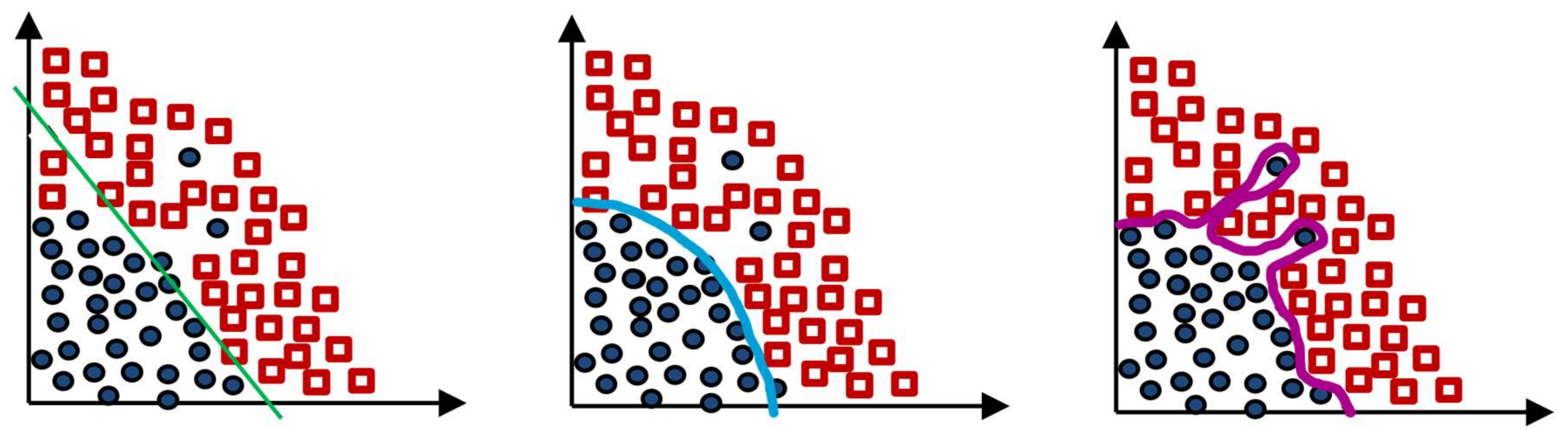
IMPROVE DNN

Dr.Varodom Toochinda

Dept. of Mechanical Engineering,

Kasetsart University

BIAS AND VARIANCE



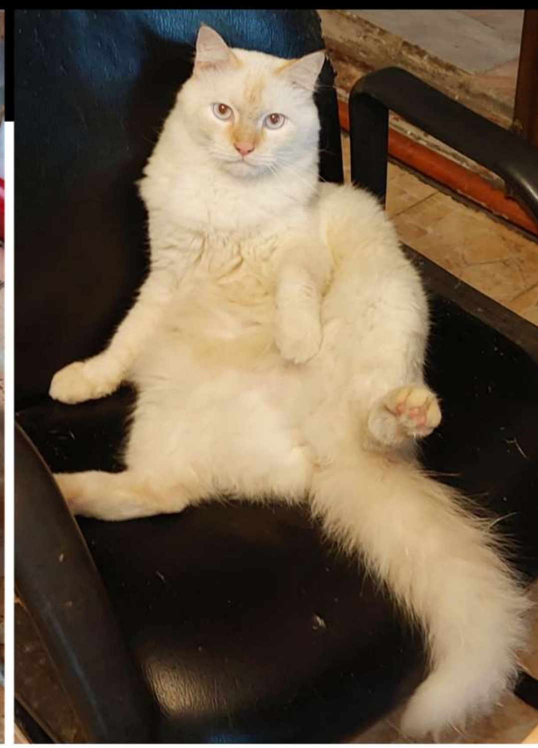
(a) High bias (underfitting)

(b) About right

(c) High variance (overfitting)

CAT/DOG CLASSIFICATION

y = 0



y = 1

- **A.** ได้ค่าผิดพลาดจากชุดข้อมูลฝึกเท่ากับ **0.5%** และค่าผิดพลาดจากชุดข้อมูลพัฒนาเท่ากับ **1%** จัดได้ว่าเป็นโมเดลที่ดี คือได้ทั้งค่าเอนเอียงและความแปรปรวนต่ำ
- **B.** ได้ค่าผิดพลาดจากชุดข้อมูลฝึกเท่ากับ **1%** และค่าผิดพลาดจากชุดข้อมูลพัฒนาเท่ากับ **12%** คือกรณีความแปรปรวนสูง จะเห็นว่าโมเดลมีความแม่นยำสูงกับข้อมูลฝึก แต่ทำงานได้ไม่ดีกับข้อมูลพัฒนาเนื่องจากการฟิตเกิน
- **C.** ได้ค่าผิดพลาดจากชุดข้อมูลฝึกเท่ากับ **20%** และค่าผิดพลาดจากชุดข้อมูลพัฒนาเท่ากับ **21%** คือกรณีค่าเอนเอียงสูง โมเดลทำงานได้ไม่ดีตั้งแต่กับชุดข้อมูลการฝึก และก็ได้ผลใกล้เคียงกันกับชุดข้อมูลพัฒนา คือมีการฟิตต่ำไป
- **D.** ได้ค่าผิดพลาดจากชุดข้อมูลฝึกเท่ากับ **20%** และค่าผิดพลาดจากชุดข้อมูลพัฒนาเท่ากับ **30%** คือกรณีทั้งค่าเอนเอียงและความแปรปรวนสูง เป็นโมเดลที่ด้อยที่สุดสำหรับทั้ง **4** กรณีที่ยกตัวอย่างมา

สมมุติว่ามนุษย์สามารถจำแนกภาพได้โดยมีความผิดพลาดเป็นศูนย์

หากพบว่าโมเดลมีค่าเอนเอียงสูง (ต้องการปรับปรุงสมรรถนะต่อข้อมูลฝึก)

- เพิ่มขนาดของ DNN
- เพิ่มจำนวนรอบการฝึก หรือทดลองเปลี่ยนตัวหาค่าเหมาะที่สุด
- อาจทดลองเปลี่ยนสถาปัตยกรรมของโครงข่ายประสาทเทียม

See bias_variance.ipynb

หากพบว่าโมเดลมีความแปรปรวนสูง (ต้องการปรับปรุงสมรรถนะต่อข้อมูลพัฒนา)

- เพิ่มจำนวนข้อมูล
- ลดการฟิตเกินโดยวิธีเช่นเรกูลาร์ไรเซชันหรือดรอปเอาต์
- อาจทดลองเปลี่ยนสถาปัตยกรรมของโครงข่ายประสาทเทียม

REGULARIZATION

Logistic regression

- **L1** $$J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \|w\|_2^2$$

- **L2** $$J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \|w\|_1$$

REGULARIZATION

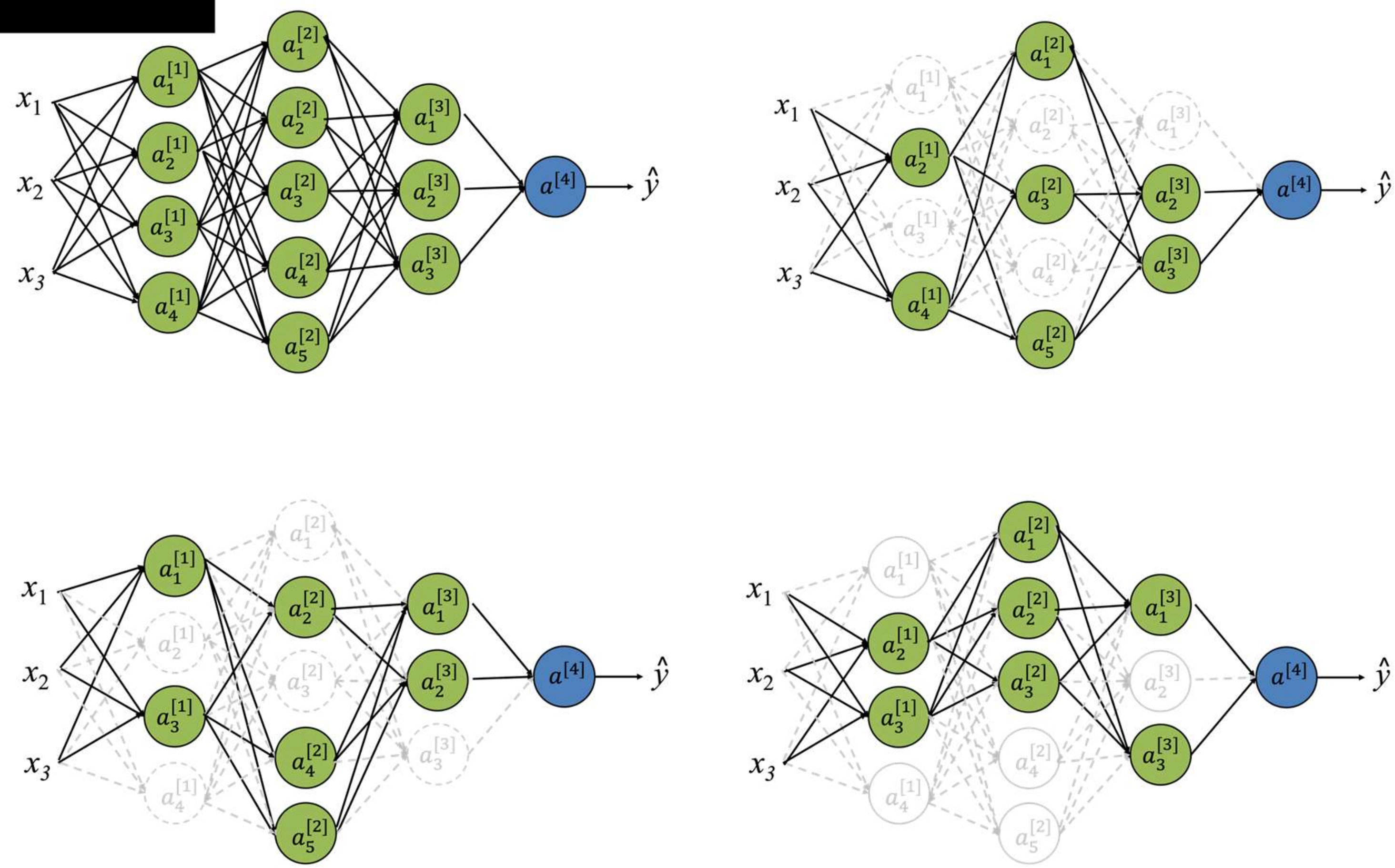
DNN

- L2 $$J(W^{[1]}, b^{[1]}, \ldots, W^{[L]}, b^{[L]}) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \sum_{l=1}^{L} \|W^{[l]}\|_F^2$$

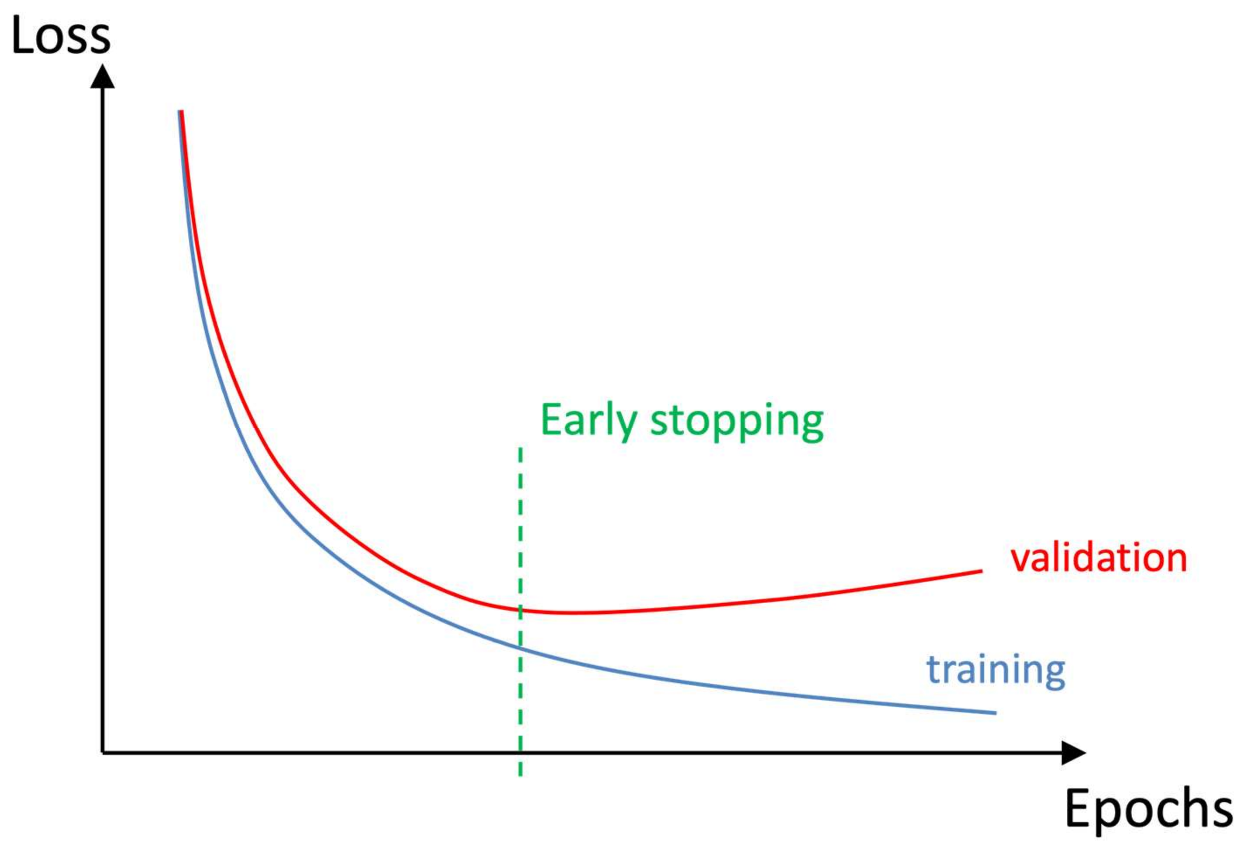
Frobenius norm ➡ $$\|W^{[l]}\|_F^2 = \sum_{i=1}^{n^{[l]}} \sum_{j=1}^{n^{[l-1]}} \left(w_{i,j}^{[l]}\right)^2$$

DROPOUT

original DNN



EARLY STOPPING

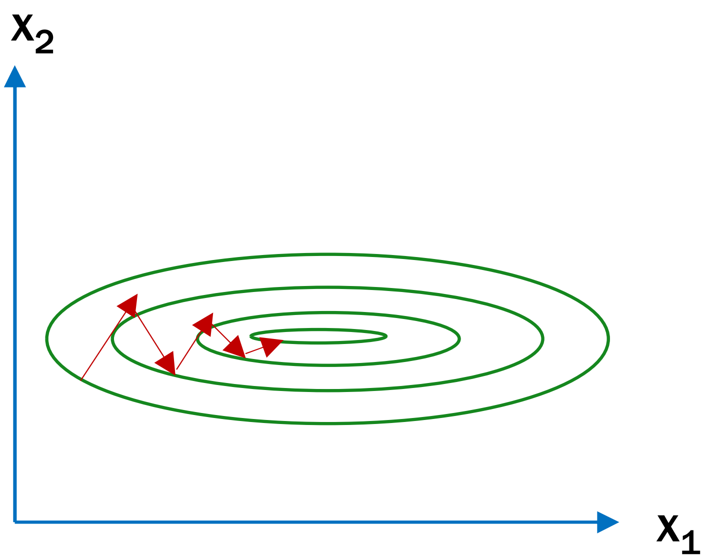


NOTEBOOK DEMO

- regularization.ipynb
- dropout.ipynb
- early_stopping.ipynb

IMPROVE MODEL TRAINING

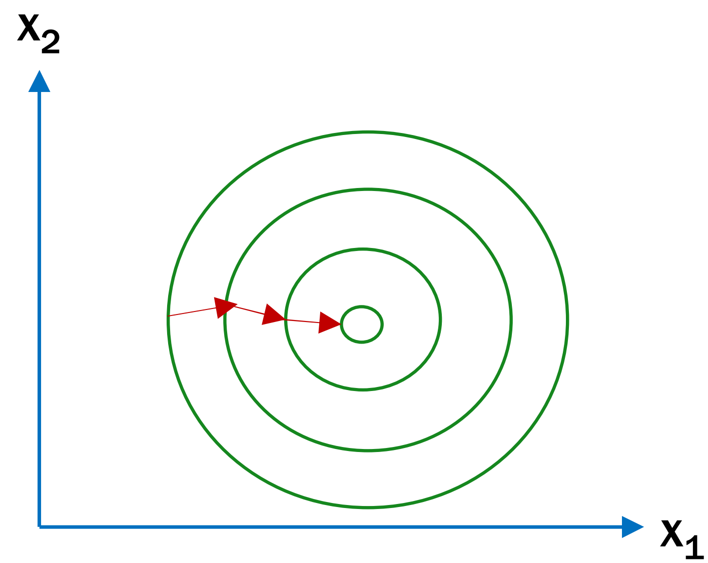
NORMALIZING INPUTS



$$\tilde{x} = \frac{x - \mu}{\sigma}$$

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)^2$$

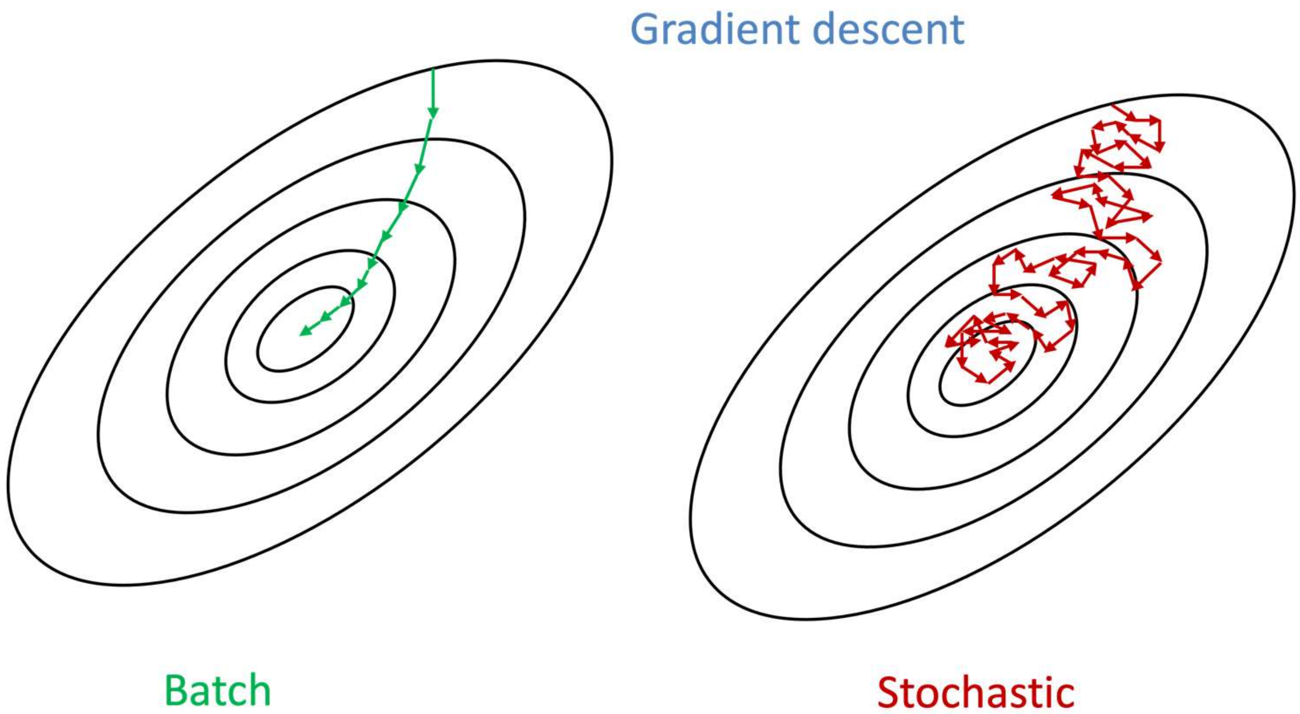


WEIGHT INITIALIZATION

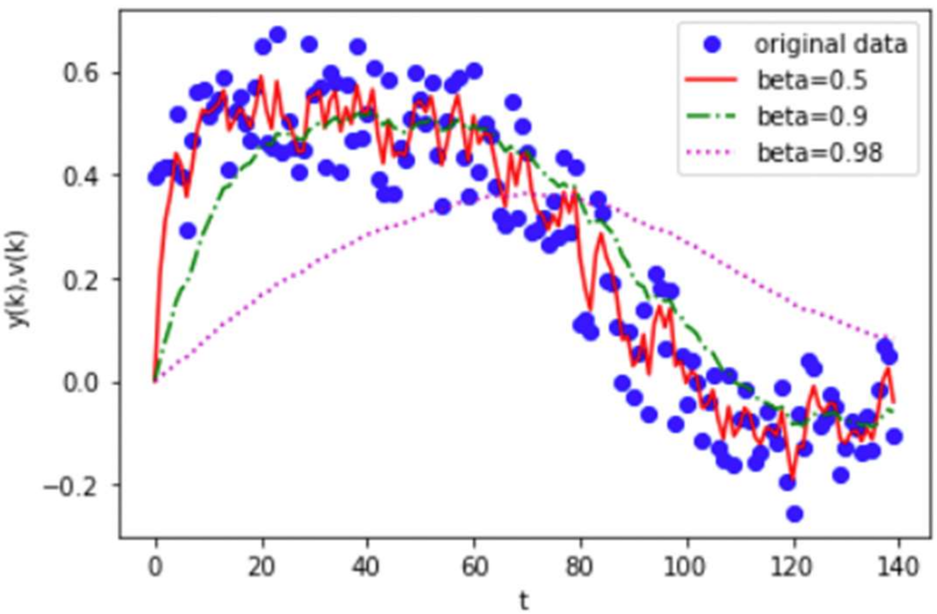
- W = 0 or W = constant results in symmetry. So must use random values
- Must not be too large nor too small.
- see normalize_input_and_initialization.ipynb

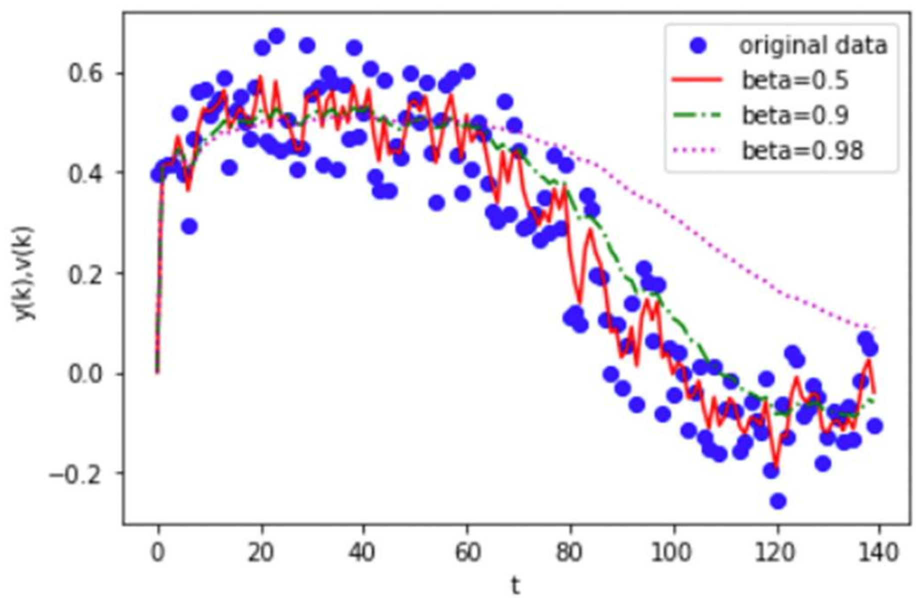
OPTIMIZERS

GRADIENT DESCENT



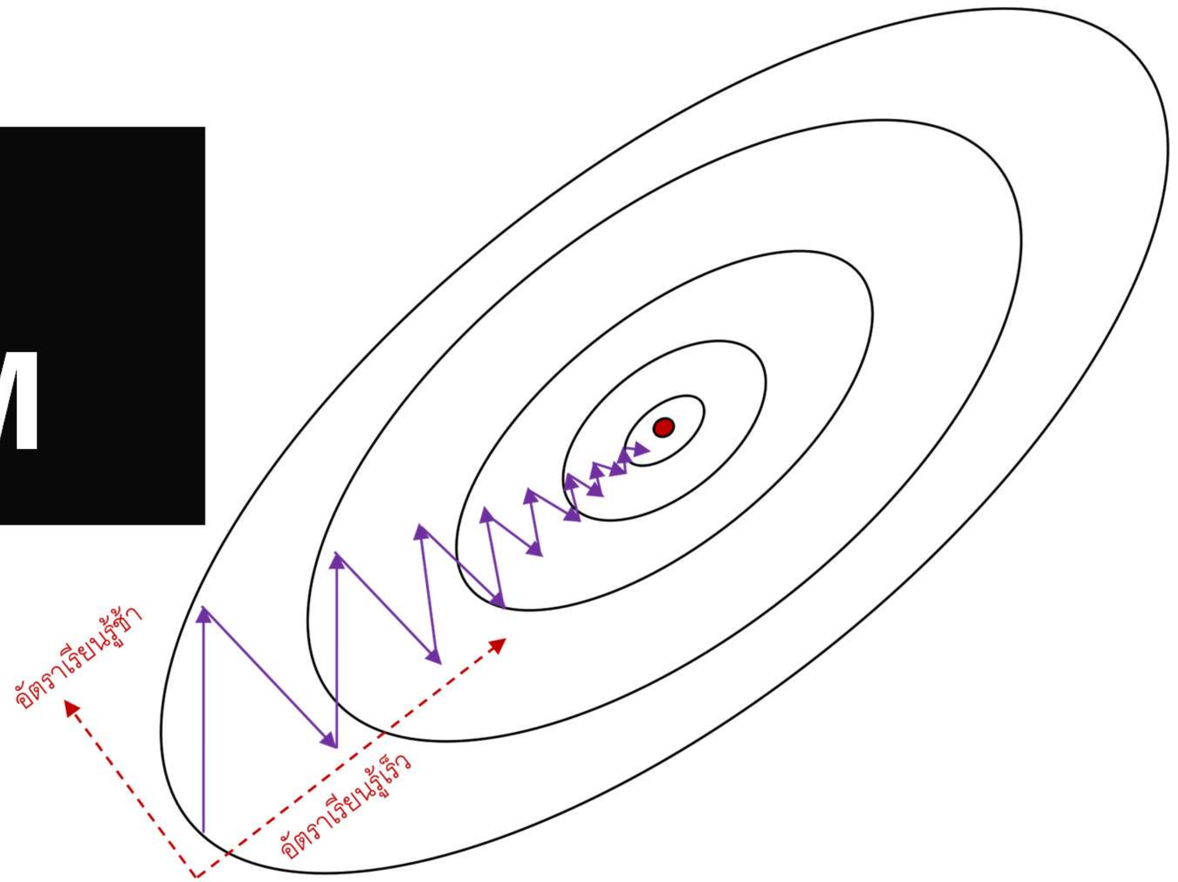
EXPONENTIALLY WEIGHTED AVERAGES





see ewa.ipynb

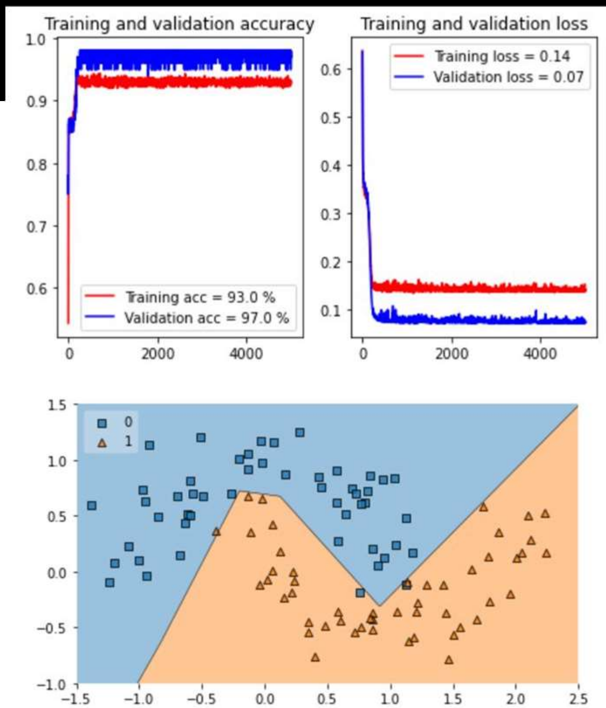
GD WITH MOMENTUM



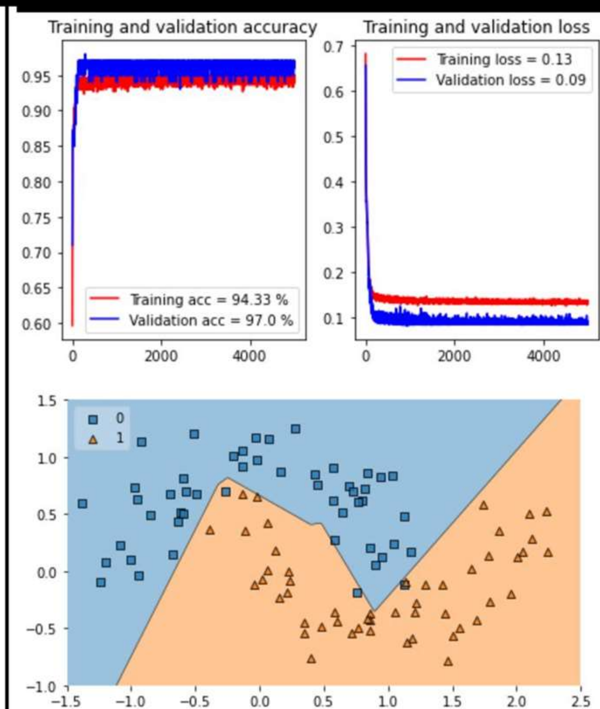
OTHER OPTIMIZERS

- RMSProp
- Adam
- see optimizers.ipynb

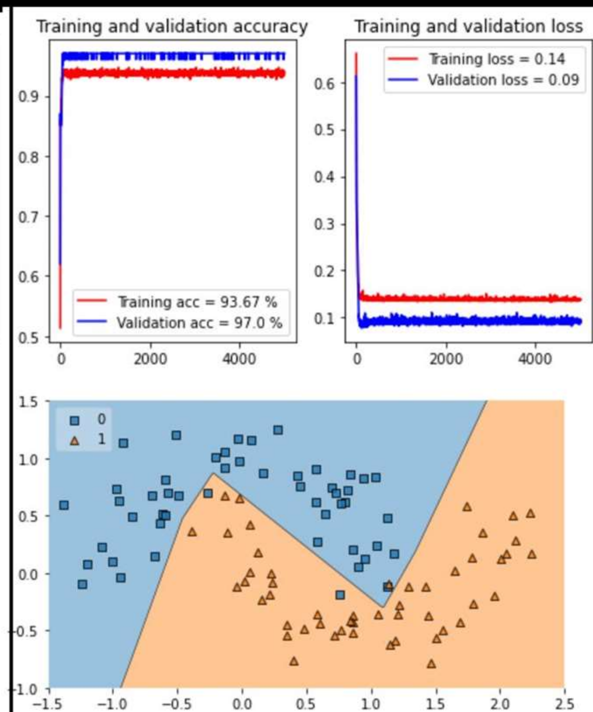
COMPARISION



momentum



RMSprop



Adam

LEARNING RATE DECAY

$$\alpha = \frac{1}{1 + decay\_rate * epoch\_no}\alpha_0$$

$$\alpha = 0.95^{epoch\_no}\alpha_0$$

$$\alpha = \frac{k}{\sqrt{epoch\_no}}\alpha_0$$

BATCH NORMALIZATION

see batch_normalization.ipynb

$$\mu = \frac{1}{m} \sum_{i=1}^{m} z^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} (z^{(i)} - \mu)^2$$

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

normalized

$$\boxed{\tilde{z}^{(i)} = \gamma z_{norm}^{(i)} + \beta}$$

learnable parameters