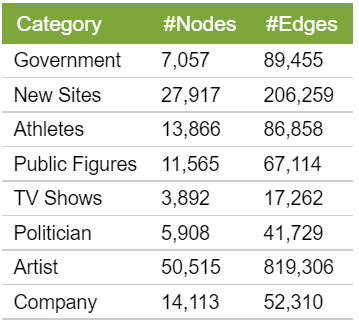
# Facebook Link Prediction Report

第8組

組員: 李彥霖 M11015094、馬葆芸 M11015030、王樸M11015072、魏向晨M11015084

## **Abstract**

　　連結預測是預測網路中兩個實體 (identity)之間是否存在連結的問題。連結預測的示例包括預測[社交網絡中用戶之間的友誼連結、預測](https://en.wikipedia.org/wiki/Social_network)[引文網絡](https://en.wikipedia.org/wiki/Citation_network)中的共同作者連結以及預測[生物網絡](https://en.wikipedia.org/wiki/Biological_network)中基因和蛋白質之間的相互作用。連結預測也可以具有時間方面，其中，給定資料集的快照 (snapshot), 目標是預測資料集在時刻 t+1的連結 ,。在電子商務中，連結預測通常是向用戶推薦商品的子任務。在引文數據庫的策展中，它可用於記錄重複數據刪除。在本文中，我們比較了不同單維度特徵對於連結預測效能的影響，分別採用了 Gradient boosting decision tree 和 Random Forest 兩種互相比較。同時也採用了多維度特徵 GEMSEC, Deep walk 衡量這兩類特徵的實驗結果。

## **Dataset**

　　使用 Snap Facebook pages (2017/11) 公開資料集，節點代表 Facebook 上的公眾粉絲專頁(人物,公司等)，節點之間的邊代表互相按讚的關係。資料經過重新索引以強調匿名性。總共包含的8種不同類型的粉絲專業，各類型之間的資料互相獨立。右圖列出不同類型的資料節點和邊的數量。

## **Feature**

　　我們提取資料集中的各種特徵作為分析演算法的輸入，以比較不同特徵的重要性，評量分析預測效能。以下列出:

#### Shortest Path Length :

兩節點之間最短路徑距離。無相連路徑設距離為無限大。

#### Jaccard Distance :

兩節點各自的鄰居集合的互斥程度。0%表示兩節點鄰居集合完全一致。100%表示兩節點各自的鄰居集合互為獨立集。

#### Cosine Distance



表示二節點之間的相異程度，為1-cosine similarity。

#### Same Community

表示兩節點是否屬於相同的社群 (community) ，以二元分類0,1表示。社群計算採 Louvian 演算法。

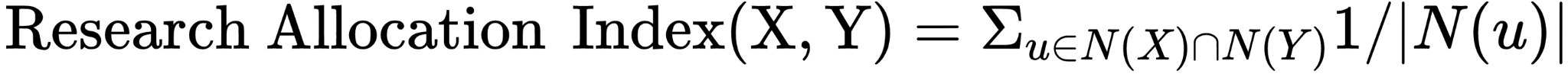
#### Common Neighbors

表示兩節點的共同鄰居數量。

#### Adamic Adar

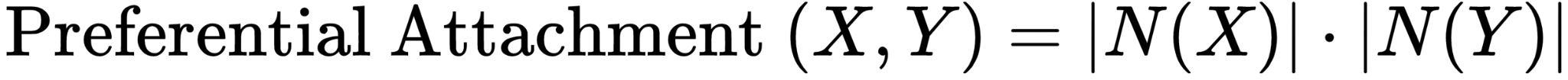
計算方式如上，表示兩節點之間的共同連結量。

#### Resource Allocation Index



計算方式如上，表示一個節點可以通過他們共同的鄰居向另一個節點發送的資源

#### Preferential Attachment



計算方式如上，為兩個節點之鄰居數量相乘。

#### SVD

以奇異值數量=6進行svd 分解，每個node 分別於U和V 矩陣對應一個6為向量，將節點對之u向量進行內積，形成”svd\_dot\_u” feature;將節點對之v向量進行內積，形成 “svd\_dot\_v” feature。

#### GEMSEC

計算 node embedding 的同時學習節點的群集 (clustering) 關係。節點放置在抽象特徵空間中，其中頂點特徵最小化保留採樣頂點鄰域的負對數似然性，並通過機器學習正則化結合已知的社交網絡屬性。

#### GEMSEC with Regularization

在前述方法的基礎上，loss function 採納了 adamic-adar 特徵作為正則化 (regularization)依據。

#### DeepWalk

利用隨機遊走形式提取圖中節點的 node embedding 資訊。

#### DeepWalk with Regularization

同上述 GEMSECWithRegularization 方法，採納了 adamix-adar 特徵作為正則化 (regularization)依據。

#### Node2vector

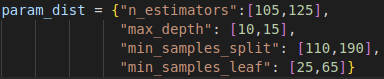
node2vec是結合了DFS和BFS隨機遊走的deepwalk方法。優化的目標是給定每個頂點條件下，令其近鄰頂點出現的概率最大。

## **Learning Model**

　　我們將透過Random Forest及Gradient Boosting Decision Tree這兩種模型進行學習，進行Link Prediction。

　　以下將對於兩種學習模型以及我們實作中的參數設置進行介紹：

* Random Forest



上圖為四項參數:n\_estimators, max\_depth, min\_samples\_split, min\_samples\_leaf 及其值所形成的字典。將RandomForestClassifier及此字典給進 RandomizedSearchCV 函式，隨機選取五種參數設定，每種設定進行10-fold cross-validation，選出f1最高之estimator,使用訓練資料fit，並以此estimator 之Train f1, Test f1, Train ROC, Test, ROC ,Train confusion matrix, Test confusion matrix 作為最後驗結果，並分析各個不同feature的重要程度。

* Gradient Boosting Decision Tree



上圖為兩項參數:n\_estimators, max\_depth 及其值所形成的字典。將RandomForestClassifier及此字典給進 RandomizedSearchCV 函式，隨機選取五種參數設定，每種設定進行3-fold cross-validation，選出f1最高之estimator,使用訓練資料fit，並以此estimator 之Train f1, Test f1, Train ROC, Test, ROC ,Train confusion matrix, Test confusion matrix 作為最後驗結果，並分析各個不同feature的重要程度。

## **Experiments**

### **Experimental Detail**

　　在這一個部分，我們將對於本次實作的實驗設置細節進行介紹：

#### **Negative Link Sampling**

　　由於資料集內容為粉絲專業互相按讚的關係，關係皆為Positive，為使模型更好的學習到不同的連結關係，我們額外加入了Negative Link，也就是原始資料集中沒有按讚關係的粉絲專業。

　　我們透過採樣的方式選取Negative Link，取出沒有按讚關係的兩個粉絲專業作為Negative Link。而在採樣數量的設置上，為防止模型過度傾向學習某一種關係，我們選擇採樣與Positive Link相同數量的Negative Link進行學習。

　　在實作上，我們將有互相按讚的關係(Positive Link)設置為1，沒有互相按讚的關係設置為0。進行Link Prediction時，模型會預測兩個粉絲專頁間是否有互相按讚關係，若有按讚關係便輸出1，沒有按讚關係便輸出0。

#### **Training and Testing Data Splitting**

　　在分割Training Data以及Testing Data時，我們為了使模型均衡的學習到兩種關係(Positive及Negative Link)，因此我們分別對於Positive及Negative Link進行分割。

　　對於Positive Link，我們以8：2的比例進行分割，其中8為Training Data，2為Testing Data。Negative Link也是以同樣的8：2的比例分割。接著，將Positive Link的Training Data及Negative Link的Training Data進行合併；Positive Link的Testing Data及Negative Link的Testing Data進行合併。分別獲取了Training Data及Testing Data。

#### **Graph Construction**

　　本次實作中，我們透過Python的Networkx根據資料集建立Graph。另外，由於資料集為兩粉絲專頁間互相按讚的關係，因此為無向圖。

　　由於Testing Data中存在部分Positive Link，為防止模型預先學習到正確答案，因此我們在建立Graph時，將Testing Data中的Positive Link移除。

另外，為了避免計算feature時出現太多未看過的點，我們將所有點加入Train Graph 中，使其包含所有點。

#### **Evaluation Metric**

　 　　在本次實作中，我們採用f1 score作為我們評估模型訓練結果的方式。

### **Research Questions and Analysis**

　　在本次報告的實驗中，我們根據先前介紹的方法計算產生每對Link的特徵值並透過Random Forest及GDBT兩種學習模型進行Link Prediction。

　　接著，我們對於不同方法進行設置並分成三個研究問題進行探討：

* **哪一種方法下，使的Random Forest及GBDT有最好的表現?**

　　根據先前介紹的不同方法學習Link的特徵值，經過Random Forest及GDBT學習並進行Link Prediction。這部分將比較在不同的方法下，Random Forest及GBDT預測結果的f1 score，探討哪種方法對於本次使用的資料集會有最好的表現，並說明兩個粉絲專頁間在哪種情形下有互相按讚的傾向。

* **論文提出的GEMSEC運用於Link Prediction問題的表現為何?**

　　本次實作使用的資料集由《GEMSEC: Graph Embedding with Self Clustering》論文提出，論文中透過此資料集針對於Community Detection問題提出了GEMSEC的Embedding方式，我們將論文提出的GEMSEC運用於Link Prediction，並將其學習的結果與我們實作的其他方法進行比較，呈現出論文提出的GEMSEC方法在Link Prediction的表現。

* **在不同資料集中，不同方法的重要性是否有所差異?**

　　本次實驗使用的資料集由多種類型的粉絲專業所組成，其中包含了Government、New Sites、Athletes、Public Figures、TV Shows、Politician、Artist以及Company，總共8種類型的粉絲專頁類型所形成的資料集。我們將比較在不同類型的粉絲專頁下，哪些方法有較好的表現，進而探討不同類型的粉絲專業形成互相按讚關係的傾向是否有所差異。

　　接著，將對於這三個問題透過實驗數據進行分析：

#### **RQ1. 哪一種方法下，使的Random Forest及GBDT有最好的表現?**

首先針對各個方法的實作細節及設置進行介紹：

* **Common Neighbors**

　　透過Networkx中的common\_neighbors()函式進行實作，計算共同鄰居節點的數量。

* **Same Community**

　　在本次實作中，嘗試了Networkx中的connected\_components()及Python-louvain中的best\_partition()，兩種函式進行Community Detection，若兩粉絲專頁傾向屬於同一個Community設置為1，若屬不同Community則設置為0。在實驗結果發現best\_partition() 會有較好的表現，因此這部分呈現的數據是以best\_partition()進行實作。

* **Jaccard Distance**

　　使用Networkx的jaccard\_coefficient()計算兩粉絲專頁間的特徵值。

* **Cosine Similarity**

　　使用Networkx的neighbor()計算兩粉絲專頁各自的鄰居節點數量，並透過鄰居節點數量計算兩粉絲專頁間的相似程度。

* **Adamic Adar**

　　使用Networkx的adamic\_adar\_index()計算兩粉絲專頁間的特徵值。

* **Shortest Path Length**

　　使用Networkx的shortest\_path\_length()計算兩節點間的最短路徑距離。

* **Resource Allocation Index**

　　使用Networkx的resource\_allocation\_index()計算兩粉絲專頁間的特徵值。

* **Preferential Attachment**

　　使用Networkx的preferential\_attachment()計算兩粉絲專頁各自的鄰居節點數，相乘得此條Link的特徵值。

* **SVD**

　　使用SVD矩陣分解計算Graph的中節點的特徵值，針對每一對互相按讚關係的兩節點的U及V矩陣分別相乘，計算得到每對Link的特徵。

* **Node2Vec (PCA)**

　　使用Node2Vec產生每個節點的特徵Embedding，每對Link的特徵值為透過將兩節點的特徵值相加而得。由於維度較高的Embedding方法相對於前面介紹的方法會有較好的表現，為了著重於先前提到的方法，以及了解降維對於多維Embedding表現的影響，我們使用PCA()對Node2Vec產生的16維Embedding進行降維，變成維度為1的特徵值。

* **GEMSEC (PCA)**

　　論文提出的Embedding方法，維度為16，我們同樣使用PCA()對產生的Embedding進行降維，變成維度為1的特徵值。

* **GEMSEC with Regularization (PCA)**

　　論文提出的Embedding方法，維度為16，我們同樣使用PCA()對產生的Embedding進行降維，變成維度為1的特徵值。

* **DeepWalk (PCA)**

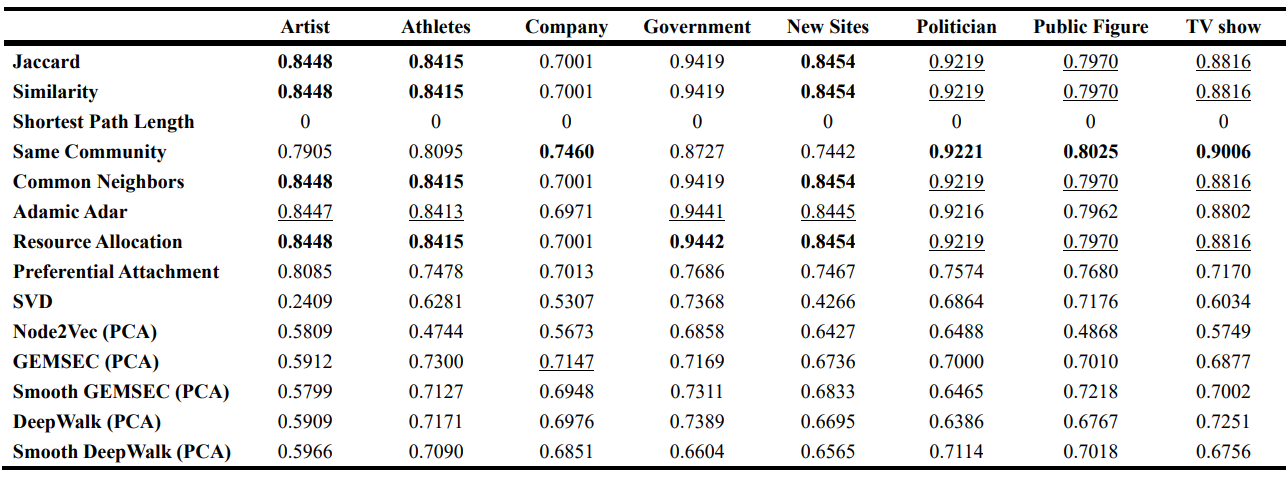
　　論文中比較對象的Embedding方法，維度為16，我們同樣使用PCA()對產生的Embedding進行降維，變成維度為1的特徵值。

* **DeepWalk with Regularization (PCA)**

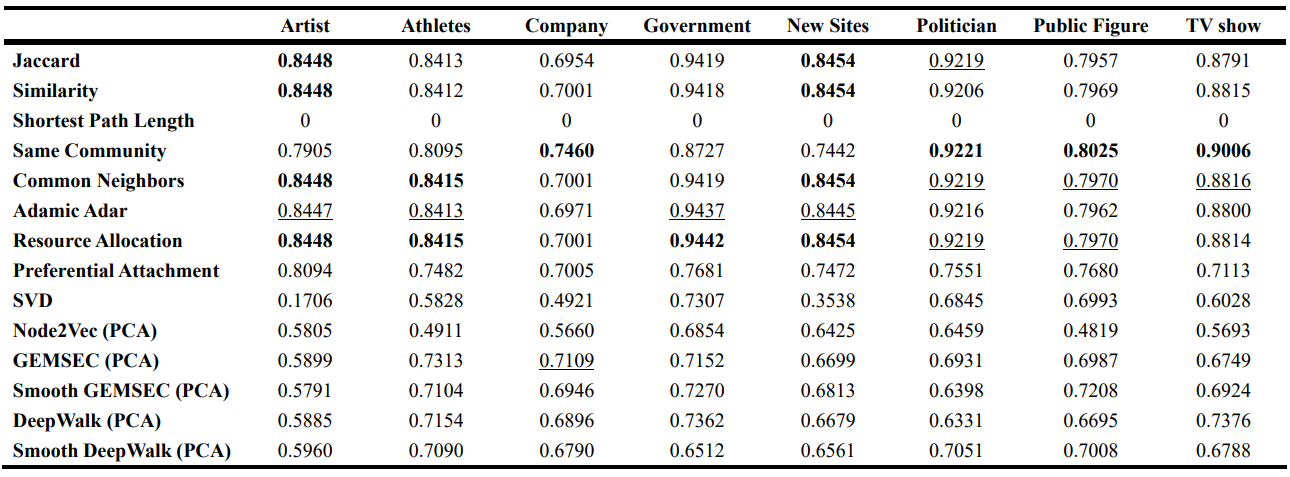
　　論文中比較對象的Embedding方法，維度為16，我們同樣使用PCA()對產生的Embedding進行降維，變成維度為1的特徵值。

　　以下數據呈現各個粉絲專頁類型的資料集在不同方法下的表現：

* **Random Forest**



* **GBDT (Gradient Boosting Decision Tree)**



　　根據上面數據進行分析，我們發現：

* Random Forest 及 GBDT兩種方式的模型學習效果差異不大，維度不多的特徵值使兩種模型學習到相近的結果。
* 根據鄰居節點計算的方法，像是Jaccard、Similarity、Common Neighbors、Adar、Resource Allocation等等，都有不錯的表現，呈現出鄰居節點對於此資料集的Link Prediction是有影響的，也說明有相近數量的鄰居或是有共同鄰居較有可能形成Link。
* 考量Community的方法Same Community，在Company、Politician、Public Figure、TV Show這四個類型的粉絲專頁都有很好的表現，說明在這四種類型的資料集中是共同Community的粉絲專頁較有機會成為互相按讚的關係。另外，在Company資料集中，同樣考量Community的GEMSEC (PCA)也有不錯的效果，呈現Community對於Company資料集的影響力。
* 對多維度Embedding進行降維的方法，像是GEMSEC (PCA)、DeepWalk (PCA)、Node2Vec (PCA)等等，效果較先前介紹的方法差一些，但部分資料集中即使經過降維，仍有0.7左右的f1 score。
* 我們觀察到Shortest Path Length對於訓練資料集有過度貼合的情形，f1 score接近0.9，並在測試資料集表現不佳，模型學習後將所有的Link預測為0，說明Shortest Path Length對於Link Prediction效果不大。

#### **RQ2. 論文提出的GEMSEC運用於Link Prediction問題的表現為何?**

首先針對各個比較方法的實驗設置進行介紹：

* **Node2vec**

　　使用Node2Vec產生每個節點的特徵Embedding，每對Link的特徵值為透過將兩節點的特徵值相加而得。

* **GEMSEC**

　　論文提出的Embedding方法，維度為16。

* **GEMSEC with Regularization**

　　論文提出的Embedding方法，維度為16。

* **DeepWalk**

　　論文中比較對象的Embedding方法，維度為16。

* **DeepWalk with Regularization**

　　論文中比較對象的Embedding方法，維度為16。

* **All Methods without High Dimension Embedding**

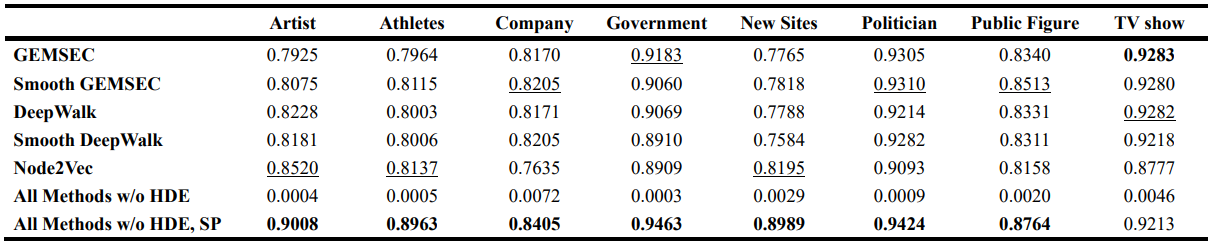
　　將先前介紹的9種方法結合(Common Neighbors、Same Community、Jaccard Distance…等)，其中不包含5個高維度Embedding經過降維的特徵值(Node2Vec (PCA)、GEMSEC (PCA)...等)。

* **All Methods without High Dimension Embedding and Shortest Path Length**

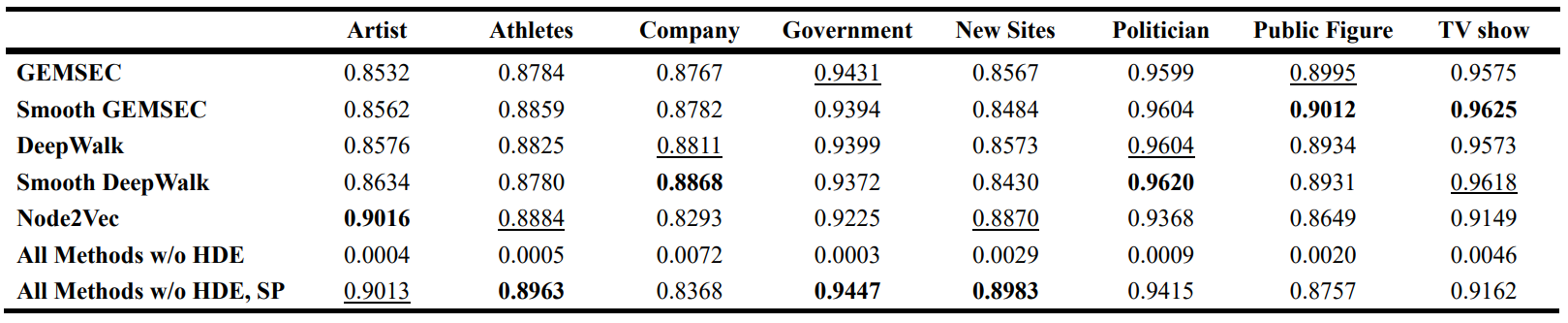
　　將先前介紹的8種方法結合(Common Neighbors、Same Community、Jaccard Distance…等)，不包含高維度Embedding經過降維的特徵值及Shortest Path Length的特徵值，其中因為Shortest Path Length對於此資料集訓練效果不佳，在下方的實驗數據將再進行詳細說明。

　以下數據呈現各個粉絲專頁類型的資料集在不同方法下的表現：

* **Random Forest**



* **GBDT (Gradient Boosting Decision Tree)**



根據上面數據進行分析，我們發現：

* Random Forest及GBDT兩種學習模型的結果在高維度特徵下，在不同方法的學習效果有明顯的差異。
* All Method w/o HDE與All Method w/o HDE, SP兩者的差異在於是否有無考量Shortest Path Length特徵，不論在Random Forest及GBDT都呈現加入Shortest Path Length會導致模型過度擬合於訓練資料集，導致在測試資料集表現不佳。
* 在Random Forest的模型訓練下，All Method w/o HDE, SP有最好的效果，呈現多方面考量不同資訊可以提高Link Prediction的準確率。另外，論文提出的GEMSEC也有不錯的表現，說明論文的方法雖然主要運用於Community Detection，但運用在Link Prediction上也有不錯的效果。
* 在GBDT的模型訓練下，大多數資料集中All Method w/o HDE, SP仍有最好的表現。

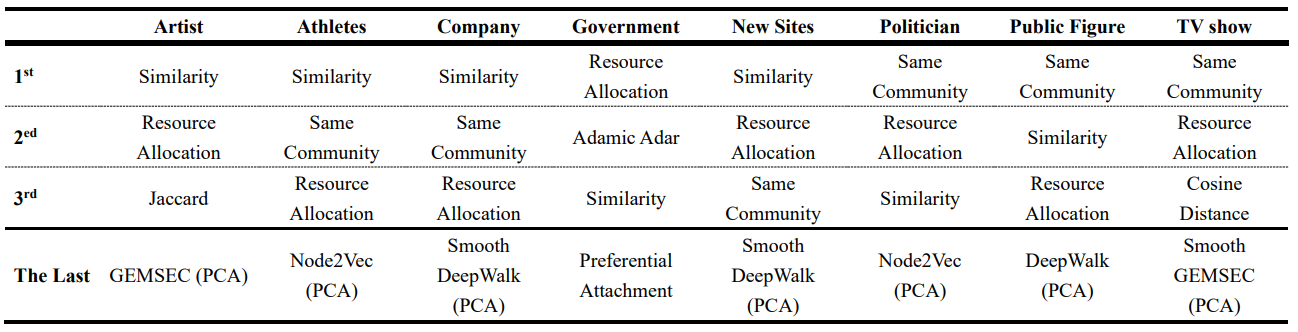
#### **RQ3. 在不同資料集中，不同方法的重要性是否有所差異?**

　　我們透過Random Forest及GBDT模型中的feature\_importances\_()函式獲取模型學習特徵時的重要性，根據模型學習不同粉絲專頁資料集時的特徵重要性，分析不同方法在不同資料集上的影響。

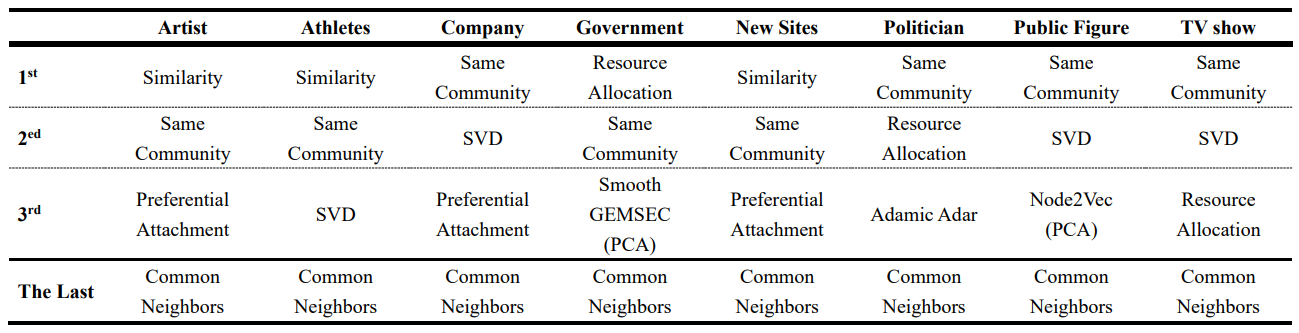
　　特徵重要性的計算方式是透過計算Mean Decrease Impurity，目的為降低預測目標的不確定性，考量根據某項特徵進行Decision Tree分裂前後的信息增益(Gini)。

　　以下數據呈現在不同資料集下，影響力最大的前三個方法以及影響力最小的方法：

* **Random Forest**



* **GBDT (Gradient Boosting Decision Tree)**



根據上面數據進行分析，我們發現：

* 在Random Forest的模型學習下，考量鄰居數量的方法及考量Community有較高的影響力。不過在將多維Embedding特徵進行降維後，模型學習的效果較不佳，說明降維對於特徵資訊的保留有所影響。
* 在GBDT的模型學習下，考量鄰居數量的相似程度及考量Community都有較大的影響。然而Common Neighbors在所有資料集的影響力都是最差的，說明對於GBDT的學習，共同鄰居數量的影響力較低一些。
* 從影響力排名中可以觀察到部分資料集有較相近的排名，說明該類型的粉絲專頁可能有較相似的互相按讚傾向。像是Company及Public Figure這兩個資料集中Same Community影響最大，說明這兩種類型的粉絲專頁屬於一個Community較可能有互相按讚的關係。