$

$$\Rightarrow \boxed{O(np^2)}$$

3b cumulative prob $\boxed{0.625}$

$$0.\boxed{1} \cdot 2^{-1} + \boxed{0} \cdot 2^{-2} + \boxed{1} \cdot 2^{-3}$$

$$\begin{array}{r} 0.625 \\ -\ 0.5 \\ \hline 0.125 \end{array} \rightarrow \cancel{\begin{array}{r} 0.125 \\ -\ 0.25 \\ \hline \end{array}} \rightarrow \begin{array}{r} 0.125 \\ -\ 0.125 \\ \hline 0 \end{array}$$

$$\Rightarrow \boxed{0.\boxed{101}000\ldots}$$

(3e) $0.375 \cdot 2 + (0.25 \cdot 2) \cdot 2 + 0.125 \cdot 3 = \boxed{2.125}$

(4) $P = \text{pos}$ $\quad H = \text{healthy}$

$N = \text{neg}$ $\quad D = \text{disease}$

$$x = P(p \mid D) = \frac{p(p)\,p(D \mid p)}{P(D)} = \frac{p(p) \cdot 0.8}{\frac{1}{500}}$$

$$\boxed{\frac{1}{\cancel{500}} x = \left(\frac{499}{\cancel{500}}(1-x) + \frac{1}{\cancel{500}} x\right)\frac{4}{5}}$$

$$p(p) = p(p, H) + p(p, D)$$

$$= p(H)\,p(p \mid H) + p(D)\,p(p \mid D)$$

$$= \frac{499}{500}(1-x) + \frac{1}{500} x$$

$$p(p \mid H) + \cancel{p(p \mid D)}^{\,p(n \mid H)} = 1$$

$$\boxed{P(n \mid D) + p(p \mid D) = 1}$$

$\boxed{x = 0.9995\ldots}$

Part 5f

$$p(y=k \mid \vec{x}) \propto p(y=k)\,\underbrace{p(x_1, x_2 \mid y=k)}$$

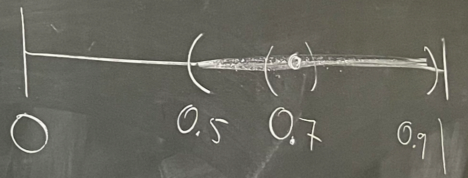
$$= p(x_1 \mid y=k)\; p(x_2 \mid x_1, y=k)$$

$$\frac{N_{x_1=v_1,\, x_2=v_2,\, y=k} + 1}{N_{x_1=v,\, y=k} + |f_2|}$$

$$\frac{N_{x_1=v_1,\, x_2=v_2,\, y=k} + 1}{N_{y=k} + |f_1| \cdot |f_2|}$$

6c classifier that always predicts female $\Rightarrow$ acc=75%

6d



6e-g

$$\bar{t} = \frac{1}{n}\sum_{i=1}^{n} x_i - y_i$$

(g) $\vec{x} = [15, 7, 10, 18, 11]$

$\vec{y} = [12, 4, 7, 15, 8]$

$$\begin{bmatrix} 12 & 7 & 10 & 18 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 15 & 4 & 7 & 15 & 11 \end{bmatrix}$$

$$-3 \quad 3 \quad 3 \quad 3 \quad -3$$

≈ 0