

UNIVERZITET U BIHAĆU
TEHNIČKI FAKULTET
BIHAĆ

Auditorne vježbe iz predmeta
VJEŠTAČKA INTELIGENCIJA I EKSPERTNI SISTEMI

Una Drakulić, MA elektrotehnike
Viši asistent

MAŠINSKO UČENJE (MACHINE LEARNING)

Mašinsko učenje (engl. *Machine Learning*, *ML*) je grana vještačke inteligencije koja omogućava računarima da **uče iz podataka** i poboljšavaju svoje performanse bez eksplicitnog programiranja.

Osnovna ideja je da algoritam pronađe **šablone (pattern-e)** i **odnose među varijablama** kako bi mogao donositi odluke, predviđanja ili klasifikacije.

Mašinsko učenje se dijeli na tri glavne kategorije:

a) Nadzirano učenje (engl. *Supervised Learning*)

U ovom pristupu algoritam uči na **označenim podacima** (engl. *labeled data*), gdje su poznati ulazi i odgovarajući izlazi.

- **Cilj:** Naučiti funkciju $f(x) \approx y$ koja preslikava ulazne podatke x u željene izlaze y .
- **Primjene:** Predikcija napona, temperature, cijene, klasifikacija kvarova, itd.
- **Algoritmi:** Linearna regresija, logistička regresija, SVM, k-NN, neuronske mreže.

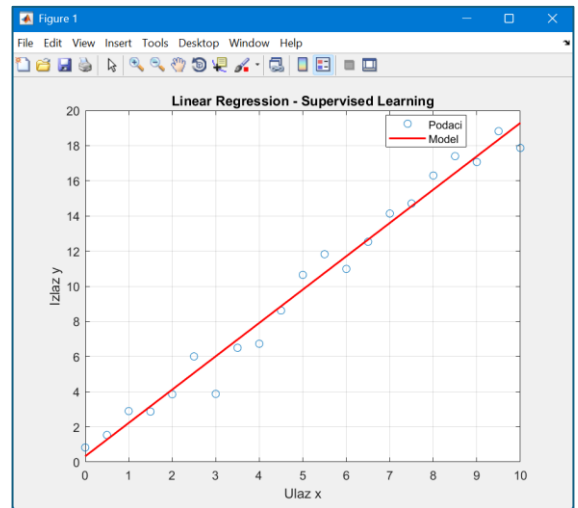
Primjer1

```
% Nadzirano učenje - Linearna regresija
x = (0:0.5:10)'; % Ulazni podaci
y = 2*x + randn(size(x)); % Izlazni podaci
sa šumom

% Treniranje modela
mdl = fitlm(x, y);

% Predikcija i prikaz
x_pred = (0:0.1:10)';
y_pred = predict(mdl, x_pred);

plot(x, y, 'o');
hold on;
plot(x_pred, y_pred, 'r', 'LineWidth',
1.5);
title('Linearna regresija - Nadzirano
ucenje');
xlabel('Ulaz x'); ylabel('Izlaz y');
legend('Podaci', 'Model');
grid on;
```



Slika 1

Dobijeni graf prikazuje tačkaste podatke i crvenu liniju koja predstavlja naučeni model linearne regresije.

Primjer2

Primjer nadziranog učenja za PV sistem, gdje su ulazi sunčeva radijacija (G) i temperatura (T), a izlazi napon (V) i struja (I) PV panela. Simulira se stvarni hibridni energetska sistem i pokazuje kako neuronska mreža može predviđati performanse panela.

Generisanje sintetičkih podataka

- Ulazi:
 - G – sunčeva radijacija [W/m²] (100–1000 W/m²)
 - T – temperatura [°C] (15–40 °C)
- Izlazi:
 - V – napon PV panela [V]
 - I – struja PV panela [A]

Koristimo linearno-nelinearni model PV panela:

$$V = V_{oc} - k_v(T - 25) - \alpha I, I = I_{sc} \cdot \frac{G}{1000}$$

```
clc; clear; close all;

%% 1. Generisanje ulaza (G i T)
num_samples = 200;
G = 100 + (1000-100).*rand(1,num_samples); % Sunčeva radijacija [W/m²]
T = 15 + (40-15).*rand(1,num_samples); % Temperatura [°C]

inputs = [G; T]; % Dimenzija: [2 × num_samples]

%% 2. Generisanje izlaza (V i I) sa šumom
V_oc = 38; % Otvoreni napon PV panela [V]
I_sc = 5; % Struja kratkog spoja [A]
k_v = 0.3; % Koeficijent temperature
alpha = 0.01; % Interni otpor

I = I_sc .* (G./1000) + 0.05*randn(1,num_samples); % Sintetički I sa šumom
V = V_oc - k_v*(T-25) - alpha*I + 0.1*randn(1,num_samples); % Sintetički V sa šumom

outputs = [V; I]; % Dimenzija: [2 × num_samples]

%% 3. Podjela na trening i test
train_ratio = 0.7;
idx = 1:round(num_samples*train_ratio);

inputs_train = inputs(:,idx);
outputs_train = outputs(:,idx);

inputs_test = inputs(:,idx(end)+1:end);
outputs_test = outputs(:,idx(end)+1:end);

%% 4. Kreiranje neuronske mreže
hidden_neurons = 10;
net = fitnet(hidden_neurons); % Mreža za regresiju

net.trainParam.showWindow = true;
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-4;

%% 5. Treniranje mreže
[net, tr] = train(net, inputs_train, outputs_train);

%% 6. Testiranje
```

```

outputs_pred = net(inputs_test);

%% 7. Vizuelizacija
figure;

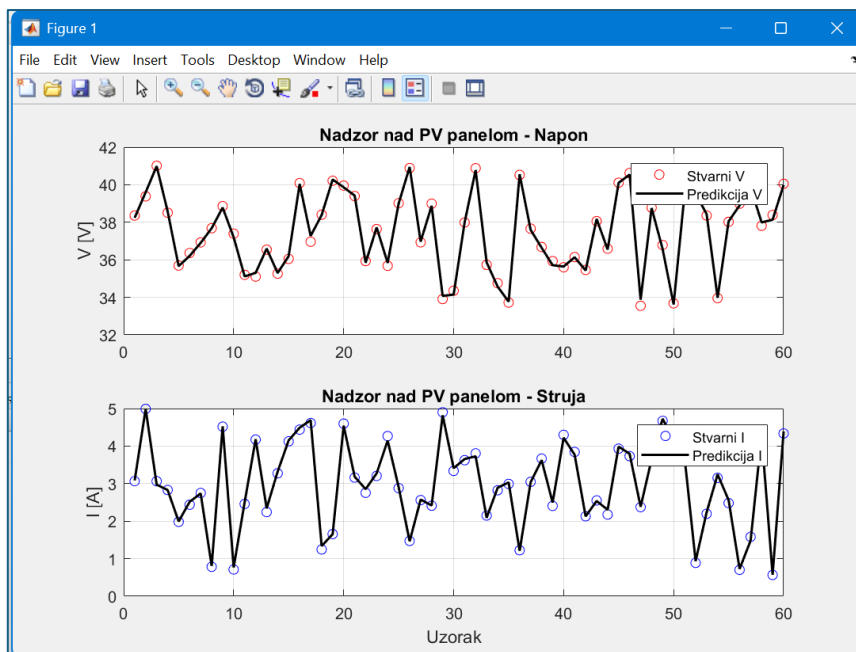
subplot(2,1,1);
plot(outputs_test(1,:), 'ro', 'DisplayName', 'Stvarni V');
hold on;
plot(outputs_pred(1,:), 'k-', 'LineWidth', 1.5, 'DisplayName', 'Predikcija V');
ylabel('V [V]');
title('Nadzor nad PV panelom - Napon');
legend; grid on;

subplot(2,1,2);
plot(outputs_test(2,:), 'bo', 'DisplayName', 'Stvarni I');
hold on;
plot(outputs_pred(2,:), 'k-', 'LineWidth', 1.5, 'DisplayName', 'Predikcija I');
ylabel('I [A]');
xlabel('Uzorak');
title('Nadzor nad PV panelom - Struja');
legend; grid on;

%% 8. Izračun MSE
mse_V = mse(outputs_test(1,:) - outputs_pred(1,:));
mse_I = mse(outputs_test(2,:) - outputs_pred(2,:));
fprintf('MSE V = %.4f, MSE I = %.4f\n', mse_V, mse_I);

```

MSE V = 0.0157, MSE I = 0.0033



Slika 2

Primjer3

Nadzor i predikcija performansi vjetroturbine pomoću neuronske mreže. Snaga vjetra se opisuje relacijom:

$$P = \frac{1}{2} \rho A C_p v^3$$

gdje su:

- $\rho = 1.225 \text{ kg/m}^3$ – gustina zraka,
- $A = \pi r^2$ – površina rotora,
- C_p – koeficijent snage (0.25–0.45),
- v – brzina vjetra [m/s].

Moment generatora se određuje kao: $T = \frac{P}{\omega}$, gdje je ω ugaona brzina, proporcionalna brzini vjetra.

```
clc; clear; close all;

%% 1. Generisanje ulaznih podataka
num_samples = 200;
v = 2 + (15-2).*rand(1,num_samples); % Brzina vjetra [m/s]

%% 2. Parametri turbine
rho = 1.225; % Gustina zraka [kg/m^3]
r = 40; % Poluprečnik rotora [m]
A = pi * r^2; % Površina rotora
Cp = 0.4; % Koeficijent snage
k_w = 0.12; % Koeficijent proporcionalnosti za ugaonu brzinu

%% 3. Izračunavanje izlaza (P i T)
P = 0.5 * rho * A * Cp .* (v.^3) / 1e6; % Snaga u MW
omega = k_w * v; % Ugaona brzina [rad/s]
T = (P * 1e6) ./ omega; % Moment [Nm]

% Dodavanje šuma radi realnosti
P = P + 0.05*randn(1,num_samples);
T = T + 10*randn(1,num_samples);

inputs = v;
outputs = [P; T]; % Izlazi: snaga i moment

%% 4. Podjela na trening i test
train_ratio = 0.7;
idx = 1:round(num_samples*train_ratio);
x_train = inputs(:,idx);
y_train = outputs(:,idx);
x_test = inputs(:,idx(end)+1:end);
y_test = outputs(:,idx(end)+1:end);

%% 5. Kreiranje i treniranje neuronske mreže
net = fitnet(10); % 10 neurona u skrivenom sloju
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-4;

[net, tr] = train(net, x_train, y_train);

%% 6. Testiranje i poređenje
y_pred = net(x_test);

figure;
subplot(2,1,1);
plot(y_test(1,:), 'ro', 'DisplayName', 'Stvarna P');
hold on;
plot(y_pred(1,:), 'k-', 'LineWidth', 1.3, 'DisplayName', 'Predikcija P');
```

```

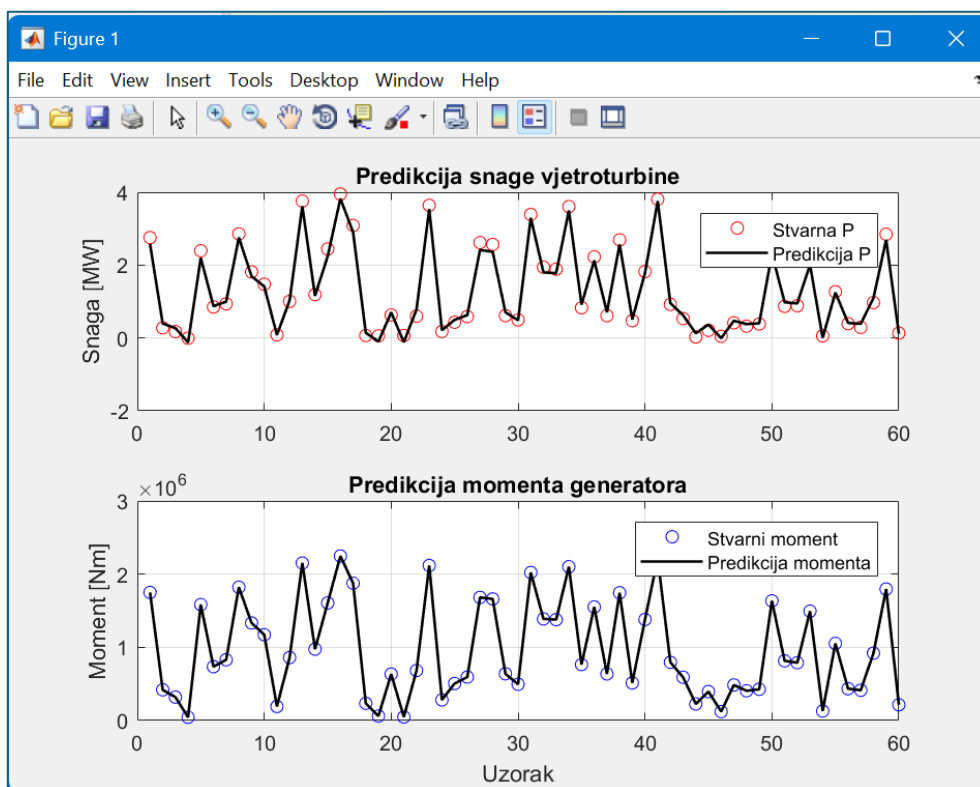
ylabel('Snaga [MW]');
title('Predikcija snage vjetroturbine');
legend; grid on;

subplot(2,1,2);
plot(y_test(2,:), 'bo', 'DisplayName', 'Stvarni moment');
hold on;
plot(y_pred(2,:), 'k-', 'LineWidth', 1.3, 'DisplayName', 'Predikcija momenta');
ylabel('Moment [Nm]');
xlabel('Uzorak');
title('Predikcija momenta generatora');
legend; grid on;

%% 7. Greška
mse_P = mse(y_test(1,:) - y_pred(1,:));
mse_T = mse(y_test(2,:) - y_pred(2,:));
fprintf('MSE snaga = %.4f, MSE moment = %.4f\n', mse_P, mse_T);

```

MSE snaga = 0.0131, MSE moment = 133.0264



Slika 3

Primjer4 Statička regresija: predikcija temperature prostorije

Potrebno je kreirati model koji može procjeniti temperaturu prostorije Troom, u datom trenutku, na osnovu vrijednosti senzora koji mjere:

- Tout — vanjska temperatura (°C)
- HVAC — trenutno opterećenje/izlaz HVAC sistema (0-1 ili 0-100% snage)
- Occ — broj prisutnih osoba
- RH — relativna vlažnost (%)

Model treba da uči na osnovu historijskih podataka, a nakon treniranja da se može koristiti za predikciju temperature u stvarnom vremenu, što omogućava inteligentno upravljanje HVAC sistemom radi poboljšanja energetske efikasnosti i udobnosti korisnika.

HVAC(engl. Heating, Ventilation, and Air Conditioning - Grijanje, ventilacija i klimatizacija) integrisani sistem koji se koristi za upravljanje temperaturom, vlažnošću i kvalitetom zraka u zatvorenim prostorima (zgradama, laboratorijama, učionicama, server sobama itd.)

```
clc; clear; close all;

% 1) Generisanje sintetičkih podataka (realistične varijacije)
N = 500;
t = (1:N);

% Vanjska temp: dnevni ciklus (sinusoidno) + promjene
Tout = 10 + 8*sin(2*pi*t/144) + 0.8*randn(1,N); % °C

% HVAC output zavisi od kontrole (nekontrolisano sintetički)
HVAC = 0.5 + 0.4*sin(2*pi*(t+20)/200) + 0.1*randn(1,N);
HVAC = min(max(HVAC, 0), 1);

% Broj osoba (diskretan) slučajan raspored
Occ = poissrnd(1.2,1,N);
Occ(Occ>5)=5;

% Relativna vlažnost (RH)
RH = 40 + 20*sin(2*pi*(t+50)/300) + 3*randn(1,N);
RH = min(max(RH,20),90);

% Stvarna sobna temp model
alpha0 = 5.0;
alpha1 = 0.55; % utjecaj vanjske temperature
alpha2 = -6.0; % HVAC hladi (negativan utjecaj ako HVAC=1 -> niža temp)
alpha3 = 0.2; % svaka osoba doda toplinu
alpha4 = 0.01; % blagi utjecaj vlažnosti
Troom = alpha0 + alpha1*Tout + alpha2*HVAC + alpha3*Occ + alpha4*RH +
0.5*randn(1,N);

% 2) Sastavi ulaze i izlaze u oblik pogodan za fitnet (dim: [features x
samples])
inputs = [Tout; HVAC; Occ; RH]; % 4 x N
targets = Troom; % 1 x N

% 3) Podjela podataka: 70% train, 15% val, 15% test
trainN = round(0.7*N);
valN = round(0.15*N);
testN = N - trainN - valN;

idx = randperm(N);
train_idx = idx(1:trainN);
```

```

val_idx = idx(trainN+1:trainN+valN);
test_idx = idx(trainN+valN+1:end);

inputs_train = inputs(:,train_idx);
targets_train = targets(:,train_idx);
inputs_val = inputs(:,val_idx);
targets_val = targets(:,val_idx);
inputs_test = inputs(:,test_idx);
targets_test = targets(:,test_idx);

% 4) Kreiranje i treniranje mreže (regresija)
hiddenNeurons = 12;
net = fitnet(hiddenNeurons,'trainlm'); % Levenberg-Marquardt

% postavi vlastitu podjelu (da MATLAB koristi naše skupove)
net.divideFcn = 'divideind';
net.divideParam.trainInd = train_idx;
net.divideParam.valInd = val_idx;
net.divideParam.testInd = test_idx;

% (opcionalno) regularizacija i tren. parametri
net.trainParam.epochs = 300;
net.trainParam.goal = 1e-4;
net.performParam.regularization = 0.01;

% Treniraj
[net,tr] = train(net, inputs, targets);

% 5) Testiranje
pred_all = net(inputs); % 1 x N (predikcije za sve uzorke)
pred_test = pred_all(:, test_idx);
true_test = targets(:, test_idx);

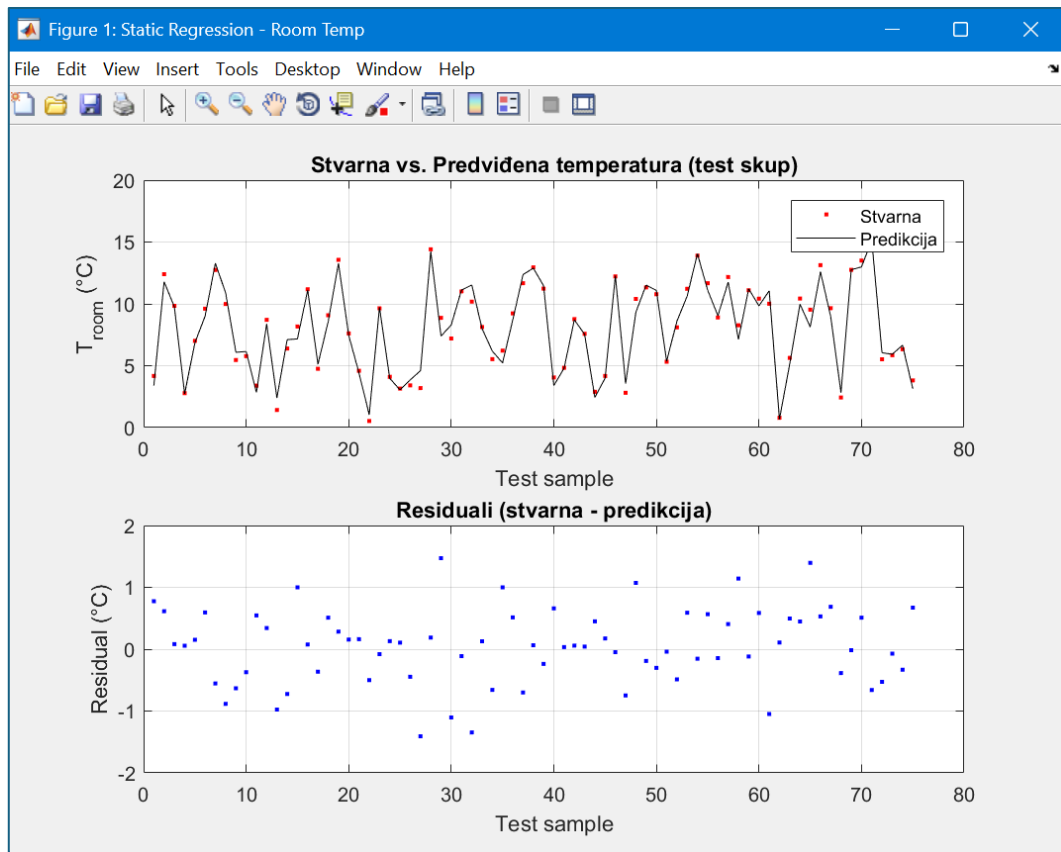
% 6) Vizualizacija
figure('Name','Static Regression - Room Temp');
subplot(2,1,1);
plot(true_test,'r. '); hold on;
plot(pred_test,'k- ');
xlabel('Test sample'); ylabel('T_{room} (°C)');
legend('Stvarna','Predikcija');
title('Stvarna vs. Predviđena temperatura (test skup)');
grid on;

subplot(2,1,2);
plot(true_test - pred_test,'b. ');
xlabel('Test sample'); ylabel('Residual (°C)');
title('Residuali (stvarna - predikcija)');
grid on;

% 7) MSE i statistika
mse_val = mse(true_test - pred_test);
mae_val = mean(abs(true_test - pred_test));
fprintf('Static model: MSE = %.4f °C^2, MAE = %.4f °C\n', mse_val, mae_val);

```

Static model: MSE = 0.3684 °C², MAE = 0.4794 °C



Slika 4

Graf residuala pomaže u pronalaženju pristranosti / sistematskih pogrešaka.

b) Nenadzirano učenje (engl. Unsupervised Learning)

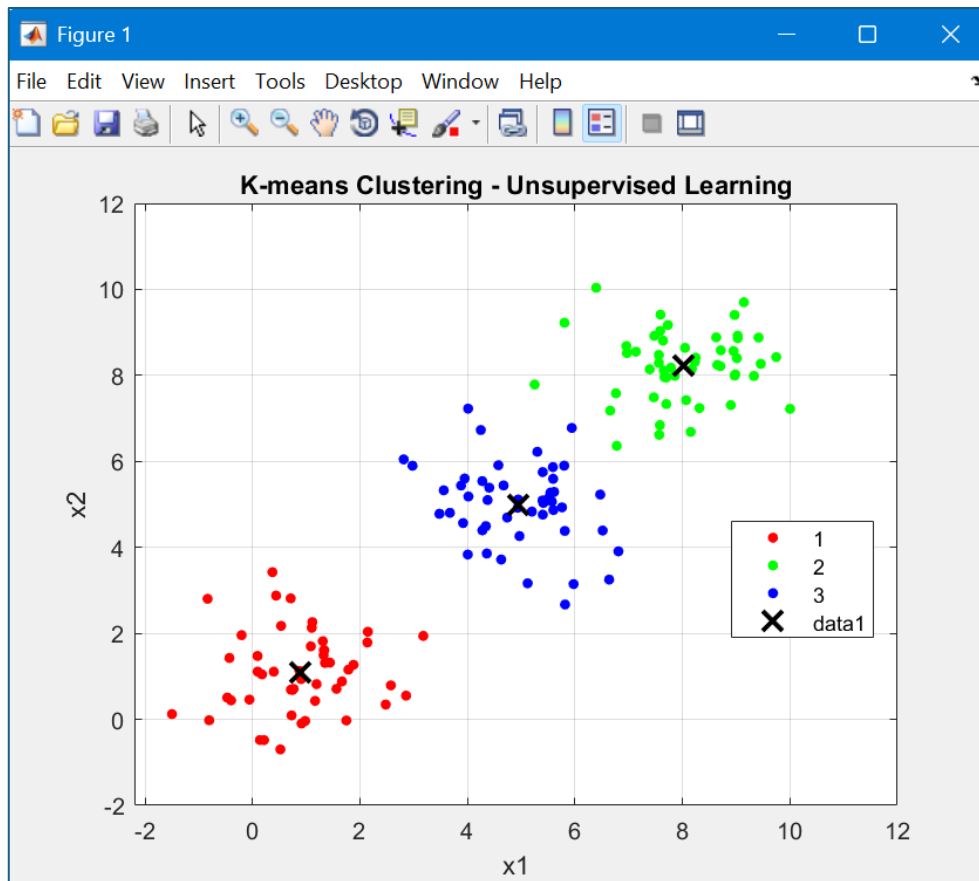
Algoritam uči bez poznatih izlaza, tj. podaci nisu označeni. Cilj je pronaći skrivene strukture u podacima, kao što su grupe ili pravci maksimalne varijacije.

- Cilj: Otkriti uzorke, klastere ili latentne dimenzije u podacima.
- Primjene: Grupisanje tipova potrošača energije, analiza radnih režima sistema.
- Algoritmi: K-means, PCA, Autoencoder.

Primjer5

```
% Nenadzirano učenje - K-means klasterovanje
data = [randn(50,2)+1; randn(50,2)+5; randn(50,2)+8];
[idx, C] = kmeans(data, 3);

gscatter(data(:,1), data(:,2), idx);
hold on;
plot(C(:,1), C(:,2), 'kx', 'MarkerSize', 12, 'LineWidth', 2);
title('K-means Clustering - Unsupervised Learning');
xlabel('x1'); ylabel('x2');
grid on;
```



Slika 5

Dobiveni podaci su grupisani u tri klastera, a centri klastera su označeni crnim „X“. Ovim se pokazuje sposobnost algoritma da sam pronade obrasce bez unaprijed poznatih oznaka.

Primjer6 Na osnovu dnevne potrošnje električne energije u domaćinstvima, sistem treba automatski da grupiše potrošače u kategorije:

- potrošači s malom potrošnjom,
- potrošači sa srednjom potrošnjom,
- potrošači s velikom potrošnjom.

*Nema unaprijed poznatih oznaka (etiketa) — sistem sam otkriva obrasce u podacima. Nenadzirano učenje se koristi kada podaci nemaju označene izlaze. Cilj je pronaći strukturu ili sličnost između uzoraka. Najpoznatija metoda je K-means clustering — algoritam koji dijeli podatke u K grupa (klastera) na osnovu međusobne sličnosti.

```
%% Primjer 5: Nenadzirano učenje - K-means clustering podataka o potrošnji
električne energije
% MATLAB R2017b kompatibilno

% Generisanje podataka (npr. potrošnja u kWh tokom dana)
rng(1); % zbog ponovljivosti rezultata
data = [randn(50,1)*2 + 10; randn(50,1)*3 + 20; randn(50,1)*4 + 35];
% data = 150 vrijednosti koje predstavljaju potrošnju energije (kWh)

% Broj grupa (klastera)
K = 3;

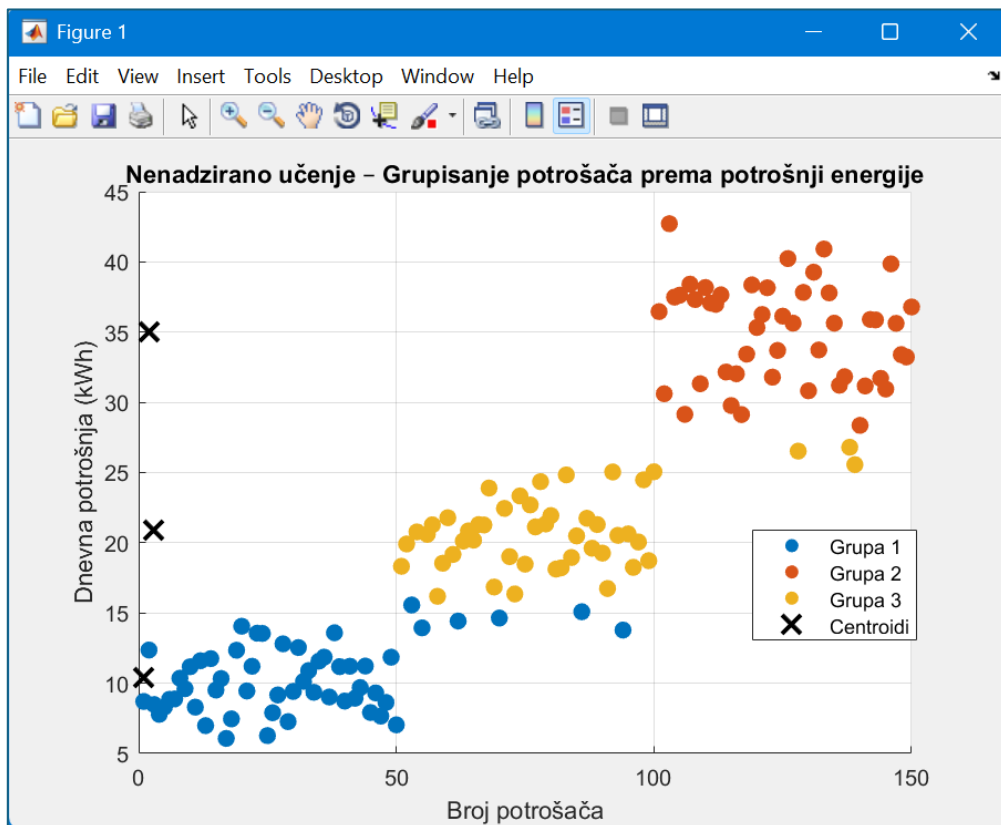
% Primjena K-means algoritma
[idx, C] = kmeans(data, K);
```

```

% Prikaz rezultata
figure;
hold on;
colors = lines(K);
for i = 1:K
    scatter(find(idx==i), data(idx==i), 60, colors(i,:), 'filled');
end
plot(C, 'kx', 'MarkerSize', 12, 'LineWidth', 2);
xlabel('Broj potrošača');
ylabel('Dnevna potrošnja (kWh)');
title('Nenadzirano učenje - Grupisanje potrošača prema potrošnji energije');
legend('Grupa 1', 'Grupa 2', 'Grupa 3', 'Centroidi');
grid on;

% Analiza grupa
disp('Prosječna potrošnja po grupi (kWh):');
disp(C);

```



Slika 6

Prosječna potrošnja po grupi (kWh):

10.4161
35.0019
20.9174

Primjer7 Grupisati radne tačke solarnog panela (*Sunčeva radijacija* (G) [W/m²], *Temperatura panela* (T) [°C], *Napon* (V) [V] i *Struja* (I) [A]) kako bi se definisali radni režimi bez unaprijed poznatih izlaza. Ulazni podaci predstavljaju mjerne vrijednosti prikupljene u različitim vremenskim uslovima rada solarnog panela.

```
clear; clc; close all;

%% 1. Ulazni podaci
% Svaka vrsta predstavlja mjerne podatke PV panela u određenim uslovima
Radiation = [600; 800; 1000; 400; 300; 950; 700; 850; 200; 500];
Temperature = [20; 25; 28; 18; 15; 27; 22; 26; 14; 19];
Voltage = [10.5; 11.8; 12.0; 9.2; 8.5; 11.9; 10.8; 11.5; 7.8; 9.5];
Current = [3.2; 4.0; 4.5; 2.8; 2.0; 4.3; 3.6; 3.9; 1.5; 2.7];

% Kreiranje matrice podataka
Data = [Radiation Temperature Voltage Current];

%% 2. Broj klastera
numClusters = 3;

%% 3. K-means klasterovanje
[cluster_idx, cluster_centers] = kmeans(Data, numClusters, 'Replicates', 10);

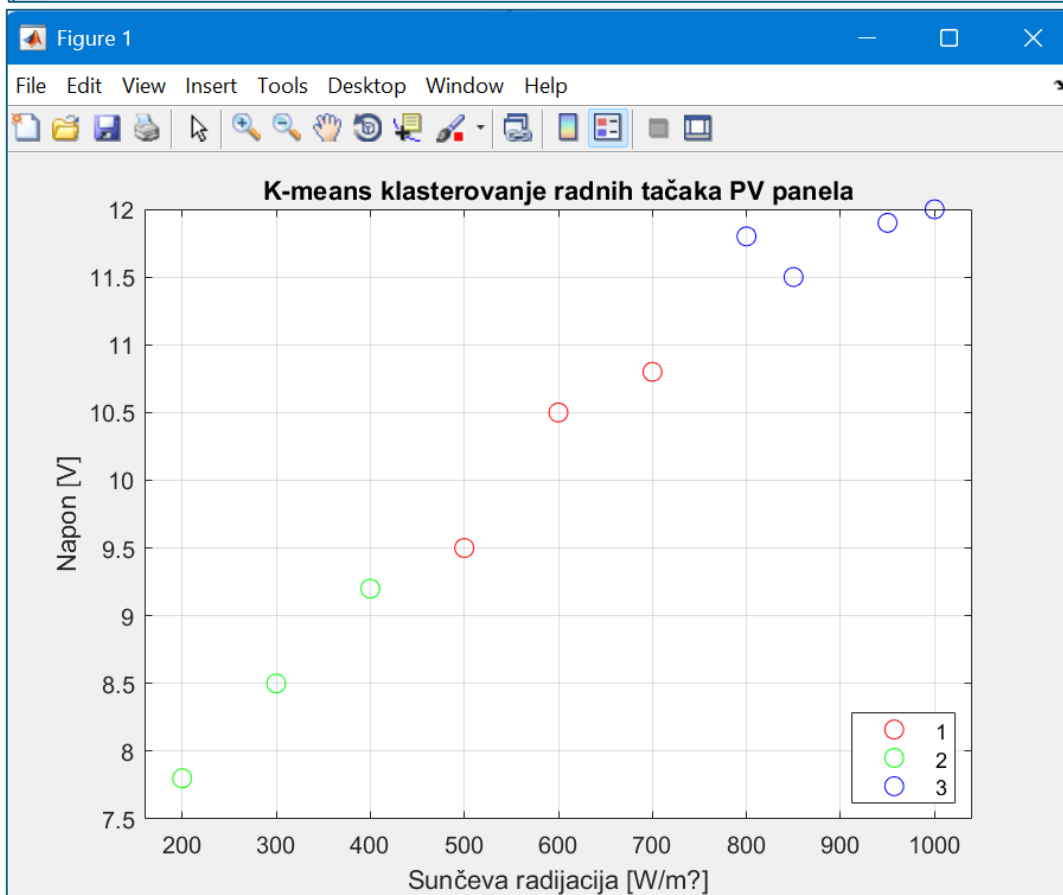
%% 4. Prikaz rezultata
disp('--- Rezultati klasterovanja ---');
Results = table(Radiation, Temperature, Voltage, Current, cluster_idx, ...
    'VariableNames',
    {'Radiation_Wm2', 'Temperature_C', 'Voltage_V', 'Current_A', 'Cluster'});
disp(Results);

%% 5. Grafički prikaz (radijacija vs. napon)
figure;
gscatter(Radiation, Voltage, cluster_idx, 'rgb', 'o', 8);
xlabel('Sunčeva radijacija [W/m?]');
ylabel('Napon [V]');
title('K-means klasterovanje radnih tačaka PV panela');
grid on;

%% 6. Prikaz centara klastera
disp('--- Centri klastera ---');
disp(array2table(cluster_centers, ...
    'VariableNames',
    {'Radiation_Wm2', 'Temperature_C', 'Voltage_V', 'Current_A'}));

%% 7. Interpretacija (komentar)
% Klaster 1 -> Niska radijacija i napon -> neefikasan režim rada
% Klaster 2 -> Umjerena radijacija i stabilan napon -> normalni radni režim
% Klaster 3 -> Visoka radijacija i visoka efikasnost -> optimalni radni režim
```

Command Window				
--- Rezultati klasterovanja ---				
Radiation_Wm2	Temperature_C	Voltage_V	Current_A	Cluster
600	20	10.5	3.2	1
800	25	11.8	4	3
1000	28	12	4.5	3
400	18	9.2	2.8	2
300	15	8.5	2	2
950	27	11.9	4.3	3
700	22	10.8	3.6	1
850	26	11.5	3.9	3
200	14	7.8	1.5	2
500	19	9.5	2.7	1
--- Centri klastera ---				
Radiation_Wm2	Temperature_C	Voltage_V	Current_A	
600	20.333	10.267	3.1667	
300	15.667	8.5	2.1	
900	26.5	11.8	4.175	



Slika 7

Primjer8 Dati su podaci o temperaturi, vlažnosti zraka, vlažnosti tla, osvjetljenosti i koncentraciji CO₂ koji se prikupljaju pomoću senzora (npr. DHT22, soil moisture sensor, BH1750, MQ135). Napisati kod koji će omogućiti analiziranje kako bi se prepoznale grupe (klasteri) radnih uslova:

- optimalni uslovi,
- pregrijavanje,
- suvi ili zasićeni uslovi.

```
clear; clc; close all;

%% 1. Simulirani senzorski podaci (npr. mjerenja u toku dana)
% temperature [°C], humidity [%], soil moisture [%], light [lux], CO2 [ppm]
Temperature = [22 24 28 35 18 20 27 30 33 25 26 21 19 32 29]';
Humidity = [60 55 50 40 80 75 52 48 45 58 54 70 72 42 49]';
SoilMoisture = [40 35 30 25 60 55 38 32 28 36 33 58 57 29 31]';
Light = [15000 20000 35000 40000 10000 12000 25000 37000 42000 21000 23000 14000 11000 38000 36000]';
CO2 = [450 480 600 720 420 430 580 700 750 500 520 460 440 680 640]';

% formiraj dataset
Data = [Temperature Humidity SoilMoisture Light CO2];

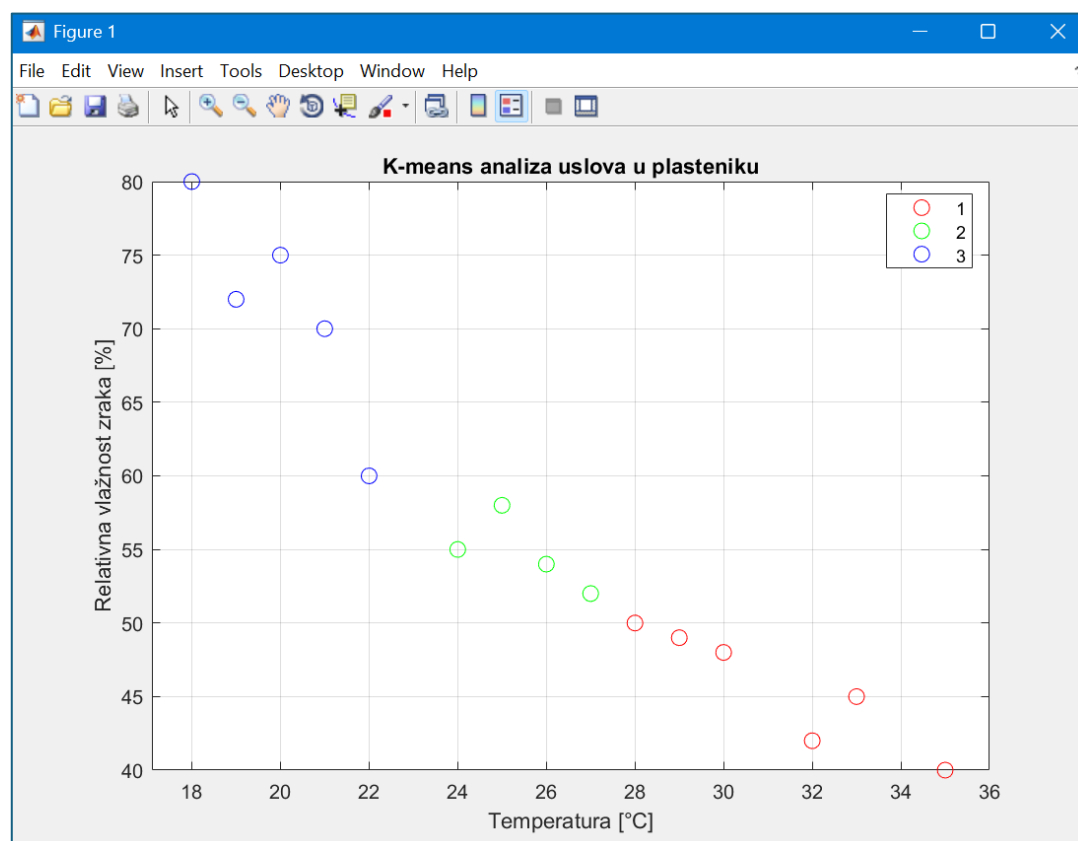
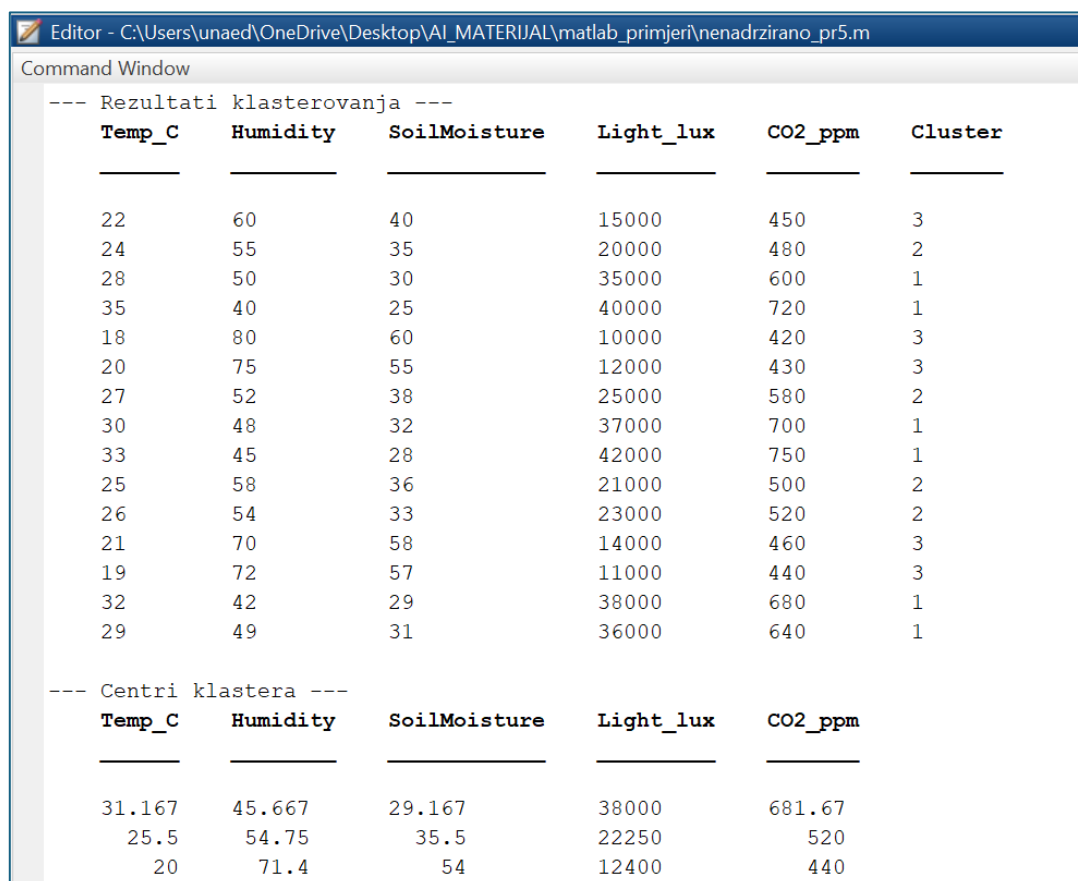
%% 2. K-means klasterovanje
numClusters = 3;
[cluster_idx, cluster_centers] = kmeans(Data, numClusters, 'Replicates', 10);

%% 3. Rezultati
disp('--- Rezultati klasterovanja ---');
Results = table(Temperature, Humidity, SoilMoisture, Light, CO2, cluster_idx, ...
    'VariableNames', {'Temp_C', 'Humidity', 'SoilMoisture', 'Light_lux', 'CO2_ppm', 'Cluster'});
disp(Results);

%% 4. Prikaz klastera (npr. Temperatura vs. Vlažnost)
figure;
gscatter(Temperature, Humidity, cluster_idx, 'rgb', 'o', 8);
xlabel('Temperatura [°C]');
ylabel('Relativna vlažnost zraka [%]');
title('K-means analiza uslova u plasteniku');
grid on;

%% 5. Centri klastera
disp('--- Centri klastera ---');
disp(array2table(cluster_centers, ...
    'VariableNames', {'Temp_C', 'Humidity', 'SoilMoisture', 'Light_lux', 'CO2_ppm'}));

%% 6. Interpretacija (komentar)
% Klaster 1 -> Optimalni uslovi (umjerena temperatura i vlaga)
% Klaster 2 -> Pregrijavanje i niska vlaga -> potreba za ventilacijom i zalijevanjem
% Klaster 3 -> Niska temperatura, visoka vlaga -> rizik od kondenzacije i gljivica
```

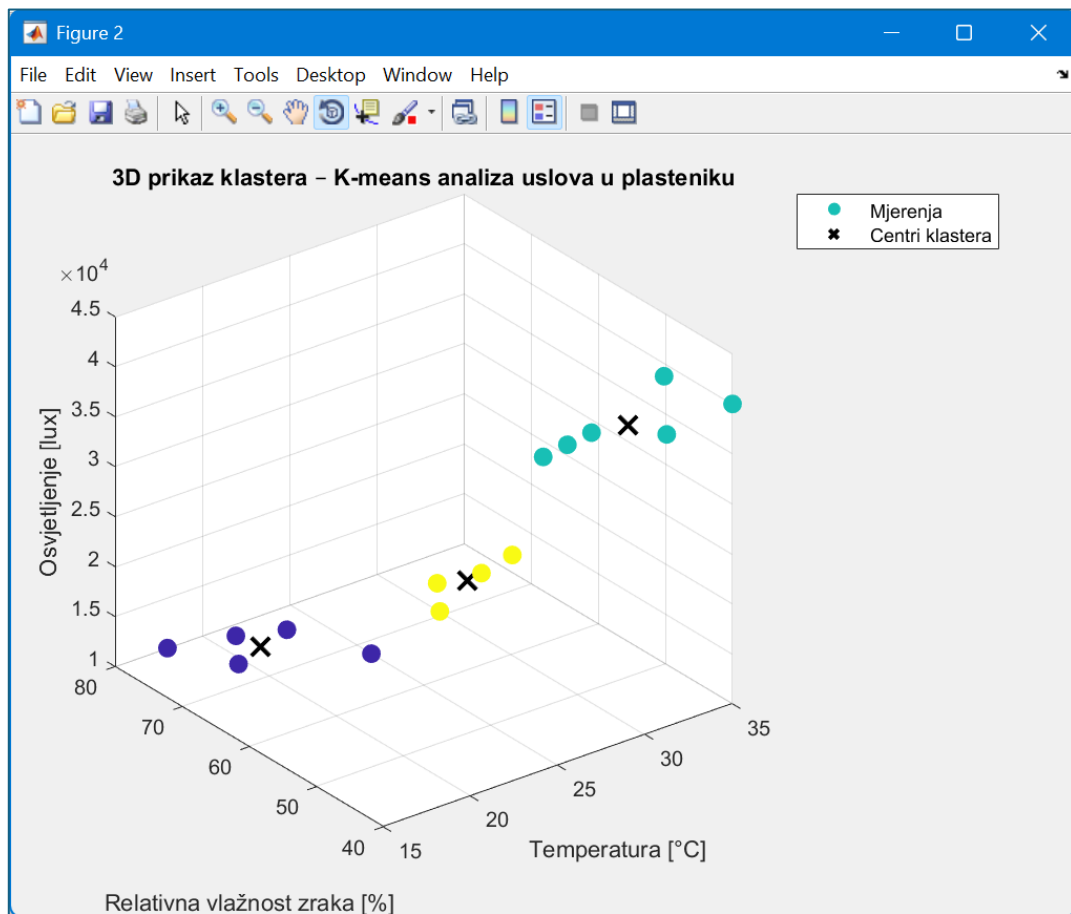


Slika 8

```

%% 3D prikaz klastera
figure;
scatter3(Temperature, Humidity, Light, 80, cluster_idx, 'filled');
hold on;
scatter3(cluster_centers(:,1), cluster_centers(:,2), cluster_centers(:,4),
150, 'kx', 'LineWidth', 2);
xlabel('Temperatura [°C]');
ylabel('Relativna vlažnost zraka [%]');
zlabel('Osvjetljenje [lux]');
title('3D prikaz klastera - K-means analiza uslova u plasteniku');
legend('Mjerenja', 'Centri klastera');
grid on;
rotate3d on;

```



Slika 9

c) Polunadzirano učenje (engl. Semi-Supervised Learning)

Ovdje algoritam koristi kombinaciju označenih i neoznačenih podataka.

Praktično, u realnim sistemima često je teško označiti sve podatke, pa model uči i iz neoznačenih primjera.

- Cilj: Poboljšati učenje korištenjem velike količine neoznačenih podataka zajedno s malim brojem označenih.
- Primjene: Detekcija grešaka u proizvodnji, analiza potrošnje energije, predikcija kvarova u sistemima.
- Algoritmi: Semi-supervised SVM, Label Propagation, Self-training.

Primjer9 Zadatak je predvidjeti optimalno stanje ventilacije u plasteniku (otvorena/zatvorena), koristeći djelimično označene podatke prikupljene senzorima:

- Temperatura zraka (°C)
- Vlažnost zraka (%)
- Vlažnost tla (%)
- Intenzitet svjetla (lux)

Od ukupnog skupa podataka, samo dio (npr. 20%) je ručno označen kao:

- 1 = ventilacija treba biti uključena
- 0 = ventilacija treba biti isključena

Ostatak podataka nema oznaku, model treba sam naučiti obrasce i dopuniti oznake koristeći semi-supervised learning.

```
%% Polunadzirano učenje - ventilacija u plasteniku
clc; clear; close all;

%% Simulacija determinističkih podataka (24 sata)
n = 100;
time = linspace(0, 24, n)';

Temperature = 20 + 10*sin(time/24*2*pi); % °C
Humidity = 50 + 20*sin(time/24*2*pi + 1); % %
Light = 1000 + 800*sin(time/24*2*pi - 1); % lux

%% Definicija stvarnog stanja ventilacije
Ventilation = double(Temperature > 28 | Humidity > 70 | Light > 1800);

%% Polunadzirano učenje - svi podaci označeni
Labels = Ventilation;
X = [Temperature, Humidity, Light];

%% Treniranje decision tree modela (složen)
trainedModel = fitctree(X, Labels, 'MaxNumSplits', 50);

%% Predikcija ventilacije
Predicted = predict(trainedModel, X);

%% Dodajemo minimalno odstupanje za vizualizaciju
epsilon = 0.05; % minimalna razlika
Predicted_plus = min(Predicted + epsilon, 1);
Predicted_minus = max(Predicted - epsilon, 0);
```

```

%% Linijski graf
figure;

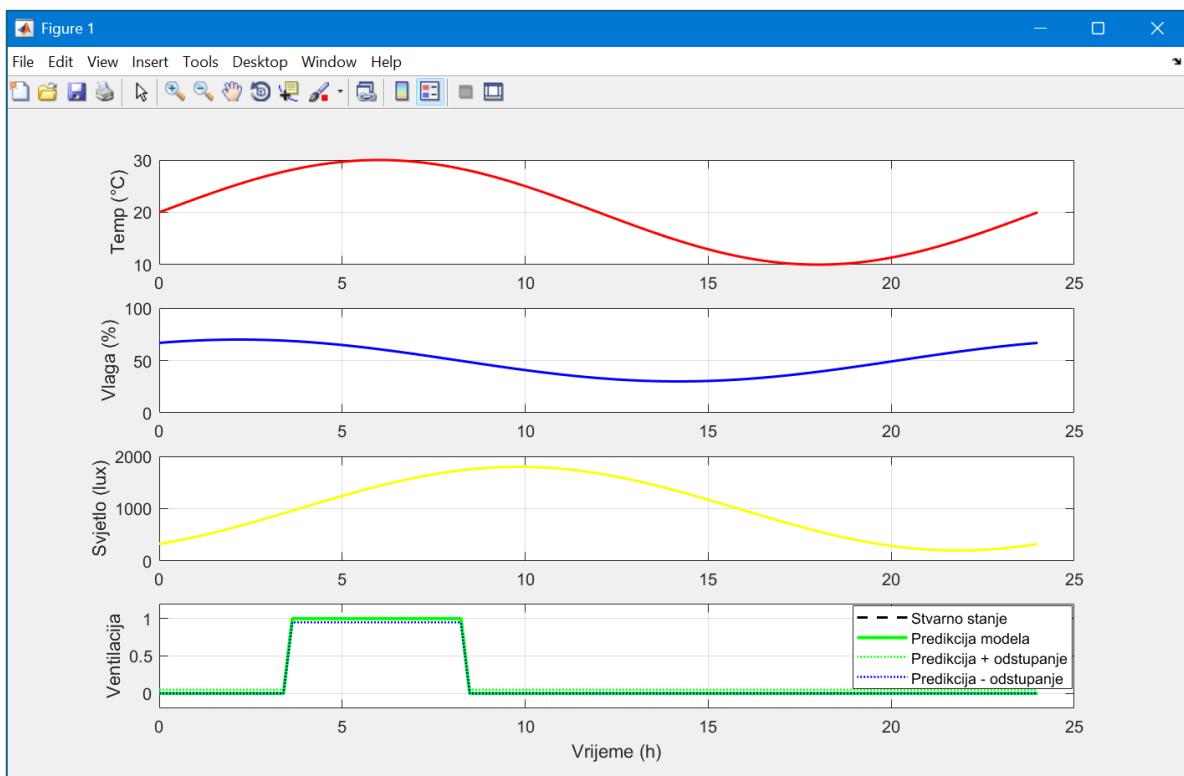
subplot(4,1,1)
plot(time, Temperature, '-r', 'LineWidth', 1.5); ylabel('Temp (°C)'); grid on;
subplot(4,1,2)
plot(time, Humidity, '-b', 'LineWidth', 1.5); ylabel('Vlaga (%)'); grid on;
subplot(4,1,3)
plot(time, Light, '-y', 'LineWidth', 1.5); ylabel('Svjetlo (lux)'); grid on;

subplot(4,1,4)
plot(time, Ventilation, '--k', 'LineWidth', 1.5, 'DisplayName', 'Stvarno stanje');
hold on;
plot(time, Predicted, '-g', 'LineWidth', 2, 'DisplayName', 'Predikcija modela');
plot(time, Predicted_plus, ':g', 'LineWidth', 1.2, 'DisplayName', 'Predikcija + odstupanje');
plot(time, Predicted_minus, ':b', 'LineWidth', 1.2, 'DisplayName', 'Predikcija - odstupanje');
xlabel('Vrijeme (h)'); ylabel('Ventilacija');
ylim([-0.2 1.2]);
legend('Location','best'); grid on;

% Naslov figure
axes('Position',[0 0 1 1],'Visible','off');
text(0.5, 1, 'Polunadzirano učenje - Minimalno odstupanje predikcije i stvarnog stanja', ...

'HorizontalAlignment','center','VerticalAlignment','top','FontSize',12,'FontW
eight','bold');

```



Slika 10

```

%% Graf sa bojama
figure; hold on;

% Stvarno stanje
area(time, Ventilation, 'FaceColor',[0.8 0.8 0.8], 'DisplayName','Stvarno
stanje');

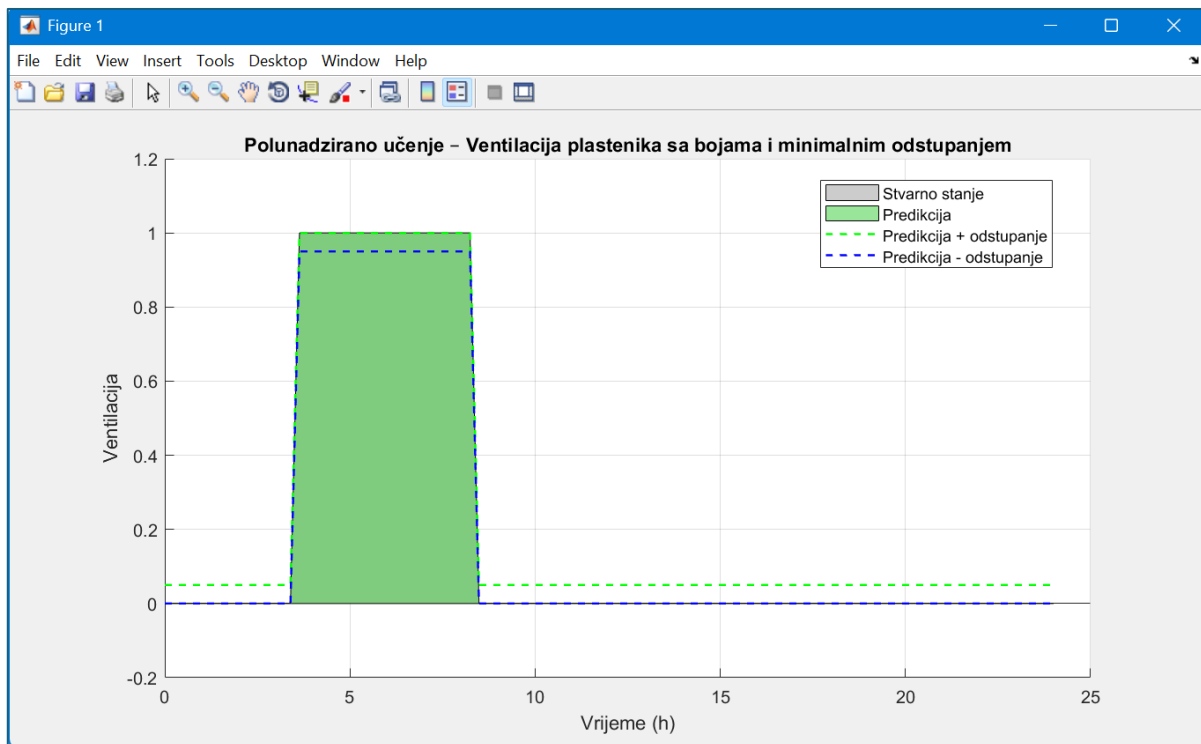
% Predikcija
area(time, Predicted, 'FaceColor',[0.2 0.8 0.2], 'FaceAlpha',0.5,
'DisplayName','Predikcija');

% Minimalno odstupanje
plot(time, Predicted_plus, '--g', 'LineWidth',1.2, 'DisplayName','Predikcija
+ odstupanje');
plot(time, Predicted_minus, '--b', 'LineWidth',1.2, 'DisplayName','Predikcija
- odstupanje');

xlabel('Vrijeme (h)');
ylabel('Ventilacija');
ylim([-0.2 1.2]);
grid on;
legend('Location','best');

% Naslov figure
title('Polunadzirano učenje - Ventilacija plastenika sa bojama i minimalnim
odstupanjem');

```



Slika 11