**データベース処理とデータ分析**

８．分類，クラスタリング

URL: http://www.kkaneko.jp/cc/dbenshu/index.html

分類（Classification）

分類とは，特徴空間におけるベクトル集合と分類ラベル集合を用いて，特徴ベクトルを分類ラベルにマッピングする識別機（classifier）を生成することである．具体例として，アヤメ（Iris）データセットでは，花の特徴ベクトル（例：(5.9, 3.0, 5.1, 1.8)）を，学習済みの識別機を用いて分類ラベル「virginica」にマッピングする．この識別機は，トレーニングデータセットとそれに対応する分類ラベル集合から学習を行う．分類は教師あり学習の一種であり，SVM（Support Vector Machine：サポートベクターマシン）などの手法を用いて実装する．

線形SVM

線形SVMは，データが線形分離可能な場合に分類を行う手法である．アヤメデータセットでは，Setosa品種を他の品種から分離する際に用いることができる．

分類（Classification）のプログラム：

import numpy as np

from sklearn import datasets

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

# アヤメデータセットの読み込み

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# データをトレーニングセットとテストセットに分割（テストデータ20%）

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# 線形SVMによる分類

svm = SVC(kernel='linear')

svm.fit(X\_train, y\_train)

y\_pred\_svm = svm.predict(X\_test)

# 精度の評価

print("SVM Accuracy:", accuracy\_score(y\_test, y\_pred\_svm))

RBFカーネルSVM

RBF（Radial Basis Function：放射基底関数）カーネルを用いたSVMは，非線形な決定境界を学習する手法である．これにより，データの複雑なパターンを捉えることができ，VersicolorとVirginica品種の分離に用いることができる．

import numpy as np

from sklearn import datasets

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

# アヤメデータセットの読み込み

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# データをトレーニングセットとテストセットに分割（テストデータ20%）

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# RBFカーネルSVMによる分類

kernel\_svm = SVC(kernel='rbf', gamma='scale')

kernel\_svm.fit(X\_train, y\_train)

y\_pred\_kernel\_svm = kernel\_svm.predict(X\_test)

# 精度の評価

print("Kernel SVM Accuracy (RBF):", accuracy\_score(y\_test, y\_pred\_kernel\_svm))

クラスタリング（Clustering）

クラスタリングは，データのラベルが未知の場合に，特徴空間内のベクトルのみを用いてデータを自然なグループに分類する教師なし学習手法である．その目的は，データの内部構造を発見し，類似したデータポイントを同一クラスタにグループ化することである．k-means法などのアルゴリズムは，データが球状に分布する場合に用いることができる．

k-meansクラスタリングのプログラム：

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

# アヤメデータセットの読み込み

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# k-means法によるクラスタリング（クラスタ数3）

kmeans = KMeans(n\_clusters=3, random\_state=42)

y\_pred\_kmeans = kmeans.fit\_predict(X)

# PCA（Principal Component Analysis：主成分分析）による2次元への次元削減と可視化

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X)

plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y, cmap='viridis', marker='o', edgecolor='k', s=50)

plt.title("True Labels")

plt.subplot(1, 2, 2)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y\_pred\_kmeans, cmap='viridis', marker='o', edgecolor='k', s=50)

plt.title("K-means Clustering")

plt.show()

GMM（Gaussian Mixture Model：ガウス混合モデル）の特徴

GMMは，データを複数のガウス分布の混合としてモデル化する手法である．主な特徴は以下の通りである：

・柔軟なクラスタ形状：各クラスタをガウス分布でモデル化することで，楕円形など複雑な形状のクラスタを表現することができる．

・確率的割り当て：データポイントのクラスタ所属を確率的に決定するため，明確な境界を持たないデータにも対応することができる．

・事前知識の活用：各ガウス分布の平均や共分散行列を事前に設定することで，ドメイン知識を反映したクラスタリングを行うことができる．

GMMによるクラスタリングのプログラム：

import numpy as np

import matplotlib.pyplot as plt

from sklearn import datasets

from sklearn.mixture import GaussianMixture

from sklearn.decomposition import PCA

# アヤメデータセットの読み込み

iris = datasets.load\_iris()

X = iris.data

y = iris.target

# GMMによるクラスタリング（クラスタ数3）

gmm = GaussianMixture(n\_components=3, random\_state=42)

y\_pred\_gmm = gmm.fit\_predict(X)

# PCAによる2次元への次元削減と可視化

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X)

plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y, cmap='viridis', marker='o', edgecolor='k', s=50)

plt.title("True Labels")

plt.subplot(1, 2, 2)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=y\_pred\_gmm, cmap='viridis', marker='o', edgecolor='k', s=50)

plt.title("GMM Clustering")

plt.show()