CSE 475: Statistical Methods in AI **Monsoon 2018**

Lec 5: Linear Methods-III

Lecturer: C. V. Jawahar *Date: Aug. 16, 2018*

5.11 Gradient Descent Procedure

We are interested in finding the optimal $\mathbf{w}$ or $\mathbf{w}^*$ corresponding to the minima of $J(\mathbf{w})$. We know the gradient descent optimization procedure for this as:

1. Start with an arbitrary $\mathbf{w}$, $k = 0$
2. Improve $\mathbf{w}$ as
$$\mathbf{w}^{k+1} \leftarrow \mathbf{w}^k - \eta \nabla J$$
3. $k \leftarrow k + 1$
4. Repeat steps 2-3 until some convergence criteria is met.

Convergence criteria can be (i) the change in weight $\mathbf{w}$ or something similar. It is quite intuitive to see that the solution is improving in each iteration.

5.11.1 Concerns

There are many concerns to us at this stage to appreciate this algorithm fully:

- Intuitively, it works, and we appreciate in 1D. What about in a general situations? How do we argue? How did we come up with this update rule?
- There seems to be an adhoc η, learning rate, in the equation. How do we fix this? Indeed, we can find a good η in every iteration. Therefore, it may be worth to write η as η^k, truely.
- What about non-convex functions? That is also a major concern. Surprisingly, this is one of the most favorite methods for optimizing non-convex functions.
- Is this the best update procedure? Can we have better update rules than this?

There are many such very relevant concerns. Let us see if we can find answers to some of these in this lecture.

5.11.2 A trivial case

Consider a function that is linear. (draw $f()$ as a line and x and y are two points on the x axis. Assume we know $f(x)$. How do we compute $f(y)$?

$$f(y) = f(x) + (y - x)f'(x)$$

Simple.!!?

What does it say? If we know xm $f(x)$ and $f'(x)$, then we can compute the function at a new point y. If we want to know where $f(y) = 0$, we can solve $f(x)+(y-x)f'(x) = 0$ and find the y of our interest.

Our problem is not this simple, because the functions of our interest are more complex.

5.12 Taylor Series

We know from the past about a function getting expressed in terms of neighbors. Taylor series is a representation of a function as the sum of terms involving function's derivatives at a single point. It is an infinite series, as long as derivatives do not vanish. You might have studied it in different forms. Hope one of the following equations reminds you the details from your past maths lectures.

$$f(x + h) = f(x) + hf'(x) + \frac{h^2}{2} f''(x) + \dots$$

$$f(y) = f(x) + (y - x)f'(x) + \frac{(y - x)^2}{2} f''(x) + \dots$$

$$f(\mathbf{y}) = f(\mathbf{x}) + [\mathbf{y} - \mathbf{x}] \cdot \nabla \mathbf{f}(\mathbf{x}) + \frac{1}{2}[\mathbf{y} - \mathbf{x}]^T \mathbf{H}(\mathbf{x})[\mathbf{y} - \mathbf{x}] + \dots$$

$$\begin{aligned} f(\mathbf{w^{k+1}}) &= f(\mathbf{w^k}) + [\mathbf{w^{k+1}} - \mathbf{w^k}] \cdot \nabla f(\mathbf{w^k}) \\ &+ \frac{1}{2}[\mathbf{w^{k+1}} - \mathbf{w^k}]^T \mathbf{H}[\mathbf{w^{k+1}} - \mathbf{w^k}] + \dots \end{aligned}$$

By now, you should realize why Taylor series come at this stage!!

We may also assume our update is more generic and as

$$\mathbf{w}^{k+1} \leftarrow \mathbf{w}^k + \mathbf{s} \qquad (5.10)$$

$$f(\mathbf{w^{k+1}}) = f(\mathbf{w^k}) + \mathbf{s}^T \nabla f + \frac{1}{2} \mathbf{s}^T \mathbf{H} \mathbf{s} + \dots$$

Note that we ∇ and Hessians are evaluated at $\mathbf{w}^k$, It is not written here explicitly in some of these equations, to minimize the clutter.

We are now in a position to have a closer look at the GD in the next section. Here our objective function J needs to be expanded and optimized

5.13 Closer Look at GD

Let us first reproduce the two equations that we may need:

$$\begin{aligned} J(\mathbf{w^{k+1}}) &= J(\mathbf{w^k}) + [\mathbf{w^{k+1}} - \mathbf{w^k}] \cdot \nabla J(\mathbf{w^k}) \\ &+ \frac{1}{2}[\mathbf{w^{k+1}} - \mathbf{w^k}]^T \mathbf{H} [\mathbf{w^{k+1}} - \mathbf{w^k}] + \dots \end{aligned}$$

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w^k}) + \mathbf{s}^T \nabla J + \frac{1}{2}\mathbf{s}^T \mathbf{H}\mathbf{s} + \dots$$

Let us try to answer three relevant questions:

1. What happens when our familiar gradient descent update rule is used in a general situation?
2. What is the optimal learning rate?
3. Is this the best update rule that we can have?

5.13.1 GD Improves in each step

Let us remember our update rule as

$$\mathbf{w^{k+1}} = w^k - \eta \nabla J(\mathbf{w}^k)$$

or the change in weight (or weight uodate)

$$\mathbf{w}^{k+1} - \mathbf{w}^k = \mathbf{s} = -\eta \nabla J(\mathbf{w}^k)$$

Also let us look at the first order approximation of the laylor series as:

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w^k}) + \mathbf{s}^T \nabla J(\mathbf{w}^k)$$

Substituting for $\mathbf{s}$

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w}^k) - \eta \nabla J(\mathbf{w}^k)^T \nabla J(\mathbf{w}^k)$$

or

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w}^k) - \eta((\text{a non negative quantity})$$

Therefore, when η is positive, we can argue that the objective is improving in every iteration. When the gradient is zero, no change in the objective and the iterations stops.

One can extend this argument to say that as long as the function is bounded below, we will be able to get to the minima. Depending on η, we may take more iterations. This naturally lead to the question of optimal learning rate.

5.13.2 Optimal learning rate

What is the best learning rate η? Let us reproduce the relevant equations and look at this problem:

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w^k}) + \mathbf{s}^T \nabla J + \frac{1}{2}\mathbf{s}^T \mathbf{H}\mathbf{s}$$

$$\mathbf{w}^{k+1} - \mathbf{w}^k = \mathbf{s} = -\eta \nabla J(\mathbf{w}^k)$$

Substituting

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w^k}) - \eta \nabla \mathbf{J^T} \nabla \mathbf{J} + \frac{\eta^2}{2} \nabla \mathbf{J}^T \mathbf{H} \nabla \mathbf{J}$$

We want to optimize for η, Let us optimize for η by differentiating with respect to η and equating to zero.

This leads to

$$-||\nabla \mathbf{J}||^2 + \eta \nabla \mathbf{J}^T \mathbf{H} \nabla \mathbf{J} = 0$$

or

$$\eta = \frac{||\nabla \mathbf{J}||^2}{\nabla \mathbf{J}^T \mathbf{H} \nabla \mathbf{J}}$$

Note that this needs change at every iteration k since J and $\mathbf{H}$ are also function of $\mathbf{w^k}$.

This assumes the second order approximation of the function. Though such optimal learning rates could improve the convergence at every point, it also adds an additional computation in each iteration.

5.13.3 Better Update Rule

Let us also see if there is a better update rule. i.e., is there a better $\mathbf{s}$?

Let us revisit:

$$J(\mathbf{w^{k+1}}) = J(\mathbf{w^k}) + \mathbf{s}^T \nabla J + \frac{1}{2}\mathbf{s}^T \mathbf{H}\mathbf{s}$$

Differentiating with respect to $\mathbf{s}$ and equating to zero:

$$\nabla \mathbf{J} + \mathbf{H}\mathbf{s} = 0$$

Or

$$\mathbf{s} = -\mathbf{H}^{-1} \nabla \mathbf{J}$$

This leads to a better update rule known as Newton's updates.

Q: Write the psudocode for the newtons method for optimization.

Newton's method is known to provide faster convergence for the optimization. Indeed the second order approximations we use here is more accurrate than the first order one.

Q: However, Newton's method is not very popular in practice. Why is this? You will see plenty of discussions online. Read and appreciate.

5.14 Discussions

Gradient descent algorithms are the most popular and the most effective ones to train many machine learning models (including deep neural networks). The analysis we had here is quite minimal. Deeper understanding is good to have. If you are interested in this line, here are two references (second one a bit more advanced).

- Leon Bottou, "Stochastic Gradient Descent Tricks, 2012
- Leon Bottou, Frank E. Curtis, Jorge Nocedal "Optimization Methods for Large-Scale Machine Learning", 2018

I advise you to read the first, even if you do not understand everything. Second is for people who can walk that extra mile. Both are beyond the scope of this course, and the exams for sure.