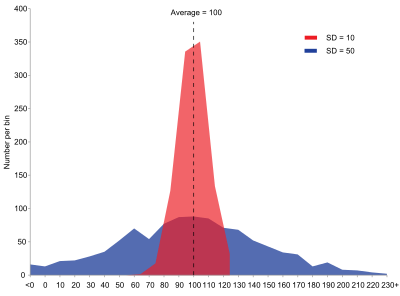
1. **什麼是偏差(bias)、方差(變異數、variable)之間的均衡?**
   * Variable為一隨機變數的離散程度。

EX: 紅色群體與藍色群體均值都為100，但紅色群體變異數為100、標

準差為10，藍色群體變異數為2500、標準差為50，代表藍色群體

裡離散程度比較大。

**Bias** => 為模型預測出來的值與真實值之間的誤差。Bias可能會導致欠擬

合，使模型預測值很難有較高的準確性，也很難從訓練集訓練出來

的模型應用到測試集。

**Variance** => 為模型每一次輸出結果與模型輸出期望之間的誤差，即模型的

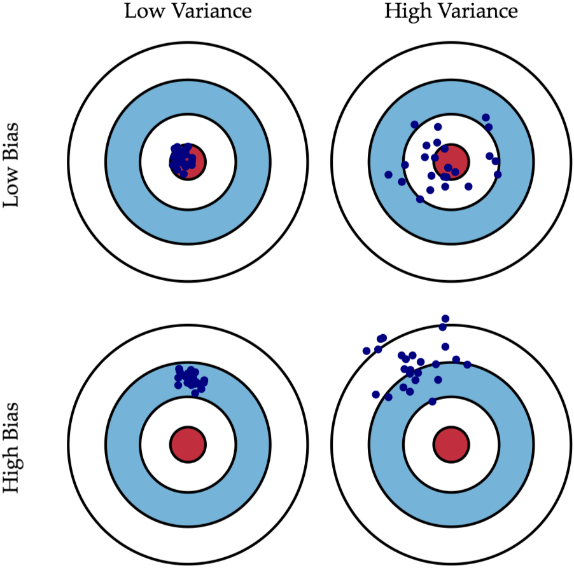
穩定性。若Variance過高，則會導致演算法對高維度變化過於

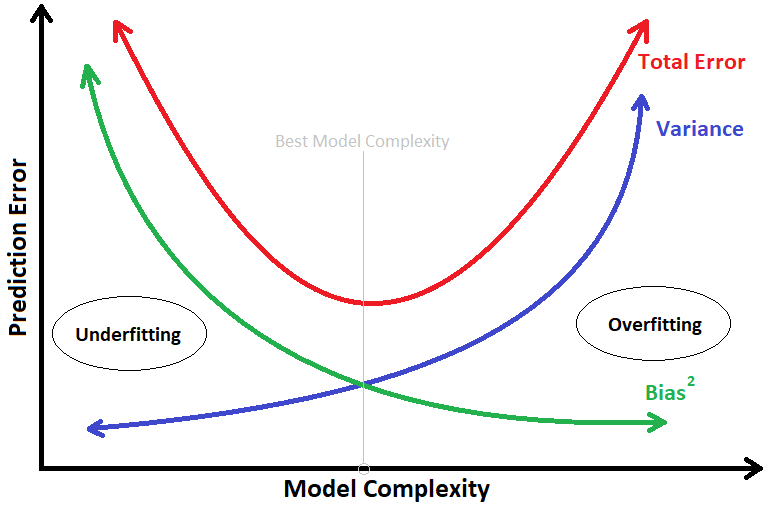
敏感，也就是會導致模型過擬合。

**Bias-Variance Tradeoff** => 為對模型同時加入偏差、方差與一點噪聲。透過

權衡 Bias Error 跟 Variance Error 來使得總誤差

( Total Error ) 達到最小。



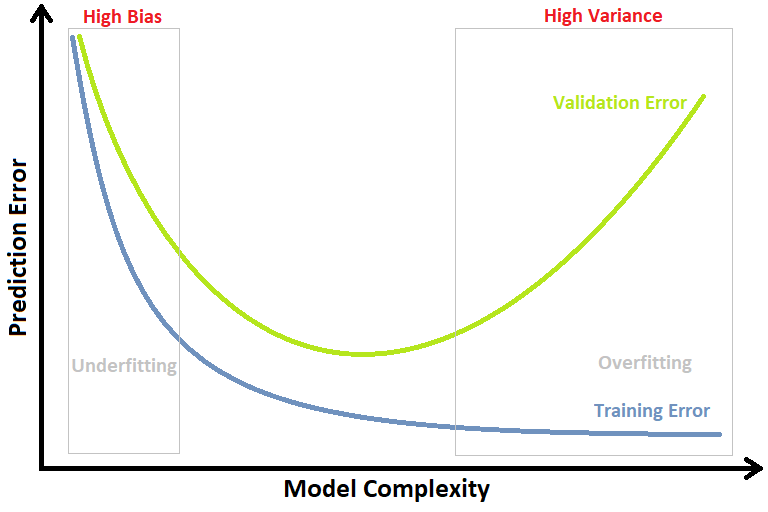


1. **如何找到之間的均衡?**

在實務上我們更常透過模型外在的表現來判斷它現在是 underfitting 還是 overfitting，再透過調整模型的超參數( Hyperparameter ) 來調整模型的複雜度。實際操作上，一般我們會將 dataset 切割成 training set 跟 validation set，training set 用於訓練模型；而 validation set 將不會參與訓練，用於評估模型是否 overfitting。

如果這部分一樣把 模型的複雜度 與 模型預測的誤差 畫成圖表的話，見

下圖：



可以觀察到：

* 在 Underfitting 的時候，不論是在 Training set 還是 Validation set 的

Error 都很高。

* 在 Overfitting 的時候，Training set 的 Error 已經將降低了，但 Validation set 上的 Error 會很高。

​有了這兩個重要的觀察，在訓練的時候我們就可以很容易的判斷 model fitting 的好不好了！

1. **Overfitting vs underfitting**

**【Underfitting】**

* 前面有提到，發生 Underfitting 的根本原因是由於**模型太過簡單**，所以根本的解決方案就是提高模型的複雜度，可以透過：

1. 增加訓練的疊代次數
2. 調整超參數 ( 修改模型架構 )
3. 生成更多的特徵來訓練模型
4. 如果有使用正規化 ( Regularization ) 可先將其移除
5. 更換一個更複雜的模型

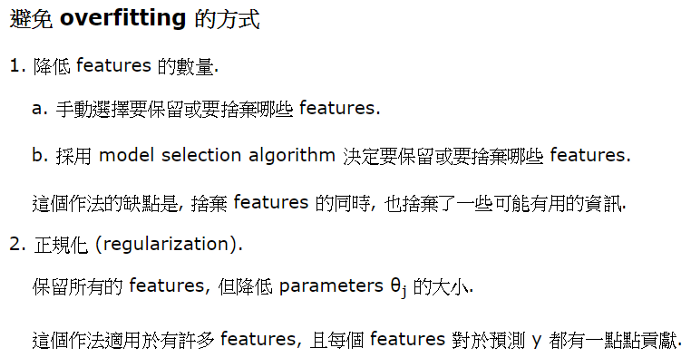
**【Overfitting】**

* 前面有提到，發生 Overfitting 的根本原因是由於**模型太過複雜**，所以根本的解決方案就是降低模型的複雜度，可以透過：

1. Early Stopping
2. 增加訓練資料
3. 降低特徵維度
4. 如果沒有使用正規化 ( Regularization ) 可以將其加入
5. 調整超參數 ( 修改模型架構 )
6. 更換一個較為簡單的模型​
7. 保持模型盡可能地簡單：通過考量較少的變數和參數來減少方差，達到資料中消除部分噪音的效果。
8. 使用交叉檢驗的手段如：k-folds cross-validation。
9. 使用正則化的技術如：LASSO方法來懲罰模型中可能導致過擬合的參數。

結論:

**欠擬合**的模型將具有高訓練和高測試誤差，而**過擬合**模型將具有極低的訓練誤差但具有高測試誤差。



1. **KNN和K-means聚類有什麼不同?**

**【K-Nearest Neighbors】**

其觀念是根據資料彼此之間的距離進行分類，距離哪一種類別最近，就分到哪一類，所以 KNN 常被用於分類問題。

思想與「近朱者赤，近墨者黑」有異曲同工之妙！

* **KNN 的演算法步驟如下：**
  + KNN 中的 K 代表找出距離你最近的 K 筆資料進行分類。
  + 假設 K 是 3，代表找出前三筆相近的資料。
  + 當這 K 筆資料被選出後，就以投票的方式統計哪一種類的資料量最多，然後就被分到哪一個類別。
* KNN 不只可以處理離散型的資料，想可以處理連續型的資料。
* K值選擇為預測準確程度的關鍵，且最好選擇奇數以避免投票平常手的狀 況發生。
* K 值越大，可能會包含不相關的樣本點進來，K 值太小又會被會為是噪音的干擾（雜訊），因為參考的樣本數不夠多，通常都會把 K 值控制在樣本數平方根以下，是最好的結果。
* 因為我們不知道哪一個 K 值是最好的，我們可以去切割樣本去做交叉 驗證找到適合的 K 值。選擇適合的距離計算方式。

**KNN 的優點：**

* 簡單易懂
* 資料型態不受限（離散型：取眾數，連續型：取平均值）
* 在多種類別預測有較好的表現

**KNN 的缺點：**

* **計算本成本高**：因為他要不斷地計算點跟點之間的距離
* 資料不平衡時容易產生預測不準確的狀況發生

**【k-means】**

K-means是非監督式學習。

**K-means運作概念步驟:**

1. 我們先設定好要分成多少(k)群。

2. 然後在feature space(x軸身高和y軸體重組出來的2維空間，假設資料是d維，則會組出d維空間)隨機給k個群心。

3. 每個資料都會所有k個群心算歐式距離(歐基李德距離Euclidean distance，其實就是直線距離公式，從小學到大的那個距離公式，這邊距離當然也可以換成別種距離公式，但基本上都還是以歐式距離為主)。

4. 將每筆資料分類判給距離最近的那個群心。

5. 每個群心內都會有被分類過來的資料，用這些資料更新一次新的群心。

6. 一直重複3–5，直到所有群心不在有太大的變動(收斂)，結束。

**【knn vs k-means】**

**KNN**

* 分類
* 監督式學習
* 不需要學習（不去學習一個判別式函數(損失函數)而是要記住整個訓練集）
* 在範圍內與旁邊的鄰居進行比較，並以這幾個鄰居（訓練樣本）中出現最多的分類標籤作為最終新樣本數據的預測標籤

**K-Means**

* 分群
* 非監督式學習（沒有label的資料）
* 需要訓練
* 比較主觀作分類，由人決定k值，目測要分成幾類

**相同及相異的地方**

* 相同：都是透過一個中心的點，由旁邊的點再去做分析（並不完全相同）
* 相異：兩種算法之間的最大區別是，
  1. K-means本質上是無監督學習，而KNN是監督學習
  2. K-means是聚類算法，KNN是分類（或回歸）算法。
  3. K-means算法把一個數據集分割成cluster，每個cluster裡的點相互靠近，必須一直計算改變 cluster centroid，就好像班上的核心人物一直在改變。

KNN算法嘗試基於其k（可以是任何數目）個周圍鄰居來對未標記的觀察進行分類。它也被稱為懶惰學習法，因為它涉及最小的模型訓練。所以k對整個模型非常重要。

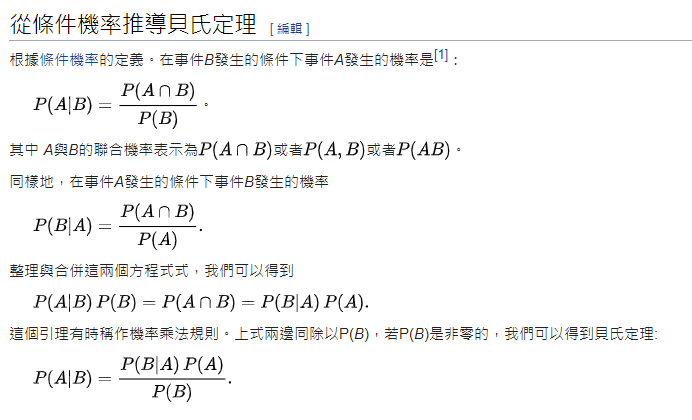
1. **監督式學習與非監督式學習有甚麼不同?**

監督學習需要有label的資料。例如，為了進行classification（一項受監督的學習任務），您需要首先標記將用於train模型的資料，以便將資料分類到標記的組中。相反的，無監督學習不需要明確標記資料。

1. **什麼是貝葉式定理? 如何應用在機器學習環境?**

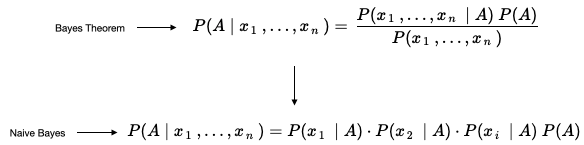
貝葉斯定理描述了當你不能準確知悉一個事物的本質時，你可以依靠與事物特定本質相關的事件出現的多少去判斷其本質屬性的概率。 它給出了已知先驗知識下事件的後驗概率。

P(A|B) =



1. **Bayes them Vs Naïve bayes**

請記住，這就是為什麼它被稱為“樸素”(naive)的原因，因為所有特徵的條件概率都是相互獨立計算的。樸素貝葉斯算法在獨立性和刪除分母的幫助下得到了簡化。



1. **L1、L2正則之間有甚麼不同?**

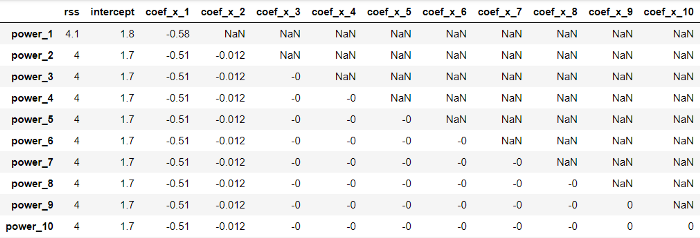
兩函數都可以幫助降低模型的複雜度。

**L1**:

https://lh6.googleusercontent.com/ZkeC-2_tsStTiQsIybfSywYVdvIAlkN9xUutsUobLxfqgaJbhu96fHXfNRGhu4bfmF54fwDgEs_gvmRuEhecBQCPHO4lZ0DVNzq5hkO2MmYR7KLAJE1JbCcVyDiP2eFl9EezKfo2

L1 正則化，也稱為 L1 範數或 Lasso（在回歸問題中），通過將參數縮小到 0 來對抗過度擬合。這使得一些特徵過時了。

這是特徵選擇的一種形式，因為當我們為特徵分配 0 權重時，我們將特徵值乘以 0 返回 0，從而消除了該特徵的重要性。如果我們模型的輸入特徵的權重接近 0，我們的 L1 範數將是稀疏的。輸入特徵的選擇將具有等於零的權重，其餘的將是非零的。



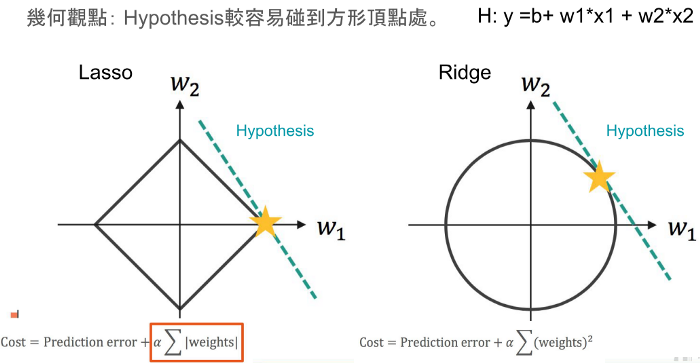
Lasso(L1) Regularization相較於Ridge(L2) Regularization會產生較多零的

coefficient，這個特性可以用來做重要Feature Extraction。

**LASSO:**

1.針對Ridge Regression的沒法做variable selection的問題提出來的

2.L1 penalty雖然算起來麻煩，沒有解析解，但可以把某些係數shrink到0啊。

3.但是不consistent啊，而且當n很小時至多只能選出n個變量；而且不能做group selection。

**L2**:

https://lh3.googleusercontent.com/IgXPFQMDqPVlStgz3OEwtcPuiL1Y7-3fNeCcKBTK6o_Jn0D47qBWSyD-QBpCp11nOUzmIXJ3tt5ON7-gzlJcK_N2U5DCJ606oKWXFFo1tsxXkMdAnd-W6y4n99A4brnSh_GAiTf0

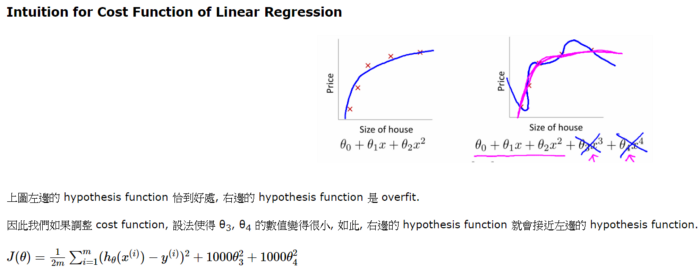
L2 正則化，或 L2 範數，或 Ridge（在回歸問題中），通過強制權重變小但不使權重恰好為 0 來對抗過度擬合。

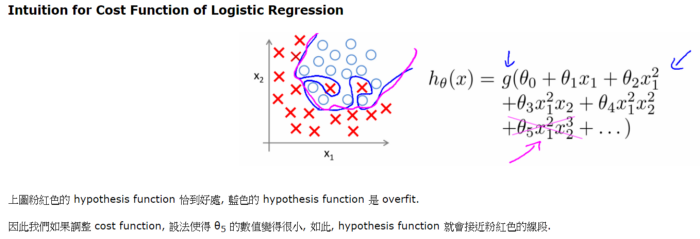
使用 L2 正則化時要考慮的一個主要問題是它對異常值不穩健。平方項會放大異常值的誤差差異。然後，正則化將嘗試通過懲罰權重來解決這個問題。

**L1和L2正則化的區別**：

* L1 正則化懲罰權重的絕對值之和，而 L2 正則化懲罰權重的平方和。
* L1 正則化解決方案是稀疏的。L2 正則化解決方案是非稀疏的。
* L2 正則化不執行特徵選擇，因為權重僅減少到接近 0 而不是 0 的值。L1 正則化具有內置的特徵選擇。
* L1 正則化對異常值具有魯棒性，而 L2 正則化則不然。

**應用**: 避免overfitting



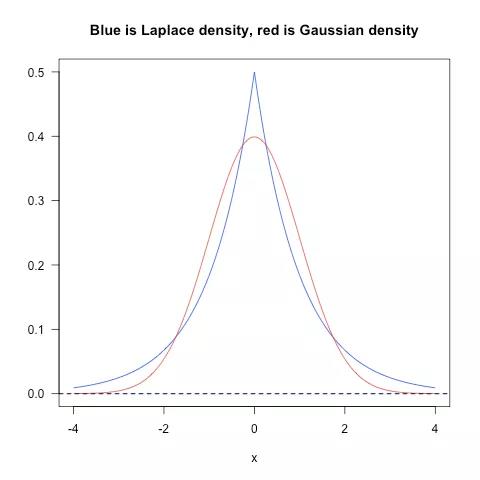


* + **魯棒性解釋:**

在機器學習領域，總是看到“演算法的魯棒性”這類字眼，比如這句 –

L1範數比L2范數魯棒。

Huber從穩健統計的角度系統地給出了魯棒性3個層面的概念：

* 模型具有較高的精度或有效性，這也是對於機器學習中所有學習模型的基本要求；
* 對於模型假設出現的較小偏差，只能對演算法性能產生較小的影響； 主要是：雜訊（noise）
* 對於模型假設出現的較大偏差，不可對演算法性能產生“災難性”的影響；主要是：離群點（outlier）
* **結論**:
* L2正则，对应的是加入2范数，使得对权重进行衰减，从而达到惩罚损失函数的目的，防止模型过拟合。保留显著减小损失函数方向上的权重，而对于那些对函数值影响不大的权重使其衰减接近于0。相当于加入一个gaussian prior。
* L1正则 对应得失加入1范数，同样可以防止过拟合。它会产生更稀疏的解，即会使得部分权重变为0，达到特征选择的效果。相当于加入了一个laplacean prior。
* https://medium.com/ai%E5%8F%8D%E6%96%97%E5%9F%8E/learning-model-l1-l2-regularization%E5%B7%AE%E7%95%B0-8d7fc089b35c

1. **什麼是深度學習，它與機器學習算法之間有什麼關係。**

深度學習是與神經網路有關的機器學習的一個子集：如何使用反向傳播和神經科學中的某些原理來更精確地建模大量未標記或半結構化資料。從這個意義上說，深度學習是一種無監督的學習演算法，它通過使用神經網路來學習資料的表示。

1. **交叉檢驗如何用在時間序列數據上?**

與標準的k-folds 交叉檢驗不同，資料不是隨機分佈的，而是具有時序性的。如果模式出現在後期，模型仍然需要選擇先前時間的資料，儘管前期對模式無影響。我們可以如下這麼做：

fold1：training[1], test[2]

fold2：training[1 2], test[3]

fold3：training[1 2 3], test[4]

fold4：training[1 2 3 4], test[5]

fold5：training[1 2 3 4 5], test[6]

1. **如何處理一個不平衡的數據集**

過採樣與欠採樣

<https://medium.com/%E6%95%B8%E5%AD%B8-%E4%BA%BA%E5%B7%A5%E6%99%BA%E6%85%A7%E8%88%87%E8%9F%92%E8%9B%87/smote-enn-%E8%A7%A3%E6%B1%BA%E6%95%B8%E6%93%9A%E4%B8%8D%E5%B9%B3%E8%A1%A1%E5%BB%BA%E6%A8%A1%E7%9A%84%E6%8E%A1%E6%A8%A3%E6%96%B9%E6%B3%95-cdb6324b711e>

1-收集更多資料，甚至資料集中的不平衡。

2-對資料集重新取樣以糾正不平衡。

3-在你的資料集中嘗試一個不同的演算法。

這裡重要的是，您對不平衡資料集可能造成的損害以及如何平衡具有敏銳的感知。

1. **什麼時候要使用分類而不是回歸?**

分類產生離散值並將資料集轉換為嚴格的類別，而回歸則提供連續的結果，使您能夠更好地區分各個點之間的差異。

如果您希望結果反映資料集中資料點對某些明確類別的歸屬性（例如：如果您希望知道某個名稱是男性還是女性，而不僅僅是它們與男性和女性名稱之間的關聯性），則可以使用分類而不是回歸。

1. **說明使用集成學習會很有用**

集成學習通過組合一些基學習演算法來優化得到更好的預測性能，通常可以防止模型的過擬合使模型更具有魯棒性。

你可以列舉一些集成學習的例子，如bagging、boosting、stacking等，並且瞭解他們是如何增加模型預測能力的。

1. **如何評估您的機器學習模型的有效性**

首先你需要將資料分成訓練集和測試集，或者使用給交叉驗證方法分割。然後你需要選擇度量模型表現的metrics，如F1數、準確率、混淆矩陣等。

更重要的是，根據實際情況你需要理解模型度量的輕微差別，以便於選擇正確的度量標準。

1. **如何處理數據中丟失或損壞的數據?**

您可以在資料集中找到丟失/損壞的資料，然後刪除這些行或列，或者決定用另一個值替換它們。

在pandas中，有兩種非常有用的方法：isNull（）和dropna（），這兩種方法將幫助您查找缺少或損壞資料的資料列，並刪除這些值。如果要用預留位置值（例如0）填充無效值，可以使用fillna（）方法。

1. **每個演算法各自介紹**

監督：

1. 決策樹：

當您有線性決策邊界時，您可以使用決策樹。一個例子是根據他們的智商對人進行分類：

140 歲以上 - 天才

120 歲以上 - 高於平均水平

等等。您有一個優勢，您可以將某些意義賦予決策。

2.支持向量機：

支持向量機主要用於二進制分類。（它們可以間接用於多類分類，但我不太了解。我用 SVM 做過人臉識別，但描述它會變得過於技術化。）SVM 有優勢（與決策樹相比）您可以將它們用於非線性決策邊界。但缺點是你不能賦予意義（比如為什麼特徵是 x 導致樣本被分類為 y）。SVM 也需要很長時間來訓練。

3. 安：

人工神經網絡可用於多類分類和非線性邊界。但是 ANN 的一個問題是，您必須憑經驗嘗試不同的拓撲結構才能找到最好的拓撲結構。此外，您通常需要大量的訓練樣本（比您在 SVM 中所需的要多得多）才能達到良好的準確性。因此，訓練需要很長時間。您還可以獲得屬於某個類的樣本的概率估計。（Scikit-learn 的 SVM 函數會生成概率，但我不確定他們是如何做到的。他們可能使用到決策邊界的距離或其他東西，但 ANN 直接給你概率。）

4. 隨機森林：

隨機森林可用於分類和回歸。你有一堆決策樹，每個決策樹都由一個（不一定不相交的）特徵子集組成。每棵樹都為該類投票。樣本被歸類為得票最多的類別。隨機森林還可以為您提供特徵重要性的指示。它是最受歡迎的分類器之一，因此您應該幾乎總是嘗試一下。

5. 線性回歸\嶺回歸：

當您想要預測連續值而不是分類時使用。例如，回歸可用於交通預測。嶺回歸減少了預測的方差。方差是測試誤差的組成部分之一，另一個是偏差。

6.樸素貝葉斯分類器：

當特徵條件獨立時，您可以使用樸素貝葉斯分類器。我已經將它用於 RGB 中非常簡單的對象識別，其中三個通道被認為是不相關的。

7. 貝葉斯網絡：

當您需要對因果關係建模時使用貝葉斯網絡，例如醫學診斷。

無監督：

1. 隱馬爾可夫模型：

如果您的數據隨時間發生變化，例如活動識別；你可以使用 HMM。與 ANN 類似，HMM 要求您憑經驗得出拓撲。

我不知道 HMM 的替代品，所以也許你應該查一下。

2. 聚類：

當您需要根據某些特徵對點進行分組時，請使用 k-means 聚類。當您需要根據連通性對點進行分組時，請使用譜聚類。

您還應該查看半監督學習方法。

1. **什麼是稀疏特徵？**

具有稀疏數據的特徵是具有大部分零值的特徵。這與缺少數據的特徵不同。稀疏特徵的示例包括 one-hot-encoded 單詞的向量或分類數據的計數。另一方面，具有密集數據的特徵主要具有非零值。

1. **稀疏數據和缺失數據有什麼區別？**

當有缺失數據時，意味著很多數據點是未知的。另一方面，如果數據是稀疏的，則所有數據點都是已知的，但大多數數據點的值為零。

為了說明這一點，有兩種類型的特徵。具有稀疏數據的特徵具有已知值（= 0），但具有缺失數據的特徵具有未知值（= null）。不知道空值行中應該包含哪些值。

1. **稀疏特徵的常見問題包括：**
2. 如果模型有很多稀疏特徵，就會增加模型的空間和時間複雜度。線性回歸模型將擬合更多的係數，並且基於樹的模型將具有更大的深度來解釋所有特徵。
3. 如果特徵具有稀疏數據，模型算法和診斷措施可能會以未知方式表現。[Kuss [2002]](https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.1421" \t "_blank)表明，當數據稀疏時，擬合優度檢驗是有缺陷的。
4. 如果特徵太多，模型會擬合訓練數據中的噪聲。這稱為過擬合。當模型過擬合時，它們在投入生產時無法泛化到更新的數據。這會對模型的預測能力產生負面影響。
5. 一些模型可能低估了稀疏特徵的重要性，並優先考慮更密集的特徵，即使稀疏特徵可能具有預測能力。基於樹的模型以這樣的行為而臭名昭著。例如，與類別較少的特徵相比，隨機森林高估了類別較多的特徵的重要性。
6. **處理稀疏特徵的方法**
   1. **模型中移除特徵**

稀疏特徵會引入噪聲，模型會拾取噪聲並增加模型的內存需求。為了解決這個問題，可以從模型中刪除它們。例如，從文本挖掘模型中去除稀有詞，或者[去除低方差的特徵](https://scikit-learn.org/stable/modules/feature_selection.html#removing-features-with-low-variance)。但是，在此過程中不應刪除具有重要信號的稀疏特徵。

LASSO 正則化可用於減少特徵的數量。基於規則的方法（例如設置方差閾值以在模型中包含特徵）也可能很有用。

**2. 使特徵密集**

* 主成分分析 (PCA) **-** PCA 方法可用於將特徵投影到主成分的方向，並從最重要的成分中進行選擇。
* [*特徵散列*](https://en.wikipedia.org/wiki/Feature_hashing)- 在特徵散列中，可以使用散列函數將稀疏特徵分箱成所需數量的輸出特徵。必須注意選擇大量的輸出特徵以防止散列衝突。

**3. 使用對稀疏特徵具有魯棒性的模型**

某些版本的機器學習模型對稀疏數據具有魯棒性，可以用來代替更改數據的維度。例如，[熵加權 k-means 算法](https://ieeexplore.ieee.org/abstract/document/4262534)比常規 k-means 算法更適合這個問題。

**結論**

稀疏特徵在機器學習模型中很常見，尤其是以 one-hot 編碼的形式。這些特徵可能會導致機器學習模型出現過度擬合、特徵重要性不准確和高方差等問題。建議對稀疏特徵進行特徵散列或去除特徵等方法進行預處理，以減少對結果的負面影響。

1. **ROC曲線原理(Q4)**
2. **定義精度和召回率(Q5)**

**NLP Test**

* 1. **瞭解Google最新的模型bert嗎？**

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding，提出的 BERT 模型在 11 個 NLP 任務上的表現刷新了記錄，包括問答 Question Answering (SQuAD v1.1)，推理 Natural Language Inference (MNLI)等

* BERT 可以用來幹什麼？

BERT 可以用於問答系統，情感分析，垃圾郵件過濾，命名實體識別，文檔聚類等任務中，作為這些任務的基礎設施即語言模型。

* BERT 的代碼也已經開源：

google-research/bert

我們可以對其進行微調，將它應用於我們的目標任務中，BERT 的微調訓練也是快而且簡單的。

例如在 NER 問題上，BERT 語言模型已經經過 100 多種語言的預訓練，這個是 top 100 語言的清單：

只要在這 100 種語言中，如果有 NER 資料，就可以很快地訓練 NER。

* 1. **什麼是NLP？**

自然語言處理（NLP） 是計算機科學，人工智能和語言學的交叉領域。目標是讓計算機處理或“理解”自然語言，以執行語言翻譯和問題回答等任務。

隨著語音接口和聊天機器人的興起，NLP正在成為信息時代最重要的技術之一，同時它也是人工智能的關鍵部分。充分理解和表達語言的含義是一個非常困難的目標。為什麼？因為人類的語言很特別。

人類語言有什麼特別之處？

* 1. 人類語言是專門為傳達說話人的意圖而構建的系統。這不僅僅是一個環境信號，更是一個有意識的交流。
  2. 人類語言大多是離散/符號的/分類的信號系統，大概是因為信號可靠性更高。
  3. 一種語言的分類符號可以用幾種方式編碼為通信信號：聲音，手勢，寫作，圖像等。人類語言只是其中的一種。
  4. 人類語言是不明確的（與編程和其他正式語言不同）。因此，在表達、學習和使用語言/情境/情境/文字/視覺知識對人類語言方面存在高度複雜性。
  5. **什麼是詞嵌入？**

詞嵌入是一種對文本算法學習後的表示形式，甚至，你可以理解為一個單詞在算法中的儲存形式。大家知道存入計算機的都是0101的數值化序列，這裡也是同理，詞嵌入就是將文本數值化以方便擬合算法。這種將單詞或者文檔數字化表示的方式被認為是深度學習在自然語言處理任務中最具有挑戰性的問題之一。

* 1. **詞嵌入算法有哪些?**

詞嵌入是從文本語料中學習到的一種將單詞表示為預定義大小的實值向量形式。學習過程一般與某個神經網絡的模型任務一同進行，比如文檔分類。

**下面介紹3中詞嵌入技術。**

* 1. **Embedding Layer**

這裡並沒有專業的名次，使用就稱之為Embedding Layer吧。

Embedding Layer是特定的自然語言處理任務（例如語言建模或者文檔分類）上的與神經網絡模型共同學習更新的詞嵌入方法。

使用Embedding Layer通常步驟一般是先預處理語料文本，將每個單詞轉化成one-hot形式的編碼。而此單詞對應的詞向量其實是算法模型的其中一部分，詞向量用預定義的維度來表示，大小我們隨機初始化。在這裡Embedding Layer其實就是神經網絡的input layer，而詞向量矩陣即是input layer 到 hidden layer中間的權值矩陣。

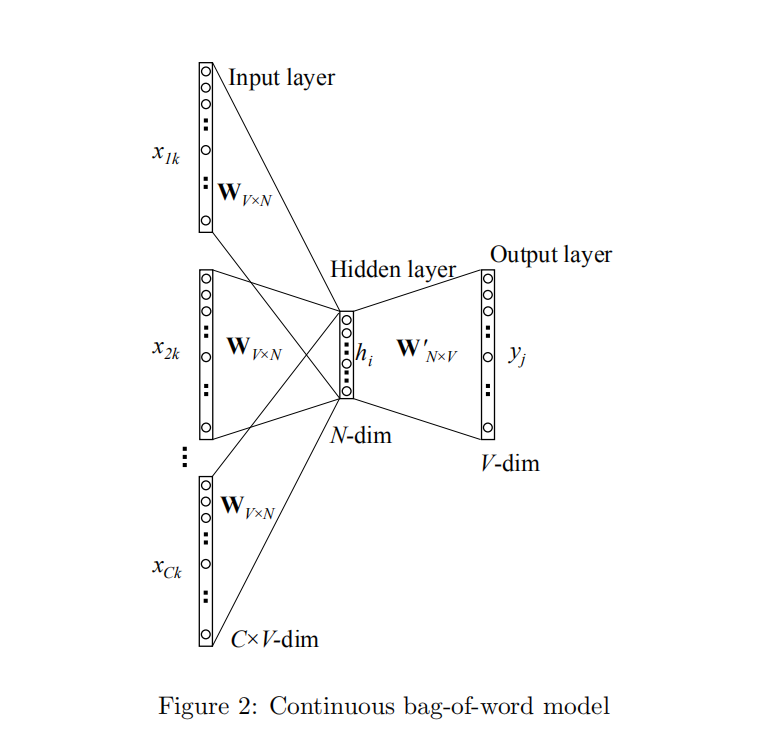
這種學習嵌入層的方法需要大量的訓練數據並且可能很慢，但是確實能學習到針對特定文本數據和NLP任務的詞嵌入。

* 1. **Word2vec**
     1. Continuous Bag-of-Word Model（CBOW）

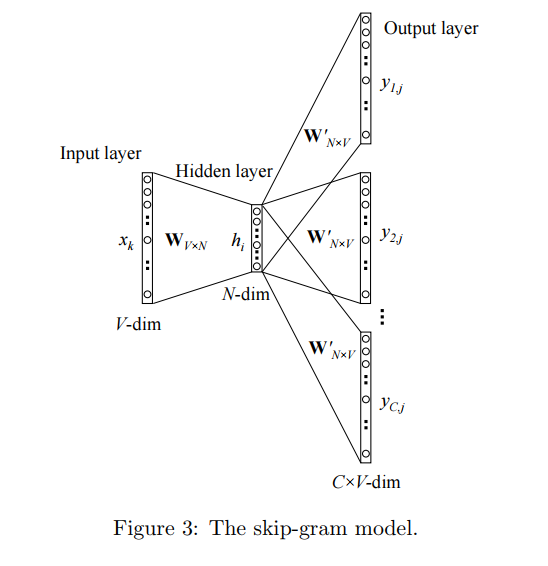
=> 有多個詞彙或單個詞彙組合

連續詞袋模型做出這樣的假設一個單詞出現的可能性與其上下文的單詞組成有關，這意味著對於CBOW模型我們可以通過給定的上下文單詞環境來預測一個目標單詞，就像一個二元模型。

當給予附近單詞，來預測中心單詞

對應這裡就是給予今天/不錯來預測中心詞是天氣的概率。

* + 1. skip-gram模型

skip-gram與上一章介紹的CBOW正好相反，skip-gram是通過中心單詞（target word）來預測其上下文的單詞（context words）。

* 1. **GloVe: Global Vectors for Word Representation**
  2. **fastText**

fastText的核心思想就是：將整篇文檔的詞及n-gram向量疊加平均得

到文檔向量，然後使用文檔向量做softmax多分類。這中間涉及到兩個技巧：字符級n-gram特徵的引入以及分層Softmax分類。FastText 對原始 word2vec 向量的主要改進是包含了字符 n-gram，它允許為沒有出現在訓練數據中的單詞計算單詞表示。

* 1. **什麼是TF-IDF算法?**

TF-IDF(term frequency–inverse document frequency)是一種用於信息檢索與數據挖掘的常用加權技術，常用於挖掘文章中的關鍵詞，而且算法簡單高效，常被工業用於最開始的文本數據清洗。

TF-IDF有兩層意思，一層是"詞頻"（Term Frequency，縮寫為TF），另一層是"逆文檔頻率"（Inverse Document Frequency，縮寫為IDF）。

假設我們現在有一片長文叫做《量化系統架構設計》詞頻高在文章中往往是停用詞，“的”，“是”，“了”等，這些在文檔中最常見但對結果毫無幫助、需要過濾掉的詞，用TF可以統計到這些停用詞並把它們過濾。當高頻詞過濾後就只需考慮剩下的有實際意義的詞。

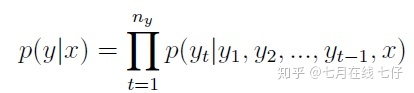
但這樣又會遇到了另一個問題，我們可能發現"量化"、"系統"、"架構"這三個詞的出現次數一樣多。這是不是意味著，作為關鍵詞，它們的重要性是一樣的？事實上系統應該在其他文章比較常見，所以在關鍵詞排序上，“量化”和“架構”應該排在“系統”前面，這個時候就需要IDF，IDF會給常見的詞較小的權重，它的大小與一個詞的常見程度成反比。

當有TF(詞頻)和IDF(逆文檔頻率)後，將這兩個詞相乘，就能得到一個詞的TF-IDF的值。某個詞在文章中的TF-IDF越大，那麼一般而言這個詞在這篇文章的重要性會越高，所以通過計算文章中各個詞的TF-IDF，由大到小排序，排在最前面的幾個詞，就是該文章的關鍵詞。

* 1. **如何理解Seq2Seq Attention模型 ?**

seq2seq是一個Encoder–Decoder結構的網絡，它的輸入是一個序列，輸出也是一個序列，Encoder中將一個可變長度的信號序列變為固定長度的向量表達，Decoder將這個固定長度的向量變成可變長度的目標的信號序列。

首先，我們使用x={x1，x2，…，xn}代表輸入的語句，y={y1,y2,…,yn}代表輸出的語句，yt代表當前輸出詞。在理解seq2seq的過程中，我們要牢記我們的目標是：



即輸出的yt不僅依賴之前的輸出{y1,y2,…,yt−1}，還依賴輸入語句x，模型再怎麼變化都是在上述公式的約束之下。

* 1. **關於意圖式 (Intent-based) 跟流程式 (flow-based) Chatbot**

<https://medium.com/botbonnie/nlp-%E9%97%9C%E6%96%BC%E6%84%8F%E5%9C%96%E5%BC%8F-intent-based-%E8%B7%9F%E6%B5%81%E7%A8%8B%E5%BC%8F-flow-based-chatbot-1555fbfc322c>

