SHAP値の考え方を理解する

木構造編

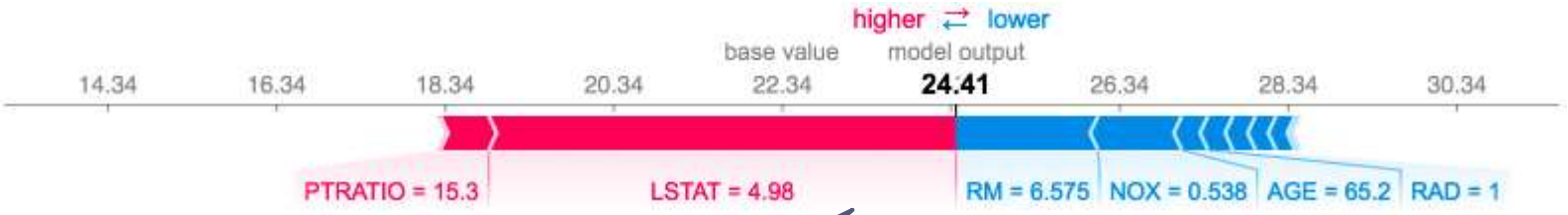
概要

- SHAPライブラリの計算の仕組みを知るために、木構造を対象とした以下の論文を読んだ
  Consistent Individualized Feature Attribution for Tree Ensembles
  https://arxiv.org/pdf/1802.03888.pdf

- まだ理解できていない部分はあるが、重要そうな考え方は理解できたので本資料を作成してみた

SHAP値って何？

- SHapley Additive exPlanationsの略
- ゲーム理論で用いられる考え方をベースにしている
- 予測の時にどの説明変数が影響したかを示すことができる（Feature importances_の予測バージョン）



例えば、PTRATIOが15.3であることやLSTATが4.98であることは
予測値を増やす要因となっていることが読み取れる

https://github.com/slundberg/shap より

SHAPの使い方

- Pip install shapから簡単に使える。以下のURL等を参照

公式ドキュメント

https://github.com/slundberg/shap

https://shap.readthedocs.io/en/latest/#

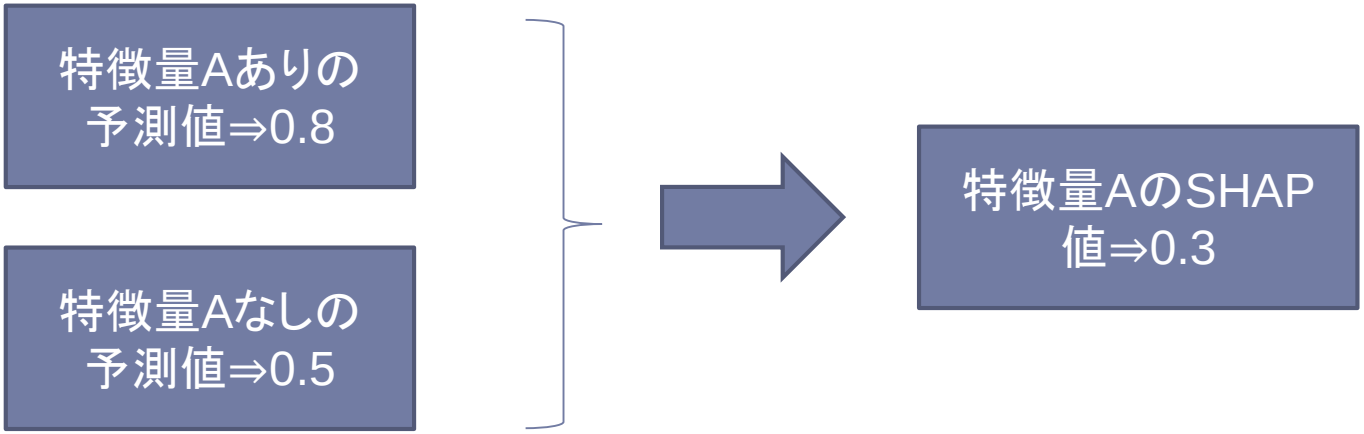
自作ブログ

https://own-search-and-study.xyz/2019/10/05/shap-all-methods/

SHAP値の計算に必要なこと

- 「ある特徴量は予測にどう貢献したか」を計算する必要がある

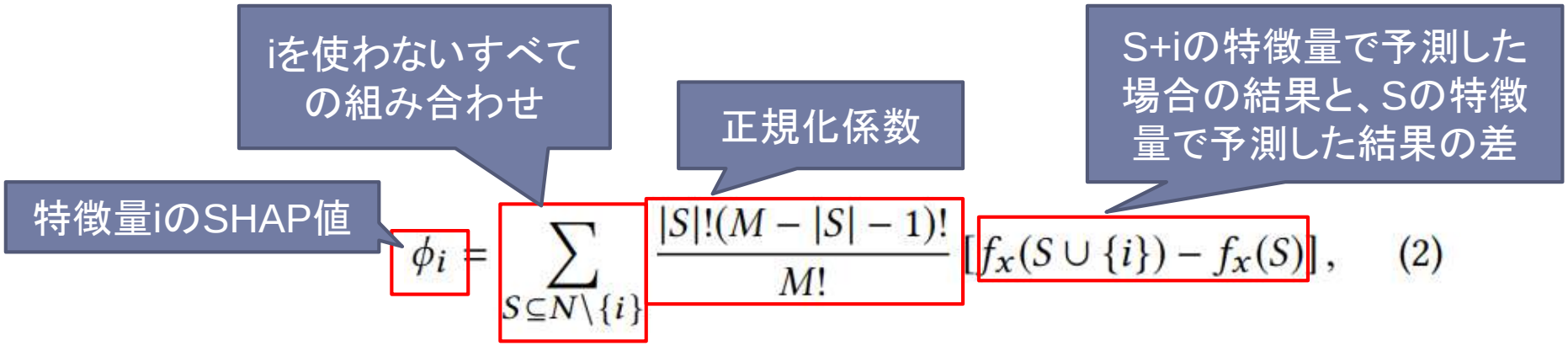
⇒ゲーム理論の考え方（Shapley value）を使う（次ページ）



この部分をどう算出するか？

SHAP値の基本的な考え方

- 特徴量iがあることで、予測がどのように変わったかを全組み合わせについて算出して平均をとる



https://ja.wikipedia.org/wiki/%E3%82%B7%E3%83%A3%E3%83%BC%E3%83%97%E3%83%AC%E3%82%A4%E5%80%A4 より

i：SHAP値を求めたい特徴量
M：特徴量の総数
N：特徴量の全組み合わせの集合（順番考慮）
S：特徴量iを含まない全組み合わせの集合の1つ（順番考慮）
fx：予測モデル

Sのイメージ

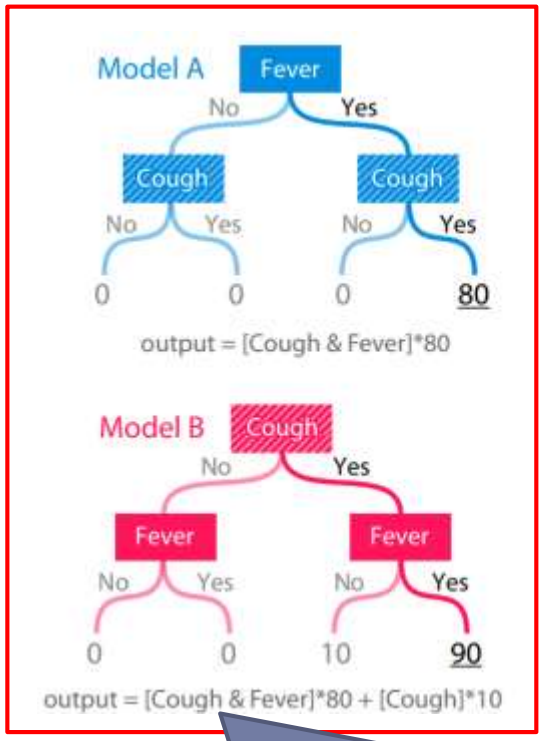
- 特徴量A,B,Cがあり、CのSHAP値を求めたいとするとSは以下の候補がある（Φは空集合）
- Cがどのような状況で追加されたかの全てを考慮するという考え方

Case	順番	S
1	ABC	{A, B}
2	ACB	{A}
3	BAC	{B, A}
4	BCA	{B}
5	CAB	{Φ}
6	CBA	{Φ}

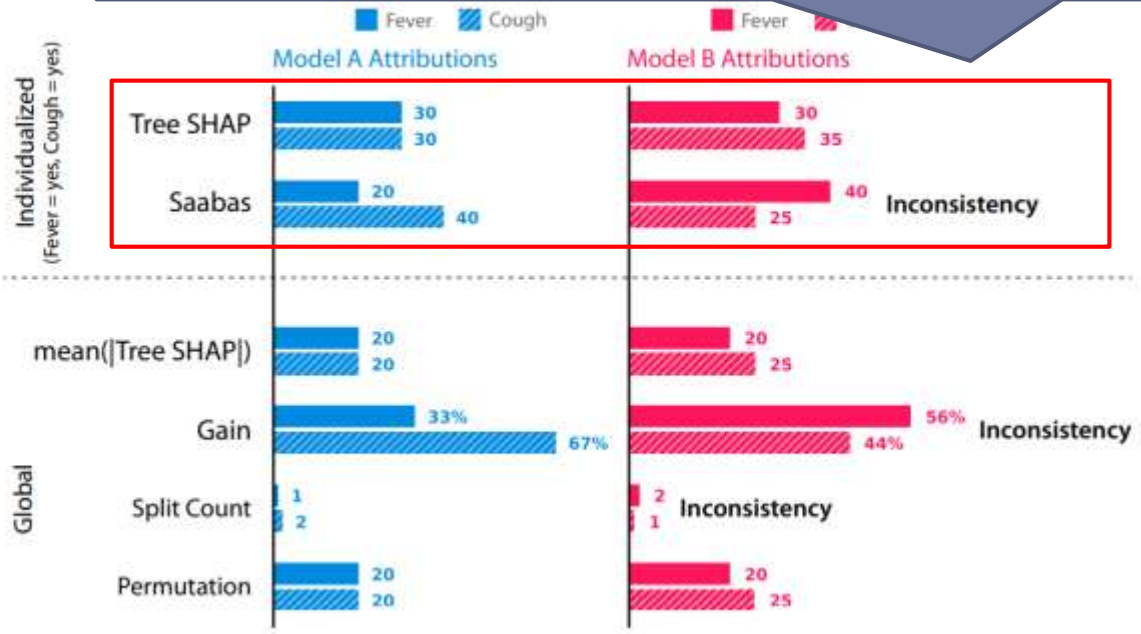
全順列を考えて、Cより前にあるものを抽出したのがS

何故Shapley valueの考え方が必要なのか？

- 論文では、Shapley valueの考え方がないと、似たようなモデルで一貫性のある結果が出ないと主張している



影響度の解釈が、モデルA/BでSHAPは一貫しているが、従来ロジック（Saabas）では倍半分違う



https://arxiv.org/pdf/1802.03888.pdf より

ほぼ同じことを表した2つのモデル
（熱と咳が出るなら風邪）

ここまでの流れをドラゴンボールで説明

- 悟空(G)、ピッコロ(P)、ヤムチャ(Y)が交代でフリーザ軍1000人を相手にするシチュエーションにおいてヤムチャの貢献を定量化したい。
- 単純にはヤムチャが何人倒すかを見ればよさそうだ。
- ヤムチャが何番目に登場するかで貢献度は変わる。

Case	順番	S	Fx(S)	Fx(S∪Y)	ヤムチャが倒した人数
1	GPY	{G, P}	1000人	1000人	0人
2	GYP	{G}	1000人	1000人	0人
3	PGY	{P, G}	1000人	1000人	0人
4	PYG	{P}	900人	960人	60人
5	YPG	{Φ}	0人	300人	300人
6	YGP	{Φ}	0人	300人	300人

ヤムチャのSHAP値
＝110人
（0+0+0+60+300+300)
/6

具体的なSHAP値の算出（その１）

- fx(S)=E[f(x)|xS]を以下のアルゴリズムで求めるとする
- 木を辿っていって、最後についたノードの値を返すアルゴリズム。ただし、モデルの分岐の特徴量がSに選ばれていない場合データ量の重みづけて両方を辿った場合を考慮する

Algorithm 1 Estimating $E[f(x) \mid x_S]$

```
procedure EXPVALUE(x, S, tree = {v, a, b, t, r, d})
    procedure G(j, w)
        if v_j ≠ internal then
            return w · v_j
        else
            if d_j ∈ S then
                return G(a_j, w) if x_{d_j} ≤ t_j else G(b_j, w)
            else
                return G(a_j, wr_{a_j}/r_j) + G(b_j, wr_{b_j}/r_j)
            end if
        end if
    end procedure
    return G(1, 1)
end procedure
```

葉にたどり着いた時の処理

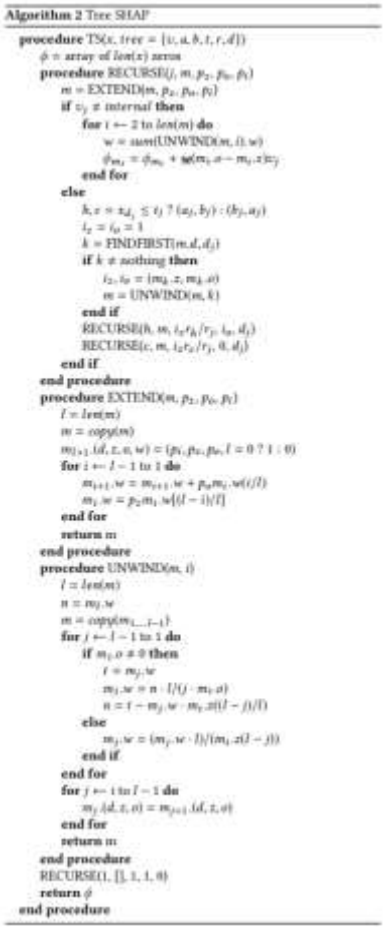
葉にたどりつくまで時の処理

モデルの分岐の特徴量がSに選ばれていない場合の処理

x：特徴量の値
S：特徴量の部分集合
以下は、ノード数のベクトル
v：ノードの目的変数の値
a：子ノードのindex（左）
b：子ノードのindex（右）
t：ノードの分岐の値
r：ノードのデータの量
d：ノードの特徴量のindex

具体的なSHAP値の算出（その２）

- Sの組み合わせは、各特徴量を使う/使わないの組み合わせとなるため、すべての特徴量のSHAP値を求めるためには、2^Mの計算を実施する必要がある
- 論文では、計算を工夫することで多項式時間で計算するアルゴリズムを検討した



Algorithm 2 Tree SHAP

procedure TS(x, $tree = (v, a, b, t, r, d)$)
  ϕ = array of $len(x)$ zeros
  procedure RECURSE(j, m, p_z, p_o, p_i)
    m = EXTEND(m, p_z, p_o, p_i)
    if $v_j \neq$ *internal* **then**
      for $i \leftarrow 2$ to $len(m)$ **do**
        w = sum(UNWIND(m, i).w)
        $\phi_{m_i} = \phi_{m_i} + w(m_i.o - m_i.z)v_j$
      end for
    else
      $h, x = x_{d_j} \leq v_j$? $(a_j, b_j) : (b_j, a_j)$
      $i_z = i_o = 1$
      k = FINDFIRST($m.d, d_j$)
      if $k \neq$ *nothing* **then**
        $i_z, i_o = (m_k.z, m_k.o)$
        m = UNWIND(m, k)
      end if
      RECURSE(h, m, $i_z r_h / r_j$, i_o, d_j)
      RECURSE(x, m, $i_z r_x / r_j$, 0, d_j)
    end if
  end procedure
  procedure EXTEND(m, p_z, p_o, p_i)
    $l = len(m)$
    $m = copy(m)$
    $m_{l+1}.(d, z, o, w) = (p_i, p_z, p_o, l = 0\ ?\ 1 : 0)$
    for $i \leftarrow l - 1$ to 1 **do**
      $m_{i+1}.w = m_{i+1}.w + p_o m_i.w(i/l)$
      $m_i.w = p_z m_i.w(l - i)/l$
    end for
    return m
  end procedure
  procedure UNWIND(m, i)
    $l = len(m)$
    $n = m_l.w$
    $m = copy(m_{1 \ldots l-1})$
    for $j \leftarrow l - 1$ to 1 **do**
      if $m_i.o \neq 0$ **then**
        $t = m_j.w$
        $m_j.w = n \cdot l/(j \cdot m_i.o)$
        $n = t - m_j.w \cdot m_i.z(l - j)/l$
      else
        $m_j.w = (m_j.w \cdot l)/(m_i.z(l - j))$
      end if
    end for
    for $j \leftarrow i$ to $l - 1$ **do**
      $m_j.(d, z, o) = m_{j+1}.(d, z, o)$
    end for
    return m
  end procedure
  RECURSE(1, [], 1, 1, 0)
  return ϕ
end procedure

長いので、詳細の解読は断念

一度の計算で、すべてのSを考慮しながら
計算する作りになっている

今後の予定

- SHAPライブラリのツリー構造以外の理論を追いかける
- ツリー構造のアルゴリズム2の中身を理解する