

On Logistic Regression: Gradients of the Log Loss, Multi-Class Classification, and Other Optimization Techniques

Karl Stratos

June 20, 2018

Recall: Logistic Regression

- ▶ **Task.** Given input $x \in \mathbb{R}^d$, predict either 1 or 0 (on or off).

Recall: Logistic Regression

- ▶ **Task.** Given input $\mathbf{x} \in \mathbb{R}^d$, predict either **1** or **0** (**on** or **off**).
- ▶ **Model.** The probability of on is parameterized by $\mathbf{w} \in \mathbb{R}^d$ as a dot product squashed under the **sigmoid/logistic** function $\sigma : \mathbb{R} \rightarrow [0, 1]$.

$$p(1|\mathbf{x}, \mathbf{w}) := \sigma(\mathbf{w} \cdot \mathbf{x}) := \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{x})}$$



Recall: Logistic Regression

- ▶ **Task.** Given input $\mathbf{x} \in \mathbb{R}^d$, predict either **1** or **0** (**on** or **off**).
- ▶ **Model.** The probability of on is parameterized by $\mathbf{w} \in \mathbb{R}^d$ as a dot product squashed under the **sigmoid/logistic** function $\sigma : \mathbb{R} \rightarrow [0, 1]$.

$$p(1|\mathbf{x}, \mathbf{w}) := \sigma(\mathbf{w} \cdot \mathbf{x}) := \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{x})}$$



The probability of off is

$$p(0|\mathbf{x}, \mathbf{w}) = 1 - \sigma(\mathbf{w} \cdot \mathbf{x}) = \sigma(-\mathbf{w} \cdot \mathbf{x})$$

Recall: Logistic Regression

- ▶ **Task.** Given input $\mathbf{x} \in \mathbb{R}^d$, predict either **1** or **0** (**on** or **off**).
- ▶ **Model.** The probability of on is parameterized by $\mathbf{w} \in \mathbb{R}^d$ as a dot product squashed under the **sigmoid/logistic** function $\sigma : \mathbb{R} \rightarrow [0, 1]$.

$$p(1|\mathbf{x}, \mathbf{w}) := \sigma(\mathbf{w} \cdot \mathbf{x}) := \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{x})}$$



The probability of off is

$$p(0|\mathbf{x}, \mathbf{w}) = 1 - \sigma(\mathbf{w} \cdot \mathbf{x}) = \sigma(-\mathbf{w} \cdot \mathbf{x})$$

- ▶ Today's focus:
 1. **Optimizing the log loss by gradient descent**
 2. **Multi-class classification** to handle more than two classes
 3. **More on optimization:** Newton, stochastic gradient descent

Overview

Gradient Descent on the Log Loss

Multi-Class Classification

More on Optimization

- Newton's Method

- Stochastic Gradient Descent (SGD)

Negative Log Probability Under Logistic Function

- ▶ Using $y \in \{0, 1\}$ to denote the two classes,

$$\begin{aligned} -\log p(y|\mathbf{x}, \mathbf{w}) &= -y \log \sigma(\mathbf{w} \cdot \mathbf{x}) - (1 - y) \log \sigma(-\mathbf{w} \cdot \mathbf{x}) \\ &= \begin{cases} \log(1 + \exp(-\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 1 \\ \log(1 + \exp(\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 0 \end{cases} \end{aligned}$$

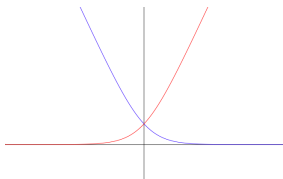
is convex in \mathbf{w} .

Negative Log Probability Under Logistic Function

- ▶ Using $y \in \{0, 1\}$ to denote the two classes,

$$\begin{aligned} -\log p(y|\mathbf{x}, \mathbf{w}) &= -y \log \sigma(\mathbf{w} \cdot \mathbf{x}) - (1 - y) \log \sigma(-\mathbf{w} \cdot \mathbf{x}) \\ &= \begin{cases} \log(1 + \exp(-\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 1 \\ \log(1 + \exp(\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 0 \end{cases} \end{aligned}$$

is convex in \mathbf{w} .

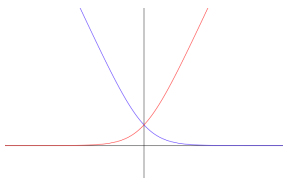


Negative Log Probability Under Logistic Function

- ▶ Using $y \in \{0, 1\}$ to denote the two classes,

$$\begin{aligned} -\log p(y|\mathbf{x}, \mathbf{w}) &= -y \log \sigma(\mathbf{w} \cdot \mathbf{x}) - (1 - y) \log \sigma(-\mathbf{w} \cdot \mathbf{x}) \\ &= \begin{cases} \log(1 + \exp(-\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 1 \\ \log(1 + \exp(\mathbf{w} \cdot \mathbf{x})) & \text{if } y = 0 \end{cases} \end{aligned}$$

is convex in \mathbf{w} .



- ▶ Gradient given by

$$\nabla_{\mathbf{w}} (-\log p(y|\mathbf{x}, \mathbf{w})) = -(y - \sigma(\mathbf{w} \cdot \mathbf{x}))\mathbf{x}$$

Logistic Regression Objective

- ▶ Given iid samples $S = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$, find \mathbf{w} that minimizes the empirical negative log likelihood of S (“log loss”):

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p\left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}\right)$$

- ▶ Gradient?

Logistic Regression Objective

- ▶ Given iid samples $S = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$, find \mathbf{w} that minimizes the empirical negative log likelihood of S (“log loss”):

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p\left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}\right)$$

- ▶ Gradient?

$$\nabla J_S^{\text{LOG}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)}) \right) \mathbf{x}^{(i)}$$

Logistic Regression Objective

- ▶ Given iid samples $S = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$, find \mathbf{w} that minimizes the empirical negative log likelihood of S (“log loss”):

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p\left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}\right)$$

- ▶ Gradient?

$$\nabla J_S^{\text{LOG}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)}) \right) \mathbf{x}^{(i)}$$

- ▶ Unlike in linear regression, there is no closed-form solution for

$$\mathbf{w}_S^{\text{LOG}} := \arg \min_{\mathbf{w} \in \mathbb{R}^d} J_S^{\text{LOG}}(\mathbf{w})$$

Logistic Regression Objective

- ▶ Given iid samples $S = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$, find \mathbf{w} that minimizes the empirical negative log likelihood of S (“log loss”):

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p\left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}\right)$$

- ▶ Gradient?

$$\nabla J_S^{\text{LOG}}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)})\right) \mathbf{x}^{(i)}$$

- ▶ Unlike in linear regression, there is no closed-form solution for

$$\mathbf{w}_S^{\text{LOG}} := \arg \min_{\mathbf{w} \in \mathbb{R}^d} J_S^{\text{LOG}}(\mathbf{w})$$

- ▶ But $J_S^{\text{LOG}}(\mathbf{w})$ is **convex** and **differentiable**! So we can do gradient descent and approach an optimal solution.

Gradient Descent for Logistic Regression

Input: training objective

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p \left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w} \right)$$

number of iterations T

Output: parameter $\hat{\mathbf{w}} \in \mathbb{R}^n$ such that $J_S^{\text{LOG}}(\hat{\mathbf{w}}) \approx J_S^{\text{LOG}}(\mathbf{w}_S^{\text{LOG}})$

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \dots T - 1$,

$$\theta^{t+1} = \theta^t + \frac{\eta^t}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}$$

3. Return θ^T .

Overview

Gradient Descent on the Log Loss

Multi-Class Classification

More on Optimization

- Newton's Method

- Stochastic Gradient Descent (SGD)

From Binary to m Classes

- ▶ A logistic regressor has a single $\mathbf{w} \in \mathbb{R}^d$ to define the probability of “on” (against “off”):

$$p(1|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w} \cdot \mathbf{x})}{1 + \exp(\mathbf{w} \cdot \mathbf{x})}$$

From Binary to m Classes

- ▶ A logistic regressor has a single $\mathbf{w} \in \mathbb{R}^d$ to define the probability of “on” (against “off”):

$$p(1|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w} \cdot \mathbf{x})}{1 + \exp(\mathbf{w} \cdot \mathbf{x})}$$

- ▶ A **log-linear model** has $\mathbf{w}^y \in \mathbb{R}^d$ for each possible class $y \in \{1 \dots m\}$ to define the probability of the class equaling y :

$$p(y|\mathbf{x}, \theta) = \frac{\exp(\mathbf{w}^y \cdot \mathbf{x})}{\sum_{y'=1}^m \exp(\mathbf{w}^{y'} \cdot \mathbf{x})}$$

where $\theta = \{\mathbf{w}^t\}_{t=1}^m$ denotes the set of parameters

From Binary to m Classes

- ▶ A logistic regressor has a single $\mathbf{w} \in \mathbb{R}^d$ to define the probability of “on” (against “off”):

$$p(1|\mathbf{x}, \mathbf{w}) = \frac{\exp(\mathbf{w} \cdot \mathbf{x})}{1 + \exp(\mathbf{w} \cdot \mathbf{x})}$$

- ▶ A **log-linear model** has $\mathbf{w}^y \in \mathbb{R}^d$ for each possible class $y \in \{1 \dots m\}$ to define the probability of the class equaling y :

$$p(y|\mathbf{x}, \theta) = \frac{\exp(\mathbf{w}^y \cdot \mathbf{x})}{\sum_{y'=1}^m \exp(\mathbf{w}^{y'} \cdot \mathbf{x})}$$

where $\theta = \{\mathbf{w}^t\}_{t=1}^m$ denotes the set of parameters

- ▶ Is this a valid probability distribution?

Softmax Formulation

- ▶ We can transform any vector $z \in \mathbb{R}^m$ into a probability distribution over m elements by the **softmax function** $\text{softmax} : \mathbb{R}^m \rightarrow \Delta^{m-1}$,

$$\text{softmax}_i(\mathbf{z}) := \frac{\exp(\mathbf{z}_i)}{\sum_{j=1}^m \exp(\mathbf{z}_j)} \quad \forall i = 1 \dots m$$

Softmax Formulation

- ▶ We can transform any vector $z \in \mathbb{R}^m$ into a probability distribution over m elements by the **softmax function** $\text{softmax} : \mathbb{R}^m \rightarrow \Delta^{m-1}$,

$$\text{softmax}_i(z) := \frac{\exp(z_i)}{\sum_{j=1}^m \exp(z_j)} \quad \forall i = 1 \dots m$$

- ▶ A log-linear model is a linear transformation by a matrix $W \in \mathbb{R}^{m \times d}$ (with rows $w^1 \dots w^m$) followed by softmax:

$$p(y|\mathbf{x}, W) = \text{softmax}_y(W\mathbf{x})$$

Negative Log Probability Under Log-Linear Model

- ▶ For any $y \in \{1 \dots m\}$,

$$-\log p(y|\mathbf{x}, \theta) = \underbrace{\log \left(\sum_{y'=1}^m \exp(\mathbf{w}^{y'} \cdot \mathbf{x}) \right)}_{\text{constant wrt. } y} - \underbrace{\mathbf{w}^y \cdot \mathbf{x}}_{\text{linear}}$$

is convex in each $\mathbf{w}^l \in \theta$.

Negative Log Probability Under Log-Linear Model

- ▶ For any $y \in \{1 \dots m\}$,

$$-\log p(y|\mathbf{x}, \theta) = \underbrace{\log \left(\sum_{y'=1}^m \exp(\mathbf{w}^{y'} \cdot \mathbf{x}) \right)}_{\text{constant wrt. } y} - \underbrace{\mathbf{w}^y \cdot \mathbf{x}}_{\text{linear}}$$

is convex in each $\mathbf{w}^l \in \theta$.

- ▶ For any $l \in \{1 \dots m\}$, the gradient wrt. \mathbf{w}^l is given by

$$\nabla_{\mathbf{w}^l} (-\log p(y|\mathbf{x}, \theta)) = \begin{cases} -(1 - p(l|\mathbf{x}, \theta))\mathbf{x} & \text{if } l = y \\ p(l|\mathbf{x}, \theta)\mathbf{x} & \text{if } l \neq y \end{cases}$$

Log-Linear Model Objective

- ▶ Given iid samples $\mathcal{S} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$, find \mathbf{w} that minimizes the empirical negative log likelihood of \mathcal{S}

$$J_{\mathcal{S}}^{\text{LLM}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p\left(y^{(i)} \mid \mathbf{x}^{(i)}, \theta\right)$$

This is the so-called **cross-entropy loss**.

- ▶ Again convex and differentiable, can be optimized by gradient descent to reach an optimal solution.

Overview

Gradient Descent on the Log Loss

Multi-Class Classification

More on Optimization

Newton's Method

Stochastic Gradient Descent (SGD)

One Way to Motivate Gradient Descent

- ▶ At current parameter value $\theta \in \mathbb{R}^d$, choose update $\Delta \in \mathbb{R}^d$ by minimizing the first-order Taylor approximation around θ with squared l_2 regularization:

$$J(\theta + \Delta) \approx \underbrace{J(\theta) + \nabla J(\theta)^\top \Delta + \frac{1}{2} \|\Delta\|_2^2}_{J_1(\Delta)}$$

$$\Delta^{\text{GD}} = \arg \min_{\Delta \in \mathbb{R}^d} J_1(\Delta) = -\nabla J(\theta)$$

One Way to Motivate Gradient Descent

- ▶ At current parameter value $\theta \in \mathbb{R}^d$, choose update $\Delta \in \mathbb{R}^d$ by minimizing the first-order Taylor approximation around θ with squared l_2 regularization:

$$J(\theta + \Delta) \approx \underbrace{J(\theta) + \nabla J(\theta)^\top \Delta + \frac{1}{2} \|\Delta\|_2^2}_{J_1(\Delta)}$$
$$\Delta^{\text{GD}} = \arg \min_{\Delta \in \mathbb{R}^d} J_1(\Delta) = -\nabla J(\theta)$$

- ▶ Equivalent to minimizing a second-order Taylor approximation but with **uniform curvature**

$$J(\theta) + \nabla J(\theta) \Delta + \frac{1}{2} \Delta^\top I_{d \times d} \Delta$$

Newton's Method

- ▶ Actually minimize the second-order Taylor approximation:

$$J(\theta + \Delta) \approx J(\theta) + \nabla J(\theta)^\top \Delta + \underbrace{\frac{1}{2} \Delta^\top \nabla^2 J(\theta) \Delta}_{J_2(\Delta)}$$

$$\Delta^{\text{NEWTON}} = \arg \min_{\Delta \in \mathbb{R}^d} J_2(\Delta) = -\nabla^2 J(\theta)^{-1} \nabla J(\theta)$$

Newton's Method

- ▶ Actually minimize the second-order Taylor approximation:

$$J(\theta + \Delta) \approx J(\theta) + \nabla J(\theta)^\top \Delta + \underbrace{\frac{1}{2} \Delta^\top \nabla^2 J(\theta) \Delta}_{J_2(\Delta)}$$

$$\Delta^{\text{NEWTON}} = \arg \min_{\Delta \in \mathbb{R}^d} J_2(\Delta) = -\nabla^2 J(\theta)^{-1} \nabla J(\theta)$$

- ▶ Equivalent to gradient descent after a change of coordinates by $\nabla^2 J(\theta)^{1/2}$

Overview

Gradient Descent on the Log Loss

Multi-Class Classification

More on Optimization

Newton's Method

Stochastic Gradient Descent (SGD)

Loss: a Function of All Samples

- ▶ Empirical loss $J_S(\theta)$ is a function of the **entire data** S .*

$$\underbrace{\frac{1}{2n} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right)^2}_{J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{-\frac{1}{n} \sum_{i=1}^n \log p \left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w} \right)}_{J_S^{\text{LOG}}(\mathbf{w})}$$

*We're normalizing by a constant for convenience without loss of generality.

Loss: a Function of All Samples

- ▶ Empirical loss $J_S(\theta)$ is a function of the **entire data** S .*

$$\underbrace{\frac{1}{2n} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right)^2}_{J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{-\frac{1}{n} \sum_{i=1}^n \log p \left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w} \right)}_{J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ So the gradient is also a function of the entire data.

$$\underbrace{-\frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\nabla J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}}_{\nabla J_S^{\text{LOG}}(\mathbf{w})}$$

*We're normalizing by a constant for convenience without loss of generality.

Loss: a Function of All Samples

- ▶ Empirical loss $J_S(\theta)$ is a function of the **entire data** S .*

$$\underbrace{\frac{1}{2n} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right)^2}_{J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{-\frac{1}{n} \sum_{i=1}^n \log p \left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w} \right)}_{J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ So the gradient is also a function of the entire data.

$$\underbrace{-\frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\nabla J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}}_{\nabla J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ Thus one update in gradient descent requires summing over all n samples.

*We're normalizing by a constant for convenience without loss of generality.

Gradient Estimation Based on a Single Sample

- ▶ What if we use a **single uniformly random** sample $i \in \{1 \dots n\}$ to estimate the gradient?

$$\underbrace{- \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\hat{\nabla}^{(i)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}}_{\hat{\nabla}^{(i)} J_S^{\text{LOG}}(\mathbf{w})}$$

Gradient Estimation Based on a Single Sample

- ▶ What if we use a **single uniformly random** sample $i \in \{1 \dots n\}$ to estimate the gradient?

$$\underbrace{- \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(i)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(i)} J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ In expectation, the gradient is consistent:

$$\mathbf{E}_i \left[\widehat{\nabla}^{(i)} J_S(\mathbf{w}) \right] = \frac{1}{n} \sum_{i=1}^n \widehat{\nabla}^{(i)} J_S(\mathbf{w}) = \nabla J_S(\mathbf{w})$$

Gradient Estimation Based on a Single Sample

- ▶ What if we use a **single uniformly random** sample $i \in \{1 \dots n\}$ to estimate the gradient?

$$\underbrace{- \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(i)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\left(y^{(i)} - \sigma \left(\mathbf{w} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(i)} J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ In expectation, the gradient is consistent:

$$\mathbf{E}_i \left[\widehat{\nabla}^{(i)} J_S(\mathbf{w}) \right] = \frac{1}{n} \sum_{i=1}^n \widehat{\nabla}^{(i)} J_S(\mathbf{w}) = \nabla J_S(\mathbf{w})$$

- ▶ This is **stochastic gradient descent (SGD)**: estimate the gradient with a single random sample. This is justified as long as the gradient is *consistent* in expectation.

SGD with Mini-Batches

- ▶ Instead of estimating the gradient based on a single random example i , use a random “mini-batch” $B \subseteq \{1 \dots n\}$.

$$\underbrace{-\frac{1}{|B|} \sum_{i \in B} (y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)}) \mathbf{x}^{(i)}}_{\hat{\nabla}^{(B)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\frac{1}{|B|} \sum_{i \in B} (y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)})) \mathbf{x}^{(i)}}_{\hat{\nabla}^{(B)} J_S^{\text{LOG}}(\mathbf{w})}$$

SGD with Mini-Batches

- ▶ Instead of estimating the gradient based on a single random example i , use a random “mini-batch” $B \subseteq \{1 \dots n\}$.

$$\underbrace{-\frac{1}{|B|} \sum_{i \in B} \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(B)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\frac{1}{|B|} \sum_{i \in B} \left(y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)}) \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(B)} J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ Still consistent: $\mathbf{E}_B \left[\widehat{\nabla}^{(B)} J_S(\mathbf{w}) \right] = \nabla J_S(\mathbf{w})$

SGD with Mini-Batches

- ▶ Instead of estimating the gradient based on a single random example i , use a random “mini-batch” $B \subseteq \{1 \dots n\}$.

$$\underbrace{-\frac{1}{|B|} \sum_{i \in B} \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(B)} J_S^{\text{LS}}(\mathbf{w})}$$

$$\underbrace{\frac{1}{|B|} \sum_{i \in B} \left(y^{(i)} - \sigma(\mathbf{w} \cdot \mathbf{x}^{(i)}) \right) \mathbf{x}^{(i)}}_{\widehat{\nabla}^{(B)} J_S^{\text{LOG}}(\mathbf{w})}$$

- ▶ Still consistent: $\mathbf{E}_B \left[\widehat{\nabla}^{(B)} J_S(\mathbf{w}) \right] = \nabla J_S(\mathbf{w})$
- ▶ Mini-batches allow for a more stable gradient estimation.
 - ▶ SGD is a special case with $|B| = 1$.

Stochastic Gradient Descent

Input: training objective $J(\theta) \in \mathbb{R}$ of form

$$J(\theta) = \frac{1}{n} \sum_{i=1}^n J_i(\theta)$$

, number of iterations T

Output: parameter $\hat{\theta} \in \mathbb{R}^n$ such that $J(\hat{\theta})$ is small

1. Initialize $\hat{\theta}$ (e.g., randomly).
2. For $t = 0 \dots T - 1$,
 - 2.1 For $i \in \{1 \dots n\}$ in **random order**,

$$\hat{\theta} \leftarrow \hat{\theta} - \eta^{t,i} \nabla J_i(\hat{\theta})$$

3. Return $\hat{\theta}$.

Stochastic Gradient Descent for Linear Regression

Input: training objective

$$J_S^{\text{LS}}(\mathbf{w}) := \frac{1}{2} \sum_{i=1}^n \left(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)} \right)^2$$

number of iterations T

Output: parameter $\hat{\mathbf{w}} \in \mathbb{R}^n$ such that $J_S^{\text{LS}}(\hat{\mathbf{w}}) \approx J_S^{\text{LS}}(\mathbf{w}_S^{\text{LS}})$

1. Initialize $\hat{\mathbf{w}}$ (e.g., randomly).
2. For $t = 0 \dots T - 1$,
 - 2.1 For $i \in \{1 \dots n\}$ in **random order**,

$$\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}} - \eta^{t,i} \left(y^{(i)} - \hat{\mathbf{w}}^\top \cdot \mathbf{x}^{(i)} \right) \mathbf{x}^{(i)}$$

3. Return $\hat{\mathbf{w}}$.

Stochastic Gradient Descent for Logistic Regression

Input: training objective

$$J_S^{\text{LOG}}(\mathbf{w}) := -\frac{1}{n} \sum_{i=1}^n \log p \left(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w} \right)$$

number of iterations T

Output: parameter $\hat{\mathbf{w}} \in \mathbb{R}^n$ such that $J_S^{\text{LOG}}(\hat{\mathbf{w}}) \approx J_S^{\text{LOG}}(\mathbf{w}_S^{\text{LOG}})$

1. Initialize $\hat{\mathbf{w}}$ (e.g., randomly).
2. For $t = 0 \dots T - 1$,
 - 2.1 For $i \in \{1 \dots n\}$ in **random order**,

$$\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}} + \eta^{t,i} \left(y^{(i)} - \sigma \left(\hat{\mathbf{w}} \cdot \mathbf{x}^{(i)} \right) \right) \mathbf{x}^{(i)}$$

3. Return $\hat{\mathbf{w}}$.

Summary

- ▶ **Logistic regression:** binary classifier that can be trained by optimizing the log loss
- ▶ **Log-linear model:** multi-class classifier that can be trained by optimizing the cross-entropy loss
- ▶ **Newton's method:** local search using the curvature of the loss function
- ▶ **SGD:** gradient descent with stochastic gradient estimation
 - ▶ Cornerstone of modern large-scale machine learning