

日本ディープラーニング協会
エンジニア(E)資格対応

機械学習講座 (理論と実践)

Study AI

①機械学習の基本的な手法を理解し実装する

- ▶ 線形回帰
- ▶ ロジスティック回帰
- ▶ 主成分分析
- ▶ K平均法など

①機械学習の基本的な手法を理解し実装する

②機械学習モデリングの流れを理解

機械学習モデリングプロセス (プロセスは深層学習でも同じ)

1. 問題設定

どのような課題を
機械学習に解決させるか(※)

2. データ選定

どのようなデータを使うか

3. データの前処理

モデルに学習させられるよう
にデータを変換

4. 機械学習
モデルの選定

どの機械学習モデルを
利用するか

5. モデルの学習
(パラメータ推定)

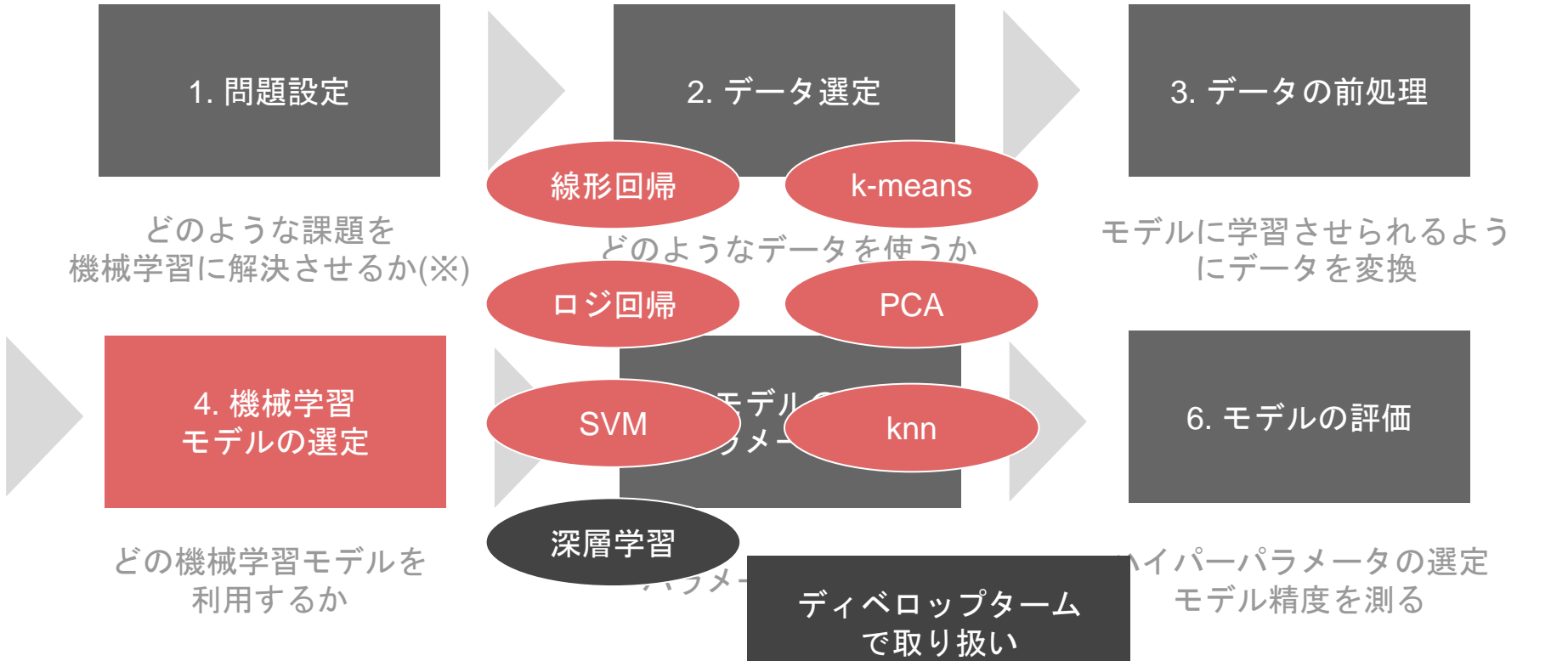
パラメータの決め方

6. モデルの評価

ハイパーパラメータの選定
モデル精度を測る

※問題設定時に必ずしも機械学習を使う必要はないが本講義は機械学習を使うことを前提として進める

機械学習モデリングプロセス (プロセスは深層学習でも同じ)



※問題設定時に必ずしも機械学習を使う必要はないが本講義は機械学習を使うことを前提として進める

本講義で扱う内容

学習種類	タスク	機械学習モデル	パラメータの 推定問題	モデル 選択・評価
教師あり学習	予測	線形回帰・非線形回帰	最小2乗法・尤度最大化	ホールドアウト法 交差検証法
	分類	ロジスティック回帰	尤度最大化 (最尤法)	
		最近傍・K-近傍アルゴリズム		
		サポートベクターマシン	マージン最大化	
教師なし学習	クラスタリング	K-meansアルゴリズム		無し
	次元削減	主成分分析	分散最大化	

※シラバスに含まれる手法のみ

線形回帰とロジスティック回帰でモデリングプロセスを体験

学習種類	タスク	機械学習モデル		パラメータの 推定問題	モデル 選択・評価
教師あり学習	予測	線形回帰・非線形回帰		最小2乗法・尤度最大化	ホールドアウト法 交差検証法
	分類	ロジスティック回帰		尤度最大化 (最尤法)	
		最近傍・K-近傍アルゴリズム			
		サポートベクターマシン	マージン最大化		
教師なし学習	クラスタリング	K-meansアルゴリズム			無し
	次元削減	主成分分析	分散最大化		

※シラバスに含まれる手法のみ

数式は省きエッセンスのみ紹介

学習種類	タスク	機械学習モデル		パラメータの 推定問題	モデル 選択・評価
教師あり学習	予測	線形回帰・非線形回帰		最小2乗法・尤度最大化	ホールドアウト法 交差検証法
	分類	ロジスティック回帰		尤度最大化 (最尤法)	
		最近傍・K-近傍アルゴリズム			
		サポートベクターマシン	マージン最大化		
教師なし学習	クラスタリング	K-meansアルゴリズム			無し
	次元削減	主成分分析	分散最大化		

※シラバスに含まれる手法のみ

本講義で扱う内容

機械学習と
モデリングプロセス
(50 min)

休憩
(10)

「線形回帰」
講義・ハンズオン (toyデータ)
(50 min)

休憩
(10)

「非線形回帰」
「ロジスティック回帰」
講義
(50 min)

休憩
(60)

「ロジスティック回帰」
講義・ハンズオン (kaggleデータ)
(50 min)

休憩
(10)

主成分分析
講義・ハンズオン (toyデータ)
(50 min)

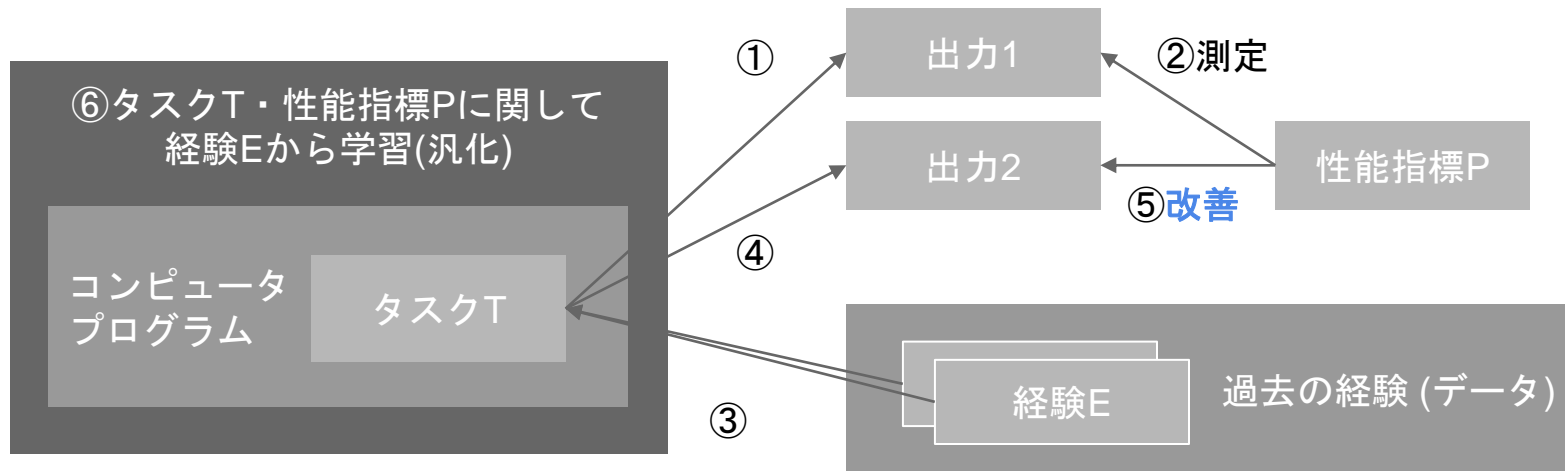
休憩
(10)

k-meansとknn
講義・ハンズオン (toyデータ)
(50 min)

講義本編

● 機械学習

- コンピュータプログラムは、タスクT(アプリケーションにさせたいこと)を性能指標Pで測定し、その性能が経験E(データ)により改善される場合、タスクTおよび性能指標Pに関して経験Eから学習すると言われている (トム・ミッチェル 1997)
- 人がプログラムするのは認識の仕方ではなく学習の仕方 (数学で記述)



線形回帰モデル

1. 講義

- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. 線形回帰モデルの説明
 - iii. パラメータの推定
 - iv. モデルの評価

1. ハンズオン

- a. pandasとnumpyの紹介
- b. scikit-learnの紹介
- c. 住宅価格予測

線形回帰モデル

● 回帰問題

- ある入力(離散あるいは連続値)から出力(連続値)を予測する問題
 - 直線で予測 ▶ 線形回帰
 - 曲線で予測 ▶ 非線形回帰

● 回帰で扱うデータ

- 入力 (各要素を説明変数または特徴量と呼ぶ)
 - m 次元のベクトル ($m=1$ の場合はスカラー)
- 出力 (目的変数)
 - スカラー値 (目的変数)

説明変数

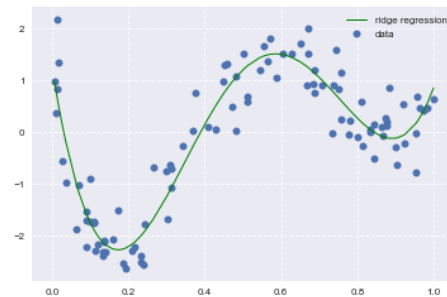
$$\mathbf{x} = (x_1, x_2, \dots, x_m)^T \in \mathbb{R}^m$$

目的変数

$$y \in \mathbb{R}^1$$



線形



非線形

線形回帰モデル

教師データ

● 線形回帰モデル

- 回帰問題を解くための機械学習モデルの一つ
- 教師あり学習(教師データから学習)
- 入力とm次元パラメータの線形結合を出力するモデル
 - 慣例として予測値にはハットを付ける (正解データとは異なる)

$$\{(\mathbf{x}_i, y_i); i = 1, \dots, n\}$$

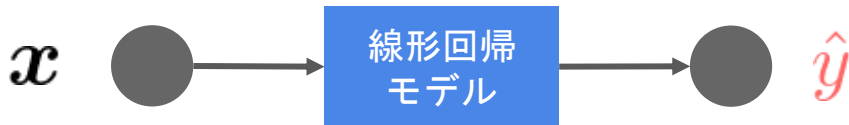


パラメータ

$$\mathbf{w} = (w_1, w_2, \dots, w_m)^T \in \mathbb{R}^m$$

線形結合

$$\hat{y} = \mathbf{w}^T \mathbf{x} + w_0 = \sum_{j=1}^m w_j x_j + w_0$$

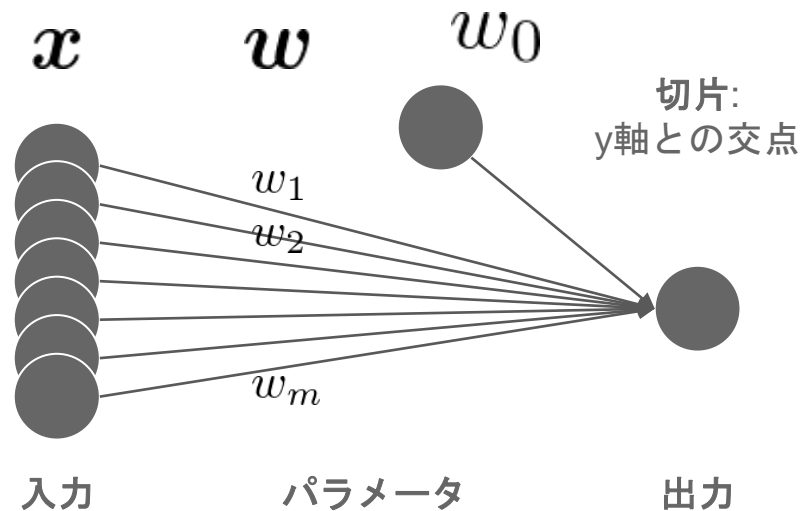


- 線形結合 (入力とパラメータの内積)
 - 入力ベクトルと未知のパラメータの各要素を掛け算し足し合わせたもの
 - 入力ベクトルとの線形結合に加え、切片も足し合わせる
 - (入力のベクトルが多次元でも)出力は1次元(スカラー)となる

線形結合

$$\hat{y} = \underline{w}^T x + \underline{w_0} = \sum_{j=1}^m \underline{w_j} x_j + \underline{w_0}$$

パラメータは未知



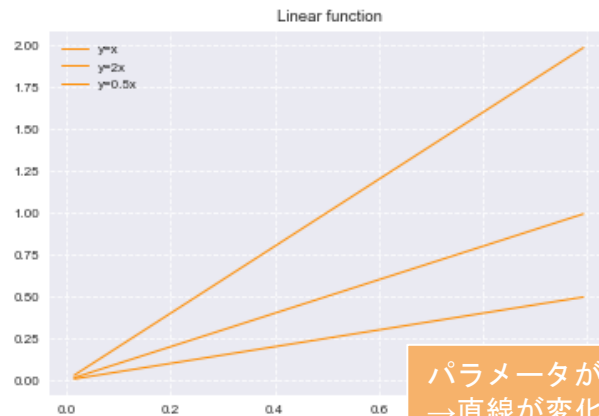
● モデルのパラメータ

- モデルに含まれる推定すべき未知のパラメータ
- 特徴量が予測値に対してどのように影響を与えるかを決定する重みの集合
 - 正の(負の)重みをつける場合、その特徴量の値を増加させると、予測の値が増加(減少)
 - 重みが大きければ(0であれば)、その特徴量は予測に大きな影響力を持つ(全く影響しない)
- 切片
 - y軸との交点を表す

$$\hat{y} = \underline{\mathbf{w}}^T \mathbf{x} + \underline{w_0} = \sum_{j=1}^m \underline{w_j} x_j + \underline{w_0}$$

パラメータは未知

最小二乗法により推定



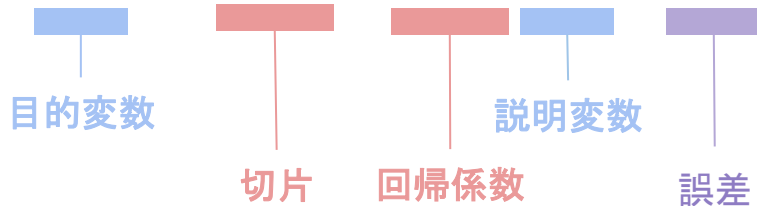
パラメータが変化
→ 直線が変化
→ 全ての線を表示可能

線形回帰モデル

- 説明変数が1次元の場合($m=1$)
 - 単回帰モデルと呼ぶ
- データへの仮定
 - データは回帰直線に誤差が加わり観測されていると仮定

モデル数式

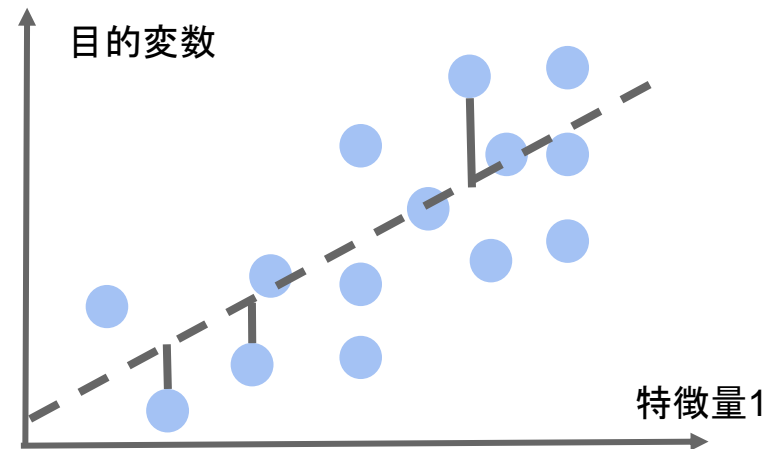
$$y = w_0 + w_1 x_1 + \varepsilon$$



既知：入力データ

未知：学習で決める

幾何学的意味



線形回帰モデル

- 連立方程式

- それぞれのデータをモデル式へ当てはめるとn個の式が導出される

連立方程式

未知パラメータ

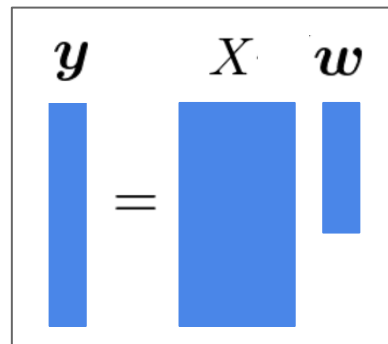
$$y_1 = w_0 + w_1 x_1 + \varepsilon_1$$

$$y_2 = w_0 + w_1 x_2 + \varepsilon_2$$

$$y_n = w_0 + w_1 x_n + \varepsilon_n$$

行列表現

$$\mathbf{y} = X\mathbf{w} + \boldsymbol{\varepsilon}$$


$$\mathbf{y} = X\mathbf{w}$$

$$X = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$$

$$\mathbf{y} = (y_1, y_2, \dots, y_n)^T$$

$$\mathbf{x}_i = (1, x_i)^T$$

$$\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n)^T$$

$$\mathbf{w} = (w_0, w_1)^T$$

線形回帰モデル

- 説明変数が多次元の場合($m>1$)
 - 線形重回帰モデル
 - 単回帰は直線 ▶ 重回帰は曲面
- データへの仮定
 - データは回帰曲面に誤差が加わり観測されていると仮定

モデル数式

$$y = w_0 + w_1x_1 + w_2x_2 + \varepsilon$$

目的変数

切片

回帰係数

回帰係数

誤差

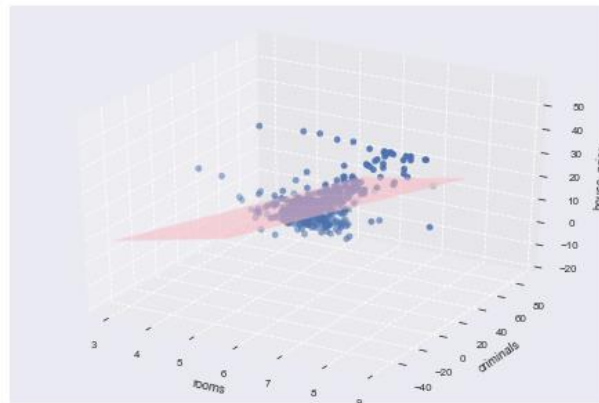
説明変数

説明変数

既知：入力データ

未知：学習で決める

幾何学的意味



線形回帰モデル

単回帰の場合と全く同じ

未知パラメータは
最小二乗法により推定

連立方程式

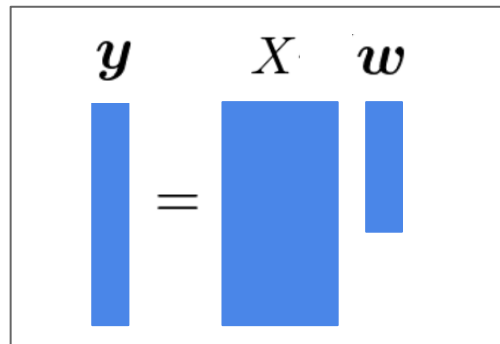
$$y_1 = w_0 + w_1 x_{11} + w_2 x_{12} + \cdots + w_m x_{1m} + \varepsilon_1$$

$$y_2 = w_0 + w_1 x_{21} + w_2 x_{22} + \cdots + w_m x_{2m} + \varepsilon_2$$

$$y_n = w_0 + w_1 x_{n1} + w_2 x_{n2} + \cdots + w_m x_{nm} + \varepsilon_n$$

行列表現

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}$$



$$\mathbf{x}_i = (1, x_{i1}, \cdots, x_{im})^T$$

$$\mathbf{y} = (y_1, y_2, \cdots, y_n)^T$$

$$\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)^T$$

$$\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n)^T$$

$$\mathbf{w} = (w_0, w_1, \cdots, w_m)^T$$

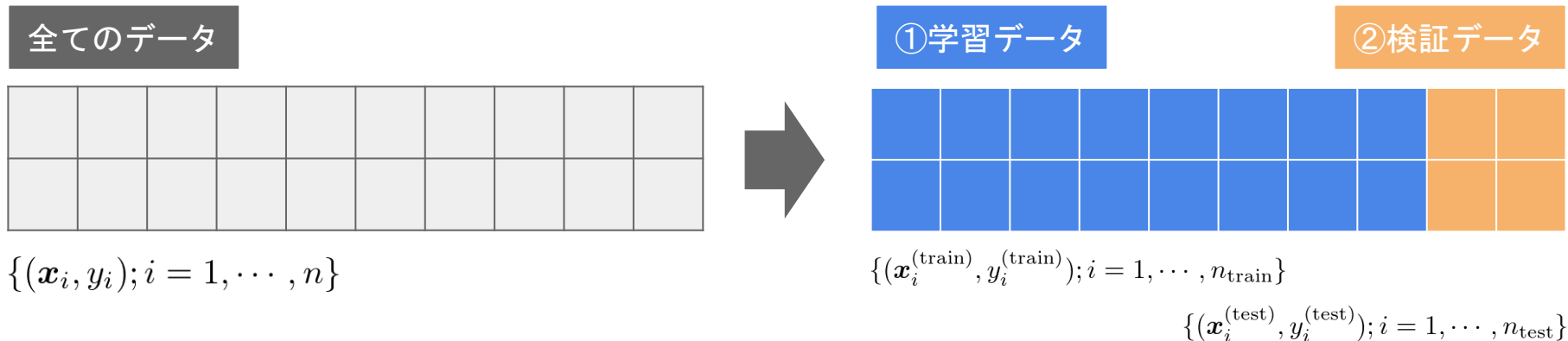
データの分割とモデルの汎化性能測定

- データの分割

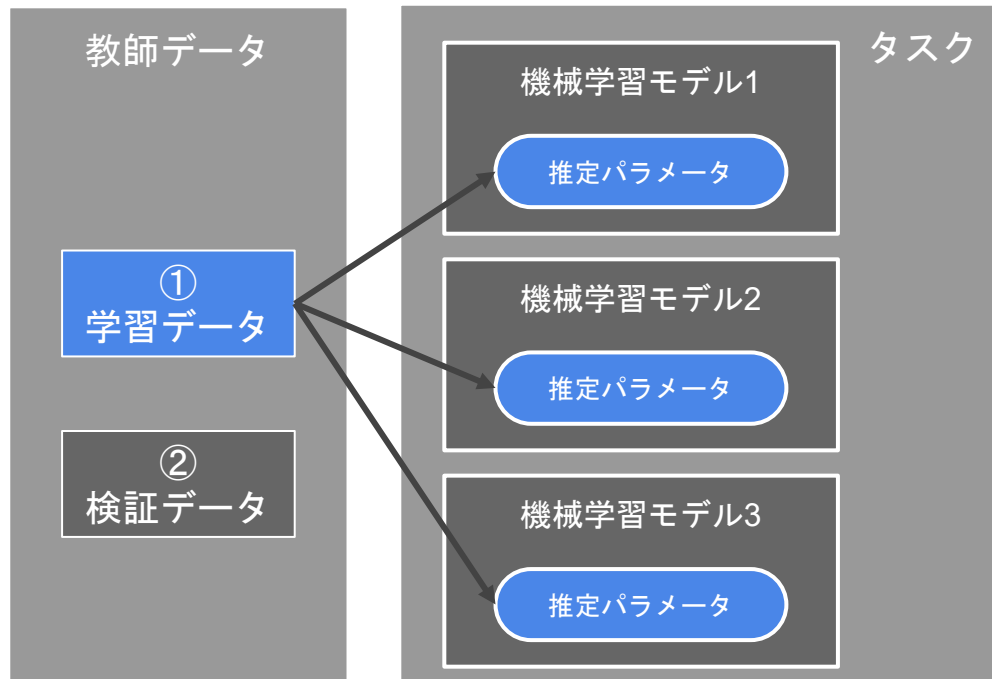
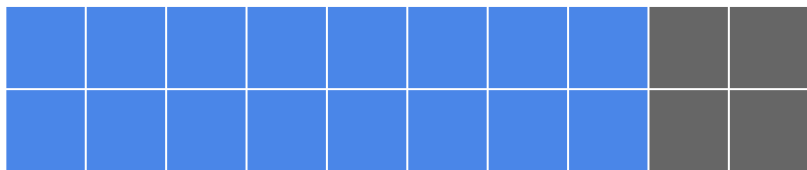
- 学習用データ：機械学習モデルの学習に利用するデータ
- 検証用データ：学習済みモデルの精度を検証するためのデータ

- なぜ分割するか

- モデルの汎化性能(Generalization)を測定するため
- データへの当てはまりの良さではなく、未知のデータに対してどれくらい精度が高いかを測りたい



データの分割とモデルの汎化性能測定



学習データの使い方

$$\{(\boldsymbol{x}_i^{(\text{train})}, y_i^{(\text{train})}); i = 1, \dots, n_{\text{train}}\}$$

を利用して推定パラメータを学習

データの分割とモデルの汎化性能測定

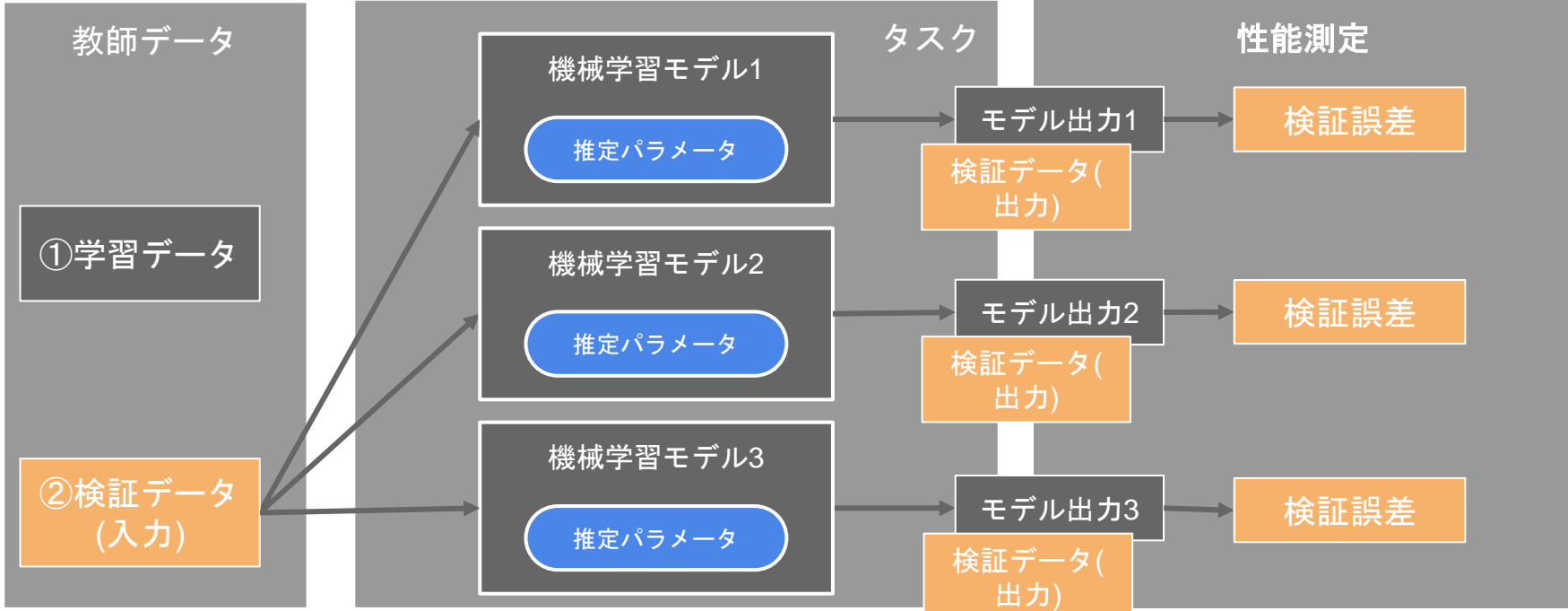
学習に使ったデータと別のデータで検証
(モデルの未来のデータへの
当てはまりの良さ = 汎化性能)

学習誤差

$$MSE_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

検証誤差

$$MSE_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i^{(\text{test})} - y_i^{(\text{test})})^2$$



$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

データの分割とモデルの汎化性能測定

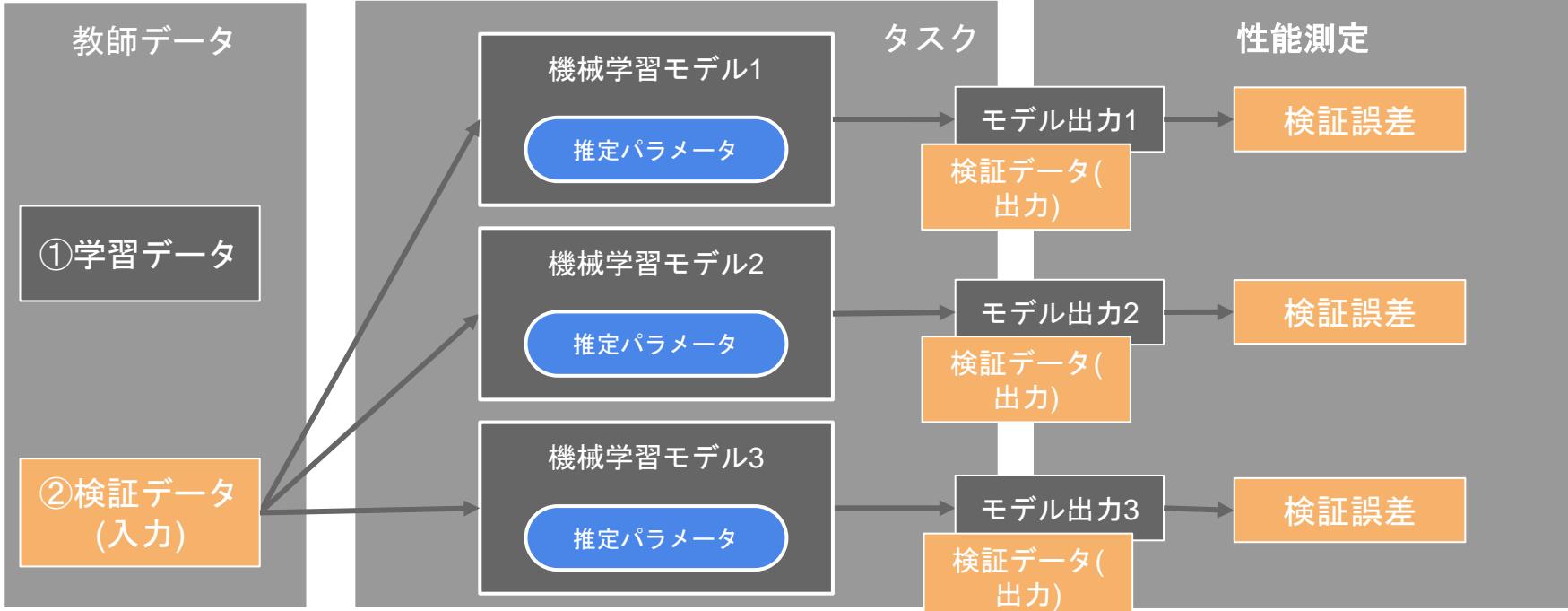
学習に使ったデータと別のデータで検証
(モデルの未来のデータへの
当てはまりの良さ = 汎化性能)

学習誤差

$$MSE_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

検証誤差

$$MSE_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i^{(\text{test})} - y_i^{(\text{test})})^2$$



$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

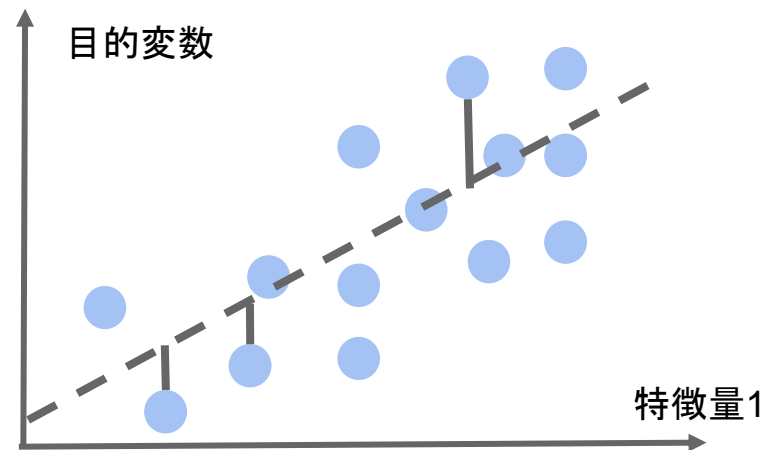
線形回帰モデルのパラメータは最小二乗法で推定

- 平均二乗誤差 (残差平方和)

- データとモデル出力の二乗誤差の和
 - 小さいほど直線とデータの距離が近い
 - 動画解説
- パラメータのみに依存する関数
 - データは既知の値でパラメータのみ未知

- 最小二乗法

- 学習データの平均二乗誤差を最小とするパラメータを探索
- 学習データの平均二乗誤差の最小化は、その勾配が0になる点を求めれば良い



$$\text{MSE}_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

線形回帰モデル

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w} \in \mathbb{R}^{m+1}} \text{MSE}_{\text{train}}$$

MSEを最小にするようなw(m次元)

$$\frac{\partial}{\partial \mathbf{w}} \text{MSE}_{\text{train}} = 0$$

MSEをwに関して微分したものが0
となるwの点を求める

$$\Rightarrow \frac{\partial}{\partial \mathbf{w}} \left\{ \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2 \right\} = 0$$

平均二乗誤差(残差平方和)を微分

行列表現

$$\Rightarrow \frac{1}{n_{\text{train}}} \frac{\partial}{\partial \mathbf{w}} \left\{ (X^{(\text{train})} \mathbf{w} - \mathbf{y}^{(\text{train})})^T (X^{(\text{train})} \mathbf{w} - \mathbf{y}^{(\text{train})}) \right\}$$

$$\Rightarrow \frac{1}{n_{\text{train}}} \frac{\partial}{\partial \mathbf{w}} \left\{ \mathbf{w}^T X^{(\text{train})T} X^{(\text{train})} \mathbf{w} - 2\mathbf{w}^T X^{(\text{train})T} \mathbf{y}^{(\text{train})} + \mathbf{y}^{(\text{train})T} \mathbf{y}^{(\text{train})} \right\}$$

{ } 展開

$$\Rightarrow 2X^{(\text{train})T} X^{(\text{train})} \mathbf{w} - 2X^{(\text{train})T} \mathbf{y}^{(\text{train})} = 0$$

行列微分計算

$$\Rightarrow \hat{\mathbf{w}} = (X^{(\text{train})T} X^{(\text{train})})^{-1} X^{(\text{train})T} \mathbf{y}^{(\text{train})}$$

回帰係数

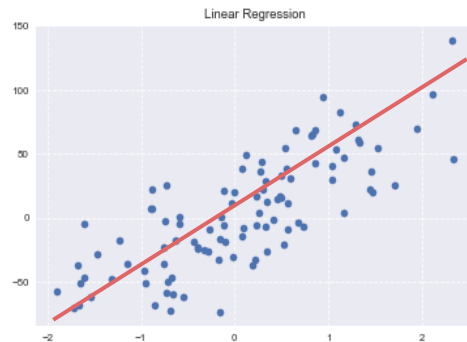
線形回帰モデル

回帰係数

$$\hat{\boldsymbol{w}} = (X^{(\text{train})T} X^{(\text{train})})^{-1} X^{(\text{train})T} \boldsymbol{y}^{(\text{train})}$$

予測値

$$\hat{\boldsymbol{y}} = X_{n_{\text{new}} \times m+1} (X^{(\text{train})T} X^{(\text{train})})^{-1} X^{(\text{train})T} \boldsymbol{y}^{(\text{train})}$$



- [topic] 最尤法による回帰係数の推定

- 誤差を正規分布に従う確率変数を仮定し尤度関数の最大化を利用した推定も可能
- 線形回帰の場合には、最尤法による解は最小二乗法の解と一致

線形回帰モデル

1. 講義

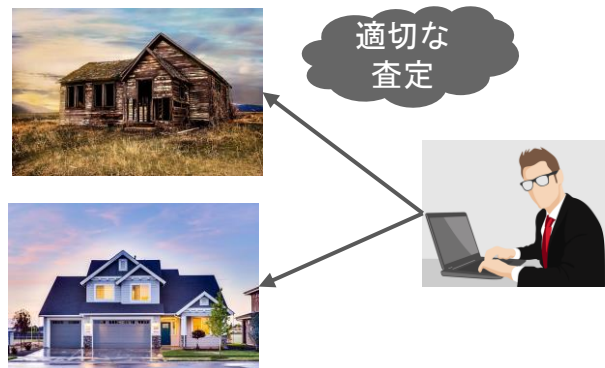
- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. 線形回帰モデルの説明
 - iii. パラメータの推定
 - iv. モデルの評価

1. ハンズオン

- a. pandasとnumpyの紹介
- b. scikit-learnの紹介
- c. 住宅価格予測

線形回帰モデル

- 設定
 - ボストンの住宅データセットを線形回帰モデルで分析
 - 適切な査定結果が必要
 - 高すぎても安すぎても会社に損害がある
- 課題
 - 部屋数が4で犯罪率が0.3の物件はいくらになるか？



データセット名	ボストンの住宅データセット (ボストン市の住宅価格)		
レコード数	506	<div>Boston Housingデータセットは、現在倫理的な問題から非推奨となっております。</div> <div>配布ソースコード「<code>skl_regression.ipynb</code>」は使用できませんので、動画閲覧で学習を進めていただきますようお願いいたします。</div> <div>※レポートでの実装演習キャプチャーの提出は不要です。</div>	
カラム数	14		
詳細	UCI Machine Learning Repository: Housing Data Set		
備考	よく線形回帰モデルのテストセットとして使用される		

項番	変数	記号名	備考
1	犯罪発生率	CRIM	(人口単位)
2	25000平方フィート以上の住宅 <u>区間</u> の割合	ZN	
3	非小売業の土地面積の割合	INDUS	(人口単位)
4	チャールズ川沿いかどうかフラグ	CHAS	(川沿いなら1、そうでなければ0)
5	窒素感化物の濃度	NOX	(pphm単位)
6	一戸あたりの平均部屋数	RM	2,3など(単位は部屋)
7	1940年よりも前に建てられた家屋の割合	AGE	
8	ボストンの主な5つの雇用圏までの重み付き距離	DIS	
9	幹線道路へのアクセス指数	RAD	
10	10000ドルあたりの <u>所得税</u> 率	TAX	
11	教師あたりの生徒の数	PTRATIO	(人口単位)
12	$1000(B_k - 0.63)^2$ として計算される量	B	(B _k はアフリカ系アメリカ人居住者の割合(人口単位))
13	<u>低所得者</u> の割合	LSTAT	
14	住宅価格	MEDV	中央値(単位は1000ドル)

Scikit-learn: PythonのOpen-Sourceライブラリ

特徴：様々な回帰・分類・クラスタリングアルゴリズムが実装されており、Pythonの数値計算ライブラリのNumPyとSciPyとやりとりするように設計されている。(先ほどのPandasはデータ表示・解析に利用)

URL: <https://scikit-learn.org/stable/>



利用するライブラリの説明

Pandas:

Excel のような2次元のテーブルを対象にしたデータ解析を支援する機能を提供するライブラリです。DataFrame が pandasのメインとなるデータ構造で2次元のテーブルを表します。

Numpy:

値計算を効率的に行うための拡張モジュールである。効率的な数値計算を行うための型付きの多次元[配列](#)（例えばベクトルや行列などを表現できる）のサポートをPythonに加えるとともに、それら进行操作するための大規模な[高水準の数学関数](#)ライブラリを提供する。

統計処理、機械学習を行う際に共によく利用

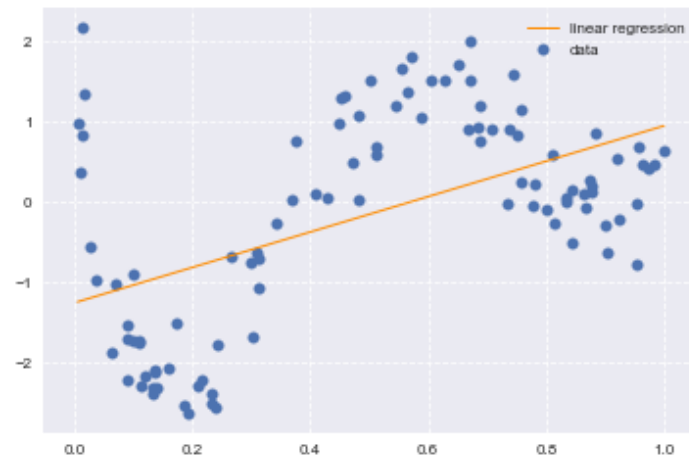
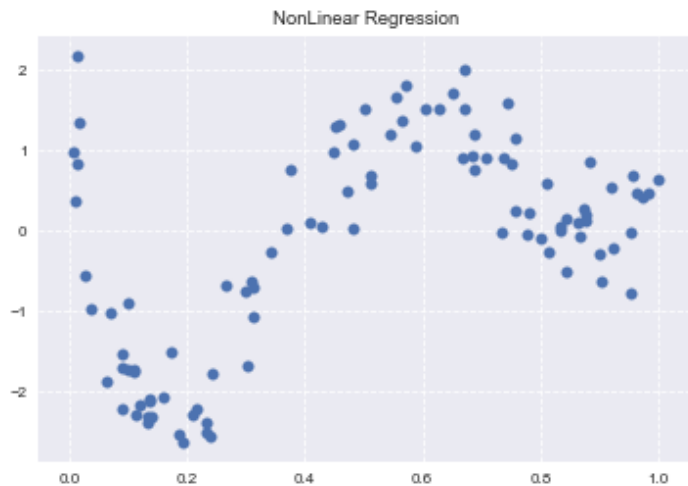
非線形回帰モデル

1. 講義

- a. アウトライン
- b. 数学的定式化
 - i. 非線形回帰モデルの説明
 - ii. パラメータの推定
 - iii. バイアスバリエアンスのトレードオフ
 - iv. クロスバリデーション

非線形回帰モデル

- 複雑な非線形構造を内在する現象に対して、非線形回帰モデリングを実施
 - データの構造を線形で捉えられる場合は限られる
 - 非線形な構造を捉えられる仕組みが必要



● 基底展開法

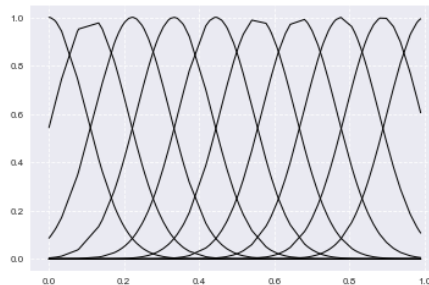
- 回帰関数として、基底関数と呼ばれる既知の非線形関数とパラメータベクトルの線型結合を使用
- 未知パラメータは線形回帰モデルと同様に最小2乗法や最尤法により推定

$$y_i = f(\mathbf{x}_i) + \varepsilon_i$$

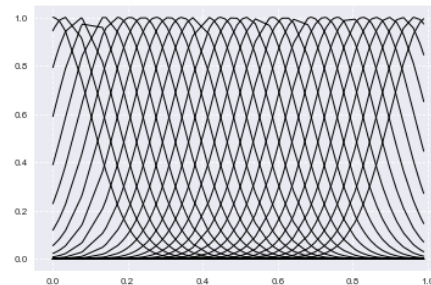
$$y_i = w_0 + \sum_{j=1}^m w_j \phi_j(\mathbf{x}_i) + \varepsilon_i$$

● よく使われる基底関数

- 多項式関数
- ガウス型基底関数
- スプライン関数 / Bスプライン関数



ガウス型基底 (10)

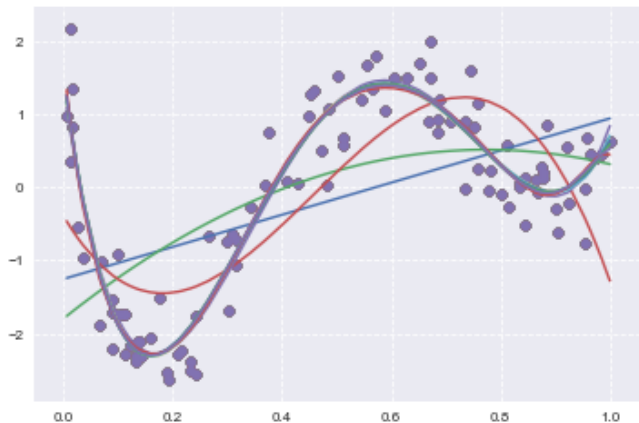


ガウス型基底 (30)

1次元の基底関数に基づく非線形回帰

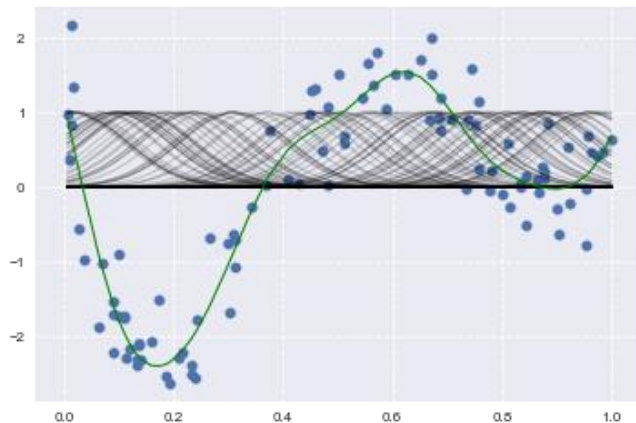
多項式 (1~9次)

$$\phi_j = x^j$$



ガウス型基底

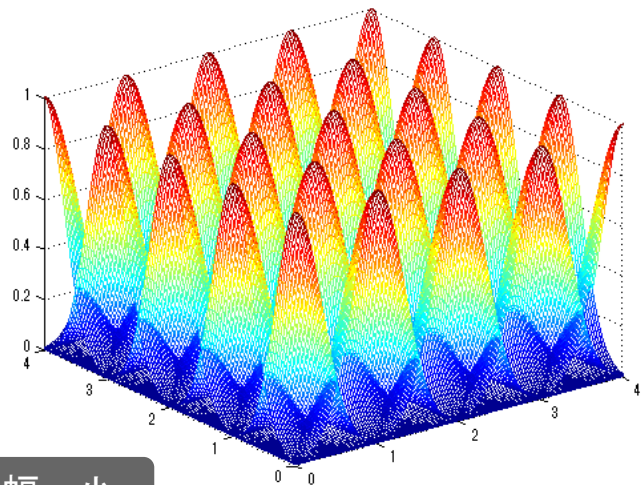
$$\phi_j(x) = \exp \left\{ -\frac{(x - \mu_j)^2}{2h_j} \right\}$$



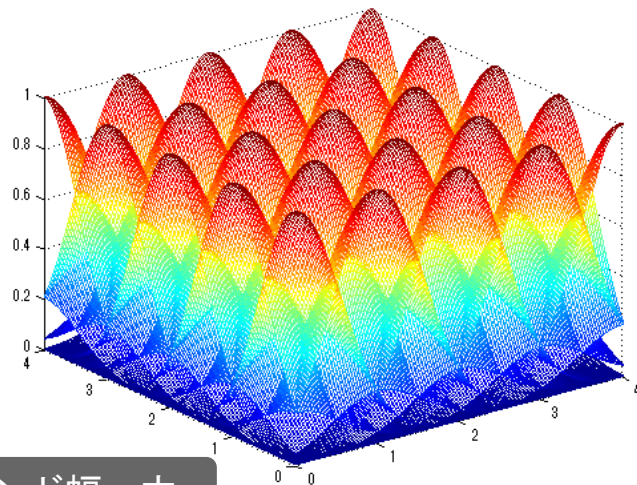
2次元の基底関数に基づく非線形回帰

2次元ガウス型基底関数

$$\phi_j(\mathbf{x}) = \exp \left\{ \frac{(\mathbf{x} - \boldsymbol{\mu}_j)^T (\mathbf{x} - \boldsymbol{\mu}_j)}{2h_j} \right\}$$



バンド幅 : 小



バンド幅 : 大

非線形回帰モデル

説明変数

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{im}) \in \mathbb{R}^m$$

説明変数の数

非線形関数ベクトル

$$\phi(\mathbf{x}_i) = (\phi_1(\mathbf{x}_i), \phi_2(\mathbf{x}_i), \dots, \phi_k(\mathbf{x}_i))^T \in \mathbb{R}^k$$

基底関数の数

非線形関数の計画行列

$$\Phi^{(train)} = (\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \dots, \phi(\mathbf{x}_n))^T \in \mathbb{R}^{n \times k}$$

最尤法による予測値

$$\hat{\mathbf{y}} = \Phi(\Phi^{(train)T} \Phi^{(train)})^{-1} \Phi^{(train)T} \mathbf{y}^{(train)}$$

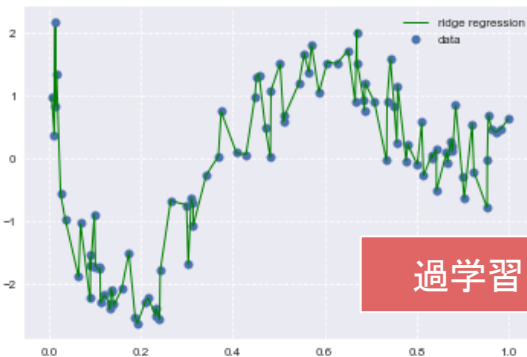
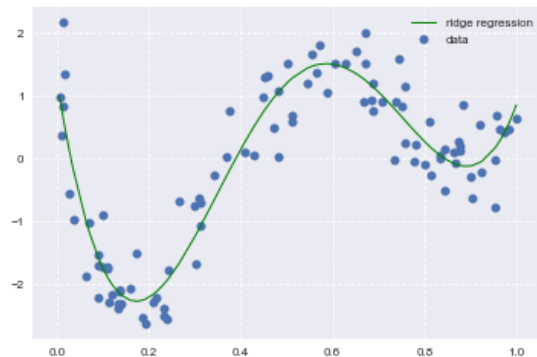
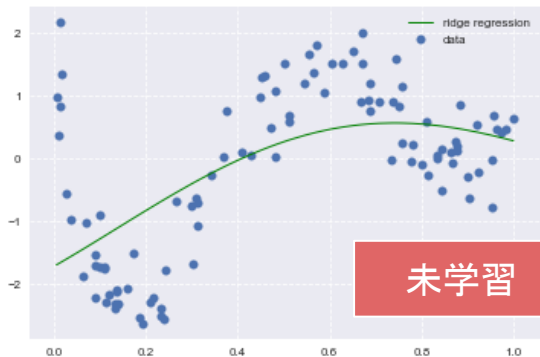
基底展開法も線形回帰と同じ枠組みで推定可能

非線形回帰モデル

- 未学習(underfitting)と過学習(overfitting)

- 学習データに対して、十分小さな誤差が得られないモデル→未学習
 - (対策)モデルの表現力が低いため、表現力の高いモデルを利用する
- 小さな誤差は得られたけど、テスト集合誤差との差が大きいモデル→過学習
 - (対策1) 学習データの数を増やす
 - (対策2) 不要な基底関数(変数)を削除して表現力を抑止
 - (対策3) 正則化法を利用して表現力を抑止

モデルの複雑さを
調整する2つの方法



表現力の低いモデル

適切な表現力のモデル

表現力の高いモデル

※線形回帰モデルの場合には、変数の組み合わせによりモデルの複雑さが変わる (変数選択問題)

- 不要な基底関数を削除

- 基底関数の数、位置やバンド幅によりモデルの複雑さが変化
- 解きたい問題に対して多くの基底関数を用意してしまうと過学習の問題がおこるため、適切な基底関数を用意 (CVなどで選択)

- 正則化法 (罰則化法)

- 「モデルの複雑さに伴って、その値が大きくなる正則化項(罰則項)を課した関数」を最小化
- 正則化項(罰則項)
 - 形状によっていくつもの種類があり、それぞれ推定量の性質が異なる (次スライド)
- 正則化(平滑化)パラメータ
 - モデルの曲線のなめらかさを調節 □適切に決める必要あり

$$S_{\gamma} = (\mathbf{y} - \Phi \mathbf{w})^T (\mathbf{y} - \Phi \mathbf{w}) + \gamma R(\mathbf{w}) \quad \gamma(> 0)$$

$n \times k$

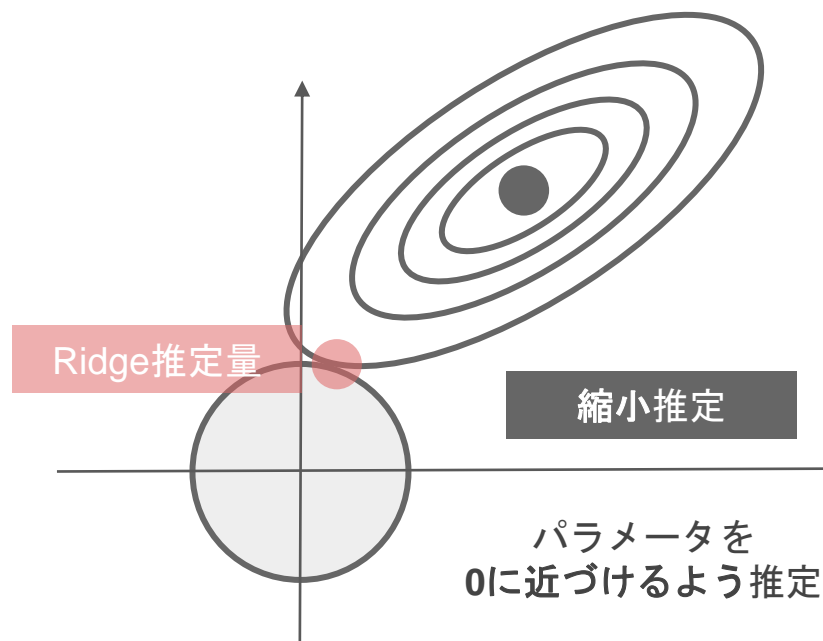
基底関数の数(k)が増加するとパラメータが増加し、
残差は減少 (モデルが複雑化)

モデルの複雑さに伴う罰則

非線形回帰モデル

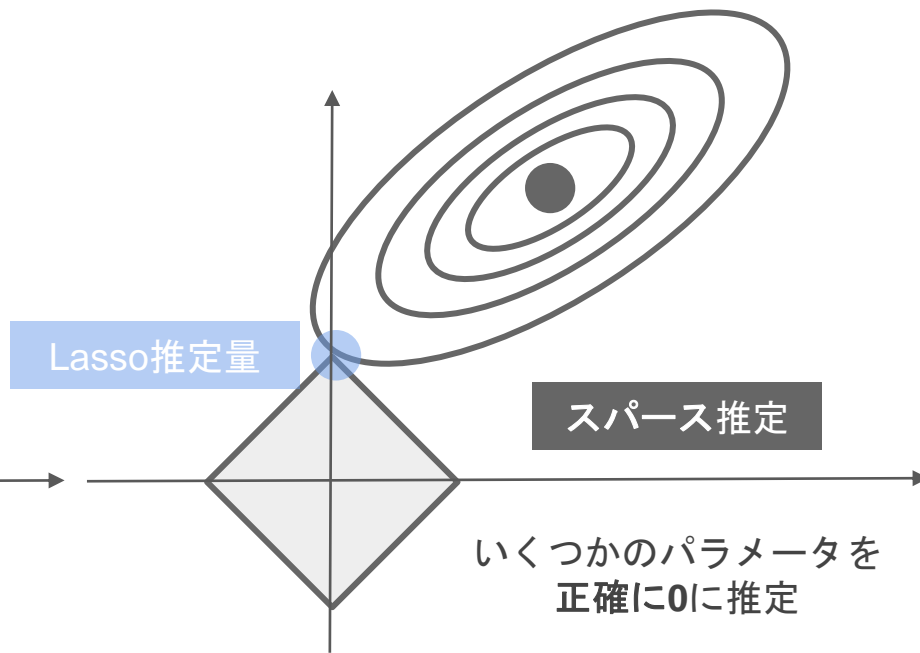
- 正則化項(罰則項)の役割

- 無い ▶ 最小2乗推定量
- L2ノルムを利用 ▶ Ridge推定量
- L1ノルムを利用 ▶ Lasso推定量



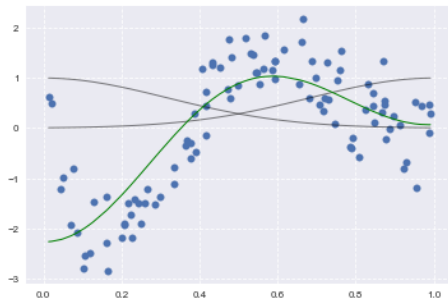
- 正則化パラータの役割

- 小さく ▶ 制約面が大きく
- 大きく ▶ 制約面が小さく

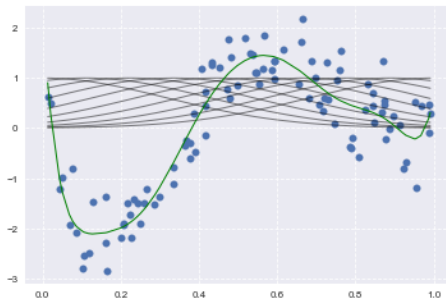


非線形回帰モデル

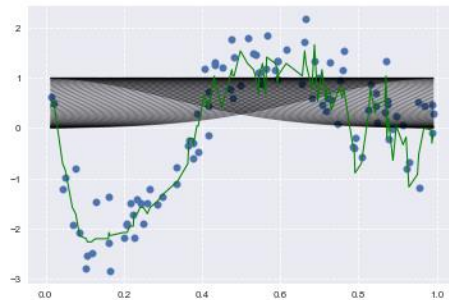
基底関数: 2



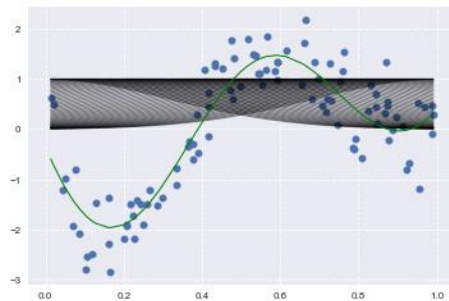
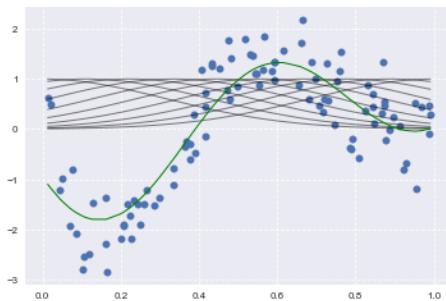
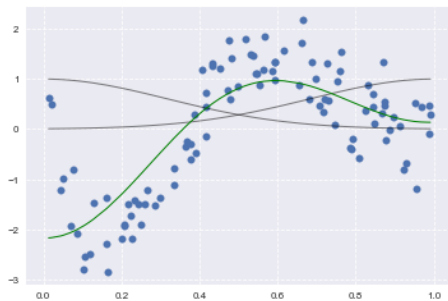
基底関数: 10



基底関数: 50



正則化
パラメータ: 0



正則化
パラメータ: 0.1

※サンプル数は100

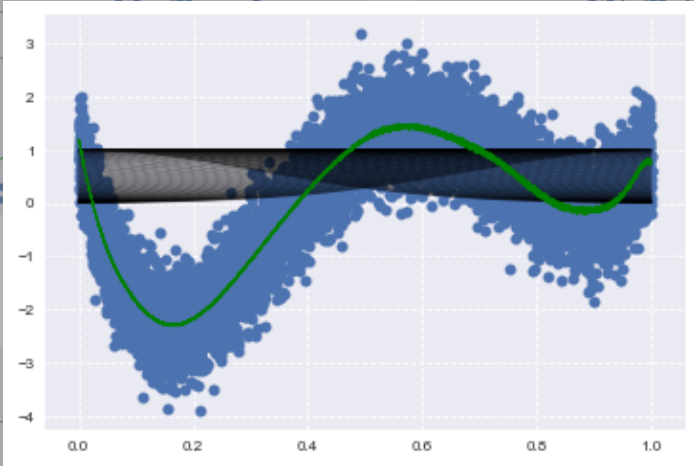
非線形回帰モデル

正則化
パラメータ:0

正則化
パラメータ:0

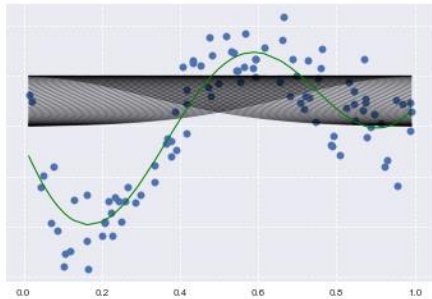
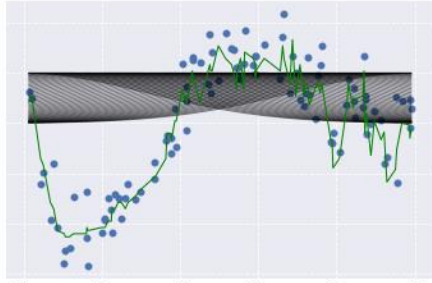
正則化
パラメータ:0.1

基底関数: 50



サンプル数: 10000

基底関数: 50



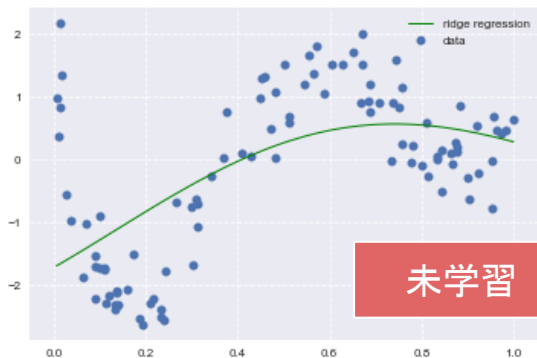
非線形回帰モデル

基底関数の個数

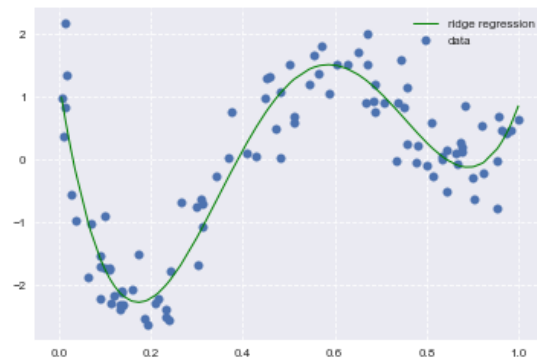
基底関数の位置

基底関数のバンド幅

正則化パラメータ



表現力の低いモデル



適切な表現力のモデル



表現力の高いモデル

適切なモデル (汎化性能が高いモデル)は交差検証法で決定

- 汎化性能

- 学習に使用した入力だけでなく、これまで見たことのない新たな入力に対する予測性能
 - (学習誤差ではなく)汎化誤差 (テスト誤差)が小さいものが良い性能を持ったモデル。
 - 汎化誤差は通常、学習データとは別に収集された検証データでの性能を測ることで推定
- バイアス・バリエンス分解
 - <https://www.hellocybernetics.tech/entry/2017/01/24/100415>

訓練誤差

$$\text{MSE}_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

テスト誤差

$$\text{MSE}_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i^{(\text{test})} - y_i^{(\text{test})})^2$$

データの分割とモデルの汎化性能測定

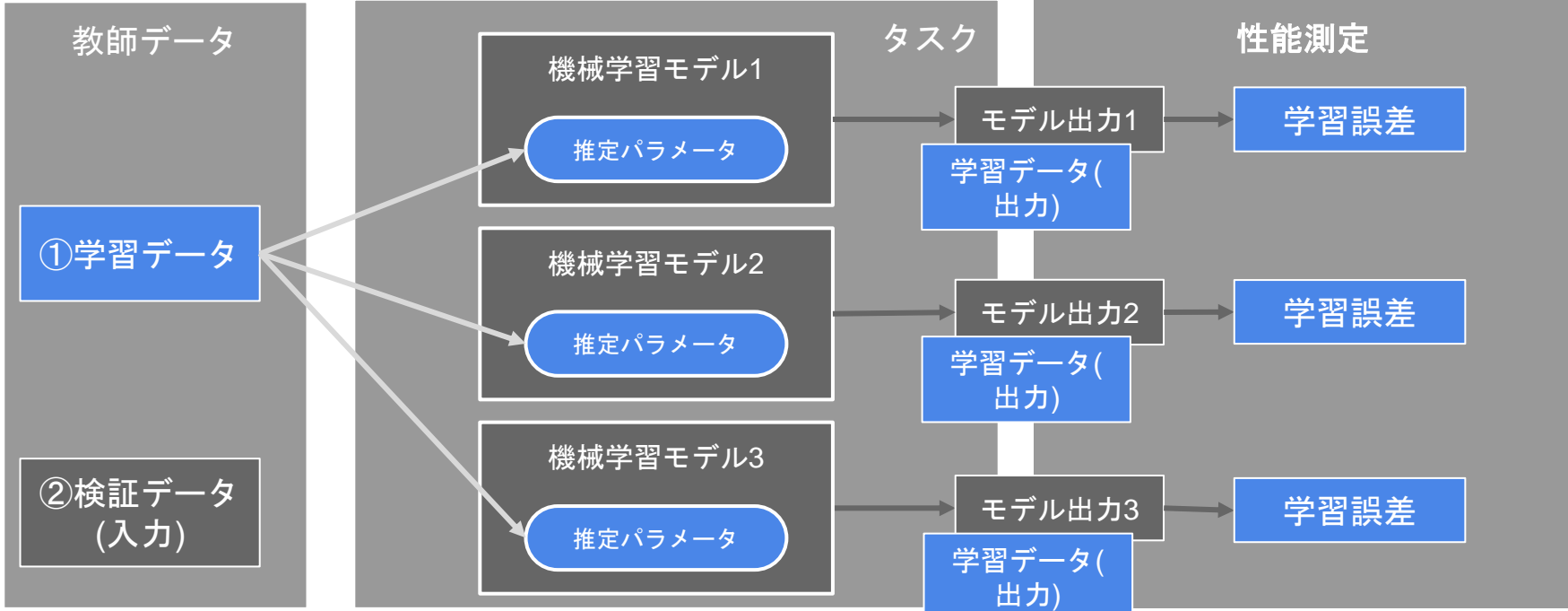
学習に使ったデータと同じデータで検証
(モデルの学習データへの
当てはまりの良さ)

学習誤差

$$MSE_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

検証誤差

$$MSE_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i^{(\text{test})} - y_i^{(\text{test})})^2$$



$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

データの分割とモデルの汎化性能測定

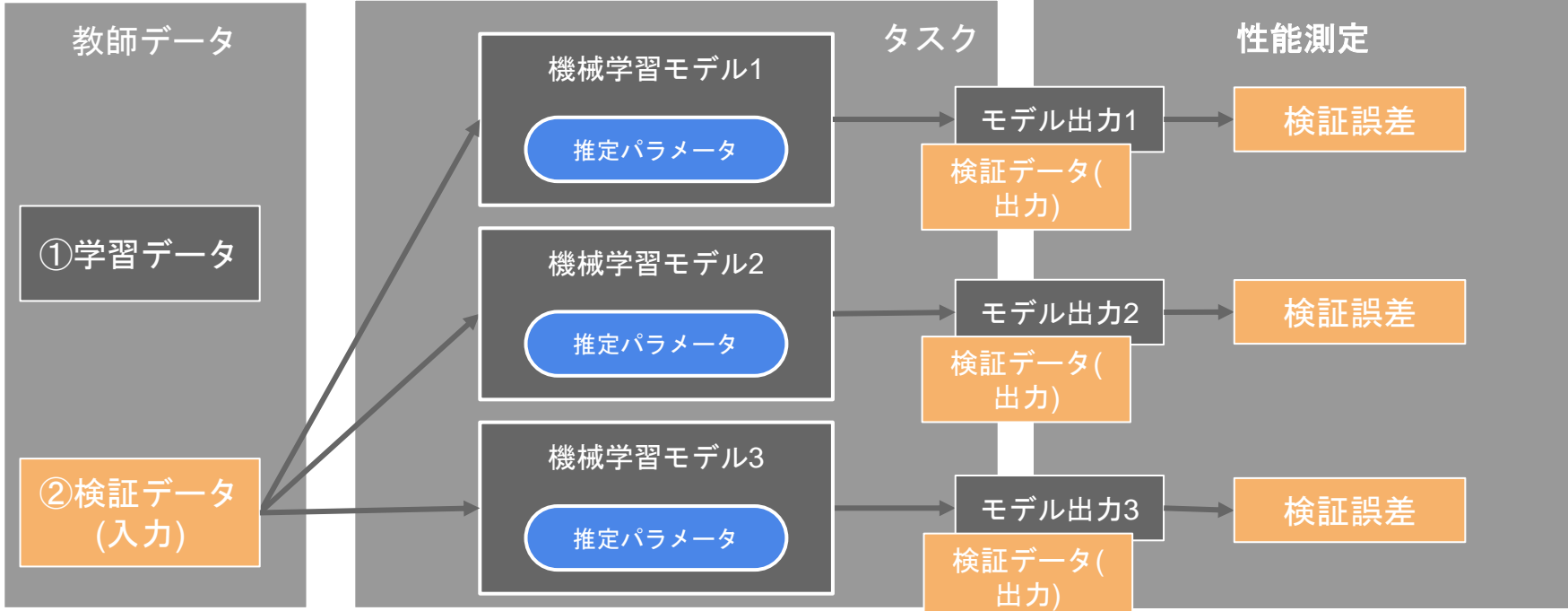
学習に使ったデータと別のデータで検証
(モデルの未来のデータへの
当てはまりの良さ = 汎化性能)

学習誤差

$$MSE_{\text{train}} = \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} (\hat{y}_i^{(\text{train})} - y_i^{(\text{train})})^2$$

検証誤差

$$MSE_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i^{(\text{test})} - y_i^{(\text{test})})^2$$

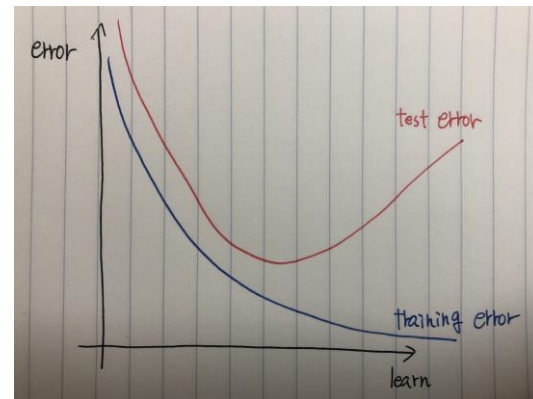
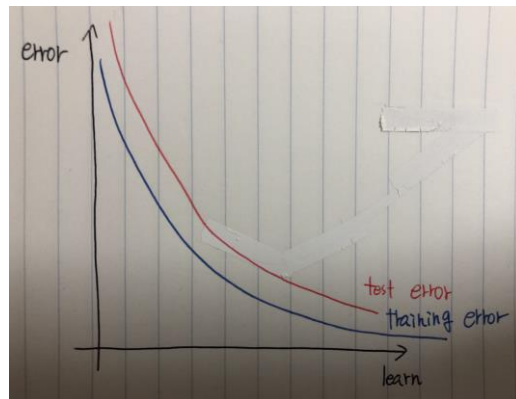
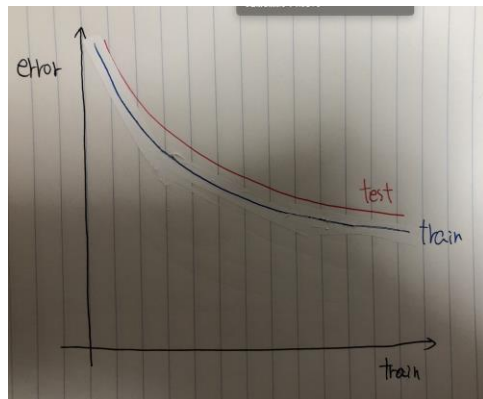


$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

$$\{(\mathbf{x}_i^{(\text{test})}, y_i^{(\text{test})}); i = 1, \dots, n_{\text{test}}\}$$

非線形回帰モデル

- 手元のモデルがデータに未学習しているか過学習しているか
 - 訓練誤差もテスト誤差もどちらも小さい▶汎化しているモデルの可能性
 - 訓練誤差は小さいがテスト誤差が大きい▶過学習
 - 訓練誤差もテスト誤差もどちらも小さくならない▶未学習
 - 回帰の場合には陽に解が求まります (学習誤差と訓練誤差の値を比較)



- ホールドアウト法

- 有限のデータを学習用とテスト用の2つに分割し、「予測精度」や「誤り率」を推定する為に使用
 - 学習用を多くすればテスト用が減り学習精度は良くなるが、性能評価の精度は悪くなる
 - 逆にテスト用を多くすれば学習用が減少するので、学習そのものの精度が悪くなることになる。
 - **手元にデータが大量にある場合を除いて、良い性能評価を与えないという欠点**がある。
- 基底展開法に基づく非線形回帰モデルでは、基底関数の数、位置、バンド幅の値とチューニングパラメータをホールドアウト値を小さくするモデルで決定する



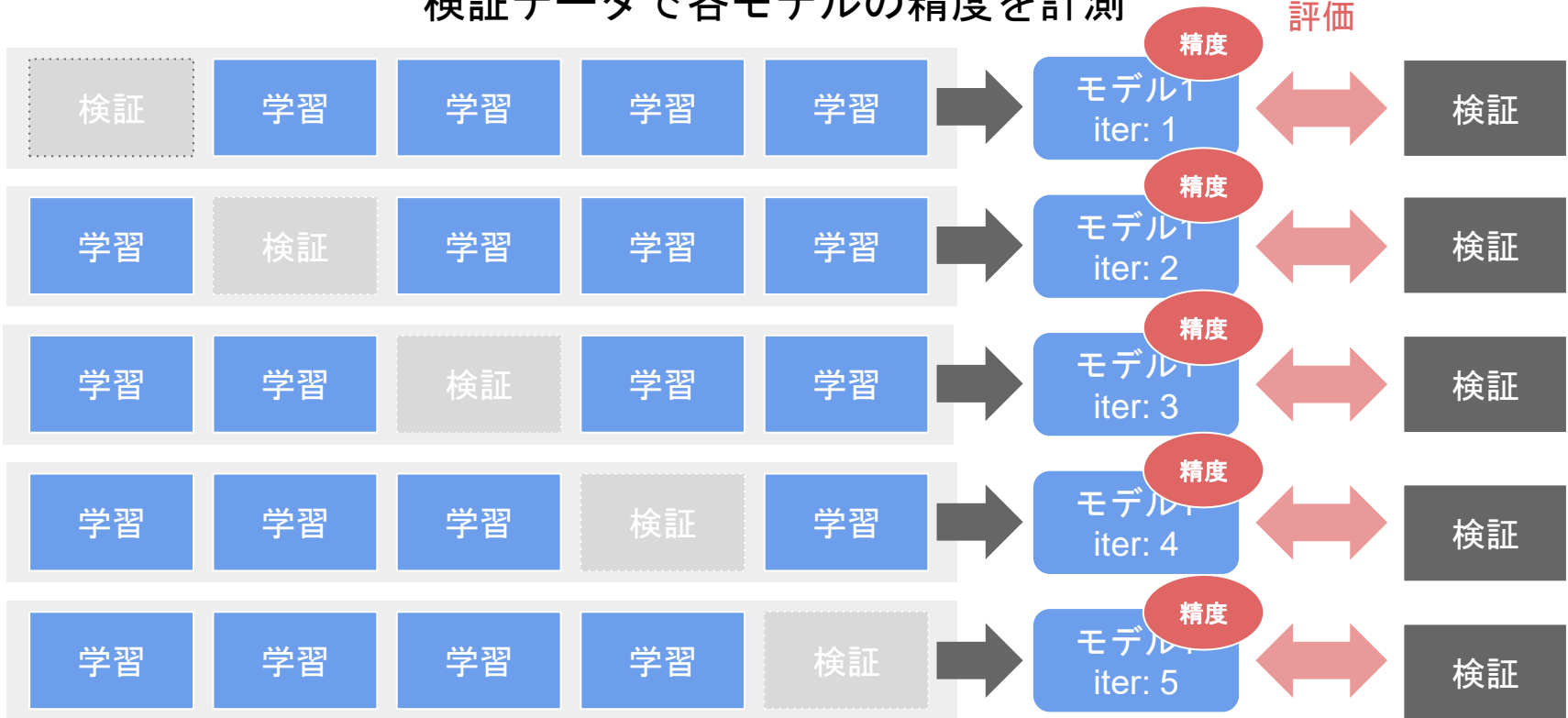
クロスバリデーション(交差検証)

データを学習用と評価用に分割 (5分割の例)

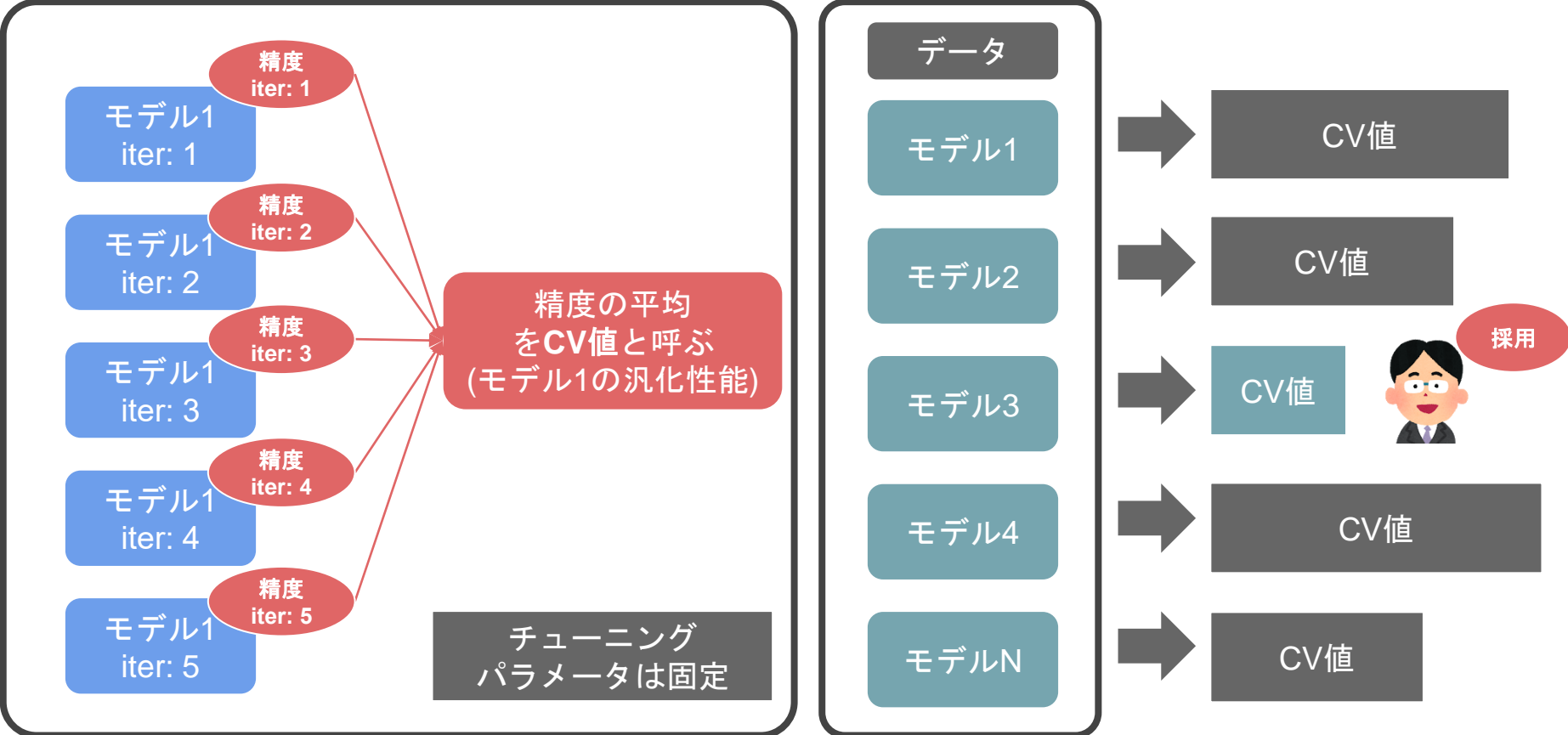


クロスバリデーション(交差検証)

検証データで各モデルの精度を計測



非線形回帰モデル



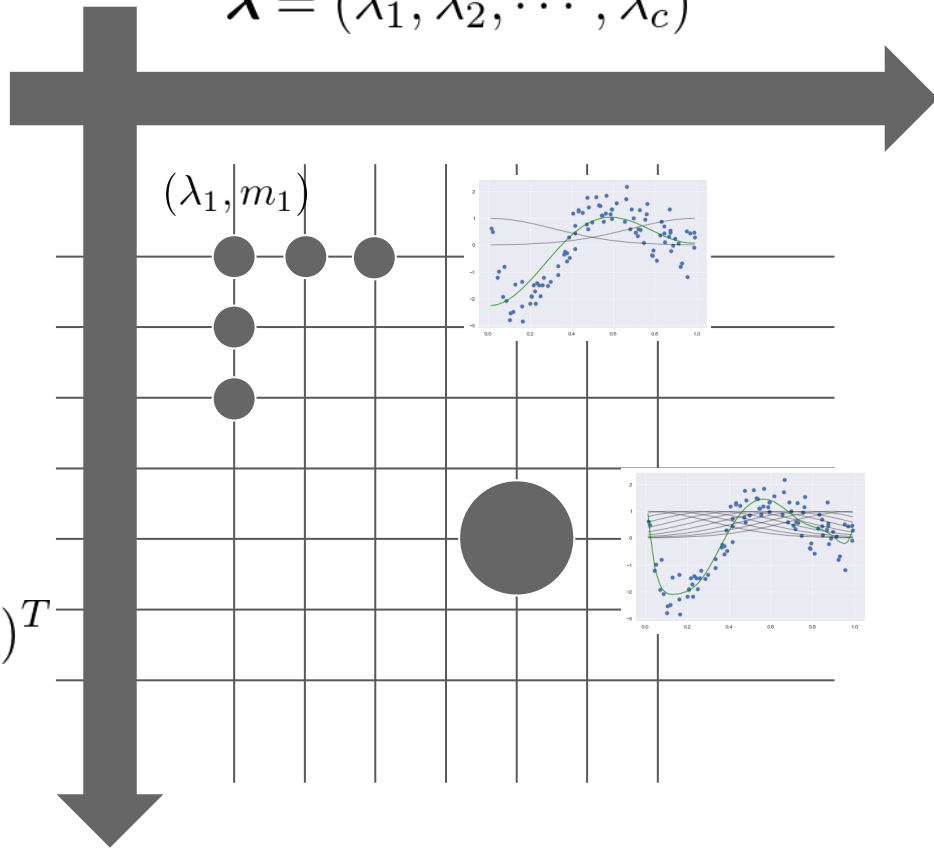
※あるモデルをホールドアウトで検証したところ70%の精度、CVで65%だった場合でも、汎化性能の推定としてはCVを利用するようにしてください。

- グリッドサーチ

- 全てのチューニングパラメータの組み合わせで評価値を算出
- 最も良い評価値を持つチューニングパラメータを持つ組み合わせを、「いいモデルのパラメータ」として採用

$$\mathbf{m} = (m_1, m_2, \dots, m_c)^T$$

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_c)^T$$



分類タスク

ロジスティック 回帰モデル

1. 講義

- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. ロジスティック回帰モデルの説明
 - iii. パラメータの推定
 - iv. モデルの評価

1. ハンズオン

- a. タイタニックデータ解析
- b. 特徴量抽出

ロジスティック回帰モデル

- 分類問題 (クラス分類)

- ある入力(数値)からクラスに分類する問題

- 分類で扱うデータ

- 入力 (各要素を説明変数または特徴量と呼ぶ)
 - m次元のベクトル (m=1の場合はスカラー)
- 出力 (目的変数)
 - 0 or 1の値
- タイタニックデータ、IRISデータなど

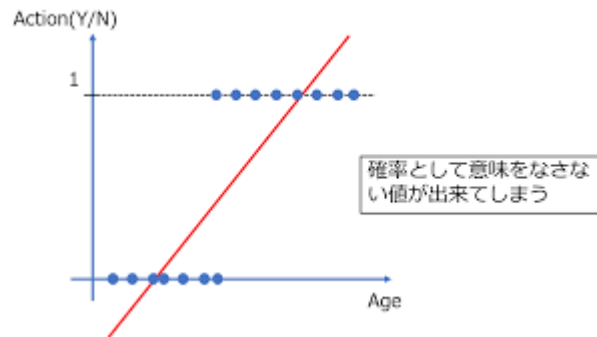
説明変数

$$\mathbf{x} = (x_1, x_2, \dots, x_m)^T \in \mathbb{R}^m$$

目的変数

$$y \in \{0, 1\}$$

0か1



教師データ

$$\{(\mathbf{x}_i, y_i); i = 1, \dots, n\}$$



ロジスティック回帰モデル

● ロジスティック線形回帰モデル

- 分類問題を解くための教師あり機械学習モデル(教師データから学習)
 - 入力と m 次元パラメータの線形結合をシグモイド関数に入力
 - 出力は $y=1$ になる確率の値になる

パラメータ

$$\mathbf{w} = (w_1, w_2, \dots, w_m)^T \in \mathbb{R}^m$$

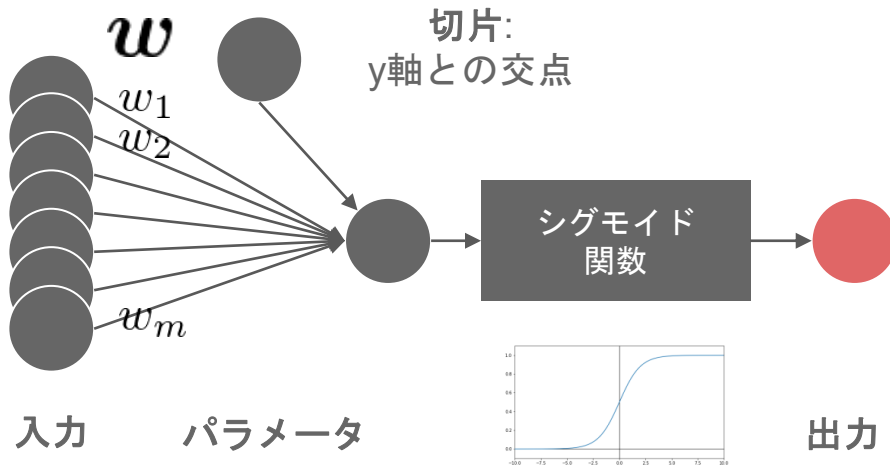
線形結合

$$\hat{y} = \mathbf{w}^T \mathbf{x} + w_0 = \sum_{j=1}^m w_j x_j + w_0$$

例) titanicデータ

- ・ 部屋のクラス
- ・ 年齢
- ・ 性別など

\mathbf{x}



生死データ
(0なら生還、1なら死亡)

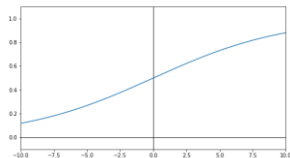
- シグモイド関数

- 入力の実数・出力は必ず0~1の値
- (クラス1に分類される)確率を表現
- 単調増加関数

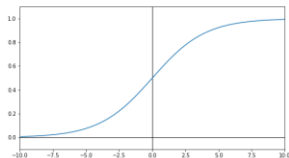
- パラメータが変わるとシグモイド関数の形が変わる

- aを増加させると, $x=0$ 付近での曲線の勾配が増加
- aを極めて大きくすると, 単位ステップ関数($x<0$ で $f(x)=0$, $x>0$ で $f(x)=1$ となるような関数)に近づきます
- バイアス変化は段差の位置

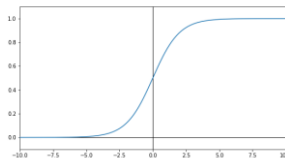
$$\sigma(x) = \frac{1}{1 + \exp(-ax)}$$



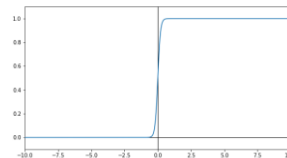
a=0.2



a=0.5



a=1



a= 10

- シグモイド関数の性質

- シグモイド関数の微分は、シグモイド関数自身で表現することが可能
- 尤度関数の微分を行う際にこの事実を利用すると計算が容易

$$\begin{aligned}\frac{\partial \sigma(x)}{\partial x} &= \frac{\partial}{\partial x} \left(\frac{1}{1 + \exp(-ax)} \right) \\ &= (-1) \cdot \{1 + \exp(-ax)\}^{-2} \cdot \exp(-ax) \cdot (-a) \\ &= \frac{a \exp(-ax)}{\{1 + \exp(-ax)\}^2} = \frac{a}{1 + \exp(-ax)} \cdot \frac{1 + \exp(-ax) - 1}{1 + \exp(-ax)} \\ &= a\sigma(x)(1 - \sigma(x))\end{aligned}$$

連鎖律

ロジスティック回帰モデル

- シグモイド関数の出力をY=1になる確率に対応させる
 - データの線形結合を計算
 - シグモイド関数に入力▶出力が確率に対応
 - [表記] i番目データを与えた時のシグモイド関数の出力をi番目のデータがY=1になる確率とする

求めたい値

シグモイド
関数

$$P(Y = 1|\mathbf{x}) = \sigma(w_0 + w_1x_1 + \cdots + w_mx_m)$$

説明変数の実現値が
与えられた際にY=1になる確率

データのパラメーターに対する線形結合

表記

$$p_i = \sigma(w_0 + w_1x_{i1} + \cdots + w_mx_{im})$$

ロジスティック回帰モデル

数式

$$P(Y = 1|\mathbf{x}) = \sigma(w_0 + w_1x_1)$$

データが与えられた時
に $Y=1$ になる確率

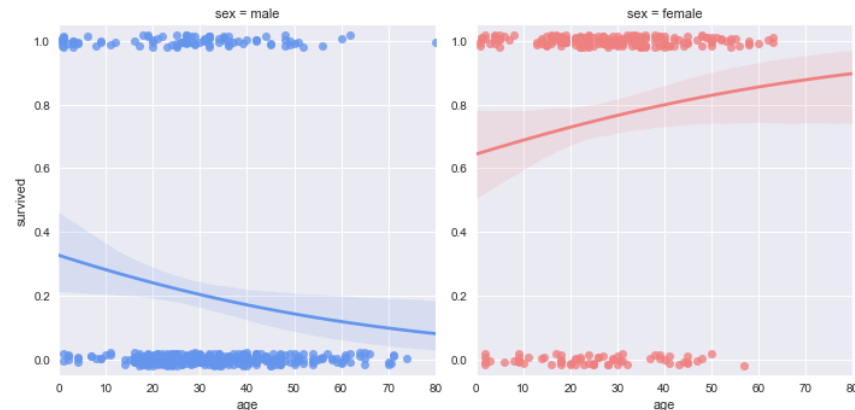
切片 回帰係数

説明変数

既知：入力データ

未知：学習で決める

幾何学的な意味



データ Y は確率が0.5以上ならば1・未満なら0と予測

最尤推定

ロジスティック回帰モデル

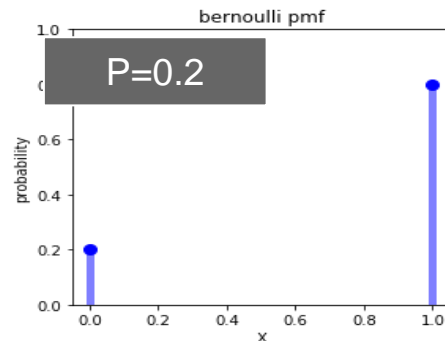
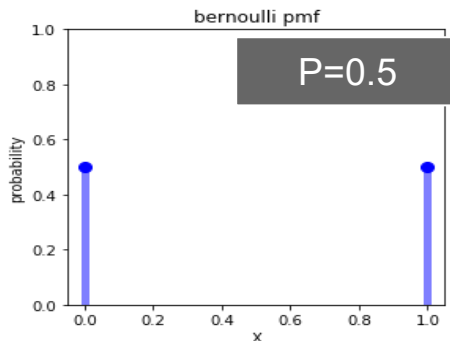
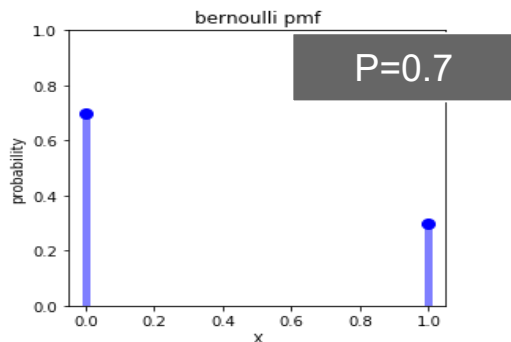
- 世の中には様々な確率分布が存在する
 - 正規分布、t分布、ガンマ分布、一様分布、ディリクレ分布・・・
 - ロジスティック回帰モデルではベルヌーイ分布を利用する
- ベルヌーイ分布
 - 数学において、確率 p で1、確率 $1-p$ で0をとる、離散確率分布 (例：コインを投げ)
 - 「生成されるデータ」は分布のパラメータによって異なる (この場合は確率 p)

ベルヌーイ分布に
従う確率変数 Y

$$Y \sim Be(p)$$

$$P(y) = p^y(1-p)^{1-y}$$

$Y=0$ と $Y=1$ になる
確率をまとめて表現



- ベルヌーイ分布のパラメータの推定

- ある分布を考えた時、そのパラメータ(既知)によって、生成されるデータは変化する
 - 0.3と0.8
- データからそのデータを生成したであろう尤もらしい分布(パラメータ)を推定したい
 - 最尤推定



ロジスティック回帰モデル

- 同時確率

- あるデータが得られた時、それが同時に得られる確率
- 確率変数は独立であることを仮定すると、それぞれの確率の掛け算となる

- 尤度関数とは

- データは固定し、パラメータを変化させる
- 尤度関数を最大化するようなパラメータを選ぶ推定方法を最尤推定という

1回の試行で
 $y=y_1$ になる確率

$$P(y) = p^y(1-p)^{1-y}$$

 : 既知  : 未知

n回の試行で
 $y_1 \sim y_n$ が同時に
起こる確率(p固定)

$$P(y_1, y_2, \dots, y_n; p) = \prod_{i=1}^n p^{y_i} (1-p)^{1-y_i}$$

$y_1 \sim y_n$ のデータが
得られた際の
尤度関数

$$P(y_1, y_2, \dots, y_n; p) = \prod_{i=1}^n p^{y_i} (1-p)^{1-y_i}$$

- ロジスティック回帰モデルの最尤推定

- 確率 p はシグモイド関数となるため、推定するパラメータは重みパラメータとなる
- $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$ を生成するに至った尤もらしいパラメータを探す

$$P(Y = y_1 | \mathbf{x}_1) = p_1^{y_1} (1 - p_1)^{1-y_1} = \sigma(\mathbf{w}^T \mathbf{x}_1)^{y_1} (1 - \sigma(\mathbf{w}^T \mathbf{x}_1))^{1-y_1}$$

$$P(Y = y_2 | \mathbf{x}_2) = p_2^{y_2} (1 - p_2)^{1-y_2} = \sigma(\mathbf{w}^T \mathbf{x}_2)^{y_2} (1 - \sigma(\mathbf{w}^T \mathbf{x}_2))^{1-y_2}$$

...

$$P(Y = y_n | \mathbf{x}_n) = p_n^{y_n} (1 - p_n)^{1-y_n} = \sigma(\mathbf{w}^T \mathbf{x}_n)^{y_n} (1 - \sigma(\mathbf{w}^T \mathbf{x}_n))^{1-y_n}$$



: 既知



: 未知

ロジスティック回帰モデル

$y_1 \sim y_n$ のデータが
得られた際の尤度関数

確率変数が独立を仮定
▶ 確率の積に分解可能

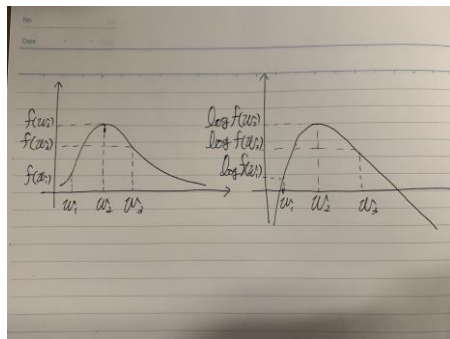
$$\begin{aligned} P(y_1, y_2, \dots, y_n | w_0, w_1, \dots, w_m) &= \prod_{i=1}^n p_i^{y_i} (1 - p_i)^{1-y_i} \\ &= \prod_{i=1}^n \sigma(\mathbf{w}^T \mathbf{x}_i)^{y_i} (1 - \sigma(\mathbf{w}^T \mathbf{x}_i))^{1-y_i} \\ &= L(\mathbf{w}) \end{aligned}$$

尤度関数は
パラメータのみに依存する関数

尤度関数Eを最大とするパラメータを探索

- 尤度関数を最大とするパラメータを探す (推定)

- 対数をとると微分の計算が簡単
 - 同時確率の積が和に変換可能
 - 指数が積の演算に変換可能
- 対数尤度関数が最大になる点と尤度関数が最大になる点は同じ
 - 対数関数は単調増加 (ある尤度の値が $x_1 < x_2$ の時、必ず $\log(x_1) < \log(x_2)$ となる)
- 「尤度関数にマイナスをかけたものを最小化」し、「最小2乗法の最小化」と合わせる



$$\begin{aligned} E(w_0, w_1, \dots, w_m) &= -\log L(w_0, w_1, \dots, w_m) \\ &= -\sum_{i=1}^n \{y_i \log p_i + (1 - y_i) \log(1 - p_i)\} \end{aligned}$$

- 勾配降下法 (Gradient descent)

- 反復学習によりパラメータを逐次的に更新するアプローチの一つ
- η は学習率と呼ばれるハイパーパラメータでモデルのパラメータの収束しやすさを調整
- 参考URL: <https://qiita.com/masatomix/items/d4e5fb3b52fa4c92366f>

- なぜ必要か？

- [線形回帰モデル (最小2乗法)] ▶ MSEのパラメータに関する微分が0になる値を解析に求めることが可能
- [ロジスティック回帰モデル (最尤法)] ▶ 対数尤度関数をパラメータで微分して0になる値を求める必要があるのだが、解析的にこの値を求めることは困難である。

$$\boldsymbol{w}(k+1) = \boldsymbol{w}^k - \eta \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}}$$



- 対数尤度関数を係数とバイアスに関して微分

$$\begin{aligned}\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} &= \sum_{i=1}^n \frac{\partial E_i(\mathbf{w})}{\partial p_i} \frac{\partial p_i}{\partial \mathbf{w}} \\&= \sum_{i=1}^n \left(\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right) \frac{\partial p_i}{\partial \mathbf{w}} \\&= \sum_{i=1}^n \left(\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right) p_i(1-p_i) \mathbf{x}_i \\&= - \sum_{i=1}^n (y_i(1-p_i) - p_i(1-y_i)) \mathbf{x}_i \\&= - \sum_{i=1}^n (y_i - p_i) \mathbf{x}_i\end{aligned}$$

連鎖律

対数尤度関数の
pに関する微分

シグモイド関数の微分

式を整理

ロジスティック回帰モデル

- パラメータが更新されなくなった場合、それは勾配が0になったということ。少なくとも反復学習で探索した範囲では最適な解がもとめられたことになる。

$$\boldsymbol{w}^{(k+1)} = \boldsymbol{w}^{(k)} + \eta \sum_{i=1}^n (y_i - p_i) \boldsymbol{x}_i$$

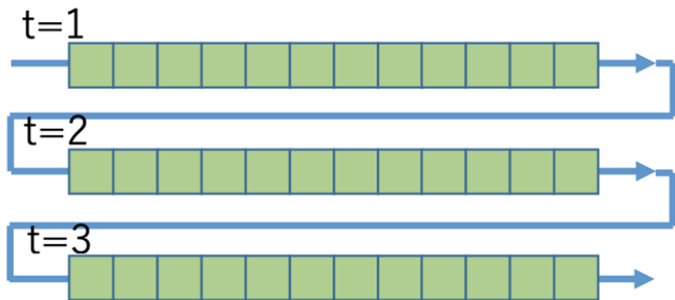
- 勾配降下法では、パラメータを更新するのにN個全てのデータに対する和を求める必要がある。
 - nが巨大になった時にデータをオンメモリに載せる容量が足りない、計算時間が莫大になるなどの問題がある
 - 確率的勾配降下法を利用して解決

- 確率的勾配降下法 (SGD)

- データを一つずつランダムに(「確率的」に)選んでパラメータを更新
- 勾配降下法でパラメータを1回更新するのと同じ計算量でパラメータをn回更新できるので効率よく最適な解を探索可能

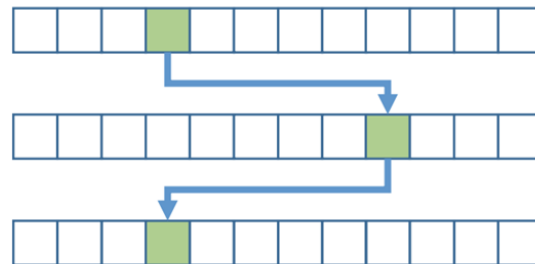
$$w(k+1) = w^k + \eta(y_i - p_i)x_i$$

普通の勾配降下法：



確率的勾配降下法：

毎回の更新でデータの一つ(または少量)しか見ない



ロジスティック回帰モデル

- 学習済みの「ロジスティック回帰モデル」の性能を測る指標について見ていく。
- その前に混同行列 (confusion matrix)について説明
 - 各検証データに対するモデルの予測結果を4つの観点(表)で分類し、それぞれに当てはまる予測結果の個数をまとめた表

		検証用データの結果	
		positive	negative
モデルの 予測結果	positive	真陽性 (True Positive) ~正しく positive と判別した個数	偽陰性 (False Positive) ~間違えて positive と判別した個数
	negative	偽陽性 (False Negative) ~間違えて Negative と判別した個数	真陰性 (True Negative) ~正しく Negative と判別した個数

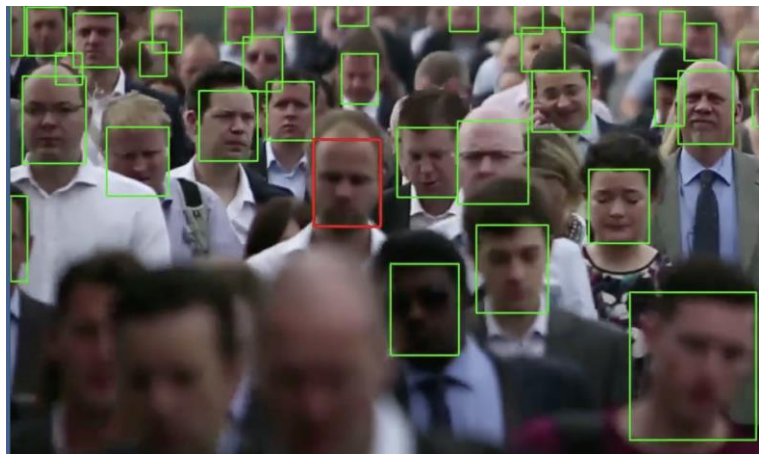
- 分類の評価方法

- 「正解率」がよく使われる
- 正解した数 / 予測対象となった全データ数
 - メールのスパム分類
 - スпам数が80件・普通のメールが20件であった場合
 - 全てをスパムとする分類器の正解率は80%となる。
- どのような問題があるか？
 - 分類したいクラスにはそれぞれ偏りがあることが多い
 - この場合、単純な正解率はあまり意味をなさないことがほとんどです。

$$\frac{TP + TN}{TP + FN + FP + TN}$$

● 分類の評価方法

- 表情から怪しい人物を検知する動画分析ソリューション
 - 前スライドの正解率が適当ではない例 ▶ リコールやプレシジョンを使って評価する
 - 異常値検知のタスクぽいが、説明の簡易化のため分類で解くとする



- False Positive
 - 正常な人を間違えて異常としてしまう。
 - 赤い枠が増えるので、確認の工数がかかかかる。
- False Negative
 - 異常な人を間違えて正常としてしまう。
 - 本当に検知したい脅威が見えなくなる。

ロジスティック回帰モデル

● 再現率 (Recall)

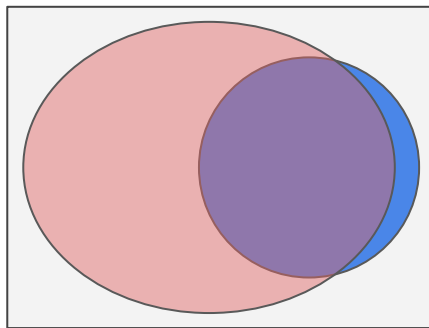
- 「本当にPositiveなもの」の中からPositiveと予測できる割合 (NegativeなものをPositiveとしてしまう事象については考えていない)
- 「誤り(False Positive)が多少多くても抜け漏れは少ない」 予測をしたい際に利用
- 使用例) 病気の検診で「陽性であるものを陰性と誤診 (False Negative)」としてしまうのを避けたい。「陰性を陽性であると誤診 (False Positive)」とするものが少し増えたとしても再検査すればいい。

$$\frac{TP}{TP + FN}$$

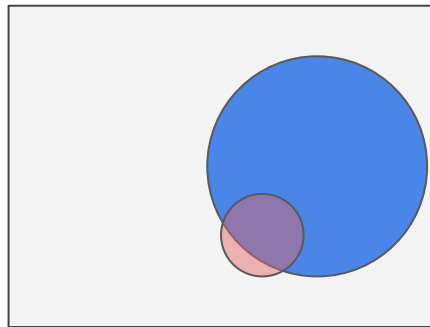
● モデルがPositiveと予測したデータ ● 本当にPositiveであるデータ

		ラベル (答え)	
		positive	negative
モデルの 予測結果	positive	True Positive	False Positive
	negative	False Negative	True Negative

False Negative
が小さい



Recall 高



Recall 低

● 適合率 (Precision)

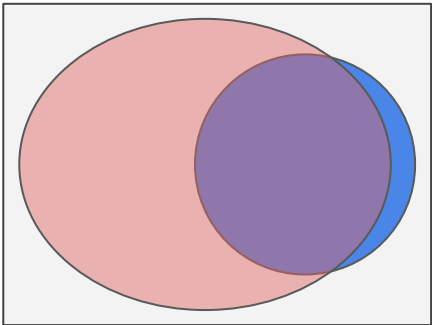
- モデルが「Positiveと予測」したものの中で本当にPositiveである割合 (本当にPositiveなものをNegativeとしてしまう子については考えていない)
- 見逃し(False Negative)が多くてもより正確な「予測をしたい際に利用
- 「重要なメールをスパムメールと誤判別」されるより「スパムと予測したものが確実にスパム」である方が便利。スパムメールを検出できなくても(False Negative)、自分でやればいい。

$$\frac{TP}{TP + FP}$$

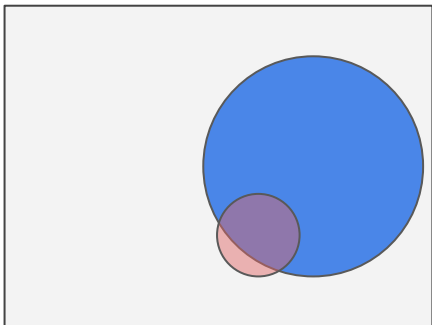
● モデルがPositiveと予測したデータ ● 本当にPositiveであるデータ

		ラベル (答え)	
		positive	negative
モデルの 予測結果	positive	True Positive	False Positive
	negative	False Negative	True Negative

False Positive
が小さい



Precision 低



Precision 高

- F値

- 理想的にはどちらも高いモデルがいいモデルだが、両者はトレードオフの関係にあり、どちらかを小さくすると、もう片方の値が大きくなってしまう。
- PrecisionとRecallの調和平均
 - RecallとPrecisionのバランスを示している。高ければ高いほどRecallとPrecisionがともに高くなる。
 - <http://ibisforest.org/index.php?F%E5%80%A4>

ロジスティック回帰モデル

- 設定
 - タイタニックの乗客データを利用しロジスティック回帰モデルを作成
 - 特徴量抽出を試みる
- 課題
 - 年齢が30歳で男の乗客は生き残れるか？

データセット名	タイタニックデータ
レコード数	891 / 471
カラム数	11
詳細	kaggle
備考	分類アルゴリズムでよく利用される

ロジスティック回帰モデル



項番	変数	記号名	値
1	PassengerId	乗客ID	1, 2 ...
2	Survived	生存・死者情報	1 -> 生存者(Alive) / 0 ->死者(Dead)
3	Pclass	乗客の社会階級	1 (High) / 2 (Middle) / 3 (Low))
4	Name	乗客名	-
5	Sex	性別	男性： male / 女性： female
6	Age	年齢	22.0 / 38.0 など
7	SibSp	兄弟および配偶者の数	0 / 1/ 2 など
8	Parch	親もしくは子供の数	0 / 1 など
9	Ticket	チケットNo	A/5 21171 など
10	Fare	運賃	7.2500 など
11	Cabin	船室	C85など
12	Embarked	乗船した港（3つ）	C: Cherbourg; Q: Queenstown, S: Southampton

欠損あり

欠損あり

テストデータ
には無し

項番	変数	記号名	値
1	PassengerId	乗客ID	1, 2 ...
2	Survived	生存・死者情報	1 -> 生存者(Alive) / 0 ->死者(Dead)
3	Pclass	乗客の社会階級	1 (High) / 2 (Middle) / 3 (Low))
4	Name	乗客名	-
5	Sex	性別	男性： male / 女性： female
6	Age	年齢	22.0 / 38.0 など
7	SibSp	兄弟および配偶者の数	0 / 1/ 2 など
8	Parch	親もしくは子供の数	0 / 1 など
9	Ticket	チケットNo	A/5 21171 など
10	Fare	運賃	7.2500 など
11	Cabin	船室	C85など
12	Embarked	乗船した港（3つ）	C: Cherbourg; Q: Queenstown, S: Southampton

平均

欠損あり

欠損あり

テストデータ
には無し

項番	変数	記号名	値
1	PassengerId	乗客ID	1, 2 ...
2	Survived	生存・死者情報	1 -> 生存者(Alive) / 0 ->死者(Dead)
3	Pclass	乗客の社会階級	1 (High) / 2 (Middle) / 3 (Low))
4	Name	乗客名	-
5	Sex	性別	男性 : male / 女性 : female
6	Age	年齢	22.0 / 38.0 など
7	SibSp	兄弟および配偶者の数	0 / 1/ 2 など
8	Parch	親もしくは子供の数	0 / 1 など
9	Ticket	チケットNo	A/5 21171 など
10	Fare	運賃	7.2500 など
11	Cabin	船室	C85など
12	Embarked	乗船した港 (3つ)	C: Cherbourg; Q: Queenstown, S: Southampton

ラベル

入力データ

まずは2変数に絞る

主成分分析

1. 講義

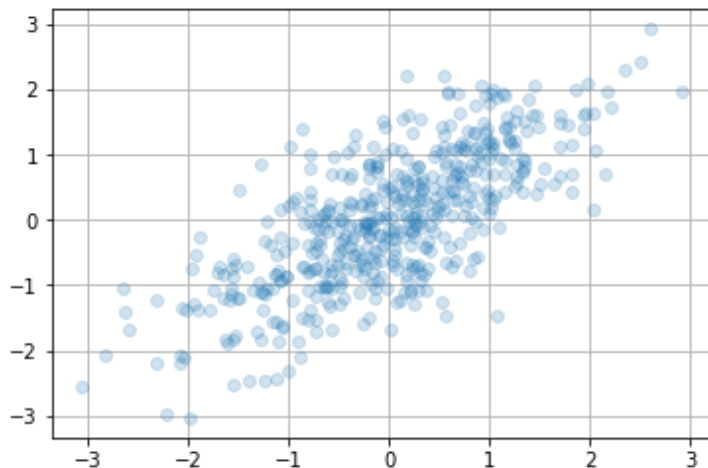
- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. 主成分分析の説明
 - iii. 主成分と寄与率

1. ハンズオン

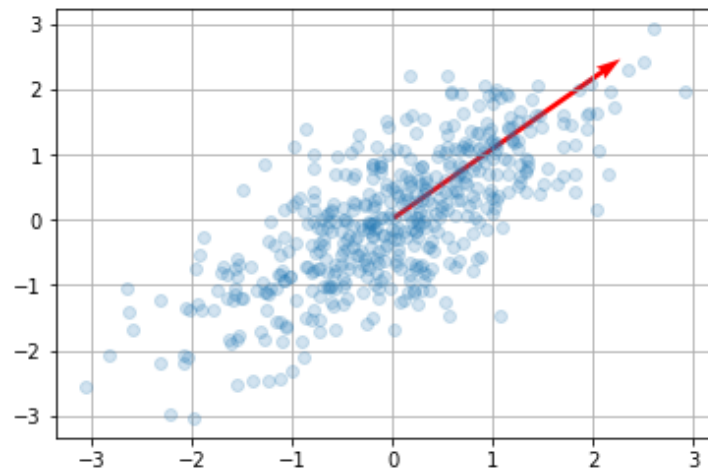
- a. 乳がんデータの分析

主成分分析

- 多変量データの持つ構造をより少数個の指標に圧縮
 - 変数の個数を減らすことに伴う、情報の損失はなるべく小さくしたい
 - 少数変数を利用した分析や可視化(2・3次元の場合)が実現可能



相関0.8の正規分布
から生成したデータ



2次元を1次元に圧縮

主成分分析

学習データ

$$\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{im}) \in \mathbb{R}^m$$

平均 (ベクトル)

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$$

データ行列

$$\bar{X} = (\mathbf{x}_1 - \bar{\mathbf{x}}, \dots, \mathbf{x}_n - \bar{\mathbf{x}})^T \in \mathbb{R}^{n \times m}$$

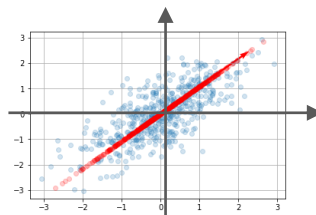
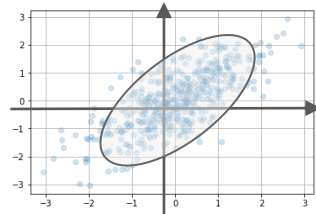
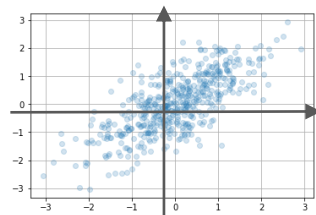
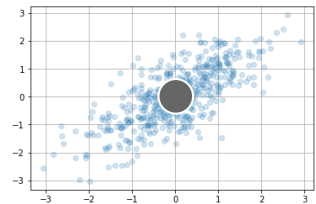
分散共分散行列
(復習用)

$$\Sigma = \text{Var}(\bar{X}) = \frac{1}{n} \bar{X}^T \bar{X}$$

線形変換後の
ベクトル

$$\mathbf{s}_j = (s_{1j}, \dots, s_{nj})^T = \bar{X} \mathbf{a}_j \quad \mathbf{a}_j \in \mathbb{R}^m$$

※jは射影軸のインデックス

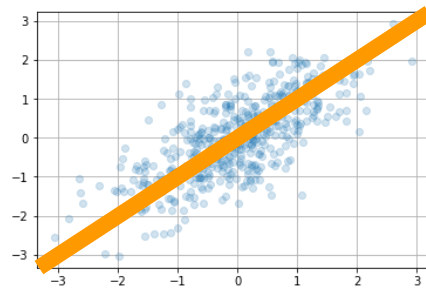
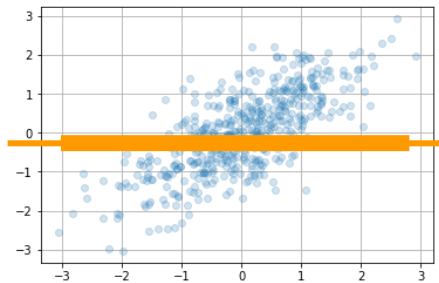
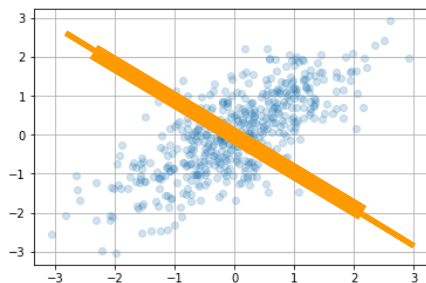


主成分分析

- 係数ベクトルが変われば線形変換後の値が変化

- 情報の量を分散の大きさと捉える
- 線形変換後の変数の分散が最大となる射影軸を探索

$$\mathbf{s}_j = (s_{1j}, \dots, s_{nj})^T = \bar{X} \mathbf{a}_j \quad \mathbf{a}_j \in \mathbb{R}^m$$



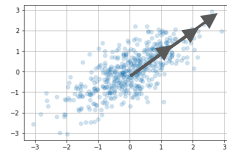
線形変換後の分散

—
$$Var(\mathbf{s}_j) = \frac{1}{n} \mathbf{s}_j^T \mathbf{s}_j = \frac{1}{n} (\bar{X} \mathbf{a}_j)^T (\bar{X} \mathbf{a}_j) = \frac{1}{n} \mathbf{a}_j^T \bar{X} \bar{X} \mathbf{a}_j = \mathbf{a}_j^T Var(\bar{X}) \mathbf{a}_j$$

主成分分析

- 以下の制約付き最適化問題を解く

- ノルムが1となる制約を入れる(制約を入れないと無限に解がある)



目的関数

$$\arg \max_{\mathbf{a} \in \mathbb{R}^m} \mathbf{a}_j^T \text{Var}(\bar{X}) \mathbf{a}_j$$

制約条件

$$\mathbf{a}_j^T \mathbf{a}_j = 1$$

- 制約つき最適化問題の解き方

- ラグランジュ関数を最大にする係数ベクトルを探索 (微分して0になる点)

ラグランジュ乗数

ラグランジュ関数

$$E(\mathbf{a}_j) = \mathbf{a}_j^T \text{Var}(\bar{X}) \mathbf{a}_j - \lambda(\mathbf{a}_j^T \mathbf{a}_j - 1)$$

目的関数

制約条件

主成分分析

- ラグランジュ関数を微分して最適解を求める

- 元のデータの分散共分散行列の固有値と固有ベクトルが、上記の制約付き最適化問題の解となる

微分 $\frac{\partial E(\mathbf{a}_j)}{\partial \mathbf{a}_j} = 2\text{Var}(\bar{X})\mathbf{a}_j - 2\lambda\mathbf{a}_j = 0$

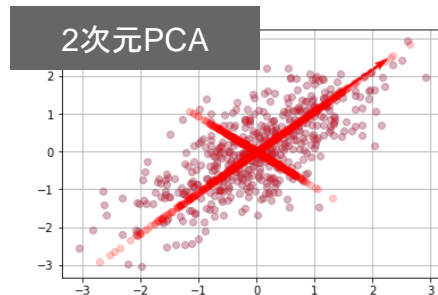
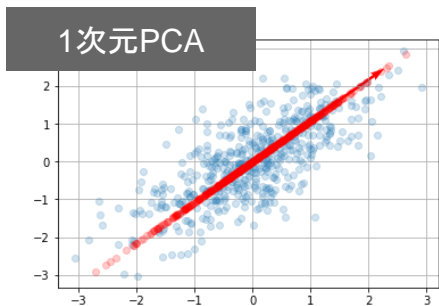


解 $\text{Var}(\bar{X})\mathbf{a}_j = \lambda\mathbf{a}_j$

- 射影先の分散は固有値と一致

$$\text{Var}(\mathbf{s}_1) = \mathbf{a}_1^T \text{Var}(\bar{X})\mathbf{a}_1 = \lambda_1 \mathbf{a}_1^T \mathbf{a}_1 = \lambda_1$$

- 分散共分散行列は正定値対称行列 ▶ 固有値は必ず0以上・固有ベクトルは直行



主成分分析

分散共分散行列
を計算

固有値問題を解く

(最大)m個の固有値と
固有ベクトルのペアが出現

k番目の固有値(昇順)
並べ、対応する固有ベクトル
を第k主成分と呼ぶ

● 寄与率

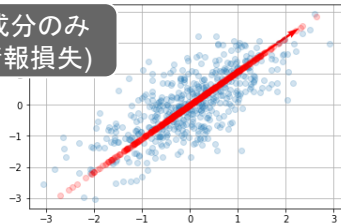
- 第1~元次元分の主成分の分散は、元のデータの分散と一致
 - 2次元のデータを2次元の主成分で表示した時、固有値の和と元のデータの分散が一致
 - 第k主成分の分散は主成分に対応する固有値

$$V_{total} = \sum_{i=1}^m \lambda_i$$

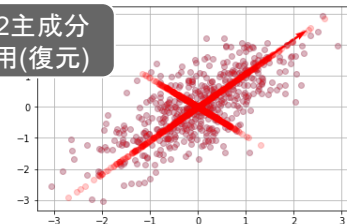
元データの
総分散

主成分の
総分散

第1主成分のみ
利用(情報損失)



第1~2主成分
を利用(復元)



- 寄与率：第k主成分の分散の全分散に対する割合 (第k主成分が持つ情報量の割合)
- 累積寄与率：第1-k主成分まで圧縮した際の情報損失量の割合

$$c_k = \frac{\lambda_k}{\sum_{i=1}^m \lambda_i}$$

第k主成分の
分散

主成分の
総分散

$$r_k = \frac{\sum_{j=1}^k \lambda_j}{\sum_{i=1}^m \lambda_i}$$

第1~k主成
分の分散

主成分の
総分散

主成分分析

- 設定
 - 乳がん検査データを利用しロジスティック回帰モデルを作成
 - 主成分を利用し2次元空間上に次元圧縮
- 課題
 - 32次元のデータを2次元上に次元圧縮した際に、うまく判別できるかを確認

データセット名	乳がん検査データ
レコード数	569
カラム数	33
詳細	scikit learn
備考	

k近傍法

1. 講義

- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. 最近傍法・k近傍法説明
 - iii. アルゴリズム

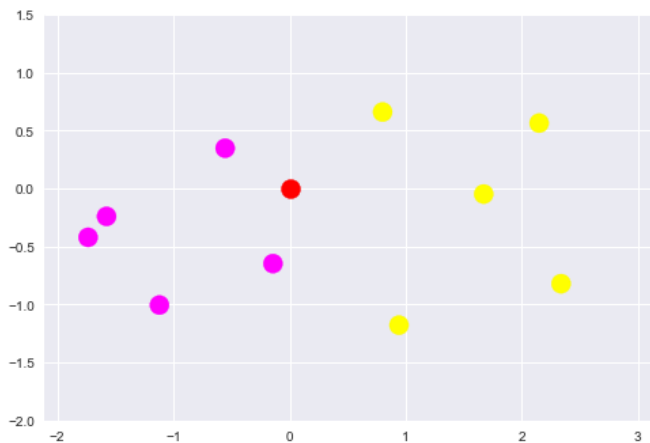
1. ハンズオン

- a. syntheticデータ分析

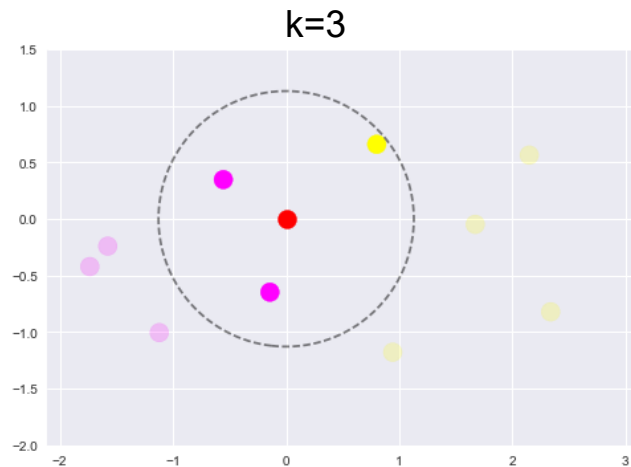
k近傍法 (kNN)

- 分類問題のための機械学習手法

最近傍のデータを k 個取ってきて、それらがもっとも多く所属するクラスに識別



新しいデータ(赤色)を分類する

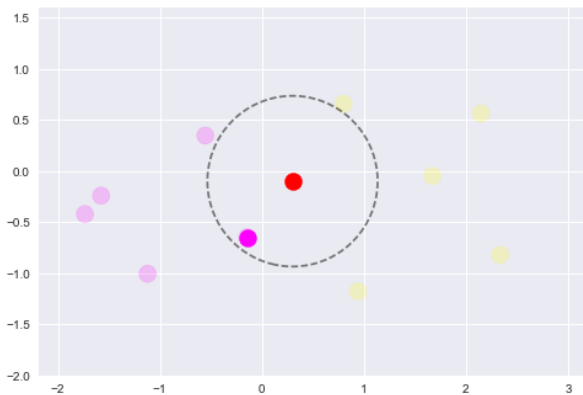


近傍の点は紫2個、黄1個なので
紫クラスに分類する

k近傍法 (kNN)

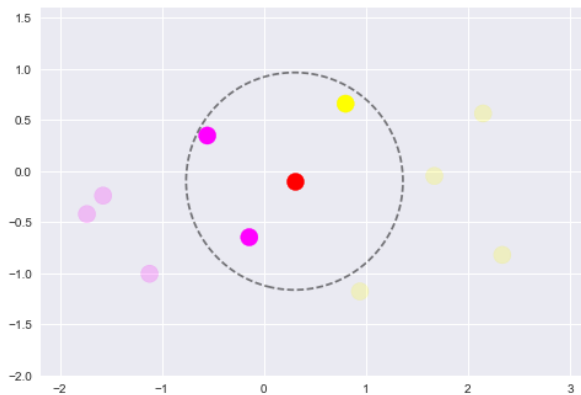
- kを変化させると結果も変わる

k=1 (最近傍法)



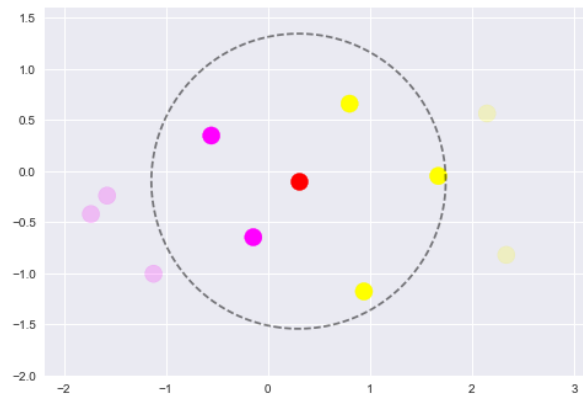
紫1個、黄0個で
紫クラスに分類

k=3



紫2個、黄1個で
紫クラスに分類

k=5

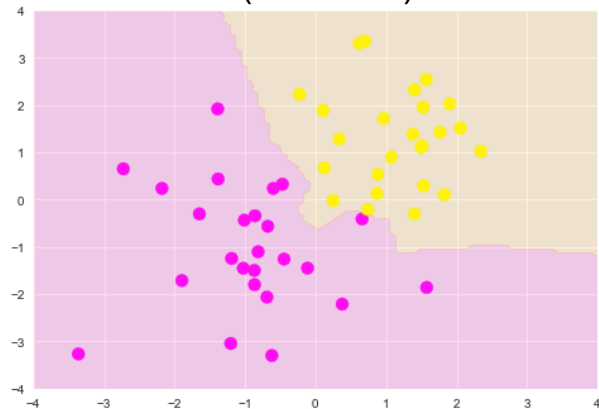


紫2個、黄3個で
黄クラスに分類

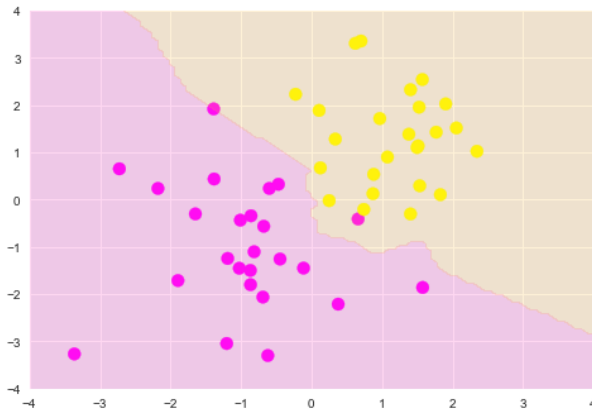
k近傍法 (kNN)

- k を大きくすると決定境界は滑らかになる

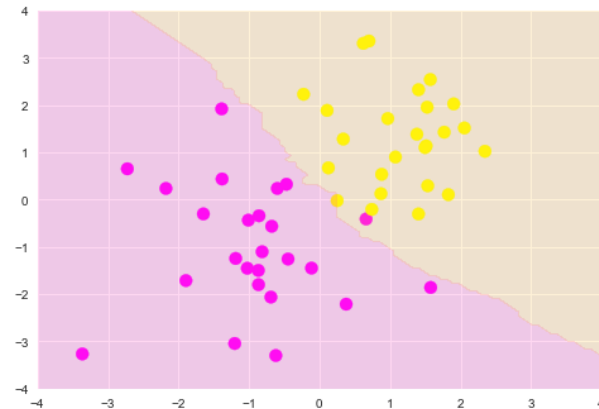
$k=1$ (最近傍法)



$k=3$



$k=10$



k近傍法 (kNN)

- ハンズオン設定
 - 人口データを分類
 - 配布済みのjupyter notebookを利用
- 課題
 - 人口データと分類結果をプロットしてください

データセット名	人口データ
レコード数	-
カラム数	-
詳細	-
備考	

k-means

1. 講義

- a. アウトライン
- b. 数学的定式化
 - i. データ形式の説明
 - ii. k平均法の説明
 - iii. アルゴリズム

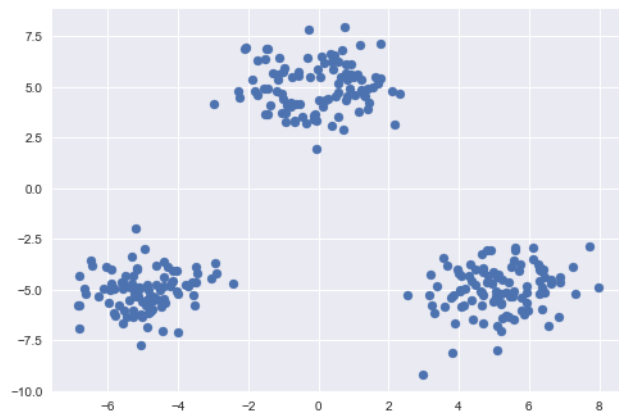
1. ハンズオン

- a. syntheticデータ分析

k-平均法 (k-means)

- 教師なし学習
- クラスタリング手法
- 与えられたデータをk個のクラスタに分類する

クラスタリング・・・特徴の似ているもの同士をグループ化



k平均法(k-means)のアルゴリズム

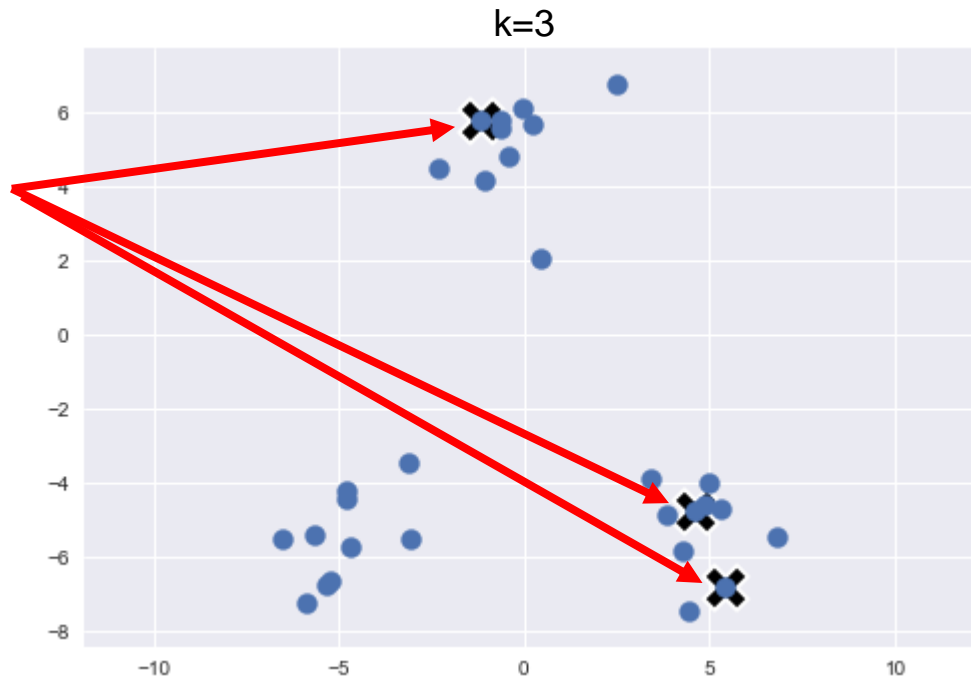
- 1) 各クラスタ中心の初期値を設定する
- 2) 各データ点に対して、各クラスタ中心との距離を計算し、最も距離が近いクラスタを割り当てる
- 3) 各クラスタの平均ベクトル（中心）を計算する
- 4) 収束するまで2, 3の処理を繰り返す

次から各手順の詳細を説明する

手順①

- 各クラスタ中心の初期値を設定する

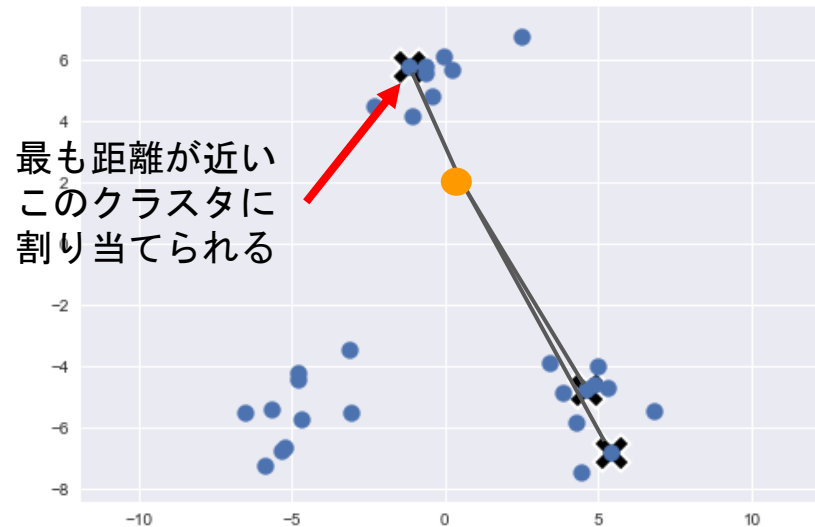
最初のクラスタ中心を
ランダムに選ぶ



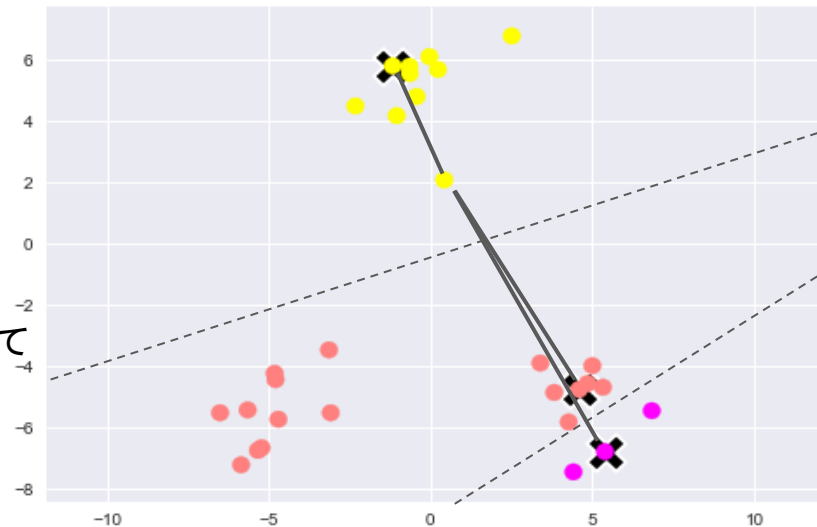
k-平均法 (k-means)

手順②

- 各データ点に対して、各クラスタ中心との距離を計算し、最も距離が近いクラスタを割り当てる

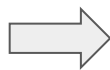
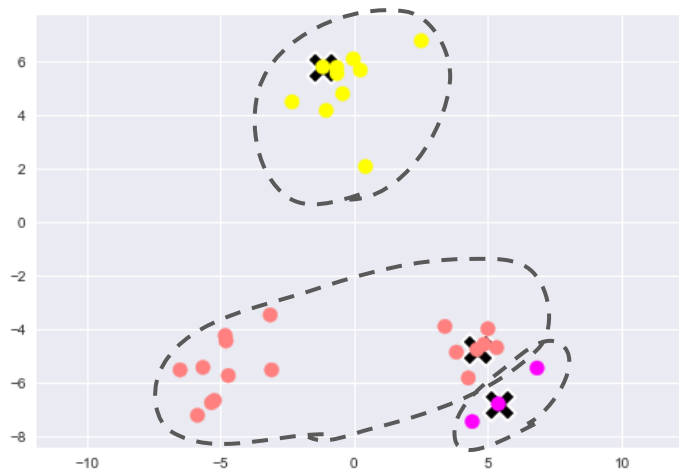


データ全て
に対して

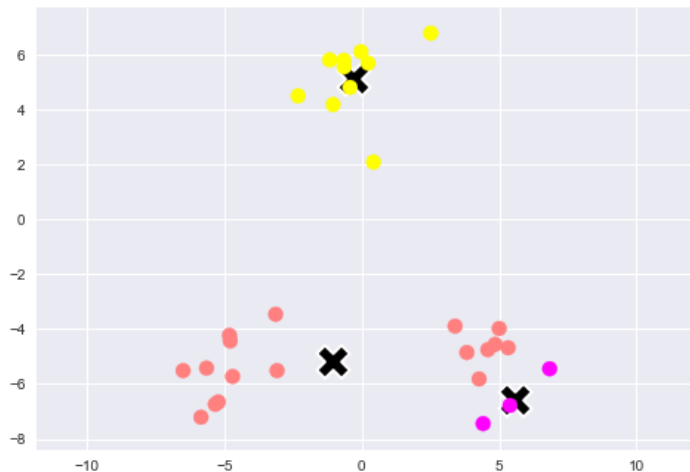


手順③

- 各クラスタの平均ベクトル（中心）を計算する

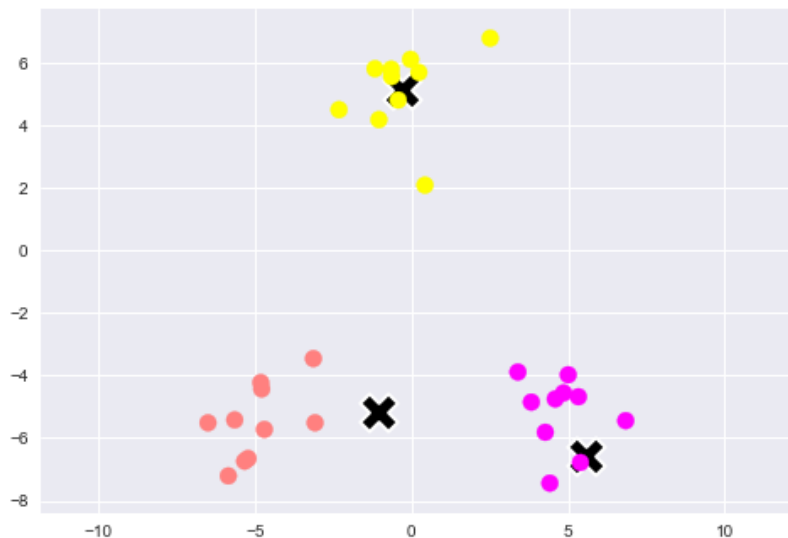


中心を更新

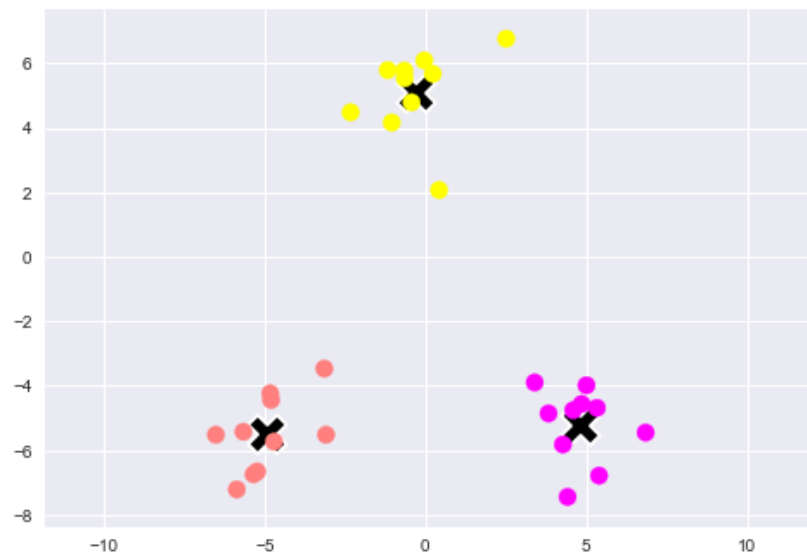
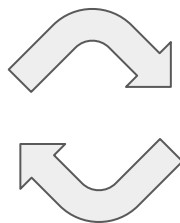


手順④

- クラスタの再割り当てと、中心の更新を繰り返す



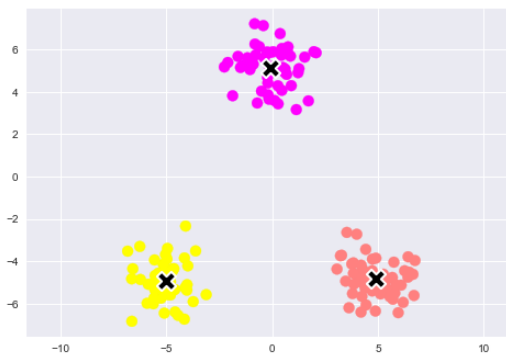
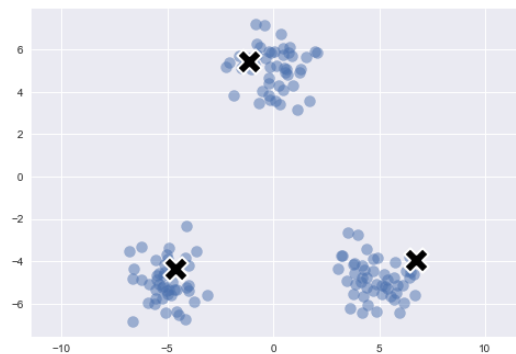
クラスタの再割り当て



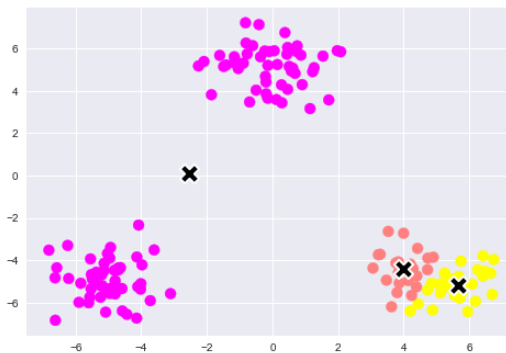
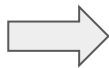
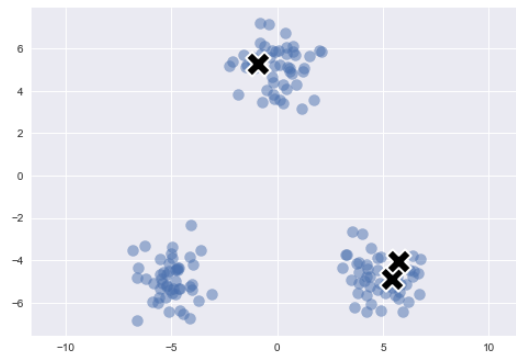
中心の更新

k-平均法 (k-means)

- 中心の初期値を変えるとクラスタリング結果も変わってしまう



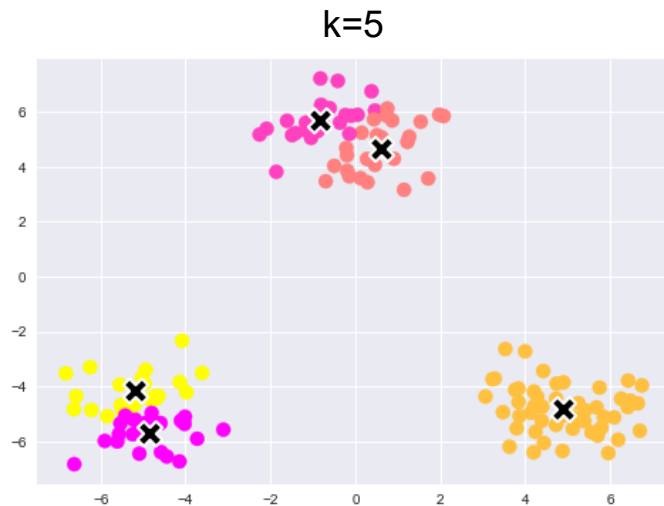
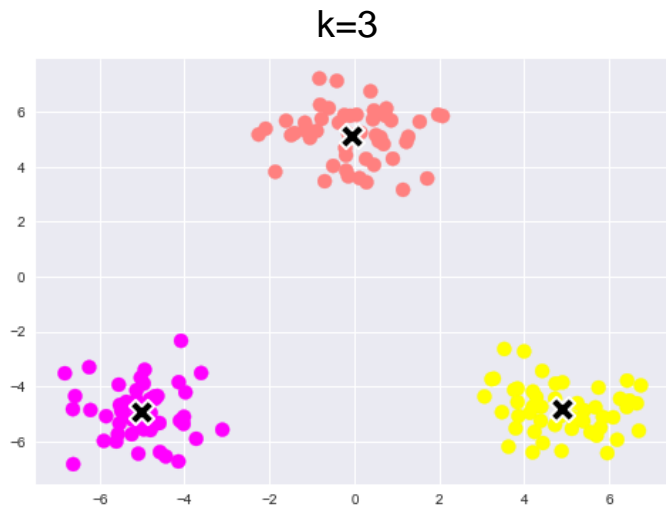
初期値が離れる
▶
うまくクラスタリングできる



初期値が近い
▶
うまくクラスタリングできない

k平均法(k-means)

- k の値を変えるとクラスタリング結果も変わる



Appendix