Linear Regression II

COSC 410: Applied Machine Learning
Fall 2025
Prof. Forrest Davis

September 11, 2025

Warm-up

1. Discuss with your neighbor the strangest food combination you've tried.
2. Given $g(x, z) = 2x^3 + 3z$:
   (a) What is the derivative of $g(x, z)$ with respect to x (i.e., $\frac{\partial g(x,z)}{\partial x}$)?
   (b) What is the derivative of $g(x, z)$ with respect to z (i.e., $\frac{\partial g(x,z)}{\partial z}$)?

Logistics

- Codelet 2 on Linear Regression Due Friday
- Office Hours today 3-6PM 331 Bernstein

Learning Objectives

- Apply stochastic gradient descent
- Reason about the choice of closed form and gradient based solutions to optimization
- Articulate the relationship between probability and model fit

Summary: We lay out the optimization problem for linear regression and propose two solutions: an analytical one and an algorithmic one. We then consider the choice between these and conclude with a deeper dive into the foundations of linear regression.

Refresh Aims

IN LINEAR REGRESSION, WE ARE SEEKING a set of parameters that best fits our data. Last class, we defined best fit with respect to **Mean-Squared Error** (MSE):

$$\text{MSE} := \frac{1}{m}\sum_{i=0}^{m-1}(y^{(i)} - \hat{y}^{(i)})^2 \tag{1}$$

The aim, then, of linear regression is to produce a line whose predictions are closest (on average) to the observed values. We find this line by minimizing our loss over our dataset:

$$\arg\min_{\mathbf{w}\in\mathbb{R}^{n+1}} \text{MSE}(\mathbf{X}, \mathbf{y}; \mathbf{w}) \tag{2}$$

We will consider two approaches to minimizing this loss: (1) the closed form (or analytic) solution and (2) the application of an iterative algorithm.

Optimization Approaches

Closed Form

UNLIKE MOST ML MODELS and optimization problems, linear regression has an analytic solution. This solution is where the gradient of our cost function is 0. In Figure 1, this is the bottom of the basin.

First, let's consider reformulating MSE over all of our samples.[1]

$$\arg\min_{\mathbf{w}\in\mathbb{R}^{n}} \text{MSE}(X, y; \mathbf{w}) = \frac{1}{m}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \tag{3}$$

Let's find the minimum (notice we reorder some terms to make it slightly easier).

$$\partial_{\mathbf{w}} \frac{1}{m}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 = 0$$

Incremental Algorithm: Gradient Descent Sketch

Not all functions are so easily solved analytically, as we will see later in the course. A more general approach to optimization is **gradient descent**.

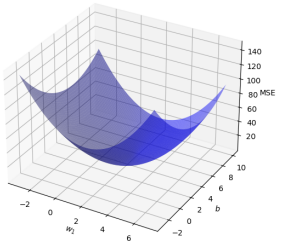

Figure 1: Mean-squared error with different parameters.

[1] Notice the use of the **vector norm**, $\|\mathbf{v}\|$, which is defined as:

$$\|\mathbf{v}\| = \sqrt{\sum_{i=1}^{n} v_i^2}$$

Squaring the norm gets us back to our MSE.

General Scheme

- Using a (twice) differentiable loss function
- Propose a initial guess for parameters
- While not converged:
  1. Determine the direction to move each parameter closer to a minima
  2. Increment each parameter by some amount in that direction
  3. If changes in the parameters or the loss are under some threshold, then converged

Gradient Descent: Dante's Other Inferno

IN ORDER TO GROUND GRADIENT DESCENT, we have to determine or set a few things:

1. How we determine direction to move
2. How much to change parameters
3. How much data to use in each update

Let's formulate the problem in a more concrete way to help address each choice. Say we have a dataset of 4 samples each with 3 features:

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} \\ x_1^{(3)} & x_2^{(3)} & x_3^{(3)} \\ x_1^{(4)} & x_2^{(4)} & x_3^{(4)} \end{bmatrix} \text{ and } \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \\ y^{(4)} \end{bmatrix}$$

We then have 4 parameters, the 3 from each feature and the 1 for the bias:

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \text{ and } \quad b$$

We have been building towards an elegant solution to both the direction and magnitude problems above (where to move and by how much). Namely, we can use the **partial derivative** of our loss function with respect to each parameter to determine both the direction

to move (the reverse of the sign of the gradient) and the magnitude (we take smaller gradients to mean we are close to the relevant minimum). Our update rule, then, looks like:[2]

$$w_i^{\text{next step}} = w_i - \eta \frac{\partial \text{Loss}(\mathbf{X}; \mathbf{w})}{\partial w_i} \tag{4}$$

[2] In Equation 4, w_i is the relevant parameter, $w_i^{\text{next step}}$ is the parameter updated from a step of gradient descent and η is the learning rate (a hyperparameter).

This is an iterative algorithm, where we update our weights in increments looping over our whole dataset some number of times (each loop over our entire dataset is called a **epoch**). In calculating the loss, in order to update the parameters, we consider a partition of our data. There are three broad categories:

1. **Stochastic Gradient Descent**: We update after each sample
2. **Batch Gradient Descent**: We update after calculating the loss over the entire dataset
3. **Mini-Batch Gradient Descent**: We update after calculating the loss over a subset of our data (between one sample and the entire dataset). Each subset is called a **mini-batch** or, confusingly, a **batch**.

Practice Problems

Given a dataset $\mathbf{X} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & -1 \\ 3 & 1 & 2 \\ -1 & -2 & 1 \end{bmatrix}$ with labels $\mathbf{y} = \begin{bmatrix} 2 \\ 1 \\ 0 \\ -1 \end{bmatrix}$ and initial parameters $\mathbf{w} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$ and $b = 1$:

1. Calculate the changes to the weights after one step of stochastic gradient descent with a learning rate of 0.5

2. What is the new $\hat{y}$ for the first sample after the first step of gradient descent?

Reflections on Optimization

WHY WOULDN'T WE ALWAYS USE the closed form solution for linear regression? Let's consider the computational complexity of the relevant linear algebra operations. For a full list check out this Wikipedia page.

There are two operations to focus on, matrix multiplication and the calculation of a matrix's inverse. Matrix multiplication applies for us for $\mathbf{X}^\mathsf{T}\mathbf{X}$ which is $O(n^2m)$. Calculating the inverse is $O(n^3)$. That means the closed solution is $O(n^2m + n^3)$, being dominated by the inverse, while a single step of gradient descent is $O(n^2m)$.[3]

[3] One step is $\mathbf{w} - \eta\frac{2}{m}(\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} - \mathbf{X}^\mathsf{T}\mathbf{y})$ which is $O(n^2m + n^2 + nm)$.

Question

When should we favor gradient descent?

Before Next Class

- Read and pre-class quiz
- Work on Codelet 2