教師なし学習

aki-goto-27

機械学習の種類（ざっくり）

機械学習

- 教師あり学習
  - 回帰　　連続値の予測
  - 分類　　種類の判別
- **教師なし学習**
  - クラスタリング（後述）　　データをグループ化していく
  - 次元削減(後述)　　データの特徴量の次元を減らす
- 強化学習　　報酬を最大化するために行動する方法を学ぶ

教師あり学習との違い

教師あり学習	教師なし学習
正解が示されている	正解が示されていない
入力から正しい出力を予測することが得意	データ内のパターンや構造を発見することが得意
回帰や分類に強い	次元削減やクラスタリングに強い

教師なし学習のメリット・デメリット

メリット

- 正解・不正解が不明瞭な場合にも利用できる
- 人間が発見できていない新たなパターンを見つけることができる

デメリット

- 正解がないため、学習結果の精度が低くなる傾向がある
- 発見したパターンが役に立たない可能性がある

機械学習の種類（ざっくり）

機械学習

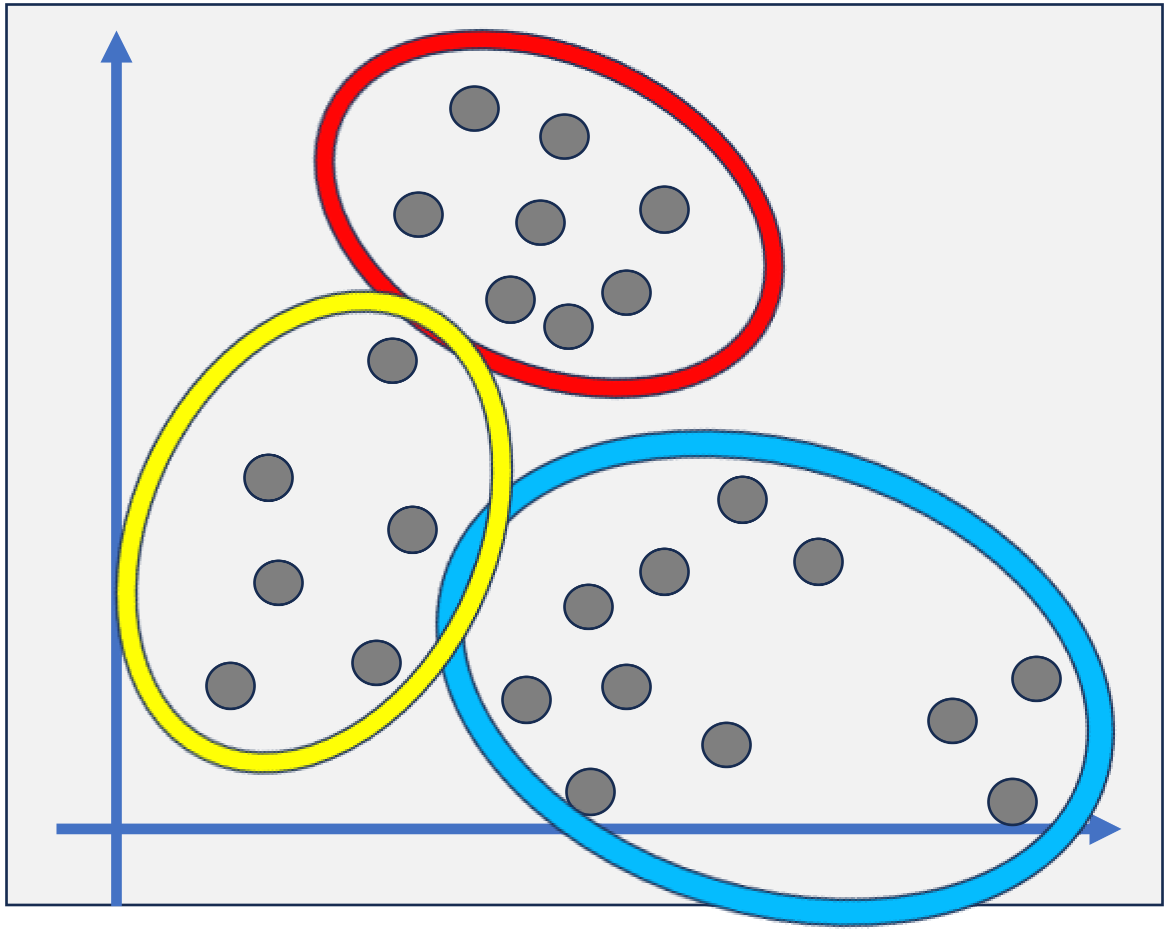
- 教師あり学習
  - 回帰　連続値の予測
  - 分類　種類の判別
- **教師なし学習**
  - **クラスタリング** (後述)　データをグループ化していく
  - 次元削減(後述)　データの特徴量の次元を減らす
- 強化学習　報酬を最大化するために行動する方法を学ぶ

クラスタリングとは？

グループを作って振り分ける

目的

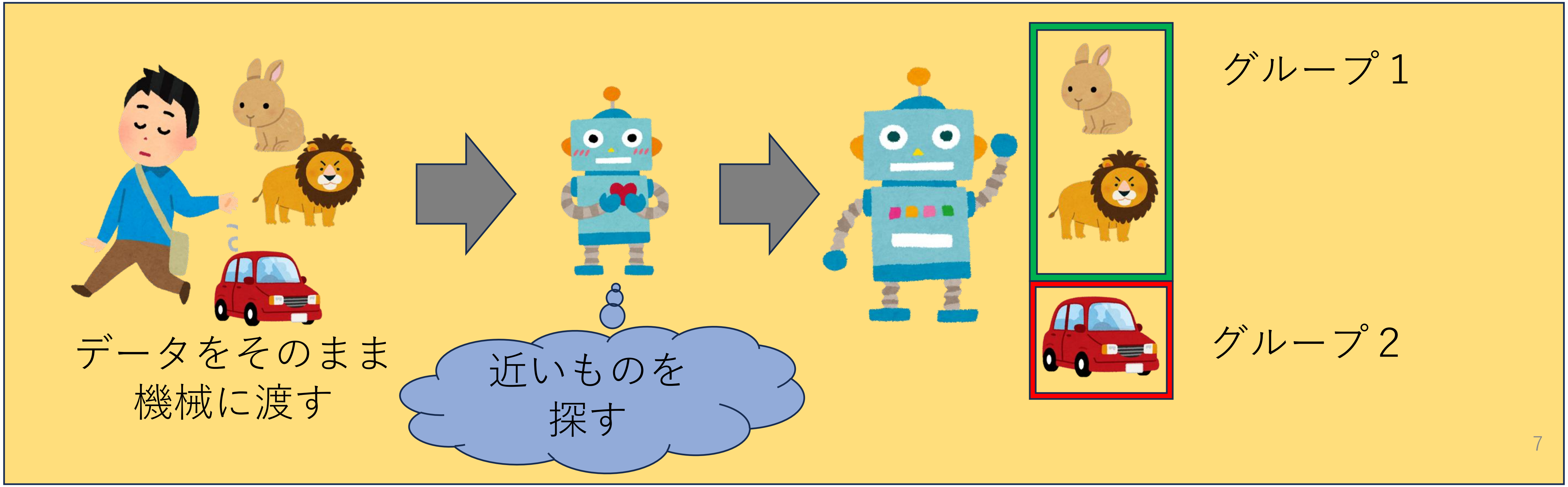
- **パターンの発見**
- **類似性の特定**
- **データの構造を理解する**



クラス分類とクラスタリングの違い（概要）

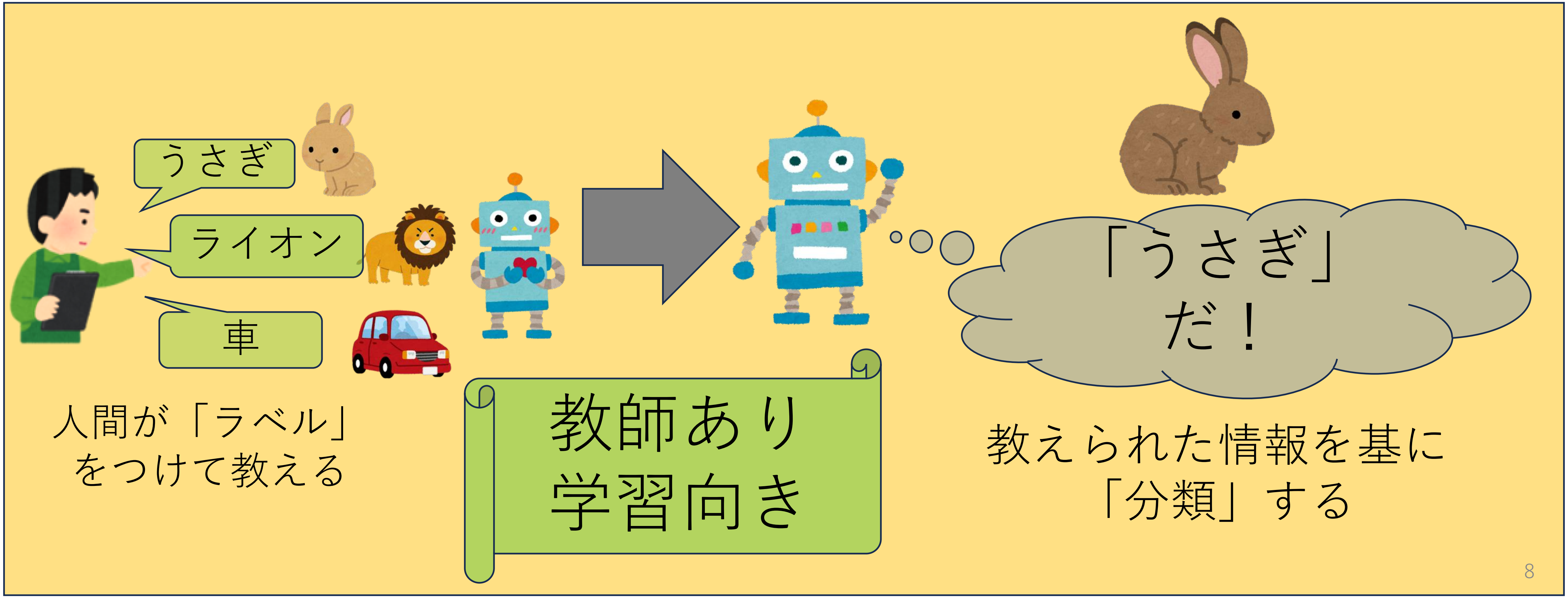
・クラスタリング

できたグループに対する解釈は人間次第
正しいかどうかも不明
例). 尻尾がついてるかついてないか？
生き物かどうか？



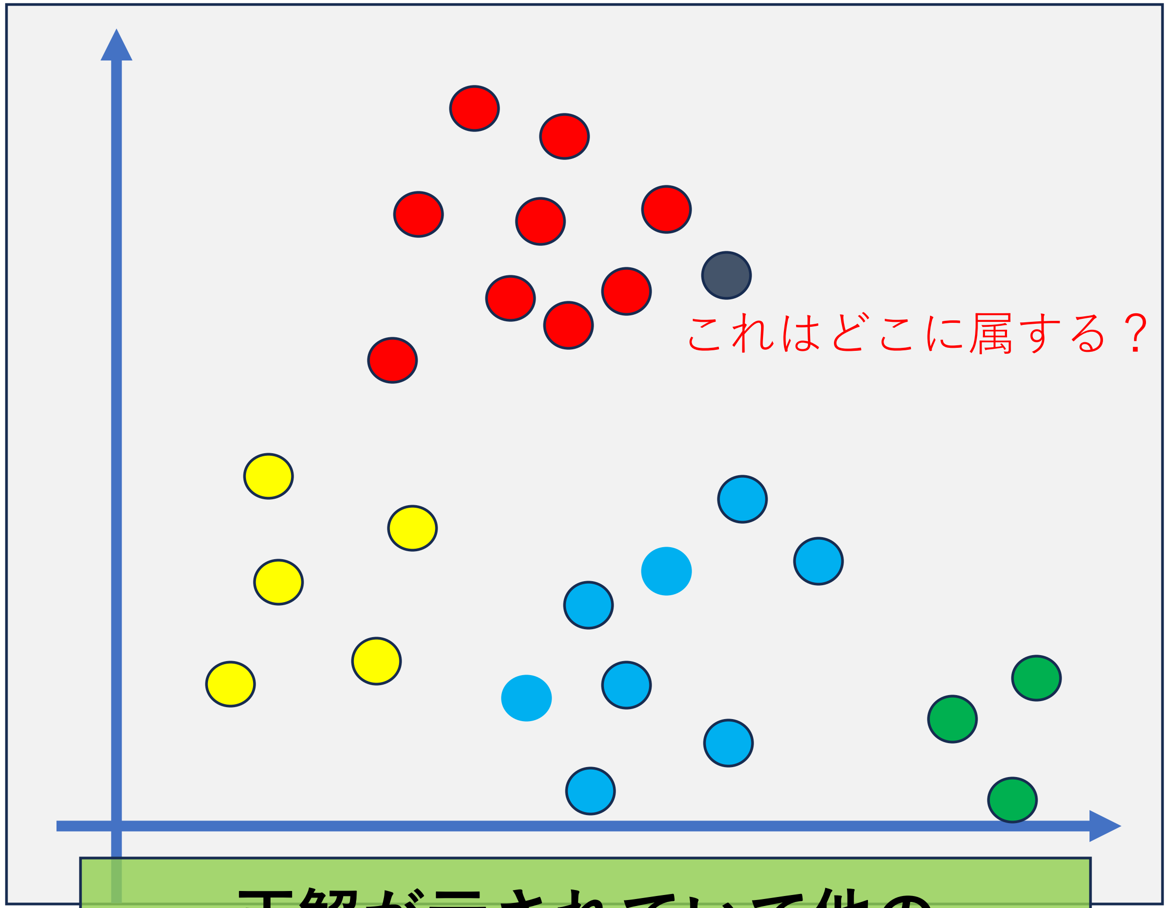
クラス分類とクラスタリングの違い（概要）

・クラス分類



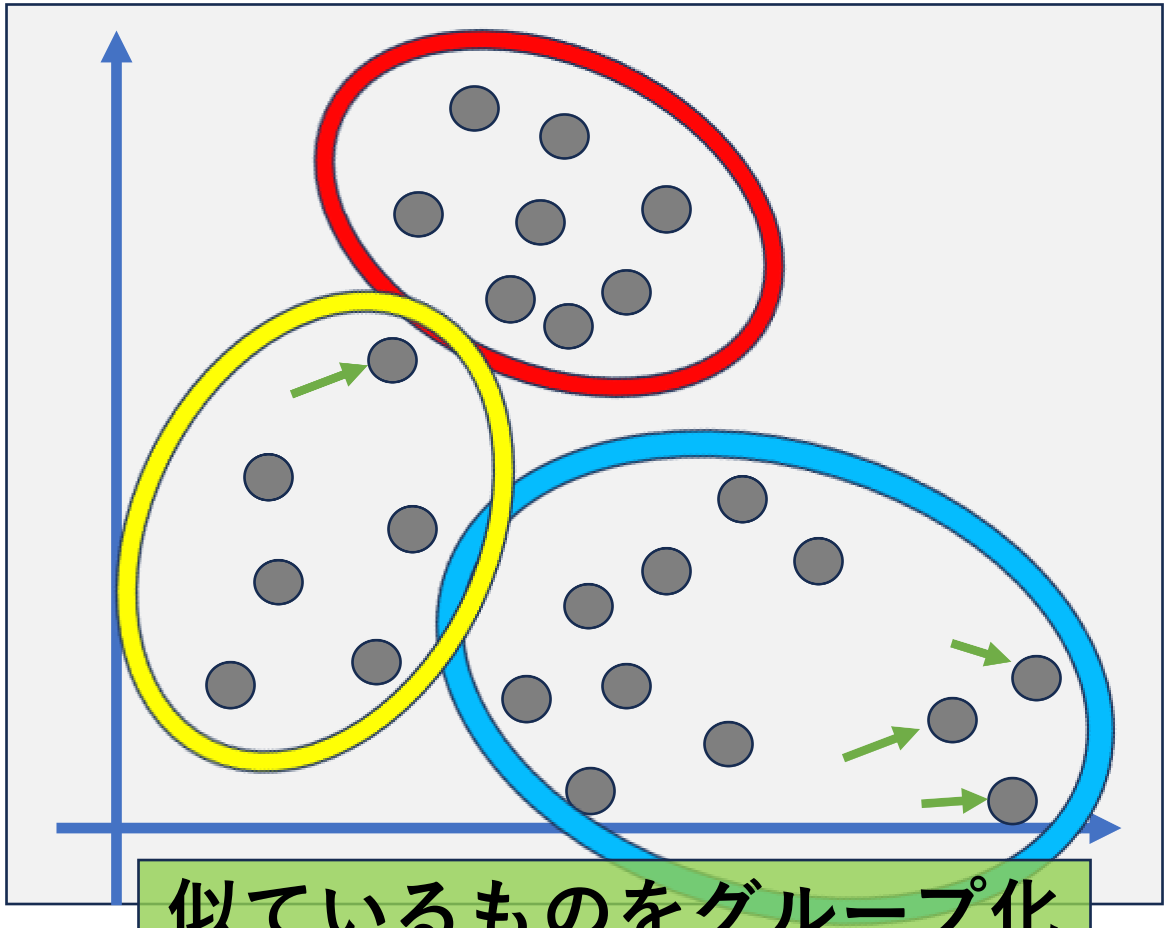
クラス分類とクラスタリングの違い（概要）

・クラス分類



正解が示されていて他の
データがどこに属すか判別する

・クラスタリング



似ているものをグループ化
正しいかは分からない

機械学習の種類（ざっくり）

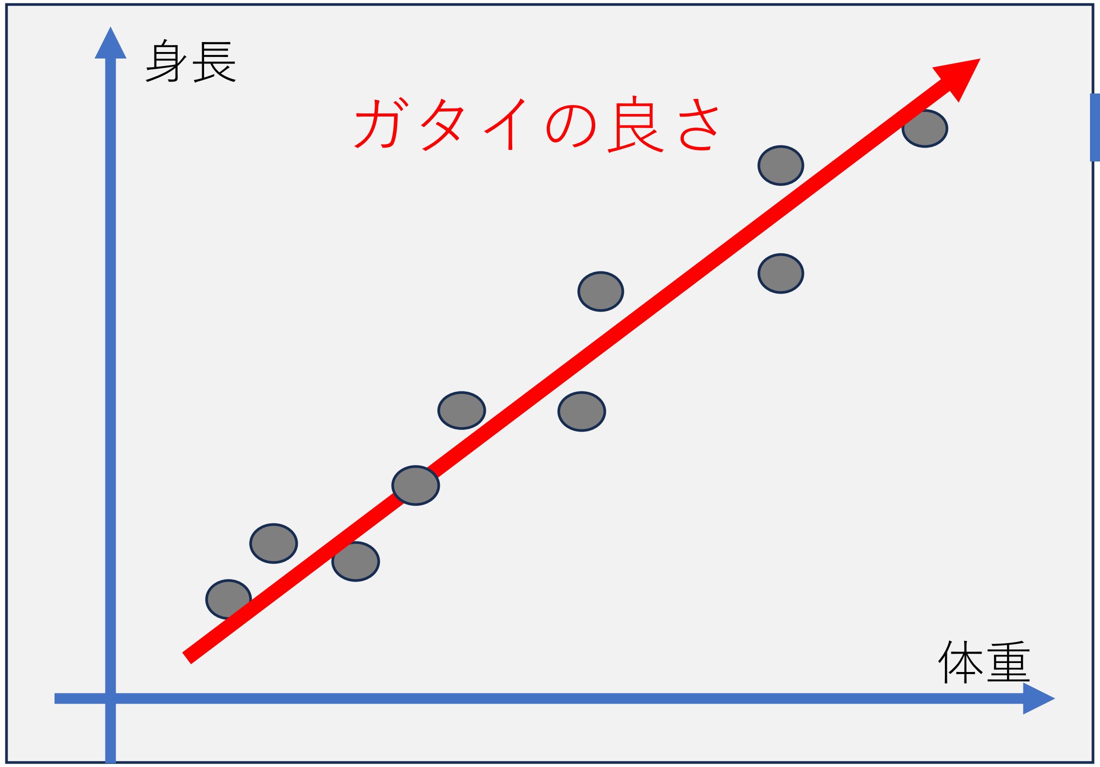
機械学習

- 教師あり学習
  - 回帰　連続値の予測
  - 分類　種類の判別
- **教師なし学習**
  - クラスタリング (後述)　データをグループ化していく
  - **次元削減**(後述)　データの特徴量の次元を減らす
- 強化学習　報酬を最大化するために行動する方法を学ぶ

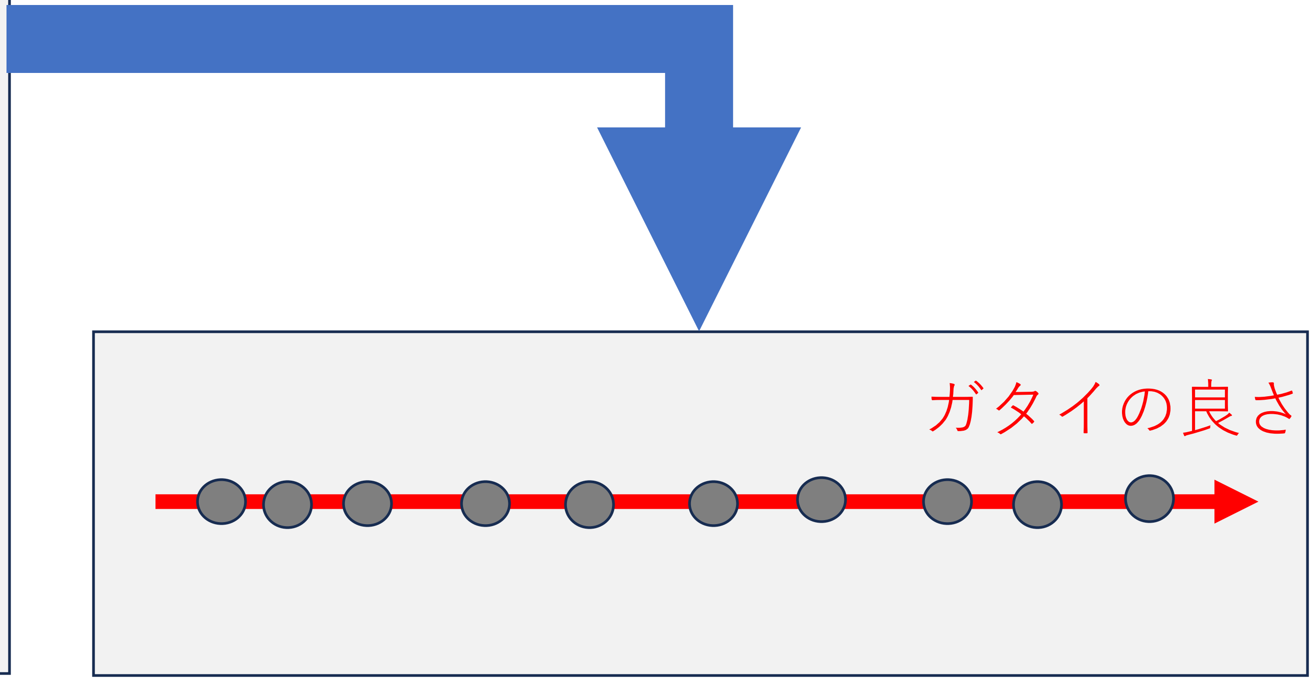
次元削減とは？

高次元からなる情報を、その意味を保ったまま

低次元の情報に落とし込むこと

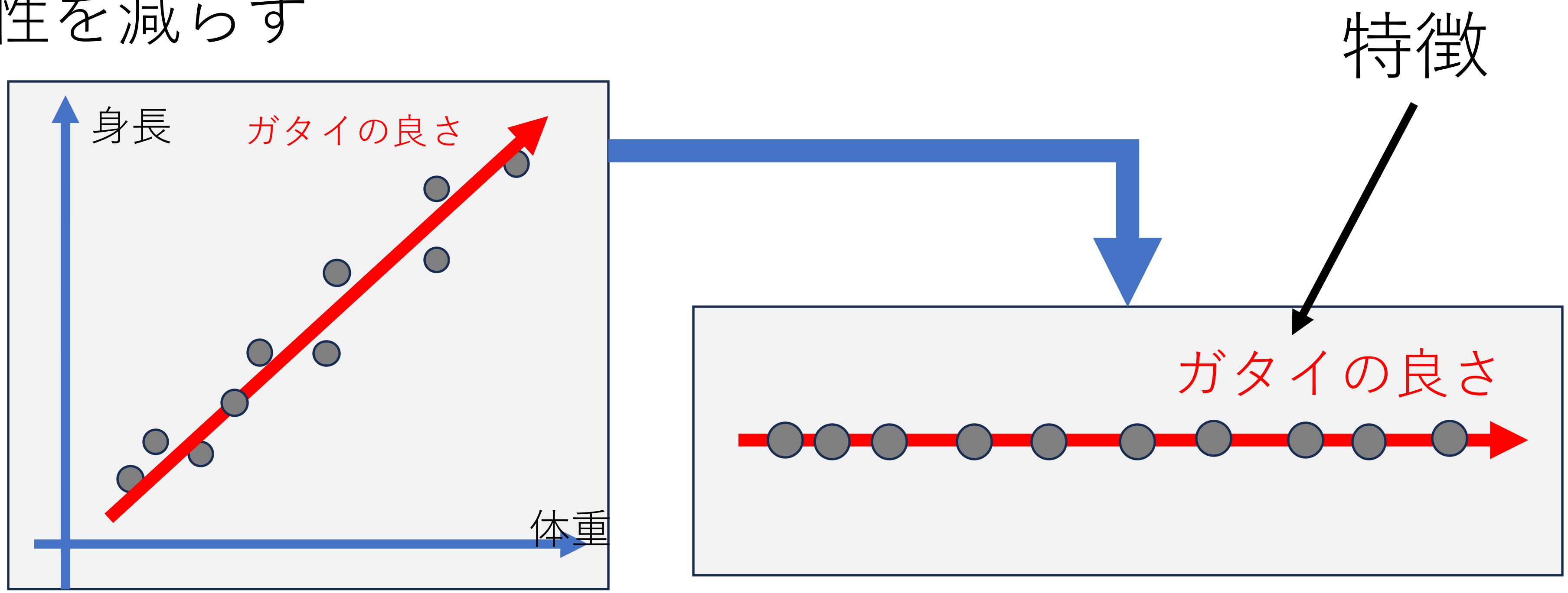


２次元から１次元への次元削減



次元削減を行う目的

- 特徴抽出
- データの可視化と理解
- 計算効率の向上
- ノイズや冗長性を減らす



次元削減のデメリット

- **情報の損失**　元のデータの特徴が失われる
- **選択バイアス**　手法の選択によって結果や精度が変わる
- **解釈の難しさ**　元のデータとどのように関連しているか判別が困難

教師なし学習の手法の一例

次元削減
- 主成分分析（PCA）
- SNE

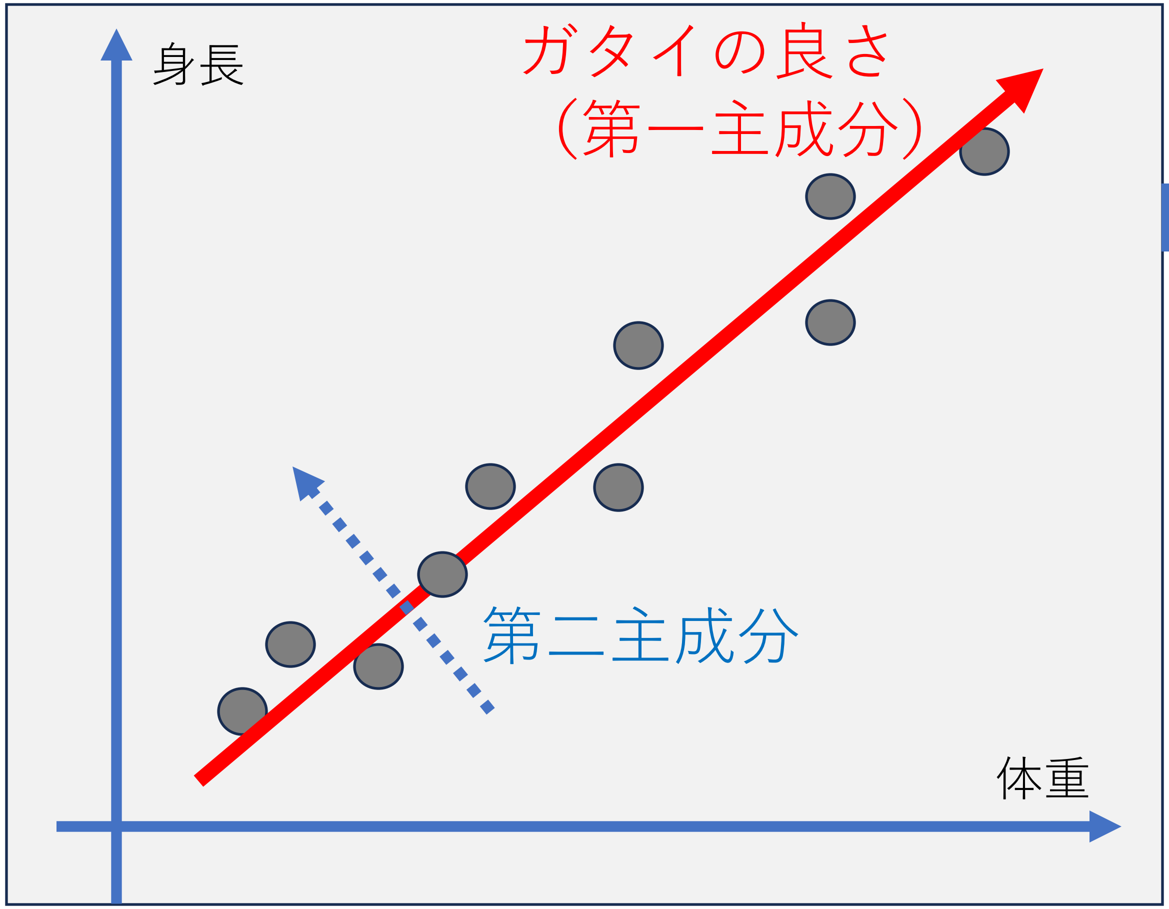
クラスタリング
- k-mean

主成分分析（PCA）

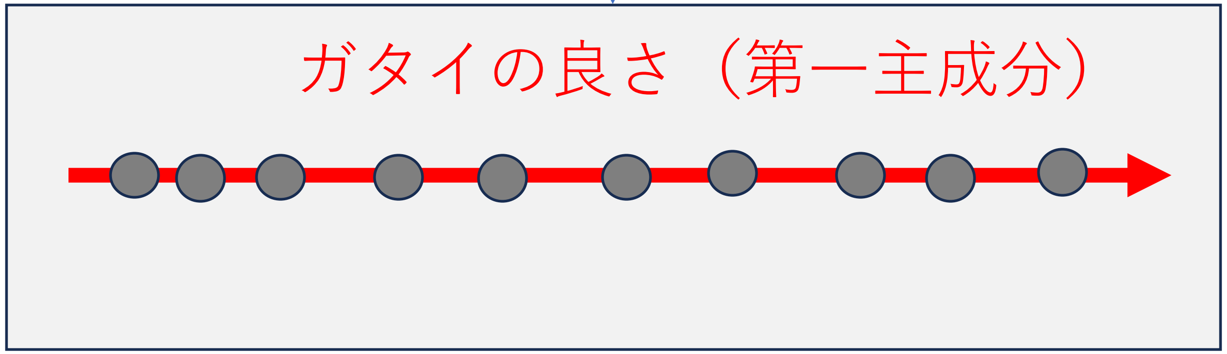
・次元削減（可視化）の最も基本的な手法

・最も情報量の多い「軸」を取り出し圧縮

例えば



２次元から１次元への次元削減



さきほどのこれも主成分分析だと捉えられる

主成分分析（方法の流れ）

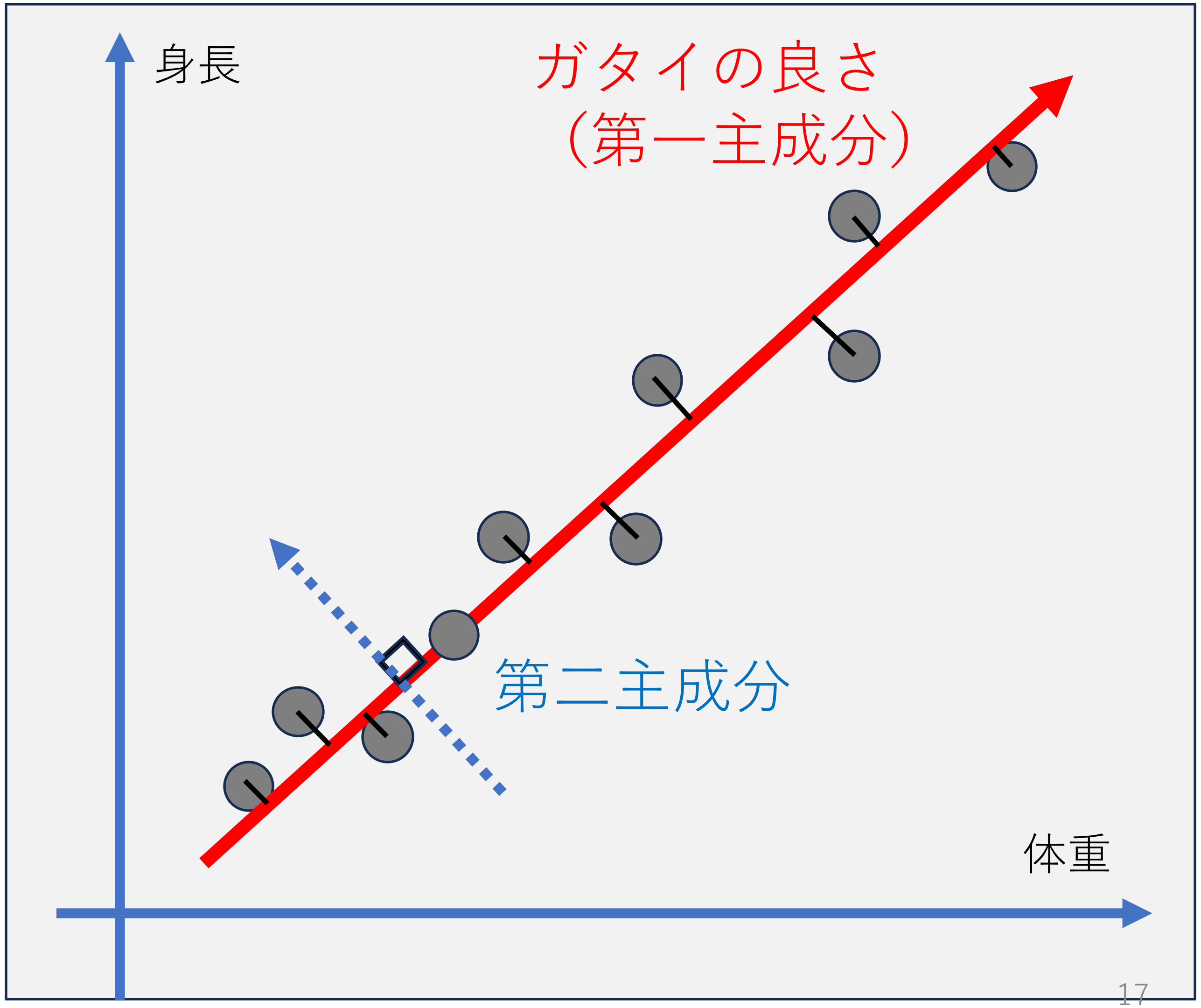
主成分（軸）の決め方

第１主成分

射影したデータの
分散を最大化

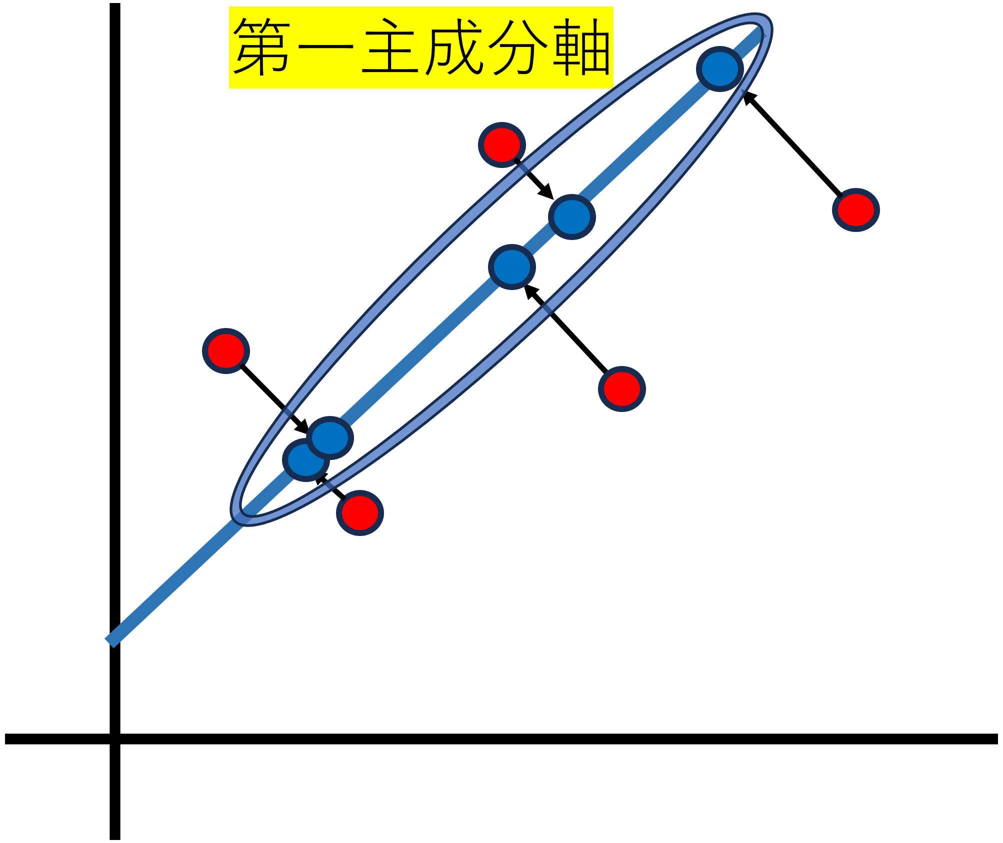
第２主成分以降

他の主成分に
直交しつつ
分散を最大化



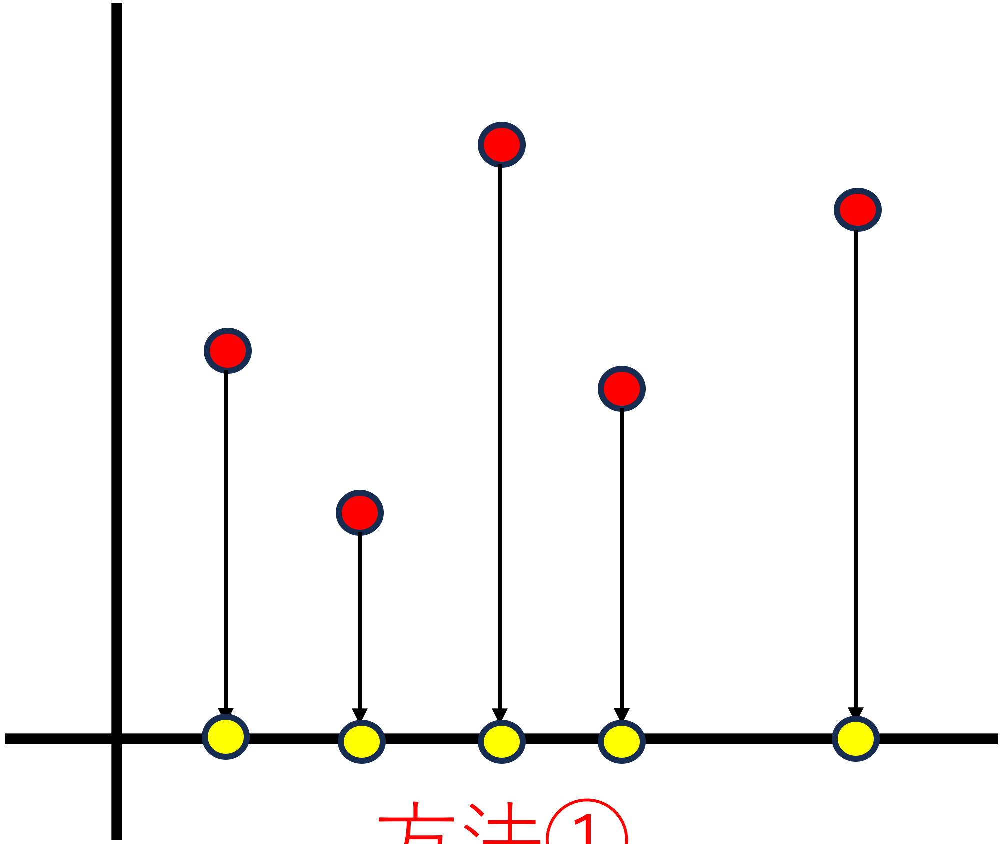
第１主成分

射影したデータの分散が最大になるような軸を探す

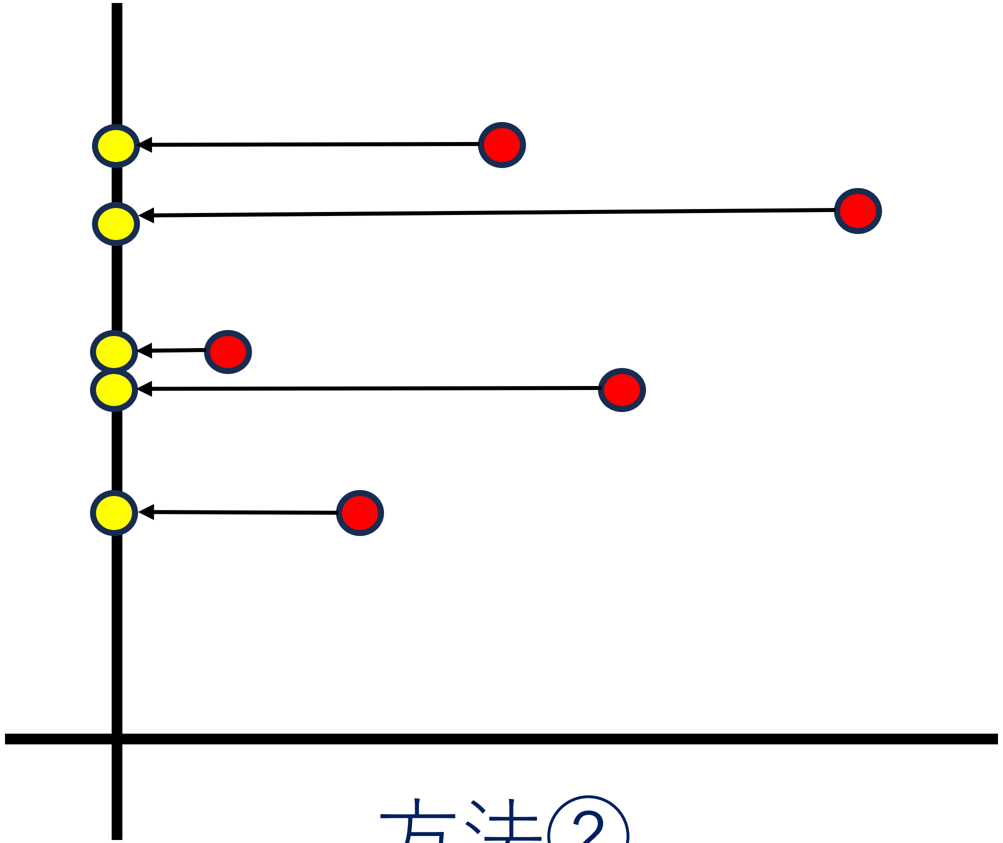


なぜ分散を最大化するのか①

２次元のデータを１次元に圧縮することを考える



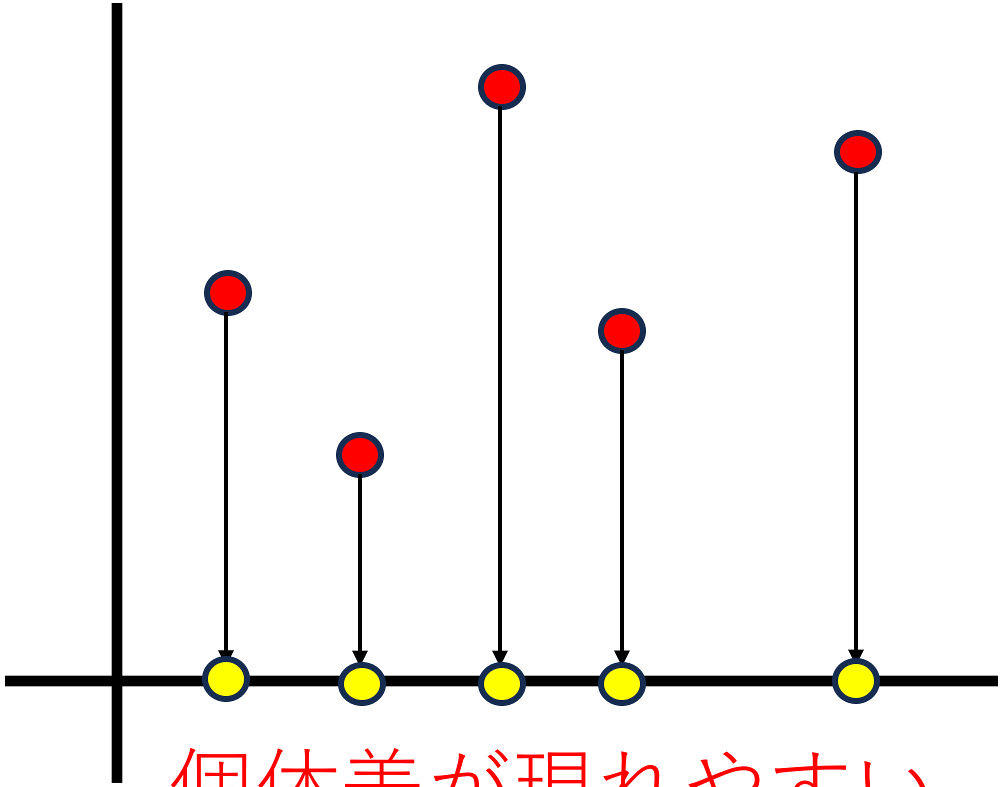
方法①
縦軸の情報の損失



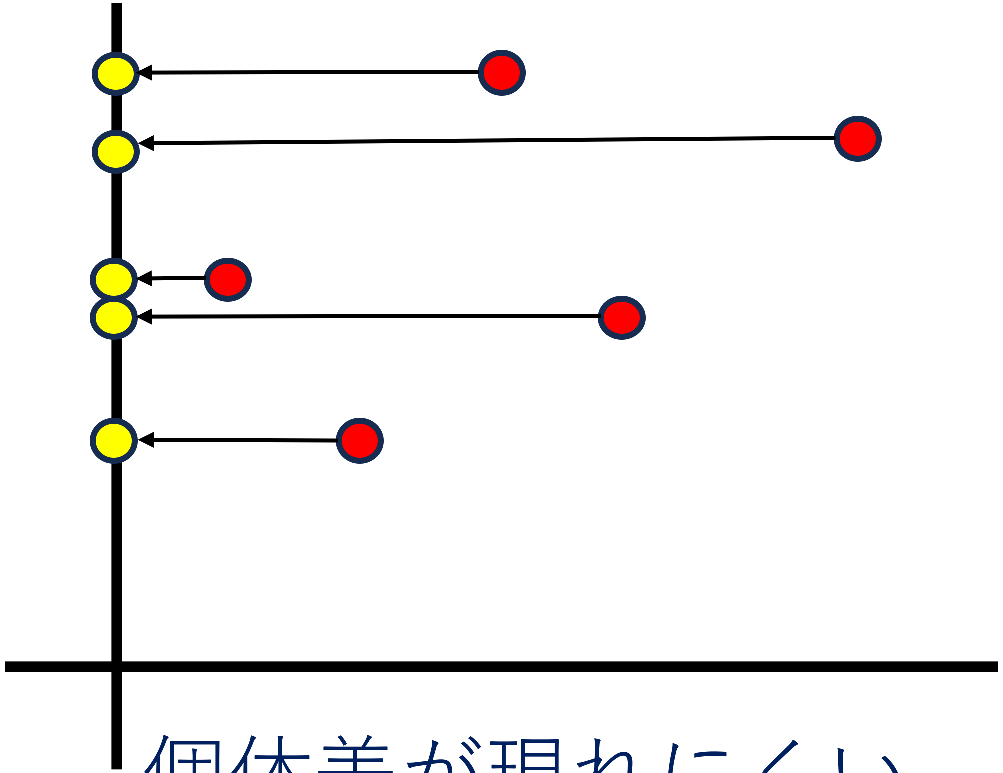
方法②
横軸の情報の損失

なぜ分散を最大化するのか②

射影したデータのばらつきが大きいほど
元のデータの情報を多く含んでいると考えられる



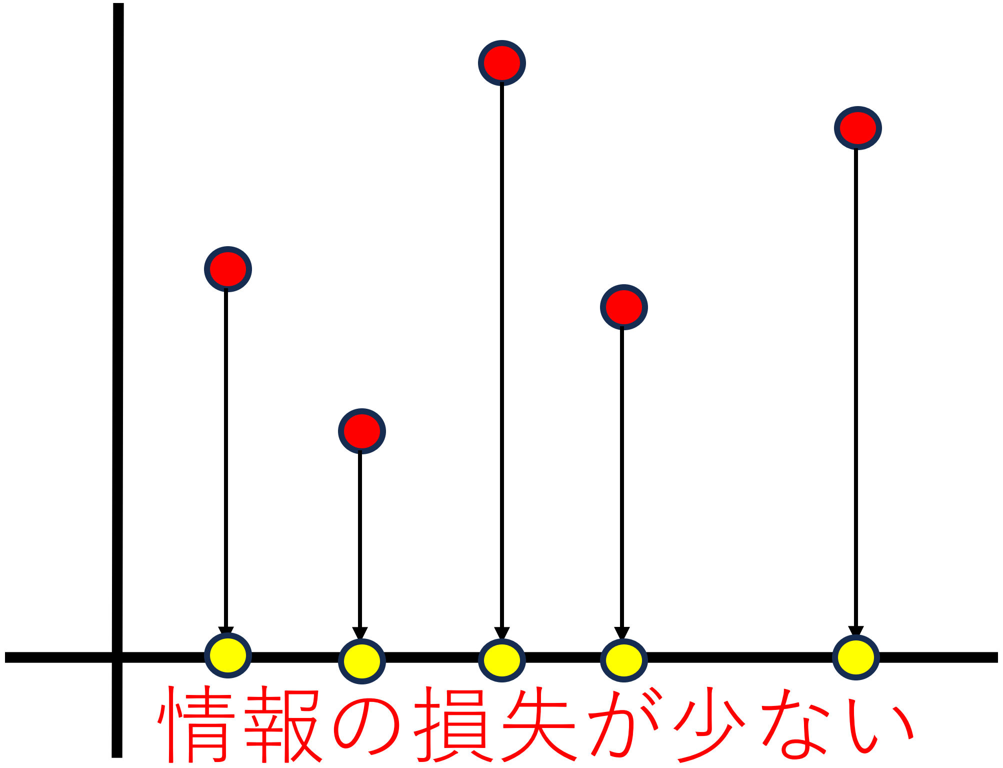
個体差が現れやすい

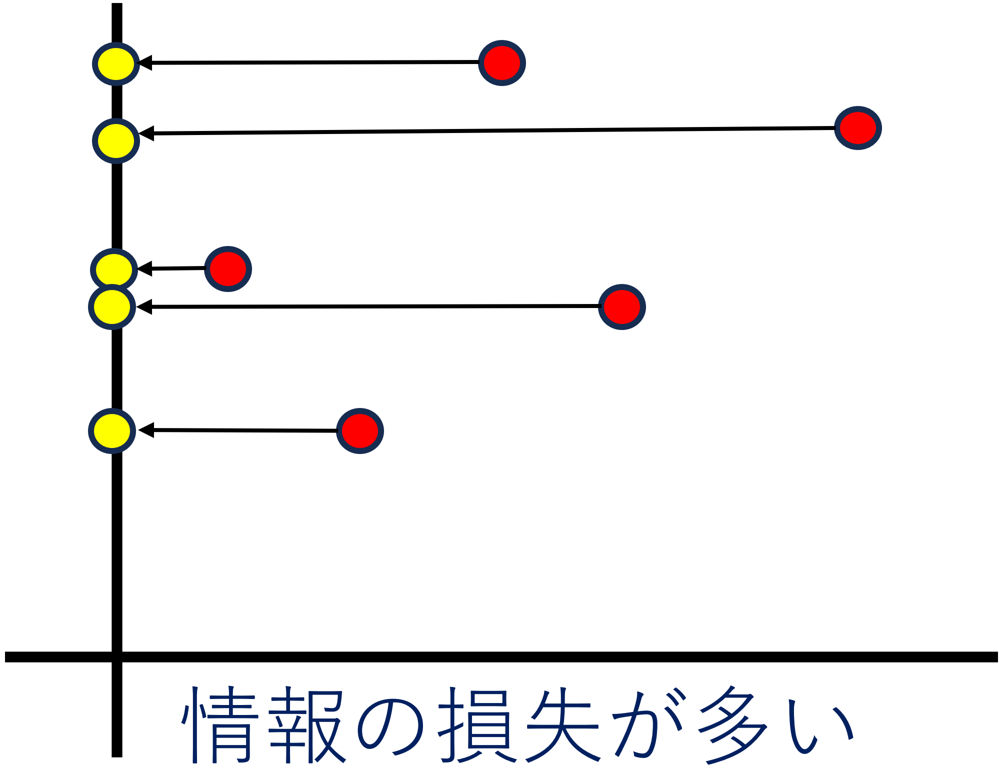


個体差が現れにくい

なぜ分散を最大化するのか③

元データの情報の損失が
できるだけ小さくなる軸を探したい



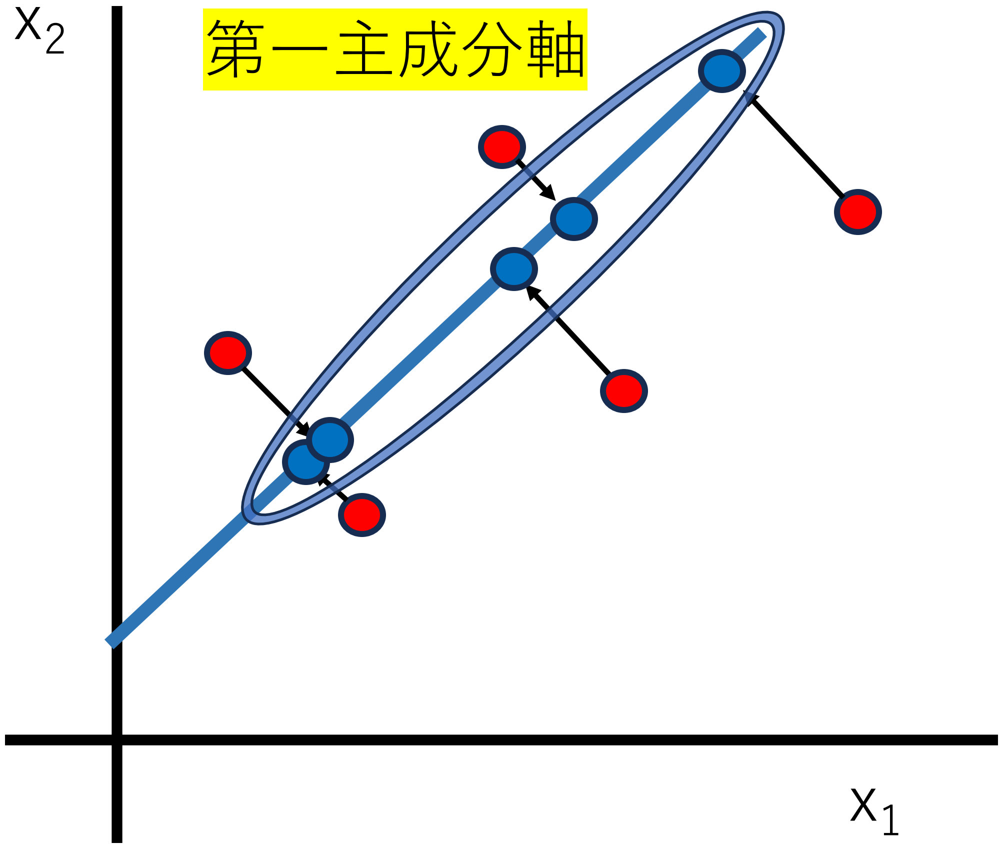


射影したデータの分散が最大となる軸を探す！

第１主成分

2次元の場合の例

射影したデータの分散が最大になるような軸を探す



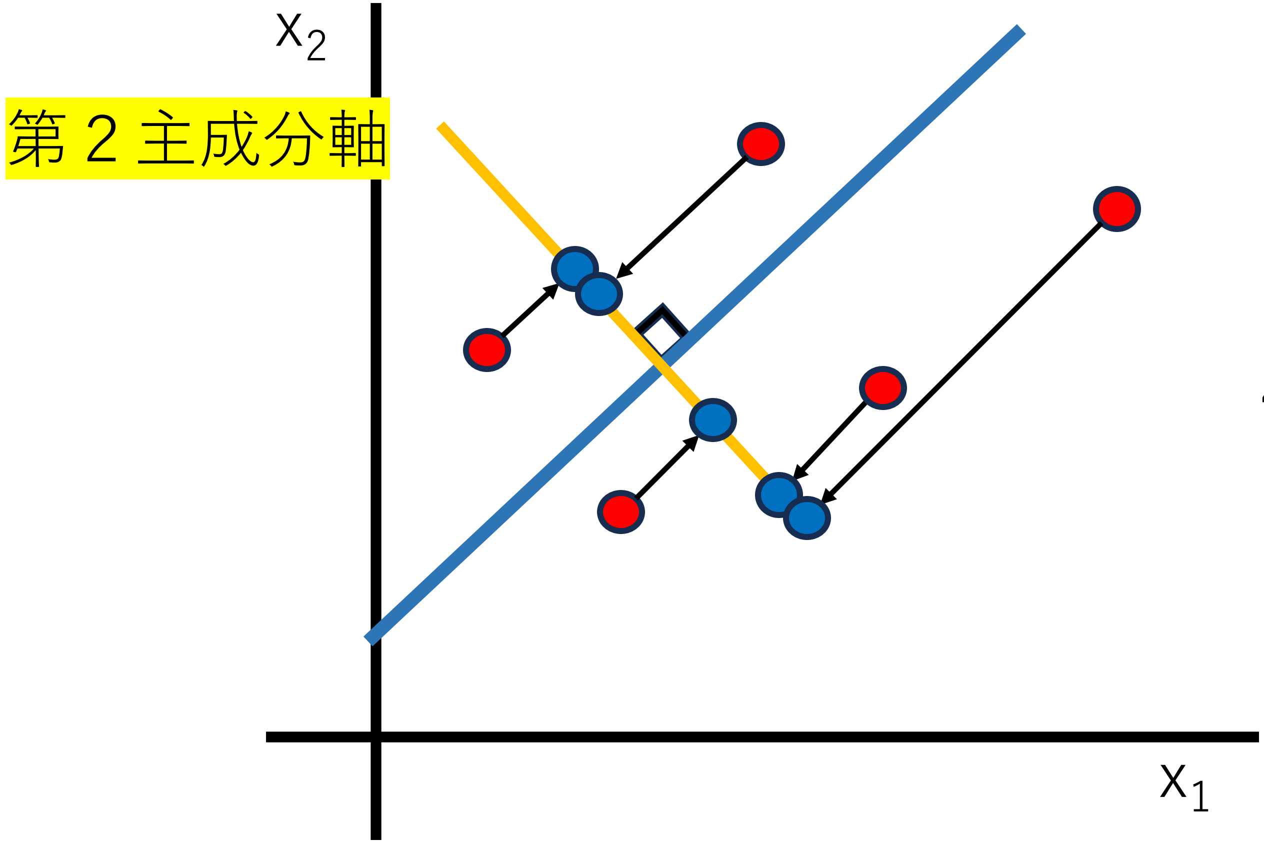
$$Z_1 = a_{11}x_1 + a_{12}x_2$$

Z ： 主成分
a ：係数
x ：成分

第２主成分以降

2次元の場合の例

第１主成分と直交する軸の中で、
軸上に射影したデータの分散が最大となる軸を探す



$$Z_2 = a_{21}x_1 + a_{22}x_2$$

Z ： 主成分
a ：係数
x ：成分

計算で軸を決める手法（超重要！）

主成分の軸

$$Z_1 = \sum_{K=1}^{p} a_{1k}x_k \quad \cdots \quad Z_m = \sum_{K=1}^{p} a_{mk}x_k$$

主成分の分散が最大の時の係数a_1,a_2を求める→主成分分析

主成分の分散の値は説明変数の
分散共分散行列の固有値λの値と一致

⇩ つまり

最大固有値に属する固有ベクトル$[a^1,a^2]^t$を求めれば
主成分分析ができる

多変数の場合の主成分分析の例

個体と変数	**x_1**	**x_2**	**…**	**x_p**
1	x_{11}	x_{21}	…	x_{p1}
2	x_{12}	x_{22}	…	x_{p2}
…	…	…		
n	x_{1n}	x_{2n}	…	x_{pn}

分散共分散行列S
を求める

$$S = \begin{bmatrix} s_1^2 & s_{12} & \cdots & s_{1p} \\ s_{12} & s_2^2 & & s_{2p} \\ \vdots & & \ddots & \vdots \\ s_{1p} & s_{2p} & \cdots & s_p^2 \end{bmatrix}$$

多変数の場合の主成分分析の例

Sの固有値λを求める

$$\begin{vmatrix} s_1^2-\lambda & s_{12} & \cdots & s_{1p} \\ s_{12} & s_2^2-\lambda & \cdots & s_{2p} \\ \vdots & & & \vdots \\ s_{1p} & s_{2p} & \cdots & s_p^2-\lambda \end{vmatrix} = 0$$

λはp個の解を持つ

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \geq \lambda_p \geq 0$$

大きいものから順に第一主成分の係数、第二主成分の固有値…というかたち

多変数の場合の主成分分析の例

$$\begin{bmatrix} s_1^2 & s_{12} & \cdots & s_{1p} \\ s_{12} & s_2^2 & & s_{2p} \\ \vdots & & \ddots & \vdots \\ s_{1p} & s_{2p} & \cdots & s_p^2 \end{bmatrix} \begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{ip} \end{bmatrix} = \lambda_i \begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{ip} \end{bmatrix}$$

※$a_{i1}^2 + a_{i2}^2 + \cdots + a_{ip}^2 = 1$

固有値を代入　固有ベクトルを導出

第一主成分　$z_1 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1p}x_p$

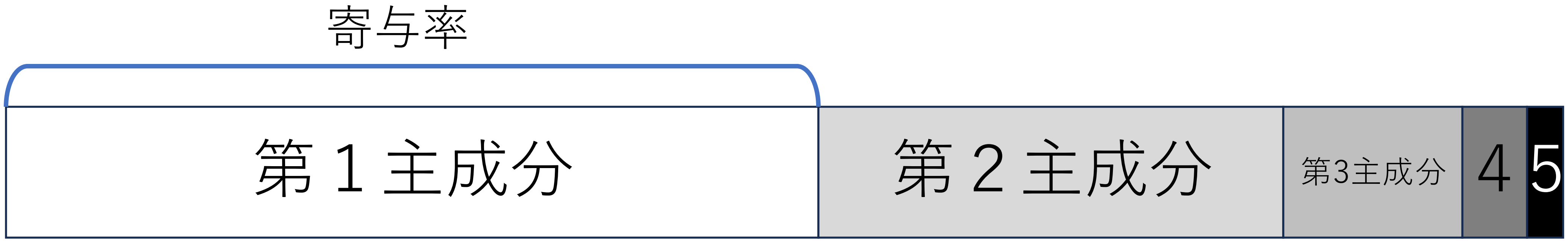
が得られ、λ_2、λ_3で行うと、第二、第三主成分が求められる

寄与率・累積寄与率

寄与率

その主成分によってデータ全体の何%を説明できるか

$$\frac{\lambda_i}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}$$

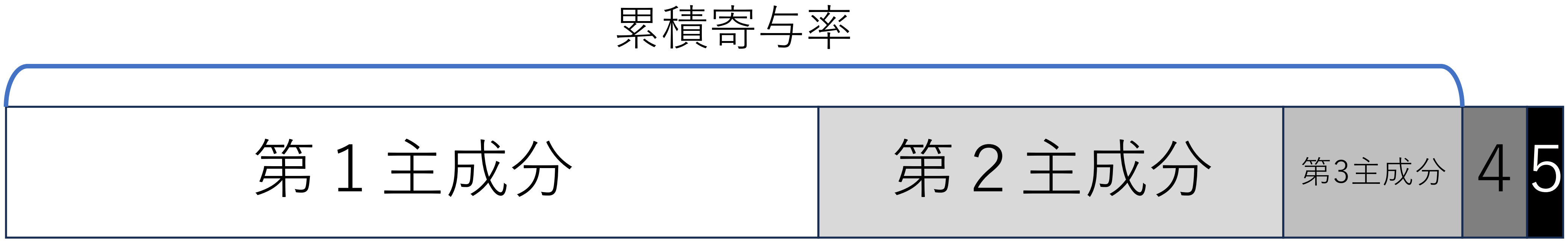


寄与率・累積寄与率

累積寄与率

第一主成分からその主成分によってデータ全体の何％を説明できるか

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_i}{\lambda_1 + \lambda_2 + \cdots + \lambda_p}$$



主成分分析（ざっくりイメージ）

＜全データ＞

要素A	要素 B	要素C	要素D	要素E

↓ 主成分分析

＜全データ＞　寄与率

第１主成分	第２主成分	第3主成分	4	5

A~E全ての要素が
大きかれ小さかれ入っている

４と５は全体への影響が小さいので無視すると、第１～３だけで全体のほとんどを表せる。
→次元削減

PCAの弱点

類似しているデータを

低次元上でも近くに保つこと

→異なるデータを低次元上でも遠くに保つこと

（分散を大きくする）に焦点を当てたアルゴリズムだから

そこで！

データの局所的な構造の維持を目的とした次元削減技術が発展

次元削減
- 主成分分析（PCA）
- SNE

クラスタリング
- k-mean

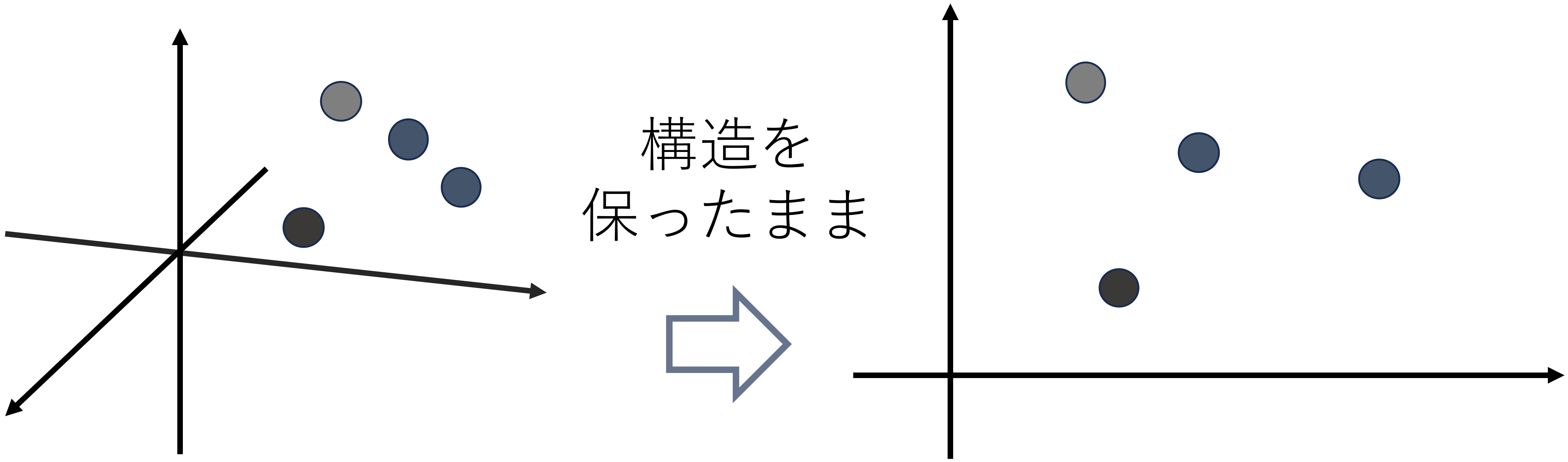
SNE

局所構造を保持しつつ

次元削減するアルゴリズム

概要

- データポイント間の「近さ」を「確率」で表す
- 高次元空間でも低次元空間でも
  点の分布はガウス分布に従うと仮定



SNEの方法（流れ）

①近傍の確率の計算

②低次元での確率の計算

③損失関数の設計

④Perplexity(困惑度)

⑤損失関数の最適化

何故確率で表すのか？

- 次元削減した後でも元の密度を保つため
- 高次元では距離だけでは不十分
  →密度が不均一だから
- 最適化に勾配降下法等のアプローチが可能

SNEの方法

①近傍の確率の計算

②低次元での確率の計算

③損失関数の設計

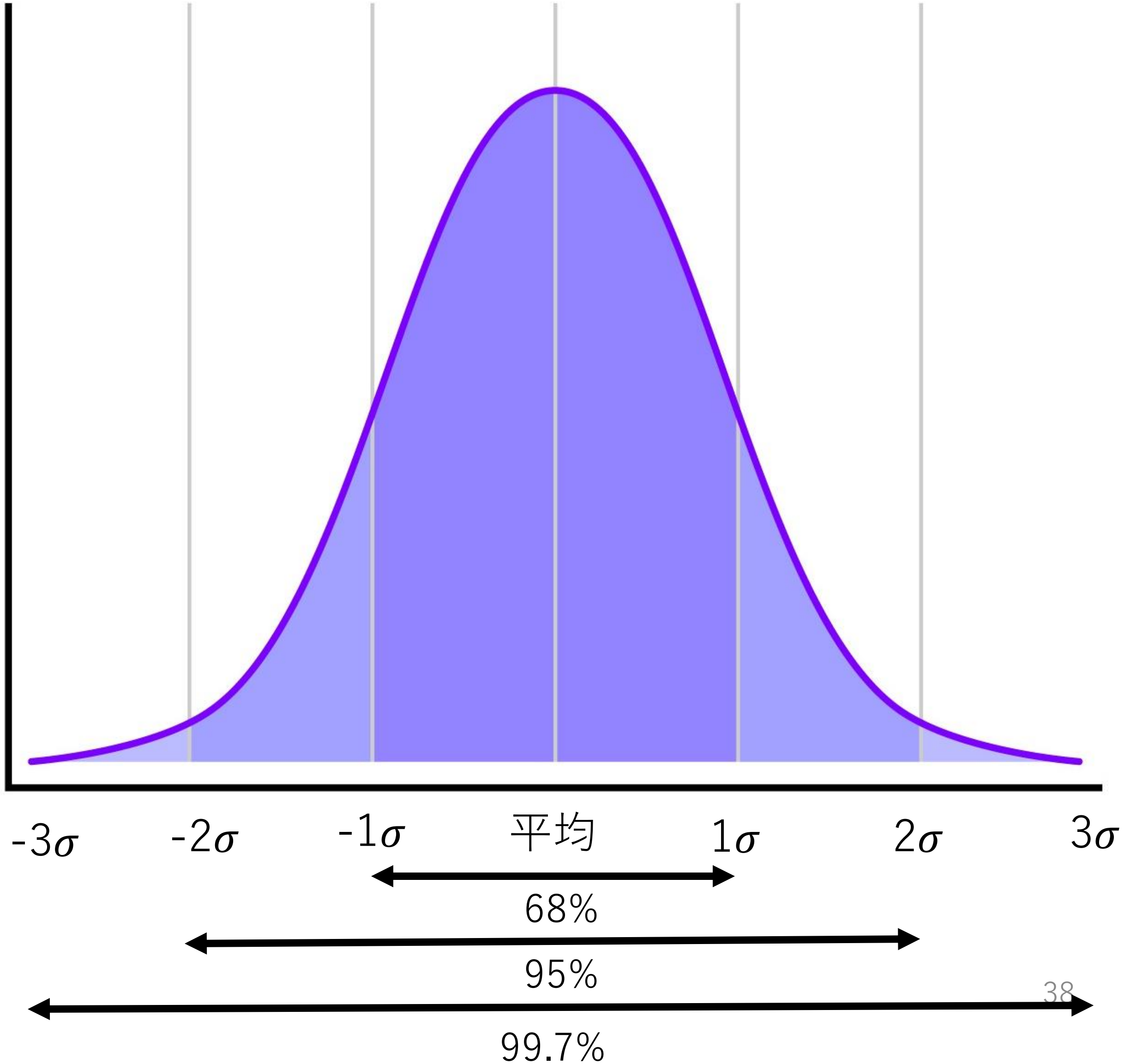
④Perplexity(困惑度)

⑤損失関数の最適化

前提知識

ガウス分布

$$\frac{1}{\sqrt{2\pi\sigma^2}}\exp(-(x-\mu)^2/2\sigma^2)$$



σ ：標準偏差
σ^2：分散
μ ：平均

①近傍の確率の計算

データポイント間の距離 → 条件付き確率．に変換

データポイントx_iとx_jの類似度を、条件付き確率$p_{j|i}$として表現

x_jはx_iを中心とした**正規分布**に基づいて選択されると仮定

$$p_{j|i} = \frac{\exp(-\frac{\|x_i - x_j\|^2}{2\sigma_i{}^2})}{\sum_{k\neq i}\exp(-\frac{\|x_i - x_k\|^2}{2\sigma_i{}^2})}$$

jが近傍である確率

全体の確率

x_i, x_j ：データポイント
分散$\sigma_i{}^2$は後述で調整
$p_{i|i} = 0$

②低次元での確率の計算

次元削減後のデータポイントも条件付き確率に変換

データポイントy_iとy_jの類似度を、条件付き確率$q_{j|i}$として表現

y_jはy_iを中心とした**正規分布**に基づいて選択されると仮定

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

y_i, y_j ：データポイント
${\sigma_i}^2$は$1/\sqrt{2}$で固定
$q_{i|i} = 0$

SNEの方法

①近傍の確率の計算

②低次元での確率の計算

③損失関数の設計

④Perplexity(困惑度)

⑤損失関数の最適化

③損失関数の設計

KLダイバージェンス　確率分布どうしの差異(距離)を測る

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

①近傍の確率の計算

データポイント間の距離 → 条件付き確率．に変換

データポイントx_iとx_jの類似度を、条件付き確率$p_{j|i}$として表現

x_jはx_iを中心とした**正規分布**に基づいて選択されると仮定

$$p_{j|i} = \frac{\exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2{\sigma_i}^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\left\|x_i - x_k\right\|^2}{2{\sigma_i}^2}\right)}$$

jが近傍である確率

全体の確率

${\sigma_i}^2$は後述で調整

$$p_{i|i} = 0$$

④Perplexity σ^2の決定

Perplexity（困惑度）の定義

$$\mathrm{Perp}(P_i) = 2^{H(P_i)}$$

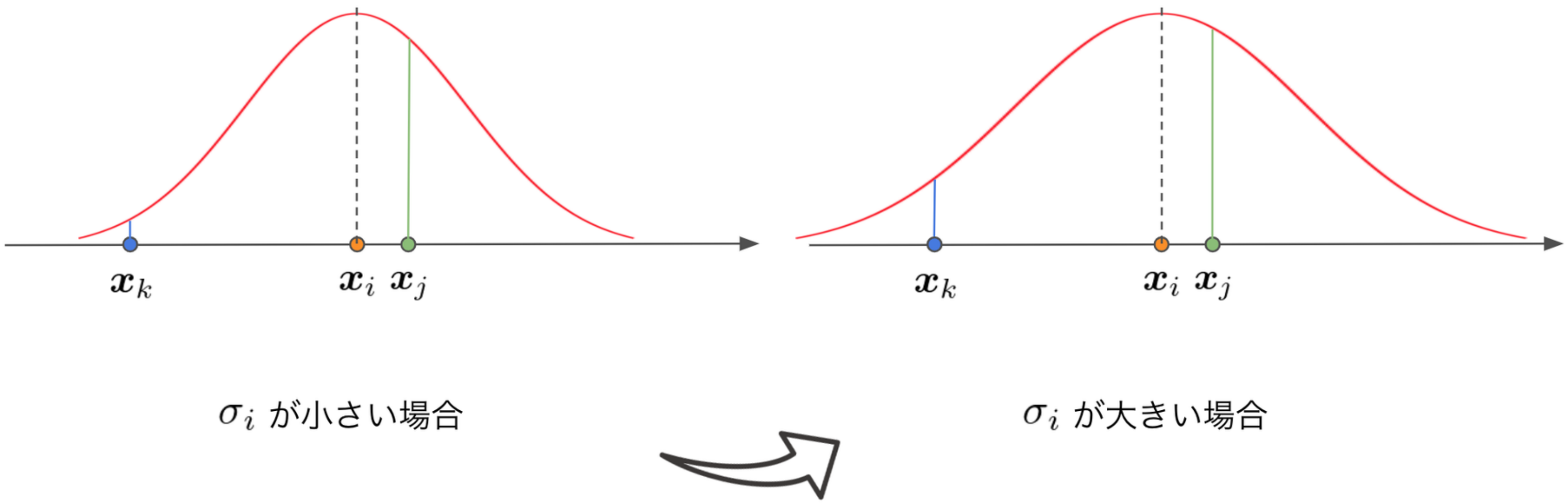
一般的にPerplexityは10～50

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$$

シャノン・エントロピーを介して
σ^2を求める

Perplexity $\begin{cases} \text{大} \quad \sigma^2\text{も大きくなる} \\ \text{小} \quad \sigma^2\text{も小さくなる} \end{cases}$

なぜ$\boldsymbol{\sigma}^2$が大事なのか



$\boldsymbol{x}_k$ の全体に占める割合が大きくなる

分散を変えると
・近い点を重視(分散小)するか
・遠い点を重視(分散大)するかを切り替えられるから

⑤損失関数の最適化

勾配降下法

$$\frac{\delta C}{\delta y_i} = 2\sum_j \left(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j}\right)(y_i - y_j)$$ 勾配

$$Y^{(t)} = Y^{(t-1)} + \eta \frac{\delta C}{\delta Y} + \alpha(t)\left(Y^{(t-1)} - Y^{(t-2)}\right)$$ 更新式

発展系：t-SNE

SNEとの相違点

1. 対称なコスト関数を使用

条件付き確率では非対称なコスト関数となり　最適化が困難になる

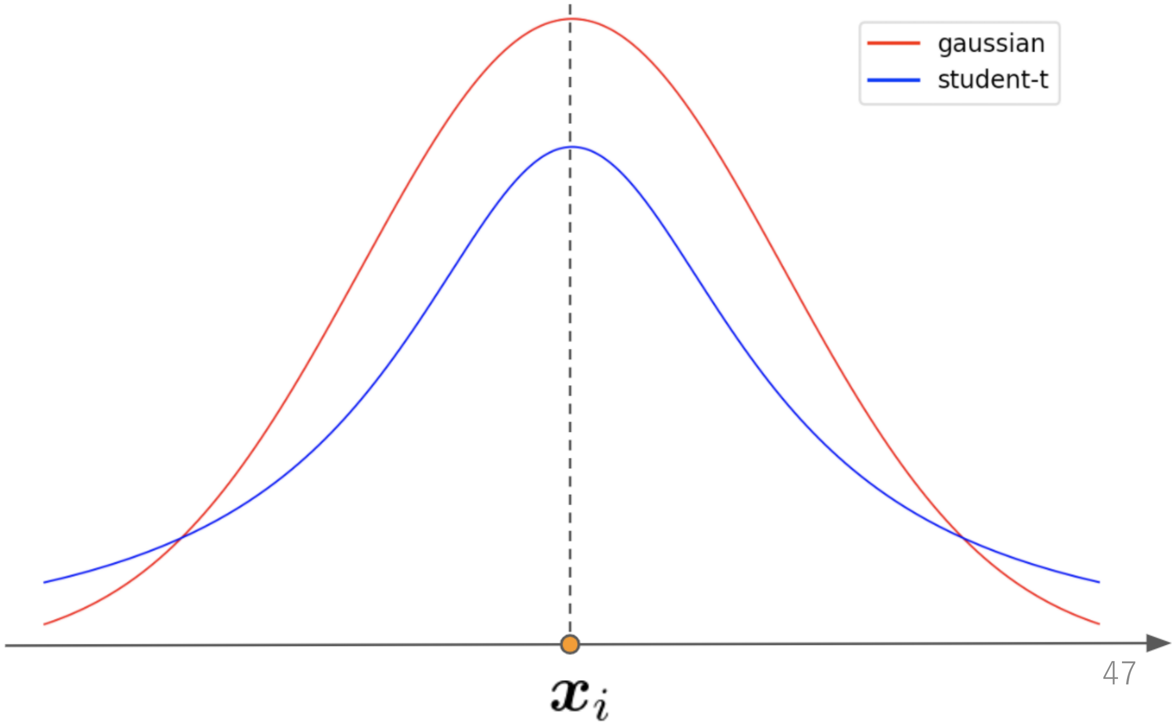
同時確率を使用することで解決

2. 圧縮後の計算に Student - t 分布を使用

ガウス分布では近い部分を重視しすぎる

必要以上に点が「混み合って」しまう

Student - t 分布を使用し緩和



発展系：t-SNE

p_{ji}には変化なし、平均を取るような処理

$$p_{ji} = \frac{p_{i|j} + p_{j|i}}{2n}$$

nはサンプル数

で同時確率にすることで対称化

t分布を使用するのでqは変化する

$$q_{ji} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k,k \neq i}(1 + \|y_i - y_j\|^2)^{-1}}$$

KLダイバージェンス

$$C = \sum_i KL(P||Q) = \sum_i \sum_j p_{ji} \log \frac{p_{ji}}{q_{ji}}$$

発展系：t-SNE

勾配降下法

$$\frac{\delta C}{\delta y_i} = 4\sum_j (p_{ji} - q_{ji})(y_i - y_j)\left(1 + \|y_i - y_j\|^2\right)^{-1}$$ 勾配

$$Y^{(t)} = Y^{(t-1)} + \eta \frac{\delta C}{\delta Y} + \alpha(t)\left(Y^{(t-1)} - Y^{(t-2)}\right)$$ 更新式

発展系：t-SNE

SNEとの相違点

1. 対称なコスト関数を使用

条件付き確率では非対称なコスト関数となり　最適化が困難になる

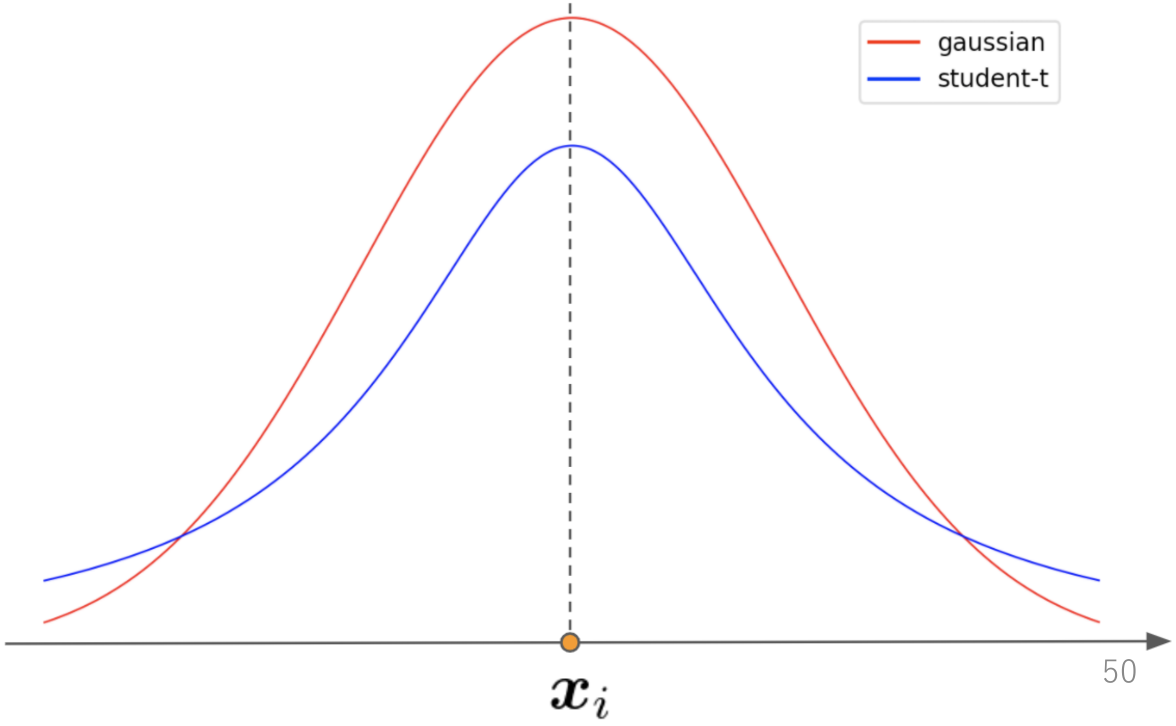
同時確率を使用することで解決

2. 圧縮後の計算に Student - t 分布を使用

ガウス分布では近い部分を重視しすぎる

必要以上に点が「混み合って」しまう

Student - t 分布を使用し緩和

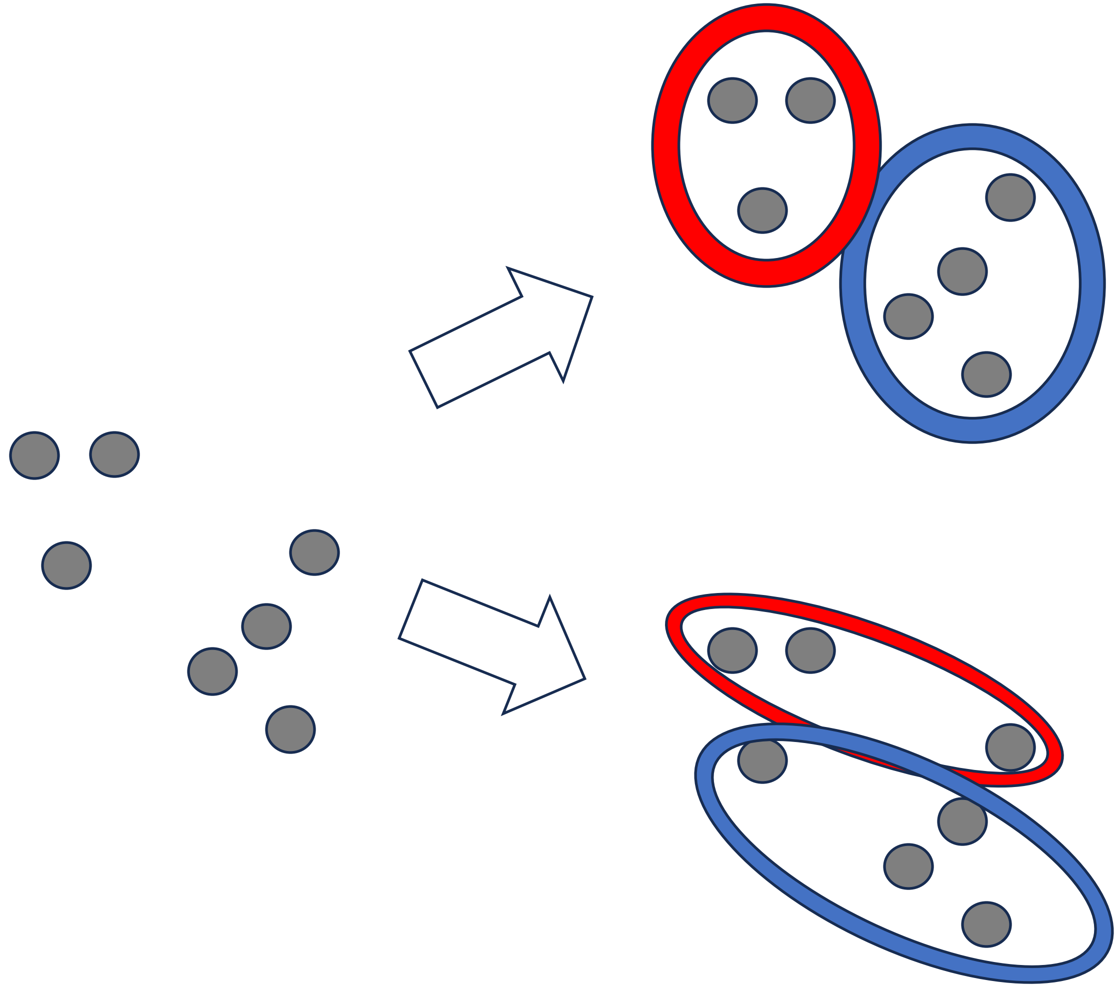


k-means（k平均法）

クラスタリングアルゴリズムの中で

最も基本的なアルゴリズム

理想的なクラスタリング（そもそも）

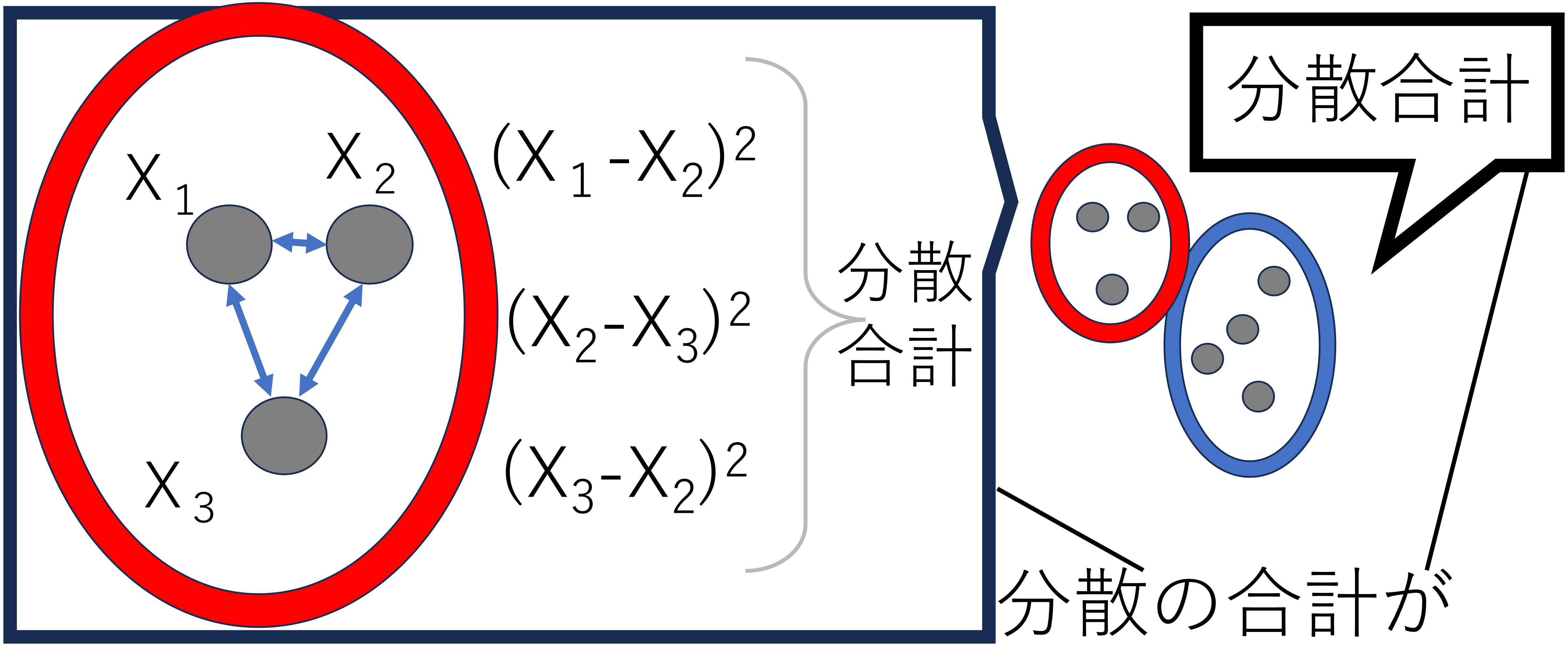


クラスタ内の分散：小

→理想に近づく

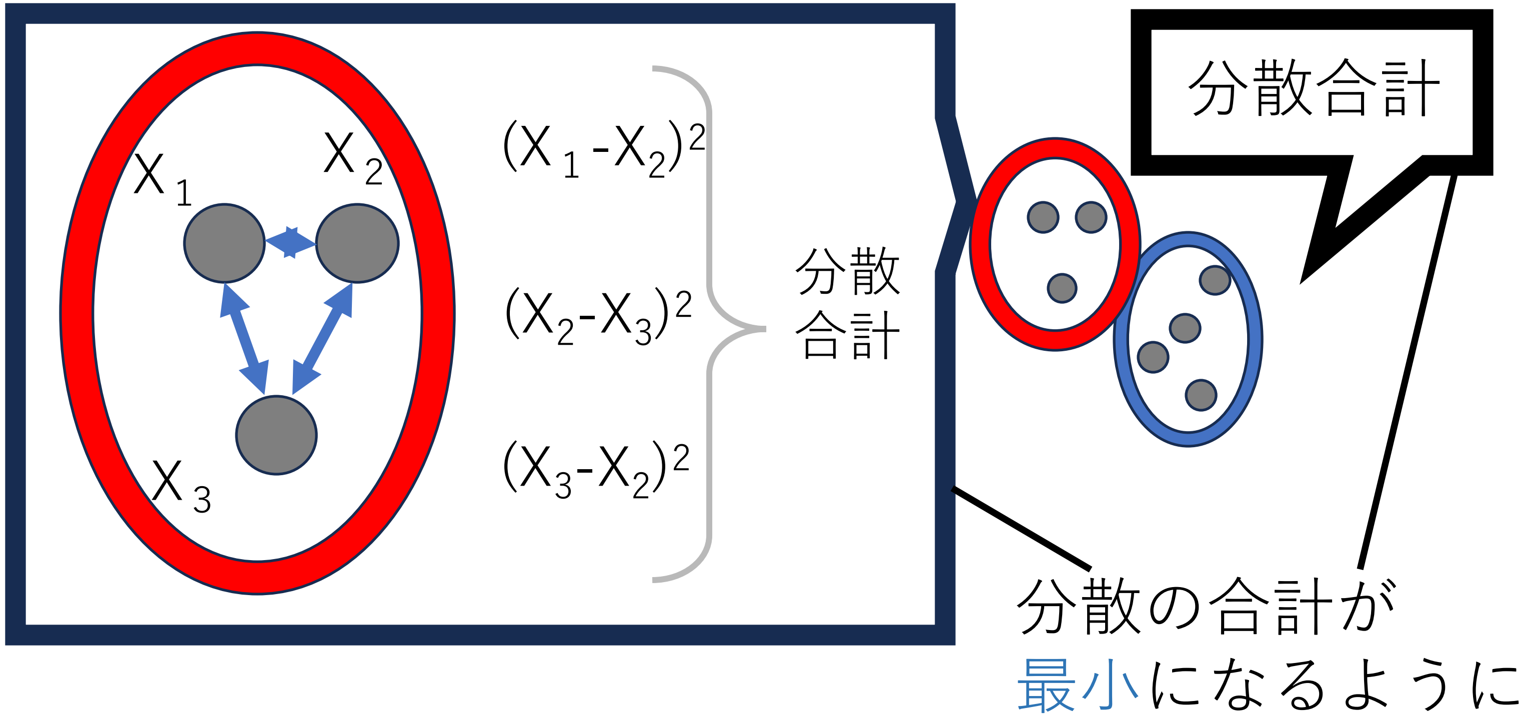
クラスタ内の分散：大

クラスタの分散の測り方



分散の合計が
最小になるように

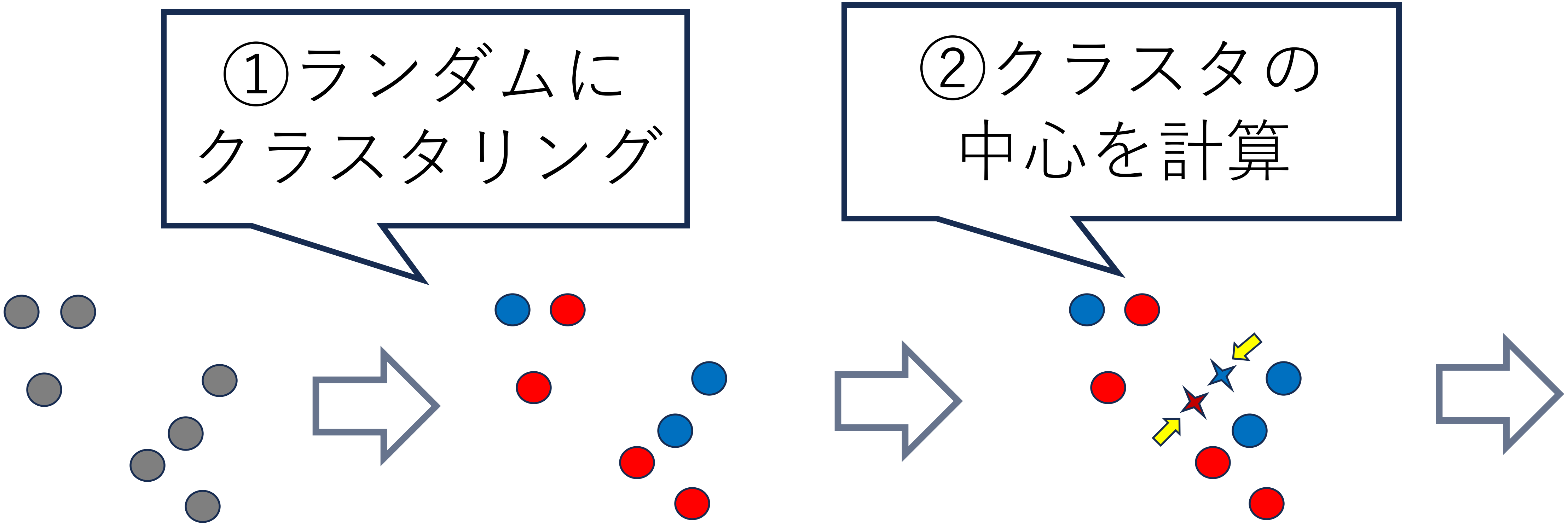
クラスタの分散の測り方



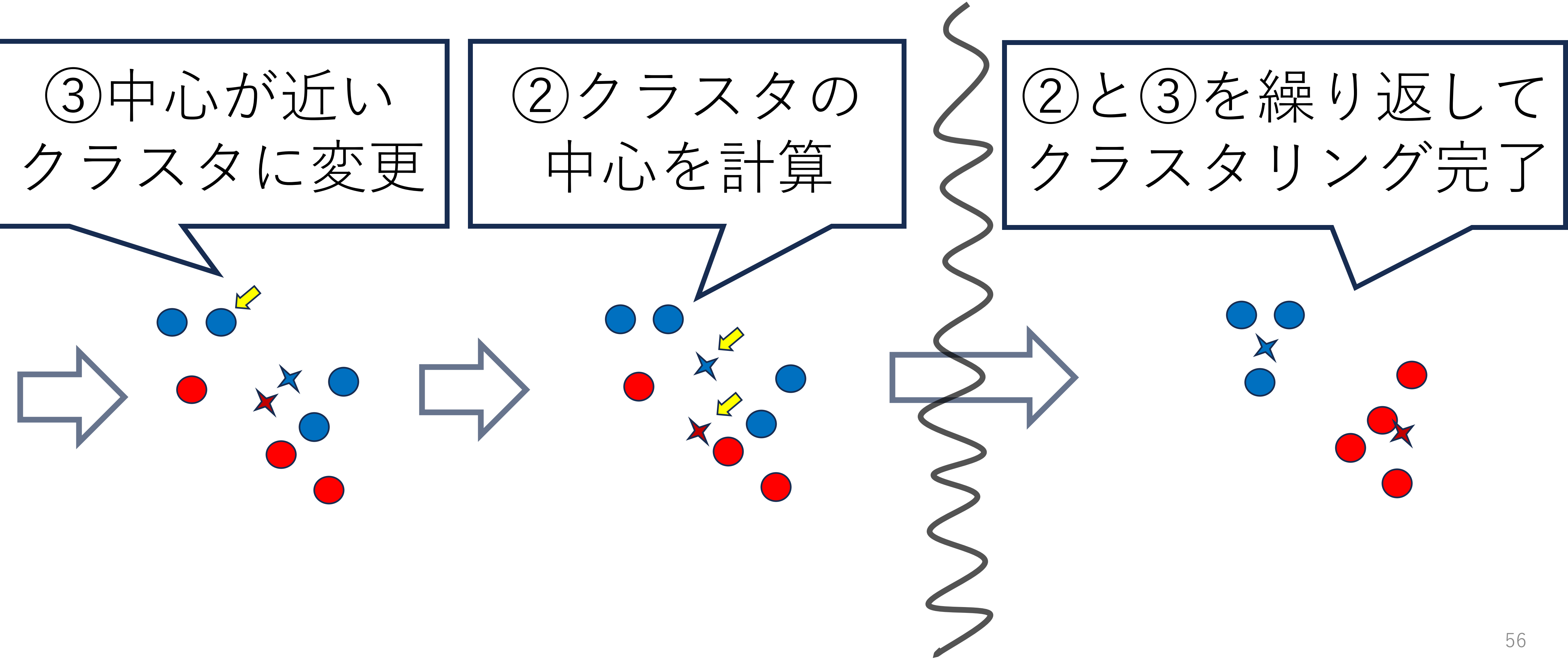
k-meansではこれの
局所解を求める

k-meansのアルゴリズム

事前に幾つに分けるか (K) だけ指示を与えておく



k-meansのアルゴリズム



k-meansの注意点

①最適な結果が得られるとは限らない
（局所解）

②**K**（クラスタ数）の決め方がとても重要

解決策

①最適な結果に近づけるために

複数回試行を行って

最も分散の合計が少ないものを採用

②Kを決めるために

データの性質等の観点から仮説を立てて決める

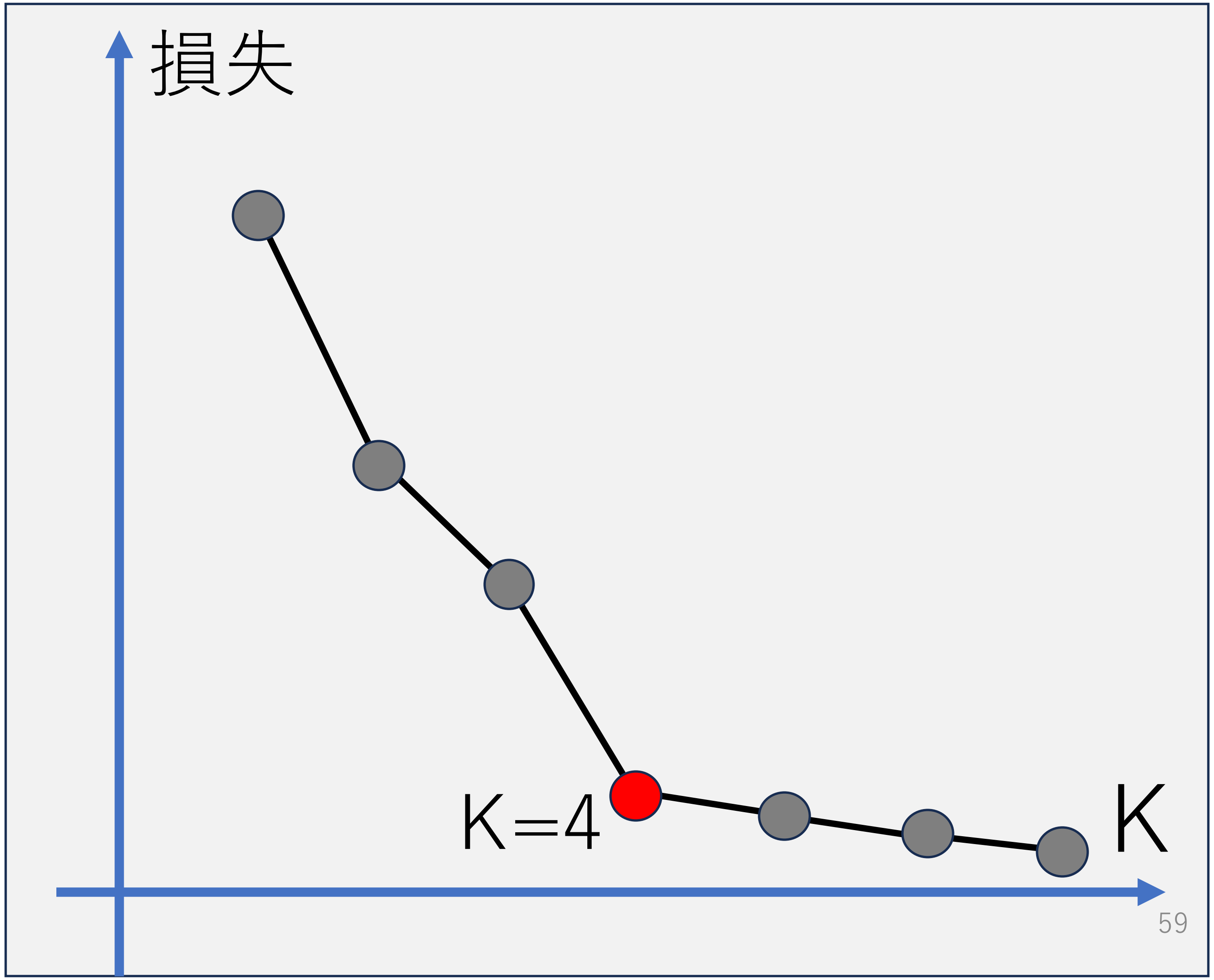
②それでもKを決められない時は

Elbow method

損失が急に下がり

それ以降緩やか

になる部分を採用



k-neighbor（k近傍法）

教師なし学習より分類等の

教師あり学習に向いている

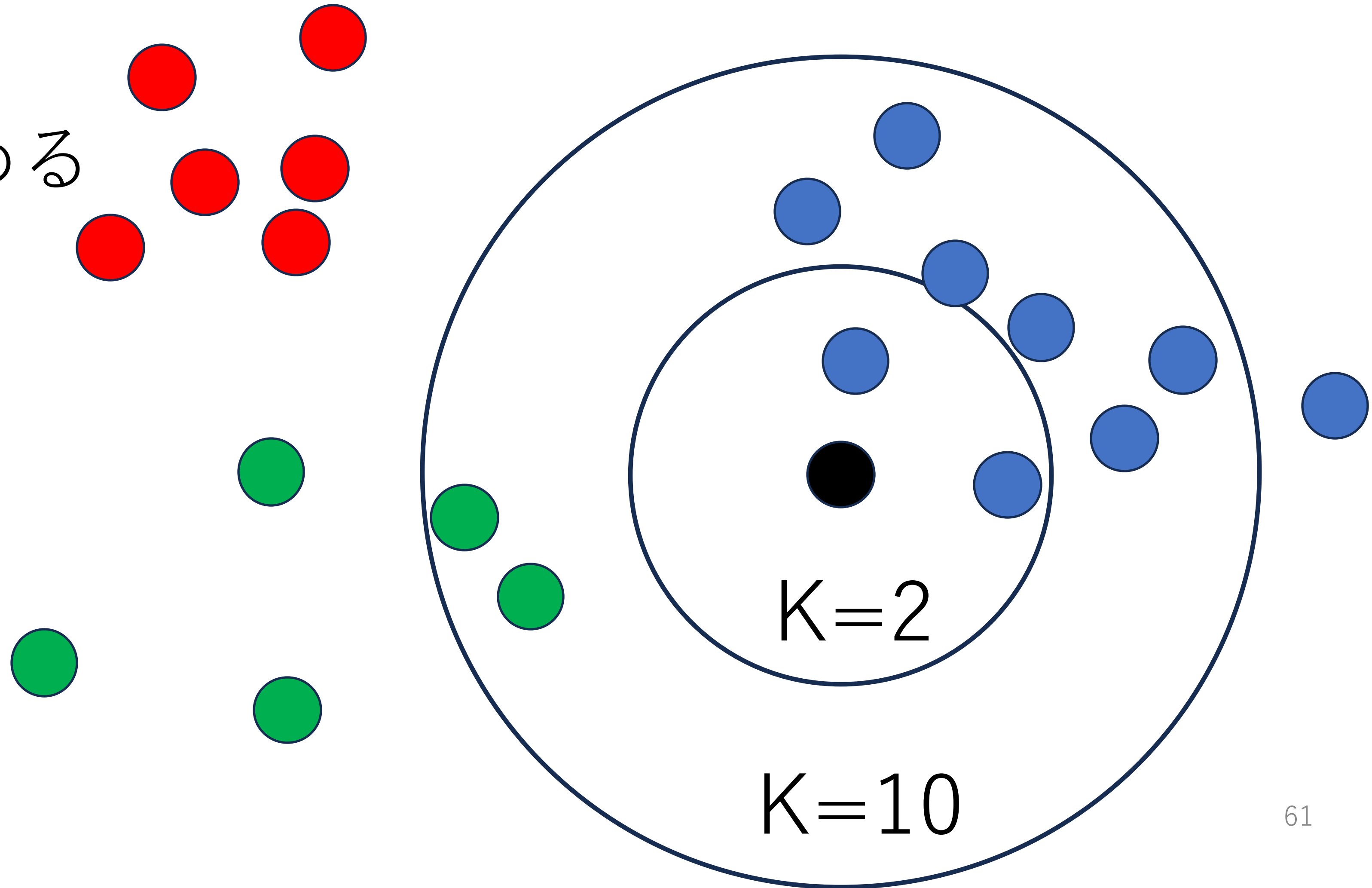
k-neighborのアルゴリズム

①Kの値を決める

②距離が近いものを求める

③多数決

※Kとは最近傍点の数のこと



高次元の距離（n次元）

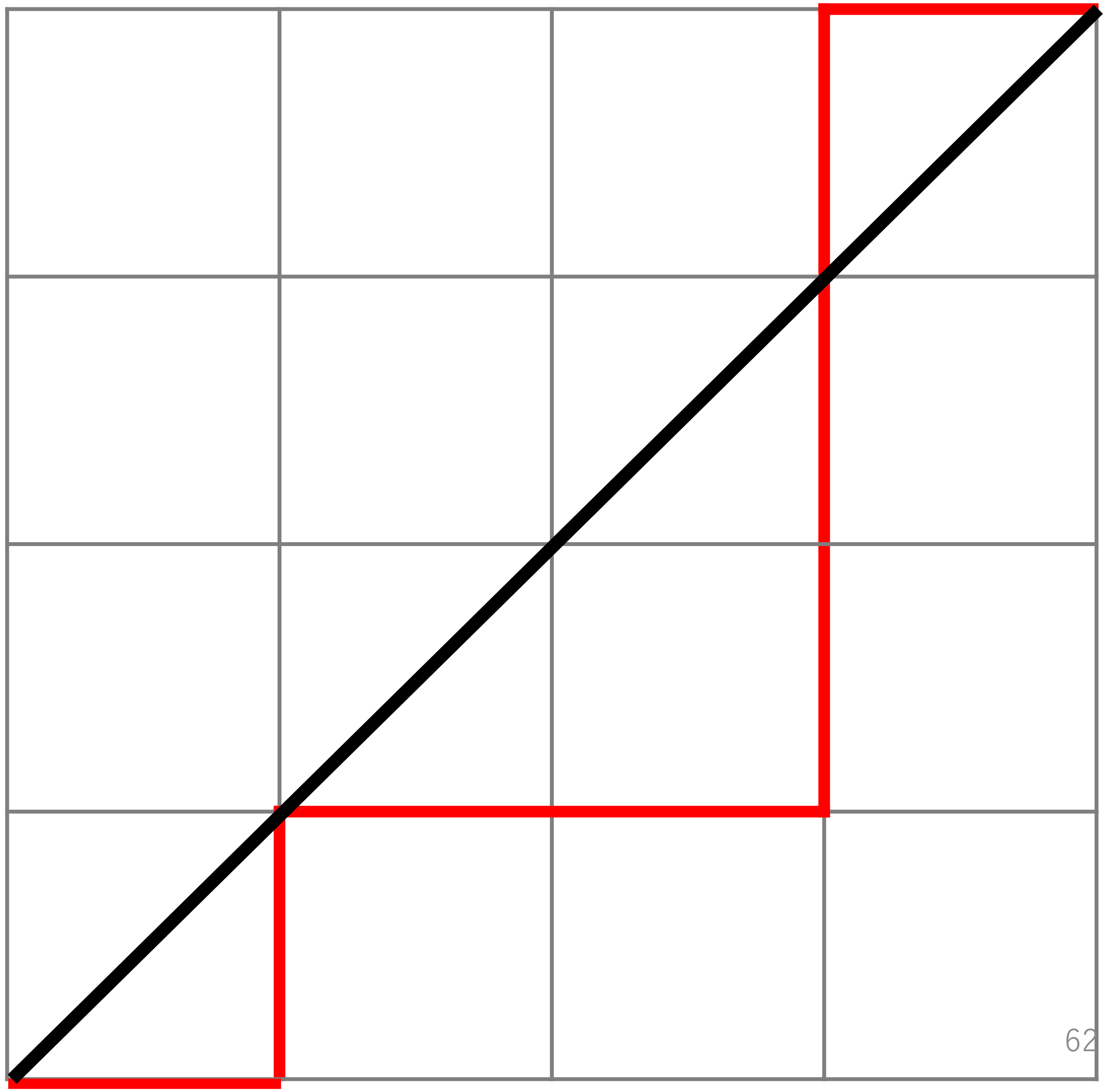
2次元の例

①ユークリッド距離

$$d(x,y)=\sqrt{\sum_{k=1}^{n}(x_k-y_k)^2}$$

②マンハッタン距離

$$d(x,y)=\sum_{k=1}^{n}|x_k-y_k|$$



k-neighbor

使い所

　①分類問題

　②データ前処理

欠点

　①データ数が多くなると計算コストが爆増

　②次元の呪い

まとめ.　教師なし学習

次元削減

・主成分分析

多次元データを少数の主成分に圧縮する分析手法

・SNE

データポイントの類似性を保ちながら次元削減する手法
発展系：t-SNE

クラスタリング

・k-mean

データを最も近いクラスタの中心に割り当てて
クラスタリングする手法

・k-neighbor

最も近いk個のデータポイントに基づいて分類する手法