Vol.2017-NL-231 No.6
Vol.2017-SLP-116 No.6
2017/5/15

深層ディシジョンフォレストによる疎密混合データの解析

大北 剛[1,a)]　井上 創造[1,b)]

概要：ランダムフォレストは, 弱学習器によりアンサンブル学習を実現するが, この機構を用いて目的変数を設定すれば説明変数の重要度を求めるタイプの学習に用いることができる. 深層ディシジョンフォレストは, 決定木におけるユニークなルーチングをベルヌーイ確率変数により確率的に表現し, 微分可能とする. このことにより, 決定木のディープ化においてよく見られる局所的でグリーディな最適化ではなく, グローバルな最適化を行なう版であり, 精度が上昇したことが報告されている. 実験では, 介護施設におけるインシデント報告というマルチモーダルなデータを用いて, なぜ事故が起こったのかという原因究明の解析を深層ディシジョンフォレストとランダムフォレストを比較する.

1. イントロダクション

ランダムフォレスト [1] は, 弱学習器によりアンサンブル学習を実現するが, この機構を用いて目的変数を設定すれば説明変数の重要度を求めるタイプの学習に用いることができる. 深層ディシジョンフォレスト [5] は, 決定木におけるユニークなルーチングをベルヌーイ確率変数により確率的に表現し, 微分可能とする. このことにより, 決定木のディープ化においてよく見られる局所的でグリーディな最適化ではなく, グローバルな最適化を行なう版であり, 精度が上昇したことが報告されている. 実験では, 介護施設におけるインシデント報告という疎密の混合するデータを用いて, なぜ事故が起こったのかという原因究明の解析を深層ディシジョンフォレストとランダムフォレストを比較する.

2. 深層ディシジョンフォレスト

Kontschieder[5] にしたがい, 深層ディシジョンフォレストを記述する. 入力を $\mathcal{X}$, 出力を $\mathcal{Y}$ とする分類問題を考える. 決定木は, 分割ノード（内部ノード）と推定ノード（葉ノード）よりなる木構造の分類器である. 分割ノードと推定ノードをそれぞれ $\mathcal{N}$, $\mathcal{L}$ によりインデックスづけする. 推定ノード $\ell \in \mathcal{L}$ は $\mathcal{Y}$ 上の確率分布 π_ℓ をもつ. 分割ノード $n \in \mathcal{N}$ は θ によりパラメータ化された決定関数 $d_n(\cdot;\theta) : \mathcal{X} \to [0,1]$ を割り当てられる.

決定木のノードを θ によりパラメータ化したとする. このとき, 木 T からのサンプル x に対する確率分布の推定は

$$P_T[y|x,\theta,x] = \sum_{\ell\in\mathcal{L}} \pi_{\ell_y}\mu_\ell(x|\theta) \tag{1}$$

と表現される. なお, $\pi = (\pi_\ell)_{\ell\in\mathcal{L}}$ とし, π_{ℓ_y} をサンプルが葉ノード ℓ に到達してクラス y となる確率とする. また, $\mu_\ell(x|\theta)$ をルーチング関数と呼び,

$$\mu_\ell(x|\theta) = \prod_{n\in\mathcal{N}} d_n(x;\theta)^{\mathbf{1}_{\ell\leftarrow n}}\overline{d}_n(x;\theta)^{\mathbf{1}_{n\to\ell}} \tag{2}$$

と表現されるとする. なお, $\overline{d}_n(x;\theta) = 1 - d_n(x;\theta)$ とし, $\mathbf{1}$ を指示関数とする.

決定ノードにおいて確率的ルーチングを行なうことが決定木と異なる点である. σ をシグモイド関数とし, $f_n(\cdot;\theta)$ を $\mathcal{X}$, θ を変数とする関数とする. すると, 確率的ルーチング $d_n(x;\theta)$ を

$$d_n(x;\theta) = \sigma(f_n(x;\theta)) \tag{3}$$

と表現できる.

さらに, フォレストを k 本の決定木のアンサンブル $\mathcal{F}(= \{T_1,\ldots,T_k\})$ とする. すると, 入力 x に対する確率分布の推定は, $\mathcal{F}$ における各々の木の出力を平均化することで得られる. つまり,

$$P_F[y|x] = \frac{1}{k}\sum_{h=1}^{k} P_{T_h}[y|x] \tag{4}$$

と表現される.

データ集合 $T(\subset \mathcal{X}\times\mathcal{Y})$ に対して, ログ損失関数 $L(\theta,\pi;x,y)$ を用いて,

$$R(\theta,\pi;T) = \frac{1}{|T|}\sum_{(x,y)\in T} L(\theta,\pi;x,y) \tag{5}$$

という経験損失を最小化する. なお, ログ損失関数

[1] 九州工業大学
〒 804-8580, 北九州市戸畑区仙水町 1-1
[a)] tsuyoshi.okita@gmail.com
[b)] sozo@mns.kyutech.ac.jp

$L(\theta, \pi; x, y)$ をトレーニング集合 $(x, y) \in T$ に対するログ損失項としており,

$$L(\theta, \pi; x, y) = -\log(P_T[y|x, \theta, \pi]) \tag{6}$$

として定義される. この最適化は, 決定ノードの学習と葉ノードの学習という二つの要素を学習するものとなる. 各エポックにおいて, 決定ノードの学習 (ミニバッチ分) と葉ノードの学習を交互に両方行なう.

決定ノードの学習は以下の通りである. 確率的勾配降下法を用いて θ に対して経験損失最小化法を適用する. すると,

$$\theta^{(t+1)} = \theta^{(t)} - \eta \frac{\partial R}{\partial \theta}(\theta^{(t),\pi;B}) \tag{7}$$

となる. θ に関する損失関数 L の勾配は以下のように表現され,

$$\frac{\partial L}{\partial \theta}(\theta, \pi; x, y) = \sum_{n \in N} \frac{\partial L(\theta, \pi; x, y)}{\partial f_n(x; \theta)} \frac{\partial f_n(x; \theta)}{\partial \theta} \tag{8}$$

決定木に依存する f_n に関する損失関数 L の勾配は

$$\frac{\partial L(\theta, \pi; x, y)}{\partial f_n(x; \theta)} = d_n(x; \theta) A_{n_r} - \overline{d}_n(x; \theta) A_{n_\ell} \tag{9}$$

と表現される. なお, A_m を一般的なノード $m \in \mathcal{N}$ に対して

$$A_m = \frac{\sum_{\ell \in L_m} \pi_{\ell_y} \mu_\ell(x|\theta)}{P_T[y|x, \theta, \pi]} \tag{10}$$

と定義されるとする. なお, 中間層のない版を用いている.

次に葉ノードの学習は式 5 を θ を固定して, π に関して最小値を求める. これも Kontschieder[5] にしたがっている. つまり,

$$\min_{\pi} R(\theta, \pi; T) \tag{11}$$

となる. これは

$$\pi_{\ell_y}^{(t+1)} = \frac{1}{Z_l^{(t)}} \sum_{(x,y') \in T} \frac{\mathbf{1}_{y=y'} \pi_{\ell_y}^{(t)} \mu_\ell(x|\theta)}{P_T[y|x, \theta, \pi]} \tag{12}$$

を逐次的に解くこととなる. なお, Kontschieder[5] が, この値が収束することを証明している.

以上は, 一つの深層決定木の学習であったが, これを k 木のフォレスト F にする. これも Kontschieder[5] にしたがう. k 木のフォレストは θ を共有することができるが, 異なる決定関数と構造, 葉ノードにおける確率分布の推定 π が独立でなければならない. このためには π に問題はなく (フォレスト F が独立した π をもっていれば, エポックが進んでも確率分布の推定 π は独立のままである), θ は各ミニバッチにおいてランダムに選択すればよい.

3. 実装と重要度の計算

実装に際しても Kontschieder[5] にしたがったが, 重要度に関する記述はなかったため, これに関しては補った.

決定木の決定関数 d_n を $f_n(x; \theta_n) = \theta_n^T x$ として k 本の木を並列に並べた層を決定木の層とした. したがって, 全体のネットワーク構造は, 入力層, ReLU を活性化関数とする全結合層, シグモイドを活性化関数とする全結合層, そして, 決定木の層からなる.

ランダムフォレストにおける重要度は, フォレストのそれぞれの木において, 特定の変数に対して, その項目の中身をシャッフルした場合にそれらによりどれだけ推定値が悪くなったかを測定した値である [1]. この値が大きく悪くなれば重要度が高いと考える. それぞれの変数ごとにフォレスト内のすべての木上でこの重要度を平均する. 深層ディシジョンフォレストにおいても同様のことを行なったが, 特定の変数は決定木のノードに縛られていないため, まず, 特定の変数に関してシャッフルしたテスト集合を用意し, 単純に k 個の深層ディシジョンフォレスト内の各々の木に対して, クラスごとに精度がどう落ちるかを測定した。つまり, シャッフルしたテスト集合に対して

$$A_\ell = \frac{\pi_{\ell_y} \mu_\ell(x|\theta)}{P_T[y|x, \theta, \pi]} \tag{13}$$

という量がどう変化するかをクラス毎に測定し, シャッフルしないテスト集合に対して得られる値との差の絶対値をクラス上で平均したものを重要度と考えた. なお, シャッフルしたテスト集合はそれぞれのクラスに対し N 種類用意し, それらに対して上記を測定した. その上で平均値を取り, これとシャッフルしないテスト集合に対して得られる値との差の絶対値を考えた. 以下の実験においては $N = 10$ を用いた.

4. 実験結果

4.1 データ

実験においては, 介護施設におけるインシデントデータを用いた [6]. 項目は, 発生時間, 年齢, 介護度, 認知度, 利用区分, 事故歴, 事故の種類, 発生場所, 目的, 詳細, 重症度, 診断よりなる. 詳細のみが自然言語の自由形式の記述で行なわれる. その他は離散値による記述もしくはいくつかのカテゴリに属する名詞として記述される.

「詳細」項目における日本語による記述は, 1 件は複数行を含むことがあり, 件数の数は 5, 190 件となり, 265, 397 語を含む. また, 数字, 漢字, アルファベット, 許容範囲の記号 (. , 「」など) 以外の文字をチェックした. 半角コードと全角コードが混在し, 一方, 表 1 に示すような記号が存在することがわかった. このためこれらは全角コードに統一して正規化した. なお, これら以外にも表記の揺れは存在したが, これらに関しての正規化は行なっていない.

データをトレーニング集合 4,106 タプル, ディベロップメント集合 456 タプル, テスト集合 507 タプルに固定したやり方で分割した. データ 5 は利用区分を目的変数として, その他を説明変数とした. 同様にデータ 7, データ 10, データ 11 は事故の種類, 重症度, 診断をそれぞれ目的変数とし,

その他を説明変数とした. 分類におけるクラスの数は表 2 に示した通りである.

4.2 アルゴリズムに関する詳細

実験 1 はデータの大きさが $m = 11$ で, すべての項目を離散値の ID に変換した. 実験 2 はデータの大きさが $m = 12$ で, 上記の実験 1 で用いたデータに自然言語による表現を追加した.

ランダムフォレストは sklearn[*1]を用いた. 一方, 深層ディシジョンフォレストは Lasagne[*2]/Theano[8][*3]ベースの実装[*4]をベースとして用いたが変更を加えている.

ランダムフォレストにおいては, トレーニング集合を用いてクラス分類器を構築し, テスト集合に対して性能（精度と OOB スコア）を測定した. つまり, ランダムフォレストにおいては, ディベロップメント集合を用いたパラメータなどのチューニングは行なっていないが, 深層ディシジョンフォレストと条件を同じにするため, ディベロップメント集合を設定した. したがって, トレーニング集合の大きさを若干犠牲にしている.

一方, 深層ディシジョンフォレストにおいては, ディベロップメント集合を用いて, ハイパーパラメータのチューニングを行なった. チューニングは以下のパラメータを用いたベイジアン最適化 [7] による. 関係するパラメータは以下の通りであり, 各々の選択は経験により行なった. したがって, 総当りでチューニングを行なったわけではない.

depth	$[2, 3, 4, 5]$
n__estimators	$[10, 20, 30, 40]$
n__hidden	$[64, 128, 256]$
learning__rate	$[0.1, 0.01, 0.001, 0.0001]$
num__epochs	$[500, 2000]$
batch__size	$[100, 200]$
momentum	$[0.0, 0.8]$
dropout	$[0.0, 0.2, 0.25, 0.3, 0.4, 0.8]$
update	$[adagrad, sgd, adam, RMSprop]$

表 3 ハイパーパラメータのチューニングで用いた探索範囲

ディベロップメント集合に対して高い精度をあげたモデルに対し, 固定テスト集合に対して, 精度を測定した. 損失関数はクロスエントロピーを用いた.

形態素解析に関しては JUMAN を用いた [3].

4.3 実行環境

両者ともインテルの CPU i7-7920HQ 上で実行し, さらに深層ディシジョンフォレストに関しての実行は NVIDIA GTX1080 上で行なった.

4.4 実験 1 の詳細

上述したように実験１は離散値のみのデータからなる. 実験１の実験結果のテスト集合に対する実験結果は以下の通りである.

	ランダムフォレスト		深層ディシジョンフォレスト	
	テスト	ディベロップ	テスト	ディベロップ
データ 5	0.619	0.678	0.635	0.560
データ 7	0.688	0.735	0.692	0.699
データ 10	0.714	0.789	0.716	0.663
データ 11	0.481	0.548	0.487	0.458

表 4 テスト集合/ディベロップメント集合に対する実験結果 (精度) を示す.

いずれの項目においても, 深層ディシジョンフォレストはランダムフォレストの精度をそれぞれ 2.6%, 0.6%, 0.3%, 1.2%上回った. 興味深いことは深層ディシジョンフォレストはディベロップメント集合に対し, あまり良い性能は示していないが, テスト集合では良い性能を示している. これは汎化性能が良いことを示していると思われる. なお, 前述したがランダムフォレストはディベロップメント集合をト

シンボル	正規化後の表現
リットル記号 (半角)	リットル
℃	度
『	「
』	」
…	(消去)
cm (半角と全角)/センチ	センチ
mm (半角と全角)/ミリ	ミリ
右手大 (囲み文字４) 指の痛み	右手大四指の痛み
皮膚剥離を（囲み文字１）発見する	皮膚剥離を発見する
※本人様による事後報告	本人様による事後報告
２Ｘ２の表皮剥離	２ｘ２の表皮剥離
第 II 指の爪	第二指の爪
ベット⇔トイレ間	ベットとトイレ間
て転倒⇒外傷なし	て転倒したが外傷なし
１リットル→１．２５リットル	１リットルから１．２５リットル
１１：４５→下痢が続いており	１１：４５から下痢が続いており
ＳＰＯ２/SpO2	ＳＰＯ²
Dr	医師
NS, Ns	看護師
NC, Nc	ナースコール

表 1 インシデントデータにおける記号データに関する正規化.

データ 5	15 クラス
データ 7	16 クラス
データ 10	4 クラス
データ 11	15 クラス

表 2 四種の分類タスクにおける目的変数のクラスの数

[*1] http://scikit-learn.org
[*2] https://lasagne.readthedocs.io/en/latest/
[*3] http://deeplearning.net/software/theano/
[*4] https://github.com/SkidanovAlex/ShallowNeuralDecisionForest

レーニングにおいて使っていない.

チューニングにおける最適化アルゴリズムの選択を眺めると, データ 5, データ 7, データ 11 は adagrad [2] の性能が良く, データ 10 は adam [4] の性能が良かった. また, データ 10 においては誤差サーフェイスが起伏に富んでいるため, 学習率の小さいものが良い性能を上げた.

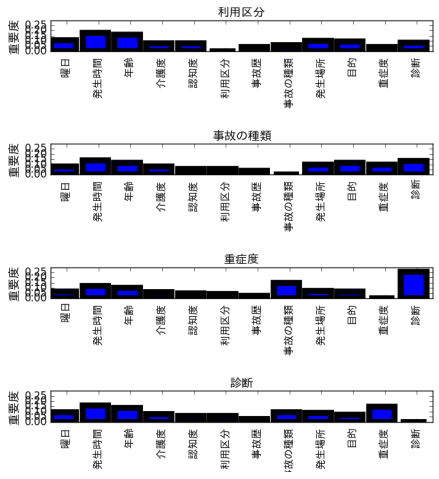

図 1 リレーショナルデータのみに対するランダムフォレストによる重要度のプロット

4.5 実験 2 の詳細

実験 2 の実験結果のテスト集合に対する実験結果は以下の通りである. 自由文を形態素解析 [3] を行ない, ストップワードを除くと, 7,745 項目を得た. 各々の項目に対し, 頻度を素性としたものを, 実験 1 と同様にトレーニング集合, ディベロップメント集合, テスト集合に分割した. これを元に, 同様にランダムフォレスト, 深層ディシジョンフォレストで分類タスクを解かせた.

ランダムフォレストは, 表 4 と表 5 を比較して明らかなように, データ 5, 7, 10, 11 のいずれにおいても混在データの方が 10%, 3%, 2%, 22%精度が良い. 自然言語という記述を 7,745 項目という疎な形のデータが密な離散データの精度を補っている. 同様の傾向が深層ディシジョンフォレストでも見られる.

	ランダムフォレスト		深層ディシジョンフォレスト	
	テスト	ディベロップ	テスト	ディベロップ
データ 5	0.690	0.662	0.694	0.660
データ 7	0.736	0.728	0.742	0.722
データ 10	0.715	0.704	0.730	0.707
データ 11	0.570	0.561	0.581	0.570

表 5 マルチモーダルデータに対するテスト集合/ディベロップメント集合に対する実験結果 (精度) を示す.

5. 結論

本論文においては, 分類タスクにおけるランダムフォレストを深層化した深層ディシジョンフォレストをランダムフォレストと比較した. 最初の実験においては離散値のみからなるデータを用いた. すべての項目においてわずかながら深層ディシジョンフォレストの精度がランダムフォレストを上回った. 二つ目の実験においては自然言語の記述からストップワードを除いた語をという疎なデータを最初の実験の密な離散値のデータに加えた. ここにおいても同様にすべての項目においてわずかながら深層ディシジョンフォレストの精度がランダムフォレストを上回った.

参考文献

[1] L. Breiman, Random forests, Machine Learning, 45:5-32, 2001.
[2] John Duchi, Elad Hazan, Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Journal of Machine Learning Research 12, page 2121-2159, 2011.
[3] 河原大輔, 黒橋禎夫: 自動構築した大規模格フレームに基づく構文・格解析の統合的確率モデル, 自然言語処理, Vol.14, No.4, pp.67-81, (2007.7).
[4] Diederik P. Kingma, Jimmy Ba, Adam: A Method for Stochastic Optimization, the 3rd International Conference for Learning Representations, San Diego, 2015
[5] Peter Kontschieder, Madalina Fiterau, Antonio Criminisi, Samuel Rota Bulo, Deep Neural Decision Forests, The IEEE International Conference on Computer Vision (ICCV), pp. 1467-1475, 2015.
[6] 峯崎 智裕, 井上 創造, ”介護サービス向上に向けた介護事故事例テキストの分析”, マルチメディア, 分散, 協調とモバイル (DICOMO2016) シンポジウム, pp. 1663-1669, 2016.
[7] Jasper Snoek, Hugo Larochelle and Ryan Prescott Adams. Practical Bayesian Optimization of Machine Learning Algorithms. Neural Information Processing Systems, 2012.
[8] Theano Development Team, ”Theano: A Python framework for fast computation of mathematical expressions”, arXiv e-prints, abs/1605.02688, 2016.