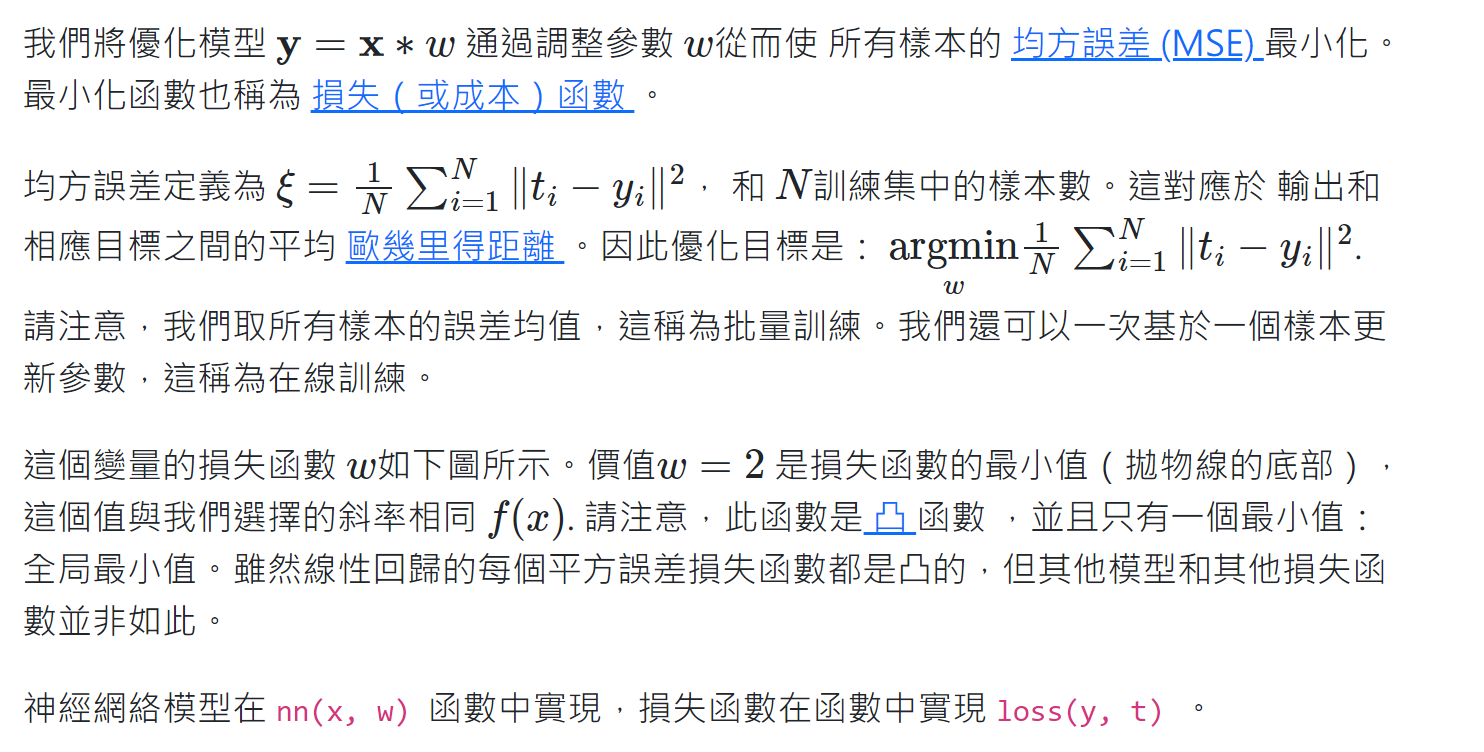
1. Define a loss-function for SVM

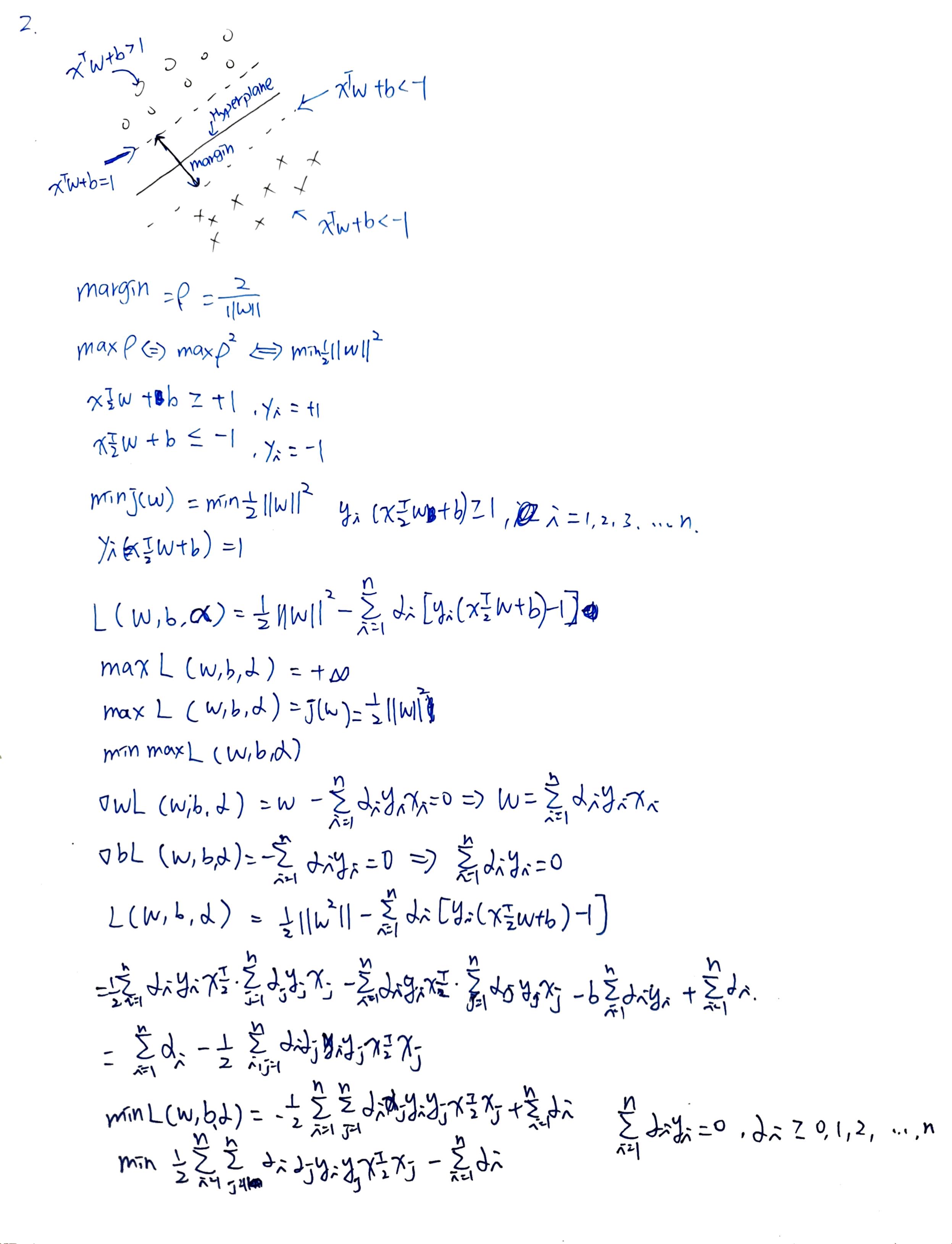
在機器學習中，鉸鏈損失是一個用於訓練分類器的損失函數。鉸鏈損失被用於「最大間格分類」，因此非常適合用於支持向量機 (SVM)。對於一個預期輸出 ，分類結果的鉸鏈損失定義為

特別注意：以上式子的應該使用分類器的「原始輸出」，而非預測標籤。例如，在線性支持向量機當中，，其中 是超平面參數， 是輸入資料點。

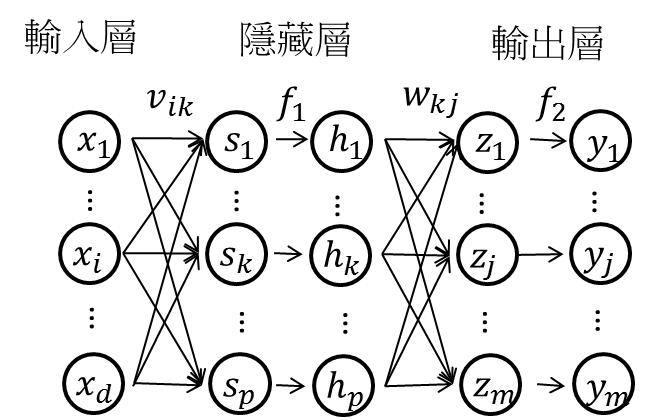
當和同號（意即分類器的輸出是正確的分類），且 時，鉸鏈損失。但是，當它們異號（意即分類器的輸出是錯誤的分類）時，隨線性增長。套用相似的想法，如果 ，即使和同號（意即分類器的分類正確，但是間隔不足），此時仍然會有損失。



1. Derivation of optiomal for a binary SVM



1. Derivation of optimal s for a 3-layers MLP



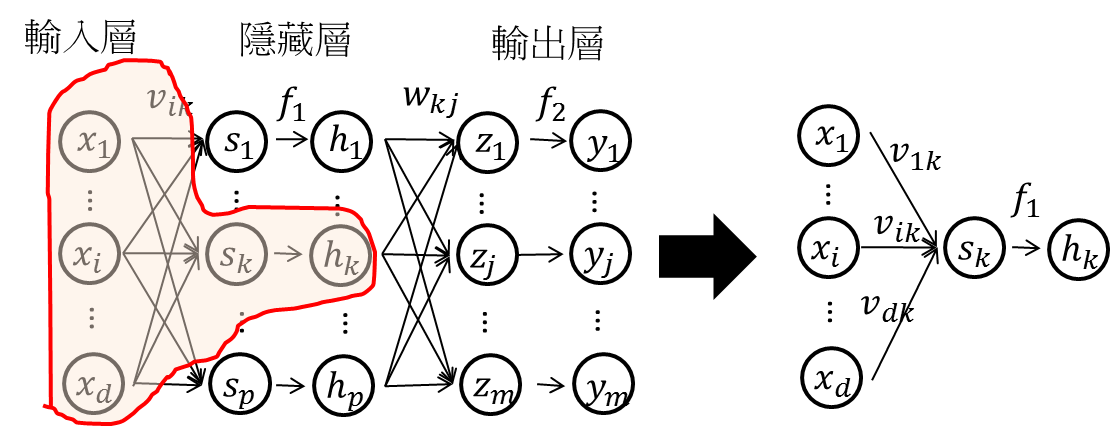
假設有個MLP的結構，共有*n*筆樣本，每個樣本對應*m*個輸出值。隱藏層只有一層設定為*p*個hidden node。https://miro.medium.com/max/598/1*RRinWFzwEtNgRqLUSbvG6A.png

前向傳遞(Forward propagation): 較簡單 (只有線性合成，和非線性轉換)

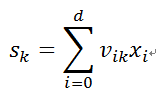
反向傳遞 (Backward propagation): 較複雜 (因為多微分方程)

**前向傳遞 (Forward propagation)**

**輸入層到隱藏層**

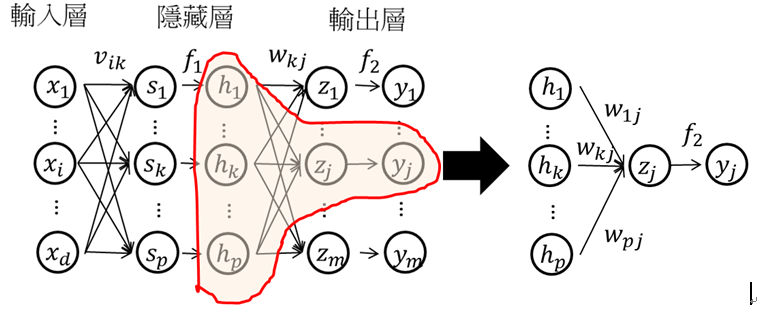


輸入層到隱藏層的值為s*k*,*k*=1,…,*p*，為輸入訊號的加權線性和(v*ik*為第i個輸入到第k個hidden node的權重)。

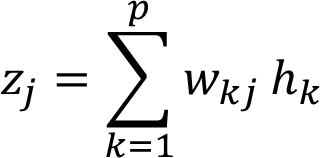


經過 非線性轉換/激活函數(activation function，f*1*)後，得到hidden node的輸出

https://miro.medium.com/max/189/1*vUq7J0WaKOi2BiskwJxDXg.png

**隱藏層到輸出層**

隱藏層到輸出層的值為*zj,j=1,…,m*，為hidden node輸出的加權線性和(w*kj*為第k個hidden node輸出到第j個輸出值的權重)



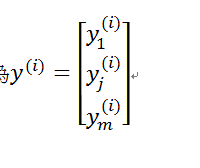
經過 非線性轉換/激活函數(activation function，*f2*)後，得到推估的輸出值

https://miro.medium.com/max/189/1*UgBUl0j61peuNkii7f2q6A.png

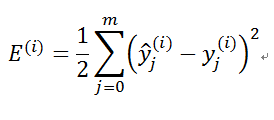
**反向傳遞 (Backward propagation)**

反向傳遞的目的就是利用最後的目標函數(loss/cost function)來進行參數的更新，一般來說都是用誤差均方和(mean square error)當作目標函數。如果誤差值越大，代表參數學得不好，所以需要繼續學習，直到參數或是誤差值收斂。

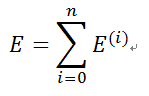
x^(i)為第i筆資料的輸入值，其輸出值為



其目標的誤差為



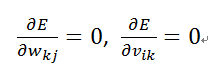
所有樣本的誤差和當作目標函數



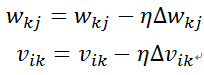
最佳化的目的就是讓「所有樣本的誤差均方和」越小越好，所以目標是

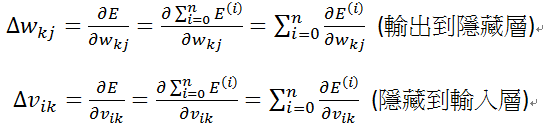
https://miro.medium.com/max/168/1*-0Dn_DmrstRny8nApFnejA.png

所以要找到最佳參數解(參數只有w*kj*和v*ik*)，最簡單的方式就是微分方程式等於0找解



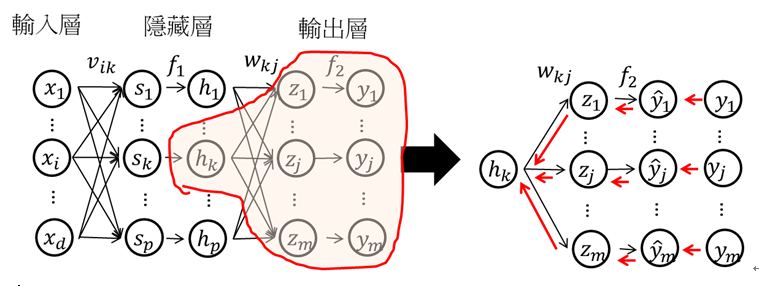
但參數量多無法直接找到唯一解(後面公式有偏微分後的結果，很難直接找到唯一解)，所以還是需要依賴gradient descent找最佳解。

假設讀者對gradient descent有基本認識。  
利用gradient descent找最佳參數解(參數只有w*kj*和v*ik*) 

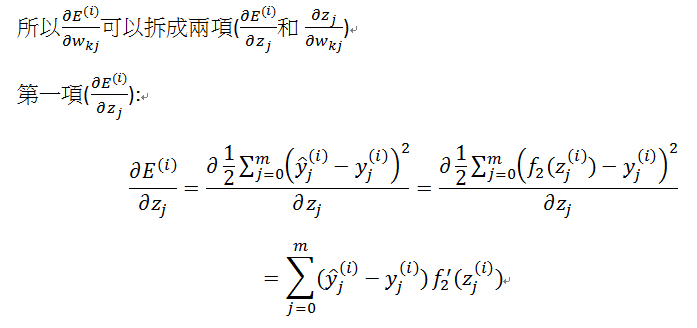
其中η為學習率(learning rate)，

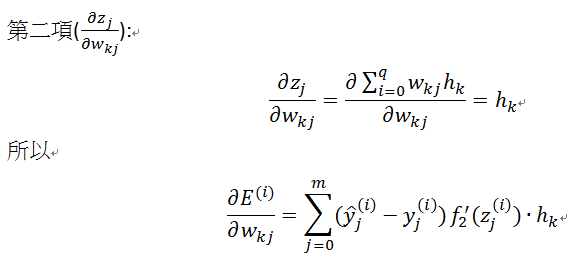
基本上微分解無法直接算出，因此用chain rule方式，可以更有效得到解，以下針對不同層別的連結算倒傳遞 (只針對一個樣本去計算)

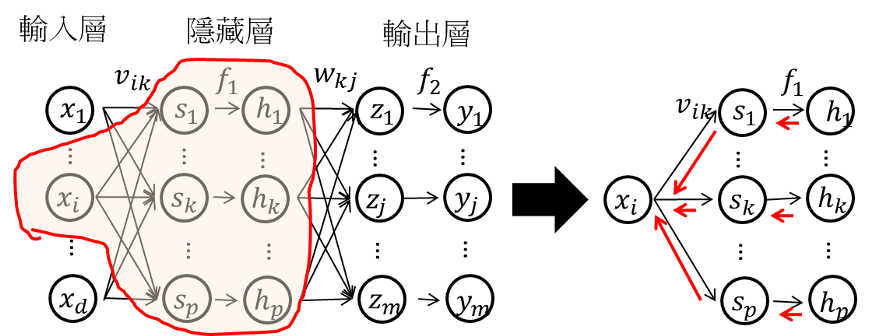
**輸出到隱藏層(w*kj*)**

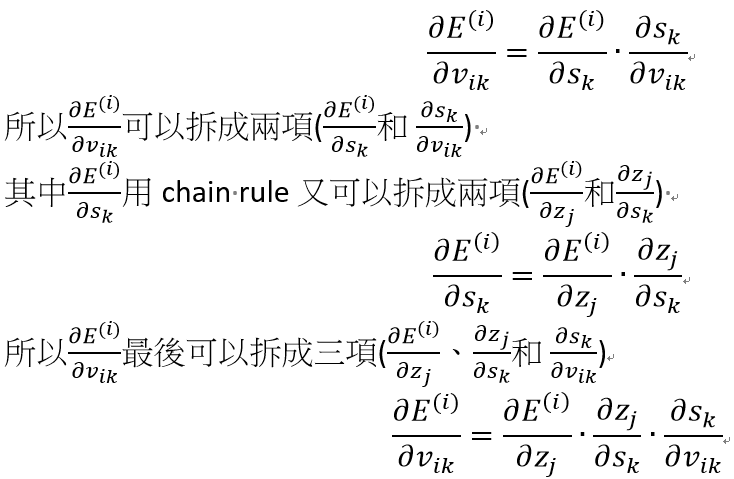


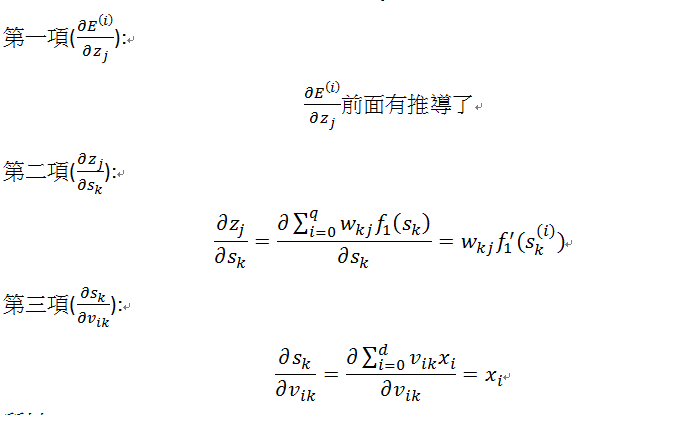
chain rule:



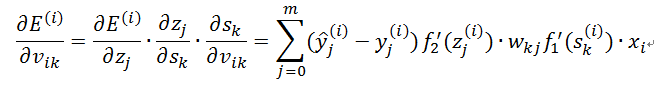


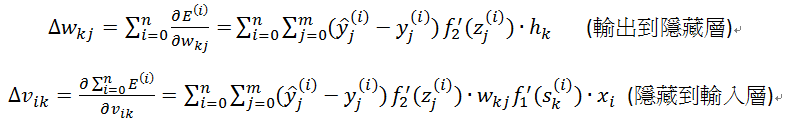
**隱藏層到輸入層(v*ik*)** 

chain rule: 

2020/08/20 修改所以E/V可以拆成兩項的筆誤

所以

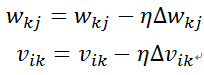


最後把n個樣本所有gradient加起來得到參數的update

到這邊倒傳遞也推導完成了，看完有沒有覺得很簡單(這邊只是符號多了一點而已，基本上用到的數學應該沒有很多)。  
PS:裡面有一個重點，非線性轉換/激活函數(*f1,f2*)在倒傳遞時都有微分，所以在選擇激活函數時**必須**要選擇可微分函數。

結論

MLP神經網路只是在利用gradient descent找最佳參數解



最後帶入MLP內的前向傳遞 (Forward propagation)即可得到最後的預測值。