

# ランダムフォレスト

波部 斉<sup>1</sup>

**概要:** ランダムフォレストはアンサンブル学習による機械学習アルゴリズムの一つである。複数の決定木 (tree) を弱識別器として用い、その結果を統合 (forest) して正しい結果を得る。Leo Breiman による 2001 年の論文以降様々な分野に適用されてきたが、近年コンピュータビジョン分野でも盛んに用いられている。本稿では、その概要、最近の適用例、また、利用時の留意点などについて概観する。

**キーワード:** 決定木, アンサンブル学習, ブートストラップ

## Random Forests

HITOSHI HABA<sup>1</sup>

**Abstract:** Random Forests is a machine learning framework that consists of many decision trees. It can be categorized as an ensemble classifier in which each decision tree performs as a weak classifier. Since it was originally developed by Leo Breiman in 2001, it has been applied to various application scenarios including computer vision. Especially, recent successful results have been attracting researcher's attention. This report introduces a generalized definition of Random Forests and discusses important factors when one would like to apply it to their purpose.

**Keywords:** Decision Tree, Ensemble Learning, Bootstrap

### 1. はじめに

コンピュータビジョンの様々なタスクの中で、パターン認識・識別は大きな注目を集めている。Support Vector Machine[7] や Adaboost[10] などさまざまな手法が利用されているが、最近の研究例では Random Forests[4] が用いられている場面を見ることが多い。

Random Forests はアンサンブル学習の一種であり Leo Breiman によって 2001 年に提案されたものである。パターン認識をはじめとして、回帰、クラスタリングに利用できる特徴をもっている。コンピュータビジョン関連の分野では、Lepetit による特徴点対応付け [16] や、Shotton による画像セグメンテーション [24] が知られていたが、ここ

数年で特に応用例が増えているように思われる。その理由としては、単純な構造にかかわらず良好な結果が得られていること、ますます大規模になる (教師) データベースに適していること、学習・評価が高速に行えること、(先に述べたように) パターン識別だけでなく回帰など様々な問題に適用可能であることなどが挙げられる。

このような状況で、もはや Random Forests はコンピュータビジョンの定番ツールとして位置づけられるものとなっているといえる。本稿では、そのアルゴリズムと特徴、および様々な適用例を示した上で、実際に利用する上での留意点などにも言及して Random Forests を用いようとする学生・初学者の一助になることを目指している。

インターネット上をみると Random Forests に関する多数の文献を見つけることができる [2], [6], [8], [20]。本稿はそれらを参考にしながら最新の事例や考察を加えながらまとめたものである。以下、本文中では [8] の表記法に習い

<sup>1</sup> habe@info.kindai.ac.jp  
近畿大学理工学部情報学科  
Department of Informatics, School of Science and Engineering, Kinki University  
3-4-1, Kowakae, Higashi-Osaka-shi, Osaka 577-8502, Japan

ながら解説を進めていく\*1。

## 2. Random Forests の基本

Random Forests はその名が示しているとおり、複数の木 (tree) を用いて森 (forest) を構成して識別などを行う機械学習アルゴリズムである。ここでいう木は決定木 (decision tree) のことであり、個々の決定木は高い識別性能をもつわけではないが、それらを複数用いてそれぞれの結果を補うことによって高い予測性能を得ることが一つの特徴である。これは機械学習の分野ではアンサンブル学習 (ensemble learning) と呼ばれており、個々の決定木がアンサンブル学習における弱識別器 (weak classifier) に相当している。

このようなアイデアは決して新しいわけではなく、弱識別器としての決定木を統合して強識別器を構成する方法について議論が行われてきた。例えば、T. K. Ho による [14][15] ではランダムな特徴選択によって複数の決定木を構築する方法を提案している。

これに対して、Breiman による Random Forests[4] は自らが提案したバギング (bagging) [3] を利用して弱識別器となる決定木を構築するものである。現在、コンピュータビジョンのコミュニティで広く用いているのはこの Breiman による提案をもとにしたものである。以下ではその学習・評価の流れを紹介する。

### 2.1 決定木

決定木 (decision tree) は、図 1 に示すように、それぞれの接点 (node) ではその子のうちどれに進むべきかを決定する分割関数 (split function) が与えられ、木の葉 (leaf) に最終的な出力結果が対応付けられたものである。

$d$  次元の入力データ  $\mathbf{v} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d$  が決定

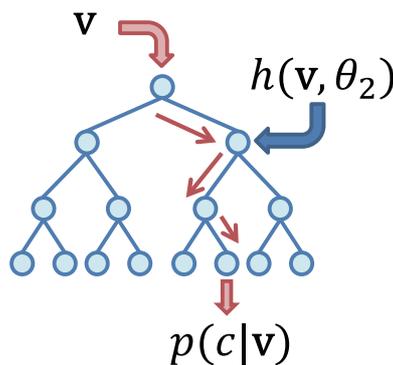


図 1 決定木

\*1 本稿で扱う “Random Forests” は、Random Forest, Randomized Decision Trees, Randomized Decision Forests など多くの名称で呼ばれているがいずれもほぼ同じものを指している。この中で本稿では Leo Breiman にならって Random Forests という呼称を採用する。なお米国において Random Forests は統計解析ソフトウェアの商標として登録されている (78642027) ので注意が必要である。

木に与えられたとき、それはまず根に入力される。根では、分割関数の評価結果によっていずれかの子ノードにデータが入力される。以降、この処理が再帰的に繰り返されて葉ノードに達したときに処理が終了し、葉ノードに対応付けられた結果が出力される。

これを基本として以降のように Random Forests が構成される。なお、一般の決定木はあるノードに対する子の数は任意であるが、Random Forests では図に示しているように二分決定木 (binary decision tree) が主に用いられる\*2。

#### 2.1.1 ノードでの分割関数

二分決定木の場合、ノード  $j$  における分割関数を、

$$h(\mathbf{v}, \theta_j) \in \{0, 1\}, \quad (1)$$

の形で書くことができる。ここで  $\mathbf{v}$  がノード  $j$  に到達したデータ、 $\theta$  が分割関数を決定するパラメータであり、具体的には  $\theta = (\phi, \psi, \tau)$  で構成される。 $\phi$  は  $d$  次元のデータ  $\mathbf{v}$  から幾つかの次元 (特徴) を抽出するフィルタ、 $\psi$  は分割基準 (特徴空間中の超平面など) の幾何特徴を決めるパラメータ、 $\tau$  は分割を行う際の閾値を示している。

たとえば、線形な分割関数を用いる場合には一般に

$$h(\mathbf{v}, \theta_j) = [\tau_1 > \phi(\mathbf{v}) \cdot \psi > \tau_2], \quad (2)$$

と定義できる。ここで関数  $[\cdot]$  は引数が真のときに 1 を、偽のときに 0 を返すものとする。もし、 $\tau_1 = \infty$  あるいは  $\tau_2 = -\infty$  であれば単一の不等式の真偽で分割する単純なものとなる。

フィルタ  $\phi$  は扱うデータに応じて定義する必要があるが、コンピュータビジョンの分野で広く用いられているのは、図 2 のように処理対象の局所パッチの中である 2 点を取り出し、その差分を計算するというものである [11], [16]。これは非常に単純な演算であるが、照明条件変化などの影響を受けない安定した特徴量であると言える。

なお、図 2 では濃淡画像の 2 画素の差分をとっているが、それだけではなく、パッチ内で計算した SIFT 特徴量を加えた中から 2 つの要素を選ぶことも行われている。さらに、後に述べるデプスマップを用いたパターン識別や、

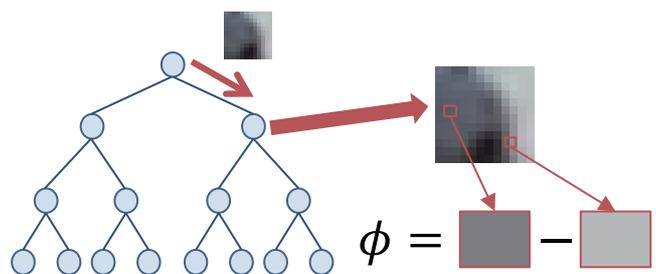


図 2 特徴の抽出

\*2 [8] では複数の子ノードをもつ決定木でも大きな性能向上が望めないと述べられている。

CT 画像からの回帰処理の場合も同様に 2 つの要素を選んでその差を特徴量としている。

このような特徴選択をした場合は特徴量はスカラー値となるため、ある閾値との大小関係を評価して分割関数の出力とすることになる。

### 2.1.2 決定木の出力

このように構成された決定木の葉ノードでは、学習時に割り当てられたデータ（集合）を利用して、評価時の予測に用いるデータが対応付けられる。例えば、クラス識別に用いる場合には、クラスラベル  $c$  の事後確率  $p(c|v)$ （の予測値）が葉ノードに対応付けられる。

以降では、この決定木を基本要素として構成される Random Forests の学習について述べていく。

## 2.2 Random Forests の学習

これまでに述べていたように、Random Forests は複数の決定木（以下、個数を  $T$  とする）から構成されるが、学習過程ではそれぞれの決定木で独立して学習が行われる。

与えられる学習データ全体を  $S = \{v\}$  とすると、決定木毎に学習データ  $S_0 (\in S)$  が選ばれて学習が開始する。決定木の学習は古典的な“C4.5”などと同様に、根から順に各ノードにおいてそこでの学習データを適切に分割する分割関数を選んでいき、ある特定の条件を満たすときに学習を停止してそのノードを葉ノードとする。以下にその流れを解説する。

### 2.2.1 ランダム性の導入

Random Forests の大きな特徴は、個々の決定木のその学習の際に「ランダム性」を巧妙に導入しているところにある。これによって、それぞれの決定木が弱識別器として相補的に働き、過学習を防いで高い汎化性能を得ることができる。

具体的に、ランダム性は (1) 個々の決定木を学習する際の学習データのサンプリング（バギング）と (2) 決定木の各ノードでの分割関数学習の 2 点で導入されている。前者は通常用いられているバギングと同様に  $t$  番目の決定木の学習に利用する学習データ  $S_0^t \in S$  を選択する際に、 $S$  から（重複を許して）ランダムに抽出する。

後者について、1 式で定義された分割関数のパラメータ  $\theta_j$  の取り得る値の集合を  $\mathcal{T}$  とすると  $\mathcal{T}$  は一般に非常に大

きな集合になるため、その一部  $\mathcal{T}_j \in \mathcal{T}$  だけを利用することになる。その選択をランダムに行うというものである。

ランダム性の“度合い”は比  $|\mathcal{T}_j|/|\mathcal{T}|$  で制御できる。通常  $|\mathcal{T}|$  は常に一定であるためパラメータ  $\rho = |\mathcal{T}_j| \in \{1, \dots, |\mathcal{T}|\}$  によってランダム性が表現できる。もし、 $\rho = |\mathcal{T}|$  であればすべての決定木で同じ学習データを用いることになるのでそこにランダム性は存在せず、 $\rho = 1$  であれば完全にランダムになって相関の無い決定木が得られる。なお、通常  $\rho$  はすべての決定木のすべてのノードで一定の値とすることが多い。

### 2.2.2 ノードでの分割関数の学習

根ノードに学習データ集合  $S_0^t \in S$  が与えられれば、根ノードにおいて最適な分割関数を求め、以降ではその子ノードにおいても再帰的に同様の処理を繰り返していく。

決定木に幅優先探索を適用したときに走査する順にノードに 0 から順の番号を振る。すなわち、根ノードが 0 となり、その子ノードが左から 1 と 2 となる（図 3）。このとき、ノード  $i$  の学習（分割関数の選択）に用いる学習データを  $S_i = \{v\}$  とし、分割関数によって分割され、左と右の子ノードに割り当てられる学習データをそれぞれ  $S_i^L$  と  $S_i^R$  とする。このような表記を用いると、それぞれの分割ノード  $j$  において、 $S_j = S_j^L \cup S_j^R$ ,  $S_j^L \cap S_j^R = \emptyset$ ,  $S_j^L = S_{2j+1}$ ,  $S_j^R = S_{2j+2}$  が成立する。

ノードにおける分割関数  $h$ （1 式）の学習は  $h$  のパラメータ  $\theta$  の決定に他ならない。ノード  $j$  の分割関数の最適パラメータ  $\theta_j^*$  を

$$\theta_j^* = \arg \max_{\theta_j \in \mathcal{T}_j} I_j \quad (3)$$

で求める。 $I_j$  は目的関数であり、その一般的な定義は

$$I_j = I(S_j, S_j^L, S_j^R, \theta_j), \quad (4)$$

となる。

$I_j$  の具体的な定義はタスクによって異なってくるが、ここで共通して利用するのがエントロピーと情報利得である。以降ではその具体例をパターン識別を例にとって解説する。

今、集合  $S$  内のデータにクラスラベル  $c \in C$  が対応付いているとき、そのエントロピー（平均情報量）は

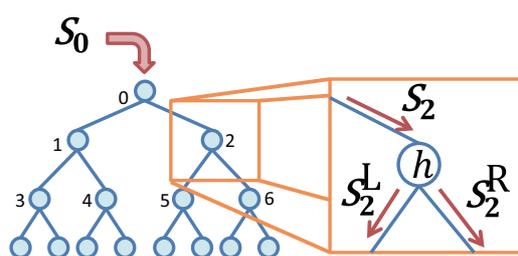


図 3 決定木ノードでの分割関数

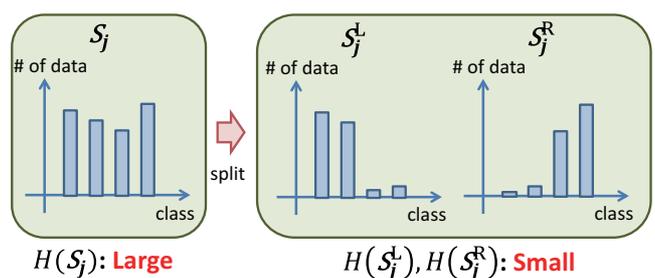


図 4 データ集合の分割によるエントロピーの変化

$$H(S) = - \sum_{c \in C} p(c) \log(p(c)), \quad (5)$$

である。\$S\$ 内に単一のクラスしか存在しないとき、\$H(S)\$ はゼロになり、全てのクラスが等確率で存在するときに \$H(S)\$ は最大になる。ノードでの分割によって得られる情報利得はエントロピーを用いて、

$$I = H(S) - \sum_{i \in 1,2} \frac{|S^i|}{|S|} H(S^i) \quad (6)$$

と書け、右辺第1項が分割前、第2項が分割後のエントロピーを示している。図4に示すように、\$I\$ が大きいということは、分割前後でデータ集合を比べたとき、分割後の方がクラス分布の偏りが大きいことを示す。

分割関数は異なるクラスを異なる子ノードに割り当てるような働きをすることが望ましいので、\$I\$ を3式の目的関数として学習を行う。

この結果求めた \$\theta\_j^\*\$ を用いて学習データ \$S\_j\$ を \$S\_j^L\$ と \$S\_j^R\$ に分割し、それぞれを子ノードの学習データとして再帰的に学習を続けていく。

### 2.2.3 学習の停止

以上の学習アルゴリズムはノードに割り当てられる学習データ集合の数が1になるまで継続することができるが、それでは過学習となってしまうため、適切な停止条件を導入することが必要である。

そのための方法としては、(1) あらかじめ設定した最大高さ \$D\$ に達した場合に停止する、(2) ノードに割り当てられた学習データの個数が一定値以下になった場合に停止する、(3) 分割による情報利得が一定値以下になった場合に停止する、などがあり、通常はそれらが同時に用いられている。

## 2.3 Random Forests の評価

以上の結果得られた Random Forests を用いた与えられたデータの評価を行う。この際の処理には一切ランダム性はないことに注意が必要である。

図5のように、入力データ \$\mathbf{v}\$ は \$T\$ 個の決定木すべてに並列に入力される。決定木ではノードに割り当てられた分割関数 \$h\$ に従って左右どちらの子ノードに進むべきかが決定され、最後には葉ノードに到達する。

葉ノードでは、あらかじめ対応付けられた予測値が得ら

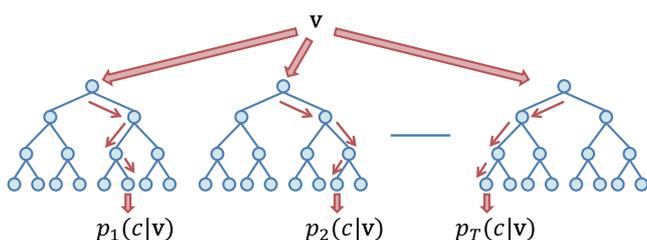


図5 複数の決定木から得られた予測結果の統合

れる。例えば、先述のようにクラスラベル \$c\$ の識別を行う際には、事後確率が得られる。ここで、\$t \in \{1, \dots, T\}\$ 番目の決定木によって得られた事後確率の予測値を \$p\_t(c|\mathbf{v})\$ とする。

これらを統合する際には、単純な相加平均をとって、

$$p(c|\mathbf{v}) = \frac{1}{T} \sum_{t=1}^T p_t(c|\mathbf{v}), \quad (7)$$

を得る場合が多いが、他には、すべてを掛け合わせた

$$p(c|\mathbf{v}) = \frac{1}{Z} \prod_{t=1}^T p_t(c|\mathbf{v}), \quad (8)$$

を用いることもある。この場合、\$Z\$ は \$p(c|\mathbf{v})\$ が確率として意味をもつようにする正規化係数である<sup>\*3</sup>。掛け合わせた場合は、分布のピークがより先鋭になる効果がある。

以上はクラス識別の際の例であったが、回帰などの場合も同様の考え方が適用できる。

## 3. 利用上の留意点

2節で述べた Random Forests の学習・評価の流れをみると、いくつかのパラメータなど、ユーザーが選択・決定しなければならない部分があることが分かる。具体的には、(1) 決定木の個数 \$T\$、(2) 決定木の最大高さ \$D\$、(3) ランダム性を制御するパラメータ \$\rho\$、(4) ノードでの分割関数、(5) 学習時の目的関数、(6) 適用先に応じた特徴選択、などがある。これらは、予測精度、汎化性能、計算効率など Random Forest の性能すべての面に大きく影響を及ぼす。

[8]によれば、\$T\$ を大きくすれば予測精度が向上するが、規模が大きくなるほど、学習・評価の効率が低下する。また、先述のとおり \$D\$ を大きくすると過学習が発生する。Breiman [4]は、ランダム性が決定木の間の相関関係に影響を与え、全体の性能を保持するにあたって非常に重要であることを示している。

以上、一般的な Random Forests の定義および大まかな処理の流れであった。以降ではパターン識別など個々の応用例を紹介し、そこで特に留意しなければならない事項について論じる。

## 4. Random Forests によるパターン識別

パターン識別は Random Forests がもっとも多く適用されている応用分野である。ここでは、与えられたデータ \$\mathbf{v}\$ に対して離散値のクラスラベル \$c \in \{c\_k\}\$ を対応付けることが目的となる。

Random Forests によるパターン識別の利点としてよく挙げられる事項としては、(1) (SVMなどと異なり)自然に多クラス識別に適用できる、(2) 確率的な予測出力を得ることができる、(3) 未知のデータに対する汎化能力が高

<sup>\*3</sup> それぞれの木は確率的に独立ではないため、この計算は厳密には正しくないことに注意が必要である

い、(4) 並列実行可能であり、データ毎の評価回数が少なくなるため効率が良い、があり、この中で (1) がパターン識別特有の利点である。

2 節で示した Random Forests の定義はパターン識別を念頭においていたため、分割関数 (1 式)、その最適化目的関数 (3 式)、情報利得 (6 式)、予測する事後確率 (7 式) をそのまま用いればパターン識別を実行できる。

#### 4.1 パラメータ選択の影響

Random Forests をパターン識別に用いるとき、3 節で述べた一般的な注意事項はそのまま当てはまる。

まず、決定木の個数  $T$  は先述のとおり、多ければ多いほど予測性能を向上させることができる。これを直感的に示すために図 6 に示すような二つのクラス  $c = 0, 1$  の識別問題を考える。これに対してクラスの期待値  $\sum_{c \in \{0, 1\}} c \cdot p(c|x)$  を求めるとすると、望ましいものは図 6(a) のように二つのクラスの間を滑らかに変化するものである。

これに対し、一つの決定木のノードが示す識別境界がクラス間に存在していたとすると、それに対応する期待値は図 6(b) のように階段状になる。先述のとおり、分割関数の最適化ではランダムに選んだ候補から最も良いものを選ぶため、異なる決定木では識別境界は様々な位置をとり  $T$  が増加するにつれて期待値も (a) のように滑らかなものになると期待される。

つぎに、木の高さ  $D$  についても先に述べたとおり、 $D$  が大きいほど過学習に陥る可能性が高くなるが、これについては、データ  $v$  の次元やその分布によって最適な値が決まってくるのである程度の試行錯誤が要求される。

ノードにおける分割関数の定義についても、複雑なものを用いればより多種のクラス境界面を表現できるようになるが、同時に評価にかかる計算コストも増大するためここでもトレードオフが存在している。

ランダム性については、あまりランダム性が高くなると

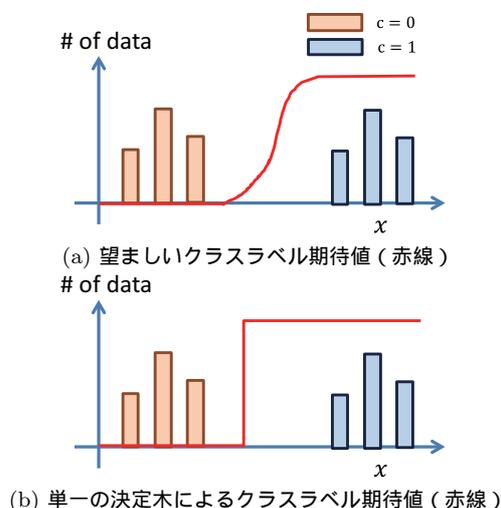


図 6 複数の決定木の統合による予測精度向上

決定木同士がまったく別のものとなってしまっていて、その結果を平均化したものの信頼性が低くなることも指摘されている。

#### 4.2 応用例

パターン識別の応用例として特筆すべきものは、Shotton らによる単一のデプスマップからの実時間人物姿勢推定 [23] であろう。これは、Microsoft の Kinect for Xbox 360 向けに開発された技術であり、ユーザが自分の身振り・手振りによってゲームなどの操作を行うことを可能にした。コンピュータビジョンの学会でも高く評価され、CVPR 2011 のベストペーパーを受賞している。

この手法は、Kinect センサーで得られた一枚のデプスマップを入力として、そこに写っている人物の頭部・手・足などの部分に分割・識別することを目的としている。これは、デプスマップ上の位置  $p \in \mathbb{R}^2$  を入力として、人体部位のクラスラベル  $c \in \{\text{lefthand, righthand, head, } \dots\}$  を予測するパターン認識問題に他ならない\*4。

[23] では、分割関数で用いる特徴量として、 $p$  とその周囲でもう一点を選んで、この二点でのデプスマップの値の差を特徴量としている。これは非常に単純なものであるが、多くの学習データを用いて学習を行うことで非常に良好な結果が得られている。

### 5. Random Forests による回帰

#### 5.1 パターン識別問題と回帰問題との比較

パターン識別問題と回帰問題の大きな違いは、出力が離散的なクラスラベルか連続的な予測値になるかという点である。数式で書けば、入力データ  $v$  に対して、変数  $y \in \mathcal{Y} \subseteq \mathbb{R}^n$  がとる確率密度分布  $p(y|v)$  を推定する問題になる。これによって、Random Forests による回帰 [1] のモデルはパターン識別の場合と若干異なることになる。以下、項目毎にパターン識別問題との比較を行っていく。

##### Random Forests 回帰の利点

回帰に用いた場合にも Random Forests 一般の利点はそのまま当てはまる。さらに、決定木を用いた予測は特徴空間を分割してそれぞれについて予測を行うことに他ならないため、非線形回帰問題を部分問題に分割してそれぞれを単純な形で解くことも可能となる。図 7 は、決定木の葉ノードに対応する部分空間において線形近似を行って非線形回帰を行っている様子を示している。

##### 決定木の出力

この分割した空間において、教師データから  $p(y|v)$  を推定しなければならない。最も簡単な例としては、確率分布の平均値を線形モデルで  $y(x) = \sum_{i=0}^n w_i x_i$  と

\*4 Shotton らは 31 個の部位に分割している。

表現し、その分散とあわせて保持するものである。

### 分割関数の設定

決定木のノードで評価される分割関数  $h$  はパターン識別のときと同じように定義できる。

### 分割関数の学習

学習時に分割関数のパラメータ  $\theta$  の決定は 3 式に従って行われるが、その目的関数  $I_j$  の定義はパターン識別の場合と異なり、以下ようになる。

$$I_j = \sum_{\mathbf{v} \in \mathcal{S}_j} \log(|\Lambda_{\mathbf{y}}|) - \sum_{1 \in \{L, R\}} \left( \sum_{\mathbf{v} \in \mathcal{S}_j^1} \log(|\Lambda_{\mathbf{y}}|) \right). \quad (9)$$

ここで、 $\Lambda_{\mathbf{y}}$  は条件付き共分散行列で、先に述べた線形モデルから導出できる。本式の導出については文献 [8] を参照されたい。

### 決定木出力の統合

このようにして学習された各決定木から得られた出力を統合する際には、7 式と同じように、その平均

$$p(\mathbf{y}|\mathbf{v}) = \frac{1}{T} \sum_t p_t(\mathbf{y}|\mathbf{v}) \quad (10)$$

を計算すれば良い。

## 5.2 パラメータ選択の影響

回帰問題に適用した場合のパラメータ選択の影響に関しても、パターン識別の場合とほぼ同じ議論が成立する。

例えば、決定木の数  $T$  は多ければ多いほど予測精度を向上できる。先に述べたように、回帰を行う場合は特徴空間を分割してそれぞれに確率密度分布を推定するため、単一の決定木では分割した空間の境界で予測値が不連続になる。複数の決定木から得られる結果を平均することで、その影響をキャンセルすることができ、全体的に妥当な結果

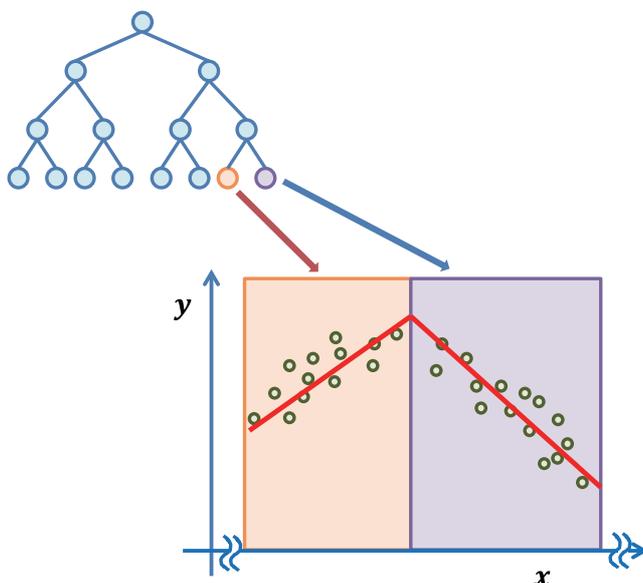


図 7 部分領域への分割による非線形回帰

を得ることが可能となる。

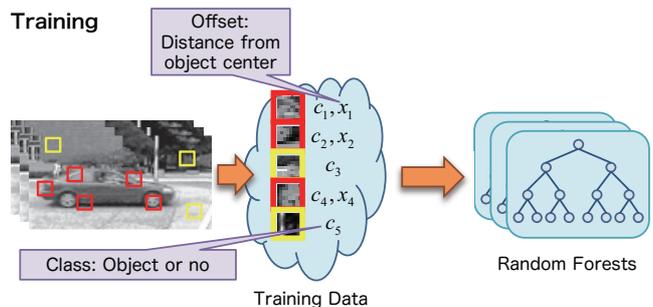
木の高さ  $D$  についても同様に、 $D$  を大きくすることは特徴空間を細かく分割していくことに相当する。これは、先に述べた不連続性が発生する可能性が増大することに他ならない。つまり、回帰問題の場合も過学習によって予測する確率分布が不連続になる。

## 5.3 応用例

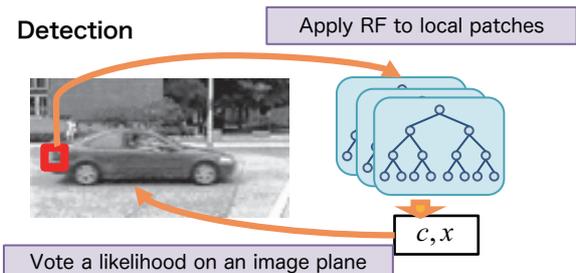
Random Forests を回帰問題に適用した例としては、画像中の物体検出 [11], [17] や 3 次元 CT 画像からの臓器の検出 [9] が挙げられる。いずれも、画像の画素や CT 画像のボクセルを入力として、そこから検出対象物体までの相対位置 (オフセット量) を出力としている。入力データの様々な場所で回帰による位置推定を行い、その結果を投票してスコアが高い位置を位置検出結果として出力している。

図 8 は物体検出の概要を示しており、(a) に示す学習フェーズでは、学習データに含まれる大量の局所パッチについて、そのクラスラベル (物体に属するか否か) とオフセット量 (物体中心までの距離) を与えて Random Forests を用いて学習する。(b) に示す検出フェーズでは、入力画像の局所パッチを Forests に入力して、予測された物体中心位置にクラスラベル推定値に基づくスコアを投票する。

このような事例に Random Forests を適用したときの一つの利点としては、学習結果の解釈が容易であることが挙げられる。学習の結果得られた決定木の葉ノードにおいて、確率分布に高いピークが存在していれば、そこに対応するデータからの物体位置推定の確度が高いことを意味してい



(a) クラスラベルとオフセットの学習



(b) Random Forests を用いた物体検出

図 8 Random Forests による回帰を用いた物体検出

る。これから、確度が高い葉ノードに対応付く学習データを見れば、検出物体を特徴づける情報が何かを知ることができる。例えば、臓器の場合はその周囲の骨格や、物体の場合はそれに特有な形状などを可視化することができる。

## 6. その他の適用例

Random Forests はパターン識別と回帰以外にも様々な応用に適用されている。以下では、[8] で紹介されているものについて簡単に述べる。

### 6.1 確率分布の推定

これまでと異なり、ラベル無しのデータ  $\{v\}$  が与えられているとする。それから  $\{v\}$  が従っている確率分布を推定する問題にも Random Forests を適用することができる。

まず、個々の決定木によってデータ集合をクラスタリングする。そのためには、ノードにおける分離関数を最適化する目的関数を教師無しのエントロピーを用いて定義する。詳細な定義は割愛するが、定性的には正規分布を仮定したときの当てはまり度合いが高くなるようにクラスタリングしていく。こうして得られた決定木の葉ノードには、教師データの部分集合が対応付けられているが、その分布を単一の正規分布で近似する。

新たなデータ  $v$  が与えられたときには、それぞれの決定木に入力して確率  $p_t(v)$  を得た上で、その平均  $p(v) = 1/T \sum_{t=1}^T p_t(v)$  を確率の推定値とする。

このように確率分布の推定を行う手法としては、ノンパラメトリックなものとしては Parzen 推定、パラメトリックなものとしては混合ガウス分布も知られているが、Random Forests を用いた方がデータへの適応性がよく、より自然な推定結果が得られることが示されている [8]。

### 6.2 多様体学習

多様体学習は、前項と同様にラベル無しのデータが与えられたときに、その分布をコンパクトに表現するより小さな次元の空間への写像を得る問題である。これも Random Forests で実現することができる。

そのためまず、前項の確率分布推定と同様の処理を行って学習データをクラスタリングする。その後、学習データ同士の類似度を表現する類似度行列 (affinity matrix) を得る。最後に、その類似度を保持しながら低次元の特徴空間に射影する写像を求める。

この中で、複雑なデータについて類似度行列を求める問題は一般に困難であるが、Random Forests を用いることで効率的に類似度行列を導出可能である。

### 6.3 半教師付き学習

ここまでは教師付き学習と教師なし学習の双方について述べてきたが、その中間である半教師付き学習も行うこと

ができる。パターン識別問題を考えるとすれば、少数のクラスラベル付きの学習データと、多数のクラスラベル無し学習データが得られたときに、後者に適切なクラスラベルを割り当てる問題に帰着される。

これは、ここまで述べてきた教師付き学習用の Random Forests と教師無し学習用の Random Forests を組み合わせて実現することができる。学習時に用いる目的関数は前者において利用される情報利得  $I_j^s$  と後者における  $I_j^u$  の重み付き和  $I_j = I_j^u + \alpha I_j^s$  を用いて定義する。これによって得られる決定木はラベル無し・有り双方のデータをクラスタリングする。ラベル無しデータについては最も近いラベル付きデータのクラスラベルを割り当てる。

### 6.4 Random Forests の改良・変形

Extremely Randomized Trees[13] はノードでの分割関数の学習を省略してランダムに選んだものを採用するものである。2 節で述べたモデルでは  $\rho = 1$  に相当する。つまり、ランダムに選んだ分割関数をそのまま使用するものである。一方、Random ferns[19] も Random Forests の一種であり、決定木の同じレベルのノードでは同じ分割関数を利用するものである。いずれも、Random Forests を簡略化したもので予測性能は低下するが、データ数が少ない場合でも過学習しにくいというメリットがある。

また、新たに得られたデータを用いてモデルを更新するオンライン学習を実現したもの [22] も提案されている。ここでは、決定木において新たなデータが到達した葉ノードにおいて、あらかじめ対応付けられていた確率分布を新データを用いて更新している。

## 7. 実装

Random Forests はすでに多くの分野で使用されているため、様々な実装が存在する。以下に簡単に紹介する。

Open CV[18] 広く利用されている OpenCV でも CvRTrees クラスとして実装されている。

Waffles[12] 機械学習手法のライブラリ Waffles の中でも Random Forests が実装されている。Linux, Windows, OSX で利用可能である。

randomforest-matlab[21] MATLAB 向けの実装である。

Leo Breiman による実装 [5] FORTRAN77 による著者らによる実装。

これらを用いても良いが、Random Forests は単純な構造であるので、自らの手で実装することもそれほど困難ではない。

## 8. おわりに

本稿では Random Forests をコンピュータビジョンへ応用する学生・初学者を対象として、その基本アイデアと

様々な適用事例・利用上の留意点などをまとめた．浅学な著者であり，また，紙面に限りもあるため，万が一漏れや誤りがあればご指摘をいただければ幸いです．

現在でも多くの研究事例が新たに公開されているので，それを踏まえ，本稿では足りない部分を補った最新版を <http://www.habe-lab.org/habe/RfTutorial/> で公開する予定である．

## 謝辞

本稿をまとめるにあたって，東芝欧州研究所ケンブリッジ研究所の牧淳人博士に貴重なご意見をいただいた．また，冒頭にあげたインターネット上の参考資料 [2], [6], [8], [20] がなければ本稿をまとめることはできなかった．紙面を借りて感謝申し上げる．なお，本稿をまとめるための検討には科研費 23700210 の補助を受けている．

## 参考文献

- [1] Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J.: *Classification and Regression Trees*, Statistics/Probability Series, Vol. 19, Wadsworth (1984).
- [2] Breiman, L.: <http://stat-www.berkeley.edu/users/breiman/RandomF>.
- [3] Breiman, L.: Bagging Predictors, *Machine Learning*, Vol. 24, No. 2, pp. 123–140 (1996).
- [4] Breiman, L.: Random Forests, *Machine Learning*, Vol. 45, No. 1, pp. 5–32 (2001).
- [5] Breiman, L. and Cutler, A.: <http://www.stat.berkeley.edu/~breiman/RandomForests/cc.software.htm>.
- [6] Computer Vision Tutorial: Random Forests とその応用 — 中部大学藤吉研究室 : <http://www.vision.cs.chubu.ac.jp/CVTutorial/>.
- [7] Cortes, C. and Vapnik, V.: Support-vector networks, *Machine Learning*, Vol. 20, No. 3, pp. 273–297 (1995).
- [8] Criminisi, A., Shotton, J. and Konukoglu, E.: Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning., Technical Report MSR-TR-2011-114, Microsoft Research (2011).
- [9] Criminisi, A., Shotton, J., Robertson, D. and Konukoglu, E.: Regression Forests for Efficient Anatomy Detection and Localization in CT Studies, *Medical Computer Vision Recognition Techniques and Applications in Medical Imaging*, Vol. 6533, pp. 106–117 (2011).
- [10] Freund, Y.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *Journal of Computer and System Sciences*, Vol. 55, No. 1, pp. 119–139 (1997).
- [11] Gall, J., Yao, A., Razavi, N. and Van Gool, L.: Hough Forests for Object Detection, Tracking, and Action Recognition, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 33, No. 11, pp. 2188–2202 (2011).
- [12] Gashler, M. S.: Waffles: A Machine Learning Toolkit, *Journal of Machine Learning Research*, Vol. MLOSS 12, pp. 2383–2387 (2011).
- [13] Geurts, P., Ernst, D. and Wehenkel, L.: Extremely randomized trees, *Machine Learning*, Vol. 63, No. 1, pp. 3–42 (2006).
- [14] Ho, T. K.: Random decision forests, *Document Analysis and Recognition 1995 Proceedings of the Third International Conference on* (Kavavaugh, M. and Storms, P., eds.), Proceedings of the 3rd International Conference on Document Analysis and Recognition, Vol. 1, IEEE, pp. 278–282 (1995).
- [15] Ho, T. K.: The random subspace method for constructing decision forests, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 8, pp. 832–844 (1998).
- [16] Lepetit, V., Laguerre, P. and Fua, P.: Randomized Trees for Real-Time Keypoint Recognition, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR05*, Vol. 2, pp. 775–781 (2005).
- [17] Okada, R.: Discriminative generalized hough transform for object detection, *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2000–2005 (2009).
- [18] OpenCV: <http://opencv.willowgarage.com/wiki/>.
- [19] Ozuysal, M., Calonder, M., Lepetit, V. and Fua, P.: Fast keypoint recognition using random ferns., *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 3, pp. 448–461 (2010).
- [20] Random Forests — Wikipedia: [http://en.wikipedia.org/wiki/Random\\_forest](http://en.wikipedia.org/wiki/Random_forest).
- [21] Randomforest-matlab: <http://code.google.com/p/randomforest-matlab/>.
- [22] Saffari, A., Leistner, C., Santner, J., Godec, M. and Bischof, H.: On-line Random Forests, *2009 IEEE 12th International Conference on Computer Vision Workshops ICCV Workshops*, Vol. 32, No. 2, pp. 1393–1400 (2009).
- [23] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A. and Blake, A.: Real-Time Human Pose Recognition in Parts from Single Depth Images, *CVPR 2011*, Vol. 2, IEEE, pp. 1297–1304 (2011).
- [24] Shotton, J., Johnson, M. and Cipolla, R.: Semantic texton forests for image categorization and segmentation, *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 1–8 (2008).