説明可能な機械学習のための拡張シャープレイ値の厳密指数時間アルゴリズム

大泉　翼† 有村　博紀††

† 北海道大学工学部　〒 060-0814 札幌市北区北 14 条西 9 丁目
†† 北海道大学大学院情報科学研究院　〒 060-0814 札幌市北区北 14 条西 9 丁目
E-mail: †monkukui@eis.hokudai.ac.jp, ††arim@ist.hokudai.ac.jp

あらまし　シャープレイ値は，経済学の投票指数として提案されたが，この数年，機会学習モデルにおける単一の変数の重要性尺度として，深層ニューラルネット等の非線形予測モデルを対象に，シャープレイ値の計算方法が集中的に研究され始めている．本論文では，シャープレイ値を複数の変数の組合せに拡張した「拡張シャープレイ値」を新たに定義し，与えられた予測モデルと入力データから，すべての変数の組合せに対する拡張シャープレイ値の値を一度に計算する問題を考察する．この問題に高速ゼータ変換を適用することで，素朴な $\mathcal{O}^*(3^n)$ 時間アルゴリズムを高速化して，$\mathcal{O}^*(2^n)$ 時間アルゴリズムを与える．ここに，n は予測モデルの変数の数である．

キーワード　機械学習，シャープレイ値，解釈可能性，動的計画法，高速ゼータ変換

1 はじめに

1.1 背景と目的

近年では機械学習手法により学習された予測モデルの解釈可能性（*interpretability*）の研究に注目が集まりつつある．一般に，機械学習モデルの性能は予測精度で評価され，誤差が小さいほど良いとされる．一方で，機械学習モデルの予測を意思決定に用いるとき，その予測根拠の提示が必要とされる場合がある．機械学習モデルの予測根拠を正しく提示することができれば，応用上での信頼を獲得することや，モデルの改善方法を発見する手助けになること，モデル化されるまでの過程を理解するのを助けることにつながる．これまでには，機械学習モデルの解釈可能性に関する研究が数多くなされている [1] [2] [3] [4]．本研究では，特定の予測結果に対する，単一変数の重要性尺度を表すシャープレイ値に焦点を当てる．

シャープレイ値（*Shapley value*）[5] [6] とは，ゲーム理論において，プレーヤーの貢献度を表す指標であり，1953 年に *Lloyd S. Shapley* によって提案された．近年，深層学習等の機械学習モデルの予測結果に対する単一の変数の重要性尺度として，シャープレイ値が広く注目されている [2]．単一の変数だけでなく，複数の変数組に対する重要度を計算することができれば，学習モデルの予測根拠により妥当な解釈を与えることができると考えた．

本研究では，複数の変数組に対する拡張シャープレイ値を定義し，すべての変数組に対する拡張シャープレイ値を計算する問題を考察する．さらに，その問題を解く効率良いアルゴリズムを与える．

1.2 主結果

本研究の主結果を以下に示す．

- 変数集合 $P = \{1, \ldots, n\}$ と，集合関数 $v : 2^P \to \mathbb{R}$，変数組 $\Delta \subseteq P$ に対して，拡張シャープレイ値 $\phi_\Delta(P, v)$ を定義した．
- すべての変数組 $\Delta \subseteq P$ に対して，拡張シャープレイ値 $\phi_\Delta(P, v)$ を計算する問題 *ESV* を定義した．
- *ESV* を効率よく解くアルゴリズムを与えた．

拡張シャープレイ値 $\phi_\Delta(P, v)$ を 1 つ計算するのは $\mathcal{O}^*(n^2)$ 時間かかる．ナイーブなアルゴリズムでは，$\mathcal{O}(2^n)$ 個ある変数組 Δ それぞれに対して拡張シャープレイ値を計算するので $\mathcal{O}^*(4^n)$ 時間を要する．これを改善するために，厳密指数時間アルゴリズム分野で知られている高速ゼータ変換 [7] を適用することで，時間計算量 $\mathcal{O}^*(2^n)$ と空間計算量 $\mathcal{O}^*(2^n)$ でこの問題を解くことができた．ここに，$\mathcal{O}^*(f(n))$ とは，多項式因子を無視した表記法である．より正確には，関数 f と g に対して，$f(n) = \mathcal{O}(g(n)\mathrm{Poly}(n))$ であるとき，$f(n) = \mathcal{O}^*(g(n))$ と表される．ここに，$\mathrm{Poly}(n)$ は n に関する多項式である．

2 準　　備

2.1 特性関数形ゲーム

協力ゲームにはいくつかの表現方法が存在するが，ここでは特性関数形ゲームの定義を与える．

非負整数 n に対して，プレイヤー集合を $P = \{1, \ldots, n\}$ とする．プレイヤー集合 P の各要素は**プレイヤー**（*player*）と呼び，部分集合 $S \subseteq P$ を**提携**（*coalition*），集合関数 $v : 2^P \to \mathbb{R}$ を**特性関数**（*characteristic function*）と呼ぶ．それぞれの $S \subseteq P$ に対して $v(S)$ は，提携 S に属するプレイヤーたちが協力して行動した際に S 全体が得ることができる利得を表す．よって以降では，$v(S)$ を S の利得と呼ぶ．ここで，特性関数形ゲームの定義を述べる．

定義 1. 特性関数形ゲームとは，プレイヤー集合 $P = \{1, \ldots, n\}$ と特性関数 $v : 2^P \to \mathbb{R}$ の組 (P, v) である．ここに，n は非負

表 1　$P=\{1,2,3\}$ におけるプレイヤー 1 の限界貢献度

全体提携の形成過程	プレイヤー 1
$1\to 2\to 3$	0
$1\to 3\to 2$	0
$2\to 1\to 3$	$v(\{1,2\})-v(\{2\})$
$2\to 3\to 1$	$v(\{1,2,3\})-v(\{2,3\})$
$3\to 1\to 2$	$v(\{1,3\})-v(\{3\})$
$3\to 2\to 1$	$v(\{1,2,3\})-v(\{2,3\})$

表 2　$P=\{1,2,3\}$ におけるプレイヤー組 $\{1,2\}$ の限界貢献度の例

全体提携の形成過程	プレイヤー組 $\{1,2\}$
$1\to 2\to 3$	0
$1\to 3\to 2$	0
$2\to 1\to 3$	0
$2\to 3\to 1$	0
$3\to 1\to 2$	$v(\{1,2,3\})-v(\{3\})$
$3\to 2\to 1$	$v(\{1,2,3\})-v(\{3\})$

整数である．

2.2　シャープレイ値

特性関数形ゲーム (P,v) において，P の利得 $v(P)$ を，各プレイヤー間でどのように分配するかについて考える．提携に対するプレイヤーの貢献度に応じて利得を分配する方法が，1953 年に *Lloyd S. Shapley* によって定義された．

任意のプレイヤー $i\in P$ と任意の提携 $S\subseteq P\setminus\{i\}$ に対して，$v(S\cup\{i\})-v(S)$ を S に対するプレイヤー i の限界貢献度と呼ぶ．これは，プレイヤー i が提携 S に加わることによって生じる利得の増分と解釈できる．それぞれのプレイヤーが順番に加わっていく $|P|!$ 通りの提携形成が等確率で起こることを仮定する．このとき，プレイヤー $i\in P$ の限界貢献度の期待値 $\mathbb{E}[v(S\cup\{i\})-v(S)]$ が，シャープレイ値 $\phi_i(P,v)$ であり，次のように定義する．

定義 2. プレイヤー集合を $P=\{1,\dots,n\}$ とし，v を集合関数 $v:2^P\to\mathbb{R}$ とする．このとき，特性関数形ゲーム (P,v) におけるプレイヤー $i\in P$ のシャープレイ値 $\phi_i(P,v)$ とは，

$$\phi_i(P,v):=\sum_{S\subseteq P\setminus\{i\}}\frac{|S|!(|P|-|S|-1)!}{|P|!}\left(v(S\cup\{i\})-v(S)\right) \tag{1}$$

である．

定義 3. すべてのプレイヤー $i\in P$ に対する，シャープレイ値 ϕ_iP,v を並べた n 個の要素からなるベクトル $\phi(v,P)=(\phi_i(v,P))_{i\in P}$ を，特性関数形ゲーム (P,v) におけるシャープレイ値という．

とりうる $|P|!$ 通りの順列すべてを考えたとき，提携 S に対するプレイヤー i の貢献度 $v(S\cup\{i\})-v(S)$ は $|S|!(|P|-|S|-1)!$ 回出現する．すなわち，定義 2 は，$P\setminus\{i\}$ の部分集合すべてに対して，$|S|!(|P|-|S|-1)!(v(S\cup\{i\})-v(S))$ を足し合わせて $|P|!$ で割ったものである．

$P=\{1,2,3\}$ のときのプレイヤー 1 の限界貢献度の例を表 1 に示す．

3　拡張シャープレイ値と全部分集合シャープレイ値問題

本節では，単一のプレイヤー $i\in P$ に対するシャープレイ値 $\phi_i(P,v)$ を拡張し，複数のプレイヤー組 $\Delta\subseteq P$ に対する拡張シャープレイ値 $\phi_\Delta(P,v)$ を定義する．

さらに，すべての $\Delta\subseteq P$ に対して，拡張シャープレイ値 $\phi_\Delta(P,v)$ を計算する問題（*ESV*）を定義する．

3.1　拡張シャープレイ値

非負整数 n に対して，$P=\{1,\dots,n\}$ をプレイヤー集合とし，$v:2^P\to\mathbb{R}$ を提携 $S\subseteq P$ の利得を表す集合関数とする．プレイヤー $i\in P$ に対するシャープレイ値 $\phi_i(P,v)$ を拡張し，複数のプレイヤーの組合せ $\Delta\subseteq P$ に対する**拡張シャープレイ値**（*extended Shapley value*）$\phi_\Delta(P,v)$ を以下のように導入する．

任意のプレイヤー組 $\Delta\subseteq P$ と任意の提携 $S\subseteq P\setminus\Delta$ に対して，$v(S\cup\Delta)-v(S)$ を S に対するプレイヤー組 Δ の限界貢献度と呼ぶ．これは，複数のプレイヤーからなるプレイヤー組 Δ が提携 S に同時に加わることによって生じる利得の増分と解釈できる．それぞれのプレイヤーが順番に加わっていく $|P|!$ 通りの提携形成が等確率で起こることを仮定する．このとき，プレイヤー組 $\Delta\subseteq P$ の限界貢献度の期待値 $\mathbb{E}[v(S\cup\Delta)-v(S)]$ が，拡張シャープレイ値 $\phi_\Delta(P,v)$ であり，次のように定義する．

定義 4. プレイヤー集合を $P=\{1,\dots,n\}$ とし，v を集合関数 $v:\to\mathbb{R}$ とする．このとき，特性関数形ゲーム (P,v) におけるプレイヤー組 $\Delta\subseteq P$ の拡張シャープレイ値 $\phi_\Delta(P,v)$ とは，

$$\phi_\Delta(P,v):=\sum_{S\subseteq P\setminus\Delta}\frac{|S|!|\Delta|!\left(|P|-|S|-|\Delta|\right)!}{|P|!}\left(v\left(S\cup\Delta\right)-v\left(S\right)\right) \tag{2}$$

と定義する．

定義 5. すべてのプレイヤー組 $\Delta\subseteq P$ に対する，拡張シャープレイ値 $\phi_\Delta(P,v)$ を並べた 2^n 個の要素からなるベクトル $\Phi(v,P)=(\phi_\Delta(P,v))_{\Delta\subseteq P}$ を，特性関数形ゲーム (P,v) における拡張シャープレイ値と呼ぶ．

とりうる $|P|!$ 通りの順列すべてを考えたとき，提携 S に対するプレイヤー組 Δ の限界貢献度 $v(S\cup\Delta)-v(S)$ は $|S|!|\Delta|!(|P|-|S|-|\Delta|)!$ 回出現する．すなわち，定義 5 は，$P\setminus\Delta$ の部分集合すべてに対して，$|S|!|\Delta|!(|P|-|S|-|\Delta|)!(v(S\cup\Delta)-v(S))$ を足し合わせて $|P|!$ で割ったものである．$P=\{1,2,3\}$ のときのプレイヤー組 $\{1,2\}$ の限界貢献度の例を，表 2 に示す．

3.2　全部分集合拡張シャープレイ値問題

本論文で考察する問題を述べる．

定義 6. 全部分集合拡張シャープレイ値問題（*all subset extended Shapley value problem, ESV*）とは，全体集合 $P=\{1,\dots,n\}$ と

Algorithm 1 *ESV* を解くナイーブアルゴリズム

Require: 変数集合 $P = \{1, \ldots, n\}$，集合関数 v.
Ensure: 拡張シャープレイ値 $\Phi(v, P)$

```
1: init_factorial(); // 階乗テーブルを構築する
2: for Δ ⊆ P do
3:     E ← 0;
4:     for S ⊆ P \ Δ do
5:         E ← E + (|S|!|Δ|!(|P|−|S|−|Δ|)!)/|P|! (v(S ∪ Δ) − v(S));
6:     end for
7:     「φ_Δ(P, v) = E」と出力する;
8: end for
```

集合関数 $v : 2^P \to \mathbb{R}$ を入力として受け取り，拡張シャープレイ値 $\Phi(v, P) = (\phi_\Delta(P, v))_{\Delta \subseteq P}$ を出力する問題である．

上の定義で，ベクトル $\Phi(v, P)$ の長さは 2^n なので，この問題を解くための時間計算量は $\Omega(2^n)$ である．

3.3 ナイーブアルゴリズム

ESV を解くナイーブな方法を Algorithm 1 に示す．Algorithm 1 は，すべての $\Delta \subseteq P$ に対して，拡張シャープレイ値 $\phi_\Delta(P, v)$ を一つずつ計算する．Algorithm 1 の 1 行目の init_factorial では，0! から $n!$ までの階乗値を持つ階乗テーブル $\mathrm{fact}_i := i!$ を，すべての $i \subseteq \{0, 1, \ldots, n\}$ に対して構築する．$\mathrm{fact}_0 = 1$ とさだめ，それぞれの $i \subseteq \{1 \ldots, n\}$ に対して，$\mathrm{fact}_i = \mathrm{fact}_{i-1} \cdot i$ にしたがって階乗テーブルを計算する．以降では，階乗テーブル fact_i を参照することで，1 から n までの階乗値を得ることができる．Algorithm 1 の 3 行目で，$E \leftarrow 0$ として，拡張シャープレイ値を計算するための変数を初期化する．Algorithm 1 の 4 行目から 6 行目で，すべての集合 $S \subseteq P \setminus \Delta$ に対する関数値を，式 5 にしたがって E に加算する．Algorithm 1 の 7 行目で，集合 Δ に対する拡張シャープレイ値 $\phi_\Delta(P, v)$ として，E を出力する．

ここで，Algorithm 1 の時間計算量に関する定理を与える．

定理 1. 入力として，全体集合 $P = \{1, \ldots, n\}$ と，集合関数 $v : 2^P \to \mathbb{R}$ が与えられたとき，Algorithm 1 の時間計算量は $\mathcal{O}(3^n \cdot t_v(n))$ である．ここに，$t_v(n)$ は関数 v を計算するのにかかる時間である．

証明. Arigorithm 1 の 1 行目で，階乗テーブルの構築に $\mathcal{O}(n)$ 時間かかる．以降では，階乗テーブルを参照することで，任意の $i \subseteq \{0, 1, \ldots, n\}$ に対して，階乗値 $i!$ を $\mathcal{O}(1)$ で得ることができる．Algorithm 1 の 4 行目から 6 行目は，集合 $P \setminus \Delta$ の部分集合 S に対する関数値の総和を，式 5 にしたがって計算する．$P \setminus \Delta$ の部分集合は $2^{|P \setminus \Delta|}$ 個であるため，ループ回数は $2^{|P \setminus \Delta|}$ 回である．$|P \setminus \Delta| = k$ となる S の通り数は ${}_nC_k$ 通りであるので，Algorithm 1 の 5 行目は全体を通して $\sum_{k=0}^{n} {}_nC_k \cdot 2^k$ 回実行される，

Algorithm 2 *ESV* を解く提案アルゴリズム

Require: 変数集合 $P = \{1, \ldots, n\}$，集合関数 v.
Ensure: 拡張シャープレイ値 $\Phi(v, P)$

```
1: すべての Δ ⊆ P に対して，α(Δ) を計算する;
2: すべての Δ ⊆ P に対して，β(Δ) を計算する;
3: for Δ ⊆ P do
4:     「φ_Δ(P, v) = (α(Δ) − β(Δ)) · |Δ|!」と出力する;
5: end for
```

$$
\begin{aligned}
&\sum_{k=0}^{n} {}_nC_k \cdot 2^k \\
&= \sum_{k=0}^{n} {}_nC_k \cdot 2^k \cdot 1^{n-k} \\
&= (2+1)^n \\
&= 3^n
\end{aligned}
$$

と式変形ができるので，Algorithm 1 の 5 行目は 3^n 回実行される．よって，Algorithm 1 の時間計算量は $\mathcal{O}(3^n \cdot t_f(n))$ となる． □

4 提案アルゴリズム

全部分集合拡張シャープレイ値問題（*ESV*）に対して高速ゼータ変換を適用することで，Algorithm 1 より高速なアルゴリズムを与える．

4.1 基本アイデア

拡張シャープレイ値 $\Phi_\Delta(P, v)$ を式変形し，2 つの項に分ける．それぞれの項に高速ゼータ変換を適用させて，高速に計算することを考える．

拡張シャープレイ値 $\Phi_\Delta(P, v)$ を，

$$
\begin{aligned}
&\frac{\Phi_\Delta(P, v)}{|\Delta|!} \\
&= \sum_{S \subseteq P \setminus \Delta} \frac{|S|!(|P| - |S| - |\Delta|)!}{|P|!} \left[f(S \cup \Delta) - f(S) \right] \\
&= \underbrace{\sum_{S \subseteq P \setminus \Delta} \frac{|S|!(|P| - |S| - |\Delta|)!}{|P|!} f(S \cup \Delta)}_{\alpha(\Delta)} \\
&\quad - \underbrace{\sum_{S \subseteq P \setminus \Delta} \frac{|S|!(|P| - |S| - |\Delta|)!}{|P|!} f(S)}_{\beta(\Delta)}
\end{aligned}
$$

と式変形し，第 1 項 $\alpha(\Delta)$ と第 2 項 $\beta(\Delta)$ を定め，それぞれ独立に計算することを考える．すべての $\Delta \subseteq P$ に対して，$\alpha(\Delta)$ と $\beta(\Delta)$ を計算することができれば，ただちに全部分集合拡張シャープレイ問題（*ESV*）の解を得ることができる．すべての $\Delta \subseteq P$ に対して，$\alpha(\Delta)$ と $\beta(\Delta)$ を計算することができると仮定して，*ESV* を解く提案手法を Algorithm 2 に示す．

以降では，$\alpha(\Delta)$ と $\beta(\Delta)$ を計算する手法について説明する．

4.2 第 1 項 $\alpha(\Delta)$ の計算

4.2.1 第 1 項 $\alpha(\Delta)$ に対する部分問題

拡張シャープレイ値の第 1 項 $\alpha(\Delta)$ を

$$\alpha(\Delta) = \sum_{S \subseteq P \setminus \Delta} \frac{|S|!(|P| - |S| - |\Delta|)!}{|P|!} v(S \cup \Delta) \tag{3}$$

と定義する．次に，すべての $\Delta \subseteq P$ に対して，$\alpha(\Delta)$ の効率良い計算方法を与える．

集合 T を $T = S \cup \Delta$ とおき，$\sum$ の S に関するループ条件を，T に関するループ条件に置き換えることを考える．定義より，Δ は P の部分集合であるので，

$$\begin{aligned} & \emptyset \subseteq S \subseteq P \setminus \Delta \\ \iff & \emptyset \cup \Delta \subseteq S \cup \Delta \subseteq (P \setminus \Delta) \cup \Delta \\ \iff & \Delta \subseteq T \subseteq P \end{aligned}$$

となり，$\emptyset \subseteq S \subseteq P \setminus \Delta$ と $\Delta \subseteq T \subseteq P$ は同値である．加えて，$S \subseteq P \setminus \Delta$ より，$S = T \setminus \Delta$ が成り立つので，$\alpha(\Delta)$ は

$$\begin{aligned} \alpha(\Delta) &= \sum_{\emptyset \subseteq S \subseteq P \setminus \Delta} \frac{|S|!(|P| - |S| - |\Delta|)!}{|P|!} v(S \cup \Delta) \\ &= \sum_{\Delta \subseteq T \subseteq P} \frac{|T \setminus \Delta|!(|P| - |T \setminus \Delta| - |\Delta|)!}{|P|!} v(T) \\ &= \sum_{\Delta \subseteq T \subseteq P} \frac{|T \setminus \Delta|!(|P| - |T| + |\Delta| - |\Delta|)!}{|P|!} v(T) \\ &= \sum_{\Delta \subseteq T \subseteq P} \frac{|T \setminus \Delta|!(|P| - |T|)!}{|P|!} v(T) \\ &= \sum_{\Delta \subseteq T \subseteq P} |T \setminus \Delta|! g(T) \end{aligned}$$

と式変形できる．ここに，$g(T) := \frac{(|P|-|T|)!}{|P|!} v(T)$ とおいた．すべての $\Delta \subseteq P$ に対して $\sum_{\Delta \subseteq T \subseteq P} |T \setminus \Delta|! g(T)$ の値を計算することができれば，ただちに，すべての $\Delta \subseteq P$ に対する $\alpha(\Delta)$ の値が求まる．よって，以下で問題を新たに定義し，その問題を解くことを考える．

定義 7. 拡張シャープレイ値部分問題 1（*extended Shapley Value subproblem 1, ESVSP1*）とは，全体集合 $P = \{1, \ldots, n\}$ と集合関数 $f : 2^P \to \mathbb{R}$ を入力として受け取り，すべての $S \subseteq P$ に対する $f\alpha(S)$ からなるベクトル $[f\alpha] = (f\alpha(S))_{S \subseteq P}$ を出力する問題である．ここに，

$$f\alpha(S) = \sum_{S \subseteq X \subseteq P} |X \setminus S|! f(S) \tag{4}$$

である．

4.2.2 第 1 項 $\alpha(\Delta)$ に対するアルゴリズム

高速ゼータ変換を用いて *ESVSP1* を解く方法を Algorithm 3 に示す．

アルゴリズムの基本アイデアを述べる．高速ゼータ変換に基づいた動的計画法によって，所望の値を得る．

Algorithm 3 *ESVSP1* を解く高速なアルゴリズム

Require: 全体集合 $P = \{1, \ldots, n\}$，集合関数 $f : 2^P \to \mathbb{R}$.
Ensure: ベクトル $[f\alpha] = (f\alpha(S))_{S \subseteq P}$

```
for S ⊆ P do
    fα_{0,0}(S) ← 0! f(S);
end for
for i ∈ {1, ..., n} do
    for j ∈ {0, ..., n} do
        for S ⊆ P do
            fα_{i,j}(S) = { j · fα_{i-1,j-1}(S ∪ {i}) + fα_{i-1,j}(S),   if i ∉ S,
                          { fα_{i-1,j}(S),                              if i ∈ S.
        end for
    end for
end for
for S ⊆ P do
    A ← 0;
    for j ∈ {0, ..., n} do
        A ← A + fα_{n,j}(S);
    end for
    「fα(S) = A」と出力する;
end for
```

動的計画法のアルゴリズムを与える．すべての $S \subseteq P$ と，$i \in \{0, \ldots, n\}$，$j \in \{0, \ldots, n\}$ に対して，ゼータ変換テーブル $f\alpha_{i,j}(S)$ を

$$f\alpha_{i,j}(S) := \sum_{\substack{S \subseteq X \subseteq S \cup \{1, \ldots, i\}, \\ |X \setminus S| = j}} |X \setminus S|! f(X). \tag{5}$$

と定義する．ゼータ変換テーブル $f\alpha_{i,j}(S)$ は，$X \setminus S$ のサイズ j によって値を区別して保持している．ゼータ変換テーブルの定義より，$\alpha(S) = \sum_{j=0}^{n} f\alpha_{n,\,j}(S)$ である．アルゴリズムは，動的計画法によってゼータ変換テーブル $f\alpha_{i,j}(S)$ を順に計算していくことにより，すべての $S \subseteq P$ に対する $\alpha(S)$ を得る．

動的計画法によるゼータ変換テーブルの更新方法について述べる．まず，ステージ 0 で，すべての $S \subseteq P$ に対して $f\alpha_{0,0}(S) = 0! f(S)$ とテーブルを初期化する．つぎに，ステージ $i \geqq 1$ で，以下の漸化式にしたがって，すべての $j \in \{0, 1, \ldots, n\}$ とすべての $S \subseteq P$ に対する $f\alpha_{i,j}(S)$ を計算する．

$$f\alpha_{i,j}(S) = \begin{cases} j \cdot f\alpha_{i-1,j-1}(S \cup \{i\}) + f\alpha_{i-1,j}(S), & \text{if } i \notin S, \\ f\alpha_{i-1,j}(S), & \text{if } i \in S. \end{cases} \tag{6}$$

最後に，ステージ n が完了したときに，定義より $\alpha(S) = \sum_{j=0}^{n} f\alpha_{n,j}(S)$ なので，すべての $S \subseteq P$ に対して *ESVSP1* の解が得られる．

アルゴリズムの動作例として，$P = \{1, 2\}$ の時の，高速ゼータ変換により得られる $f\alpha_{i,j}(S)$ のテーブルを表 3 に示す．階乗の値によって，j が一意に定まるため，j に関するセルは陽に区別せずまとめて表示する．また表を簡潔にするため，ここでは $g(\{1,2\})$ を単に $g\{1,2\}$ と表記するものとする．

表 3　$P=\{1,2\}$ における $\alpha_{i,j}(\Delta)$ テーブル

i	$\alpha_i\{\}$	$\alpha_i\{1\}$
0	$0!g\{\}$	$0!g\{1\}$
1	$0!g\{\}+1!g\{1\}$	$0!g\{1\}$
2	$0!g\{\}+1!g\{1\}+1!g\{2\}+2!g\{1,2\}$	$0!g\{1\}+1!g\{1,2\}$
i	$\phi_i\{2\}$	$\phi_i\{1,2\}$
0	$0!g\{2\}$	$0!g\{1,2\}$
1	$0!g\{2\}+1!g\{1,2\}$	$0!g\{1,2\}$
2	$0!g\{2\}+1g!\{1,2\}$	$0!g\{1,2\}$

4.2.3　部分問題 1 に関する計算量解析

Algorithm 3 の計算量に関する定理を与える．

補題 1. 入力として，全体集合 $P=\{1,\dots,n\}$ と集合関数 $f:2^P\to\mathbb{R}$ が与えられたとき，Algorithm 3 の時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ であり，空間計算量は $\mathcal{O}(2^n\cdot n+s_f(n))$ である．ここに，$t_f(n)$ は関数 f を計算するのにかかる時間であり，$s_f(n)$ は関数 f を計算するのに必要なスペースである．

証明. 時間計算量について議論する．Algorithm 3 の 1 行目のループ回数は 2^n であり，$f\alpha_{0,0}(S)$ の初期化にかかる時間計算量は $\mathcal{O}(2^n\cdot t_f(n))$ である．Algorithm 3 の 7 行目の漸化式を計算するのにかかる時間計算量は $\mathcal{O}(1)$ である．Algorithm 3 の 4 行目から 10 行目では，すべての $i\in\{1,\dots,n\}$ と，$j\in\{0,1,\dots,n\}$，$S\subseteq P$ に対して，漸化式にしたがって $f\alpha_{i,j}(S)$ を計算しており，$\mathcal{O}(2^n\cdot n^2)$ 時間かかる．以上により，Algorithm 3 の実行にかかる時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ である．

空間計算量について議論する．式 (4.4) より，ステージ i での漸化式の計算に必要な値は，ステージ $i-1$ の結果のみであるので，アルゴリズムを通して必要な作業領域は $\mathcal{O}(2^n\cdot n+t_f(n))$ である．

以上により，Algorithm 3 の時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ であり，空間計算量は $\mathcal{O}(2^n\cdot n+s_f(n))$ であることが示された．□

4.3　第 2 項 $\beta(\Delta)$ の計算

4.3.1　第 2 項 $\beta(\Delta)$ に対する部分問題

拡張シャープレイ値の第 2 項 $\beta(\Delta)$ を

$$\sum_{S\subseteq P\setminus\Delta}\frac{|S|!(|P|-|S|-|\Delta|)!}{|P|!}v(S) \tag{7}$$

と定義する．次に，すべての $\Delta\subseteq P$ に対して，$\beta(\Delta)$ の効率良い計算方法を与える．

集合 U を $U=P\setminus\Delta$ とおく．このとき，$S\subseteq P\setminus\Delta$ より，$\Delta=P\setminus U$ が成り立つので，$\beta(\Delta)$ は

$$\begin{aligned}\beta(\Delta)&=\sum_{\emptyset\subseteq S\subseteq P\setminus\Delta}\frac{|S|!(|P|-|S|-|\Delta|)!}{|P|!}v(S)\\&=\sum_{\emptyset\subseteq S\subseteq U}\frac{|S|!(|P|-|S|-|P\setminus U|)!}{|P|!}v(S)\\&=\sum_{\emptyset\subseteq S\subseteq U}\frac{|S|!(|U|-|S|)!}{|P|!}v(S)\\&=\sum_{\emptyset\subseteq S\subseteq U}|U\setminus S|!h(S)\end{aligned}$$

と式変形できる．ここに，$h(S):=\frac{|S|!}{|P|!}v(S)$ とおいた．すべての $U\subseteq P$ に対して $\sum_{S\subseteq U}|U\setminus S|!h(S)$ を計算することができれば，ただちに，すべての $\Delta\subseteq P$ に対する $\beta(\Delta)$ が求まる．よって，以下で問題を新たに定義し，その問題を解くことを考える．

定義 8. 拡張シャープレイ値部分問題 2（*extended Shapley Value subproblem 2, ESVSP2*）とは，全体集合 $P=\{1,\dots,n\}$ と集合関数 $f:2^P\to\mathbb{R}$ を入力として受け取り，すべての $S\subseteq P$ に対する $f\beta(S)$ からなるベクトル $[f\beta(S)]=(f\beta(S))_{S\subseteq P}$ を出力する問題である．ここに，

$$f\beta(S)=\sum_{X\subseteq S}|X\setminus S|!f(S) \tag{8}$$

である．

4.3.2　第 2 項 $\beta(\Delta)$ に対するアルゴリズム

高速ゼータ変換を用いて *ESVSP1* を解く方法を Algorithm 4 に示す．

アルゴリズムの基本アイデアを述べる．Algorithm 4 と同様に，ゼータ変換テーブルを，集合 $X\setminus S$ のサイズの情報を保持できる形で定義する．そうすることにより，動的計画法の計算が可能になる．

Algorithm 4 *ESVSP2* を解く高速なアルゴリズム

Require: 全体集合 $P=\{1,\dots,n\}$，集合関数 $f:2^P\to\mathbb{R}$.
Ensure: ベクトル $[f\beta(S)]=(f\beta(S))_{S\subseteq P}$

```
for S ⊆ P do
    fβ_{0,0}(S) ← 0!f(S);
end for
for i ∈ {1,...,n} do
    for j ∈ {0,...,n} do
        for S ⊆ P do
            fβ_{i, j}(S) = { fβ_{i-1, j}(S),                  if i ∉ S,
                           { j·fβ_{i-1,j}(S\{i}) + fβ_{i-1,j}(S),  if i ∈ S.
        end for
    end for
end for
for S ⊆ P do
    B ← 0;
    for j ∈ {0,...,n} do
        B ← B + fβ_{n,j}(S);
    end for
    「fβ(S) = B」と出力する;
end for
```

動的計画法のアルゴリズムを与える．すべての $S \subseteq P$ と，$i \in \{0,1,\ldots,n\}$，$j \in \{0,1,\ldots,n\}$ に対して，ゼータ変換テーブル $f\beta_{i,j}(S)$ を

$$f\beta_{i,j}(S) := \sum_{\substack{S\setminus\{1,\ldots,n\}\subseteq X\subseteq S,\\ |S\setminus X|=j}} |S \setminus X|! f(X).$$

と定義する．ゼータ変換テーブル $f\beta_{i,j}(S)$ は，$S \setminus X$ のサイズ j によって値を区別して保持している．ゼータ変換テーブルの定義より，$\beta(S) = \sum_{j=0}^{n} f\beta_{n,\,j}(S)$ である．アルゴリズムは，動的計画法によってゼータ変換テーブル $f\beta_{i,j}(S)$ を順に計算していくことにより，すべての $S \subseteq P$ に対する $\beta(S)$ を得る．

動的計画法を用いたテーブルの更新方法について述べる．まず，ステージ 0 で，すべての $S \subseteq P$ に対して $f\beta_{0,0}(S) = 0! f(S)$ とテーブルを初期化する．つぎに，ステージ $i \geqq 1$ で，以下の漸化式にしたがって，すべての $j \in \{0,1,\ldots,n\}$ とすべての $S \subseteq P$ に対する $f\beta_{i,j}(S)$ を計算する．

$$f\beta_{i,j}(S) = \begin{cases} f\beta_{i-1,j}(S), & \text{if } i \notin S, \\ j \cdot f\beta_{i-1,j}(S \setminus \{i\}) + f\beta_{i-1,j}(S), & \text{if } i \in S. \end{cases} \tag{9}$$

最後に，ステージ n が完了したときに，定義より $\beta(S) = \sum_{j=0}^{n} f\beta_{n,\,j}(S)$ なので，すべての $S \subseteq P$ に対して *ESVSP2* の解が得られる．

4.3.3 部分問題 2 に関する計算量解析

Algorithm 4 の計算量に関する定理を与える．

補題 2. 入力として，全体集合 $P = \{1,\ldots,n\}$ と集合関数 $f : 2^P \to \mathbb{R}$ が与えられたとき，Algorithm 4 の時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ であり，空間計算量は $\mathcal{O}(2^n \cdot n + s_f(n))$ である．ここに，$t_f(n)$ は関数 f を計算するのにかかる時間であり，$s_f(n)$ は関数 f を計算するのに必要なスペースである．

証明. 時間計算量について議論する．Algorithm 4 の 1 行目のループ回数は 2^n であり，$f\beta_{0,0}(S)$ の初期化にかかる時間計算量は $\mathcal{O}(2^n \cdot t_f(n))$ である．Algorithm 4 の 7 行目の漸化式を計算するのにかかる時間計算量は $\mathcal{O}(1)$ である．Algorithm 4 の 4 行目から 10 行目では，すべての $i \in \{1,\ldots,n\}$ と，$j \in \{0,1,\ldots,n\}$，$S \subseteq P$ に対して，漸化式にしたがって $f\beta_{i,j}(S)$ を計算しており，$\mathcal{O}(2^n \cdot n^2)$ 時間かかる．以上により，Algorithm 4 の実行にかかる時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ である．

空間計算量について議論する．式 (4.7) より，ステージ i での漸化式の計算に必要な値は，ステージ $i-1$ の結果のみであるので，アルゴリズムを通して必要な作業領域は $\mathcal{O}(2^n \cdot n + s_f(n))$ である．

以上により，Algorithm 4 の時間計算量は $\mathcal{O}(2^n(t_f(n)+n^2))$ であり，空間計算量は $\mathcal{O}(2^n \cdot n + s_f(n))$ であることが示された． □

4.4 提案アルゴリズムの計算量解析

以上の議論により，すべての $\Delta \subseteq P$ に対して，シャープレイ値の第 1 項 $\alpha(\Delta)$ と第 2 項 $\beta(\Delta)$ が，それぞれ $\mathcal{O}(2^n(t_v(n)+n^2))$ 時間と $\mathcal{O}(2^n \cdot n + s_f(n))$ メモリで計算可能である．したがって，本論文の主結果である，次の定理を与える．

定理 2. 入力として，全体集合 $P = \{1,\ldots,n\}$ と集合関数 $v : 2^P \to \mathbb{R}$ が与えられたとき，Algorithm 2 の時間計算量は $\mathcal{O}(2^n(t_v(n)+n^2))$ であり，空間計算量は $\mathcal{O}(2^n \cdot n + s_v(n))$ である．ここに，$t_v(n)$ は関数 v を計算するのにかかる時間であり，$s_v(n)$ は関数 v を計算するのに必要なスペースである．

証明. 補題 1 と補題 2 より，明らか． □

5 実　　験

本節では，提案手法の実験結果を示し，その考察を行う．

全部分集合拡張シャープレイ値問題（*ESV*）を解くための 2 つのアルゴリズムに対して，入力の全体集合 $P = \{1,\ldots,n\}$ のサイズを変化させて実行し，速度を計測した．ナイーブな手法である Algorithm 1 と提案手法である Algorithm 2 を比較した．

6 環境および設定

本研究の実験で用いた計算機の環境は以下の通りである．CPU は Intel(R) Core(TM) i5-7360U CPU @ 2.30GHz，メモリは 8 GiB，OS は macOS Mojave 10.14.6 である．すべてのプログラムは Apple LLVM version 10.0.1 (clang-1001.0.46.4) を用いてコンパイルした．オプションに -O3 を用いた．

本研究の実験で扱う全部分集合シャープレイ値問題（*ESV*）は，入力として全体集合 $P = \{1,\ldots,n\}$ と集合関数 $f : 2^P \to \mathbb{R}$ を受け取る．本研究の実験において，*ESV* の入力である集合関数 $f : \mathbb{R}^n \to \mathbb{R}$ として，二層ニューラルネットワークを用いた．入力に対して関数値 f を計算するのにかかる時間計算量を $t_f(n)$ で表すと，$t_f(n,k) = \mathcal{O}(nk)$ である．ここに，k はニューラルネットワークのユニット数とする．

入力サイズ n に対して，ナイーブな手法である Algorithm 1 は，$\mathcal{O}(3^n \cdot t_f(n))$ 時間かかる手法であり，提案手法である Algorithm 2 は，$\mathcal{O}(2^n(n^2+t_f(n)))$ 時間かかる手法である．これらの 2 つのアルゴリズムに対して，全体集合 P のサイズ n を変化させて実行し，計算が終了するまでの速度を計測した．

7 実 験 結 果

実験の結果を表 4 に示す．実行時間が 1 時間を超えたものについては，計測を打ち切った．

この結果より，提案アルゴリズムの方が高速に動作することがわかる．

7.1 終 わ り に

本論文では，機械学習モデルの変数重要度として応用される，シャープレイ値を取り上げ，それを拡張した「拡張シャープレイ値」を定義した．加えて，すべての変数組合せに対して，拡張シャープレイ値を求める問題（*ESV*）を定義し，それを解く効率良いアルゴリズムを与えた．さらに，考案したアルゴリズ

表 4　全部分集合拡張シャープレイ値問題（*ESV*）を解くのにかかった時間（秒）

n	ナイーブアルゴリズム	提案アルゴリズム
1	0.006	0.003
2	0.004	0.003
3	0.003	0.004
4	0.003	0.005
5	0.004	0.006
6	0.008	0.004
7	0.012	0.004
8	0.033	0.005
9	0.096	0.009
10	0.295	0.017
11	0.928	0.036
12	2.929	0.074
13	9.229	0.157
14	29.090	0.353
15	91.465	0.778
16	286.771	1.769
17	896.701	3.950
18	2829.511	8.323
19	-	17.679
20	-	42.649
21	-	95.684
22	-	258.296
23	-	663.746

ムとナイーブな方法を計算機実験により実行速度を比較した．

今後の課題として，拡張シャープレイ値の機械学習への応用を，工学的・理論的に考察することが挙げられる．拡張シャープレイ値が，学習モデルの変数組合せの重要度として応用されることを期待して，研究を進める．加えて，拡張シャープレイ値に対する別の問題を定義し，それを解くアルゴリズムを考察することが挙げられる．例えば，サイズが k 以下の変数組合せに限定して，拡張シャープレイ値を求める問題や，集合関数 f を単純な線形モデルに限定して，拡張シャープレイ値を求める問題などが考えられる．

文　献

[1] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[2] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc., 2017.

[3] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

[4] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1885–1894. JMLR. org, 2017.

[5] Eyal Winter. The shapley value. *Handbook of game theory with economic applications*, Vol. 3, pp. 2025–2054, 2002.

[6] Lloyd S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, Vol. 2, No. 28, pp. 307–317, 1953.

[7] Fedor V. Fomin and Dieter Kratsch. *Exact exponential algorithms*. Springer Science & Business Media, 2010.

[8] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pp. 598–617. IEEE, 2016.

[9] Thomas H. Cormen, Charles E. Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[10] Yasuko Matsui and Tomomi Matsui. NP-completeness for calculating power indices of weighted majority games. *Theoretical Computer Science*, Vol. 263, No. 1-2, pp. 305–310, 2001.

[11] Takeaki Uno. Efficient computation of power indices for weighted majority games. In Kun-Mao Chao, Tsan-sheng Hsu, and Der-Tsai Lee, editors, *Algorithms and Computation*, pp. 679–689. Springer Berlin Heidelberg, 2012.