支持信賴

支持度:count(x&y)/all
信賴度:count(x&y)/count(x)

C1L1C2L2

The Apriori Algorithm
分析資料筆數-考慮是否達到門檻-重新列出-根據新表寫出可能的組合-分析資料筆數.....
(c1-l1-c2-l2…)
(不存在新表內的元素不能寫進可能的組合)

HASH

Mod分兩部分
利用mod 將資料填進hash表『ac=13 13mod7=6』
格子內出現的組數
刪除元素後的排列並對應格子內出現的組數
最終低於門檻者再刪除

決策樹

決策樹：
由上而下遞迴分割建構的方法，一開始所有資訊都在樹根，有絕對明確的的屬性。(若為連續性資料，則將所有資料事先離散化。)選擇屬性後遞迴分割，基於啟發式或統計量來選測試屬性。

3個stop：

1. All samples for a given node belong to the same class所有樣本的屬同一類
2. There are no remaining attributes for further partitioning屬性用完仍無法分類完畢（解法：多數決）
3. There are no samples left沒有樣本

貝氏定理

p(A|B)=P(B|A)P(A)/P(B)
P(buy,yes)=9/14　　p(buy,no)=5/14
p(屬性｜yes/no)
P(x|yes/no) 所有屬性的yes/no相乘
p(x|yes/no)*p(buy|yes/no)
Naïve assumption ：較高者

k_means

把n個點（可以是樣本的一次觀察或一個實體）劃分到k個群集中，使得每個點都屬於離他最近的均值（此即群集中心）對應的群集，以之作為群集的標準。
Given k, the k-means algorithm is implemented in 4 steps:

1. Partition objects into k nonempty subsets把所有資料分成k個子集合（找最近的點）

2. Compute seed points as the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.歸類一群後重新計算質心，質心是分群資料裡的中心點
3. Assign each object to the cluster with the nearest seed point. 重新分群後再觀察質心變化
4. Go back to Step 2, stop when no more new assignment.重複步驟2,直到質心沒有改變

缺點：

1. Applicable only when mean is defined, then what about categorical data? Replace means of clusters with modes, which is the most representative (frequent) object.僅在定義平均值時適用，分類數據呢？最具代表性的分群模式去取代平均
2. Need to specify k, the number of clusters, in advance 必須事先給定k
3. The k-means algorithm is sensitive to outliers since an extremely large value may substantially distort the centroid.對極端值很敏感，容易因為極值而嚴重扭曲，使質心失真

Density-Based Clustering Methods

密度可達：對於樣本集合D，給定一串樣本點p1,p2….pn，p= p1,q= pn,假如對象pi從pi-1直接密度可達，那麼對象q從對象p密度可達。

密度相連：存在樣本集合D中的一點o，如果對象o到對象p和對象q都是密度可達的，那麼p和q密度相聯。

可以發現，密度可達是直接密度可達的傳遞閉包，並且這種關系是非對稱的。密度相連是對稱關系。DBSCAN目的是找到密度相連對象的最大集合。

DBSCN通過檢查數據及中每點的eps領域來搜尋cluster 如果p點的eps領域包含的點多餘MinPts個，則創建一個以p為核心對象的cluster。DBSCAN 迭代的劇集從這些核心對象直接密度可達的對象，這個過程可能涉及一些密度的可達cluster的合併。當沒有新的點添加到任何cluster時，該過程結束

The vector model

判斷兩份檔案的相似性

兩筆資料的夾角越小相似度越高，相似度越高排名越前面

$Sim(q,d_j) = cos\theta = [vecd_j \bullet vecq]/(|d_j| * |q|) = [\sum w_ij * w_iq]/|d_j| * |q|)$

$W_ij = tf(i,j) * idf(i)$

證明：

Let,

N be the total number of docs in the collection 全部文件數

n_i be the number of docs which contain k_i 文件包含k_i的個數

$freq(i,j)$ raw frequency of k_i within d_j

A normalized tf factor is given by

$tf(i,j) = freq(i,j)/max(freq(l.j))$

where the maximum is computed over all terms which occur within the document d_j

The idf factor is computed as

$idf(i) = log(N/ni)$

the log is used to make the values of tf and idf comparable.
The best term-weighting schemes use weights which are give by
$w_ij = freq(i,j)/max(freq(l,i)) * log(N/n_i)$

Inverted file

1. Vocabulary search: the words present in the query are searched in the vocabulary詞彙檔出現在哪
2. Retrieval occurrences: the lists of the occurrences of all words found are retrieved找出出現在哪些文章以及頻率跟權重
3. Manipulation of occurrences: the occurrences are processed to solve the query解決查詢資料