# 

T.C.

CUMHURİYET ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

# arkadaş öneri sistemi

## ERHAN BİLGİN

## 2014123071

### *Özet:*

Projede lojistik regrasyon kullanılarak bir arkadaş öneri sistemi geliştirilecektir. *Bu öneri sistemi kullanıcı tarafından seçilen bir kişi için arkadaş önerisinde bulunacaktır.*

***Anahtar Kelimeler:*** lojistik regrasyon, windows form uygulaması, öneri sistemi

*Abstract :*

The project will develop a friend suggestion system using logistic regression. This suggestion system will propose a friend for a user selected by the user.

***Keywords:*** logistic regression, windows form application, suggestion system

## Giriş

Projede kullanılan teknolojiler şunlardır ;

1-) Visual Stuido 2017

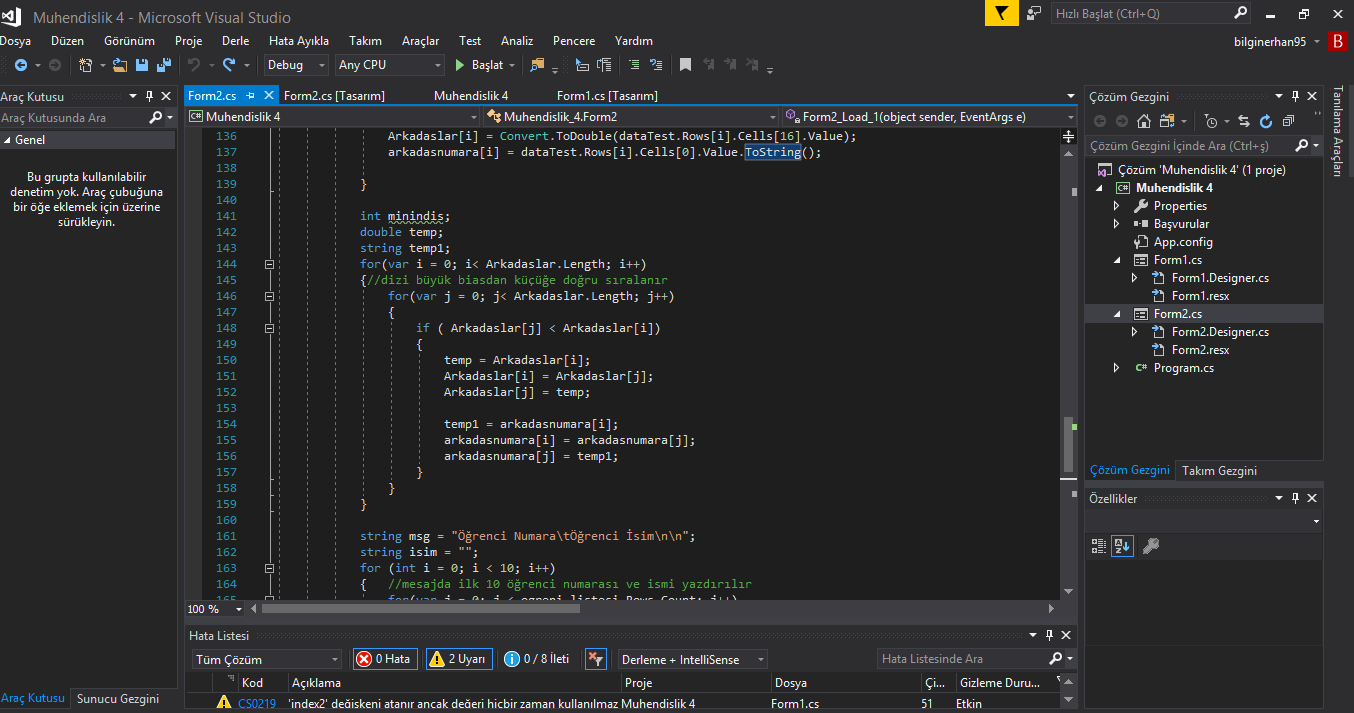
Visual Studio 2017 idesi kullanılarak bir Windows form uygulaması gerçekleştirilmiştir.Veriler csv dosyalarından okunarak hesaplanmıştır

## UYGULAMA

Uygulamayı için kullanılan ide ve diller

* Visual Studio 2017 (Şekil 1.1)

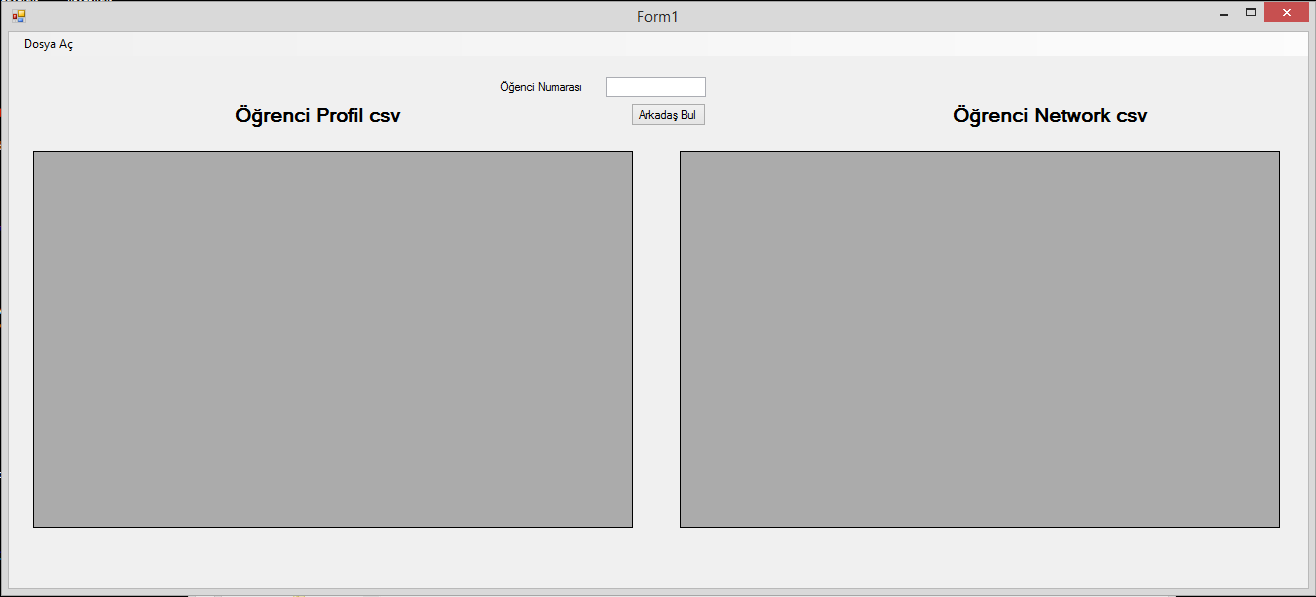
Projenin Geliştirildiği ortam



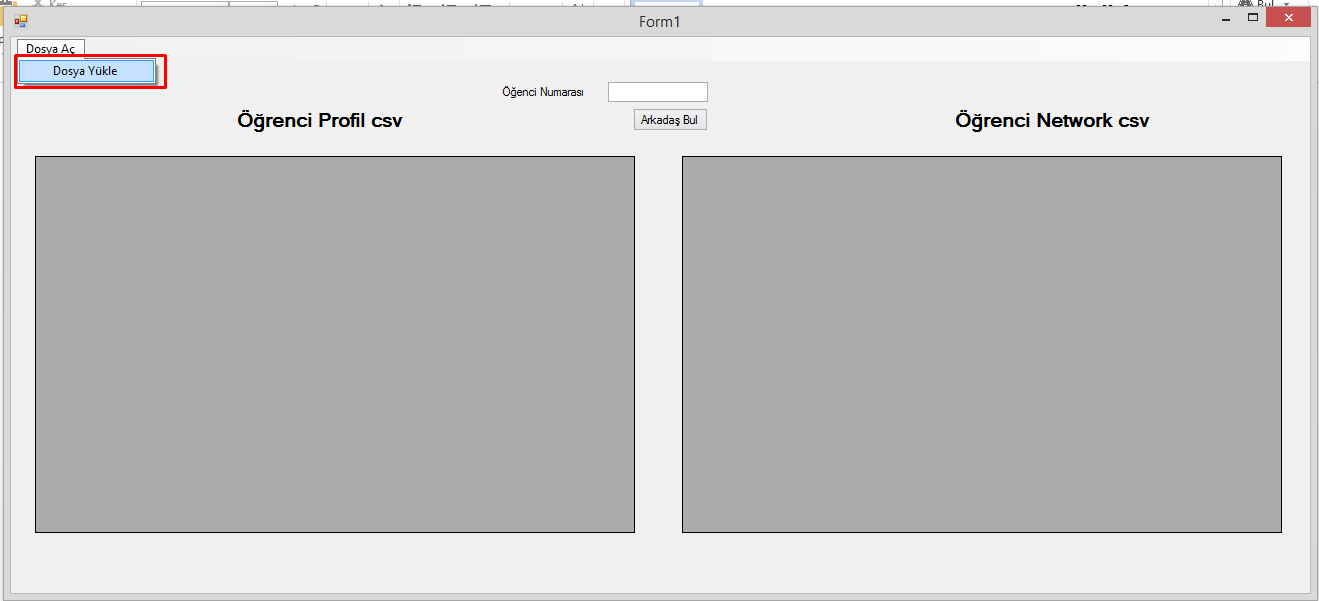
**Şekil 1.1 :** Visual Studio 2017 geliştirme ortamı

## SONUÇLAR

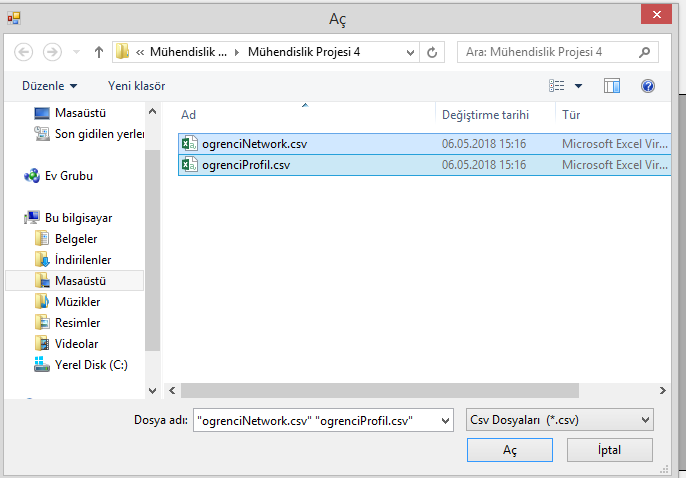
Proje bir visual studio projesi olduğundan size verilen rar dosyası içerisindeki .exe uzantılı dosyayı çalıştırarak uygulamaya erişebilrisiniz.Şekil 1.2 de ki bir ekran sizi karşılayacaktır



**Şekil 1.2 :** Uygulama Ekranı

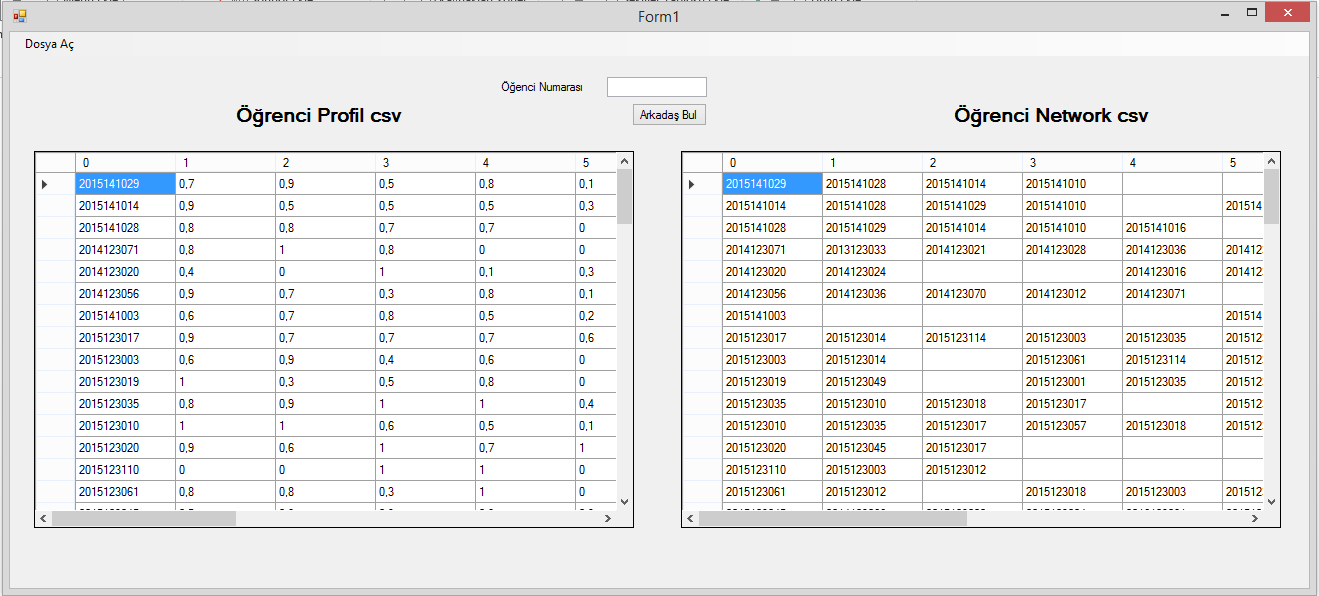
Sol üst tarafta bulunan dosya aç sekmesinde bulunan dosya yükleye basarak .csv uzantılı dosyaları yükleyebilirsiniz. Şekil 1.3 ve şekil 1.4 de gösterilmiştir.

**Şekil 1.3 :** Uygulama giriş ekranı



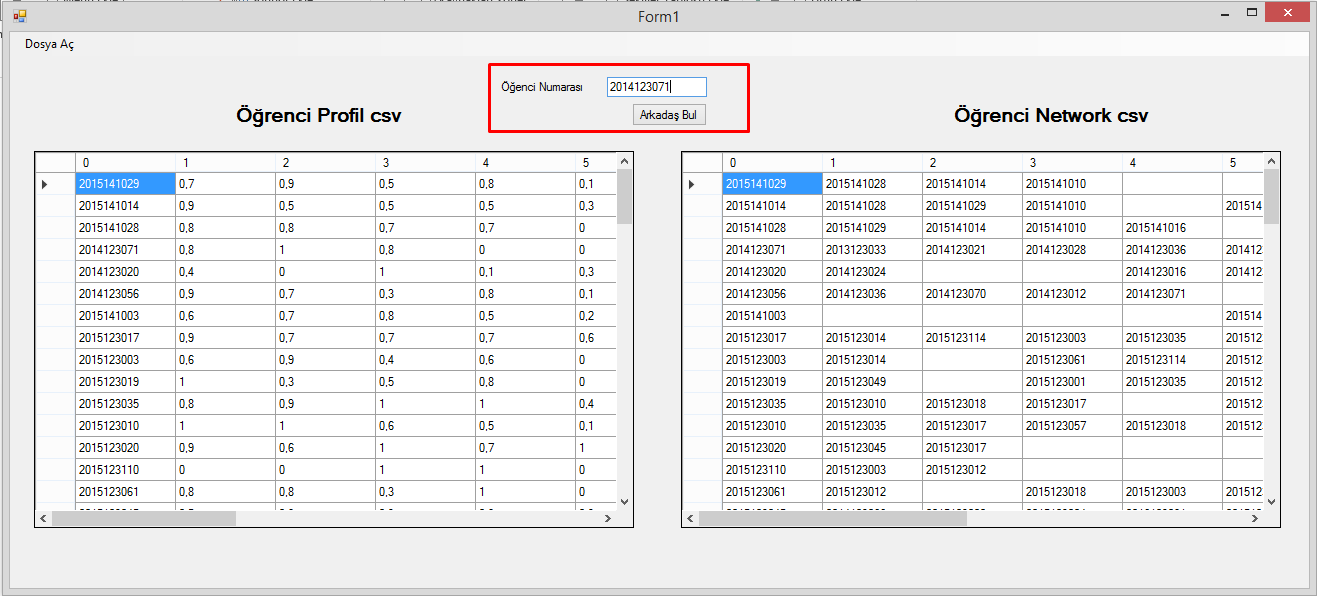
**Şekil 1.4 :** Dosya yükleme ekranı

Seçilen dosyaları Aç butonuna bastıktan sonra uygulama üzerinde gridlere .csv uzantılı dosyalar içindeki dataları aktarmaktadır. Şekil 1.5 de gösterilmiştir.



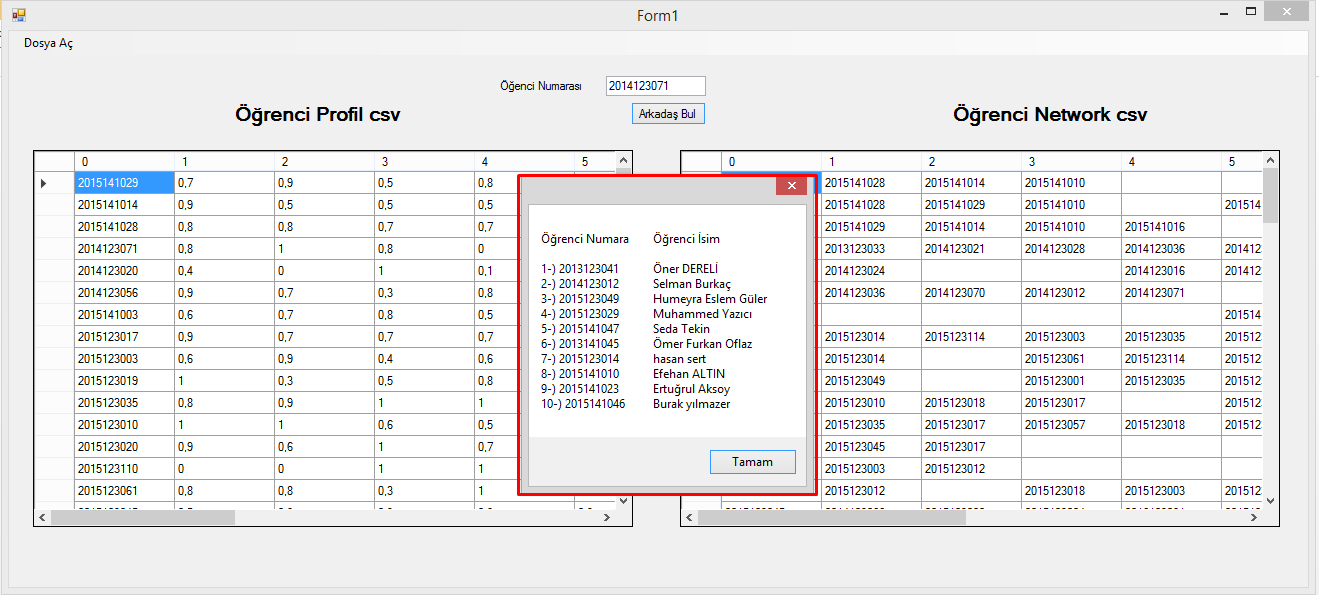
**Şekil 1.5 :** Csv datalarının gridde gösterilmesi

Daha sonra yüklenen veriler üzerinde hesaplama yapmak için Öğrenci numarası bölümüne listeden bir öğrenci seçerek Arkadaş Bul butonuna basın. Şekil 1.6 da gösterilmiştir.



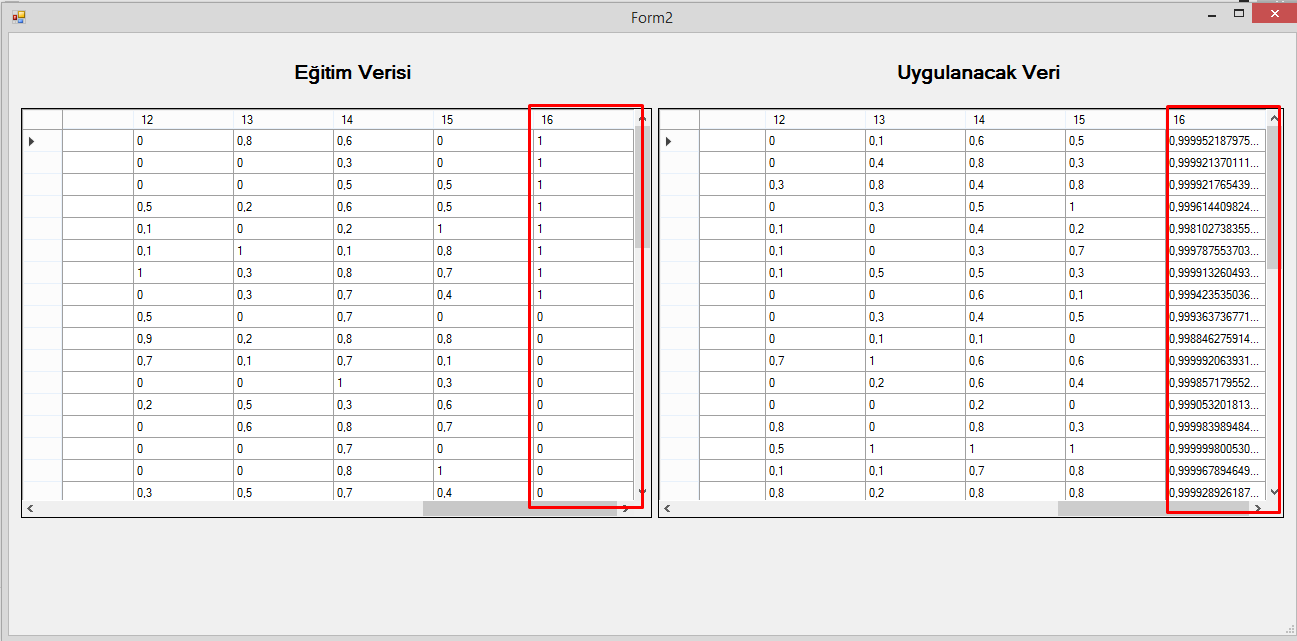
**Şekil 1.6 :** Arkadaş Arama

Arkadaş Bul butonuna bastıktan sonra program lojistik regrasyon yaparak uygulama ekrana bir mesaj penceresi gösterir bu pencerede yazılan kişiye 10 adet arkadaş önerisinde bulunur. Şekil 1.7 de gösterilmiştir.



**Şekil 1.7 :** Arkadaş Önerileri

Ekrandaki Tamam butonuna bastıktan sonra Uygulama sonuçları başka bir form da gösterilmektedir Resimdeki sol taraftaki kırmızı alan öğrenci networkden alınan kişinin arkadaş olanlar 1 diğerleri 0 olarak atanmıştır bu veriler kullanılarak biaslar hesaplanmıştır ve bu biaslar sağ taraftaki kısma uygulandıktan sonra kişilerin arkadaş olma oranları grid üzerinde gösterilmektedir..Şekil 1.8 de gösterilmiştir.



**Şekil 1.8 :** Simulate Butonuna basmadan önce

## KAYNAK KODLAR

Kaynak kodların github linki : <https://github.com/BilginErhan/MuhProje4>

Form1.cs

\*------------------------------------------------------------------\*

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Muhendislik\_4

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void dosyaYükleToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog op = new OpenFileDialog();//dosya almak için filedialog nesnesi

op.Filter = "Csv Dosyaları |\*.csv"; //sadece csv dosyalarını seçmesi için filtreleme işlemi

op.Multiselect = true; //çoklu seçim özelliği

if (op.ShowDialog() == DialogResult.OK)//openfilediaglog penceresni açar ve aç diyince

{

string[] dosyaYollari = op.FileNames;//dosya yolları ve

string[] dosyaAdlari = op.SafeFileNames; //isimlerini alır

for(var i = 0; i < dosyaYollari.Length; i++)//dosya yolları üzerinde dönerek verileri okur

{

string dosyaYolu = dosyaYollari[i];//dosya yolunu alır

DataTable dt = new DataTable();//data table nesnesi oluşturulur

string[] lines = System.IO.File.ReadAllLines(dosyaYolu);//tüm satırla io nesnesi ile okunur

if (lines.Length > 0)//eğer boş değilse

{

string firsLine = lines[0];//ilk satır alınır

string[] headerLabels = firsLine.Split(',');//, ile split edilir

int j = 0;//Header Oluştur

foreach(string headerWord in headerLabels)

{//header nesnesi oluşturulur

dt.Columns.Add(new DataColumn(j.ToString()));

j++;

}

for (int r = 0; r < lines.Length; r++)

{//data içerisinde gez datayı oluştur

string[] dataWords = lines[r].Split(',');

DataRow dr = dt.NewRow();

int index = 0, index1=0,index2=0;

foreach (string headerWord in headerLabels)

{

if (dosyaAdlari[i].Equals("ogrenciProfil.csv") && index != 0)

{//, ile ayrılan datalar ogrenciprofil ise puanları 10 da 1 oranında düşür

//bias hesaplarken double değişkeninin boyutu yetmemektedir o yüzden bu yola başvuruldu

double d1 = Convert.ToDouble(dataWords[index]) \* 0.1;

//double d1 = Convert.ToDouble(dataWords[index]);

dr[index++] = d1.ToString();

index1 = index;

}

else

{

dr[index++] = dataWords[index1++];

}

}

//okunan satırlar datatable a atılır

dt.Rows.Add(dr);

}

}

if (dt.Rows.Count >0 && dosyaAdlari[i].Equals("ogrenciProfil.csv"))

{//eğer okunan dosya ne ise ilgili datagridview atılır

DataProfilcsv.DataSource = dt;

}

if (dt.Rows.Count > 0 && dosyaAdlari[i].Equals("ogrenciNetwork.csv"))

{

DataNetworkcsv.DataSource = dt;

}

}

}

}

private void button1\_Click(object sender, EventArgs e)

{//arkadaş bul butonuna basıldığında

int sonuc = 0;

for(int k = 0; k < DataNetworkcsv.Rows.Count-1; k++)

{//datagrid view üzerinde dönerek textbox içerisine yazılan numarayı arar

//buluduğunda yeniArkadasMatrisi fonksiyonuna gider

if (DataNetworkcsv.Rows[k].Cells[0].Value.ToString().Equals(Arama.Text))

{

yeniArkdasMatrisi(k);

sonuc = 1;

break;

}

}

if (sonuc ==0)

{//bulamaz ise hata mesajı döndürür

MessageBox.Show(Arama.Text + " Nolu Öğrenci Bulunamadı!!");

}

}

public void yeniArkdasMatrisi(int rowIndex)

{

ArrayList arkadasRow = new ArrayList();

for( var i = 0; i< DataNetworkcsv.Rows[rowIndex].Cells.Count; i++)

{//data netword üzerinde dönülerek

string str = DataNetworkcsv.Rows[rowIndex].Cells[i].Value.ToString();

for(int k = 0; k< DataProfilcsv.Rows.Count - 1; k++)

{//data profil üzerinde dönülür

string str1 = DataProfilcsv.Rows[k].Cells[0].Value.ToString();

if (str.Equals(str1) && !str.Equals(Arama.Text))

{//network üzerinde eşlesen kayıt olursa kendisi hariç aradasrow arraylistine atılır

arkadasRow.Add(k);

}

}

}

DataTable yeniData = new DataTable();

for(int k = 0; k < 17; k++)

{//yeni data oluşturulur

yeniData.Columns.Add(new DataColumn(k.ToString()));

}

int deneme;

foreach (int index in arkadasRow)

{//ilk satırlara kişinin arkadaşları gelmektedir

DataRow row = yeniData.NewRow();

for(var i = 0; i< DataProfilcsv.Rows[index].Cells.Count; i++)

{

row[i] = DataProfilcsv.Rows[index].Cells[i].Value.ToString();

}

row[16] = "1";//en son column 1 olarak atanır

yeniData.Rows.Add(row);//data table ekleme yapılır

}

for(var i = 0; i< DataProfilcsv.Rows.Count-1; i++)

{//data profil üzerinde gezilerek geri kalan kişileri datatable a ekleme yapılır

int index = arkadasRow.IndexOf(i);

if (index == -1)

{

DataRow row = yeniData.NewRow();

if (!DataProfilcsv.Rows[i].Cells[0].Value.ToString().Equals(Arama.Text))

{

for (int j = 0; j < DataProfilcsv.Rows[i].Cells.Count; j++)

{

row[j] = DataProfilcsv.Rows[i].Cells[j].Value.ToString();

}

row[16] = "0";//arkadaş olmadıkları için 0 atanır column

yeniData.Rows.Add(row);//satırlar datatable a eklenir

deneme = yeniData.Rows.Count;//debug için

}

}

}

//kenidis hariç olduğu için 90 data değil 89 adet kayıt mevcuttur.

//DataProfilcsv.DataSource = yeniData;

Form2 frm = new Form2();

frm.veri = yeniData;//yeni data form2 ye aktarılır

frm.arkadasSayisi = arkadasRow.Count;//arkadaş sayısı diğer forma aktarılır

frm.ShowDialog();

}

}

}

\*------------------------------------------------------------------\*

Form2.cs

\*------------------------------------------------------------------\*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Muhendislik\_4

{

public partial class Form2 : Form

{

public DataTable veri { get; set; }//form1 den gelen verileri karşılar

public int arkadasSayisi { get; set; }

public double[] ebias = new double[16];//bias dizileri

public double[] ybias = new double[16] { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,1 };

DataTable egitim;//iki adet datasetimiz

DataTable test;

public Form2()

{

InitializeComponent();

}

private void Form2\_Load\_1(object sender, EventArgs e)

{

//Oledb bağlantımız excell den öğrenci numara ve isim çekmek için

OleDbConnection baglanti = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source =C:\\Users\\P1274\\Desktop\\University\\Mühendislik Projesi\\Mühendislik Projesi 4\\ogrencilistesi.xlsx; Extended Properties = Excel 12.0");

baglanti.Open();

string sorgu = "select \* from [Sheet1$] ";

OleDbDataAdapter data\_adaptor = new OleDbDataAdapter(sorgu, baglanti);

baglanti.Close();

DataTable ogreni\_listesi = new DataTable();

data\_adaptor.Fill(ogreni\_listesi);

//excell den çekilen data ogrenci\_listesi data table a atılır

//form1 den gelen toplam veri datalojistik gridine atılır

DataLojistik.DataSource = veri;

int sayi = arkadasSayisi;//arkadaş sayısı alnır

int sayi1 = DataLojistik.Rows.Count;

egitim = new DataTable();//data table ların nesneleri oluşturulur

test = new DataTable();

for (int k = 0; k < 17; k++)

{//2 adet datatable ımızın headerları oluşturulur

egitim.Columns.Add(new DataColumn(k.ToString()));

test.Columns.Add(new DataColumn(k.ToString()));

}

//ilk datagridimizde toplam kaç kişi olacağı belirlenir

int ilk\_dongu = (DataLojistik.Rows.Count - 1 - arkadasSayisi) / 2;

//for döngüsünde bu belirlenir

for (var i = 0; i< ilk\_dongu + arkadasSayisi; i++)

{

//arkadaş olanlar ve kalan verinin yarısı kadar döner

//bunları egitim data table a atılır

DataRow row = egitim.NewRow();

for (var j = 0; j < DataLojistik.Rows[i].Cells.Count;j++)

{

row[j] = DataLojistik.Rows[i].Cells[j].Value;

}

egitim.Rows.Add(row);

}

for (var i = ilk\_dongu + arkadasSayisi; i < DataLojistik.Rows.Count-1; i++)

{//kalan datayı test data table a atılır

DataRow row1 = test.NewRow();

for (var j = 0; j < DataLojistik.Rows[i].Cells.Count; j++)

{

row1[j] = DataLojistik.Rows[i].Cells[j].Value;

}

test.Rows.Add(row1);

}

//datalojistik gridi eğitim verisi

DataLojistik.DataSource = egitim;

//datatest gridi uygulanacak test verisi

dataTest.DataSource = test;

//bias hesaplama//

double reg = 0;

for(var i = 0; i < 100; i++)

{//100 iterasyon sayısı

reg = 0;

for(var j = 0; j < DataLojistik.Rows.Count-1; j++)

{//eğitim verisi üzerinde dön

//bu kısımda bias[0] hesaplanır

reg = reg + bias(DataLojistik.Rows[j]) - Convert.ToDouble(DataLojistik.Rows[j].Cells[16].Value);

}

ebias[0] = ybias[0] - (0.0001\* (1 / ((double)DataLojistik.Rows.Count-1)) \* reg);

reg = 0;

for (var q = 1; q <= 15; q++)

{//bu kısımda diğer bias değerleri hesaplanır

for (var j = 0; j < DataLojistik.Rows.Count-1; j++)

{

reg = reg + ( bias(DataLojistik.Rows[j]) - Convert.ToDouble(DataLojistik.Rows[j].Cells[16].Value) ) \* Convert.ToDouble(DataLojistik.Rows[j].Cells[q].Value);

//reg = reg \* Convert.ToDouble(DataLojistik.Rows[j].Cells[q].Value);

}

ebias[q] = ybias[q] - (0.0001 \* (1 / ((double)DataLojistik.Rows.Count-1)) \* reg);

reg = 0;

}

for(var z = 0; z<16; z++)

{//son olarak güncel bias değerleri ybias a atılır

ybias[z] = ebias[z];

}

}

for (var j = 0; j < dataTest.Rows.Count-1; j++)

{//test verisi üzerinde bias değerleri kullanılır

dataTest.Rows[j].Cells[16].Value = bias(dataTest.Rows[j]).ToString();

}

double[] Arkadaslar = new double[dataTest.Rows.Count-1];//En büyük değer çıkan 10 arkadaş

string[] arkadasnumara = new string[dataTest.Rows.Count-1];//hesaplama

for (var i = 0;i< dataTest.Rows.Count-1;i++)

{//for içiersinde dönerek tüm datayı bu dizilere atılır

Arkadaslar[i] = Convert.ToDouble(dataTest.Rows[i].Cells[16].Value);

arkadasnumara[i] = dataTest.Rows[i].Cells[0].Value.ToString();

}

int minindis;

double temp;

string temp1;

for(var i = 0; i< Arkadaslar.Length; i++)

{//dizi büyük biasdan küçüğe doğru sıralanır

for(var j = 0; j< Arkadaslar.Length; j++)

{

if ( Arkadaslar[j] < Arkadaslar[i])

{

temp = Arkadaslar[i];

Arkadaslar[i] = Arkadaslar[j];

Arkadaslar[j] = temp;

temp1 = arkadasnumara[i];

arkadasnumara[i] = arkadasnumara[j];

arkadasnumara[j] = temp1;

}

}

}

string msg = "Öğrenci Numara\tÖğrenci İsim\n\n";

string isim = "";

for (int i = 0; i < 10; i++)

{ //mesajda ilk 10 öğrenci numarası ve ismi yazdırılır

for(var j = 0; j < ogreni\_listesi.Rows.Count; j++)

{

if (ogreni\_listesi.Rows[j][0].ToString().Equals(arkadasnumara[i].ToString()))

{

isim = ogreni\_listesi.Rows[j][1].ToString();

}

}

//Öğrenci isim ve numaraları bir düzene göre sıralanır

msg += (i+1)+"-) "+arkadasnumara[i].ToString()+"\t"+isim+"\n";

}

MessageBox.Show(msg);

}

public double bias(DataGridViewRow g1)

{//bias fonksiyonu hesaplaması

double sonuc;

double e = ybias[0];

for(var i = 1; i < 16; i++)

{

e = e + ( Convert.ToDouble(g1.Cells[i].Value) \* ybias[i] );

}

e = -1 \* e;

double b1 = Math.Exp(e);

b1++;

sonuc = 1 / b1;

return sonuc;

}

}

}

\*------------------------------------------------------------------\*