Gradient Descent

Dimitris Papadimitriou

INFO 251: Applied Machine Learning

March 4, 2020

Optimization (General Formulation)

- Most ML problems can be formulated as optimization problems

$$\min_{z_1, z_2, \ldots, z_n} J(z_1, z_2, \ldots, z_n)$$

- where $z_1, z_2 \ldots z_n$ are our parameters
- In the regression setting these would be the intercept and the slopes
- Closed form solution usually not possible $\rightarrow$ **Gradient Descent**

Gradient Descent Intuition

- Gradient always points in the direction of greatest increase of the function
- Thus negative gradient points towards steepest descent
- We expect that taking steps in the direction of the negative derivative will lead us to the minimum
- How big steps though?

Linear Regression

- Assume $y_i = a + bx_i + \epsilon_i,\ i = 1, \ldots, N$
- Objective of OLS: $J(\alpha, \beta) = \frac{1}{2N}\sum_{i=1}^{N}(y_i - \alpha - \beta x_i)^2$
- Estimate α, β by $\min_{\alpha,\beta} \frac{1}{2N}\sum_{i=1}^{N}(y_i - \alpha - \beta x_i)^2$
- "In general we optimize a function with respect to some variables by setting the derivatives w.r.t. those variables equal to zero and solving for those parameters"
- Closed form solution

$$\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = (X^T X)^{-1} X^T Y$$

where $X = \begin{bmatrix}1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N\end{bmatrix}$ and $Y = \begin{bmatrix}y_1 \\ y_2 \\ \vdots \\ y_N\end{bmatrix}$

Issues With That...

- So why GD if we have closed form solution?

Issues With That...

- So why GD if we have closed form solution?
- For OLS inverting $X^T X$ can be very demanding

Issues With That...

- So why GD if we have closed form solution?
- For OLS inverting $X^T X$ can be very demanding
- Also OLS objective is easily differentiable but this is not the case in general!

Gradient Descent

- In class we derived:

$$\frac{\partial J(\alpha,\beta)}{\partial \alpha} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \alpha - \beta x_i)$$

$$\frac{\partial J(\alpha,\beta)}{\partial \beta} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \alpha - \beta x_i)x_i$$

- GD steps:

$$\alpha \leftarrow \alpha - R\frac{\partial J(\alpha,\beta)}{\partial \alpha}$$

$$\beta \leftarrow \beta - R\frac{\partial J(\alpha,\beta)}{\partial \beta}$$

Or more concisely

$$\tilde{\beta} \leftarrow \tilde{\beta} - R\nabla J(\tilde{\beta}),$$

where $\tilde{\beta} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$ and $\nabla J(\tilde{\beta}) = \begin{bmatrix}\frac{\partial J(\alpha,\beta)}{\partial \alpha} \\ \frac{\partial J(\alpha,\beta)}{\partial \beta}\end{bmatrix}$

Gradient Descent Algorithm

Algorithm 1 GD

```
1: Initialize $\alpha, \beta$, select $R$
2: for $i = 1, 2, \ldots$ do
3:
       $$\alpha \leftarrow \alpha - R\frac{\partial J(\alpha, \beta)}{\partial \alpha}$$
       $$\beta \leftarrow \beta - R\frac{\partial J(\alpha, \beta)}{\partial \beta}$$
4:     If convergence criterion achieved break
5: end for
```

Gradient Descent

- Convergence? YES! Generally in local minima
- Convergence criterion: When parameters no longer change, i.e. gradients are zero
- Choose learning rate? Small enough so that it does not diverge but large enough so that it converges fast e.g $10^{-2}, 10^{-3} \dots$
- Fancier rules e.g. line search $R = \text{argmin}_r J(x - r\nabla_x J(x))$ for $t \in \mathbb{R}_+$

Mini-Batch and Stochastic Gradient Descent

- Stochastic
- Instead of using all data points for gradient calculation use only one point
- Choose it randomly
- Diminishing step size e.g. 1/#iterations

Mini-Batch and Stochastic Gradient Descent

- Stochastic
- Instead of using all data points for gradient calculation use only one point
- Choose it randomly
- Diminishing step size e.g. 1/#iterations
  - Pros: Computationally more efficient
  - Cons: Slower to converge
- Mini-Batch GD best of both worlds
  - Same notion as SGB only instead of 1 point we choose a batch K
  - typical values for K are $8, 16, 24, 32$ etc

SGD and Mini-Batch GD Algorithms

Algorithm 2 Mini-Batch GD

```
1: Initialize $\alpha, \beta$, select $R$, batch-size $K$
2: for $i = 1, 2, \ldots$ do
3:     Randomly shuffle data
4:     for $j = 1, 2, \ldots, \lfloor N/K \rfloor$ do
5:         Choose $j$th batch (batch(j)) from data
6:
               $\alpha \leftarrow \alpha - R\frac{\partial J^j(\alpha,\beta)}{\partial \alpha}$
               $\beta \leftarrow \beta - R\frac{\partial J^j(\alpha,\beta)}{\partial \beta}$
7:     end for
8:     If convergence criterion achieved break
9: end for
```

where for our bivariate regression problem

$$\frac{\partial J^j(\alpha,\beta)}{\partial \alpha} = \frac{1}{K}\sum_{k \in batch(j)} y_k - \alpha - \beta x_k, \quad \frac{\partial J^j(\alpha,\beta)}{\partial \beta} = \frac{1}{K}\sum_{k \in batch(j)} (y_k - \alpha - \beta x_k) x_k$$

Implementation Details

- Remember when shuffling your data to shuffle y and x simultaneously
- i.e. If initially

after shuffling and not

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad y = \begin{bmatrix} y_2 \\ y_3 \\ y_1 \end{bmatrix} x = \begin{bmatrix} x_2 \\ x_3 \\ x_1 \end{bmatrix} \quad y = \begin{bmatrix} y_2 \\ y_1 \\ y_3 \end{bmatrix} x = \begin{bmatrix} x_1 \\ x_3 \\ x_2 \end{bmatrix}$$

- In non-convex scenaria you should try different initialization points and compare

Linear Regression (Closed form Solution)

- Remember $||z||_2^2 = z^T z = z_1^2 + z_2^2 + \cdots + z_n^2 = \sum_{i=1}^{N} z_i^2$
- Vector calculus:
- Gradient is

$$\nabla_z J(z) = \nabla_{z_1,\ldots,z_n} J(z_1, \ldots, z_n) = \begin{bmatrix} \frac{\partial J(z_1,\ldots,z_n)}{\partial z_1} \\ \vdots \\ \frac{\partial J(z_1,\ldots,z_n)}{\partial z_n} \end{bmatrix}$$

- Also $\nabla_z \, z^T Az = 2Az$ (for any square symmetric matrix)
- $\nabla_z a^T z = a$ and $\nabla_z z^t a = a$ for any vector $a \in \mathbb{R}^n$

- Objective can be written as

$$J(\alpha, \beta) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - a - \beta x_i)^2$$

- Objective can be written as

$$J(\alpha, \beta) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - a - \beta x_i)^2$$

$$\frac{1}{2N} \left\| \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \alpha - \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \beta \right\|_2^2$$

- Objective can be written as

$$J(\alpha,\beta) = \frac{1}{2N}\sum_{i=1}^{N}(y_i - a - \beta x_i)^2$$

$$\frac{1}{2N}\left\|\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}\alpha - \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}\beta\right\|_2^2 = \frac{1}{2N}\left\|\overbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}^{Y} - \overbrace{\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_n \end{bmatrix}}^{X}\overbrace{\begin{bmatrix} \alpha \\ \beta \end{bmatrix}}^{\tilde{\beta}}\right\|_2^2$$

- Objective can be written as

$$J(\alpha, \beta) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - a - \beta x_i)^2$$

$$\frac{1}{2N} \left\| \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \alpha - \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \beta \right\|_2^2 = \frac{1}{2N} \left\| \overbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}^{Y} - \overbrace{\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_n \end{bmatrix}}^{X} \overbrace{\begin{bmatrix} \alpha \\ \beta \end{bmatrix}}^{\tilde{\beta}} \right\|_2^2$$

$$= \frac{1}{2N} ||Y - X\tilde{\beta}||_2^2$$

- Objective can be written as

$$J(\alpha,\beta) = \frac{1}{2N}\sum_{i=1}^{N}(y_i - a - \beta x_i)^2$$

$$\frac{1}{2N}\left\|\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}\alpha - \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}\beta\right\|_2^2 = \frac{1}{2N}\left\|\overbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}^{Y} - \overbrace{\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_n \end{bmatrix}}^{X}\overbrace{\begin{bmatrix} \alpha \\ \beta \end{bmatrix}}^{\tilde{\beta}}\right\|_2^2$$

$$= \frac{1}{2N}||Y - X\tilde{\beta}||_2^2 = \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta})$$

Linear Regression (Closed Form Solution)

- gradient

$$
\begin{aligned}
J(\alpha, \beta) = J(\tilde{\beta}) &= \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta}) \\
&= \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY)
\end{aligned}
$$

Linear Regression (Closed Form Solution)

- gradient

$$\begin{aligned} J(\alpha, \beta) = J(\tilde{\beta}) &= \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta}) \\ &= \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY) \end{aligned}$$

$$\begin{aligned} \nabla_{\tilde{\beta}}J(\tilde{\beta}) &= \nabla_{\tilde{\beta}}\frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY) \\ &= \frac{1}{2N}(2X^TX\tilde{\beta} - X^TY - X^TY) = \frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY) \end{aligned}$$

Linear Regression (Closed Form Solution)

- gradient

$$J(\alpha, \beta) = J(\tilde{\beta}) = \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta})$$
$$= \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY)$$

$$\nabla_{\tilde{\beta}}J(\tilde{\beta}) = \nabla_{\tilde{\beta}}\frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY)$$
$$= \frac{1}{2N}(2X^TX\tilde{\beta} - X^TY - X^TY) = \frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY)$$

- Set derivative (gradient) equal to zero

Linear Regression (Closed Form Solution)

- gradient

$$\begin{aligned} J(\alpha,\beta) = J(\tilde{\beta}) &= \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta}) \\ &= \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY) \end{aligned}$$

$$\begin{aligned} \nabla_{\tilde{\beta}} J(\tilde{\beta}) &= \nabla_{\tilde{\beta}} \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY) \\ &= \frac{1}{2N}(2X^TX\tilde{\beta} - X^TY - X^TY) = \frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY) \end{aligned}$$

- Set derivative (gradient) equal to zero
- $\frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY) = 0 \rightarrow X^TX\tilde{\beta} = X^TY$

Linear Regression (Closed Form Solution)

- gradient

$$J(\alpha, \beta) = J(\tilde{\beta}) = \frac{1}{2N}(Y - X\tilde{\beta})^T(Y - X\tilde{\beta})$$
$$= \frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY)$$

$$\nabla_{\tilde{\beta}}J(\tilde{\beta}) = \nabla_{\tilde{\beta}}\frac{1}{2N}(Y^TY + \tilde{\beta}^TX^TX\tilde{\beta} - Y^TX\tilde{\beta} - \tilde{\beta}^TX^TY)$$
$$= \frac{1}{2N}(2X^TX\tilde{\beta} - X^TY - X^TY) = \frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY)$$

- Set derivative (gradient) equal to zero
- $\frac{1}{2N}(2X^TX\tilde{\beta} - 2X^TY) = 0 \rightarrow X^TX\tilde{\beta} = X^TY$
- $\boldsymbol{\tilde{\beta} = (X^TX)^{-1}X^TY}$