Machine Learning

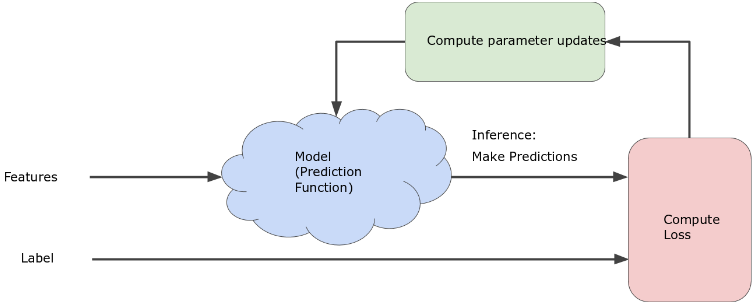
Pawel Wocjan

University of Central Florida

Fall 2020

An Iterative Approach

- The figure below depicts the iterative trial-and-error process that machine learning algorithms use to train a model:



Gradient Descent

- The iterative approach diagram contains a green box "compute parameter updates."
- Let us now discuss the gradient descent algorithm that is used to update the parameters.

Gradient Descent

- We consider the linear regression model

$$\hat{y} = f_w(x) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

where $w = (w_0, w_1, \ldots, w_n)$.

Gradient Descent

- The loss function (mean squared error)

$$\mathcal{L} : \mathbb{R}^{n+1} \to \mathbb{R}$$

  depends on the parameters $n+1$ parameters $b, w_1, \ldots, w_n$.

- The loss is given by

$$\begin{aligned}
\mathcal{L} &= \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2 \\
&= \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} (f_w(x^{(i)} - y^{(i)})^2 \\
&= \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(\sum_{j=1}^{n} w_j x_j + b - y^{(i)} \right)^2
\end{aligned}$$

Gradient Decent

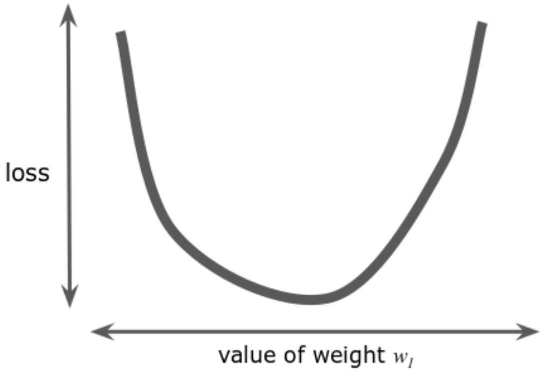
- For $n = 1$, the loss function $\mathcal{L}$ depends on two parameters – the bias term $b = w_0$ and the weight w_1 – and defines a surface in 3D.
- For $n > 1$, the loss function $\mathcal{L}$ cannot be visualized so easily.

Gradient Decent

- To simplify the plots, we assume that $n = 1$ and the bias term $b = w_0$ is fixed to be 0.
- Then, the loss function $\mathcal{L}$ depends only on w_1 and defines a curve.

Gradient Descent

- The resulting plot of the loss function $\mathcal{L}$ is be convex.
- Simply speaking, it means that it is bowl-shaped like this:

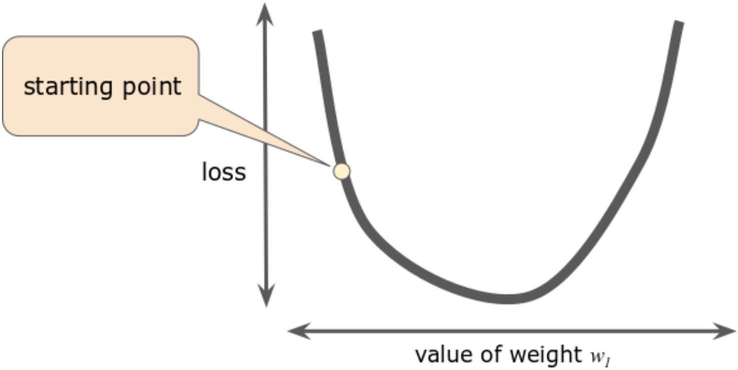


Gradient Descent

- It turns out that even in the general case the loss function $\mathcal{L}$ is convex.
- This is important because problems have only one minimum.
- Calculating the loss function for all parameter values $w_0, \ldots, w_n \in \mathbb{R}^{n+1}$ would be an inefficient way of finding the minimum.
- Let's examine a better mechanism – very popular in machine learning – called **gradient descent**.

Gradient Descent

- The first stage in gradient descent is to pick a starting value.
- The starting point doesn't matter much; therefore, many algorithms simply set $w_i = 0$ or set the w_i to random values.



Gradient Descent

- The gradient descent algorithm then calculates the gradient of the loss function $\mathcal{L}$ at the starting point.
- The gradient

$$\nabla \mathcal{L} \in \mathbb{R}^{n+1}$$

is a vector whose entries

$$(\nabla \mathcal{L})_i$$

are given by the partial derivatives

$$\partial \mathcal{L} / \partial w_i$$

of the loss function $\mathcal{L}$ with respect to the weights w_i.

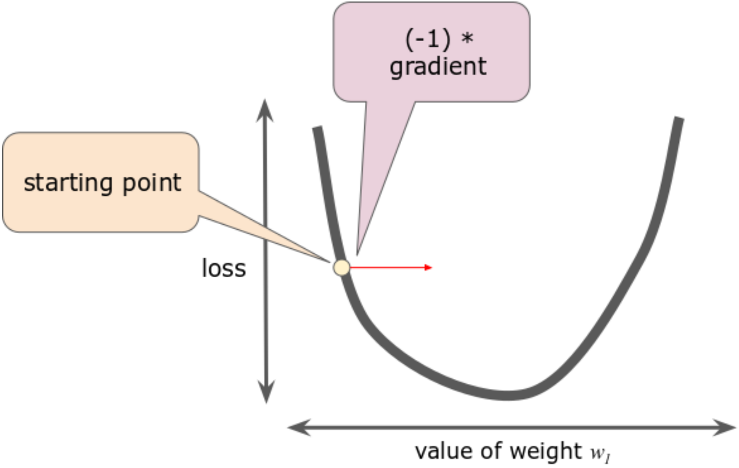
Gradient Descent

- The $\nabla\mathcal{L}$ gradient has both a direction and a magnitude.
- The gradient points which way is "warmer" or "colder."
- The gradient always points in the direction of steepest increase in the loss function.

  For the case $n = 1$ and the bias $w_0 = b$ is fixed to be 0, the gradient of the loss function $\mathcal{L}$ is simply the slope of the curve $\mathcal{L}$, that is, the derivative with respect to w_1.

Gradient Descent

- The gradient descent algorithm takes a step in the direction of the negative gradient $-\nabla\mathcal{L}$ to reduce the loss.

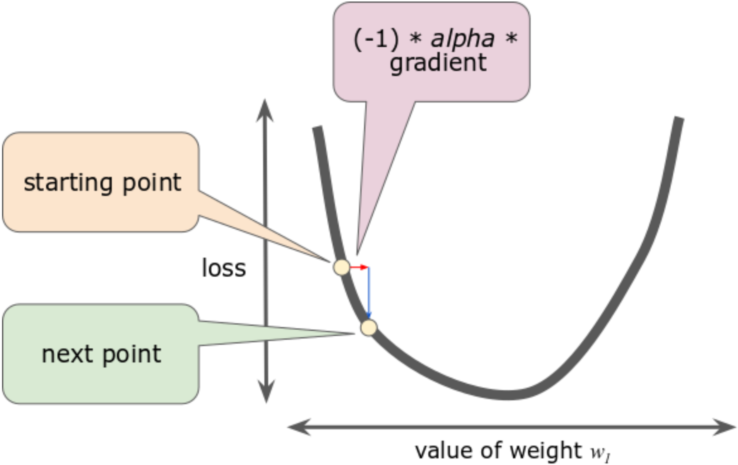


Gradient Descent

- More precisely, the gradient descent algorithm updates the starting point as follows:

$$w \leftarrow w - \alpha \nabla \mathcal{L}$$

where α is the learning rate.



Key Terms

- gradient
- gradient descent
- step

Learning Rate

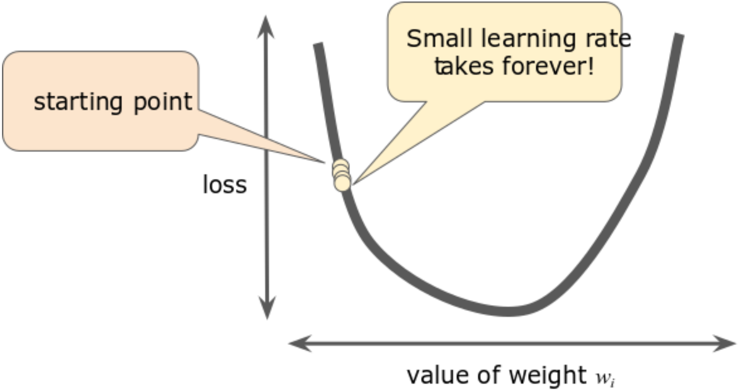
- The gradient vector has both a direction and a magnitude.
- The gradient descent algorithm multiplies the gradient by a scalar known as the learning rate (also sometimes called step size) to determine the next point.

Learning Rate

- The learning rate is a so-called **hyperparameter**.
- A hyperparameter is a parameter that is external to the model.

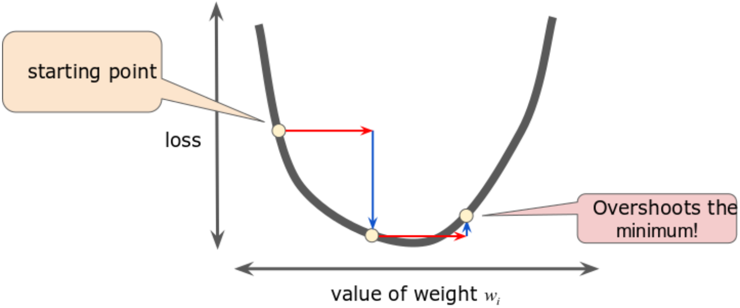
Learning Rate

- If the learning rate that is too small, learning will take too long:



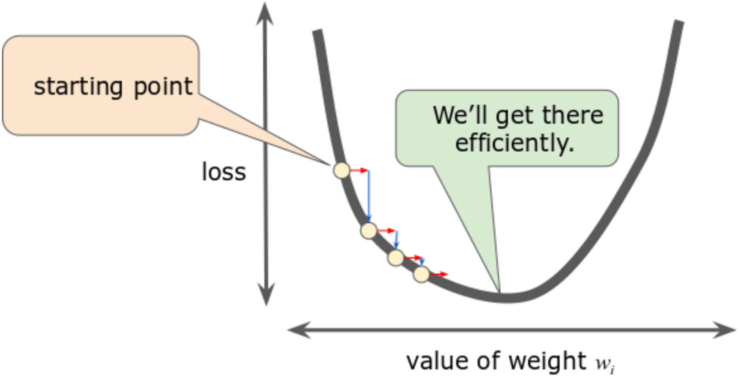
Learning Rate

- If the learning rate is too large, the next point will perpetually bounce haphazardly across the bottom of the well:



Learning Rate

- There's a Goldilocks learning rate for every linear regression problem.



Key Terms

- hyperparameter
- learning rate
- step size