**2018電子商務技術 期末考**

**\*\* 考試時間共3小時**

**\*\* 若遇計算，請四捨五入取至小數點第3位**

1. **參考下圖weather資料回答以下問題：  
   (b) 如何運用”ChiSquaredAttributeEval”檢驗屬性的必要性？(10%)**

****

用以比較input attribute 與 output attribute 的關聯性，若關聯性越大，Chi-Square value越大，只對nominal attribute有效，numeric attribute的 Chi-Square Value 為 0

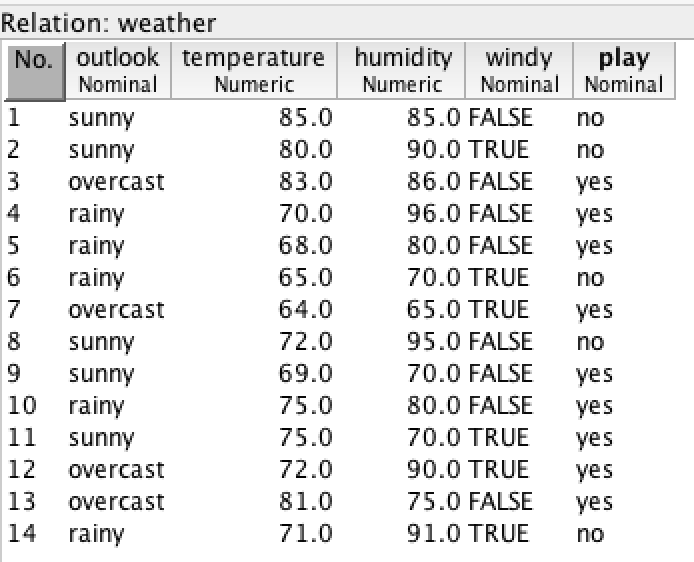
**(c) 請以equal-width binning方式轉換humidity的值，bin個數為3。(10%)**

Interval = (96-65)/3 = 10.3 (取11)

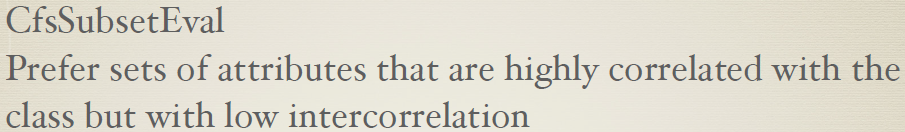
Bin 1 (-inf-76) = 65、70、70、70、75

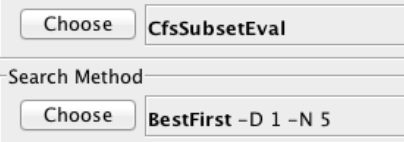
Bin 2 (76-87) = 80、80、85、86

Bin 3 (87-inf) =90、90、91、95、96

****

1. **參考下圖回答問題：  
   (a) CfsSubsetEval的功能是什麼？(5%)**

 **(b) 為何需要挑選search method？(5%)**

需要利用挑選 search method 找出最佳的 attribute set。  
****

1. **某SOM模型有三個input nodes、三個output nodes A, B, C（彼此之間的距離為1），而連線之間的權重如下，請根據此模型回答問題。**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

1. **輸入以下6個instances，一輪後，此SOM模型的結果為何？(10%)**

**X1(1.1, 1.7, 1.8)、X2(0, 0, 0)、X3(0, 0.5, 1.5)、X4(1, 0, 0)、X5(0.5, 0.5, 0.5)、X6(1, 1, 1)**

**其中：r(t) = 0.5, d(t) = 1**

**FOR X1(1.1, 1.7, 1.8)**

D(YA) =

D(YB) =

D(YC) = winner

**Update weights**

wij(new) = wij(current) + r(t)\*(Xi - wij(current))

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

W1A = 0.83 + 0.5 \* (1.1-0.83) = 0.965

W2A = 0.5 + 0.5 \* (1.7-0.5) = 1.1

W3A = 0.81 + 0.5 \* (1.8-0.81) = 1.305

W1B = 0.0 + 0.5 \* (1.1-0.0) = 0.55

W2B = 0.23 + 0.5 \* (1.7-0.23) = 0.965

W3B = 0.3 + 0.5 \* (1.8-0.3) = 1.05

W1C = 0.61 + 0.5 \* (1.1-0.61) = 0.855

W2C = 0.95 + 0.5 \* (1.7-0.95) = 1.325

W3C = 1 + 0.5 \* (1.8-1) = 1.4

1. **利用(a)的結果判斷X5(0.5, 0.5, 0.5)的類別。(5%)**
2. **試比較Bagging與Boosting的異同。 (10%)**

**Bagging**

從訓練資料中隨機抽取(取出後放回，n<N)樣本訓練多個分類器(要多少個分類器自己設定)，每個分類器的權重一致最後用投票方式(Majority vote)得到最終結果，而這種抽樣的方法在統計上稱為Bootstrapping。如果模型不是分類問題而是預測的問題，分類器部份也可以改成regression，最後投票方式改成算平均數即可。如果是用Bagging會希望單一分類器能夠是一個效能比較好的分類器。

Bagging的優點在於原始訓練樣本中有噪聲資料(不好的資料)，透過Bagging抽樣就有機會不讓有噪聲資料被訓練到，所以可以降低模型的不穩定性。

**Boosting**

Boosting算法是將很多個弱的分類器(weak classifier)進行合成變成一個強分類器(Strong classifier)，和Bagging不同的是分類器之間是有關聯性的，是透過將舊分類器的錯誤資料權重提高，然後再訓練新的分類器，這樣新的分類器就會學習到錯誤分類資料(misclassified data)的特性，進而提升分類結果。

缺點為Boosting對訓練資料的噪聲非常敏感，如果一筆訓練資料噪聲資料很多，那後面分類器都會集中在進行噪聲資料上分類，反而會影響最終的分類性能，造成overfitting。

**Bagging與Boosting的區別之處：**

**訓練樣本:**

Bagging: 每一次的訓練集是隨機抽取(每個樣本權重一致)，抽出可放回，以獨立同分布選取的訓練樣本子集訓練弱分類器。

Boosting: 每一次的訓練集不變，訓練集之間的選擇不是獨立的，每一是選擇的訓練集都是依賴上一次學習得結果，根據錯誤率(給予訓練樣本不同的權重)取樣。

**分類器:**

Bagging: 每個分類器的權重相等。

Boosting: 每個弱分類器都有相應的權重，對於分類誤差小的分類器會有更大的權重。

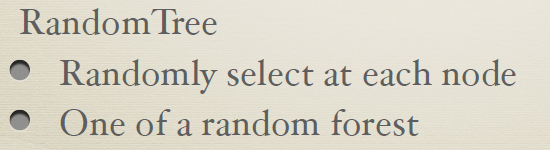
**每個分類器的取得:**

Bagging: 每個分類器可以並行生成。

Boosting: 每個弱分類器只能依賴上一次的分類器順序生成。

1. **試比較RandomSubSpace, RandomTree 與Random Forest。(10%)**

**RandomSubSpace：**概念跟bagging很像，不同的是bagging是從訓練樣本去抽樣產生不同的訓練集來訓練分類器，但Random subspace是feature bagging，從特徵中去抽樣，然後訓練多個分類器做合成，通常用在非常高維度的資料中。

**RandomTree：**

**Random Forest：**Bagging + Decision tree，即每棵決策樹都是一個分類器（假設現在針對的是分類問題），如果訓練集大小為N，對於每棵樹而言，隨機且有放回地從訓練集中的抽取N個訓練樣本（這種採樣方式稱為bootstrap sample方法），作為該樹的訓練集，N棵樹會有N個分類結果。而隨機森林集成了所有的分類投票結果，將投票次數最多的類別指定為最終的輸出。

1. **試依據下述Keras程式回答問題：   
   (a) 試描述此CNN網路的架構。必須寫出各層的節點數。（10％）**

**卷積層 (Convolution Layer)**

CNN 的作法是先透過濾鏡(filter)來提取圖片特徵(feature) 例如取影像中的邊緣、銳利化、模糊化等等，透過不同的 filter 來盡可能取得影像特徵，然後再將這些特徵作為後面 Neuron 的輸入。

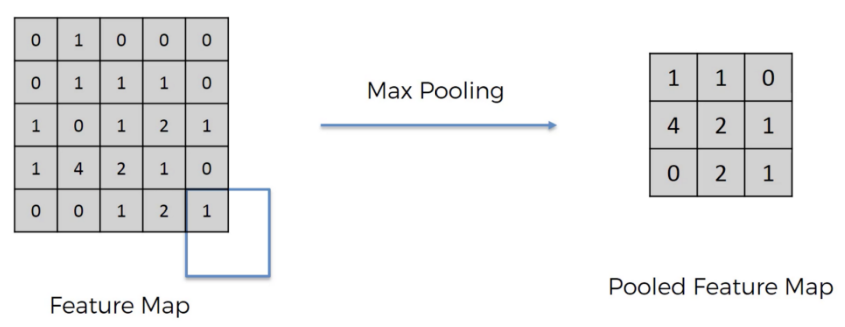
**池化層 (Pooling Layer)**

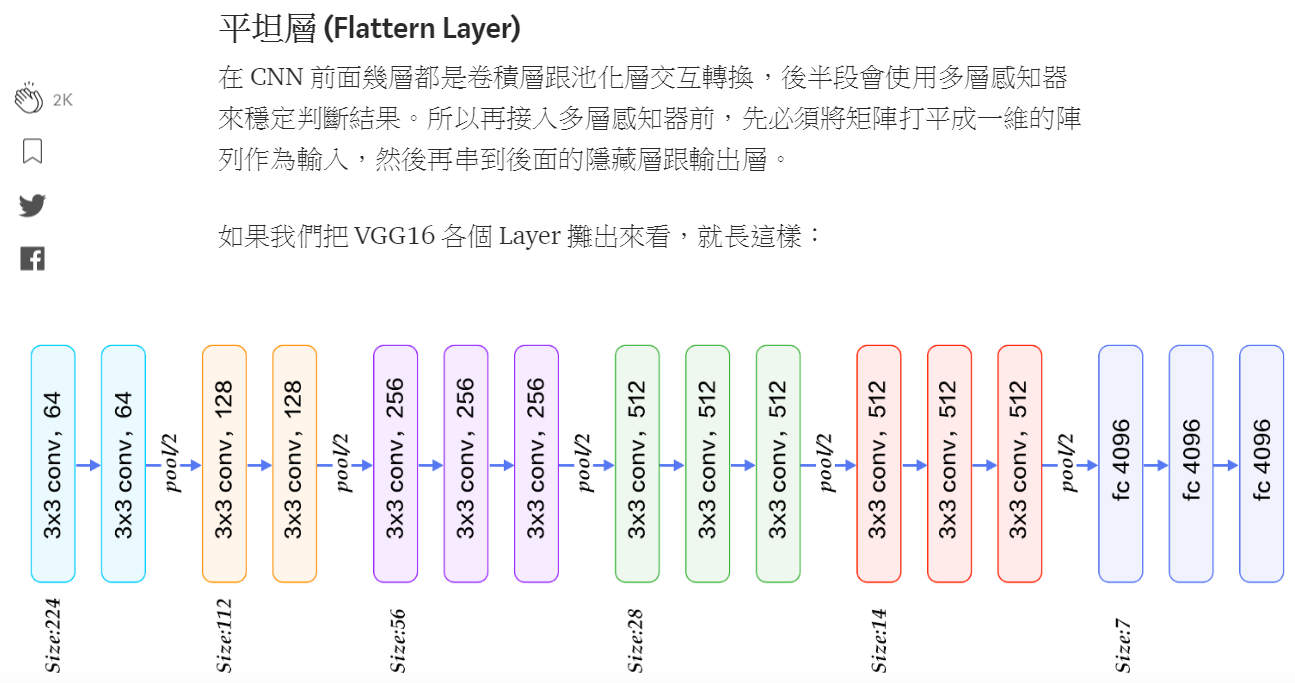
✴形成較⼩的圖像，但不影響特徵

✴減少網路中的節點數

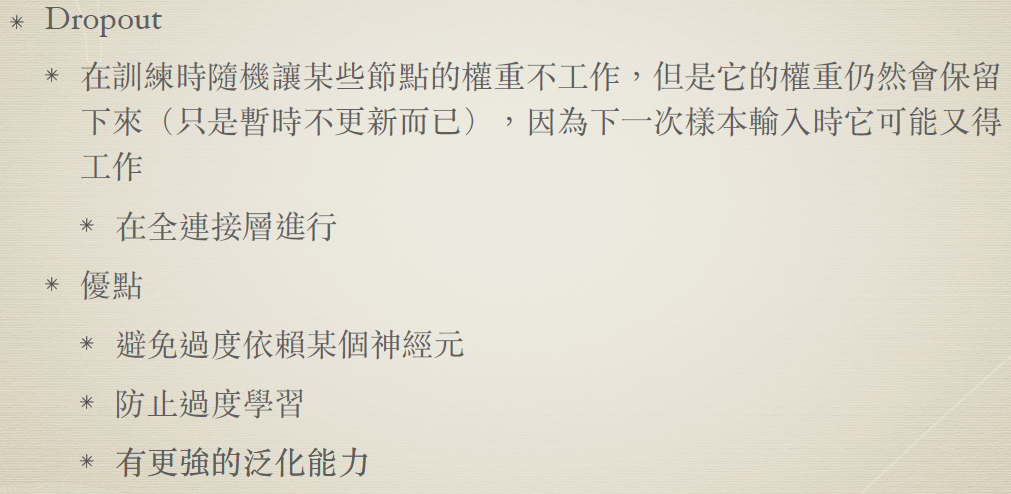
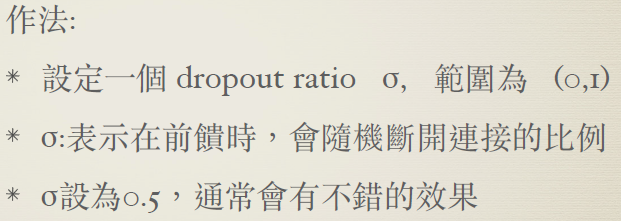
✴能夠有更低的維度，減少計算量

✴不容易過度學習，當參數過多的時候很容易造成過度學習



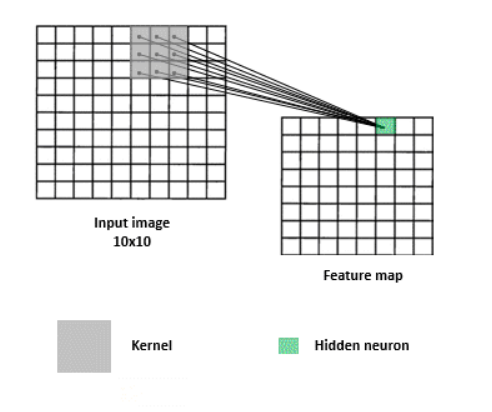


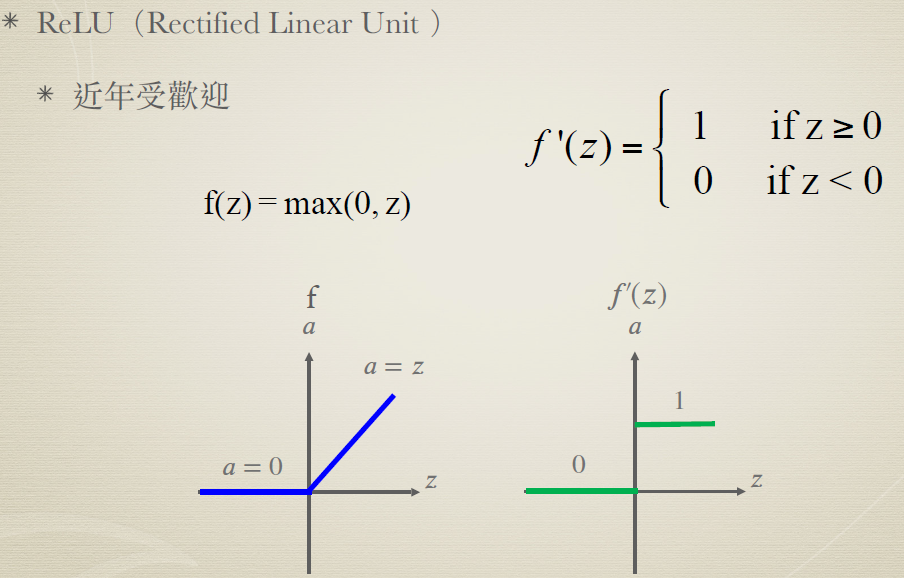
#### 輸出層

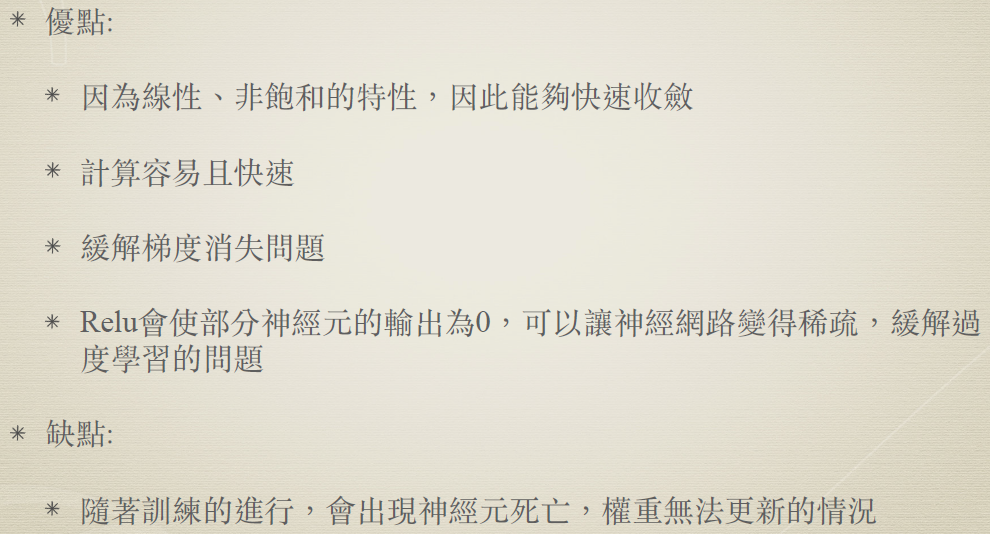
最終會輸出 10 個類別，所以在最後一層有 10 個神經元，總共有 10個輸入值，我們需要將這 10 個輸入值轉化成各個類別的「機率」 (介於 0 到 1 間)，所以在最後一層會套用 softmax 的激活函式 (activation function)

**(b) 就第一個卷積層，說明權重分享的做法。（5％）**

權重共享：不同的圖像或者同⼀張圖像**共⽤⼀個卷積核**，減少重複的卷積核。同⼀張圖像當中可能會出現相同的特徵，共享卷積核能夠進⼀步**減少權重數⽬**，可以⼤⼤的**減少權重參數的訓練**。

 **(c) 說明“relu”的算法與意義。（5％）**





**(d) 說明判斷output的算法。（5％）**

將結果轉換為機率[0,1]，輸出總和為1，輸出預測值越⼤，表有較⾼機率為該類別(class)

***model2.add(Convolution2D(10,3,3, input\_shape=(32,32,1)))***

30\*30\*10 ***model2.add(MaxPooling2D((2,2)))***

15\*15\*10 ***model2.add(Convolution2D(25,3,3))***

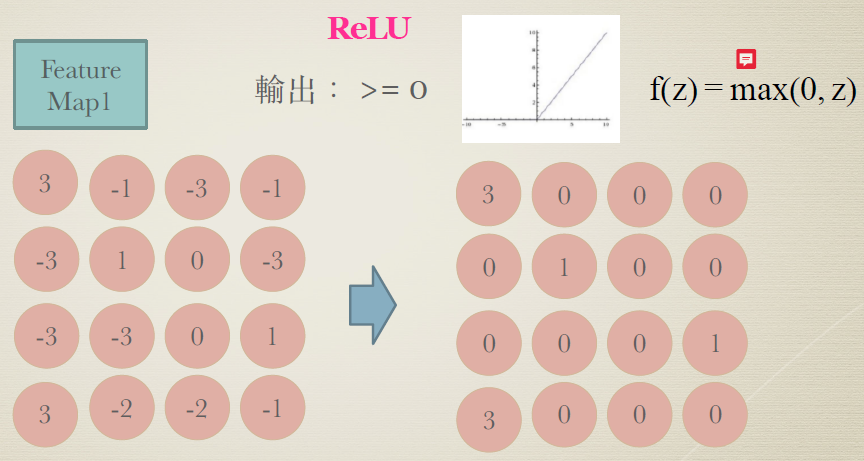
13\*13\*25 ***model2.add(MaxPooling2D((2,2)))***

7\*7\*25 ***model2.add(Flatten())***

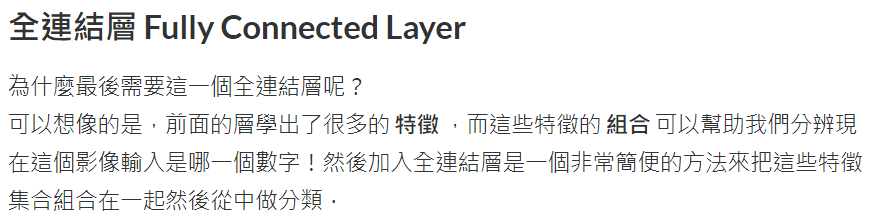
7\*7\*25 (透過Flatten轉為一維陣列) ***model2(Dense(output\_dim=100))***

100 (全連接層) ***model2(add(Activation(‘relu’))***

Non-linearity

 ***model2(Dense(output\_dim=10))***

10

***model2(add(Activation(‘softmax’))***



1. **試描繪以下MLP的網路架構（含各層的節點、連線及其權重）。(10%)**

