02.1-DeepLearning-General Guidance

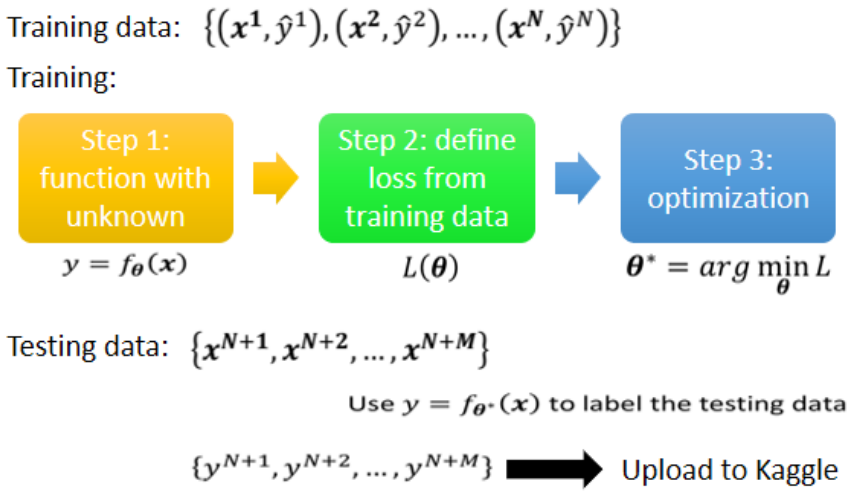
1. Framework of ML

1.1 訓練資料與測試資料

Training data: $\{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \dots, (x^N, \hat{y}^N)\}$

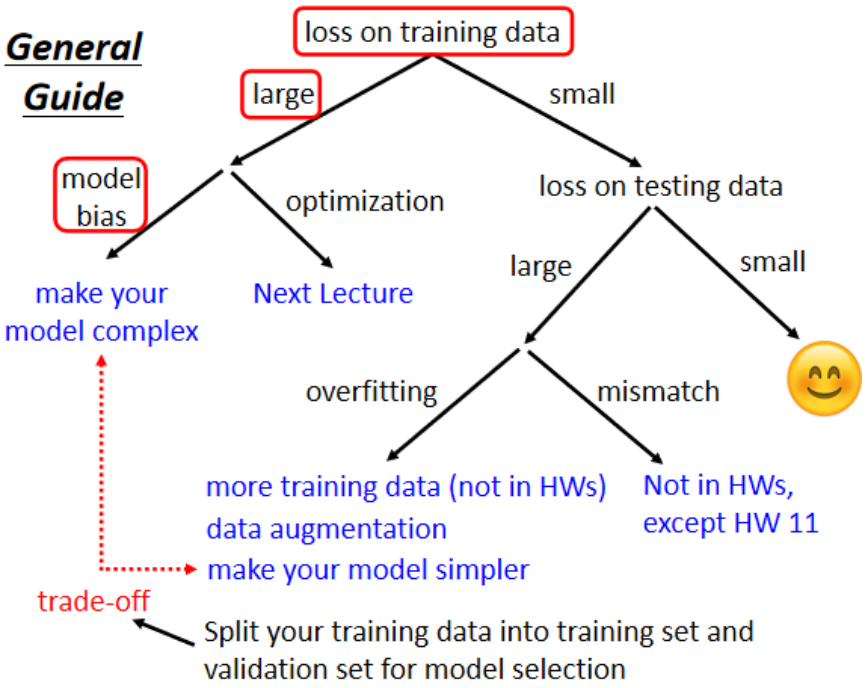
Testing data: $\{x^{N+1}, x^{N+2}, \dots, x^{N+M}\}$

1.2 訓練的過程



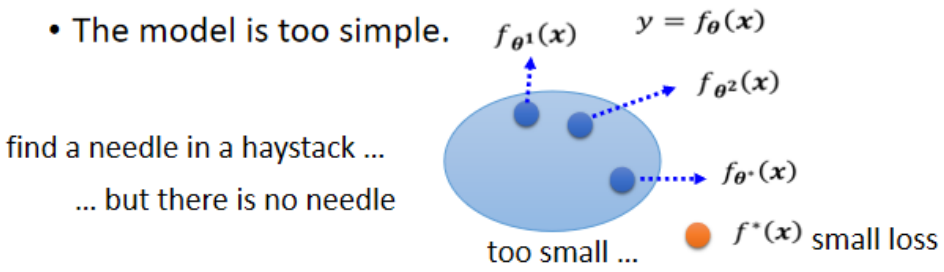
1. 寫出一個有未知參數的 function，參數用 θ 來表示
2. 確定損失函數，判斷 function 的參數 θ 好不好
3. optimization，得到使損失函數最小的參數 θ^*

2. General Guide



2.1 訓練資料上的 Loss

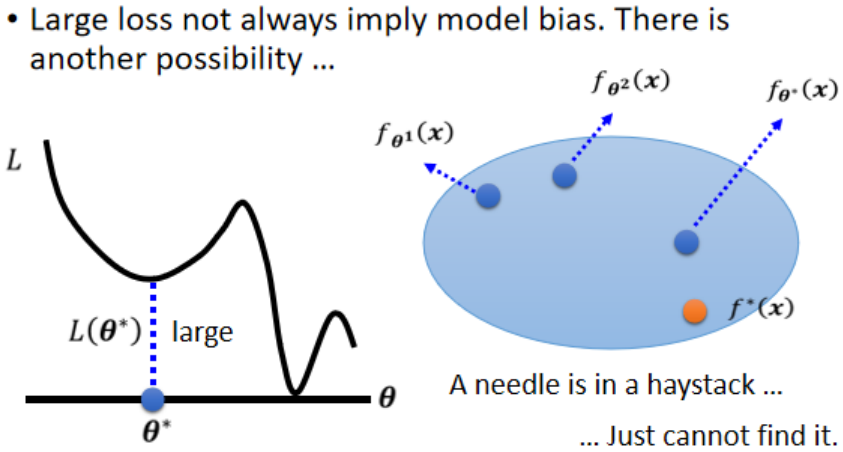
2.1.1 Model Bias



所有的 function 集合起來得到一個 function 的 set。但是這個 function 的 set 太小了，沒有包含任何一個 function 可以讓 loss 變低 ⇒ **可以讓 loss 變低的 function 不在 model 可以描述的範圍內**

⇒ **解決方法：重新設計一個 Model**，一個更複雜的、更有彈性的、有未知參數的、需要更多 features 的 function

2.1.2 Optimization



可能會卡在 local minima（局部極小值/鞍點）的地方，沒有辦法找到真的可以讓 loss 很低的參數

2.1.3 如何區分兩種情況？

- Start from shallower networks (or other models), which are easier to train.

  看到一個從來沒有做過的問題，可以先跑一些比較小、比較淺的 network，或甚至用一些不是 deep learning 的方法 ⇒ **比較容易做 optimize，較不會有 optimization 失敗的問題**

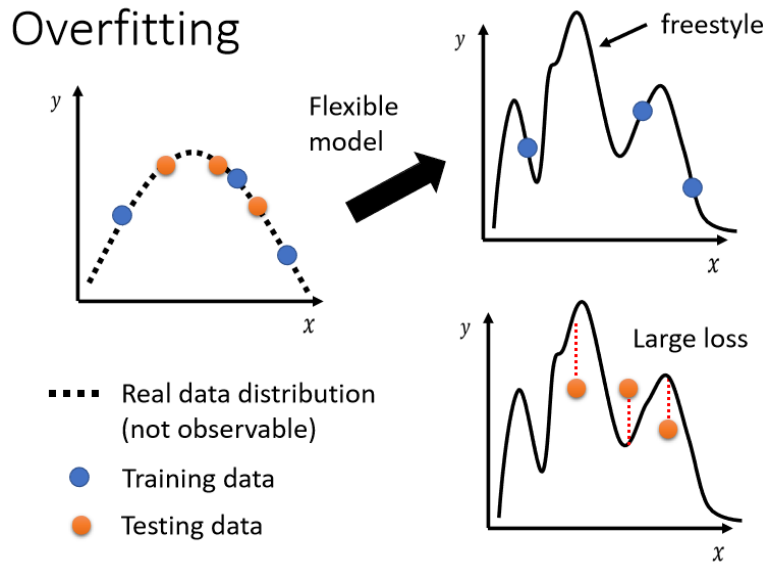
- If deeper networks do not obtain smaller loss on training data, then there is optimization issue.

如果發現深的 model 跟淺的 model 比起來，**深的 model 明明彈性比較大**，但 **loss 卻沒有辦法比淺的 model 壓得更低**，那就代表 **optimization 有問題**

2.2 測試資料上的 Loss

2.2.1 Overfitting

training 的 loss 小，testing 的 loss大，有可能是 overfitting。如果你的 model 它的**自由度很大**的話，它會**產生非常奇怪的曲線**，導致訓練集上的結果好，但是測試集上的 loss 很大



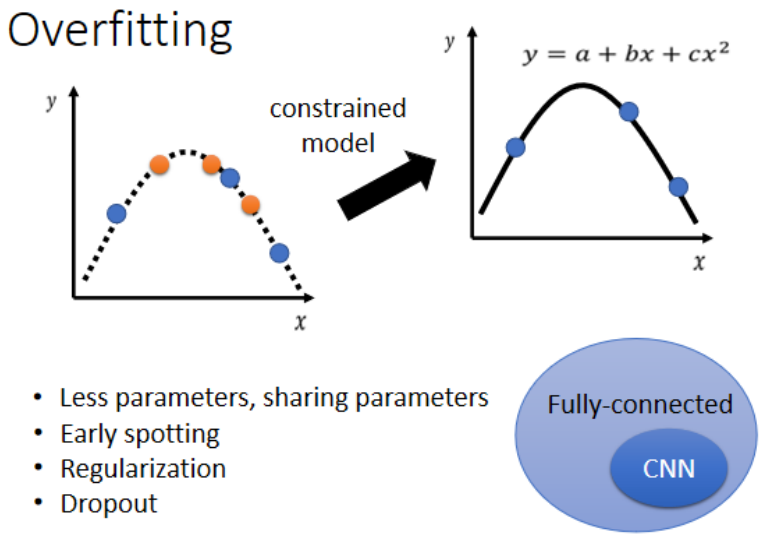
解決：

1. 增加訓練集

雖然你的 model 它的彈性可能很大，但是因為數據樣本非常非常的多，它就可以限制住

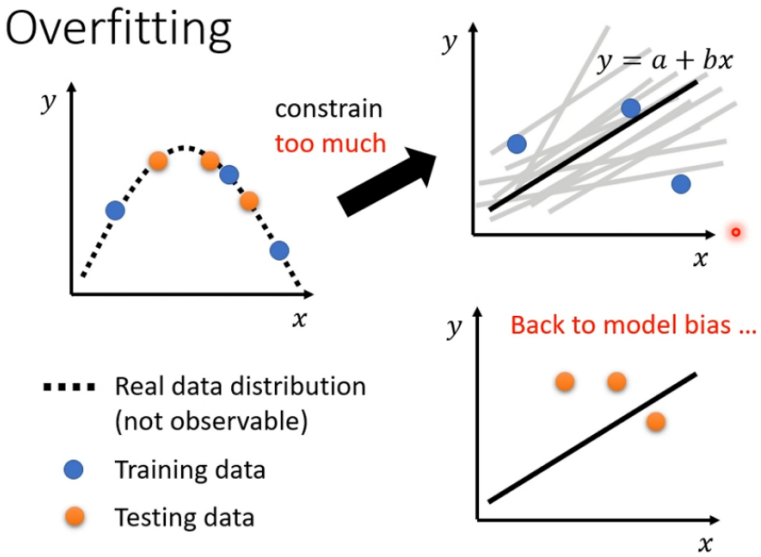
Data Augmentation：用一些對於問題的理解，從已有的數據中創造出新的數據（注意合理性）

2. 限制模型，使之不要有那麼大的彈性

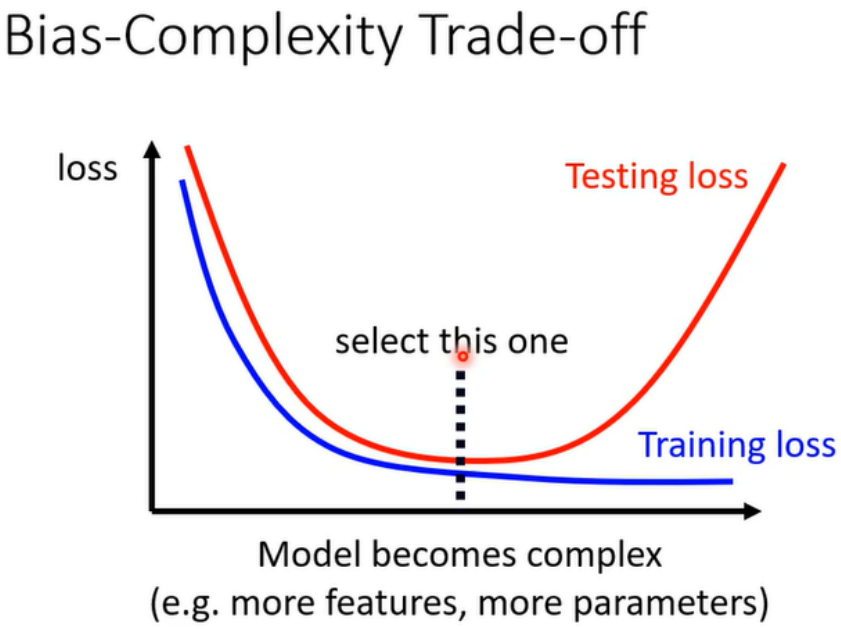


- 給**比較少的參數（比如神經元的數目）；模型共用參數 (**03-CNN**)**
- 使用**比較少的 features**
- Early Stopping
- Regularization
- Dropout

2.3 不要加過大的彈性 ⇒ Model bias



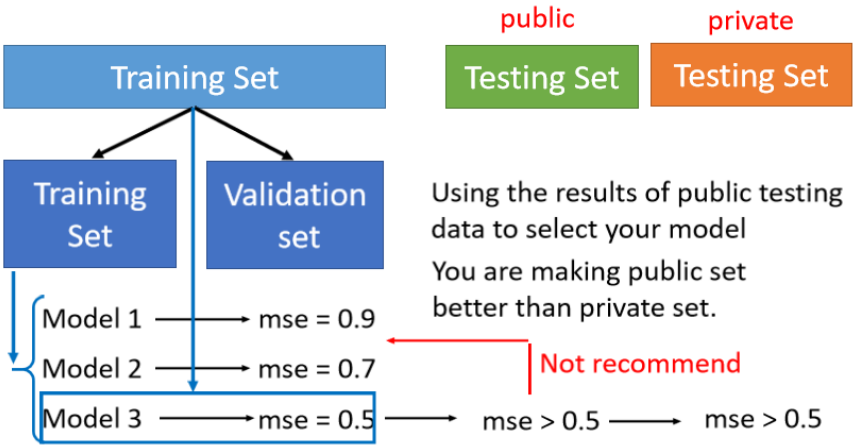
2.4 Bias-Complexity Trade-off



所謂比較複雜，是它可以**包含的 function 比較多，它的參數比較多，這個就是一個比較複雜的 model**

隨著 model 越來越複雜，**training 的 loss 可以越來越低，**當 model 越來越複雜的時候，剛開始 **testing 的 loss 會跟著下降**；但是當複雜的程度超過某一個程度以後，**testing 的 loss 就突然暴增**

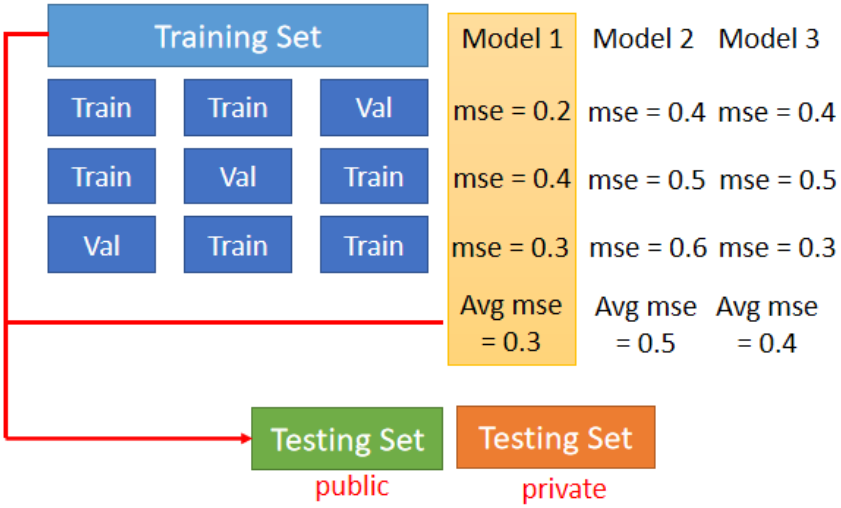
2.5 Cross Validation



1. 把 training 的資料分成兩部分，一部分叫作 training set，一部分是 validation set
2. 在 validation set 上面去衡量它們的分數，根據 validation set 上面的分數去挑選結果，不要管在 public testing set 上的結果以避免 overfiting

2.6 N-fold Cross Validation

N-fold Cross Validation 就是先把訓練集切成 N 等份，切完以後拿其中一份當作 Validation Set，另外 N-1 份當 training set，重覆 N 次



把多個模型在這三個 setting 下通通跑過一次，把 N 種狀況的結果都平均起來，看看誰的結果最好；最後再把選出來的 model（這裡是 Model 1），再拿全部的 training set（public） 訓練 Model 1，訓練完畢後，再用在 testing set（private） 上面

2.7 Mismatch

訓練集跟測試集的分佈是不一樣的，依照對數據本身的理解來判斷