Machine Discovery Homework 2

網媒所 R05944014 何文琦
化工系 B01504044 宋易霖

一、摘要

在社群網路上，因為注重隱私，有些資料被隱藏了起來，像是我們無法得知這名使用者是否「喜歡」這篇文章，但是在這次的作業中，我們可以透過一些相關的資料，來發現使用者和文章之間的連結，而那些相關資料包含使用者總人數(user.txt)、使用者的朋友(relation.txt)以及文章(以下稱文章為item)為何人所擁有、所屬的類別還有與使用者連結的總數(message.txt)。我們主要是根據老師提供的論文裡的演算法來對資料作 training，推論出使用者和文章之間的連結。又因為可以繳交兩份預測的結果，我們用了兩種不同的方法作計算，以下針對各個方法說明其資料結構和演算法。

二、資料結構

〈方法一〉

自訂新的 Structure 來儲存 User、Item 和 Category。

User：儲存 User 的 id、所有的朋友和所擁有的 item。

Item：儲存 Item 的 id、所有擁有這個 item 的使用者、類別和連結數。

Category：儲存 Category 的 id、所有這個類別中的 item。

分別用三個 Vector 來存所有的 User、Item 和 Category。

因為總 Item 數太多，為了可以更快速找到 Item 的資料，我們又用了一個 Map 資料結構來儲存 Item 的 id，可以直接對應到 Item Vector 中的位置，所以只要搜尋 Map 中的 Item 的 id，就可以直接找到 Vector 中所有關於 Item 的資訊。

〈方法二〉

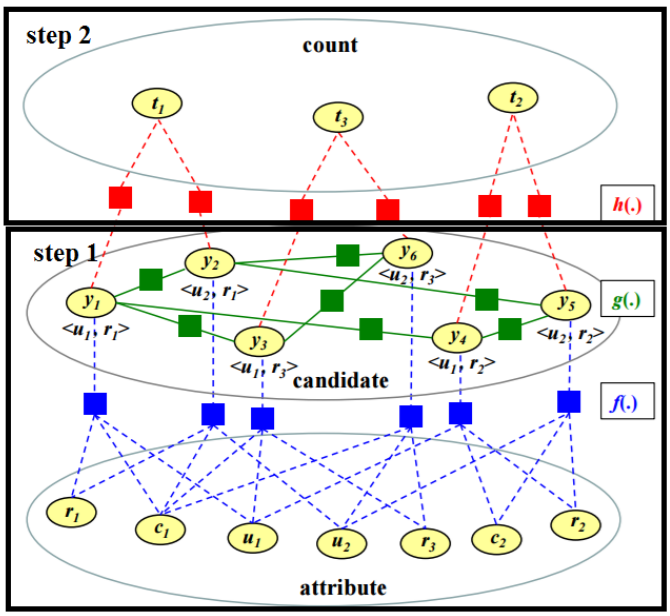
自訂新的 Structure 儲存 User、Item 何 Category。

User：儲存 User 的 id、所有的朋友和所擁有的 item，用 set 來儲存 item，方便要搜尋 item 時可在 log(n)內完成。

Item：儲存 Item 的 id、所有擁有這個 item 的使用者、類別和連結數，使用者和類別都用 set 來存。

Category：儲存 Category 的 id、所有這個類別中的 item，item 用 set 來存。

三、Model



(*Unsupervised Link Prediction Using Aggregative Statistics on Heterogeneous Social Networks*", KDD' 13)

我們參考了 Unsupervised Link Prediction Using Aggregative Statistics on Heterogeneous Social Network 所提出的 Model，但為了加快速度和一些做作業時的發現，我們對此 model 做出了修改。

1. 我們 y 的總數沒有取全部的配對，有因應不同的做法做了不同取樣(參照結果部分)。
2. 我們在 training 的時候並沒有考慮 h(.)這個參數，而是在 training 完畢以後，利用題目所說的 50%左右是 like 來取 test 中的前 50%的 pair 當作是 1。

四、演算法

針對每一個 user 和 item 的配對(以下簡稱 pair)，訂定多個 feature 來作 score 的計算，得出每個 pair 的 score 之後做高低的排名，而越高分的 pair 之間有聯結的機率越大。假定一個 pair 用 p(a, b)表示，a 為這個 pair 中的 user，b 則是 item，我們選定描述這個 pair 的 feature 有六個，這些 feature 乘上 Theta 所得出的值就是每個 pair 的分數，最終的目的主要就是 train 出一個最適合的 Theta 值來判斷出每個 pair 有連結的機率是多少。以下是我們訂定的六個 Feature :

(1) 使用者 a 擁有的所有 item 總數
(2) 使用者 a 擁有的所有朋友總數
(3) 使用者 a 是否擁有這個 item b
(4) 使用者 a 所有擁有這個 item b 的朋友總數
(5) item b 的類別中所有的 item 總數
(6) 使用者 a 擁有的 item 所屬的類別中，和 item b 的類別相同的總數

Training 過程

針對一個 item 和全部的 User 形成的 pair 來做 score 的計算，依照分數的高低做排名，然後根據這個 item 的連結總數取前幾名來分成 upper 組，其餘的 pair 則是歸類在 lower 組，分別計算 upper 和 lower 中的 pair 每一個 feature 的分數平均，利用 upper 和 lower 的差值乘上一個參數(learning rate)，再加到 theta 上做更新，即完成一次 training。

Training 的過程中我們用了一些技巧來調整他們的數字，因為每個 feature 算出來的值會差異蠻大的，像是朋友數有些是 100 位、一個類別中的 item 有幾千個，而這個 user 是否擁有 item 只會有 1 或 0 的結果，為了讓數值平均一些，每個 feature 最後都是取它的 log 值來做計算。而最後在更新 theta 的時候，也會根據不同的 feature 有不同的 learning rate 來做變動。

〈方法一〉

因為最終要預測的結果有一半是有連結，另一半是沒有連結，所以我們對所有的要預測的 pair 做高低排名，找出前 50%的 pair 給定預測值為 1，後 50%則是 0。

〈方法二〉

大致與方法一相同，但是 data 不是取 log 處理，而是直接將 data 做 mean normalize。($X_{new}= \frac{x- \bar{x}}{\text{range of x}}$)

五、結果

經過測試得到最好的 learning rate 為 : [0.01, 0.01, 1, 1, 1, 0.1]，很直觀的可以發現因為 feature(1)、(2)的值介於 100 左右，feature(5)的值皆小於 100，其餘的值皆在 10 之內，經過 learning rate 的調整，valid 資料整體的正確率可以上升到 97%左右。而在 test2 中因為 feature 的數值和 valid 得出來的結果不太一樣，所以針對 test2 給了不同的 learning rate : [0.01, 0.001, 1, 1, 1, 0.1]。

〈方法一〉

Training 的 item 數越多的話，正確率也會越來越好，但是考慮到 training 越多 item 之後會慢慢收斂，還有執行時間的問題，我們還是只選取了一些 item 做 training。Test1 選了三萬個 item 作 training，test2 則是 1000 個。

〈方法二〉

因為所有 item 和 user 的配對太多了，因此我們在實際 training 時是對

Training data 隨機取樣，分別做了固定所有 User 都取但 item 只取約 500~4000 個和固定所有 Item 都取但 user 只取 5~10 倍 like 的數目(取樣的個數會因為 test1、test2 的 item、user 數目做變化，基本上 test2 因 data 較大，因此只取比較少的樣本)。最後因為在 validate 上的表現，取所有 user 的方法答對率較高，因此選擇此方法。

六、工作分配

何文琦：實作方法一
宋易霖：實作方法二

七、參考資料

Unsupervised Link Prediction Using Aggregative Statistics on Heterogeneous Social Networks", KDD' 13