知能システム（k-means 法），2018/12/14，大枝真一

1　はじめに

機械学習とは，与えられたデータからパラメータを学習し，データの背後にある規則性を見つけ出すことである．そして，学習によって得られた機械によって，未来のデータの予測や分類を行う．大量のデータを取得することが可能となり，そして大量のデータを処理することも可能となった現代において，機械学習は大きな期待と役割を担っている．

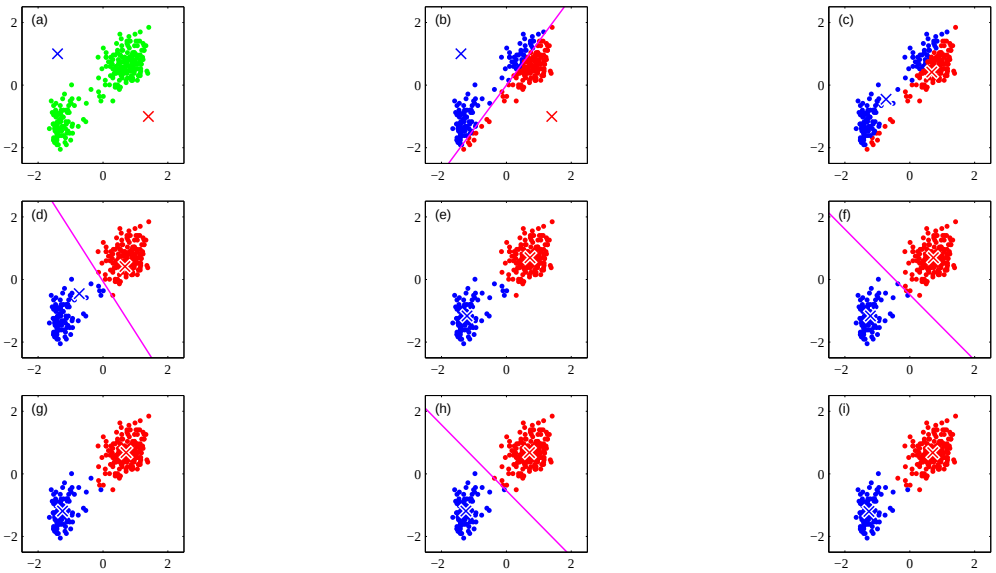
機械学習には教師あり学習と教師なし学習がある．教師あり学習としては，最小二乗法，ニューラルネットワークを学んでいる．本授業では，教師なし学習として，k-means 法を学ぶ．

2　k-means 法

k-means 法は教師なし学習によるクラスタリングの基礎となる手法である．クラスタリングとは，類似のデータをグループ化することである．例えば，大量の文書データが与えられたとき類似する文書毎にクラスタリングして自動仕分けしたり，多次元データからなる血液検査データから似た患者をクラスタリングして規則性の知識発見を行っている．k-means 法は，“似たデータ”どうしを自動的にクラスタリングする手法である．

以下，k-means 法のアルゴリズムの説明を行う．データの数を n，クラスタの数を k とおく．

1. クラスタ $V_c(c = 1, \cdots, k)$ を初期化する．
2. 各データ $\boldsymbol{x}_i(i = 1, \cdots, n)$ に対してランダムに各クラスタ $V_c(c = 1, \cdots, k)$ に割り振る．
3. 各クラスタの重心を計算する．
4. 各データ $\boldsymbol{x}_i$ と各クラスタ V_c との距離を求め，データを最も近いクラスタに割り当て直す．
5. これを繰り返し，クラスタの割り当て更新が行われなくなったら終了する．



3 実装

3.1 データ生成

まず，計算機実験ができるように対象となるデータを生成する．データの次元数は 2 次元とすると画面に表示できるため，ここでは 2 次元データとする．クラスタサイズを 3 として，そのクラスタを中心とするようなデータを生成する．

```
# 2 次元プロットデータ（3 クラス）を作成する
import numpy as np
# 各データの個数
NUM = 30
# クラスター中心
cluster1 = [0,0]
cluster2 = [0.5,0.5]
cluster3 = [0,1]
# データを正規乱数で生成
x1 = np.random.normal(cluster1[0], 0.2, NUM)
y1 = np.random.normal(cluster1[1], 0.2, NUM)
x2 = np.random.normal(cluster2[0], 0.3, NUM)
y2 = np.random.normal(cluster2[1], 0.1, NUM)
x3 = np.random.normal(cluster3[0], 0.15, NUM)
y3 = np.random.normal(cluster3[1], 0.15, NUM)

# データを結合
xx = []
xx.extend(x1)
xx.extend(x2)
xx.extend(x3)
yy = []
yy.extend(y1)
yy.extend(y2)
yy.extend(y3)
data = np.c_[xx, yy]

# ファイル出力
np.savetxt("data.csv", data, delimiter=",",fmt='%10.4f')
```

3.2 データを読み込む

生成したデータを読み込む．読み込んだデータをプロットして確認する．

```
# 2 次元プロットデータ（3 クラス）のデータを読み込んで表示する
import numpy as np
import matplotlib.pyplot as plt

# データを読み込む
data = np.loadtxt("data.csv", delimiter=",")

# データをプロット
plt.scatter(data[:,0], data[:,1], color='b', marker='o', s=20)
# グリッド表示
plt.grid(True)
# 表示
plt.show()
```

3.3 k-means 法の作成

k-means 法のアルゴリズムに従って実装する．

```
import numpy as np
# 2 点間距離を測る関数
def distance(a, b):
    dist = 0.0
    for i in range(len(a)):
        dist += (a[i] - b[i])**2
    dist = np.sqrt(dist)
    return dist

# データを読み込む
data = np.loadtxt("data.csv", delimiter=",")
val = np.array([2,3])
dist = distance(val, data[0])
print(dist)
あとは頑張る
```

3.4 k-means 法のライブラリを使う

Python には豊富な機械学習ライブラリが用意されている．手法を理解し，実装ができるレベルであるならば，わざわざ実装しなくてもライブラリを使うべきである．しかしながら，手法の理解が不十分である場合，ライブラリを利用しても実行結果の解釈はできない．まずは手法をしっかり理解し，自分自身で実装できるようになることが大切である．

```
import numpy as np
from sklearn.cluster import KMeans

# データを読み込む
data = np.loadtxt("data.csv", delimiter=",")

# k-means クラスタリング
kmeans_model = KMeans(n_clusters=3, random_state=10).fit(data)

# 分類先となったラベルを取得する
labels = kmeans_model.labels_

# ラベルを表示する
for label, dat in zip(labels, data):
    print(label, dat)

# プロットする
import matplotlib.pyplot as plt
for i in range(len(data)):
    if labels[i]==0:
        plt.scatter(data[i,0], data[i,1], color='r', marker='o', s=20)
    elif labels[i]==1:
        plt.scatter(data[i,0], data[i,1], color='g', marker='o', s=20)
    else:
        plt.scatter(data[i,0], data[i,1], color='b', marker='o', s=20)
plt.grid(True)
plt.show()
```