A Brief Introduction to Gradient Descent

(And A Closer Look at Stochastic Gradient Descent)

James Horine

January 19, 2018

A Gentle Reminder

"Essentially, all models are wrong, but some are useful"

- Box

Motivation

Typically, statistically driven data problems in data science have closed form solutions.

Motivation

Typically, statistically driven data problems in data science have closed form solutions.

That means that the “formula” may be written down with pen and paper.

Motivation

Typically, statistically driven data problems in data science have closed form solutions.

That means that the “formula” may be written down with pen and paper.

$\Longrightarrow$ Sometimes this leads to easy solutions that are computationally expensive.

Recall the OLS Regression Problem

If we assume that the data has a *population* model $\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \text{ and } \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

One may show, with some linear algebra and calculus, that under some general conditions, the Ordinary Least Squares estimates for the parameter vector, β is given by

$$\hat{\beta}_{OLS} = (X`X)^{-1}X`y \tag{1}$$

We make the assumption here that the deviations, or residuals from the *predicted* regression hyper surface are $\epsilon_i \sim_{iid} N(0, \sigma_x)$

Recall the OLS Regression Problem

The Normality assumptions on the residuals already rather strong, but what else is there?

1. Does regression make sense to use?
2. Do the data form a joint multivariate normal distribution?
3. If not, are there extensions to OLS theory that make sense to use? GLMs?
4. What is the complexity of the data?
5. What does the response manifold look like? Is $\mathbb{Y} \in \mathbb{R}^p$ where $p < \infty$...? $p << \infty$...? $p <<< \infty$...?
6. What about multicollinearity?
7. and
8. and...

Optimization

So, what of other methods - In particular, numerical optimization of an objective/cost/loss function that describes the error in a closed form way?

Gradient Descent

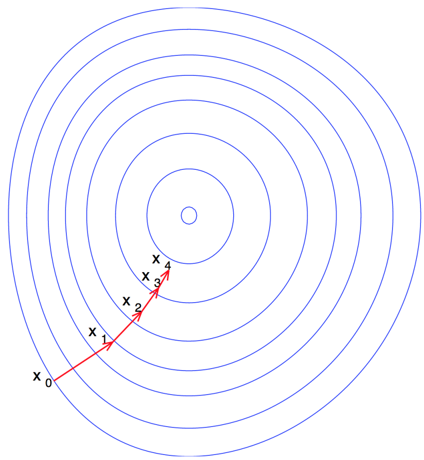

Figure: Gradient Descent Path

Fantastic animated viewer available here:
`http://vis.supstat.com/2013/03/gradient-descent-algorithm-with-r/`

Gradient Descent

Under some... general* conditions we may (and shall) implement Gradient Descent...
The Algorithm:

1. Initialize at $\beta_{\textbf{init}}$
2. refine the value
3. Repeat until we fail to make any more reasonable progress towards the extrema of interest. Convergence not covered here today.

Gradient Descent

Under some... general* conditions we may (and shall) implement Gradient Descent...
The Algorithm:

1. Initialize at $\beta_{\textbf{init}}$
2. refine the value
3. Repeat until we fail to make any more reasonable progress towards the extrema of interest. Convergence not covered here today.

Each update or refinement is: $\beta_{\text{new}} = \beta_{\text{old}} \pm \gamma \bullet \nabla f(\beta_{\text{old}})$

$\implies \gamma$ is our *step size*, or learning rate, or learning parameter

Gradient Descent

Let us define the objective function for regression as

$$S = \sum_{i=1}^{n} \epsilon_i \tag{2}$$

$$S^2 = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i})^2 \tag{3}$$

Then, with a bit of calculus, we may obtain p-many derivatives as

$$\frac{\delta S}{\delta \beta_j} = \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i}) X_{p_i} \quad \text{for j} > 0 \tag{4}$$

$$\frac{\delta S}{\delta \beta_j} = \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i}) \quad \text{for j} = 0 \tag{5}$$

Gradient Descent

Let us define the objective function for regression as

$$S = \sum_{i=1}^{n} \epsilon_i \tag{2}$$

$$S^2 = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i})^2 \tag{3}$$

Then, with a bit of calculus, we may obtain p-many derivatives as

$$\frac{\delta S}{\delta \beta_j} = \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i}) X_{p_i} \quad \text{for j} > 0 \tag{4}$$

$$\frac{\delta S}{\delta \beta_j} = \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 X_{1_i} - \ldots - \beta_p X_{p_i}) \quad \text{for j} = 0 \tag{5}$$

We may now write out the analytic form of the gradient!

$$\nabla(S) = \left\langle \frac{\delta S}{\delta \beta_0}, \frac{\delta S}{\delta \beta_1}, \ldots, \frac{\delta S}{\delta \beta_p} \right\rangle \tag{6}$$

$$\nabla(S) = \sum_{j=1}^{p} \frac{\delta S}{\delta \beta_j} \tag{7}$$

Gradient Descent

However, there are some concessions:

Gradient Descent

However, there are some concessions:

1. Not guaranteed to find the global extrema!

Gradient Descent

However, there are some concessions:

1. Not guaranteed to find the global extrema!
   - multiple random restarts
   - high dimensional problems makes this challenging

Gradient Descent

However, there are some concessions:

1. Not guaranteed to find the global extrema!
   - multiple random restarts
   - high dimensional problems makes this challenging
2. Choosing the step size γ, or the learning rate is not always straight forward

Gradient Descent

However, there are some concessions:

1. Not guaranteed to find the global extrema!
   - multiple random restarts
   - high dimensional problems makes this challenging
2. Choosing the step size γ, or the learning rate is not always straight forward
   - algorithms exist to help with this...but they introduce many more parameters

Gradient Descent

However, there are some concessions:

1. Not guaranteed to find the global extrema!
   - multiple random restarts
   - high dimensional problems makes this challenging
2. Choosing the step size γ, or the learning rate is not always straight forward
   - algorithms exist to help with this...but they introduce many more parameters
3. Each iteration requires computing the total gradient over all the data...
   - Stochastic Gradient Descent!

Stochastic Gradient Descent

Under some more general assumptions, we may implement Stochastic Gradient Descent:

1. Shuffle the data / Draw a random subset of the data of size M $<<$ N
2. For $i \in 1, \ldots, M$
   do: $\{$

   "stuff"

   $\}$
3. Repeat until mild convergence

Stochastic Gradient Descent

Under some more general assumptions, we may implement Stochastic Gradient Descent:

1. Shuffle the data / Draw a random subset of the data of size M $<<$ N
2. For $i \in 1, \ldots, M$
   do: {

      "stuff"

   }
3. Repeat until mild convergence

What is stuff?

Stochastic Gradient Descent

Under some more general assumptions, we may implement Stochastic Gradient Descent:

1. Shuffle the data / Draw a random subset of the data of size M << N
2. For $i \in 1, \ldots, M$
   do: $\{$

   "stuff"

   $\}$
3. Repeat until mild convergence

What is stuff?

"stuff" is exactly as before, where we update the parameter vector with the gradient!

$$\beta_{\text{new}} = \beta_{\text{old}} \pm \gamma' \bullet \nabla f(\beta_{\text{old}}) \tag{8}$$

Stochastic Gradient Descent

Now, I rarely change notation, unless I have a really good reason to.
Notice:

- GD: $\beta_{\text{new}} = \beta_{\text{old}} \pm \gamma \bullet \nabla f(\beta_{\text{old}})$
- SGD: $\beta_{\text{new}} = \beta_{\text{old}} \pm \gamma' \bullet \nabla f(\beta_{\text{old}})$

In the SGD setting, We are using γ', which has the same step size property. However, I want to make sure I do not wander around aimlessly, so I now define:

at initialization:

$$\gamma' = \gamma \tag{9}$$

with each epoch repetition:

$$\gamma'_{new} = \gamma'_{old} \bullet \omega \tag{10}$$

That is to say, I wish to *decay* the *learning rate* by $\omega \in [0, 1]$ at each epoch!

Appendix

general* $:=$

1. $(\mathbb{H} := L_2(\mathbb{R}^n)|\forall f, g \in \mathbb{R}^n, < f, g >= \int_\mu (\bar{f} \bullet g) d\mu$
   - (or $\mathbb{C}^n$ for that matter...)
2. f,g are square integrable, i.e. $\int\limits_{-\infty}^{\infty} |f(x)|^2 dx < \infty$)
3. f, g are convex, i.e.: $\forall x_1, x_2 \in X, \forall t \in [0,1] : f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$

Sources

1. https://en.wikipedia.org/wiki/Gradient_descent
2. http://jamesmc.com/coms4771/slides/gd.pdf
3. http://lear.inrialpes.fr/workshop/osl2013/slides/osl2013_bach.pdf
4. https://github.com/sachinruk/deepschool.io
5. https://arxiv.org/pdf/1609.04747.pdf
6. https://en.wikipedia.org/wiki/Convex_function