

現場で使える機械学習・データ分析
基礎講座 (DAY3)
SkillUP AI

本講座の構成

DAY1	DAY2
<ul style="list-style-type: none">機械学習概論<ul style="list-style-type: none">人工知能とは機械学習とは機械学習アルゴリズムの実装とワークフロー機械学習アルゴリズム概観教師あり学習の基礎<ul style="list-style-type: none">線形回帰ロジスティック回帰多変量モデルへの拡張モデルの評価指標<ul style="list-style-type: none">回帰問題 (MAE/MSE/RMSE)分類問題 (精度/適合率/再現率/F1-score)	<ul style="list-style-type: none">モデルの検証・正則化<ul style="list-style-type: none">訓練誤差と汎化誤差過学習正則化 (L2/L1)ホールドアウト法・交差検証法前処理<ul style="list-style-type: none">正規化 / 標準化無相関化 / 白色化教師あり学習の発展的トピック<ul style="list-style-type: none">サポートベクターマシン

本講座の構成

DAY3	DAY4
<ul style="list-style-type: none">• 前処理<ul style="list-style-type: none">• 特徴選択• 教師あり学習の発展的トピック<ul style="list-style-type: none">• 木モデル (決定木・ランダムフォレスト)• ニューラルネットワーク	<ul style="list-style-type: none">• 教師あり学習の発展的トピック<ul style="list-style-type: none">• 深層学習• k-最近傍法• 教師なし学習<ul style="list-style-type: none">• クラスタリング• 特徴抽出・次元削減• モデルの改善<ul style="list-style-type: none">• ハイパーパラメータ最適化

DAY3の目次

- グループワーク
 - 通し課題の進捗共有
- 特徴選択
 - 特徴選択の意義・次元の呪い
 - 特徴選択のアプローチ
 - 説明変数間の相関係数に基づく選択
 - ステップワイズ法
 - LASSO（特徴選択としてのL1正則化）
- 木モデル
 - 決定木
 - アンサンブル学習
 - ランダムフォレスト
 - アダブースト
- ニューラルネットワーク
 - ニューラルネットワークとパーセプトロン
 - 最急降下法
 - 誤差逆伝播法
 - 活性化関数
- グループワーク
- 質疑応答

特徴選択

1. 特徴選択の意義・次元の呪い
2. 特徴選択のアプローチ
3. 説明変数間の相関係数に基づく選択
4. ステップワイズ法
5. LASSO（特徴選択としてのL1正則化）

ビッグデータを扱う上での問題は何だろう？

- 昨今、画像やテキストなど様々なデータを多く収集できるようになり、機械学習の応用事例も様々なものが現れている
 - ビックデータ化に伴い、その傾向は年々加速
- 画像やテキストなどビッグデータを扱う上で問題になるポイントは？
- まずは画像やテキストのデータの形式を確認してみよう

テキストデータの例

- Twitterのデータを収集し、単語を拾い上げるような場面で出てくるデータの形式
- 文章ごとに単語の出現回数（頻度）を表現している

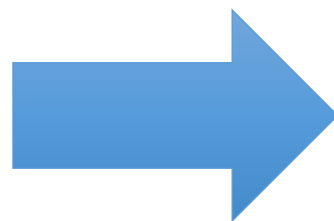
		単語							m:単語の個数		
		id	今日	は	電車	で	会社	に	行く	...	m
文章	1	1	3	1	2	2	1	1	...	0	
	2	0	2	1	0	3	0	2	...	4	
	3	1	2	0	5	0	0	1	...	0	
	4	0	0	3	0	1	2	1	...	3	
n:文章数	
	n	0	2	1	3	0	0	0	...	4	

画像データの例

- 画像認識系の問題を解くときによく出会うデータの形式
- 黒い部分を0、白い部分を1と表現している



0と1に変換



8pixel

10pixel

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0
0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

テキストや画像はデータの大きさが膨大になりやすい

- テキストデータであれば、単語の数が増えるほどデータの大きさが大きくなっていく
 - 日常には非常に様々な単語が飛び交っているため、膨大な大きさとなる
- 画像データであれば、画像が大きいほどデータの大きさが大きくなる
 - フルHD画質の画像であれば、1920pixel×1080pixel

次元の呪い

次元の呪いとは？

次元の呪い（じげんののろい、英: The curse of dimensionality）という言葉は、リチャード・ベルマンが使ったもので、（数学的）空間の次元が増えるのに対応して問題の算法が指数関数的に大きく（英語版）なることを表している。

「次元の呪い」『フリー百科事典 ウィキペディア日本語版』。
2013年11月21日 (木) 03:17 UTC、URL: <http://ja.wikipedia.org>

- 次元とは、**変数の数**のこと
- 次元(変数)が増えると、**計算時間が指数関数的に増える**
- 近年のビッグデータ化に伴い、次元(変数)はますます増える傾向

高次元データの難点

- 高次元データは、次元の呪いによって計算時間が増えるだけでなく人間にとっても解釈が難しいデータ
 - 一体どれが重要な変数なのか、特定するのが非常に大変
- 高次元データは、機械学習アルゴリズムにおいても学習が難しいデータ
 - 次元が高いほど、データ数をたくさん用意しないとオーバーフィッティングを起こしやすい

変数を削減しよう！

- 次元の呪いを回避し、解釈性を上げながらも、予測精度はできるだけ悪くならないように変数を減らしたい
- どのように減らす？
- 変数の不要 / 必要はどのように判断すればよい？

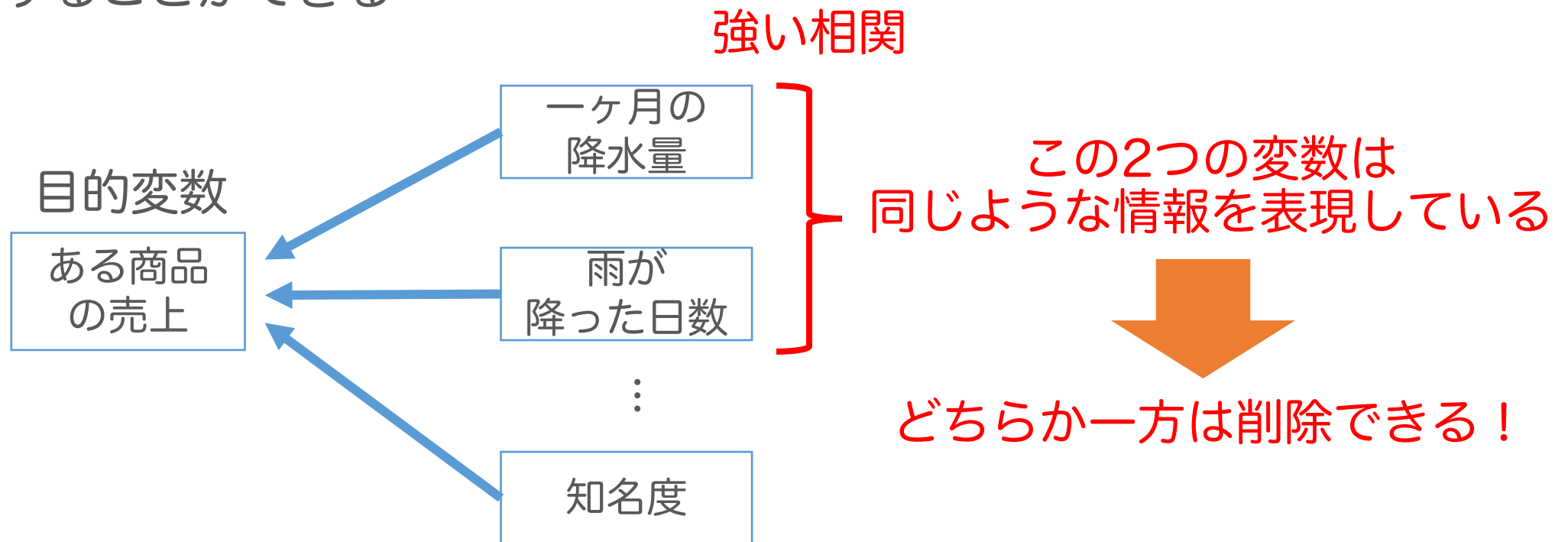
特徴選択

- 変数の良し悪しを決める基準とアルゴリズムによって、変数を削減することを**特徴選択**と呼ぶ
- 特徴選択は主に3つのアプローチに分けられる
 - 詳しくはこちらを参照：<http://www.jmlr.org/papers/v3/guyon03a.html>

	フィルタ法	ラッパー法	埋め込み法
計算時間	◎	X	○
予測精度	△	◎	○
	相関などの統計量を使って選択する方法。一般的に高速だが、精度の点で難がある。	モデルの学習と特徴選択を何度も繰り返すことでベストな組み合わせを見つける方法。時間はかかるが、予測精度は高い。	モデルの学習と同時に使用する変数を学習する方法。計算時間と精度のバランスがよい。

フィルタ法の例：説明変数間の相関係数に基づく選択

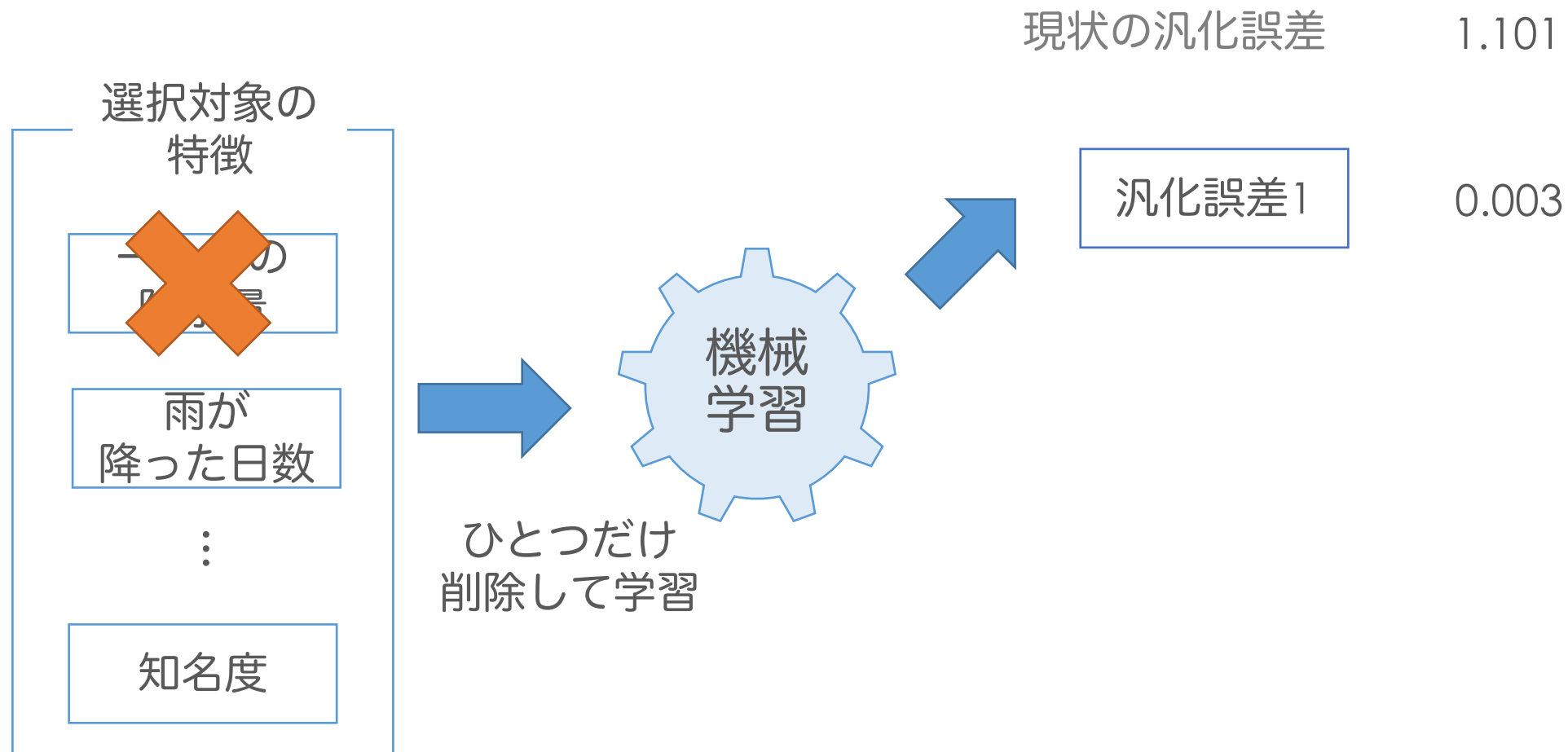
- 相関係数が高い説明変数の組を見つけ、片方を削除する方法はフィルタ法の1つである
- 相関係数が高い場合、一方の変数のみで事象を説明できることが多いため削除することができる



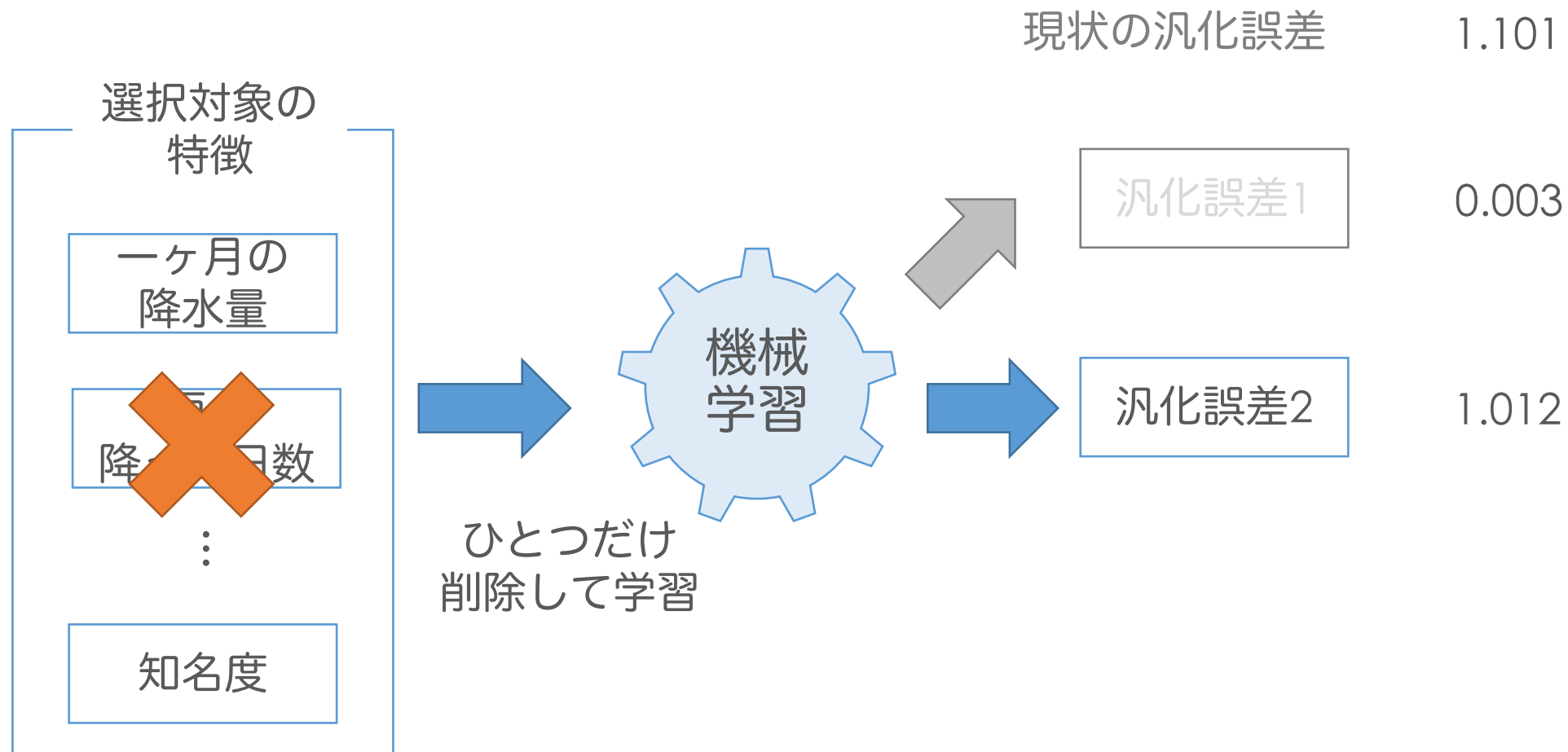
ラッパー法の例：ステップワイズ法

- ステップワイズ法とは、変数の削除（あるいは追加）→モデルの学習→汎化誤差の評価を繰り返すことで特徴選択を行う方法である
- 以下は減少法の説明である。ほかにも増減法など様々な種類がある
 1. 現状の汎化誤差を求める
 2. 変数をひとつ削除し、モデルを学習したあと汎化誤差を求める
 3. 変数をもとに戻したあと、新たに別の変数を削除し、モデルを学習する。そのあと汎化誤差を求める。
 4. 全ての変数について3を行ったあと、最も汎化誤差が減少した特徴を削除したままにして、2に戻る
 5. 汎化誤差が減少しなくなれば終了

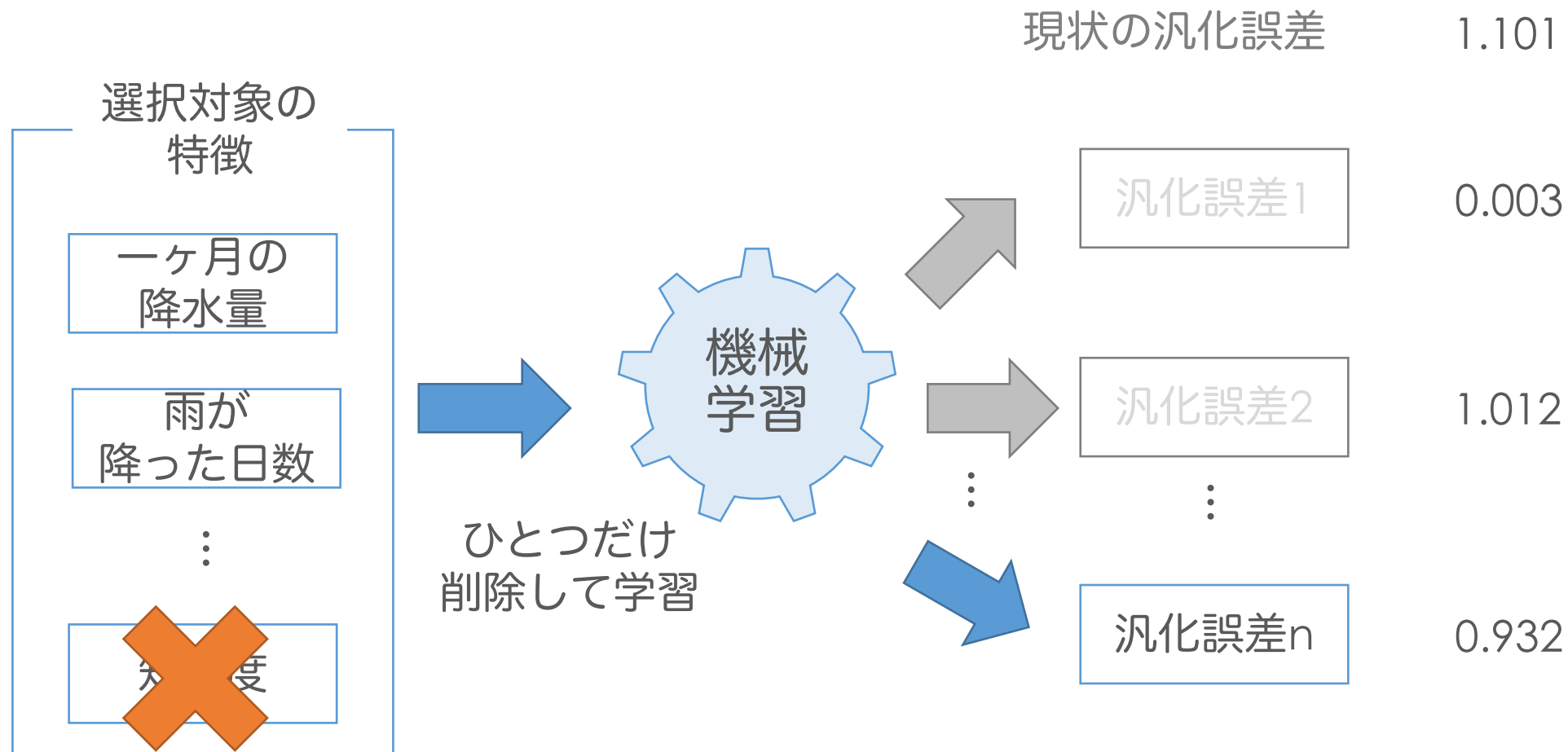
ラッパー法の例：ステップワイズ法



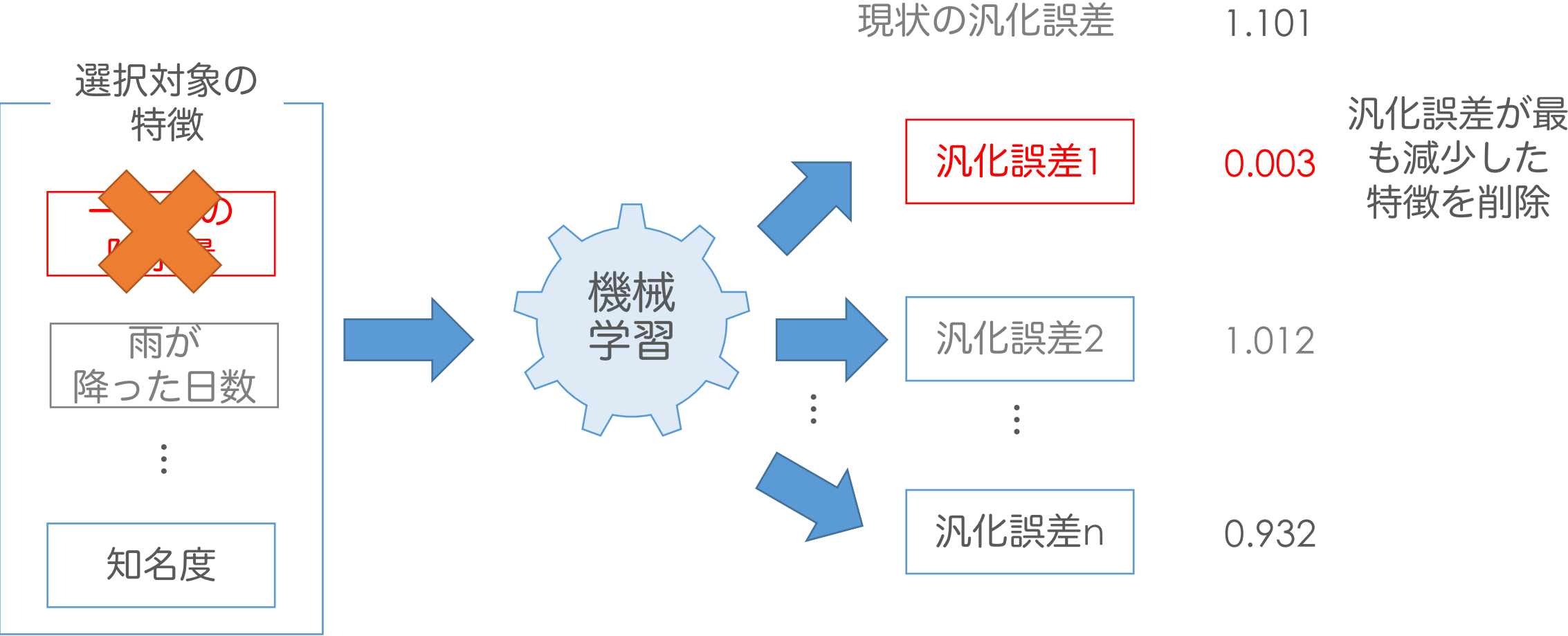
ラッパー法の例：ステップワイズ法



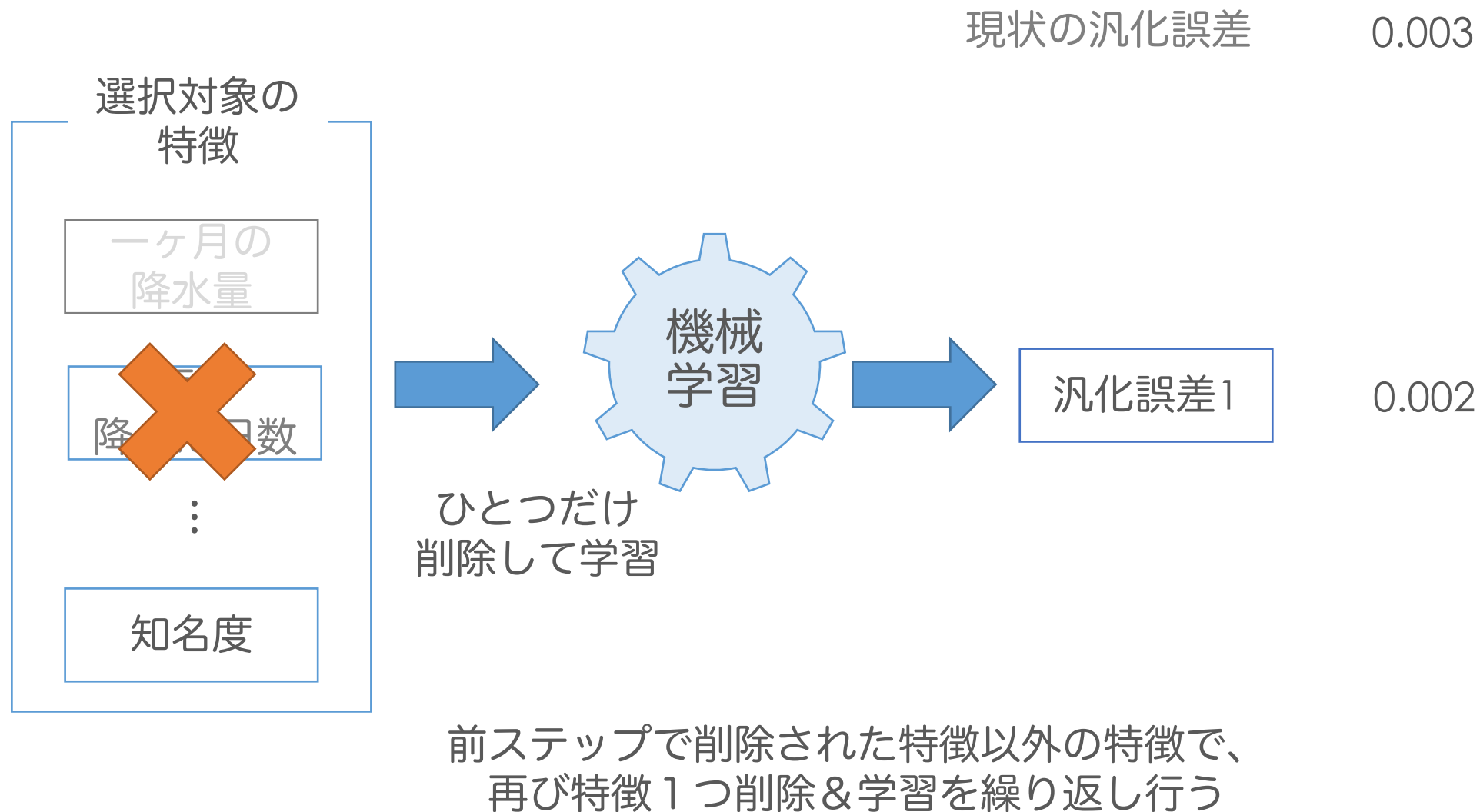
ラッパー法の例：ステップワイズ法



ラッパー法の例：ステップワイズ法

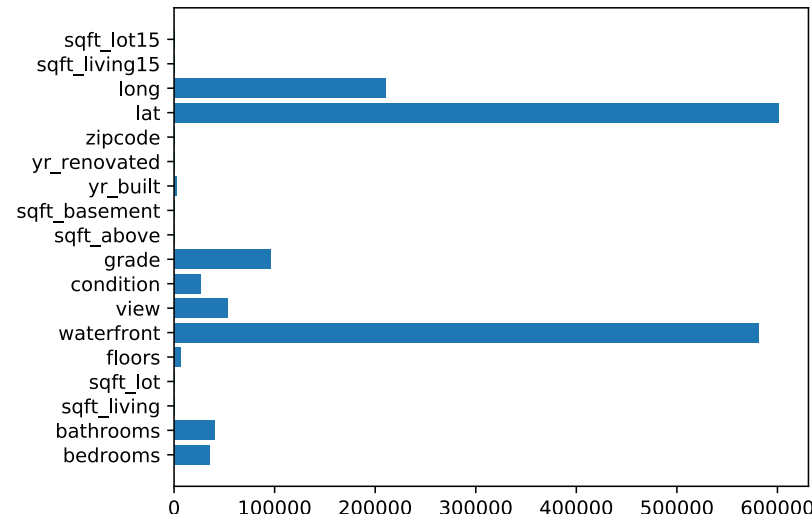


ラッパー法の例：ステップワイズ法



埋め込み法の例：LASSO（特徴選択としてのL1正則化）

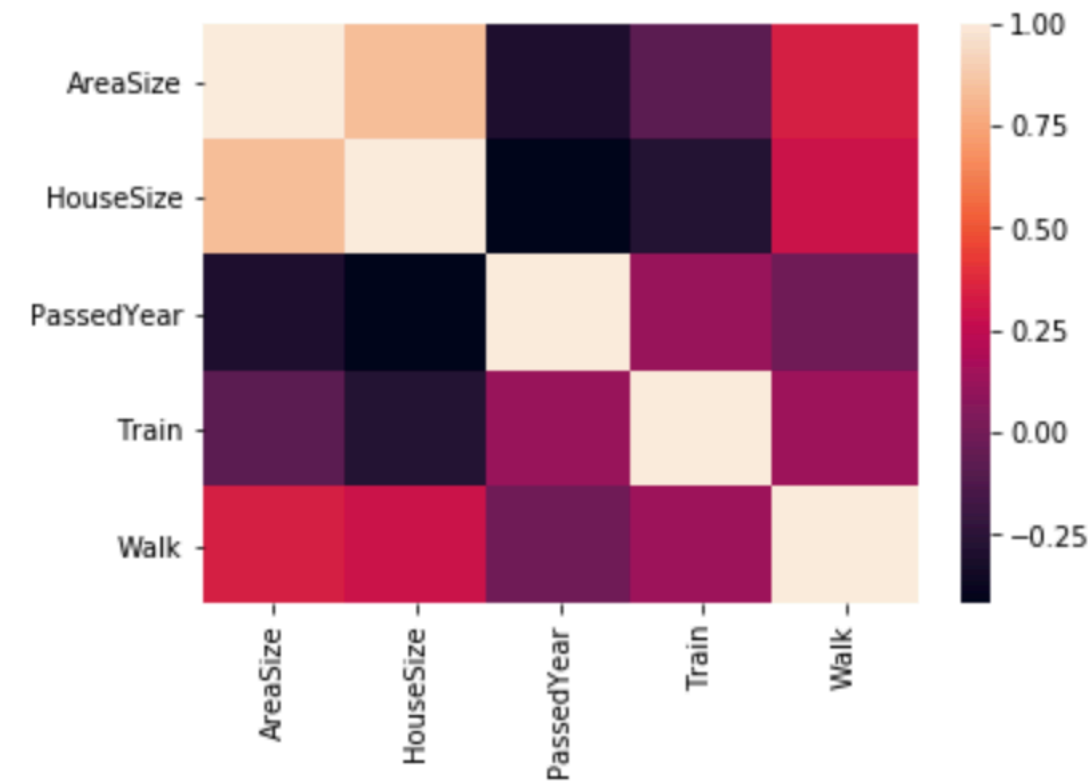
- L1正則化は、係数をゼロにする働きがある
- 学習後に係数が小さいものを削除することで特徴選択を行う
- 学習後に係数の大きさをグラフにすることで、どの変数が用いられているか確認することができる（係数プロット）



[演習] 1_feature_selection_filter.ipynb

- 説明変数間の相関係数に基づいて説明変数を削減してみよう

	AreaSize	HouseSize	PassedYear	Train	Walk
AreaSize	1.000000	0.843471	-0.303278	-0.074319	0.336687
HouseSize	0.843471	1.000000	-0.420226	-0.276636	0.291113
PassedYear	-0.303278	-0.420226	1.000000	0.124133	-0.020027
Train	-0.074319	-0.276636	0.124133	1.000000	0.138155
Walk	0.336687	0.291113	-0.020027	0.138155	1.000000



[演習] 2_feature_selection_wrapper.ipynb

- ステップワイズ法による特徴選択を確認しよう

```
# estimatorにモデルをセット
# 今回は回帰問題であるためLinearRegressionを使用
estimator = LinearRegression(normalize=True)

# RFECVは交差検証によってステップワイズ法による特徴選択を行う
# cvにはFold (=グループ) の数, scoringには評価指標を指定する
# 今回は回帰なのでneg_mean_absolute_errorを評価指標に指定 (分類ならaccuracy)
rfecv = RFECV(estimator, cv=10, scoring='neg_mean_absolute_error')
```

```
train_label = df_house["price"]
train_data = df_house.drop("price", axis=1)

y = train_label.values
X = train_data.values

# fitで特徴選択を実行
rfecv.fit(X, y)
```

```
# 特徴のランキングを表示 (1が最も重要な特徴)
print('Feature ranking: \n{}'.format(rfecv.ranking_))
```

Feature ranking:
[1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 2]

[演習] 3_feature_selection_embedded.ipynb

- LASSOによる特徴選択を確認しよう
- 係数をプロットし、予測値に寄与している変数を確認しよう

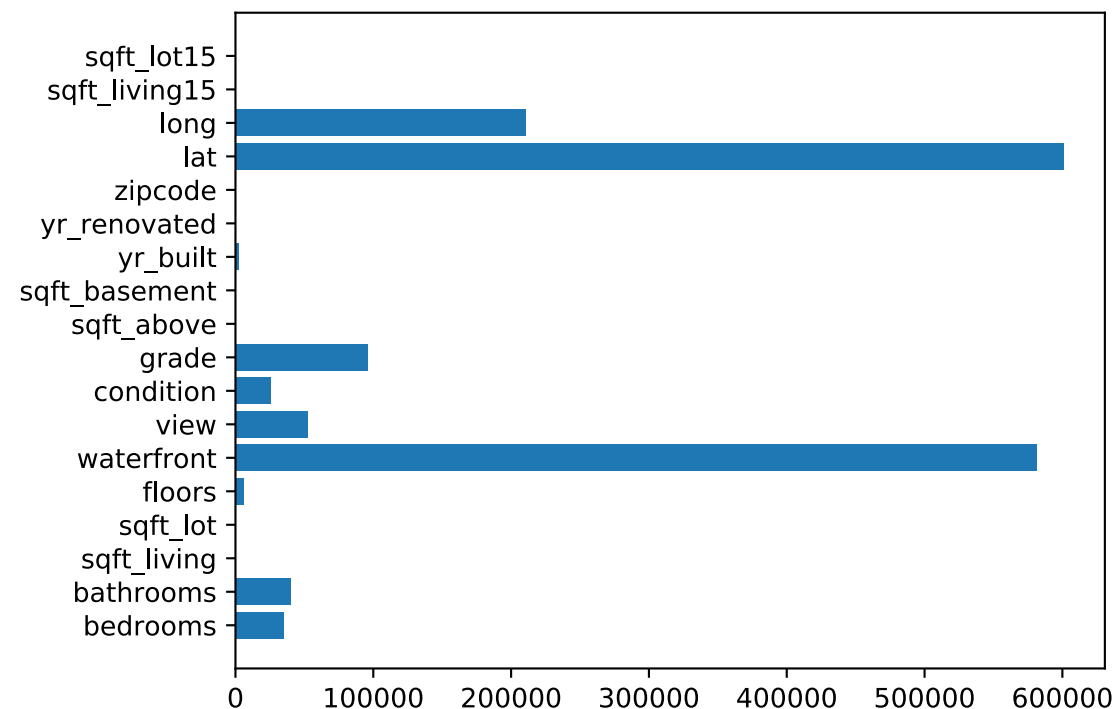
```
# estimatorにモデルをセット
# LassoCVを使って、正則化の強さは自動決定
estimator = LassoCV(normalize=True, cv=10)

# モデルの情報を使って特徴選択を行う場合は、SelectFromModelを使う
# 今回は係数が1e-5以下である特徴を削除する
# 係数のしきい値はthresholdで指定する
sfm = SelectFromModel(estimator, threshold=1e-5)
```

```
train_label = df_house["price"]
train_data = df_house.drop("price", axis=1)
```

```
y = train_label.values
X = train_data.values
```

```
# fitで特徴選択を実行
sfm.fit(X, y)
```



木モデル

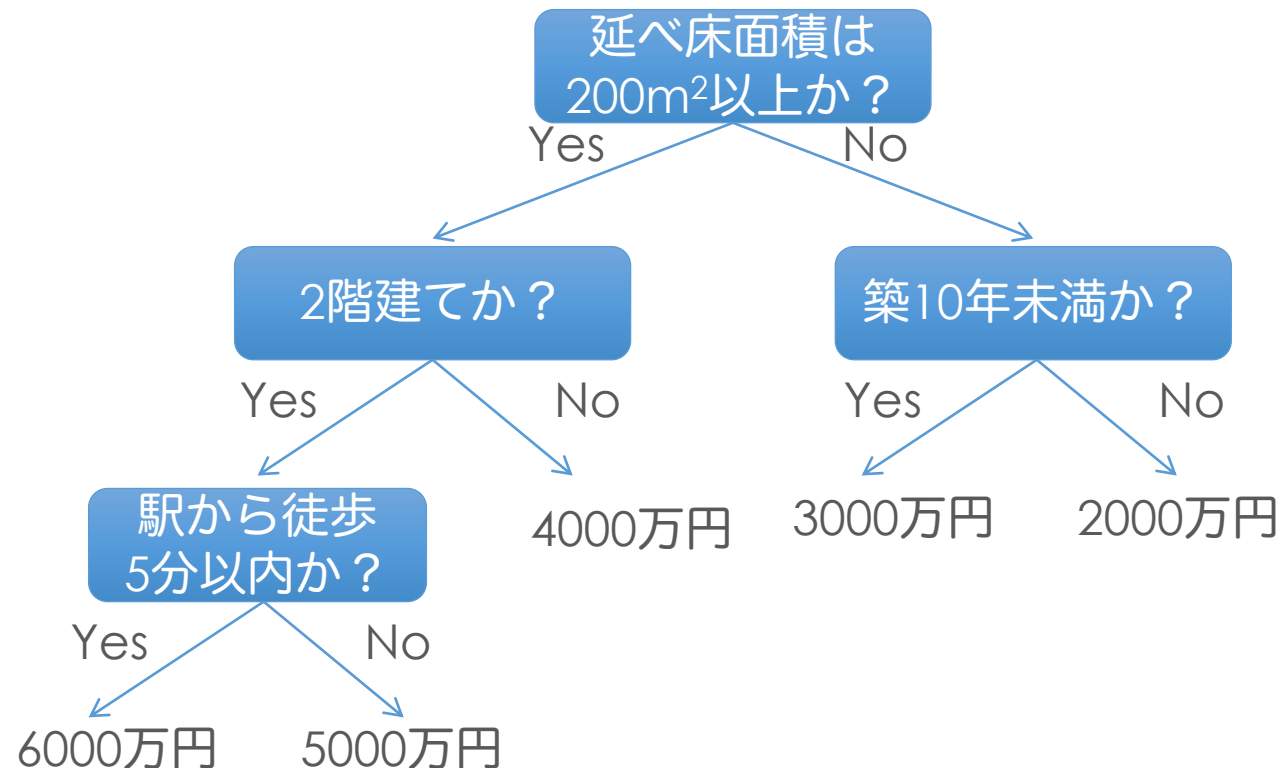
1. 決定木
2. アンサンブル学習
3. ランダムフォレスト
4. アダブースト

人間の専門家による判断プロセスを思い出してみよう

- 住宅価格の見積もりを例に考えてみよう
- 機械学習以前は、知識を持った専門家が経験と知識を組み合わせで見積もり価格を出していた
 - 今もこのように専門家による判断を用いる場面は多いだろう
- この専門家の判断プロセスはどのように構成されているだろうか？
- 自分で判断した場面を思い出してみよう。どのように判断しただろうか？

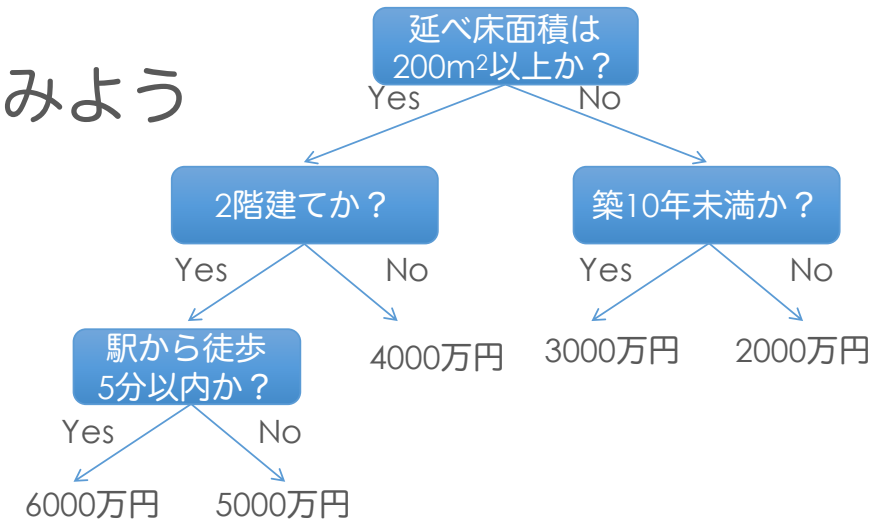
条件分岐による判断プロセス

- 住宅価格の見積もりであれば、以下のプロセスが考えられるだろう
- いくつかのルールを組み合わせ判断した場面は多いはず！



「木」の形をした判断プロセス

- 先程の「木」の形をしたトップダウン型の判断プロセスは、人間において解釈のしやすい形であった
- 「木」構造の判断プロセスを作ることができれば、予測過程の解釈も容易なモデルを作ることができそうだ
- まずは、この「木」を人の手で作ること考えてみよう
- 問題点はどこにあるだろうか？



条件分岐を人の手で作るのは難しい

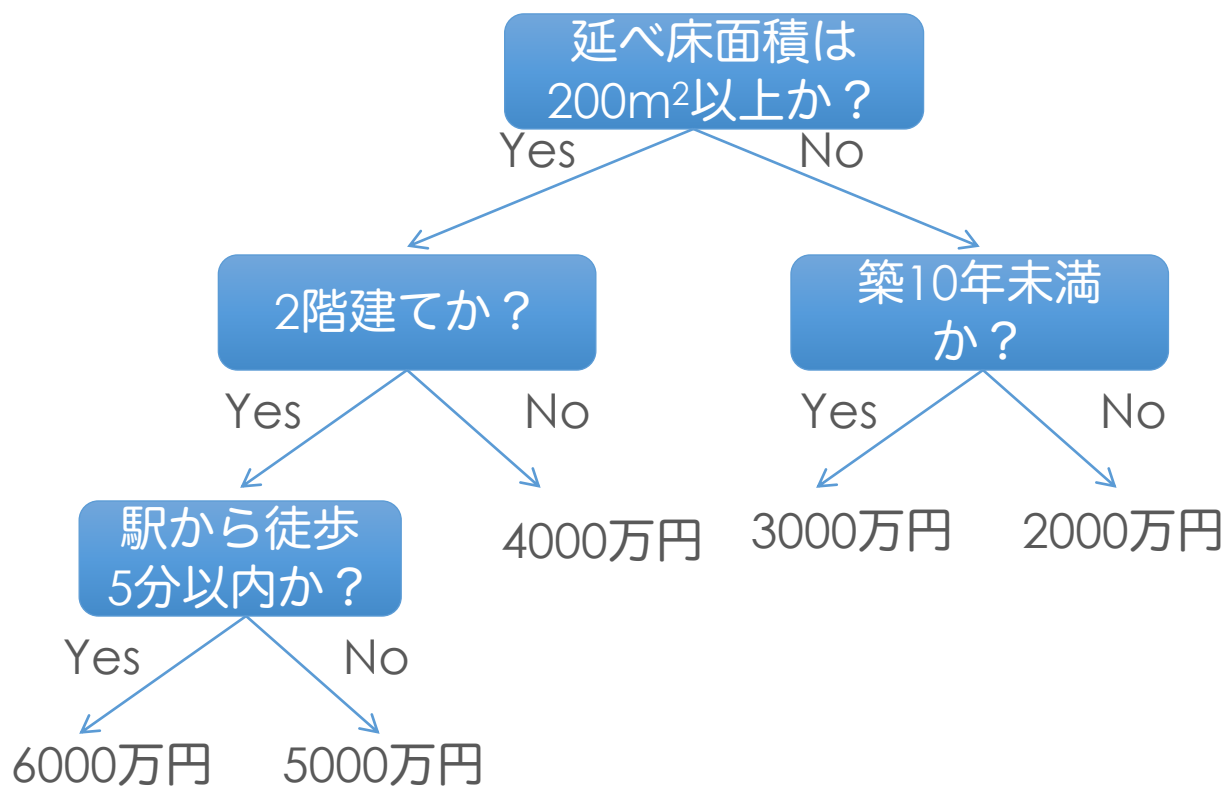
- 「木」に含まれる条件分岐をどのように作るかが悩ましいポイント
- プログラムに落とし込むには、知識を「明確」にしなければならない
 - 「もしAならばBである」という形で表現する必要がある
- 専門家でも全ての知識を明確に説明できるわけではない
 - 人間は判断プロセスの全てを言葉で説明できるだろうか？
 - 直感や勘といった言語化できない知識もあるはず
 - プログラムに矛盾は許されないが、人間はどうだろうか？

決定木とは

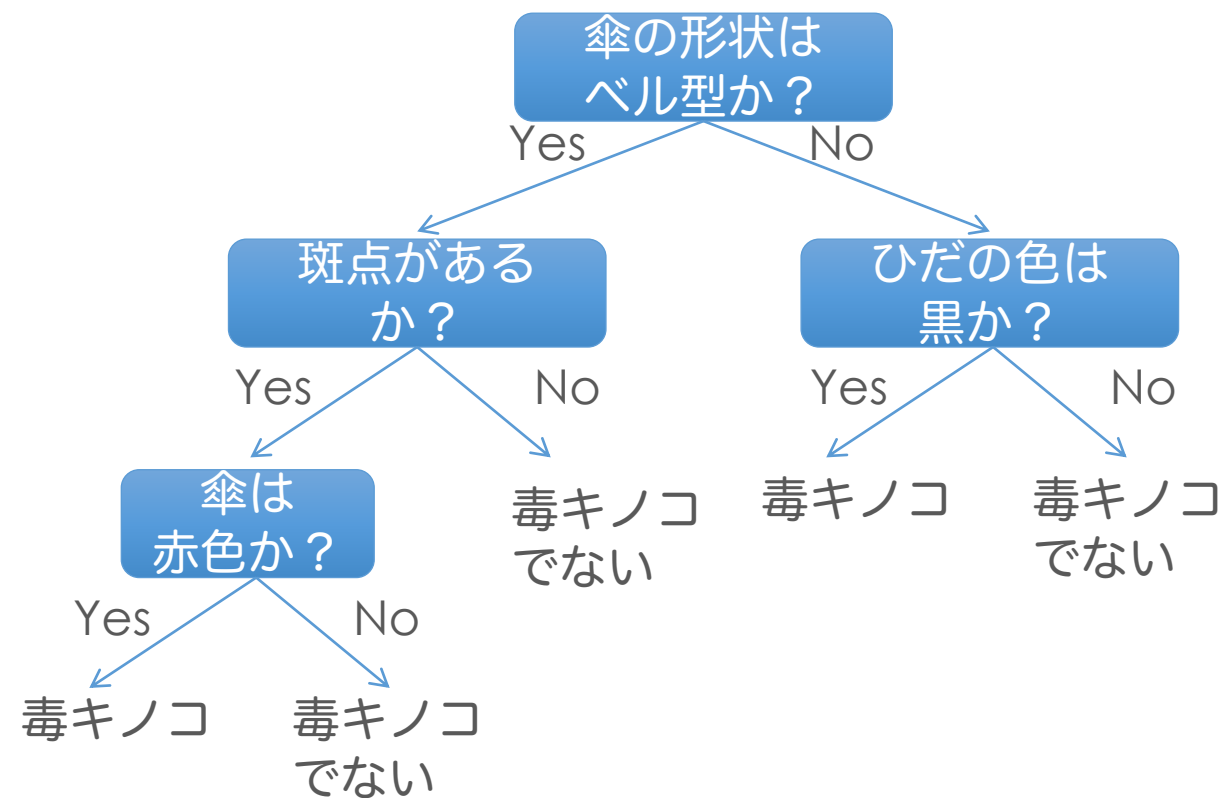
- そこで、**決定木**によって条件分岐を自動的に作成する
- 決定木とは、単純な条件分岐の組み合わせでモデルを構成する手法
- 決定木では、条件分岐を増やしていくことで、木を成長させていく
 - 基本的には、左と右に分かれる2分木を用いる
- どの条件分岐が良いかは、**不純度**によって判断する
 - 言い換えれば、不純度を損失関数としている
- 現在のノードの不純度と、分岐後の左右ノードの不純度合計を比較し、最も不純度が減る条件分岐を採用する

決定木の例

回帰の例



分類の例



不純度 (Impurity) とは

- データがどれだけばらついているかを表す指標
 - 分類では完全に分類できれば、不純度はゼロとなる
 - 回帰では1つのノードに対応するデータの分散がゼロであれば、不純度はゼロ
- 不純度を測る方法は色々だが、分類問題ではジニ係数やエントロピー、回帰問題では二乗誤差がよく用いられる

- 情報量とは、**事象の驚き度合い**を数値化した指標
- 到底起きそうもない事象が起きたことを知った→情報量が大きい
 - 例) 砂漠に雨が降った→到底起きそうにない出来事なので情報量 大
- いつでも起きそうな事象が起きたことを知った→情報量は小さい
 - 例) 東京に雨が降った→いつでも起きそうな出来事なので情報量 小

【例】

例えば、東京に雨が降ったという情報よりも、砂漠に雨が降った(確率が低い事象)という情報の方が驚くはずである。

ある日に、東京に雨が降る確率 $p(x)$ を0.3とすれば、その情報量は、

$$h(x) = -\log_2 0.3 = 1.737$$

となり、ある日に、砂漠に雨が降る確率 $p(x)$ を0.01とすれば、その情報量は、

$$h(x) = -\log_2 0.01 = 6.644$$

となる。

$$I(x) = -\log_2 p(x)$$

$I(x)$: ある事象 x の情報量 (単位はbits)

$p(x)$: ある事象 x が起きる確率

エントロピー（平均情報量）

- エントロピーとは、情報量を平均化したもので**事象のばらつき具合**を表す
- 大半は同じ事象しか起きない→エントロピーが小さい
 - 例) 砂漠の天気→ほとんどが晴れなのでエントロピー 小
- 観測するたびに事象がばらついている→エントロピーが大きい
 - 例) 東京の天気→晴れだったり雨だったりバラバラなのでエントロピー 大

$$H = \sum_i p(x_i) I(x_i) = - \sum_i p(x_i) \log_2 p(x_i)$$

H : エントロピー

$I(x_i)$: ある事象 x_i に対する情報量

【例】

例えば、ある日の東京の天気の確率が、 $p(\text{晴れ})=0.5$ 、 $p(\text{雨})=0.3$ 、 $p(\text{曇り})=0.2$ とすると、そのエントロピーは、

$$H = -\sum p(x) \log_2 p(x) = 1.485$$

となり、ある日の砂漠の天気の確率が、 $p(\text{晴れ})=0.9$ 、 $p(\text{雨})=0.01$ 、 $p(\text{曇り})=0.09$ とすると、そのエントロピーは、

$$H = -\sum p(x) \log_2 p(x) = 0.516$$

となる。

東京の天気は、確率変数 x がばらついており、砂漠の天気に比べ予測が難しいと言える。

ジニ係数 (Gini Index)

- 誤分類する確率を平均化した指標
- あるノード t において、クラス x_i が選ばれる確率を $p(x_i|t)$ としたとき、ジニ係数はあるノードにおいて誤分類する確率の期待値として定義される

$$G(t) = \sum_i^K p(x_i|t) \underbrace{(1 - p(x_i|t))}_{\text{誤分類する確率}} = 1 - \sum_i^K p(x_i|t)^2$$

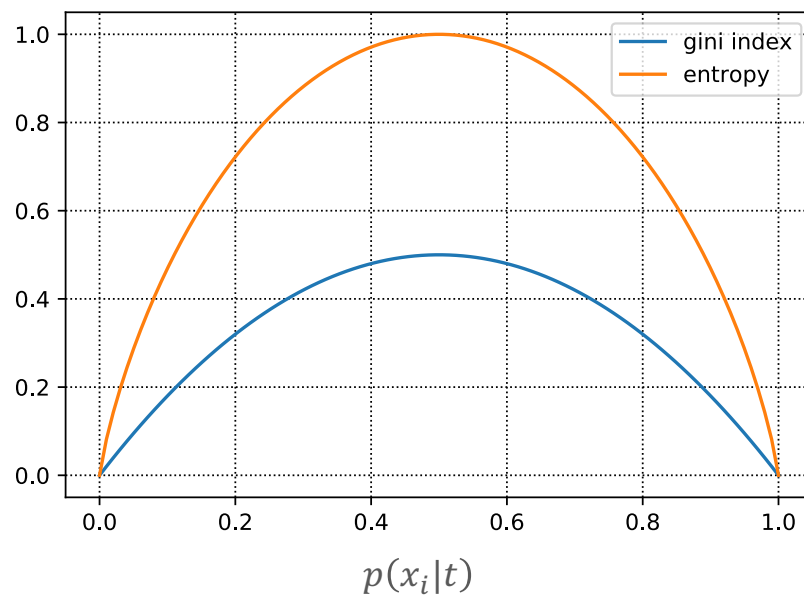
G : ジニ係数

K : クラス数

$p(x_i|t)$: あるノード t において、クラス x_i が選ばれる確率

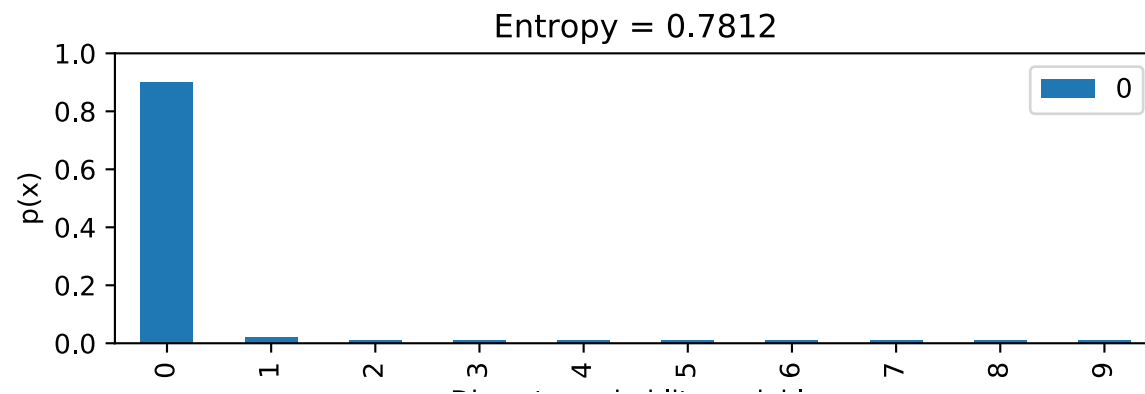
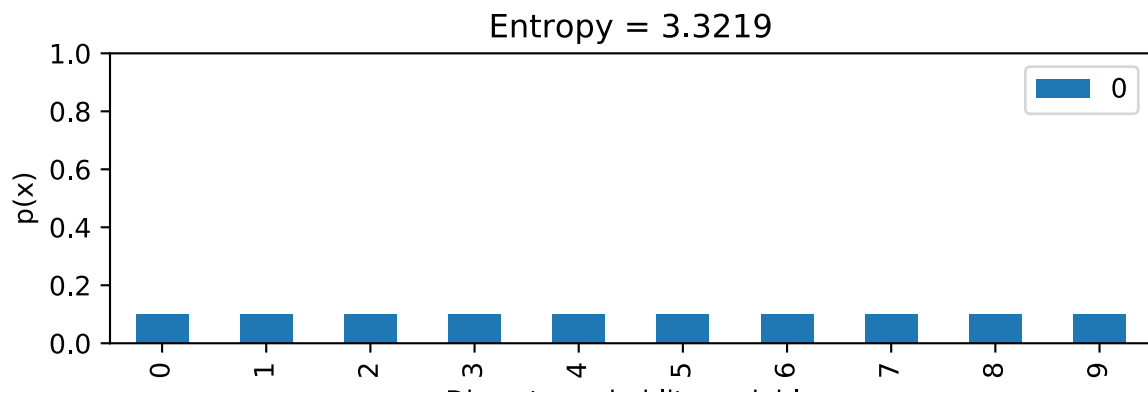
二値分類問題におけるエントロピーとジニ係数の変化

- 以下では、二値分類問題の場合を考える
- クラス x_i が選ばれる確率を $p(x_i|t)$ とした場合、 $p(x_0|t) = p(x_1|t) = 0.5$ の時に、エントロピーとジニ係数はともに最大となる
 - $p(x_0|t) = p(x_1|t) = 0.5$ は、どちらのクラスになるか全く予想がつかない状態と言える



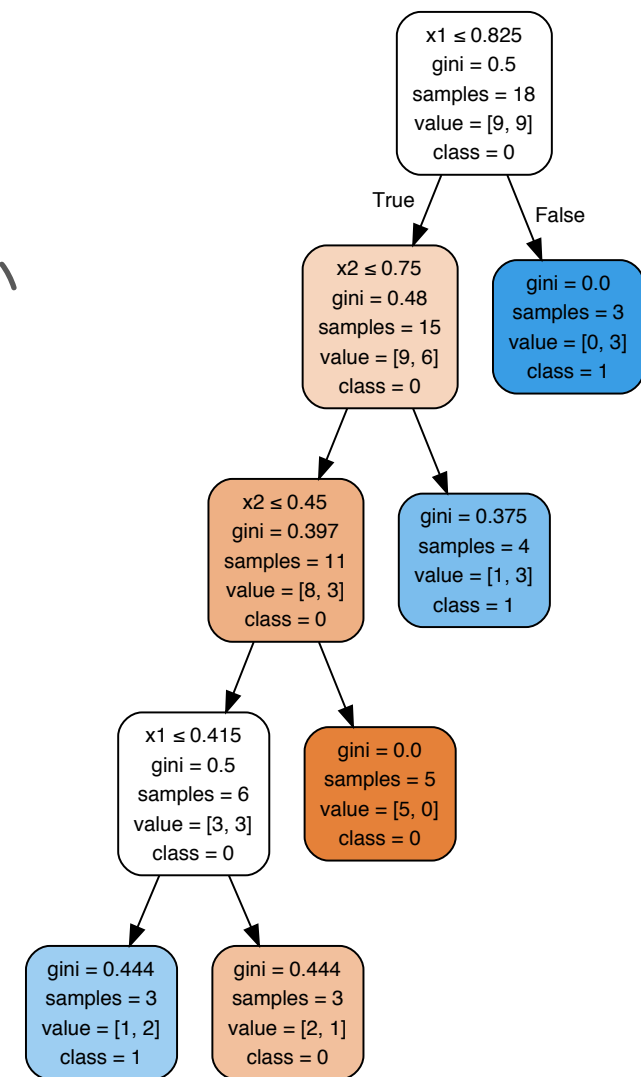
[演習] 4_entropy_and_gini.ipynb

- 情報量やエントロピーを実際の数値を用いて計算してみましょう
- 2クラス、多クラス問題におけるエントロピーやジニ係数の変化を確認しましょう



[演習] 5_descision_tree.ipynb

- scikit-learnによる決定木の実装を確認しよう
- 決定木を可視化し、どのような条件分岐が作られたか
を見てみよう



単一のモデルで大丈夫？

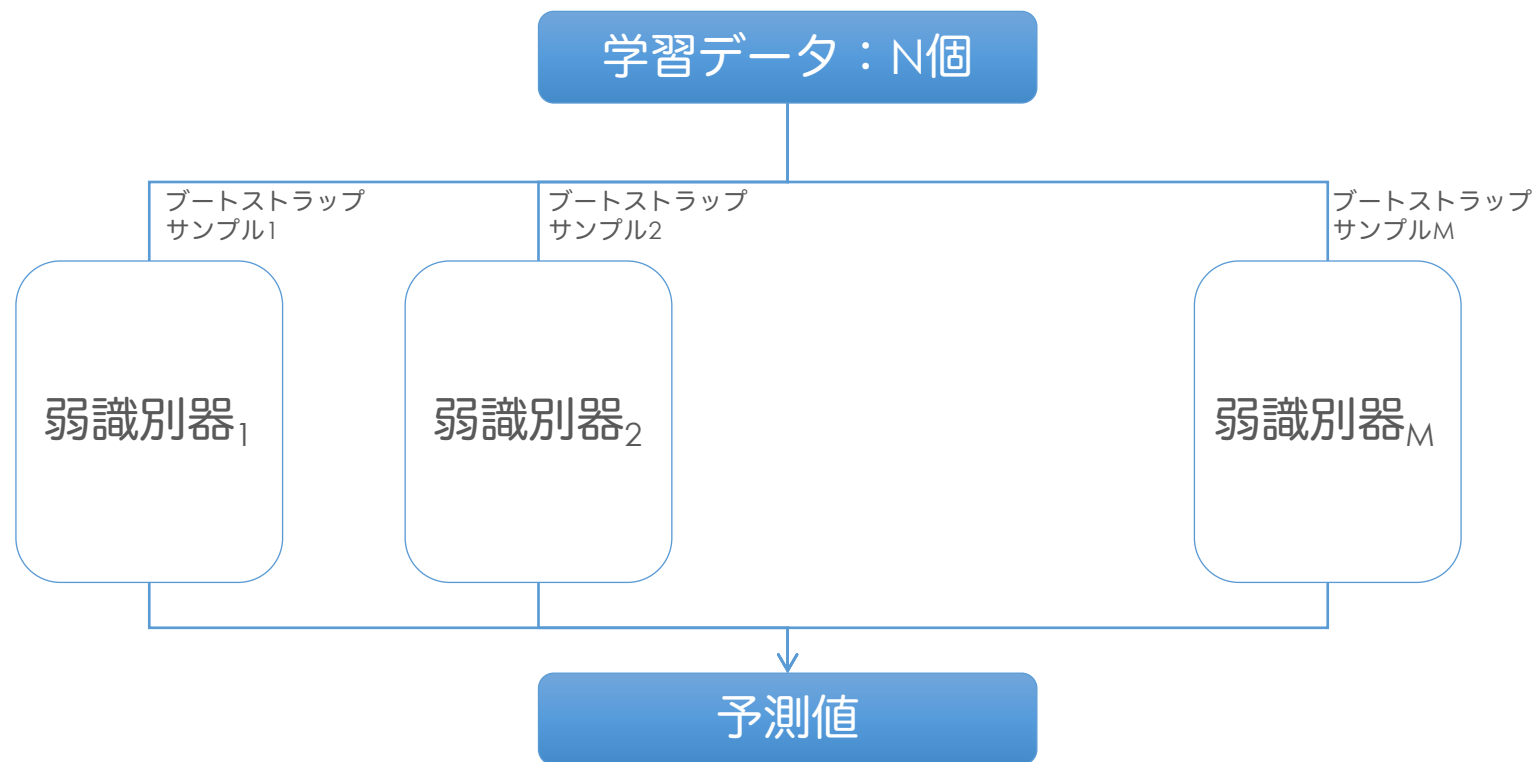
- 今まではひとつのモデルで予測していた
- たとえば、人間では複数人の意見を考慮することで、より未知の事象に対して頑健な意思決定を行ってきた
 - 多数決による意思決定が良い例
- 機械学習モデルにおいても、複数組み合わせることで、より頑健な予測結果を得ることはできないだろうか？

アンサンブル学習

- 複数の学習済みモデルの予測結果を組み合わせることで、汎化性能を改善する手法のこと
 - オーバーフィッティングしやすい決定木では特に有効なテクニック
- 組み合わせるモデルは異なっても構わないし、むしろ、多様であるほど頑健な結果が得られやすい
 - 決定木+ロジスティック回帰+サポートベクターマシンでもOK
- 今回は代表的な手法であるバギングとブースティングについて解説

バギング (Bagging) とは？

- バギング(**B**ootstrap **AGG**regat**ING**)とは、ブートストラップサンプルを用いて、複数の識別器を並列的に学習させていく方法
- 新しい入力に対する予測は、それら識別器の分類なら多数決、回帰なら平均で決定

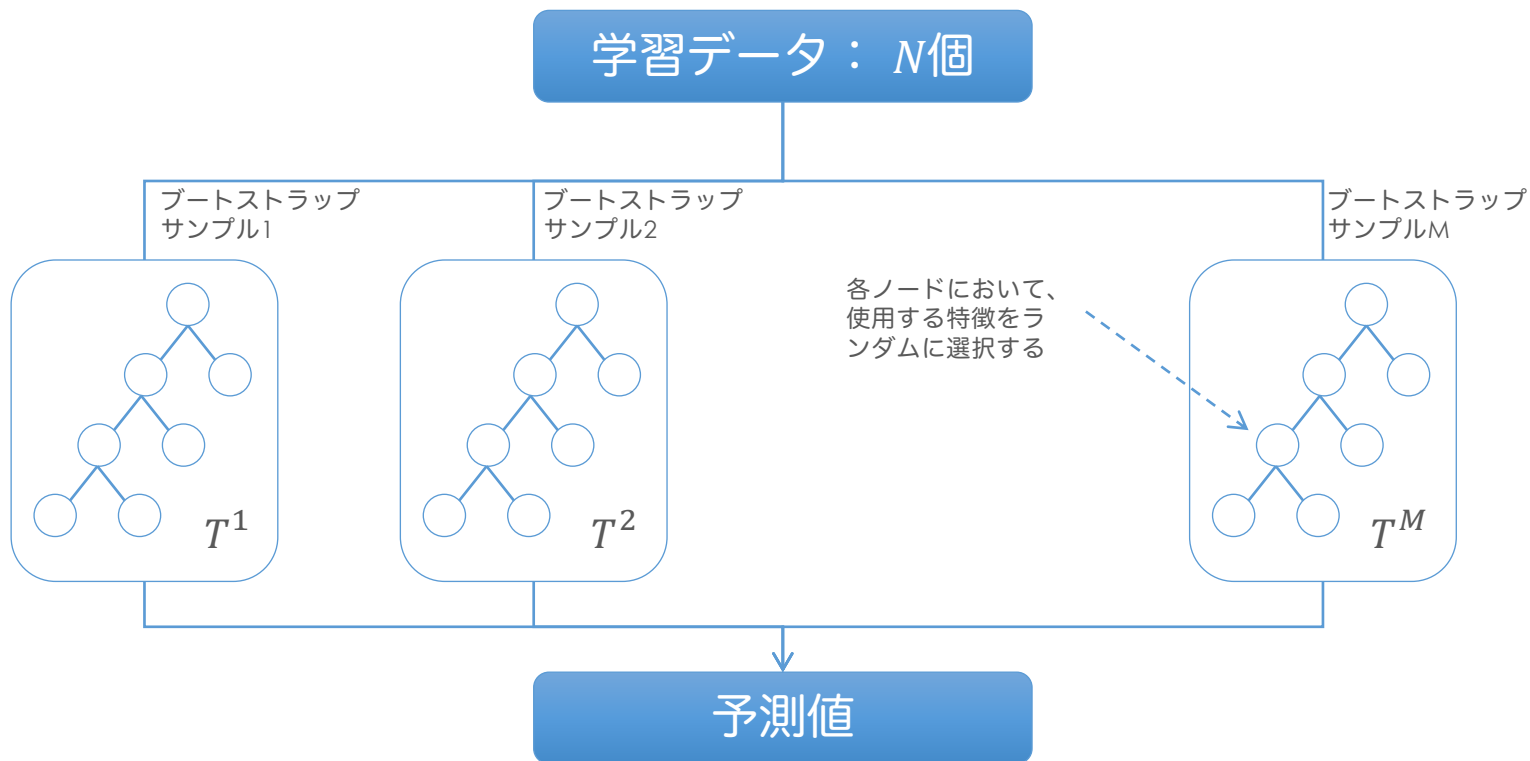


ブートストラップ (Bootstrap) とは？

- ブートストラップとは、ランダムにデータをサンプルする手法の1つ
- サンプルングの方法
 - 1) N 個のデータ点 $X = \{x_1, x_2, x_3, \dots, x_N\}$ があるとする。
 - 2) X からランダムに N 個のデータを復元抽出することによって、新たなデータ集合 X_b をつることができる
 - ✓ 復元抽出とは、同じデータを複数回選んでもいいという選び方のこと。これにより、 X のいくつかの点は、 X_b に複数回出現するが、一方、 X_b にはっていない点も存在することになる。
 - 3) この試行を L 回繰り返すことにより、サイズが N のデータ集合が L セット生成される

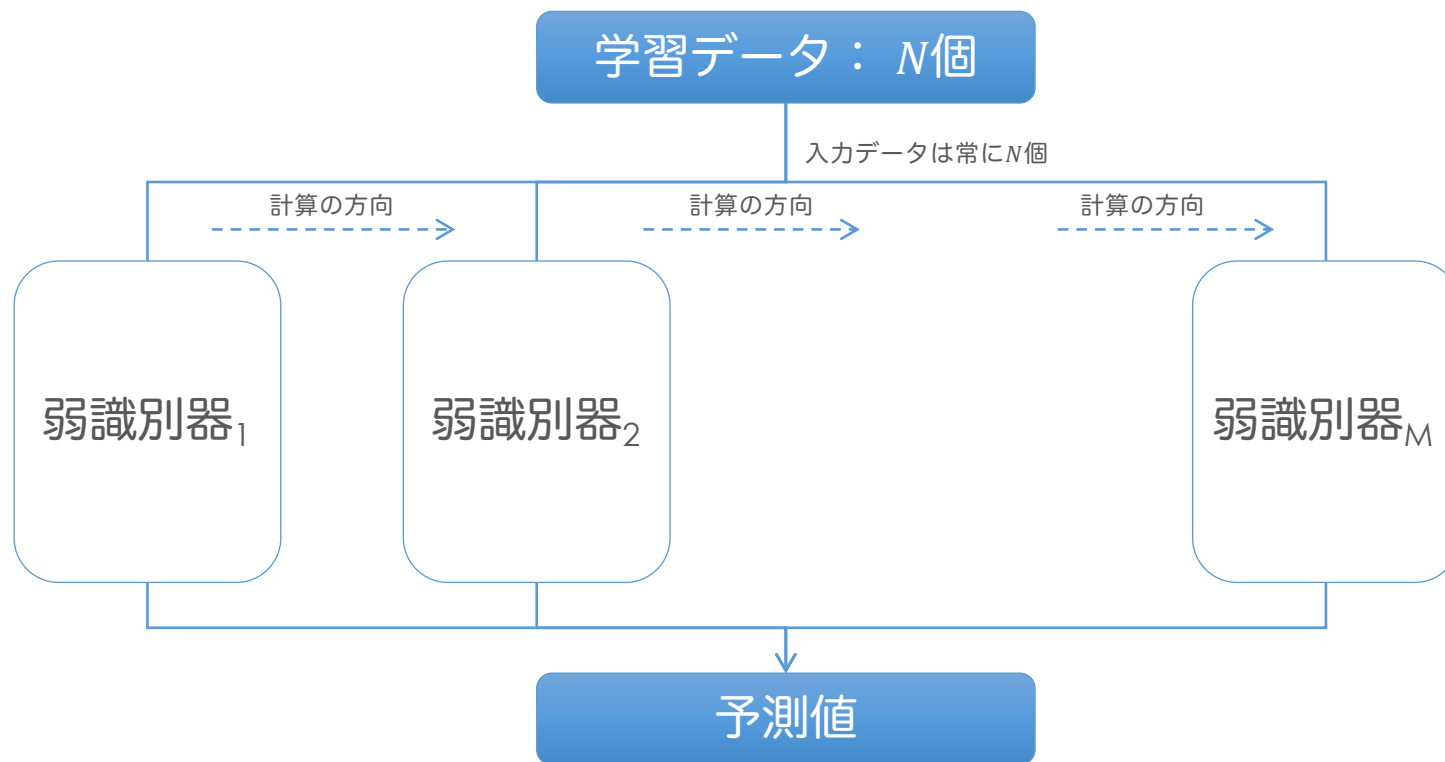
ランダムフォレスト (Random Forest) とは？

- バギングをベースに、弱識別器に決定木を用いた方法
- 決定木の各非終端ノードにおいて、識別に用いる特徴をあらかじめ決められた数だけランダムに選択することで、多様な決定木を生成できるようにした方法



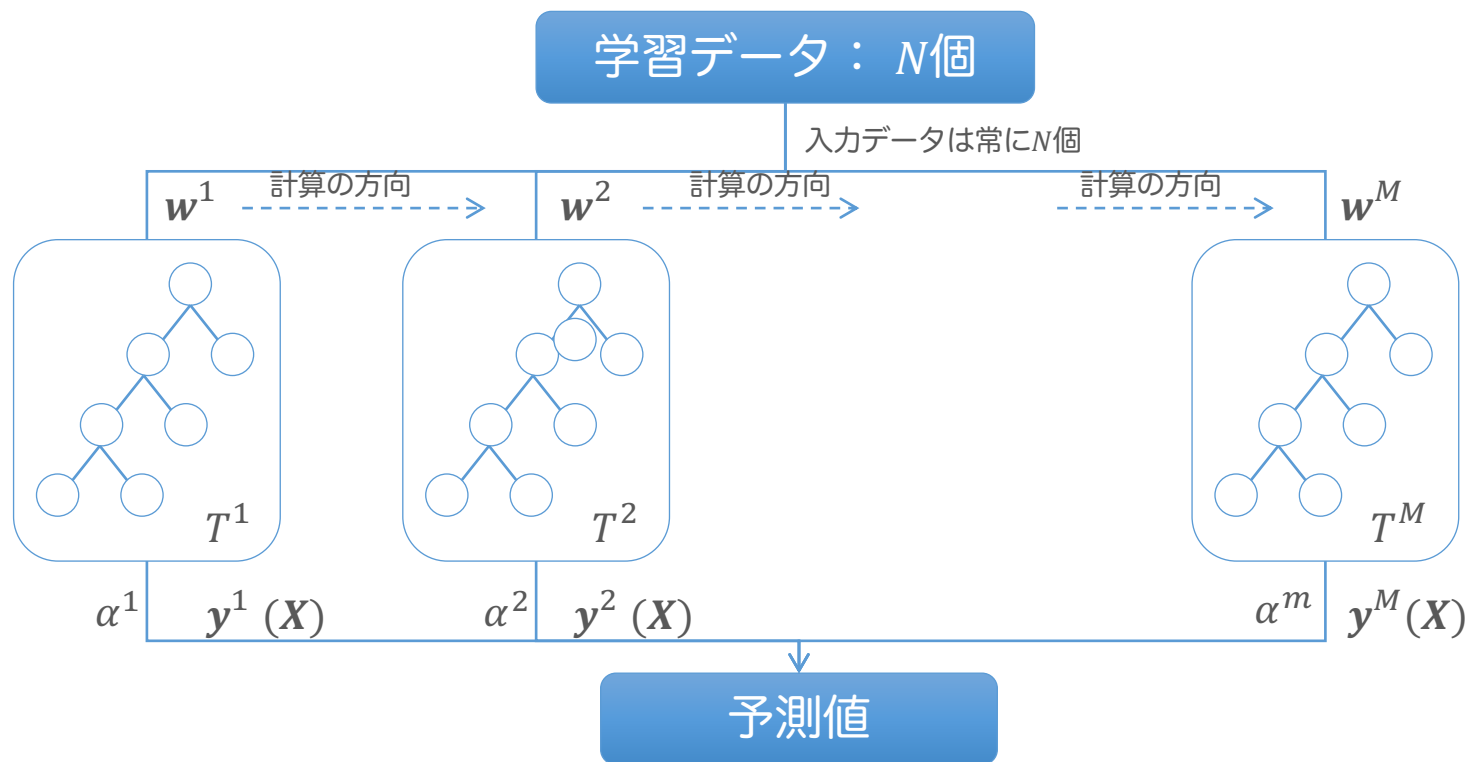
ブースティング (Boosting) とは？

- 複数の弱識別器を用意して、学習を直列的にすすめる方法
- 前の弱識別器の学習結果を参考にしながら、次の弱識別器を生成していく



アダブースト (Adaboost) とは？

- アダブースト(**Ad**aptive **boost**ing)とは、ブースティングを基にしたアンサンブル学習法
- 誤った学習データの重みを大きくし、正しく識別された学習データの重みを小さくすることで、後に学習する識別器ほど、誤りの多い学習データに集中して学習するようになる
- 新しい入力に対して予測する際は、分類ならば重み付きの多数決でクラスを決め、回帰ならM個の加重平均の値を予測値とする



参考：アダブーストの計算手順

・多クラス分類問題に対応したアダブーストの計算手順を以下に示す

1. 重みを $w_i^1 = \frac{1}{N}$, ($i = 1, 2, \dots, N$)に初期化する
2. $m = 1, \dots, M$ について以下を繰り返す
 - A) 重み w_i^m を用いて、識別器 T^m を学習
 - B) 誤り率を計算する

重み w を考慮した最適な決定木をつくる
(重み w は、 $p(x_i|t)$ を算出する際に使用される)

$$err^m = \frac{\sum_{i=1}^n w_i^m I(\mathbf{y}^m(\mathbf{X}_i) \neq t_i)}{\sum_{i=1}^n w_i^m}$$

ここで、 $I(\mathbf{y}^m(\mathbf{X}_i) \neq t_i)$ は、識別関数 T^m の出力 $\mathbf{y}^m(\mathbf{X}_i)$ が教師データ t_i と一致した時0、一致しなかった時1となる指示関数

- C) 重み α^m 計算する

$$\alpha^m = \ln \frac{1 - err^m}{err^m} + \ln(K - 1)$$

K :クラス数

- D) 重み w_i^m を更新する

$$A) \quad w_i^{m+1} = w_i^m \exp \left(\alpha^m I(\mathbf{y}^m(\mathbf{X}_i) \neq t_i) \right), i = 1, 2, \dots, N$$

- E) 重み w_i^{m+1} を正規化する(合計1になるようにする)

3. 予測結果としてのクラス $C(\mathbf{X}_i)$ を出力する

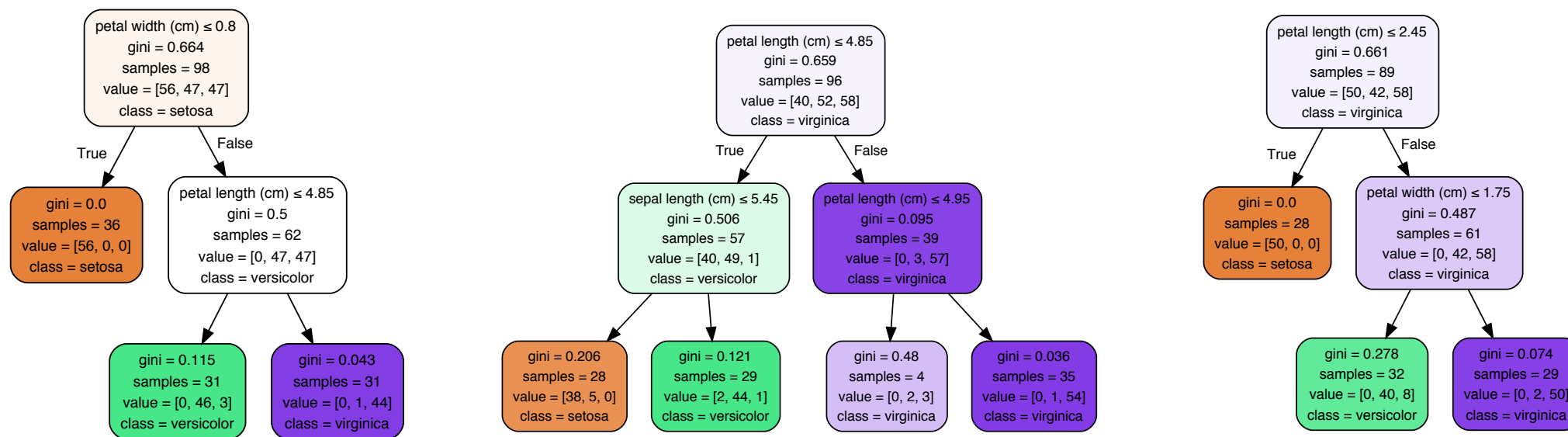
$$C(\mathbf{X}_i) = \arg \max_k \sum_{m=1}^M \alpha^m I(\mathbf{y}^m(\mathbf{X}_i) = k)$$

参照論文

Ji Zhu et al., Multi-class AdaBoost,
<https://web.stanford.edu/~hastie/Papers/samme.pdf>

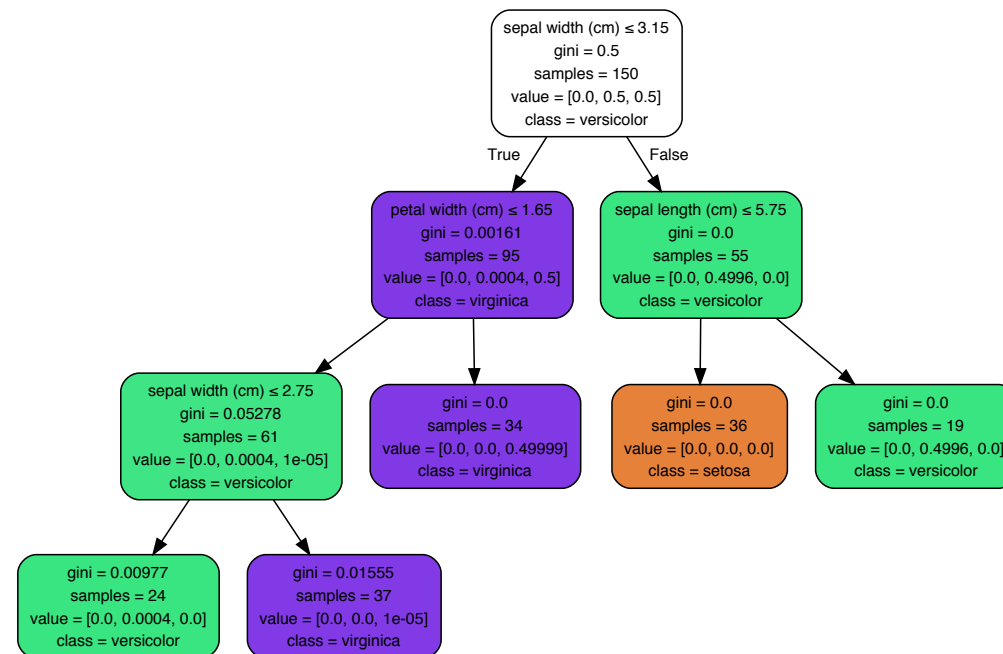
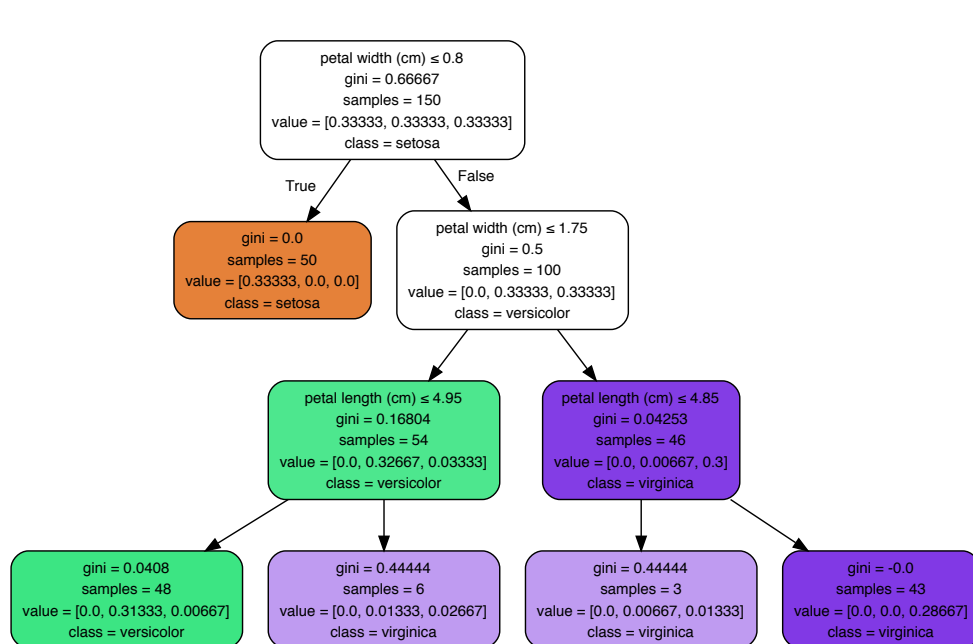
[演習] 6_random_forest.ipynb

- ランダムフォレストを用いて、アヤメの分類を行ってみましょう
- 不純度の評価方法や木の深さなどのハイパーパラメータを変更し、どのような変化があるか確認しましょう



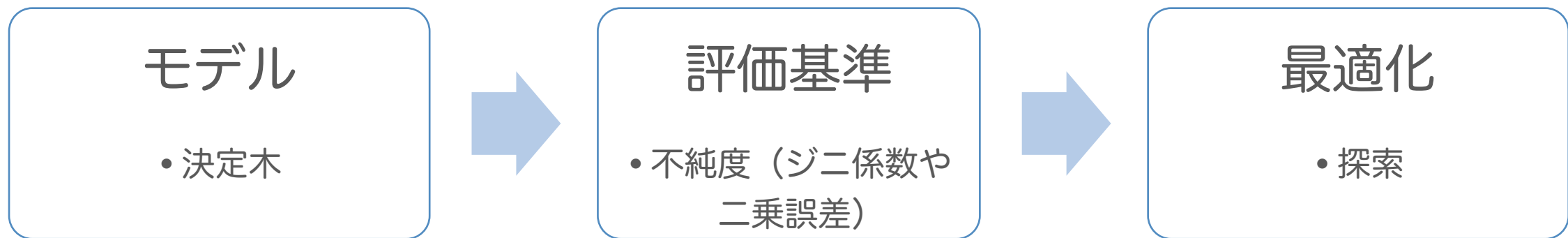
[演習] 7_adaboost.ipynb

- アダブーストと決定木を用いて、アヤメの分類を行ってみましょう
- 不純度の評価方法や木の深さなどのハイパーパラメータを変更し、どのような変化があるか確認しましょう



木モデルまとめ（モデル、評価基準、最適化の観点から）

- 決定木、ランダムフォレスト：条件分岐を増やしていくことで、木を成長させていくモデル
- 不純度をもとに分岐するための条件を最適化していく



ニューラルネットワーク

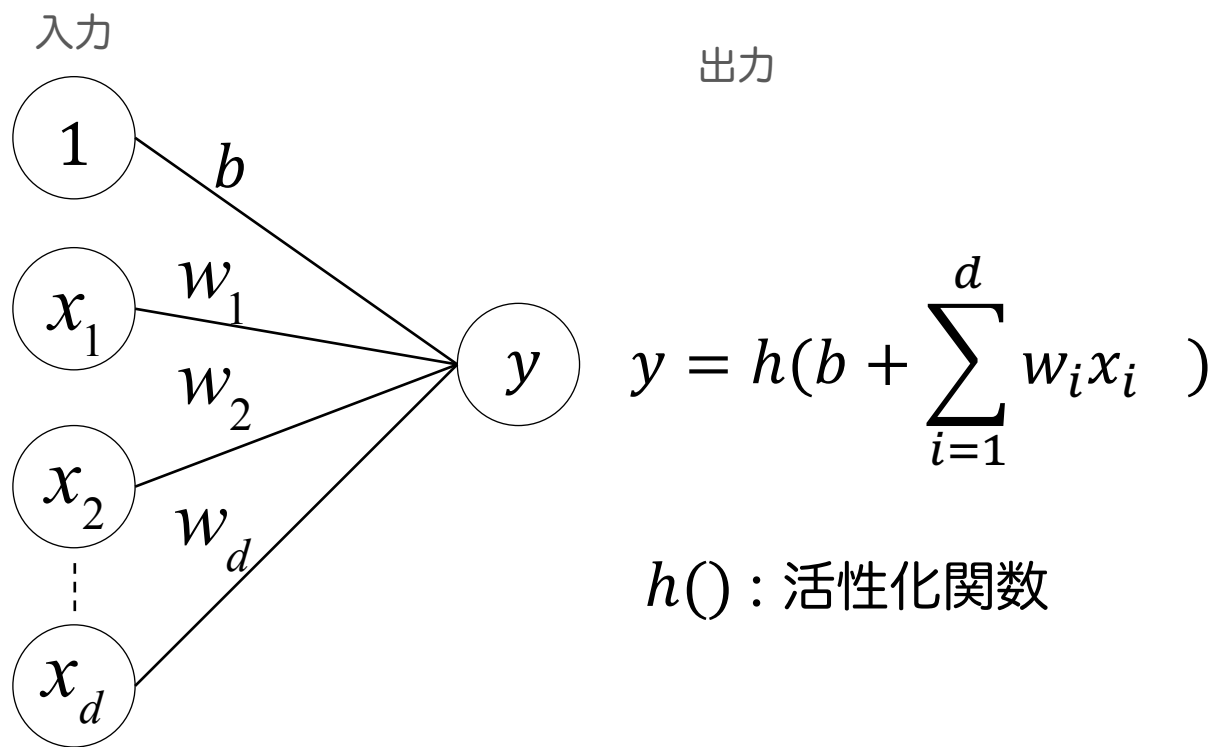
1. ニューラルネットワークとパーセプトロン
2. 最急降下法
3. 誤差逆伝播法
4. 活性化関数

ニューラルネットワークとは

- ニューラルネットワークとは、人間の脳内にある神経回路網を数式的なモデルで表現したアルゴリズム
- パーセプトロンは、もっとも単純なニューラルネットワークモデルであり、入力層と出力層で構成される
- パーセプトロンに中間層を加えたものが多層パーセプトロン

パーセプトロン

- パーセプトロンとは、入力層と出力層だけで構成されたネットワークモデルのこと
- 入力に重みをかけた総和を計算し、その結果を活性化関数に通したものが出力となる
- 活性化関数にはステップ関数などが用いられる



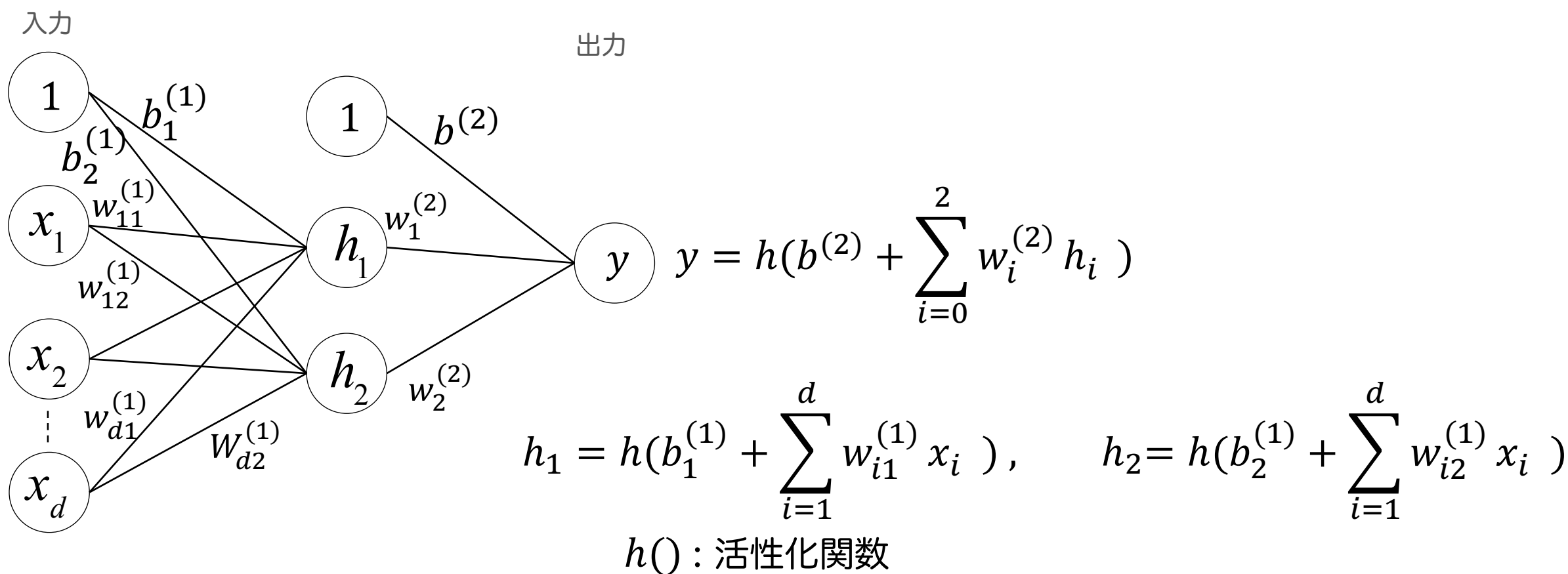
多層パーセプトロン

重みの記号

$w_{12}^{(1)}$ 第1層目の重み

前層の1つ目のノード 次層の2つ目のノード

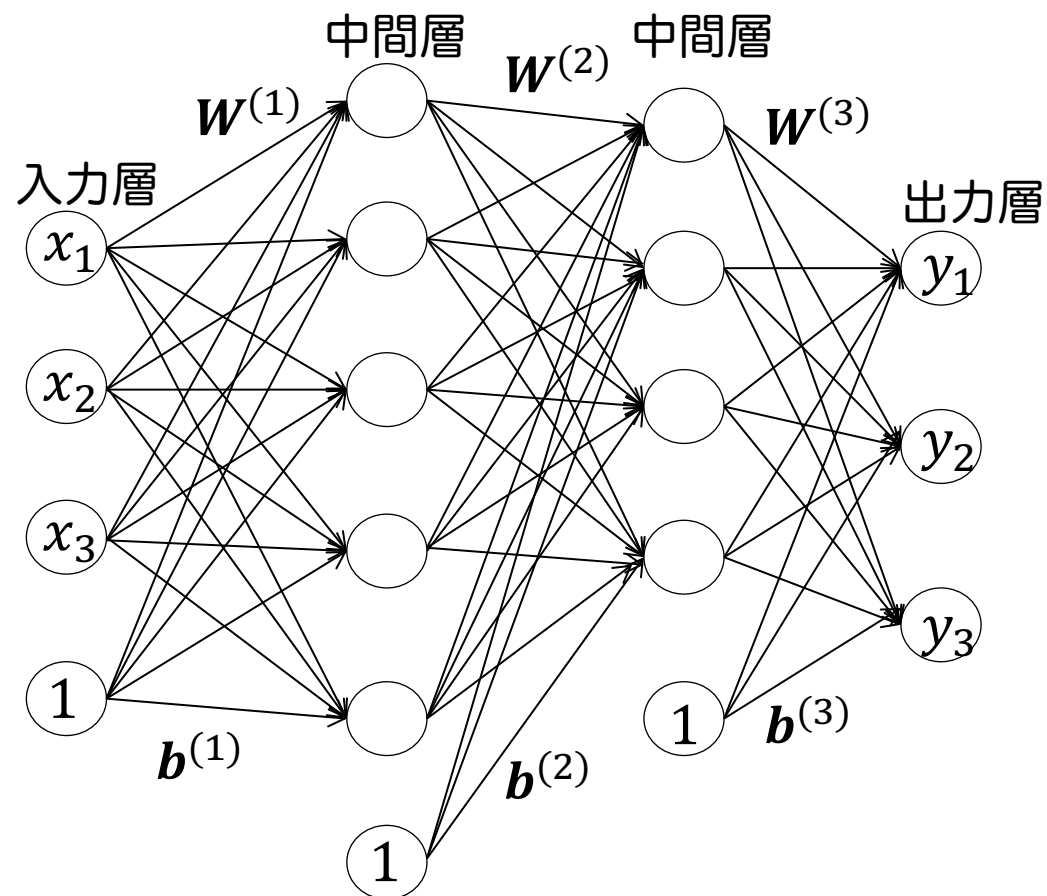
- 多層パーセプトロンとは、入力層と出力層の間に中間層(隠れ層)があるパーセプトロンのこと



ニューラルネットワーク

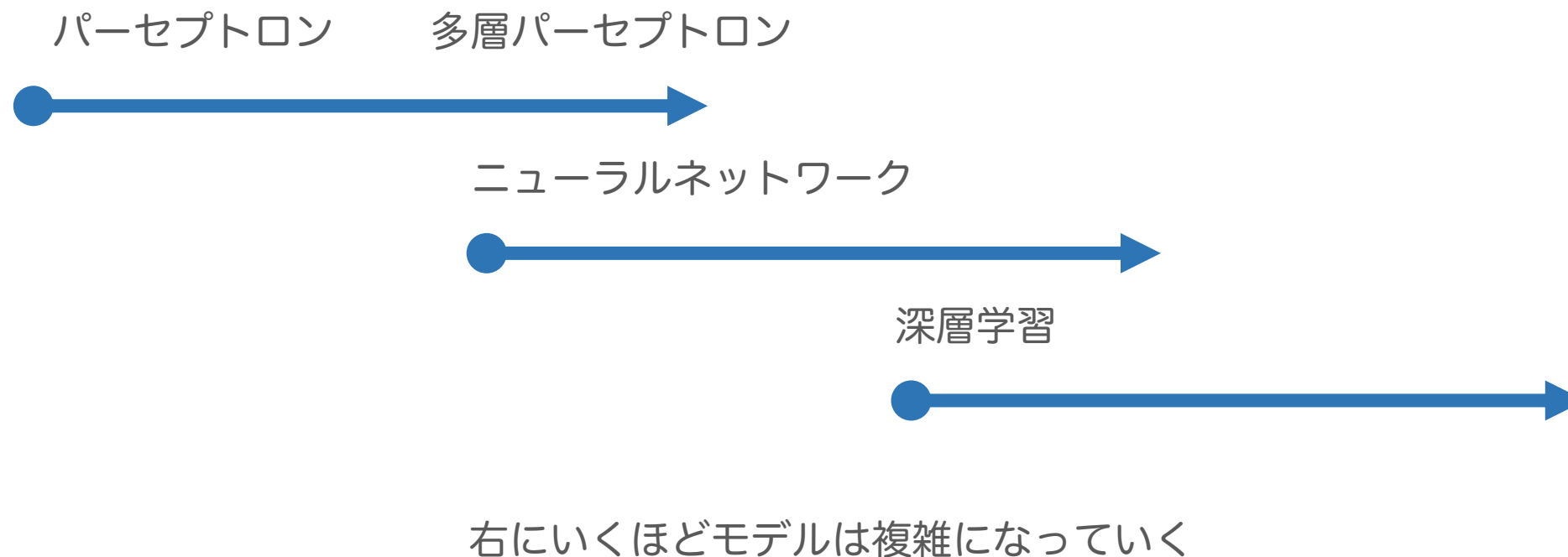
- 多層パーセプトロンとほぼ同じ意味

ニューラルネットワークの例



パーセプトロンから深層学習まで

- パーセプトロン、ニューラルネットワーク、深層学習の言葉の境界はあいまい
- きちんと明確な定義が与えられているわけではない

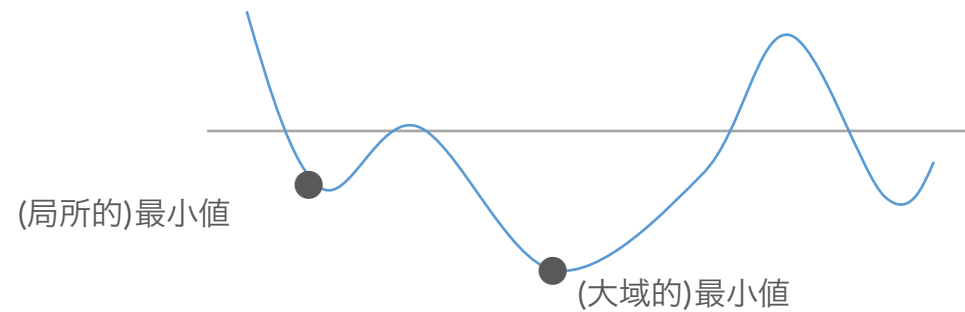
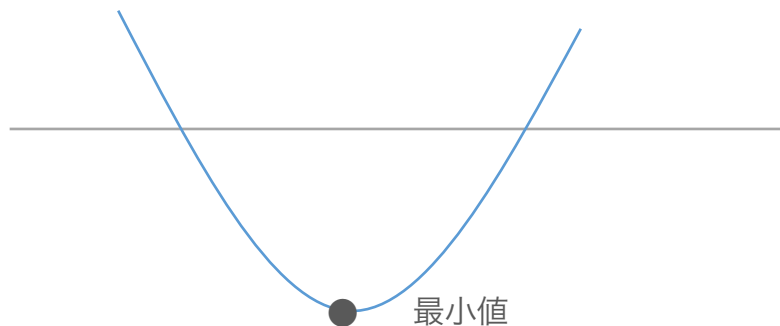


重みはどのように学習すべき？

- ニューラルネットワークにおいて、出力値を決定づけるパラメータは重みベクトルとバイアスベクトルであることがわかった
- （訓練）誤差を最小にする重みとバイアスはどのように学習すべき？
- 誤差関数として、回帰なら二乗誤差、分類ならクロスエントロピー誤差がよく用いられる

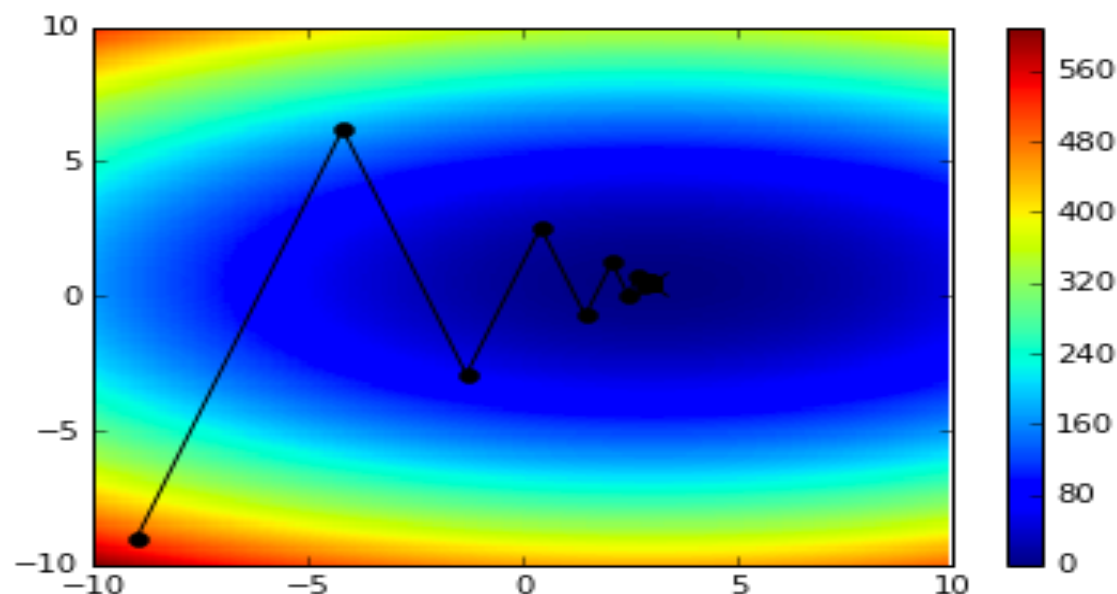
最小値の探索

以下のような関数がある時、どうやって最小値を求めればいいでしょうか？



最急降下法

- 最急降下法とは、関数の勾配が最も急な方向に探索の方向を取りながら最小点にたどり着く方法
- 最も急な方向を探すのに、偏微分を用いる
- ニューラルネットワークの重みとバイアスはこの方向によって学習を行う



ニューラルネットワークの実際の計算では、確率的勾配降下法を用いる。この方法では、データを確率的に選んで勾配を算出しパラメータを更新する。基本原理は、最急降下法とおなじ。

$$f(x, y) = 2x^2 + 3y$$

$$\frac{\partial f}{\partial x} = 4x$$

$$\frac{\partial f}{\partial y} = 3$$

勾配ベクトルとは

$$f(x, y) = 2x^2 + 3y$$

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) = (4x, 3)$$

ナブラ記号を用いる
と勾配ベクトルを簡
潔に表現できる

勾配ベクトル

[演習] 8_gradient_decent.ipynb

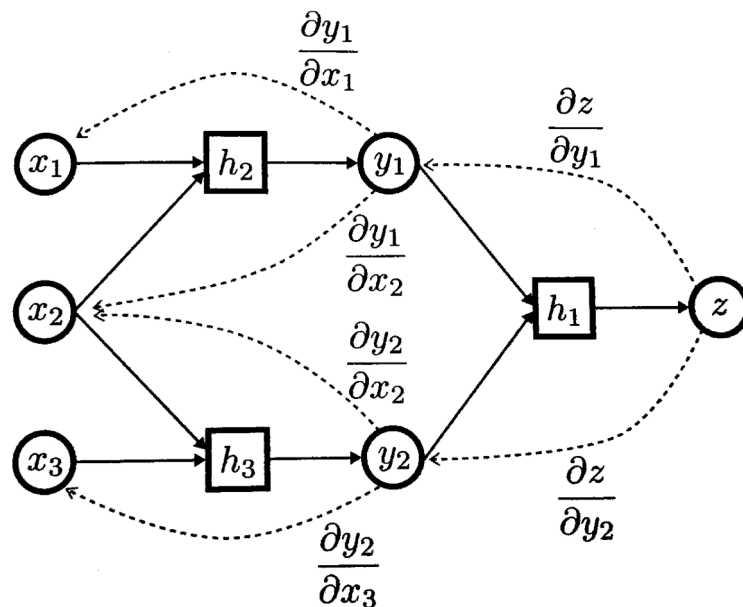
- 最急降下法の振る舞いを、実際のプログラムで確認しましょう
- 初期値や学習率を変更したとき、どのような変化が起こるか確認してみましょう

重みはどのように学習すべき？

- 重みを学習するためには、勾配ベクトル（＝各重みごとの偏微分値）を求めれば良いことがわかった
 - 偏微分値は一般的に繰り返し計算によって求められる
- もっと層を深くして、もっとユニットの数を増やすそうとすると、重みの数も当然増えていく
- たとえば10万個の重みパラメータがあった場合、その全てについて偏微分値を繰り返し計算で求めるのは非常に時間がかかる
- この問題を解決するためには、どのように改善すればよいだろうか？

誤差逆伝播法

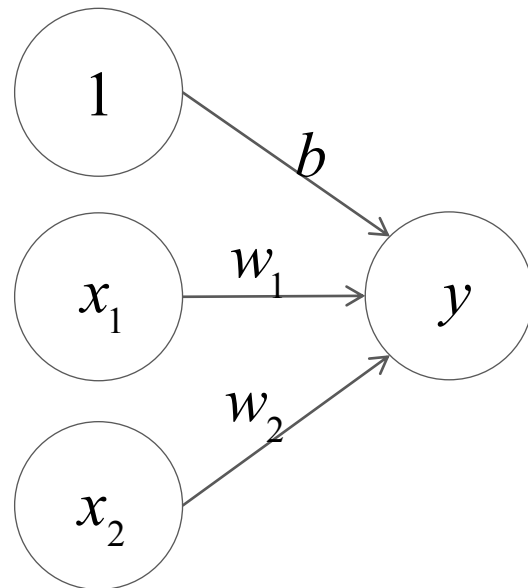
- 勾配降下法で最小値を求めるには各変数の微分値（偏微分値）が必要
- 誤差逆伝播法は、出力層から入力層へと向かって偏微分値を伝播させていく方法
 - 連鎖律の原理(合成関数の微分) を用いている



$$\frac{\partial z}{\partial x_1} = \frac{\partial z}{\partial y_1} \frac{\partial y_1}{\partial x_1}$$

純粋に繋げばいいわけではない！

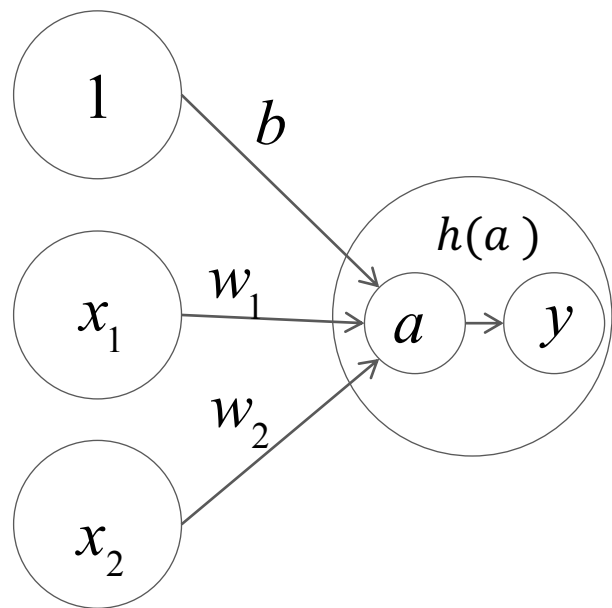
- ニューラルネットワークの一部を取り出してみる
- この結合を以下のように計算すると、単なる線形結合になってしまう
 - すなわち、線形回帰モデルとほぼ同じ計算モデルになってしまう
- 非線形関係を捉えられるようにするには、どのように改善すべき？



$$y = b + w_1x_1 + w_2x_2$$

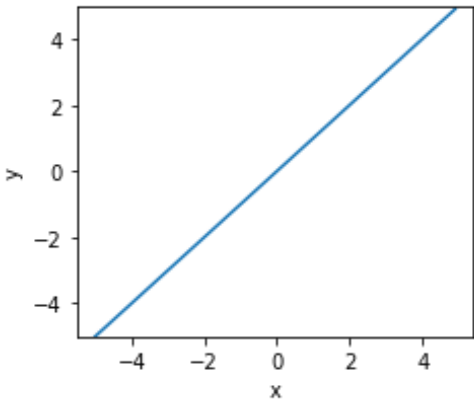
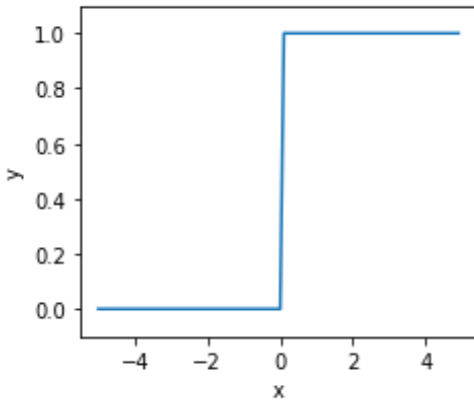
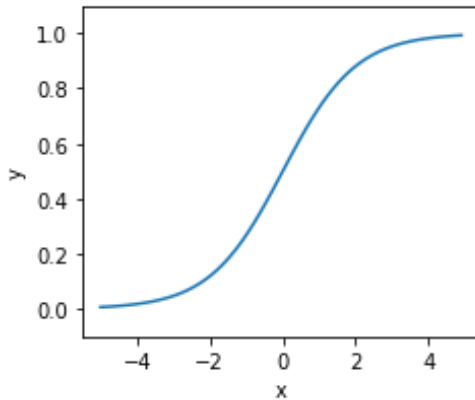
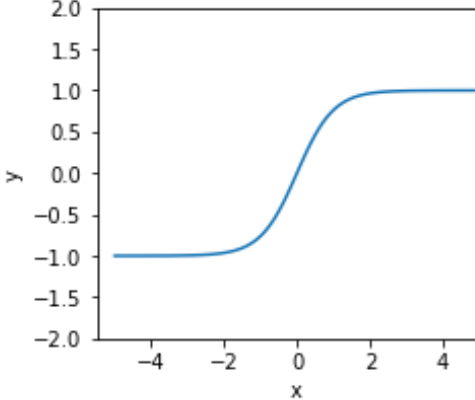
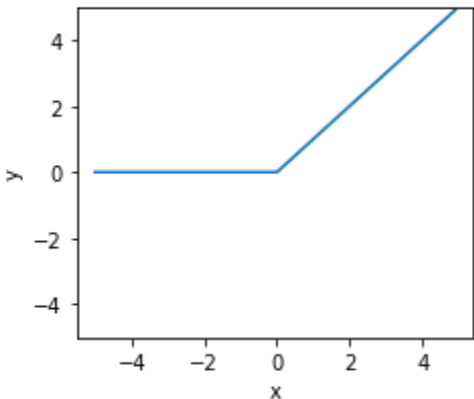
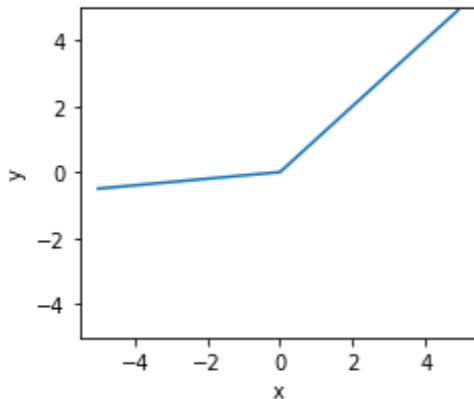
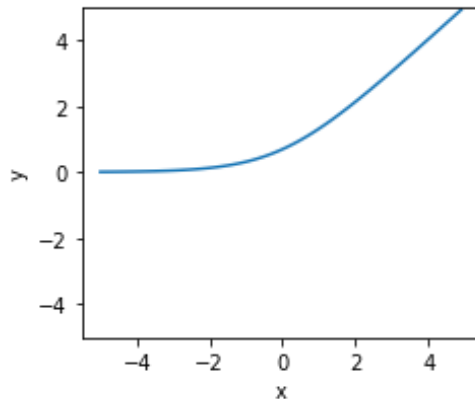
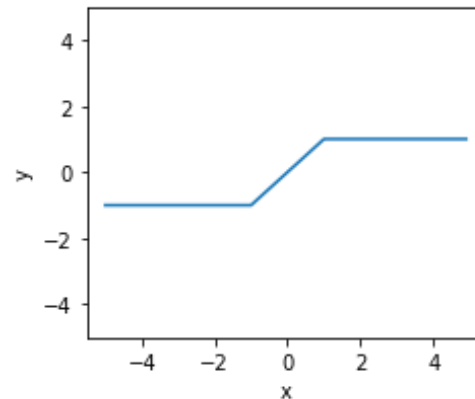
活性化関数

- 出力を非線形関数によって変換することで解決
- 出力に施す非線形関数 $h(a)$ は活性化関数と呼ばれる
- 活性化関数 $h(a)$ があると、 x と y の関係が非線形化され、ニューラルネットワークでモデリングする意義が出てくる



$$y = h(b + w_1x_1 + w_2x_2)$$

活性化関数の種類

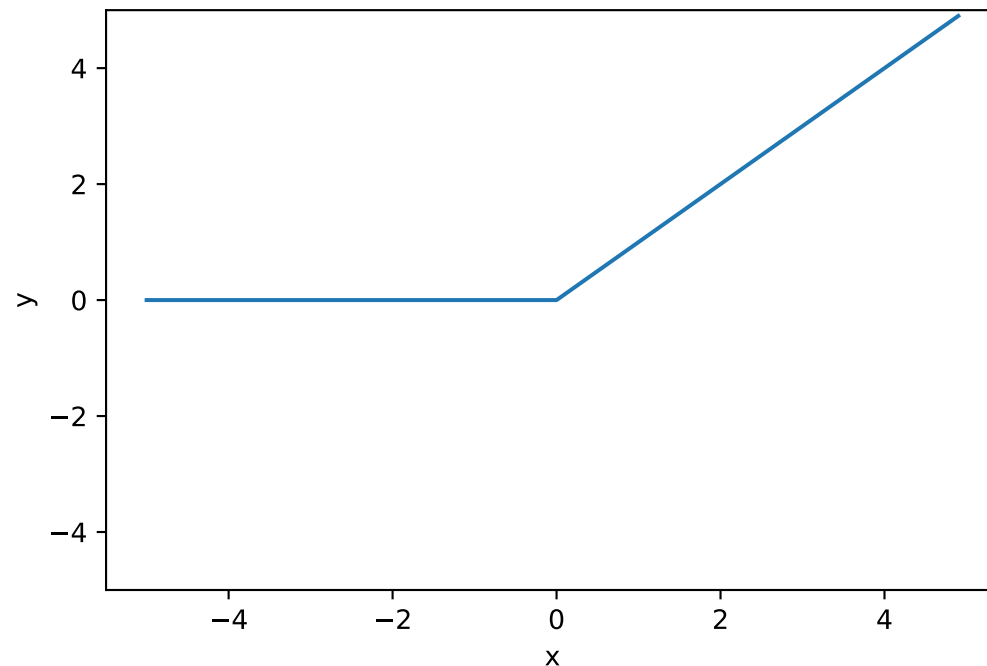
恒等写像関数	ステップ関数	シグモイド関数	tanh関数
			
ReLU関数	LeakyReLU関数	Softplus関数	Hardtanh関数
			

[演習] 9_NN_practice.ipynb

- ニューラルネットワークを実装するのに便利なライブラリTensorFlow & Kerasを用いて、アヤメの分類を行ってみましょう
- ニューラルネットワークの学習に関するハイパーパラメータを変更し、変化を確認してみましょう

[演習] 10_activate_function_trainee.ipynb

- 活性化関数の実装とその形状を確認してみましょう
- 活性化関数として最もよく用いられるReLU関数を実装してみましょう



ニューラルネットワークまとめ（モデル、評価基準、最適化の観点から）

- ニューラルネットワーク：入力を重み付けするパーセプトロンを複数かつ多層に重ねたモデル
- パーセプトロンの出力に活性化関数を通すことで、データの非線形の関係を抑えることができる

モデル

- ニューラルネットワーク



評価基準

- 回帰→二乗誤差
- 分類→交差エントロピー



最適化

- 確率的勾配降下法

[グループワーク] シンプルなモデルと複雑なモデル

- ランダムフォレスト、ニューラルネットワークなどの複雑で強力なモデルを取り上げましたが、線形回帰やロジスティック回帰などシンプルなモデルもまだまだ現役です
- シンプルなモデルが未だに使われているのはなぜでしょうか？その理由をできるだけ多く挙げてみましょう（5分）
- それをもとに複雑なモデルを用いるべきケースを多く挙げてみましょう（10分）
- 最後に以上2つで議論した結果を全体に向けて発表しましょう
（各グループ2分ずつ）

Any Questions ?