1. Basic machine learning concepts
2. Comparison of supervised learning, unsupervised clustering, and data query.(W2\_ML\_Alg\_v8 p.8.9)

: Supervised learning (監督式學習):有label

1. 它是指從訓練數據集中建立一個模型，並根據這個模型推斷出新的案例。
2. 訓練數據由輸入對象和預期輸出（標籤）組成，可以是連續值或預測分類標籤。

unsupervised clustering(非監督式學習): 無label

1. 是指給定事先未標註的數據集，自動對輸入數據集進行分類或分組。

data query: 下sql 語法抓資料，不知道要考什麼。

1. Accuracy vs. good quality for AI model

Accuracy: 是一個值與其真實值的接近程度

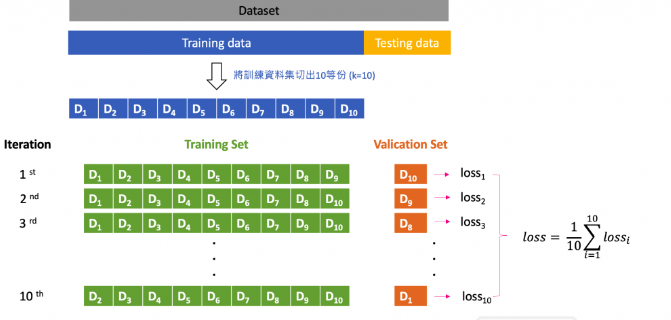
Quality: 模組滿足需求的程度

1. Over fitting v.s Under fitting

:Over fitting 在訓練時表現佳，測試時表現差; Under fitting在訓練時就很差。

1. K-fold cross-validation

: 在 K-Fold 的方法中我們會將資料切分為 K 等份，K 是由我們自由調控的，以下圖為例：假設我們設定 K=10，也就是將訓練集切割為十等份。這意味著相同的模型要訓練十次，每一次的訓練都會從這十等份挑選其中九等份作為訓練資料，剩下一等份未參與訓練並作為驗證集。因此訓練十回將會有十個不同驗證集的 Error，這個 Error 通常我們會稱作 loss 也就是模型評估方式。模型評估方式有很多種，以回歸問題來說就有 MSE、MAE、RMSE...等。最終把這十次的 loss 加總起來取平均就可以當成最終結果。透過這種方式，不同分組訓練的結果進行平均來減少方差，因此模型的性能對數據的劃分就不會那麼敏感。



1. Decision Tree and Random Forest(W2\_ML\_Alg\_v8 ch3、W3\_Scikit-learn ch2)

: Decision Tree:

1. 用於分類和回歸的監督學習方法
2. 通過學習從數據特徵推斷出的簡單決策規則來預測目標變量的值。
3. 推理算法: 以自上而下遞歸的方式建立樹結構，無法處理連續數值，必須先轉換數值屬性
4. 生成決策樹:
5. 建立樹狀結構，依照二分類法不斷執行。
6. 移除造成noise或例外的分支(修剪樹枝)
7. 決策樹的目的: 分類未知的樣本
8. 決策樹停止條件:
   1. 當分支的子集中的所有樣本都屬於同一類別時
   2. 所有的屬性都窮盡了，通過多數表決的方式，用樣本數較多的類來表示這個葉節點。
   3. 選擇某個屬性後，生成的某個分支根本沒有測試樣本
9. 節點特徵順序的三種常用算法:(運算過程去看W2\_ML\_Alg\_v8 p.40-45)
   1. ID3 Algorithm: information gain 最大的特徵優先。
   2. C4.5 algorithm: 訊息貨幣化(?)Gain ratio。
   3. CART algorithm: Gini 最小的特徵優先。
10. 決策樹優點與缺點
    1. 優點: 直觀，訓練快速。
    2. 缺點: 易忽略屬性相關性，不限制停止條件時會overfitting。

: Random Forest:

* + 1. 一個包含多個決策樹的分類器
    2. Random Forest Process:
       - 1. 隨機取n個樣本(有放回)
         2. 從所有特徵中隨機選取k個特徵，對於選取的樣本，利用這些特徵構建決策樹
         3. 重複上述步驟m次，生成m棵決策樹，形成隨機森林
         4. 每棵樹決策後，最後投票決定類別
    3. 優點:
       - 1. 無需特徵選擇就可以處理非常高維的數據
         2. 可以判斷不同特徵之間的相互作用
         3. 可以判斷特徵的重要性
         4. 訓練速度快，容易做並行化方法
         5. 採樣過程的隨機性，不易過度學習(較不會overfitting)
    4. 缺點:
       - 1. 在某些雜訊或回歸上會過度學習
         2. 對於不同價值屬性的數據，價值劃分越多的屬性對隨機森林的影響越大。

1. GradientBoosting Trees

: <https://medium.com/@cwchang/gradient-boosting-%E7%B0%A1%E4%BB%8B-f3a578ae7205>

: https://towardsdatascience.com/introduction-to-boosted-trees-2692b6653b53

1. Regression(W2\_ML\_Alg\_v8 ch4)
2. Linear regression & Logistic regression
   * 1. Linear regression :
        + 1. 一元一次方程式y = ax+b(a=斜率,b=y截距)
          2. 估計最佳擬合直線
     2. Logistic regression
        + 1. Logistic回歸中討論的變量Y主要是分類變量，尤其是分為兩類的變量（例如：是或否，是或否等）。
     3. Linear regression v.s Logistic regression
        + 1. 對連續數據使用回歸模型來預測數據
          2. Linear 中的Y通常是連續變量。
          3. Logistic中的Y主要是分類變量
          4. 示意圖



* + 1. 與perceptron區別:
       - 1. activation function is a Sigmoid Function
         2. the Loss function is a Cross entropy error function(W2 p.64), which also has a Regularization Term, or a Penalty Term, to avoid overfitting.
    2. 正規化(Regularization)
       - 1. 正則化在學習時給予懲罰，使決策邊界更平滑
         2. 目標函數 = 損失函數之和 + 正則化項
         3. L2正則化：使用權重參數的平方值作為損失函數的懲罰項
         4. L1正則化：使用權重參數的絕對值作為損失函數的懲罰
    3. Logistic regression 優點:
       - 1. 易於實現，廣泛應用於工業問題
         2. 計算量很小，速度很快，佔用存儲資源少
         3. 方便觀察樣本概率得分
         4. 計算成本低，易於理解和實施
    4. Logistic regression 缺點:
       - 1. 當特徵空間很大時，Logis 回歸的表現不是很好
         2. 容易under fitting，一般不是很準確
         3. 不能很好地處理大量的多類特徵或變量
         4. 只能處理兩個分類問題並且必須是線性可分的
         5. 對於非線性特徵，需要進行變換

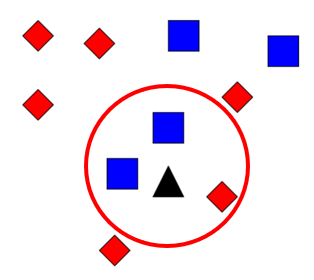
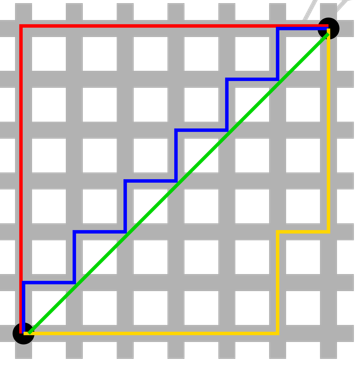
1. Correlation for features(variables)

: 相關性是衡量兩個或多個變量之間線性關係的指標。通過相關性，我們可以從一個變量預測另一個變量。使用相關性進行特徵選擇背後的邏輯是好的變量與目標高度相關

1. SVM(w2 ch5、 w3 ch3)
   * 1. 監督式機器學習，需要預先定義每個組的類型。
     2. 主要用於二元分類問題
     3. 平面無法分割的就提高維度，在高維空間中尋找一個超平面作為類別分割(W2 p.76)
     4. SVM的重要優點是可以處理線性不可分的情況
     5. SVM的基本概念很簡單，就是找到一個決策邊界，使兩個類別之間的margin最大化，從而可以將它們分開。
     6. 優點:
        + 1. 可以解決高維問題，即大特徵空間
          2. 解決少樣本機器學習問題
          3. 能夠處理非線性特徵的相互作用缺點
     7. 缺點:
        + 1. 一般SVM只支持二分類
          2. 當觀測樣本較多時，效率不是很高
          3. 非線性問題沒有通用解，有時很難找到合適的核函數
2. Kernel function(w2 p.77-79、W3 p.43)

使用線性分類器來解決非線性問題，將線性不可分數據轉換為線性可分數據，從而為可能的輸出找到最佳邊界。  
ppt原文: Dimensional conversion can be calculated using vector products, that is, kernel functions.

1. KNN(w2 ch6、w3 ch4)
   * 1. k-NN是通過k個最近鄰的分類結果來確定物體的類別(多票數)
     2. K值會影響預測效率。
     3. 假設K=3，意旨挑選三個最近的鄰居，並用多數決決定他的分類。依此類推。

* + 1. 演算法過程
       - 1. 決定K值
         2. 計算每個測試樣本和訓練樣本之間的所有距離
         3. 挑選K個鄰居
         4. 統計這個k近鄰的分類
         5. 按分類數最多的就是這個測試樣本的類別
         6. 重複2-5次，直到所有樣本都被分類完
    2. 計算距離公式
       - 1. Euclidean Distance(歐基里得距離):兩點相減的平方總合開根號。(綠線)
         2. Manhattan Distance(曼哈頓距離): 兩點相減取絕對值相加。(紅線)
         3. Common distance functions(名可夫斯基距離):p=1 是曼哈頓、p=2是歐基里得
         4. 小小計算: A(0,0) B(6,6)

Euclidean = =6

Manhattan = = 12

* + 1. 找最近k個鄰居演算法
       - 1. Ball Tree:遞歸地將數據劃分為由中心C和半徑r定義的節點，使得節點中的每個點都位於由r和C定義的超球內。
         2. Kd Tree: 是一棵二元樹，其中每個節點都是一個k維的點。所有非葉節點都可以看作一個超平面，將空間劃分為兩個半空間。
    2. 優點:簡單，無須訓練，無須估計參數
    3. 缺點:
       - 1. 樣本數多時，計算量大。
         2. 樣本數不平均時，預測偏差比較大。
         3. 超過3個類別，往往會出現平局，平局應該取默認值，這樣會讓K-NN失去精度

1. Bayes rules(w2 ch7、w3 ch5)
   * 1. Naïve Bayes' Classifier是基於特徵相互獨立的假設。
     2. 文檔分類是根據詞頻判斷文檔的類別或其他（如垃圾郵件、合法性、體育或政治等）的問題
     3. 各種算術請見ppt (w2 p.102-118)
     4. 優點: 易於實施，在大多數情況下取得了良好的效果
     5. 缺點: 現實中特徵是相互依賴的，會損失準確率，若要解決這些相依性條件的問題，需使用Bayesian Belief Networks。
     6. 又是一連串算數(w2 p.120-128)
2. 各類參數
   * 1. Logistic Regression

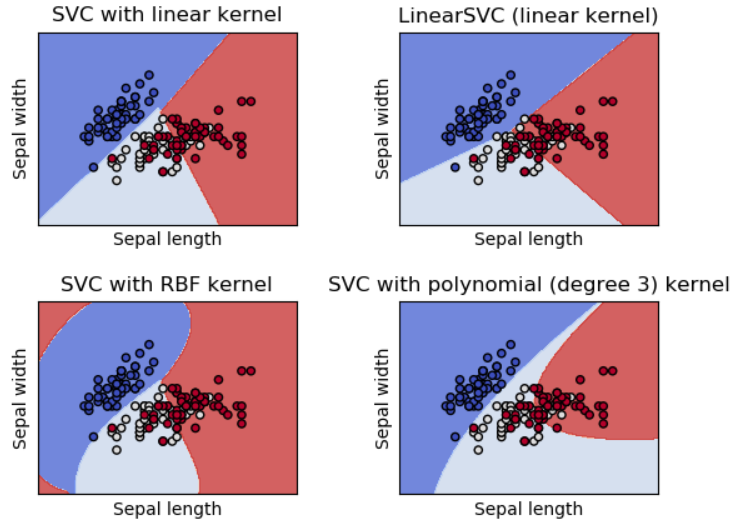
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| Penalty | string | 正則化過程中損失函數(loss)的懲罰規範。 | L2 | L1、 none elasticnet、 |
| C | float | 正則化強度與C成反比，必須為正數，C越大越容易過度擬合 | 1.0 |  |
| class\_weight | string | 類別的權重 | None | Dict、  balanced |
| solver | Int/float | 優化問題的算法。  (a) 數據量少: liblinear  (b) 數據量大: sag、saga  (c) 多類別問題: newton-cg、 sag、 saga、lbfgs。  (d) newton-cg、lbfgs、sag、saga可以處理 L2 或none。  (e) liblinear 和 saga 可以處理 L1 懲罰  (f) saga支持 elasticnet 懲罰  (g) iblinear不支持none | lbfgs | newton-cg、  liblinear、  sag、saga |
| multi\_class | string | (a)Ovr:二元分類  (b)solver='liblinear' 時，multinomial不可用。  (c)如果數據是二分類數據或者solver='liblinear'，設置'auto'時，會自動選擇'ovr'；否則，將自動選'multinomial' | auto | Ovr、multinomial |
| n\_jobs | Int、none | 並行計算使用的處理器數量，默認為1 如果設置為-1，將使用所有處理器 | none |  |
| verbose | int | 用於啟用/禁用迭代中間輸出的日誌記錄 | 0 | 1 |

* + 1. Random Forest Classifier

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| n\_estimators | integer | 子樹的數量。更多的子樹可以讓模型有更好的性能，但同時也會增加計算量 | 100 |  |
| criterion | float | CART樹劃分時對特徵的評價標準 | Gini | Gain、entropy |
| max\_depth | string | 樹的最大深度，如果沒有，則在不限制深度的情況下構建子樹，或者直到所有葉子包含的樣本少於 min\_samples\_split 樣本。 | None |  |
| min\_samples\_split | Int/float | 拆分內部節點所需的最小樣本數。  (a) int:該值為最小量  (b) float: 該值\*樣本數為最小量。 | 2 |  |
| max\_features | Int/float/ string/ None | 找到最佳分割時要考慮的特徵數量。 | auto | Sqrt(跟 suto一樣)、log2、None |
| bootstrap | boolean | 構建樹時的採樣方式  (a)True:使用隨機樣本的子集來構建每棵樹  (b)false:使用整個數據集用於構建每棵樹 | True | False |

* + 1. Support Vector Classifier

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| C | float | 正則化強度與C成反比，必須為正數，C越大越容易過度擬合 | 1.0 |  |
| kernel | string | 跟切割曲線有關  (a)linear:線性函數  (b)poly：多項式函數  (c)rbf：常態分佈曲線  (d)sigmod：sigmod 函數  (e)precomputed：內核矩陣（自定義矩陣） | rbf | Linear、poly、sigmoid、precomputed |
| degree | Int |  | 3 |  |
| gamma | String/  float | Kernel = rbf,poly,sigmoid | scale | auto |
| Coef() | Int、none | 設定整條線的位移 | none |  |
| class\_weight | 重點是Kernel，懶得寫了。 | | | |
| Decision \_function \_shape |



* + 1. K neighbors Classifier

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| n\_neighbor | int | 鄰居數 | 5 |  |
| weights | float | 用來預測  Uniform:權重一樣  Distance:越近的權重用大 | uniform | distance |
| algorithm | string | 計算最近鄰居的演算法  Auto會自動配對最適合的演算法 | None | ball\_tree、kd\_tree 、brute、auto |
| metric | String/ callable | 計算距離的公式 | minkowski | minkowski |
| p | integer | 1:曼哈頓  2:歐基里得 | 2 | Ovr、multinomial |
| n\_jobs | Int、none | 並行計算使用的處理器數量，默認為1 如果設置為-1，將使用所有處理器 | none |  |

* + 1. Gaussian NB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| priors | array-like | ?????? | | |
| var\_smoothing | float |  | 1e-9 |  |

* + 1. Multinomial NB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | type | means | default | Other value |
| alpha | int | Laplas smoothing，設為零的話就不會smoothing | 1.0 |  |
| fit\_prior | boolean | 是否考慮事前機率 | true | false |
| class\_prior | array-like |  | None |  |