

プログラムのページ (担当 鈴木 誠道)

75-02 パターン付加の手続による教師なし類別プログラム

鳥 岡 豊 士*

1. ま え が き

著者は、さきに KL-直交系¹⁾によって求められるパターンの特徴量 (固有値) に距離の概念を導入した教師なし類別のアルゴリズムを明らかにした²⁾。しかし、このアルゴリズムは複雑で多くの処理時間を要する。そこで、この欠点を除くため改良したアルゴリズムを求めたので、このアルゴリズムを用いて行なった教師なし2類別のプログラムと結果を示す。

2. パターン付加手続による教師なし2類別法

2つのクラスからなるパターン $X_i^{(k)}$ の集合 S を

$$S = \{X_i^{(k)} | k=1, 2, i=1 \sim N_1+N_2\} \quad (1)$$

k : クラスを表示

N_1, N_2 : クラス 1, 2 のパターン数

とすると、パターンを教師なしで類別するための類別基準 C は、クラス 1 とクラス 2 の自己相関行列を G_1, G_2 とすれば

$$C = t_r[G_0^2] \quad (2)$$

$$G_0: G_0 = G_1 - G_2$$

$$\begin{aligned} &= \frac{1}{N_1} \sum_{i=1}^{N_1} X_i^{(1)} X_i^{(1)T} \\ &\quad - \frac{1}{N_2} \sum_{i=1}^{N_2} X_i^{(1)} X_i^{(2)T} \end{aligned}$$

T : 転置ベクトル。

ととまる。そして、この C はクラス間の特徴量の差の平方を表す。従って、最適な類別が行なわれたとき C は最大の値を取る。パターン付加の教師なしアルゴリズムは以上の結果をもとに、一方のクラスのパターン $X_r^{(k)}$ を他のクラスに付加したときの類別基準値の変化量 ΔC

$$\Delta C = K_1 \cdot t_r[G_0^2] + K_2 \cdot t_r[G_0 \cdot \Delta G_0] \quad (3)$$

K_1, K_2 : N_1, N_2 から求められる係数。

ΔG_0 : パターン $X_r^{(k)}$ を付加した結果による G_0 の変化量。

を求め、 $\Delta C \geq 0$ のときパターン $X_r^{(k)}$ を他のクラスに加えながら最適な類別を行なう類別法である。詳しい説明は 2) を参照されたい。

3. 記号と Procedure の内容の説明

プログラム中で使用される記号と手続き (Procedure) の内容を簡単に説明する。

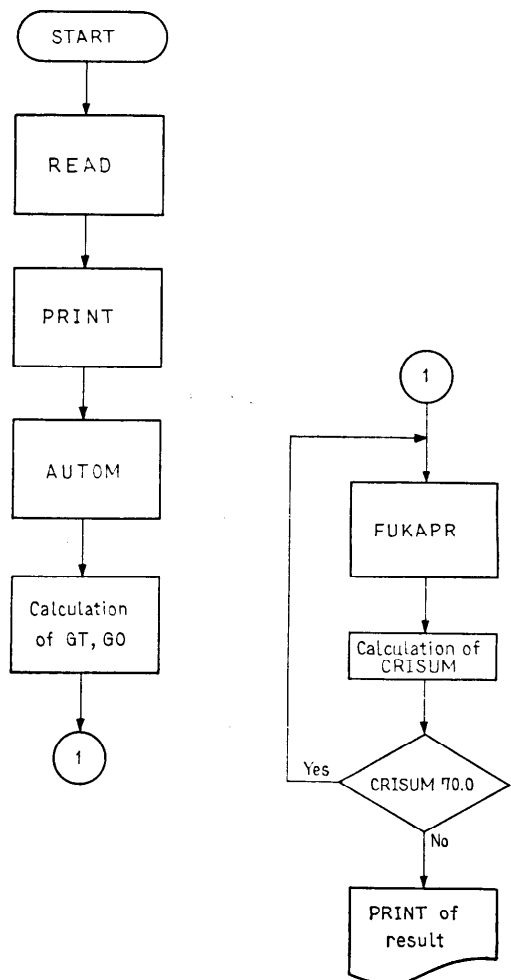


Fig. 1 Flow chart

* 山口大学短期大学部

```

begin
  comment real length=6;
  integer I,J,K,L,M,COUNT,N1,N2;
  real CRI,CRISUM;
  Readinteger(L); Readinteger(M);

  begin
    real array CL1,CL2(1:L,1:M);G0,G1,G2,GT,GV(1:M,1:M);

    procedure READ(X,I,J,n);
      value I,J;
      integer I,J,n;
      real array X;
      begin
        integer I,m;
        real p;
        p:=0;
        for I:=1 step 1 until I do
          begin
            p:=0.0;
            for m:=1 step 1 until J do
              begin
                Readreal(X(I,m));
                p:=X(I,m)*X(I,m);
              end;
            if p>0.0 then
              for m:=1 step 1 until J do
                X(I,m)=X(I,m)/sqrt(p);
              end;
            n:=n+1;
            end else go to END1;
          end;
        END1:
      end;
      end;

    procedure AUTOM(X,Y,J,n);
      value J,n;
      integer J,n;
      real array X,Y;
      begin
        integer I,m,k;
        for I:=1 step 1 until J do
          for m:=1 step 1 until J do
            Y(I,m)=0.0;
            k:=0;
          end;
        BACK:
        k:=k+1;
        for I:=1 step 1 until J do
          for m:=1 step 1 until J do
            Y(I,m)=Y(I,m)+X(I,k)*X(I,k)/float(n);
            if k=n then go to BACK else go to END2;
          end;
        END2:
      end;

    procedure PRINT(X,I,J,k,n);
      value I,J,k,n;
      integer I,J,k,n;
      real array X;
      begin
        integer I,m;
        for I:=1 step 1 until I do
          for m:=1 step 1 until J do
            begin
              if m=1 then CRLF;
              Printfix(X(I,m),k,n);
            end;
          end;
        end;

    procedure FUKAPR(X,Y,Z,W,T,i,n1,n2,c,s);
      value s,i;
      integer i,n1,n2;
      real c,s;
      real array X,Y,Z,W,T;
      begin
        integer I,m,k,k1;
        real a,b,HENKA;
        c:=0.0; k:=0;

      REPEAT:
        k:=k+1;
        k1:=(n1-1)*(n2+1);
        a:=float(n1*n2)/float(k1);
        b:=1.0/float(k1);
        HENKA:=0.0;
        for I:=1 step 1 until I do
          for m:=1 step 1 until I do
            W(I,m)=(Z(I,m)-float(n1*n2)*X(I,1)*X(I,m))*b*s;
          for I:=1 step 1 until I do
            for m:=1 step 1 until I do
              HENKA=HENKA+(a*(2-1.0)*(Z(I,m)*Z(m,1))
                +2.0*a*(W(I,m)*Z(I,m)));
            if HENKA>0.0 then go to L1 else go to L2;
          end;
        L1:
        c:=c+HENKA;
        for I:=1 step 1 until I do
          begin
            Z(n2+1,I)=X(I,1);
            for m:=1 step 1 until I do
              Z(I,m)=a*(Z(I,m)+W(I,m));
            end;
            for I:=k step 1 until n1 do
              for m:=1 step 1 until I do
                X(I,m)=X(I+1,m);
              n1:=n1-1; n2:=n2+1; k:=k-1;
            end;
          if k=n1 then go to REPEAT;
        end;

```

```

START:
  READ(CL1,L,M,N1); CRLF;
  Printstring('THE NUMBER OF PATTERNS IN CLASS1');
  Space(30); Printinteger(N1); CRLF(2); Space(30);
  Printstring('INITIAL STATE OF CLASS1'); CRLF(3);
  PRINT(CL1,L,M,6,4); CRLF(3);
  READ(CL2,L,M,N2);
  Printstring('THE NUMBER OF PATTERNS IN CLASS2');
  Space(30); Printinteger(N2); CRLF(2); Space(30);
  Printstring('INITIAL STATE OF CLASS2'); CRLF(3);
  PRINT(CL2,L,M,6,4); CRLF(3);

  INISET:
    AUTOM(CL1,G1,M,N1);
    AUTOM(CL2,G2,M,N2);
    for I:=1 step 1 until M do
      for J:=1 step 1 until M do
        begin
          G0(I,J)=G1(I,J)-G2(I,J);
          GT(I,J)=float(N1)*G1(I,J)+float(N2)*G2(I,J);
        end;
    COUNT:=0;

  SHICRI:
    COUNT:=COUNT+1;
    CRISUM:=0.0;
    FUKAPR(CL1,CL2,G0,GV,GT,M,N1,N2,CRI,1.0);
    CRISUM=CRISUM+CRI;
    FUKAPR(CL2,CL1,G0,GV,GT,M,N2,N1,CRI,-1.0);
    CRISUM=CRISUM+CRI;
    if CRISUM=0.0 then
      begin
        Printstring('CHECK OF VALUE CRISUM'); Space(13);
        Printfix(CRISUM,8,4); Space(12);
        Printstring('KURIKAESHI KAISU'); Space(12);
        Printinteger(COUNT);
        go to SHICRI;
      end else go to RESULT;

  RESULT:
    Space(30); Printstring('FINAL STATE OF CLASS1');
    CRLF(2); PRINT(CL1,L,M,6,4); CRLF(3); Space(30);
    Printstring('FINAL STATE OF CLASS2'); CRLF(2);
    PRINT(CL2,L,M,6,4); CRLF(3);
  end;
end

```

Fig. 2 処理に用いたプログラム

記号

N_1, N_2 : クラス 1 とクラス 2 のパターン数.

COUNT: パターン付加の手段 (FUKAPR) の使用回数.

CRI: 1 方のクラスのパターンを他方のクラスに付加することによって生ずる類別基準値の総和.

CRISUM: パターン付加によって生ずる類別基準値の総和.

CL1: クラス 1 のパターン集合の格納配列名.

CL2: クラス 2 のパターン集合の格納配列名.

LM: CL1, CL2 の配列の次元, M は入力パターンの次元に相当.

G_1, G_2 : クラス 1 とクラス 2 の自己相関行列.

$G_0: G_0 = G_1 - G_2$.

GT: ΔG_0 に表われる処理中一定な行列.

$GT = N_1 \cdot G_1 + N_2 \cdot G_2$.

手続き

READ: 入力パターンの読み取り.

パターンの正規化.

パターン数 N_1, N_2 の決定.

PRINT: CL1, CL2 の打出し,

k, m は桁数, 小数点以下の桁数の指定.

AUTOM: 自己相関行列 G_1, G_2 の計算.

FUKAPR: パターン付加による教師なし類別法もあり, 類別基準値の変化分 (HENKA) の計算.

HENKA > 0 のときパターンの付加.

CL1, CL2 内のパターンの再配列.

Fig. 1 (p. 246 参照) にプログラムの大略的なフローチャートを示す.

4. 実験と結論

このプログラムの有効性を示すため, 手書き文字 A, B から再構成された 9 次元パターン³⁾ 34 個を類別させた. この結果, 処理による誤まりは 1 個であった. しかも, このパターンは分布状態から見ると明らかに誤まりとなることが判る. したがって, このプログラ

ムを用いて, パターン集合 S を教師なしで分類出来ることが分る. なお, このプログラムでは初期状態としてパターンが任意に 2 つのクラスに分けられているものとして処理を行なった.

Fig. 2 (前頁参照) に処理に用いたプログラムを示す.

参 考 文 献

- 1) S. Watanabe: Knowing and Gessing, pp. 380 ~403 Wiley (1969).
- 2) 瀬良豊士: 直交空間での教師なし類別について, 情報処理, Vol. 14, No. 10 (1973).
- 3) 瀬良豊士: 3×3 素子に Karhunen-Loève 直交系を用いたパターン認識, 情報処理, Vol. 13, No. 4, pp. 210~217 (1972).

(昭和 49 年 5 月 30 日受付)