Data Mining Lab2(Baysian Classification)

* 實際資料庫: iris.arff
* 資料屬性類別(平均值&標準差): iris\_mean\_variable.txt
* 輸出檔案: BaysianClassificationResult.txt
* 執行檔: BaysianClassification.java
* 將資料夾”DataMininghw2”放置eclipse專案區中並匯入執行(或使用cmd編譯及執行BaysianClassification.java檔)。
* Naive Bayes Classifiers (單純貝氏分類器)：

如果我們假設在給定的資料集中，每一維的資料都是獨立的，在此假設下，每一類資料的PDF可以簡化成此類資料在每一維的PDF的乘積。換句話說，我們可以先算出每一類資料在每一個維度所對應的PDF，然後將同一類資料的數個PDF進行連乘，就可以得到這一類資料的PDF。我們的假設可以使用數學式子來表示如下：

**p(X|C) = P(X1|C)P(X2|C) ... P(Xd|C)**

其中 X = [, , ..., ] 是一個特徵向量，而 C 代表一個特定類別。

* + 就離散型資料欄位(屬性)而言：

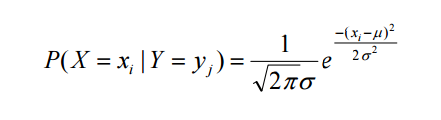
對於某個離散型資料欄位 X 而言，其條件機率P(X =|Y = y)是指在 y類別中的部份訓練資料，具有特別的屬性值 。

* + 就連續型資料欄位(屬性)而言：

可以假設該連續變數符合某個機率分配，然後使用訓練資料來估計該分配的參數。

常態分配(高斯分配)最常被用來表示連續變數的類別條件機率。

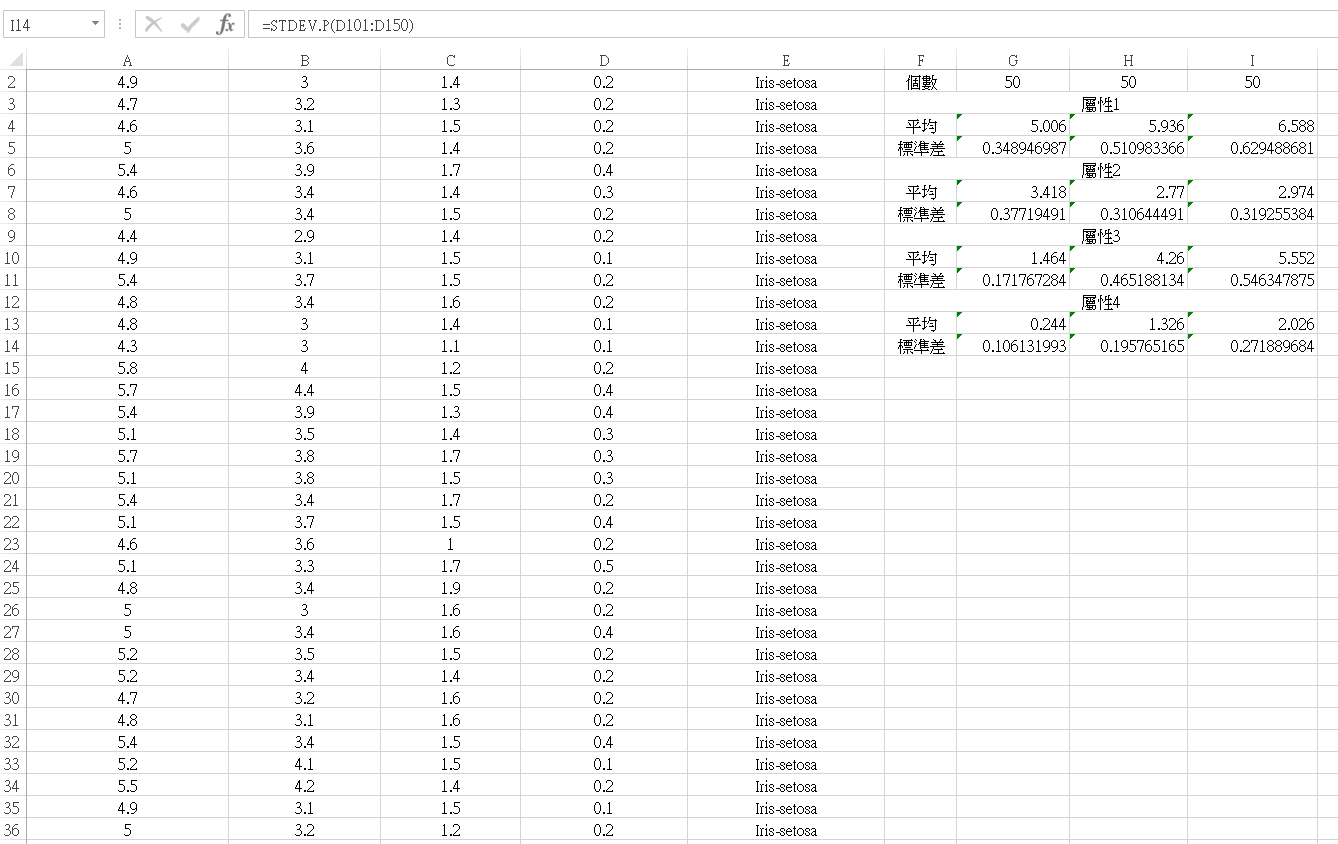
* + - 該分配有兩個參數：平均數μ與變異數
    - 對於每個類別而言，連續型資料欄位 X的類別條件機率如下：



* + - 參數μ可以經由連續型資料欄位 X中、屬於類別之樣本平均數

估計而來。而可以從樣本變異數估計而來。

* Iris資料集屬性與類別之間平均值與標準差計算
  + 使用excel軟體，將Iris資料打入excel檔中，再利用excel軟體的計算平均值與標準差的功能，計算出四個屬性(sepallength,sepalwidth,petallength,petalwidth)分別與三個類別(Iris-setosa, Iris-versicolor,Iris-virginica)之間的平均值與標準差。(可查看IRIS資料.xlsx)
  + 將計算好的平均值與標準差儲存到iris\_mean\_variable.txt中。



* Baysian Classification程式碼(BaysianClassification.java)

程式主要結構:

* + Import檔: java.io.\*
  + 主類別: BaysianClassification
    - 靜態變數: InputFileName //輸入檔名

MVInputFileName //輸入檔名

OutputFileName //輸出檔名

sepallength[] //iris資料集屬性(sepallength)陣列

sepalwidth[] //iris資料集屬性(sepalwidth)陣列

petallength[] //iris資料集屬性(petallength)陣列

petalwidth[] //iris資料集屬性(petalwidth)陣列

class1[] //iris資料集類別陣列

mean\_variable[][] //iris資料集屬性與類別之間平均值標準差(二維陣列)

Result[] //iris資料集Mining結果陣列

two\_pi\_sqrt //圓周率乘2開根號

exp //自然指數

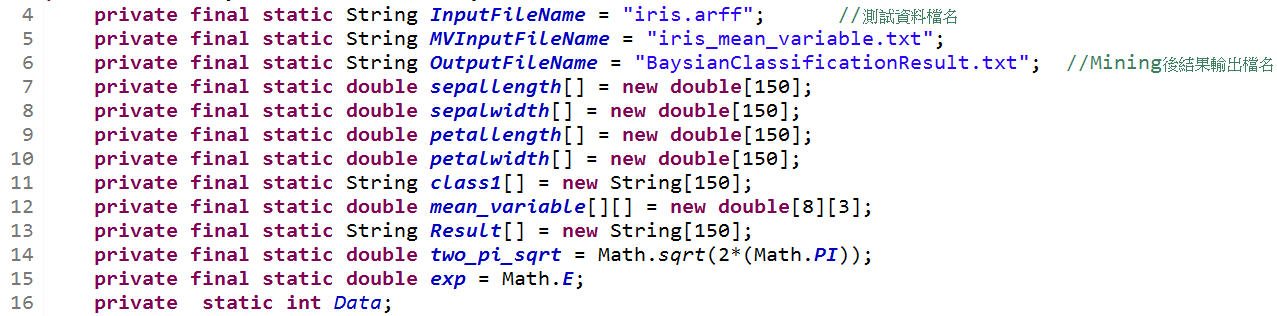
Data //記錄iris資料筆數

* + - 函式: main() //程式進入點

sop() //列印方法

compu\_iris() //計算資料屬於每個類別的機率以及判斷資料所屬類別

* Import檔
* java.io.\*: 檔案IO所需的套件包。(可視需求import會用到的子類別進來就好)
* 主類別BaysianClassification-靜態變數

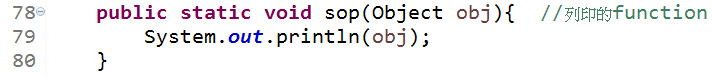


* 第4行: private final static String **InputFileName** = "retail.txt"; 設置要輸入的檔案名稱。
* 第4行: private final static String **InputFileName** = "retail.txt"; 設置要輸入的檔案名稱。
* 第5行: private final static String **OutputFileName** = "MiningResult.txt"; 設置Mining的結果要寫出的檔案名稱。
* 主類別BaysianClassification-函式
* main()



1. 第27行: 程式的進入點，throws IOException為輸入輸出丟出例外。
2. 第28~29行: 宣告讀寫檔案以及緩衝讀寫的變數，處理檔案輸入以及輸出文字型態(Big5)。
3. 第30行: 宣告字串形態變數(Line,TempString)。
4. 第31行: 宣告字串陣列TempArray，陣列大小為資料筆的屬性數。
5. 第32行: 宣告一個ArrayList(TestDataList)泛型為字串。
6. 第33~39行: 使用while迴圈將資料一行一行讀進來，用Line接收每一行資料，直到讀到的是null則跳出迴圈。用TempString接收Line的每一行內容。將TempString(每一行資料)以","作為切割點進行切割，存到TempArray陣列中。最後用for迴圈將TempArray的資料依序存到TestDataList中。
7. 第40行: 宣告整數型態變數TestDataCount，為TestDataList除以資料筆屬性個數(資料筆個數)。
8. 第41行: 宣告整數型態變數count為0。
9. 第42行: 宣告double型態二維陣列(TestData)，陣列大小為資料筆個數，長度為每個資料筆的屬性個數。
10. 第43~48行: 利用兩個(巢狀)for迴圈，利用count作為index取出TestDataList的字串，再將字串轉換成double依序存到二維陣列中。
11. 第49行: 宣告一個KMeans類別變數km，將TestData作為引數傳入。
12. 第50行: 利用KMeans類別變數km，呼叫SetNumOfClusters方法設定分群數。
13. 第51行: 宣告整數型態二維陣列(Results)，接收km的DoPartition()所回傳的整數型態二維陣列(每一群集的資料筆編號)。
14. 第52行: 宣告double型態二維陣列(TestDataDistance)，接收km的GetDisTestData()所回傳的double型態二維陣列(每一資料筆分別與原點之間的距離平方)。
15. 第53行: 宣告整數型態陣列(DataInCluster)，接收km的GetNumOfObservations()所回傳的整數型態陣列(每一群集的資料筆數)。
16. 第54~74行: 將Mining結果(原始資料輸入檔名，每一資料筆分別與原點之間的距離平方，分群數，每一群集的資料筆數，每一群集的資料筆編號以及座標)寫出到設定的檔案。
17. 第75~76行: 寫入以及寫出的檔案關閉。
18. 第77行: 列印Mining結束的訊息。

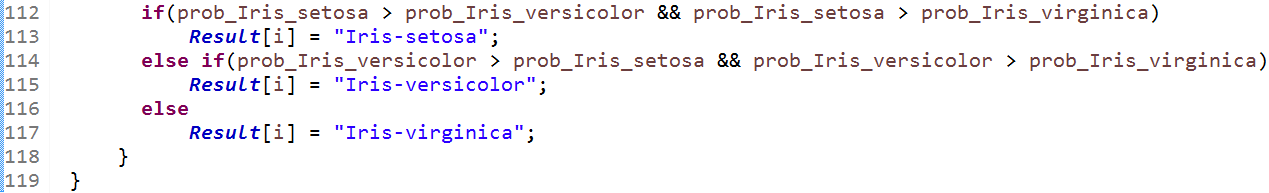
* sop()



1. 第78~80行: sop()函式執行列印功能。

* compu\_iris()





1. 第83行: 宣告double二維陣列(TestData)，儲存原始資料。
2. 第84行: 宣告double二維陣列(DisTestData)，儲存每一筆資料與原點之間的距離平方。
3. 第85行: 宣告整數型態變數(NumOfClusters)，儲存分群數。
4. 第86行: 宣告整數型態變數(TestCases)，儲存資料筆數。
5. 第87行: 宣告整數型態變數(TestAttributes)，儲存每筆資料的屬性數。
6. 第88行: 宣告double二維陣列(Centroid)，儲存分群後每群的質心座標。
7. 第89行: 宣告整數二維陣列(AryResultClusters)，儲存分群後每一群的資料筆編號。
8. 第90行: 宣告整數陣列(ClusterNo)，儲存分群後每一群的個數。

* 執行結果