EHB328 - Birinci Odev Ikinci Kisim

Gauss Dagilimina Gore Veriyi Olusturmak

In [1]:
```
import numpy as np
import math  as mt
```

In [14]:
```
def GenerateGaussianData(N,aves,covs):
    #gets number of features
    #used in the rest of the code
    no_of_features=np.size(aves)

    #av_holder=np.zeros(no_of_features)
    # holds averages for features
    #for i in loc:
    #    av_holder=i
    #since I decided to do each class separately
    #I removed this part


    std_dev_holder=np.zeros(no_of_features)
    #holds standard deviations for features
    for i in range(no_of_features):
        std_dev_holder[i]=mt.sqrt(covs[i,i])

    #N sample size specified
    #samples from Gaussian distribution
    samples=np.zeros((N,no_of_features))
    for i in range(N):
        for l in range(no_of_features):
            samples[i,l]=np.random.normal(aves[l],std_dev_holder[l])

    return samples
```

In [18]:
```
#Distribution Parameters for first Class
mean_c1=np.array([3,2])
cov_c1=np.array([[0.5,0],[0,0.5]])
```

In [24]:
```
#we want a hundred samples
N=100
```

In [25]:
```
#sample the Gaussian distribution
#hundred times for the first class
samples_c1=GenerateGaussianData(N,mean_c1,cov_c1)
#print(samples_c1)
```

In [ ]:
```
#Distribution Parameters for second class
mean_c2=np.array([5,4])
cov_c2=np.array([[1,0],[0,1]])
```

In [27]:
```
#Sample Gaussian for the second class
samples_c2=GenerateGaussianData(N,mean_c2,cov_c2)
#print(samples_c2)
```

Ayirt Edici Fonksiyonu Tanimlamak

P(C1)=P(C2) olarak siniflarin gelme olasiliklari birbirlerine esittir diye varsaydim.

In [ ]:
```
#taking each class to have equal probabilities
P_C=0.5
```

Her iki sinifin da kovaryans matrislerinin kosegen disi elemanlarini sifir oldugu icin ders notlarindaki ayirt edici fonksiyonu (discriminant function) kullandim.

Quadratic discriminant analysis (QDA) is closely related to linear discriminant analysis (LDA), where it is assumed that the measurements from each class are normally distributed. Unlike LDA however, in QDA there is no assumption that the covariance of each of the classes is identical. When the normality assumption is true, the best possible test for the hypothesis that a given measurement is from a given class is the likelihood ratio test. https://en.wikipedia.org/wiki/Linear_discriminant_analysis https://en.wikipedia.org/wiki/Quadratic_classifier#Quadratic_discriminant_analysis

Not: Kovaryans matrisleri birbirlerine esit olmadigi icin bir sonraki paragrafta ekledigim Vikipedi baglantilarindaki fonksiyonlari kullanmam gerektigini dusunmustum. Fakat o sayfalarda da bahsettigi uzere asil yapilan islem sonsal olasik dagilimi (P(C|x)) kullanilarak secim yapilmasi. Bu sebeple ders notlarindaki fonksiyonu kullanarak hata yapmadigimi dusunuyorum.

In [ ]:
```
def discriminant_fnc(data_2_predict,ave,cov_matrix,N,P_C):
    data_transposed=np.reshape(data_2_predict,(2,1))
    dot_result=np.dot(data_2_predict-ave,cov_matrix)
    intermediary=data_transposed-np.reshape(ave,(2,1))
    dot_result=np.dot(dot_result,intermediary)

    g= (-1/2)*dot_result + (-1/2)*N*np.log(np.linalg.det(cov_matrix)) + np.log(P_C)

    return g
```

"np.reshape" kullanilmasinin sebebi formulde oldugu sekliyle ic carpabilmektir. "np" kutuphanesi ile tanimladigimiz arrayler "1x..." seklinde oluyor. Ic carpimda ise "1x2""*2x2*""2x1" olmasi gerekiyor. Sonuncu "2x1" vektoru elde etmek icin averaj matrisini "np.reshape" ile tekrar sekillendiriyoruz.

Karar verici fonksiyon ayni veriyi kullanarak iki sinifa da ait olan ayirt edici fonksiyonlari ayri ayri cagirarak, verinin hangi sinifa ait olduguna karar verir. Hangi sinifa ait ayirt edici fonksiyonun geri getirdigi g degeri daha yuksekse verinin o sinifa ait olduguna karar verilir.

Simdi onceki bolumde urettigimiz verinin hepsi icin karar verici fonksiyonu cagiracagiz.

In [ ]:
```

```