機械学習（３）

概要

1. 機械学習の基礎
   - 概論
     - 教師あり学習
     - 教師なし学習
     - 強化学習
2. 教師あり学習
   - ランダムフォレスト
   - サポートベクターマシン

3. 教師なし学習と確率モデリング
   - クラスタリング
   - ナイーブベイズモデル
   - 混合ガウスモデル
   - クロスバリデーション・モデル選択
   - ベイズ推論

4. データ構造と機械学習アルゴリズム
   - テーブルデータ
   - 行列データ
   - 時系列データ
   - グラフデータ

補助資料：http://small-island.work/trial/

まず始めに

教師なし学習の基本的な方法の一つが確率モデリング

確率モデリングをすることの大きな利点

- 明確なモデルを組み込むことで比較的少ないデータ数での学習が可能
- モデル間の比較（モデル選択）→ クラスタ数の決定などが可能
- 欠損データを扱うことができる
- モデルとしてわかっている知識を活用することができる

問い：なぜ確率モデリングを学ぶ必要があるのか

回答：

・ツールを正しく使う

・独自のモデル・問題に特化したモデルを作ることもできる

・有名な深層学習モデルを学ぶ上での基礎
（VAE, GANなど）

この講義の目的

目的

１．確率モデルをブラックボックスとして利用できるようになる

２．確率モデルの
入り口（何を前提としているか）と
出口（どういった形式の出力が取り出せるか）
を理解して、正しく利用するができるようにする

３．細かい手法よりも考え方を重視
奥が深い話題もあるが概要的な話やどうやって使うかがメイン
（独自モデルに関しては別機会に…）

関連分野

人工知能

計算と人工知能

1950年： チューリングテスト

19[illegible]トマス会議

機械学習

論理・探索

第1次AIブーム: 推論と探索
1956年－1974年

1972年： Prolog

第2次AIブーム:
知識
（エキスパートシステム）
1980年－1987年

機械学習

1967年：k-means法
(教師なし学習：
クラスタリング)

1989年：Q学習
(強化学習)

1992年：非線形SVM
(教師あり学習)

1990年～：統計的機械学習
- 確率モデル
- 汎化誤差理論
- ベイズモデル

ニューラルネットワーク

1958年：パーセプトロン

1969年：パーセプトロンの限界

1986年：誤差逆伝搬法

深層学習

2006年：オートエンコーダー

第3次AIブーム:
深層学習
2006年－

確率の復習

(離散)確率変数 X, 実現値 x, 確率 $P(X = x)$

同時確率　$P(X = x, Y = y)$

$$P(X = 0, Y = 0) = 40/100$$

周辺確率　$P(Y = y) = \Sigma_x P(x, y)$

$$P(Y = 0) = 40/100 + 20/100$$

条件付確率　$P(X = x|Y = y) = \dfrac{P(X = x, Y = y)}{P(Y = y)}$

$$P(X = 0|Y = 0) = \frac{40/100}{30/50} = 2/3$$

$X \backslash Y$	0	1
0	40	10
1	20	30

度数分布表

ベイズの定理

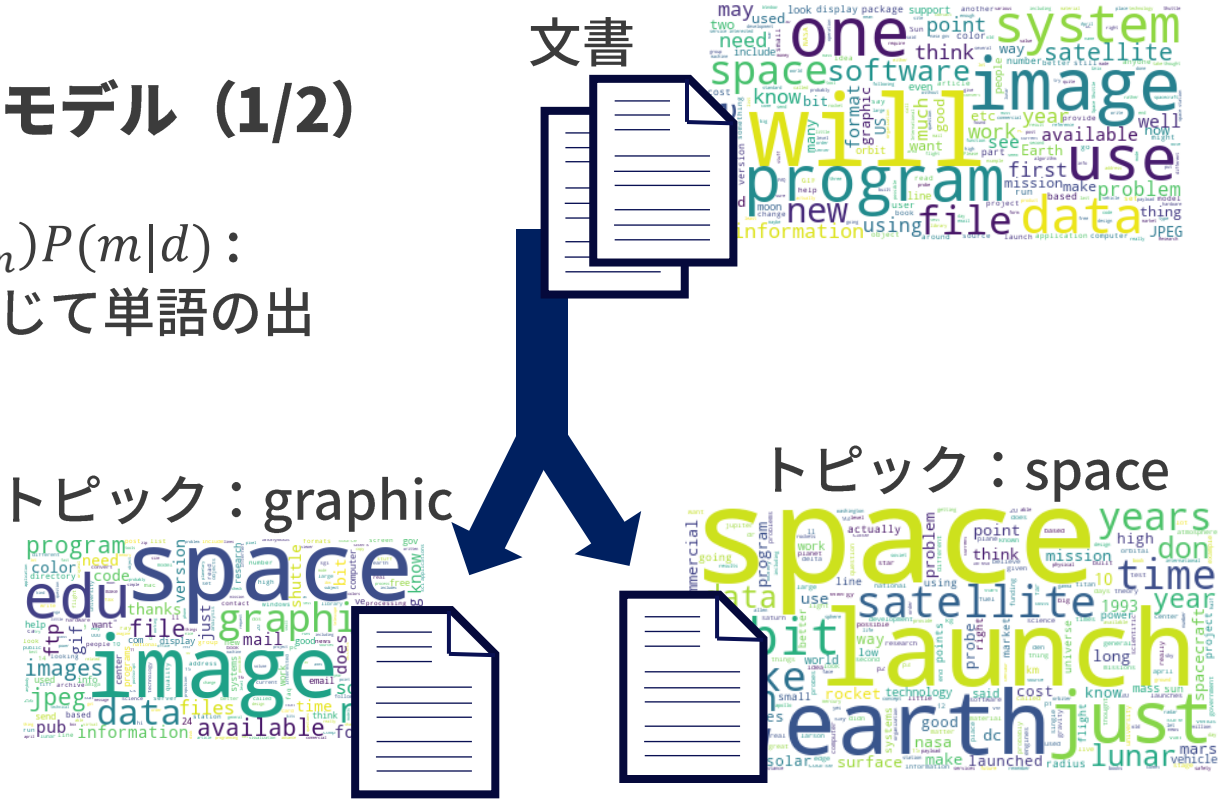
$$P(Y = y|X = x) = \frac{P(X = x|Y = y)\, P(Y = y)}{P(X = x)}$$

$$P(Y = 0|X = 0) = \frac{2/3 \cdot 3/5}{1/2} = 1/5$$

確率を使った様々なモデル（1/2）

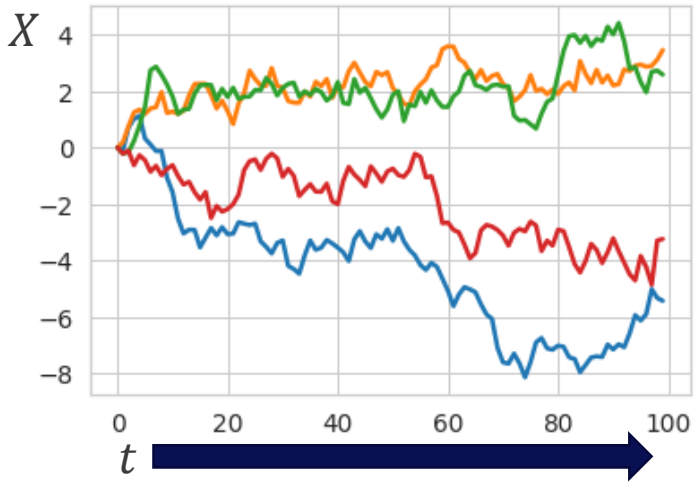
<u>トピックモデル</u>　$P(w|\phi_m)P(m|d)$：

文書dのトピックmに応じて単語の出現分布ϕ_mが決まる



<u>状態遷移モデル</u>　$P(X_t|X_{t-1})$：

次の状態が前の状態によって決まる

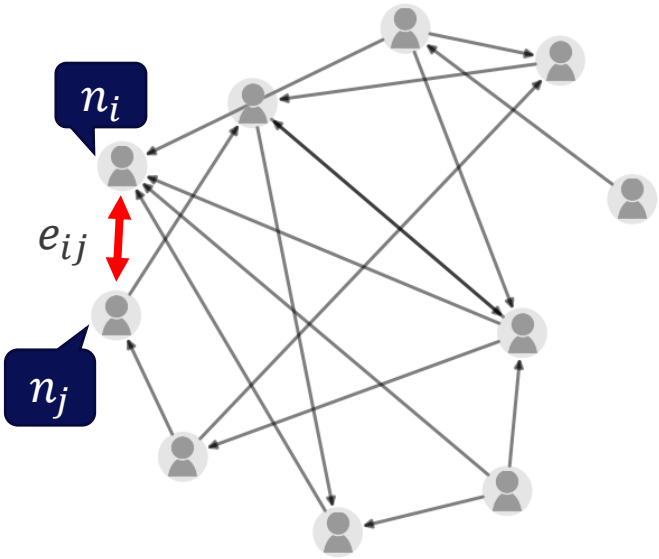


確率を使った様々なモデル（2/2）

ランダムグラフ　$P(e_{ij}|n_i, n_j)$：

iさんとjさんが友達e_{ij}の確率は

iさんとjさんの趣味n_i, n_jで決まる

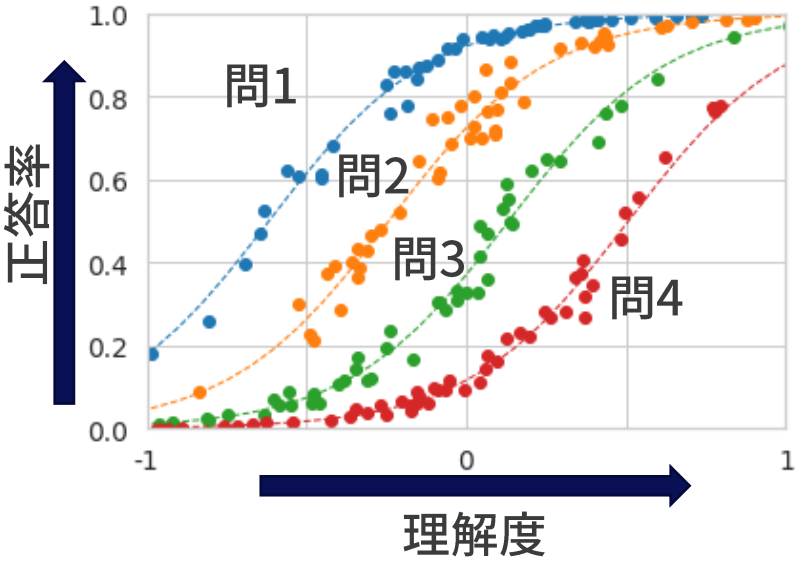


項目反応理論　$P(T_i|\theta, b_i)$：

問題iの点数はT_i、

受験者の理解度θと

難易度b_iによって決まる

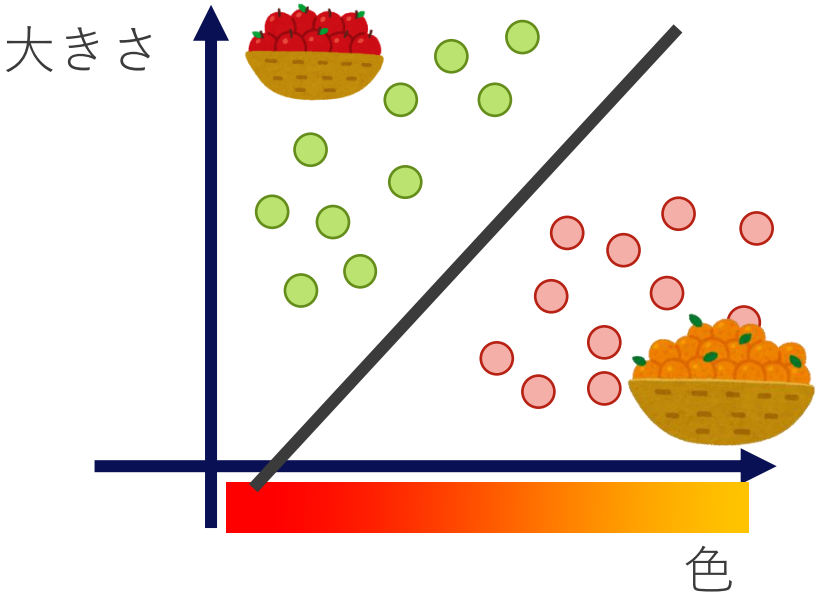


教師あり学習（識別問題）と 教師なし学習（クラスタリング）

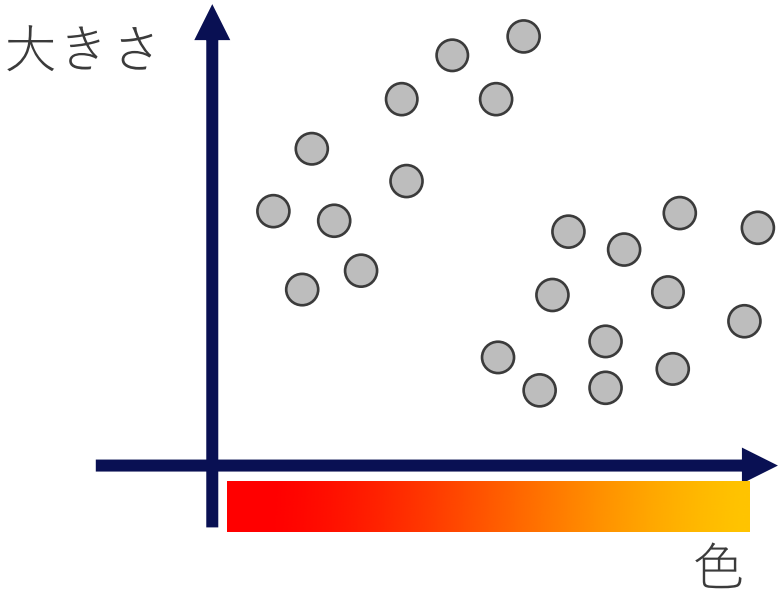
識別問題（第二回講義）：
データをクラスに識別する問題

クラスタリング：
データを複数の集合(クラスタ) に分離する問題

与えられた教師データからそれら
を分離する境界を計算

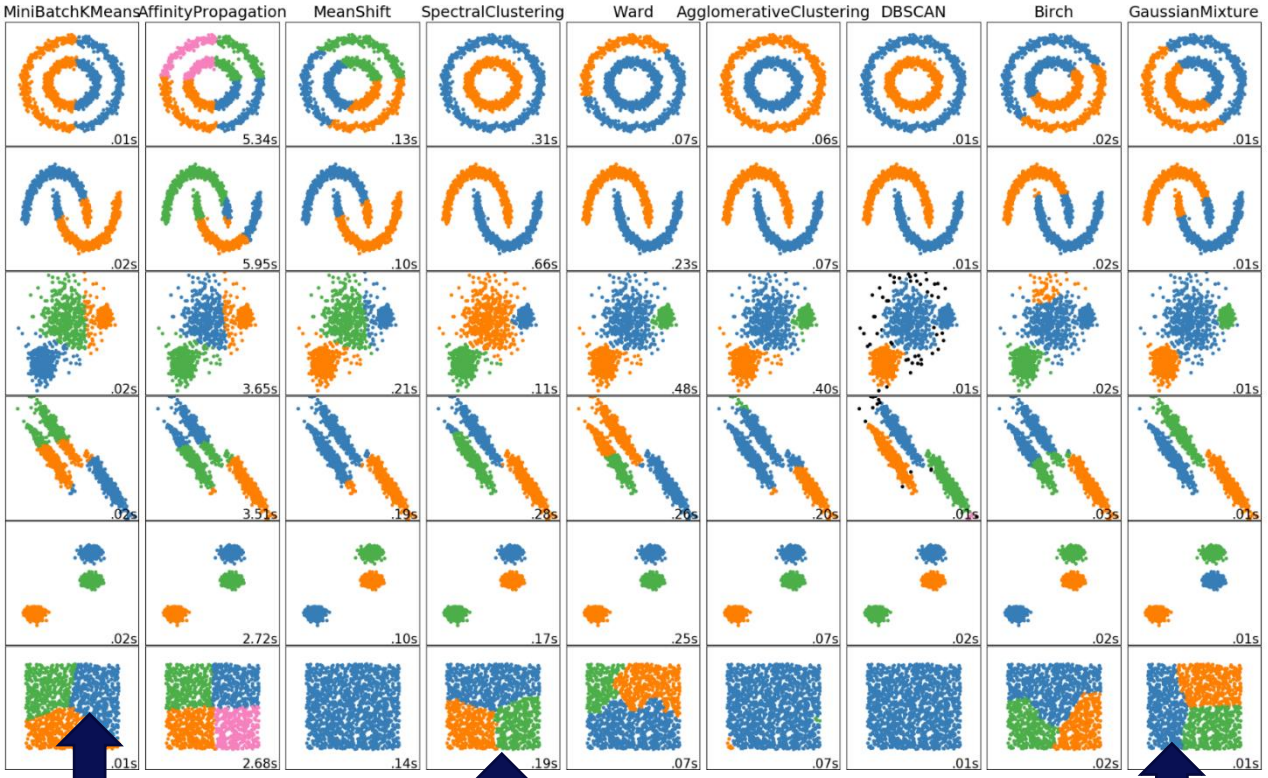


どちらがリンゴかミカンか教えなくとも
二つの集団があることがわかる



クラスタリング

正解がないことが多いのでモデルの特性をよく理解して利用する必要がある



第一回で解説済み
今回も登場

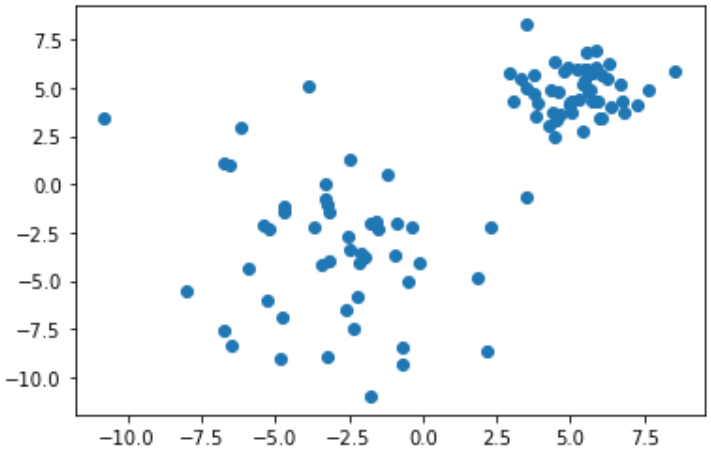
次回説明

今回説明

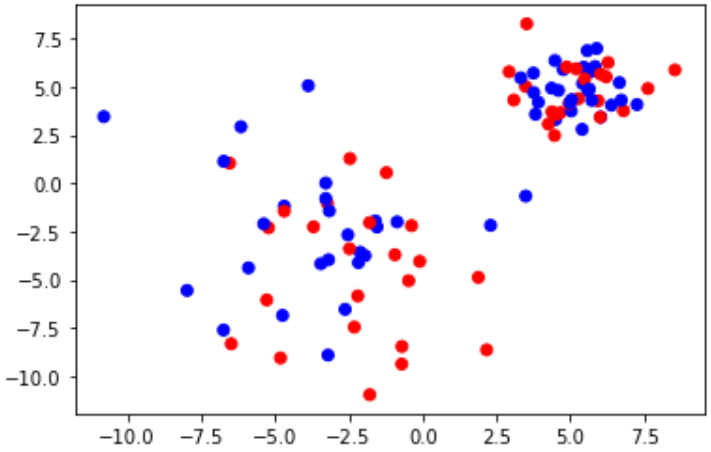
https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py

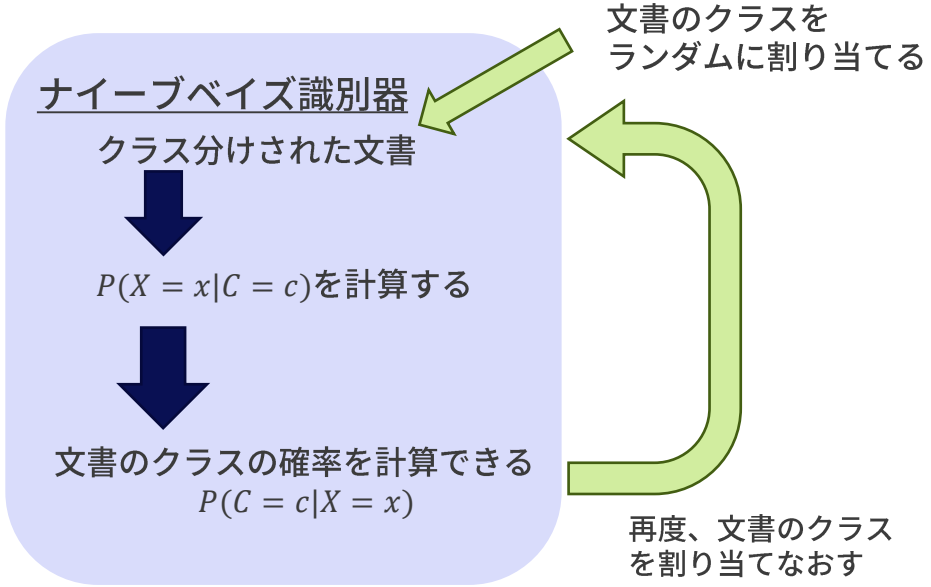
初期化（ナイーブベイズクラスタリング）

元データ



クラスをランダムに割り当てる

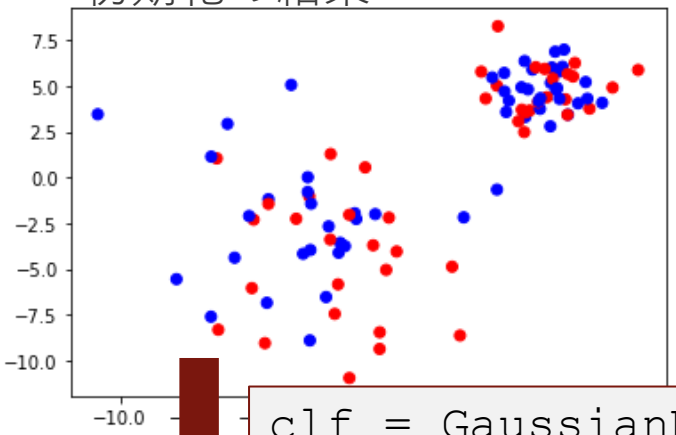




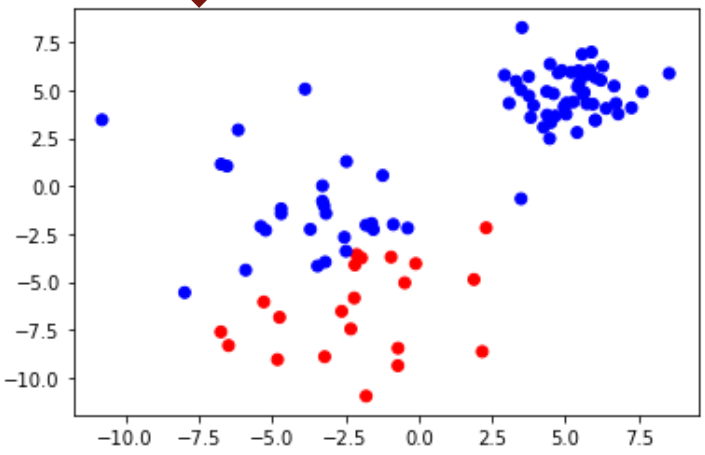
c: クラス
（スパムかどうか）
x: 単語の有無

繰り返し1回目（ナイーブベイズクラスタリング）

初期化の結果




```
clf = GaussianNB()
clf.fit(X, Y)
newY=clf.predict(X)
```



文書のクラスを
ランダムに割り当てる

ナイーブベイズ識別器

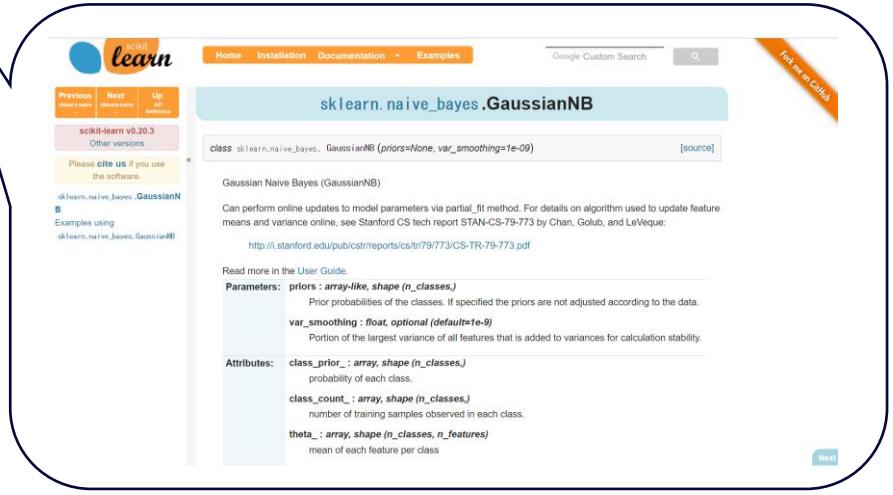
クラス分けされた文書

$P(X = x|C = c)$を計算する

文書のクラスの確率を計算できる

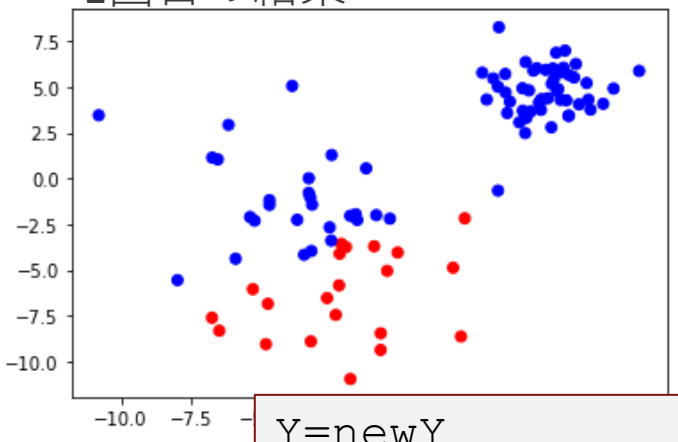
$P(C = c|X = x)$

再度、文書のクラス
を割り当てなおす



繰り返し2回目（ナイーブベイズクラスタリング）

1回目の結果




```
Y=newY
clf.fit(X, Y)
newY=clf.predict(X)
```

ナイーブベイズ識別器

クラス分けされた文書

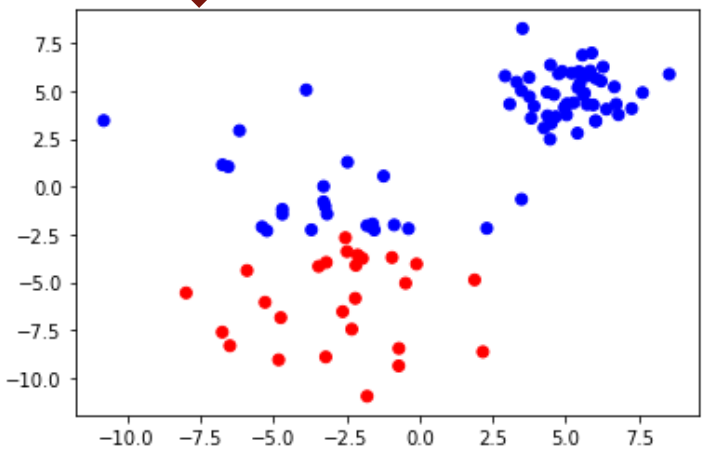
$P(X = x|C = c)$を計算する

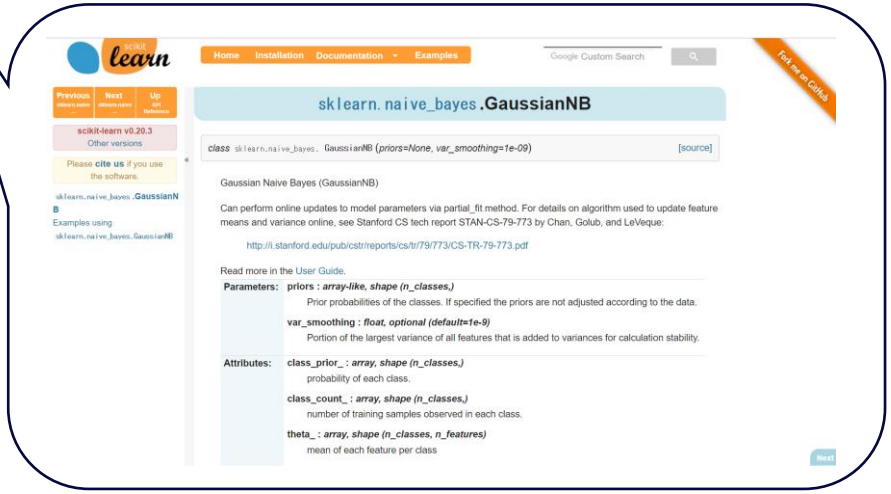
文書のクラスの確率を計算できる

$P(C = c|X = x)$

文書のクラスを
ランダムに割り当てる

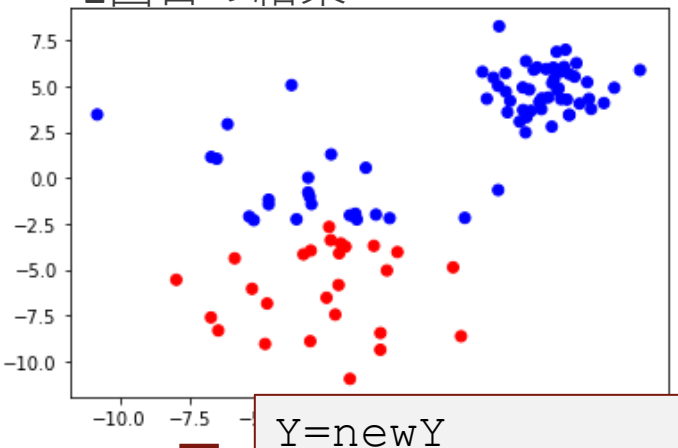
再度、文書のクラス
を割り当てなおす



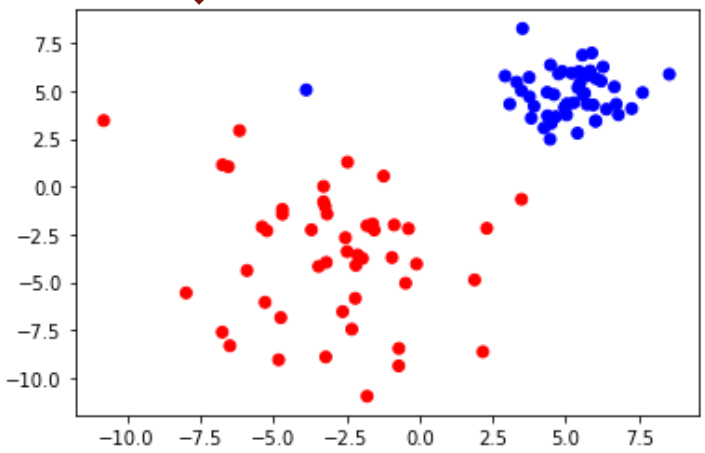


繰り返し3回目（ナイーブベイズクラスタリング）

2回目の結果




```
Y=newY
clf.fit(X, Y)
newY=clf.predict(X)
```



文書のクラスを
ランダムに割り当てる

ナイーブベイズ識別器

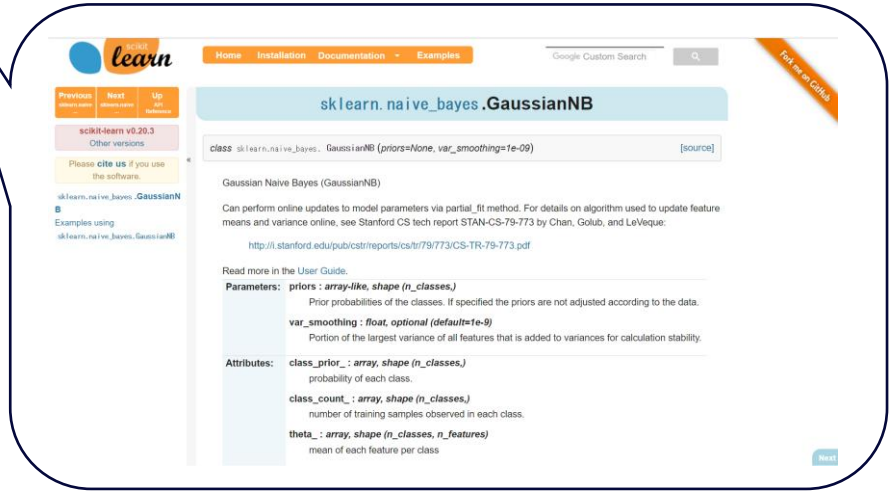
クラス分けされた文書

$P(X = x|C = c)$を計算する

文書のクラスの確率を計算できる

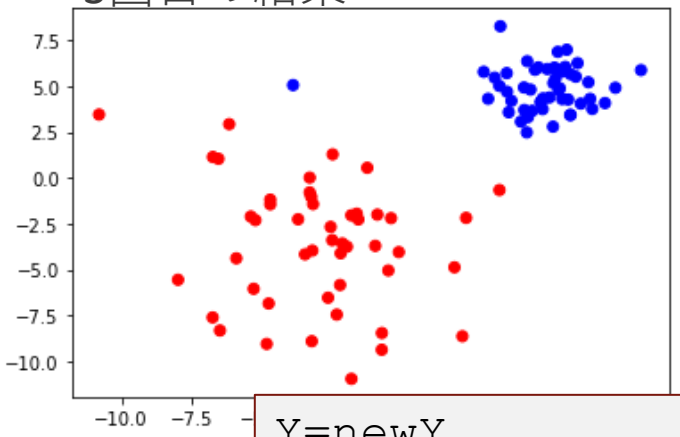
$P(C = c|X = x)$

再度、文書のクラス
を割り当てなおす

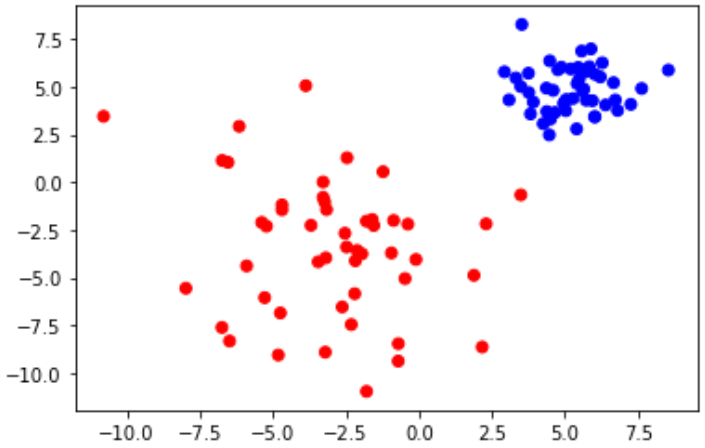


繰り返し4回目（ナイーブベイズクラスタリング）

3回目の結果




```
Y=newY
clf.fit(X, Y)
newY=clf.predict(X)
```



文書のクラスを
ランダムに割り当てる

ナイーブベイズ識別器

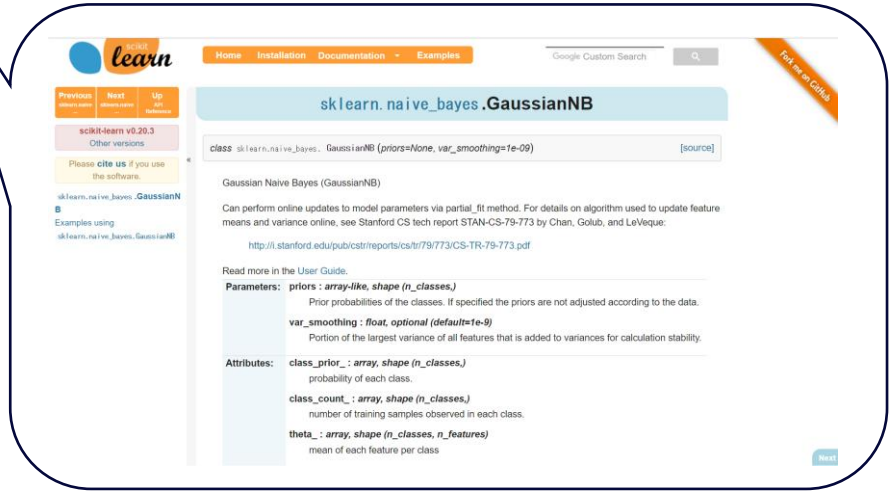
クラス分けされた文書

$P(X = x|C = c)$を計算する

文書のクラスの確率を計算できる

$P(C = c|X = x)$

再度、文書のクラス
を割り当てなおす



確率モデルの種類

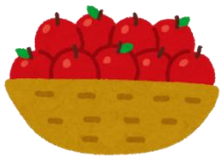
確率モデル：確率的な事象を含む数理モデル

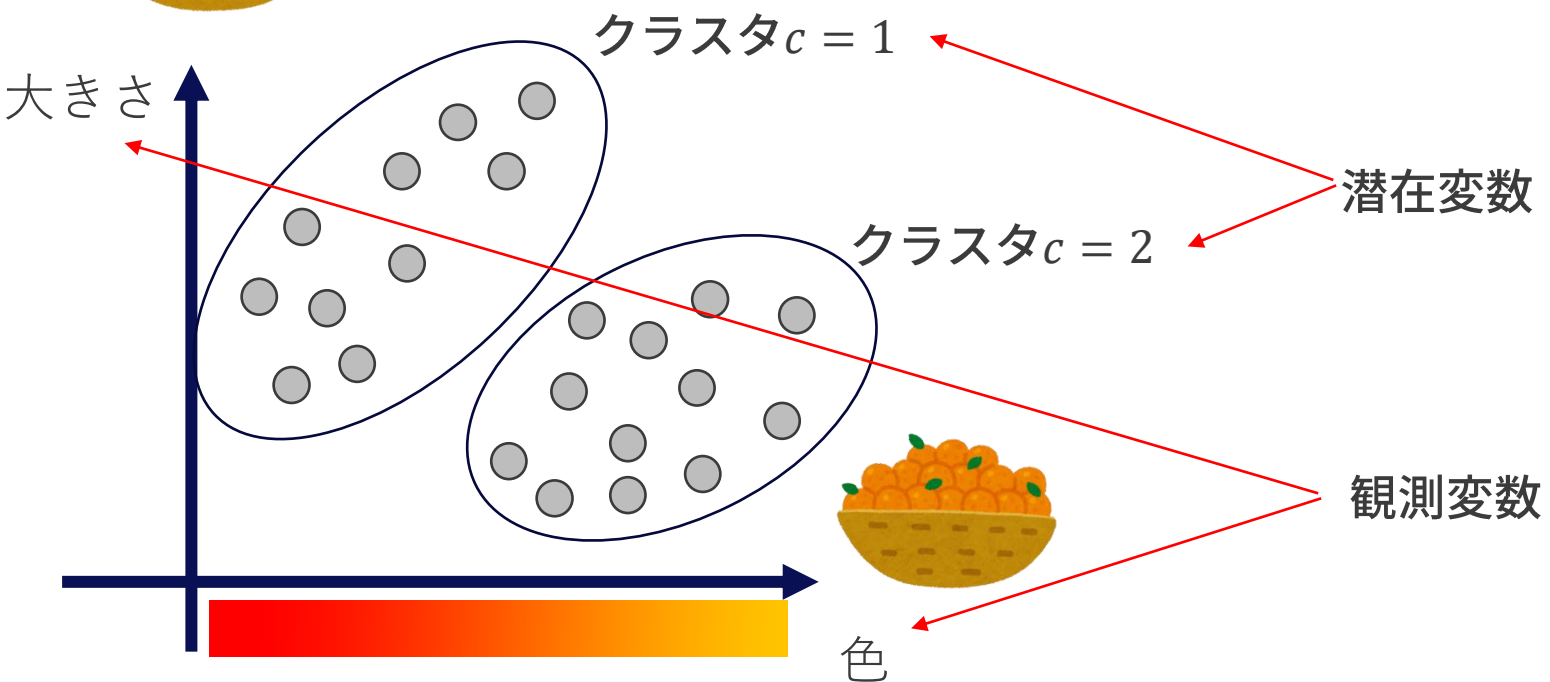
観測変数：観測できる確率変数
e.g.（常に与えられる）データ

潜在変数：観測できない確率変数
（潜在的には存在しているとみなす変数）
e.g. クラスタ，　識別対象

スパムメールの例：
メールに出てくる単語の有無
を表す変数

スパムメールの例：
スパムメールかどうか？





生成モデルと識別モデル

潜在変数に関して確率モデルは以下の二つのモデルに分けられる

観測変数：x
潜在変数：z

生成モデル

- 同時確率をモデル化
  「$p(x,z) = \ \ldots$ 」
- $p(z|x)$ と $p(x|z)$ が相互に計算可能
- クラスタリングや識別・予測など幅広く使われる

例
- ナイーブベイズ
- 混合ガウスモデル
- ベイジアンネットワーク
- カルマンフィルタ

識別モデル

- 条件付き確率をモデル化
  「$p(z|x) = \ \ldots$ 」
- $p(z|x)$のみ計算可能（$p(x|z)$はできるとは限らない）
- 識別・予測に主に用いられる

例
- ロジスティック回帰
- 出力を確率とみなしたニューラルネット
（多層パーセプトロン）

（再訪）ナイーブベイズモデル

スパムメールフィルタ：特定の語がメールの文中に含まれているかどうかでスパムメールを識別する問題

スパムメールを集めてきて、その中の単語の数を数えると計算できる

（キーワードが含まれるスパムメールの数/全スパムメールの数）

$P(\text{あなただけに} = 1 | \text{スパム} = 1)$
$P(\text{今だけ} = 1 | \text{スパム} = 1)$
$P(\text{【重要】} = 1 | \text{スパム} = 1)$
……

スパムでないメールを集めてきて、その中の単語の数を数えると計算できる

$P(\text{あなただけに} = 1 | \text{スパム} = 0)$
$P(\text{今だけ} = 1 | \text{スパム} = 0)$
$P(\text{【重要】} = 1 | \text{スパム} = 0)$
……

ランダムにメールを集めてきて、その中にどれだけスパムメールがあるかを数えると計算できる

$P(\text{スパム} = 0)$　　$P(\text{スパム} = 1)$

同時分布が定義されている

$$P(x|c)P(c) = P(x, c)$$

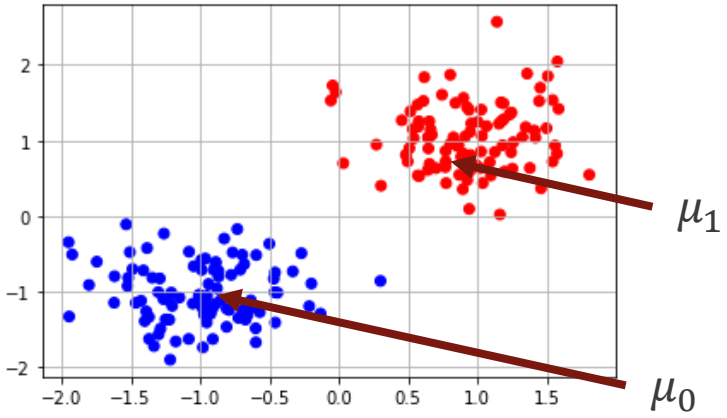
潜在変数 c: クラスタ（スパムかどうか）
観測変数 x: ある単語の有無

クラスタリングでよく使われる手法：混合ガウスモデルの雑な説明

混合ガウスモデル(Gaussian mixture model: GMM)：$p(x|c)\ P(c)$

$N(x|\mu_c, \sigma_c^2)$

$P(c=0)=\pi_0$
$P(c=1)=\pi_1$



同時確率を定義しているので生成モデル

単純な分布が足し合わさっている形式のモデルを「混合モデル」と呼び、
各種クラスタリングでよく利用される

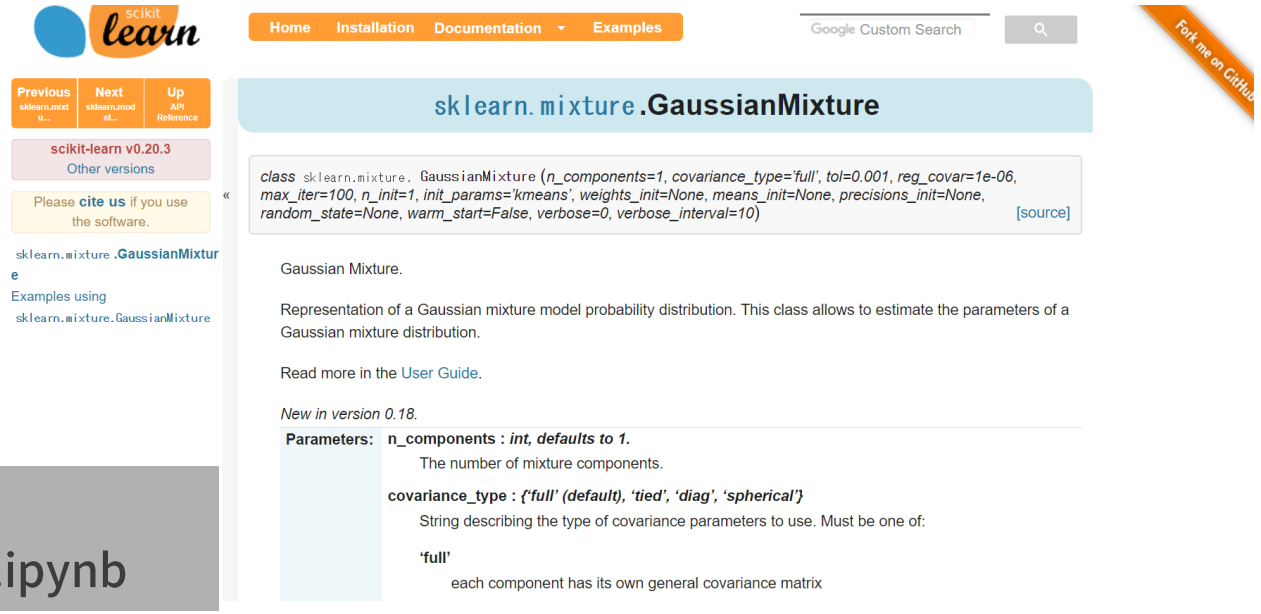
ナイーブベイズとの違いは観測がガウス分布（連続値）

（また通常個々のガウス分布は変数間の相関を持っている、相関を持たない場合を考えることもある：共分散行列のパラメータの数は入力の次元数の２乗のオーダーなのでデータ数が少ない場合は難しいため）

ナイーブベイズと類似の繰り返し計算でクラスタリングを行う。
（こういったアルゴリズムをExpectation-Maximization (EM)アルゴリズムというが、詳細は後述）

課題

K-means 法でクラスタリングを行うプログラムを実行し、
混合ガウス分布でクラスタリングを行うプログラムに変更してみましょう
余裕がある人はplotする軸を変更してどんな感じで分離されるかチューニングしてみましょう



参照

lecture_clustering.ipynb

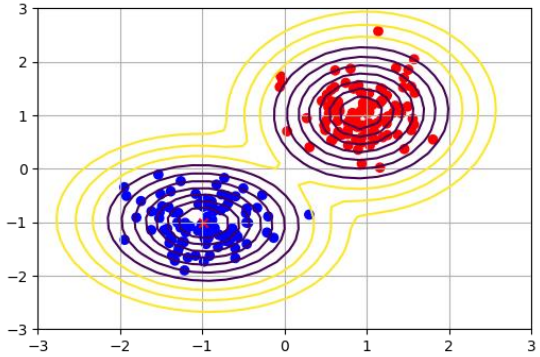
```
from sklearn.mixture import GaussianMixture
model = GaussianMixture(n_components=2)
```

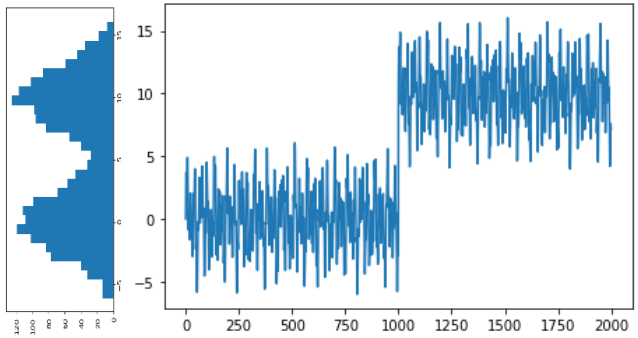
混合ガウスモデルの用途

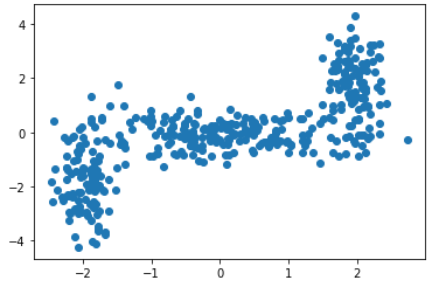
混合ガウスモデル(Gaussian mixture model: GMM)：$p(x|c)\ P(c)$

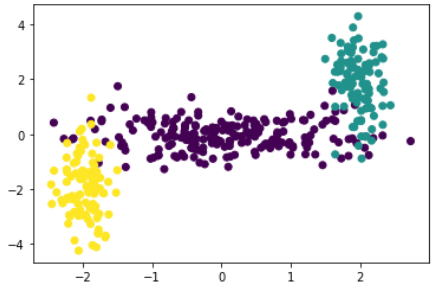
$N(x|\mu_c, \sigma_c^2)$

$P(c=0)=\pi_0$
$P(c=1)=\pi_1$









各種確率が計算できるので、

クラスタリングと類似の用途

⇒量子化・離散化

教師ありと教師なしの両方が計算できる

⇒半教師ありにも拡張可能（一部にラベル）

確率計算

⇒外れ値の検出・異常値検出・変化点の検出

よくわからない分布の近似

⇒類似データどうかの判定

混合ガウスモデル(Gaussian mixture model: GMM)のExpectation-Maximization (EM)アルゴリズム

混合ガウスモデル(Gaussian mixture model: GMM)：$p(x|c)\ P(c)$

$N(x|\mu_c, \sigma_c^2)$

$P(c=0)=\pi_0$
$P(c=1)=\pi_1$

最尤法：尤度を最大化するようにパラメータ（平均・分散など）を推定したい（シンプルな場合の最尤法については第４回確率統計を参照）

尤度：$p(\boldsymbol{X};\boldsymbol{\mu},\boldsymbol{\sigma}^2,\boldsymbol{\pi})=\sum_Z p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2,\boldsymbol{\pi})$

z_{ic}:データiがcに属するかどうか

潜在変数$\boldsymbol{Z}$を周辺化しており単純に最大化するのは難しい

→ $\log p(\boldsymbol{X};\boldsymbol{\mu},\boldsymbol{\sigma}^2) \geq Q$ となるQを最大化する　pの最大化と$\log p$の最大化は等価

以下の操作を繰り返すと尤度を最大化できる

$$Q = E_{\boldsymbol{Z}|\boldsymbol{X}}[\log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2,\boldsymbol{\pi})]$$

繰り返し

$$\mu_c \leftarrow \mathrm{argmax}_{\mu_c} Q$$
$$\sigma_c^2 \leftarrow \mathrm{argmax}_{\sigma_c^2} Q$$
$$\pi_c \leftarrow \mathrm{argmax}_{\pi_c} Q$$

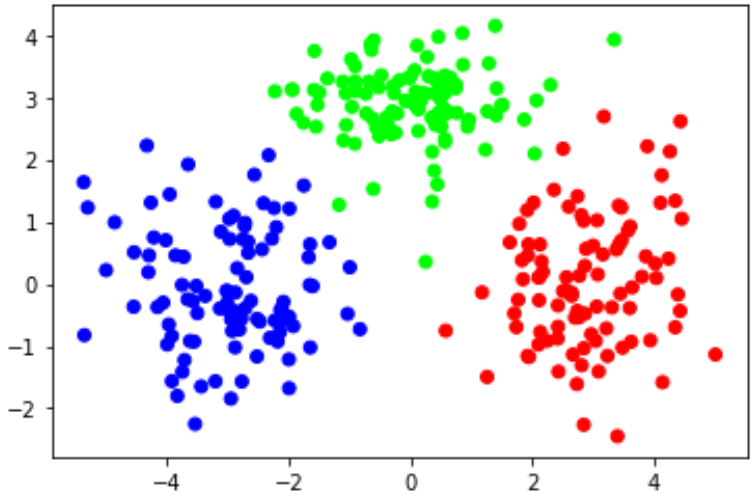
期待値の定義
$E_{Z|X}[Y]=\int p(Z|X)Y dZ$

π_cはクラスcに所属する確率

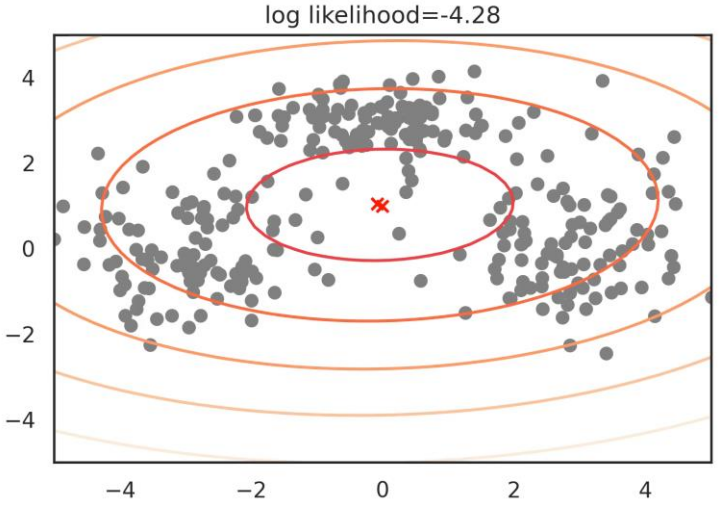
期待値計算と最大化計算の繰り返しになるので Expectation-Maximization (EM)アルゴリズムと呼ばれる

混合ガウスモデルによるクラスタリングの振る舞い

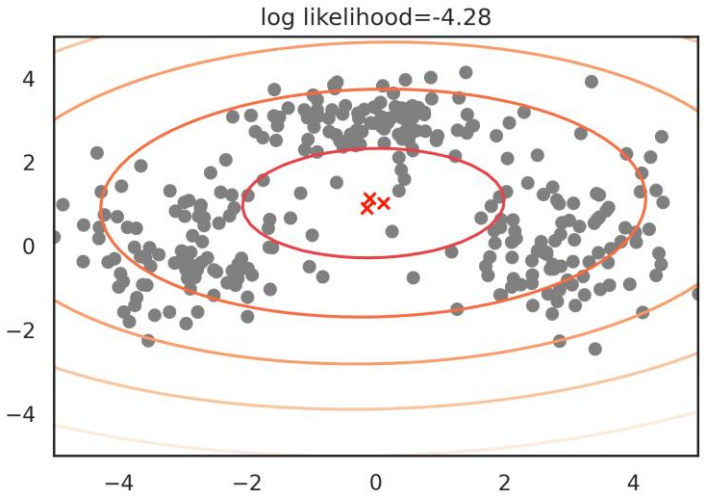
元データ



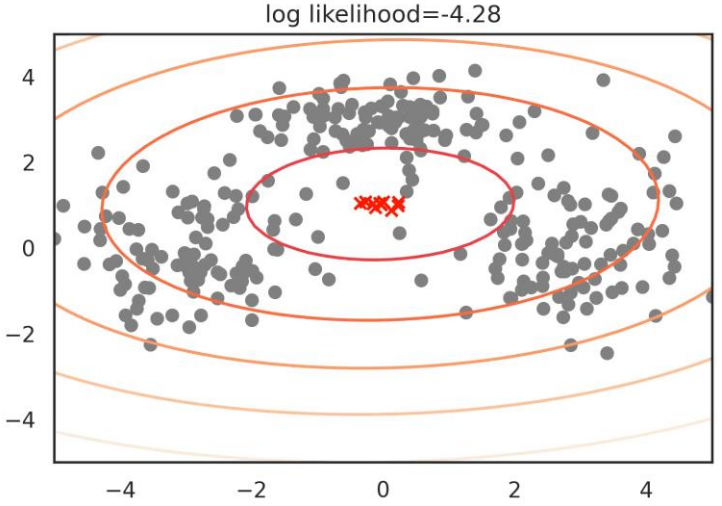
クラスタ数：2



クラスタ数：3



クラスタ数：10



尤度とQ関数に実際の混合ガウスモデルの式を入れた場合の計算

$\log p(\boldsymbol{X};\boldsymbol{\mu},\boldsymbol{\sigma}^2) \geq Q$となる$Q = E_{\boldsymbol{Z}|\boldsymbol{X}}[\log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)]$を最大化する

データとクラスタに関する確率変数を導入する

z_{ic}:データiがクラスcに属しているときに１それ以外の時０の確率変数

$$p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2) = \prod_i \prod_c \pi_c^{z_{ic}} p(x_i|c;\mu_c,\sigma_c^2)^{z_{ic}}$$

πはクラスに所属する確率

$P(c=0) = \pi_0$

$P(c=1) = \pi_1$

$\pi = (\pi_0, \pi_1)$

$$Q = E_{\boldsymbol{Z}|\boldsymbol{X}}[\log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)]$$
$$= \sum_c \sum_i \gamma_{ic} \log p(x_i|c;\mu_c,\sigma_c^2)\, \pi_c$$
$$= \sum_c \sum_i (\gamma_{ic} \log p(x_i|c;\mu_c,\sigma_c^2) + \gamma_{ic} \log \pi_c)$$

尤度との関係は以下

$\mathrm{p}(\boldsymbol{X};\mu,\sigma^2) = \sum_{\mathrm{Z}} \mathrm{p}(\boldsymbol{X},\boldsymbol{Z};\mu,\sigma^2)$

$\mu_c \leftarrow \mathrm{argmax}_{\mu_c} Q$
$\sigma_c^2 \leftarrow \mathrm{argmax}_{\sigma_c^2} Q$
$\pi_c \leftarrow \mathrm{argmax}_{\pi_c} Q$

$$\mu_c \leftarrow \frac{\sum_i \gamma_{ic} x_i}{N_c}$$
$$\sigma_c^2 \leftarrow \frac{\sum_i \gamma_{ic}(x_i-\mu_c)(x_i-\mu_c)^T}{N_c}$$
$$\pi_c \leftarrow \frac{N_C}{\sum_{c'} N_{c'}}$$

$$\gamma_{ic} = E_{z|x_i}[z_{ic}] = \frac{\pi_c N(x_i|\mu_c,\sigma_c^2)}{\sum_{c'} \pi_{c'} N(x_i|\mu_{c'},\sigma_{c'}^2)}$$

$$N_c = \sum_i \gamma_{ic}$$

EMアルゴリズムの特性

- 必ず尤度は増加する（非減少）
- 局所最適解しか得られないため、初期値を変えてリスタートが必要
- 学習のために調整すべきパラメータが少ない

そのまま実装すると、いくつかの問題がある
- 尤度が無限大になる場合がある
  （クラスタにデータが一点しかない場合、分散０のとき尤度無限大）
- 共分散行列の逆行列計算に失敗する
  （クラスタ内のデータが一直線に並んでしまったりする場合）

実際にはEMアルゴリズムを使いたい状況では、
少し変更を加えたMAP-EMアルゴリズムと呼ばれる拡張を用いる

混合ガウスモデル(Gaussian mixture model: GMM)のHARD-EM（Viterbi trainingとも）

EMアルゴリズム
尤度を最大化

$$p(\boldsymbol{X};\boldsymbol{\mu},\boldsymbol{\sigma}^2) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$$

HARD-EMアルゴリズム
同時確率を最大化

$$p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$$

$\boldsymbol{Z}$の要素 z_{ic} はデータiがクラスc に属しているときに１それ以外の時０の確率変数

π_cはクラスcに所属する確率

以下の操作を繰り返す

1. $Q = E_{\boldsymbol{Z}|\boldsymbol{X}}[\log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)\,]$
2. $\mu_c \leftarrow \mathrm{argmax}_{\mu_c}\, Q$
   $\sigma_c^2 \leftarrow \mathrm{argmax}_{\sigma_c^2}\, Q$
   $\pi_c \leftarrow \mathrm{argmax}_{\pi_c}\, Q$

以下の操作を繰り返す

1. $\boldsymbol{Z} \leftarrow \mathrm{argmax}_{\boldsymbol{Z}} \log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$
2. $\mu_c \leftarrow \mathrm{argmax}_{\mu_c} \log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$
   $\sigma_c^2 \leftarrow \mathrm{argmax}_{\sigma_c^2} \log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$
   $\pi_c \leftarrow \mathrm{argmax}_{\pi_c} \log p(\boldsymbol{X},\boldsymbol{Z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)$

混合ガウスモデルのHard-EM計算

$$p(\boldsymbol{X}, \boldsymbol{Z}; \boldsymbol{\mu}, \sigma^2) = \prod_i \prod_c \pi_c^{z_{ic}} p(x_i|c; \mu_c, \sigma_c^2)^{z_{ic}}$$ を最大化

分散とクラスタの所属確率は以下で固定
$\sigma_c^2 = 1$
$\pi_c = 1/|\text{クラスタ数}|$

1. $\boldsymbol{Z} \leftarrow \text{argmax}_{\boldsymbol{Z}} \log p(\boldsymbol{X}, \boldsymbol{Z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2)$

$z_{i\cdot} \leftarrow \text{argmax}_{z_{i\cdot}} \log p(x_i|c; \mu_c, \sigma_c^2)^{z_{ic}}$

z_{ic}:データiがクラスタcに属しているときに１それ以外の時０

$\log p(x_i|c; \mu_c, \sigma_c^2) \propto -(x_i - \mu_c)^2$が最大の$c$のみ$z_{ic} = 1$でそれ以外の$c$は$z_{ic} = 0$

$$z_{ic} = 1, \quad c \leftarrow \text{argmin}_c (x_i - \mu_c)^2$$

データx_iのクラスタをμ_cが最も近いクラスに割り当てる

2. $\mu_c \leftarrow \text{argmax}_{\mu_c} \log p(\boldsymbol{X}, \boldsymbol{Z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2)$

$$\mu_c \leftarrow \text{argmax}_{\mu_c} \sum_i z_{ic} \log p(x_i|c; \mu_c, \sigma_c^2)$$

クラスタ内のデータ集合をCとすると

$$\mu_c \leftarrow \text{argmin}_{\mu_c} \sum_{x_i \in C} (x_i - \mu_c)^2$$

クラスタ内のデータ集合の重心をμ_cとする

k-means法
=特殊な混合ガウスモデルのHard-EMアルゴリズム

k-meansの疑問点（第一回「機械学習」講義より）

この繰り返しは必ず止まるのか？

- 同時確率を最大化しているので、
  （上限が存在すれば）必ず収束・停止する
- 経験上、EMアルゴリズムよりも早く収束・停止する

各クラスタのサイズや形を変えたい場合はどうするのか？

分布を変えて同様のアルゴリズムを構築

- ガウス分布の分散（クラスタの大きさ）を変える
- ガウス分布以外の分布を利用する

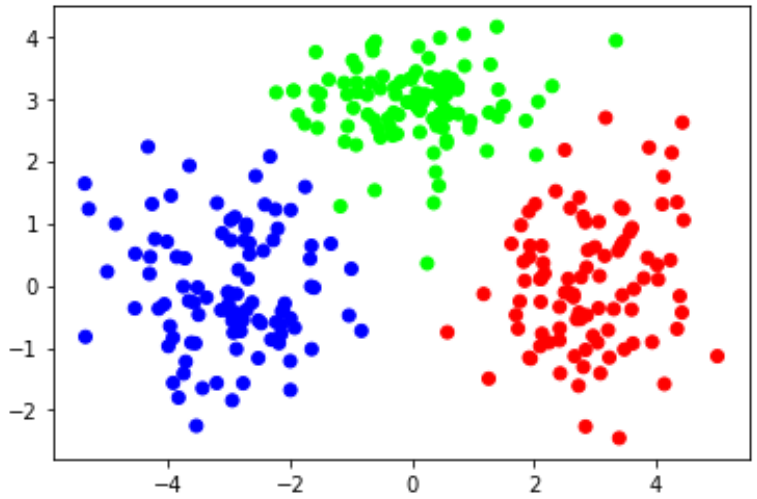
クラスタの数kをどうやって決めるのか？

次以降で説明する「モデル選択」を使用できる

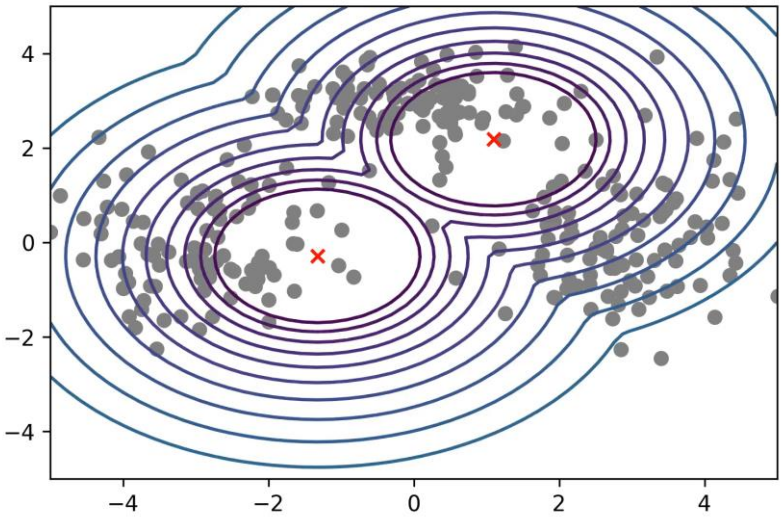
これにより、クラスタ数を決定することができる

k-means法によるクラスタリングの振る舞い

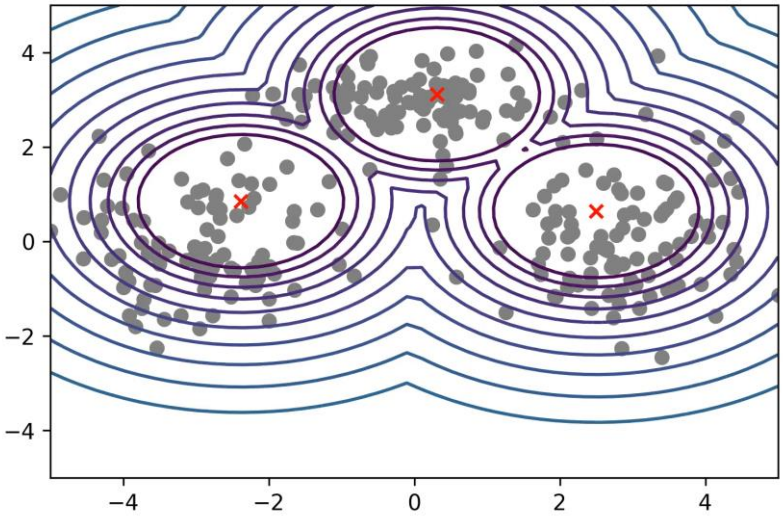
元データ



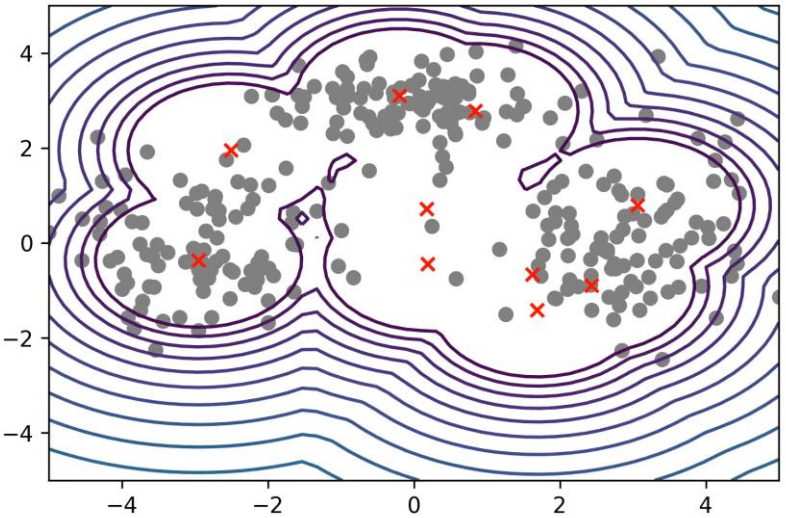
クラスタ数：2



クラスタ数：3



クラスタ数：10

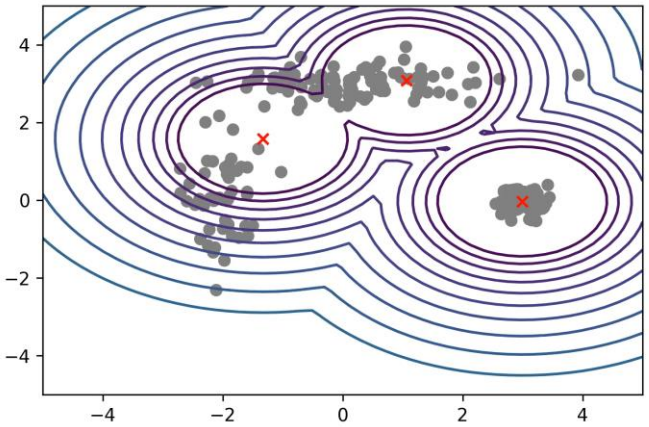


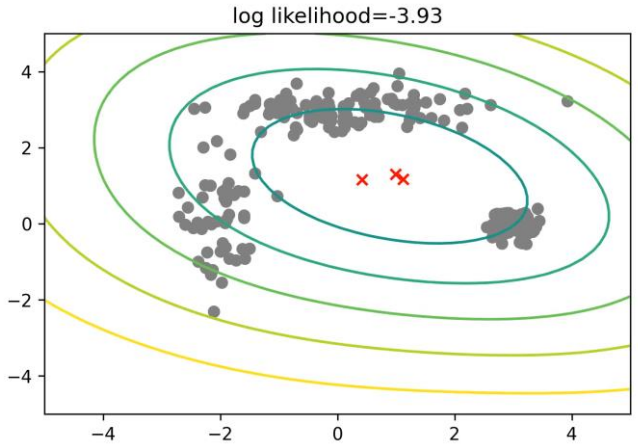
K-means(GMM + Hard-EM) vs GMM+EM

K-means
(GMM + Hard-EM)

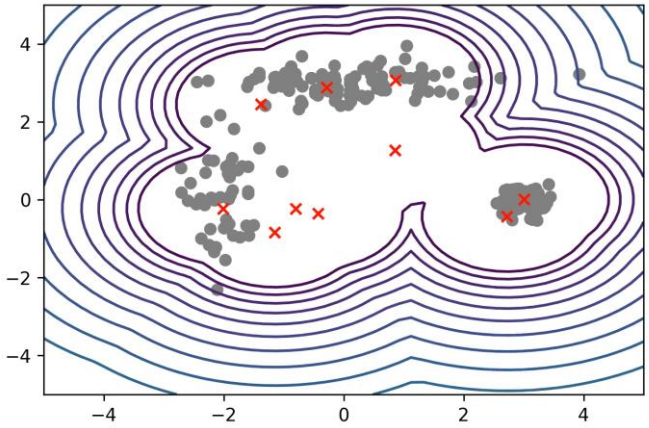
GMM+EM

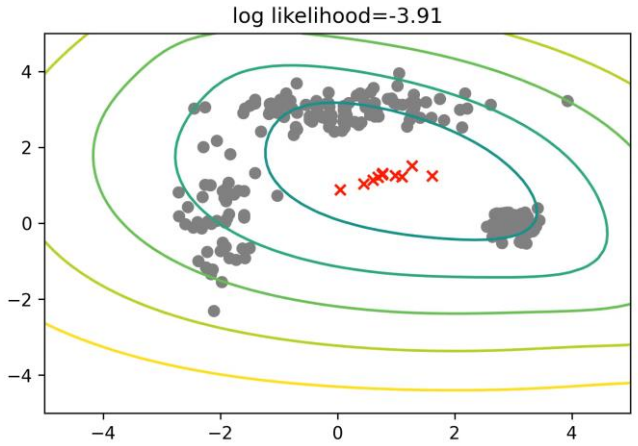
クラスタ数=3





クラスタ数=10





確率モデル前半まとめ

- 尤度を最大化することでパラメータを推定できる（最尤法）

- 潜在変数があるモデルではＥＭアルゴリズムを用いることで最尤法を実行できる
  ＥＭアルゴリズムの亜種：MAP-EM, Hard-EM, …

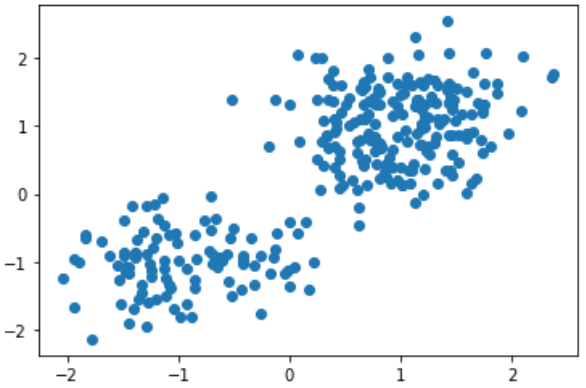
- 特に混合ガウスモデルが有名でよく使われる

- K-means法は混合ガウスモデルと関連が深い

モデル選択：どのモデルが最もいいモデル？

モデル選択の問題として解くことができる問題

- クラスタ数を決める問題
- 特徴量を決める問題
- モデルのハイパーパラメータを決める問題



モデル選択には汎化誤差（未知のデータに対する誤差）を用いたい

（第２回：機械学習講義）

→実データでは計算できないため、代替となる誤差・基準を用いる

確率モデルの場合は誤差として「負の対数尤度(negative log likelihood)」を用いる

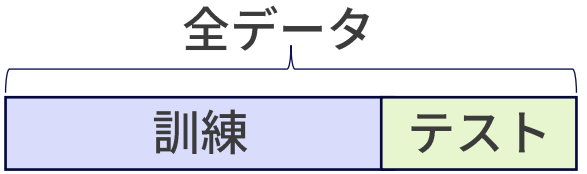
（負の対数尤度の最小化＝尤度の最大化）

交差検証

未知のデータは用意できないので、手元にあるデータを分割し、
一方を訓練データ、もう一方をテストデータとして誤差の評価を行う

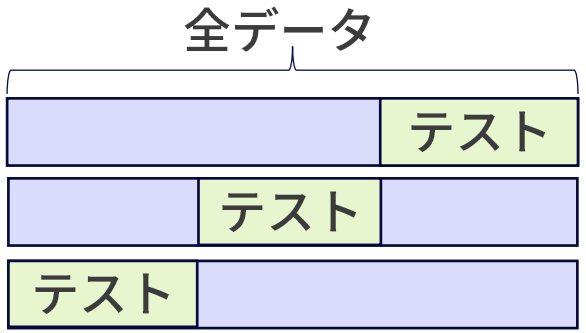
ホールドアウト検証

- 全データセットを（無作為に）テストデータと訓練データに分け、テストデータの誤差で評価をする



K-分割交差検証（k-fold cross-validation）

- 全データセットをK群に分割し、1群をテストとし、残りを訓練データとする。をK通り変更し、その誤差（の平均）テストとする群で評価をする



Leave-one-out交差検証（Leave-one-out cross-validation）

- K-分割交差検証の K を標本サイズにした場合と同じ
- 全データセットから一つの事例をテストとし、残りを訓練とする。全事例が一回ずつテストとなるよう繰り返す。

利点：簡単にできる、様々な亜種があり、直観的に使いやすい
欠点：交差検証は時間がかかる

情報量規準

AIC (Akaike information criteria)：

モデルの分布と経験分布の差を評価することで導出された規準

$$-2\log L + 2k$$

L: 尤度
k: パラメータの数
（クラスタの数が増えるとパラメータ数が増える）
n: データ数

BIC (Bayes information criteria)：

モデルの事後分布を近似して導出された規準

$$-2\log L + k\log n$$

- これらの情報量基準は潜在変数が存在しないような状況（正則モデル）を仮定しており、そのような状況でも数値的には計算できるが、本来の意味は失われてしまう。
- 混合ガウス分布ではクラスタを表す変数は潜在変数なので、これらの基準を使うことは正しくない
  ただし、実用上はそれっぽいクラスタ数を簡単に見積もるのに使うことはある
- 計算は非常に簡単なことが多い

例：多項式フィッティングにおけるモデル選択

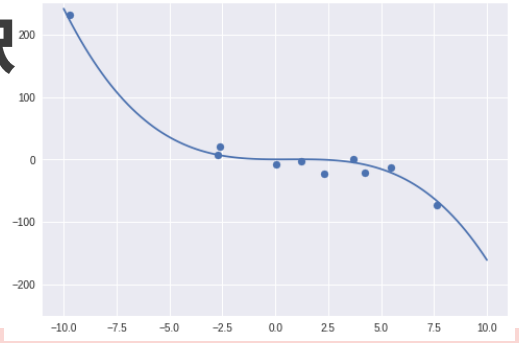
正解の曲線

$y = -0.2x^3 + 0.4x^2 - 0.1x + 0.1$

データの生成

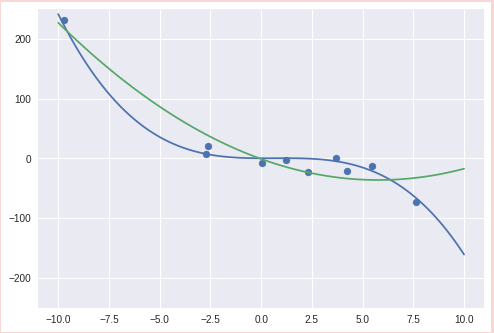
$t \sim y + \epsilon$

ϵ:ノイズ

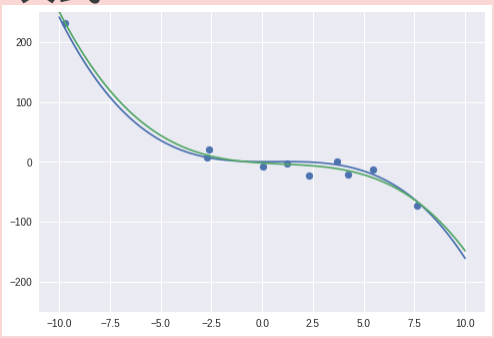


(x, t)のデータペアの集合から多項式の係数を決定する（緑の線が推定、青が正解）

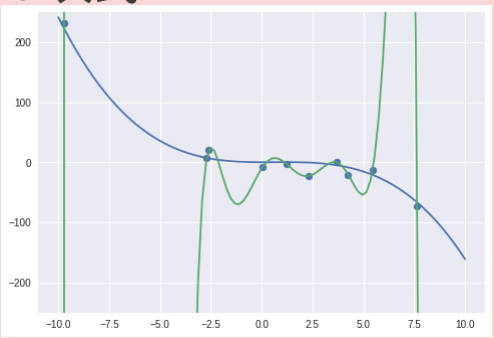
２次式



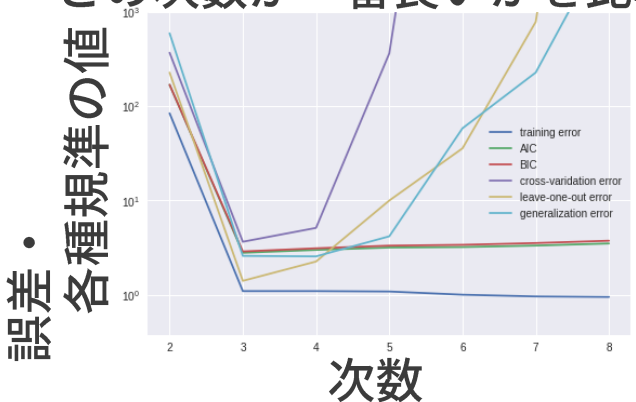
３次式



９次式



どの次数が一番良いかを比較する



Training errorを除くどの指標も汎化誤差の最も小さく、正しい次数の3次を選択できている

例：多項式フィッティングにおけるモデル選択

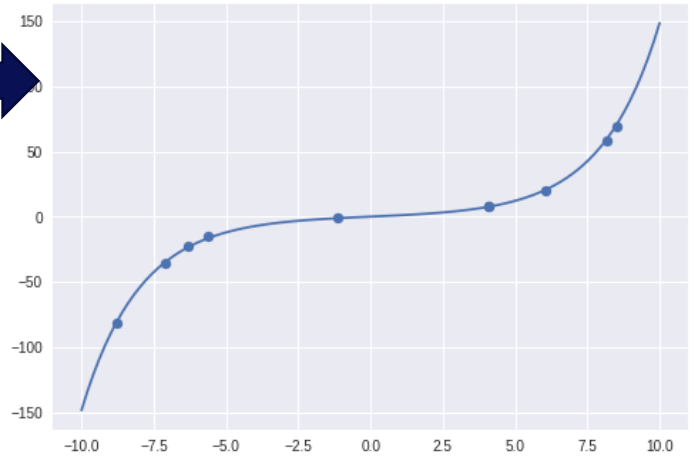
正解の曲線

$$y = \exp(x/2) - \exp(-x/2)$$

データの生成

$t \sim y + \epsilon$

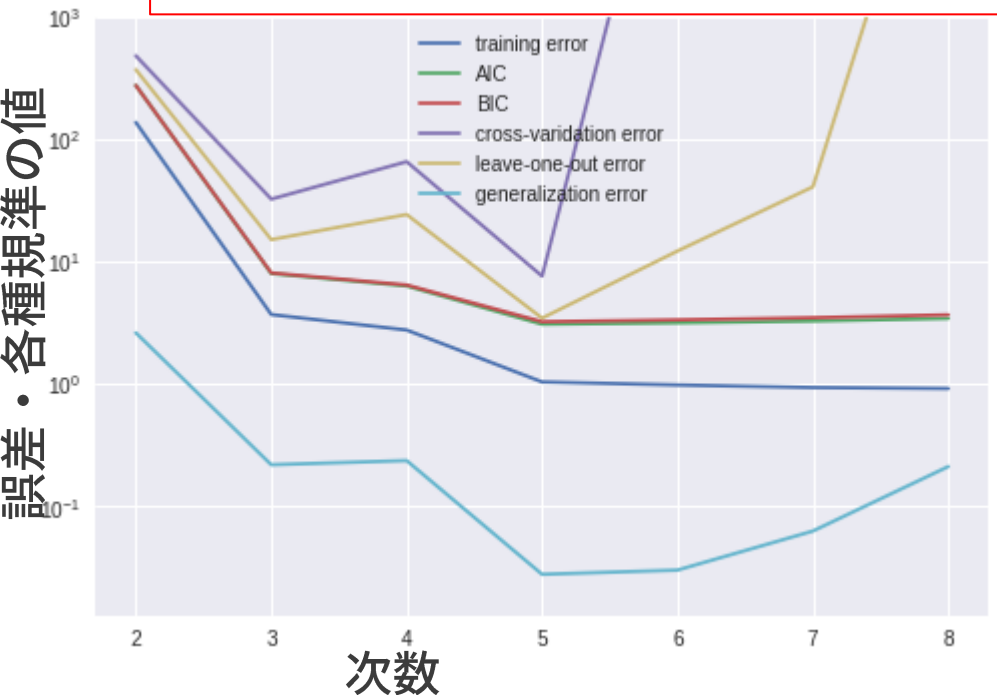
ϵ:ノイズ



(x, t)のデータペアの集合から多項式の係数を決定する

多項式の次数をどのように選んでも実は正解の曲線にはたどり着けない

（現実問題ではこういった問題の方が多いと考えられる）

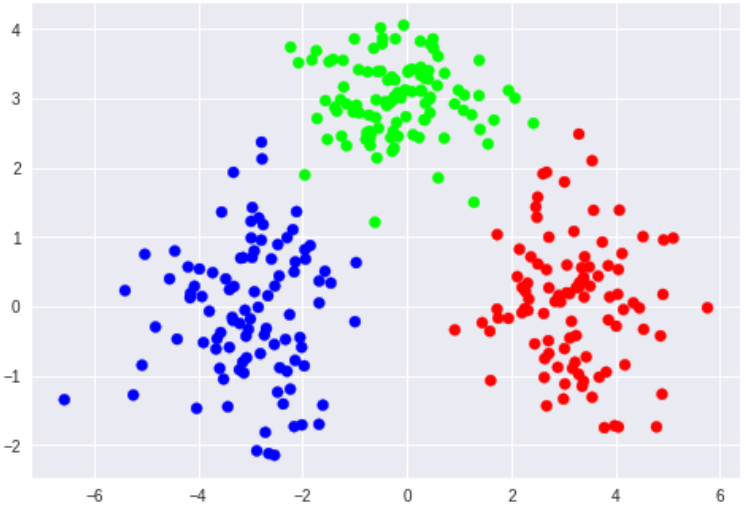


- Training errorを除くどの指標も汎化誤差の最も小さい５次を選択できている
- ３と５で小さくなっているのは奇関数の成分が効いているためと考えられる

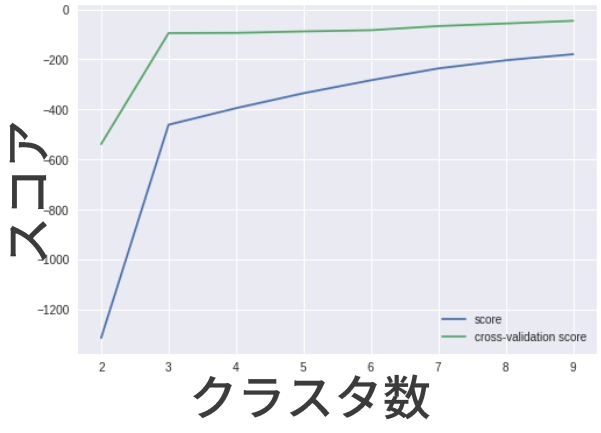
例：クラスタリングにおけるモデル選択

K-means 法によるクラスタリングのクラスタ数の決定

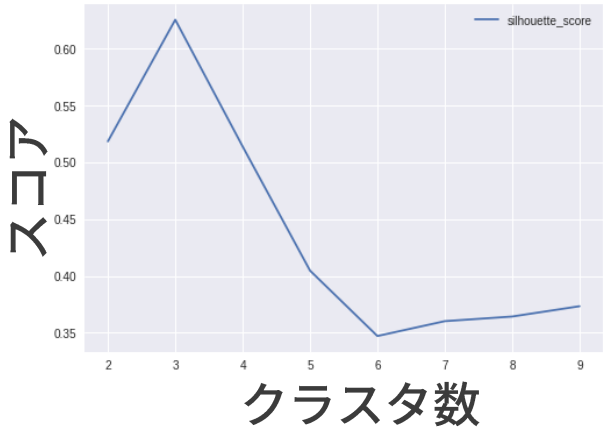
３つのガウス分布から生成したデータ→真のクラスタ数は３の時



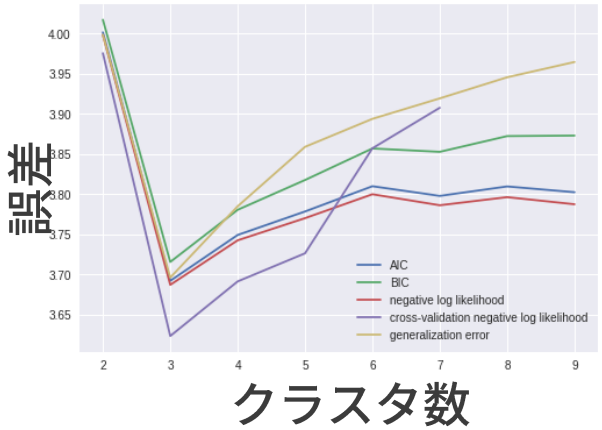
K-means 法の学習スコア（高い方がよい）



シルエット法のスコア（クラスタ間の分離度とクラスタ内の凝集度のスコア：高い方がよい）

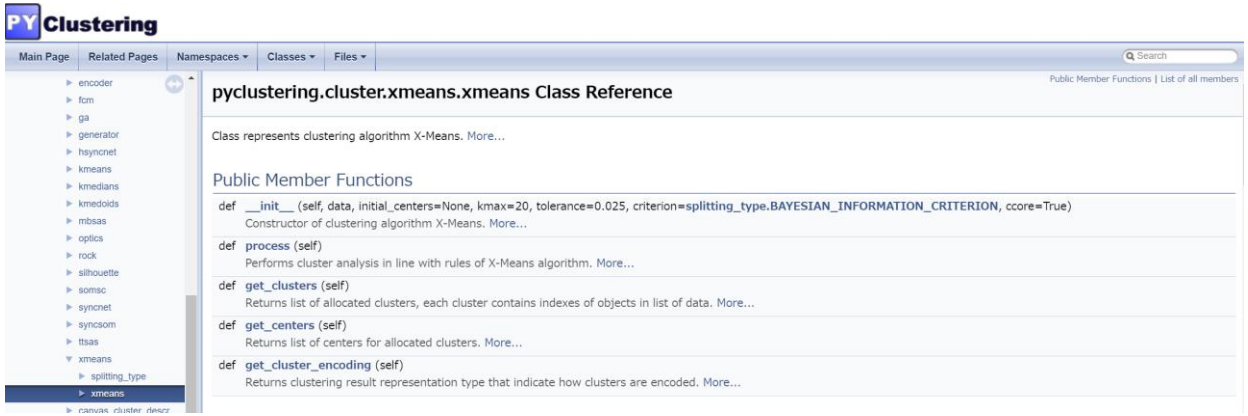


各種誤差・規準



確率モデル後半まとめ

- モデルの決定・クラスタ数の決定にはモデル選択の方法が利用できる
  交差検定や情報量規準が利用可能
- 自動決定のライブラリも内部ではこのような規準を使うことが多い
  X-means: k-meansとBICの組み合わせ



関連分野

人工知能

計算と人工知能

1950年： チューリングテスト

19[illegible]トマス会議

機械学習

論理・探索

第1次AIブーム: 推論と探索
1956年－1974年

1972年： Prolog

第2次AIブーム:
知識
（エキスパートシステム）
1980年－1987年

機械学習

1967年：k-means法
(教師なし学習：
クラスタリング)

1989年：Q学習
(強化学習)

1992年：非線形SVM
(教師あり学習)

1990年～：統計的機械学習
- 確率モデル
- 汎化誤差理論
- ベイズモデル

ニューラルネットワーク

1958年：パーセプトロン

1969年：パーセプトロンの限界

1986年：誤差逆伝搬法

深層学習

2006年：オートエンコーダー

第3次AIブーム:
深層学習
2006年－

関連分野

人工知能

計算と人工知能

1950年： チューリングテスト

19[illegible]トマス会議

機械学習

論理・探索

第1次AIブーム: 推論と探索
1956年－1974年

1972年： Prolog

第2次AIブーム:
知識
（エキスパートシステム）
1980年－1987年

機械学習

1967年：k-means法
(教師なし学習：
クラスタリング)

1989年：Q学習
(強化学習)

1992年：非線形SVM
(教師あり学習)

1990年～：統計的機械学習
- 確率モデル
- 汎化誤差理論
- ベイズモデル

ニューラルネットワーク

1958年：パーセプトロン

1969年：パーセプトロンの限界

1986年：誤差逆伝搬法

深層学習

2006年：オートエンコーダー

第3次AIブーム:
深層学習
2006年－

ベイズ推定

ベイズ推定を使う利点：

- データ数が少ない場合にもよく動作する
- クラスタ数や多項式フィッティングの次数が多い場合にも過学習を起こさない（応用上は少し多めに取っておくといったことも可能）
- クラスタ数や次数の推定もできる

考え方

データが与えられたとき、求めたい確率分布（予測・クラスタ）を直接推定する

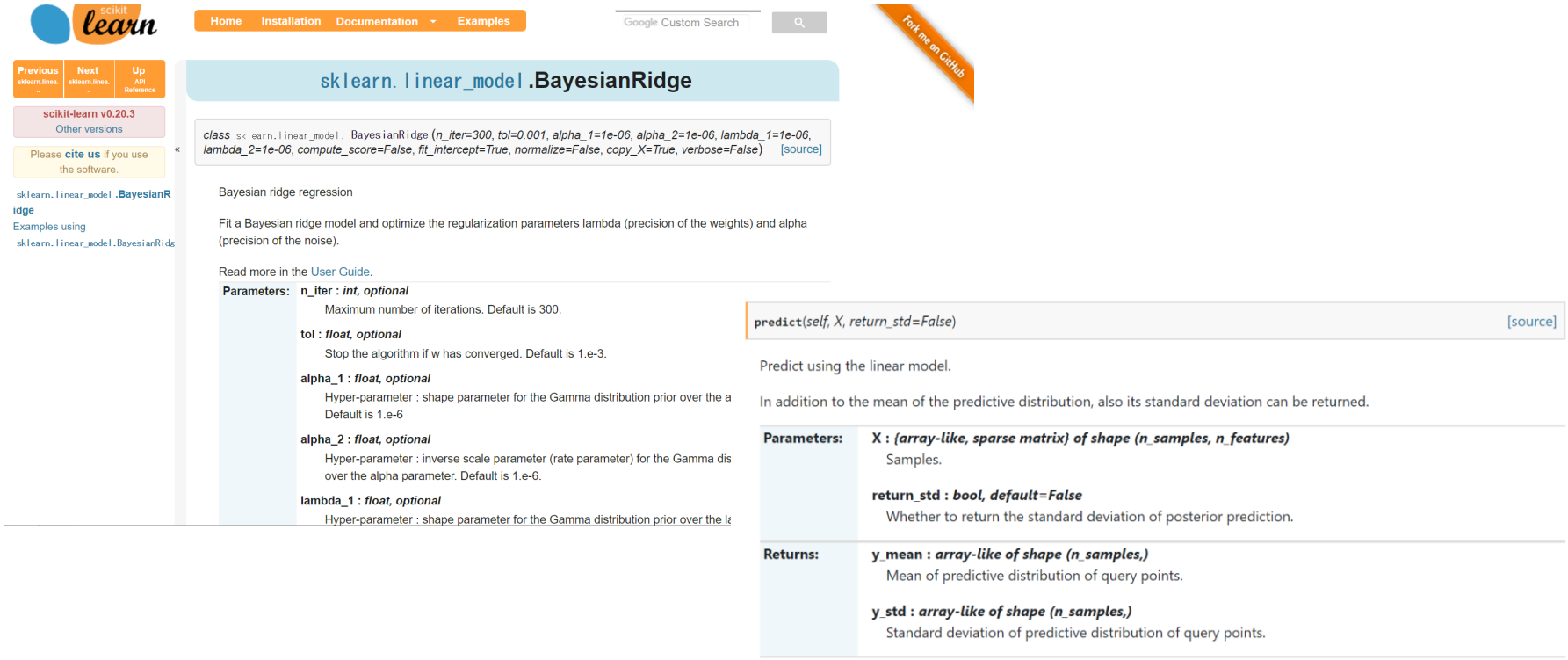
つまり、$p(Y|X)$を計算する

X: データ

Y: 求めたいもの

注：モデルは値の決まったパラメータを持たないし、訓練済みモデルといった概念も存在しない

ベイズ線形回帰



scikit learn

Home Installation Documentation Examples

Google Custom Search

Fork me on GitHub

Previous sklearn.linea... | Next sklearn.linea... | Up API Reference

scikit-learn v0.20.3
Other versions

Please cite us if you use the software.

sklearn.linear_model.BayesianRidge
Examples using sklearn.linear_model.BayesianRidge

sklearn.linear_model.BayesianRidge

class sklearn.linear_model.BayesianRidge(*n_iter=300, tol=0.001, alpha_1=1e-06, alpha_2=1e-06, lambda_1=1e-06, lambda_2=1e-06, compute_score=False, fit_intercept=True, normalize=False, copy_X=True, verbose=False*) [source]

Bayesian ridge regression

Fit a Bayesian ridge model and optimize the regularization parameters lambda (precision of the weights) and alpha (precision of the noise).

Read more in the User Guide.

Parameters:

n_iter : *int, optional*
Maximum number of iterations. Default is 300.

tol : *float, optional*
Stop the algorithm if w has converged. Default is 1.e-3.

alpha_1 : *float, optional*
Hyper-parameter : shape parameter for the Gamma distribution prior over the a
Default is 1.e-6

alpha_2 : *float, optional*
Hyper-parameter : inverse scale parameter (rate parameter) for the Gamma dis
over the alpha parameter. Default is 1.e-6.

lambda_1 : *float, optional*
Hyper-parameter : shape parameter for the Gamma distribution prior over the la

predict(*self, X, return_std=False*) [source]

Predict using the linear model.

In addition to the mean of the predictive distribution, also its standard deviation can be returned.

Parameters:

X : *{array-like, sparse matrix} of shape (n_samples, n_features)*
Samples.

return_std : *bool, default=False*
Whether to return the standard deviation of posterior prediction.

Returns:

y_mean : *array-like of shape (n_samples,)*
Mean of predictive distribution of query points.

y_std : *array-like of shape (n_samples,)*
Standard deviation of predictive distribution of query points.

使い方はその他の教師あり学習の手法とほぼ同様

予測のreturnがy_meanとy_stdがある

⇒ 単純に一点で予測するのではなく、分布で予測できる

ベイズ線形回帰：概略

最終的に計算したいもの

予測分布:$p(y_{new}|\boldsymbol{y}, \boldsymbol{X}, x_{new})$

モデル：$y = w_0 + w_1 x + w_2 x^2 + \cdots + w_m x^m = \boldsymbol{w}^T \boldsymbol{\phi}(x)$

$$p(y|\boldsymbol{w}, x) = N\left(y \mid \boldsymbol{w}^T \boldsymbol{\phi}(x), \beta^{-1}\right)$$

$$p(\boldsymbol{w}) = N(\boldsymbol{w}|0, \alpha^{-1}\mathbf{I})$$

$\boldsymbol{X}$:訓練データ
$\boldsymbol{y}$:訓練データのラベル

$$\boldsymbol{\phi}(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^m \end{bmatrix} \quad \boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}$$

$$p(y_{new}|\boldsymbol{y}, \boldsymbol{X}, x_{new}) = \int_{-\infty}^{\infty} p(y_{new}|\boldsymbol{w}, \boldsymbol{t}, \boldsymbol{X}, x_{new}) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}, x_{new})\, d\boldsymbol{w}$$

y_{new}は $\boldsymbol{w}, x_{new}$ のみに依存

$\boldsymbol{w}$はx_{new}に依存しない

$$p(y_{new}|\boldsymbol{w}, x_{new}) = N(y_{new}|\boldsymbol{w}^T \boldsymbol{\phi}(x_{new}), \sigma^2)$$

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

ベイズの定理

$\boldsymbol{w}$はガウス分布に従う（共役事前分布）

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) \propto p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})p(\boldsymbol{w})$$

ベイズ線形回帰：実際の計算（1/2）

$\boldsymbol{w}$に関する事後分布を計算する

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{X}) \propto p(\boldsymbol{y}|\,\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w})$$

$$p(y \mid \boldsymbol{X},\boldsymbol{w}) = \prod_{n=1}^{N} N(y_n \mid \boldsymbol{w}^T\boldsymbol{\phi}(x_n),\beta^{-1})$$

$$\propto \prod_{n=1}^{N} \exp\left(-\frac{\left(y_n-\boldsymbol{w}^T\boldsymbol{\phi}(x_n)\right)^2}{2\beta^{-1}}\right)$$

$$= \exp\left(-\frac{\sum_{n=1}^{N}\left(y_n-\boldsymbol{w}^T\boldsymbol{\phi}(x_n)\right)^2}{2\beta^{-1}}\right)$$

$$= \exp\left(-\frac{(\boldsymbol{y}-\boldsymbol{\Phi w})^T(\boldsymbol{y}-\boldsymbol{\Phi w})}{2\beta^{-1}}\right)$$

$$p(\boldsymbol{w}) = N\left(\boldsymbol{w}\middle|0,\alpha^{-1}\mathbf{I}\right) \propto \exp\left(-\frac{\boldsymbol{w}^T\boldsymbol{w}}{2\alpha^{-1}}\right)$$

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{X}) \propto \exp\left(-\frac{(\boldsymbol{y}-\boldsymbol{\Phi w})^T(\boldsymbol{y}-\boldsymbol{\Phi w})}{2\sigma^2}\right)\exp\left(-\frac{\boldsymbol{w}^T\boldsymbol{w}}{2\alpha^{-1}}\right)$$

$$= N(\boldsymbol{w}|\boldsymbol{m}_N,\boldsymbol{S}_N)$$

$$\boldsymbol{m}_N = \beta\boldsymbol{S}_N\boldsymbol{\Phi}^T\boldsymbol{y}$$

$$\boldsymbol{S}_N = \alpha\mathbf{I}+\beta\boldsymbol{\Phi}^T\boldsymbol{\Phi}$$

ベイズ線形回帰：実際の計算（2/2）

最終的に計算したい 予測分布:$p(y_{new}|\boldsymbol{y}, \boldsymbol{X}, x_{new})$

$$\int p(y_{new}|\boldsymbol{w}, \boldsymbol{y}, \boldsymbol{X}, x_{new}) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}, x_{new}) d\boldsymbol{w}$$

$$= \int N(y|\boldsymbol{w}^T \boldsymbol{\phi}(x_{new}), \sigma^2) \quad N(\boldsymbol{w}|\boldsymbol{m}_N, \boldsymbol{S}_N) \quad d\boldsymbol{w}$$

$$= N(\boldsymbol{w}|\boldsymbol{m}_{new}, \sigma^2_{new})$$

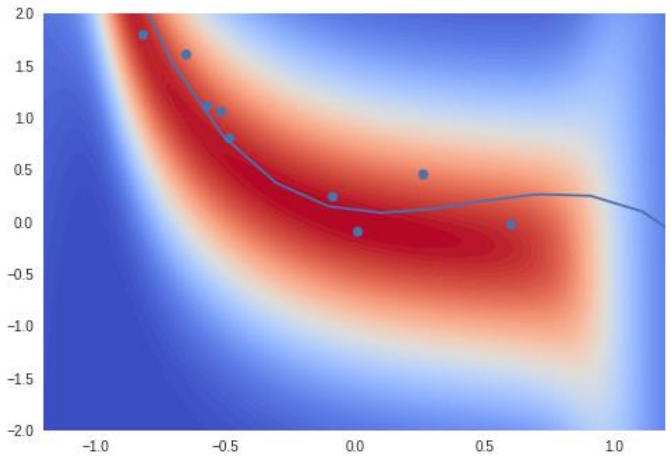
$$\boldsymbol{m}_{new} = \boldsymbol{\phi}(x_{new}) \boldsymbol{m}_N$$
$$\sigma^2_{new} = \sigma^2 + \boldsymbol{\phi}(x_{new})^T \boldsymbol{S}_N \boldsymbol{\phi}(x_{new}),$$

$\boldsymbol{t}, \boldsymbol{X}$から計算される量

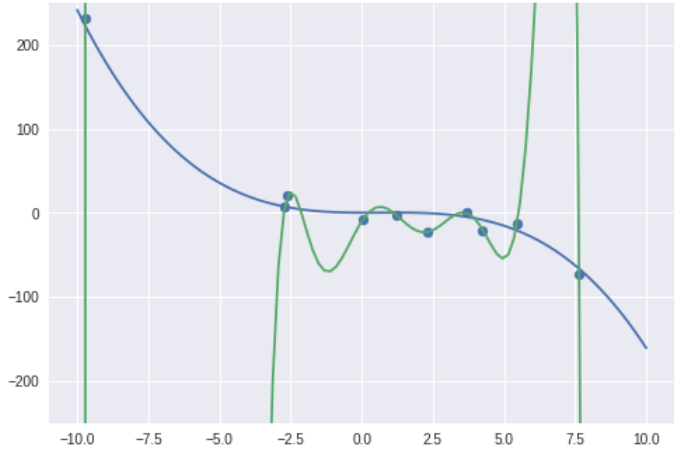
$$\boldsymbol{m}_N = \beta \boldsymbol{S}_N \boldsymbol{\Phi}^T \boldsymbol{y}$$
$$\boldsymbol{S}_N = \alpha \mathbf{I} + \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi}$$

多項式（９次式）フィッティングにおけるベイズ線形回帰と線形回帰の比較

ベイズ線形回帰の予測分布を
ヒートマップで表現



線形回帰の場合



ベイズ線形回帰を用いる場合、
大きめの次数の式でフィッティングをしても過学習しない

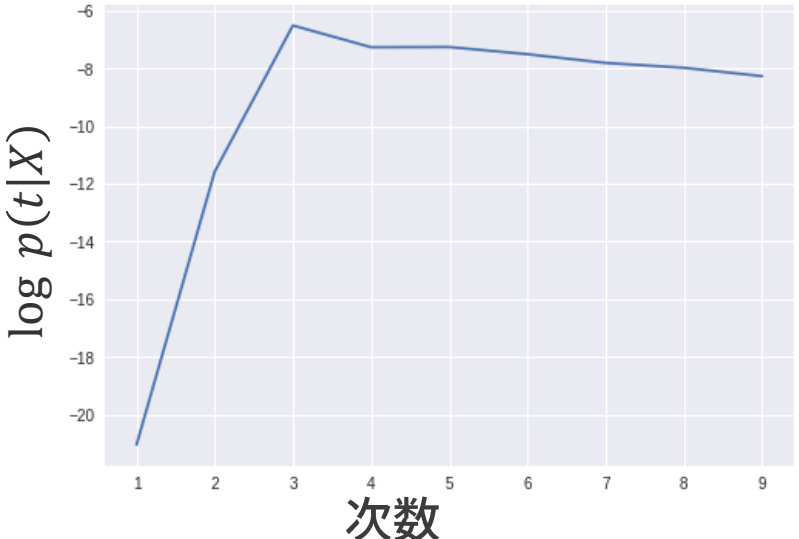
ベイズ線形回帰の評価

モデル選択（次数やハイパーパラメータの決定）は対数周辺尤度（エビデンス）を用いて計算可能

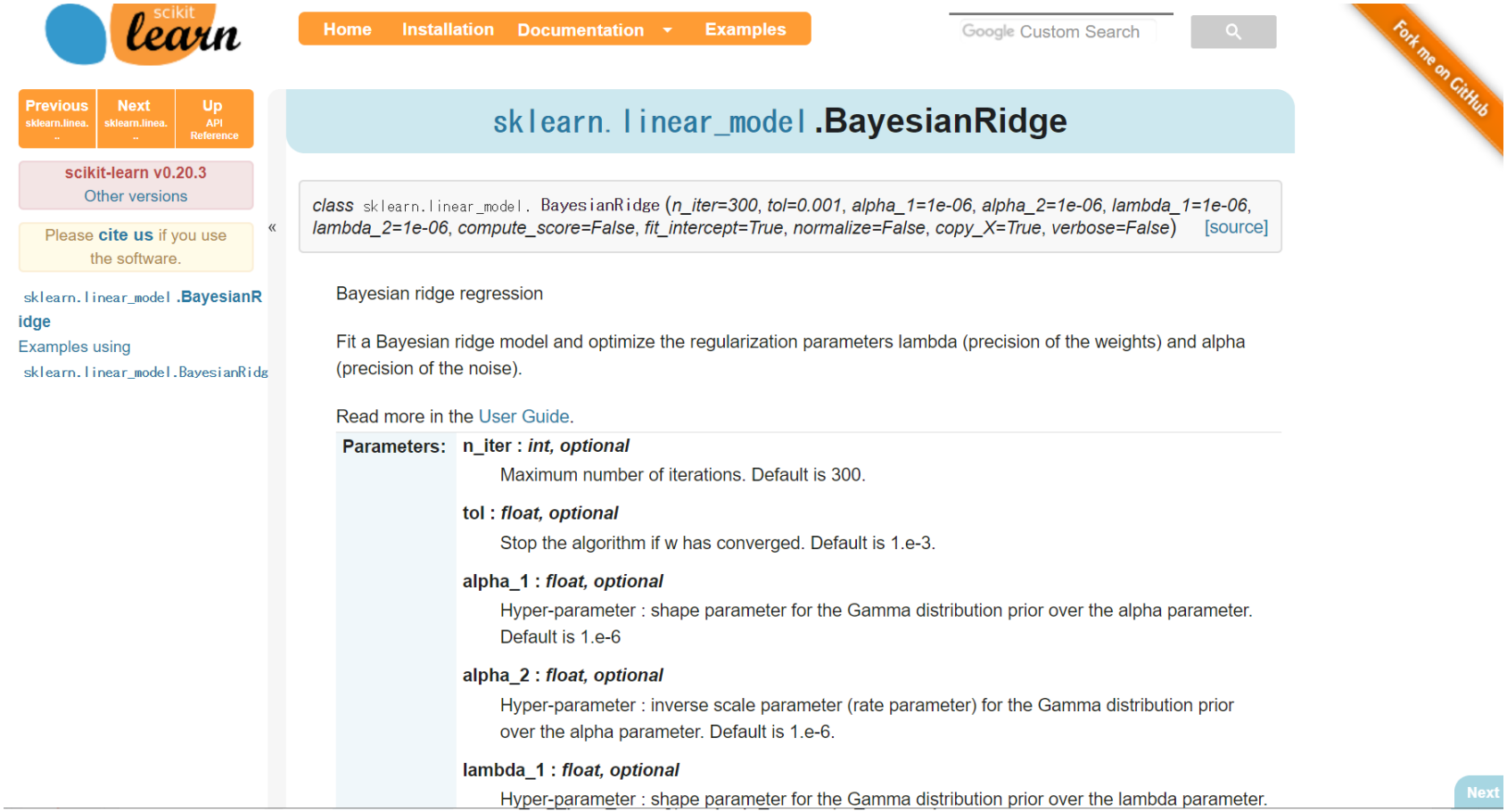
$$\log p(y|X) = \frac{M}{2}\log \alpha \ + \frac{N}{2}\log\beta - E(\boldsymbol{m}_N) - \frac{1}{2}\log|\boldsymbol{S}_N| - \frac{N}{2}\log(2\pi)$$

$$E(\boldsymbol{m}_N) = \frac{\beta}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{m}_N\|^2 + \frac{\alpha}{2}\boldsymbol{m}_N^{\boldsymbol{T}}\boldsymbol{m}_{\mathrm{N}}$$

多項式フィッティングの例では正しく
３次式を推定できることがわかる



ベイズ線形回帰：まとめ



scikit learn

Home Installation Documentation Examples

Google Custom Search

Fork me on GitHub

Previous sklearn.linea.. | Next sklearn.linea.. | Up API Reference

scikit-learn v0.20.3
Other versions

Please cite us if you use the software.

sklearn.linear_model.BayesianRidge
Examples using sklearn.linear_model.BayesianRidg

sklearn.linear_model.BayesianRidge

class sklearn.linear_model.BayesianRidge(*n_iter=300*, *tol=0.001*, *alpha_1=1e-06*, *alpha_2=1e-06*, *lambda_1=1e-06*, *lambda_2=1e-06*, *compute_score=False*, *fit_intercept=True*, *normalize=False*, *copy_X=True*, *verbose=False*) [source]

Bayesian ridge regression

Fit a Bayesian ridge model and optimize the regularization parameters lambda (precision of the weights) and alpha (precision of the noise).

Read more in the User Guide.

Parameters:

n_iter : *int, optional*
Maximum number of iterations. Default is 300.

tol : *float, optional*
Stop the algorithm if w has converged. Default is 1.e-3.

alpha_1 : *float, optional*
Hyper-parameter : shape parameter for the Gamma distribution prior over the alpha parameter. Default is 1.e-6

alpha_2 : *float, optional*
Hyper-parameter : inverse scale parameter (rate parameter) for the Gamma distribution prior over the alpha parameter. Default is 1.e-6.

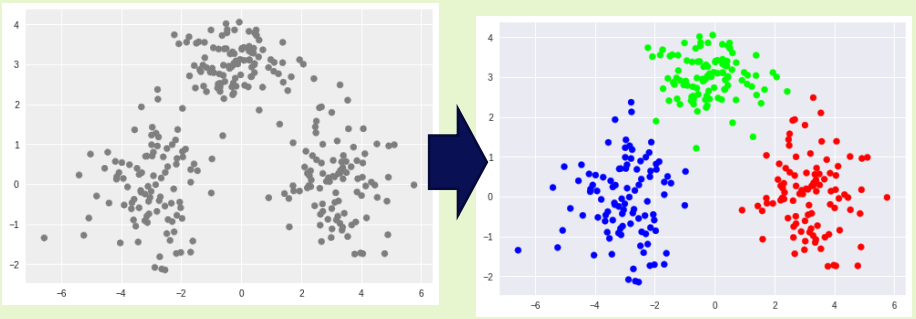
lambda_1 : *float, optional*
Hyper-parameter : shape parameter for the Gamma distribution prior over the lambda parameter.

Next

混合ガウスモデルのベイズ推定

最終的に計算したいもの

クラスタに関する事後分布: $p(\boldsymbol{Z}|\boldsymbol{X})$



$\boldsymbol{X}$: データセット

$\boldsymbol{Z}$: 各データの所属するクラスタ

$\boldsymbol{Z}$の要素 z_{ic} はデータiがクラスc に属しているとき１、それ以外の時０

- 事後分布の計算がクラスタリングに対応
- 対数周辺尤度を計算することで、クラスタ数の決定も可能
- 過学習しないため、クラスタ数を多めに設定してもよい

クラスタに関する事後分布の計算

ベイズの定理

$$p(\boldsymbol{Z}|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|\boldsymbol{Z})p(\boldsymbol{Z})}{p(\boldsymbol{X})}$$

$$p(\boldsymbol{Z}|\boldsymbol{X}) \propto p(\boldsymbol{X}|\boldsymbol{Z})p(\boldsymbol{Z})$$

ガウス分布ではないので計算困難

$$p(\boldsymbol{X}) = \int_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z})\, d\boldsymbol{Z}$$

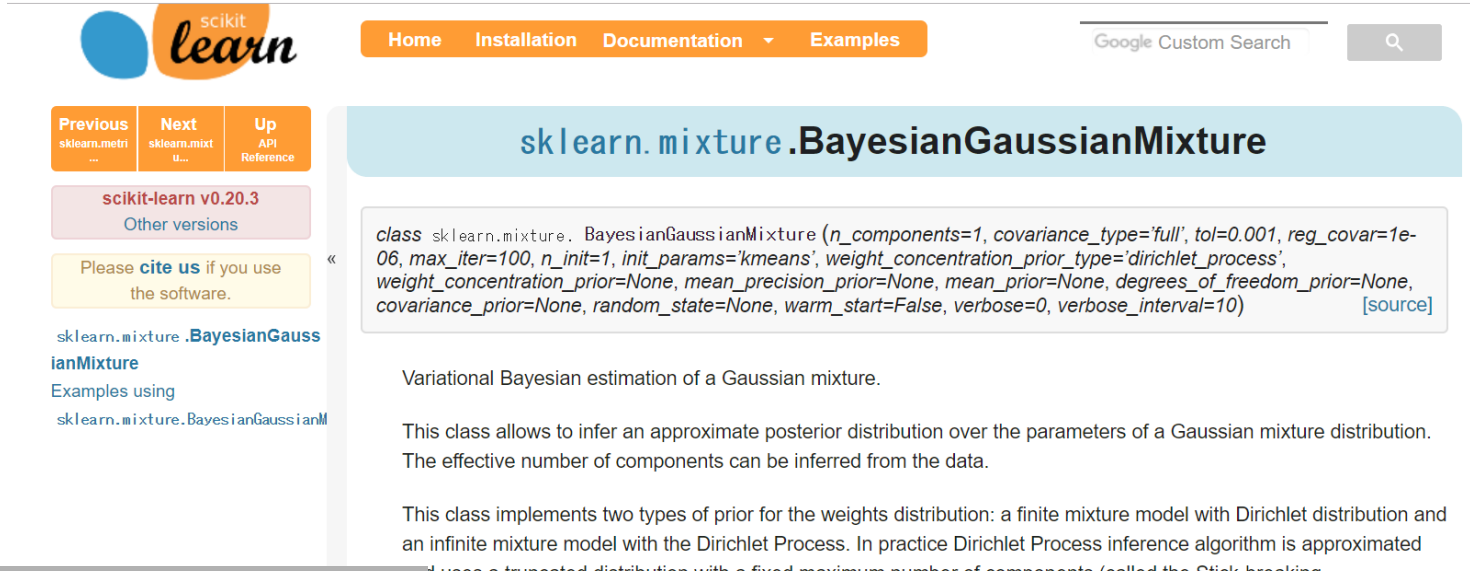
$\boldsymbol{Z}$の取りうる値はクラスタ数のデータ数乗通り→計算困難

複雑なモデルのベイズ推定：近似する

- 事後分布の近似が計算できれば、クラスタリングできる
- 対数周辺尤度の近似を計算できれば、クラスタ数の決定が可能
- 過学習しないため、大きめのクラスタ数にしても問題が起こりにくい

課題

K-means 法でクラスタリングを行うプログラムをベイズ推定の混合ガウス分布でクラスタリングを行うプログラムに変更してみましょう



参照
lecture_clustering.ipynb

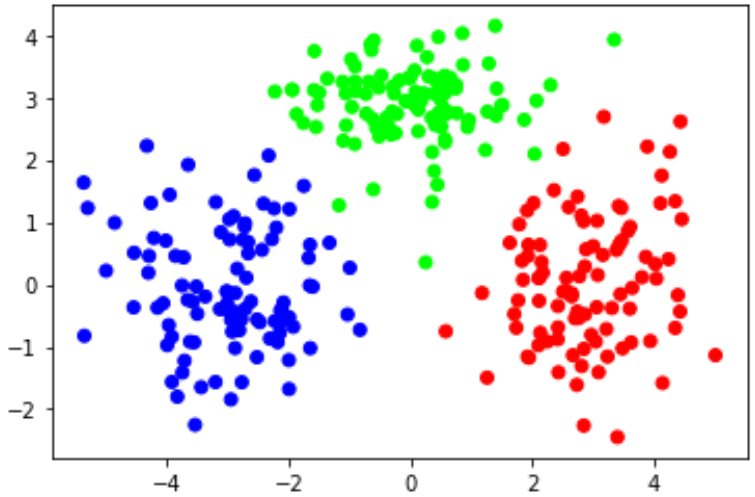
内部では事後分布の近似を計算している

```
from sklearn.mixture import BayesianGaussianMixture

model = BayesianGaussianMixture(n_components=2)
```

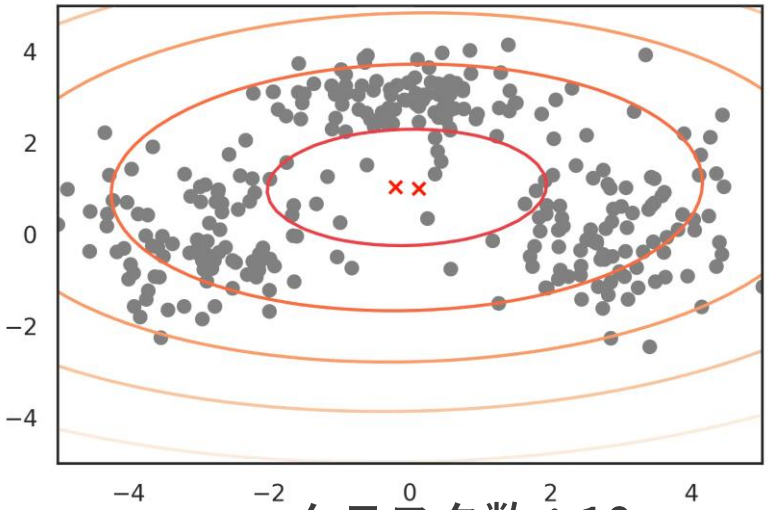
ベイズ混合ガウスモデルによるクラスタリングの振る舞い

元データ



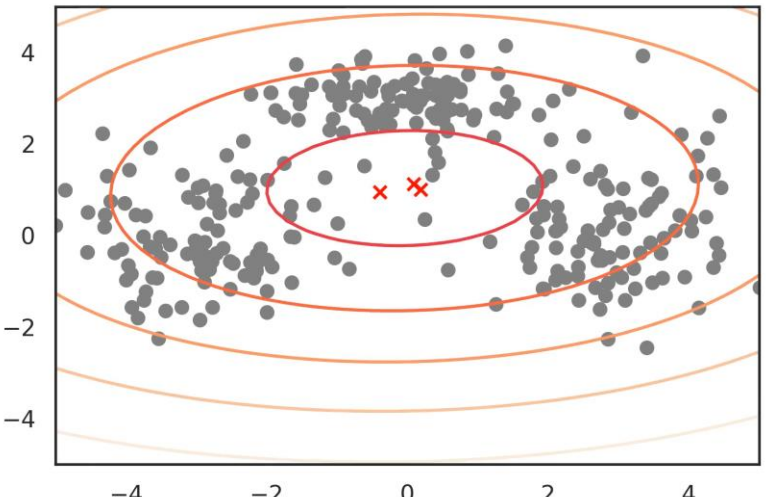
クラスタ数：2

ELBO=-7.64e+02



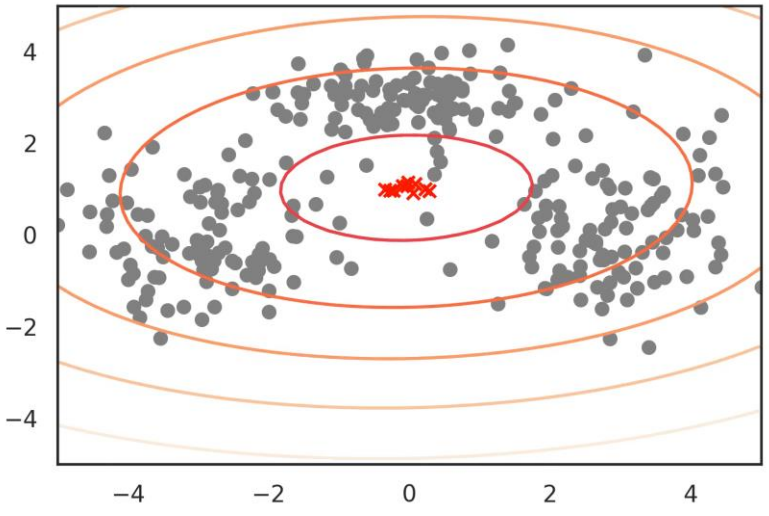
クラスタ数：3

ELBO=-7.75e+02



クラスタ数：10

ELBO=-8.47e+02



複雑なモデルのベイズ推定：近似する

- 事後分布の近似が計算できれば、クラスタリングできたことになる
- 対数周辺尤度の下限を計算できれば、クラスタ数の決定も可能
- 過学習もしにくいので、大きめのクラスタ数にしても問題が起こりにくい

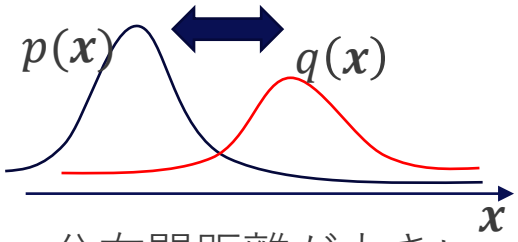
計算したいクラスタに関する事後分布 $p(\boldsymbol{Z}|\boldsymbol{X})$ を別の簡単な分布$q(\boldsymbol{Z})$で近似する

$$q(\boldsymbol{Z}) = \mathrm{argmin}_{q(\boldsymbol{Z})} KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X}))$$

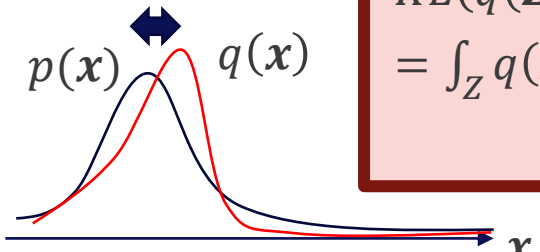
$q(\boldsymbol{Z})$ にどんな分布を持ってくるかを考える必要がある。最近ではニューラルネットを使った近似などが流行

KLダイバージェンス
- q とp の分布の近さを表す
- $q = p$ならば０で、 $q \neq p$ならば０以上の値をとる

$$KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X})) = \int_Z q(\boldsymbol{Z}) \log \frac{q(\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})} dZ$$



分布間距離が大きい



分布間距離が小さい

途中計算

$$q(\boldsymbol{Z}) = \mathrm{argmin}_{q(\boldsymbol{Z})} KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X}))$$

$$E_{q(\boldsymbol{Z})}[Y] = \int_{\boldsymbol{Z}} Y q(\boldsymbol{Z})\, d\boldsymbol{Z}$$

$$\begin{aligned} & KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X})) \\ &= E_{q(\boldsymbol{Z})}[\log q(\boldsymbol{Z})] - E_{q(\boldsymbol{Z})}[\log p(\boldsymbol{Z}|\boldsymbol{X})] \\ &= E_{q(\boldsymbol{Z})}[\log q(\boldsymbol{Z})] - E_{q(\boldsymbol{Z})}[\log p(\boldsymbol{Z},\boldsymbol{X})] + \log p(\boldsymbol{X}) \end{aligned}$$

$$KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X})) = -ELBO + \log p(\boldsymbol{X})$$

$$ELBO \equiv E_{q(\boldsymbol{Z})}[\log p(\boldsymbol{Z},\boldsymbol{X})] - E_{q(\boldsymbol{Z})}[\log q(\boldsymbol{Z})]$$

周辺対数尤度：エビデンス
データが決まると定数

近似誤差（KLダイバージェンス）の最小化は$ELBO$（Evidence lower bound）の最大化

$ELBO$はエビデンスの下限：

$$ELBO = \log p(\boldsymbol{X}) + KL(q(\boldsymbol{Z}) \,||\, p(\boldsymbol{Z}|\boldsymbol{X}))$$

変分ベイズ法：
Coordinate ascent mean-field variational inference (Variational-Bayes EM: VB-EM)

$$ELBO \equiv E_{q(\boldsymbol{Z})}[\log p(\boldsymbol{Z},\boldsymbol{X})] - E_{q(\boldsymbol{Z})}[\log q(\boldsymbol{Z})]$$

最大化

$$ELBO = E_{q(z_j)}\left[E_{q(\boldsymbol{z}_{-j})}[\log p(\boldsymbol{Z},\boldsymbol{X})]\right] - E_{q(z_j)}[\log q(z_j)] + \mathbf{const}$$

$$= E_{q(z_j)}\left[E_{q(\boldsymbol{z}_{-j})}[\log p(\boldsymbol{Z},\boldsymbol{X})] - \log q(z_j)\right] + \mathbf{const}$$

$q(z_j)$と$q(\boldsymbol{z}_{-j})$に分離できる分布をqに選ぶ必要がある

$$= -KL\left(\exp E_{q(\boldsymbol{z}_{-j})}[\log p(\boldsymbol{Z},\boldsymbol{X})] \;||q(z_j)\right) + \mathbf{const}$$

$$q^*(z_j) \propto \exp E_{q(\boldsymbol{z}_{-j})}[\log p(\boldsymbol{Z},\boldsymbol{X})]$$

$q(\boldsymbol{z}_{-j})$は$\boldsymbol{z}_j$以外の変数に関する確率

ある変数を計算するのに残りの変数を使っているので、
反復計算になっている

一変数z_jのみの比例関係かつqは簡単な分布を選択するのでこの計算は容易

課題

対数周辺尤度の下界を計算をして、クラスタ数を決定してみましょう
（対数周辺尤度の下界が大きいほど適したモデルといえる）

converged_ : *bool*

True when convergence was reached in fit(), False otherwise.

n_iter_ : *int*

Number of step used by the best fit of inference to reach the convergence.

lower_bound_ : *float*

Lower bound value on the likelihood (of the training data with respect to the model) of the best fit of inference.

weight_concentration_prior_ : *tuple or float*

The dirichlet concentration of each component on the weight distribution (Dirichlet). The type depends on `weight_concentration_prior_type`:

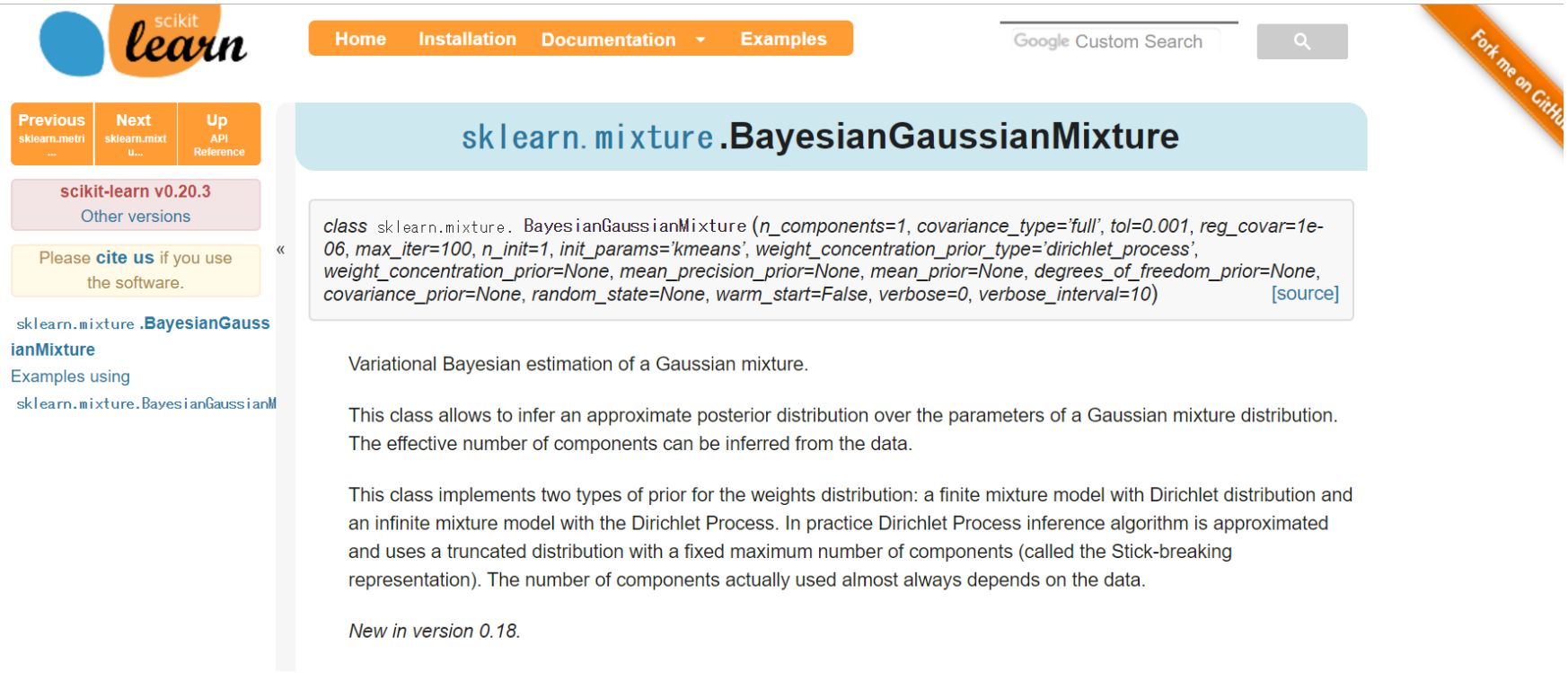
```
(float, float) if 'dirichlet_process' (Beta parameters),
float          if 'dirichlet_distribution' (Dirichlet parameters).
```

ation puts more mass in the center and will lead to more components being
concentration parameter will lead to more mass at the edge of the simplex.

参照
lecture_clustering.ipynb

```
model = BayesianGaussianMixture(n_components=2)
print(model.lower_bound_)
model = BayesianGaussianMixture(n_components=3)
print(model.lower_bound_)
```

混合ガウスモデルの変分ベイズ法（近似計算）：まとめ



scikit learn

Home Installation Documentation Examples

Google Custom Search

Fork me on GitHub

Previous sklearn.metri... | Next sklearn.mixtu... | Up API Reference

scikit-learn v0.20.3
Other versions

Please cite us if you use the software.

sklearn.mixture.BayesianGaussianMixture
Examples using sklearn.mixture.BayesianGaussianM

sklearn.mixture.BayesianGaussianMixture

class sklearn.mixture.BayesianGaussianMixture(*n_components=1, covariance_type='full', tol=0.001, reg_covar=1e-06, max_iter=100, n_init=1, init_params='kmeans', weight_concentration_prior_type='dirichlet_process', weight_concentration_prior=None, mean_precision_prior=None, mean_prior=None, degrees_of_freedom_prior=None, covariance_prior=None, random_state=None, warm_start=False, verbose=0, verbose_interval=10*) [source]

Variational Bayesian estimation of a Gaussian mixture.

This class allows to infer an approximate posterior distribution over the parameters of a Gaussian mixture distribution. The effective number of components can be inferred from the data.

This class implements two types of prior for the weights distribution: a finite mixture model with Dirichlet distribution and an infinite mixture model with the Dirichlet Process. In practice Dirichlet Process inference algorithm is approximated and uses a truncated distribution with a fixed maximum number of components (called the Stick-breaking representation). The number of components actually used almost always depends on the data.

New in version 0.18.

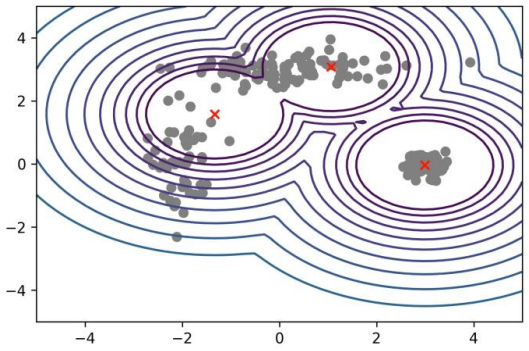
K-means(GMM + Hard-EM) vs GMM+EM vs GMM+VB-EM

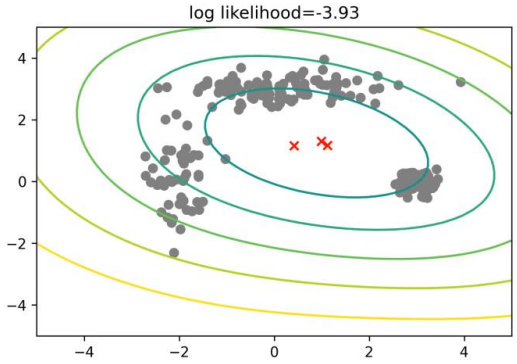
K-means
(GMM + Hard-EM)

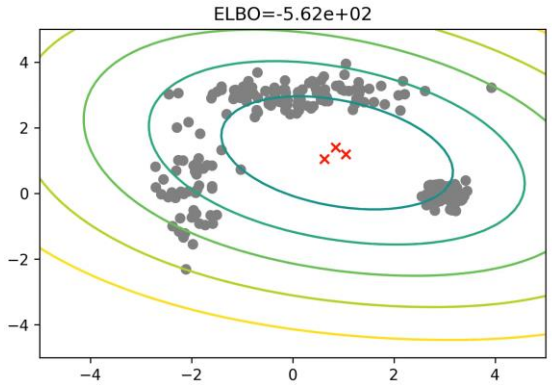
GMM+EM

GMM+VB

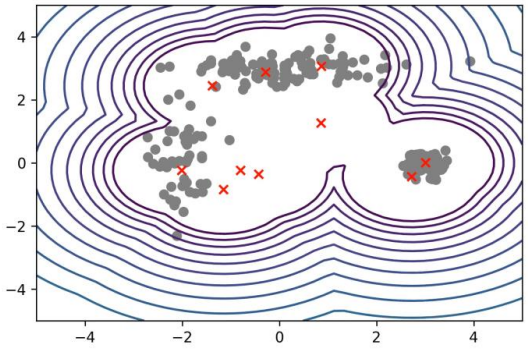
クラスタ数=3

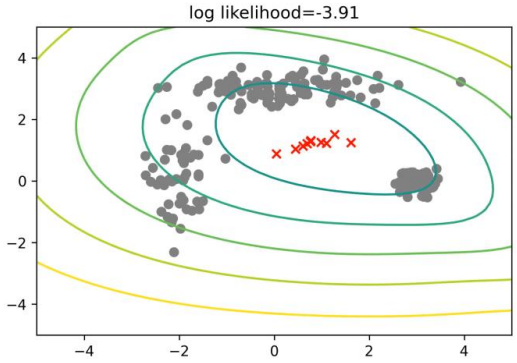


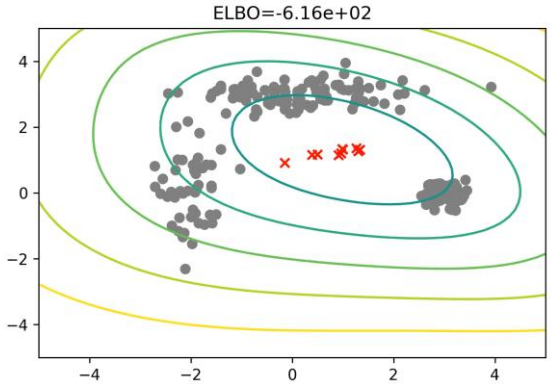




クラスタ数=10







まとめ

- 手続き的な理解だけではなくもう一歩踏み込んだ理解ができるとライブラリの使い方もよく理解できる
  - ナイーブベイズモデル・k-means法・混合ガウスモデル（EM・ベイズ）
- モデル選択の基準
  - 交差検証
  - AIC・BIC
  - 対数周辺尤度（エビデンス）（下界）
- ベイズ推定と確率モデル
  - 分布でほしいものを推定する→事後分布や予測分布を考える
  - ベイズ推定を使うと過学習やパラメータチューニングにおいてかなり多くの労力から解放される
  - 複雑なモデルでは計算困難なので変分近似（やサンプリング）を使う