

スライド、コードはこちらからダウンロードできます

https://github.com/yuyay/DEIM2022_XAI_tutorial

DEIM2022 チュートリアル

説明可能な機械学習入門 ～ Feature Attribution法 ～

千葉工業大学
人工知能・ソフトウェア技術研究センター
吉川 友也

自己紹介



- 吉川 友也
- 千葉工業大学
人工知能・ソフトウェア技術研究センター
主任研究員
- 博士（工学）
 - 2015年 奈良先端科学技術大学院大学
- 研究トピック
 - 機械学習（説明可能AI、深層学習、カーネル法、ベイズモデル等）
 - コンピュータビジョン（動作認識、OCR等）

本チュートリアルの内容

- 目標
 - LIME・SHAPについての基礎と実装方法を理解する
- 内容の難易度
 - 学部4年生レベル（？）
- 想定する前提知識
 - 線形回帰
 - ニューラルネット（NN）

アジェンダ

1. 背景 (10分)

- XAIが研究されている理由
- XAIがどんな場面で役立つか？

2. Feature Attribution法入門 (40分)

- LIME, SHAPについて紹介

3. Pythonによる実装の紹介【録画】 (10分)

4. 最近のFeature Attribution法に関する研究 (25分)

5. Q&A (5分)

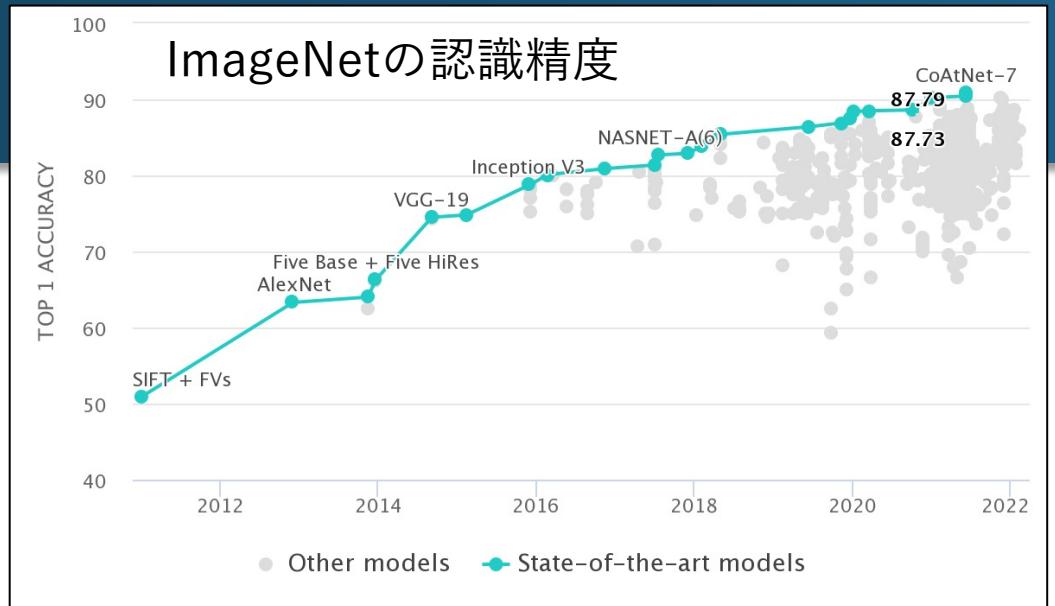


もし質問があればWebEx上の
Q&Aに書いてください
録画を流している間に
できるだけ返します

背景

背景 機械学習（特にNN）の進歩

- 2012年頃から、画像・音声・言語等様々な分野でニューラルネットが台頭
- 様々な場面で実用化されている
(or されつつある)



<https://paperswithcode.com/sota/image-classification-on-imagenet>

顔認証



<https://jpn.nec.com/biometrics/face/about.html>

機械翻訳



音声合成



自動運転



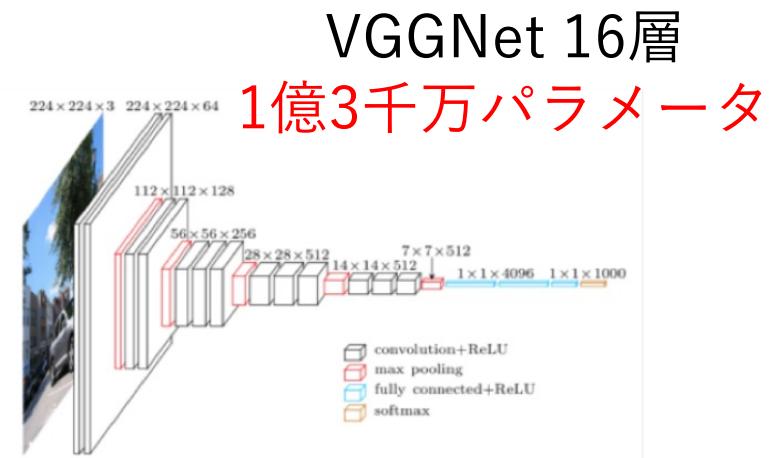
<https://gigazine.net/news/20170125-tesla-fully-self-driving-car/>

背景 ブラックボックスな機械学習

高精度の機械学習法ほど複雑な構造となり
内部の処理を人間が理解することは困難



なぜ予測が当たる/外れるのか人間が理解できない



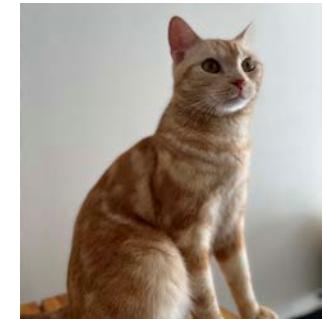
ガンです



どこを見て
判断している？

簡単そうなのに
なぜ間違えた？

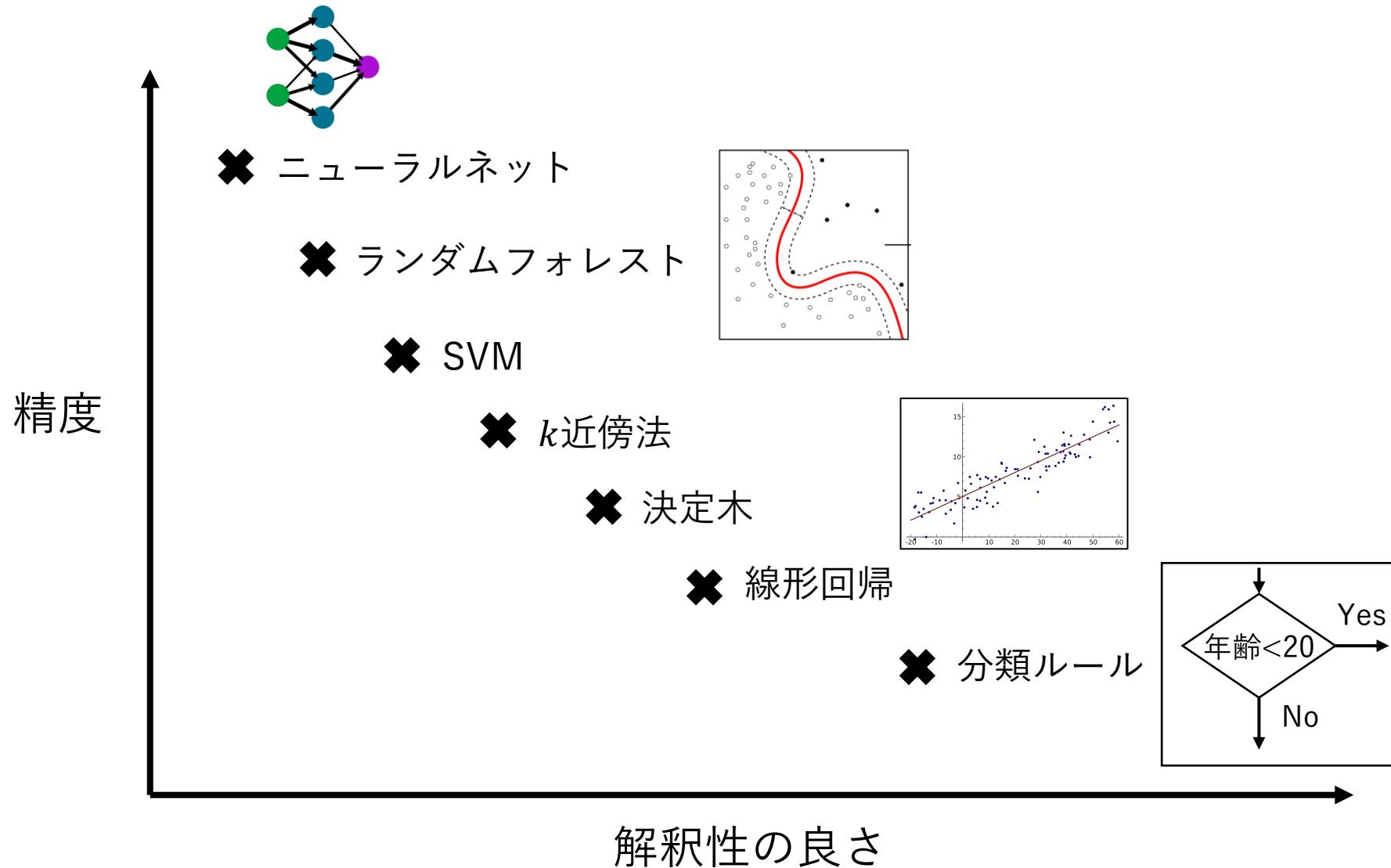
犬です



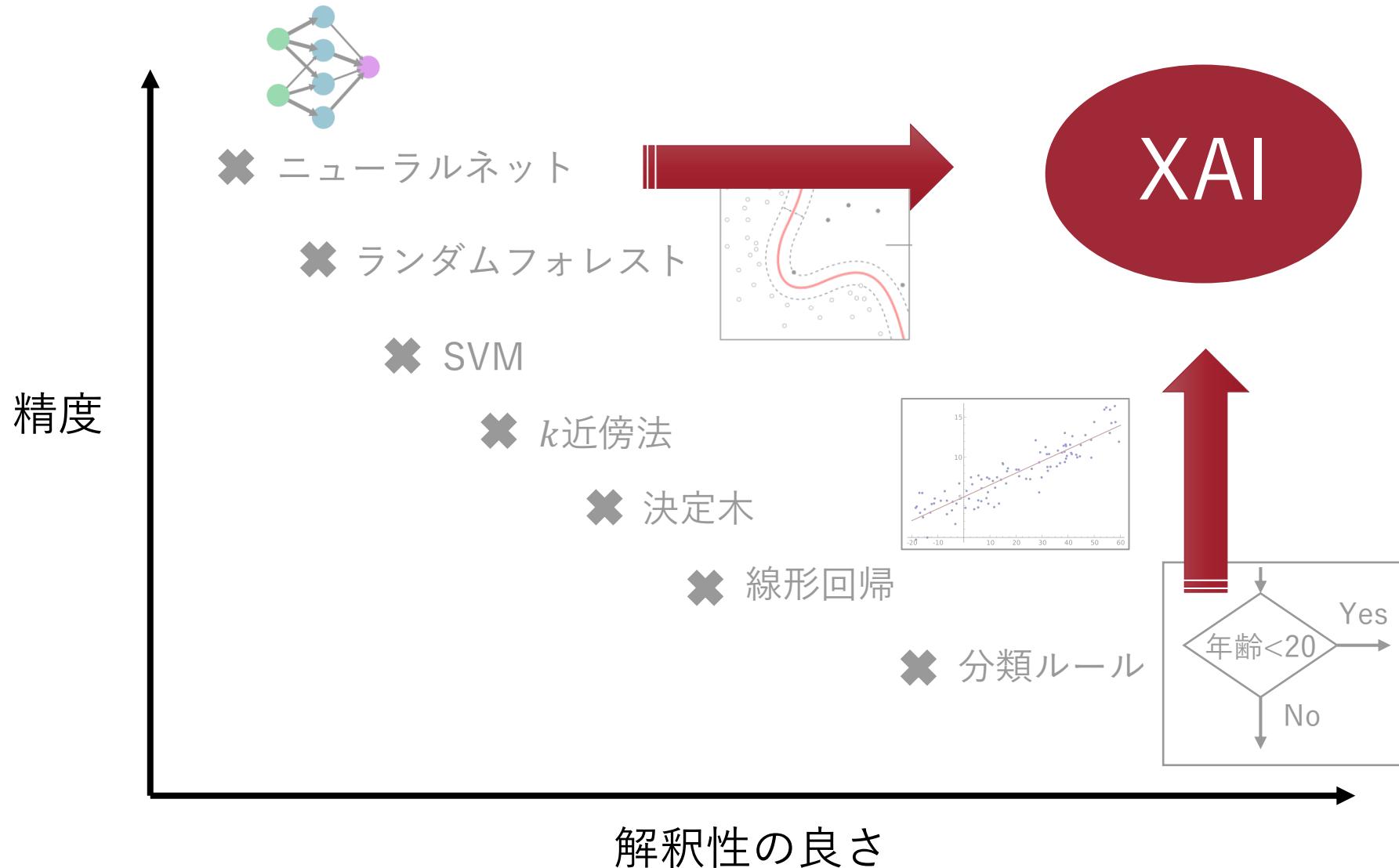
背景 AI（機械学習）の透明性についての社会のニーズ

- より責任が伴う場でのAIの利用が進んでいる
 - 医療、自動運転等の誤動作を避けなければいけない場面
 - 与信審査等の公平性・信頼性が求められる場面
- 一方で、AIに対する不信感、恐怖感の声もある
- AIが社会に受け入れられ適正に利用されるために、世界でルール作りが行われている
 - 日本: 人間中心のAI社会原則
“公平性及び透明性のある意思決定とその結果に対する説明責任が適切に確保”
 - OCDE: OECD Principles on AI
“AIシステムについて、(中略)、透明性を確保し責任ある情報開示を行うべきである”

AIの透明性を改善するまでの課題 “精度”と“解釈性”的トレードオフ

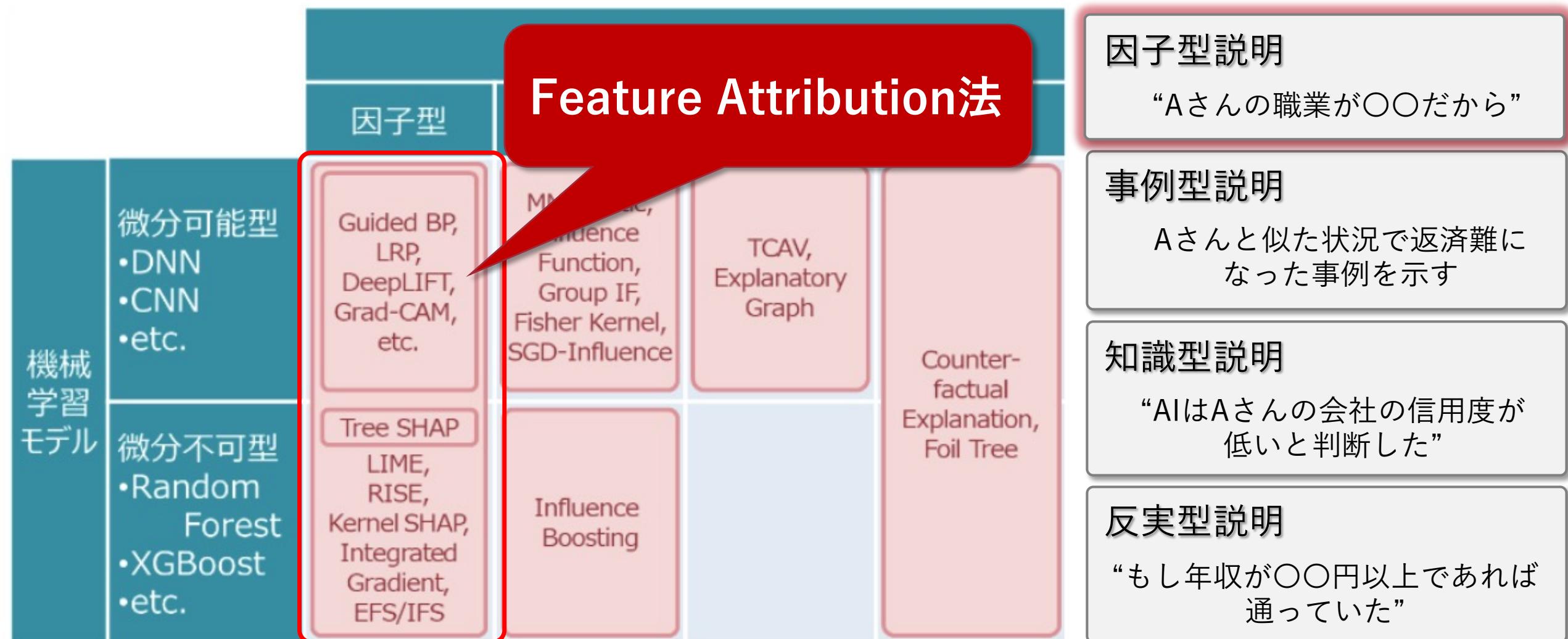


AIの透明性を改善するまでの課題 “精度”と“解釈性”的トレードオフ



局所説明の種別					
	因子型	事例型	知識型	反実型	
機械学習モデル	微分可能型 •DNN •CNN •etc.	Guided BP, LRP, DeepLIFT, Grad-CAM, etc.	MMD-critic, Influence Function, Group IF, Fisher Kernel, SGD-Influence	TCAV, Explanatory Graph	因子型説明 “Aさんの職業が○○だから”
	微分不可型 •Random Forest •XGBoost •etc.	Tree SHAP LIME, RISE, Kernel SHAP, Integrated Gradient, EFS/IFS	Influence Boosting	Counter- factual Explanation, Foil Tree	事例型説明 Aさんと似た状況で返済難に なった事例を示す
					知識型説明 “AIはAさんの会社の信用度が 低いと判断した”
					反実型説明 “もし年収が○○円以上であれば 通っていた”

図は[恵木 2020]より引用



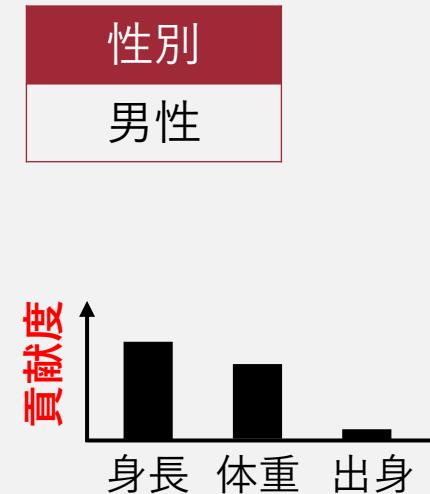
図は[恵木 2020]より引用

Feature Attribution法

予測結果を特徴量の貢献度で説明する手法

表形式データの場合

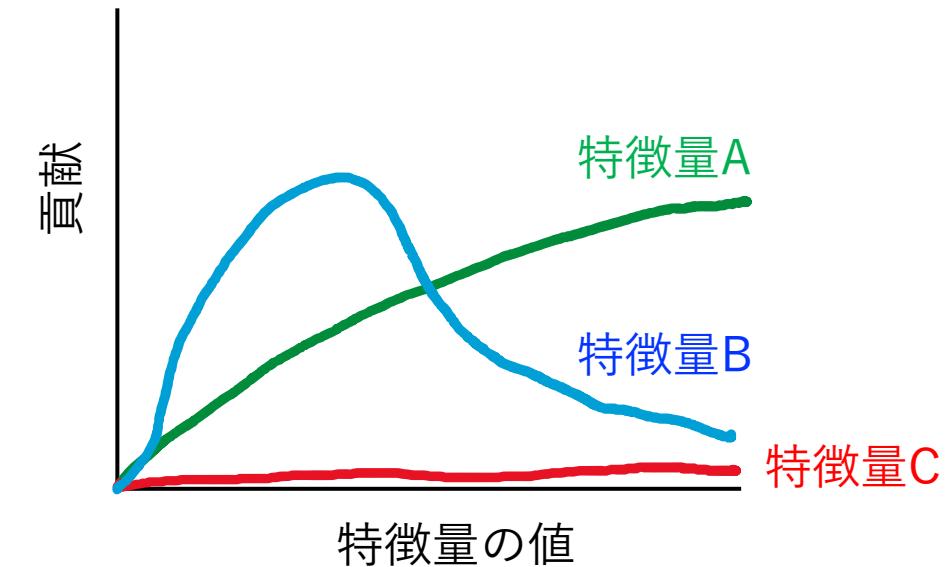
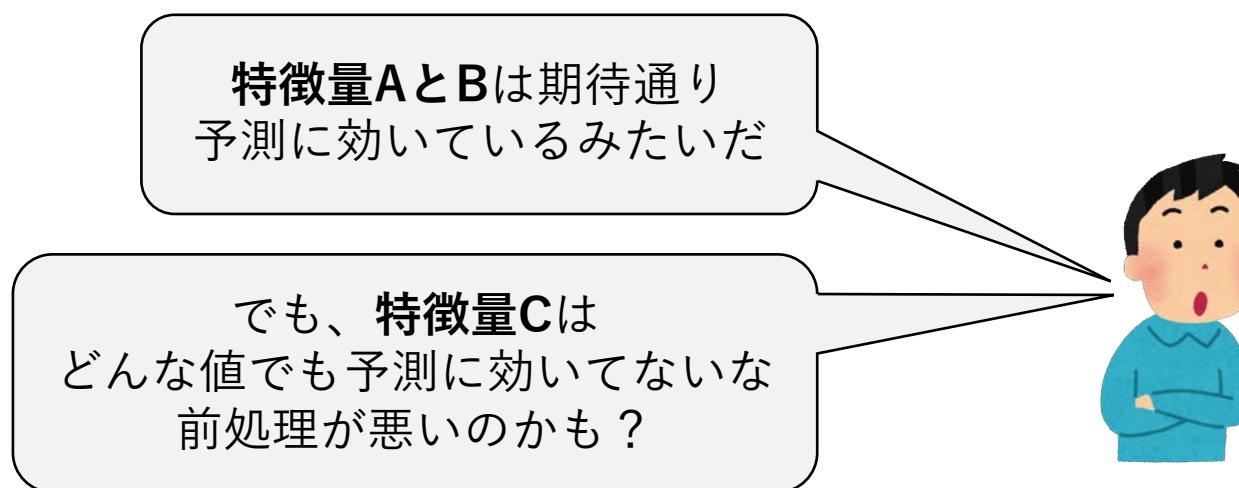
特徴量		
身長	体重	出身
172	63	東京



- 代表的な手法が **LIME** と **SHAP**
 - 特徴量の貢献度を足し合わせると予測値になる

特徴量の貢献が説明できるメリット 予測に効く・効かない特徴量を調べられる

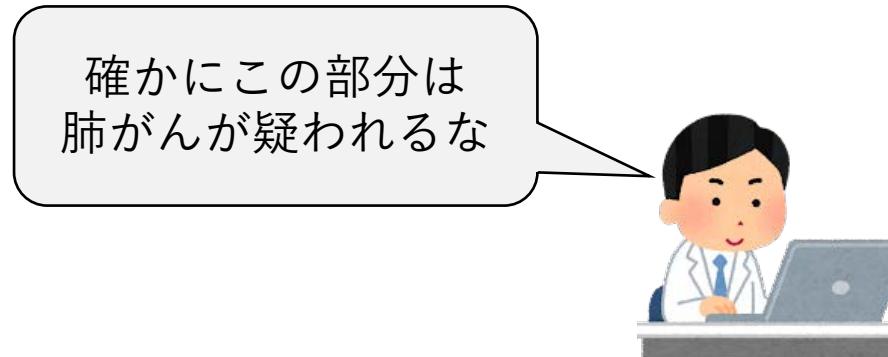
- 特徴量が予測にどう効いているのかを見れば、モデルの傾向を理解することが可能
 - さらに性能を上げるための**特徴量エンジニアリング**の補助になる



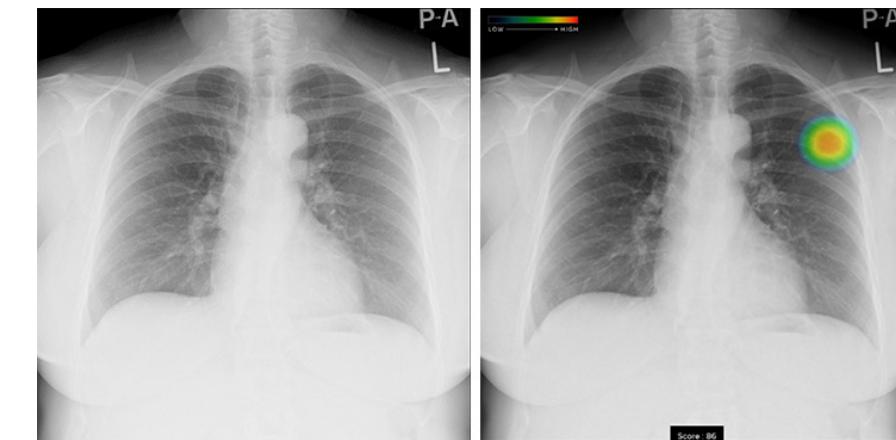
特徴量の貢献が説明できるメリット AIによる予測結果をユーザが受け入れる判断材料になる

- 重要な意思決定が必要な場面では、最終的に人間がAIの予測結果の妥当性を判断することが必要
- 予測の根拠として特徴量の貢献を提示することで、人間の意思決定の補助が可能

例: 医療画像診断におけるAI



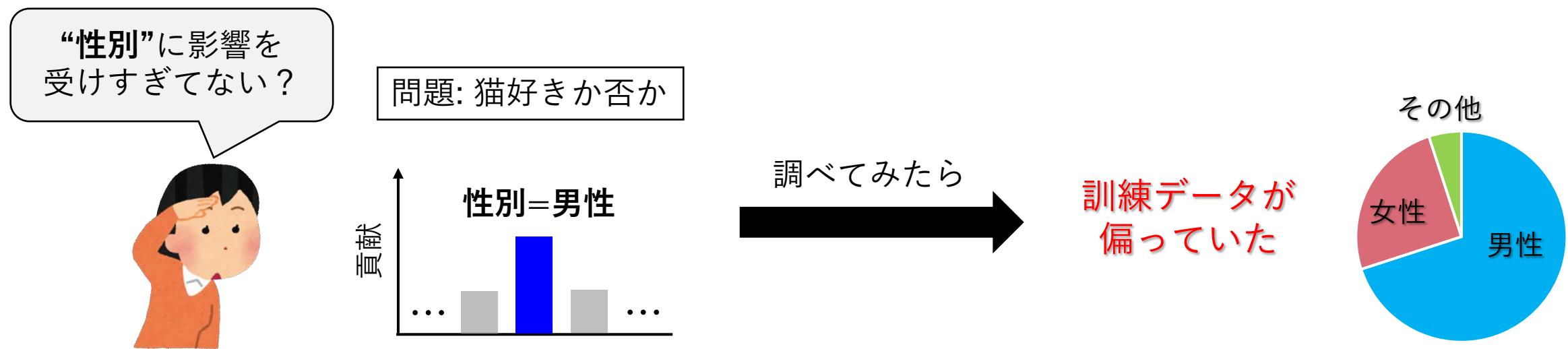
モデルの予測: 肺がん



画像は <https://www.fujifilm.com/jp/ja/news/list/6823> から引用

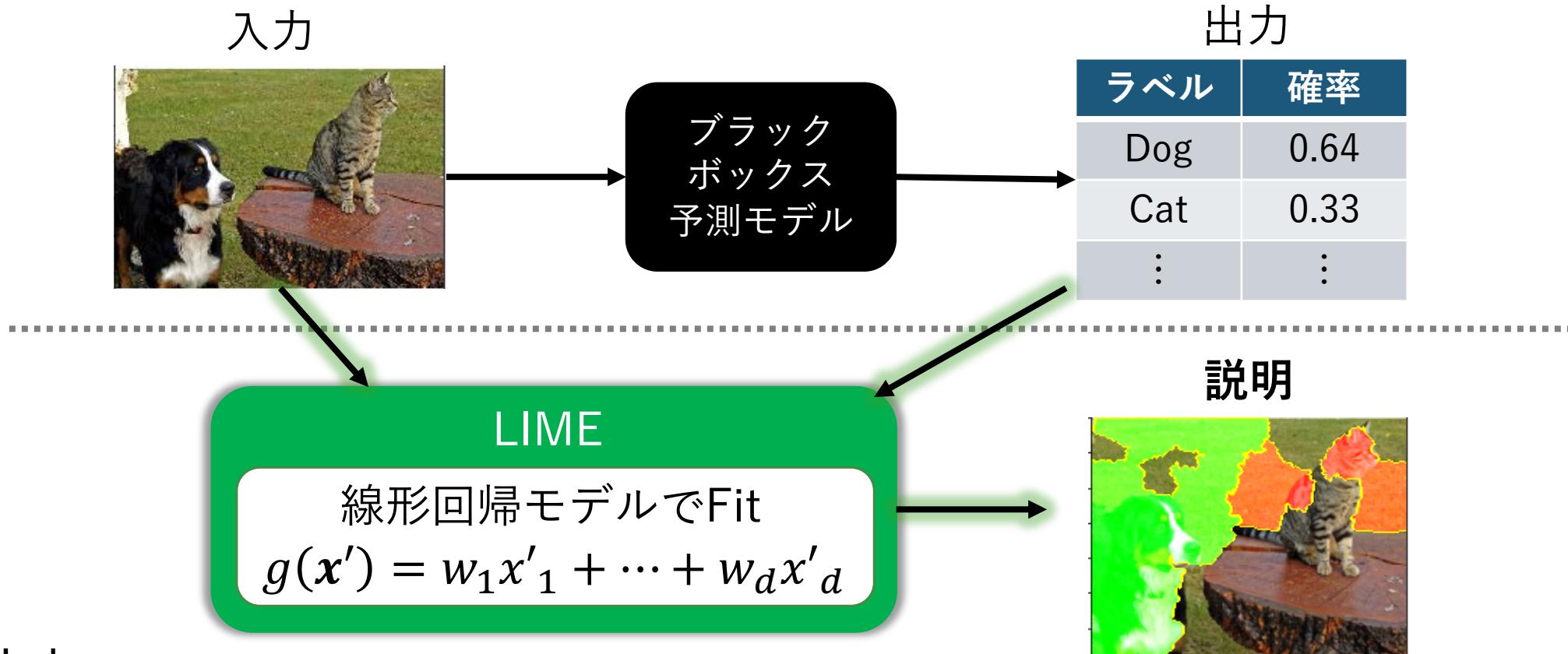
特徴量の貢献が説明できるメリット
モデルが公平な予測を行っているか知ることができる

- ・訓練データ中の特徴量の偏りのせいで、モデルは不公平な予測をする可能性がある
- ・特徴量の貢献を知ることで、モデルの不公平さに早く気が付き、訓練データを修正するなどの対策が取れる



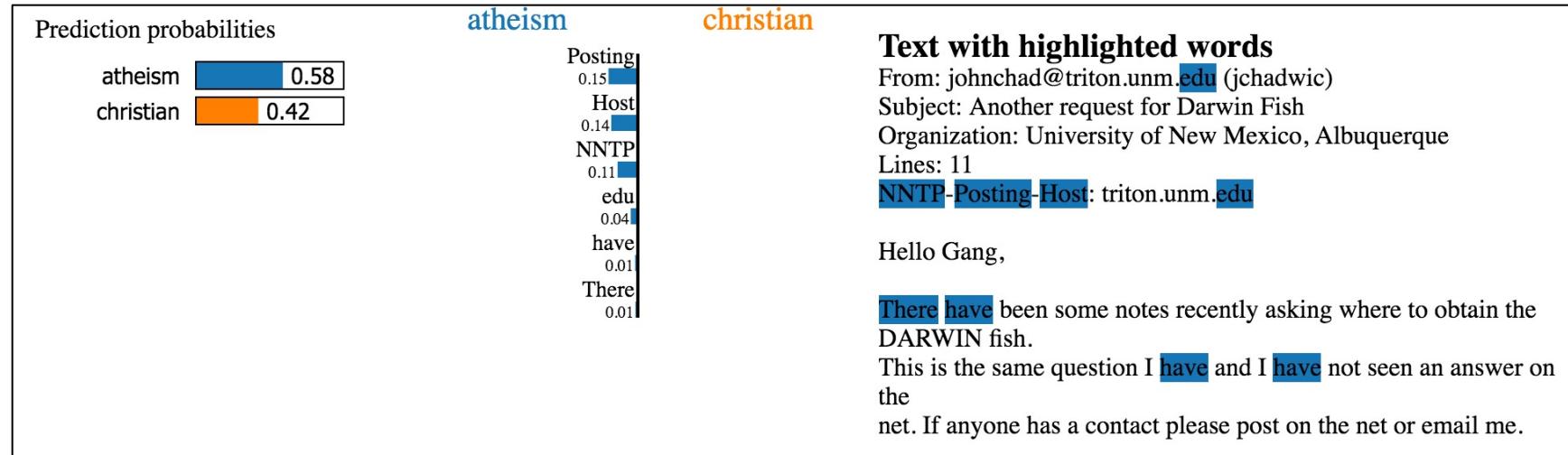
Feature Attribution法入門 ～LIME & SHAP～

- ・ ブラックボックスな予測モデルの出力を線形回帰モデルで近似することで各特徴量が予測値に対してどう貢献するかを説明する手法

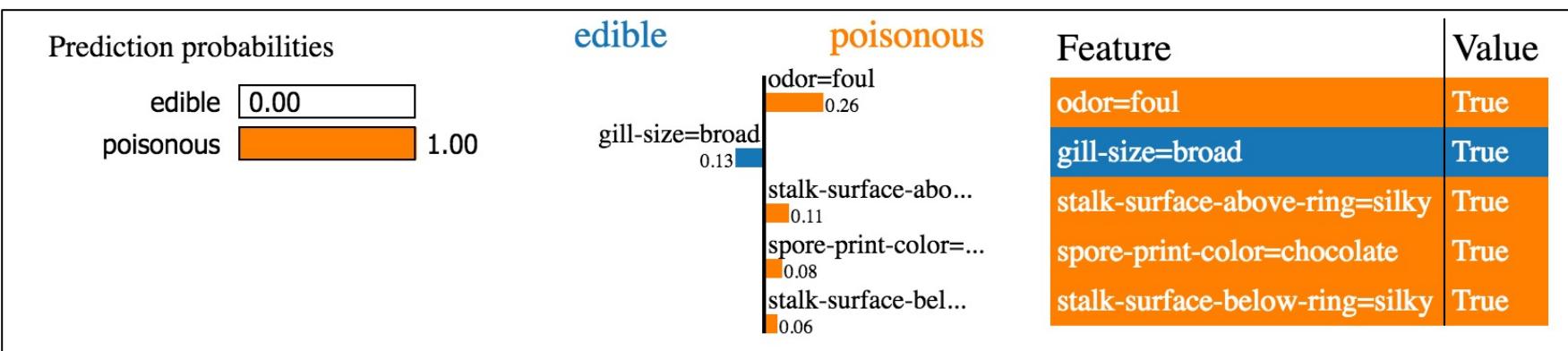


LIMEによる説明の例

テキスト分類



表形式データに対する分類



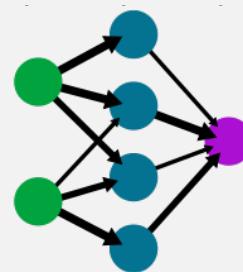
画像分類



LIME利用の流れ

1. 予測モデルを用意

- 予測モデル $f: \mathbb{R}^d \rightarrow \mathbb{R}$ を学習



2. LIMEの学習

- テスト事例 $x \in \mathbb{R}^d$ の近傍 $z \in \mathbb{R}^d$ を用いて、予測モデルの出力 $f(x)$ を近似する線形回帰モデル $g: \mathbb{R}^d \rightarrow \mathbb{R}$ を学習

$$g(x) = w_1x_1 + \cdots + w_dx_d$$

$w \in \mathbb{R}^d$: 係数パラメータ

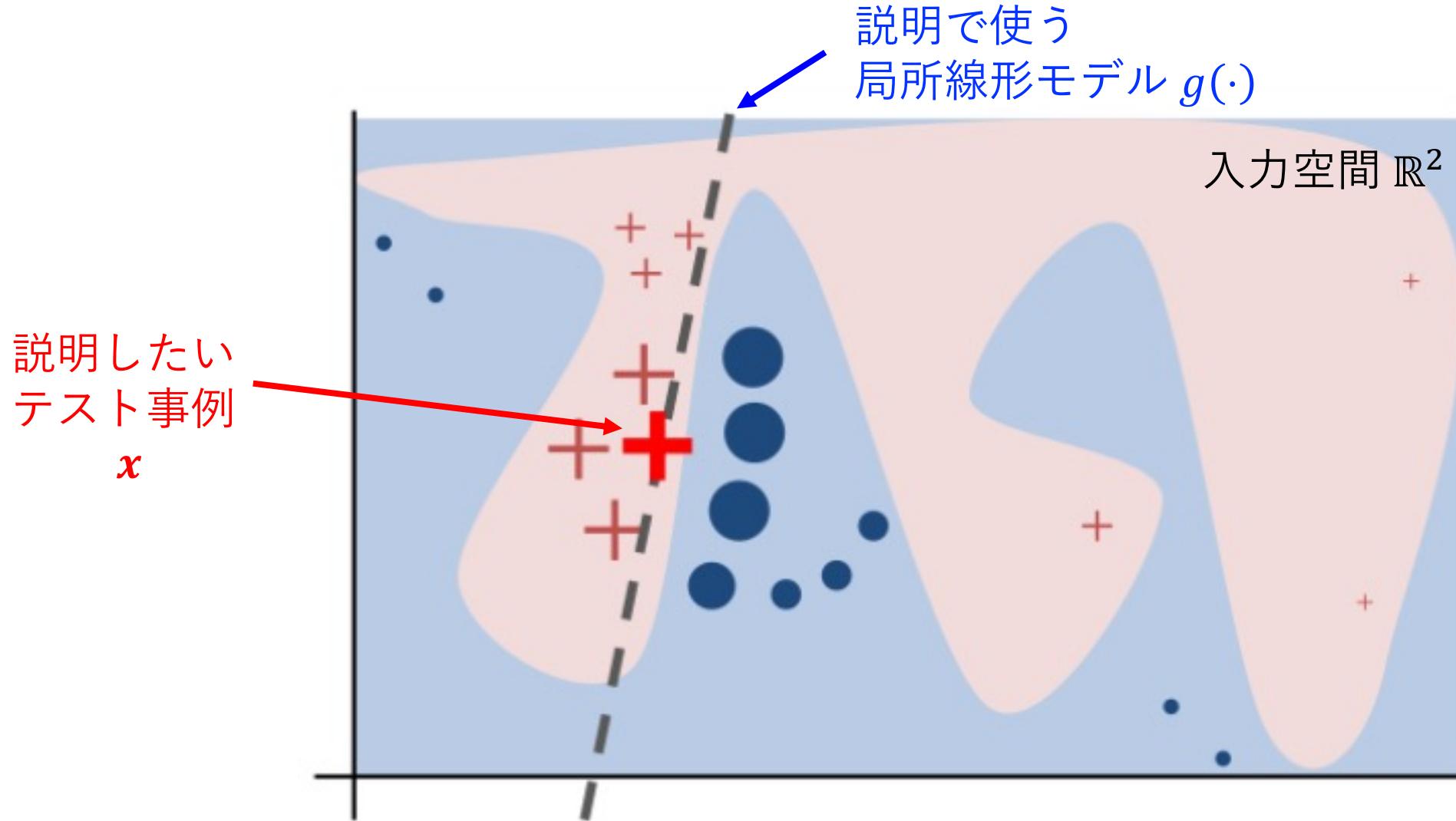
3. LIMEによる説明を解釈

- ユーザは、線形回帰モデル g の係数（重み） $w \in \mathbb{R}^d$ を見て、出力 $f(x)$ に対する各特徴量の貢献度を理解する



そこを見て
判断しているのか～

LIMEにおける説明モデルの直感的なイメージ 青クラスと赤クラスの二値分類の場合



図は[Ribeiro+ 2016] Fig. 3から引用

「リッチな入力」と「解釈しやすい入力」の利用

- 何を見て予測モデルが判断しているのか明らかにするために、元々の入力に加えて、解釈しやすい表現（解釈表現）を説明のために利用

テキスト

元表現 x : 単語の系列

For Christmas Song visit my channel ! ;)

解釈表現 x' : bag-of-wordsベクトル

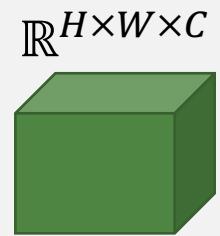
For
Christmas
Song
visit
channel
my
;) !

$$\{0,1\}^d$$

0 1 1 1 … 1 0 1

画像

元表現 x : RGBピクセル値



解釈表現 x' : Superpixelのマスクベクトル



$$\{0,1\}^d$$

0 1 1 1 … 1 0 1

LIMEの説明モデルでは解釈表現を入力とする: $g(x') = w_1x_1 + \dots + w_dx_d$

LIMEの説明モデルの学習 = 重み付き最小二乗法を解く

- テスト事例の解釈表現 \mathbf{x}' の近傍集合 $\mathcal{N}_{\mathbf{x}'}$ からランダムに疑似解釈表現 \mathbf{z}' を生成し、それに対する予測モデルの出力 $f(h(\mathbf{z}'))$ を近似するように線形回帰の係数 \mathbf{w} を最適化
 - その際、 \mathbf{z}' は \mathbf{x}' との類似度（カーネル）に応じて重み付け

$\mathcal{N}_{\mathbf{x}'}$: \mathbf{x}' の近傍集合

$h(\mathbf{z}')$: 解釈表現 \mathbf{z}' から元表現を復元する関数

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \sum_{\mathbf{z}' \sim \mathcal{N}_{\mathbf{x}'}} \pi(\mathbf{x}', \mathbf{z}') (f(h(\mathbf{z}')) - \mathbf{w}^\top \mathbf{z}')^2 + \lambda \Omega(\mathbf{w})$$

$$\pi(\mathbf{x}', \mathbf{z}') = \exp(-\|h(\mathbf{x}') - h(\mathbf{z}')\|_2^2 / \sigma^2)$$

$$\Omega(\mathbf{w}) = \|\mathbf{w}\|_2^2 \quad (\text{正則化項})$$

LIMEの実装

- 近傍事例の作り方がデータの種類によって異なる
 - 表形式データ
 - テキストデータ
 - 画像データ
- LIME著者がどのような実装をしているかを紹介

 [marcotcr/lime](#)

Lime: Explaining the predictions of any machine learning classifier

 JavaScript  9.5k  1.6k

表形式データに対するLIME

- 表形式データ
 - 数値特徴とカテゴリカル特徴が混在
 - 1行が1事例に対応

数値特徴				カテゴリカル特徴		ラベル
身長	体重	出身	職業			
172	63	東京	学生			
161	50	神奈川	会社員			
:	:	:	:			
159	49	静岡	公務員			

表形式データに対するLIME：数値特徴の近傍事例の作り方

前処理

- 訓練データを用いて**数値特徴の値を量子化**（量子化の方法は選択可能）

疑似解釈表現 \mathbf{z}'

- どのビンを選ぶかをランダムに決定。入力と同じだったら1、違えば0

疑似元表現 $h(\mathbf{z}')$

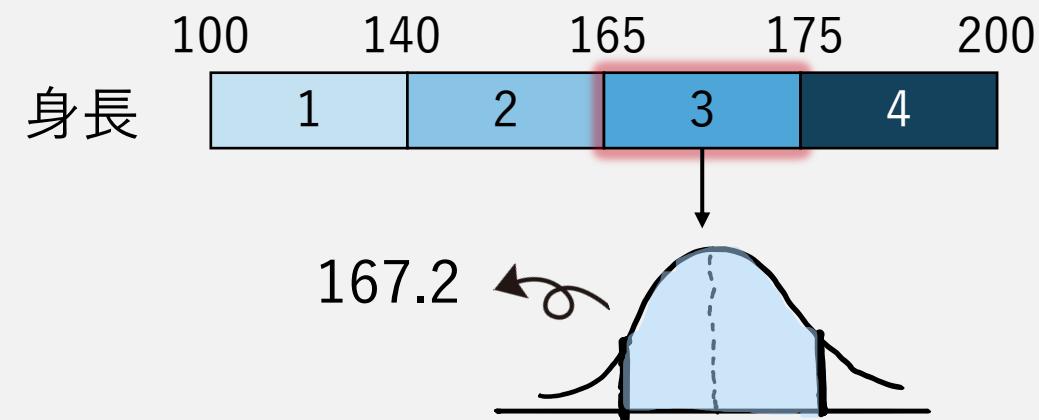
- 選ばれたビンに対応する分布から値をサンプリング

例：四分位量子化

準備 訓練データを用いて各特徴量の四分位数を求める

生成

ランダムにビンを決定し、そのビンにおける平均・標準偏差・最小・最大を用いて
切断正規分布から近傍の特徴量の値を生成



表形式データに対するLIME：カテゴリカル特徴の近傍事例の作り方

前処理

- 訓練データから特徴の頻度割合を計算

疑似元表現 $h(\mathbf{z}')$

- その割合に基づいて、ランダムに特徴を生成

疑似解釈表現 \mathbf{z}'

- 対応する解釈可能な入力では、元の入力と生成された特徴が同じなら1、そうでなければ0の二値で表現

1. 特徴の割合を計算

出身	北海道	…	東京	…	沖縄
割合	0.03	…	0.12	…	0.01

3. 入力と比較

生成された
特徴値

東京

入力における
特徴値

出身
東京



2. 特徴を生成

東京



テキストデータに対するLIME

テスト事例の元表現 x

For Christmas Song visit my channel ! ;)



近傍事例をランダム生成

疑似元表現 z

[MASK] Christmas [MASK] visit my [MASK] ! ;)

疑似解釈表現 z'

[0 1 0 1 1 0 11]

- テスト事例に対して、ランダムに単語をマスクして疑似元表現を生成
 - マスクされた単語は [MASK] のような特別な単語に置き換え
- 疑似解釈表現では、マスクするところを0、それ以外を1とする二値ベクトル

画像データに対するLIME 画像のセグメンテーションを利用

近傍事例構築の手順

1. Superpixelアルゴリズムを用いて、
画像のセグメンテーションを求める
2. 各領域をマスクするかどうかを0/1
で表現したものを擬似解釈表現 \mathbf{z}' とし、
ランダムに生成する
3. 擬似解釈表現 \mathbf{z}' に従ってマスクした画像
を擬似元表現 $h(\mathbf{z}')$ とする

テスト事例



セグメン
テーション
→

$d = 100$ 個の領域



ランダムに生成

各領域をマスクするか
どうかを表すベクトル

$$\mathbf{z}' \in \{0,1\}^d$$

0 1 1 … 1 0 1

(マスクする場合が0)

→



画像は https://scikit-image.org/docs/0.11.x/auto_examples/plot_segmentations.html から引用

LIMEの説明モデルの学習 = 重み付き最小二乗法を解く

- テスト事例の解釈表現 \mathbf{x}' の近傍集合 $\mathcal{N}_{\mathbf{x}'}$ からランダムに疑似解釈表現 \mathbf{z}' を生成し、それに対する予測モデルの出力 $f(h(\mathbf{z}'))$ を近似するように線形回帰の係数 \mathbf{w} を最適化
 - その際、 \mathbf{z}' は \mathbf{x}' との類似度（カーネル）に応じて重み付け

$\mathcal{N}_{\mathbf{x}'}$: \mathbf{x}' の近傍集合

$h(\mathbf{z}')$: 解釈表現 \mathbf{z}' から元表現を復元する関数

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \sum_{\mathbf{z}' \sim \mathcal{N}_{\mathbf{x}'}} \pi(\mathbf{x}', \mathbf{z}') (f(h(\mathbf{z}')) - \mathbf{w}^\top \mathbf{z}')^2 + \lambda \Omega(\mathbf{w})$$

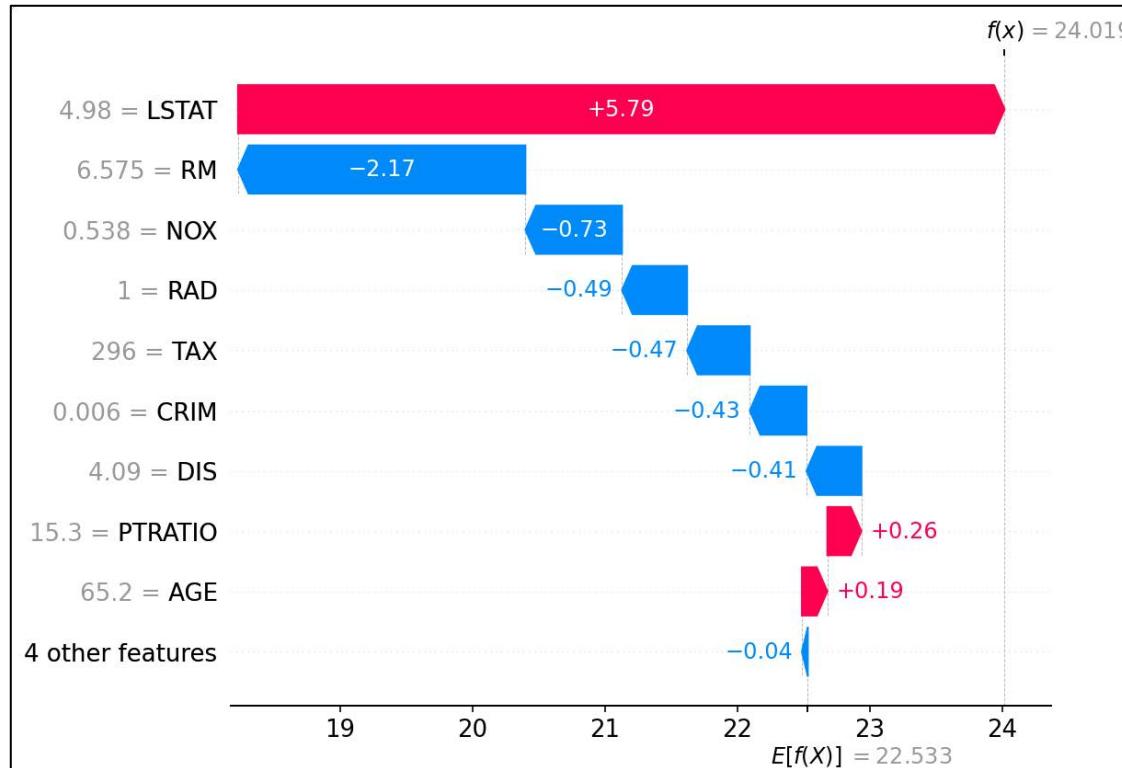
$$\pi(\mathbf{x}', \mathbf{z}') = \exp(-\|h(\mathbf{x}') - h(\mathbf{z}')\|_2^2 / \sigma^2)$$

$$\Omega(\mathbf{w}) = \|\mathbf{w}\|_2^2 \quad (\text{正則化項})$$

Feature Attribution法の代表例2： SHapley Additive exPlanations (SHAP)

[Lundberg+ 2017]

- 機械学習法（ブラックボックス）の予測値に対する各特徴量の貢献を
Shapley値の近似（SHAP値）を用いて説明する手法



<https://github.com/slundberg/shap>



“Lloyd Shapley in Stockholm 2012” by Bengt Nyman

Lloyd Shapley (1923~2016)
経済学者・数学者
2012年ノーベル経済学賞

SHAP理解のための準備：協力ゲーム理論

- 例：3名のプレイヤー(A, B, C)によるアルバイト [岡田 2011]
 - プレイヤーの組み合わせ方で報酬が変わる
 - 報酬 (=24) をプレイヤーにどのように分配するのがよいか？

▼表 6.1／参加者と報酬の関係

参加者	報酬
Aさん	6
Bさん	4
Cさん	2
Aさん・Bさん	20
Aさん・Cさん	15
Bさん・Cさん	10
Aさん・Bさん・Cさん	24



表は[森下 2021]より引用

貢献度に応じて報酬をプレイヤーに分配する方法 1/2

- ・貢献度を測るためにプレイヤーの限界貢献度を導入
 - 限界貢献度：プレイヤーがアルバイトに参加したとき、参加しなかった場合と比べて、報酬がいくら増えるか

例：Aさんの限界貢献度計算

- 誰がすでに参加しているかで変化

前提条件	現在の状況	限界貢献度
参加なし	Aのみ参加	$6 - 0 = 6$
Bのみ参加	A・Bが参加	$20 - 4 = 16$
Cのみ参加	A・Cが参加	$15 - 2 = 13$
B・Cが参加	A・B・Cが参加	$24 - 10 = 14$

▼表 6.1／参加者と報酬の関係

参加者	報酬
Aさん	6
Bさん	4
Cさん	2
Aさん・Bさん	20
Aさん・Cさん	15
Bさん・Cさん	10
Aさん・Bさん・Cさん	24

表は[森下 2021]より引用

貢献度に応じて報酬をプレイヤーに分配する方法 2/2

- ・アルバイトに参加する順番の影響を排除するために、全てのプレイヤーの参加順の組み合わせで限界貢献度の平均を計算
- ・この平均の限界貢献度をShapley値と呼ぶ

▼ 表 6.2／参加順を考慮した限界貢献度

表は[森下 2021]より引用

参加順	Aさんの限界貢献度	Bさんの限界貢献度	Cさんの限界貢献度
Aさん→Bさん→Cさん	6	14	4
Aさん→Cさん→Bさん	6	9	9
Bさん→Aさん→Cさん	16	4	4
Bさん→Cさん→Aさん	14	4	6
Cさん→Aさん→Bさん	13	9	2
Cさん→Bさん→Aさん	14	8	2

平均 11.5

平均 8

平均 4.5

各プレイヤーの
Shapley値

Shapley値の数式表現

j 番目のプレイヤーのShapley値 ϕ_j

$$\phi_j = \frac{1}{|J|!} \sum_{S \subseteq J \setminus \{j\}} |S|! (|J| - |S| - 1)! (\nu(S \cup \{j\}) - \nu(S)) = \mathbb{E}_{S \subseteq J \setminus \{j\}} [\nu(S \cup \{j\}) - \nu(S)]$$

↑
組み合わせの出現回数

↑
 S にプレイヤー j が
参加したときの限界貢献度

- J : 全プレイヤーの集合 $|J|$: 全プレイヤーの数
- $S \subseteq J \setminus \{j\}$: J からプレイヤー j を除いた集合に対する任意の部分集合
- $\nu(\cdot) \rightarrow \mathbb{R}$: 報酬関数
 - $\nu(S \cup \{j\})$: プレイヤー集合 S にプレイヤー j を追加したときの報酬
 - $\nu(S)$: プレイヤー集合 S に対する報酬

SHAPにおけるShapley値

= 特徴量を追加したときの予測値に対する平均限界貢献度

アルバイトゲームとSHAPの概念の対応関係

アルバイトゲーム	\Leftrightarrow	SHAP
プレイヤー	\Leftrightarrow	特徴量
報酬 $v(S)$	\Leftrightarrow	予測モデルの出力 $f(\mathbf{z}_S)$
S : プレイヤー集合	\Leftrightarrow	$\mathbf{z}_S = h(\mathbf{z}'_S)$: 特徴量集合 S についてのみ観測された元表現

予測モデルの出力 $f(\mathbf{z}_S)$ の計算

観測されていない特徴量集合 ($=\bar{S}$)について値が決まらないので期待値を求める

$$f(\mathbf{z}_S) = \mathbb{E}_{\mathbf{z}_{\bar{S}}|\mathbf{z}_S}[f(\mathbf{z})] \approx \mathbb{E}_{\mathbf{z}_{\bar{S}}}[f(\mathbf{z})]$$

特徴量間の独立性を仮定

SHAPにおけるShapley値の性質

性質1. 局所的な精度

説明モデル $g(x')$ は予測モデル $f(x)$ と一致

$$f(x) = g(x') = \mathbb{E}_x[f(x)] + \sum_{j \in J} w_j x'_j$$

性質2. 欠損性

ある解釈表現 x'_j が0のとき、対応するShapley値も0

$$x'_j = 0 \Rightarrow w_j = 0$$

性質3. 一貫性

予測モデルが変わり、特徴量の貢献度が
変われば、Shapley値も同様に変わる

2つの予測モデル f, f' に対して

$$f'(z) - f'(z'_{\setminus j}) \geq f(z) - f(z'_{\setminus j})$$

ならば、各々に対するShapley値 w, w' は

$$w'_j \geq w_j$$

大きすぎるShapley値の計算量に対するSHAPの工夫

特徴量 j のShapley値

$$\phi_j = \frac{1}{|J|!} \sum_{S \subseteq J \setminus \{j\}} |S|! (|J| - |S| - 1)! (E[f(h(\mathbf{z}'))|\mathbf{z}'_{S \cup \{j\}}] - E[f(h(\mathbf{z}'))|\mathbf{z}'_S])$$

- 単純にやると、 $\mathcal{O}(2^{|J|})$ 回予測モデルの出力を求める必要がある
 - 特徴量の数 $|J| = 100$ なら、 $\approx 10^{30}$ 回！
- 計算量を減らすための近似法
 - 予測モデル非依存、サンプリングによる近似 (KernelSHAP)
 - 木構造ベース予測モデルに特化した近似 (TreeSHAP)
 - ニューラルネットベース予測モデルに特化した近似 (DeepSHAP)

- 実は、カーネル $\pi(x', z')$ として Shapley カーネル を用いれば、
LIME と同様に重み付き最小二乗法で SHAP 値を求めることが可能

SHAP 値 w の推定

$$w^* = \operatorname{argmin}_w \sum_{z' \sim \mathcal{N}_{x'}} \pi(x', z')(f(h(z')) - w^\top z')^2$$

$h(z')$: 解釈表現 z' から
対応する元表現 z への変換

ただし、 $\pi(x', z')$ は Shapley カーネル :

$$\pi(x', z') = \frac{(|J| - 1)}{C(|J|, |z'|) |z'| (|J| - |z'|)}$$

$C(|J|, |z'|)$:
 $|J|$ から $|z'|$ 取る組み合わせの数

KernelSHAP

Shapleyカーネルを用いた重み付き最小二乗法

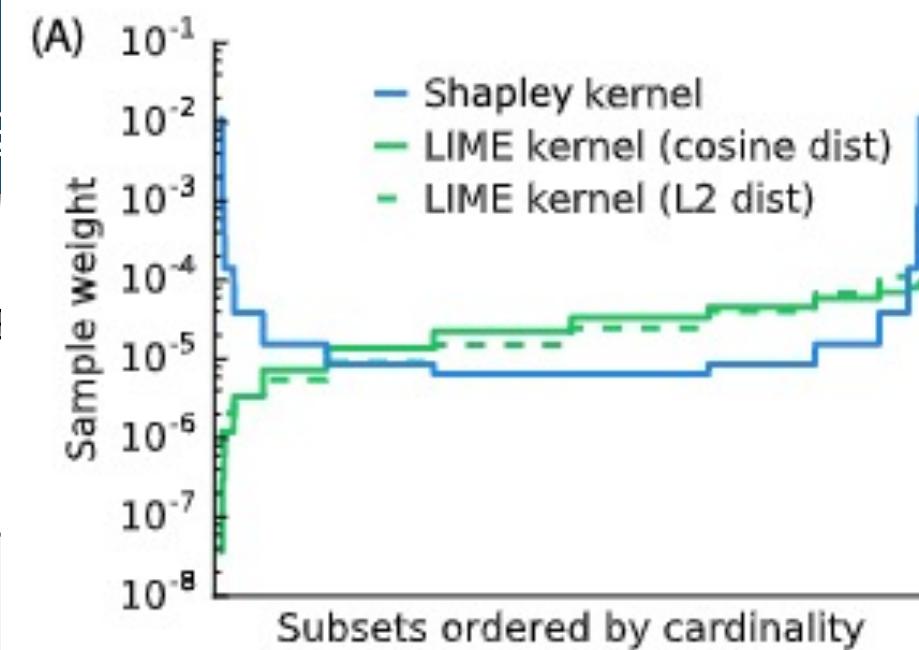
- 実は、カーネル $\pi(x', z')$ として Shapley カーネルを LIME と同様に重み付き最小二乗法で SHAP 値を求め

SHAP 値 w の推定

$$w^* = \operatorname{argmin}_w \sum_{z' \sim \mathcal{N}_{x'}} \pi(x', z')(f(h(z')) - w^\top z')^2$$

ただし、 $\pi(x', z')$ は Shapley カーネル：

$$\pi(x', z') = \frac{(|J| - 1)}{C(|J|, |z'|) |z'| (|J| - |z'|)}$$



$h(z')$ ：解釈表現 z' から
対応する元表現 z への変換

$C(|J|, |z'|)$ ：
 $|J|$ から $|z'|$ 取る組み合わせの数

Shapleyカーネルの妥当性

時間がないので…

[Lundberg+ 2017]のSupplimental Materialを読んでください！

Shapley Kernel Proof

November 3, 2017

The Shapley kernel is the sample weight given to each binary vector $z' \in \{0, 1\}^M$:

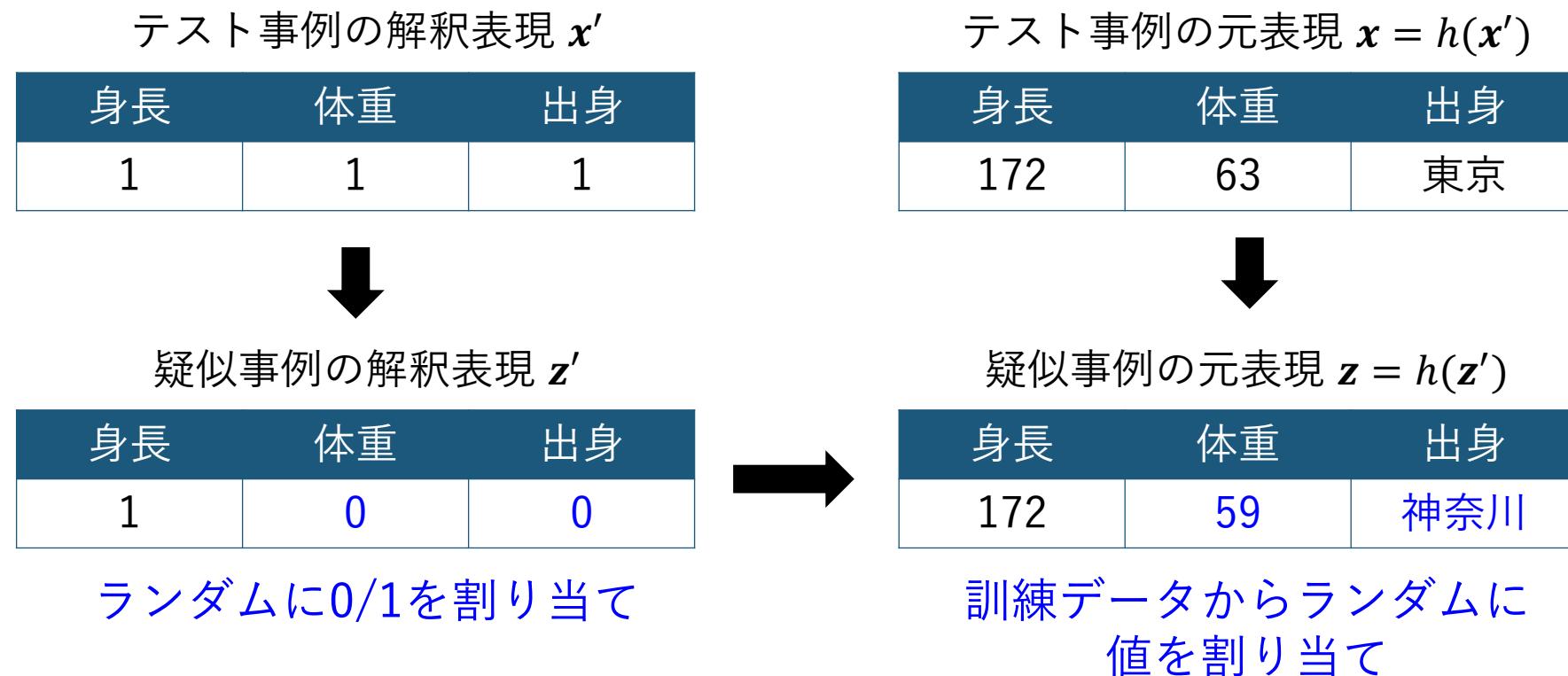
$$k(z') = k(M, s) = \frac{M - 1}{\binom{M}{s} s(M - s)}$$

where $s = |z'|$, the number of ones in z' .

Let X be the matrix of all possible binary vectors of length M with 2^M rows and M columns. We use the

表形式データに対する疑似事例の生成

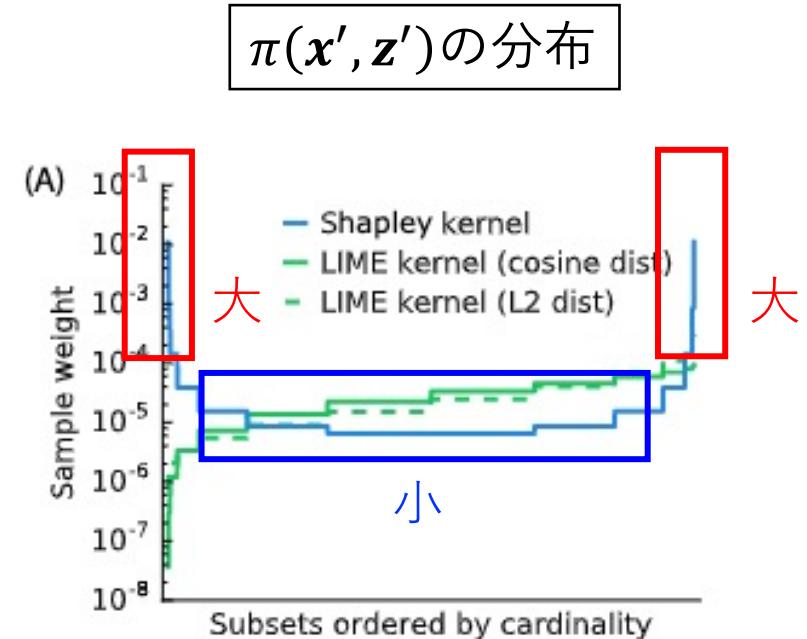
- LIMEとは異なり、 $z'_j = 0$ のときは訓練データ中の特徴量 j の値から一つ選ぶ



少ない疑似事例でSHAP値を求める工夫

$$\text{SHAP値の計算式: } \mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \sum_{\mathbf{z}' \sim \mathcal{N}_{x'}} \pi(\mathbf{x}', \mathbf{z}') (f(h(\mathbf{z}')) - \mathbf{w}^\top \mathbf{z}')^2$$

- 解釈表現 \mathbf{z}' がほとんど 0 or 1 から成るときに
カーネル $\pi(\mathbf{x}', \mathbf{z}')$ が大きな値
 - これらが係数（SHAP値） \mathbf{w} の学習への影響大
 - そのような特徴の組み合わせを優先的に使うことでSHAP値の計算式を近似



[Lundberg+ 2017]のFig. 2から引用

K 個の疑似事例を用いる場合のサンプリング法

1. $m = 1$ とする
2. m 個の特徴と $|J| - m$ 個の特徴で 1 or 0 になる疑似解釈事例 \mathbf{z}' を全て生成
3. ここまで生成した疑似事例の数が K 未満なら、 $m = m + 1$ して 2 に戻る
4. カーネル $\pi(\mathbf{x}', \mathbf{z}')$ を再計算

生成される疑似事例の解釈表現 \mathbf{z}'

$m = 1$ のとき $\rightarrow 2 \times |J|$ 個

$1\ 0\ 0 \cdots 0\ 0\ 0$	…	$0\ 0\ 0 \cdots 0\ 0\ 1$
$0\ 1\ 1 \cdots 1\ 1\ 1$	…	$1\ 1\ 1 \cdots 1\ 1\ 0$

$m = 2$ のとき $\rightarrow 2 \times C(|J|, 2)$ 個

$1\ 1\ 0 \cdots 0\ 0\ 0$	…	$0\ 0\ 0 \cdots 0\ 1\ 1$
$0\ 0\ 1 \cdots 1\ 1\ 1$	…	$1\ 1\ 1 \cdots 1\ 0\ 0$

LIMEとSHAPのまとめ

- LIME：予測モデルの出力を線形回帰モデルの係数で説明
- SHAP：Shapley値に基づいて、予測モデルにおける特徴量の貢献度を説明
- LIMEと(Kernel)SHAPは、重み付き最小二乗法で解くことが可能
 - ただし、カーネル関数と正則化項が異なる

目的: $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \sum_{\mathbf{z}' \sim \mathcal{N}_{x'}} \pi(\mathbf{x}', \mathbf{z}') (f(h(\mathbf{z}')) - \mathbf{w}^\top \mathbf{z}')^2 + \lambda \Omega(\mathbf{w})$

LIME

$$\pi(\mathbf{x}', \mathbf{z}') = \exp(-\|h(\mathbf{x}') - h(\mathbf{z}')\|_2^2 / \sigma^2)$$

$$\Omega(\mathbf{w}) = \|\mathbf{w}\|_2^2$$

(Kernel)SHAP

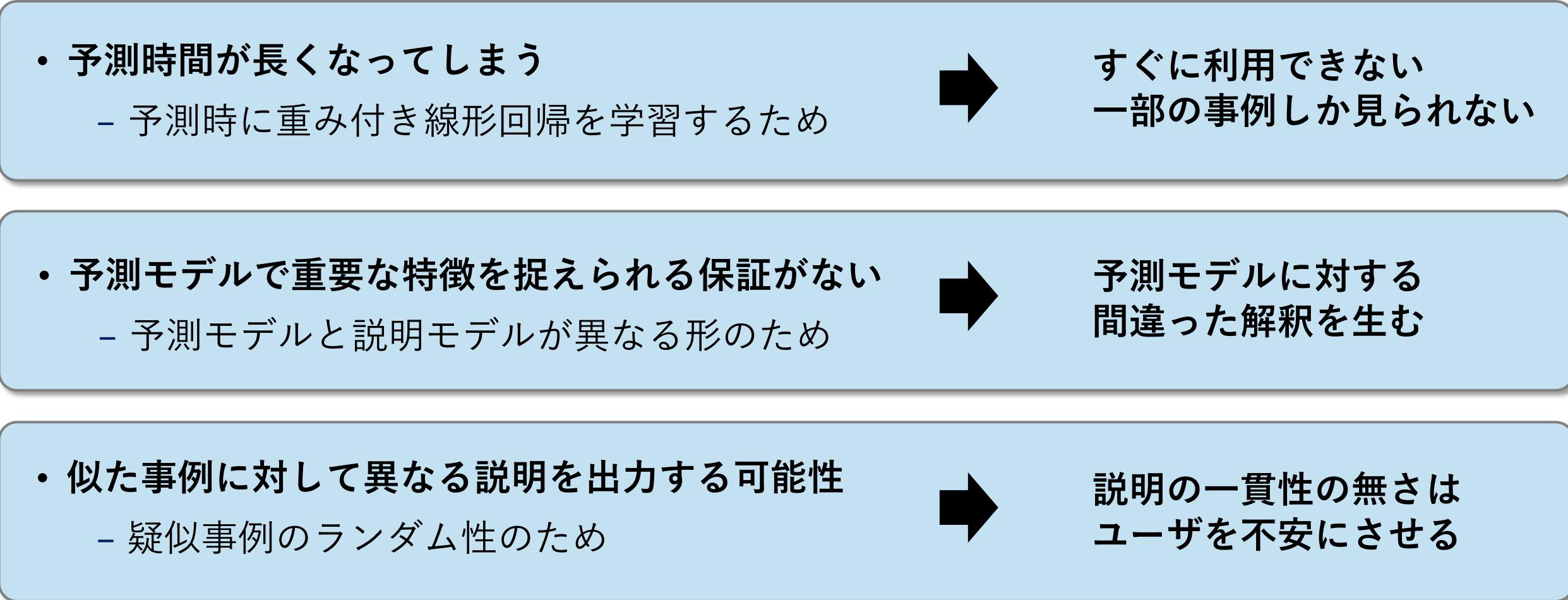
$$\pi(\mathbf{x}', \mathbf{z}') = \frac{(|J| - 1)}{C(|J|, |\mathbf{z}'|)|\mathbf{z}'|(|J| - |\mathbf{z}'|)}$$

$$\Omega(\mathbf{w}) = 0$$

Pythonによる実装の紹介

Feature Attribution法に関する 最近の研究

LIME・SHAPの懸念点

- 予測時間が長くなってしまう
 - 予測時に重み付き線形回帰を学習するため
 - 予測モデルで重要な特徴を捉えられる保証がない
 - 予測モデルと説明モデルが異なる形のため
 - 似た事例に対して異なる説明を出力する可能性
 - 疑似事例のランダム性のため
- 
- すぐに利用できない
一部の事例しか見られない
 - 予測モデルに対する
間違った解釈を生む
 - 説明の一貫性の無さは
ユーザを不安にさせる

LIME・SHAPの懸念点

- 予測時間が長くなってしまう
 - 予測時に重み付き線形回帰を学習するため
- 予測モデルで重要な特徴を捉えられる保証がない
 - 予測モデルと説明モデルが異なる形のため
- 似た事例に対して異なる説明を出力する可能性
 - 疑似事例のランダム性のため

速度

忠実性

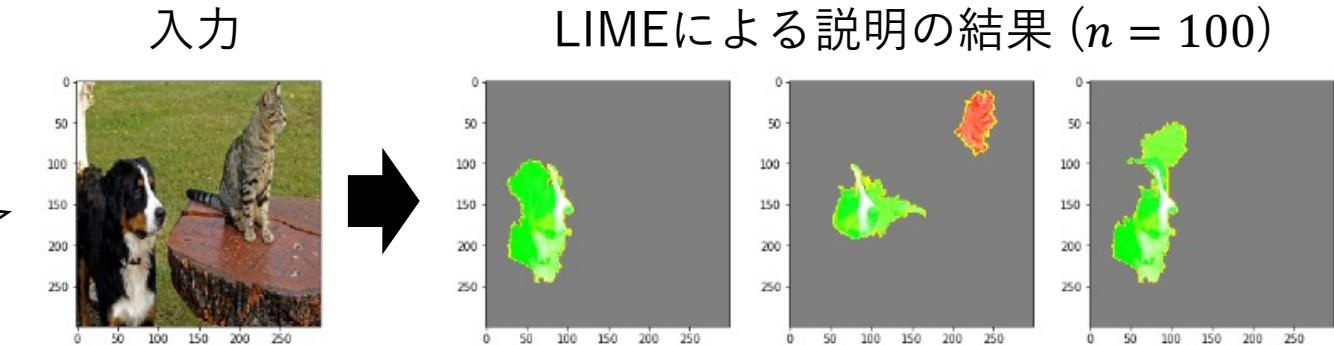
一貫性

LIMEをベイズ化することで一貫性を上げる Bayesian LIME (BayLIME)

一貫性

[Zhao+ 2021]

疑似事例が少ないとき
LIMEは説明の一貫性がない



事前分布を入れることで疑似事例数が少なくとも一貫した説明が出やすくする

$$\text{尤度: } Pr(\mathbf{y} | \boldsymbol{\beta}, \mathbf{X}, \alpha) = \prod_{i=1}^n \mathcal{N}(y_i | \mathbf{x}_i \boldsymbol{\beta}, \alpha^{-1})$$

$$\text{事前分布: } Pr(\boldsymbol{\beta} | \boldsymbol{\mu}_0, \mathbf{S}_0) = \mathcal{N}(\boldsymbol{\beta} | \boldsymbol{\mu}_0, \mathbf{S}_0)$$

$$\begin{aligned}\text{事後分布: } Pr(\boldsymbol{\beta} | \mathbf{y}, \mathbf{X}, \alpha, \boldsymbol{\mu}_0, \mathbf{S}_0) &\propto Pr(\mathbf{y} | \boldsymbol{\beta}, \mathbf{X}, \alpha) Pr(\boldsymbol{\beta} | \boldsymbol{\mu}_0, \mathbf{S}_0) \\ &= \mathcal{N}(\boldsymbol{\beta} | \boldsymbol{\mu}_n, \mathbf{S}_n)\end{aligned}$$

$$\text{事後分布の平均: } \boldsymbol{\mu}_n = \mathbf{S}_n^{-1} \lambda \mathbf{I}_m \boldsymbol{\mu}_0 + \mathbf{S}_n^{-1} \alpha \mathbf{X}^\top \mathbf{W} \mathbf{X} \boldsymbol{\beta}_{MLE}$$

パラメータ λ と α の大きさで
事前平均 $\boldsymbol{\mu}_0$ と最尤推定量 $\boldsymbol{\beta}_{MLE}$ の
影響の大きさが決まる

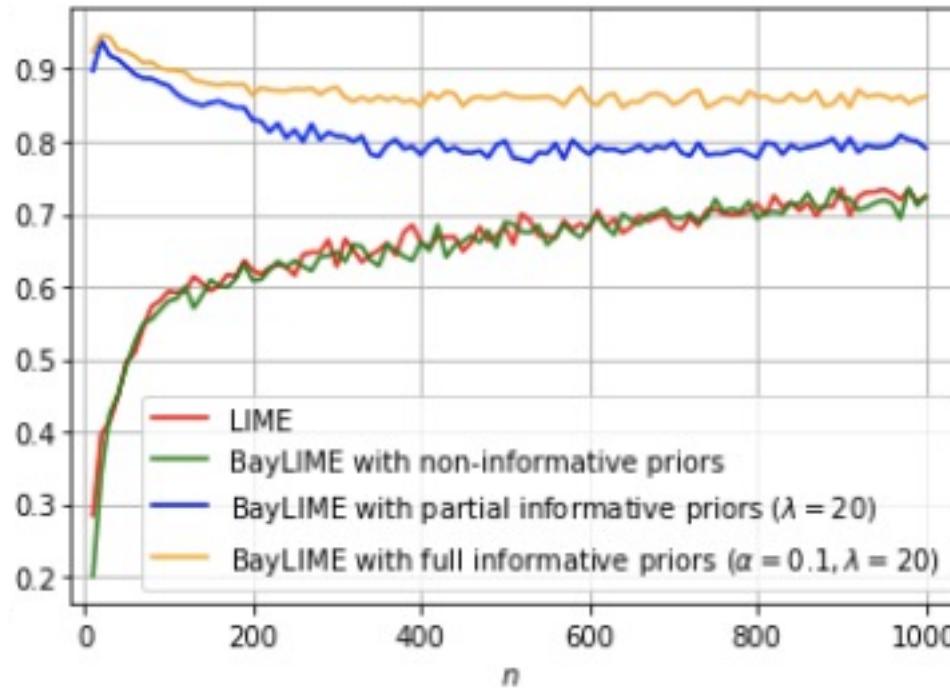
LIMEをベイズ化することで一貫性を上げる Bayesian LIME (BayLIME)

一貫性

[Zhao+ 2021]

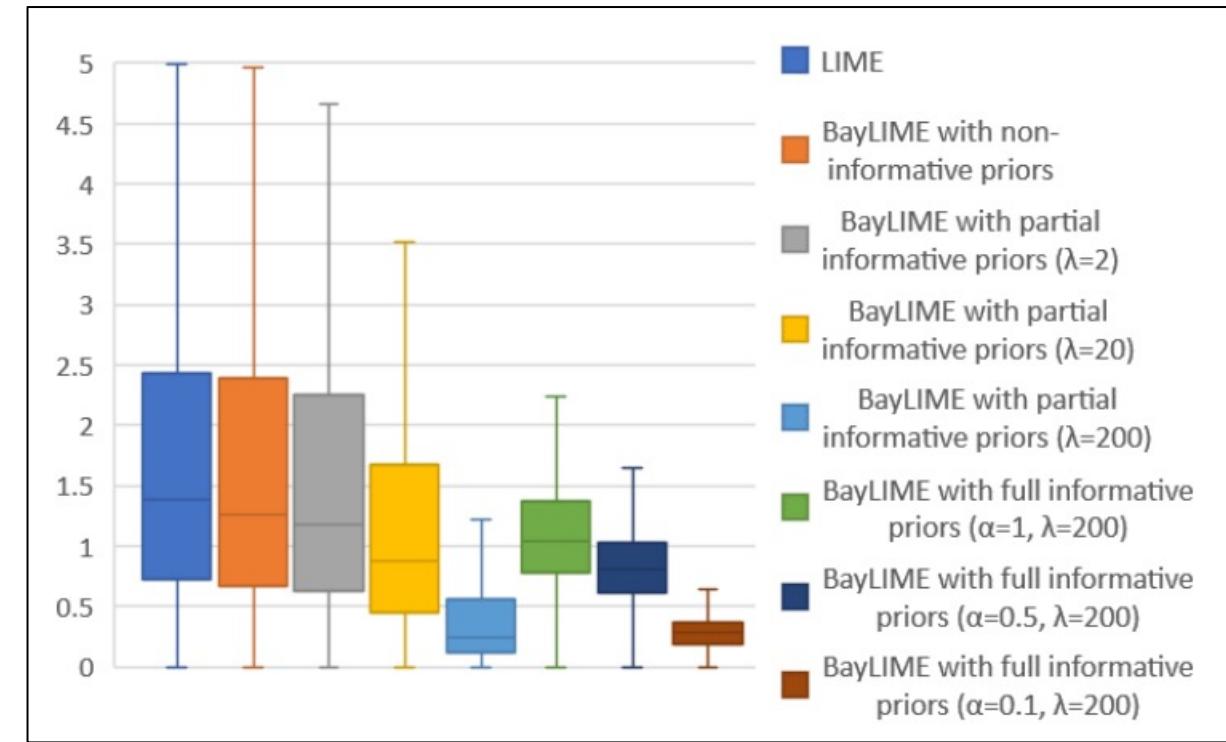
実験結果

同じテスト事例に対する説明の一貫性



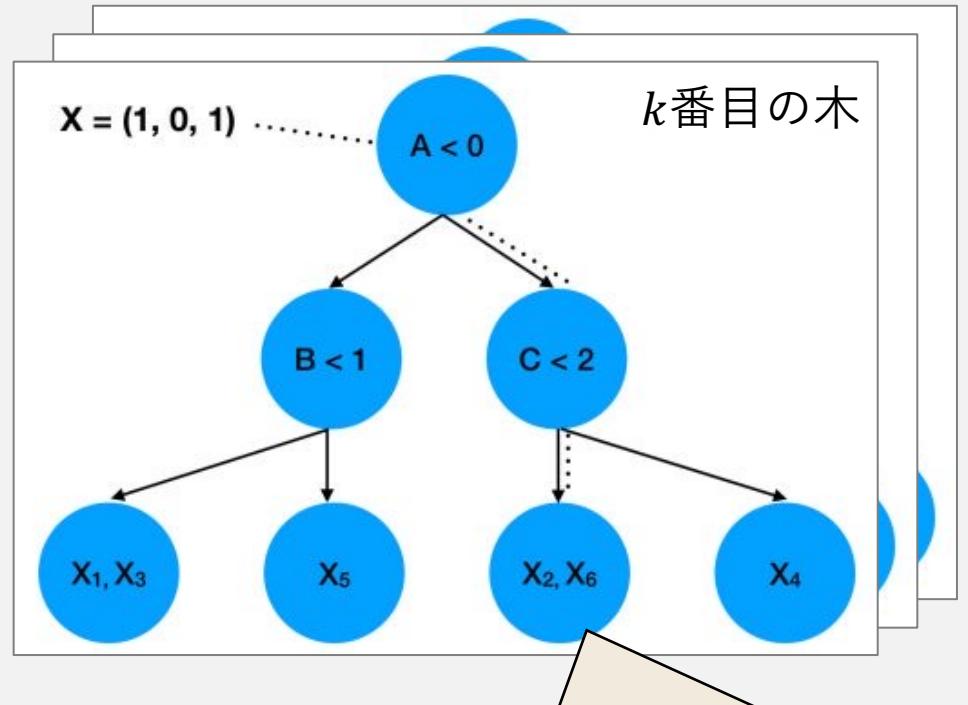
μ_0, λ, α について適切な事前知識があれば
一貫性のある説明を出力可能

カーネルのバンド幅パラメータに対する頑健さ



μ_0, λ, α について適切な事前知識があれば
どんなバンド幅の値でも説明は安定

1. アンサンブル木を学習し、テスト事例 X と同じ葉に到達する訓練事例をカウント



2. 同じ葉に所属する回数に応じて訓練事例を重み付け

訓練事例 X_i の重み

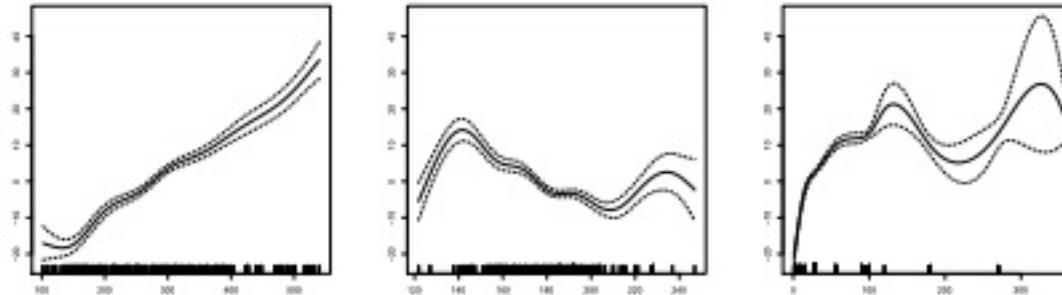
$$w_i = \frac{X_i \text{ と } X \text{ が同じ葉に属する回数}}{X \text{ が属する葉の数}}$$

3. 重み付けした訓練事例を用いて重み付き線形回帰モデルを学習

$$\hat{\beta}_x = \operatorname{argmin}_{\beta} \sum_{i=1}^n w_i (\beta^T x_i - y_i)^2.$$

図は <https://blog.ml.cmu.edu/2019/07/13/towards-interpretable-tree-ensembles/> から引用

- 各特徴量が予測値への貢献を決める関数 f_j を持つ
 - 関数 f_j の形を見れば、特徴量 x_j がどのように予測値に貢献するのか分かる



一般化加法モデル (GAM)

$$y = \beta_0 + \sum_{j \in J} f_j(x_j)$$

- 関数 f_j の形：平滑化スプラインで決定

特徴量のペアワイズ項も考慮するGAMモデル GA²M / Explainable Boosting Machine (EBM)

速度

忠実性

[Lou+ 2013, Nori+ 2019]

- 特徴量の各ペアにも関数を置くことで、
特徴量の相互作用で予測値が変わる場合も
捉えられる

結果的に
予測性能が上がる

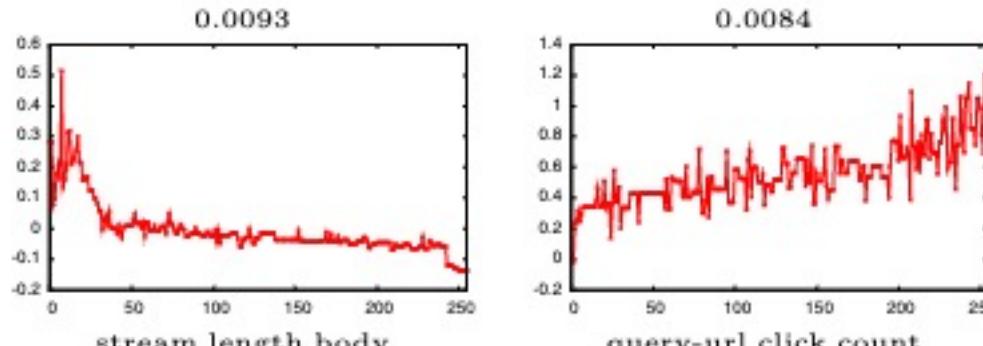
Model	Mean
Linear Regression	1.52±0.79
GAM	1.00±0.00
GA ² M FAST	0.84±0.20

ペアワイズ項を考慮したGAM

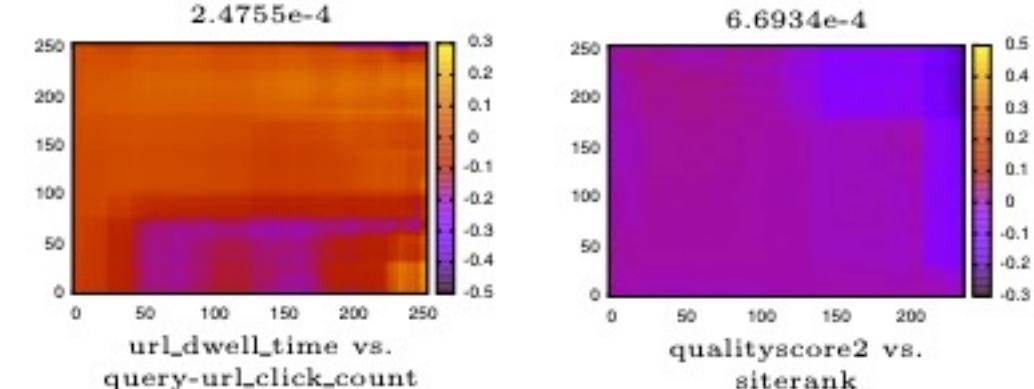
$$y = \beta_0 + \sum_{j \in J} f_j(x_j) + \sum_{j \neq k} f_{j,k}(x_j, x_k)$$

学習された特徴量の関数

個々の特徴量



ペアワイズ特徴量



特徴量のペアワイス項も考慮するGAMモデル GA²M / Explainable Boosting Machine (EBM)

速度

忠実性

[Lou+ 2013, Nori+ 2019]

基本の学習アルゴリズム

Algorithm 1 GA²M Framework

```
1:  $\mathcal{S} \leftarrow \emptyset$ 
2:  $\mathcal{Z} \leftarrow \mathcal{U}^2$ 
3: while not converge do
4:    $F \leftarrow \arg \min_{F \in \mathcal{H}^1 + \sum_{u \in \mathcal{S}} \mathcal{H}_u} \frac{1}{2} E[(y - F(\mathbf{x}))^2]$ 
5:    $R \leftarrow y - F(\mathbf{x})$ 
6:   for all  $u \in \mathcal{Z}$  do
7:      $F_u \leftarrow E[R|x_u]$ 
8:      $u^* \leftarrow \arg \min_{u \in \mathcal{Z}} \frac{1}{2} E[(R - F_u(x_u))^2]$ 
9:      $\mathcal{S} \leftarrow \mathcal{S} \cup \{u^*\}$ 
10:     $\mathcal{Z} \leftarrow \mathcal{Z} - \{u^*\}$ 
```

\mathcal{S} : 選択済み特徴量集合

\mathcal{Z} : 特徴量ペアの集合

現在の \mathcal{S} を用いてGA²Mを最適化
残差 R を求める

各特徴量ペアにおいて予測器 F_u を学習
最も残差を減らせる特徴量ペアを選択

u^* を選択済み特徴量集合 \mathcal{S} に追加

※予測時は高速だが学習には時間がかかる

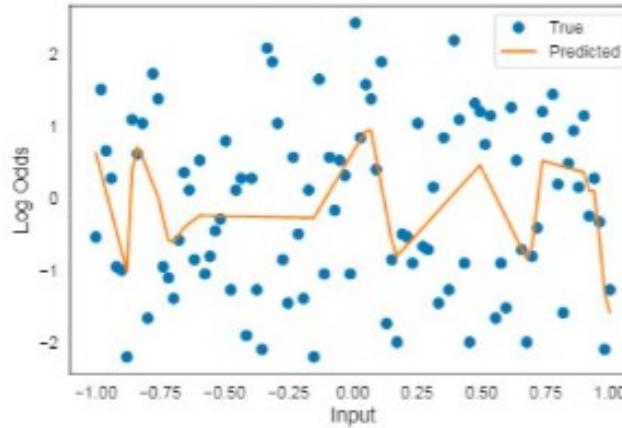
Neural Additive Model (NAM)

[Agarwal+ 2021]

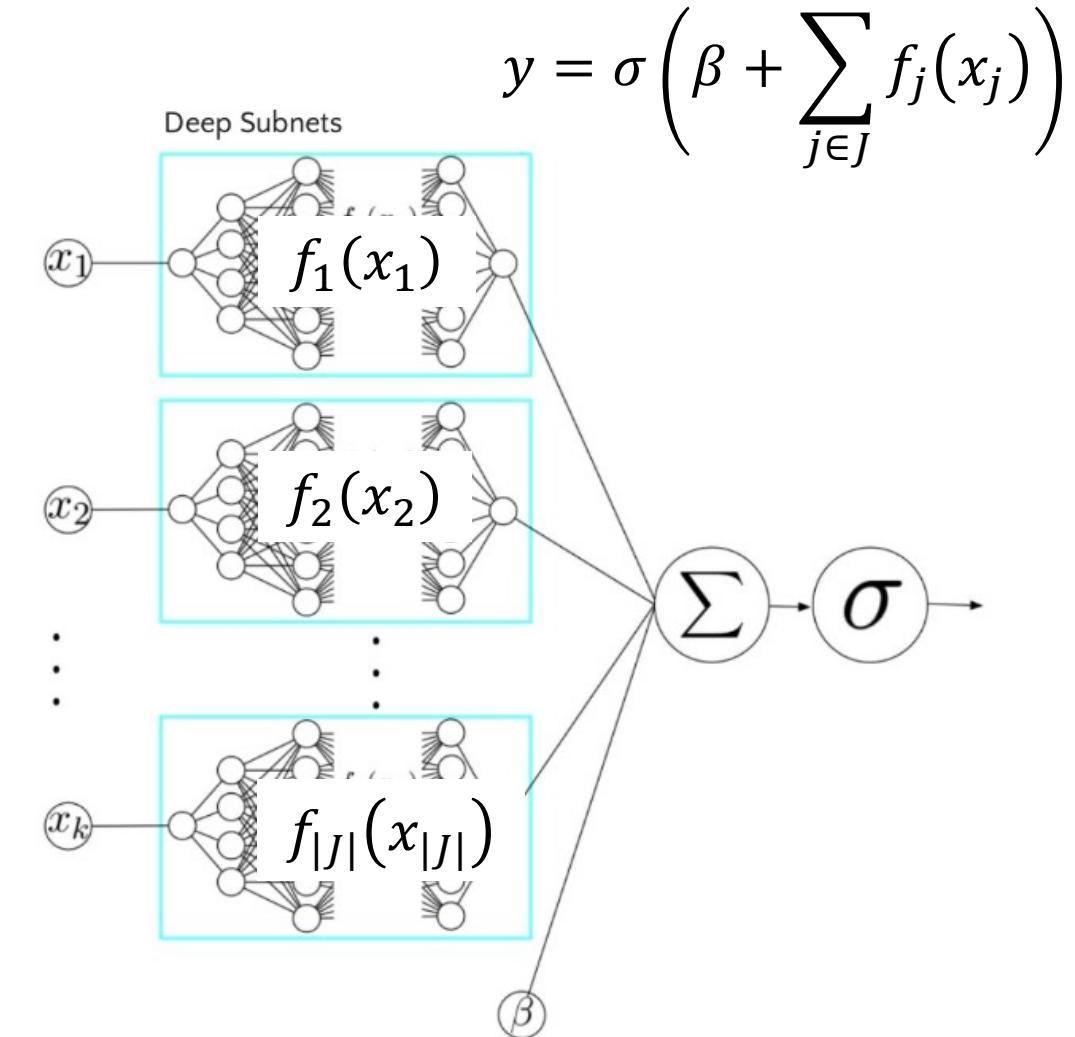
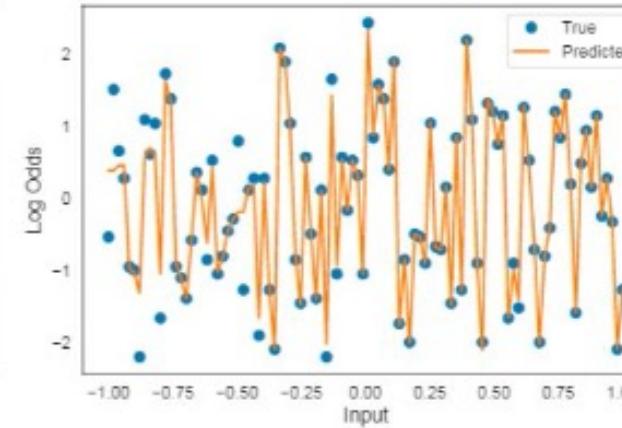
- GAMの各特徴量の関数 $f_j(x_j)$ をNN化
 - 学習速度向上
- EBMのように関数の値が大きく変化できるように、特別な活性化関数を用いる

$$\text{ExU: } h(x) = f(e^w * (x - b))$$

NAM w/ ReLU



NAM w/ ExU



Neural Additive Model (NAM)

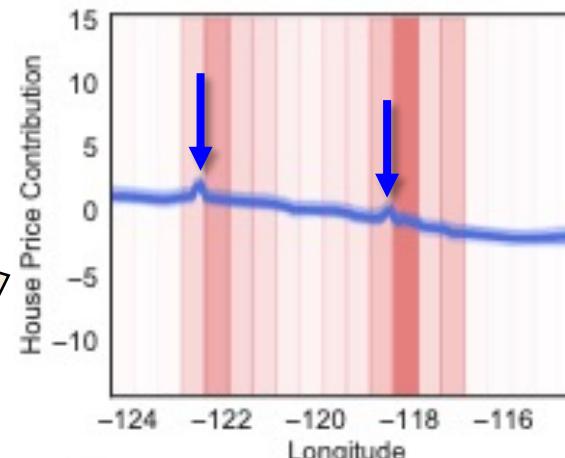
[Agarwal+ 2021]

予測性能

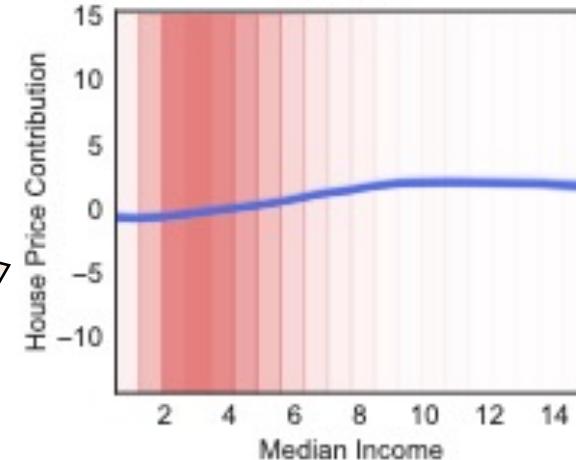
Model	MIMIC-II (AUC)	Credit (AUC)	CA Housing (RMSE)	FICO (RMSE)
Log./Linear Reg.	0.791 ± 0.007	0.975 ± 0.010	0.728 ± 0.015	4.344 ± 0.056
CART	0.768 ± 0.008	0.956 ± 0.004	0.720 ± 0.006	4.900 ± 0.113
NAMs	0.830 ± 0.008	0.980 ± 0.002	0.562 ± 0.007	3.490 ± 0.081
EBMs	0.835 ± 0.007	0.976 ± 0.009	0.557 ± 0.009	3.512 ± 0.095
XGBoost	0.844 ± 0.006	0.981 ± 0.008	0.532 ± 0.014	3.345 ± 0.071
DNNs	0.832 ± 0.009	0.978 ± 0.003	0.492 ± 0.009	3.324 ± 0.092

特徴量の関数の可視化（住宅価格データ）

サンフランシスコや
カリフォルニアがある経度のところで
価格が跳ね上がる
ことを捉えている



収入の中央値に
対しては
線形に変化



予測と説明が一致するモデル（吉川の研究）

ニューラルネット特化型モデル

Yuya Yoshikawa, and Tomoharu Iwata,

“Neural Generators of Sparse Local Linear Models for Achieving Both Accuracy and Interpretability,” Information Fusion, 2022.

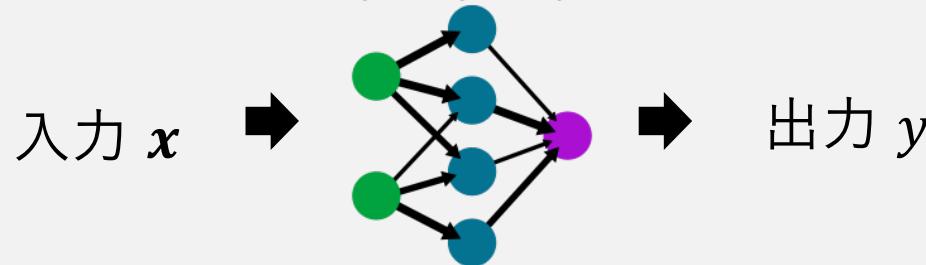
ガウス過程回帰特化型モデル

Yuya Yoshikawa, and Tomoharu Iwata,

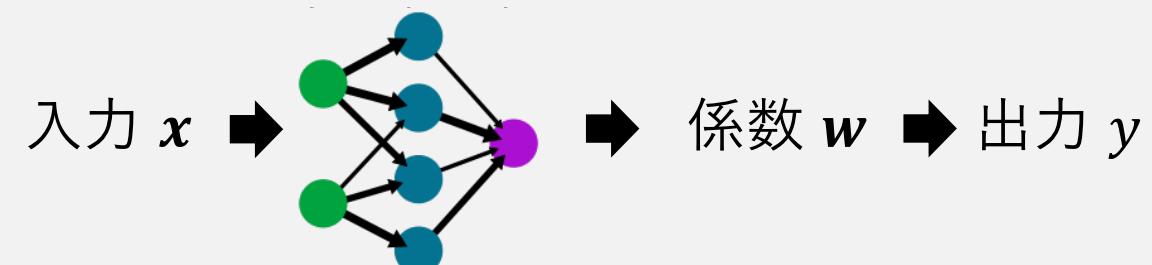
“Gaussian Process Regression With Interpretable Sample-wise Feature Weights,” IEEE Transactions on Neural Networks and Learning Systems, 2021.

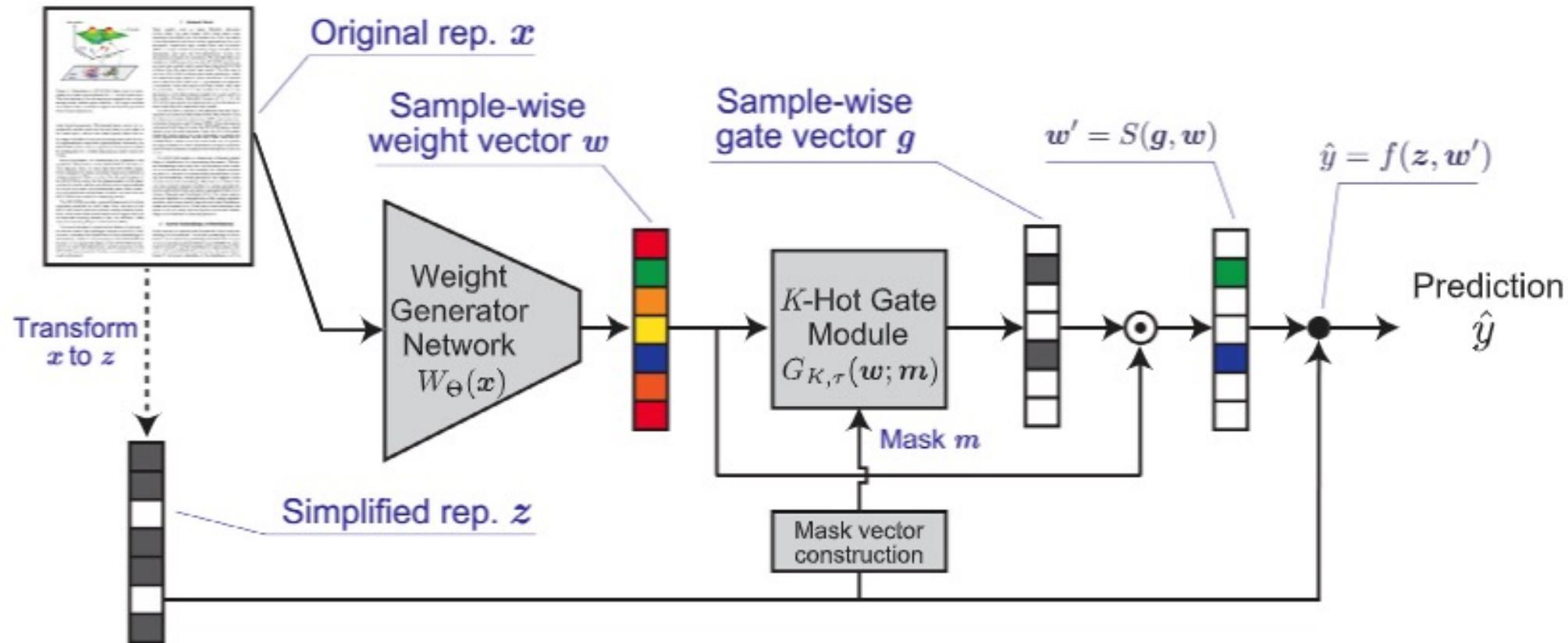
- NGSLLのキーポイント
 - 局所線形モデルの係数 w を生成するニューラルネットを作ることで、事例ごとに異なる解釈可能な係数が推定できる
 - ニューラルネットなので予測性能は高いまま維持できる
 - 予測と説明のギャップがない (= 忠実性が高い)

通常のニューラルネット



NGSLL

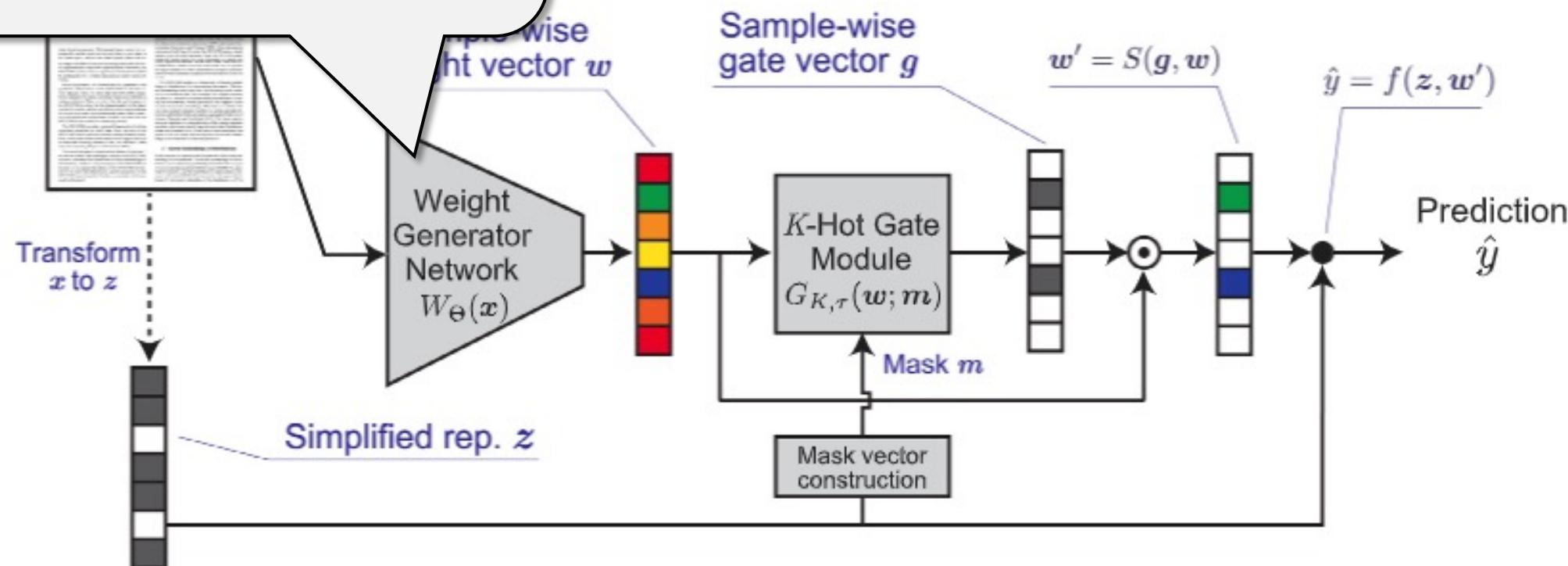


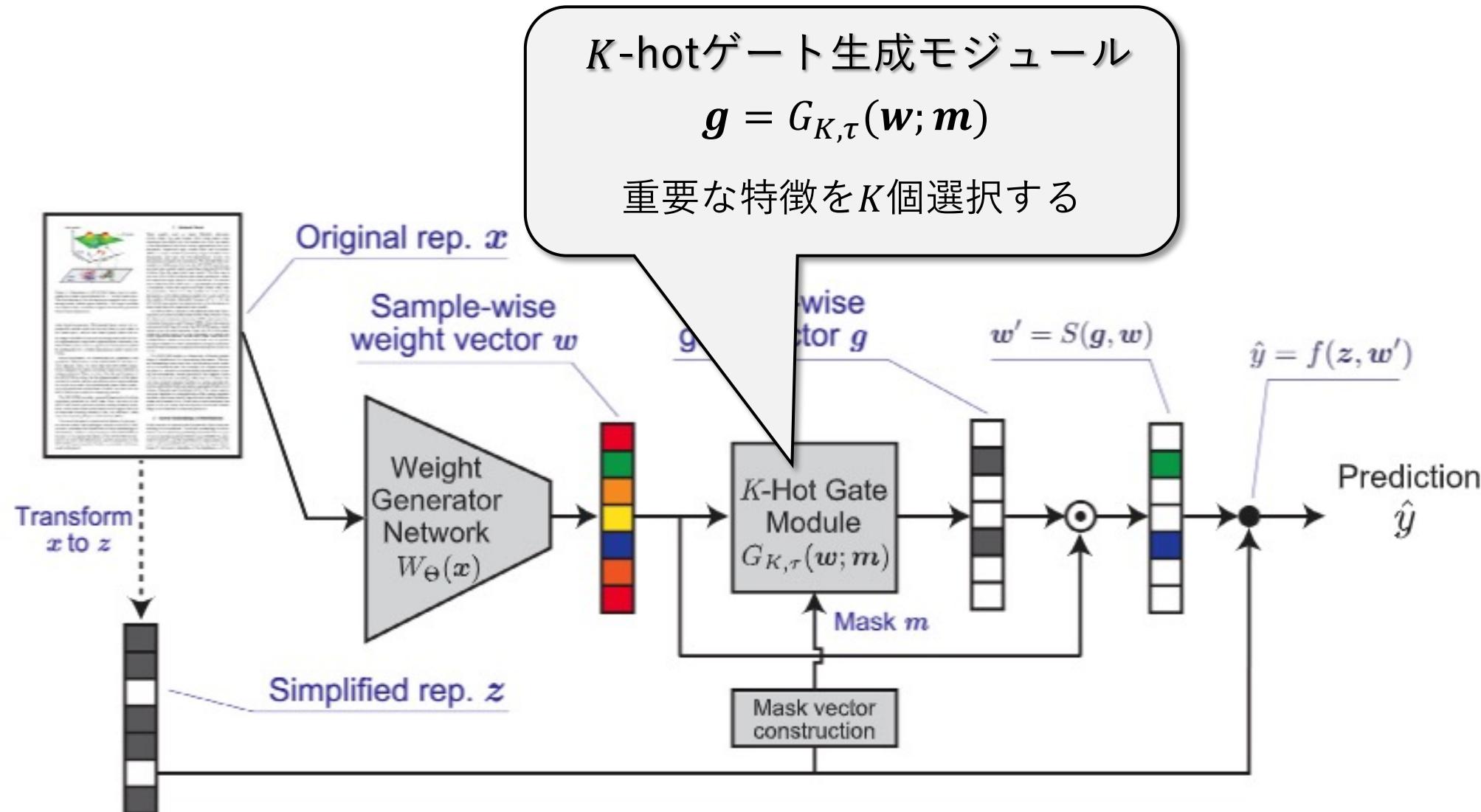


係数生成ネットワーク

$$\mathbf{w} = W_{\Theta}(\mathbf{x})$$

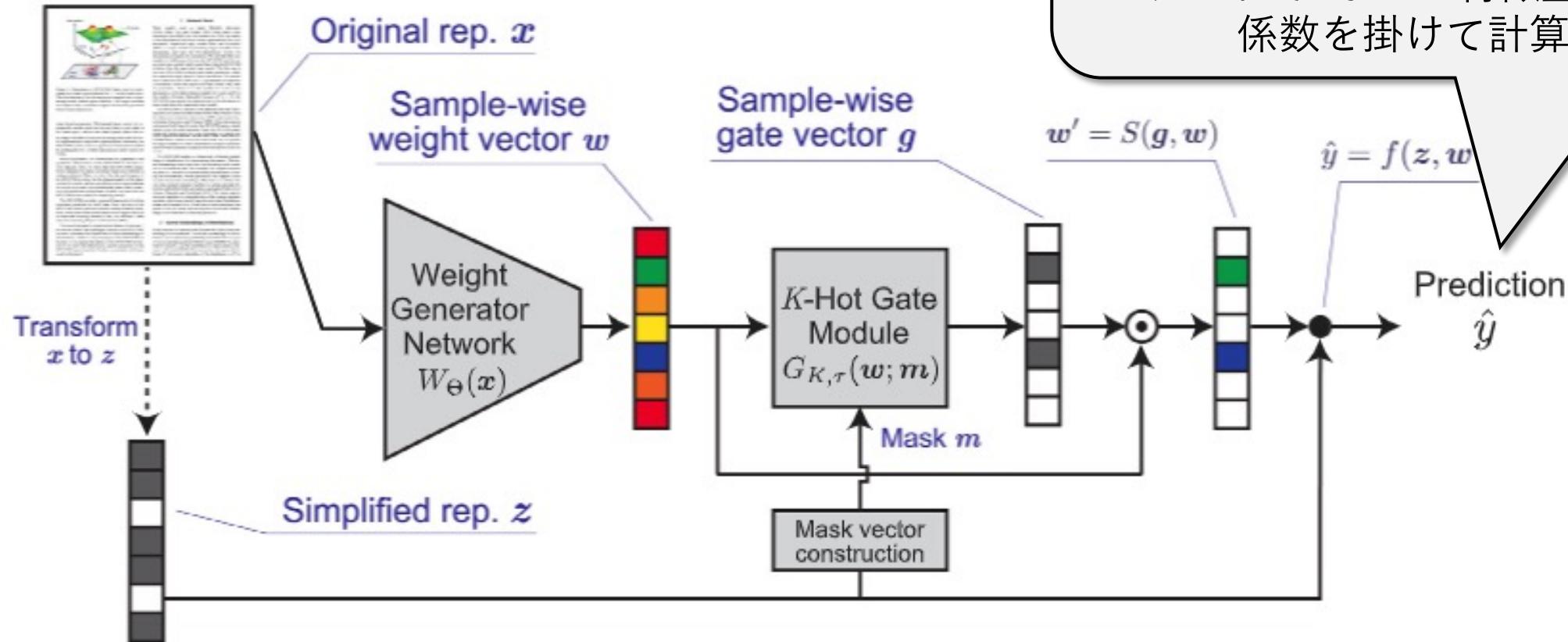
既存のNN (例: CNN)
を使用可能





予測式 $y = \sum_{j \in J} w_j g_j z_j$

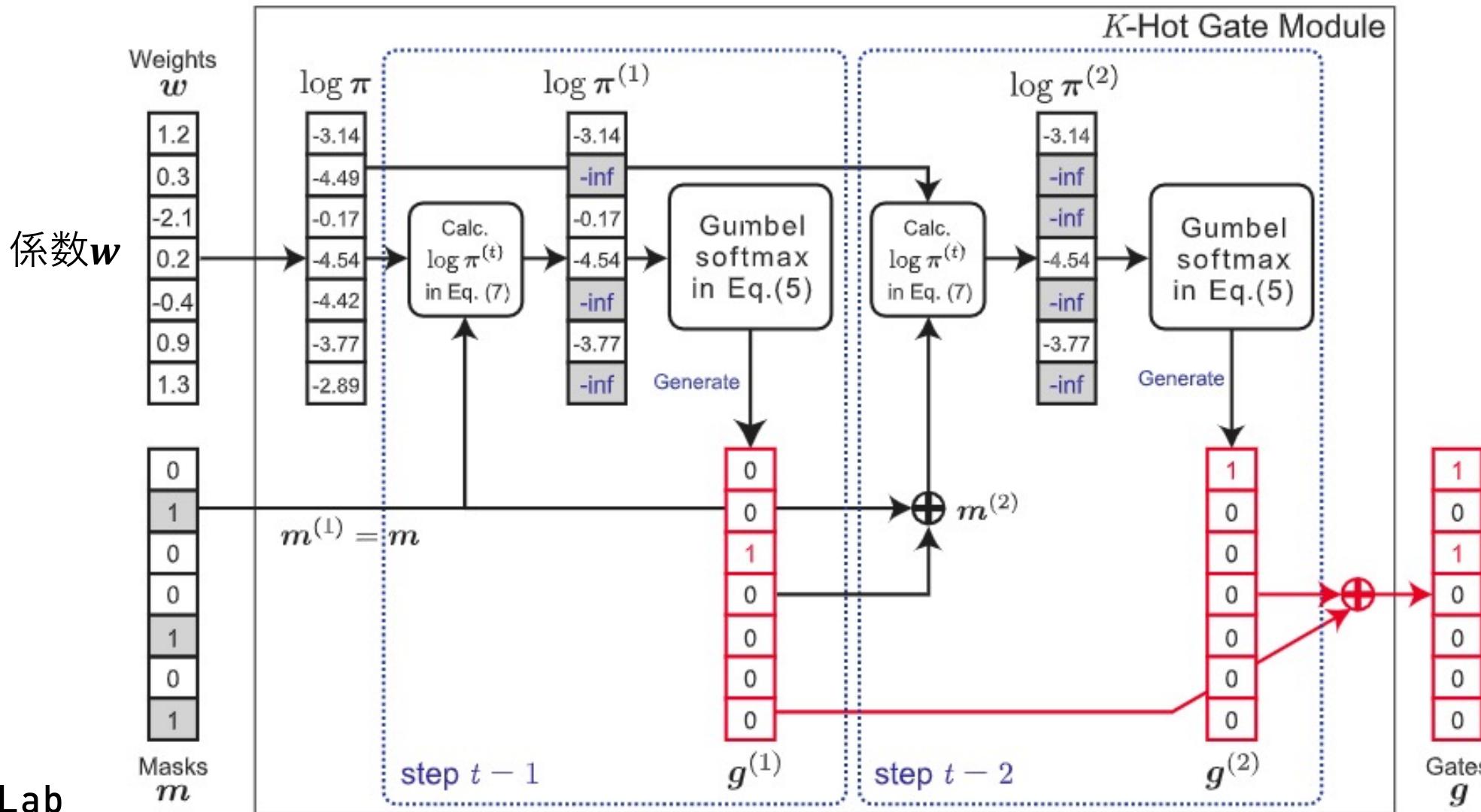
ゲートで1が立つ特徴量のみ
係数を掛けて計算



スパースな係数を生成する仕組み

重要な係数を一つづつ選び、それだけ通す0/1ゲートベクトルを作る

Gumbel softmaxをK回適用（ただし、一度選ばれた特徴は二度と選ばない）



ニューラルネット特化型モデル スパースな局所線形モデルのニューラル生成器 (NGSLL)

速度

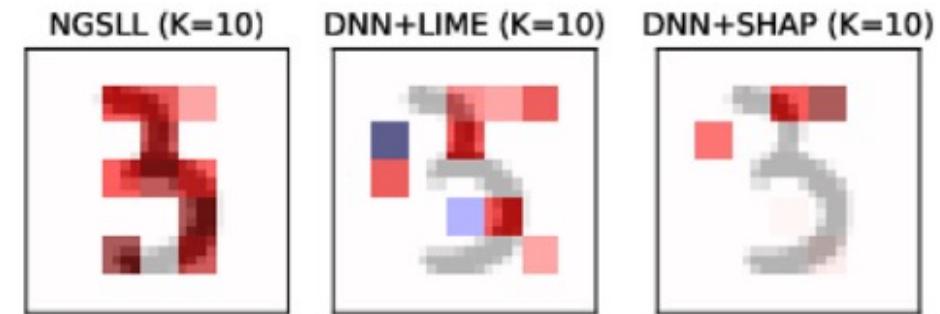
忠実性

[Yoshikawa+ 2022]

係数 w の生成結果

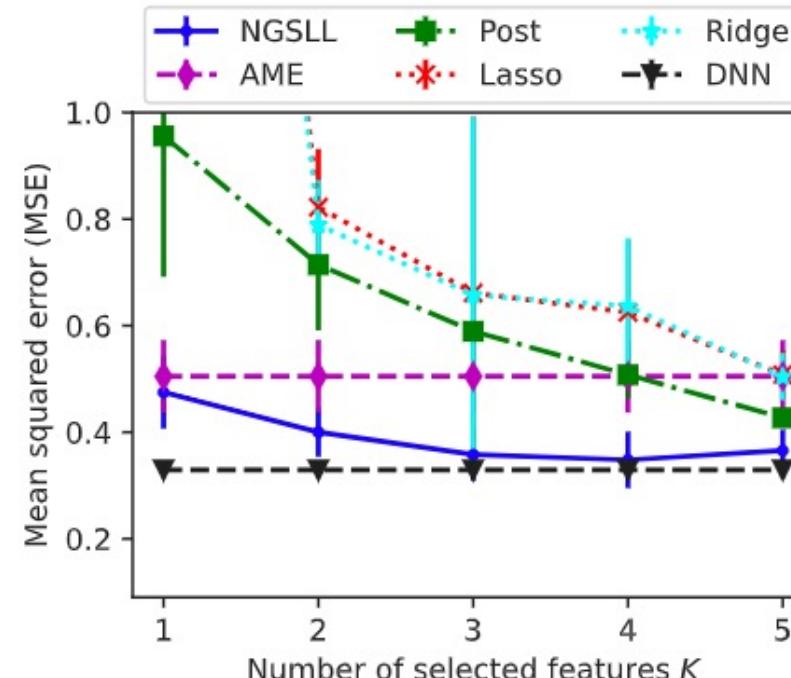
白ピクセルは0、黒ピクセルは1の特徴量の値
→ 黒ピクセルに対する係数のみが予測値に影響

NGSLLは適切に係数を割り当てている



予測性能

NGSLLは
少ない特徴量でも
低い予測誤差



予測時の計算時間

	Test
NGSLL ($K = 1$)	0.66 (± 0.01)
NGSLL ($K = 5$)	0.72 (± 0.01)
NGSLL ($K = 10$)	0.82 (± 0.01)
AME	3.11 (± 0.05)
DNN+LIME	11 553.09 (± 34.36)
DNN+SHAP	10 861.75 (± 7.93)
Post	0.13 (± 0.01)

予測時、NGSLLはForward計算のみ
なのでかなり高速

- 局所説明付きガウス過程回帰 (GPX)

- 局所線形モデルの係数をガウス過程から生成された関数で決定するようにモデル化
- 従来のガウス過程回帰と同等の予測性能
- 予測と説明のギャップがない (= 忠実性が高い)

ガウス過程回帰 (GP)

$$y = f(x) + \epsilon$$

$$f(\cdot) \sim \mathcal{N}(0, K)$$

自己説明型ガウス過程回帰 (GPX)

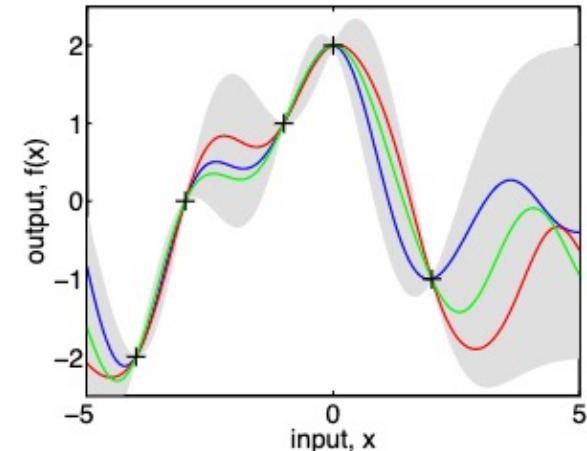
$$y = f(x) + \epsilon$$

$$f(x) = \sum_{j \in J} w_j(x) x_j \quad w_j(\cdot) \sim \mathcal{N}(0, K)$$

GPXの利点

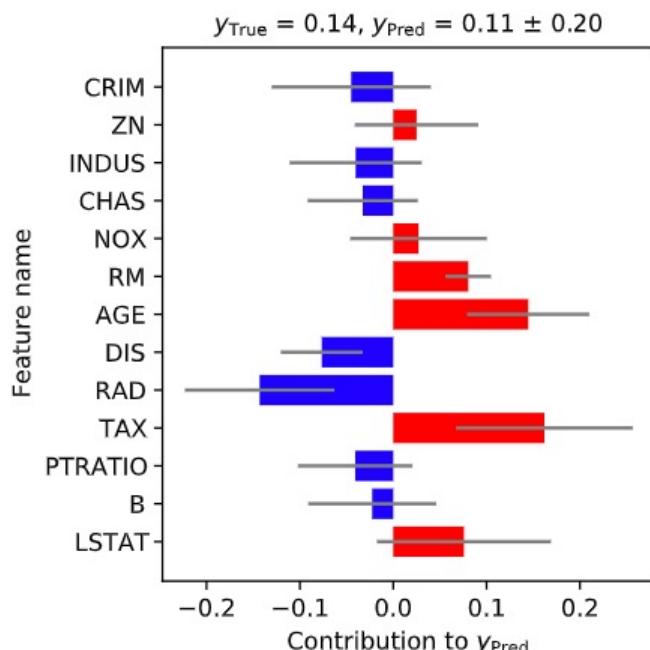
利点 1

- ガウス過程を仮定すると、係数 w が近傍事例で滑らかに変化するようになる → 一貫性が向上



利点 2

- 係数 w の事後分布が得られる
 - 不確実性の評価が可能



忠実性スコア

	GPX (ours)	GPR/SE+LIME	GPR/SE+SHAP
Digits	0.888 ± 0.003	0.384 ± 0.038	0.651 ± 0.013
Abalone	0.898 ± 0.017	0.775 ± 0.024	0.914 ± 0.007
Diabetes	0.966 ± 0.008	0.844 ± 0.027	0.928 ± 0.010
Boston	0.898 ± 0.026	0.693 ± 0.035	0.869 ± 0.009
Fish	0.902 ± 0.016	0.672 ± 0.043	0.826 ± 0.033
Wine	0.749 ± 0.027	0.575 ± 0.009	0.734 ± 0.008

GPXは予測モデルと説明モデルが
同じ振る舞いをしている

計算時間

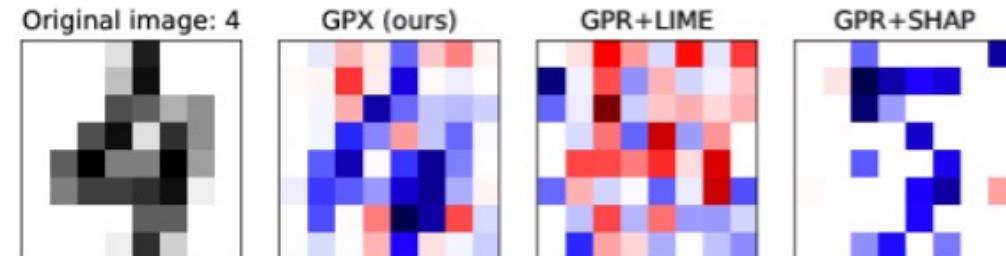
	GPX (ours)	GPR/SE+LIME	GPR/SE+SHAP
Training	5.74	5.53	5.76
Prediction	24.57	155.54	2643.06
Total	30.31	161.07	2648.82

GPXは予測時に比較的速い
通常のGP同様の高速化が適用可能

一貫性スコア

	GPX (ours)	GPR/SE+LIME	GPR/SE+SHAP
Digits	1.153 ± 0.015	2.410 ± 0.079	2.274 ± 0.110
Abalone	3.094 ± 0.249	11.809 ± 0.453	11.625 ± 0.362
Diabetes	1.164 ± 0.065	1.870 ± 0.119	1.400 ± 0.147
Boston	1.452 ± 0.081	4.180 ± 0.426	2.176 ± 0.304
Fish	1.497 ± 0.075	> 10 ⁵	1.634 ± 0.057
Wine	1.531 ± 0.113	> 10 ⁶	> 10 ⁵
Paper	0.004 ± 0.002	5.535 ± 0.161	4.548 ± 0.098
Drug	0.067 ± 0.014	16.976 ± 0.399	12.037 ± 0.513

GPXは類似事例で似た説明を出力可能

係数 w の生成結果

GPXのPythonライブラリ公開中

<https://github.com/yuyay/gpx>

```
from sklearn.metrics import mean_squared_error
from gpx import GPXRegressor

'''Training
X_tr: input data (numpy array), with shape of (n_samples, n_X_features)
y_tr: target variables (numpy array), with shape of (n_samples,)
Z_tr: simplified input data (numpy array), with shape of (n_samples, n_Z_features). The same as X_tr is
OK.
'''
model = GPXRegressor().fit(X_tr, y_tr, Z_tr)

'''Prediction
y_mean: the posterior mean of target variables
y_cov: the posterior variance of target variables
w_mean: the posterior mean of weights
w_cov: the posterior variance of weights
'''
y_mean, y_cov, w_mean, w_cov = model.predict(X_te, Z_te, return_weights=True)

'''Evaluation'''
mse = mean_squared_error(y_te, y_mean)
print("Test MSE = {}".format(mse))
```

解釈性の性能評価

- 正解データが存在しないケースがほとんどなので、**直接的な定量評価は困難**
- 代わりに、**間接的な定量評価**が用いられる
 - **Fidelity**
 - > 近傍事例で予測モデルと説明モデルの予測値が一致しているか？
 - **Stability**
 - > 近傍事例で説明が類似しているか？
 - **Faithfulness**
 - > 特徴量の貢献度とその特徴量を除いたときの予測値の変化の大きさは相関しているか？

LIMEによる説明を改善させる正則化 Explanation-based Optimization (ExpO)

忠実性

一貫性

[Plumb+ 2020]

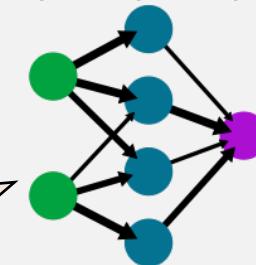
- LIMEによる説明のFidelityとStabilityを上げるために、予測モデルの学習時に正則化を入れる

$$\hat{f} := \arg \min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^N (\mathcal{L}(f, x_i, y_i) + \gamma \mathcal{R}(f, N_{x_i}^{\text{reg}}))$$

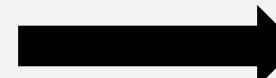
係数 w の Fidelity, Stability を上げる正則化

学習のイメージ

近傍で説明を類似させつつ、
予測誤差が下がるように
予測モデル f を更新

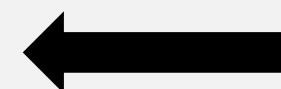


予測モデル f



各事例の係数 w

説明法
(LIME)



現在の予測モデル
でLIMEを学習

LIMEによる説明を改善させる正則化 Explanation-based Optimization (ExpO)

忠実性

一貫性

[Plumb+ 2020]

回帰データセットでの実験結果

Metric	Regularizer	autompgs	communities	day [†] (10 ⁻³)	housing	music	winequality.red	MSD
予測誤差	None	0.14 (0.03)	0.49 (0.05)	1.000 (0.300)	0.14 (0.05)	0.72 (0.09)	0.65 (0.06)	0.583 (0.018)
	FIDELITY	0.13 (0.02)	0.46 (0.03)	0.002 (0.002)	0.15 (0.05)	0.67 (0.09)	0.64 (0.06)	0.557 (0.0162)
	1D-FIDELITY	0.13 (0.02)	0.55 (0.04)	5.800 (8.800)	0.15 (0.07)	0.74 (0.07)	0.66 (0.06)	0.548 (0.0154)
Point Fidelity	None	0.040 (0.011)	0.100 (0.013)	1.200 (0.370)	0.14 (0.036)	0.110 (0.037)	0.0330 (0.0130)	0.116 (0.0181)
	FIDELITY	0.011 (0.003)	0.080 (0.007)	0.041 (0.007)	0.057 (0.017)	0.066 (0.011)	0.0025 (0.0006)	0.0293 (0.00709)
	1D-FIDELITY	0.029 (0.007)	0.079 (0.026)	0.980 (0.380)	0.064 (0.017)	0.080 (0.039)	0.0029 (0.0011)	0.057 (0.0079)
Neighb. Fidelity	None	0.041 (0.012)	0.110 (0.012)	1.20 (0.36)	0.140 (0.037)	0.112 (0.037)	0.0330 (0.0140)	0.117 (0.0178)
	FIDELITY	0.011 (0.003)	0.079 (0.007)	0.04 (0.07)	0.057 (0.018)	0.066 (0.011)	0.0025 (0.0006)	0.029 (0.007)
	1D-FIDELITY	0.029 (0.007)	0.080 (0.027)	1.00 (0.39)	0.064 (0.017)	0.080 (0.039)	0.0029 (0.0011)	0.0575 (0.0079)
Stability	None	0.0011 (0.0006)	0.022 (0.003)	0.150 (0.021)	0.0047 (0.0012)	0.0110 (0.0046)	0.00130 (0.00057)	0.0368 (0.00759)
	FIDELITY	0.0001 (0.0003)	0.005 (0.001)	0.004 (0.004)	0.0012 (0.0002)	0.0023 (0.0004)	0.00007 (0.00002)	0.00171 (0.00034)
	1D-FIDELITY	0.0008 (0.0003)	0.018 (0.008)	0.100 (0.047)	0.0025 (0.0007)	0.0084 (0.0052)	0.00016 (0.00005)	0.0125 (0.00291)

Fidelity、Stabilityはほぼ確実に改善し、予測誤差も同等かそれ以下に小さくなる

LIMEによる説明を改善させる正則化 Explanation-based Optimization (ExpO)

忠実性

一貫性

[Plumb+ 2020]

User study

タスク設定

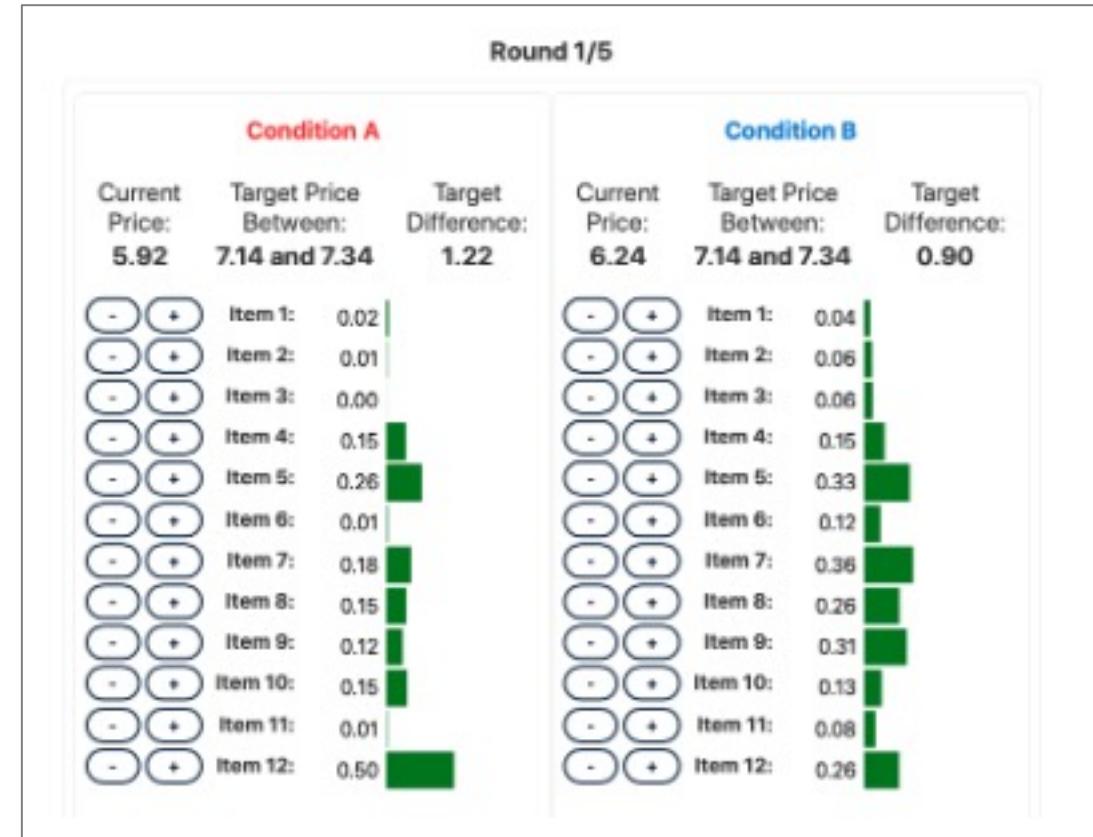
- 被験者に特徴量の値を色々変えてもらい
予測モデルの出力がある範囲にしてもらう
- 特徴量の値を変えるごとに貢献度を表示し
それを見て被験者は次変える特徴量を決める

実験結果

ExpOを使って学習したモデルは、説明を見れば
少ないステップで目的の範囲の値にできる



人間の感覚と合う説明と予測モデルの挙動になっている



Condition	Steps	Usefulness	Expectation
None	11.45	11	11
ExpO	8.00	28	26
No Preference	N.A.	15	17

より直接的な評価に向けて 特徴量を改ざんし、説明法がそれを正しく検出できるかを評価

[Zhou+ 2021]

- 特徴量の一部に「改ざん」を行い、予測モデルが改ざん部分だけを使えば予測可能な状況を作り出す
- その上で、説明法が改ざんした特徴量が重要と判定できるかを評価

改ざんした画像



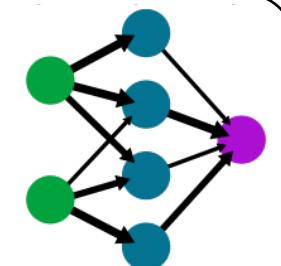
説明法

LIME
SHAP
Grad-CAM
⋮

特徴量の貢献度



改ざんしたデータで
学習した予測モデル



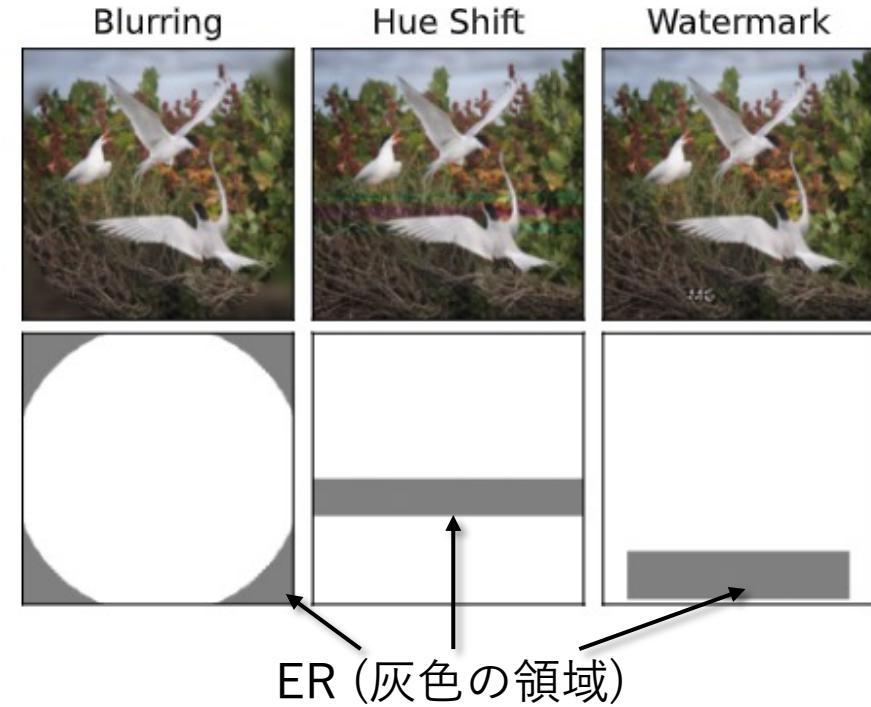
より直接的な評価に向けて 特徴量を改ざんし、説明法がそれを正しく検出できるかを評価

[Zhou+ 2021]

評価の手続き（画像の二値分類の例）

1. 正例だけ特徴量（ピクセル）を「改ざん」し、変化させた領域(Effective Region; ER)を記録
 - 改ざんした特徴量のみとラベルが相関するよう、ラベルは事前にランダムに変える
2. 改ざんしたデータセットで認識モデルを学習
 - 認識モデルは改ざんした特徴量のみで高精度の認識が可能
3. 認識モデルをLIME等で説明
 - どれだけ改ざんした領域の貢献度を高くできているかを Attr%(.) で評価

改ざんの例



$$\text{Attr}^{\%}(F) = \frac{\text{改ざん領域 } F \text{ の貢献度の総和}}{\text{全特徴量の貢献度の総和}}$$

より直接的な評価に向けて 特徴量を改ざんし、説明法がそれを正しく検出できるかを評価

[Zhou+ 2021]

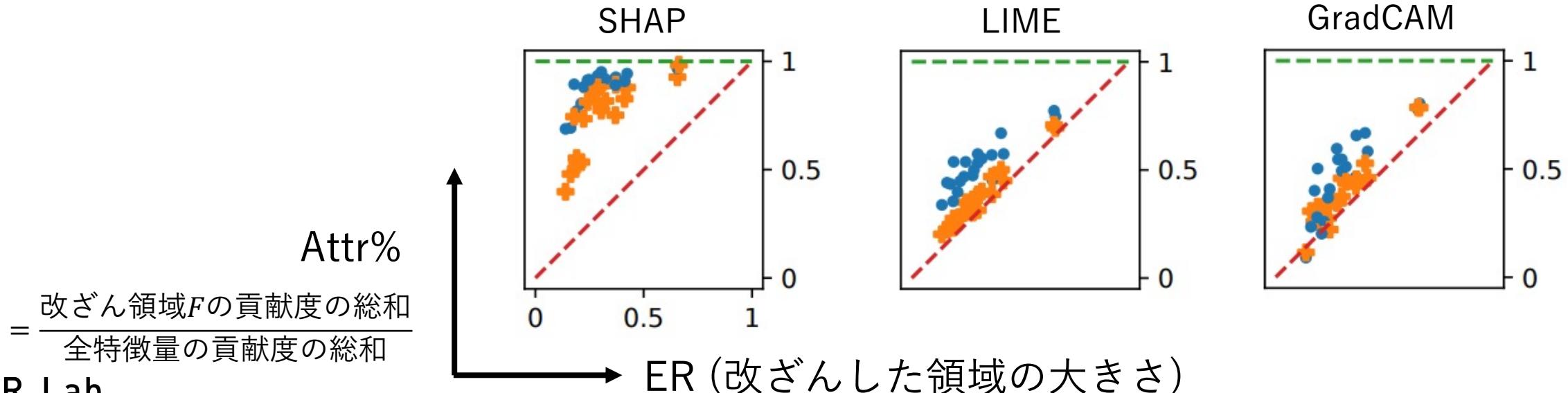
実験（どれだけ正確に改ざん部分を当てられているか）

期待

改ざんした部分のみの貢献度を高くする（すなわち $Attr\% \approx 1.0$ ）

結果

- 予測モデルの精度は高いにも関わらず、どんな手法も $Attr\% \approx 1.0$ にならない
- その中ではSHAPは良い方だった



まとめ

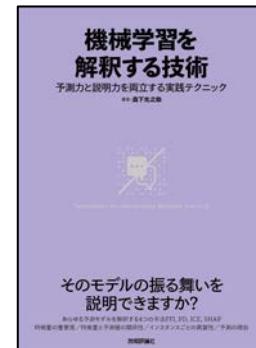
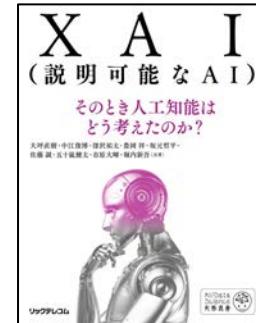
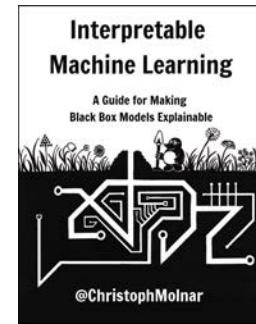
- Feature Attribution法の代表的手法について紹介
 - LIME：予測モデルの出力を局所線形モデルの係数で説明
 - SHAP：Shapley値に基づいて、予測モデルにおける特徴量の貢献度で説明
- PythonによるLIMEの実装例の紹介
- 関連研究について、速度・忠実性・一貫性の観点でまとめて紹介
 - BayLIME, MAPLE, GAM, GA²M, EBM, NAM, NGSLL, GPX, ExpO, 等

スライド、コードはこちらからダウンロードできます

https://github.com/yuyay/DEIM2022_XAI_tutorial

参考図書：日本語で読めるXAIの専門書

- **Interpretable Machine Learning** (Webで無料)
 - XAI手法について網羅的に書かれた良書
 - (株)HACARUSによる日本語訳が公開されている
- XAI(説明可能なAI) --そのとき人工知能はどう考えたのか?
 - LIME, SHAP等の実装例が豊富で実務向き
 - ビジネス寄りの視点での解説
- **機械学習を解釈する技術**
～予測力と説明力を両立する実践テクニック
 - 数式を用いた解説が豊富で、しっかり勉強したい人向け



参考文献 1/3

- [恵木 2020] 恵木正史. “XAI(eXplainable AI)技術の研究動向.” 日本セキュリティ・マネジメント学会誌, vol. 34, no. 1, 2020, https://www.jstage.jst.go.jp/article/jssmjournal/34/1/34_20/_pdf/-char/ja.
- [Ribeiro+ 2016] Ribeiro, Marco Tulio, et al. “‘Why Should I Trust You?’: Explaining the Predictions of Any Classifier.” *arXiv:1602.04938 [cs, Stat]*, Feb. 2016. *arXiv.org*, <http://arxiv.org/abs/1602.04938>.
- [Lundberg+ 2017] Lundberg, Scott M., and Su-In Lee. “A Unified Approach to Interpreting Model Predictions.” *Advances in Neural Information Processing Systems 30*, edited by I. Guyon et al., Curran Associates, Inc., 2017, pp. 4765–74.
- [岡田 2011] 岡田章. “ゲーム理論 新版.” 有斐閣, 2011.
- [森下 2021] 森下光之助. “機械学習を解釈する技術～予測力と説明力を両立する実践テクニック.” 技術評論社, 2021.
- [Yoshikawa+ 2021] Yoshikawa, Yuya, and Tomoharu Iwata. “Gaussian Process Regression With Interpretable Sample-Wise Feature Weights.” *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, Dec. 2021, <https://doi.org/10.1109/TNNLS.2021.3131234>.
- [Yoshikawa+ 2022] Yoshikawa, Yuya, and Tomoharu Iwata. “Neural Generators of Sparse Local Linear Models for Achieving Both Accuracy and Interpretability.” *An International Journal on Information Fusion*, vol. 81, May 2022, pp. 116–28.

参考文献 2/3

- [Lou+ 2013] Lou, Yin, et al. “Accurate Intelligible Models with Pairwise Interactions.” *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, 2013, pp. 623–31.
- [Nori+ 2019] Nori, Harsha, et al. “InterpretML: A Unified Framework for Machine Learning Interpretability.” *arXiv [cs.LG]*, 19 Sept. 2019, <http://arxiv.org/abs/1909.09223>. arXiv.
- [Agarwal+ 2021] Agarwal, Rishabh, et al. “Neural Additive Models: Interpretable Machine Learning with Neural Nets.” *arXiv [cs.LG]*, 29 Apr. 2020, <http://arxiv.org/abs/2004.13912>. arXiv.
- [Hastie+ 1986] Hastie, Trevor, and Robert Tibshirani. “Generalized Additive Models.” *Schweizerische Monatsschrift Fur Zahnheilkunde = Revue Mensuelle Suisse D'odonto-Stomatologie / SSO*, vol. 1, no. 3, Aug. 1986, pp. 297–310.

参考文献 3/3

- [Plumb+ 2018] Plumb, Gregory, et al. “Model Agnostic Supervised Local Explanations.” *arXiv [cs.LG]*, 8 July 2018, <https://proceedings.neurips.cc/paper/2018/file/b495ce63ede0f4efc9eec62cb947c162-Paper.pdf>. arXiv.
- [Plumb+ 2020] Plumb, Gregory, et al. “Regularizing Black-Box Models for Improved Interpretability.” *arXiv [cs.LG]*, 18 Feb. 2019, <http://arxiv.org/abs/1902.06787>. arXiv.
- [Zhao+ 2021] Zhao, Xingyu, et al. “BayLIME: Bayesian Local Interpretable Model-Agnostic Explanations.” *arXiv [cs.AI]*, 5 Dec. 2020, <http://arxiv.org/abs/2012.03058>. arXiv.
- [Zhou+ 2022] Zhou, Yilun, et al. “Do Feature Attribution Methods Correctly Attribute Features?” *arXiv [cs.LG]*, 27 Apr. 2021, <http://arxiv.org/abs/2104.14403>. arXiv.

