Exercise 2 - Linear regression

Đặng Linh Anh

December 2020

1. Đọc hiểu code (file `D5, D6, D7, D8`) về cách train bài toán linear regression theo cách thông thường và vectorization cho m and N sample.

2. So sánh L1 loss (absolute error) và L2 loss (squared error). Nêu ưu nhược điểm của chúng.

Lời giải

i. **L1 loss (absolute error)**

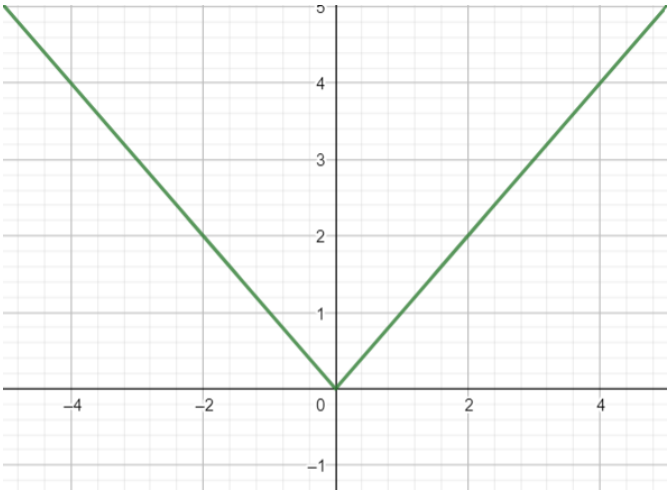
$$L = |\hat{y} - y| \tag{1}$$

$$\Rightarrow \frac{dL}{d\theta} = \frac{\hat{y} - y}{|\hat{y} - y|} \times \mathbf{x} \tag{2}$$

$$\Leftrightarrow \frac{dL}{d\theta} = \begin{cases} \mathbf{x}, & \text{khi } \hat{y} - y \geq 0 \\ -\mathbf{x}, & \text{khi } \hat{y} - y < 0 \end{cases} \tag{3}$$

Cập nhật tham số:

$$\theta = \theta \pm \eta \times \mathbf{x} \tag{4}$$



Hình 1: Đồ thị hàm loss L1

- Ưu điểm
  Ta thấy, với một sample xác định, giá trị $|L'_\theta|$ không thay đổi. Do đó, tốc độ thay đổi của tham số trong quá trình train khá ổn định, đặc biệt đối với nhiễu. Tuy nhiên, sự ổn định này chỉ thể hiện tốt đối với các giá trị lớn.
- Nhược điểm
  Khi model tiến gần đến vị trí global minima, loss $L1$ gây ra hiện tượng loss dao động cực kỳ bất ổn quanh điểm cực tiểu. Bởi, tốc độ thay đổi của tham số vẫn giữ nguyên nên khi tiếp tục train làm cho loss vượt qua vị trí cực tiểu cần tiến tới. Lúc này, mô hình cực kỳ nhạy với nhiễu, có thể làm cho loss tăng đáng kể, tức phân kỳ ra khỏi global minima.

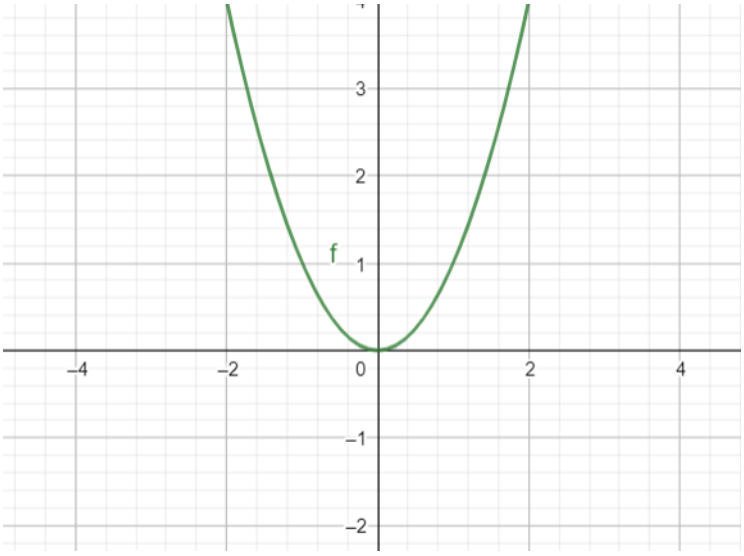
ii. **L2 loss (squared error)**.

$$L = (\hat{y} - y)^2 \tag{5}$$

$$\Rightarrow \frac{dL}{d\theta} = 2(\hat{y} - y)\mathbf{x} \tag{6}$$

Cập nhật tham số;

$$\theta = \theta - \eta \times 2(\hat{y} - y)\mathbf{x} \tag{7}$$



Hình 2: Đồ thị hàm loss L2

- Ưu điểm
  Khi giá trị $|\hat{y} - y|$ nhỏ, $\frac{dL}{d\theta}$ nhỏ, θ được cập nhật chậm nên dễ đi tới global minima, đồng thời, mô hình ổn định với nhiễu.
- Nhược điểm
  Khi giá trị $|\hat{y} - y|$ lớn, $\frac{dL}{d\theta}$ lớn (tuyến tính theo $\hat{y} - y$) nên mô hình khá nhạy với nhiễu. Giả sử, khi nhiễu sample $\mathbf{x}$ lớn, $\Delta\theta = \eta \times 2(\hat{y} - y)\mathbf{x}$ rất lớn, làm cho θ thay đổi rất nhanh gây khó khăn cho quá trình train.

3. Tìm hiểu về Hubber loss và cách Hubber loss khắc phục nhược điểm của 2 hàm loss trên. Cài đặt Hubber loss cho các chương trình ở file `D5, D6, D7, D8`.

Lời giải

Hàm Hubber loss được định nghĩa như sau:

$$\mathbf{L}_\delta(\hat{y}-y) = \begin{cases} \frac{1}{2}(\hat{y}-y)^2, & \text{khi } |\hat{y}-y| \leq \delta \\ \delta(|\hat{y}-y| - \frac{1}{2}\delta), & \text{khi } |\hat{y}-y| > \delta \end{cases} \tag{8}$$

Hàm này có dạng bình phương đối với các giá trị $|\hat{y}-y|$ nhỏ và có dạng tuyến tính khi $|\hat{y}-y|$ có giá trị lớn. Từ các ưu và nhược điểm của loss $L1$ và $L2$ kể trên, có thể thấy, Hubber loss đã khắc phục tốt các nhược điểm cũng như tận dụng các ưu điểm của loss $L1$ và $L2$. Cụ thể:

- Khi giá trị $|\hat{y}-y|$ nhỏ, Hubber loss mang dạng của loss $L2$, θ được cập nhật chậm nên dễ đi tới global minima, đồng thời, mô hình ổn định với nhiễu.
- Khi giá trị $|\hat{y}-y|$ lớn, Hubber loss mang dạng của loss $L1$, sự thay đổi của tham số trong quá trình train khá ổn định, đặc biệt đối với nhiễu
- Tuy nhiên, để Hubber loss hoạt động tốt, cần phải tính chỉnh tham số δ của hàm cho phù hợp với từng bài toán khác nhau.

Cài đặt Linear regression vơi Hubber loss:

$$\frac{dL}{d\theta} = \frac{dL}{d\hat{y}}\frac{d\hat{y}}{d\theta} \tag{9}$$

$$\Rightarrow \frac{dL}{d\theta} = \begin{cases} (\hat{y}-y)\mathbf{x}, & \text{khi } |\hat{y}-y| \leq \delta \\ \delta \times \frac{\hat{y}-y}{|\hat{y}-y|}, & \text{khi } |\hat{y}-y| > \delta \end{cases} \tag{10}$$

Chương trình Linear regression với Hubber loss:

(a) `D5 - m Samples`:
Tính toán loss và gradient cho mô hình dùng các hàm sau:

Listing 1: D5 - m Samples

```
def compute_hubber_loss(y, z, delta=1.5):
    if abs(z-y) <= delta:
        return 1/2*(z-y)**2
    else:
        return delta*(abs(z-y)-1/2*delta)

def compute_hubber_grad(x, y, z, delta=1.5):
    if abs(z-y) <= delta:
        return (z-y)*x
    else:
        return delta*(z-y)/abs(z-y)
```

Code: LINK-D5

(b) `D6 - m Samples - Vectorization`:
Tính toán loss và gradient cho mô hình dùng các hàm sau:

Listing 2: D6 - m Samples - Vectorization

```
def compute_hubber_loss(y, z, delta=5):
    return np.where(np.abs(z-y.T) <= delta,
                    1/2*np.multiply(z-y.T, z-y.T),
                    delta*(np.abs(z-y.T)-1/2*delta))

def compute_hubber_grad(x, y, z, m, delta=5):
    abs_stacked = np.tile(np.abs(z-y.T), (m, 1))
    return np.where(abs_stacked <= delta,
                    np.multiply(np.tile(z-y.T, (m, 1)), x),
                    np.tile(delta*(z-y.T)/np.abs(z-y.T), (m, 1)))
```

Code: LINK-D6

(c) `D7 - N Samples`:
Tính toán loss và gradient cho mô hình dùng các hàm sau:

Listing 3: D7 - N Samples

```
def compute_hubber_loss(y, z, delta=7):
    if abs(z-y) <= delta:
        return 1/2*(z-y)**2
    else:
        return delta*(abs(z-y)-1/2*delta)

def compute_hubber_grad(x, y, z, delta=7):
    if abs(z-y) <= delta:
        return (z-y)*x
    else:
        return delta*(z-y)/abs(z-y)
```

Code: LINK-D7

(d) `D8 - N Samples - Vectorization`:
Tính toán loss và gradient cho mô hình dùng các hàm sau:

Listing 4: D8 - N Samples - Vectorization

```
def compute_hubber_loss(y, z, delta=5):
    return np.where(np.abs(z-y.T) <= delta,
                    1/2*np.multiply(z-y.T, z-y.T),
                    delta*(np.abs(z-y.T)-1/2*delta))

def compute_hubber_grad(x, y, z, m, delta=5):
    abs_stacked = np.tile(np.abs(z-y.T), (m, 1))
    return np.where(abs_stacked <= delta,
                    np.multiply(np.tile(z-y.T, (m, 1)), x),
                    np.tile(delta*(z-y.T)/np.abs(z-y.T), (m, 1)))
```

Code: LINK-D8

4. Xây dựng công thức linear regression cho N sample khi vector $\mathbf{x}$ có dạng sau

$$\mathbf{x} = \begin{bmatrix} sample1 \\ \cdots \\ sampleN \end{bmatrix}$$

Sắp xếp theo chiều ngang, khác với chiều dọc trong bài học.

Lời giải

Cài đặt (n features):

$$\mathbf{x} = \begin{bmatrix} x_1^{(1)} & \cdots & x_n^{(1)} & 1 \\ x_1^{(2)} & \cdots & x_n^{(2)} & 1 \\ & \cdots & & \\ x_1^{(N)} & \cdots & x_n^{(N)} & 1 \end{bmatrix} \tag{11}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \cdots \\ y_N \end{bmatrix} \tag{12}$$

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \cdots \\ \hat{y}_N \end{bmatrix} \tag{13}$$

$$\theta = \begin{bmatrix} w_1 \\ \cdots \\ w_n \\ b \end{bmatrix} \tag{14}$$

$$\frac{dL}{d\theta} = \begin{bmatrix} dw_1 \\ \cdots \\ dw_n \\ db \end{bmatrix} \tag{15}$$

$$\tag{16}$$

Forward:

$$\hat{\mathbf{y}} = \mathbf{x}\theta \tag{17}$$

Loss:

$$L = \frac{1}{N} \sum_{1 \le j \le N} (\hat{y}_j - y_j)^2 \tag{18}$$

$$\Rightarrow L = \frac{1}{N} \times \underbrace{\begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}}_{N} \times [(\hat{\mathbf{y}} - \mathbf{y}) \odot (\hat{\mathbf{y}} - \mathbf{y})] \tag{19}$$

Gradient:

$$\mathbf{A} = 2 \times (\hat{\mathbf{y}} - \mathbf{y}) \tag{20}$$

$$\frac{dL}{d\theta} = \underbrace{\begin{bmatrix} \mathbf{A} & \cdots & \mathbf{A} \end{bmatrix}}_{n+1} \odot \mathbf{x} \tag{21}$$

Cập nhật tham số:

$$\theta := \theta - \eta \times \frac{dL}{d\theta} \tag{22}$$

5. Cài đặt linear regression cho bài toán advertising theo 2 cách (cách thông thường và vectorization) dùng m sample (mini-batch gradient descent).

Lời giải

(a) Thông thường
LINK CODE

(b) Vectorization
LINK CODE