به نام خدا

مینی پروژه سوم

مبانی سیستم های هوشمند

دکتر علیاری

شیرین مهدی حاتم

۹۹۳۰۸۹۳

۱۳۰۷
دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق

پاییز ۱۴۰۲

سوال اول:

میخواهیم یک سیستم استنتاج فازی برای یک سیستم با دو ورودی و یک خروجی را پیاده‌سازی کنیم.

برای کران مرتبه اول داریم:

```
clc; clear; close all;
alfa = -1;
beta = 1;
x1 = alfa:0.01:beta;
x2 = alfa:0.01:beta;
N = 51;
h = 0.04;
g_bar = zeros(N + N, 1);
e_il = zeros(N, 1);
e_i2 = zeros(N, 1);
[x1, x2] = meshgrid(x1, x2);
```

این بخش متغیرها را مقداردهی اولیه می‌کند، شامل محدوده مقادیر ورودی (`x1`) و (`x2`)، تعداد مجموعه‌های فازی (`N`) و مقادیر گریب برای `x1` و `x2` با استفاده از `meshgrid`.

۱۳۰۷
دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق

```
for i1=1:N
    for i2=1:N
    e_i1(i1,1) = -1 + h*(i1-1);
    e_i2(i2,1) = -1 + h*(i2-1);
        if i1==1
            mu_A_x1 = trimf(x1,  [-1,-1,-1+h]);
        elseif i1==N
            mu_A_x1 = trimf(x1,[1-h, 1, 1]);
        else
            mu_A_x1 = trimf(x1,[-1+h*(i1-2), -1+h*(i1-1), -1+h*(i1)]);
        end

        if i2==1
            mu_A_x2 = trimf(x2,  [-1,-1,-1+h]);
        elseif i2==N
            mu_A_x2 = trimf(x2,[1-h, 1, 1]);
        else
            mu_A_x2 = trimf(x2,[-1+h*(i2-2), -1+h*(i2-1), -1+h*(i2)]);
        end


       g_bar(k+1,1) = 1./(3+e_i1(i1,1)+e_i2(i2,1));
       num = num + g_bar(k+1,1).*mu_A_x1.*mu_A_x2;
       den=den+mu_A_x1.*mu_A_x2;
       k=k+1;
    end
end
```

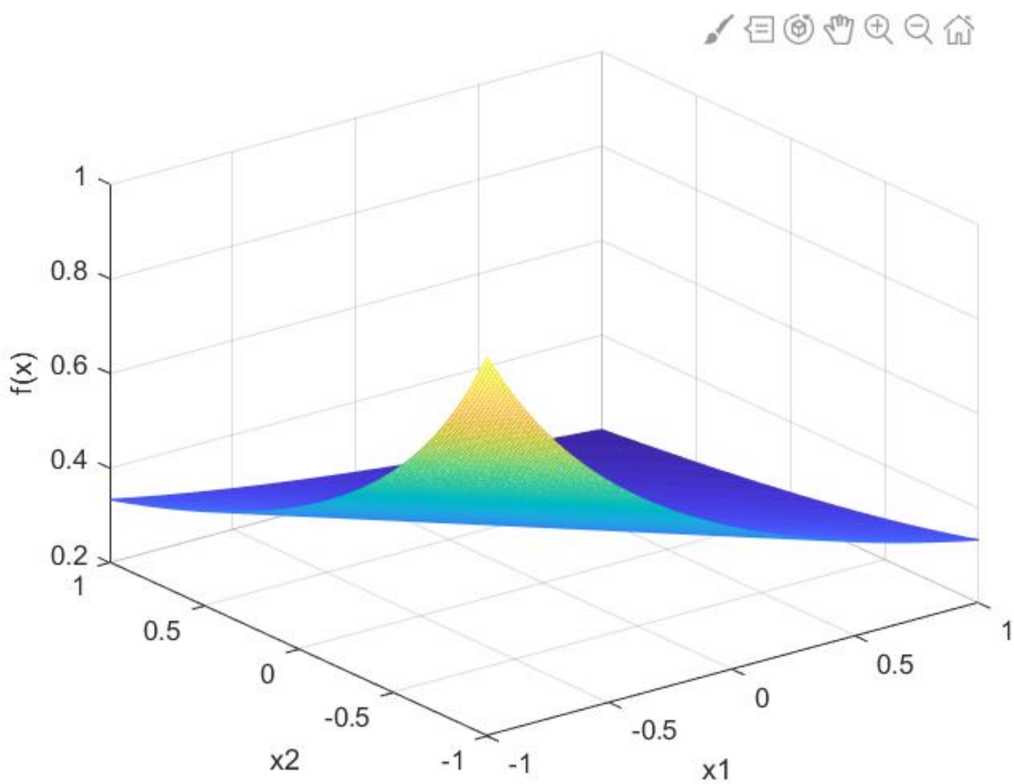
این بخش توابع توزیع تعریف شده و سپس بر روی مجموعه‌های فازی حلقه if می‌زند و توابع مثلثی عضویت (`mu_A_x1` و `mu_A_x2`) را بر اساس تکرار فعلی تعریف می‌کند. سپس `g_bar`، `num` و `den` را برای استفاده‌های بعدی محاسبه می‌کند.

این بخش استنتاج را محاسبه می‌کند. سپس g_bar، num وden را برای استفاده‌های بعدی محاسبه می‌کند

```
f_x = num./den;
g_x = 1./(3+x1+x2);
```

```
figure1 = figure('Color', [1 1 1]);
mesh(x1, x2, f_x);
xlabel('x1');
ylabel('x2');
zlabel('f(x)');
```

در نهایت، دراین بخش با استفاده از تابع `mesh` یک نمودار سه‌بعدی از خروجی فازی (`f_x`) ایجاد می‌کنیم و محورها را برچسب‌گذاری می‌کنیم.



دانشکده مهندسی برق

برای کران مرتبه دوم نیز داریم:

```
clc; clear; close all;
```

```
alfa = -1;
beta = 1;
h = 0.25;
N = 9;
x1 = alfa:0.01:beta;
x2 = x1;
[~, n1] = size(x1);
[~, n2] = size(x2);
e1 = beta * ones(1, N+1);
e2 = beta * ones(1, N+1);
```

ابتدا مقدار دهی اولیه میکنیم تمام متغیرهارا(اندازه گام و متغیرها وبازه ایی که در آن الفا و بتا با گام ۰.۰۱ نشان داده میشوند)

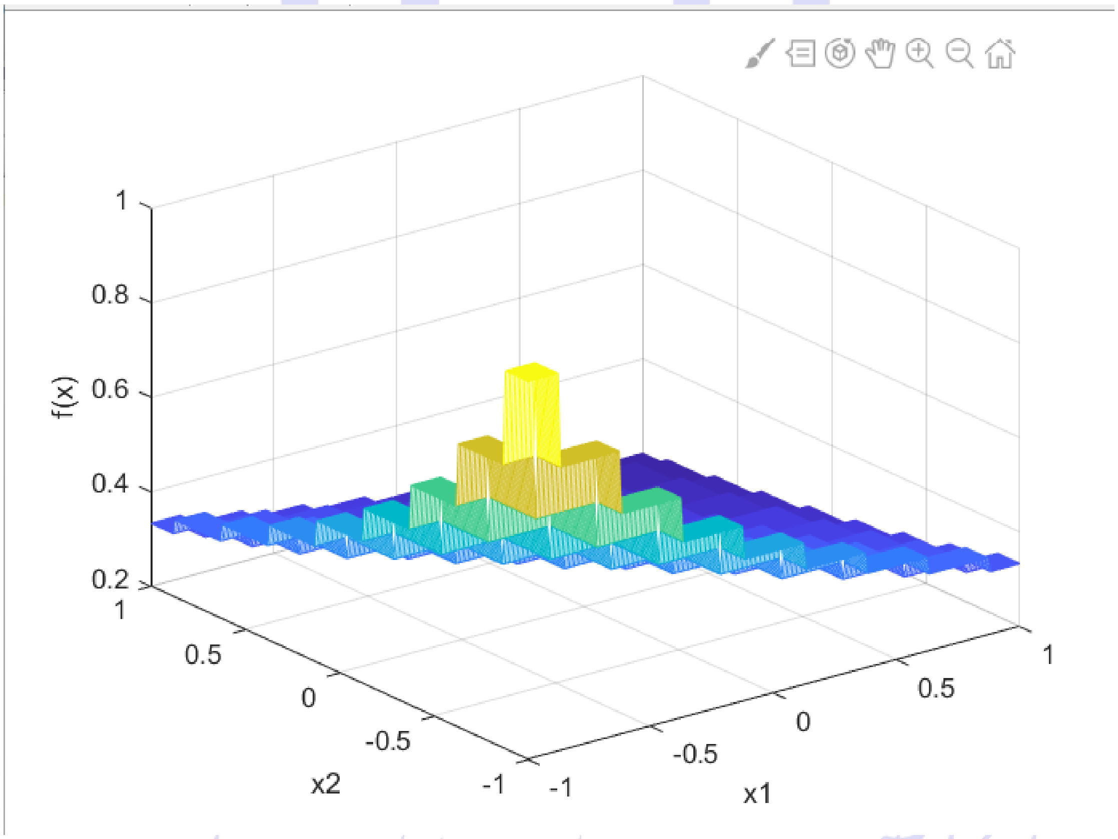
سپس شبکه را تولید میکنیم:

```
for j = 1:N
    e1(j) = alfa + h * (j-1);
    e2(j) = alfa + h * (j-1);
end
```

```
f_x=zeros(n1,n2);
for k1=1:n1
    for k2=1:n2

        i1=min(find(e1<=x1(1,k1),1,'last'),find(e1>=x1(1,k1),1));
        i2=min(find(e2<=x2(1,k2),1,'last'),find(e2>=x2(1,k2),1));

        if x1(1,k1)>=e1(1,i1)&& x1(1,k1)<=.5*(e1(1,i1)+e1(1,1+i1))&& x2(1,k2)>=e2(1,i2)&& x2(1,k2)<=.5*(e2(1,i2)+e2(1,1+i2))
            p=0;
            q=0;
        elseif x1(1,k1)>=e1(1,i1)&& x1(1,k1)<=.5*(e1(1,i1)+e1(1,1+i1))&& x2(1,k2)>=0.5*(e2(1,i2)+e2(1,1+i2))&& x2(1,k2)<=e2(1,1+i2)
            p=0;
            q=1;
        elseif x1(1,k1)>=.5*(e1(1,i1)+e1(1,1+i1))&& x1(1,k1)<=e1(1,1+i1)&& x2(1,k2)>=e2(1,i2)&& x2(1,k2)<=0.5*(e2(1,i2)+e2(1,1+i2))
            p=1;
            q=0;
        elseif x1(1,k1)>=.5*(e1(1,i1)+e1(1,1+i1))&& x1(1,k1)<=e1(1,1+i1)&& x2(1,k2)>=0.5*(e2(1,i2)+e2(1,1+i2))&& x2(1,k2)<=e2(1,1+i2)
            p=1;
            q=1;
        end

        f_x(k1,k2)=1/(3+e1(1,i1+p)+e2(1,i2+q));
    end
end
```

این بخش از کدمقدار تابع را در هر جفت از نقاط با حلقه های تو در تو در شبکه بررسی میکند.

```
[x1, x2] = meshgrid(x1, x2);
figure1 = figure('Color', [1 1 1]);
mesh(x1, x2, transpose(f_x));
xlabel('x1');
ylabel('x2');
zlabel('f(x)');
```



دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق

نتیجه گیری

همانطور که دیدیم در طراحی سیستم با کران مرتبه اول از ۲۶۰۱ قاعده فازی و در کران مرتبه دوم قاعده فازی ۸۱ استفاده کردیم.

در هردو روش دقت تقریب ۰.۱ فرض شد پس در محاسبه با کران دوم حجم محاسبات کاهش پیدا کرد.

سوال سوم:

میخواهیم یک سیستم شناخت فازی را پیاده‌سازی کنیم. منظور از "سیستم شناخت فازی" این است که از اطلاعات فازی (به جای اطلاعات دقیق) برای کنترل یا مدل‌سازی یک سیستم استفاده می‌شود. در این کد، یک مدل فازی برای تقریب یک سیگنال ورودی پیشنهاد می‌شود.

حالا به توضیح هر بخش از کد می‌پردازم:

```
%% Initializing
M=4; %Number of membership functions
num_train = 100; % Number of training data points
total_data_points = 500;
learning_rate = 0.01; % A constant stepsize
```

در این بخش، متغیرها و ثابت‌های مورد استفاده در کد تعریف می‌شوند، همچنین مقادیر اولیه برای متغیرها تعیین می‌شود.

طراحی مقادیر اولیه برای عضویت:

```
x_bar=zeros (num_train, M);
g_bar=zeros (num_train, M);
sigma=zeros (num_train, M);
y=zeros(total_data_points, 1);
u=zeros(total_data_points, 1);
x=zeros(total_data_points, 1);
y_hat=zeros(total_data_points, 1);
f_hat=zeros(total_data_points, 1);
z=zeros(total_data_points, 1);
g_u=zeros(total_data_points, 1);
error = zeros(total_data_points, 1);

rng(53); % Set the random seed
u(1)=-1+2*rand;
y(1)=0;
g_u(1)=0.6*sin(pi*u(1))+0.3*sin(3*pi*u(1))+0.1*sin(5*pi*u(1));
f_hat(1)=g_u(1);
```

در این مرحله، مقادیر اولیه برای پارامترهای عضویت (مانند مراکز، پهنای توزیع، و وزن‌ها) مشخص می‌شود.

```
rng(53); % Set the random seed
u(1)=-1+2*rand;
y(1)=0;
g_u(1)=0.6*sin(pi*u(1))+0.3*sin(3*pi*u(1))+0.1*sin(5*pi*u(1));
f_hat(1)=g_u(1);
```

```
u_min=-1;
u_max=1;
h=(u_max-u_min)/(M-1);
for k=1:M
    x_bar(1, k)=-1+h*(k-1);
    u(1,k) =x_bar(1, k);
    g_bar(1,k)=0.6*sin(pi*u(1,k))+0.3*sin(3*pi*u(1,k))+0.1*sin(5*pi*u(1,k));
end

sigma(1,1:M) = (max(u(1,:))-min(u(1,:)))/M;

x_bar(2,:)=x_bar(1, :);
g_bar(2,:)=g_bar(1, :);
sigma(2, :)=sigma(1,:);
x_bar_initial=x_bar(1, :);
sigma_initial=sigma(1, :);
y_bar_initial=g_bar(1,:);
```

آموزش سیستم فازی:

```
for q=2:num_train
    b=0;
    a=0;
    x(q)=-1+2*rand;
    u(q)=x(q);

g_u(q)=0.6*sin(pi*u(q))+0.3*sin(3*pi*u(q))+0.1*sin(5*pi*u(q
));

    % Calculate output of the identified model
    for l=1:M
        z(l)=exp(-((x(q)-x_bar(q,l))/sigma(q, l))^2);
        b=b+z(l);
        a=a+g_bar(q, l)*z(l);
    end

    f_hat(q) = a/b;
```

```
    % Update identified model output
    y(q+1) = 0.3*y(q)+0.6*y(q-1)+g_u(q);
    % Update identified model output
    y_hat(q+1) = 0.3*y(q)+0.6*y(q-1)+f_hat(q);

    % Update fuzzy sets parameters using recursive least squares
    for l=1:M
        g_bar(q+1,l)=g_bar(q,l)-learning_rate*(f_hat(q)-
g_u(q))*z(l)/b;
        x_bar(q+1,l)=x_bar(q,l)-learning_rate*((f_hat(q)-
g_u(q))/b)*(g_bar(q,l)-f_hat(q))*z(l)*2*(x(q)-
x_bar(q,l))/(sigma(q,l)^2);
        sigma      (q+1,l)=sigma(q,       l)-learning_rate*((f_hat(q)-
g_u(q))/b)*(g_bar(q,l)-f_hat(q))*z(l)*2*(x(l)-
x_bar(q,l))^2/(sigma(q,l)^3);
    end
    % Calculate error for visualization
    error(q) = g_u(q) - f_hat(q);
end

x_bar_final=x_bar(num_train,:);
sigma_final=sigma(num_train,:);
g_bar_final=g_bar(num_train,:);
```

در این بخش، سیستم فازی با استفاده از روش کمترین مربعات بازگشتی آموزش داده می‌شود. این بخش شامل یک حلقه است که از داده‌های آموزش استفاده می‌کند تا پارامترهای مدل فازی را به‌روزرسانی کند.

۱۳۰۷

اعتبار سنجی شبکه:

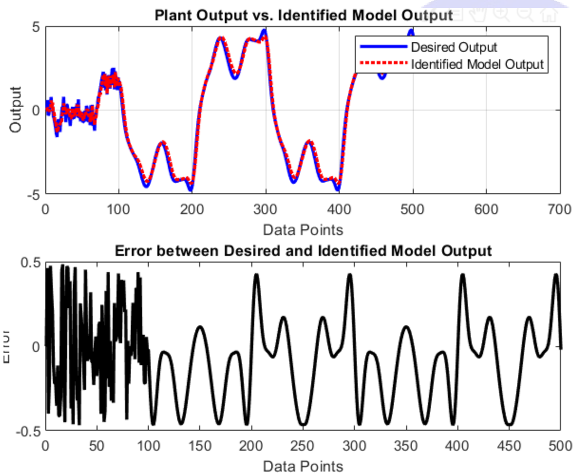
دانشگاه صنعتی خواجه نصیرالدین طوسی

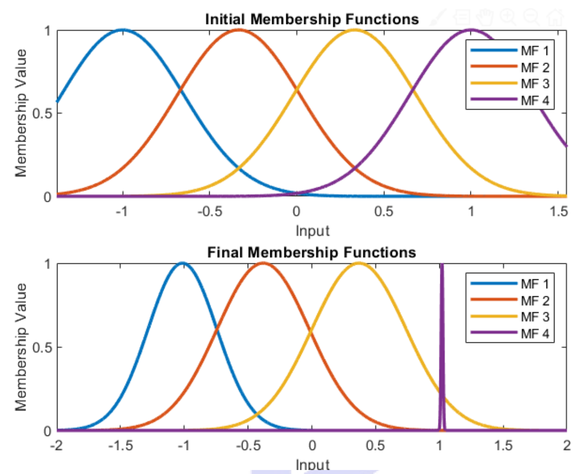
```
for q=num_train:total_data_points
    b=0;
    a=0;
    x(q)=sin(2*q*pi/200);
    u(q)=x(q);

    g_u(q)=0.6*sin(pi*u(q))+0.3*sin(3*pi*u(q))+0.1*sin(5*pi*u(q));

    % Calculate output of the identified model
    for l=1: M
        z(l) = exp(-((x(q)-x_bar(num_train,l))/sigma(num_train, l))^2);
        b = b+z(l);
        a = a+g_bar(num_train, l)*z(l);
    end
     f_hat(q) = a/b;
     y(q+1) = 0.3*y(q)+0.6*y(q-1)+g_u(q);
     y_hat(q+1) = 0.3*y(q)+0.6*y(q-1)+f_hat(q);
     % Calculate error for visualization
     error(q) = g_u(q) - f_hat(q);
end
```

دانشکده مهندسی برق

در این مرحله، مدل فازی آموزش داده شده با استفاده از داده‌های تست ارزیابی می‌شود.

در ادامه و بخش نهایی کد نیز به سراغ رسم نمودار ها میرویم:





۱۳۰۷
دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق

سوال ۴)

ابتدا کتابخانه های پایتون را ایمپورت کرده و دیتاست را لوئ میکنیم:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
!pip install --upgrade --no-cache-dir gdown
!gdown 1IPbQgkOCqQxsXEwQ_FtmoYC0WV3H4-2T

data = pd.read_csv('/content/covid.csv')
data
```

	Fever	Cough	Breathing issues	Infected
0	No	No	No	No
1	Yes	Yes	Yes	Yes
2	Yes	Yes	No	No
3	Yes	No	Yes	Yes
4	Yes	Yes	Yes	Yes
5	No	Yes	No	No
6	Yes	No	Yes	Yes
7	Yes	No	Yes	Yes
8	No	Yes	Yes	Yes
9	Yes	Yes	No	Yes
10	No	Yes	No	No
11	No	Yes	Yes	Yes
12	No	Yes	Yes	No
13	Yes	Yes	No	No

آنتروپی و اینفورمیشن گین را بصورت دستی از اول تعریف کرده و حساب میکنیم:

```
def entropy(labels):
    p = labels.value_counts() / len(labels)
    return -np.sum(p * np.log2(p + 1e-10))  # Added a small epsilon to prevent log(0) issue

def information_gain(data, feature, target):
    # Entropy of parent
    entropy_parent = entropy(data[target])

    # Entropy of child
    entropy_child = 0
    for value in data[feature].unique():
        subset = data[data[feature] == value]
        wi = len(subset) / len(data)
        entropy_child += wi * entropy(subset[target])

    return entropy_parent - entropy_child
data.iloc[:, :-1].columns
[information_gain(data, feature, 'Infected') for feature in data.iloc[:, :-1].columns]
```

در ادامه به تعریف گره ها میپردازیم:

```
def __init__(self, feature=None, label=None):
    self.feature = feature
    self.label = label
    self.children = {}

def __repr__(self):
    if self.feature is not None:
        children_repr = ', '.join(f'{value}: {child}' for value, child in self.children.items())
        return f'DecisionNode(feature="{self.feature}", children={{ {children_repr} }})'
    else:
        return f'LeafNode(label="{self.label}")'
```

حالا نوبت ساختن و تعریف تابع درخت است:

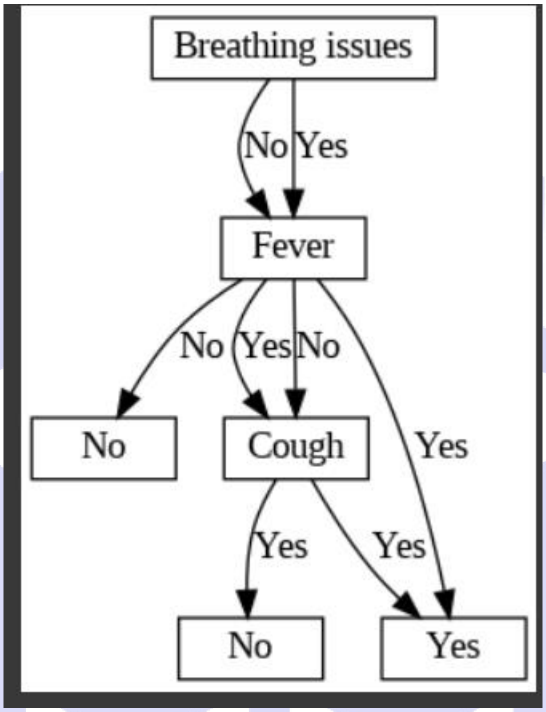
```
def make_tree(data, target):

    # leaf node?

    if len(data[target].unique()) == 1:

        return Node(label=data[target].iloc[0])


    features = data.drop(target, axis=1).columns

    if len(features) == 0 or len(data) == 0:
```

```
        return Node(label=data[target].mode()[0])


    # calculate information gain
    gains = [information_gain(data, feature, target) for feature in features]


    # greedy search to find best feature
    max_gain_idx = np.argmax(gains)
    best_feature = features[max_gain_idx]


    # make a node
    node = Node(feature=best_feature)


    # loop over the best feature
    for value in data[best_feature].unique():
        subset = data[data[best_feature] == value].drop(best_feature, axis=1)
        # display(subset)


        node.children[value] = make_tree(subset, target)
    return node
```

در ادامه هم درخت را پرینت میکنیم تا نتایج را ببینیم:

```
[8] tree = make_tree(data, 'Infected')
    tree

DecisionNode(feature="Breathing issues", children={ No: DecisionNode(feature="Fever", children={ No: LeafNode(label="No"), Yes: DecisionNode(feature="Cough", children={ Yes: LeafNode(label="No") }) }), Yes: DecisionNode(feature="Fever", children={ Yes: LeafNode(label="Yes"), No: DecisionNode(feature="Cough", children={ Yes: LeafNode(label="Yes") }) }) })
```

در ادامه درخت ساخته شده را رسم میکنیم:



بخش دوم سوال ۴)

کتابخانه ها را اضافه کرده و سپس دیتاست رو بارگزاری میکنیم:

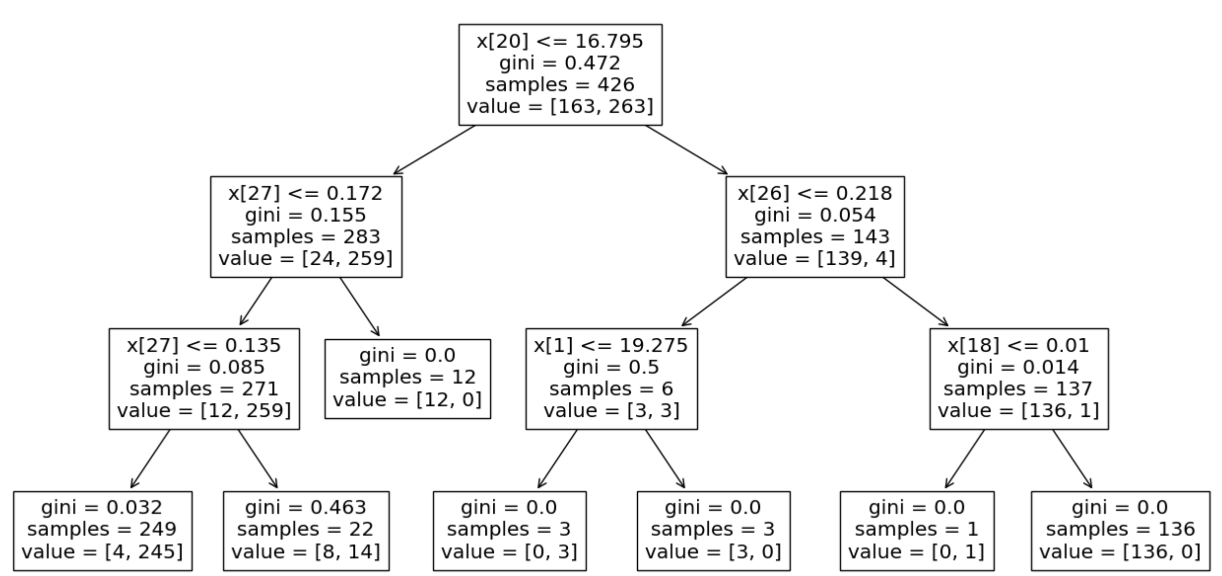
```
[15] from sklearn.datasets import load_breast_cancer
     from sklearn import tree
     cancer = load_breast_cancer()
     X, y = cancer.data, cancer.target
     X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=13)
     print(X_train.shape, X_test.shape)
     print(y_train.shape, y_test.shape)

(426, 30) (143, 30)
(426,) (143,)
```

```
clf = tree.DecisionTreeClassifier(max_depth=3, random_state=43, ccp_alpha=0)
clf.fit(X_train, y_train)

DecisionTreeClassifier
DecisionTreeClassifier(ccp_alpha=0, max_depth=3, random_state=43)
```

با دستور clf تعیین میکنیم که میخواهیم که طبقه گر با استفاده از درخت تصمیم بسازیم.

clf( ) بر روی دادههای آموزشی با استفاده از روش برازش آموزش داده میشود

این فرآیند شامل یادگیری الگوها و روابط درون دادهها برای پیشبینی موارد دیدید و دیده نشده است.



میبینیم که درخت تولید شده دارای عمق نسبتا زیاد و ۱۳ گره است.

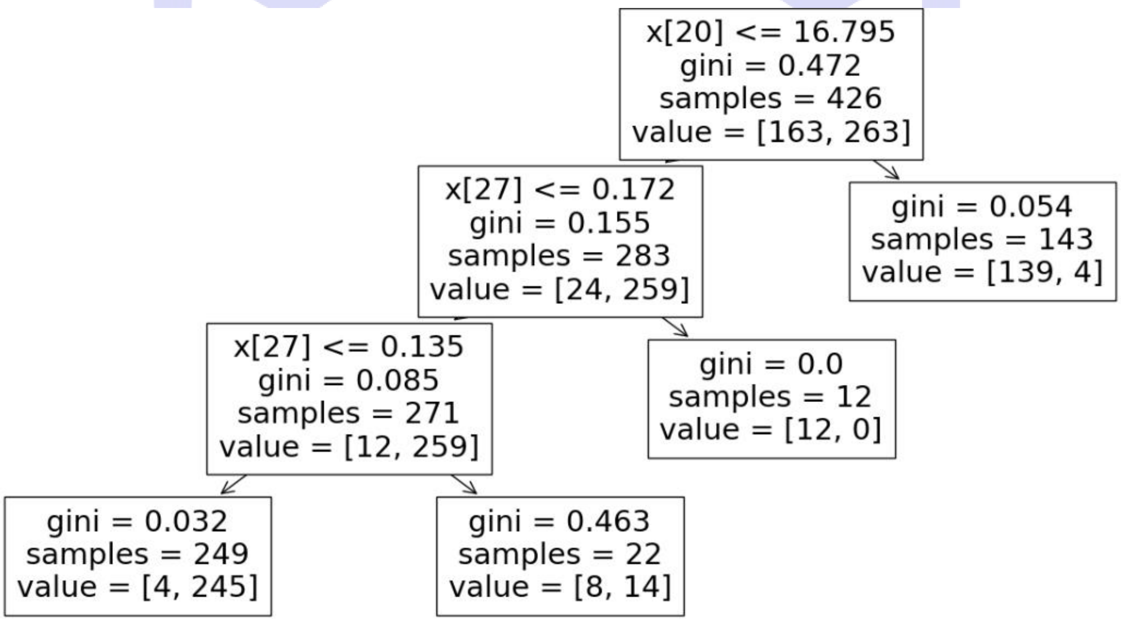
با توجه به شکل چون روت نود مربوط به ویژگی شماره ۲۰ است پس طبیعتا ویژگی شماره ۲۰ اهمیت بیشتری داشته است.

cpp alpha and max depth

```
clf = tree.DecisionTreeClassifier(max_depth=3, random_state=43, ccp_alpha=0.01)
clf.fit(X_train, y_train)
clf.predict(X_test)
clf.score(X_test, y_test)
```

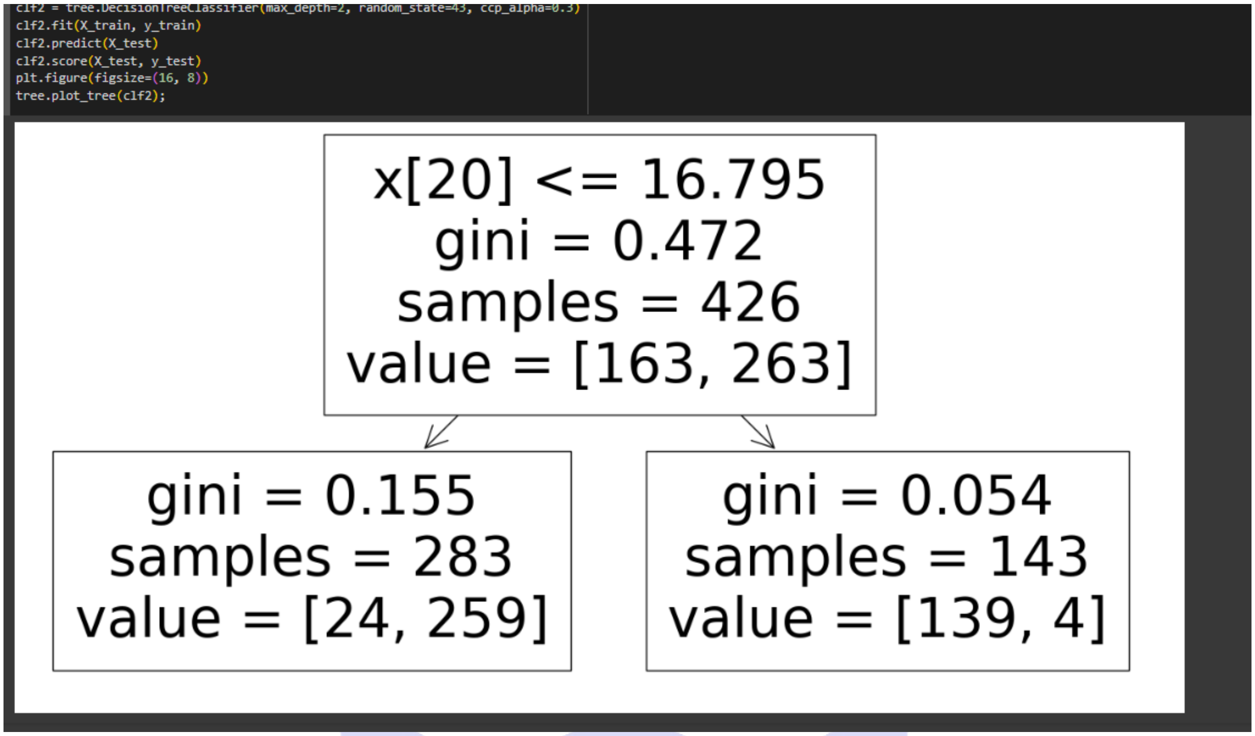
در اینجا مقادیر مکس دپث و سی سی پی آلفا رو تغییر میدیم تا حجم درخت نیز تغییر کند

اینجا پارامتر سی سی پی الفا رو برابر با ۰.۰۱ قرار دادیم در صورتی ک قبل ۰ بود.



تعداد گره ها و به صورتی عمق درخت کاهش یافته اما دقت ان ثابت مانده است.

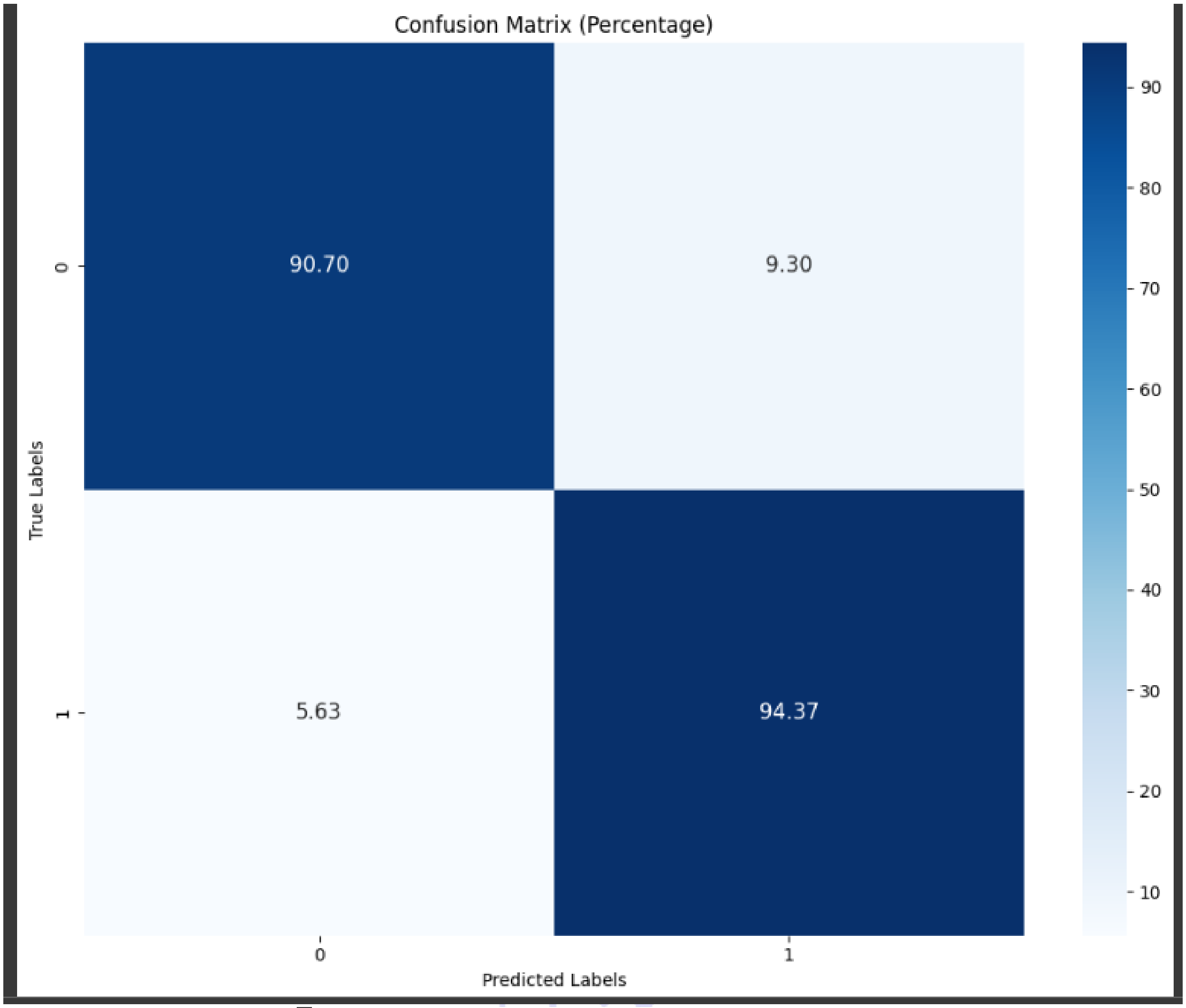
دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق



این بار هم سی سی پی الفا و هم مکس دپث را تغییر دادم میبینیم که دقا کمی کاهش داشته و حجم درخت هم چشمگیر کاهش یافته است.

حال باید بررسی کنیم میزان خطا و دقت و شاخص های دیگر و کانفیوژن ماتریس را

۱۳۰۷
دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق




```
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.91      0.91        43
           1       0.94      0.94      0.94        71

    accuracy                           0.93       114
   macro avg       0.93      0.93      0.93       114
weighted avg       0.93      0.93      0.93       114
```

دانشکده مهندسی برق

https://colab.research.google.com/drive/1el-7EtCCfjNyCRZ2Cenyp11JsBH84jAB?usp=sharing