# Optimization Techniques in Regression and Classification

Linear and logistic regression are foundational machine learning models that need effective methods to find optimal parameters. Let's look at the key techniques and why they matter.

## Linear Regression: Normal Equation vs. Gradient Descent

Both methods minimize Mean Squared Error (MSE) to find the best weights (θ), but they work very differently.

### The Normal Equation

This analytical method solves for θ directly in one calculation:

$\theta = (X^T X)^{-1} X^T y$

**Pros:** No hyperparameter tuning needed, gives exact solution, no feature scaling required.

**Cons:** Computationally expensive with $O(n^3)$ complexity, becomes impractically slow with many features (imagine inverting a 100,000 × 100,000 matrix), and fails if $(X^T X)$ isn't invertible.

### Gradient Descent

An iterative approach that starts with an initial guess and gradually improves it by stepping in the direction that reduces the cost function.

**Pros:** Scales excellently to large datasets, works for many optimization problems including neural networks.

**Cons:** Requires tuning the learning rate and iteration count, needs feature scaling for good convergence, only approximates the solution.

**Rule of thumb:** Use Normal Equation for datasets with fewer than 10,000 features. For larger datasets, Gradient Descent is your only practical option.

## Logistic Regression: The Softmax Function

In multi-class classification, your model produces raw scores (logits) for each class—unbounded numbers like [1.2, 2.9, -0.5]. But you need probabilities that are between 0 and 1 and sum to 1.

## How Softmax Works

The softmax function converts logits to probabilities:

$P(y = j \mid z) = e^{z_j} / \Sigma_k e^{z_k}$

It does this in two steps:

1. **Exponentiation** - Makes all values positive ($e^{z_j}$) and amplifies differences
2. **Normalization** - Divides by the sum so everything adds up to 1

Those example logits [1.2, 2.9, -0.5] become probabilities like [0.17, 0.84, 0.09]—now you can clearly see class 2 is most likely.

The name "softmax" comes from it being a smooth, differentiable version of "argmax" (which would just pick the maximum). This differentiability is crucial for training with Gradient Descent.

**Bottom line:** Softmax is the bridge between your model's raw outputs and interpretable probabilities, making both prediction and training possible.