

**EGE UNIVERSITY**

**FACULTY OF ENGINEERING**

**COMPUTER ENGINEERING DEPARTMENT**

**204 DATA STRUCTURES (3+1)**

**2020–2021 FALL SEMESTER**

**PROJECT-1 REPORT**

**(Arrays, Matrices, Methods, Random Numbers)**

**DELIVERY DATE**

14/12/2020 23:59

**PREPARED BY**

05180000037, Sinan DÖŞEYİCİ

İçindekiler

[1) POINTS IN A 2D PLANE 2](#_Toc57390051)

[1.a Rastgele Nokta Üretimi 2](#_Toc57390052)

[1.a.1 Kodlar 2](#_Toc57390053)

[1.a.2 Ekran görüntüleri 2](#_Toc57390054)-3-4

[1.a.3 Açıklama 4](#_Toc57390055)

[1.b Uzaklık Matrisi 4](#_Toc57390056)

[1.b.1 Kodlar 4-5](#_Toc57390057)

[1.b.2 Ekran görüntüleri 5](#_Toc57390058)

[1.b.3 Açıklama 5](#_Toc57390059)

[2) CLASSIFICATION USING K-NEAREST NEIGHBORS (KNN) ALGORITHM 5](#_Toc57390060)

[2.a KNN ile sınıflandırma 5-6](#_Toc57390061)

[2.a.1 Algoritma sözde-kodu 6](#_Toc57390062)

[2.b Banknot sınıflandırma 6](#_Toc57390063)

[2.b.1 Kodlar 6](#_Toc57390064)

[2.b.2 Ekran görüntüleri 6-7](#_Toc57390065)

[2.b.3 Açıklama 7](#_Toc57390066)

[2.c Başarı ölçümü 7](#_Toc57390067)

[2.c.1 Kodlar 7](#_Toc57390068)

[2.c.2 Ekran görüntüleri 8](#_Toc57390069)

[2.c.3 Açıklama 9](#_Toc57390070)

[2.d Listeleme 9](#_Toc57390071)

[2.d.1 Kodlar 9](#_Toc57390072)

[2.d.2 Ekran görüntüleri 10-11](#_Toc57390073)

[2.d.3 Açıklama 12](#_Toc57390074)

# 1) POINTS IN A 2D PLANE

//The platform, version, and programming language used

## 1.a Rastgele Nokta Üretimi

### 1.a.1 Kodlar

class Program

{

static double[,] Nokta(Random random,int n, int height, int widht) // rastgele oluşturulan sayıların iki boyutlu listede değerlerinin atanması

{

double[,] noktalar = new double[height, widht];

for (int i = 0; i < n; i++)

{

noktalar[i, 0] = random.NextDouble()\*widht; //x koordinatı

noktalar[i, 1] = random.NextDouble()\*height; //y koordinatı

Console.WriteLine("Nokta: " + noktalar[i, 0] + " , " + noktalar[i, 1]);

}

return noktalar;

}

}

static void Main(string[] args)

{

Random random = new Random();

Console.WriteLine(" X koordinatı Y koordinatı");

Console.WriteLine("\n--------------n=10 için olan değerler--------------");

double[,] noktalar1 = Nokta(random,10, 100, 100);

Console.WriteLine("\n--------------n=100 için olan değerler-------------");

double[,] noktalar2 = Nokta(random,100, 100, 100);

Console.ReadKey();

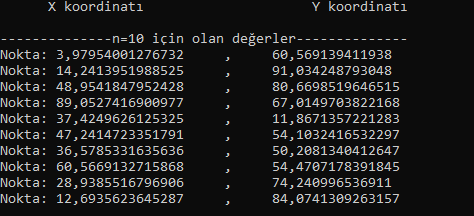
}

}

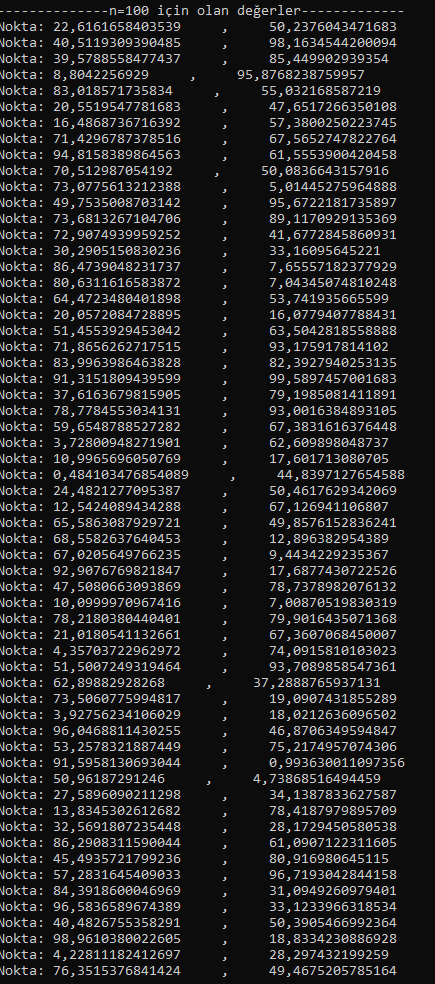
}

### 1.a.2 Ekran görüntüleri

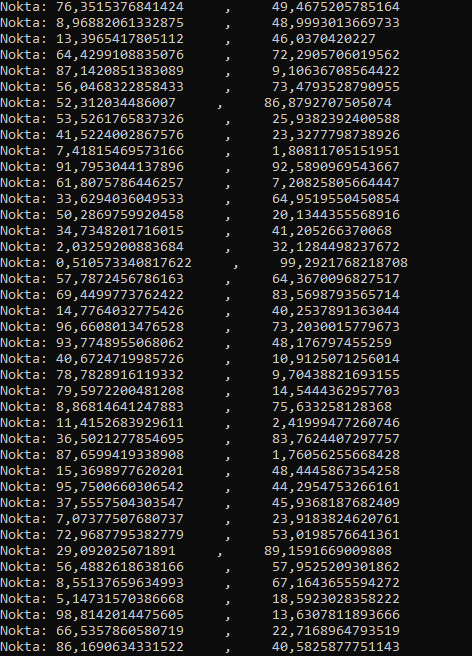
N =10 için olan rastgele nokta üretimi değerleri



N =100 için olan değerler 1. Kısım ( 100 değerde konsola sığmadığı için 2 parçaya böldüm )



N =100 için olan değerler 2. Kısım ( 100 değerde konsola sığmadığı için 2 parçaya böldüm )



### 1.a.3 Açıklama

Öncelikle çift boyutlu bir “noktalar” listesi açıp i değerleri ile noktaları dönüp 0 ve 1 değerleri ile y eksenini(her bir kolonu) döndüm, bu listeye random sayılar oluşturarak her bir değeri ekledim ve print işlemlerini yaptım.

## 1.b Uzaklık Matrisi

### 1.b.1 Kodlar

static double[,] DistanceMatrix(int n,double[,] noktalar) // uzaklık matrisi oluşturulması

{

double[,] uzaklıkMatrisi = new double[n, n];

double uzaklık =0.0;

for(int i=0; i<n; i++)

{

for (int j = 0; j < n; j++)

{ // x'ler ve y'ler farkının kareler toplamının karekökünün alınması

uzaklık = Math.Sqrt(Math.Pow(noktalar[i, 0] - noktalar[j, 0], 2) + Math.Pow(noktalar[i, 1] - noktalar[j, 1], 2));

// bulunan değerlerin çift boyutlu uzaklıkMatrisine atanması

uzaklıkMatrisi[i, j] = uzaklık;

Console.Write(String.Format("{0,10:0.0}", uzaklıkMatrisi[i, j]));

}

Console.WriteLine();

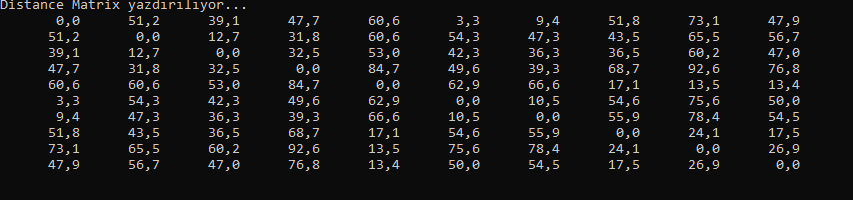
}

return uzaklıkMatrisi ;

Console.WriteLine("Distance Matrix yazdırılıyor...");

double[,] uzaklıklar1 = DistanceMatrix(10,noktalar2);

### 1.b.2 Ekran görüntüleri

N =10 için Distance Matrix değerleri

### 1.b.3 Açıklama

Daha sonrasında bu değerlere “DistanceMatrix” adlı çift boyutlu listede i = x ekseni j = y ekseni olmak üzere her bir nokta arasındaki uzaklık değerlerini listenin her bir elemanı olarak aldım. Daha sonra da main içerisinde print işlemlerini yapıp bitirdim.

2) CLASSIFICATION USING K-NEAREST NEIGHBORS (KNN) ALGORITHM

//The platform, version, and programming language used

## 2.a KNN ile sınıflandırma

### 2.a.1 Algoritma

static double Knn(int k, double[,] veriSeti, double[] kullanıcıDeğerleri)

{

List<double[]> uzaklıklar = new List<double[]>();

for (int i = 0; i < veriSeti.GetLength(0); i++) //uzaklıklar arrayList'ine uzaklıkların eklenmesi

{

double[] dizi = new double[6];

for (int y = 0; y < 5; y++)

{

dizi[y] = veriSeti[i, y];

}

dizi[5] = uzaklıkHesapla(dizi, kullanıcıDeğerleri);

uzaklıklar.Add(dizi);

}

listeSıralama(uzaklıklar);

int birSayısı = 0;

int sıfırSayısı = 0;

for (int i = 0; i < k; i++)

{

if (uzaklıklar[i][4] == 0)

{

sıfırSayısı++;

}

else

{

birSayısı++;

}

Console.Write(String.Format("{0,10:0.000000}", uzaklıklar[i][0]));

Console.Write(String.Format("{0,10:0.000000}", uzaklıklar[i][1]));

Console.Write(String.Format("{0,10:0.000000}", uzaklıklar[i][2]));

Console.Write(String.Format("{0,10:0.000000}", uzaklıklar[i][3]));

Console.Write(String.Format("{0,3:0}", uzaklıklar[i][4]));

Console.Write(String.Format("{0,10:0.000000}", uzaklıklar[i][5]));

Console.WriteLine();

}

double sınıflandırma = 0;

if (birSayısı < sıfırSayısı)

{

sınıflandırma = 0;

}

else if (sıfırSayısı < birSayısı)

{

sınıflandırma = 1;

}

else

{

sınıflandırma = uzaklıklar[0][4];

}

return sınıflandırma;

listeSıralama(uzaklıklar);

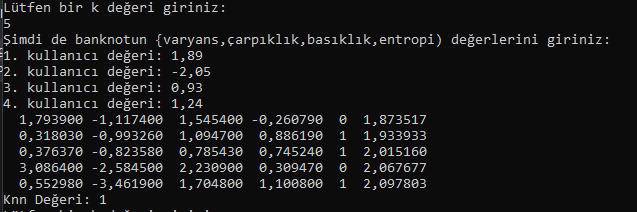
}

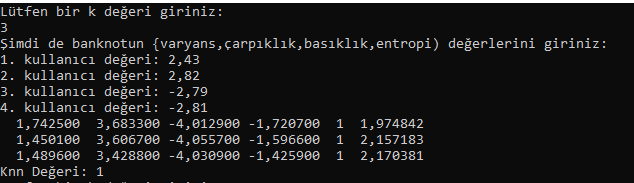
## 2.b Banknot sınıflandırma

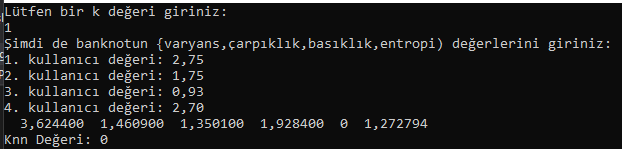
### 2.b.1 Kodlar

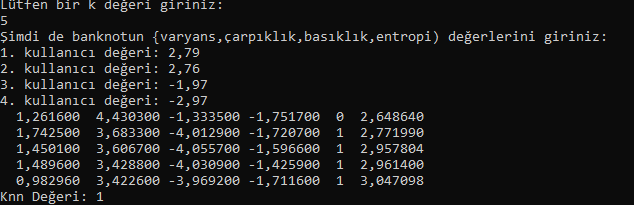
Banknot sınıflandırma kodu da Knn algoritması içerisindedir. (Yukarıda paylaştım.)

### 2.b.2 Ekran görüntüleri









Proje açıklamasının yapılıdığı pdfte Tablo-2de verilen ilk iki değer ve son iki değerin ekran görüntülerini buraya koydum. Dilerseniz programı çalıştırdığınızda diğer verileri de deneyebilirsiniz.

### 2.b.3 Açıklama

Öncelikle jenerik bir double liste oluşturuyorum daha sonra bu jenerik listeye add methoduyla uzaklıkHesapla adlı fonksiyondan aldığım uzaklık değerlerini ekliyorum. Daha sonra knn algoritmasını daha kolay bir şekilde çalıştırmak için listemi sıralıyorum. Listeyi sıralamamın sebebi ise ilk k kadar değeri bulup onlar arasından tür sıralaması yapmamın kolay olması için. Sort işlemini yaptıktan sonra for döngüsü açıp I<k kadar döndürüyorum çünkü sortlu olan listemden gelecek değerler k tane en yakın değerler olmuş olucak 4. kolonundaki değerler ise türü belirticek burada da tür kontrolü yapıp 4. kolondaki değer 0 ise 0sayısını artırıyorum değilse 1 sayısını artırıyorum ve bu şekilde de tür sayısını bulmuş oluyorum ve ekrana banknot değerlerini, k kadar yakın olan banknot türleri ve uzaklıklarını printliyorum.

## 2.c Başarı ölçümü

### 2.c.1 Kodlar

List<double[]> kontrolEdilecek200Data = new List<double[]>();//200 veri bunun double[6]

double[,] kontrolHavuzu = new double[1172,5];//geri kalan 1172 veriyi tutucak double[6]

for (int i = 662; i < 762; i++) // veri setindeki tür değerlerinin 0 değerlerinden 1 değerlerine geçerkenki son 100 değerin alınması

{

double[] dizi = new double[5];

for (int y = 0; y < 5; y++)

{

dizi[y] = veriSeti[i, y];

}

kontrolEdilecek200Data.Add(dizi);

}

for (int i = 1272; i < 1372; i++) //veri setindeki son 100 değerin alınması

{

double[] dizi = new double[5];

for (int y = 0; y < 5; y++)

{

dizi[y] = veriSeti[i, y];

}

kontrolEdilecek200Data.Add(dizi);

}

for (int i = 0; i < 662; i++) // veri setinde alınan 100 değerden sonra kalan değerlerin alınması

{

for (int y = 0; y < 5; y++)

{

kontrolHavuzu[i, y] = veriSeti[i, y];

}

}

for (int i = 662; i+100 < 1272; i++) // veri setinde alınan 100 değerden sonra kalan değerlerin alınması

{

for (int y = 0; y < 5; y++)

{

kontrolHavuzu[i,y] = veriSeti[i+100, y];

}

}

Console.WriteLine("Lütfen bir k değeri giriniz:");

k = Convert.ToInt32(Console.ReadLine());

double kontrolEdilecekDataDöndür;

double başarıSayacı=0;

for(int i=0; i<200; i++)

{

kontrolEdilecekDataDöndür = Knn(k, kontrolHavuzu, kontrolEdilecek200Data[i]);

Console.WriteLine("gerçek: "+kontrolEdilecek200Data[i][4]+"tahmin: "+kontrolEdilecekDataDöndür);

if (kontrolEdilecekDataDöndür == kontrolEdilecek200Data[i][4] )

{

başarıSayacı++;

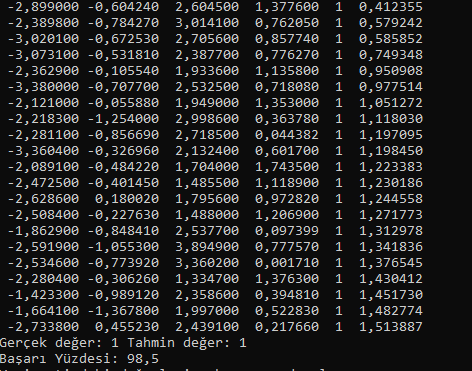
}

}

Console.WriteLine("Başarı Yüzdesi: " + (başarıSayacı / 200) \* 100);

### 2.c.2 Ekran görüntüleri

K=21 değeri için ekran görüntüsünü koyuyorum dilerseniz istediğiniz değeri deneyebilirsiniz. Başarı oranı = 98,5 (Her bir değer için print yapıyorum ama sonucu göstermek için burda sadece en alttaki yerin ekran görüntüsünü attım, videoda detaylı gösteriyorum.



### 2.c.3 Açıklama

Öncelikle kontrol edilecek 200 data için bir liste ve sonra geri kalan değerler için de çift boyutlu bir liste oluşturuyorum. Daha sonrasında tür değerlerinde 0’lardan 1’e dönüşen noktaya kadar son 100 değeri, 1’e dönüştükten sonraki 100 değeri ayırıp bunları da for döngüleriyle açtığım listelere ekliyorum. Daha sonrasında bir başarı sayacı oluşturarak tahmin ettiğim değer ile gerçek değer eşitse artırıyorum ve başarı yüzdesini ekrana printliyorum.

## 2.d Listeleme

### 2.d.1 Kodlar

string[] herBirDeğerSatırı = System.IO.File.ReadAllLines(@"C:\\Users\\Casper\\Desktop\\data\_banknote\_authentication.txt"); // dosya içerisindeki verileri okuma ve herBirDeğer listesinde satırları tutma

double[,] veriSeti = new double[herBirDeğerSatırı.Length, 5];

//Veri setindeki verilerin iki boyutlu veriSeti dizisine değişkenlere göre aktarılması (varyans,basıklık,entropi...)

for (int i = 0; i < herBirDeğerSatırı.Length; i++)

{

int j = 0;

int indexBoyutu = 0; // Substring methodu kullanırken ,'den sonra gelecek değerin o satırdaki değişkenin kaç haneden olduğunun bulunması için kullanılacak değişken

int sayac = 0; // Substring methodu kullanırken ,'e kadar alınacak değerlerin her seferinde hangi index'ten başladığını saymak için kullanılacak sayaç

for (int m = 0; m < herBirDeğerSatırı[i].Length; m++)

{

if (herBirDeğerSatırı[i].Substring(m, 1) == "," || (m + 1) == herBirDeğerSatırı[i].Length) // string manipüle edilmesi, ","'e kadar olan değerlerden parçalama işlemi yapılması

{

indexBoyutu = m - sayac; //ilk , işaretini görüp index boyutu hesaplaması

if ((m + 1) == herBirDeğerSatırı[i].Length)

{

indexBoyutu = 1;

}

veriSeti[i, j] = double.Parse(herBirDeğerSatırı[i].Substring(sayac, indexBoyutu).Replace(".", ",")); // .'lı şekilde olduğunda program hata veriyor ve 3.6661 değerini 36661 olarak alıyor bu yüzden yerine , atadım

sayac = m + 1; // sayaç her , gördüğünde m+1 değerini almalı ki diğer ,'ü gördüğünde kaçıncı indexten başlayacağını ve ne kadar uzunlukta olacağını bulabilelim

j++;

}

}

}

// Veri setindeki değerlerin ekrana yazdırılması

Console.WriteLine("Veri setindeki değerlerin ekrana yazdırılması");

for (int x = 0; x < herBirDeğerSatırı.Length; x++)

{

for (int y = 0; y < 5; y++)

{

Console.Write(veriSeti[x, y] + " ");

if (y == 4)

{

Console.WriteLine("");

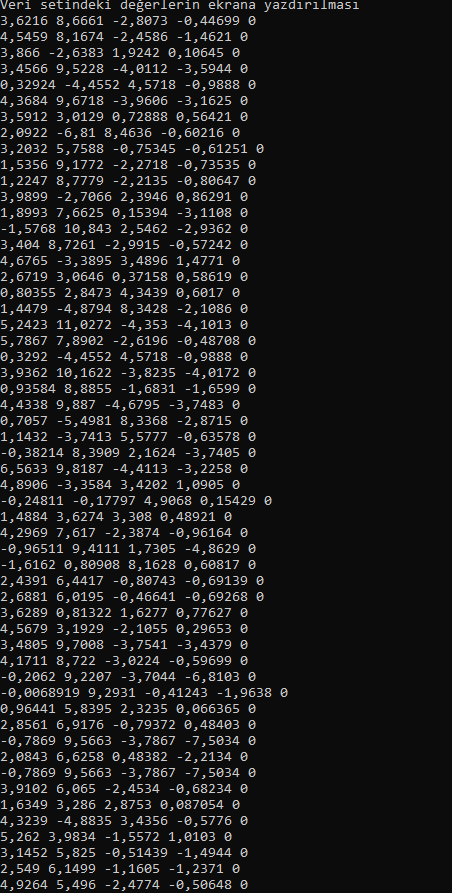
}

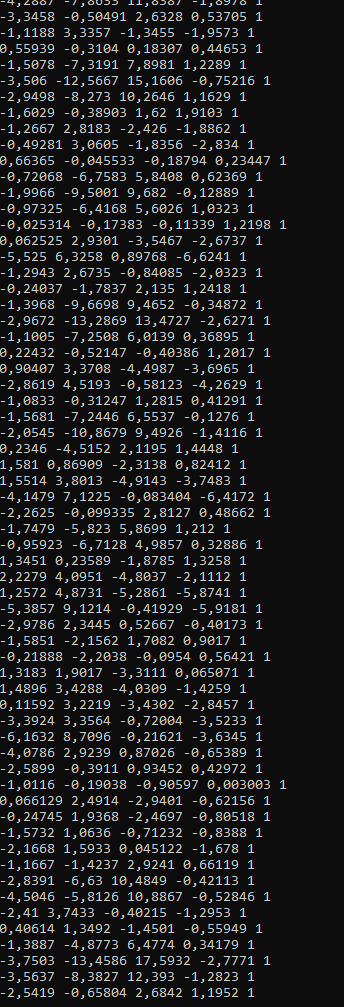
}

}

### 2.d.2 Ekran görüntüleri

Tüm değerlerin ss’lerini alıp buraya atmak uzun süreceği için ilk değerler ve son değerlerin ekran görüntülerini buraya koyuyorum;

 Veri setinin ilk değeri ve onun altındakii değerler

Veri setinin en son değeri ve onun üstündeki değerler

### 2.d.3 Açıklama

Öncelikle veri setini indirip veri seti dosyasının yolunu her bir değerin satırını tutacağım string değişkene atayarak veri setindeki dosyaları okuttum. Daha sonrasında veri seti içerisindeki değerleri tutacağım 2 boyutlu bir liste açarak verileri veriSeti adlı 2 boyutlu listenin her bir satır ve sütununda tutmak için string manipüle işlemini yaptım. String manipülasyonda satırdaki her haneyi “,” ‘e kadar kontrol ederek virgule gördüğü yerde o değeri alıp veri seti içerisindeki her bir satır,sütun değeri için ekleme yapmasını sağladım ve bu döngünün de satır bitene kadar olmasını sağladım index boyutu adlı değişkende kontrol ettiğim her bir ,’e kadar olan sayının uzunluğunu tuttum, sayaç adlı değişkende ise bu eklediğim değerlerden sonra indexin tekrar bir sonraki değerleri tutması için artırma işlemi yaparak kaçıncı indexte olduğumu tuttum ve çift boyutlu listeme ekleme işlemlerini yapttım. Çıktı alma kısmında ise 2 for ile değerlerimi dönerek ekrana bastırdım.