2E2-4

クラス確率密度の高速な推定による近似 kNN 識別

Fast density estimation for approximated k neatest neighbor classification

小林 卓夫[*1]
Takao Kobayashi

清水 郁子[*2]
Ikuko Shimizu

[*1] [*2] 東京農工大学
Tokyo University of Agriculture and Technology

In this paper we propose a method for fast density estimation of samples, which allows to accelerate classification based on the k nearest neighbor method significantly. Our chief idea is that many trials of a rough estimation of probability density function are performed and they are integrated by Bayes' theorem.

1. はじめに

k 最近傍識別法(kNN 法)は最も良く知られた古典的なパターン認識手法である[Cover 1969][Fukunaga 1975]. kNN 法は多クラス識別にもそのまま適用でき, 十分なサンプルが与えられれば高い識別率が得られるため, 実用的にも強力な手法である.

kNN 法は学習サンプル数に比例した識別時間がかかるため, その高速化についてはこれまでに多くの研究がされてきた.

計算幾何学における最近傍探索アルゴリズムの研究では kd tree[Arya 1993], Locality Sensitive Hashing (LSH)[Indyk 1998] などが知られており, これによって kNN 識別処理を高速化することができる. しかしながら, 最近傍探索の計算量はプロトタイプ数に依存するため学習サンプル数が増加すれば識別時間の増大は避けられない. 本来, kNN 識別に対して最近傍探索を経由することはより難しい問題を解くことになるといえる.

一方, kd tree においてクラス識別境界のみに着目して高速化・メモリ削減を行う識別手法の kd-decision tree が提案されている[Shibata 2003]. しかしながら, 識別器作成に多大な時間がかかることと, 候補情報が得られないという問題がある.

kNN 識別は未知入力に対するクラス確率密度関数を推定することによって識別処理を行う. 従って, 未知入力の近傍の密度関数が高速に推定できればよい. 本稿では, 確率密度関数の粗い推定を多数回行い, それをベイズの定理を使って統合するという方法を提案する. 提案手法の識別処理は学習サンプル数に依存せず定数時間であり, また, その学習は学習サンプル数に比例した時間で行える.

2. 確率密度関数の推定

kNN 識別は, 未知入力 $\mathbf{x}$ を中心とした k 個の学習サンプルが含まれる超球内のクラス密度を推定することで行う(図 1).

n_j を超球内の各クラス C_j のサンプル数とする($\Sigma n_j = k$). このとき比 n_j / k は $\mathbf{x}$ の近傍のクラス確率密度の粗い推定となる.

$$p(\mathbf{x}|C_j) \cong \frac{n_j}{k} \cdot \frac{p(\mathbf{x})}{\Pr(C_j)} \tag{1}$$

よって条件付確率 $\Pr(C_j \mid \mathbf{x})$は次のようになる.

$$\Pr(C_j|\mathbf{x}) = p(\mathbf{x}|C_j) \cdot \frac{\Pr(C_j)}{p(\mathbf{x})} \cong \frac{n_j}{k} \tag{2}$$

連絡先:小林 卓夫, 東京農工大学工学部, 〒184-8588 東京都小金井市中町 2-24-16, tkobaya@cc.tuat.ac.jp

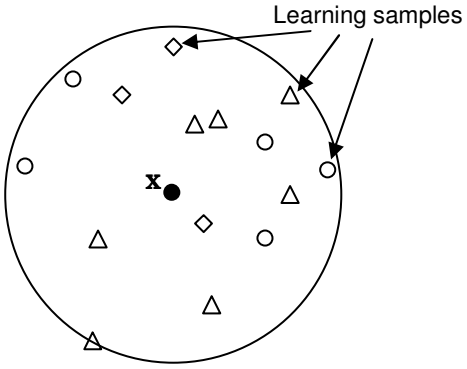

図 1 kNN 法の仕組み

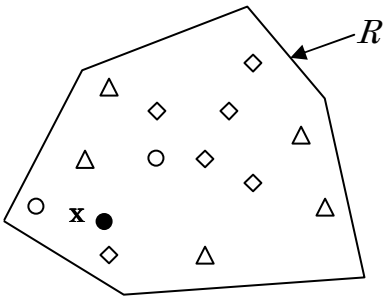

図 2 領域 R 内のサンプル数計測による x の近傍のクラス確率密度の推定

ここで, 確率密度関数の推定に $\mathbf{x}$ を中心とした超球領域を使用する代わりに, 次のように行うことを考える.

特徴空間 S を多数の領域に分割する. すなわち各領域は互いに重ならず, かつ全ての領域の和集合は S に等しい. 従って, 全てのサンプルはどれかの領域に必ず属する. このとき $\mathbf{x}$ が属している領域 R に含まれる各クラスの学習サンプルの比率を $\mathbf{x}$ の近傍における密度の近似と仮定する.

$$p(\mathbf{x}|C_j) \cong \Pr(\mathbf{x} \in R \,|\, C_j) \cong \frac{n_j}{\sum n_i} \cdot \frac{\Pr(\mathbf{x} \in R)}{\Pr(C_j)} \tag{3}$$

より,

$$\Pr(C_j|\mathbf{x}) \cong \Pr(C_j|\mathbf{x} \in R) \tag{4}$$

であるから,

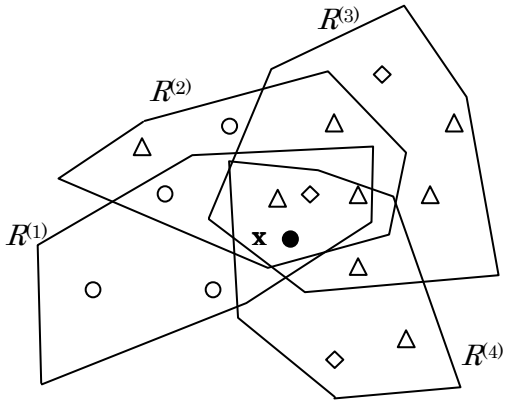

図 3 多数の試行による x 近傍のクラス確率密度の推定

$$\Pr(C_j|\mathbf{x}\in R)\cong\frac{n_j}{\sum n_i} \tag{5}$$

とクラス確率が見積もられる. 全ての領域 R に対してそれぞれの各クラスのサンプル数 n_j はバケット法により簡単に求められるので, この方法で識別を行えば学習・識別処理ともに非常に高速である.

しかしながら, 全ての領域が球形をしているわけではないし, しかも **x** はいつも領域の中心に来るわけではない. 従って, この方法ではおそらく kNN 法に比べてかなり低い識別能力しか達成されないであろう.

そこで次のように考える. もし, 式(5)の見積もりを多数回試行すれば, 見積もられる値の信頼性を高めることができるのではないか.

そこで空間分割のし方を多種類用意する. 空間分割が L 通りあるとして, **x** はそれぞれの場合において領域 $R^{(l)}$に属しているとする. 図 3 ではわれわれのアイデアを 2 次元的に図解している. これによると, それぞれの領域は相当部分が互いに重なり合っているため, それぞれの見積もりが条件付独立を仮定できないので, 多数回試行しても見積もりの精度は上がらないと思えるかもしれない. しかしながら, 数値実験により高次元空間の場合には領域の重なりは非常に小さくなることがわかっている(本研究では概ね 15 次元以上の場合を扱っている). 従って, 見積もりはそれぞれ独立であると仮定して議論を進める.

x がクラス C_jに属する条件付確率は

$$\Pr(C_j|\mathbf{x})\cong\Pr(C_j|R^{(1)}\cdots R^{(L)}) \tag{6}$$

により推定され, 条件付独立を仮定してベイズの定理を適用すると,

$$\Pr(C_j|R^{(1)}\cdots R^{(L)})= \Pr(C_k)\cdot\frac{\prod_{l=1}^{L}\Pr(R^{(l)}|C_j)}{\prod_{l=1}^{L}\Pr(R^{(l)})}=\frac{\prod_{l=1}^{L}\Pr(C_j|R^{(l)})}{\Pr(C_j)^{L-1}} \tag{7}$$

となる. 識別とはクラス確率の最大となるものを採用するので, 確率の対数を取ってもよい.

$$\underset{j}{\operatorname{argmax}}\{\Pr(C_j|\mathbf{x})\}=\underset{j}{\operatorname{argmax}}\{\log(\Pr(C_j|\mathbf{x}))\} \tag{8}$$

これらより,

$$\log(\Pr(C_j|\mathbf{x}))\cong\log(\Pr(C_j|R^{(1)}\cdots R^{(L)}))= -(L-1)\log(\Pr(C_j))+\log\left(\sum_{l=1}^{L}\Pr(C_j|R^{(l)})+\varepsilon\right) \tag{9}$$

を得る. ここで, ε は log 0 の計算を回避するラプラスコレクションである.

x に近いサンプルほど **x** と同じ領域に属する確率が高いので, 確率密度関数の推定値に大きく寄与する. 従ってこの方法はある種の重み付き kNN 法を近似すると考えられる. kNN 法には多数のバリエーションがあり, サンプルを単純に数えるのではなく, **x** からの距離に応じた重みを付ける方法がある[Dudani 1991].

3. 従来手法との関連

われわれはベクトルの分散表現を用いた高速な学習・識別方法を提案した[Kobayashi 2009]. 分散表現とは, ベクトルを複数のベクトルで表現することであり, これにより高速に識別・探索などのアルゴリズムが実現できる.

ベクトル **x** が分散表現により L 個のベクトル q_1, …, q_Lで表現されるとき,

$$\Pr(C_k|x)=\Pr(C_k|q_1,\cdots,q_L) \tag{10}$$

であるとして, Naïve Bayes が適用できるなら,

$$\Pr(C_k|q_1,\cdots,q_L)=\Pr(C_k)\cdot\frac{\prod_{l=1}^{L}\Pr(q_l|C_k)}{\prod_{l=1}^{L}\Pr(q_l)} =\Pr(C_k)^{-(L-1)}\cdot\prod_{l=1}^{L}\Pr(C_k|q_l) \tag{11}$$

となり, これは式(7)と同じである. Naïve Bayes は文書分類などで成功している手法であるが[Joachims 1996], 式(11)においては q_l を出現した単語と見立てることができ, 文献[Kobayashi 2009]では非常な高次元空間における線形識別関数として考察している.

分散表現は具体的に次のような方法を用いる. Q は非常に多数の点を持つ有限集合であり, かつ任意の **x** に対して **x** から Q の最近傍が高速に求められるものとする. **x** から Q の最近傍を Q(**x**)と表記することにする. L 種類の集合 Q_1, …, Q_L を適切に用意して q_l=Q_l(**x**) (l=1, …, L) とすることにより, 良い効果が得られることが示されている[小林 2006-2].

Q(**x**)という操作は x に対してハッシュコードを計算するとも解釈でき, そしてそれは, Locality Sensitive Hashing スキームの条件を満たす[Indyk 1998][Charikar 2002]. すなわち, 2 つの点 **x** と **y** が近いほど確率

Pr(Q(x)=Q(y))

は 1 に近くなる. また, Q(**x**)という操作は空間を Q の点によりボロノイ領域分割することに他ならないから, この観点では本稿で提案する領域分割を行う方法に包含される. 4 章では従来提案されている以外の新しいさまざまな領域分割方法を提案するとともに有効性を示す.

次に LSH スキームとの関連を説明する. LSH スキームを利用した最近傍探索手法が提案されている. すなわち,

$$sim(p,q)\cong\Pr(H(p)=H(q)) \tag{12}$$

であることを利用し, L 個のハッシュ関数 H_lを用いて,

$$\sum_{l=1}^{L}\delta\left(H_l(p_i),H_l(q)\right)\geq\theta \qquad (13)$$

であるプロトタイプ p_iをクエリ q の最近傍候補として，それらについてのみ詳細に距離計算をすることで高速な探索を行う．ここで δはクロネッカーのデルタ関数であり，θ は一般に 1 とする．

式(6)と(12)，(9)と(13)は対応している．提案手法のデータ構造は LSH による最近傍探索で使用するハッシュテーブルの中のプロトタイプ情報がクラス情報と重み係数に置き換わったものになる．そして本方式では，プロトタイプ情報を使用せず最近傍探索問題を解かないで直接クラス識別を行うため，識別時間は学習サンプル数のオーダーにはならずにクラス数に比例する．従って非常に高速である．

なお，提案手法の特殊な形と解釈できる手法は LSH が提案されたよりも早く考案されて実用化され，産業的有用性が示されている[Kobayashi 1997]．

4. 効果的なハッシュ関数

多くのパターン認識では学習・識別において特徴ベクトルの大きさを正規化している．そのため，本稿でも特徴空間として特に球面空間を扱う．すなわち特徴ベクトルの属する空間 S は

$$S\subseteq\left\{\mathbf{x}\,\middle|\,\|\mathbf{x}\|=1\right\}$$

であるとする．

高速に計算できるハッシュ関数 H が存在するとき，ランダムな直交行列 T を使って新しいハッシュ関数 H'を次のように定義できることは知られている[Datar 2004][小林 2006-1]．

$$H'(\mathbf{x})=H\circ T(\mathbf{x})=H(\mathbf{Tx})$$

こうすることでハッシュ関数を必要なだけ生成できる．

では識別に有効なハッシュ関数は具体的にどのようなものが存在するだろうか．[小林 2006-1]ではバイナリ空間$\{-1,+1\}^{d'}$を使用した方法が提案されている．ただしこの方法では特徴ベクトル次元 d に対して領域分割数は最大 2^d 個という制約がある．

本研究ではこれら以外にも特徴ベクトルから効率的に計算できるさまざまな計算操作を用意する．これら操作を組み合わせることによりハッシュ関数は

$$H(\mathbf{x})=\left(f_1(\mathbf{x}),\cdots,f_T(\mathbf{x})\right)$$

と構築される．ここで $f_t\ (1\leqq t\leqq T)$ は計算操作を行う関数である．

まず，ベクトルから高速に計算できるさまざまな操作を以下に挙げる．ここで，ベクトル $\mathbf{x}=(x_1,\ldots x_d)^t$ に対して，i_1, …, i_d および j_1, …, j_d は以下であるとする．

$$\left|x_{i_1}\right|>\left|x_{i_2}\right|>\cdots>\left|x_{i_d}\right|,$$

$$x_{j_1}>x_{j_2}>\cdots>x_{j_d}.$$

また，$1\leq a\leq d$であるとする．

(i) $sign\left(x_i\right)$

(ii) $sign\left(\left|x_i\right|-\left|x_j\right|\right)$

(iii) $sign\left(x_{i_a}\right)\cdot i_a$

(iv) j_a

(v) $\{j_{n_1},\ldots,j_{n_a}\}$ ($\mathbf{x}$ を a 個の整数の集合に変換する)

これらを適切に組み合わせてハッシュ関数を作成する．このようにして出来るハッシュ関数が意味するところの空間分割操作を視覚的に理解するため，3 次元球面の例を図 4 に示す．

次元数が大きい場合には，これら操作を組み合わせることにより非常に多数の領域に分割するハッシュ関数を構築すること

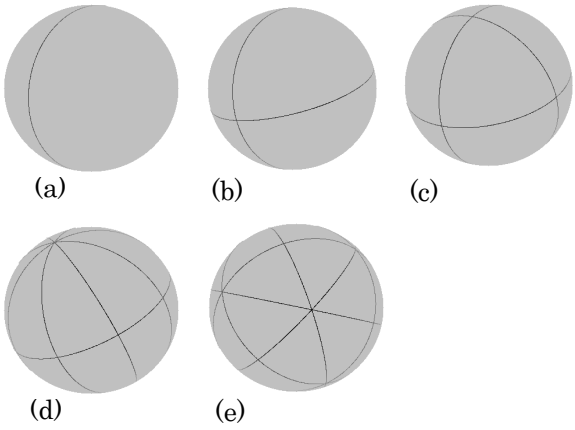

図 4　3 次元球面の分割例:

(a) $H(\mathbf{x})=sign(x_1)$.

(b) $H(\mathbf{x})=\left(sign(x_1),sign(x_2)\right)$.

(c) $H(\mathbf{x})=\left(sign(x_1),sign(x_2),sign(x_3)\right)$.

(d) $H(\mathbf{x})=\left(sign(x_1),sign(x_2),sign(x_3),sign(|x_1|-|x_3|)\right)$.

(e) $H(\mathbf{x})=\left(sign(x_{i_1})\cdot i_1,sign(x_{i_2})\cdot i_2\right)$.

が出来る．例えば，

$$H(\mathbf{x})=\left(sign(x_1),\cdots,sign(x_a)\right)$$

は，S を 2^a 個の領域に分割する．また，

$$H(\mathbf{x})=\left(sign(x_1),\cdots,sign(x_d),\right.$$
$$\left.sign(|x_1|-|x_2|),\cdots,sign(|x_{2i-1}|-|x_{2i}|)\right)$$

は，$i\leq d/2$ のとき S を 2^{d+i} 個の領域に分割し，

$$H(\mathbf{x})=\left(sign(x_1),\cdots,sign(x_d),\right.$$
$$\left.sign(|x_1|-|x_2|),\cdots,sign(|x_i|-|x_j|),\cdots,sign(|x_{d-1}|-|x_d|)\right)$$
$$(1\leq i,j\leq d\ \ (i\neq j))$$

は，S を $2^d\,d!$個の領域に分割する．

$$H(\mathbf{x})=\left(sign(x_{i_1})\cdot i_1,\cdots,sign(x_{i_a})\cdot i_a\right)$$

は，S を$2^a\,d!/a!$個の領域に分割する．

$$H(\mathbf{x})=\left(\{j_{n_1},\ldots,j_{n_a}\}\right)$$

は，S を${}_dC_a$ 個の領域に分割する．

5. 実験

提案手法の性能を検証するため，実際のデータを使って実験を行った．実験には文字認識用の BIRDS データベースの中の手書き数字 19 万サンプルを使用した(http://www.geocities.jp/onex_lab/birdsdb/birdsdb.htm)．特徴ベクトルの作成方法は文献[小林 2006-2]の方法を使用し，次元数 15，24 の場合について学習サンプル数に対応する認識率と認識時間を測定した．比較のために kNN 法の性能も測定した．

提案手法の認識性能，すなわち認識率と認識速度は使用するハッシュ関数の形および数によって変わる．図 5 および表 1 に示す提案手法の計測値は，さまざまなハッシュ関数を試した上で認識率と認識速度の両方が比較的良好となったものを採用している．

提案手法は学習サンプル数が同じ場合には kNN 法よりも幾分認識率が低いが，これは本手法は kNN 識別の近似アルゴリ

ズムであり, 密度推定の精度に起因するためと考えられる. しかしながら, 学習サンプル数を増やしていくと認識率は上昇し, kNN 法に追随していく. また, 認識時間は kNN 法よりも大幅に速く, 学習サンプル数が増えると 50 倍以上高速になる.

6. まとめ

本稿では未知ベクトルに対する高速な確率密度関数の推定による高速な kNN 識別の近似手法を提案した. 提案手法の主要なアイデアは, locality sensitive なハッシュ関数によってベクトルの近傍の粗いクラス密度の推定を行い, それを多数回行ってベイズの定理で統合するというものである. 提案手法は, 非常に高次元の空間における Naïve Bayes による線形識別を行うという過去に提案した手法と一致する. 本稿ではこれと kNN 法との関連性を明らかにした.

また, 実際に良い識別性能を示すことが出来るハッシュ関数の作成方法を提案し, 実験により有効性を確認した.

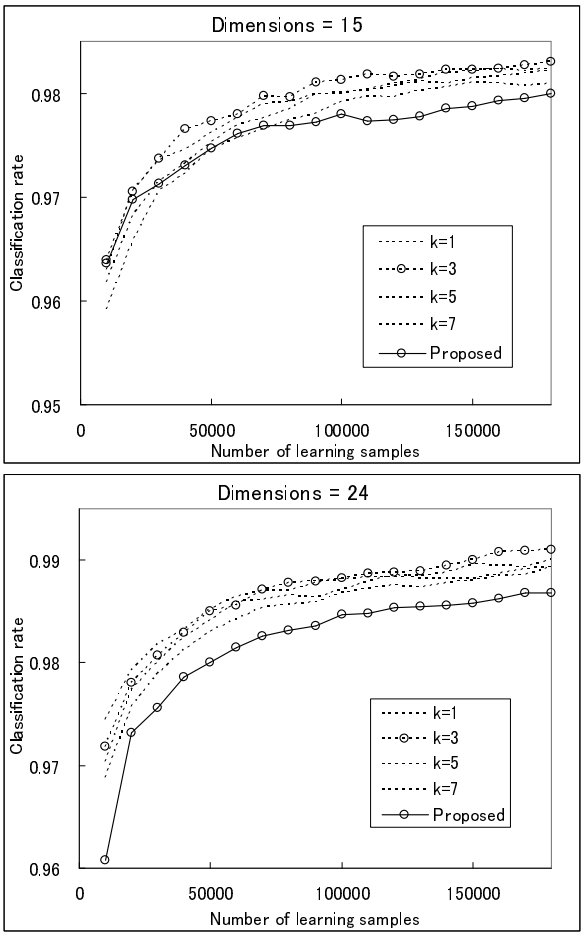

図 5 学習サンプル数と認識率の関係

表 1 認識時間の比較

次元	認識率同一の条件における学習サンプル数		認識時間の比 (3-NN 対提案手法)
	3-NN	提案手法	
15	20,000	30,000	12.2
	50,000	90,000	30.5
	80,000	180,000	50.3
24	20,000	40,000	16.9
	40,000	80,000	31.5
	70,000	180,000	58.4

今後は手法の更なる改良を行い, 実際のデータに対して適用し有効性を検証する予定である.

謝辞

本研究の一部は文科省特別教育研究費共生情報工学研究経費による.

参考文献

[Arya 1993] S. Arya, D. M. Mount, “Approxiamte Nearest Neighbor Queries in Fixed Dimensions,” In Proceedings of the 4th Simposium on Discrete Algorithms(SODA), pp.271-280, Jan. 1993.

[Charikar 2002] M. S. Charikar, Similarity Estimation Techniques from Rounding Algorithms, Proceedings of the 34th Annual ACM Symposium on Theory of Computing 2002.

[Cover 1969] T.M. Cover and P.E. Hart, Nearest neighbor pattern classification. IEEE Trans. Inform. Theory 13 (1967), pp. 21–27.

[Datar 2004] M. Datar, N. Immorlica, P. Indyk and V. S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, Proc. 19th Annual Symposium on Computational Geometry, pp.253-262, Jun. 2004.

[Dudani 1991] S.A. Dudani, The Distance-Weighted k-Nearest Neighbor Rule, Neighbor Neighbor Norms: NN Pattern Classification Techniques, IEEE Computer Society Press, Los Alamitos, CA, 1991, pp. 92–94.

[Fukunaga 1975] K. Fukunaga and L.D. Hostetler, k-nearest-neighbor Bayes-risk estimation. IEEE Trans. Inform. Theory 21 (1975), pp. 285–293.

[Indyk 1998] P. Indyk, and R. Motwani, “Approximate nearest neighbors: Towards removing the curse of dimensionality,” Proc. 30th ACM Symposium on Theory of Computing, pp.604-613, 1998.

[Joachims 1996] T. Joachims, A probabilistic analysis of the rocchio algorithm with TD.IDF for text categorization. Technical Report CMU-CS-96-118, Carnegie-Mellon Institute, 1996.

[Kobayashi 1997] T. Kobayashi, Method of and apparatus for pattern recognition and method of creating pattern recognition dictionary, United States Patent, 5,689,584, Nov. 1997

[Kobayashi 2009] T. Kobayashi and I. Shimizu, A linear classification method in a very high dimensional space using distributed representation, In: P. Perner (Ed.), Machine Learning and Data Mining in Pattern Recognition, Lnai 5632, Springer Verlag, Heidelberg, 2009, p.137-147.

[Shibata 2003] T. Shibata, T. Kato, and T. Wada. K-d decision tree: An accelerated and memory efficient nearest neighbor classifier. In ICDM’03, pp. 641–644, Nov 2003.

[小林 2006-1] 小林卓夫, 中川正樹, “分散コーディングによる高次元の最近傍探索,” 信学技報, PRMU2006-41, pp.13-18, Jun. 2006.

[小林 2006-2] 小林卓夫, 中川正樹, “線形時間学習及び定数時間識別の一パターン識別手法,”信学論(A), Vol.J89-A, no.11, pp.981-992, Nov. 2006.