電子商務研究，2006 年夏季
第四卷，第二期，頁 123-156

決策樹形式知識的合併修剪之研究

馬芳資
龍華科技大學資訊管理學系
林我聰
國立政治大學資訊管理學系

摘要

隨著知識經濟時代的來臨，知識的產生、儲存、應用、整合等已成爲重要的討論議題，本研究將針對知識整合此一議題進行探討；而在知識的呈現方式中，決策樹（Decision Tree）形式知識爲樹狀結構，可用圖形化方式加以呈現，它的結構簡單且易於瞭解，本研究擬針對決策樹形式知識來探討其知識整合的課題。本研究提出一個決策樹合併修剪方法 DTBMPA (Decision-Tree-Based Merging-Pruning Approach)以整合既有／原始的決策樹形式知識；此方法包括三個主要程序：決策樹合併、修剪，和驗證，其做法是先將兩棵原始樹經由合併程序結合成一棵合併樹，再透過修剪程序產生修剪樹，最後由驗證程序來評估修剪樹的準確度。此決策樹合併修剪方法藉由合併程序來擴大樹的知識，再利用修剪程序來修剪合併後樹的過度分支。在本研究的實驗中，合併樹的準確度優於原始一棵樹的比率有 90%，而修剪樹的準確度大於或等於合併樹的比率有 80%。在統計檢定中，合併樹和修剪樹的準確度優於原始一棵樹的準確度達顯著差異；而修剪樹與合併樹的準確度雖無顯著差異。然在節點數的比較上，修剪樹的節點數較合併樹的節點數平均約少了 15%。

關鍵字：知識整合、決策樹、決策樹合併、決策樹修剪

A Decision-Tree-Based Merging-Pruning Approach

Fang-Tz Ma
Department of IM, Lunghwa University of Science and Technology
Woo-Tsong Lin
Department of MIS, National Chengchi University

Abstract

Along with approaching of knowledge economy era, knowledge creation, retaining, application and integration are becoming the important themes for discussion in nowadays. This research tends to focus on discussion of knowledge integration and regarding subjects. In the way of knowledge representation, “decision tree” is the most common type to show the knowledge structure in a tree-shaped graphic. This “decision tree” is considerably simple and easy understanding, thus we focus on decision-tree-based knowledge in connection with knowledge integration theme. Our research proposes an approach called DTBMPA (Decision-Tree-Based Merging-Pruning Approach) to integrate the knowledge of decision trees. There are 3 steps included in this approach. In the merging step, the first step, two primitive decision trees are merged as a merged tree to enlarge the knowledge of primitive trees. In the pruning step, the second step, the merged tree from the first step is pruned as a pruned tree to cut off the bias branches of the merged tree. In the validating step, the last step, the performance of the pruned tree from the second step is validated. In the simulation experiments, the percentage accuracy for the merged tree will have 90% of chance that is greater than or equal to the accuracy for those primitive trees, and the percentage accuracy for the pruned tree will have 80% of chance that is greater than or equal to the accuracy for merged tree. And we also find that the average number of nodes of the pruned tree will have 15% less than that of the merged tree.

Keywords: Knowledge Integration, Decision Tree, Decision Tree Merging, Decision Tree Pruning

壹、前言

在機器學習(Machine Learning)範疇內的歸納學習法(Induction Learning)是從實際發生的案例資料中，以自動化的方式來進行知識的擷取與學習。歸納學習法在適當應用領域中確實可以有效地替代知識工程師，來完成知識擷取的工作。近年來決策樹歸納法已有相當多的學者投入研究，並且將演算法進行改良及提昇其效果(Bolakova, 2002; Fournier & Cremilleux, 2002; Dong & Kothari, 2001; Bradford et al., 1998; Murthy, 1997)。甚至已發展出許多商業軟體，其具有學習決策樹的功能，而且它們成功的應用在解決實際的問題中(Witten & Frank, 2000)。對於經由學習演算法的複雜運算所獲得的決策樹之後續知識整合是值

得進一步研究的課題，於是本研究擬探討決策樹形式知識的合併與修剪問題。

首先有關決策樹形式知識的合併方面，有些學者直接將決策樹轉換爲法則集(Rule Set)，然後再處理法則間重複與衝突，但是這樣的處理方法會犧牲掉原有一棵樹形知識結構的優點，且轉換爲法則後，會引發其間順序的安排、法則修剪、與解決法則衝突等問題。另外有些學者利用投票表決、加權投票、或其他演算法來綜合多棵樹的知識，而這些處理方法並沒有實際整合決策樹形式的知識，僅是外加一個結合策略(Combing Strategy)來綜合多棵樹的預測結果值。Quinlan (1998)以兩兩合併的方式把三棵決策樹合併成一棵決策樹，而他所提出的合併演算法是把第二棵樹取代第一棵樹的葉節點，如此增長了決策樹的路徑(Path)，而且使得合併後的決策樹的節點數大幅成長。爲了減少合併決策樹的節點數，他採用修剪策略，刪除沒有例子落入的葉節點，然而卻使得合併決策樹的平均準確度降低，甚至低於一棵樹的平均準確度。有鑑於此，本研究擬提出一個合併方法，讓決策樹知識結合的同時，不會大幅成長合併樹的節點數，且期能提昇其預測準確度。

其次在修剪決策樹方面，傳統決策樹修剪的目的是爲了避免演算法過於配合訓練資料來進行分群，或是因爲資料內含雜訊而形成過度分支的現象，因而造成產出的決策樹過於龐大而複雜，因此將過度分支進行修剪處理(Frank, 2000)。而學者們提出的修剪方法分爲事先修剪法(Prepruning)和事後修剪法(Postpruning)，其中前者是在建立決策樹的演算法內設定停止分支的條件，以提早結束決策樹的往下分支；而後者是在完全成長決策樹之後，再進行修剪方法，例如成本－複雜修剪法(Cost-Complexity Pruning, CCP)、最小誤差修剪法(Minimum Error Pruning, MEP)、悲觀誤差修剪法(Pessimistic Error Pruning, PEP)、臨界值修剪法(Critical Value Pruning, CVP)、減少錯誤修剪法(Reduced Error Pruning, REP)、和誤差基礎修剪法(Error-Based Pruning, EBP)等等(Mingers, 1989; Frank, 2000)。由於本研究是針對已合併後的決策樹進行修剪處理，是故僅適合採用事後修剪法，期能藉由修剪方法，讓合併樹在保有原有預測能力下，能減少其樹的複雜度(節點數)，換句話說，修剪的目的在於取得一棵較簡潔的合併決策樹。

綜合上述，本研究的目的在於提出一個決策樹形式知識的合併修剪方法。此方法能以兩兩合併的方式，將新的決策樹知識結合到即有決策樹中，並透過修剪程序來精簡合併後的決策樹，期能藉由這個方法來整合決策樹形式知識，以提昇決策樹的預測準確度。

接著，簡述本篇文章的結構：第貳節針對決策樹的合併和修剪進行文獻探討；第參節說明本研究所提出的決策樹合併修剪方法；第肆節說明本研究的實驗設計；第伍節針對實驗結果進行分析；第陸節進行總結和說明後續的研究方向。

貳、文獻探討

決策樹具有樹形資料結構，它是由數個節點(Node)和分支(Edge)組合而成。在決策理論(Decision Theory)之中，決策樹是一個決策圖及它們可能的結果，使用在建立計畫來達成目標。建構決策樹的目的在於輔助決策的制定。在機器學習裡，決策樹是一個預測模型，可用於分類和預測。如圖 1，決策樹的第一個節點稱爲根節點(Root Node)，它是運用決策樹時的唯一進入點（箭頭標示處）；根節點和其下的內部節點(Internal Node)會選取一個屬性來分群（橢圓形符號），換句話說，它們內部會提問一個問題，而節點間是由分支來連結，分支上會有答案的可能值；終端節點稱爲葉節點(Leaf Node)（矩形符號），它們是用來決定預測值(Prediction)，或給定分類的類別(Class)的決策節點(Decision Node) (Witten & Frank, 2000)。接著，本研究對於相關名詞定義、決策樹的合併方法、和修剪處理來進行文獻整理。

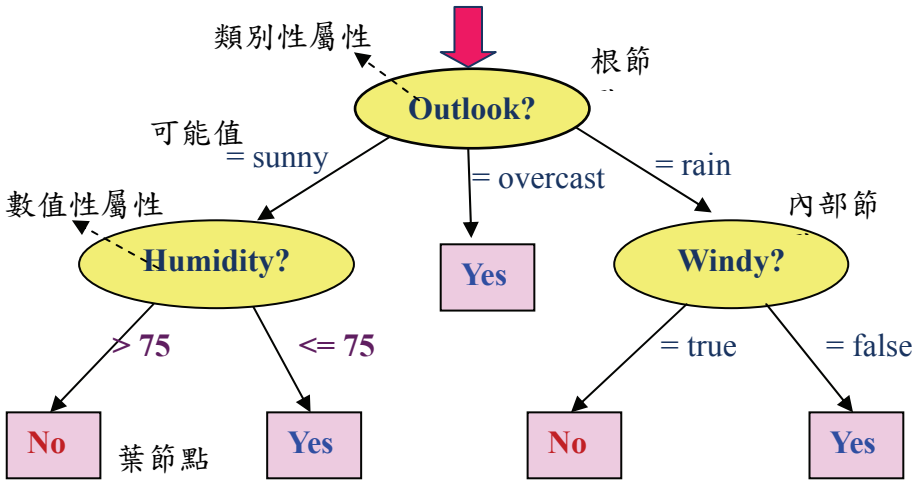


圖 1：決策樹形式知識表示法
資料來源：本研究。此案例請參考(Witten & Frank, 2000，第 8 至 11

一、名詞定義

在此本研究針對決策樹裡的重要名詞進行解釋說明：

(一) 決策樹：它是一棵樹形資料結構，其中根節點和內部節點會標示著一個問題，而從它們往下連結節點的分支上會標示著該問題的可能答案；葉節點內標示著針對整個預測問題的一個預測解答(Dunham, 2003)。本研究以 T 來代表一棵決策樹或它的一個子樹(Subtree)。

(二) 根節點：決策樹的第一個進入節點，在運用決策樹進行分類時，首先進入的節點，其下包含數個或零個子節點。

(三) 分徑屬性：用來進行往下分支所選取的屬性。本研究僅探討選取單一屬性往下分支的決策樹，對於組合多個分徑屬性的決策樹不在本研究的範圍內。

(四) 內部節點：又稱爲非末端節點(Non-terminal Node)、判斷節點(Test Node)。在決策樹的內部節點內會有一個問題來進行往下分群，產生子節點。而此問題爲詢問使用者該節點內的分徑屬性的值。

(五) 葉節點：又稱爲末端節點(Terminal Node)、樹葉(Leaf)、或決策節點。其爲決定最後預測類別的節點，它沒有子節點。本研究以 t 代表一個葉節點。

(六) 子節點(Child Node)：一個節點往下分支所產生的節點，稱爲子節點，而其上層節點稱爲父節點(Parent Node)。一個節點的子節點數目爲兩個或兩個以上。

(七) 分支：連結一個節點與另一個節點的接線，它是有方向性的，係由上而下的連結。在此分支上會標記問題的答案可能值，也就是所有等於此值的例子就會落入往下分群的子節點之中。

(八) 連結節點(Connecting Node)：合併決策樹內在合併擁有相同父路徑的節點所產生的節點，即此節點內的子節點具有相同的父路徑。此節點爲本研究進行合併處理所增加的節點類型，它不同於傳統決策樹的內部節點，在於其下所有分支的可能值都相同，且在運用決策樹來進行預測或分類時，它的每一個分支都必須要走訪，而非傳統內部節點僅選取一個分支去走訪。本研究用雙邊線的橢圓型符號來代表此類型的節點。

(九) 純度：落於葉節點的例子中，以佔最多數的類別之個數除以此葉節點之總例子數來計算。
純度＝（葉節點內佔最多數類別的個數／葉節點內總例子數）* 100%

(十) 強態葉節點：即決策樹之每條由根節點到葉節點之路徑爲一條法則，法則之好壞乃因葉節點之純度及落入此葉節點之例子數多寡來判定，純度愈高且例子數愈多，其法則之代表性較佳，即爲強態葉節點。本研究參考馬芳資 (1995) 針對強態葉節點所進行的實驗分析，其建議最佳的強態法則是設定葉節點的純度達到 85%且落於此葉節點的例子數在 10 筆以上者，亦即其區別能力的確定性程度至少達到 85%。

(十一) 強態葉節點策略：每一條路徑若有選擇連結的節點，其就會多產生一個預測值。在合併兩棵決策樹的情形下，最多會出現兩個預測葉節點，而在綜合這兩個預測葉節點所採用的策略是：首先判斷是否有強態葉節點，若有就以落入強態葉節點之信用良好和信

用不良客戶例子數來判定最終預測值；倘若沒有就判斷所有葉節點的例子數總和是否為零，若是則以系統預設目標值為最終預測值；否則以信用良好和信用不良客戶的例子總數多寡來判定最終預測值，以佔最多者為最終預測值。

(十二) 準確度：以額外的測試例子集來驗證決策樹之正確命中的比率。
準確度＝（判斷正確的筆數／測試例子集個數）* 100%

二、合併決策樹

有關結合多個方法的文獻相當多，本研究僅針對結合多個決策樹形式知識的組合策略進行探討。這些組合策略包括投票表決方式、加權計分方式(加權投票法)、利用法則集來集結決策樹之知識法則、後設決策樹(Meta Decision Trees, MDT)、仲裁樹(Arbiter Tree)、合併樹(Combiner Tree)、以及合併決策樹(Merging Decision Trees)等方法。

採用投票表決的方式來結合多棵決策樹的知識是最廣為使用的方法，Breiman(1996)提出的 Bagging 方法就是採用投票表決的方式來結合多個分類法則(Classifiers)，以得票率最高的類別做為系統預測的類別；加權計分的方式是給予每一棵決策樹不同的權重值，然後乘上其預測目標值而計算出總得分，而得分最高的類別為系統預測的目標類別。此即 AdaBoost 所採用的策略，它是以分類法則 C_i 之誤判率 ε_i 除以 $(1-\varepsilon_i)$ 的對數值為權重值，而以權重總和最大之預測值 y 為最終之預測結果值 $C^*(x)$，換句話說，它是依據分類法則的誤判率大小，把誤判率較小的分類法則，設定較高的權重值 (Bauer & Kohavi, 1999)。

Williams(1990)提出一個多重歸納學習(Multiple Induction Learning, MIL)的方法，係把多棵決策樹的知識轉成法則，然後再把它們組合在一起。由於每一棵決策樹的每一條路徑皆可轉換成一條『若--則』(If-then)的法則，而路徑上的中間節點之判斷條件是以『且(And)』的方式來進行連結，末端葉節點的目標值(類別)則置於『則(then)』的後面，而它是用來進行預測分類結果。Todorovski & Dzeroski(2000)提出後設決策樹的方法來結合多個分類法則的預測值。Chan & Stolfo(1995)提出仲裁樹和合併樹的方法，前者利用仲裁法則來綜合多棵決策樹之預測結果值；後者是利用一棵合併樹來綜合基底層的多棵決策樹的預測值。上述的這些組合方法的共同點就是無法保有原本一棵決策樹的樹狀結構之優點，接著我們探討把多棵決策樹合併成簡單的一棵決策樹之合併策略。

Quinlan(1998)提出 MiniBoosting 演算法改良 Boosting 的方法而成，以結合三棵決策樹的形式來與其他方法進行比較。而他在文章中把三棵原始決策樹，藉由合併的程序，結合成一棵簡單結構的決策樹(請參考圖 2)。Quinlan 採用兩兩合併的方法來結合三棵決策樹 C_1、C_2、和 C_3。

首先用 C_2 取代 C_1 的葉節點，形成 $C_{1,2}$；然後再用 C_3 來取代到 $C_{1,2}$ 的葉節點，最後形成一棵決策樹 $C_{1,2,3}$。

Quinlan 利用 UCI 資料集來進行驗證，在合併三棵樹的規模下，合併後決策樹 $C_{1,2,3}$ 的平均節點數是原本一棵決策樹的平均節點數的 73.09 倍，其中最大的是 174.25 倍，而最小的是 4.04 倍。他進一步去檢視合併後的決策樹，發現其間有很多分支的葉節點並無任何例子落入此節點，因此 Quinlan 進一步修剪掉這些空的葉節點。經由這個簡化(Simplifications)程序，其簡化後合併決策樹 $C_{1,2,3}'$ 的平均節點數約是原本一棵樹的 3.93 倍，其中最大是 12.08 倍，然而它的平均準確度卻變差了，比原本一棵樹還差。

這樣的合併方法主要是用新決策樹來取代舊決策樹的葉節點，如此增長了決策樹的判斷路徑，造成合併後的決策樹的節點數大量增加，雖然它可以保證每一個例子都能落入某一葉節點，但合併樹的法則結構已不同於兩棵原始決策樹的判斷法則，因此本研究擬提出一個新的合併方法來克服這個問題，讓決策樹在進行合併時，能保存原有判斷法則且節點數的增加不會大幅成長。

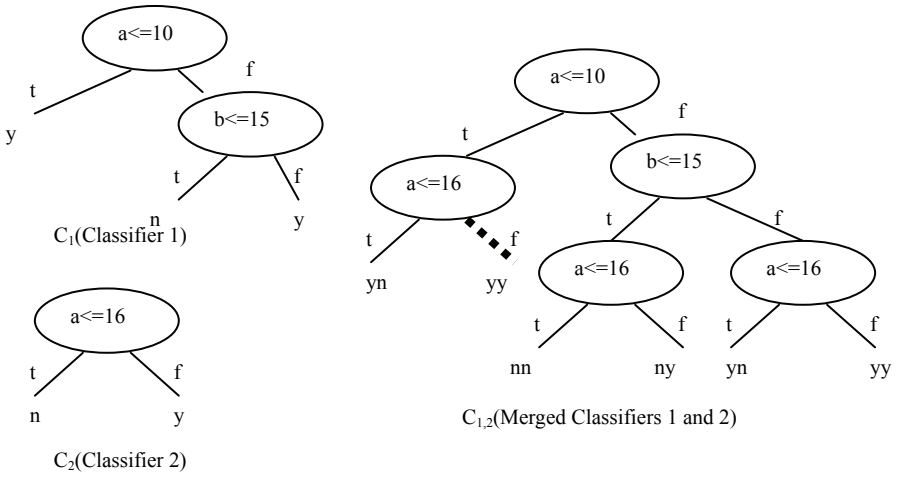


圖 2：Quinlan 合併決策樹演算法

資料來源：Quinlan, J.R., “Mini-Boosting Decision Trees,” AI Access Foundation and Morgan Kaufmann Publishers, 1998.

三、修剪決策樹

本研究針對六種不同的事後修剪策略進行文獻整理，包括成本－複雜修剪法、最小誤差修剪法、悲觀誤差修剪法、臨界值修剪法、減少錯誤修剪法、和誤差基礎修剪法等。這些方法的共通點針對已完全

成長的決策樹進行修剪處理，以及大部份的方法都會評估修剪前(子樹)和修剪後(葉節點)的估計錯誤率，倘若還原成葉節點的估計錯誤率較低，則進行子樹的修剪動作。

Breinman 等(1984)等提出一種兩階段程序的決策樹修剪法，稱爲成本－複雜修剪法，這種方法分爲兩階段，第一階段是產生一連串的樹列($T_0,T_1,T_2,\cdots T_k$)，T_0是原本的決策樹，而以下的 T_{i+1}是從 T_i修剪掉某些子樹而產生的，一直到最後的 T_k樹，它只有一個節點，就是只剩根節點的情形。第二階段是評估這些樹列，並從其中選出一棵最可靠的樹，當做最佳修剪後的樹。此法即 Breinman 等所提出之 CART 決策樹學習演算法中所使用的修剪方法。這個方法的缺點：(1)成本複雜度公式缺乏理論依據，它不像一般評估決策樹的績效指標，例如錯誤率和決策樹大小等容易理解；對於此公式，Breiman 並沒有清楚交代其根源。(2)對於最主要葉節點來取代和樹列的產生，其取捨的程序並不規則。(3)在修剪過程中，需要額外一些例子來進行評估，且運算成本高，較費時(陳重銘，1995)。

最小誤差修剪法是由 Niblett & Bratko(1986)所提出，主要是使期望錯誤率最小化，也就是找出錯誤率最低的修剪後的決策樹，其中有學者使用 Laplace 機率估計(Probability Estimates)來做爲期望錯誤率。這個方法需要另一組修剪例子集做爲評估修剪之用。它的缺點是需要額外的修剪例子集、敏感於類別數目、及在雜訊下不穩定。

悲觀誤差修剪法是由 Quinlan(1986)先生所提出，基於誤判數目和訓練例子的大小，採用二項分配之修正因子來估計錯誤率。首先檢視決策樹內的每一棵子樹，評估其錯誤率是否在一個標準差以內，倘若大於一個標準差的上限，則進行修剪。由於是以較悲觀的上限來做爲決策修剪的依據，所以稱爲悲觀誤差修剪法。優點：(1)修剪的速度比成本複雜修剪法和減少錯誤修剪法較快。(2)不需要額外的修剪例子集。缺點是在雜訊下不穩定，以及由上而下搜尋導致水平效果。

臨界值修剪法是由 Mingers(1989)所提出，它主要是在決策樹成長階段，對每個節點的重要性進行評估。在進行修剪時，設定一個臨界值，如果節點沒有達到此臨界值，便修剪掉此節點。通常臨界值給的愈高，修剪程度也就愈高，而其修剪後的樹也就愈小。臨界值的設定通常視決策樹成長過程中採用何種分徑指標方法，亦可以採用機率值或絕對值，通常機率值以 0.9500 到 0.9995 比較合適；而絕對值則是視訓練例子的性質而定。缺點是很難設定一個客觀適當的臨界值、準確度較差、和需要額外的例子來進行修剪。

減少錯誤修剪法是由 Quinlan(1987)所提出，它的做法是計算葉節點和子樹的猜錯個數，假若其分支葉節點無法降低錯誤率則進行修剪。此法首先建立完全成長的決策樹，然後利用另一組修剪例子集來進行測試此決策樹，記錄每一節點的類別出現的例子數。然後以由下

C.E.P.S.

而上的方式，對於非樹葉的節點，計算如果保留此子樹的錯誤次數和如果修剪掉此子樹的錯誤次數，倘若子樹的錯誤次數大於等於修剪掉後錯誤次數，則進行修剪，如此進行，直到任何修剪會造成錯誤率增加的時候爲止。這個方法較簡單明確，最後產生的決策樹就是符合未知修剪例子的最少錯誤分類數的修剪樹。它的缺點是：(1)需要額外的例子來進行修剪。(2)較稀有的特別例子，若不在修剪例子集中出現，將可能被忽略。

誤差基礎修剪法是由 Quinlan(1992)所提出，爲 C4.5 演算法所預設的修剪方法，它是由下而上檢視每一棵子樹，評估其預測錯誤率與把它修剪成一葉節點時的預測錯誤率，倘若後者得出的值較小，則將此子樹修剪成一葉節點。預測錯誤率是由現有訓練資料的錯誤比率來計算出來。C4.5 在計算預測錯誤率時，是採用二項式分配的機率值，其做法是給定一個可信度(Confidence Factor, CF)，以其二項式分配機率的上限值做爲預測錯誤率的值。此方法的優點是修剪的速度相對而言較其它方法快，且不需額外的修剪例子。本研究採用此修剪方法來針對合併樹進行修剪處理，此乃因爲這個方法不需要額外的修剪例子集，且其在修剪方法的比較之文獻上，有良好的預測效果(Windeatt & Ardeshir, 2001, Frank, 2000, Esposito et al., 1995)。表 1 爲此六個修剪方法之彙整比較表。

表 1：決策樹修剪方法彙整比較表

方法名稱	特徵和策略	修剪例子集※	優點及效益	缺點或限制
1. 最小成本-複雜修剪法 Minimal Cost-Complexity Pruning (Breinman et. al.,1984)	由下而上(檢視內部所有節點) 可以使用交互驗證或標準誤來估計準確度	需要 (0-SE,1-SE) 不需要 (CV)	準確度 (M,E:0-SE) 在 0-SE(E)時可獲得最好的大小的樹	在 1-SE(E)時準確度會不好 不穩定的準確度(CV)(E) 在 1-SE(E)時會過度修剪
2. 最小誤差修剪法 Minimum Error Pruning(MEP) (Niblett 和 Bratko, 1986)	由下而上 基於 M 機率估計錯誤率	需要		敏感於類別數目； 在雜訊下不穩定 低準確度 假定 Uniform 類別分配；需要額外的例子來進行修剪
Laplacian Error Estimation(1996)	使用 Laplace 機率估計來做為期望錯誤率	需要		高複雜度 需要額外的例子來進行修剪

續表 1：決策樹修剪方法彙整比較

3. 悲觀誤差修剪法 Pessimistic Error Pruning(PEP) (Quinlan,1987)	由上而下 使用連續修正因子	不需要	有效率(E) 非常準確(E, 但非 M) 修剪的速度比成本複雜修剪法和減少錯誤修剪法較快 不需要額外的修剪例子集	在雜訊下不穩定 由上而下搜尋導致水平效果
4. 臨界值修剪法 Critical Value Pruning (Mingers, 1989)	由下而上 使用一個臨界門檻值於分徑選擇指標,判斷是否保有此分支子樹	需要		準確度較差 臨界值不易設定 需要額外的例子來進行修剪
5. 減少錯誤修剪法 Reduced Error Pruning (Quinlan, 1987)	由下而上 找最適修剪樹且需修剪例子集	需要	準確度佳(M 但非 E) 好的樹大小(E) 最適修剪例子集(E)	需要額外的例子來進行修剪 較稀有的特別例子，若不在修剪例子集中出現，將可能被忽略
6. 誤差基礎修剪法 Error-Based Pruning (Quinlan, 1992)	由下而上 估計信賴區間,使用訓練例子集	不需要	非常準確 不需要額外的修剪例子集 修剪速度較快	

※ 是否需要額外之修剪例子集；SE 表示標準誤，E 為估計錯誤率，M 為猜錯的個數，CV 指的是交互驗證(Cross Validation)。

※ 資料來源：本研究整理。

參、決策樹合併修剪方法

決策樹形式的知識是一種圖形呈現知識結構的形式，它的結構很簡單且容易理解，層級式組織在推理應用上也十分清楚明確，本研究擬將經由繁複運算所得的決策樹形式知識進行樹狀結構知識的合併與修剪，期能達成樹形知識的整合。本節首先說明決策樹合併修剪方法的運作流程，接著說明決策樹的合併方法，然後再針對決策樹的修剪方法及驗證方法進行說明。

一、運作流程

本研究所提出之 DTBMPA 方法的輸入項是一棵舊的決策樹 T_i 及一棵新的決策樹 T_j，以及用來驗證的測試例子集；而輸出項包括合併樹 T_{ij}、修剪樹 T_{ij}' 及驗證結果值。其包含三個主要的處理程序，即合併程

序、修剪程序及驗證程序。整體運作流程請參考圖 3，首先讀入兩棵決策樹，經由合併程序結合成一棵合併決策樹。然後再將合併決策樹，利用修剪程序，剪掉分類效果不佳的子樹，進而產生修剪過後的決策樹；接著進入驗證程序，即以測試例子集來進行原始決策樹、合併樹、及修剪樹的評估，期能瞭解這些決策樹在預測未見過例子的準確度，即評估其預測能力。分析比較它們的預測結果，選取最佳的決策樹，做爲進行下一次合併修剪之舊的決策樹，以此運作模式，持續地擴充舊的決策樹的知識法則，以提昇其預測能力。圖 4 是以步驟的方式來說明 DTBMPA 方法的處理程序。接著，我們依序討論合併方法、修剪方法、和驗證方法之處理過程。

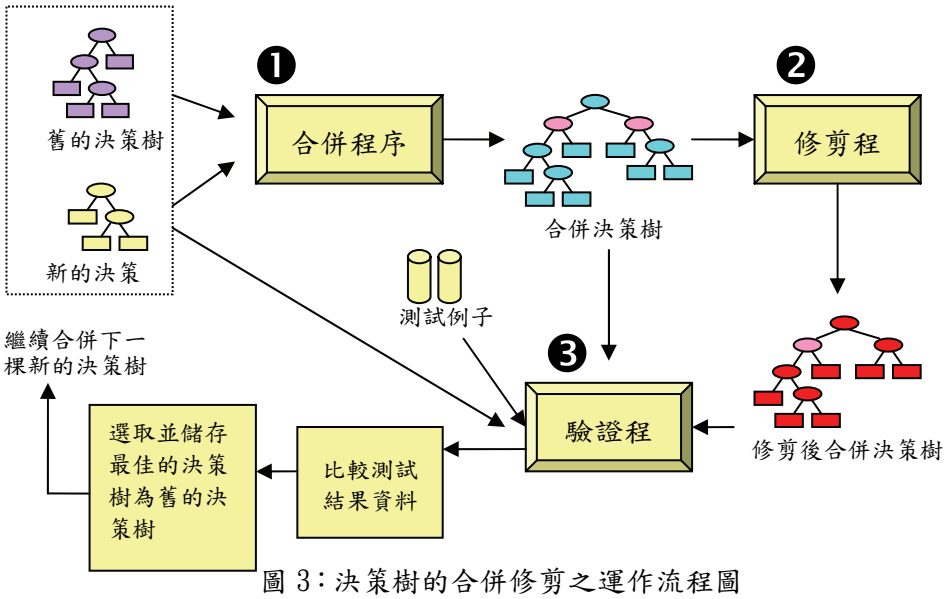


圖 3：決策樹的合併修剪之運作流程圖

輸入：舊的決策樹 T_i 、新的決策樹 T_j、及測試例子集
輸出：合併樹 T_{ij}、修剪樹 T_{ij}'及驗證結果值

1. 讀入兩棵決策樹資訊： T_i 和 T_j。
2. 合併新的決策樹 T_j，到舊的決策樹 T_i 之中；合併後的結果儲存於 T_i。
3. 重新設定決策樹 Ti 的節點編號並輸出決策樹 T_i 為 T_{ij}。
4. 修剪決策樹 T_i 並輸出決策樹 T_i 為 T_{ij}'。
5. 讀入測試例子集並進行驗證決策樹 T_i、T_j、T_{ij} 和 T_{ij}'。
6. 分析比較各組預測準確度，選取最佳的決策樹，且把它儲存為舊的決策樹，以進行下次的合併修剪。

圖 4：決策樹合併修剪方法

二、決策樹合併方法

決策樹的合併方法即利用接枝(Grafting)的技術，把新的決策樹中不同於原有決策樹的分支，利用一個連結節點，接枝到原有決策樹，以擴增原有決策樹的知識。本研究所採用的合併方法係由上而下比對兩棵決策樹，即以前序式走訪相對應的節點，當發現兩個節點型態不同，或型態相同但選取不同分徑屬性時，則在原有決策樹內新增一個連結節點，將兩個比對節點及其下所有分支子節點，接枝到此連結節點下，而成爲其子節點。圖 5 爲決策樹合併方法之處理程序。

(一) 符號定義：

以下符號定義是用來說明圖 5 之處理內容：

(1) T_i：用來表示一棵決策樹或一個子樹。

(2) N_i：用來表示一個節點。

(3) Root(T_i)：傳回決策樹 Ti 的根節點。

(4) Brother(n_i)：傳回節點 ni 的右側兄節點。

(5) Child(n_i)：傳回節點 ni 的最左的子節點。

(6) isSame(n_i, n_j)：比較節點 n_i 和 n_j 是否相等，而判斷相等的做法是兩節點有相同的流入分支且選取相同的分徑屬性，並有相同的流出分支。

(7) isRoot(p, T_i)：判斷節點 p 是否爲決策樹 T_i 的根節點。

(8) hasLeaf(n_i)：判斷節點 n_i 是否有葉節點。

(9) hasBrother(n_i)：判斷節點 n_i 是否有右側兄節點。

(10) hasFather(n_i)：判斷節點 n_i 是否有父節點。

(11) setPure(n_i)：重新計算節點的純度。

(12) setGoal(n_i)：重新找出節點內的佔最多數例子的類別。

(13) setLinks(p, q, op)：設定節點 p、q 和連結節點 op 的指標連結關係。

(14) newNode(op)：建立一個新的連結節點 op。

(15) sumCases(n_i, n_j)：計算節點 n_i 和 n_j 合併後的例子數。

(16) OpFlg：此旗標是用來設定是否有產生連結節點的情形，若其值爲 true 表示需要產生連結節點，否(false)則表示不需要。

C.E.P.S.

(17) findFlg：此旗標是用來設定是否有找到下一個比對的節點，若其值爲 true 則表示已找到，否(false)則表示未找到。

```
1. Initialize: Let ni = Root(Ti); Let nj = Root(Tj); Let OpFlg=False.
2. Compare the nodes in Ti and Tj in preorder traversal.
   Do
      IF isSame(ni, nj) THEN //比對兩個節點是否相同
         //兩節點相同時
         sumCases(ni, nj); setPure(ni); setGoal(ni);
         IF hasLeaf(ni) AND hasLeaf(nj) THEN //判斷兩者是否有
      葉節點
            ni=Child(ni); nj=Child(nj);
            next loop;
         END IF
      ELSE
         //兩節點不同時
         p=ni; q=nj; OpFlg=True;
               //判斷是否在根節點就不同
         IF isRoot(p, Ti) AND isRoot(q, Tj) THEN
            newNode(op); setLinks(p, q, op);
            Return op;
         END IF
      END IF
      //屬找下一個比對的節點
      IF hasBrother(ni) AND hasBrother(nj) THEN
         ni=Brother(ni); nj=Brother(nj);
      ELSE
         Let findFlg=False
         WHILE (hasFather(ni) AND hasFather(nj) AND
      findFlg=False)
                  ni=Father(ni); nj=Father(nj);
                  IF hasBrother(ni) AND hasBrother(nj) THEN
                     ni=Brother(ni); nj=Brother(nj); findFlg=True;
                  END IF
               END WHILE
      END IF
      IF OpFlg THEN
                newNode(op); //新增一個連結節點
                setLinks(p, q, op); //設定新的分支連結關係
      END IF
   Until (isRoot(ni, Ti) AND isRoot(nj, Tj)); //判斷是否已完成節點
   比對
```

圖 5：決策樹合併方法

C.E.P.S.

(二) 範例說明：

假設我們有兩棵決策樹－T_1 (左)和 T_2 (右)，兩棵樹皆有七個節點，四個葉節點(請參考圖 6)；它們是屬於相同問題的決策樹，即來自相同的母體的例子集所歸納出的決策樹。它們有三個分徑屬性 A、B、和 C，且每個屬性皆有兩個可能值(小括弧內標示的即為可能值)，也就是 A(a_1,a_2), B(b_1,b_2), C(c_1,c_2)；預測目標類別有兩個，即☺(信用良好客戶)和☻(信用不良客戶)。

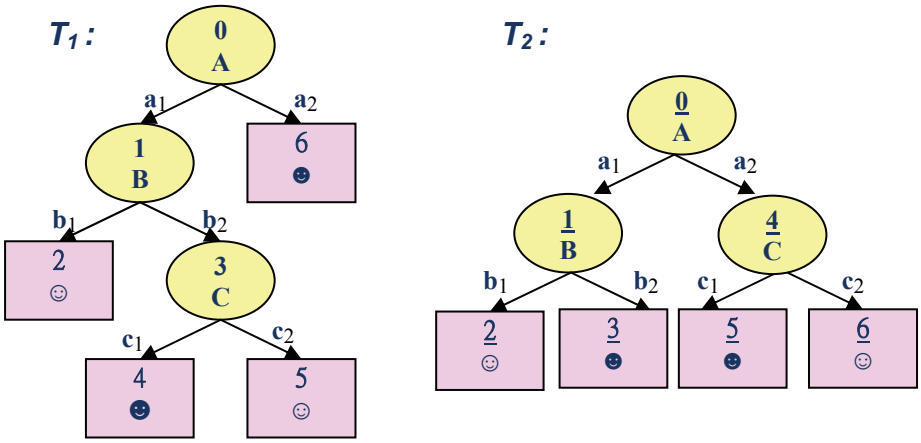


圖6：兩棵原始決策樹， T_1 (左), T_2 (右).

註：1.矩形框表示葉節點、橢圓形框表示根節點或內部節點。矩形框或橢圓形框內上方的數字表示節點的編號，下方的英文符號表示分徑屬性，各箭頭上的數值表示分徑屬性的可能值。

2.分徑屬性(可能值)：A(a_1,a_2), B(b_1,b_2), C(c_1,c_2)

3.預測目標類別：☺(信用良好客戶),☻(信用不良客戶)

合併方法係從兩棵樹的根節點開始比對，首先比對的是節點 0 和節點 0，兩者皆為內部節點，選取相同的屬性 A 來進行往下分支，所以兩者是相同的節點；接著走訪其最左邊子節點；比對節點 1 和節點 1，發現兩者皆為內部節點且選取相同的分徑屬性 B，所以繼續走訪其最左邊子節點；比對節點 2 和節點 2，發現兩者皆為葉節點，是故屬相同的節點，由於是終端節點，所以合計其節點內之例子數，並以佔最多數例子的類別做為其預測目標值(請見圖 7a 、圖 7b、和圖 7c)。

圖7a：完成第一條路徑的節點比對(比對第一個節點)

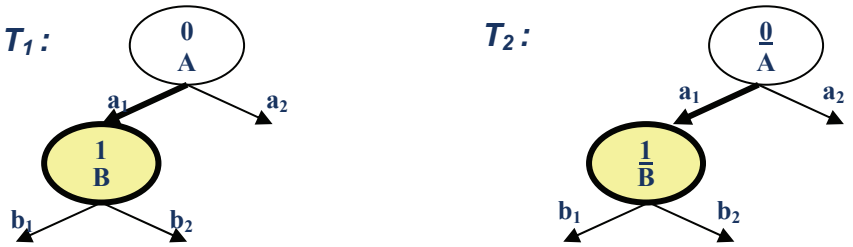


圖7b：完成第一條路徑的節點比對(比對第二個節點)

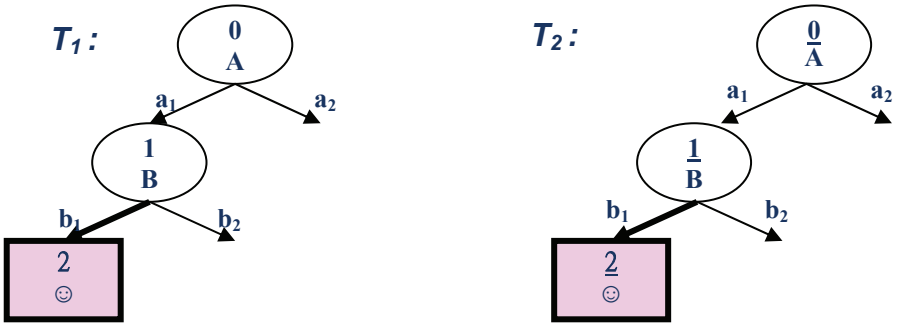


圖7c：完成第一條路徑的節點比對(比對第三個節點)

由於已到最左分支的底部，於是走訪右側兄節點，比對節點 3 和節點 3，發現節點 3 爲內部節點，而節點 3 爲葉節點，兩者是不相同，於是把節點 3 接枝到 T_1，做法是新增一個連結節點，做爲節點 3 和節點 3 的父節點(請見圖 8 和圖 9)。

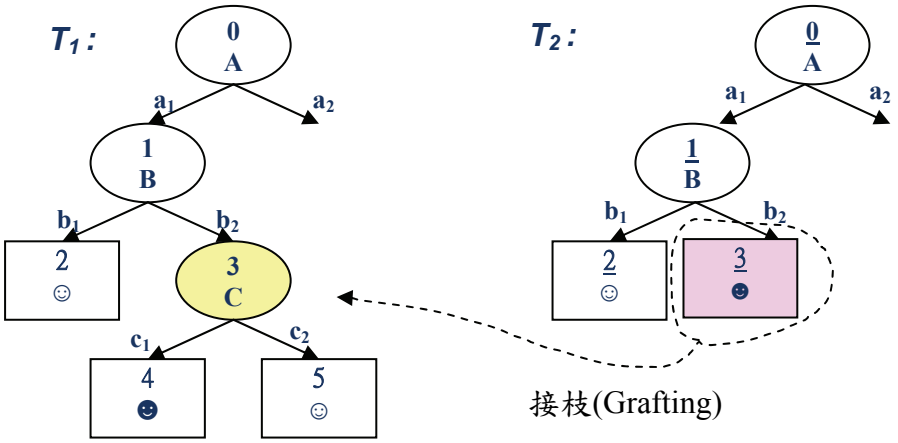


圖8：完成第二條路徑的節點比對

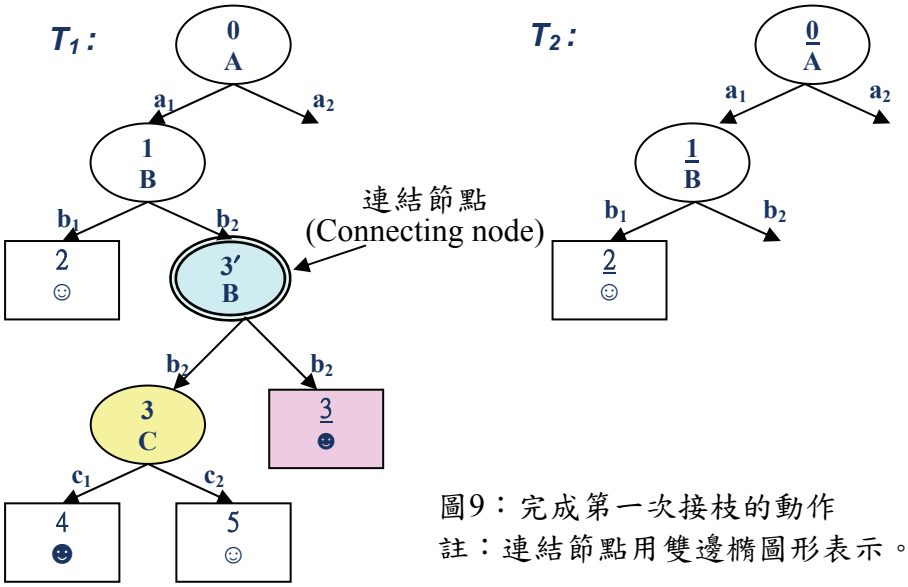


圖9：完成第一次接枝的動作
註：連結節點用雙邊橢圖形表示。

在進行接枝動作之前，就會先尋找下一個欲進行比對的節點，由於發現兩節點不同，是故其下的子節點就不需要再比對了，於是走訪節點 3 和節點 3 的父節點，判斷其是否有兄節點，於是走訪至節點 6 和節點 4；比對兩者，發現節點 6 是葉節點，而節點 4 是內部節點，兩節點不相同，於是進行第二次的接枝動作，要把節點 4 及其子節點接枝到 T_1(請參見圖 10 和圖 11)。

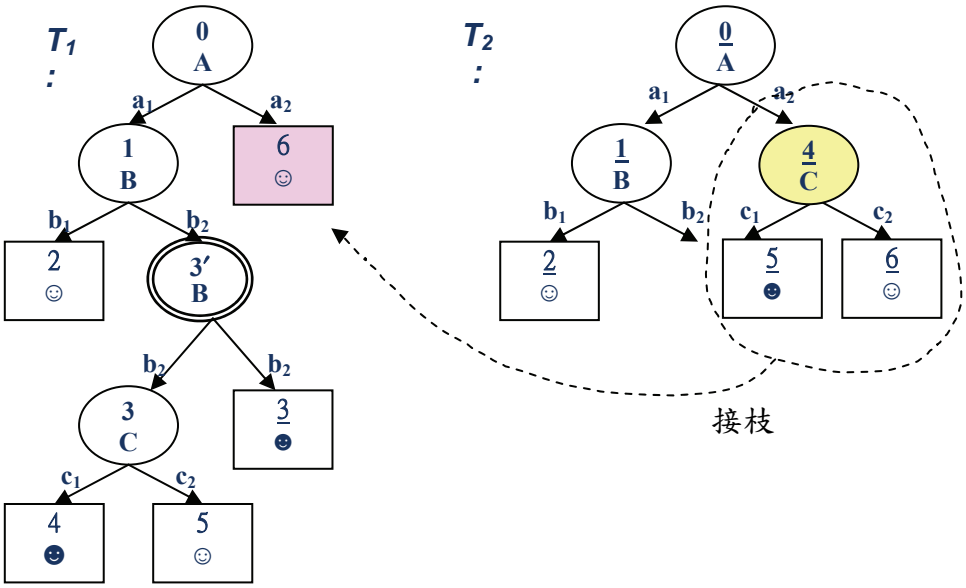


圖10：完成第三條路徑的節點比對

首先新增一個連結節點，做爲節點 6 和節點 4 的父節點，然後設定它們之間的連結關係，如此完成了第二次的接枝動作(請參考圖 11)。在接枝之前，尋訪下一個比對節點時，發現節點 6 已無兄節點而繼續走訪其父節點，發現父節點即爲根節點，而節點 4 亦同此情形，於是乎已完成兩棵樹之節點合併作業。合併後之決策樹儲存於 T_1 之中，如此做法亦可節省記憶體，有助於龐大的決策樹的合併處理。

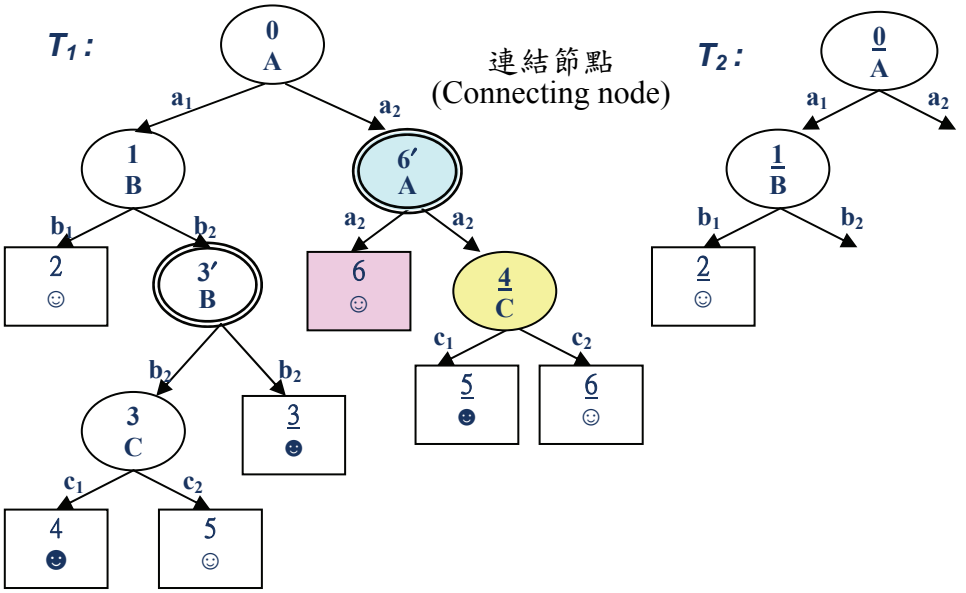


圖11：完成第二次接枝的動作

接枝動作會新增一個連結節點，重設節點的連結關係，且會計算兩合併子樹之例子總合及佔多數例子的類別和其例子數，然後儲存在連結節點之中，而此連結節點所在位置如同一般內部節點，只是其分徑屬性是同於其父節點，且其下分支的可能值皆相同。如圖 11，節點 6′的分徑屬性 A 與其父節點 0 相同，而節點 6′的分支可能值皆爲 a_2。

完成合併後，仍需進行一項動作，即節點重新編號，由於部份節點是由第二棵樹接枝過來，且又新增了連結節點，所以合併樹必須重新將其節點以前序式走訪來進行編號，如此有助於合併樹日後進行合併其他決策樹之作業。完成後的合併樹如圖 12，其中粗線的邊框的節點即新增的節點，換句話說，即藉由合併作業，新增的判斷法則，期能以此擴增其知識法則，進而提昇決策樹的預測能力。

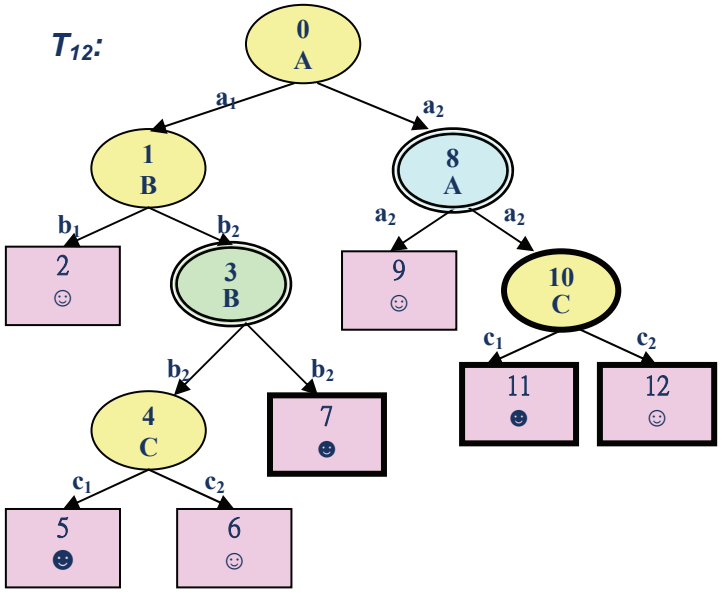


圖12：完成兩棵樹的合併動作

三、合併樹修剪方法

爲避免合併決策樹經由合併程序而愈來愈大，所以我們對於合併決策樹進行修剪。本研究係由下而上的方式，計算每一個子樹之估計錯誤率，及把它還原成葉節點之錯誤率，倘若分支無法降低錯誤率則進行修剪，也就是說，子樹的錯誤率大於還原成葉節點的錯誤率時，就把此子樹還原成一個葉節點。

(一) 葉節點之估計錯誤率（Estimated Error Rate, e_t）

$$e_t = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}, \text{（公式１）}$$

其中 E 是錯誤分類的例子數，N 是此落入此葉節點的總例子數，f=E/N 此爲傳統計算節點錯誤率的公式；當信賴水準爲 25%，則 z 值爲 0.69。

(二) 子樹之估計錯誤率 e_T

$$e_T = \sum_{i=1}^{k} \frac{N_i}{N} e_i, \text{（公式２）}$$

其中 N 是子樹 T 的根節點的總例子數，N_i是 T 的第 i 子節點的例子數，而 T 有 k 個子節點。

本研究以一個範例來說明修剪之處理方式，系統是以由下而上的方式檢視決策樹內的每一個子樹，計算其 e_T 和 e_t 值；倘若 e_t 小於 e_T 時，則進行修剪，即把子樹 T 還原成一個葉節點 t。以圖 13 爲例，其中內部節點右上方及葉點節正下方之小括弧所標示的數字，代表該節點的信用良好和信用不良之例子數。由下而上的處理，首先找到的是子樹 T_4，於是利用公式１計算 e_{t4} 的值（請參考算式１），即還原成葉節點後的估計錯誤率；然後利用公式２計算 e_{T4} 的值（請參考算式２、３、和４），即保留子樹時的估計錯誤率，結果發現 e_{T4} (=0.487) 小於 e_{t4}(=0.591)，換句話說，保留子樹 T_4 可以降低錯誤率，所以不進行修剪 T_4。以此類推，當檢視到根節點時，即完成整棵樹的修剪作業。

$$e_{t4} = \frac{\frac{7}{14} + \frac{(0.69)^2}{2\times 14} + (1.69)\sqrt{\frac{(7/14)}{14} - \frac{(7/14)^2}{14} + \frac{(7/14)^2}{4\times(14)^2}}}{1 + \frac{(0.69)^2}{14}} = 0.590676 \text{（算式１）}$$

$$e_{t5} = \frac{\frac{3}{8} + \frac{(0.69)^2}{2\times 8} + (1.69)\sqrt{\frac{(3/8)}{8} - \frac{(3/8)^2}{8} + \frac{(3/8)^2}{4\times(8)^2}}}{1 + \frac{(0.69)^2}{8}} = 0.4969736 \text{（算式２）}$$

$$e_{t6}=\frac{\frac{2}{6}+\frac{(0.69)^2}{2\times 6}+(0.69)\times\sqrt{\frac{(2/6)}{6}-\frac{(2/6)^2}{6}+\frac{(2/6)^2}{4\times(6)^2}}}{1+\frac{(0.69)^2}{6}}=0.4739882 \quad \text{（算式３）}$$

$$e_{T4}=\frac{8}{14}\times(0.49697369)+\frac{6}{14}\times(0.47398827)=0.487123 \quad \text{（算式４）}$$

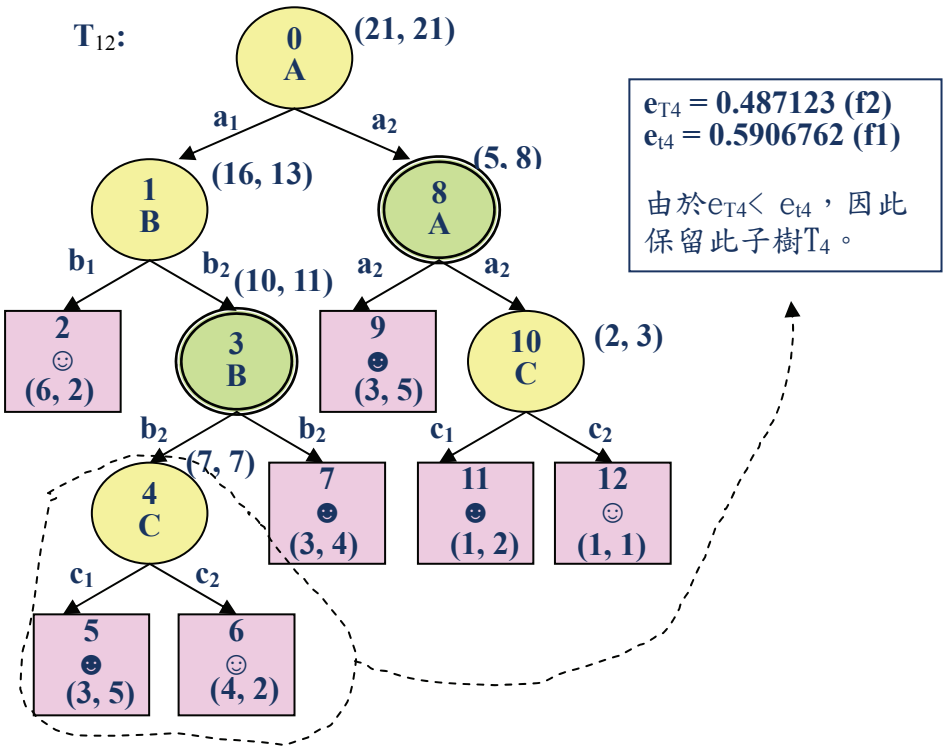


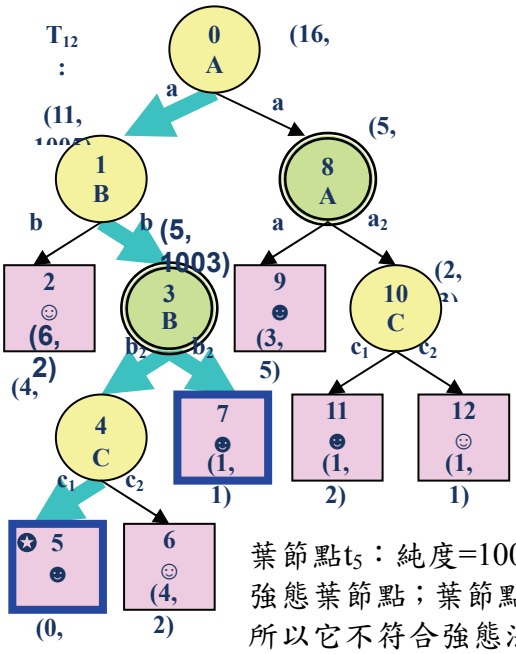
圖13：合併樹之修剪處理

四、合併樹驗證方法

本研究利用額外(Hold-out)的測試例子來進行合併樹之驗證，而由於合併樹存在連結節點，不同於傳統之決策樹結構，因此在進行測試時的處理方式也會不同。傳統決策樹在測試時，僅選取一條路徑由根節點走到葉節點，然後利用葉節點來進行分類和預測。然而在合併樹的測試中，其連結節點下的每一條路徑皆須走訪。假設連結節點有 k

個分支，則路徑上每增加一個連結節點就會增加 k 條路徑。再者，本研究提出一個強態葉節點策略，即當葉節點是屬於強態葉節點（純度在 85%以上，且例子數在十筆以上）時，才能參與判定最終預測值，而藉由篩選代表性高的葉節點來進行預測，可提昇合併樹之預測準確度。

舉例來說，在圖 14 中有一個例子 x 要來進行分類，其屬性的可能值分別為 A 等於 a_1、B 等於 b_2、和 C 等於 c_1。利用合併樹 T_{12} 來進行例子 x 的目標值預測，首先從根節點開始，其分徑屬性為 A，而例子 x 的屬性 A 的值為 a_1，所以走訪至節點 1；接著節點 1 的分徑屬性為 B，而例子 x 的屬性 B 的值為 b_2，所以走訪至節點 3，而節點 3 為連結節點，所以其下的兩個分支皆必須走訪。由左邊先走，到節點 4，其分徑屬性為 C，而例子 x 的屬性 C 的值為 c_1，所以走訪至節點 5，得到一個預測值為信用不良客戶，而此節點 5 內例子數為一千筆，且純度為 100%，是一個強態葉節點。然後再回到連結節點 3，走訪其右邊分支，到節點 7，得到第二個預測值為信用不良客戶，而此節點內信用良好與信用不良客戶例子數相同，且僅有一筆，此葉節點之代表性不高，因為純度僅有 50%，且總例子數僅兩筆，於是不考慮其預測值，即節點 7 並非為強態葉節點。因此 T_{12} 預測例子 x 是根據節點 5 的預測值，判定其為信用不良客戶。



例子：x
屬性A = a_1
屬性B = b_2
屬性C = c_1
目標值 = ?

兩條路徑如下：

- t_0 -> t_1 -> t_3-> t_4 –> t_5
- t_0 -> t_1 -> t_3 -> t_7

葉節點t_5：純度=100%、例子數=1000，所以它是一強態葉節點；葉節點t_7：純度=1/2=50%、例子數=2，所以它不符合強態法則，不是一個強態葉節點，因此不用於進行預測。是故最終的預測值為☻ 信用不良客戶。

圖14：合併樹之驗證處理

上述說明使用合併樹預測一筆例子的情形，然而對於整棵樹的績效評估，是以額外的測試例子集來評估樹的預測準確度，並以此做爲比較不同決策樹之優劣的基礎。

肆、實驗設計

本研究以發卡銀行之信用卡客戶歷史信用資料來進行實驗，以瞭解本研究所提出之 DTBMPA 方法是否有助於提昇系統預測之準確度。本節針對樣本資料、實驗流程、和研究假設及限制來進行說明。

一、樣本資料

本研究採用發卡銀行之信用卡客戶歷史信用資料來進行實驗，經過資料整理，刪除無效樣本及空缺值，共計可用樣本爲 103,653 筆。在目標值選取上，本研究以現狀碼等於零之正常流通卡爲信用良好之客戶，而以現狀碼等於壹之強制停卡者爲信用不良之客戶。

實驗樣本內所搜集到的屬性包括年齡、學歷、年收入、有無甲存、有無不動產、行業別、公司等級、職等、一般卡張數及金卡張數，以及帳款餘額和六個月的繳款記錄等，共計有十七個；其中年齡、年收入及帳款餘額爲數值性線索，其餘爲非數值性線索。期能讓系統充分學習信用良好與信用不良客戶之特徵，所以將此十七個屬性完全納入學習之範圍。

研究母體共有 103,653 筆，其中信用良好客戶有 100,763 筆，信用不良客戶有 2,890 筆。本研究採用隨機抽樣的方式，以不放回式抽出二十組訓練例子集，每組 200 筆例子，信用良好與信用不良客戶比爲 1:1，做爲產生原始決策樹之用。接著再隨機抽出七組測試例子集，每組 200 筆，以做爲驗證決策樹之準確度。

二、實驗流程

本研究所提出之 DTBMPA 方法之輸入項是原始決策樹，然而因爲決策樹取得不易，所以本研究採用原始資料放入 Weka 系統內，以其 J48 方法來產生實驗用之決策樹(Witten & Frank, 2000)。J48 方法係以 Java 語言來撰寫之決策樹歸納學習系統，其內處理方式相同於 Quinlan(1992) 先生的 C4.5 方法。Weka 機器學習系統提供一般用途的自動化分類、迴歸、聚集和屬性選取等功能，它可應用於解決資料探勘(Data Mining) 的問題(Frank et. al., 2004)。

本研究首先從原始資料庫中隨機抽樣出二十組訓練例子集，接著把它們轉檔爲 J48 系統可讀之 arff 檔案格式，然後放入 J48 系統中歸納學習出決策樹。由於其輸出爲敘述決策樹圖形結構的文字檔，所以本研究用 C 語言撰寫一轉換程式，將 J48 輸出的決策樹的文字檔，轉換

成 DTBMPA 系統可讀取之格式，此格式仍是以文字檔儲存，但內容每一列即描述決策樹內的每一個節點的資訊，包括節點編號，型態，目標值，選取屬性編號，節點例子數，目標值例子數，屬性可能值編號，數值性分割點，父節點編號，兄節點編號，子節點編號等。

接著本研究把二十棵原始決策樹以兩兩一組，分別放入 DTBMPA 系統中進行合併，然後產生出十棵合併決策樹；完成後，再把此十棵合併決策樹利用 DTBMPA 的修剪方法進行合併決策樹的修剪，然後再利用七組測試例子集進行原始決策樹、合併樹、及修剪樹之驗證，測試完成後，進行統計分析它們的測試結果。

三、研究假設及限制

(一) 所有屬性可能值限於發卡銀行既有之分類方式且所有屬性限於該發卡銀行的電腦系統所儲存之資料，以及假設其內部儲存之資料是由客戶正確填寫且行員正確輸入之資料。

(二) 本研究採用 J48 方法來產生決策樹，且以其預設之參數來進行學習。

(三) 本研究所提出的 DTBMPA 方法，其輸入項是決策樹，是故當存在有兩棵或兩棵以上的決策樹，即可進行決策樹的合併及修剪；而其中決策樹的來源，可由例子集來歸納產生，亦或是由知識庫內的法則轉換而成。

(四) 本研究所提出的 DTBMPA 方法是針對同一問題領域的決策樹形式知識的合併修剪處理，且僅探討選取單一屬性分群的決策樹形式知識。

伍、結果分析

本研究以七組額外的測試例子集來驗證二十組原始決策樹、十棵合併樹、及十棵修剪後的合併樹，總共取得二百八十次的驗證結果。首先本研究採用成對樣本檢定來進行分組比較，包括一棵樹與合併樹的準確度、合併樹與修剪樹的準確度、及一棵樹與修剪樹的準確度之成對樣本檢定；接著再將三組準確度進行無母數的 Wilcoxon 符號等級檢定；然後針對三組決策樹的節點數進行分析比較，最後說明實驗的結論。

C.E.P.S.

一、原始樹與合併樹之準確度比較

假設 u1 是平均一棵原始樹 T_i 之平均預測準確度，而 u2 是合併樹 T_{ij} 之平均預測準確度，且由於其符合常態分配，而每一對內之 T_i 和 T_{ij} 的準確度則是相依的，其相關係數爲 0.633。計算結果得知誤差的平均數是-3.62E-02 及誤差的標準差(Sd)爲 2.81E-02。本研究所欲進行之雙尾 t 檢定如表 2 所示：

H_0：(u1-u2) = 0
H_1：(u1-u2)≠0

在顯著水準 α 值爲 0.05 且自由度是 69 的情況下，由表 2 得知上界爲-2.95E-02 及下界爲-4.29E-02，而 p 值 0.00 小於 α 值爲 0.05，所以拒絕 H_0，即表示兩者有顯著的差異，而其平均值小於零，由此可見合併樹 T_{ij} 的準確度優於平均一棵原始樹 T_i 的準確度。

表 2：平均一棵原始樹與合併樹成對樣本檢定

成對變數差異					t	自由度	顯著性(雙尾)
平均數	標準差	平均數的標準誤	差異的 95% 信賴區間				
			下界	上界			
-3.617857E-02	2.81002E-02	3.35862E-03	-4.287884E-02	-2.947831E-02	-10.772	69	.000

二、合併樹與修剪樹之準確度比較

假設 u1 是合併樹 T_{ij} 之平均預測準確度，而 u2 是修剪樹 T_{ij}' 之平均預測準確度，且由於其符合常態分配，而每一對內之 T_{ij} 和 T_{ij}' 的準確度則是相依的，其相關係數爲 0.806。計算結果得知誤差的平均數是-3.35E-03 及誤差的標準差(Sd)爲 1.51E-02。本研究所欲進行之雙尾 t 檢定如表 3 所示：

H_0：(u1-u2) = 0
H_1：(u1-u2)≠0

在顯著水準 α 值爲 0.05 且自由度是 69 的情況下，由表 3 得知上界爲 2.32E-04 及下界爲-6.95E-03，而 p 值 0.066 大於 α 值爲 0.05，所以接受 H_0，即表示兩者無顯著的差異，然若將 α 值設爲 0.1 時，就會達顯著差異；再者，兩者相差的平均數爲-3.36E-03，由此可見修剪樹的準確度略勝合併樹的準確度。

C.E.P.S.

表 3：合併樹與修剪樹成對樣本檢定

成對變數差異					t	自由度	顯著性(雙尾)
平均數	標準差	平均數的標準誤	差異的 95% 信賴區間				
			下界	上界			
-3.357143E-03	1.50536E-02	1.79924E-03	-6.946538E-03	2.32252E-04	-1.866	69	.066

三、原始樹與修剪樹之準確度比較

假設 u1 是平均一棵原始樹 T_i 之平均預測準確度，而 u2 是修剪樹 T_{ij}' 之平均預測準確度，且由於其符合常態分配，計算結果得知誤差的平均數是-3.95E-02 及誤差的標準差(Sd)爲 3.32E-02。本研究所欲進行之雙尾 t 檢定如表 4 所示：

H_0：(u1-u2) = 0

H_1：(u1-u2)≠0

表 4：平均一棵原始樹與修剪樹成對樣本檢定

成對變數差異					t	自由度	顯著性(雙尾)
平均數	標準差	平均數的標準誤	差異的 95% 信賴區間				
			下界	上界			
-3.953571E-02	3.31862E-02	3.96650E-03	-4.744867E-02	-3.162275E-02	-9.967	69	.000

在顯著水準 α 值爲 0.05 且自由度是 69 的情況下，由表 4 得知上界爲-3.16E-02 及下界爲-4.74E-02，而 p 值 0.00 小於 α 值爲 0.05，所以拒絕 H_0，即表示兩者有顯著的差異，而其平均值小於零，由此可見修剪樹 T_{ij}' 的準確度優於平均一棵原始樹 T_i 的準確度。

四、Wilcoxon 符號等級檢定

本研究針對三組樣本進行無母數的 Wilcoxon 符號等級檢定，得知以下檢定結果數據。在合併樹與一棵樹的準確度比較方面，合併樹優於一棵樹的準確度有 63 次，比率爲 90%；而合併樹小於一棵樹的準確度有 6 次，比率爲 8.6%；兩者平手的次數有 1 次，比率爲 1.4%。詳細結果如表 5 所示：

表 5：合併樹與一棵樹之 Wilcoxon 符號等級檢定

Wilcoxon 符號等級檢定		個數	等級平均數	等級總和
合併樹 - 一棵樹	負等級	6(a)	12.83	77.00
	正等級	63(b)	37.11	2338.00
	等值結	1(c)		
	總和	70		
a 合併樹<一棵樹；b 合併樹>一棵樹；c 一棵樹=合併樹				

在修剪樹與合併樹的準確度比較方面，修剪樹優於合併樹的準確度有 19 次，比率為 27.1%；而修剪樹小於合併樹的準確度有 14 次，比率為 20%；兩者平手的次數有 37 次，比率為 52.9%。由此可見，兩者的準確度大部份相當；然若以修剪樹大於等於合併樹的準確度來看，其次數總合有 56 次，比率高達 80%，所以修剪樹在大部份的情形下，仍有較佳的表現。詳細結果如表 6 所示：

表 6：修剪樹與合併樹之 Wilcoxon 符號等級檢定

Wilcoxon 符號等級檢定		個數	等級平均數	等級總和
修剪樹 - 合併樹	負等級	14(d)	15.50	217.00
	正等級	19(e)	18.11	344.00
	等值結	37(f)		
	總和	70		
d 修剪樹<合併樹；e 修剪樹>合併樹；f 合併樹=修剪樹				

在修剪樹與一棵樹的準確度比較方面，修剪樹優於一棵樹的準確度有 60 次，比率為 85.7%；而修剪樹小於一棵樹的準確度有 8 次，比率為 11.4%；兩者平手的次數有 2 次，比率為 2.9%。由此可見，修剪樹的準確度大於等於一棵樹的準確度的比率為 86%。詳細結果如表 7 所示：

C.E.P.S.

表 7：修剪樹與一棵樹之 Wilcoxon 符號等級檢定

Wilcoxon 符號等級檢定		個數	等級平均數	等級總和
修剪樹 - 一棵樹	負等級	8(g)	10.25	82.00
	正等級	60(h)	37.73	2264.00
	等值結	2(i)		
	總和	70		
g 修剪樹<一棵樹；h 修剪樹>一棵樹；i 一棵樹=修剪樹				

接著，以檢定統計量來看三者之兩兩比較，由表 8 中得知，在顯著水準 α 值爲 0.05 的情況下，合併樹顯著優於一棵樹的準確度；修剪樹亦顯著優於一棵樹的準確度；而修剪樹與合併樹之準確度並無顯著差異。

表 8： Wilcoxon 符號等級檢定的檢定統計量

檢定統計量(b)			
	合併樹 - 一棵樹	修剪樹 - 合併樹	修剪樹 - 一棵樹
Z 檢定	-6.761(a)	-1.146(a)	-6.668(a)
漸近顯著性 (雙尾)	.000	.252	.000
a 以負等級為基礎。			
b Wilcoxon 符號等級檢定			

五、節點數分析比較

本研究分析三組之節點數，發現兩棵原始樹與合併樹之節點數相近，而修剪樹的節點數小於合併樹和兩棵原始樹的節點數約 15%。由於此二十棵原始樹之選取的分徑屬性差異較大，所以在合併時，十組中有九組在比對兩棵樹時，發現兩棵樹在根節點時，就選取不同的分徑屬性，因此合併樹的節點數無法降低。平均而言，修剪樹的節點數約是合併樹的 85%，減少約 15%(請參考圖 15)。

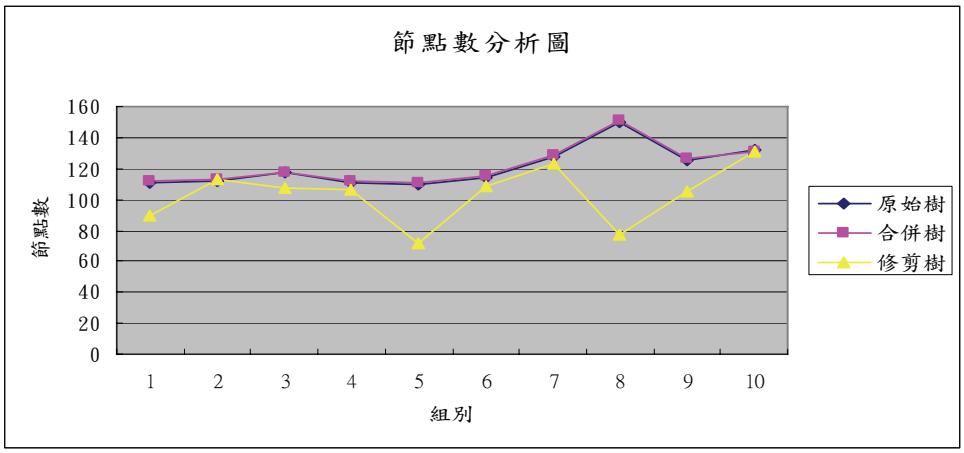


圖 15：節點數比較圖

六、運算成本的探討

關於本研究所提之 DTBMPA 方法的時間複雜度(Time Complexity)，我們依下列兩種情形來進行探討：

(一)已有決策樹而無原始例子集的情形：

當有現成的兩棵決策樹時，我們可以直接利用 DTBMPA 方法來進行樹的合併修剪，以產生一棵合併決策樹。而對於決策樹的來源，可以是由知識庫內法則轉換而來，或由人類領域專家訪談而來。此爲一般情況，因決策樹經常是藉由彙整專家之經驗法則、知識而得來，而非藉由原始資料產生；或原始資料已不可考；此時由於沒有原始資料，因此無法比較採用本研究提出之 DTBMPA 方法所產生決策樹與合併原始資料所產生決策樹的運算成本。

(二)同時擁有決策樹及其原始例子集的情形：

對於同時擁有決策樹及其有原始例子集時，我們比較 DTBMPA 方法的運算成本和把兩組例子集合併，然後利用 C4.5 演算法來產生一棵決策樹的運算成本，詳細說明如下所示：

1. 本研究所提之 DTBMPA 的運算成本

本研究所提出的合併兩棵決策樹程式，是以前序式走訪比對兩棵決策樹的節點，其搜尋方法都是從根向葉，沿著樹前進，判斷途中的每一個節點，予以處理。假設 k 爲節點數，n 爲葉節點的數目，p 爲 Ti 所擁有的選擇關係節點個數；在最差的情況下，需比對所有節點，則其運算成本爲 O(k)；再者，在比對每一個節點時，若 Ti 有連結節點，則會進行逐步比對其下的子節點的動作，是故其運算複雜度是 O(p)；而在尋找下一個比對節點時，當此節點爲最右側兄節點且無子節點時，會往其父節點走訪，是故當此節點爲路徑之最底層且最右側葉節

點時，最差情形下需往上走 log n (樹的高度)次，才能到達下一個比對節點，是故其運算成本 O(log n)。其次針對決策樹修剪方面，子樹取代的複雜度是 O(n)。綜合上述，DTBMPA 的總運算成本是 O(k log n)+O(kp)+ O(n)，其中 p 爲數值較小的常數，因爲在已合併 p 棵樹的 Ti 內，它的連結節點其下分支個數最大爲 p-1，是故相對於葉節點數 n 而言，它是較小的數值，所以公式３爲合併決策樹程式的運算成本。

O(k log n) (公式３)

2.合併兩組例子集來建立決策樹的運算成本

首先我們以 C4.5 演算法在建立一棵決策樹的運算成本爲例，假設訓練例子包含 n 個例子和 m 個屬性，而在最差的情況下決策樹有 n 個葉節點，則其高度是 log n。而在每一個樹的高度，必須處理全部 n 個例子，所以在樹的高度是 log n 時，一個屬性的運算成本就是 O(n log n)；在每一個節點處理所有 m 個屬性的話，其總運算成本就是 O(mn log n)。因此，在第一階段建立決策樹的運算成本是 O(mn log n)。其次，針對決策樹修剪方面，子樹取代的複雜度是 O(n)；子樹提昇(Subtree Lifting)的總運算成本是 $O(n(\log n)^2)$。因此在考量所有處理程序後，C4.5 演算法歸納出一棵決策樹的複雜度如公式４ (Witten & Frank, 2000)。

$$O(mn\log n)+O(n(\log n)^2) \quad \text{(公式４)}$$

假設我們有兩組例子集，S_1和 S_2。S_1有 n_1個例子，而 S_2有 n_2個例子，且它們同樣有 m 個屬性。依據公式４，若把兩組例子合併，以全部例子來產生一棵決策樹，則其運算成本如下：

$$O(m*(n_1+n_2)*\log(n_1+n_2))+O((n_1+n_2)(\log(n_1+n_2)^2) \quad \text{(公式５)}$$

綜合上述分析，用 DTBMPA 方法來產生一棵決策樹的運算成本，比起結合兩組例子集來重新學習一棵決策樹，較節省運算成本。

七、小結

經由上述的實驗，本研究歸納出下列結論：

(一) 在合併兩棵樹的規模下，合併樹的準確度比原始一棵樹的準確度有顯著提昇。就七十次的準確度比較之中，合併樹優於一棵樹之比率爲 90%，由此可見，藉由樹形知識合併，可以擴大決策樹的知識涵蓋面，提昇其預測未知例子的準確度。

(二) 在進行兩棵樹的實驗下，修剪樹與合併樹的準確度並無顯著差異，然就實驗數據(準確度)的比較，修剪樹大於合併樹的比率爲 27.1%，如果再加上等於的比率就高達爲 80%，可見修剪樹的準確度有小幅的提昇。

(三) 在分析決策樹的複雜度(節點數)方面，合併樹的節點數相當於兩棵原始樹的節點數相加。在最差的情形下，DTBMPA 方法輸出的合併樹節點數是的兩棵輸入樹的節點數加總再加一個連結節點。然而比較合併樹與修剪樹的節點數，得知修剪樹的平均節點數約爲

合併樹的平均節點數的百分之八十五，換句話說，修剪方法使得合併樹的節點數減少約 15%。

(四) 應用面的探討：本研究利用信用卡客戶信用資料來進行驗證 DTBMPA 方法，而整合後的決策樹形式知識可應用於審核信用卡申請者的資料，讓電腦系統事先篩選信用不良之申請者，然後再讓授信人員進一步審核申請者資料，如此可以輔助授信專家進行審卡作業。此外，決策樹的應用可分為下列幾項說明：

1. 處理大量資料問題：由於持卡人的消費記錄是與日俱增，面對大量的消費資料該如何放入決策樹演算法中進行學習，以瞭解信用良好與信用不良的客戶之消費樣態。因此發卡銀行可以先分散地學習出多棵決策樹，然後再利用 DTBMPA 方法來逐步地進行樹的合併修剪，以彙整出一棵完整的決策樹，如此可以解決大量資料的問題。
2. 達成知識累積的目的：發卡銀行不需每次都重新去產生一棵決策樹，而可利用 DTBMPA 方法來逐步地加入新的資料所產生的決策樹；利用新加入的決策樹來擴充原有決策樹的知識。此外，發卡銀行亦可將公司內部之授信法則或專家知識法則，轉換成樹形知識法則，然後利用 DTBMPA 方法來整合至合併決策樹內，以提昇決策樹內部知識的廣度，以達累積知識的目的。
3. 應用於決策樹形式知識之線上預測系統架構：此架構包含有知識學習、知識儲存、知識整合、知識溝通、和知識使用等五種功能。其中在知識整合方面，即可採用 DTBMPA 方法來進行整合多棵決策樹形式的知識，期能讓決策樹形式的知識在保有簡單樹狀結構下，進行知識法則之擴充(馬芳資，2003)。

陸、結論與建議

在知識經濟時代中，知識成為最重要的關鍵生產要素。企業家必須調整心態將知識累積視同資本累積般地重視，並且有效結合知識管理與人工智慧技術，才能提昇企業產能，創造契機 (曾憲雄等，2005)。而決策樹形式知識有著簡單易懂的樹形知識結構，且可以用圖形化的方式呈現給決策者，讓他能清楚瞭解決策樹的決策過程。因此本研究針對決策樹形式知識，提出一個方法來進行其知識整合，以達成企業累積知識的目的。

其次有鑑於 Quinlan(1998)先生所提出的合併樹方法，其合併後決策樹的節點數呈乘數(multiplicative)成長，再經由修剪程序後，合併樹的平均準確度低於一棵樹的平均準確度，因此本研究提出一個合併修剪方法，讓合併樹的節點數呈加數(additive)成長，同時提昇樹預測準確度，然後再透過修剪方法讓合併樹在保有原有預測能力下，進行精簡

的動作，期能產生一棵簡潔且有較佳預測能力的合併樹。

本研究提出的 DTBMPA 方法包括三個主要程序，即合併、修剪、和驗證。其運作流程首先把原始兩棵樹經由合併程序來結合成一棵合併樹，然後再透過修剪程序來產生修剪樹，最後由驗證程序來評估比較樹的準確度。其中合併程序是由上而下比對兩棵樹的節點，把相同父路徑的節點利用一個連結節點來結合，藉由此來擴大既有決策樹的分支法則知識；而修剪程序是由下而上走訪合併樹，把無法提昇準確度之分支節點還原成一個葉節點，其主要目的在於剪掉合併樹的過度分支。

本研究利用信用卡客戶信用資料來進行模式的驗證，經由實驗數據得知合併樹的準確度優於一棵樹的比率有 90%，而修剪樹的準確度大於或等於合併樹的比率有 80%。在統計檢定中，合併樹和修剪樹的準確度優於一棵樹的準確度達顯著差異，而修剪樹與合併樹的準確度無顯著差異。再者，在節點數的比較上，修剪方法使得合併樹的節點數減少了 15%。由此可見，本研究提出的 DTBMPA 方法在合併決策樹形式知識上有良好的表現。

此外，就管理意涵而言，企業可以運用本研究提出的 DTBMPA 方法來整合公司內部的法則知識，而法則知識的來源可由訪談專家而來，或將其日常營運所累積之交易記錄，透過決策樹學習演算法來歸納出決策樹，然後再進行知識整合。其應用面可用於財務、生產排程、銷售、和其他相關的決策問題的決策樹形式的知識累積。藉由決策樹形式知識的整合，讓企業擁有更周全的知識，以輔助決策者制定決策，提昇決策的品質。

在後續研究方面，本研究擬進一步探討當進行多次的合併修剪後的決策樹，如何利用重整方法來調整此樹的樹形，讓決策樹不會因為合併程序而變得過於龐大，且讓它在使用時運作會較有效率。再者，探討當外在環境變化時，DTBMPA 方法如何順應環境變化而進行合併樹內的知識調整。以及探討當有多個預測節點進行決策時，該如何設計有效判定最終預測值的策略。

參考文獻

馬芳資，林我聰，決策樹形式知識之線上預測系統架構，圖書館學與資訊科學，2003 年 10 月，Vol. 29，No.2，pp.60-76。

馬芳資，信用卡信用風險預警範例學習系統之研究，第十屆全國技職及職業教育研討會，技職研討會，商業類 I，1995 年，pp.427-436。

陳重銘，結合直線最適法於決策樹修剪之影響研究，國立中山大學資訊管理研究所碩士論文，1995 年。

曾憲雄，黃國禎，人工智慧與專家系統－理論／實務／應用，旗標出

版股份公司，台北市，2005 年，pp.1-25。

Bauer, E. & Kohavi, R., An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants, Journal of Machine Learning Vol. 36, Nos. 1/2, July/August 1999, pp.105-139.

Bolakova, I., Pruning Decision Trees to Reduce Tree Size, Proceedings of the international conference--Traditional And Innovations In Sustainable Development Of Society, Rezekne, Latvia, February 28 - March 2, 2002, pp.160-166, ISBN 9984-585-02-6.

Bradford, J. & Kunz, C. & Kohavi, R. & Brunk, C. & Brodley, C.E., Pruning decision trees with misclassification costs, In Proceedings of Tenth European Conference on Machine Learning(ECML-98), Berlin, 1998, pp. 131-136.

Breiman, L. Friedman, J.H. Olshen, R. & Stone, C., Classification and Regression Trees, Belmont, California: Wadsworth, 1984.

Breiman, L., Bagging predictors, Machine Learning, 1996, Volume 24, pp.123-140.

Chan, P.K. & Stolfo, S.J., Learning arbiter and combiner trees from partitioned data for scaling machine learning, In Proc. Intl. Conf. on Knowledge Discovery and Data Mining, 1995, pp.39-44.

Dong M. & Kothari, R., Classifiability Based Pruning of Decision Trees, Proc. International Joint Conference on Neural Networks(IJCNN), 2001, Volume 3, pp.1739-1743.

Dunham, M.H. Data Ming: Introductory and Advanced Topics, Pearson Education, Inc., 2003.

Esposito, F. & Malerba, D. & Semeraro, G., A Further Study of Pruning Methods in Decision Tree Induction, Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics, 1995, pp.211-218.

Fournier D. & Crémilleux B., A Quality Index for Decision Tree Pruning, Knowledge-Based Systems, 2002 ,Volume 15, Elsevier, pp.37-43.

Frank, E., Hall, M., Trigg, L., Holmes, G. and Witten, I.H., Data mining in bioinformatics using Weka, Bioinformatics Advance Access, published online Bioinformatics, 8 April. Oxford University Press, 2004.

Frank, E., Pruning Decision Trees and Lists, Department of Computer Science, University of Waikato, Hamilton, New Zealand. 2000.

Mingers, J., An empirical comparison of pruning methods for decision tree induction, Machine Learning, 1989,Volume 4, pp.227-443.

Murthy, S.K., On growing better decision trees from data, Doctoral dissertation, University of Maryland, 1997.

Niblett, T. & Bratko, I., Learning decision rules in noisy domains. In Bramer, M. A. (Ed.), Research and Development in Expert Systems III. Proceedings of Expert Systems '86, Brighton 1986, pp. 413-420. Madison, Wisconsin: Morgan Kaufmann, San Francisco, CA.

Quinlan, J. R., MiniBoosting Decision Trees, Journal of Artificial Intelligence Research, 1998.

Quinlan, J.R., C4.5: Programs for Machine Learning, San Mateo: Morgan Kaufmann, 1992.

Quinlan, J.R., Machine Learning: An Artificial Intelligence Approach, Volume 2, chapter The effect of noise on concept learning, Los Altos, CA: Morgan Kaufmann, 1986, pp.149-166.

Quinlan, J.R., Simplifying decision trees. International Journal of Man-Machine Studies, 1987, 27(3), pp.221-234.

Todorovski, L. & Dzeroski, S., Combining Multiple Models with Meta Decision Trees, In Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2000, pp.54-64.

Williams, G., Induction and Combining Multiple Decision Trees, Ph.D. Dissertation, Australian National University, Canberra, Australia, 1990.

Windeatt T. & Ardeshir G., An empirical comparison of pruning methods for ensemble classifiers, Proc. of Int. Conf Intelligent Data Analysis, Sept 13-15, Lisbon, Portugal, Lecture notes in computer science, Springer-Verlag, 2001, pp.208-217.

Witten, I. H. & Frank, E, Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations, Morgan Kaufmann, 2000.

致謝

感謝二位匿名審查委員對本文內容之指正與指導，讓文章更具可讀性、完整性，並提升論文的品質。

作者簡介

馬芳資爲國立政治大學資訊管理學系博士候選人，現爲龍華科技大學資訊管理學系講師，電子郵件信箱爲 ponymar@mail.lhu.edu.tw。

林我聰為美國加州大學柏克萊分校工業工程博士，現為國立政治大學資訊管理學系教授，電子郵件信箱為 lin@mis.nccu.edu.tw。

C.E.P.S.