Improvement Speed & Precision

จากการหา Weight

$$W = W + \frac{\alpha}{N} \times b^{T}(Y - Yhat)$$

$$Yhat = XbW$$

มีการคูณ Matrix (Operation มากๆ) = ช้า

$Xb = [N \times D]$

$Y - Yhat = [N \times K]$

$D \times N \times K$

$Xb = [N \times D]$

$W = [D \times K]$

$N \times D \times K$

Ex

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \times B = \begin{bmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{bmatrix}$$

operation $= 3 \times 2 \times 4$

3×2

2×4

ลด Operation เพิ่ม Speed

Input

$$\begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & x_1^4 \\ x_2^1 & x_2^2 & x_2^3 & x_2^4 \\ x_3^1 & x_3^2 & x_3^3 & x_3^4 \\ x_4^1 & x_4^2 & x_4^3 & x_4^4 \\ x_5^1 & x_5^2 & x_5^3 & x_5^4 \end{bmatrix}$$

1) Batch Mode (Batch Gradient Descent)
<ใช้ N ทั้งหมด>

$$\begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & x_1^4 \\ x_2^1 & x_2^2 & x_2^3 & x_2^4 \\ x_3^1 & x_3^2 & x_3^3 & x_3^4 \\ x_4^1 & x_4^2 & x_4^3 & x_4^4 \\ x_5^1 & x_5^2 & x_5^3 & x_5^4 \end{bmatrix}$$

2) Mini-Batch <ใช้ N บ้างตัว หรือ Random>

$$\begin{bmatrix} x_1^1 & x_1^2 & x_1^3 & x_1^4 \\ x_2^1 & x_2^2 & x^3 & x_2^4 \\ x_5^1 & x_5^2 & x_5^3 & x_5^4 \end{bmatrix}$$

3) Stochastic <ใช้ N = 1>

$$\begin{bmatrix} x_4^1 & x_4^2 & x_4^3 & x_4^4 \end{bmatrix}$$

Regularization

<กรณีลดตัวแปรไม่ได้ เพราะมีความสัมพันธ์กัน -> จะลดความสำคัญของตัวแปรลงแทน (Feature)>

1) Lasso Regression (L1 Regularization)

- **Regression**

$$Loss = mse + \lambda \sum_{i=1}^{N} |w_i|$$

- **Classification**

$$Loss = entropy + \lambda \sum_{i=1}^{N} |w_i|$$

$$mse = \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}$$

$$entropy = -\sum_{i=1}^{N} \frac{y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)}{N}$$

feature บางตัวที่ไม่สำคัญ ค่า Weight จะน้อย หรือ < 1 จะทำให้สมการใกล้ 0
เหมาะกับ Data ที่มีการกระจัดกระจายของข้อมูล

2) Ridge Regression (L2 Regularization)

- **Regression**

$$Loss = mse + \lambda \sum_{i=1}^{N} w^2$$

- **Classification**

$$Loss = entropy + \lambda \sum_{i=1}^{N} w^2$$

จะเฉลี่ย Weight เท่าๆ กันไม่ให้อิงกับ feature ใดมากไป

-> ค่าน้อย ทำให้ค่าสมการต่ำลง เช่น 0.01^2 = 0.001

-> ค่ามาก ทำให้ค่าสมการเพิ่มขึ้น เช่น 10^2 = 100

3) Elastic Net (L1+L2 Regularization)

- **Regression**

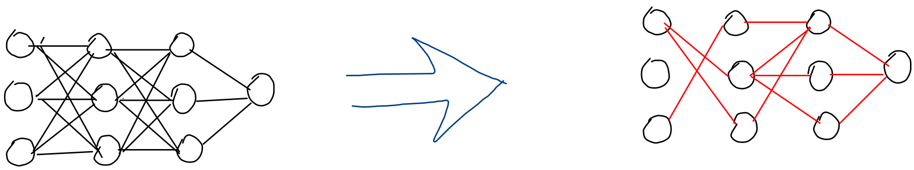
$$Loss = mse + \lambda \sum_{i=1}^{N} |w_i| + \lambda \sum_{i=1}^{N} w^2$$

- **Classification**

$$Loss = entropy + \lambda \sum_{i=1}^{N} |w_i| + \lambda \sum_{i=1}^{N} w^2$$

ดึงเอาข้อดีของ L1 กับ L2 มาใช้

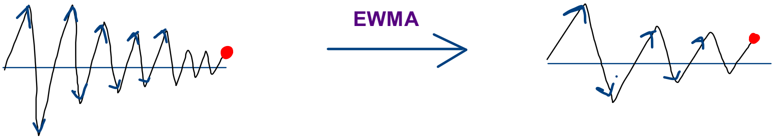
4) Dropout Regularization



5) Early Stop หยุดเมื่อได้ Loss ที่ต้องการแล้ว
(แถมๆ อาจไม่ได้จัดอยู่ใน Regularization)

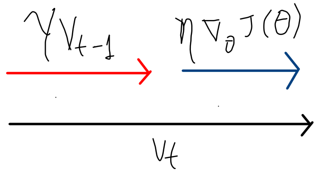
เพิ่ม Speed

1) Momentum เมื่อเราใช้ Mini-Batch (GD) จะเกิดกาสั่นของการลู่เข้าจุด Optimum(จุดต่ำสุด) ดังรูป
-> ทำให้ ต้องใช้จำนวน Epoch (รอบคำนวณ) มาก Momentum จึงเข้ามาแก้ไขปัญหานี้

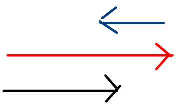


EWMA (Exponentially Weighted Average) เป็นการเฉลี่ยค่า gradient ใน epoch ก่อนหน้า

$$V_t = \gamma V_{t-1} + \eta \nabla_\theta J(\theta)$$



weight update (สีน้ำเงิน) ไปทางเดียวกัน จะทำให้เคลื่อนที่เร็ว



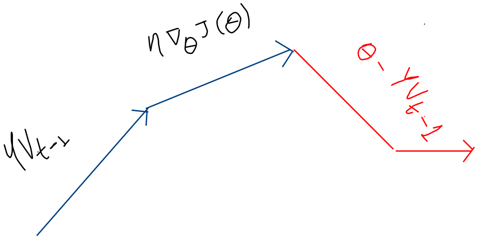
weight update (สีน้ำเงิน) ไปคนละทาง จะเคลื่อนที่น้อยลง

ข้อเสีย อาจข้ามจุด Optimum แท้จริงไป

2) Nesterov accelreated gradient

ช่วยแก้ข้อเสียของ Momentum

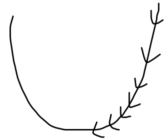
$$V_t = \gamma V_{t-1} + \eta \nabla_\theta J(\theta - \gamma V_{t-1})$$



จะช่วยควบคุมเส้นทางการเดินของ Momentum ไม่ให้ก้าวข้ามจุดต่ำสุดไป

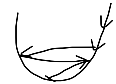
3) Adagrad

ช่วยปรับค่า Learning rate เมื่อเวลาผ่านไป
-> ปรับโดยการสเกลค่า Learning ด้วย Inverse
รากที่สอง ของผลรวมของ Gradient
ยกกำลังสองทุกตัวใน Epoch ที่ผ่านมาทั้งหมด



$\propto$ ถี่ขึ้นตามความชัน

ไม่ได้ดูที่รอบ Epoch นะ
เพราะอาจจะกระโดดข้ามได้



$$W = W - \frac{\alpha}{\sqrt{s+\epsilon}} \odot g$$

$$\theta_t = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot g_t$$

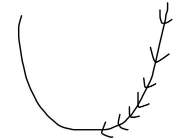
GD

GD

ข้อเสีย มีการสะสมค่าของ Gradient (GD) ในตัวหาร -> นานๆไป ค่า learning จะเข้าใกล้ 0
=> ทำให้ W ไม่ได้ต่อ ซึ่งอาจไม่ใช่จุดต่ำสุดจริงๆ

4) RMS Prop (แก้ข้อเสียของ Adagrad)

โดยเปลี่ยนจากผลรวมของ Gradient(GD) เป็น Exponentially Weighted Moving Average (ค่าเฉลี่ย) ของ GD แทน



$\propto$ **ถี่ขึ้นตามความชัน**

$$W = W - \frac{\alpha}{\sqrt{S+\epsilon}} \odot g$$

$$\theta_t = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \odot g_t$$

GD

Average

GD

Average

5) Adam (รวมเอาข้อดีของ RMS prop กับ Momentum)

มีการทำ bias corrections (V, S) ที่ช่วยให้ช่วงแรกมี bias น้อยลง
ไม่เหมือนกับ RMS prop และ Momentum ที่ช่วงแรกมี bias สูง

Momentum

$$V = \beta_1 V + (1-\beta_1) g$$

Momentum

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

RMS prop

$$S = \beta_2 S + (1-\beta_2) g^2$$

RMS

$$V_t = \beta_2 V_{t-1} + (1-\beta_2) g_t^2$$

แก้สมการใหม่ เพื่อไม่ให้
Bias เข้าใกล้ 0

$$\hat{V} = \frac{V}{1-\beta_1^t} \; ; \quad \hat{S} = \frac{S}{1-\beta_2^t}$$

สมการปรับ
Weight
สุดท้าย

$$W = W - \frac{\alpha}{\sqrt{\hat{S}} - \epsilon} \odot \hat{V}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_t} + \epsilon} \hat{m}_t$$