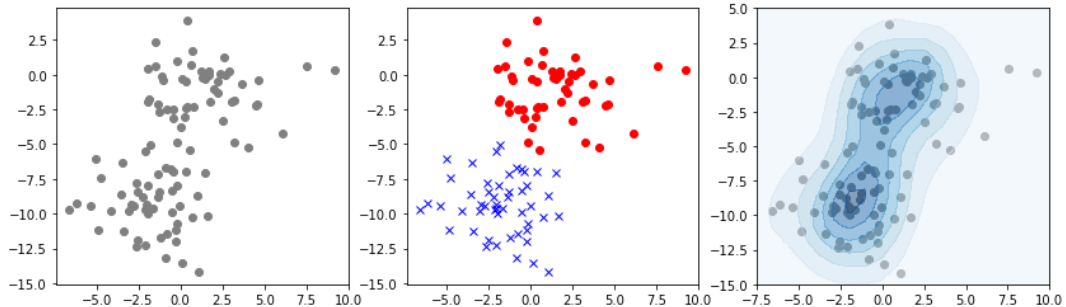
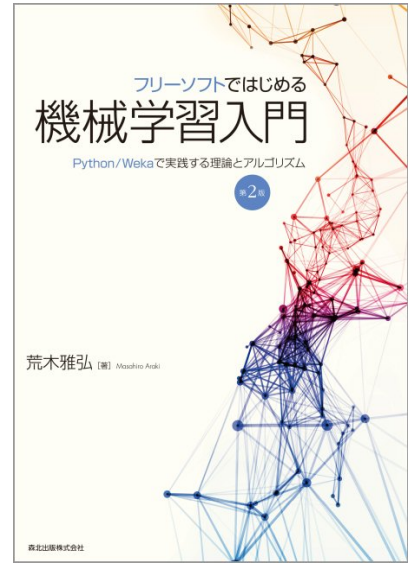
11. モデル推定



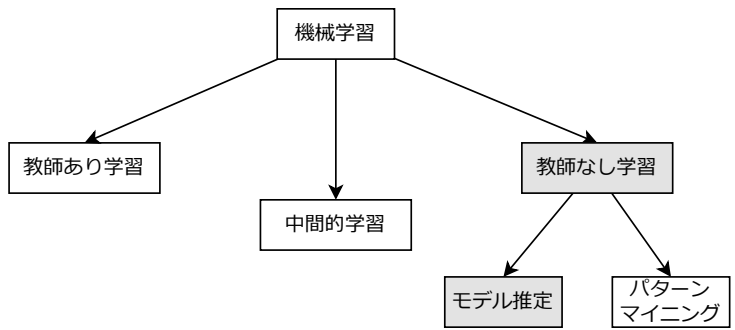
- 11.1 数値特徴に対する「教師なし・モデル推定」問題の定義
- 11.2 クラスタリング
- 11.3 異常検出
- 11.4 確率密度推定



- 荒木雅弘:『フリーソフトではじめる機械学習入門(第2版)』(森北出版, 2018年)
- スライドとJupyter notebook
- サポートページ

11.1 数値特徴に対する「教師なし・モデル推定」問題の定義(1/3)

- 問題設定
  - 教師なし学習
    - 正解なし数値ベクトル → クラスモデル
    - データ全体を説明するモデルを見つける



  - 応用例
    - 顧客セグメンテーション
    - 異常検知

11.1 数値特徴に対する「教師なし・モデル推定」問題の定義（2/3）

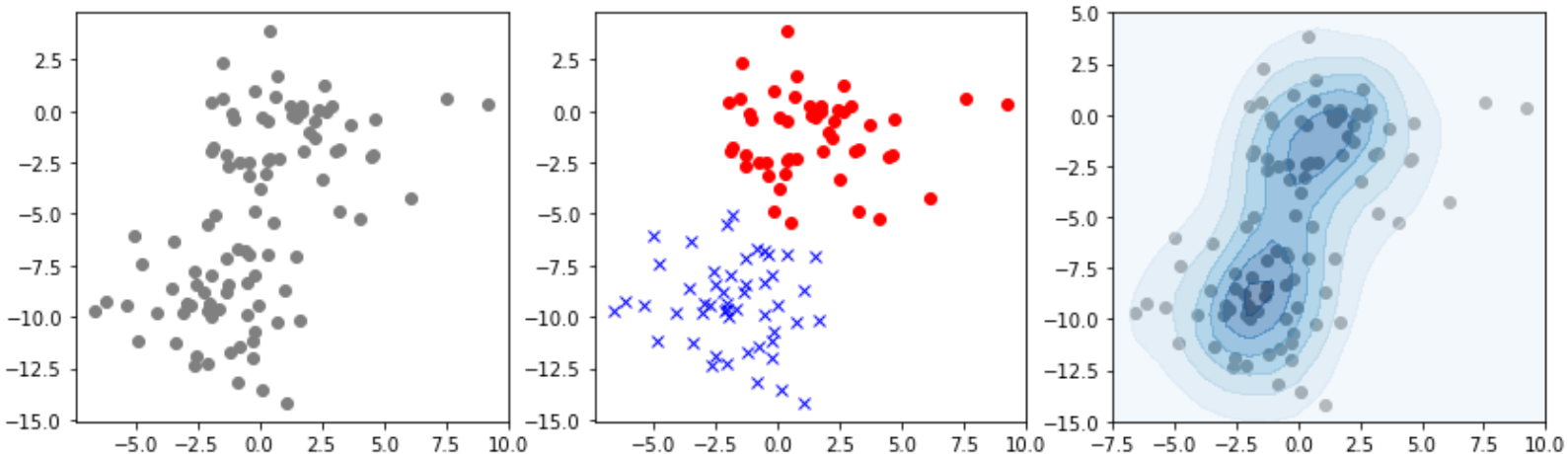
- データセット（正解なし）
  - （密な）d次元数値ベクトルの集合

$$\{\boldsymbol{x}_i\} \quad i = 1, \ldots, N$$

- モデル推定とは
  - クラスタリング
    - 個々のデータを生じさせた共通の性質をもつクラスを見つける
  - 確率密度推定
    - クラスの統計的性質を推定する
    - 与えられたデータを1クラスとみなすと、異常検知が行える

11.1 数値特徴に対する「教師なし・モデル推定」問題の定義(3/3)

- 正解なしデータ、クラスタリング結果、確率密度推定結果



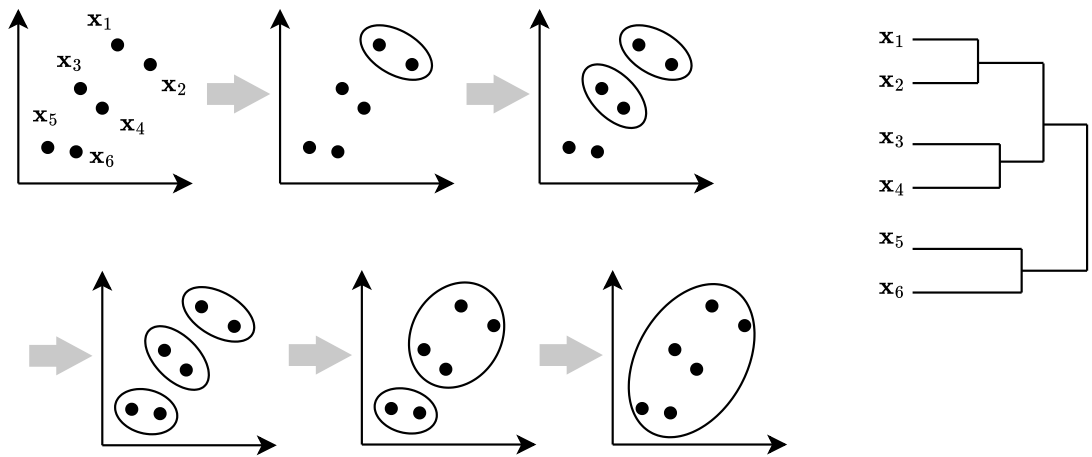
11.2 クラスタリング

- クラスタリングとは
  - 「共通の性質をもつクラス」=「特徴空間上で近い値をもつデータの集まり」と考え、データのまとまりを見つける
  - まとまり:「内的結合の小ささ」と「外的分離の大きさ」が同時に満たされる集合
    - 内的結合: 同じ集合内のデータ間の距離
    - 外的分離: 異なる集合間の距離
- クラスタリング手法の分類
  - 階層的手法
    - ボトムアップ的にデータをまとめてゆく
  - 分割最適化手法
    - トップダウン的にデータ集合を分割し、最適化してゆく

11.2.1 階層的クラスタリング（1/5）

- 階層的クラスタリングの手順

  1. 1データ1クラスタからスタート
  2. 距離（linkage）が最小のクラスタ対を求めて、1つにまとめる
  3. 2.を繰り返し、全データが1クラスタになれば終了



11.2.1 階層的クラスタリング（2/5）

- 距離（linkage）の定義とできるクラスタの傾向
  - 単連結法（single）
    - 定義：最も近いデータ対の距離
    - 傾向：クラスタが一方向に伸びやすくなる
  - 完全連結法（complete）
    - 定義：最も遠いデータ対の距離
    - 傾向：直径の小さいクラスタが優先的に形成される
  - 群平均法（average）
    - 定義：すべてのデータ対の距離の平均
    - 傾向：単連結と完全連結の中間的な形
  - Ward法（ward）
    - 定義：融合前後の「クラスタ内のデータと平均との距離の二乗和」の差
    - 傾向：極端な形になりにくく、よく用いられる基準

11.2.1 階層的クラスタリング（3/5）

- irisデータの0, 1次元目から2次元の教師なしデータを作成してクラスタリング

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.cluster import AgglomerativeClustering, KMeans, AffinityPropagation
from sklearn.mixture import GaussianMixture
from sklearn.neighbors import LocalOutlierFactor

iris = load_iris()
X = iris.data[:,0:2]

# データ表示用関数
def result_plot(X, y):
    for t in set(y):
        plt.scatter(X[y==t,0], X[y==t,1])
    plt.legend(set(y))
```

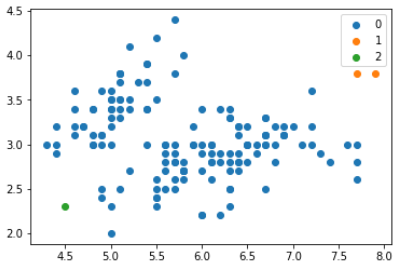
11.2.1 階層的クラスタリング（4/5）

- `AgglomerativeClustering` のパラメータ
  - `linkage`：距離の基準。デフォルトは `ward`
  - `n_clusters`：結果のクラスタ数。デフォルトは2
- メソッド
  - `fit`：正解なしデータを引数として呼び出すと、`labels_` 属性にクラスタリング結果が得られる

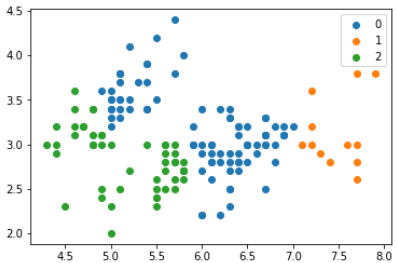
```
# クラスタ数を3に指定して階層的クラスタリング
ac = AgglomerativeClustering(n_clusters=3)
ac.fit(X)
result_plot(X, ac.labels_)
```

11.2.1 階層的クラスタリング（5/5）

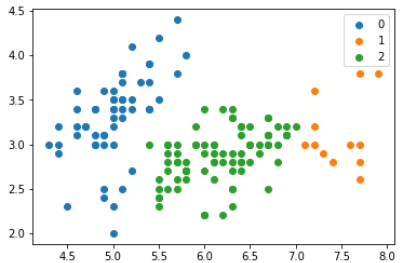
- 距離の基準とクラスタリング結果



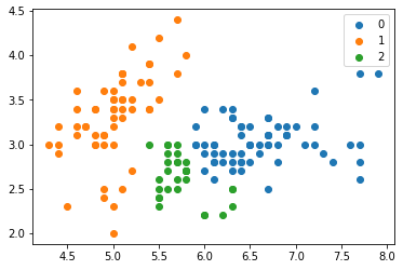
single



complete



average



ward

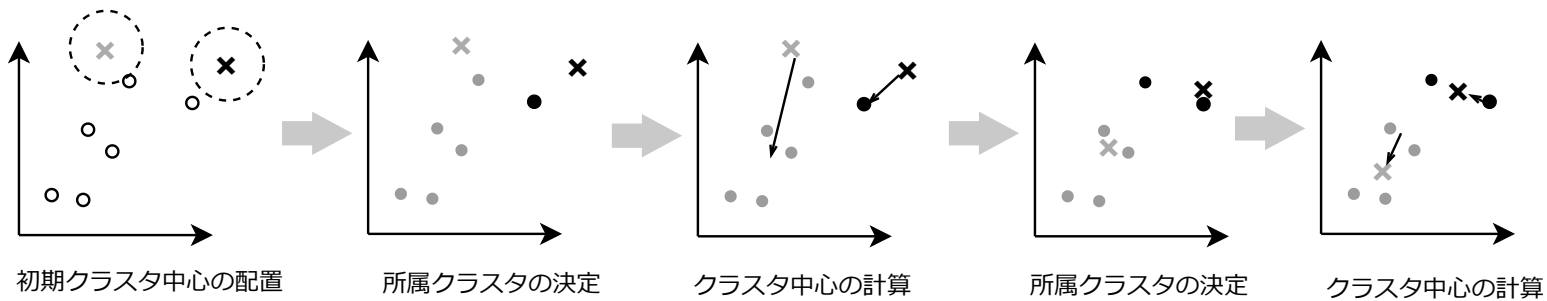
11.2.2 分割最適化クラスタリング（1/5）

- 分割最適化クラスタリングとは
  - データ分割の良さを評価する関数を定め、その評価関数の値が最適となる分割を求める
  - ただし、すべての可能な分割に対して評価値を求めることは、データ数 N が大きくなると不可能
    - 例：2分割で 2^N 通り
  - 従って、適切な初期値から探索によって準最適解を求める

11.2.2 分割最適化クラスタリング（2/5）

- k-meansアルゴリズム

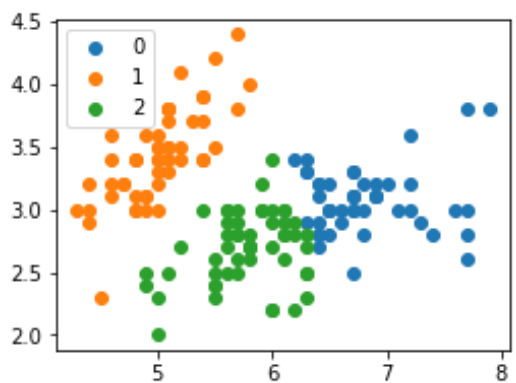
1. 分割数 k を予め与え、乱数で k 個のクラスタ中心を設定
2. 各データについて、クラスタ中心との距離に基づいて所属クラスタを決定
3. 各クラスタについて、クラスタ中心を所属データの平均ベクトルの位置に移動する
4. クラスタ中心の変化がなくなるまで 2, 3を繰り返し



11.2.2 分割最適化クラスタリング（3/5）

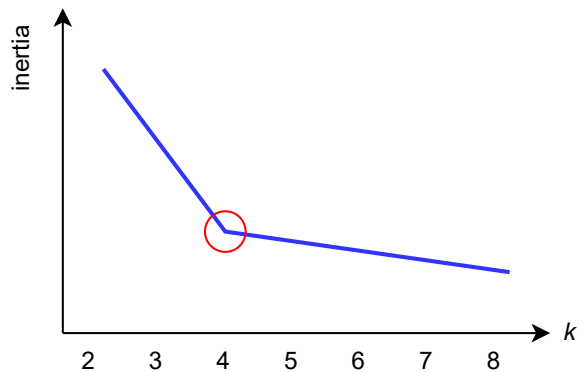
- `KMeans` のパラメータ
  - `init`：初期クラスタの決め方
    - デフォルトは初期クラスタが散らばるようにする `kmeans++`
  - `n_clusters`：クラスタ数。デフォルトは8

```
km = KMeans(n_clusters=3)
km.fit(X)
result_plot(X, km.labels_)
```



11.2.2 分割最適化クラスタリング（4/5）

- k-means法の問題点 1
  - 分割数 k を予め決めなければならない
  - 解決法
    - エルボーメソッド
      - データとクラスタ中心との平均二乗距離（inertia）を結果の評価値として、その値の減り方が鈍るところを見つける



11.2.2 分割最適化クラスタリング（5/5）

- k-means法の問題点 2
  - 得られる結果が初期値に大きく依存する
  - 解決法 ⇒ Affinity Propagation アルゴリズム
    - すべてのデータがクラスタ中心の候補
      - クラスタ中心らしさ（responsibility）とクラスタへの属しやすさ（availability）をデータ間で伝達して収束させる
      - クラスタ数を予め決める必要がない

Affinity Propagation (1/3)

- データ i とデータ k の間に定義される3つの関数
  - $s(i,k)$: データ i とデータ k の類似度。距離の反数がよく用いられる
  - $r(i,k)$: データ k がデータ i が属するクラスタの代表点となるべき証拠の累積値

$$r(i,k) \leftarrow s(i,k) - \max_{\forall k' \neq k} \{a(i,k') + s(i,k')\}$$

  - $a(i,k)$: データ i がデータ k を代表点とするクラスタに所属するべき証拠の累積値

$$a(i,k) \leftarrow \min\{0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k))\} \quad for \quad i \neq k$$

$$a(k,k) \leftarrow \sum_{i' \notin \{i,k\}} \max(0, r(i',k))$$

Affinity Propagation (2/3)

- Affinity Propagationのアルゴリズム

1. r, aの値を0で初期化
2. rを以下の式で更新(λは学習率)

$$r_{t+1}(i,k) = \lambda r_t(i,k) + (1-\lambda) r_{t+1}(i,k)$$

3. aを以下の式で更新

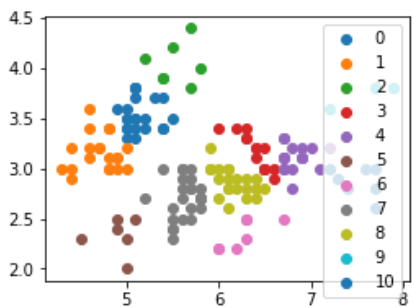
$$a_{t+1}(i,k) = \lambda a_t(i,k) + (1-\lambda) a_{t+1}(i,k)$$

4. 2,3 を収束するまで繰り返し、$r(i,i) + a(i,i) > 0$ となるものが代表点

Affinity Propagation (3/3)

- `AffinityPropagation` のパラメータ
  - `preference`: 各点の代表点としての選ばれやすさ。負にするとクラスタ数が少なくなる
- メソッド
  - `fit`: `labels_`属性にクラスタリング結果、`cluster_centers_`属性に代表点のリストが得られる

```
ap=AffinityPropagation()
ap.fit(X)
result_plot(X, ap.labels_)
```



11.3 異常検知 (1/4)

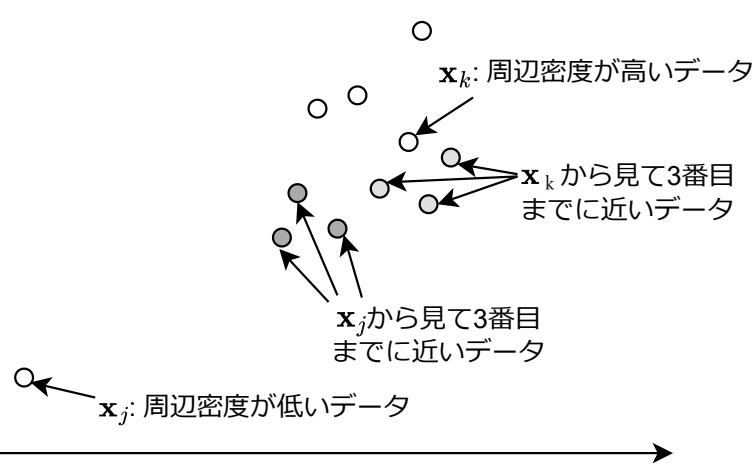
- 異常検知とは
  - 外れ値検知：データ中で、他のデータから値が外れているものを検知
  - 変化点検知：時系列信号等で観測値の振舞いの変化点を検知（例：心電図データの異常）
- 外れ値検知（静的異常検知）
  - データの分布から大きく離れている値を見つける
  - 手法
    - 観測値 $\boldsymbol{x}$ と、データの確率分布（平均 $\boldsymbol{\mu}$、共分散行列 $\boldsymbol{\Sigma}$）とのマハラノビス距離 $a(\boldsymbol{x})$ に基づいて判断する

$$a(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})$$

    - 近傍のデータ密度の違いに基づいて判断する（局所異常因子）

11.3 異常検知 (2/4)

- 局所異常因子による外れ値検知
  - 周辺密度
    - あるデータの周辺の他のデータの集まり具合
  - 局所異常因子(LOF: local outlier factor)
    - あるデータの周辺密度と、その近くの k 個のデータの周辺密度の平均との比



11.3 異常検知 (3/4)

- 局所異常因子の計算
  - 到達可能距離($\boldsymbol{x}^{(k)}$ は $\boldsymbol{x}$ に k 番目に近いデータ)

$$RD_k(\boldsymbol{x}, \boldsymbol{x}') = \max(\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|, \|\boldsymbol{x} - \boldsymbol{x}'\|)$$

  - 局所到達可能密度

$$LRD_k(\boldsymbol{x}) = (\frac{1}{k}\sum_{i=1}^{k} RD_k(\boldsymbol{x}^{(i)}, \boldsymbol{x}))^{-1}$$

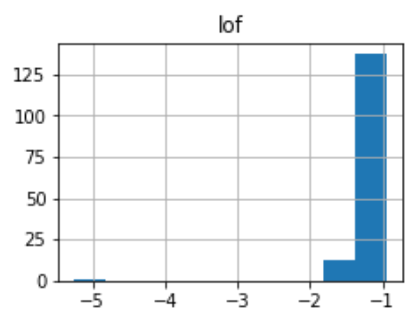
  - 局所異常因子

$$LOF_k(\boldsymbol{x}) = \frac{\frac{1}{k}\sum_{i=1}^{k} LRD_k(\boldsymbol{x}^{(i)})}{LRD_k(\boldsymbol{x})}$$

11.3 異常検知 (4/4)

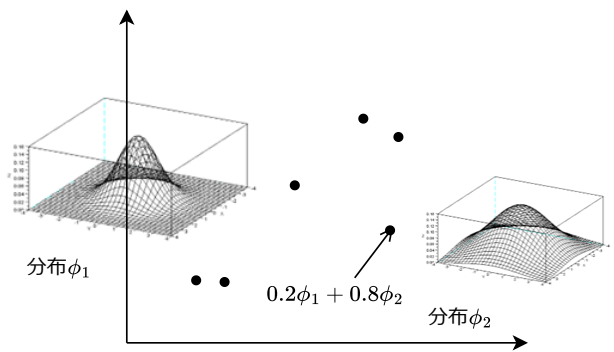
- `LocalOutlierFactor` のパラメータ
  - `n_neighbors`: 近傍とするデータ数。デフォルトは20
  - `novelty`: 新規性検出に用いるか。デフォルトは`False`

```
X, _ = load_iris(return_X_y=True, as_frame=True)
X['petal width (cm)'][0] = 2.5 + 0.76 # 異常値(最大値+1標準偏差)の混入
lof = LocalOutlierFactor()
lof.fit(X)
X['lof'] = lof.negative_outlier_factor_
X.hist(column='lof')
```



11.4 確率密度推定 (1/5)

- 教師なし学習で識別器を作る問題
  - クラスタリング結果からは、1クラス1プロトタイプの単純な識別器しかできない
  - 各クラスの事前確率や確率密度関数も推定したい
- ガウス混合分布モデル
  - データの広がりを複数の正規分布の混合で表す
  - k 個の初期分布を与え、EMアルゴリズムで最適化してゆく



11.4 確率密度推定 (2/5)

- k-means法からガウス混合分布モデルへ(EMアルゴリズム)
  - k 個のクラスタ中心を乱数で決める

    ⇒ k 個の正規分布を乱数で決める
  - クラスタ中心との距離を基準に各データをいずれかのクラスタに所属させる

    ⇒ データが分布から生成される確率に基づき、データを各クラスタに緩やかに所属させる
  - 所属させたデータをもとにクラスタ中心を再計算

    ⇒ データのクラスタへの所属度に基づき、分布のパラメータ(平均、共分散行列)を再計算

11.4 確率密度推定 (3/5)

- E (Expectation) ステップ:確率計算

$$
\begin{aligned}
p(c_m \mid \boldsymbol{x}_i) &= \frac{p(c_m)p(\boldsymbol{x}_i \mid c_m)}{p(\boldsymbol{x}_i)} \\
&= \frac{p(c_m)p(\boldsymbol{x}_i \mid c_m)}{\sum_{j=1}^{k} p(c_j)p(\boldsymbol{x}_i \mid c_j)} \\
&= \frac{p(c_m)\phi(\boldsymbol{x}_i; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}{\sum_{j=1}^{k} p(c_j)\phi(\boldsymbol{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}
\end{aligned}
$$

- M (Maximization) ステップ:分布の最尤推定

$$
\boldsymbol{\mu}_m = \frac{1}{|D|} \sum_{\boldsymbol{x}_i \in D} p(c_m \mid \boldsymbol{x}_i)\, \boldsymbol{x}_i
$$

$$
\boldsymbol{\Sigma}_m = \frac{1}{|D|} \sum_{\boldsymbol{x}_i \in D} p(c_m \mid \boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}_m)(\boldsymbol{x}_i - \boldsymbol{\mu}_m)^T
$$

11.4 確率密度推定 (4/5)

- ガウス混合分布モデルの問題点
  - 分割数 k を予め決めなければならない
- 情報量規準の最小化
  - 2分割から始めて、分割数を適応的に決定する
  - 分割の妥当性の判断：BIC (Bayesian Information Criterion)が小さくなれば分割を継続

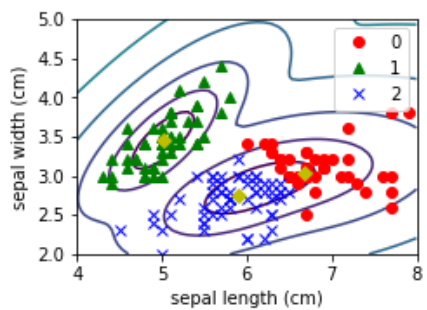
$$BIC = -2\log L + q\log N$$

    - L: モデルの尤度
    - q: モデルのパラメータ数
    - N: データ数

11.4 確率密度推定（5/5）

- `GaussianMixture` のパラメータ
  - `n_clusters`：分布の混合数。デフォルトは1
  - `covariance_type`：共分散行列のタイプ指定。デフォルトは`full`
- メソッド
  - `fit`：`means_`属性に平均ベクトル、`covariances_`属性に共分散行列が得られる

```
gmm = GaussianMixture(n_components=3, covariance_type='full')
gmm.fit(X)
```



11.5 まとめ

- モデル推定
  - データのまとまりを発見するプロセス
- 階層的クラスタリング
  - 類似度に基づいてボトムアップにデータをまとめてゆく
- 分割最適化クラスタリング
  - トップダウンでのデータの分割を最適化
- 確率密度推定
  - 分割最適化クラスタリングの一般化