Optimization

Friday, 30 April 2021 07:42

Gradient Descent

- In general used to find a minimum of any function
- In ML mostly used to find a minimum of the error function (typically MSE) dependent on model parameters

General formulation: $\min_{\vec{x} = (x_1, \dots, x_n)} f(x_1, \dots, x_n)$

ML formulation (special case of the above): $\min_{\vec{\theta} = (\theta_1, \dots, \theta_n)} \sum_{(\vec{x}, y) \in D} (y - f(\vec{x} \mid \vec{\theta}))^2$

training data

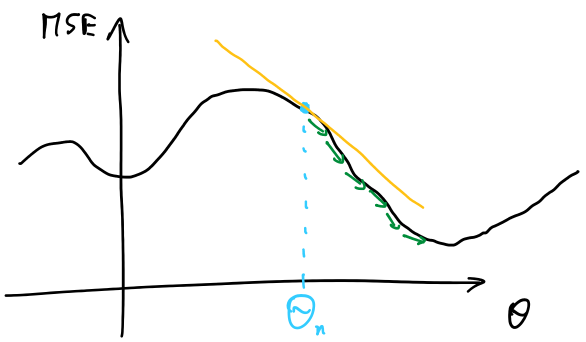
Example

$$\min_{(\theta_0, \theta_1)} \sum_{(x,y) \in D} (y - (\theta_1 x + \theta_0))^2$$

$$f(x \mid \theta) = \theta_1 x + \theta_0$$

Idea of GD

1. Start with any $\vec{\theta}$
2. Iteratively move $\vec{\theta}$ in the direction opposite to the derivative



The slope of the tangent line is equal to the derivative of the function with respect to θ

$$\frac{\partial MSE}{\partial \theta}$$

Derivative calculation for MSE

For simplicity we assume a linear model

$$f(x \mid \theta_0, \theta_1) = f(x, \theta_0, \theta_1) = \theta_1 x + \theta_0$$

$$MSE = \frac{1}{|D|} \sum_{(x,y) \in D} (y - (\theta_1 x + \theta_0))^2$$

Since $(f+g)'(x) = f'(x) + g'(x)$ and $(a f(x))' = a f'(x)$ we can make the derivative calculations for a single element in the sum (single datapoint) and then average.

$$\frac{\partial}{\partial \theta_0} (y - (\theta_1 x + \theta_0))^2 = 2 (y - (\theta_1 x + \theta_0)) \cdot \frac{\partial}{\partial \theta_0} (y - (\theta_1 x + \theta_0))$$

$$= 2 (y - (\theta_1 x + \theta_0)) (-1)$$

$$= -2 (y - (\theta_1 x + \theta_0))$$

$$\frac{\partial}{\partial \theta_1} (y - (\theta_1 x + \theta_0))^2 = 2 (y - (\theta_1 x + \theta_0)) \cdot \frac{\partial}{\partial \theta_1} (y - (\theta_1 x + \theta_0))$$

$$= 2 (y - (\theta_1 x + \theta_0)) (-x)$$

$$= -2x (y - (\theta_1 x + \theta_0))$$

Updating rule

$$\boxed{\vec{\theta}^{\,n} = \vec{\theta}^{\,n-1} - \alpha \frac{\partial}{\partial \theta} MSE(\vec{\theta})}$$

$$\theta_0^n = \theta_0^{n-1} - \alpha \frac{\partial}{\partial \theta_0} MSE(\theta_0^{n-1}, \theta_1^{n-1})$$

$$= \theta_0^{n-1} - \alpha \frac{1}{|D|} \sum_{(x,y) \in D} (-2) (y - (\theta_1 x + \theta_0))$$

$$\theta_1^n = \theta_1^{n-1} - \alpha \frac{\partial}{\partial \theta_1} MSE(\theta_0^{n-1}, \theta_1^{n-1})$$

$$= \theta_1^{n-1} - \alpha \frac{1}{|D|} \sum_{(x,y) \in D} (-2x)(y - (\theta_1 x + \theta_0))$$

Stochastic Gradient Descent (SGD)

1. Start with any $\vec{\theta}$
2. Iteratively:
   a) take a datapoint $(\vec{x}, y) \in D$
   b) calculate the derivative of MSE on this single datapoint with respect to $\vec{\theta}$
   c) shift θ in the direction opposite to the derivative

Problem: SGD can be unstable and diverge

Mini-batch Gradient Descent

Remark: most of the time when people say SGD they mean Mini-batch GD

1. Start with any $\vec{\theta}$
2. Iteratively:
   a) take a mini-batch $\{(\vec{x}, y)\} \subset D$
   b) calculate the derivative of MSE on this mini batch with respect to $\vec{\theta}$
   c) shift θ in the direction opposite to the derivative

There are many other variants of SGD used in practice:

- SGD with momentum
- Rmsprop

- SGD with momentum
- Rmsprop
- NAG
- Adam (the most popular)
- Ada Grad
- Ada Delta

Stochastic Gradient Descent for matrix factorization

$$\min_{\{p_u, q_i\}} \underbrace{\sum_{(u,i)\in K} (r_{ui} - q_i^T p_u)^2 + \lambda (\|q_i\|^2 + \|p_u\|^2)}_{\text{error}}$$

For simplicity consider embedding dim = 2

$$p_u = (p_{u1}, p_{u2})$$
$$q_i = (q_{i1}, q_{i2})$$

$$p_{u1}^{(n)} = p_{u1}^{(n-1)} - \alpha \frac{\partial}{\partial p_{u1}} \text{error}$$

$$\frac{\partial}{\partial p_{u1}} \text{error} = \frac{\partial}{\partial p_{u1}} \Big(\sum_{(u,i)\in K} (r_{ui} - (q_{i1} p_{u1} + q_{i2} p_{u2}))^2 + \lambda (q_{i1}^2 + q_{i2}^2 + p_{u1}^2 + p_{u2}^2) \Big)$$

$$= \sum_{(u,i)\in K} \left[\frac{\partial}{\partial p_{u1}} (r_{ui} - (q_{i1} p_{u1} + q_{i2} p_{u2}))^2 + \frac{\partial}{\partial p_{u1}} \lambda (q_{i1}^2 + q_{i2}^2 + p_{u1}^2 + p_{u2}^2) \right]$$

$$= \sum_{(u,i)\in K} \left[2(r_{ui} - (q_{i1} p_{u1} + q_{i2} p_{u2}))(-q_{i1}) + 2\lambda p_{u1} \right]$$

Denote

$$e_{ui} = r_{ui} - (q_{i1} p_{u1} + q_{i2} p_{u2})$$

Then

$$\frac{\partial}{\partial p_{u1}} \text{error} = \sum_{(u,i)\in K} (-2 e_{ui} q_{i1} + 2\lambda p_{u1})$$

Analogously for p_{u2}, q_{i1}, q_{i2}

$$\frac{\partial}{\partial p_{u2}} \text{error} = \sum_{(u,i) \in K} (-2 e_{ui} q_{i2} + 2\lambda p_{u2})$$

$$\frac{\partial}{\partial q_{u1}} \text{error} = \sum_{(u,i) \in K} (-2 e_{ui} p_{i1} + 2\lambda q_{u1})$$

$$\frac{\partial}{\partial q_{u2}} \text{error} = \sum_{(u,i) \in K} (-2 e_{ui} p_{i2} + 2\lambda q_{u2})$$

For SGD (error on a single datapoint):

$$(p_{u1}^{(n)}, p_{u2}^{(n)}) = (p_{u1}^{(n-1)}, p_{u2}^{(n-1)}) - \alpha \left(\frac{\partial}{\partial p_{u1}} \text{error}, \frac{\partial}{\partial p_{u2}} \text{error} \right)$$

$$= (p_{u1}^{(n-1)}, p_{u2}^{(n-1)}) - \alpha \left(-2 e_{ui} q_{i1} + 2\lambda p_{u1}, -2 e_{ui} q_{i2} + 2\lambda p_{u2} \right)$$

$$= (p_{u1}^{(n-1)}, p_{u2}^{(n-1)}) + 2\alpha \left(e_{ui} q_{i1} - \lambda p_{u1}, e_{ui} q_{i2} - \lambda p_{u2} \right)$$

Analogously for q_{i1}, q_{i2}

Rewriting the above equations in vector form gives the following formulation (renaming α to be 2α)

$$p_u^{(n)} = p_u^{(n-1)} + \alpha (e_{ui} q_i - \lambda p_u)$$

$$q_i^{(n)} = q_i^{(n-1)} + \alpha (e_{ui} p_u - \lambda q_i)$$

Least Squares for a linear model

$$y = a_1 x_1 + .. + a_k x_k + \varepsilon \qquad \text{where } \varepsilon \sim N(0, \sigma)$$

in other words

$$\hat{y} = a_1 x_1 + .. + a_k x_k$$

$$y = a_1 x_1 + \ldots + a_k x_k$$

Problem: given a dataset $D = \{(\vec{x}_n, y_n)\}$ where $x_n = (x_{n1}, \ldots, x_{nk}) \in \mathbb{R}^k$ find $a_1, \ldots, a_k \in \mathbb{R}$ which minimize the squared error

$$SE = \sum_n (y_n - (a_1 x_{n1} + \ldots + a_k x_{nk}))^2$$

SE is a quadratic function of $a_1, \ldots, a_k$ and the coefficient in front of the quadratic term is positive hence the minimum is in the point $(a_1, \ldots, a_k)$ where

$$\forall_i \ \frac{\partial}{\partial a_i} SE(a_0, \ldots, a_k) = 0$$

For simplicity of notation consider the 2D case where we need to find (a_1, a_2)

We need to find a_1, a_2 where

$$\begin{cases} \dfrac{\partial}{\partial a_1} \sum_n (y_n - (a_1 x_{n1} + a_2 x_{n2}))^2 = 0 \\ \dfrac{\partial}{\partial a_2} \sum_n (y_n - (a_1 x_{n1} + a_2 x_{n2}))^2 = 0 \end{cases}$$

We have

$$\frac{\partial}{\partial a_1} \sum_n (y_n - (a_1 x_{n1} + a_2 x_{n2}))^2 = \sum_n \frac{\partial}{\partial a_1} (y_n - (a_1 x_{n1} + a_2 x_{n2}))^2$$

$$= \sum_n 2 (y_n - (a_1 x_{n1} + a_2 x_{n2}))(-x_{n1})$$

$$= -2 \sum_n x_{n1}\left(y_n - (a_1 x_{n1} + a_2 x_{n2})\right)$$

$$\frac{\partial}{\partial a_2} \sum_n \left(y_n - (a_1 x_{n1} + a_2 x_{n2})\right)^2 = -2 \sum_n x_{n2}\left(y_n - (a_1 x_{n1} + a_2 x_{n2})\right)$$

$$\Updownarrow$$

$$\begin{cases} \sum_n x_{n1}\left(y_n - (a_1 x_{n1} + a_2 x_{n2})\right) = 0 \\ \sum_n x_{n2}\left(y_n - (a_1 x_{n1} + a_2 x_{n2})\right) = 0 \end{cases}$$

$$\Updownarrow$$

$$\begin{cases} \sum_n x_{n1} y_n = \sum_n x_{n1}^2 \, a_1 + \sum_n x_{n1} x_{n2} \, a_2 \\ \sum_n x_{n2} y_n = \sum_n x_{n1} x_{n2} \, a_1 + \sum_n x_{n2}^2 \, a_2 \end{cases}$$

$$\Updownarrow$$

$$\begin{cases} \sum_n x_{n1} y_n = a_1 \sum_n x_{n1}^2 + a_2 \sum_n x_{n1} x_{n2} \\ \sum_n x_{n2} y_n = a_1 \sum_n x_{n1} x_{n2} + a_2 \sum_n x_{n2}^2 \end{cases} \qquad (*)$$

$$\Updownarrow$$

$$\begin{cases} \sum_n x_{n1} y_n \sum_n x_{n2}^2 = a_1 \sum_n x_{n1}^2 \sum_n x_{n2}^2 + a_2 \sum_n x_{n1} x_{n2} \sum_n x_{n2}^2 \\ \sum_n x_{n2} y_n \sum_n x_{n1} x_{n2} = a_1 \sum_n x_{n1} x_{n2} \sum_n x_{n1} x_{n2} + a_2 \sum_n x_{n2}^2 \sum_n x_{n1} x_{n2} \end{cases}$$

$$\Downarrow$$

$$a_1 = \frac{\sum_n x_{n2} y_n \sum_n x_{n1} x_{n2} - \sum_n x_{n1} y_n \sum_n x_{n2}^2}{\left(\sum_n x_{n1} x_{n2}\right)^2 - \sum_n x_{n1}^2 \sum_n x_{n2}^2}$$

Denoting $X_1 = (x_{11}, x_{21}, \dots, x_{n1})$ – the first coordinate of every datapoint
$X_2 = (x_{12}, x_{22}, \dots, x_{n2})$ – the second coordinate of every datapoint

Denoting $X_1 = (x_{11}, x_{21}, \dots, x_{n1})$ – the first [illegible]
$X_2 = (x_{12}, x_{22}, \dots, x_{n2})$ – the second coordinate of every datapoint

we can write the formula for a_1 in a concise way

$$a_1 = \frac{(X_2 \cdot y)(X_1 \cdot X_2) - (X_1 \cdot y)\,\|X_2\|^2}{X_1 \cdot X_2 - \|X_1\|^2 \|X_2\|^2}$$

and for a_2:

$$a_2 = \frac{(X_1 \cdot y)(X_1 \cdot X_2) - (X_2 \cdot y)\,\|X_1\|^2}{X_1 \cdot X_2 - \|X_1\|^2 \|X_2\|^2}$$

The system of equations $(*)$ can be written in a more generic way

$$\begin{cases} \sum_n x_{n1} y_n = a_1 \sum_n x_{n1}^2 + a_2 \sum_n x_{n1} x_{n2} \\ \sum_n x_{n2} y_n = a_1 \sum_n x_{n1} x_{n2} + a_2 \sum_n x_{n2}^2 \end{cases}$$

$$\Updownarrow$$

$$\boxed{X^T y = X^T X A}$$

where

$$X = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad A = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

Then

$$X^T y = X^T X A \quad \Big| \cdot (X^T X)^{-1}$$

$$\Updownarrow$$

$$\boxed{A = (X^T X)^{-1} X^T y}$$

This form holds also for k-dimensional x vectors

$X^T X$ is a $k \times k$ dimensional matrix
$\Rightarrow$ if k is small, inverting $X^T X$ is not costly

ALS - Alternating Least Squares

Recall that the matrix factorization problem is given by

$$\min_{p_u, q_i \in \mathbb{R}^d} \sum_{(u,i) \in K} (r_{u,i} - q_i^T p_u)^2$$

If we fix the item representations q_i, then the problem becomes

$$\min_{p_u \in \mathbb{R}^d} \sum_i (r_{u,i} - (q_{i1} p_{u1} + \dots + q_{id} p_{ud}))^2$$

where this expression has to be minimized for every user over all possible values of $p_u = (p_{u1}, p_{u2}, \dots, p_{ud})$

This is the Linear Least Squares problem!

Therefore

$$p_u = (X^T X)^{-1} X^T y$$

where

$$X = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1d} \\ q_{21} & q_{22} & \cdots & q_{2d} \\ & \vdots & & \\ q_{m1} & q_{m2} & \cdots & q_{md} \end{bmatrix} \qquad y = \begin{bmatrix} r_{u1} \\ r_{u2} \\ \vdots \\ r_{um} \end{bmatrix}$$

ALS

1. Initialize all user and item representation vectors p_u and q_i with random values
2. Iterate until convergence
   (i.e. changing of representations less than ε)
   2.a. Set all item representations q_i and solve the Linear Least Squares problem for user representations p_u
   2.b. Set all user representations p_u and solve the Linear Least Squares problem for item representations q_i

Maximum Likelihood Estimation (MLE)

Consider again the following linear model

$$y = a_1 x_1 + \dots + a_k x_k + \varepsilon \qquad \text{where } \varepsilon \sim N(0, \sigma)$$

Assuming this model is true the likelihood of observing a datapoint $(x_1, x_2, \dots, x_k, y)$ in the data is equal to

$$L(\varepsilon) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\varepsilon}{\sigma}\right)^2} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2}$$

The idea behind MLE is that for a given set of observed datapoints $\{(\vec{x}_n, y_n)\} = \{(x_{n1}, x_{n2}, \dots, x_{nk}, y_n)\}$ we want to find such model parameters $a_1, a_2, \dots, a_k$ that the likelihood of observing such dataset is maximal, i.e. we want to solve

$$\max \prod \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2}$$

$$\max_{a_1,\dots,a_k} \prod_n \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y-(\dots)}{\sigma}\right)}$$

This expression can be further simplified since:

$$\arg\max_{a_1,\dots,a_k} \prod_n \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2}$$

$$= \arg\max_{a_1,\dots,a_k} \prod_n e^{-\frac{1}{2}\left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2}$$

$$= \arg\max_{a_1,\dots,a_k} e^{-\frac{1}{2}\sum_n\left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2} \qquad \left(\text{since } e^a e^b = e^{a+b}\right)$$

$$= \arg\min_{a_1,\dots,a_k} \sum_n \left(\frac{y-(a_1x_1+a_2x_2+\dots+a_kx_k)}{\sigma}\right)^2 \qquad \left(\text{since } e^{-x} \text{ is decreasing}\right)$$

$$= \arg\min_{a_1,\dots,a_k} \sum_n \left(y-(a_1x_1+a_2x_2+\dots+a_kx_k)\right)^2$$

But this is exactly Least Squares!

MLE and Least Squares are equivalent if the noise in the data is normal (Gaussian)

Note

MLE is a powerful and general method which can be used with any probability distribution, for instance Bernoulli, Binomial, Poisson, Exponential, Gamma, Beta