CS 260: Foundations of Data Science

Prof. Sara Mathieson

Spring 2025

HAVERFORD
COLLEGE

Admin

- Lab 5 released (partners optional), due after Spring Break
- Lab 3 grades will be up soon
- Today in class and lab: review and practice exam
- Thursday: continue review, focus on runtime

- **Midterm 1: next Tuesday in class**
  - Study sheet created by 1 (one page front and back)
  - No other notes or resources

Feedback forms: what do you know well

- Python
- Classes (OOP)
- Linear regression
- ROC curves
- Other evaluation metrics

Feedback forms: what needs review

- Cost function
- Stochastic gradient descent
- Bayes rule
- Probability topics

Feedback forms: other questions

- Class webpage with posted notes linked from Piazza
- Most said prefer to choose partner or work individually
- Individual appointments (can request!)
- Breaks during class (I will try!)
- Not planning to record or post notes before class (sorry!)

Why do we have a exam?

- Process of synthesizing the material on your own is essential
- Preparing the “study sheet” is designed to facilitate that process
- Review in class/lab this week (working through midterm practice exam)

Outline

- Review
  - Linear regression
  - Gradient descent
  - Matrix/vector form of Lab 3
  - Classification
  - Single feature models / decision trees
  - Evaluation metrics

Matrix/Vector form of SGD

while not converged:

  # shuffle the data (stochastic)

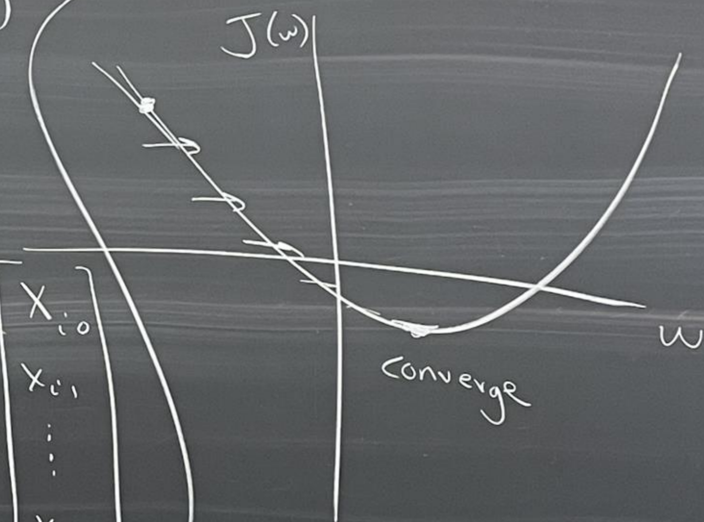
  4, 7, 2, 10 ... ← shuffle indices

  for $i = 1, 2, 3 \dots n$

$$\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix} \leftarrow \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix} - \underbrace{\alpha}_{\text{step size, learning rate}} \Big(\overbrace{h_{\vec{w}}(\vec{x}_i)}^{\text{pred}} - \overbrace{y_i}^{\text{truth}} \Big) \begin{bmatrix} x_{i0} \\ x_{i1} \\ \vdots \\ x_{ip} \end{bmatrix}$$

  test for convergence

$$|\text{new\_cost} - \text{old\_cost}| < \varepsilon$$



model $h_{\vec{w}}(\vec{x}) = \sum_{j=0}^{p} w_j x_j = \underbrace{w_0 \overbrace{x_0}^{1}}_{\text{intercept}} + w_1 x_1 \cdots w_p x_p = \vec{w} \cdot \vec{x}$

predict $(X, \vec{w})$

$$\underset{\text{matrix mult}}{X}\vec{w} = \hat{\vec{y}}$$

$$\begin{bmatrix} \\ x_{i0} \; x_{i1} \cdots x_{ip} \\ \\ \end{bmatrix}_{n \times (p+1)}^{T} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_p \end{bmatrix}_{(p+1)\times 1} = \begin{bmatrix} \vec{x}_i \cdot \vec{w} \end{bmatrix} = \begin{bmatrix} \hat{y}_i \end{bmatrix}$$

"any" regression problem

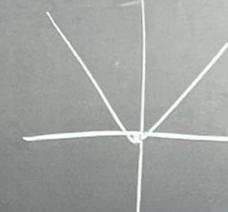
$$\text{cost}\,(X, y, \vec{w})$$

$$J(\vec{w}) = \frac{1}{2}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$= \frac{1}{2}(\vec{y} - \hat{\vec{y}}) \cdot (\vec{y} - \hat{\vec{y}})$$

$$= \boxed{\frac{1}{2}(\vec{y} - X\vec{w}) \cdot (y - X\vec{w})}$$

$$\cancel{J(\vec{w}) = \sum |y_i - \hat{y}_i|}$$



$\leq \varepsilon$

7

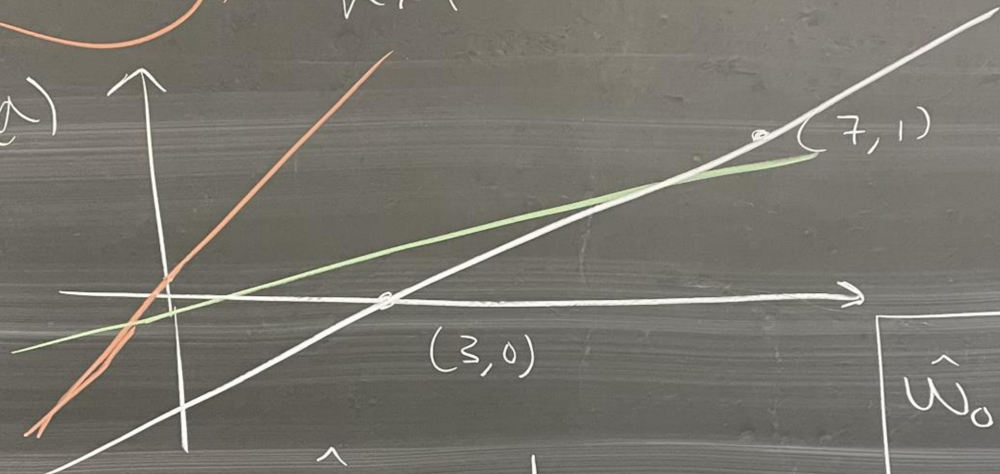
$$X = \begin{bmatrix} 1 & 3 \\ 1 & 7 \end{bmatrix}, \quad n \times (p+1)$$

column of 1's

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

$n = 2$
$p = 1$

(a)



$$\boxed{\begin{aligned} \hat{w}_0 &= -\frac{3}{4} \\ \hat{w}_1 &= \frac{1}{4} \end{aligned}}$$

$$\hat{w}_1 = \frac{1-0}{7-3} = \frac{1}{4}$$

$$y - 0 = \frac{1}{4}(x-3)$$

$$\boxed{y = -\frac{3}{4} + \frac{1}{4}x}$$

w

(b)

(c)

(b) $$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \leftarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.1\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 7 \end{bmatrix} - 1\right)\begin{bmatrix} 1 \\ 7 \end{bmatrix}$$

$$\leftarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} -0.1 \\ -0.7 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.7 \end{bmatrix}$$

(c) $$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \leftarrow \begin{bmatrix} 0.1 \\ 0.7 \end{bmatrix} - 0.1\left(\begin{bmatrix} 0.1 \\ 0.7 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 3 \end{bmatrix} - 0\right)\begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

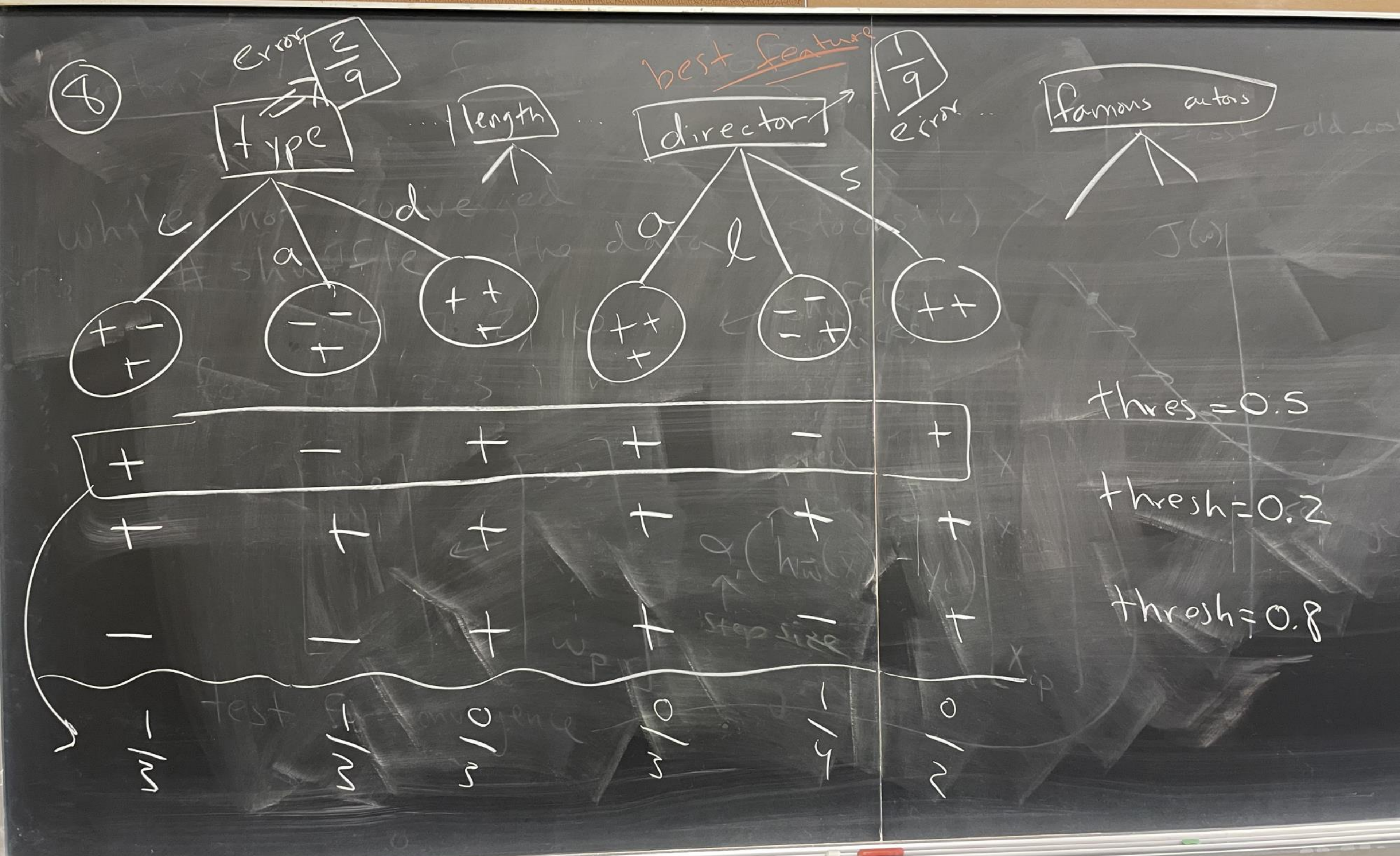
$$\leftarrow \begin{bmatrix} -0.12 \\ 0.04 \end{bmatrix}$$

ma

predi

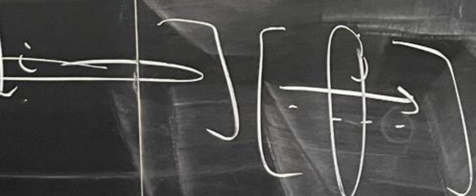
(

cost



8
error
2/9
type
c
a
d
length
best feature
director
a
l
s
1/9
error
famous actors
thres = 0.5
thresh = 0.2
thresh = 0.8
1/3
2/3
0/3
0/3
1/4
0/2

```
def matrix_mult(A, B):
    n, p = A.shape      check dims
    p, m = B.shape      inner match!
    result = np.zeros((n, m))
    for i in range(n):
        for j in range(m):
            val = 0
            for k in range(p):
                val += A[i, k]*B[k, j]      A[i] · B[j]
            result[i, j] = val
    return result.
```



Quote of the week

“Don't let the fear of the time it will take to accomplish something stand in the way of your doing it. The time will pass anyway; we might just as well put that passing time to the best possible use.” — Earl Nightingale