Gradient Descent

Johanni Brea

Introduction à l'apprentissage automatique

GYMINF 2021

Table of Contents

Gradient Descent

1. Input: loss function L, initial guess $\beta^{(0)} = \left(\beta_0^{(0)}, \ldots, \beta_p^{(0)}\right)$ learning rate η, maximal number of steps T.
2. For $t = 1, \ldots, T$
   - $\delta_i = \eta \dfrac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}\right)$
   - $\beta_i^{(t)} = \beta_i^{(t-1)} - \delta_i$
3. Return $\beta^{(T)}$

Gradient Descent

1. Input: loss function L, initial guess $\beta^{(0)} = \left(\beta_0^{(0)}, \ldots, \beta_p^{(0)}\right)$ learning rate η, maximal number of steps T.
2. For $t = 1, \ldots, T$
   - $\delta_i = \eta \frac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}\right)$
   - $\beta_i^{(t)} = \beta_i^{(t-1)} - \delta_i$
3. Return $\beta^{(T)}$

Automatic Differentiation software uses the chain rule and symbolic derivatives for primitive functions, to compute the derivative of almost any code we write.

Table of Contents

Stochastic Gradient Descent (SGD)

Computing the loss over all samples $1, \dots, n$ can be computationally costly.
A subset of the training data may be sufficient to estimate the gradient direction.

Stochastic Gradient Descent (SGD)

Computing the loss over all samples $1, \ldots, n$ can be computationally costly.
A subset of the training data may be sufficient to estimate the gradient direction.

1. Input: loss function L, initial guess $\beta^{(0)} = \left(\beta_0^{(0)}, \ldots, \beta_p^{(0)}\right)$ learning rate η, maximal number of steps T, tolerance Δ, batch size B.
2. For $t = 1, \ldots, T$
   - Determine batch of training indices $\mathcal{I}$
   - $\delta_i = \eta \dfrac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}; \mathcal{I}\right)$
   - $\beta_i^{(t)} = \beta_i^{(t-1)} - \delta_i$
3. Return $\beta^{(T)}$

where $L(\beta; \mathcal{I})$ is the loss function evaluated on the training samples with indices in $\mathcal{I}$, e.g.

$$L\left(\beta; \mathcal{I}\right) = \frac{1}{B} \sum_{i \in \mathcal{I}} \left(y_i - x_i^T \beta\right)^2$$

Stochastic Gradient Descent (SGD)

Computing the loss over all samples $1, \ldots, n$ can be computationally costly.
A subset of the training data may be sufficient to estimate the gradient direction.

1. Input: loss function L, initial guess $\beta^{(0)} = \left(\beta_0^{(0)}, \ldots, \beta_p^{(0)}\right)$ learning rate η, maximal number of steps T, tolerance Δ, batch size B.
2. For $t = 1, \ldots, T$
   - Determine batch of training indices $\mathcal{I}$
   - $\delta_i = \eta \frac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}; \mathcal{I}\right)$
   - $\beta_i^{(t)} = \beta_i^{(t-1)} - \delta_i$
3. Return $\beta^{(T)}$

where $L(\beta; \mathcal{I})$ is the loss function evaluated on the training samples with indices in $\mathcal{I}$, e.g.

$$L\left(\beta; \mathcal{I}\right) = \frac{1}{B} \sum_{i \in \mathcal{I}} \left(y_i - x_i^T \beta\right)^2$$

Example $B = 6$

	batch 1	batch 2	batch 3	ba
$\mathcal{I}$	1 8 3 13 93	9 14 2 26 31	$\cdots$	

Table of Contents

Momentum

- $\delta_i = \eta \frac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}\right)$
- $v_i^{(t)} = \mu v_i^{(t-1)} + (1-\mu)\delta_i$
- $\beta_i^{(t)} = \beta_i^{(t-1)} - v_i^{(t)}$

Adaptive Learning Rates and Momentum

Momentum

- $\delta_i = \eta \frac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}\right)$
- $v_i^{(t)} = \mu v_i^{(t-1)} + (1-\mu)\delta_i$
- $\beta_i^{(t)} = \beta_i^{(t-1)} - v_i^{(t)}$

Adaptive Learning Rates

For every parameter a different learning rate η_i can be chosen. It can also change over time.

`https://doi.org/10.1371/journal.pcbi.1007640`

Momentum

- $\delta_i = \eta \frac{\partial L}{\partial \beta_i}\left(\beta^{(t-1)}\right)$
- $v_i^{(t)} = \mu v_i^{(t-1)} + (1-\mu)\delta_i$
- $\beta_i^{(t)} = \beta_i^{(t-1)} - v_i^{(t)}$

Adaptive Learning Rates

For every parameter a different learning rate η_i can be chosen. It can also change over time.

`https://doi.org/10.1371/journal.pcbi.1007640`

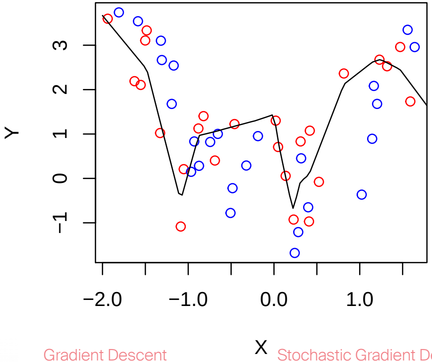
Modern methods like ADAM(W) include momentum and automatically adapting learning rates for the different parameters.

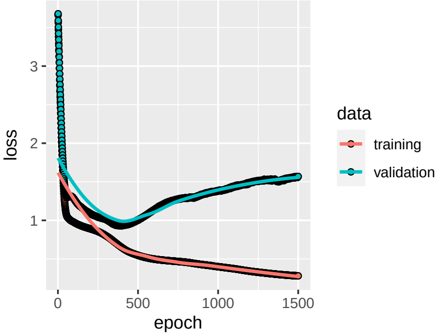
Table of Contents

Start with small weights and stop gradient descent when validation loss starts to increase.

training data validation data

black line: a flexible neural network trained with gradient descent





Quiz

- If we choose a small learning rate larger than zero, the training loss in gradient descent is decreasing in every step.

Quiz

- If we choose a small learning rate larger than zero, the training loss in gradient descent is decreasing in every step.
- If we choose a small learning rate larger than zero, the training loss in stochastic gradient descent is decreasing in every step.

Quiz

- If we choose a small learning rate larger than zero, the training loss in gradient descent is decreasing in every step.
- If we choose a small learning rate larger than zero, the training loss in stochastic gradient descent is decreasing in every step.
- Stochastic gradient descent requires less computation than full gradient descent for each update of the parameters.