Gradient Descent for Risk Optimization

Data 100: Principles and Techniques of Data Science

Sandrine Dudoit

Department of Statistics and Division of Biostatistics, UC Berkeley

Spring 2019

Outline

1. Motivation

2. Gradient Descent Optimization
   - 2.1 Optimization
   - 2.2 Batch Gradient Descent Algorithm
   - 2.3 Stochastic Gradient Descent Algorithm
   - 2.4 Convexity

3. Examples
   - 3.1 Squared Error Loss Function
   - 3.2 Huber Loss Function
   - 3.3 tips Dataset

Motivation

- Optimal statistical inference. A very broad class of statistical inference methods can be framed in terms of risk optimization.
- Least squares estimation (LSE) involves minimizing risk for the squared error loss function.
- Maximum likelihood estimation (MLE) involves minimizing risk for the negative log loss function.
- One can obtain closed-form expressions for risk minimizers for the squared error/L_2 and absolute error/L_1 loss functions: Means minimize mean squared error (MSE), while medians minimize mean absolute error (MAE).
- In general, however, there are no closed-form solutions for risk optimization, e.g., Huber loss function.

Motivation

- Instead, one can turn to numerical optimization methods such as gradient descent, simulated annealing, and genetic algorithms.

Optimization

- Suppose we wish to minimize the function $f : \mathbb{R}^J \to \mathbb{R}$, i.e., find
$$\text{argmin}_{\theta \in \mathbb{R}^J} f(\theta).$$
- The function f is referred to as objective function.
- In statistical inference, f typically corresponds to a risk function, i.e., the expected value of a loss function.
- The function f could be the empirical risk for the squared error loss function, i.e., the empirical mean squared error,
$$f(\theta) = R_2(P_n, \theta) = \frac{1}{n}\sum_{i=1}^{n} L_2(X_i, \theta) = \frac{1}{n}\sum_{i=1}^{n}(X_i - \theta)^2,$$
where $\theta \in \mathbb{R}$.

Optimization

- The function f could be the empirical risk for the Huber loss function

$$f(\theta) = R_H(P_n, \theta) = \frac{1}{n} \sum_{i=1}^{n} L_H(X_i, \theta),$$

where

$$L_H(X, \theta) = \begin{cases} \frac{1}{2}(X - \theta)^2, & |X - \theta| \leq \delta \\ \delta \left(|X - \theta| - \frac{1}{2}\delta \right), & \text{otherwise} \end{cases},$$

$\theta \in \mathbb{R}$, and $\delta \in \mathbb{R}^+$ is a tuning parameter.

Batch Gradient Descent Algorithm

- Gradient descent algorithms are iterative algorithms that seek to iteratively improve the solution to a particular optimization problem.
- That is, given a current estimate $\theta^{(t)}$, the algorithm aims to produce a next estimate $\theta^{(t+1)}$ such that $f(\theta^{(t)}) \geq f(\theta^{(t+1)})$.
- The intuition behind gradient descent algorithms is that the gradient (cf. slope) $\nabla_\theta f(\theta)$ suggests the direction in which to update θ.
    - If the gradient is negative, increase θ.
    - If the gradient is positive, decrease θ.
- Specifically, the batch gradient descent algorithm is as follows.

    1. Choose a starting value θ^0.

Batch Gradient Descent Algorithm

2. Update θ according to the following iteration

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta f(\theta^{(t)}), \tag{1}$$

   where α is a tuning parameter known as learning rate.
3. Repeat Step 2 until a stopping criterion is met.

- As with any iterative algorithm, important and practical decisions include the choice of starting value and stopping rule.
- A variety of approach may be use for selecting starting values.
  - Risk minimizer for tractable related loss function (e.g., mean).
  - Plotting, when possible, the objective function.
  - Domain knowledge.
  - Chosen at random.

Batch Gradient Descent Algorithm

  - Using multiple starting values is also advisable.
- Likewise, a variety of stopping rules can be used.
  - Stop after a fixed number of iterations.
  - Stop once θ doesn't change between iterations, i.e., $||\theta^{(t+1)} - \theta^{(t)}|| \leq \epsilon$ or $|\theta_j^{(t+1)} - \theta_j^{(t)}| \leq \epsilon_1(|\theta_j^{(t)}| + \epsilon_2)$ when elements of θ are of different magnitudes.
  - Stop once the objective function doesn't change between iterations, i.e., $|f(\theta^{(t+1)}) - f(\theta^{(t)})| \leq \epsilon$.
- The higher the learning rate α, the more "aggressive" the moves, at the risk of overshooting the minimum. The smaller the learning rate, the more precise the moves, but the more time-consuming the implementation. In some versions of the algorithm, the step size changes at each iteration.

Gradient Descent

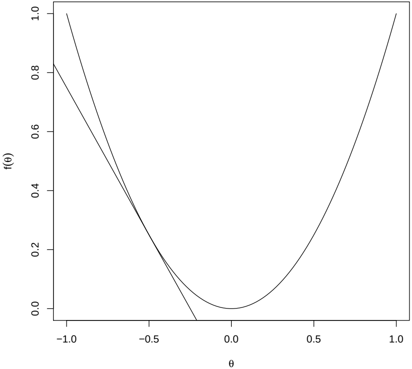

Figure 1: *Gradient descent.* The slope of the tangent line determines in which direction to update θ in order to decrease the objective function.

Stochastic Gradient Descent Algorithm

- With the above gradient descent algorithm, the gradient is computed for empirical risk based on the entire learning set

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta L(X_i, \theta^{(t)}). \tag{2}$$

- Such an approach, known as batch gradient descent, can be computationally inefficient for large datasets.
- An alternative, known as stochastic gradient descent (SGD), is to compute the gradient for a randomly chosen observation X_i, that is, have the updates

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta L(X_i, \theta^{(t)}). \tag{3}$$

Stochastic Gradient Descent Algorithm

- Stochastic gradient descent often takes steps away from the optimum, but makes more aggressive updates and often converges faster than batch gradient descent.
- Mini-batch gradient descent strikes a balance between batch gradient descent and stochastic gradient descent by using a random sample of several observations for each update.

Convexity

- Not all functions are equally easy to optimize.
- The empirical MSE has a unique global minimizer, the mean.

$$\bar{X}_n = \text{argmin}_{\theta \in \mathbb{R}} R_2(P_n, \theta) = \text{argmin}_{\theta \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} (X_i - \theta)^2.$$

- The empirical MAE could have multiple minima, the median, but these are global minima.

$$\tilde{X}_n = \text{argmin}_{\theta \in \mathbb{R}} R_1(P_n, \theta) = \text{argmin}_{\theta \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} |X_i - \theta|.$$

- Although there is no closed-form expression for the Huber risk minimizer, it is unique.

Convexity

- The above loss functions are convex functions of the parameter θ.
- A function f is convex if and only if it satisfies the following inequality

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y), \qquad \alpha \in [0,1]. \tag{4}$$

  That is, the line segment between any two points on the graph of the function lies above or on the graph.
- The function is concave if

$$f(\alpha x + (1-\alpha)y) \geq \alpha f(x) + (1-\alpha) f(y), \qquad \alpha \in [0,1].$$

Convexity

- For a twice differentiable function of a single variable, if the second derivative is greater than or equal to zero for its entire domain, then the function is convex.
- E.g. The quadratic function $f(x) = x^2$ and the exponential function $f(x) = \exp(x)$ are convex. The logarithm function $f(x) = \log(x)$ is concave.
- For convex functions, any local minimum is also a global minimum.
- Convexity of a loss function allows gradient descent to efficiently find the global risk minimizer.
- While gradient descent will converge to a local minimum for non-convex loss functions, these local minima are not guaranteed to be globally optimal.

Convexity

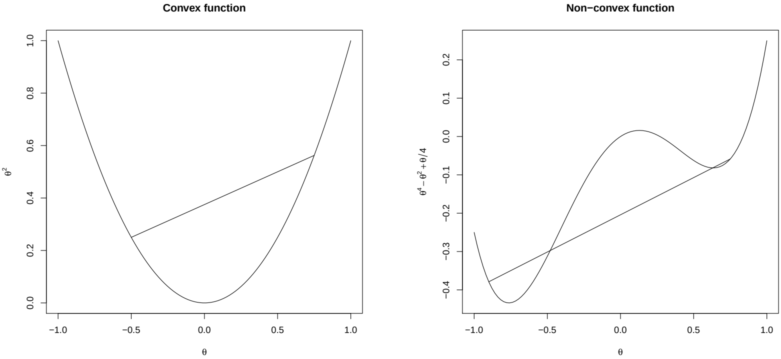

Figure 2: *Convexity.*

Squared Error Loss Function

- For the squared error loss function, the gradient (derivative) is

$$\nabla_\theta L_2(X,\theta) = \nabla_\theta (X-\theta)^2 = -2(X-\theta).$$

- The gradient descent iteration is

$$\theta^{(t+1)} = \theta^{(t)} + 2\alpha \frac{1}{n}\sum_{i=1}^{n}(X_i - \theta^{(t)}) = \theta^{(t)} + 2\alpha(\bar{X}_n - \theta^{(t)}).$$

- The empirical mean is, as expected, a fixed point of the algorithm, i.e., if $\theta^{(t)} = \bar{X}_n$, then $\theta^{(t+1)} = \bar{X}_n$.

Huber Loss Function

- For the Huber loss function, the gradient (derivative) is

$$
\begin{aligned}
\nabla_\theta L_H(X,\theta) &= \begin{cases} \nabla_\theta \frac{1}{2}(X-\theta)^2, & |X-\theta| \leq \delta \\ \nabla_\theta \delta\left(|X-\theta| - \frac{1}{2}\delta\right), & \text{otherwise} \end{cases} \\
&= \begin{cases} -(X-\theta), & |X-\theta| \leq \delta \\ -\delta\,\mathrm{sign}(X-\theta), & \text{otherwise} \end{cases}.
\end{aligned}
$$

Huber Loss Function

- The gradient descent iteration is

$$\begin{aligned}\theta^{(t+1)} &= \theta^{(t)} + \alpha \frac{1}{n} \sum_{i=1}^{n} (X_i - \theta^{(t)}) \mathrm{I}(|X_i - \theta^{(t)}| \leq \delta) \\ &\quad + \alpha\delta \frac{1}{n} \sum_{i=1}^{n} \mathrm{I}(X_i - \theta^{(t)} \geq \delta) \\ &\quad - \alpha\delta \frac{1}{n} \sum_{i=1}^{n} \mathrm{I}(X_i - \theta^{(t)} \leq -\delta).\end{aligned}$$

tips Dataset

- A particular waiter is interested in inferring the tip percentage he could expect. He collected the following data on all $n = 244$ tables he served during a month of employment: Total bill, tip, sex of customer tipping, smoking status of customer, day, time, and size of party.

	total_bill	tip	sex	smoker	day	time	size	tip_percent
1	16.99	1.01	Female	No	Sun	Dinner	2	0.06
2	10.34	1.66	Male	No	Sun	Dinner	3	0.16
3	21.01	3.50	Male	No	Sun	Dinner	3	0.17
4	23.68	3.31	Male	No	Sun	Dinner	2	0.14
5	24.59	3.61	Female	No	Sun	Dinner	4	0.15
6	25.29	4.71	Male	No	Sun	Dinner	4	0.19
				. . .				

tips Dataset

- In the lecture “Foundations of Statistical Inference” we performed exploratory data analysis (EDA) on this dataset and decided to fit a constant model for the tip percentage Y
$$\mathrm{E}[Y] = \theta.$$
- We considered three different loss functions to select an “optimal” estimator of θ:
    - the squared error loss function, for which the optimal estimator is the empirical mean,
    - the absolute error loss function, for which the optimal estimator is the empirical median,
    - the Huber loss function, for which there is no closed-form expression for the empirical risk minimizer.
- Here, we use gradient descent algorithms to optimize the empirical Huber risk.

tips Dataset

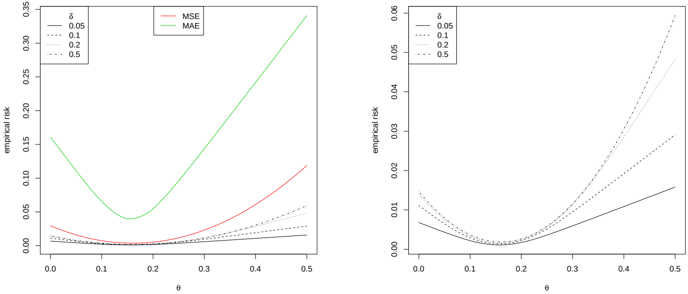

Figure 3: *tips dataset.* Empirical Huber risk, MSE, and MAE as a function of mean tip percentage θ. Right panel is zoom on Huber risk.

tips Dataset

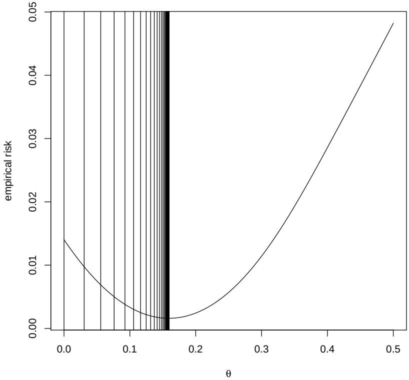

Figure 4: *tips dataset.* Batch gradient descent ($\alpha = 0.2$, $\epsilon = 10^{-6}$ for θ) for optimizing empirical Huber risk ($\delta = 0.2$),

tips Dataset

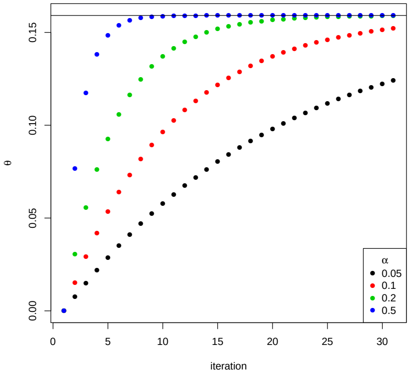

Figure 5: *tips dataset.* First 30 iterations of batch gradient descent, with different learning rates α, for optimizing empirical Huber risk ($\delta = 0.2$).

tips Dataset

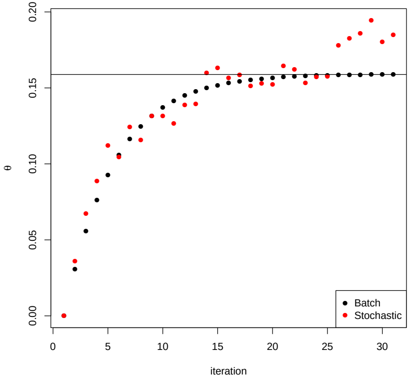

Figure 6: *tips dataset.* First 30 iterations of batch and stochastic gradient descent ($\alpha = 0.2$) for optimizing empirical Huber risk ($\delta = 0.2$).