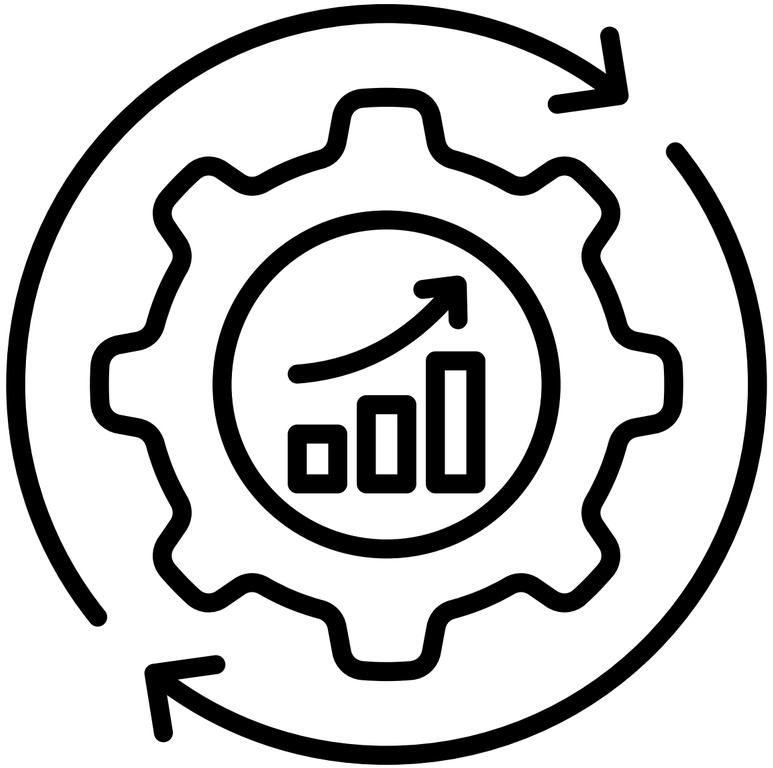
OPTIMIZER

ว30263

การเขียนโปรแกรมปัญญาประดิษฐ์

WHY WE NEED OPTIMIZERS

การสอนโมเดลในเชิงปฏิบัติ จึงมีไม่ใช่การแก้สมการโดยตรง แต่เป็นปัญหาที่อยู่ในกลุ่มที่เรียกว่า Optimizing problems โดยการมองว่า โมเดลเป้าหมาย เป็นเพียงแค่ กลุ่มของตัวแปร (parameters) ซึ่งจะมาประกอบกันเป็นสมการ และให้คำตอบอะไรบางอย่างออกมา ตามข้อมูลที่รับเข้ามา



WHY WE NEED OPTIMIZERS

การสอนโมเดล คือ การปรับแต่งตัวแปรต่างๆในโมเดล ซึ่งมักอาศัยการค่อยๆ ทดลองปรับค่าต่างๆทีละเล็กทีละน้อยซ้ำๆ ไปเรื่อยๆ เพื่อหาตัวแปรชุดที่ดีที่สุด

THE CONCEPT OF LOSS

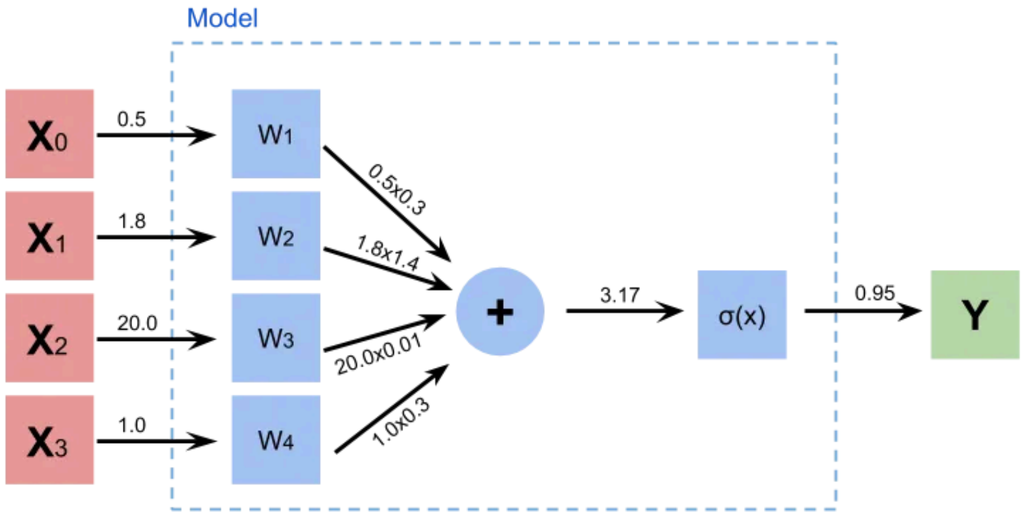
สิ่งหนึ่งที่ตามมาพร้อมๆกับ optimizer คือ คอนเซปของ “Loss” ซึ่งใช้ในการวัดผลว่า ขณะนี้ โมเดลทำหน้าที่ได้ดีแค่ไหน โดย Loss นี้ทำหน้าที่เป็นเป้าหมายของ optimizer โดย optimizer จะใช้เทคนิคต่างๆเพื่อลด loss ให้ได้มากที่สุด

THE CONCEPT OF LOSS

Loss เป็นการวัดผลการทำงานของโมเดลในระดับย่อยๆ สำหรับข้อมูลแต่ละชิ้น เช่น ข้อมูลชิ้นที่ 1 ตอบผิดไปจากความเป็นจริง 10 แต้ม, ข้อมูลชิ้นที่ 2 ตอบผิดไป 2 แต้ม แต่ Accuracy เป็นการวัดผลการทำงานของโมเดลในภาพรวม เช่น โดยเฉลี่ยจากข้อมูลทั้งหมดแล้ว โมเดลนี้มีตอบถูกทั้งหมด 80%

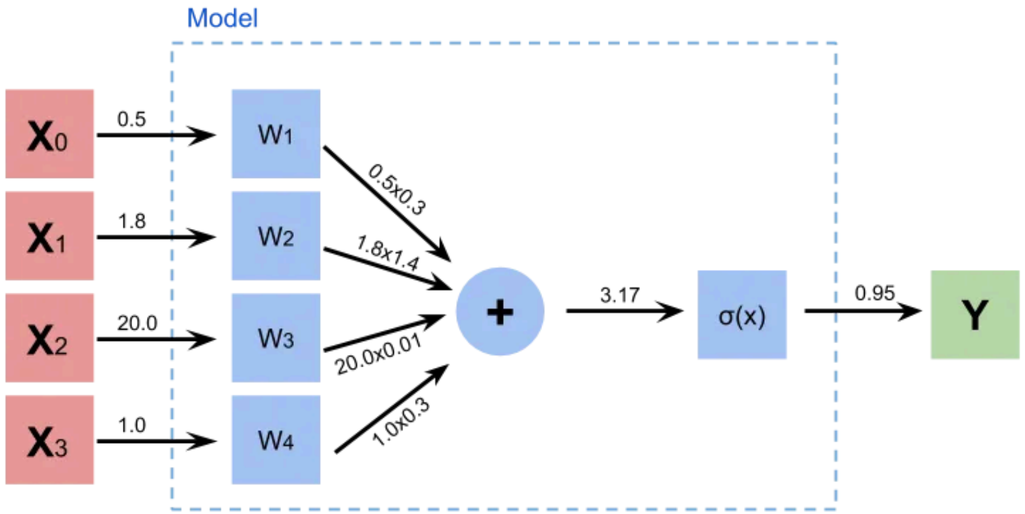
GRADIENT DESCENT

AI model เป็นเพียงแค่กลุ่มของตัวแปร ซึ่งตัวแปรเหล่านี้จะประกอบรวมกันเป็นสมการ ซึ่งจะคำนวนบางอย่างออกมาเป็นคำตอบตามที่ต้องการ



GRADIENT DESCENT

ตัวแปรแต่ละตัวในโมเดลต่างมีส่วนร่วมในการคำนวนผลลัพท์ของโมเดล ไม่ว่าจะโดยตรงหรือโดยอ้อม โดยเมื่อเปลี่ยนค่าของตัวแปรตัวใดตัวหนึ่งแน่นอนว่า จะทำให้ผลการทำนายจากโมเดลเปลี่ยนแปลง และการเปลี่ยนแปลงนี้ ก็ส่งผลไปเปลี่ยนแปลง loss อีกด้วย



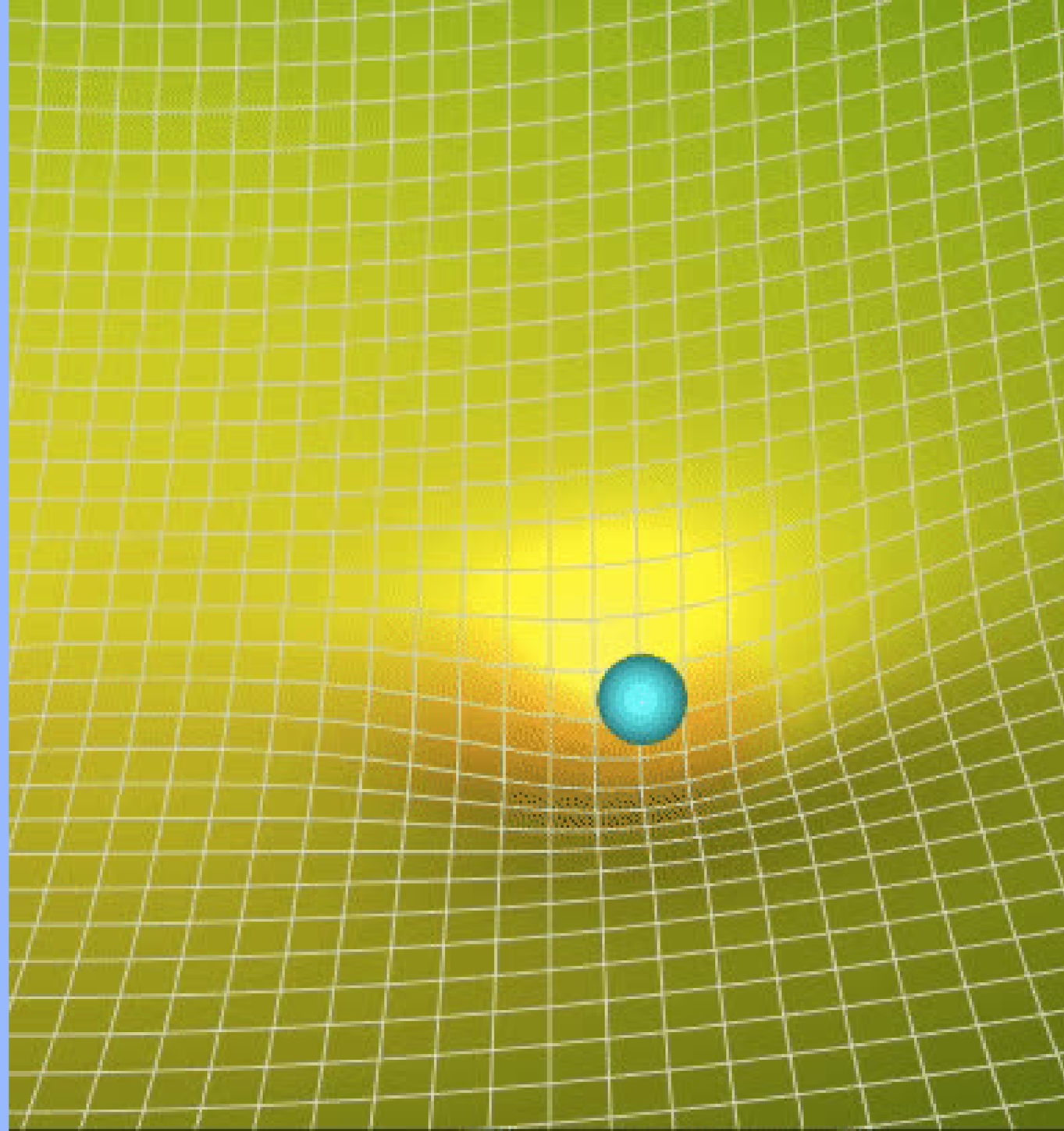
GRADIENT DESCENT

เทคนิคหนึ่งที่ดีกว่าการเดิมสุ่ม คือ การไต่ตามความชันของกราฟ (gradient) คล้ายๆกับการปีนเขา โดยใช้ความชันเป็นตัวนำทางว่าควรจะเดินต่อไปทิศทางใด

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$

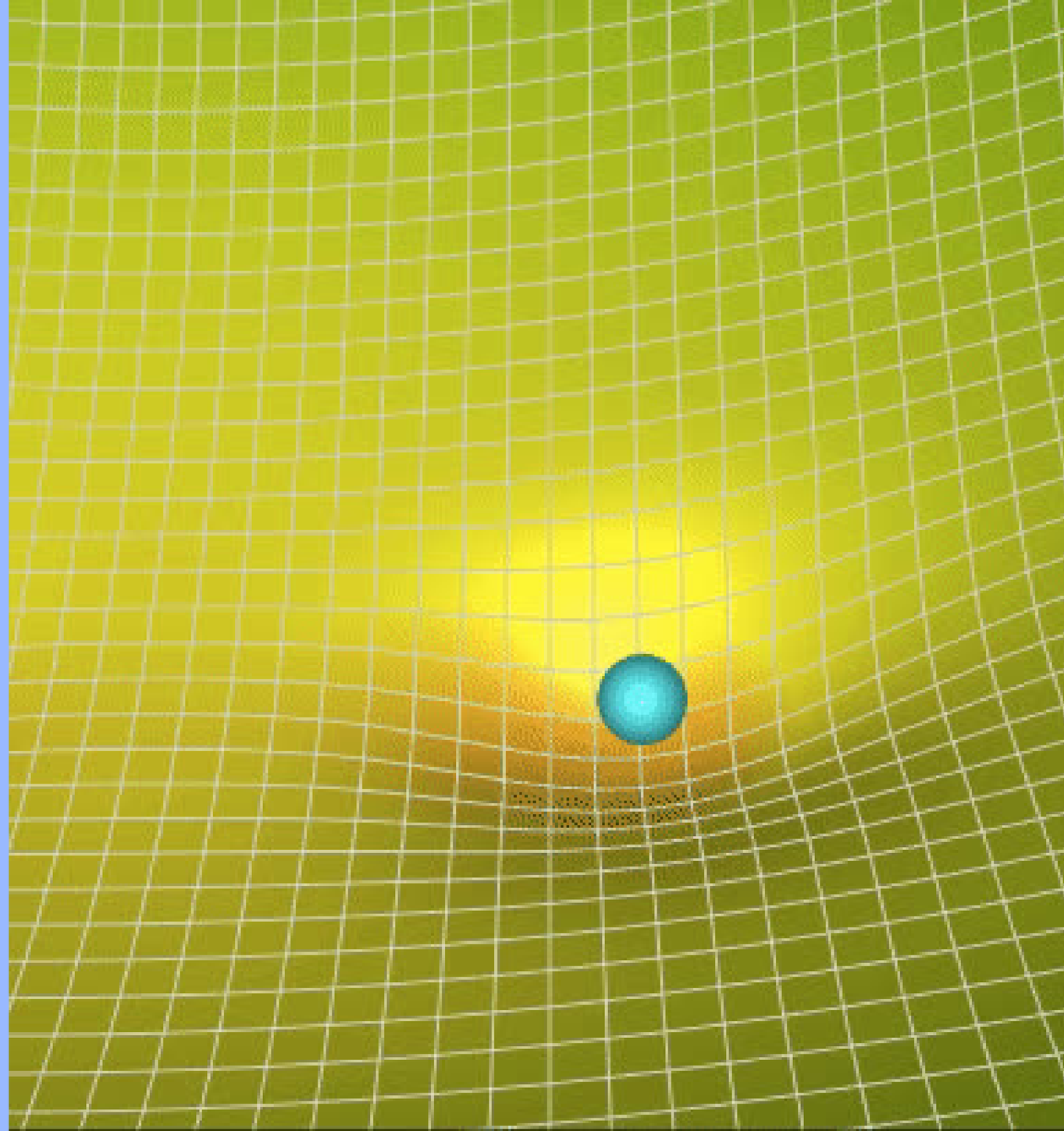
GRADIENT DESCENT

gradient ที่ถูกต้องและครบถ้วน คือ gradient ที่ได้ ต้องคำนวนจากข้อมูลๆ ทุกๆชิ้นมาเฉลี่ยรวมกัน แล้วจึงอัพเดต โมเดลหนึ่งครั้ง เรียกเทคนิคนี้ว่า Batch gradient descent



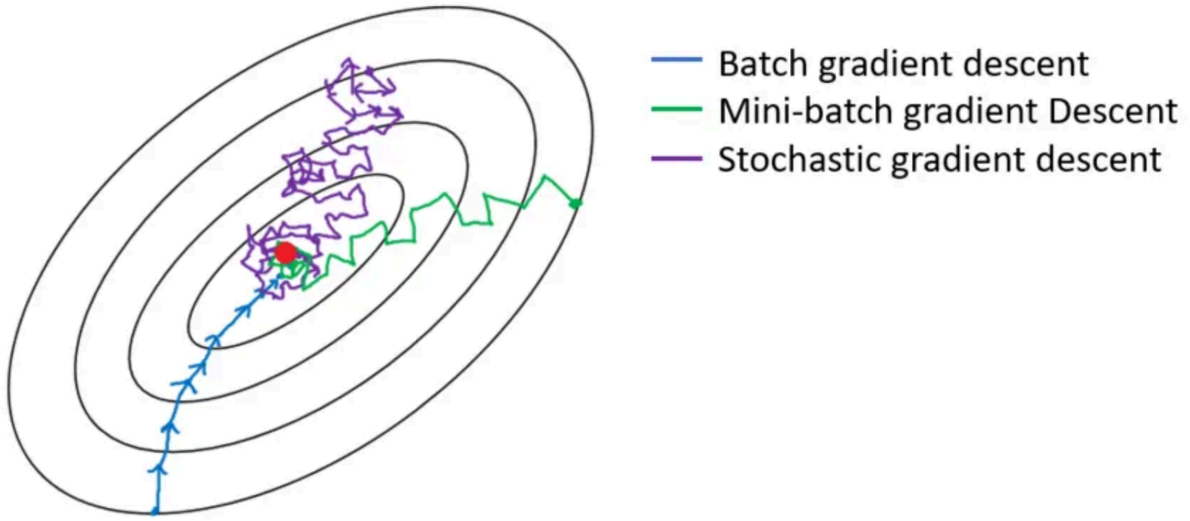
GRADIENT DESCENT

ในกรณีที่มีข้อมูลจำนวนมากๆๆๆ คือ ในขณะที่เทรน มักไม่สามารถนำข้อมูลทั้งหมดยัดลงใน RAM เพื่อคำนวน gradient พร้อมๆกันได้ทั้งหมดทีเดียว จำเป็นต้องคำนวนแค่เพียงบางส่วนก่อน



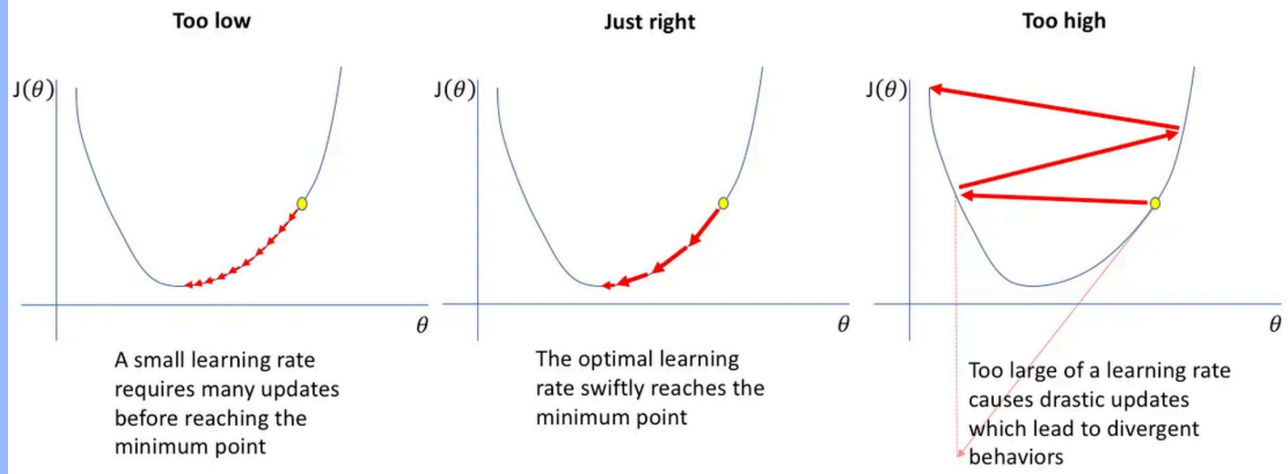
GRADIENT DESCENT

batch และ stochastic โดยไม่ใช้ทั้ง dataset อย่างใน batch และไม่ได้ใช้แค่ข้อมูล 1 ก้อนอย่างใน stochastic แต่ใช้ข้อมูลเป็น N ก้อน เรียกว่า mini-batch หรือ เรียกเทคนิคนี้ว่า mini-batch gradient descent โดยขนาด mini-batch เป็นสิ่งที่สามารถปรับจูนได้ตามต้องการ โดยมักพิจารณาจากขนาด GPU RAM ที่มี ยิ่งใช้ batch ขนาดใหญ่ ยิ่งจะทำให้การสะเปะสะปะลดลง ส่งผลให้ได้โมเดลที่ดีมากขึ้น แต่ก็ต้องลงทุนกับ GPU มากขึ้นเช่นกัน



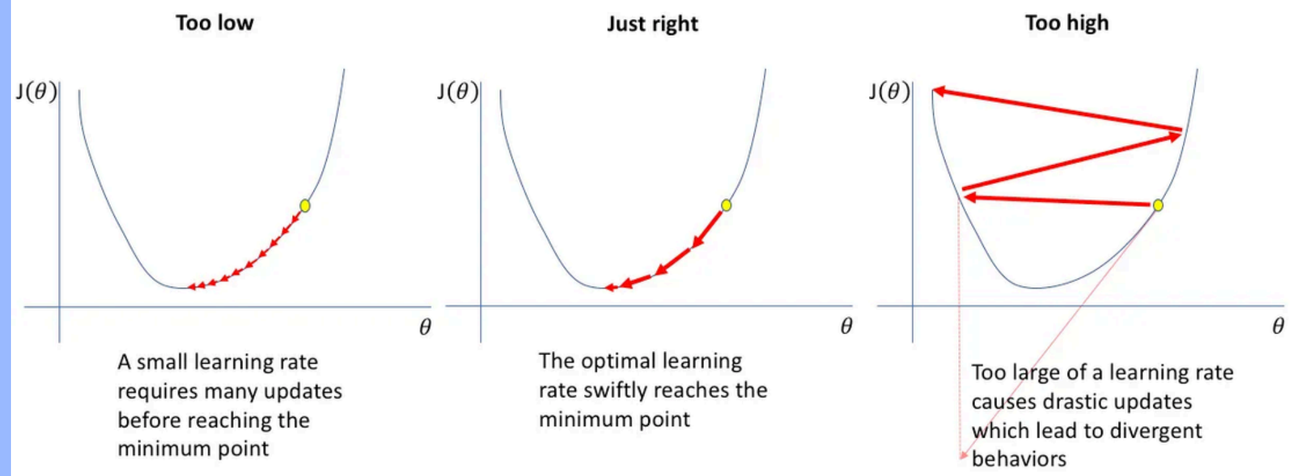
LEARNING RATE

Learning rate เป็น hyperparameter ที่ใช้ในการกำหนดขนาดของการไต่เขา (ใช้ gradient เพื่อกำหนดทิศทาง) จากการทดลองพบว่า Learning rate เป็นส่วนสำคัญอันดับต้นๆในการเทรน



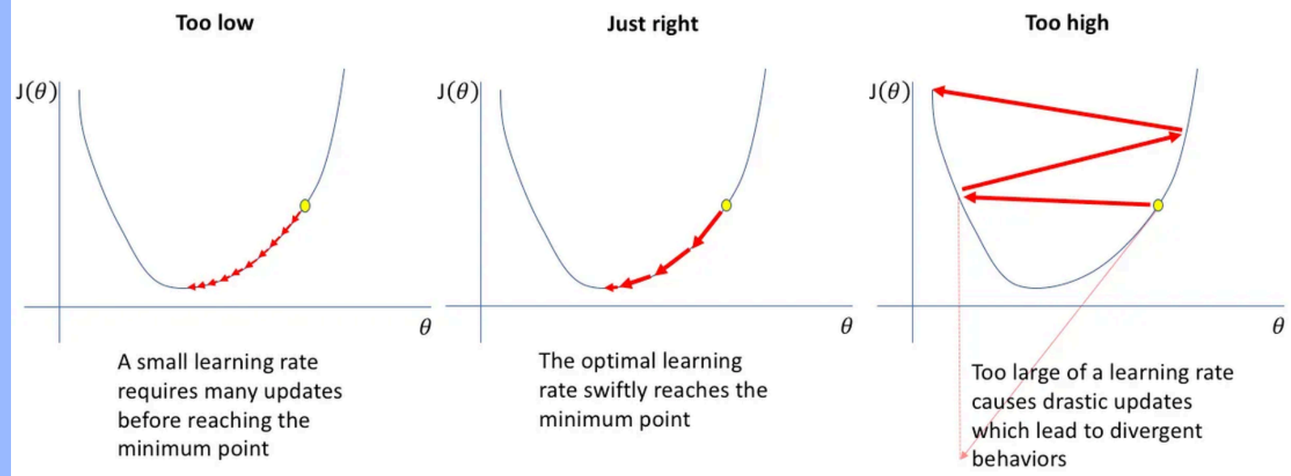
LEARNING RATE

learning rate จึงเป็นขั้นตอนสำคัญในการเทรนโมเดล ทั้งนี้ learning rate ที่เหมาะสม ขึ้นอยู่กับทั้ง loss function , โมเดล และ ข้อมูลที่ใช้สอน จึงเป็นไปได้อยากที่จะมีค่าใดค่าหนึ่งที่สามารถใช้ได้อย่างเหมาะสมกับทุกๆกรณี แต่โดยทั่วไปแล้ว learning rate จะตั้งอยู่ที่ประมาณ 0.1-0.01



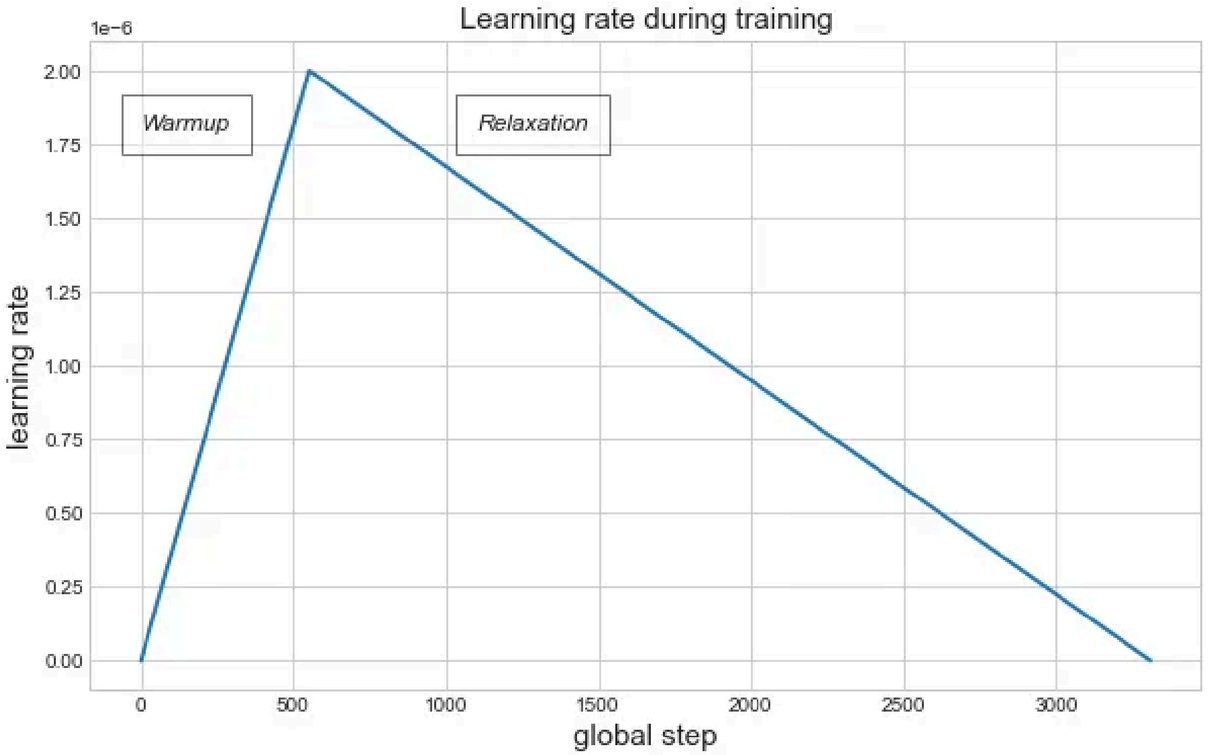
LEARNING RATE

Learning rate scheduling โดยมีที่มาจากปัญหาว่า learning rate เป็นส่วนสำคัญในการเทรนโมเดล แต่ในแต่ละช่วงของการสอนโมเดลต้องการรูปแบบการเรียนรู้ไม่เหมือนกัน โดยช่วงแรกโมเดลต้องการเรียนรู้แบบกว้างๆ เพื่อเข้าใจภาพรวมของ loss function แต่ในช่วงท้ายๆ โมเดลต้องการเรียนรู้แบบละเอียดๆ เพื่อเก็บรายละเอียด ดังนั้น Learning rate จึงไม่ควรเป็นเพียงแค่ค่าคงที่ตลอดการเทรน



LEARNING RATE

Learning rate scheduling มักจะกำหนดให้ learning rate ในช่วงแรกๆมีค่าค่อยๆเพิ่มขึ้น เรียกว่า warmup phase จากนั้นจึงค่อยๆลดลง เรียกว่า relaxing phase ทั้งนี้อัตราการเพิ่ม-ลดสามารปรับเปลี่ยนตามต้องการ แต่ทั่วไปมักให้ warmup ด้วย linear function จากนั้น relaxing ด้วย linear decay หรือ exponential decay



COST FUNCTION

cost function คือ ฟังก์ชั่นที่มีหน้าที่ช่วยในการ วัดค่าความแม่นยำของ hypothesis function ซึ่งมันคือการหาค่าเฉลี่ยของ ผลต่าง ของค่าจริงกับค่าที่คาดเดาออกมา สูตรก็ค่อนข้างตายตัว

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

COST FUNCTION

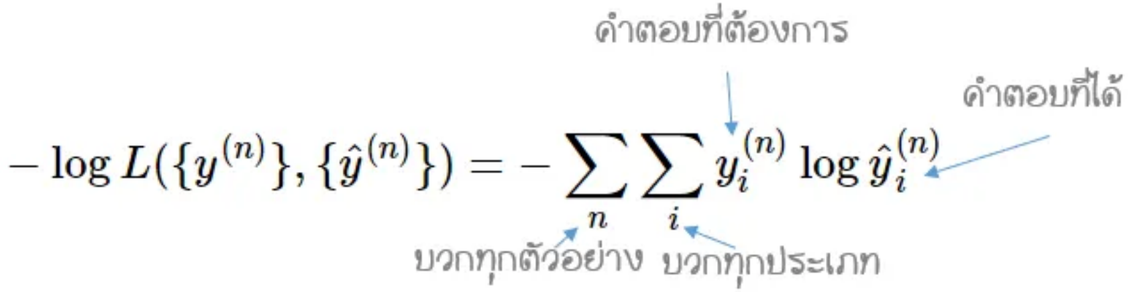
Squared error function หรือ Mean squared error พอเป็นสองชื่อนี้หลายๆ มันก็คือการหา Error ที่เกิดขึ้นว่ามีมากน้อยแค่ไหน เนื่องจากการคาดเดาผลลัพท์ของ Model ใน Machine learning นั้นไม่ควรจะเป็นผลลัพท์ที่แม่นยำ 100% หรือเปอร์เซนต์ต่ำจนเกินไป

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

Overfitting หรือ Underfitting

CROSS-ENTROPY LOSS

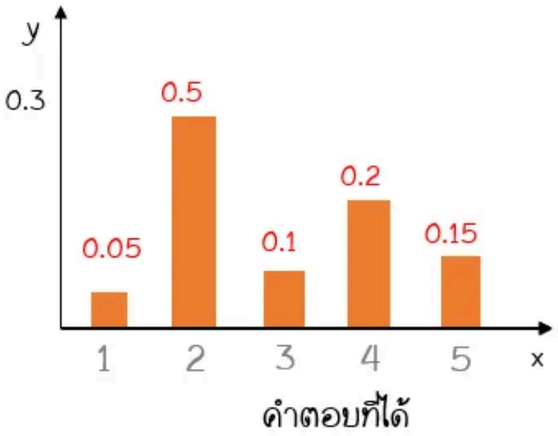
Loss Function ตัวนี้ใช้กับงานแยกประเภท ที่มีจำนวนประเภทแน่นอน เน็ตเวิร์กจะสามารถให้ผลลัพธ์เลขจำนวนเต็ม 1, 2, 3, 4, 5 เท่านั้น Loss Function ตัวนี้มาจากการเปรียบเทียบ Distribution



CROSS-ENTROPY LOSS

หากนำภาพแมวไปให้คน 100 คนบอกว่าเป็นภาพอะไร คนทั้งร้อยคนก็จะบอกว่าเป็นภาพแมว ฉะนั้นเน็ตเวิร์กของเราก็ควรจะบอกว่าเป็นภาพแมวด้วยความน่าจะเป็นเท่ากับ 1 หรือตอบแมว 100%

VS.



L1 และ L2 LOSS FUNCTION

Loss Function ประเภทนี้เหมาะกับงานที่ให้ผลลัพธ์เป็นจำนวนจริง เช่น ทำนายราคา ทำนายค่าต่างๆที่เราต้องการโดยไม่มีค่าจำกัด และสามารถมีทศนิยมได้

$$L2LossFunction = \sum_{i=1}^{n}(y_{true} - y_{predicted})^2$$

$$L1LossFunction = \sum_{i=1}^{n}|y_{true} - y_{predicted}|$$

L1 และ L2 LOSS FUNCTION

หากเน็ตเวิร์กยิ่งให้ผลลัพธ์ใกล้เคียงกับค่าที่ต้องการเท่าไหร่ ก็จะยิ่งมีผลต่างน้อย และก็จะยิ่งส่งผลให้มีค่า Loss น้อยลงเท่านั้นค่ะ และก็นำค่า Loss ของแต่ละตัวอย่างมาหาค่าเฉลี่ยอีกที ก็จะได้ค่า Loss รวมของทั้งหมด

$$L2LossFunction = \sum_{i=1}^{n} (y_{true} - y_{predicted})^2$$

$$L1LossFunction = \sum_{i=1}^{n} |y_{true} - y_{predicted}|$$

L1 และ L2 LOSS FUNCTION

Loss Function ประเภทนี้ไปใช้กับงานแยกประเภทบ้าง นั่นก็เพราะจะให้คำตอบเพี้ยน ยกตัวอย่างเช่น x อยู่ในประเภทที่ 2 หากเน็ตเวิร์กตอบว่าเป็นประเภทที่ 3 ก็จะมีค่า Loss น้อยกว่า หากเน็ตเวิร์กตอบว่าเป็นประเภทที่ 5

$$(2-3)^2 < (2-5)^2$$

L1 และ L2 LOSS FUNCTION

เมื่อพิจารณาความหมายในบริบทนี้มันไม่ใช่ หากตอบไม่ตรงประเภทที่ต้องการก็คือผิดและเสียหายเท่ากัน ตัวอย่างเช่นมีข้อมูลภาพแมว(เป็นประเภทที่ 2) ไม่ว่าจะตอบว่าเป็นภาพนก(เป็นประเภทที่ 3)หรือภาพม้า(เป็นประเภทที่ 5) ก็คือผิดเหมือนกัน ไม่มีว่าอันไหนผิดมากกว่าหรือน้อยกว่า

$$(2-3)^2 < (2-5)^2$$

L1 และ L2 LOSS FUNCTION

Loss Function เป็นเหมือนค่าวัดที่เป็นเป้าหมายให้ Network พัฒนาไปตามที่เราต้องการ หากเราเลือกใช้อย่างไม่ถูกต้อง ก็จะทำให้ได้คำตอบที่เพี้ยนได้

$$(2-3)^2 < (2-5)^2$$

งาน

ให้นักเรียนเปรียบเทียบ**batch / mini-batch / stochastic** โดยเหมาะสมที่สุดจาก **mode** ที่ดีที่สุดของสัปดาห์ที่แล้ว