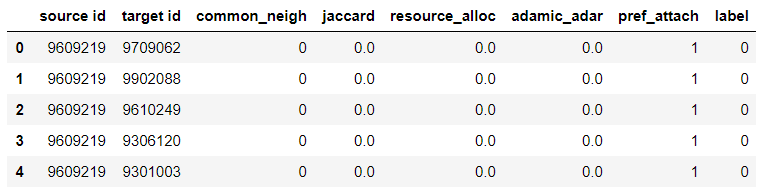
SNA HW1報告

M10615066 方誌賢

1. Your code structure and which algorithm you used.

首先因為這個dataset如果算完全部feature的話實在太久了，所以我先取Period 1 中隨機100筆資料做成network(Graph)，然後將Graph取未連線的邊(non edge)，先做成Dataframe，這個DF代表著label全為0，接著就用P1的Graph算feature(common\_neigh, jaccard\_coefficient之類)，必須要注意的是，也因為是算這些feature，所以出來的值都會是non edge的那些邊的值，算好後把這些值都放入DF中然後label全標0。

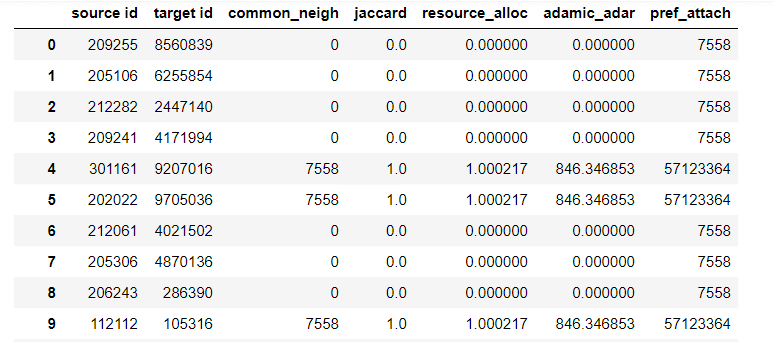


再來就是處理Period 2的部分，我將P2的DATA全當成label 1(未來有被引用的論文)，上面那個DF算完大概有18000多條non edge，也就是說label 0 大約有18000多個。label不能太少，但如果太多的話又會算很久，所以我隨機取了10000筆P2的DATA，然後建網路P2，建完網路後取補圖P2’，接著將我取得那10000筆資料建成第二個DF，然後要算feature。因為要算feature，而且是那些原本10000筆DATA的feature，所以才需要以取補圖的方式算feature(補圖算出來的邊才會是DF中的邊)。簡單來說就是我把P2的每筆DATA都預設他們label是1，然後因為要算feature，必須以補圖的方式才能算回來這些邊的feature，這樣就建完DF2了。

接著將DF2跟DF1合併，接著檢查如果label 0中的邊有重複出現在label 1中，把label 0的邊刪掉，因為這就代表那些邊未來會出現，所以label不該為0，最後將最終的DF\_FINAL順序洗亂，拿去訓練。

訓練用SVM，接著我將P1P2中所有target id全部存成LIST(叫做all target)，之後會用到。訓練完後必須建一個最終的Graph，所以將DF\_FINAL中label全為0的DATA取出(18000多筆)，先將這些建成圖，再求出non edge，會變成當初P1取的那100筆的邊。將DF\_FINAL label全為1的DATA取出建圖，接著再將這個圖做補圖，然後把P1那100筆的邊加進這個網路中，成為最後的網路。會這樣做的原因是，將TEST DATA每筆資料的點取出來後，會跟最後的網路所有的點做比較，如果沒有出現在最後網路的點中的話，會再把TEST DATA中target的點再去跟all target比較有沒有出現在裡面，如果有出現在all target我就假設這個邊未來一定會出現，所以就把TEST DATA那筆改成分類完一定label是1的pair，如果沒出現在裡面，我就把它改成分類完一定是label 0的pair，也因為這些test data的pair其實是還沒算feature的，所以我的最終網路必須那樣建，當出現預設label 一定為1的pair丟到最終網路算feature時，我給的pair是P2補圖後算完feature的pair(反正就是當初在建DF2時的隨便一個邊，因為label是1)，所以建最終網路時為了要讓這個pair的feature丟入SVM結果要是1，我才將DF\_FINAL label全為1的DATA取出建圖，接著再將這個圖做補圖，當成最後網路的一部份，好方便我控制feature。那麼label一定為0的pair我是取最終網路的non edge隨便一個值，反正幾乎所有的non edge算出來的feature都會在分類後label為0。最後如果兩個點都有出現在最終網路的node list中，就照常算feature。如果我P1 P2在當初取的資料越多，那麼最後結果就不會依賴於去比對test的點是否出現在target中。

好反正結論就是最後test data中如果有點沒出現在最後網路中，我就看他是否有出現在過去的target id中，有的會就給他分類完一定是label 1的pair，沒有的話就給label 0的pair，如果有出現在最後網路的點中，就正常算，大概做完test data會如下



然後去分類就可以得到結果了

1. The problems you meet in the HW1 and how do you deal with it.

太多問題了，資料處理花太久時間，因為如果將全部的點建成網路的話邊的數目會到9億多個，當初是卡在標label上面，標完這學期也剛好結束，所以換了一個方法，不過收穫良多，太神啦

1. What thing or concept you have learn in this homework.

當初為了要徹底了解link prediction，去coursera花了一點時間看教學，現在已經了解這些方法(common neighbor, jaccard, resource allocation)背後是怎麼算的了

1. Code execution instructions. (For TAs to run your code if it necessary.)

先執行feature\_process.py 會得到一個標好LABEL的DF

再執行train\_and\_predict.py 去訓練以及得到結果的DF