# *Pythonではじめる機械学習*

WeekendEngineerコンペティションチーム輪講会　*vol.1*

# 1章：機械学習 (Machine Learning: ML)の概要



対応ページ

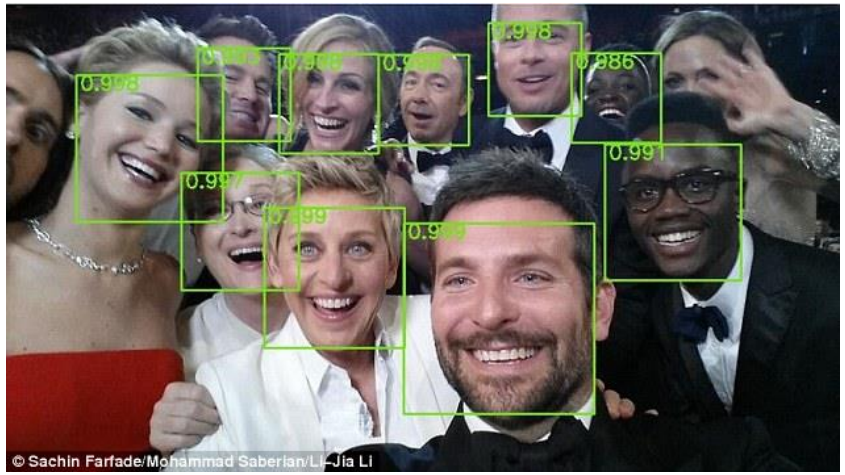
## 機械学習 (Machine Learning: ML)とは

広義的な意味

データの集合から、
みえているものからみえていないものを予測する手法

実例としては・・・


Amazonのレコメンド機能




顔認識システム

# 1章：機械学習 (Machine Learning: ML)の概要

P1,2

対応ページ

## なぜ機械学習 (ML)なのか？

「SPAMフィルタ」を例に考えてみる

＝人がルールを設計する場合

### 機械学習を使わない場合

単語のブラックリストを作成し、

その単語が
　出てきたらSPAM
　出てこなかったらSPAMではない

問題点

- ルールの設計が必要
  → 専門家が必要
- タスクが少しでも変わると
  システム全体を
  書き直さなければいけない
- 人がルールを
  機械に教え続けるには限界がある

# 1章：機械学習 (Machine Learning: ML)の概要



## なぜ機械学習 (ML)なのか？

「画像からの顔認識」を例に考えてみる

~~機械学習を使わない場合~~ ＝ 人がルールを設計する場合

機械学習を使わないとできない！

問題点
- 計算機がピクセルを「知覚」する方法が、人間が顔を認識する方法と全く異なる
  - → 人がルールを設計出来ない

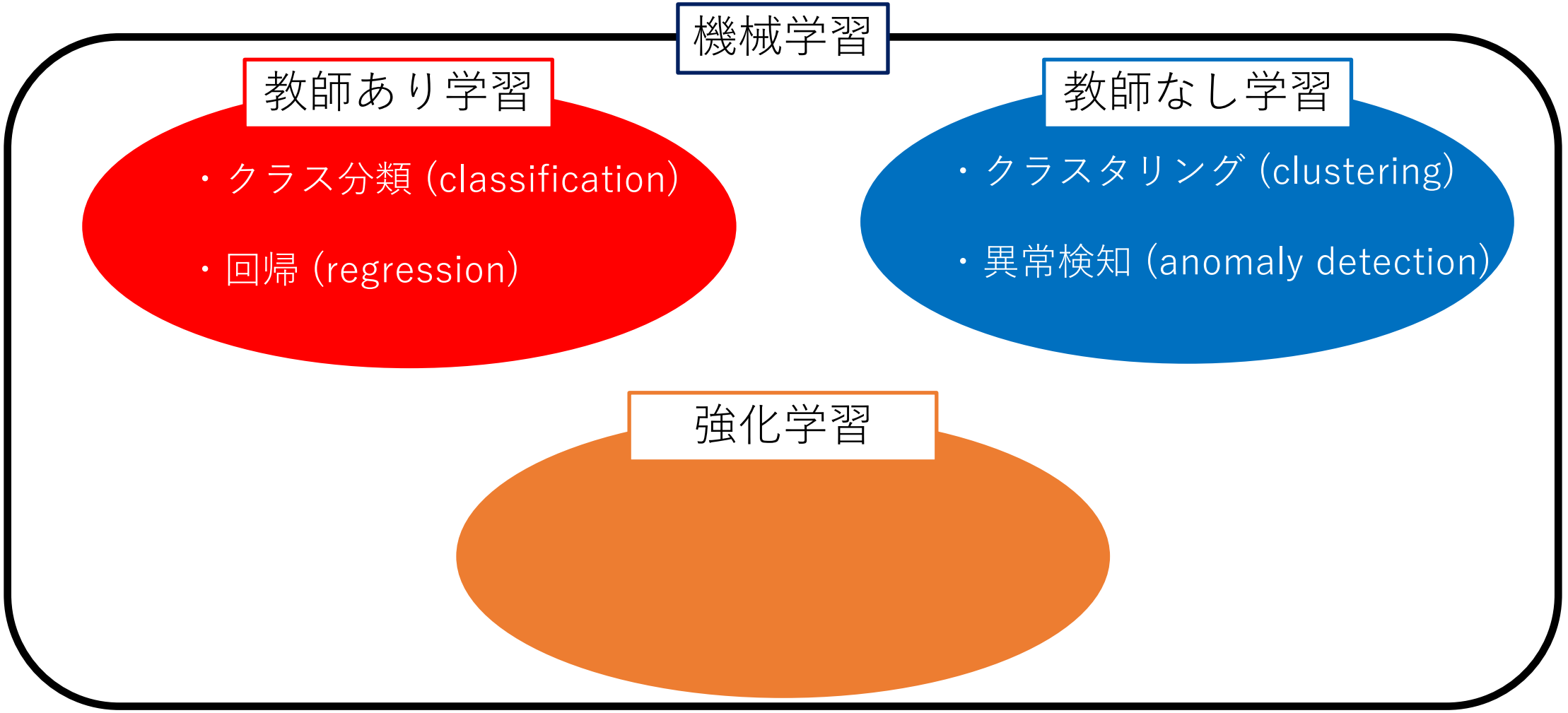
# 1章：機械学習 (Machine Learning: ML)の概要



対応ページ

## 機械学習 (ML)の分類

分類の仕方はイロイロ・・・



# 1章：機械学習 (Machine Learning: ML)の概要

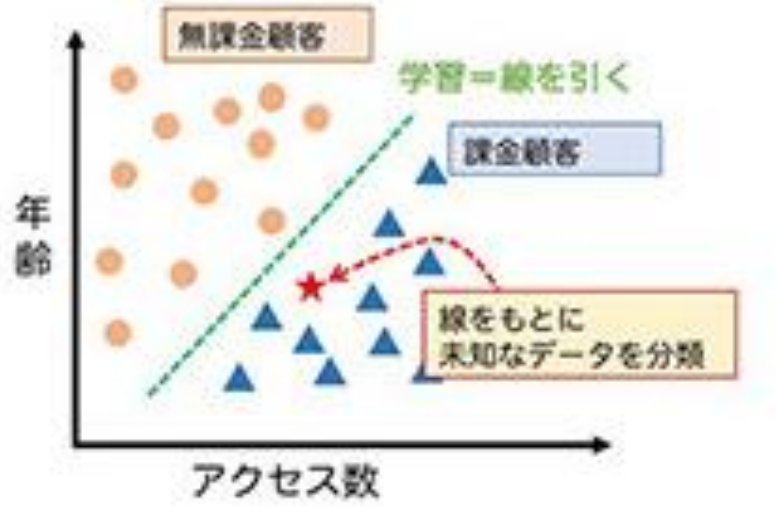


## 教師あり学習 (Supervised Learning)の概要

**学習データと教師データ (正解ラベル)**を用いて学習する手法

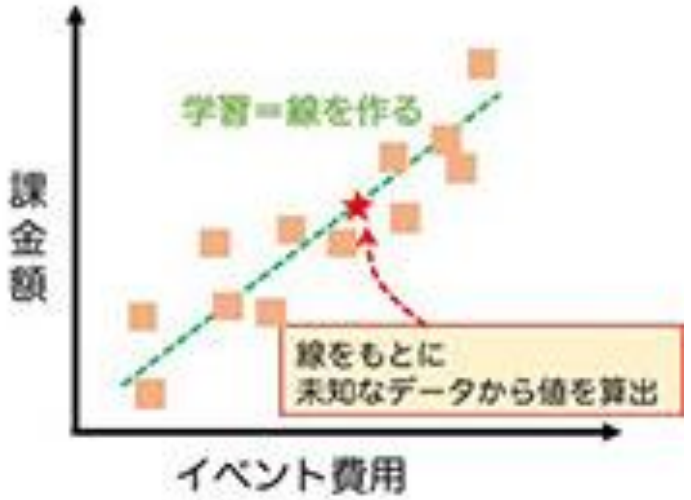
### ○クラス分類 (classification)

**クラスラベル**を予測する




### ○回帰 (regression)

**数値**を予測する



# 1章：機械学習 (Machine Learning: ML)の概要



## 教師あり学習 (Supervised Learning)の概要

**学習データと教師データ (正解ラベル)**を用いて学習する手法

例）・KaggleのTitanicコンペティション
➡2クラス分類

Titanic号の乗客が事故で
死んだか(0)生きたか(1)を予測

・MNIST (Mixed National Institute of Standards and Technology database)
➡10クラス分類

手書き数字画像「0」～「9」を予測

・iris (アヤメ)の分類
➡3クラス分類

花の画像「setosa」、「versicolor」、「virginica」を予測

・売上予測
➡回帰

# 1章：機械学習 (Machine Learning: ML)の概要

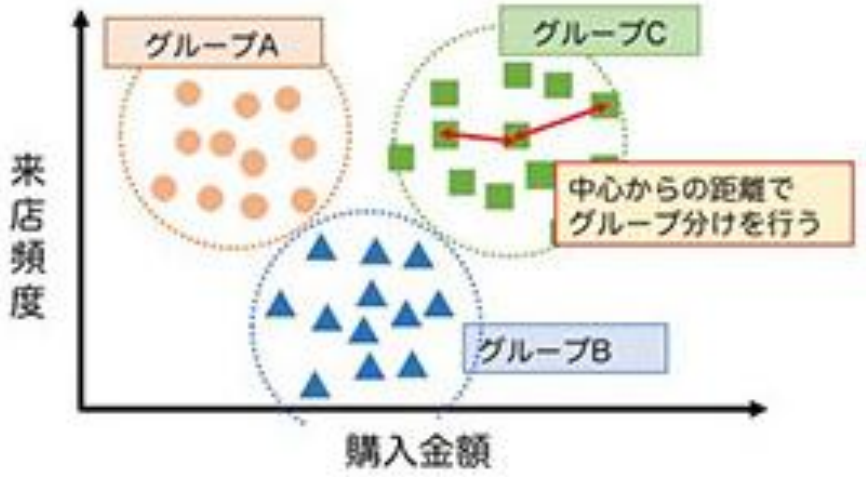


## 教師なし学習 (Unsupervised Learning)の概要

**学習データのみ**を用いて学習する手法

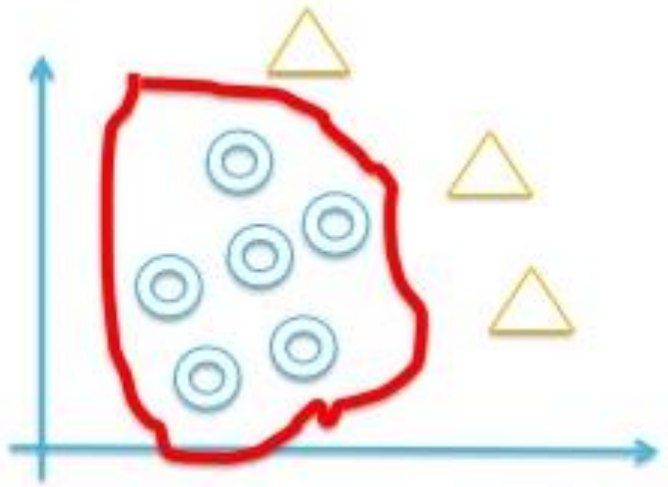
### ○クラスタリング (clustering)

**グループ分け**をする




### ○異常検知 (anomaly detection)

**異常値**を見つける



# 1章：機械学習 (Machine Learning: ML)の概要



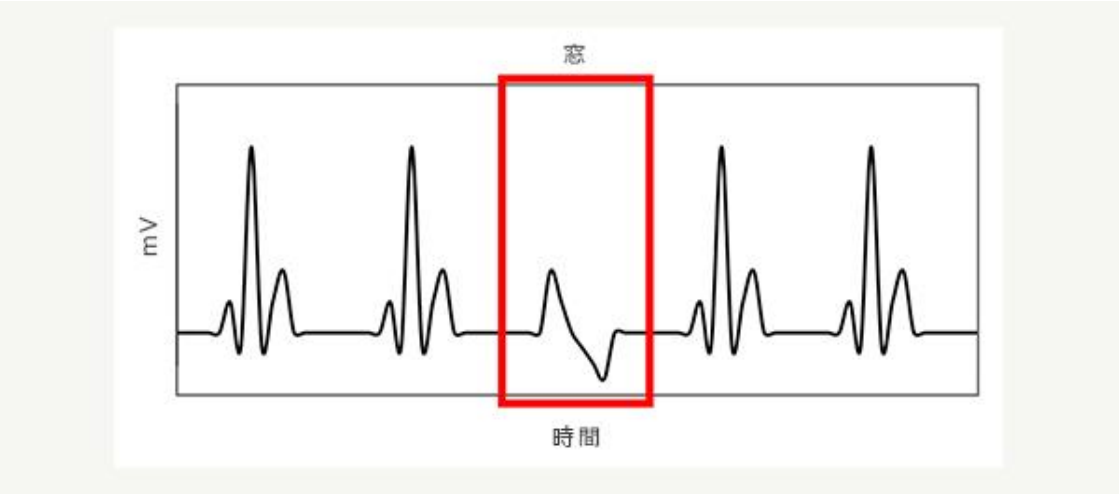
## 教師なし学習 (Unsupervised Learning)の概要

**学習データのみ**を用いて学習する手法

例）・レコメンド

おすすめの商品などを紹介

・心電図の異常検出



# 1章：機械学習（Machine Learning: ML）の概要



## タスクを知り、データを知る

大事

- **扱っているデータを理解する**
- **解決しようとしている問題とデータとの関係を理解する**

アルゴリズムには、
それぞれ得意とする
データの種類や
問題の設定がある

適当にアルゴリズムを選んでデータを投げればいいわけではない

- 解決しようとしている問題は何か？
  集めたデータでその問題が解決できるのか？
- 機械学習が適しているのか？
- 解決しようとしている問題を解くために十分なデータを集めたか？
- どのような特徴量を抽出しただろうか？
  その特徴量で正しい予測が可能だろうか？
- どのような評価を行えばよいか？

機械学習はあくまで手段であり、
目的は問題を解決すること

# 1章：機械学習 (Machine Learning: ML)の概要



## iris (アヤメ)のクラス分類

iris (アヤメ)のSepal (がく片)、Petal (花弁)の幅、長さから
「setosa」、「versicolor」、「virginica」の3品種に分類

# 1章：機械学習 (Machine Learning: ML)の概要

P14-17

対応ページ

## iris (アヤメ)のクラス分類

①データの読み込み

sklearnのライブラリからインポート

②学習データとテストデータの分割

③データの検査

④学習モデルの構築

⑤モデルの評価

# 1章：機械学習 (Machine Learning: ML)の概要



## iris (アヤメ)のクラス分類

詳細は
5章で

①データの読み込み

②学習データとテストデータの分割

③データの検査

④学習モデルの構築

⑤モデルの評価

データはラベルでソートされているため
シャッフルしないと出力ラベルが偏る

データセットをシャッフルしてから
学習データ75%、テストデータ25%に分割

# 1章：機械学習 (Machine Learning: ML)の概要

P19, 20

対応ページ

## iris (アヤメ)のクラス分類

①データの読み込み

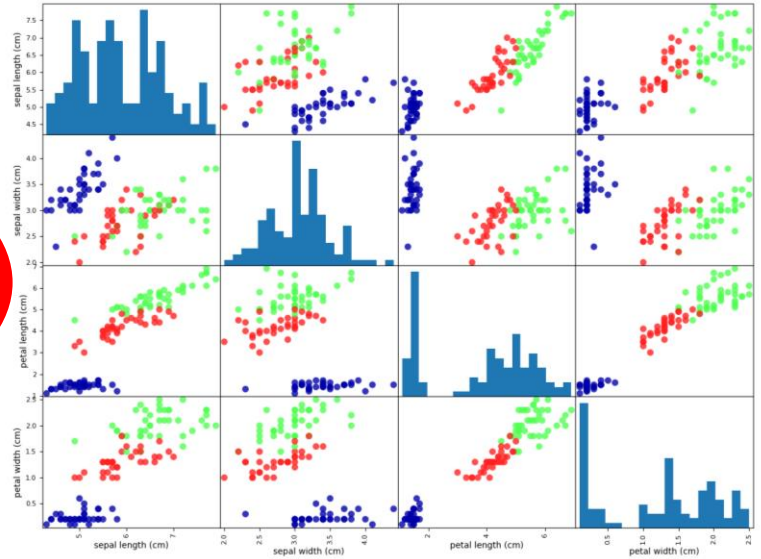
データがうまく分離しているからok

②学習データとテストデータの分割

③データの検査

④学習モデルの構築

⑤モデルの評価




ペアプロットで可視化

全ての特徴量の組み合わせをプロットしたもの

全ての特徴量の相関を同時に見ているわけではないので、
特徴量の数が少ない場合はうまくいく

# 1章：機械学習 (Machine Learning: ML)の概要

P20-22

対応ページ

## iris (アヤメ)のクラス分類

①データの読み込み

②学習データとテストデータの分割

③データの検査

④学習モデルの構築

⑤モデルの評価

学習データに近いデータに合わせる手法
(詳細は2章で)

k-近傍法 (k-Nearest Neighbors: KNN)を採用

ここではk=1

# 1章：機械学習 (Machine Learning: ML)の概要



## iris (アヤメ)のクラス分類

①データの読み込み

②学習データとテストデータの分割

③データの検査

④学習モデルの構築

⑤モデルの評価

テストデータを用いて評価する

# 2章：教師あり学習 (Supervised Learning)

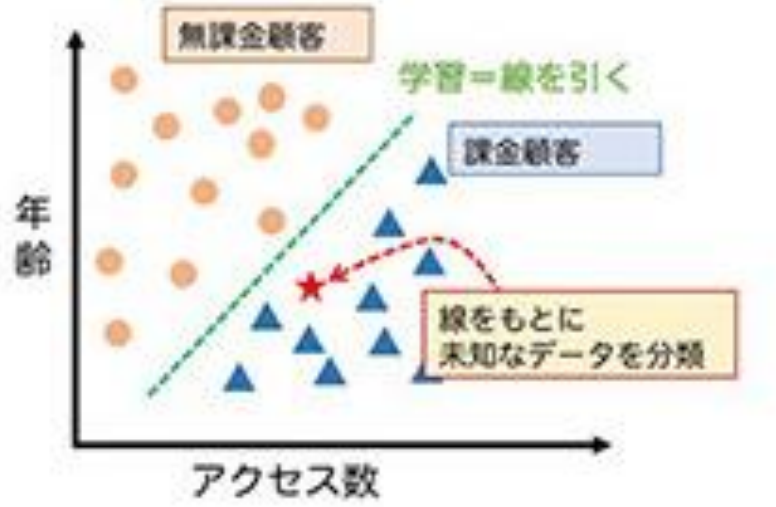


## 教師あり学習 (Supervised Learning)の概要

**学習データと教師データ (正解ラベル)**を用いて学習する手法

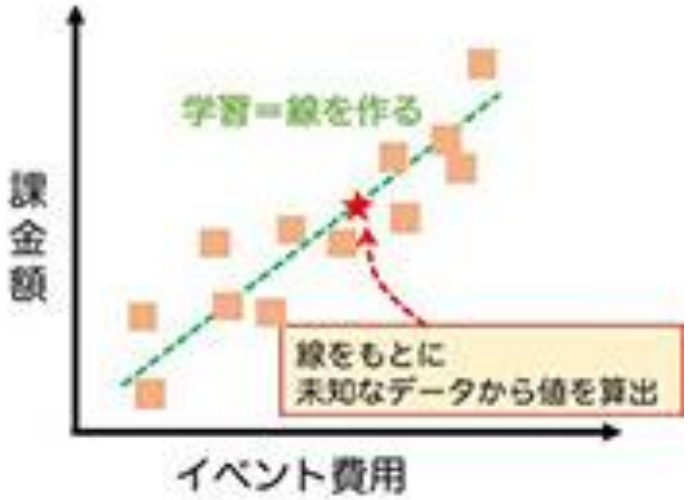
### ○クラス分類 (classification)

**クラスラベル**を予測する




### ○回帰 (regression)

**数値**を予測する



# 2章：教師あり学習 (Supervised Learning)

P28-31

対応ページ

## 汎化性能 (generalization performance)

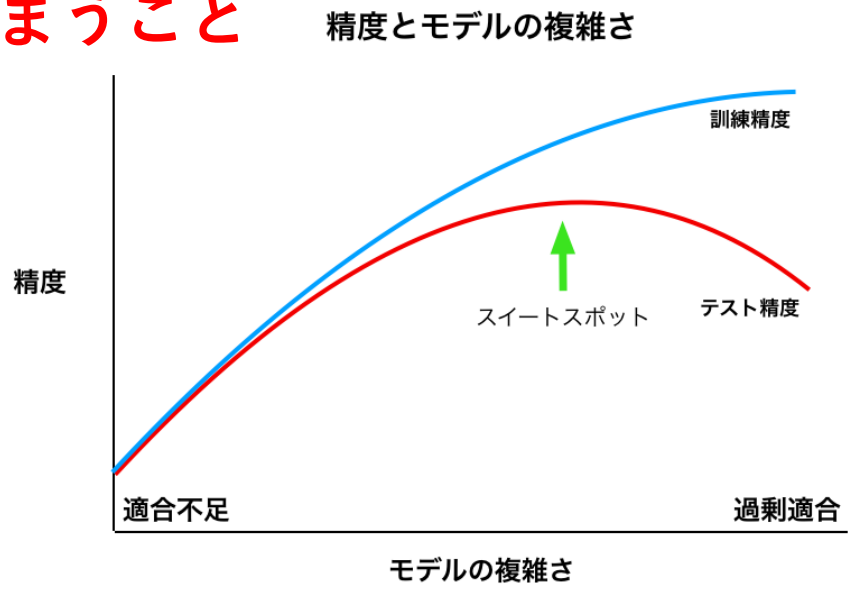
**学習データ以外のデータに対しても正しい予測ができる能力**

過学習 (over fitting) (過剰適合) ⇔ 適合不足 (under fitting)

訓練データにすらうまく学習ができないこと

**特定の学習データのみに特化した学習をしてしまうこと**

最良の汎化性能を示す
スイートスポットが
理想のモデルである

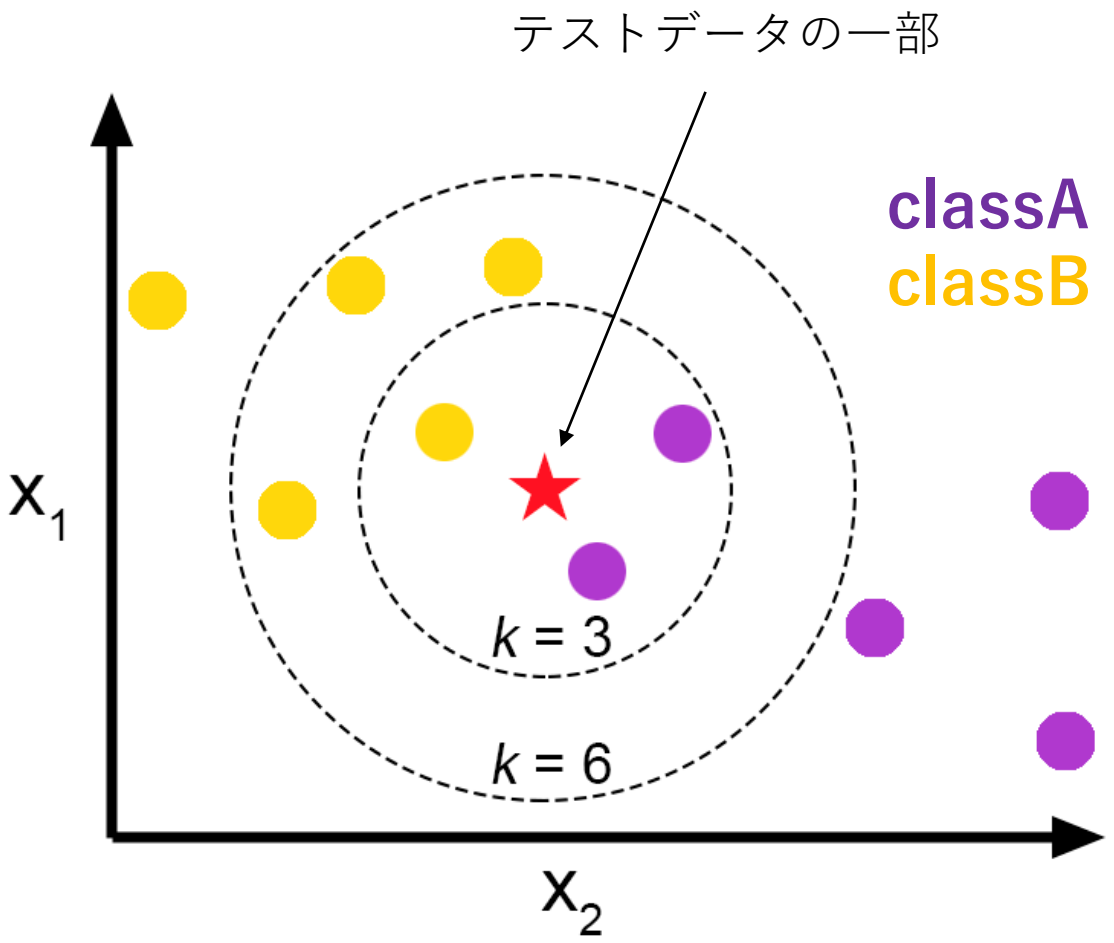


# 2章：教師あり学習 (Supervised Learning)



## k近傍法(k Nearest Neighbor: kNN)

クラス分類だけではなく、
回帰にも使える




学習データをベクトル空間上にプロットしておき、

未知のデータが得られたら、

そこから距離が近い順に任意のk個を取得し、

**多数決**でデータが属するクラスを推定する

# 2章：教師あり学習 (Supervised Learning)



## k近傍法(k Nearest Neighbor: kNN)

### 利点

・モデルが理解しやすいため、モデル構築が容易
・パラメータ調整による影響が比較的小さい

### 欠点

・多数の特徴量を持つデータセットでは十分な精度がでない
・前処理による影響が比較的大きい
・データ個数が多いほど処理速度が遅い(計算時間がかかる)

**欠点より、**
**実際にはほとんど使われていない**

# 2章：教師あり学習 (Supervised Learning)



## 線形モデル (linear model)による回帰

x :特徴量
w：xに対する重み
b :バイアス
y :予測値

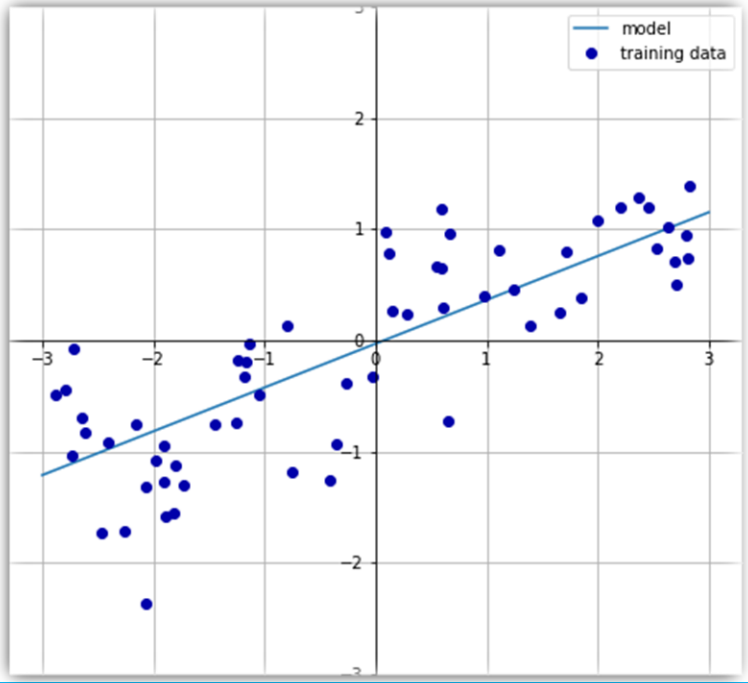
重回帰

$$y = \sum_{i=1}^{n}(w_i x_i) + b = w_1 x_1 + w_2 x_2 \cdots + w_n x_n + b$$

$$y = w * x + b$$

単回帰

低次元空間では、線形モデル
を用いるのは制約が強すぎる
かもしれない



# 2章：教師あり学習 (Supervised Learning)

P48, 49

対応ページ

## 線形回帰 (最小二乗法 (ordinary least squares: OLS))

学習データにおいて、

損失関数 (loss function)

予測値と真値の平均二乗誤差 (Mean Squared Error: MSE)
を最小にするために**重み*w*とバイアス*b***を求める

**誤差関数**

$$MSE(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$y_i$：正解値　$\hat{y}_i$：予測値

# 2章：教師あり学習 (Supervised Learning)



## リッジ回帰 (Ridge Regression)

正則化とは？

過学習を防ぐために
明示的にモデルを制約すること

*基本は線形回帰と同じだが、*
*損失関数に、L2正則化項を加える*

過学習を防ぐ

個々の特徴量が出力に与える影響をなるべく小さくしたい
➡重みwの大きさが高いものにペナルティを与える

**誤差関数**

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2+\lambda\sum_{i=1}^{n}W_i^2$$

$y_i$：正解値　$\hat{y}_i$：予測値　λ(ラムダ)：ハイパーパラメータ

# 2章：教師あり学習 (Supervised Learning)



## リッジ回帰 (Ridge Regression)

期待通り

・*線形回帰と比べ、*
　*学習データに対するスコア　➡　低い*
　*テストデータに対するスコア　➡　高い*

汎化性能が高い

・*過学習(over fitting)の危険が少ない*

・*学習データに対するスコアとモデルの簡潔さ(0に近いwの数)はトレードオフの関係*

# 2章：教師あり学習 (Supervised Learning)

P53-56

対応ページ

## Lasso

*基本は線形回帰と同じだが、*
*損失関数に、L1正則化項を加える*

いくつかの重みが完全に0になる
➡重みが0になった特徴量は無視される

**誤差関数**

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2+\lambda\sum_{i=1}^{n}|w_i|$$

$y_i$：正解値　$\hat{y}_i$：予測値　λ(ラムダ)：ハイパーパラメータ

# 2章：教師あり学習 (Supervised Learning)



## Lasso

・λ=1の場合、ほとんどのwが0になってしまい、
適合不足となった。

・λを小さくしすぎると線形回帰と同じような結果になる

・λを適度に小さくすると、
学習に使う特徴量を減らすことができ、シンプルなモデルになる
➡特徴量がたくさんあり、そのうち重要なものは
わずかしかないことが予測できる場合に有効

# 2章：教師あり学習 (Supervised Learning)



## クラス分類のための線形モデル (linear model)

2クラス分類

$$y = \sum_{i=1}^{n}(w_i x_i) + b = w_1 x_1 + w_2 x_2 \cdots + w_n x_n + b \quad > 0$$

yの値が0より大きければクラスは+1
0より小さければ - 1

- ○ロジスティック回帰 (logistic regression)
- ○線形サポートベクターマシン (Linear Support vector Machine)

# 2章：教師あり学習 (Supervised Learning)

P63-67

対応ページ

## クラス分類のための線形モデル (linear model)

多クラス分類

基本的に2クラス問題にしか適用できない

➡各クラスに対して、そのクラスと他のすべてのクラスを分類する2クラス分類を用いる

$$y = \sum_{i=1}^{n} (w_i x_i) + b = w_1 x_1 + w_2 x_2 \cdots + w_n x_n + b$$

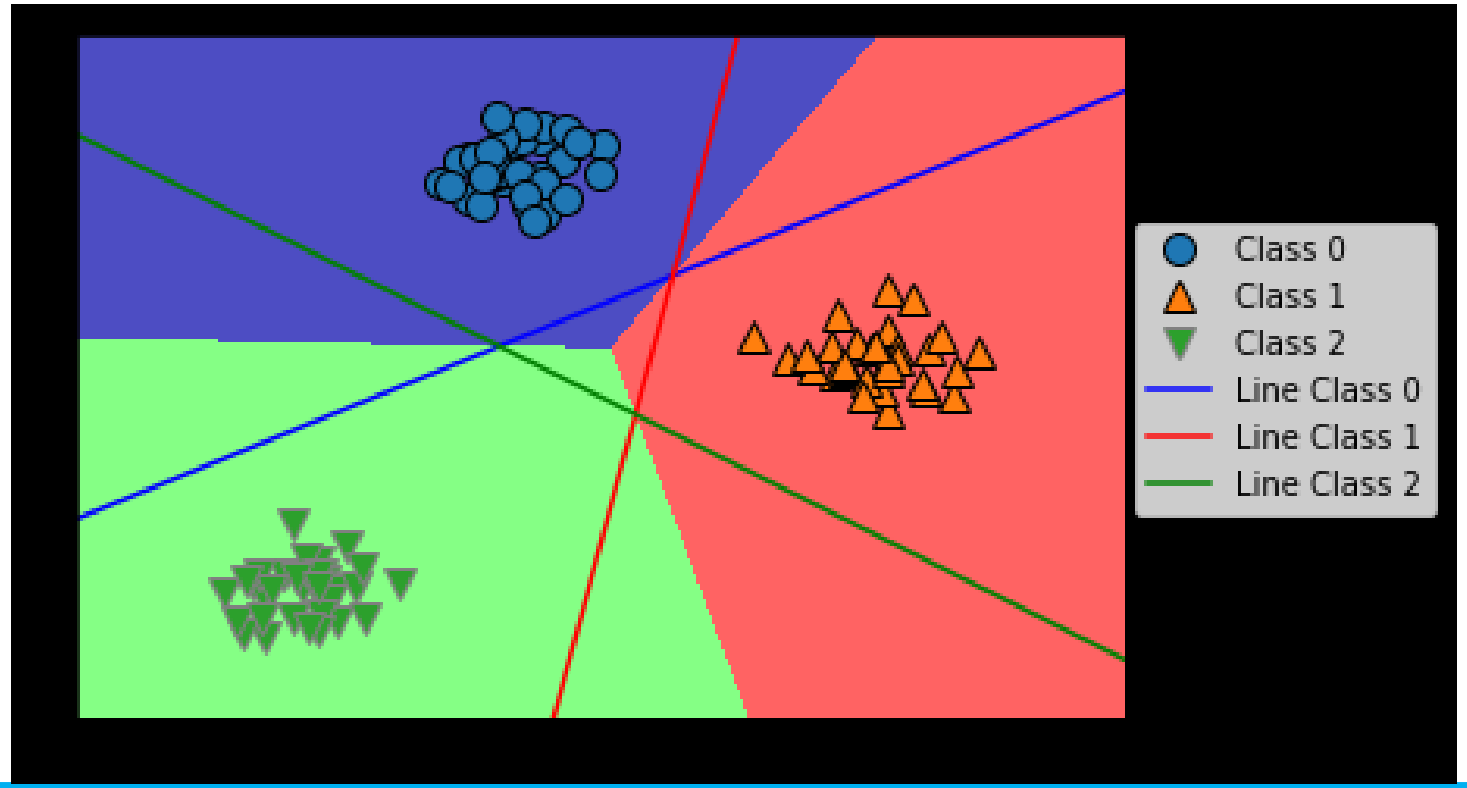
クラスごとに
重みwとバイアスbがあるからそれを用いて
クラススコアを計算

# 2章：教師あり学習 (Supervised Learning)



## クラス分類のための線形モデル (linear model)

多クラス分類



# 2章：教師あり学習 (Supervised Learning)

P46, 47

対応ページ

## ナイーブベイズ

クラス分類にしか適用できない
基本的には線形回帰と同じ

特徴

- 学習が線形回帰よりも高速
- クラス分類でしか使えない
- 線形回帰より精度が低い

scikit-learnで実装されているナイーブベイズ分類器

- GaussianNB
  ➡任意の連続値データに適用
- BernoulliNB
  ➡2値データに適用
  例)0, 1のみで構成された特徴量
- MultinomiaNB
  ➡カウントデータ(整数値)
  例)文中に出てくる単語の出現数

# 2章：教師あり学習 (Supervised Learning)

P124
-126

対応
ページ

## ここまでのまとめ

とりあえず最初に試すべきアルゴリズム

### kNN

- モデルが理解しやすく、モデル構築が容易
- パラメータ調整による影響が比較的小さい
- 多数の特徴量を持つデータセットでは十分な精度がでない
- データ個数が多いほど処理速度が遅い(計算時間がかかる)
- 前処理による影響が比較的大きい

データがうまく分離されていたら試すべき

### 線形モデル

- 処理速度が速い(計算時間がかからない)
- パラメータ調整によって過学習などの問題を解決できる
- 大きいデータや高次元のデータに適する
- パラメータ調節による影響が大きい場合があるためパラメータ調節に時間(コスト)がかかる

### ナイーブベイズ

- 線形モデルよりさらに処理速度が速い(計算時間がかからない)
- 線形モデルよりさらに高次元、大きいデータに適する
- 線形モデルより精度が劣ることが多い