Stochastic Gradient Descent for Logistic Regression

James Pustejovsky

CS115B
Brandeis University

January 28, 2025

Outline

Motivation

- Logistic Regression is a core method for classification, especially in NLP tasks.
- Why SGD?
  - Large datasets make full batch gradient descent expensive.
  - SGD updates parameters on a per-example or mini-batch basis, often converging faster in practice.
- We need to understand:
  - How logistic regression is formulated.
  - The SGD update rule.
  - When to stop (convergence criteria).

Model Setup

- Binary Classification Problem
  - Given input $\mathbf{x} \in \mathbb{R}^d$, predict $y \in \{0, 1\}$.
- Logistic Function

$$\hat{y} = \sigma(\mathbf{w}^\top \mathbf{x} + b) = \frac{1}{1 + e^{-(\mathbf{w}^\top \mathbf{x} + b)}}$$

- Interpretation: $\hat{y}$ is the probability that $y = 1$ given $\mathbf{x}$, under the model.

Loss Function: Cross-Entropy

- For a single training example $(\mathbf{x}, y)$, the loss is

$$\ell(\mathbf{w}; \mathbf{x}, y) = -\Big[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\Big].$$

- For a dataset of N examples, the empirical risk is

$$J(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}).$$

- Goal: Find $\mathbf{w}$ that minimizes $J(\mathbf{w})$.

Stochastic Gradient Descent

- Idea: Update weights using the gradient from just one (or a small batch of) training example(s) at a time.
- Generic SGD update for parameter $\mathbf{w}$:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \ell(\mathbf{w}; \mathbf{x}, y),$$

  where η is the learning rate.
- For logistic regression:

$$\nabla_{\mathbf{w}} \ell(\mathbf{w}; \mathbf{x}, y) = (\hat{y} - y)\mathbf{x}.$$

$$\implies \mathbf{w} \leftarrow \mathbf{w} - \eta\,(\hat{y} - y)\mathbf{x}.$$

Practical Stopping Criteria

SGD generally doesn't come to a hard stop. Instead, we decide we're "done" when:

- Small Parameter Updates:
  - If $\|\Delta \mathbf{w}\| < \epsilon$ for some threshold ϵ.
  - Once updates become vanishingly small, we consider the model converged.
- Loss Plateaus:
  - If the difference in training (or validation) loss

$$|J(\mathbf{w}^{(t)}) - J(\mathbf{w}^{(t-1)})| < \epsilon,$$

  - $\rightarrow$ We stop to avoid overfitting and wasted computation.
- Maximum Epochs:
  - Set a fixed number of passes (epochs) over the data.
  - If you've reached it, stop regardless.
- Validation Metrics (Early Stopping):
  - Monitor performance on a held-out set.
  - If accuracy, F1-score, or other metrics don't improve for k epochs, stop.

Convergence in Theory and Practice

- In theory, exact convergence happens when gradient $= 0$.
- In practice, floating-point precision and constant streaming of data mean we often rely on heuristics:
  - Tolerance thresholds on parameter updates.
  - Stabilizing loss or metrics.
- Rule of thumb:
  - If your model has not improved on the validation set for several epochs, you are likely near a minimum (or overfitting).
  - Once changes are negligible, it is "close enough" to call it converged.

Learning Rate and Step Size

- Learning rate (η) heavily affects convergence:
  - Too large: updates may overshoot minima and cause divergence.
  - Too small: convergence can be very slow.
- Learning rate scheduling can help:
  - Decreasing η over time (e.g., $\eta = \frac{\eta_0}{1+\alpha t}$).
  - Adaptive methods (e.g., Adam, RMSProp) for more stable convergence.

Regularization and Overfitting

- To avoid overfitting when doing many updates, logistic regression is often regularized:

$$J(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \ell(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}) + \lambda \|\mathbf{w}\|^2.$$

- Early stopping is also a regularizer:
  - Halt training when validation loss stops improving.

Conclusions

- SGD for Logistic Regression:
  - Efficient for large datasets and high-dimensional problems.
  - Each update uses just a small batch or a single sample.
- When Are We Done?
  - Parameter updates become very small.
  - Loss or metric plateau.
  - Validation set score stops improving.
- Practical Implementation:
  - Choose appropriate learning rate (schedule).
  - Employ regularization (L2, early stopping) to prevent overfitting.
  - Monitor training/validation curves to decide stopping.