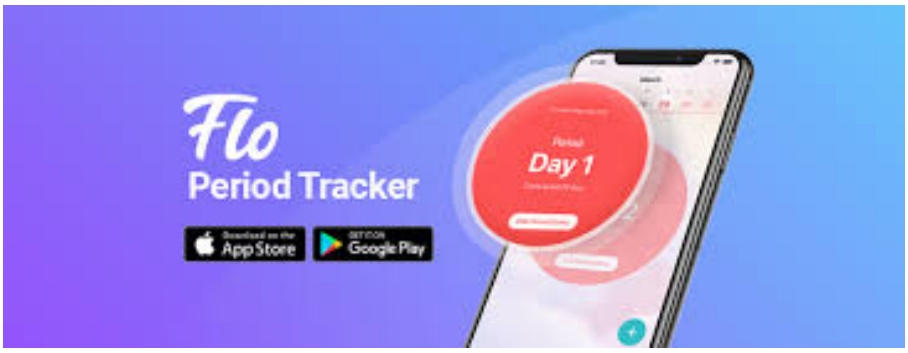
Chris Piech
CS 109

Section #7
November 15, 2025

Section 7: Machine Learning

1. **Flo. Tracking Menstrual Cycles**



Let X represent the length of a menstrual cycle: the number of days, as a continuous value, between the first moment of one period to the first moment of the next, for a given person. X is parameterized by α and β with probability density function:

$$f(X = x) = \beta \cdot (x - \alpha)^{\beta - 1} \cdot e^{-(x-\alpha)^{\beta}}$$

a. For a particular person, $\alpha = 27$ and $\beta = 2$. Write an expression for the probability that they have their period on day 29. In other words, what is $P(29.0 < X < 30.0)$? You do not need to simplify.

$$P(29.0 < X < 30.0) = \int_{29.0}^{30.0} 2 * (x - 27) * e^{-(x-27)^2} dx$$

b. For a particular person, $\alpha = 27$ and $\beta = 2$. How many times more likely is their cycle to last **exactly** 28.0 days than exactly 29.0 days? Simplify your expression.

This question is asking for the ratio between the probability density at $X = 28$ vs. the probability density as $X = 29$.

$$\frac{f(X = 28)}{f(X = 29)} = \frac{2 * (28 - 27) * e^{-(28-27)^2}}{2 * (29 - 27) * e^{-(29-27)^2}} = \frac{e^3}{2}$$

c. A person has recorded their cycle length for 12 cycles stored in a list: $m = [29.0, 28.5, \ldots, 30.1]$ where m_i is the recorded cycle length for cycle i. For this problem, assume that $\beta = 2$. Use MLE to estimate the parameter value for α. Assume that cycle lengths are IID.

You don't need a closed-form solution. Derive any necessary partial derivatives and write up to three sentences describing how a program can use the derivatives in order to choose the most likely parameter values. Here is the PDF again:

$$f(X = x) = \beta \cdot (x - \alpha)^{\beta - 1} \cdot e^{-(x-\alpha)^{\beta}}$$

As a way to test your knowledge, after section you can drop the assumption that $\beta = 2$ and use MLE to solve for both α and β.

Define our likelihood function:

$$L(\alpha, \beta) = \prod_{i=1}^{12} f(m_i)$$

We'll use log likelihood to make the math easier later:

$$LL(\alpha, \beta) = \sum_{i=1}^{12} \log f(m_i)$$

$$\alpha = \arg\max_{\alpha} LL(\alpha, \beta)$$

$$\beta = \arg\max_{\beta} LL(\alpha, \beta)$$

We can simplify the log of the PDF to:

$$\log f(m) = \log \beta + (\beta - 1) \log(m - \alpha) - (m - \alpha)^{\beta}$$

Now we take partial derivative w.r.t α and β:

$$\frac{\partial}{\partial \alpha} LL(\alpha, \beta) = \sum_{i=1}^{12} \beta (m_i - \alpha)^{\beta - 1} - \frac{\beta - 1}{m_i - \alpha}$$

$$\frac{\partial}{\partial \beta} LL(\alpha, \beta) = \sum_{i=1}^{12} \frac{1}{\beta} + \left(1 - (m_i - \alpha)^{\beta}\right) \log(m_i - \alpha)$$

We can use gradient ascent to maximize LL, using code like the snippet below. This code computes the gradient of the likelihood w.r.t. each parameter, then moves the parameters a small step in the direction of the gradient each iteration.

Note: Flo is a real "AI based" app that helps people track their period lengths. The real-world distribution is thought to be a mixture between a normal and a Weibull distribution; this problem only has you estimate parameters for a simplified Weibull.

```
num_iterations = 100000

learning_rate = 0.001  # step size

# Initialize parameters randomly
alpha = some random number
beta = some random number

for i in range(num\_iterations):
    gradient_alpha = 0
    gradient_beta = 0

    for m_i in m:  #  where m is a list of training datapoints
        gradient_alpha += 2 * (m_i - alpha) - ((beta - 1) / (m_i - alpha))
        gradient_beta += (1/beta) + log(m_i - alpha)

    alpha = alpha + (learning_rate * gradient_alpha)
    beta = beta + (learning_rate * gradient_beta)
```

2. **The Most Important Features** Let's explore saliency, a measure of how important a feature is for classification. We define the saliency of the ith input feature for a given example $(\mathbf{x}, y)$ to be the absolute value of the partial derivative of the log likelihood, with respect to that input feature $\left|\frac{\partial LL}{\partial x_i}\right|$. Below, we show both input images and the corresponding saliency of their features (in this case, pixels) for an image classification model:

Consider a trained logistic regression classifier with weights θ that predicts binary class labels, y. In this question, we allow the values of $\mathbf{x}$ to be real numbers, which doesn't change the algorithm at all (neither training nor testing).

a. (4 points) What is the Log Likelihood of a single training example $(\mathbf{x}, y)$ for this logistic regression classifier?

The log-likelihood here is the same as it was when we covered logistic regression in class: $LL(\theta) = y \cdot \log \sigma\Big(\theta^T \cdot \mathbf{x}\Big) + \Big(1 - y\Big)\log \Big[1 - \sigma\Big(\theta^T \cdot \mathbf{x}\Big)\Big]$

b. (8 points) Calculate the saliency of a single feature (x_i) for one training example $(\mathbf{x}, y)$.

We can calculate the saliency for a single feature as follows. Note that we are taking the derivative with respect to x_i, not θ_i, which is different from the usual derivative we find for MLE optimization.

$$LL(\theta) = y \log z + \Big(1 - y\Big) \log \Big(1 - z\Big) \qquad \text{where } z = \sigma\Big(\theta^T \cdot \mathbf{x}\Big)$$

$$\frac{\partial LL}{\partial x_i} = \frac{\partial LL}{\partial z} \cdot \frac{\partial z}{\partial x_i} \qquad \text{chain rule}$$

$$= \Big(\frac{y}{z} - \frac{1 - y}{1 - z}\Big) \cdot \Big(z(1 - z)\theta_i\Big) \qquad \text{partial derivatives}$$

$$\text{saliency} = \Big| \Big(\frac{y}{z} - \frac{1 - y}{1 - z}\Big) z(1 - z)\theta_i \Big|$$

c. (5 points) Show that the ratio of saliency for features i and j is the ratio of the absolute value of their weights $\frac{|\theta_i|}{|\theta_j|}$.

We can take the ratio as follows using our expression above.

$$\text{saliency for feature } i, S_i = \left| \left(\frac{y}{z} - \frac{1-y}{1-z} \right) z(1-z)\theta_i \right|, \text{ and same for } S_j$$

$$\frac{S_i}{S_j} = \frac{\left| \left(\frac{y}{z} - \frac{1-y}{1-z} \right) z(1-z)\theta_i \right|}{\left| \left(\frac{y}{z} - \frac{1-y}{1-z} \right) z(1-z)\theta_j \right|} = \frac{S_i}{S_j} = \frac{\left| \theta_i \right|}{\left| \theta_j \right|} \text{ by elimination}$$

3. **Warmup! (Optional)**

(a) True or False: The log likelihood function that is used to estimate p of a Bernoulli, for a set of observations, must always be 0 or smaller. Briefly explain why.

Yes. All probabilities are between 0 and 1, so all logs of probabilities are negative! For example, $\log(0.5) \approx -0.69$.

(b) When implementing logistic regression, a student decides to add a second intercept value. To do so they add an extra feature with value 0 to each datapoint. How will this impact training?

There will be no impact on training. When we compute $\theta^T x = \sum_{j=1}^{n} x_j \theta_j$, this new feature will have no contribution to the product, since no matter what theta value it is multiplied with, the result will still be zero.

(c) When implementing logistic regression, a student decides to incorporate an additional term that represents a non-linear relationship or "interaction" between the features. Their dataset has two features, x_1 and x_2, and a corresponding label y for each datapoint. They add the interaction term $x_1 \cdot x_2$, so that the full model is $P(Y = y|X = x) = \sigma(\theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot x_1 \cdot x_2)$. Explain briefly how this change impacts model training.

The training code would need to compute this feature (multiply x_1 by x_2) for each datapoint, so that the number of features and the length of the list of thetas increases by one. The gradient list is also one value longer, to include the gradient for the new theta. Inside the training for loop, the theta for this feature will be updated along with the others at each step.

4. **Vision Test MLE (Optional)**

> **Why is this question helpful?** In this problem you will be practicing all the mechanics and math behind logistic regression. The problem has all the same structural features (including a similar assumption), but it is not itself, logistic regression. Wahoo!

You decide that the vision tests given by eye doctors would be more precise if we used an approach inspired by logistic regression. In a vision test a user looks at a letter with a particular font size and either correctly guesses the letter or incorrectly guesses the letter.

You assume that the probability that a particular patient is able to guess a letter correctly is:

$$p = \sigma(\theta + f)$$

Where θ is the user's vision score and f is the font size of the letter. This formula uses the sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \qquad \frac{\partial \sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$$

Explain how you could estimate a user's vision score (θ) based on their 20 responses, $(f^{(1)}, y^{(1)})$ to $(f^{(20)}, y^{(20)})$, where $y^{(i)}$ is an indicator variable for whether the user correctly identified the ith letter and $f^{(i)}$ is the font size of the ith letter. Solve for all partial derivatives necessary.

We are going to solve this problem by finding the MLE estimate of θ. To find the MLE estimate, we are going to find the argmax of the log likelihood function. To calculate argmax we are going to use gradient ascent, which requires that we know the partial derivative of the log likelihood function with respect to theta.

First we write the log likelihood:

$$L(\theta) = \prod_{i=1}^{20} p^{y_i}(1-p)^{[1-y_i]}$$

$$LL(\theta) = \sum_{i=1}^{20} (y_i \log(p) + (1 - y_i) \log(1 - p))$$

Then we find the derivative of log likelihood with respect to θ. We first do this for one data point:

$$\frac{\partial LL}{\partial \theta} = \frac{\partial LL}{\partial p} \cdot \frac{\partial p}{\partial \theta}$$

We can calculate both the smaller partial derivatives independently:

$$\frac{\partial LL}{\partial p} = \frac{y_i}{p} - \frac{1 - y_i}{1 - p}$$

$$\frac{\partial p}{\partial \theta} = p[1 - p]$$

Putting it all together for one letter:

$$\begin{aligned}\frac{\partial LL}{\partial \theta} &= \frac{\partial LL}{\partial p} \cdot \frac{\partial p}{\partial \theta} \\ &= \Big[\frac{y_i}{p} - \frac{1 - y_i}{1 - p}\Big] p[1 - p] \\ &= y_i(1 - p) - p(1 - y_i) \\ &= y_i - p \\ &= y_i - \sigma(\theta + f)\end{aligned}$$

For all twenty examples:

$$\frac{\partial LL}{\partial \theta} = \sum_{i=1}^{20} y_i - \sigma(\theta + f^{(i)})$$