T.C.
YOZGAT BOZOK ÜNİVERSİTESİ
MÜHENDİSLİK MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

Bulanık Mantık Ve Yapay Sinir Ağları

Konu: Logistic Regression İle Diyabet Hastalığı Tahmini.

Hazırlayan

Adı Soyadı: Esra Yüce

Öğrenci No: 16008117051

Veri Seti

Veri seti, Diyabet hastalığının tahmini için belirli özelliklerin kullanıldığı 9 sütun (ilk 8 sütun giriş değerleri, son sütun çıkış değeri) ve 768 satırdan oluşmaktadır.

Sütun başlıkları aşağıda sırası ile verilmiştir.

Pregnancies: Hamile kalma sayısı.

Glucose: Oral glikoz tolerans testinde 2 saatlik plazma glikoz konsantrasyonu.

BloodPressure: Diyastolik kan basıncı (mm Hg)

SkinThickness: Triceps cilt kıvrımı kalınlığı (mm).

Insulin: 2 saatlik serum insülini (mu U / ml).

BMI: Vücut kitle indeksi (kg cinsinden ağırlık / (m cinsinden boy)% 2).

DiabetesPedigreeFunction: Diyabet soyağacı fonksiyonu.

Age: Yaş.

Outcome: Hasta olup(1) olmama(0) durumu.

Veri seti adresi: https://www.kaggle.com/sefakocakalay/diabets

Problem Tanıtımı

Logistic regression ile model oluşturulup, veri setindeki özellikleri kullanarak diyabet hastalığı tahmininde bulunuldu.

Uygulama Adımları

1. theta sıfır olarak başlatıldı.

2. Eğitim verileri dosyadan X matrisine (giriş) ve vektör y'ye (çıktı) alındı.

3. Tüm parametreler kullanılarak veriler çizdirildi.

4. Maliyet işlevinin en aza indirilmesi için yerleşik fminunc işlevini kullanarak theta optimize edildi.

5. Optimize edilmiş theta kullanarak maliyet ve gradyan bulundu.

6. Test verisinin olasılığını kontrol etmek için veri vektörün theta ile çarpıldı.

7. Olasılık veren çarpımın sigmoidi bulundu.

8. Son olarak, doğruluğu bulmak için tüm verilerin çıktılarını tahmin etmek için tahmin işlevi(predict) kullanıldı.

costFunction

```
% lojistik regresyon ve maliyet gradyanı için parametre olarak theta kullanmanın maliyetini hesaplar.
function [J, grad] = costFunction(theta, X, y)
m = length(y);
J = 0;
grad = zeros(size(theta));
h = sigmoid(X*theta);
J=-( y' * log(h) + (1-y') * log(1-h ))/m;
grad = (X')*(h- y);
grad= grad/m;
end
```

plotData

```
% PLOTDATA X ve y veri noktalarını yeni bir şekle dönüştürür.
% PLOTDATA (x, y), pozitif örnekler için veri noktalarını + ile, negatif noktaları ise o ile çizer.
function plotData(X, y)

figure; hold on;

pos = find(y==1);
neg = find(y==0);

plot(X(pos, 1), X(pos, 2), 'k+', 'LineWidth', 2, 'MarkerSize', 7);
plot(X(neg, 1), X(neg, 2), 'ko', 'MarkerFaceColor', 'y', 'MarkerSize', 7);

hold off;
end
```

sigmoid

```
function g = sigmoid(z)  % z'nin her bir değerinin sigmoidi hesaplandı.
g=1./(1+ exp(-z));
end
```

predict

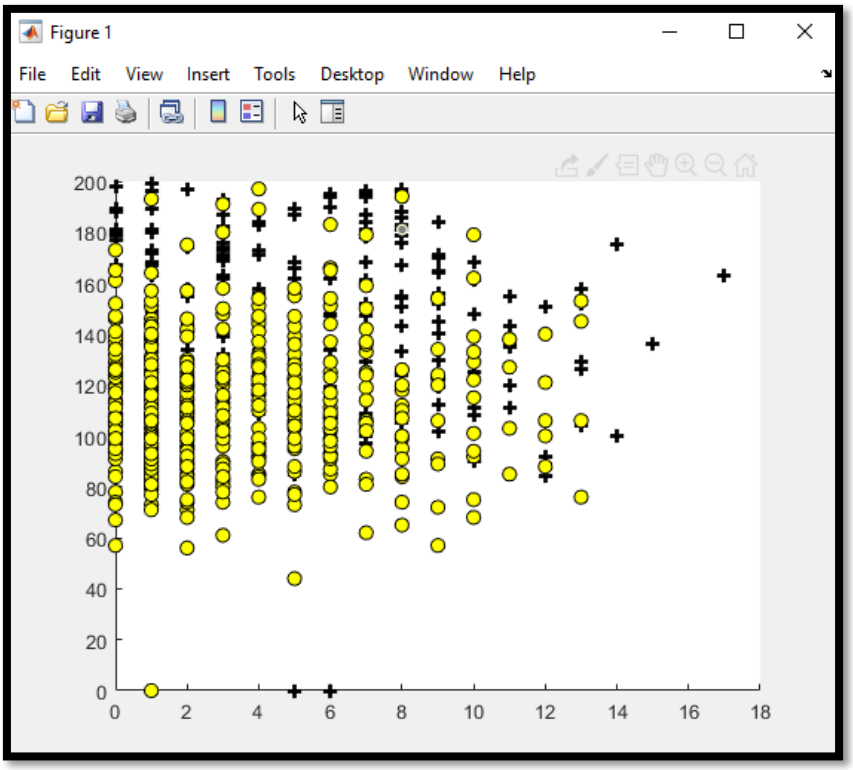
```
% Doğruluk değerini hesaplamak için X te bulunan özelliklerden tahmin yapıldı
% learned logistic kullanarak etiketin 0 mı yoksa 1 mi olduğunu tahmin edildi
function p = predict(theta, X)
m = size(X, 1);
p = zeros(m, 1);
o= X*theta;
o=sigmoid(o);
for i=1 : m
    if o(i) >=0.5
        p(i)=1;
    end
end
end
```

logisticReg

Ek-1:

```
clear ; close all; clc
%%
data = load('diabetes.txt'); % load komutu ile veri seti alındı.
X = data(:, 1:8);  % veri setinin 1'den 8'e kadarki sütunlar ve tüm satırları X'atandı.
y = data(:, 9);    % veri setinin 9. sütunu ve tüm satırları y'ye atandı.
plotData(X, y);    % Grafik için plotData fonksiyonuna X ve y girdileri verildi.
```

Ek-1 Çıktı:



Ek-2:

```
%%
[m, n] = size(X);  % X'in boyutu alındı.
X = [ones(m, 1) X]; % X'in önüne birlerden oluşan m satırlık 1 sütun eklendi.
fprintf('Theta sıfır olarak başlatılır. \n'); % fprintf komutu ile mesaj ekrana yazıldı.
init_theta = zeros(n + 1, 1); % init_theta değişkenine n+1 satır ve 1 sütunlu sıfırlardan oluşan matris atandı.
[cost, grad] = costFunction(init_theta, X, y);
% costFunction'a giriş değeri olarak init_theta, X ve y değerleri verilip çıkış olarak cost ve grad değerleri alındı.
fprintf('İlk theta^daki maliyet değeri: %f\n', cost); % cost yani maliyet değeri ekrana yazıldı.
fprintf('İlk theta^daki gradient değerleri: \n');
fprintf(' %f \n', grad); % gradient değeri ekrana yazıldı.
```

Ek-2 Çıktı:

```
Theta sıfır olarak başlatılır.
İlk theta^daki maliyet değeri: 0.693147
İlk theta^daki gradient değerleri:
 0.151042
 0.224609
 11.154297
 9.837891
 2.533854
 4.886719
 3.733008
 0.043837
 3.685547
```

Ek-3:

```
%% Test
fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n');
test_theta = [-10; 0.1; 0.05;-0.01;0.01;0.001;0.1;1;0.05];
% modelin testi için test_theta değişkenine yeni veriler eklendi.
[cost, grad] = costFunction(test_theta, X, y);
fprintf('\n Test için veriler girildi');
fprintf('\n Test için maliyet değeri: %f\n', cost);
fprintf(' Test için gradient değerleri: \n');
fprintf(' %f \n', grad);
```

Ek-3 Çıktı:

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 Test için veriler girildi
 Test için maliyet değeri: 0.887073
 Test için gradient değerleri:
 0.319286
 1.223843
 38.940643
 22.708466
 6.977634
 29.319233
 10.387760
 0.147424
 11.013632
```

Ek-4:

```
26      %%
27 -    fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n');
28 -    options = optimset('GradObj', 'on', 'MaxIter', 400);
29      % optimisazyon için seçenekler elirlendi ve options değişkenine atandı.
30 -    [theta, cost] = ...
31          fminunc(@(t)(costFunction(t, X, y)), init_theta, options);
32      % problemin minimum çözümü için optimisazyon işlemi uygulandı.
33 -    fprintf('\n Optimizasyondan sonra theta değerileri: \n');
34 -    fprintf(' %f \n', theta);
35 -    fprintf('Optimizasyondan sonra maliyet değeri: %f\n', cost);
36 -    prob = sigmoid([1 10 150 50 25 100 25 0.5 25] * theta);
```

Ek-4 Çıktı:

```
Command Window
  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

  Local minimum found.

  Optimization completed because the size of the gradient is less than
  the value of the optimality tolerance.

  <stopping criteria details>

   Optimizasyondan sonra theta değerileri:
   -8.404692
   0.123182
   0.035164
   -0.013296
   0.000619
   -0.001192
   0.089701
   0.945180
   0.014869
  Optimizasyondan sonra maliyet değeri: 0.470993
```

Ek-5:

```
37      %%
38 -    fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ \n');
39 -    fprintf('Girilen yeni değerler için diyabet olasılığı:');
40 -    fprintf(' %f \n', prob);
41 -    if prob>0.5 % prop yani olasılık değeri 0.5 ten büyük ise kişi diyabet hastasıdır.
42 -    fprintf('Diyabet hastası. \n');
43 -    else % prop yani olasılık değeri 0.5 ten küçük ise kişi diyabet hastası değildir.
44 -    fprintf('Diyabet hastası değil. \n');
45 -    end
46 -    p = predict(theta, X);
47 -    fprintf('Doğruluk: %f\n', mean(double(p == y)) * 100);
48      % predict te üretilen tahmin değeri ile veri setindeki çıkış değeri karşılaştırıldı.
49      % Ortalama değer alınıp modelin doğruluk değeri hesaplandı.
```

Ek-5 Çıktı:

```
Command Window
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Girilen yeni değerler için diyabet olasılığı: 0.603495
Diyabet hastası.
Doğruluk: 78.255208
```

İlk 10 veri sütununa baktığımızda, Preg ve age özellikleri tam sayılardır. Nüfus genellikle genç, 50 yaşın altında. Sıfır değerinin bulunduğu bazı öznitelikler verilerdeki hatalar gibi görünmektedir (örn. Plas, pres, skin, insu ve kütle) Bu veriler ya kaldırılmalı ya da dikkatlice kullanılmalıdır.

Veri kümesindeki tüm özniteliklerin dağılım grafiklerini incelersek:

Yaş ile diyabet başlangıcı arasında belirgin bir ilişki yoktur. Pedi işlevi ile diyabetin başlangıcı arasında açık bir ilişki yoktur. Bu, diyabetin kalıtsal olmadığını veya Diyabet Pedigree İşlevinin çalışılması gerektiğini gösterebilir. age, pedi, mass, insu, skin, pres ve preg için daha büyük değerlerle birleştirilmiş daha büyük plas değerleri, diyabet için pozitif test etme olasılığını daha fazla gösterme eğilimindedir.