**情報分析（第5回）　　　　　　　　　　　　　　　　　　　　　 2019年5月15日　辻慶太**

■判別分析とは

判別分析や決定木，サポートベクターマシン（SVM）は，事前にいくつかのグループが与えられた時，各個体がどのグループに属するかを自動判定してくれる手法です。前回のクラスタリングと異なる点は，クラスタリングは事前に明確なグループを設定する必要はなかったのに対し，判別分析，決定木，SVMは設定する必要がある点です。例えばNDCの0類から9類のいずれかに自動分類するというのは判別分析やSVMの仕事です。クラスタリングの仕事ではありません。

使用例：

・患者の病気診断（病気か否かの情報と，血圧，コレステロール，γGTP...の情報）：医者は，ある患者がガン患者なのかそうでないのかを，得られた検査結果から判定を迫られることがあります。

・投資先は優良企業か否か（優良かそうでないかと，財務諸表データなど）：株式の投資家は，いくつかのデータを基に，ある企業を優良企業かそうでない企業かに分類する必要があります。

・文字，音声，画像のパターン認識（「あ」か「お」かなど）：音声についてはSiriなどでおなじみでしょう。

企業は，過去の筆記試験や面接の成績から，将来役に立ってくれそうな志願者を選ばねばなりません。また，われわれ自身も，いくつかのデータをよりどころに，周囲の人が自分に対して好意を抱いているかどうかを，絶えず判別しながら生きているともいえます。世の中にはこのように，白黒付けがたいもののどうしてもその判断をしなければならないことがたくさんあります。こんなときに役立つのが判別分析やSVMと呼ばれる手法です。

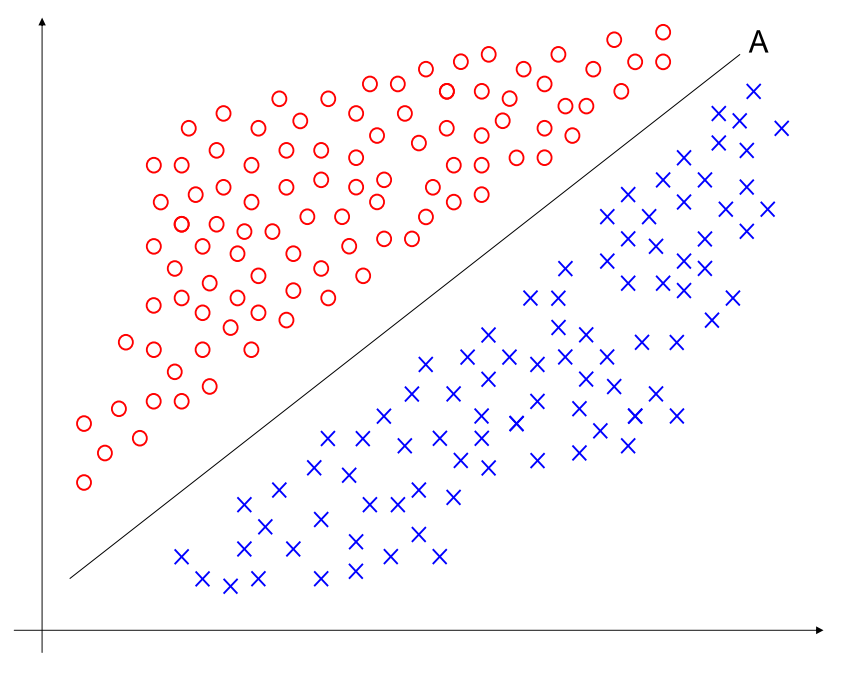
■2つのグループが：

・直線で分けられる → 線形判別関数による判別分析

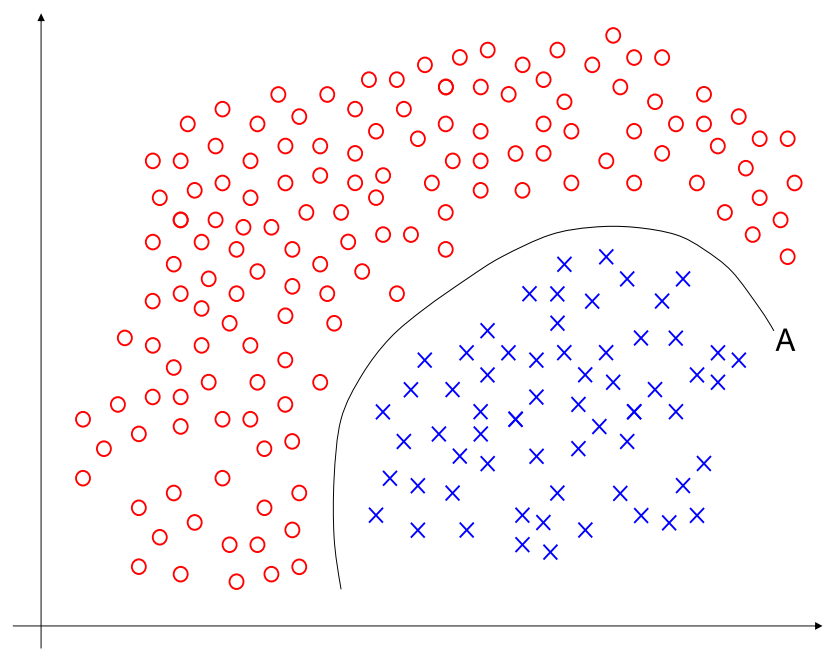
・凸な曲線で分けられる → マハラノビスの距離による判別分析

・凸な曲線では分けられない → 決定木，SVM

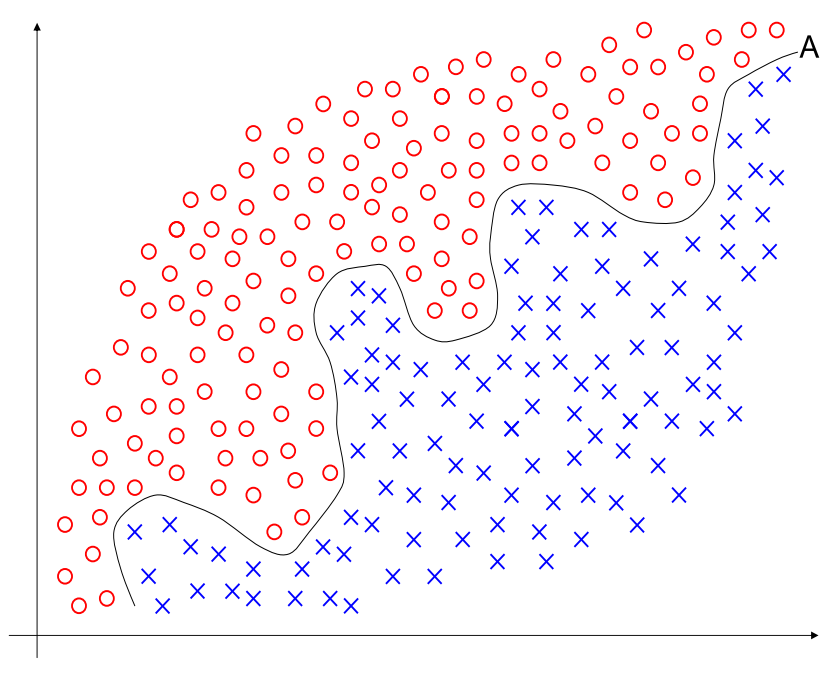
→ 上に示した通り，判別分析には線形判別関数を用いるものとマハラノビスの距離を用いるものの2種類があります。SVM も含めたそれらのイメージとしては次ページを参照して下さい。



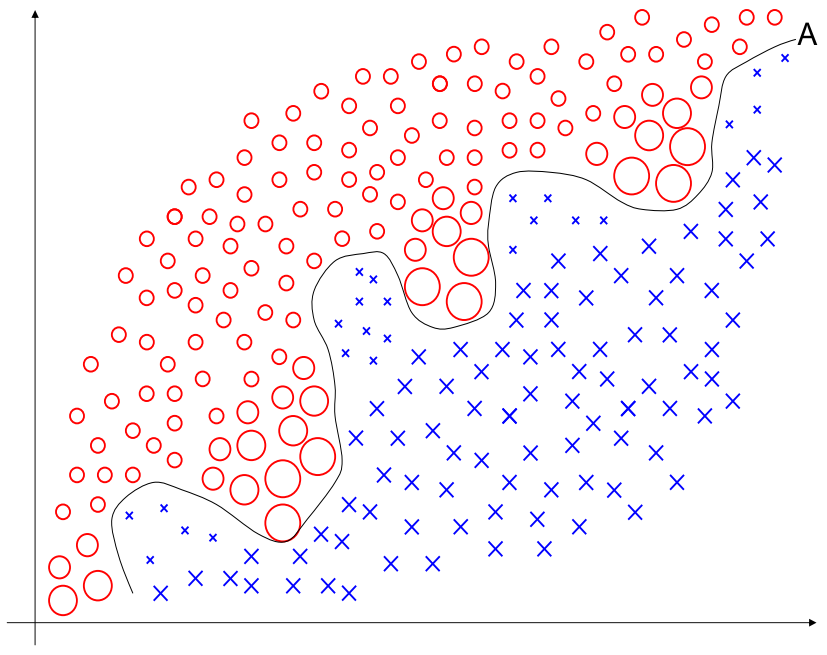
直線で分割できる場合は線形判別関数による判別分析



凸な曲線で分割できる場合はマハラノビスの距離による判別分析



凸でない曲線でしか分割できない場合は決定木やSVM。といってもSVMはこのような場合にしか使えないわけではなく，判別分析，決定木の図のような場合にも使えて，それぞれ高い精度の分割を実現します。その為，何かを判別したい時は，判別分析・決定木よりはSVMを使うのが一般的です。実はSVMより深層学習（ディープラーニング）の方が一般的なのですが，深層学習はまだ手軽に使えるものではないので割愛します。



SVMはデータの次元を拡張して（超）平面で分割します。イメージとしては，先ほどの図は2次元だったのがこの図は3次元に拡張されており，大きな○は手前に浮いていて，小さな×は遠くに沈んでいると考えて下さい。SVMはそのようにしてあいた空間に，下敷きを差し込んで分けるというイメージです。

線形判別関数による判別分析

1930年代にイギリスの優生学者，統計学者フィッシャーが提案しました。

■サー・ロナルド・エイルマー・フィッシャー（Wikipedia より）

フィッシャーはイギリスの統計学者，進化生物学者，遺伝学者で優生学者である。現代の推計統計学の確立者であるとともに，集団遺伝学の創始者の1人であり，またネオダーウィニズムを代表する遺伝学者・進化生物学者でもあった。フィッシャーは少年時代から数学の才能を発揮するとともに生物学にも興味を持った。1909年，ケンブリッジ大学に進み ... 卒業後まもなく第一次世界大戦が始まるが ... 終戦とともに新しい職探しを始め，ピアソンに招かれたものの，彼に反感を抱いてこれを断り，1919年，ハートフォードシャー州のロザムステッド農事試験場の統計研究員に就職した。ピアソンや息子のエゴン・ピアソンらとは，のちに統計学に関して大論争を起こすことになる。農事試験場では大量のデータに関する研究を行い，結果は『Studies in Crop Variation（穀物量の変動に関する研究）』という一連の報告となった。その後の数年間がフィッシャーの全盛期であり，実験計画法・分散分析・小標本の統計理論といった革新的な業績を生み出す。実際的なデータの研究から始まって新しい統計学理論へと進むのが彼の仕事の特徴であった。この仕事は1925年に最初の成書『Statistical Methods for Research Workers（研究者のための統計学的方法）』として実を結ぶ。これはその後の長きにわたり様々な分野の研究者のスタンダードとなった。1935年には『The Design of Experiments（実験計画法）』を出版しこれもスタンダードとなる。フィッシャーは分散分析や最尤法の手法を編み出し，統計学的十分性，フィッシャーの線形判別関数，フィッシャー情報行列などの概念を産んだ。

近年ではベイズ統計学が脚光を浴びていて，多くの分野で用いられています。迷惑メールを自動除去するナイーブベイズなどは有名です。が，このベイズ統計学をフィッシャーは嫌い「完全に葬り去らねばならない」と公言したりしました。統計学の権威であるフィッシャーの力は強く，近年になるまでベイズ統計学は不遇でした。ちなみにベイズ自身は一介の牧師で，彼の死後，大数学者のラプラスがベイズの定理を整理し発展させました。

線形判別関数による判別分析では，線形判別関数zを作り，zがある値より大きいならばグループ1，ある値より小さいならばグループ2，などと判定します。例えば z=12×蔵書冊数－350,218×貸出密度といった線形判別関数zを作り，その図書館の蔵書冊数と貸出密度を代入します。その結果，zが700より大きくなったらその図書館は県立図書館だろう，700より小さくなったら市立図書館だろう，などと判定します。

「そんなのもっと他の方法で調べた方が確実だろう」「電話してどっちか訊けばいいじゃないか」といったツッコミはもっともです。何となく分かりやすいかと思って図書館を例に出してみました。現実には例えば z=5×筆記試験の点数＋8×面接の点数 などとし，zが930以上だったら将来会社の役に立つ志望者として採用，zが930未満だったら将来役に立たない志望者として不採用，などとする方があり得そうです。この式は，筆記が100点でも面接が0点だったら多分役に立たないだろう，面接が100点でも筆記が0点だったらやはりだめだろう，またどちらかというと面接の点の方が重要だろう，といった経験則を式で表していると言えます。

ちなみに上では説明変量は「筆記試験の点数」と「面接の点数」という2つでした。この場合，合否の境界は直線になります。説明変量が3つの場合は合否の境界は平面になります。

ではzのイメージをつかむために，説明変量の数を2とし，平面上のデータを考えてみます。説明変量はx，yとします。またそれぞれの係数はa，bとします。即ち，zを以下のように表します：

　z=a x + b y

ここでサンプルが1, 2, ..,i, .., N個あるとすると：

　z1=a x1 + b y1

　z2=a x2 + b y2

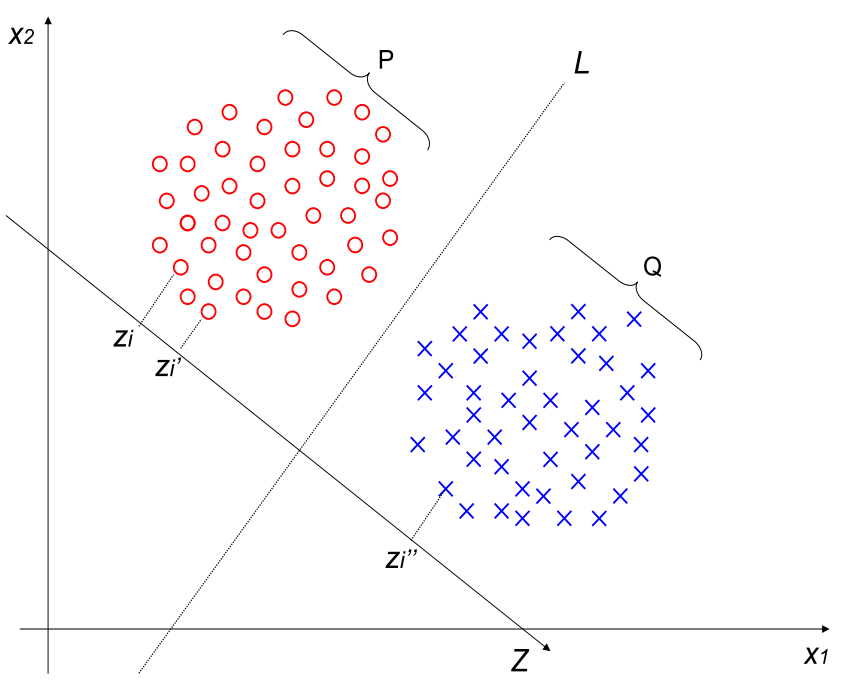
　：　：　：

　zi=a xi + b yi

　：　：　：

　zN=a xN + b yN

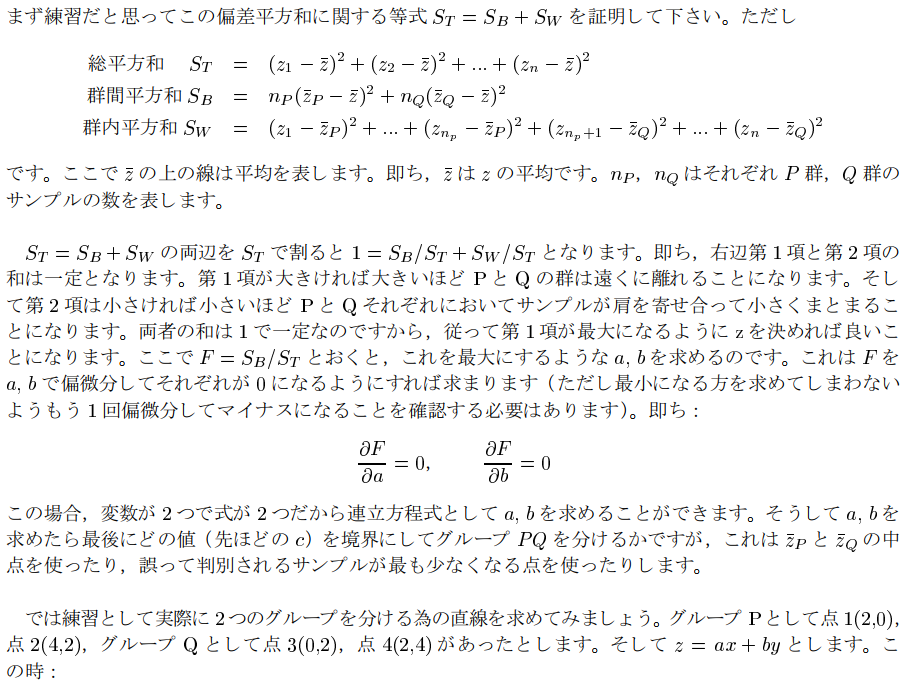
求める直線は「データを投影すると，2つのグループが一番よく分離されて見える直線（図のZ）に垂直で，かつ2つのグループの真ん中（後述）を通る直線」（=L）です。ziの値が大きかったらグループA，小さかったらグループB，などと判定します。

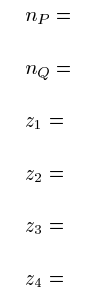


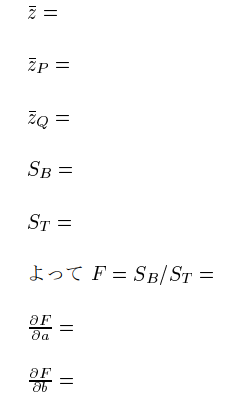
では具体的にどのように計算すれば求められるでしょうか？

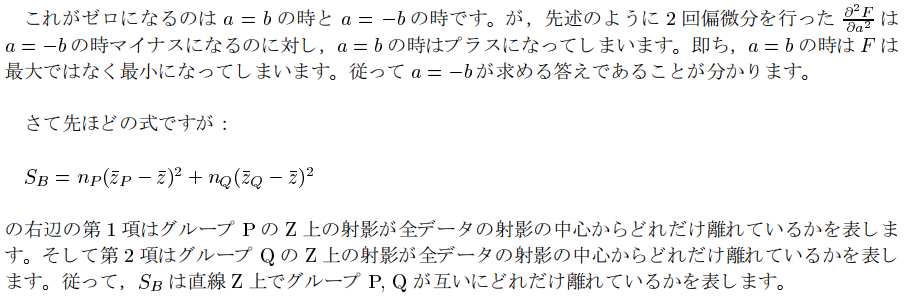
→ 一般に ST=SB+SW が成り立ちます。これを利用します。まず練習だと思ってこの偏差平方和に関する等式 ST=SB+SW を証明して下さい。

→ やっぱり面倒なのでやめましょう。先ほどのような投影図のイメージをお持ちになったなら十分です。気になる場合はおっしゃって下さい。

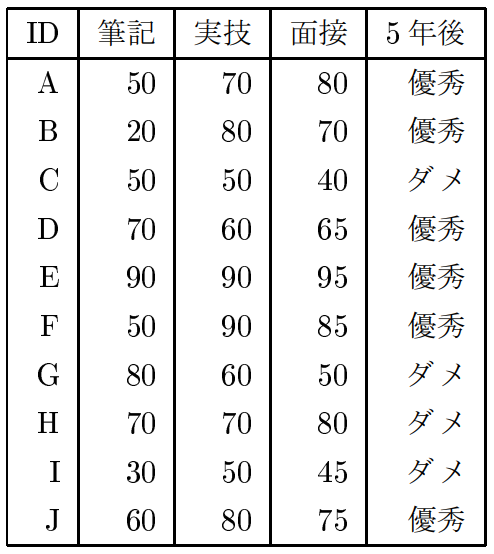




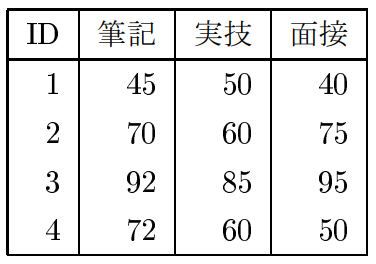




■ある会社の社員に関して，入社時の試験成績と5年後優秀な社員になったかどうかというデータが表Aのようにあります。今年は表Bのような成績の4名が入社を希望しました。誰を採用すれば優秀な社員を確保できるでしょうか？



表A. 入社時の成績と5年後



表B. 入社希望者4人

以下ではRで判別分析（線形判別関数による）をしてみます。

1) 学習用ファイルである「これまでの傾向」（Koremade\_no\_keiko.txt）とテスト用ファイルである「入社希望者」（Nyusha\_kibosha.txt）をダウンロードします（それぞれ右クリックから「対象をファイルに保存」を選択して落とした方が無難です）。ダウンロードしたら Word などで開いて中身を確認してみましょう。

2) Rを起動します。

3) まず「パッケージ」 → 「パッケージの読み込み」 → 「MASS」と選びます（「MASS」が出ない場合は，まず「パッケージのインストール」 → 「Japan (Tokyo)」 → 「MASS」と選んでから上記操作を行います）。これで判別分析を行う関数ldaが使えるようになります。

4) lda関数のヘルプが読みたい時は「?lda」と入力します。

5) 「ファイル」→「ディレクトリの変更」とクリックし，先ほどの2ファイルを置いたディレクトリに移動します。

6) > 現況 <- read.table("Koremade\_no\_keiko.txt", header=TRUE) と入力します。「現況」の部分は自分の好きな変数名で良いです。

7) > 希望者 <- read.table("Nyusha\_kibosha.txt", header=TRUE) と入力します。「希望者」の部分は自分の好きな変数名で良いです。

8) > 傾向 <- lda(5年後 ~ 筆記+実技+面接, data=現況) と入力します（「５」は全角です）。

9) > predict(傾向, 希望者)$class と入力して予想させます。結果，「[1] ダメ ダメ 優秀 ダメ」と出力されます。これは1, 2, 4番目の人は5年後「ダメ」，3番目の人は5年後「優秀」な社員になると予想されたことを意味します。

10) > 傾向 　と入力すると，線形判別関数の係数（=LD1）などが分かります。即ち，判別関数は 「-0.01992687\*筆記+0.04663433\*実技+0.04605209\*面接」 であることが分かります。

11) 学習用データをどの程度ちゃんと判別できるか，即ち，どの程度ちゃんと学習したかを知りたい時は以下のようにします：

> x <- subset(現況, select=c(1:3))

> predict(傾向, x)$class

上の1行目は「現況」変数の1列目から3列目までのデータを取りだして「x」という変数に代入することを意味します。この x を先ほどの「希望者」と同じ扱いにして predict 関数にかけます。結果「 [1] 優秀 優秀 ダメ ダメ 優秀 優秀 ダメ 優秀 ダメ 優秀」などと出力されますが，これを元のデータと比較すると2箇所で判別を誤っていることが分かります。

決定木とは

　決定木は判別分析やSVMと同様，事前にいくつかのグループ（カテゴリー）と，所属グループ不明の個体が与えられた時，各個体がどのグループに属するかを自動判定してくれる手法です。判別分析やSVMと異なる点は，決定木はいくつかの判定ルールと共に結果を表示してくれるので，なぜその個体がそのグループに分類されたのか結果を追いやすい点です。例えば「その個体は男性ですか？女性ですか？」「年齢は42歳以上ですか？」といったルールが有効だと自動的に判断し（例えば「年齢は41歳以上ですか？」より「年齢は42歳以上ですか？」の方がルールとして優れているとデータを見て自動判断してくれます），表示してくれ，かつ判定したい個体をグループ分けしてくれるのです。結果を人間が見て理解しやすい点がSVMや判別分析との大きな違いです。

■決定木の例

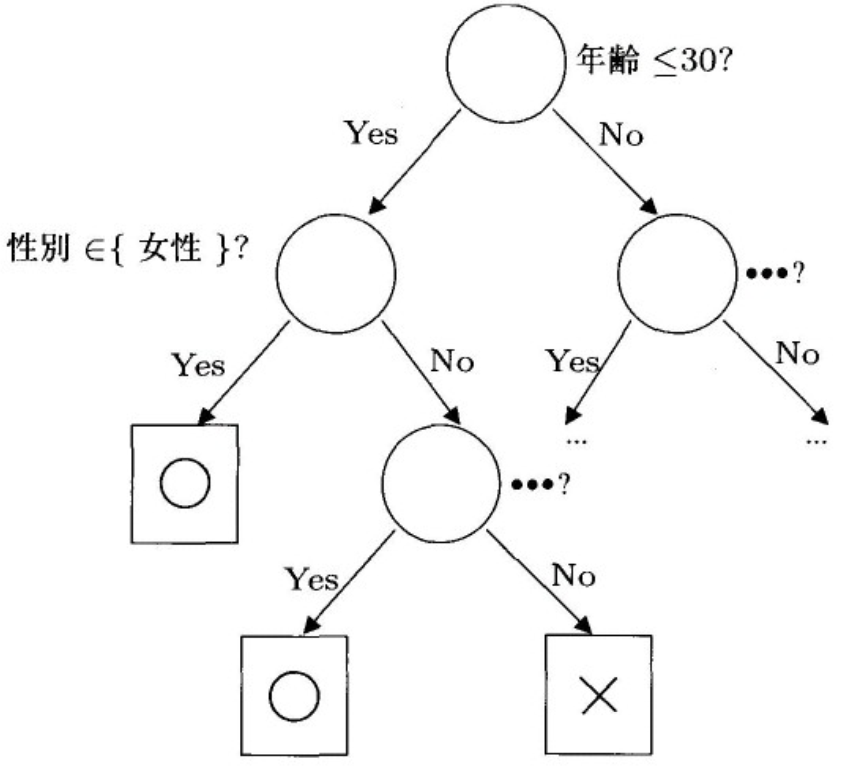
　例えば表Pのような顧客プロファイルと，その顧客の戦略商品Aの購買の有無（○が購入顧客，×が非購入顧客）を示すデータベースがあったとします。このデータベースから作成された「商品Aの購買の有無」に関する決定木の例が図Q1です。図Q1の決定木では，木の各中間ノードにデータを分類するための分割テスト（splitting test）が示されています。また，各中間ノードは分割テストを適用した結果のとり得る（2つの）値に対応する（2つの）下向きの枝をもちます。一方，各終端ノードは，ここで注目している「商品Aの購買の有無」を示すラベルをもちます。

　この例のデータベース中の各顧客は，この決定木を使って，いずれかの終端ノードに分類されます。決定木の（中間ノードの一つである）頂点ノードはデータベースの全データ集合に対応します。データベース中のある顧客は，この頂点ノードの1レコードです。このレコードは，頂点ノードの分割テストである「年齢が30以下」であるかどうかで，二つのうちのいずれかの下位ノードに分類されます。たとえば，年齢が15である顧客は，テストの適用結果であるYesに対応する左の下位ノードに分類されます。下位ノードにおいても，同様に，ノードの分割テストを適用した結果に従って，いずれかの下位ノードへ分類されます。こうした分類を繰り返すことで，各顧客は最終的にいずれかの終端ノードに分類されます。

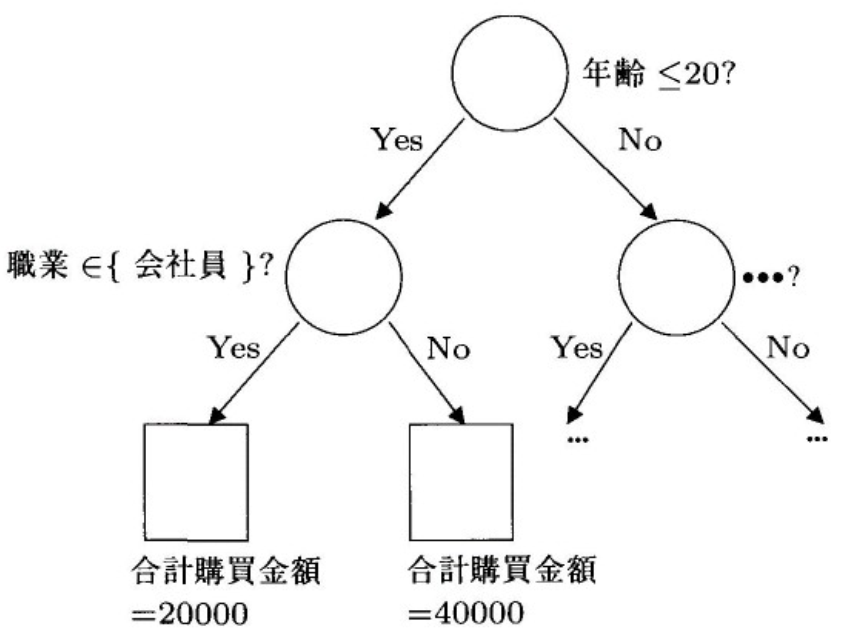
　例えばこの決定木の一番左の終端ノードに至る経路は，「年齢が30以下でかつ性別が女性という条件を満たす顧客は商品Aを購買した」という法則を表しています。従って今後たとえば「年齢が30以下」で，かつ「性別が女性」の顧客が店に入ってきたら，この顧客は商品Aを買う可能性が高いと予測して店員は接客するといった対応が可能になります。



表P. 顧客プロファイル



図Q1. 「商品Aの購買の有無」に関する決定木（分類木）の例



図Q2. 「合計購買金額」に関する決定木（回帰木）の例

　各終端ノードは，そのノードに分類される顧客の「商品Aの購買の有無」から，ラベルづけされます。一般的には，「○」の頻度が「×」の頻度より多い終端ノードのラベルは「○」，その道の場合は「×」とすることが多いです。その場合，「○」とラベルづけされた終端ノードは「商品Aを購買した」顧客グループ，「×」とラベルづけされた終端ノードは「商品Aを購買しなかった」顧客グループに対応しています。

　決定木は2種類に分けることが出来ます。「商品Aの購買の有無」のように，注目する属性値がカテゴリ型である場合は「分類木」と呼ばれることがあります。一方，注目する属性が「合計購買金額」のような数値型である場合は「回帰木」と呼ばれることがあります。図Q2に「合計購買金額」に関する回帰木の例を示します。回帰木の各終端ノードのラベルは，そのノードに分類される顧客グループの，注目している数値属性値（この場合「合計購買金額」）の平均値とする場合が多いです。「合計購買金額」が未知の顧客が，回帰木でいずれかの顧客グループに分類された場合，その顧客の「合計購買金額」の予測値を，その顧客グループの平均値とする場合，終端ノードのラベルを平均値としておくと便利なためです。

・「商品Aの購買の有無」のように，注目する属性がカテゴリ型の木 → 分類木

・「合計購買金額」のように，注目する属性が数値型の木 → 回帰木

　※2つ合わせて決定木。

■決定木の特徴

・SVMなどと異なり，人間が見て理解しやすいルールになっています。

・データが欠損値を含んでいる場合でも扱いやすい。

・非線形のデータに対して良い結果を与えることが多いです。一方，線形なデータの場合はあまりよい結果を与えないことが知られています。即ち，それなら判別分析や回帰分析の方がまだ良いです。

・ニューラルネットなどに比べ，高速に計算できます。

■用語

・木：図Qのような分岐構造（→ 木というと根が下のはずですが，なぜか逆さまにして描きます）

・頂点（ルート）ノード：木の一番上の部分

・終端（リーフ）ノード：　〃　下

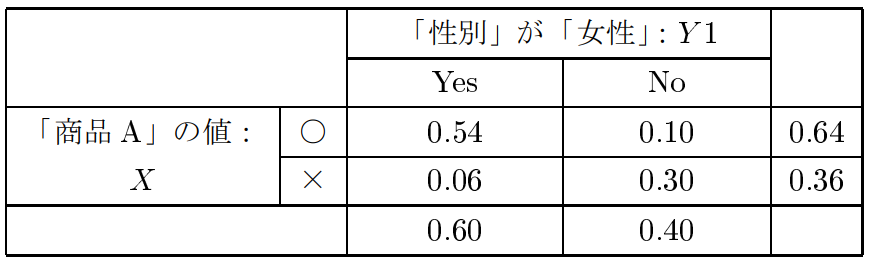
・決定木は，頂点ノードから終端ノードに至るまでのIF-THENルールの組み合わせと考えることができます。このとき終端ノードのラベルは，このIF-THENルールの結論部に相当します。

・目的属性（target attribute）：「商品Aの購買の有無」，「合計購買金額」のように，IF-THEN ルールの結論部に現れる属性

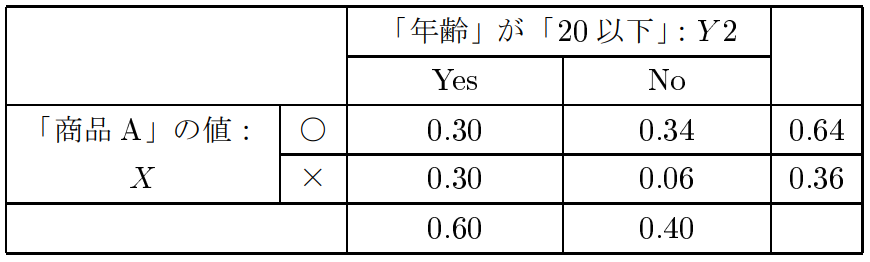
・条件属性（conditional attribute）：IF-THEN ルールの条件部に現れる属性

■決定木の求め方

　以下では相互情報量を用いた決定木構築方法について説明します。例えば「商品A」の値が○か×かを導く為の情報として表R，表Sのようなデータがあったとします。



表R. ルール1の情報



表S. ルール2の情報

　表Rから例えば，性別が女性で商品Aを買った（○）人は全体の54％であること，性別が女性で商品Aを買わなかった（×）人は全体の6％であること，であることが分かります。同様に表Sから例えば年齢が20以下で商品Aを買った（○）人は全体の30％であることなどが分かります。これらの表からは例えば以下のルールが導けます：

　　ルール1: Y1

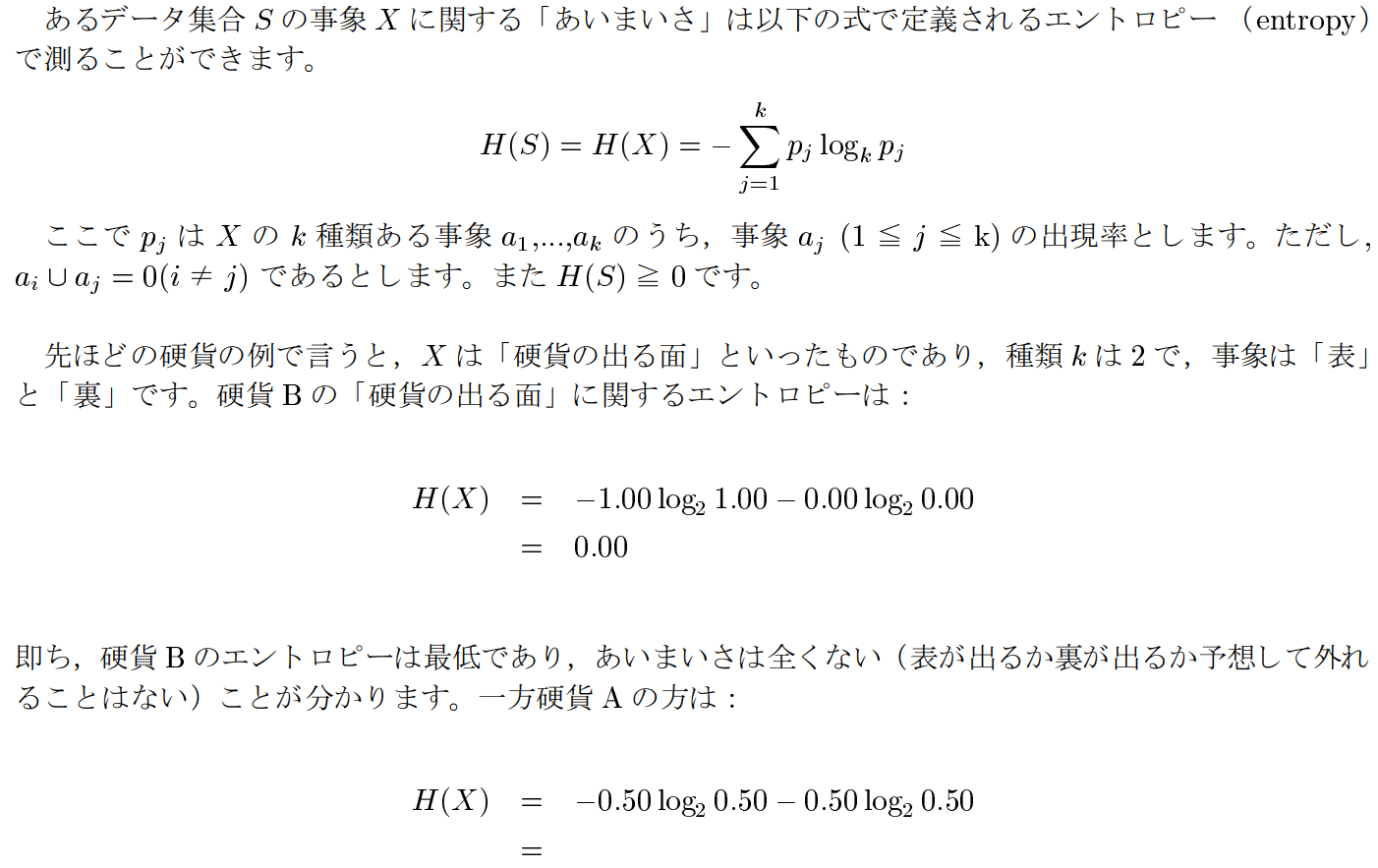
　　　（「性別」が「女性」）⇒（「商品A」が「○」）

　　ルール2: Y2

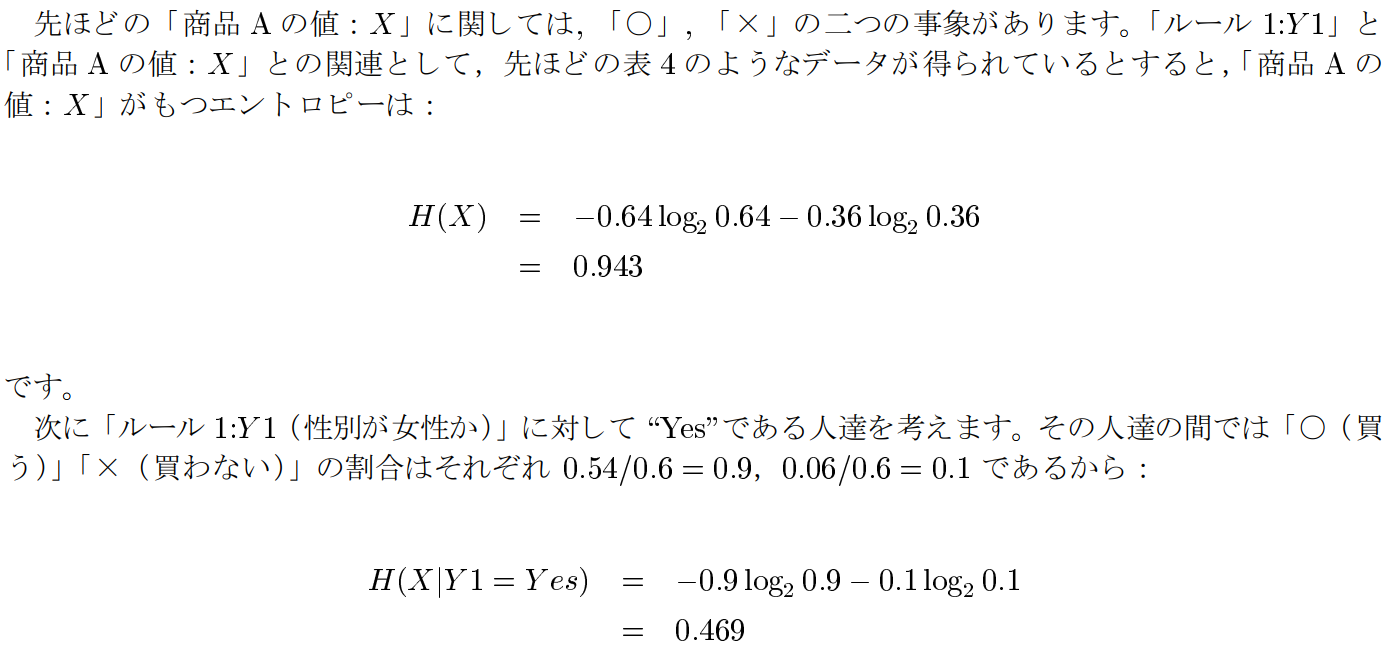
　　　（「年齢」が「20以下」）⇒（「商品A」が「×」）

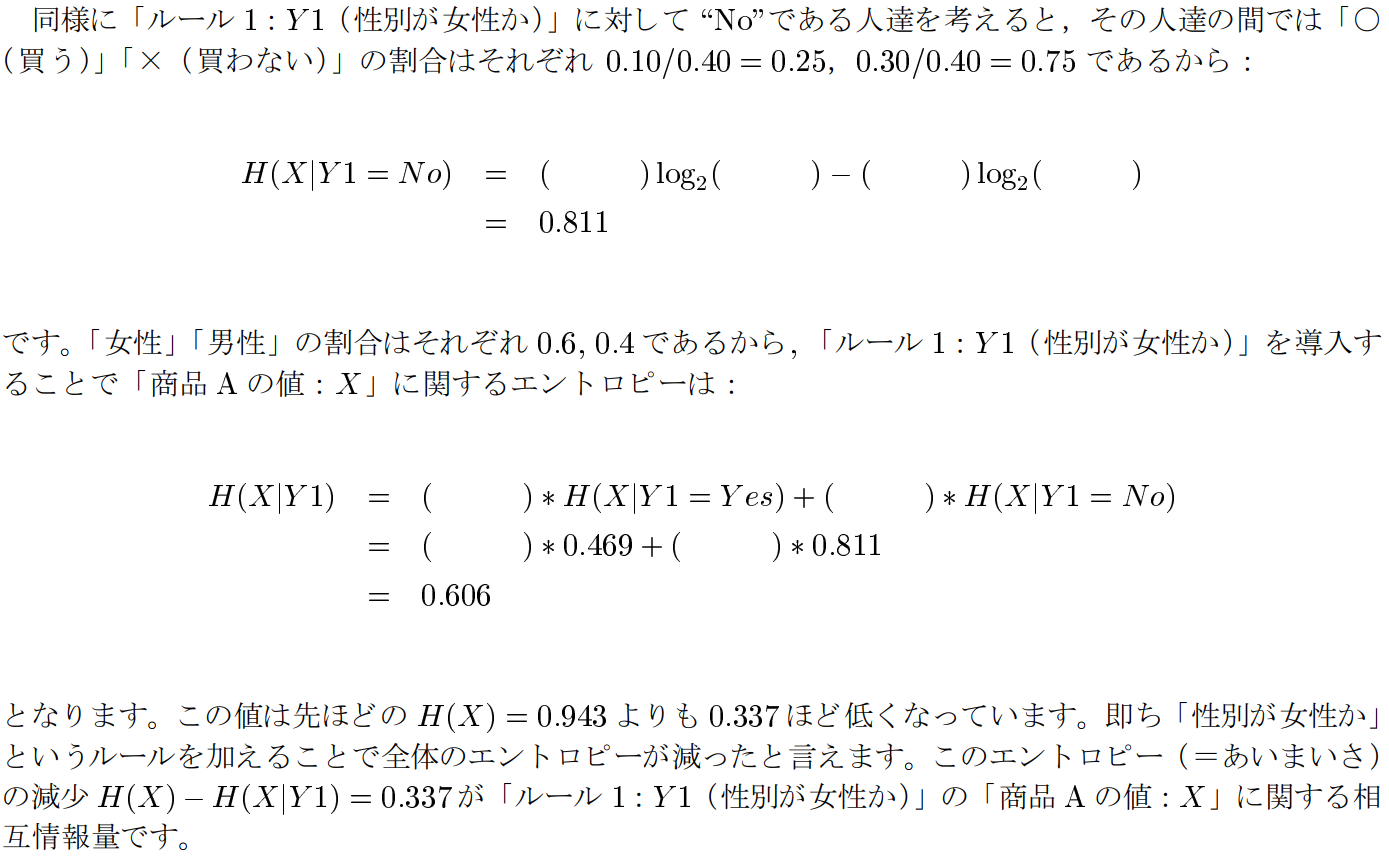
この時，ルール1とルール2だったらどちらの方が優れているでしょう？ ルールの優秀さは「あいまいさの少なさ」で評価することができます（あるいは「間違って分類してしまう確率の低さ」と言った方が分かりやすいかもしれません。が，以下では伝統に従って「あいまいさの少なさ」という表現を用いることにします）。ではこの「あいまいさ」はどのように表せるでしょうか？

　普通の硬貨Aを投げると表が出る確率は50％，裏が出る確率は50％です。ここで硬貨を真っ二つに切って貼り合わせ，両方とも表になったマンガに出て来るようないかさま硬貨Bを考えます。この硬貨は表が出る確率は100％，裏が出る確率は0％です。この時，硬貨Aを投げたとき表裏どちらが出るかに関する「あいまいさ」（予想して間違えてしまう確率）は，硬貨Bを投げたとき表裏どちらが出るかに関する「あいまいさ」（予想して間違えてしまう確率）より高いです。従って「あいまいさ」の尺度を作るとしたら上記のような大小関係になる尺度になるべきです。これにはエントロピーという以下のような尺度があります。



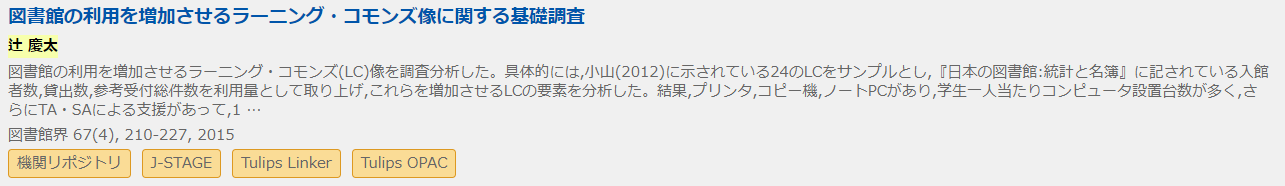
となります。即ち，硬貨Bのエントロピーより高くなることが分かります。このようにエントロピーは先ほど，あいまいさの尺度に要請した性質を満たしています。

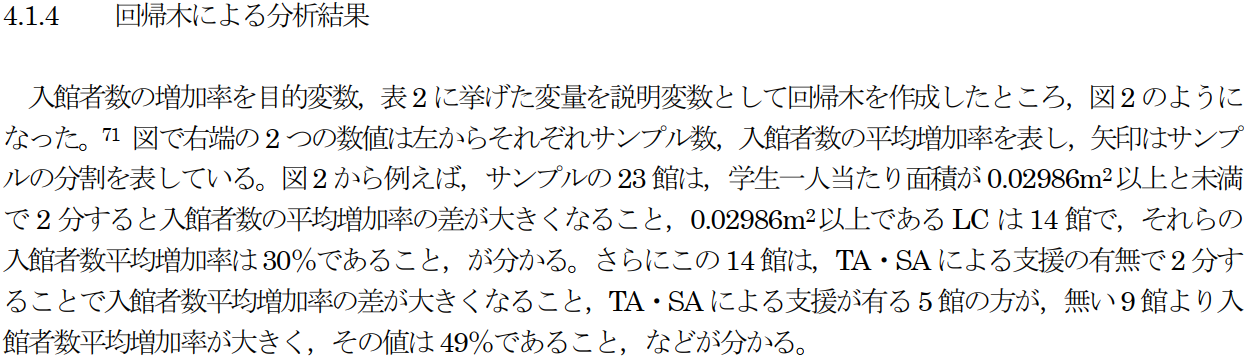


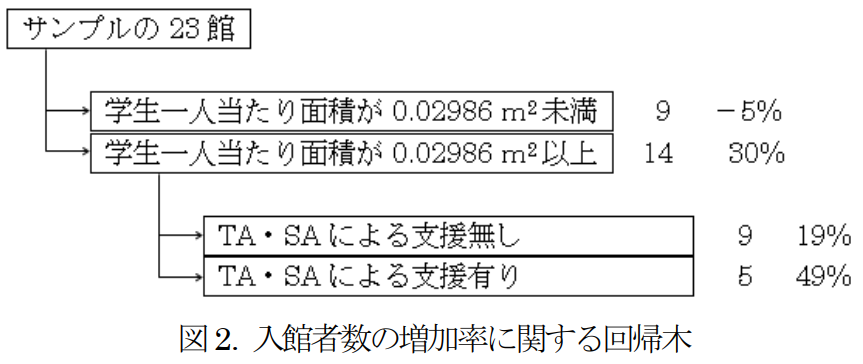


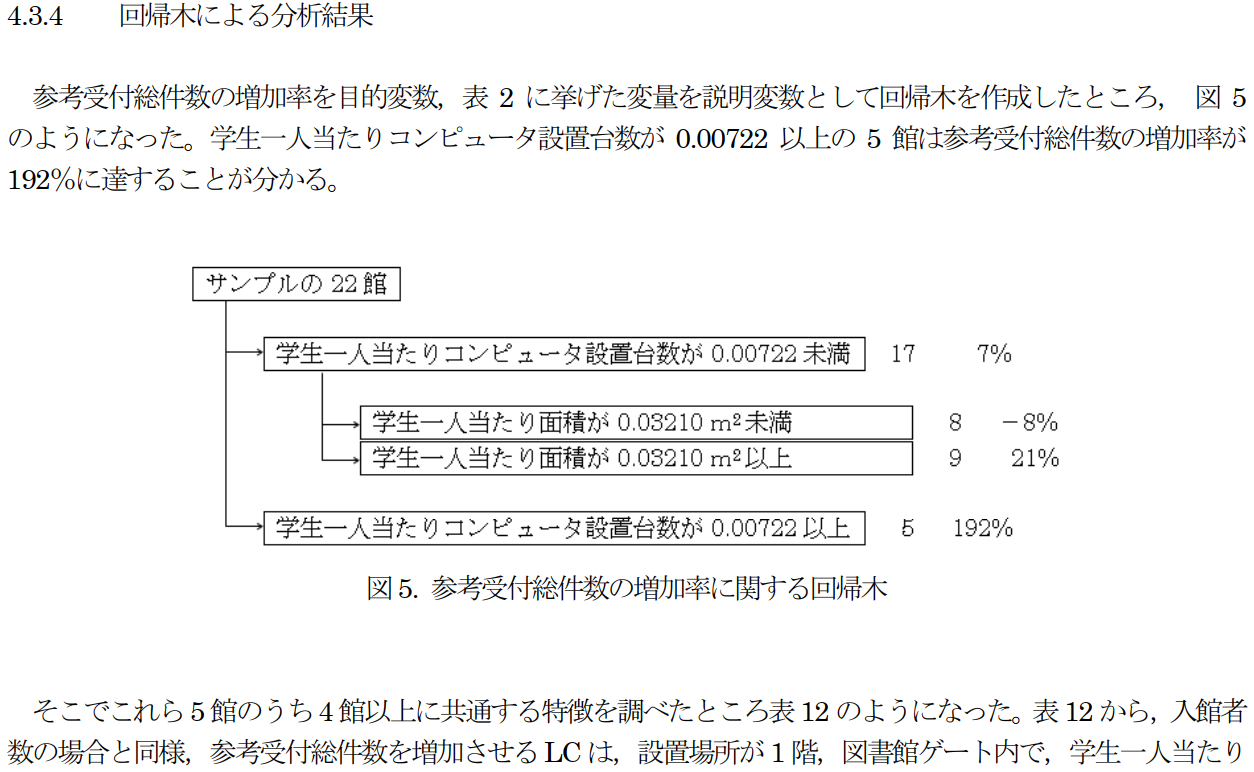
■R による決定木構築

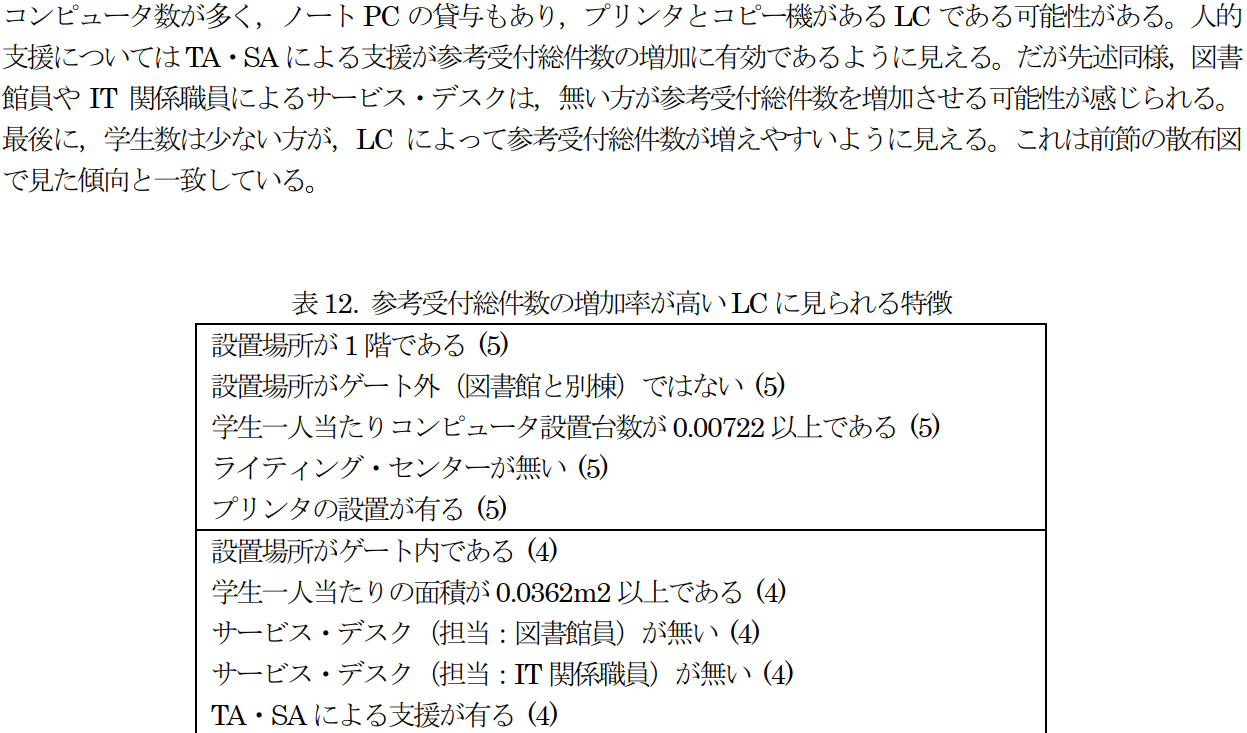
　以下の論文を書いた時，決定木（分類木ではなく回帰木の方）を作ったことがあります。以下にその論文の該当箇所を引用します。











これらの回帰木を再現してみましょう。

　まずパッケージ tree をインストールし読み込みます。「パッケージ」→「パッケージのインストール」→ ミラーサイト一覧から「Japan (Tokyo)」などを選びます → パッケージ一覧から「tree」を選びます。“tree”パッケージをインストールしたら，次にそれを読み込みます。「パッケージ」→「パッケージの読み込み」→ パッケージ一覧から「tree」を選びます。これで “tree”パッケージが使えるようになります。

　次にデータとして来館者数の増加率に関する“LC\_gate\_counts.csv”を使います。決定木は以下のように作成します：

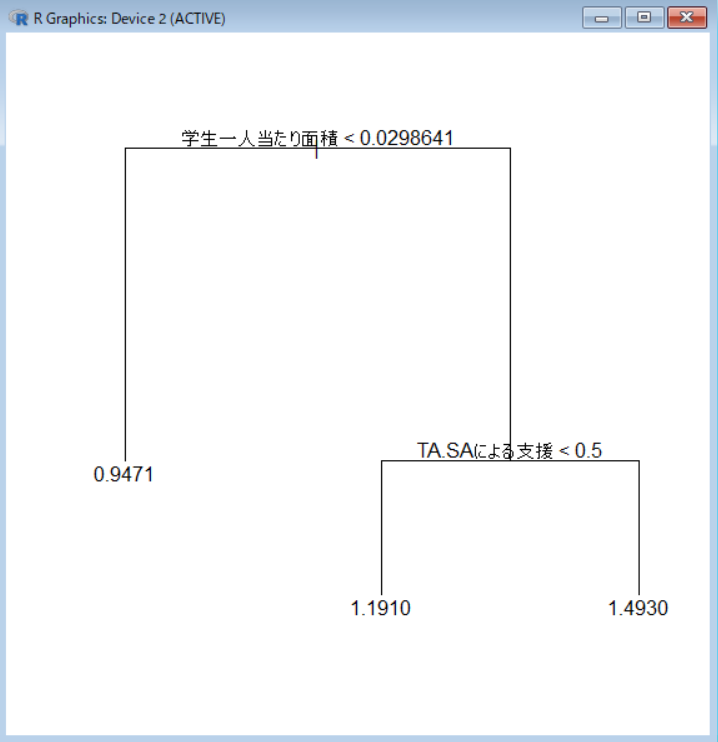
> x <- read.csv("LC\_gate\_counts.csv", row.names=1)

> kekka <- tree(増加率 ~ ., x)

> plot(kekka)

> text(kekka)

これにより下図のような出力が得られます。



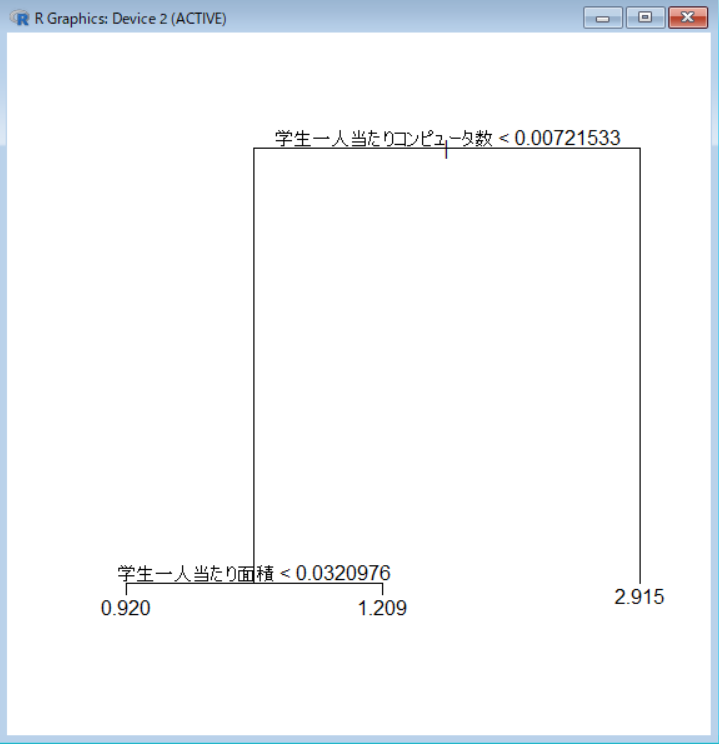
　参考受付件数については，“LC\_references.csv”を使います。決定木は以下のようにして作成します：

> x <- read.csv("LC\_references.csv", row.names=1)

> kekka <- tree(増加率 ~ ., x)

> plot(kekka)

> text(kekka)



■アソシエーションルールとは

　アソシエーションルールはAmazonの「この商品を買った人はこちらの商品も買っています」といった推薦を実現する手法です。「そんなの簡単だろう」と思うかもしれませんが，売上げにおける商品の組合せは膨大な数にのぼるので，効率的な計算方法が求められます。アソシエーションルールのアプリオリ・アルゴリズムはそのような計算（重要な組合せの特定）を効率的に行います。

■アソシエーションルールを用いた研究例：

・柏原由紀ら (2015)「動詞に着目した相関ルールを利用するメソッド名の命名支援手法の評価」研究報告ソフトウェア工学.

・伊藤寛明ら (2014)「アソシエーションルールを用いた推薦システムにおける精度と意外性の向上」研究報告数理モデル化と問題解決.

・折磊ら (2013)「アプリオリアルゴリズムとクラスタリングを用いたクロスセリング売上最大化のためのアソシエーションルールマイニング」

・遠山清人ら (2010)「アソシエーションルールを用いた電子メールのマルチラベル分類」情報処理学会全国大会講演論文集 第72回.

・杉村博 (2009)「テンプレートにもとづく時系列データからの相関ルールマイニングシステム」情報処理学会全国大会講演論文集 第71回.

・裴明花ら (2006)「重要な顧客層および相関ルール発見のための繰返し購買パターンを考慮した相関ルールマイニング」情報処理学会論文誌.

・上田芳弘ら (2002)「相関ルールを用いた組織内における文書校正支援」電子情報通信学会論文誌D.

・山本幸枝ら (2001)「相関ルールを用いた英文情報の分野判別」電子情報通信学会技術研究報告.

■アソシエーションルールの概要

例えばスーパーのレジでは様々な商品購入のデータが収集できます。一人の客の買い物かごは，どのような商品が同時に買われたかを示すデータと考えることができます。そこから，例えば｢目玉商品Aと日用品Bを購入した顧客は,同時に高級品Cと日用品Dも高い確度で購入する｣という事実がわかれば：

　　・ A，B，C，Dのセット商品を発売する。

　　・ 顧客の利便性を考えて,商品の配置を近づける。

　　・ 顧客に店内を長く歩き回ってもらうため，商品の配置を遠ざける。

　　・ AやBの特売を行う際にはCやDの在庫を増やしておく。

といった販売戦略を立てることができます。

この購買傾向は：

　{ 目玉商品A，日用品B } → { 高級品C，日用品D }

という式で表現できます。

一般にX，Yを商品の集合として（この例ではX={ 目玉商品A，日用品B }，Y={ 高級品C，日用品D }

　　　X → Y

と記述される傾向をアソシエーションルール（association rule)と呼びます。

※アソシエーションルールは相関ルールと呼ばれることもあります。ですが｢相関｣は一般に“correlation”に対する訳語として用いられているので，あまり良い呼び方ではありません。

■用語(1)：トランザクション，アイテム

買い物かごのような１つのまとまりをトランザクションと呼びます。またトランザクションを構成するもの（商品など）をアイテムと呼びます。下の場合，アイテムは {ミルク，パン，バター，ビール} です。



　　　　　表：トランザクションとアイテム

上の表から得られるアソシエーションルールとしては，{ミルク} → {パン}，{ミルク，パン} → {バター} などがあり得ます。先ほど説明した通りですが，一応言葉で書くと，{ミルク，パン} → {バター}は「ミルクとパンを同時に買う人は，バターも買う」という意味です。

さてここでアソシエーションルールの説明を正確に行えるよう以下の用語を導入します。

I = { i1，i2,...,im } をアイテムの集合とします。

T = { t1，t2,...,tn } をトランザクションの集合とします。

トランザクション ti はアイテムの（部分）集合と言えます（すなわち T ⊆ I ）。

トランザクション ti とアイテム集合 X に関して ti ⊇ X が成り立つとき, 「ti は X を含む」といいます。{パン，ミルク，ビール} というトランザクションはアイテム集合 {パン，ミルク，ビール} ，{パン，ミルク}，{ミルク，ビール}，{パン，ビール}，{パン}，{ミルク}，{ビール} を含むと言えます。

■用語(2)：サポート，確信度

X，Y をアイテムの集合とします（ただしX∩Y =φ）。この時，データベースD中のX∪Yを含むトランザクションの全トランザクションに対する割合がs％であるとき，「アソシエーションルールX → YはDにおいてs％のサポート（support：支持度とも言う）を持つ」と言います。以下ではこれをsupport（X → Y）= s％ と表記します。

例えば先ほどの表では，アイテム集合 { ミルク，パン } は5つのトランザクション中2つに現れているので 2/5=40％ のサポートを持ちます。

ではアイテム集合 { パン，バター } のサポートはいくらでしょう？

さて再びX，Yをアイテムの集合とします（ただしX∩Y =φ）。この時，データベースD中の X を含むトランザクションのうち，Y を含むものの割合がc％であるとき，「アソシエーションルールX → YはDにおいてc％の確信度（confidence）を持つ」と言います。以下ではこれをconf( X → Y )=c％ と表記します。

例えばアイテム集合 { ミルク，パン } は2つのトランザクションに現れます。このうち { バター } も含んだトランザクションは1つなので，conf ( { ミルク，パン } → { バター } )=1/2=50％ の確信度を持ちます。

※統計を習った方なら，確信度とは条件付き確率 P(Y|X) であると理解することができるでしょう。

ではアソシエーションルール{ パン } → {バター}の確信度はいくらでしょう？

「価値のある」アソシエーションルールは基本的に以下の2つを満たすルールです：

　　・確信度confが高い。

　　・サポートsupportもある程度高い。

（なぜなら，サポートが小さいアソシエーションルールは，わずかなデータにしか当てはまらないという意味で，出番の少ないルールだからです。例えば「ミルクとパンとバターとアイスとビールと猫缶とドッグフードを買った人は自転車も買っていた。このルールの確信度は100％だ」といっても，それは過去にそういう特殊な人が一人いた結果によるものならば，あまり嬉しい情報ではありません）。

■アソシエーションルールの数え上げは大変

さてm種類のアイテムを使って作ることのできるアソシエーションルールは全部で：



個あります。m 個のアイテムからアソシエーションルールに用いる k 個のアイテムを選ぶ組み合わせはmCk通りあります。そしてそれぞれの組合わせにおいて，アソシエーションルールの前提部（“→”の左側）と結論部（“→”の右側）の分け方が 2k－2 通りあるからです。

従ってアイテム数m=10の場合，アソシエーションルールは約57,000個，m=100の場合は5.15×1047個あります。それらのサポートや確信度を全て求めるのは現実的に不可能です。

　先ほど価値のあるアソシエーションルールとは，確信度が高く，サポートもある程度高いルールだと述べました。従ってすべてのアソシエーションルールが求まらなくても，価値のあるルールだけ求まれば良いと開き直りましょう（価値の少ないルールは求まらなくて良いと考えましょう）。その為のアルゴリズムを以下で紹介していきます。

■アプリオリアルゴリズム

Agrawalらは1993年，指定した確信度とサポート値（以上）をとるアソシエーションルールをすべて発見する「アプリオリアルゴリズム」を提案しました。それに基づいてIBMアルマデン研究所で開発された Quest システムは世界初の本格的なデータマイニングシステムです。

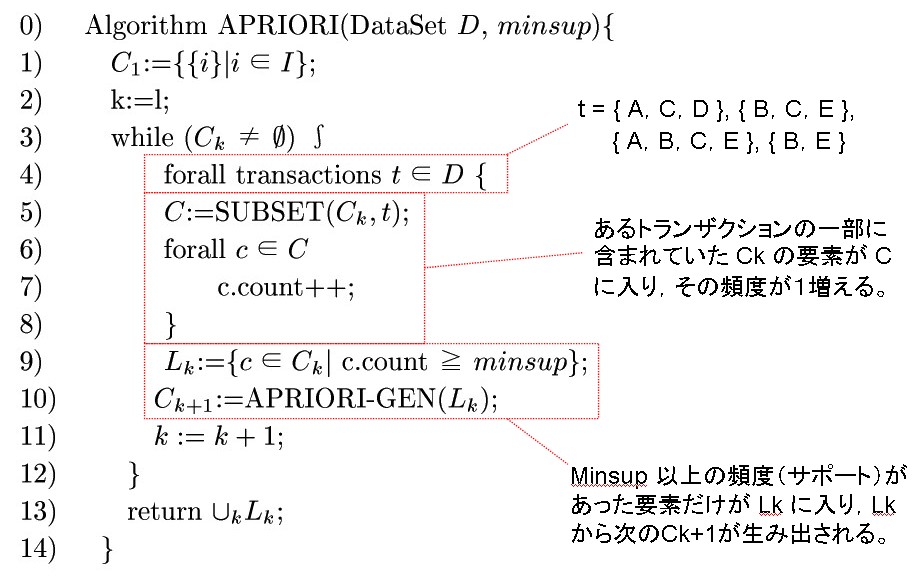
※アプリオリという名前は「先験的」な知識を使うことから来ているそうです。

基本となる考え方：

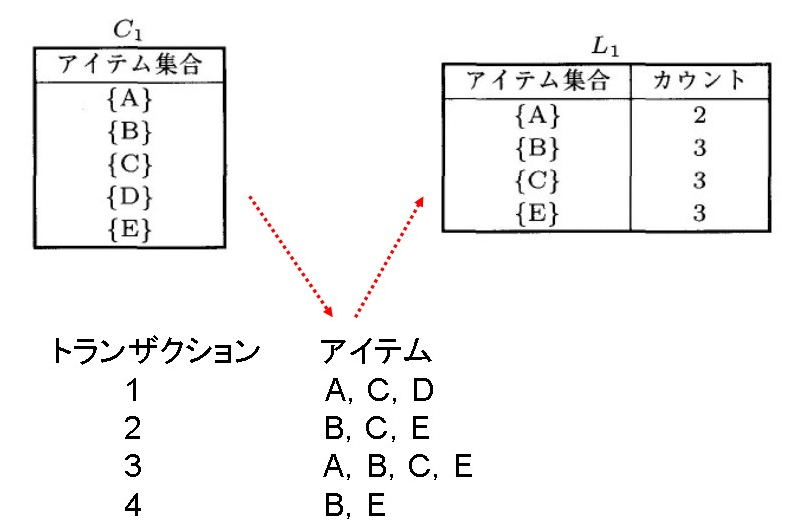
あるアイテム集合iの頻度がminsup以下だったら，そのアイテム集合iを含むアイテム集合の頻度は必ずminsup以下になる。

例えば，アイテム集合 { パン，バター } がデータベース中に5回しか現れていなかったら，これを含むあらゆるアイテム集合，例えば { パン，バター，ビール }，{ ヨーグルト，パン，バター，塩 } などは5回以下しかデータベース中に現れていないはずです。

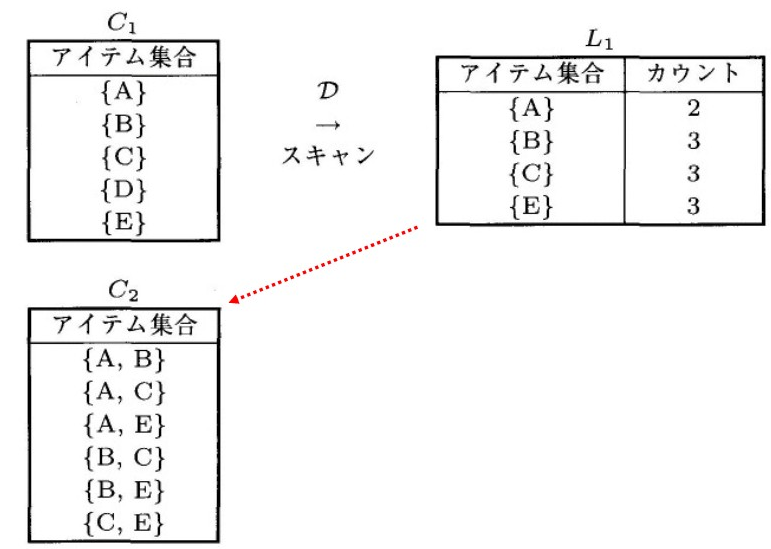
→ 従って「そういう組合せは考えない。余計な数え上げはしない」というのがアプリオリアルゴリズムの考え方になります。



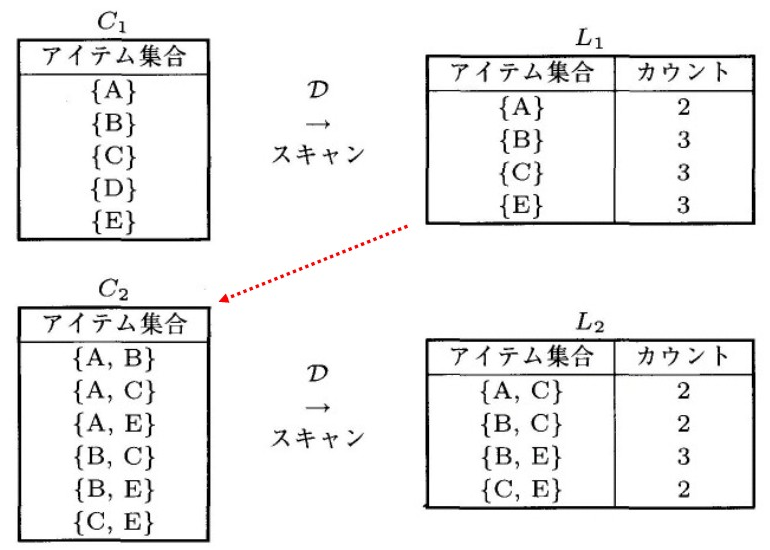
例えば：

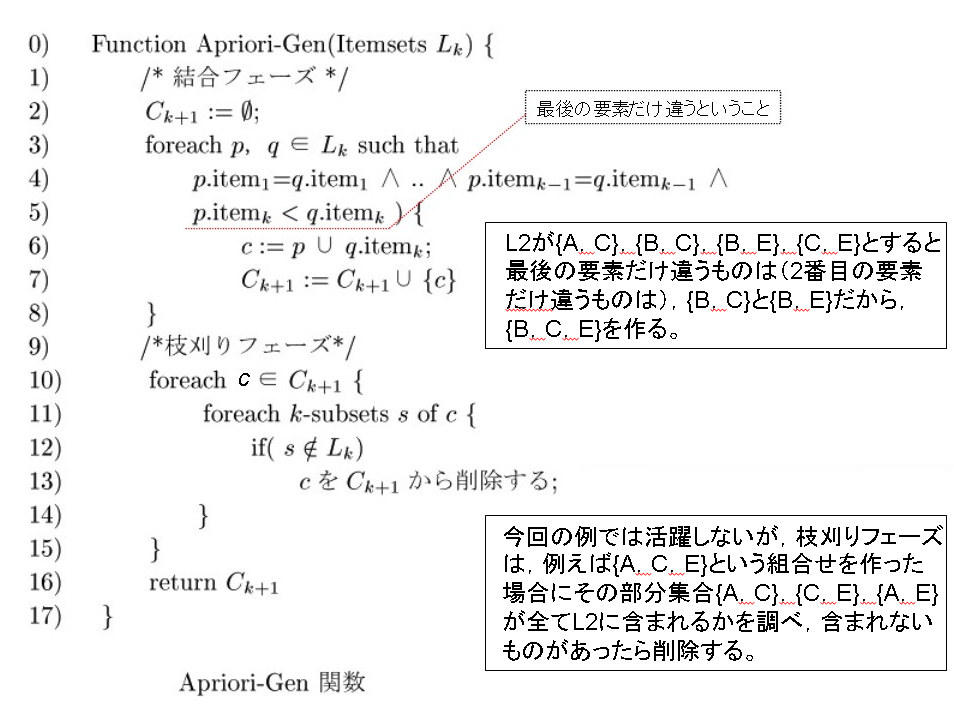


次にL1の要素から以下のようにC2を作ります：

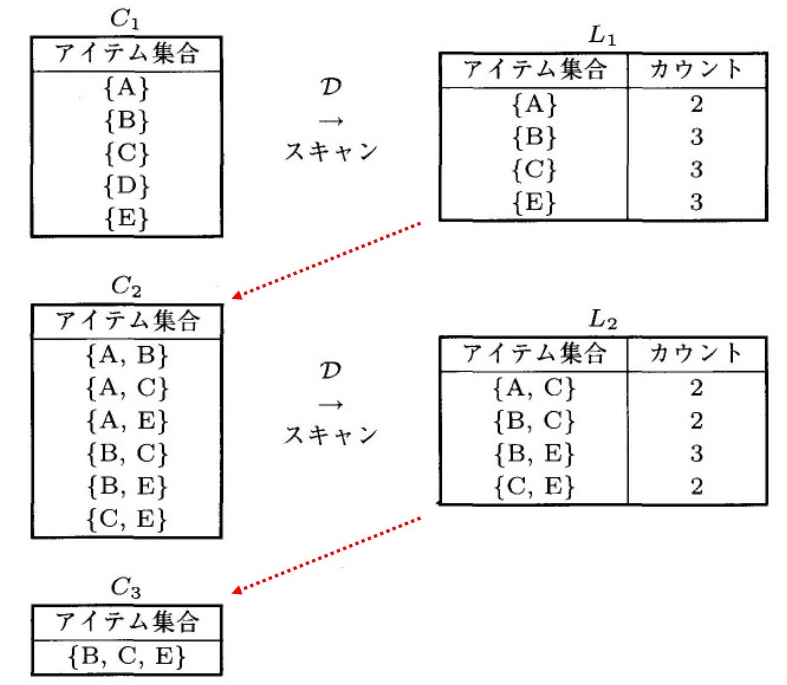


作ったC2のトランザクション中の頻度を調べ，minsup以上の要素だけL2とします：





最終的に以下のL1，L2，L3が得られ，そこからアソシエーションルールを作れば，どのルールもサポートの高いルールになります：



■R によるアソシエーションルール

(1) まずアソシエーションルールの関数を含んだパッケージ arules をインストールする必要があります。「パッケージ」 → 「CRANミラーサイトの設定」 → 「Japan (Tokyo) などを選択」 → 再度「パッケージ」 → 「パッケージのインストール」 → 「“arules”を選択」 → 再度「パッケージ」 → 「パッケージの読み込み」 → 「“arules”を選択」とします。

(2) 分析に使うファイルがあるディレクトリに移動します。左上の「ファイル」から「ディレクトリの変更」などを選んで，以下の fruits.txt のあるディレクトリに移動します。

(3) ファイルの読み込みには以下のようにして read.transactions 関数を用います：

> fr <- read.transactions("fruits.txt", format="basket", sep="\t")

(4) 次に以下のように apriori 関数を用います：

> kekka <- apriori(fr, parameter=list(supp=0.5, conf=0.01, target="rules"))

(5) 得られた結果は summary(ルール) としても見られません。次のようにして見ます：

　　　> inspect(kekka)

あるいは write(kekka, "kekka.txt") などとファイルに出力します。

(6) 上ではサポート重視でルールを出力させました。確信度重視の場合は例えば以下のようにします。

> kekka <- apriori(fr, parameter=list(supp=0.01, conf=0.5, target="rules"))

先ほどとの違いを見比べてみましょう

■練習

(1) shohin.csvを使って，サポート0.03，確信度0.75 でアソシエーションルールを作成し，表示して下さい。

(2) ある食品売上げデータgroceries.csv をダウンロードし，サポート0.001，確信度0.8 でアソシエーションルールを作成し，表示して下さい。