結構化學習應用於消費型商品推薦

陳昱瑾
中央大學資工所碩士班
julie25599@gmail.com

楊佳靜
中央大學資工所碩士班
uiuityty2003@gmail.com

陳宥涵
中央大學企管系
jmail840619@gmail.com

李胤龍
中央大學資工所碩士班
sebastien.montella@utbm.fr

張嘉惠
中央大學資工系教授
chia@csie.ncu.edu.tw

陳品良
資策會
mileschen@iii.org.tw

楊秉哲
資策會
maciaclark@iii.org.tw

谷圳
資策會
cujing@iii.org.tw

摘要

近年來網路普及，電子商務網站發展成熟，隨之而生的推薦系統發展蔚為風潮，如何幫助消費者從成千上萬的商品中，找到他所需要的商品，並在最短時間內將「對的商品」在「對的時間」推薦給「對的人」，是很重要的。然而，推薦系統背後仰賴的是大量的數據和有效的分析，對於初創發展不久的企業來說是很大的問題。

本研究屬於消費型商品推薦，與以往推薦系統主要差異為「時間序列」以及「重複購買」，消費型商品與時間有很大的關係，而且會被使用者重複購買。論文中，我們透過設計使用者特徵、商品特徵、使用者與商品的交互特徵以及時間相關的特徵，將資料透過機器學習演算法訓練預測模型。我們以時間切分後的測試資料進行預測。實驗數據顯示在 Top5 的 F-measure 比去年研究[3]高出 66.38%，提供更有效的推薦系統。

關鍵詞：消費型產品推薦、機器學習、結構化學習。

1. 緒論

近年來隨著網際網路的發展，電子商務網站發展成熟，改變了使用者瀏覽商品資訊方式與消費行為，越來越多使用者從傳統的瀏覽紙本型錄、到實體店面購買的消費形式，轉型成線上購物，電子商務縮短了交易時間和成本。但面對琳瑯滿目的產品，顧客要如何快速選購喜歡的產品，是使用者關注的議題。對於商家而言，能將對的商品投放給有需要的顧客，是商家經營商城的首要目標。

另外一方面，隨著「行動端」的迅速發展，社群網站及通訊軟體已成為使用者頻繁駐足之平台，像是 Facebook 和 Line 等。這些網站提供免費的服務，讓使用者黏著在他們的平台上，提供豐富的使用者的瀏覽行為，可讓系統端更加了解使用者，進而提升客製化行銷的機會。舉例而言，對 Netflix 公司來說，就有八成的購買行為來自於電影推薦；而 Google 及 Facebook 網站，也有 85%以上的收益是來自於廣告推播。因為有這樣的市場，使許多研究人員投入推薦系統的領域。

推薦系統仰賴大數據，依據使用者過去的行為來預測用戶對物品的「評分」或「偏好」。本論文主要分析消費型產品，像是洗面乳、衛生紙、暖暖包，每隔固定的時間，顧客便回流購買或者在特定的季節產生需求。此種消費模式和大部份僅觀看一次電影的推薦不同，主要差異為「時間因素」以及「重複購買」此兩項特質，也是我們需要面臨並克服的挑戰。

在本文中，我們使用結構化學習做為推薦系統主要的演算法。結構化學習是一種通用的學習框架，可以解決不同形式的各種複雜學習問題，例如圖像識別、序列標記等問題。我們建立一個線性模型，對每位使用者 x 計算其購買各商品 y 的分數 $F(x,y)=w\cdot\phi(x,y)$，並希望使用者有購買過商品 $\hat{y}$ 的最小分數 $P_{min}=\min_{y\in\hat{y}}F(x,y)$高於使用者沒購買商品的最大分數 $N_{max}=\max_{y\notin\hat{y}}F(x,y)$，我們定義一個使用者的 Cost 值為 N_{max} - P_{min}，接著將所有使用者的 Cost 值加總作為目標函數，並找出能最小化此目標函數的 w 來預測使用者購買某項商品的分數。

為了考慮時間因素，實驗中以時間切分訓練與測試資料，並設計時間相關的特徵，利用過去 3 個月的消費資訊做為特徵，再以接著 2 個月的消費紀錄作為有無購買之答案。實驗結果顯示 F-measure 比去年研究高出 66.38%。

2. 相關研究

推薦系統被應用於各行各業，其應用的範圍包含電影、音樂、書籍與一般商品等。大多資料屬於分數型與二元型兩種。以電影推薦為例，是利用過去使用者對電影的評分為基礎來分析，屬於分數型的資料。而商品推薦是以使用者過去是否購買過商品為依據，屬於二元型。此外是否重複推薦也會因應用主題不同而改變，如：同一部電影較不會被使用者重複觀看，故相同電影較不適合重複推薦，而音樂與消費型商品是會被使用者重複收聽或購買，故適合重複推薦。本篇研究為消費型商品推薦，屬於後者。

推薦系統策略大致上可規劃為三種技術：內容式過濾[6]、協同式過濾[11]以及結合以上兩者的混合式推薦系統。內容式過濾是根據顧客喜好的商品資料，找出特徵相似的產品，建立顧客的喜好模式，並且推薦和顧客喜好類似的產品。優點是簡單，可被用來解決冷開始問題，但內容式過濾的推薦結果可能會太相似而缺少變化。

協同式推薦系統是參考其他相似使用者所購買的產品產生預測結果，將使用者選購產品之歷史資料，分析使用者之間的相似性，來預測使用者對那些產品感興趣，又可分為 Memory-based CF [4]及 Model-based CF [2]兩項。

Memory-based CF 是利用 k-Nearest Neighbor [1]的方法當作 Filtering 的機制，並無需事先訓練模型。

Model-based CF 則是事先對歷史資料建立訓練模型，再依據模型預測結果。常用的方法是將使用者是否會購買某項商品作為分類的問題[2]，使用的方法有 SVM[9]、因素分解 [10]等，另外也有將推薦問題視為排序問題的做法，例如 SVMRank[12]以及結構化學習框架。

3. 資料介紹、特徵設計與推薦方法

本研究的資料是來自一款以集點任務作為設計導向的 App，該 App 主要目的是希望鼓勵用戶前往實體店面消費。這個 App 提供的任務共有三種型態：包括到店、掃描及消費，使用者只要完成任務即可獲得點數。接著使用者可以利用這些點數換取喜歡的禮品。由於 App 需綁定 Facebook 登入，因此使用者相關資訊，如年齡、性別、居住區域，按讚過的粉絲專頁均可獲得。此外，使用者執行任務的資訊 ，如店家資訊、任務內容及種類、消費時段、消費金額等，也都是使用者的相關資訊。

我們主要分析的商家為台灣個人美妝生活用品百貨，販售的商品有美妝、飾品、食品、五金用品等等，大多屬於消費型的產品，這些消費型商品在使用者使用完後，因此主要面對的問題為商品被重複購買。消費型的推薦與以往的推薦系統非常不同，如：書籍推薦，大多數讀者在購買一本書後，不會重覆購買同一本書。但消費型的商品是會被經常地重複購買，故重覆購買機率是消費型產品推薦的重要資訊。

另外一個關鍵因素則是時間，如前面段落所述，我們推薦的商品屬於消費型商品，這些商品是會被重複地購買的，若不考慮時間則每次推薦商品都是一樣的。由於隨著季節的推移，銷售的商品也會跟著改變，如暖暖包主要銷售的時間為冬季，夏季較少人會購買暖暖包。此外衛生紙這種消費型產品，每隔固定的時間後顧客便會回流購買。

不同與一般商品推薦，這個問題被設定在推薦商品類別給使用者，選擇商品類別的原因是給予廠商較高自主性，如廠商可以自行選擇利潤較高的商品做推薦，也可以考量其庫存狀況，推薦效期較近的產品，以方便廠商控管其貨品，推薦商品類別使廠商有較彈性的空間。

3.1. 使用者特徵

由於集點 App 是藉由和 Facebook 綁定的方式進行註冊登入，因此我們可以藉此取得使用者的相關資料，包含使用者 FB 的 ID、性別、年齡、居住地、按讚過的粉絲專頁等等。我們也延用了[3]所設計關於使用者個人的特徵，將年紀分為三類：20 歲以下、20~40 歲、40 歲以上，性別為男性或女性，居住區域分為北台灣、中台灣、南台灣、東台灣、外島並將居住地為國外的使用者定義為來台旅遊者。

我們也依使用者按讚過的粉絲專頁來分類，找出使用者可以所屬的族群，如女學生、小資女、父親等 18 個族群，每位使用者屬於一個最可能的族群。此外，我們將粉絲專頁分為食、衣、行、育、樂，並判斷使用者對哪幾類的粉絲專頁按讚。

為了了解使用者的消費習慣，我們設計了使用者消費習慣的特徵，包括使用者執行過哪項任務(到店、掃描以及消費)、其消費的金額高低分為高於平均或低於平均、購買的商品類別屬性分為吃喝類、穿用類、育樂類以及消費的時段與消費日的屬性。每個特徵皆以二元方式表示，共計 50 個使用者特徵。

3.2. 產品特徵

我們主要分析的商家為個人美妝生活用品百貨，其販售的商品有美妝、飾品、食品、五金用品等等，多達 2,177 項產品，分為 28 大類、144 中分類及 377 小分類。我們利用網路爬蟲取得產品相關資訊，並用關鍵字提取與 word2vec[8]兩種方法來表示產品特徵。

關鍵字提取是將產品中分類和小分類的 Google 搜尋的前十筆 title，透過 jieba 進行斷詞，並結合 TFIDF[5]和 TextRank[7]兩種關鍵字提取(keyword extraction)的方式選出我們的特徵。分別將 TFIDF 和 TextRank 作詞性篩選，挑出名詞、動詞、地名、動名詞四種詞性。並各別提取前五個關鍵字作聯集，作為產品特徵。

Word2vec 方式則將搜尋 Google 以及 Yahoo 前 10 筆 Snippet 紀錄下來作為語料，並對這些爬下來的資料作預處理，將日期、量詞、售價…取代掉。如：10 公克、2 盒取代為_QUANTITY_。一樣利用 jieba 進行斷詞。接著將斷詞好的資料利用 word2Vec 來訓練模型，我們將向量設為 300 維、window 大小設為 7，獲取每個商品字詞的空間向量，再以空間向量總和表示商品特徵。

3.3. 使用者購買商品之機率及時間因素特徵

除了使用者與商品本身的特徵外，我們也計算使用者購買各種商品的機率，做為其交互關係的特徵。由於部分使用者購買次數較少，使其商品購買機率多為 0，因此我們利用前述的使用者分群，並計算各群組購買各種商品的機率，將此機率做為使用者與商品交互關係的特徵。

另外由於消費性商品會被重複購買，故與時間有很大的關係，所以我們也設計了一些使用者購買商品週期的特徵，如下表 1。其中前 5 項特徵為二元分類，包含使用者是否購買過此商品、使用者是否購買超過 1 個本項商品，使用者上次購買此商品時間距離現今是否超過平均購買週期，其中平均購買週期分為 3 種：使用者購買此商品的週期、使用者平均購買週期以及商品平均被購買週期。

表 1 的 6~8 項特徵為數值型，包含使用者購買商品的累積次數、使用者上次購買此商品時間到目前區間內購買商品的次數；由於使用者購買次數差距很大，所以我們將次數取 Log 計算，藉以調整其差距。最後一項則為使用者購買商品的密度，也就是使用者上次購買此商品時間到目前期間內的平均購買個數與長期平均購買個數的比例；直覺上若比例小於 1，則依據使用者習慣購買機率會增加，若是比例大於 1 則使用者購買機率會減少。

表 1.時間相關特徵

序號	特徵設計	資料型態
1	User buy item before	binary
2	User buy item more than once	
3	| current_time − last_time | > user_item period	
4	| current_time − last_time | > user avg period	
5	| current_time − last_time | > item avg period	
6	Log(count(user, item) + 1)	numeric
7	Log(count(user, item, interval) + 1)	
8	$\left(\frac{count(user,item,interval)}{interval}\right) / \left(\frac{count(user,item)}{current\_time - firs\_time}\right)$	numeric

3.4. 結構化學習與隨機梯度下降

結構化學習提供我們一個機器學習的框架，它不再侷限於只能輸出一個數值或 label 型態的資料，而是能接受任何是結構態的輸入或輸出，例如：序列(sequence)、圖像(image)或邊框(bounding box)等，整體上我們能利用結構化學習的方式更直接的處理遇到的問題，可以說我們把任何的輸出都當作是某種結構。結構化學習主要涉及訓練以及推論兩個部分。在 Training 的部分，我們希望找到一個函數 F: X×Y→R，使 F(x, y)得到反數能夠反應 x 和 y 配對的好壞。在 Inference 的部分，則是透過方程式 $\tilde{y} = \underset{y\in Y}{\operatorname{argmax}} F(x, y)$，針對一個給定的 x 窮舉所有可能的 y，以此找到一個最好的答案。

以推薦系統來說，對於使用者 x、商品 y、以及使用者與商品的特徵向量∅(x, y)，最簡單的是線性模型 F(x, y) = w · ∅(x, y)。我們希望對於每位使用者有購買過的商品($\boldsymbol{y} \in \hat{\boldsymbol{y}}$)代入函數 F(x,y)後算出來的值(P)會大於沒購買過的商品($\boldsymbol{y} \notin \hat{\boldsymbol{y}}$)代入函數算出來的值(N)。由於使用者購買的商品不只一項，因此我們取 P_{min} 及 N_{max}，希望對所有使用者皆滿足 $P_{min} \geq N_{max}$。但對於有些使用者而言，其資料代入函數 F(x ,y)後，N_{max} 可能會大於 P_{min}，所以我們定義一個 Cost 值 C = N_{max} -P_{min}，表示錯誤的大小。我們期望有買過的資料 P_{min} 盡量能大於 N_{max}，若無法大於，也使兩者差距不要太大。最後將每位使用者算出來的 Cost 加總，整體目標函數為最小化$\sum_{i=1}^{N} C_i$。

由於訓練資料與測試資料的分佈可能不同，若 w 接近 0，可以降低單一 mismatch 的影響，所以一般會在目標函數中加入調節因子，並利用隨機梯度下降法求得最佳值，目標函數如下。

$$\text{Cost} = \frac{1}{2}||W||^2 + \sum_{i=1}^{N} C_i = \frac{1}{2}||W||^2 + \sum_{x\in X} w\cdot\emptyset(\mathbf{x},\mathbf{y_n}) - w\cdot\emptyset(\mathbf{x},\mathbf{y_p})$$

$$\nabla\text{Cost} = W + \sum_{x\in X} w\cdot\emptyset(\mathbf{x},\mathbf{y_n}) - w\cdot\emptyset(\mathbf{x},\mathbf{y_p})$$

演算法如下，對於每個使用者 x，我們將計算每位使用者(x)有購買過的商品$\boldsymbol{y} \in \hat{\boldsymbol{y}}$中分數最小的 P_{min}=$min_{y\in\hat{y}}$F(x,y)與當時的商品 y_p=$argmin_{y\in\hat{y}}$F(x,y)，同理使用者 x 沒購買過的商品$\boldsymbol{y} \notin \hat{\boldsymbol{y}}$中分數最大的 N_{max}=$max_{y\notin\hat{y}}$F(x,y)與當下的商品 y_n=$argmax_{y\notin\hat{y}}$F(x,y) ，若 P_{min}< N_{max} 我們將會更新 w。若最後資料為可分，所有訓練資料(x, ŷ)代入後，皆得到 P_{min}≥ N_{max}，換言之所有購買過的資料皆大於沒被購買的資料，我們便可得到最佳的權重向量 w，當所有使用者皆進行過一次表示一次 epoch。

Algorithm 1. Structure Learning Algorithm

Initialize w vector is random number from -1 to 1
For each user (x, ŷ)
$N_{max} = \max_{y\in\hat{y}} F(x, y)$
$P_{min} = \min_{y\in\hat{y}} F(x, y)$
if $P_{min} < N_{max}$
$y_n = \operatorname{argmax}_{y\notin\hat{y}} F(x, y)$
$y_p = \operatorname{argmin}_{y\in\hat{y}} F(x, y)$
$w = (1-\eta)\cdot w - \eta[\emptyset(x, y_n) - \emptyset(x, y_p)]$
End While;

4. 實驗

我們的資料是來自於集點 App 2015 年 1 月至 2016 年 6 月的資料，期間交易的商品個數為 259,550 個，我們篩選出消費超過 20 個商品的 1,715 人做分析，此做法可以減少資料稀疏的影響。商品的部分，我們選擇推薦商品類別，共有 144 個中類別，以類別推薦是希望給予廠商較高自主性，如廠商可以自行選擇利潤較高的商品做推薦，也可以考量其庫存狀況，推薦效期較近的產品，以方便廠商控管其貨品，推薦商品類別使廠商有較彈性的空間。

4.1. 資料準備

一次考慮 5 個月的資料，將前 3 個月的資料作為特徵，後 2 個月的購買情形做為 label。這個做法可以增加訓練資料的數量，可以解決資料不足的問題。實驗中我們選定 2015 年 1 月至 2016 年 4 月的資料作為訓練資料，共有 13 份使用者商品訓練資料。將 2016 年 5 月及 2016 年 6 月兩個月的資料作為測試資料。

4.2. 評估方法

我們使用 F-measure 來進行效能測試。在推薦系統中 Precision 固然很重要，但 Recall 也不容忽視，因為推薦系統不只要精準預測，也需要將使用者會購買的商品都找出來，所以我們選擇以 F-measure 做為評估，計算 P*recisio*n (hit *rate*)、R*ecall*、F-*measure*公式如下。

$$\text{Precision(hit rate)} = \frac{c(r\cap b)}{c(r)} \qquad \text{Recall} = \frac{c(r\cap b)}{c(b)} \qquad \text{F} = \frac{2RP}{(R+P)}$$

r 表示我們推薦給使用者的商品，b 表示使用者購買的商品。在一般的推薦系統中，推薦給使用者的商品數不會太多，因為一次推薦給使用者過多商品，會使使用者無法負擔，故我們將針對 Top1、3、5、10、15 來觀察。

4.3. 實驗結果

首先我們比較各類特徵在 Structure learning 下的效能影響，由於 Structure learning 是利用特徵相減（∅(x, y_p) ∅(x, y_n)）來更新權重，若只考慮使用者特徵，對於同一位使用者其特徵相同，無法更新，因此我們刪減特徵方式來衡量個別特徵的重要性，實驗結果如下表 2。我們發現少了前三項使用者特徵時，其表現幾乎都優於使用全部特徵（如紅色底線數字所示），這表示各類使用者特徵皆不是影響

效能的重要因素。相對的，商品特徵關鍵字以及使用者購買商品之購買機率及時間因素，效能與使用所有特徵相當。

表 2. 各類特徵效能比較

Top-k	F-measure	Increase Rate					
	all feature (2,453 dims)	without Personal Information (-11 dims)	without Facebook (-23 dims)	without Task Execution (-16 dims)	without Keyword (-2,045 dims)	without Word2vector (-300 dims)	without user-item pair feature (-58 dims)
1	0.0062	1.53	0.34	0.92	0.03	-0.10	-0.10
3	0.0250	0.10	0.34	0.00	-0.30	-0.10	0.14
5	0.0379	0.05	-0.05	0.14	0.09	0.03	0.02
10	0.0463	0.21	0.07	0.04	-0.05	0.02	0.02
15	0.0526	0.12	0.16	0.00	-0.06	-0.05	-0.07

由表 2 發現各類特徵對效能的影響皆不大，故以下我們選擇以 3.3 敘述的 58 維使用者購買商品之因素，作為後續實驗的特徵，原因為此特徵同時考慮了使用者與商品間的交互關係，且效能與其他特徵的表現相當，其維度較小運算速度較快，實驗結果效能也比較好。

我們比較在相同推薦數目下推給每位使用者相同數量或取資料的前 k 個推薦的效能。也就是說選取每個人的 Top 1 後計算其 F-measure，又因為使用者有 1,715 人，故以 User-Item pair 排序後，選取前 1,715 個 pair 並計算其 F-measure 來比較。由圖 2 發現以 User-Item pair 排序的 F-measure(圖中綠色實心 bar)，皆高出以 user 排序(圖中藍色斜線 bar)，而且差距很大，這是由於有些使用者購買較不頻繁，當他不購買時推薦給他商品皆算失敗，另一方面，對購買頻繁者來說，推薦給他較多的商品，可使他購買更多的商品，讓推薦可以更成功。

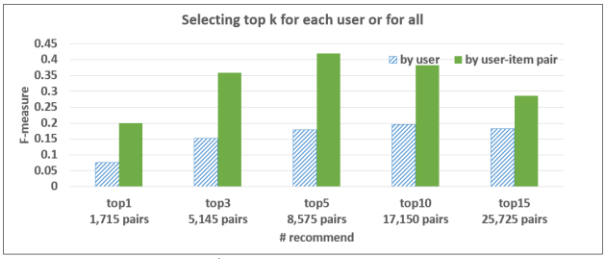

圖 1. 比較相同推薦數目下以 user 或 User-Item pair 排序的效能

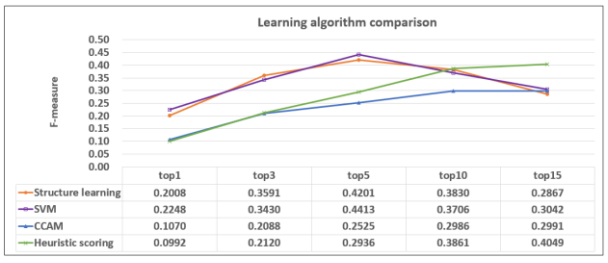

圖 2. 比較不同方法的效能

我們比較不同方法下的效能，如圖 3。比較 Structure learning、SVM、CCAM 及 Heuristic scoring 四種方法。發現 SVM[1]與 Structure Learning 互有高低，兩者的 F-measure 相差不大，在 Top3 和 Top10 時 Structure learning 的效能優於其他方法，SVM 在 Top5 的 F-measure 達到最佳的 0.4413。在 Top10 和 Top15 時 Heuristic scoring 的效能較好。

5. 結論

一般說來消費型的商品在使用者用完後會被重覆的購買，故消費型商品的推薦需考慮商品被購買的重覆性，而且與時間序列有很大的關係。過去的電影推薦系統主要是預測使用者喜歡哪部電影，同一部電影較不會被重複觀看。由於電影推薦與消費型商品推薦不同，所以我們改變特徵設計與資料準備的方式。

在本篇論文我們設計出與時間相關的使用者購買商品特徵，並搭配時序相關的資料準備方式，利用前 3 個月的購買狀況作為特徵，再將後 2 個月的購買狀況做為 label。接著搭配推薦所有資料的前 k 個來評估，Structure Learning 與 SVM 效能較好，而且效能相當，Structure Learning 的運算時間較短，可以滿足推薦的即時性。

致謝

本研究依經濟部補助財團法人資訊工業策進會「106 年度 服務系統體系驅動新興事業研發計畫(2/4)」辦理

REFERENCES

[1] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3), 175-185.

[2] Billsus, D., & Pazzani, M. J. (1998, July). Learning Collaborative Information Filters. In Icml (Vol. 98, pp. 46-54).

[3] Chen, Yu-Ching, et al. "User behavior analysis and commodity recommendation for point-earning apps." Technologies and Applications of Artificial Intelligence (TAAI), 2016 Conference on. IEEE, 2016.

[4] Delgado, J., & Ishii, N. (1999). Memory-based weighted majority prediction. In SIGIR Workshop Recomm. Syst. Citeseer.

[5] Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). Mining of massive datasets. Cambridge university press.

[6] Melville, P., Mooney, R. J., & Nagarajan, R. (2002, July). Content-boosted collaborative filtering for improved recommendations. In Aaai/iaai (pp. 187-192).

[7] Mihalcea, R., & Tarau, P. (2004, July). TextRank: Bringing Order into Text. In EMNLP (Vol. 4, pp. 404-411).

[8] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[9] Murty, M. N., & Raghava, R. (2016). Support Vector Machines and Perceptrons: Learning, Optimization, Classification, and Application to Social Networks. Springer.

[10] Rendle, S. (2012). Factorization machines with libfm. ACM Transactions on Intelligent Systems and Technology (TIST), 3(3), 57

[11] Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filtering recommender systems. In The adaptive web (pp. 291-324). Springer Berlin Heidelberg.

[12] Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. Journal of machine learning research, 6(Sep), 1453-1484.

[1] SVM 與 Structure Learning 使用相同特徵。