The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

近傍法による2値化と頻出パターンマイニングを用いた代表点抽出

Finding Representatives via Nearest Neighbor Based Binarization and Frequent Pattern Mining

米田 友花 *1　杉山 麿人 *2　鷲尾 隆 *1
Yuka Yoneda　Mahito Sugiyama　Takashi Washio

*1大阪大学産業科学研究所
The Institute of Scientific and Industrial Research
*2国立情報学研究所
National Institute of Informatics

We propose to find representative data points from continuous data via a two-step procedure: We first binarize data points based on the nearest neighbor search, followed by performing frequent pattern mining on the binarized data. Since frequent patterns correspond to combinations of data points shared by many other data points as their neighbors, they are expected to well summarize the entire dataset. We empirically show that representative data points detected by our method have competitive quality with random sampling in the classification scenario.

1. はじめに

機械学習技術を用いた大量データの解析が，様々な分野で盛んに行われている [3]．例えば，センサーによって温度や心拍数などを連続値データとして大量に取得し，製品開発に役立てることができる．大量のデータを解析することで初めて分かる変数間の関係性など，それまでの人の手による解析では得られなかった知見を得ることができる可能性がある．しかし，大量データ解析には，大きく二つの問題点がある．(1) 取得したデータが大きくなるほど，計算機によって一括で扱うことが困難となり，詳細な解析が行えない．(2) データにはセンシングの際の雑音や複製されたデータが含まれてしまうことがあり，すべてが有用なデータであるとは限らない．雑音データと有用なデータを混同したままデータを解析すると，間違った結果を導く可能性がある．また雑音の少ない大量データであっても効率的に大量のデータ解析を行うためには，できるだけ同じようなデータの処理は避けたい．そこで，大量のデータから不要なデータを取り除き，有用なデータを抽出する手法が求められている．

この目的を達成するため，本研究では，連続値データからデータの代表点を抽出する手法を提案する．提案手法ではまず，連続値データを k 近傍法を用いて二値化する．そして，頻出パターンマイニングを用いることで，頻出するデータ点の組合せを得る．これらのデータ点は，多くのデータ点に対して近傍として共有されているため，データ集合を代表する点となっている．

本稿は以下のような構成となっている．まず 2. 節で提案手法を説明する．最初に概要を述べた後，2.1 節では二値化，2.2 では頻出パターンマイニングについて述べる．3. 節で提案手法の動作例を示した後，4. 節で提案手法を実験によって評価し，結果を考察する．最後に 5. 節で本研究の結論と今後の課題について述べる．

2. 提案手法

提案手法の概要について述べる．提案手法には，2 つの段階があり，第 1 段階は k 近傍法による二値化，第 2 段階は頻出パターンマイニングである．表 1 に概念図を示す．

連絡先: 米田友花, 大阪大学産業科学研究所, 06-6879-8540, 大阪府茨木市美穂ケ丘 8-1, yoneda@ar.sanken.osaka-u.ac.jp

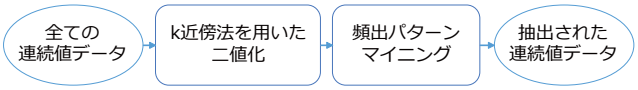

図 1: 提案手法の概念図

具体的には，第 1 段階では，連続値をとるデータにおいて，各データ点に対して他のデータ点との距離を計算し，k 番目に近い点までのデータ点であるかどうかを示す二値データに変換する．この二値データは，パターンマイニングにおけるトランザクションデータに対応する．第 2 段階では，第 1 段階で得たトランザクションデータに対して頻出パターンマイニングを適用することで，複数のトランザクションに頻出するデータ点の組合せを得る．これによって，多くのデータ点に近傍として共有されているデータ点を見つけることができ，データの分布を推定することなくデータの代表点を検出できる．提案手法のアルゴリズムの擬似コードを，Algorithm 1 に示す．2 つの段階におけるそれぞれのアルゴリズムは，2.1 節と 2.2 節で説明する．

提案手法への入力データは，連続値をとる多次元ベクトルの集合である．各データ点は，n 次元の実数値ベクトル $\boldsymbol{x}_i = (x_i^1, x_i^2, \ldots, x_i^n) \in \mathbb{R}^n$ $(i = 1, \ldots, N)$ である．

2.1 二値化

k 近傍法を用いた二値化では，連続値の特徴を要素にもつ各特徴ベクトルを二値化することで，二値の特徴ベクトルを得る．変換された二値特徴ベクトルは，元の特徴空間においてデータ点同士が近いかどうかという情報を保持している．

各データ点 $\boldsymbol{x}_i \in \mathbb{R}^n$ を二値化した後の N 次元二値ベクトルを $\boldsymbol{d}_i = (d_i^1, d_i^2, \ldots, d_i^N) \in \{0,1\}^N$ $(i = 1, \ldots, N)$ と表記する．各要素 d_i^j の値は以下の定義によって与えられる．

$$d_i^j = \begin{cases} 1 & \boldsymbol{x}_j \text{が} \boldsymbol{x}_i \text{から} l \text{番目} (l \leq k) \text{に近い点であるとき,} \\ 0 & \text{その他の場合.} \end{cases}$$

ここで，k は入力パラメーターである．

以下では，二値化の動作について説明する．入力データは n 次元実数値ベクトルであり，出力は N 次元二値ベクトルである．まず，あるデータ点 $\boldsymbol{x}_i$ に注目し，他の点 $\boldsymbol{x}_j$ との距離を計算する．すべてのデータ点との距離を計算した後，$\boldsymbol{x}_j$ と

の距離が k 番目までに近い場合，d_i^j に 1 を代入する．$\boldsymbol{x}_j$ が k 番目よりも遠い点であった場合，d_i^j に 0 を代入する．これらの処理を全てのデータ点に対し実行することで，二値化された N 次元特徴ベクトル $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_N$ を得る．

2.2 頻出パターンマイニング

提案手法では，入力データから得られた二値ベクトルに対して，頻出パターンマイニングを適用することで，例外的なデータを取り除き，データの代表点の組合せを得る．

頻出パターンマイニング [1] とは，組合せの集合から頻出する組合せを取り出すことを目的とするデータマイニング手法の 1 つである．あらかじめ与えられた集合 A について，A の各要素 $i \in A$ を**アイテム**（item）と呼び，A の部分集合 $F \subseteq A$ を**アイテム集合**（itemset）と呼ぶ．**トランザクション**（transacition）T はアイテム集合である．頻出パターンマイニングにおいて解析の対象となるデータは，**トランザクションデータ**（transaction data）と呼ばれるトランザクションの集合 $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$ である．あるアイテム集合 F について，データベース $\mathcal{D}$ における F を含むトランザクションの割合を**サポート**（support）と呼ぶ．あるアイテム集合 F について，データベース $\mathcal{D}$ における F を含むトランザクションの割合を**サポート**（support）と呼び，F のサポート $S(F)$ は $S(F) = |\{T \in \mathcal{D} \mid F \subseteq T\}|/|\mathcal{D}|$ で定義される．サポートは，アイテム集合 F が頻出であるかどうかの判定に用いる．サポートの値が，**最小サポート**（minimum support）s と呼ばれるユーザーが与える閾値以上となるとき，この F を**頻出アイテム集合**（frequent itemset）と呼ぶ．本研究では，全ての頻出アイテム集合のうち，特に**極大頻出アイテム集合**（maximal frequent itemset）と呼ばれるアイテム集合の発見に注目した．極大頻出アイテム集合とは，任意のアイテム $a \in A$ に対して $F \cup \{a\}$ のサポートが s より小さくなるアイテム集合 F のことを指す．極大頻出アイテム集合の部分集合は，必ず頻出アイテム集合となることが分かっている．本研究では，頻出パターンマイニングを適用して極大頻出アイテム集合を得るために，LCM (Linear time Closed itemset Miner) [5] を使用した．

提案手法では，まず二値ベクトルの集合をトランザクションデータに変換する．各二値ベクトル $\boldsymbol{d}_i$ に対して，対応するトランザクションを T_i とする．二値ベクトル $\boldsymbol{d}_i$ に対して，その要素のうち 1 である要素のインデックスをすべて T_i に格納する．つまり，各 T_i は

$$T_i = \{j \in \{1, 2, \ldots, N\} \mid d_i^j = 1\} \tag{1}$$

で与えられる．頻出パターンマイニングに入力するトランザクションデータは，インデックス集合の集合 $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$ である．提案手法において，頻出パターンマイニングによって得られる極大頻出アイテム集合を $\mathcal{S} = \{S_1, S_2, \ldots, S_L\}$ とする．これが提案手法の出力となる．本稿では，この組合せ集合の和集合をとることで，データの代表点を得る．具体的には，$\bigcup \mathcal{S} = S_1 \cup S_2 \cup \cdots \cup S_L \subseteq \{1, 2, \ldots, N\}$ で与えられる．この和集合 $\bigcup \mathcal{S}$ は，元の実数値データのインデックスを示す 1 から N の数字の集合であり，$\bigcup \mathcal{S}$ に含まれるインデックスに対応するデータ点を抽出する．

3. 提案手法の動作例

提案手法の動作例を紹介する．表 1 に示した特徴数 $n = 2$，データ数 $N = 7$ の実数値データを入力とし，まず $k = 2$ として k 近傍法により二値化して，表 2 を得る．そして，各二値

Algorithm 1 提案手法のアルゴリズム

```
Input k, s
for all i = 1 to N do
    for all j = 1 to N do
        Calculate the distance between x_i and x_j
    end for
    for all j = 1 to N do
        if the distance between x_i and x_j is lth smallest
        with l ≤ k then
            d_i^j = 1
        else
            d_i^j = 0
        end if
    end for
end for
for all i = 1 to N do
    Generate T_i by Equation (1)
end for
T ← {T_1, T_2, ..., T_N}
Apply T to frequent pattern mining
                        with the minimun support s
S ← maximum frequent itemsets
Outout ⋃S
```

表 1: 入力の連続値データ

	特徴 1	特徴 2
$\boldsymbol{x}_1$	1.1	2.1
$\boldsymbol{x}_2$	1.9	1.8
$\boldsymbol{x}_3$	2.2	1.2
$\boldsymbol{x}_4$	3.9	5.1
$\boldsymbol{x}_5$	5.2	2.5
$\boldsymbol{x}_6$	5.3	5.4
$\boldsymbol{x}_7$	2.4	4.3

ベクトルを表 3 のようにトランザクションデータに変換した後，最小サポート 4 を満たす極大頻出アイテム集合を頻出パターンマイニングによって得る．この例において抽出されたデータ点は $\boldsymbol{x}_2$ および $\boldsymbol{x}_4$ である．また，図 2 において入力の実数値データ及び抽出後の実数データをプロットした．横軸は特徴 1，縦軸は特徴 2 を表している．

4. 評価実験

提案手法を人工データに適用して性能評価を行い，その結果について考察する．まず，4.1，4.2 節で実験を行った条件について述べる．その後 4.3 節で実験結果を示し，4.4 節で結果について考察する．

4.1 実験環境

実験に用いた計算機は，Intel Core i7-4790 CPU 3.60 GHz のプロセッサ，実験メモリ 8.00 GB，OS は Windows 8 Enterprise 64 bit である．提案手法および比較手法は，ともに Python 言語（version 3.5.2）で記述されている．LCM 本体は，著者によって公開されている C 言語での virsion5.3 の実装 [*1] を用いた．

*1 http://research.nii.ac.jp/~uno/codes-j.htm

表 2: 表 1 に示したデータを二値化して得られる二値データ

	1	2	3	4	5	6	7
$\boldsymbol{d}_1$	1	1	1	0	0	0	0
$\boldsymbol{d}_2$	1	1	1	0	0	0	0
$\boldsymbol{d}_3$	1	1	1	0	0	0	0
$\boldsymbol{d}_4$	0	0	0	1	0	1	1
$\boldsymbol{d}_5$	0	0	0	1	1	1	0
$\boldsymbol{d}_6$	0	0	0	1	1	1	0
$\boldsymbol{d}_7$	0	1	0	1	0	0	1

表 3: 表 2 から得たトランザクションデータ

	トランザクション
T_1	{1,2,3}
T_2	{1,2,3}
T_3	{1,2,3}
T_4	{4,6,7}
T_5	{4,5,6}
T_6	{4,5,6}
T_7	{2,4,7}

4.2 実験方法

本稿では，人工的に生成したデータを用いて評価実験を行った．生成した人工データは，10 個の正規分布からなるデータ数が N，特徴数が n のデータである．生成したデータは，1 つの大きなクラスタと 9 つの小さいクラスタからなり，それぞれデータ数の割合が 0.55 および 0.05 となるように異なる正規分布から生成した．正規分布のそれぞれの平均値はランダムに定めて，分散は 1 に固定した．本実験では，$k = 10$ の k 近傍法を用いた二値化を適用した．提案手法を適用することで，データの代表点を適切に抽出できているかを確認するため，10 個の正規分布からなるデータそれぞれに異なるクラスを割り当てた．そして，(1) データ抽出前の全データ，(2) 提案手法を適用したデータ，また (3) 提案手法と同じデータ数だけランダムに抽出したデータをそれぞれ決定木分析に入力し，10 分割の交差検証法で分類精度を評価した．決定木分析には，CART [2] を採用した．Python の機械学習ライブラリ scikit-learn [4] では，決定木分析の標準的手法として採用されており，本実験でもこの実装を用いた．

本実験においては，2 種類の実験を行った．(i) データ数，特徴数を固定し，最小サポートを変化させたときの，提案手法とランダム抽出における抽出点の数と分類精度の測定および (ii) データ数，特徴数を変化させたときの，全データを使用した場合，提案手法，ランダム抽出における抽出点の数と分類精度の測定である．ただし，実験 (ii) において，提案手法における最小サポートは，極大頻出アイテム集合が存在する最大限の値をそれぞれのデータ数や特徴数に応じて設定した．また，実験結果の値は，小数第 3 位を四捨五入したものを記載した．

4.3 実験結果

実験 (i) では，データ数を 1000，特徴数を 2 で固定し，最小サポートを変化させた場合の提案手法およびランダム抽出における抽出点の数と分類精度を測定した．結果を表 4 に示す．最小サポートが 15 から 17 に増加すると，抽出されるデータ数の平均が 144.9 から 52.5 に減少し，よりデータが圧縮できている．さらに，分類精度はランダム抽出と同程度であり，ランダム抽出においては分類精度が 0.97 から 0.94 の 0.03 だけ下がったことに対し，提案手法では 0.93 から 0.92 の 0.01 の低下にとどまった．

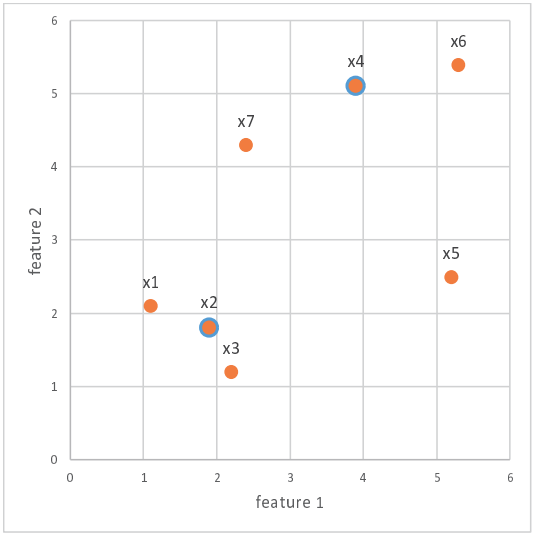

図 2: 実数値データのプロット．提案手法で選び出された点を青色で示す．

表 4: データ数 1000 特徴数 2 のデータにおいて最小サポートを増加させたときの実験結果

最小サポート	抽出点数	分類精度	
		提案手法	ランダム抽出
15	144.9	0.93	0.97
16	88.8	0.93	0.96
17	52.5	0.92	0.94

次に，実験 (ii) において，特徴数を 2 で固定し，データ数を 1000 から 4000 に変化させたときの全データを使用した場合，提案手法およびランダム抽出における抽出点数と分類精度を測定した．結果を表 5 に示す．まず，データ数を 1000 から 4000 まで変化させると，抽出点数が 52.5 から 115.3 と 2 倍程度増加することが分かった．また，分類精度はどの手法も高い値になっている．

最後に，実験 (ii) において，データ数を 3000 で固定し，特徴数を 2 から 10 に変化させたときの全データを使用した場合，提案手法およびランダム抽出における抽出点数と分類精度を測定した．結果を表 6 に示す．まず，特徴数を変化させると，抽出点数が 119.5 から 377.5 へ増加した．ここでも，分類精度はどの手法も同水準となった．

4.4 実験の考察

実験 (i) の表 4 から，データの抽出点の数が大幅に減少したとき，提案手法の方がランダム抽出よりも分類精度の低下が抑えらている．これは，提案手法の頻出パターンマイニングによって，周辺に同じようなデータがたくさんあるデータ点を抽出することができるため，抽出した点が少ない場合でもある程度の分類精度を保つことができるためと考えられる．

また実験 (ii) の表 5 から，データ数を多くしたとき，データ数 1000 のときは提案手法とランダム抽出の分類精度の差が 0.02 だったことに対しデータ数 4000 のときは 0.01 と縮まった．そのため，データが大きくなるほど，提案手法の方がラン

表 5: データ数 N を増加させたときの実験結果

データ数	特徴数	抽出点数	分類精度		
			全データ使用	提案手法	ランダム抽出
1000	2	52.5	0.98	0.92	0.94
2000	2	91.6	0.95	0.93	0.94
3000	2	119.5	0.99	0.96	0.96
4000	2	115.3	0.99	0.98	0.99

表 6: 特徴数 n を増加させたときの実験結果

データ数	特徴数	抽出点数	分類精度		
			全データ使用	提案手法	ランダム抽出
3000	2	119.5	0.99	0.96	0.96
3000	5	359.5	0.99	0.98	0.99
3000	10	377.5	1.00	0.99	0.99

ダム抽出より適切にデータの代表点を抽出することができると考えられる．これは，同じ特徴数のデータにおいては，データ数が増えると，各クラスタのデータの密度が大きくなり，提案手法での頻出パターンマイニングでデータの代表点を適切に抽出することができたからであると考えられる．

最後に実験 (ii) の表 6 から，特徴数を多くしたとき，抽出されたデータ数は大幅に増加し，分類精度は提案手法とランダム抽出は同じように上昇した．これは，データが高次元になると，データ数に対し元の特徴空間が指数関数的に大きくなるため，それぞれのデータ同士の距離の差が少なくなる．その結果，データ周辺にあるデータの数の偏りがなくなり，提案手法における頻出パターンマイニングによって適切なデータを抽出することと，ランダム抽出のようにランダムにデータを抽出することの差がなくなったためと考えられる．

5. おわりに

本稿では，実数値データの集合からデータの代表点を抽出する手法を提案した．この手法では，実数値データを k 近傍法を用いた二値化によって二値データに変換したあと，頻出パターンマイニングによってデータの代表点を抽出する．実験による性能評価の結果，提案手法をもちいることで，特にデータ数が増加させた場合に，ランダムにデータを抽出する手法よりも分類精度を向上させることができた．

一方，今回の評価実験においては，データ数が比較的小さい場合の実験にとどまった．より大規模なデータに対して，提案手法がどのようなふるまいをするのかを評価し考察していく必要がある．また本稿では，提案手法の出力である極大頻出アイテム集合の和集合を取ることで，データ集合の代表点を得た．ここで，和集合を取ることなくアイテム集合を直接利用することで，クラスタリングなどの解析が直接可能である．今後は，このような提案手法の発展に取り組む予定である．

参考文献

[1] C. C. Aggarwal and J. Han, editors. *Frequent Pattern Mining*. Springer, 2014.

[2] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[3] J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of Massive Datasets*. Cambridge University Press, 2014.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] T. Uno, T. Asai, Y. Uchida, and H. Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *Discovery Science*, pages 16–31, 2004.