*VII-3-4-2. 混合ガウスモデル(#scikit -learn、GMMによる非階層的クラスター分析)*

VII-3-4-2-i。準備とデータの読み込み

|  |
| --- |
| #Ssci-kit -learn、GMMによる非階層的クラスター分析  #必要なライブラリーのインポート  #[A]必要なライブラリーの読み込み  import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  %matplotlib inline  from sklearn.model\_selection import train\_test\_split  from scipy.cluster.hierarchy import linkage,dendrogram,fcluster  from sklearn import cluster, preprocessing,mixture  import decimal  decimal.getcontext().prec=6  #[B]データの読み込み  df =pd.read\_csv("sample10.csv")  #データフレームをつくる  dfX=pd.DataFrame(df)  X0=df.values  X=np.delete(X0,0,1) |

VII-3-4-2-ii. 混合ガウスモデルを作り実行

|  |
| --- |
| X=X  x1=1 #作図する平面の選択x軸  x2=2 #作図する平面の選択y軸  x1=x1-1  x2=x2-1  x=X[:,x1]  y=X[:,x2]  nc=5 #クラスター数の決定  #実行  gmm=mixture.GaussianMixture(n\_components=nc,covariance\_type='full',max\_iter=250,n\_init=10,init\_params='kmeans')  z\_gmm=gmm.fit(X)  predict1=z\_gmm.predict(X)  print(predict1) #クラスの判別  print(gmm.means\_) #各クラスの重心  print(gmm.covariances\_) #各クラスの分散共分散行列  print(gmm.weights\_) #混合係数  print(gmm.aic(X)) #赤池情報基準  print(gmm.bic(X)) #ベイズ情報基準  print(gmm.score(X))#平均対数尤度  #結果を図示  #平面を決定  plt.figure (1,figsize=(4,4))  plt.scatter(x,y,c=predict1)  plt.show() |

VII-3-4-2-iii.元データのクラス分けを図示。

|  |
| --- |
| #元のクラス分け  X=X  x1=1  x2=2  x1=x1-1  x2=x2-1  x=X[:,x1]  y=X[:,x2]  z=X0[:,0]  plt.figure (1,figsize=(4,4))  plt.scatter(x,y,c=z)  plt.show() |

VII-3-4-2-iv.K-means法によるクラス分け図示

|  |
| --- |
| X=X  x1=1  x2=2  x1=x1-1  x2=x2-1  x=X[:,x1]  y=X[:,x2]  km=cluster.KMeans(n\_clusters=5)  z\_km=km.fit(X)  plt.figure(1,figsize=(4,4))  plt.scatter(x,y,c=z\_km.labels\_)  plt.scatter(z\_km.cluster\_centers\_[:,x1],z\_km.cluster\_centers\_[:,x2],s=250,marker='\*',c=[0,1,2,3,4])  plt.show()  print(z\_km.cluster\_centers\_) |

リストVII-3-4-2-v.作図のための関数定義

|  |
| --- |
| #関数の定義  #ガウス関数の定義  def gauss(x,mu,sigma):  N,D=x.shape  c1=-(D/2)\*np.log(2\*np.pi)  det\_sigma=np.linalg.det(sigma)  c2=-(1/2)\*np.log(det\_sigma)  inv\_sigma=np.linalg.inv(sigma)  c3=x-mu  c4=np.dot(c3,inv\_sigma)  c5=np.zeros(N)  for d in range(D):  c5=c5+c4[:,d]\*c3[:,d]  c5=-c5/2  p=c1+c2+c5  p=np.exp(p)  return p  #混合ガウスモデル  def mixgauss(x,pi,mu,sigma):  N,D=x.shape  K=len(pi)  p=np.zeros(N)  for k in range(K):  p=p+pi[k]\*gauss(x,mu[k,:],sigma[k,:,:])  return p  #個々入力データの尤度  def likelihood(xx,mu,pi,sigma):  N,D=xx.shape  ppk1=np.zeros((N,K))  ppl1=np.zeros((N,K))  g1=np.zeros((N,K))  S1=np.zeros((N))  for k in range(K):  g1[:,k]=gauss(xx,mu[k,:],sigma[k,:,:] )  ppk1[:,k]=pi[k]\*g1[:,k]  for n in range(N):  for k in range(K):  S1[n]=S1[n]+ppk1[n,k]  for k in range(K):  ppl1[n,k]=g1[n,k]\*ppk1[n,k]/S1[n]  ratio1=np.zeros((N,K))  for n in range(N):  SS1=0  for k in range(K):  SS1=SS1+ppl1[n,k]  for k in range(K):  ratio1[n,k]=ppl1[n,k]/SS1  return g1,ppk1,ppl1,ratio1  #データの図示  #混合ガウス等高線表示  import matplotlib.pyplot as plt  from mpl\_toolkits.mplot3d import axes3d  %matplotlib inline  def show\_contour\_mixgauss(pi,mu,sigma):  xn=40 #解像度  x0=np.linspace(X\_range0[0],X\_range0[1],xn)  x1=np.linspace(X\_range1[0],X\_range1[1],xn)  xx0,xx1=np.meshgrid(x0,x1)  A=xx0.reshape(xn\*xn,1)  B=xx1.reshape(xn\*xn,1)  x=np.c\_[B,A]  f=mixgauss(x,pi,mu,sigma)  f=f.reshape(xn,xn)  f=f.T  plt.contour(x0,x1,f,10,color="grey")  #混合ガウス3D表示  def show3d\_mixgauss(ax,pi,mu,sigma):  xn=40 #解像度  x0=np.linspace(X\_range0[0],X\_range0[1],xn)  x1=np.linspace(X\_range1[0],X\_range1[1],xn)  xx0,xx1=np.meshgrid(x0,x1)  A=xx0.reshape(xn\*xn,1)  B=xx1.reshape(xn\*xn,1)  x= np.c\_[B,A]  f=mixgauss(x,pi,mu,sigma)  f=f.reshape(xn,xn)  f=f.T  ax.plot\_surface(xx0,xx1,f,rstride=2,cstride=2,alpha=0.3,color='blue',edgecolor='black') |

VII-3-4-2-vi.等高線図と３D確率分布

|  |
| --- |
| Pi2=gmm.weights\_  Mu2=gmm.means\_  Sigma2=gmm.covariances\_  X\_range0=[-2,2] #項目1の範囲  X\_range1=[-2,2] #項目2の範囲  #混合ガウス関数（等高線と3d)  #等高線図と3d  Fig=plt.figure(1,figsize=(8,3.5))  Fig.add\_subplot(1,2,1)  show\_contour\_mixgauss(Pi2,Mu2,Sigma2)  plt.grid(True)  Ax=Fig.add\_subplot(1,2,2,projection='3d')  show3d\_mixgauss(Ax,Pi2, Mu2,Sigma2)  Ax.set\_xlabel('$x\_0$',fontsize=14)  Ax.set\_ylabel('$x\_1$',fontsize=14)  Ax.view\_init(40,-60)  plt.xlim(X\_range0)  plt.ylim(X\_range1)  plt.show() |

VII-3-4-2-vii.所属クラスの判別

|  |
| --- |
| Y=[[0.5,0.5],[-0.6,0.4],[-0.3,-0.6],[0.6,-0.3],[0,0]]  predict2=z\_gmm.predict(Y)  print(predict2) |

VII-3-4-2-viii. 任意の座標のクラスに属する尤度と確率

|  |
| --- |
| #  x=[[-0.21877,-0.28051],[-0.6,0.4],[-0.3,-0.6],[0.6,-0.3],[0.0]]  N=len(x)  D=len(x[0])  xx=np.zeros((N,D))  for n in range(N):  xx[n]=x[n]  K=nc  Gamma2,Ppk2,Ppl2,Ratio2=likelihood(xx,Mu2,Pi2,Sigma2)  print(Ppl2)  print(Ratio2) |