s973350，杜明軒，資工 4C

實作 **KNN(k-nearest neighbor)-**以預測學生是否會被當掉為例

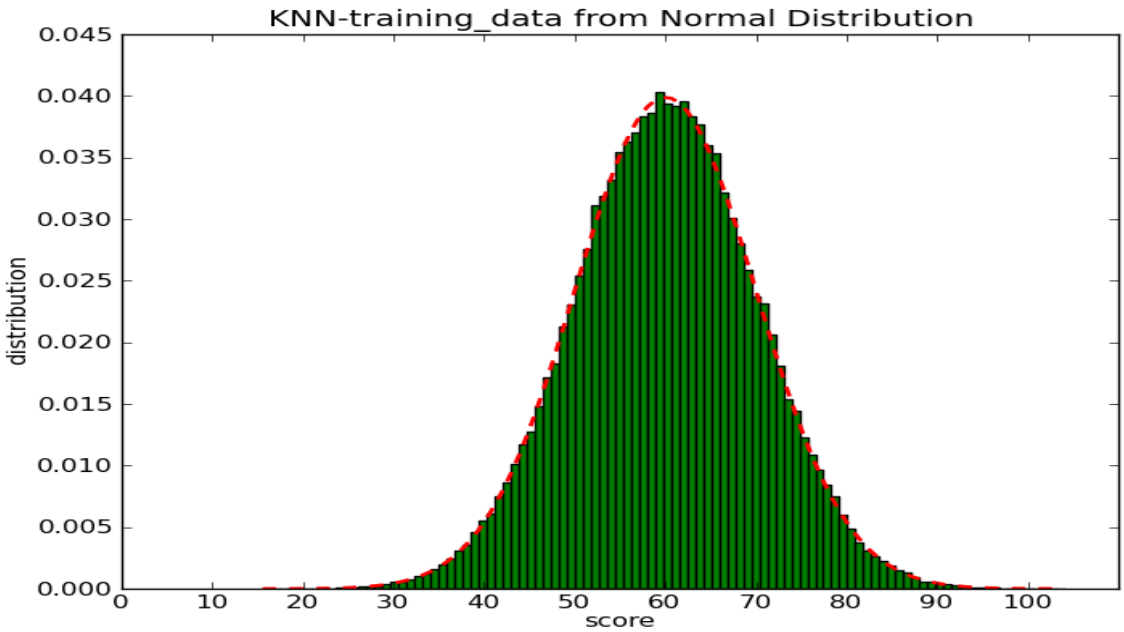
KNN 屬於監督式分類演算法(Supervised Classfication Algorithm)，屬於實作簡單且預測相當準確的演算法，在 2008 年的時候，Xindong 等人歸納了 10 個在 Data Mining 最具代表性的演算法[1]，其中一個提到的演算法就是 KNN，目前也有許多基於 KNN 的改良演算法不斷被提出來。

本次報告以 Python2.6 實作一個簡單的 KNN 為例，藉由 KNN 來預測學生的分數是否可以拿到該課堂的學分，我們期望 KNN 可以判斷分數為六十分以上的同學可以拿到該課堂的學分，在一維的情況下，我們採用 *Euclidean Distance* 來當作點跟點之間的距離，透過這次的實作及實驗讓我了解 KNN 具體的過程。

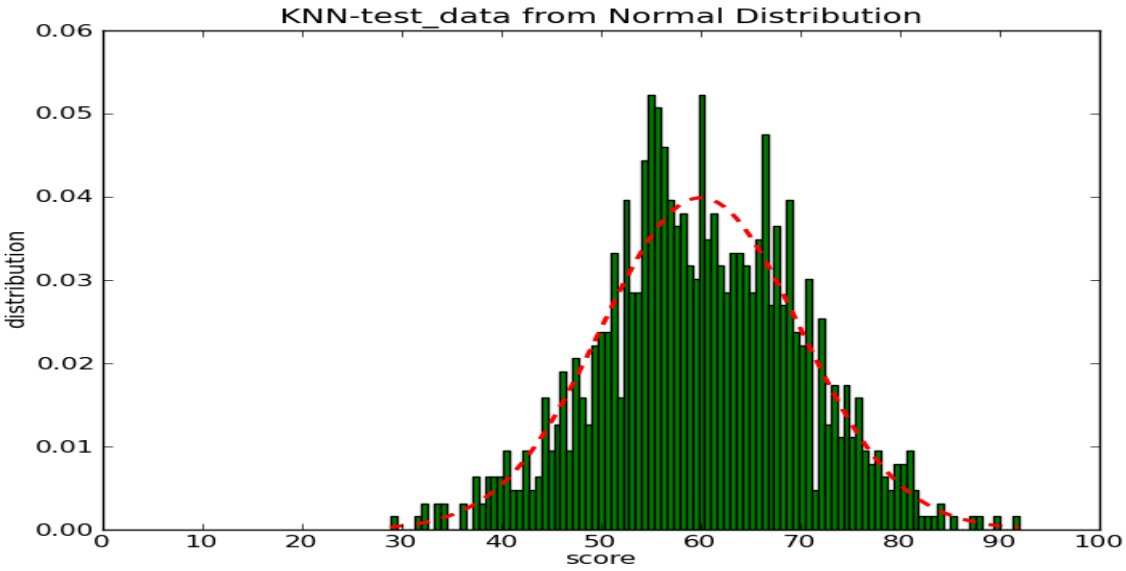
在實驗之前，我們必須先產生一組 Training Data 來訓練 KNN，我們對於一個 *Normal Distribution* 隨機均勻取樣十萬筆資料當作訓練資料集，其中 *Normal Distribution* 的參數設定為 Mean=60，Standard deviation=10。會採用 *Normal Distribution* 的原因在於我們希望我們的資料是公平均勻分佈且符合現實生活中的情況。由於我們只有兩個類別(pass or fail)，因此我們必須假設 KNN 的參數 k 為奇數，在此我們假設 k=101，以避免兩個類別的鄰居數一樣多的情況。

從實驗結果我們發現，由於我們設定 *Normal Distribution* 的 mean=60，表示所取樣出來的 Training Data 中大約有一半的學生會 pass(60 分以上)，一半的學生會 fail(60 分以上)，在此必須強調的是，我們並沒有告訴 KNN 一件事實:60 分以上的會 pass，60 分以下的會 fail。我們給 KNN 的只有資料的內容(學生的分數)跟資料所對應的類別(pass or fail)。但是 KNN 也能精準的判斷該學生是否可以 pass，這就是監督式分類演算法的特色。當然，KNN 也有誤判的情形，這與我們採用 Normal Distribution 以及其設定的 mean 是很有關係的，會使得有些學生莫名其妙被 KNN 給當掉，這可以對 Normal Distribution 做調整，藉以改善 KNN 誤判的情形。

為了要確定我們從 *Normal Distribution* 取樣出來的資料也會服從 *Normal Distribution*，我們把隨機取樣總共十萬筆的 Training Data 的 Histogram 表示如下:



可以發現上圖也是趨近於以 mean=60 為中心，兩邊分別以正負 3 個標準差對稱的 *Normal Distribution*。(似乎也可以由 *Central Limit Theorem* 得知)，紅色虛線部份為 *Normal Distribution*(mean=60,standard deviation=10)的母體分佈。



現在來進行實驗一(exp1.py)，我們對同樣的 Distribution 隨機取樣一千筆資料來作為我們的測試資料集，當然測試資料集也會趨近於 *Normal Distribution*(mean=60,standard Deviation=10)，由於只有一千筆資料，所以不會像上述的 Training Data 吻合的這麼漂亮，如上圖是一個隨機取樣的測試資料集。

我們將隨機取樣的一千筆的測試資料交給 KNN 來做分類，誤判的資料會輸出並且存在 victim.txt 中，若 victim.txt 是空的表示沒有誤判的資料，每次執行程式的時候都會隨機取樣十萬筆訓練資料集以及一千筆的測試資料集。實驗結果幾乎不會有誤判的情形，但是由於我們的訓練資料集是隨機的，所以在某些隨機訓練資料集上面，KNN 在判斷邊界值的時候(例如分數為 60 分的同學)會誤判，這在實驗二(exp2.py)中可以發現。解決方法有兩個，第一個就是把 *normal distribution* 的 mean 取比 60 大一點點，例如取 mean=65，這相當於把整個 distribution 往右邊移動，使得在比 60 大的訓練資料點可以變多，例如[60,70]區間的資料點可以多一些，使得分數在 60 這個資料點的右邊的鄰居(可以拿到學分的訓練資料點)可以變多，因為有較多的鄰居屬於會過(pass)的類別，所以這有機會可以降低邊界值誤判的可能性。第二個解決方法是把較好的訓練資料集保留下來，也就是保留不會讓 KNN 對於邊界值有誤判情形的訓練資料集合，讓 KNN 只固定讀取該訓練資料集。

以下是實驗二(exp2.py)執行的截圖，其中第一個所產生的訓練資料集會使得 KNN 對於 60 分這個邊界值有誤判的情形。但是重新產生訓練資料集之後，對於 60 分這個邊界值又可以正確的判斷。因此可以發現不同的訓練資料集，雖然是從母體參數一樣的 *Normal Distribution* 中取樣，但還是會對同一個資料點有不同的判斷，其中一個解決方法就是保留第二個重新產生的訓練資料集，讓 KNN 固定讀取該資料集。

```
you can press <C-P> to terminate experiment
Random select  100000  samples from Normal Distribution...
Sample from Normal Distribution is done...
The number of  49725  students from samples pass the class
The number of  50275  students from samples fail the class
Please input a score of student to predict if he/she would pass :) 60
the student with score  60  will fail
Regenerate the training data set ? (Y/N)y
Random select  100000  samples from Normal Distribution...
Sample from Normal Distribution is done...
The number of  50341  students from samples pass the class
The number of  49659  students from samples fail the class
Please input a score of student to predict if he/she would pass :) 60
the student with score  60  will pass
```

Reference:

1. http://www.cs.uvm.edu/~icdm/algorithms/10Algorithms-08.pdf

心得：

KNN 其實對於非均勻的資料判斷很差，以這個例子而言的話，假如我們被當掉的訓練資料集的資料點分佈於 20 分到 30 分之間(總共有 10 個點)，會拿到學分的訓練資料集的資料點分佈於 60 分到 90 分之間，其中 60 分到 70 分佔了 20 個點，這樣如果我餵給 KNN 的測試資料點為 50 分到 60 分的話，幾乎會出現誤判的情況。因此訓練資料集的分佈狀況對於 KNN 是非常重要的。：)

如果 k=1 的話，其實不怎麼公平，假設我們的測試資料點是 59，離他最近的訓練資料點為 60，這樣子在 1NN 的情況下，系統會誤判 59 分可以拿到學分，但是其實離 59 分很近的點也有 57、56、55...等等，所以如果延伸 k>1 的情況的話，準確率應當會提升。

PS:

老師如果有任何的 comment 的話，可以聯絡我的信箱: qrnnis2623891@gmail.com