DATA SCIENCE 5B

Q-LEARNING

OLEH KELOMPOK - 7

AGNES ADELIA PUTRI (233307031)
CHANDRA DINA SEFRILLIAN (233307038)
ELDA SERLYA DWI SEVIANA (233307044)
KHOIRUL FAULAH NUR R (233307052)

www.qlearning.com

HIPOTESIS

jika sebuah agen ditempatkan dalam suatu lingkungan dan diberi kesempatan untuk mencoba berbagai aksi dengan mendapatkan reward (nilai) atau hukuman pada setiap langkah, maka dengan menggunakan algoritma Q-Learning agen tersebut akan belajar dari pengalamannya. Melalui proses mencoba berulang kali dan mengubah nilai Q di dalam Q-Table, agen diduga lama-kelamaan dapat menemukan pola tindakan yang menghasilkan reward total lebih besar, sehingga mampu mengambil keputusan yang semakin baik dari waktu ke waktu.

TEORI

Q-LEARNING

PENGERTIAN Q – LEARNING

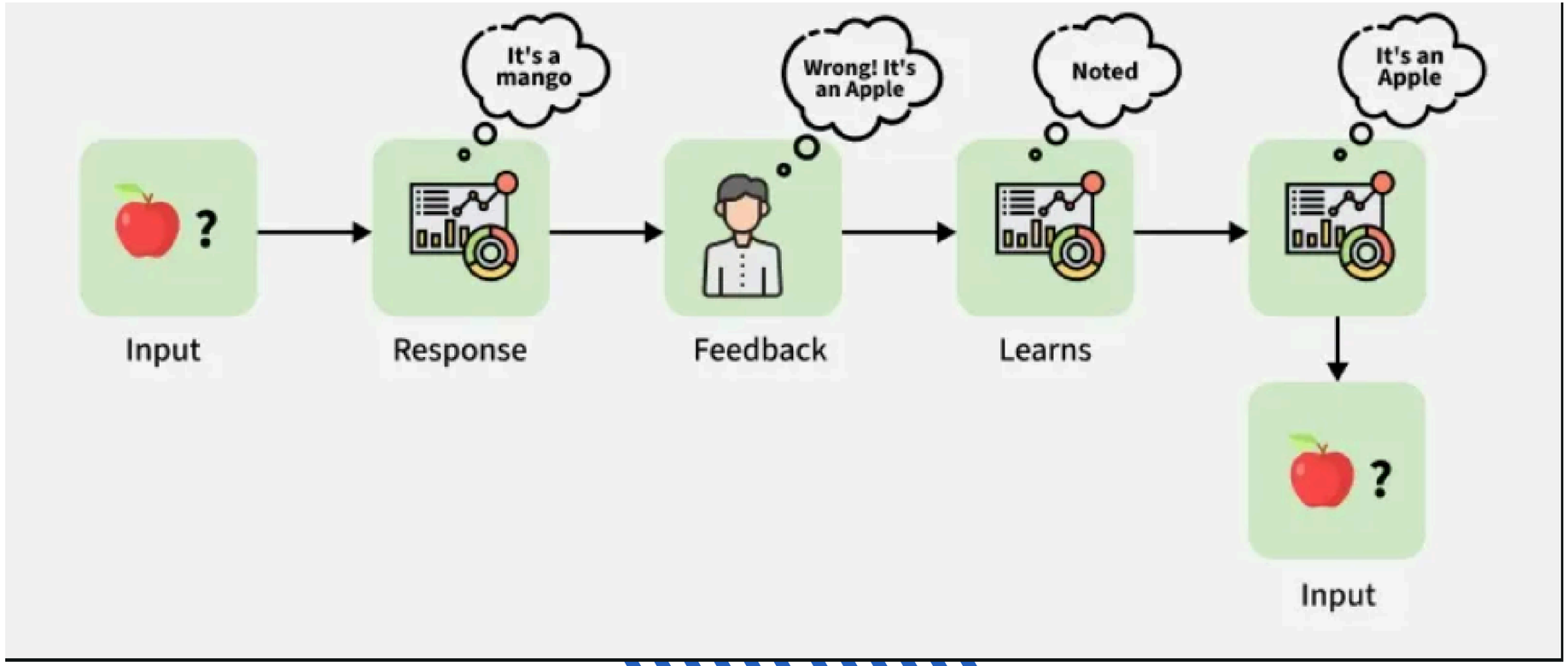
Q-Learning adalah algoritma Reinforcement Learning yang mengajarkan agen mengambil keputusan terbaik lewat proses trial-and-error. Berbeda dengan metode Machine Learning lain, Q-Learning tidak membutuhkan dataset apa pun. Algoritma ini tidak belajar dari data yang sudah tersedia, tetapi belajar langsung dari interaksi agen dengan lingkungan. ini menggunakan tabel nilai (Q-Table) untuk menyimpan nilai kualitas suatu tindakan (action) pada kondisi tertentu (state). Selama belajar, agen mencoba berbagai aksi, menerima reward atau hukuman, lalu memperbarui nilai Q menggunakan persamaan Bellman. Dengan cara ini, agen dapat menemukan strategi optimal tanpa perlu mengetahui aturan lingkungan terlebih dahulu.

TUJUAN Q-LEARNING

- menemukan tindakan (action) terbaik pada setiap kondisi (state)
- memaksimalkan total reward jangka panjang
- mempelajari nilai kualitas tindakan melalui pegalaman (trial-and-error)
- menghasilkan strategi optimal tanpa mengetahui model lingkungan
- membantu egen mengambil keputusan secara efisien dan konsisten

CONTOH SEDERHANA

Misal ada sebuah sistem yang melihat sebuah apel tetapi salah berkata, "Itu mangga." Sistem tersebut diberi tahu, "Salah! Itu apel." Sistem tersebut belajar dari kesalahan sebelumnya. Saat ditunjukkan apel berikutnya, sistem tersebut dengan benar berkata, "Itu apel." Proses coba-coba dengan umpan balik ini mirip cara kerja Q-Learning.



KOMPONEN UTAMA Q-LEARNING

1. State (S) – Kondisi yang Sedang Terjadi
State adalah gambaran kondisi atau situasi yang sedang berlangsung pada saat tertentu. State berisi informasi penting mengenai lingkungan yang menjadi dasar dalam menentukan langkah atau keputusan selanjutnya. Setiap proses pengambilan keputusan dalam Q-Learning selalu dimulai dari state yang sedang dihadapi.

2. Action (A) – Tindakan yang Dapat Dilakukan
Action adalah pilihan tindakan atau langkah yang dapat dilakukan pada suatu state tertentu. Tindakan ini menentukan bagaimana proses pembelajaran akan bergerak dari satu kondisi ke kondisi berikutnya. Pemilihan tindakan berpengaruh langsung pada perubahan state dan reward yang diperoleh.

KOMPONEN UTAMA Q-LEARNING

3. Reward (R) – Umpan Balik dari Lingkungan

Reward merupakan nilai yang diberikan oleh lingkungan sebagai respon dari tindakan tertentu. Reward berfungsi sebagai penanda apakah suatu tindakan memberikan dampak positif atau negatif. Nilai inilah yang digunakan untuk memperbarui Q-Value sehingga proses dapat berlangsung secara bertahap menuju hasil optimal.

4. Policy (π) – Aturan dalam Memilih Tindakan

Policy adalah aturan atau strategi yang digunakan untuk menentukan tindakan mana yang harus dipilih pada kondisi tertentu. Policy mengatur keseimbangan antara mencoba tindakan baru (exploration) dan menggunakan tindakan yang sudah diketahui memberikan hasil baik (exploitation).

KOMPONEN UTAMA Q-LEARNING

5. Q-Value – Nilai yang Menggambarkan Kualitas Tindakan

Q-Value adalah nilai yang menunjukkan seberapa baik atau menguntungkan suatu tindakan pada state tertentu dalam jangka panjang. Nilai ini diperbarui secara berkala dengan mempertimbangkan reward yang diterima serta prediksi nilai terbaik pada state berikutnya.

RUMUS Q-LEARNING

$$Q(s, a) = Q(s, a) + \alpha [r + \gamma \max Q(s', a') - Q(s, a)]$$

Keterangan:

- α (learning rate): seberapa cepat agen belajar
- γ (discount factor): seberapa peduli agen pada reward masa depan
- r: reward yang diterima
- s': state berikutnya

ALUR KERJA Q-LEARNING

Proses Q-Learning dimulai dengan menginisialisasi Q-Table, kemudian memilih tindakan melalui exploration atau exploitation. Setelah tindakan dilakukan, lingkungan memberikan reward dan proses berpindah ke state baru. Nilai Q diperbarui menggunakan rumus Q-Learning, dan langkah-langkah ini diulangi hingga nilai Q stabil.

HIPOTESA FUNCTION Q-LEARNING

Hipotesa function pada Q-Learning adalah fungsi yang memetakan setiap pasangan state dan action menjadi sebuah nilai yang disebut Q-value. Fungsi ini digunakan untuk memperkirakan seberapa baik suatu tindakan (action) ketika dilakukan pada kondisi tertentu (state). Hipotesa function ini dihitung menggunakan metode Action-Value Function (Q-Function) dan diperbarui dengan pendekatan Temporal Difference Learning (TD Learning), kemudian direpresentasikan dalam bentuk Q-Table.

COST FUNCTION Q-LEARNING

Dalam Q-Learning, cost function adalah ukuran selisih (error) antara:

- Q-value saat ini, dan
- Target Q-value, yaitu reward yang diterima diitambah nilai terbaik dari state berikutnya.

Cost function menggunakan TD Error (Temporal Differance Error) sebagai ukuran cost-nya. Tujuannya adalah meminimalkan error tersebut sehingga Q-value menjadi semakin akurat

$$\text{Cost} = \left(r + \gamma \max_{a'} Q(s', a') - Q(s, a)\right)$$

Keterangan :

- r = reward
- y (gamma) = discount factor
- Q(s,a) = nilai Q saat ini
- maxa Q’(s’,a’) = nilai Q terbaik dari state berikutnya

KERANGKA BERPIKIR

Q-LEARNING

KERANGKA BERPIKIR

1. INISIALISASI & PEMAHAMAN LINGKUNGAN

Agen ditempatkan dalam lingkungan tanpa pengetahuan awal (tabula rasa).
Detail Konsep
- Agen tidak memiliki dataset awal.
- Belajar murni dari interaksi langsung.
- Tujuan: Memaksimalkan reward jangka panjang.

2. IDENTIFIKASI KOMPONEN UTAMA

Mendefinisikan elemen-elemen yang membentuk dunia Q-Learning.
Detail Konsep
- State (S): Kondisi saat ini.
- Action (A): Pilihan tindakan.
- Reward (R): Umpan balik (positif/negatif).
- Policy (π): Strategi memilih aksi.

3. PEMBENTUKAN Q-TABLE

Membuat tabel memori untuk menyimpan nilai kualitas tindakan.
Detail Konsep
- Tabel berisi pemetaan State x Action.
- Awalnya diinisialisasi dengan 0 atau random.
- Q-Value merepresentasikan prediksi keuntungan masa depan.

4. EKSEKUSI AKSI (EXPLORATION VS EXPLOITATION)

Agen memilih tindakan berdasarkan Policy yang ditentukan.
Detail Konsep
- Exploration: Mencoba hal baru (ambil acak).
- Exploitation: Menggunakan pengetahuan terbaik saat ini (ambil max Q).
- Menyeimbangkan risiko dan peluang.

5. EVALUASI & PENERIMAAN REWARD

Lingkungan memberikan respon terhadap aksi agen.
Detail Konsep
- Agen menerima Reward (r) instan.
- Agen mengamati State baru (s').
- Menentukan apakah tindakan itu baik atau buruk.

6. PEMBARUAN PENGETAHUAN (BELLMAN UPDATE)

Mengupdate nilai Q-Table menggunakan rumus matematika.
Detail Konsep
- Q(s,a) baru = Q(s,a) lama + α [TD Error]
- Memperhitungkan Learning Rate (α).
- Memperhitungkan masa depan via Discount Factor (γ).

7. KONVERGENSI & OPTIMASI

Proses berulang hingga agen menemukan strategi optimal.
Detail Konsep
- Nilai Q menjadi stabil (konvergen).
- Error (selisih ekspektasi dan realita) diminimalkan.
- Agen otomatis tahu rute terbaik di setiap keadaan.

STUDI KASUS

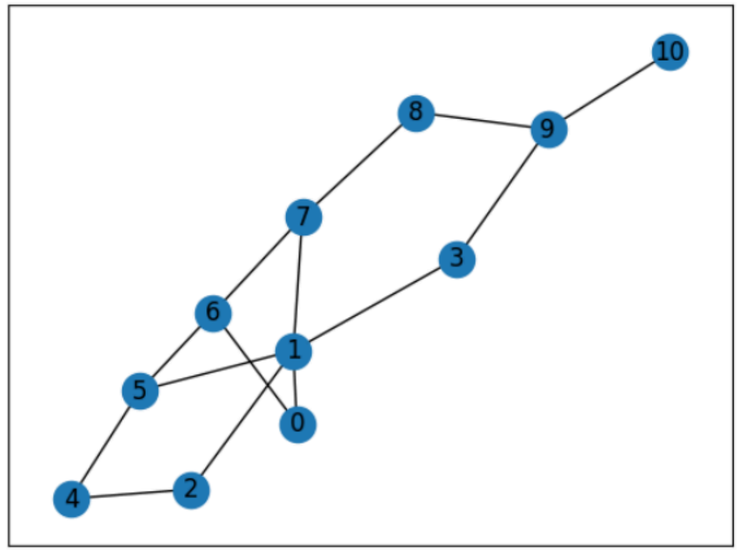
Q-LEARNING

CONTOH STUDI KASUS

seorang agen atau detektif belajar mencari rute terbaik di dalam sebuah jaringan titik (graf) untuk mencapai lokasi bandar narkoba yang direpresentasikan sebagai node 10. Agen menggunakan algoritma Q-Learning untuk mencoba berbagai jalur dan belajar dari reward yang diberikan, sehingga lama-kelamaan menemukan rute yang paling efisien. Selain itu, studi kasus ini juga memanfaatkan informasi lingkungan berupa posisi polisi dan jejak narkoba, sehingga agen tidak hanya fokus sampai ke bandar, tetapi juga berusaha memilih rute yang lebih aman dengan menghindari node yang berisiko.

KODE PROGRAM Q-LEARNING

```python
#Definisi environment (lingkungan)
edges = [(0, 1), (1, 5), (5, 6), (5, 4), (1, 2),
         (1, 3), (9, 10), (2, 4), (0, 6), (6, 7),
         (8, 9), (7, 8), (1, 7), (3, 9)]

goal = 10
G = nx.Graph()
G.add_edges_from(edges)
pos = nx.spring_layout(G)
nx.draw_networkx_nodes(G, pos)
nx.draw_networkx_edges(G, pos)
nx.draw_networkx_labels(G, pos)
pl.show()
```



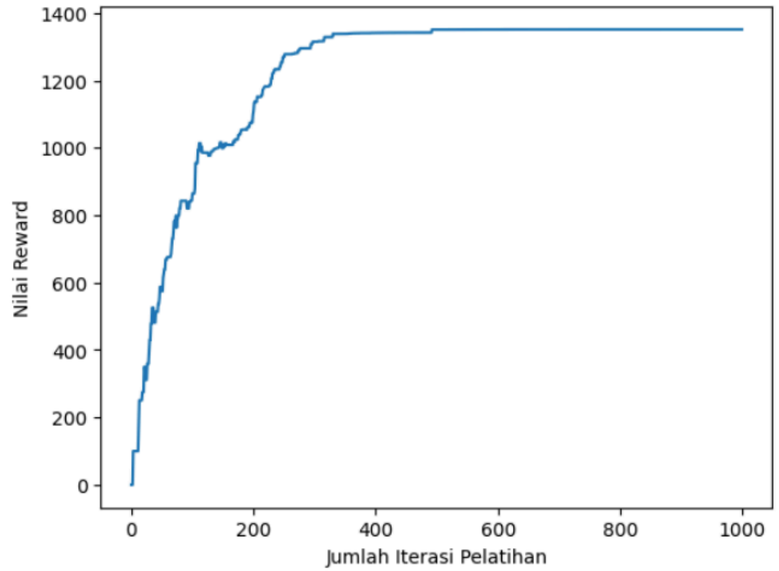

```python
# LANGKAH 2: Membuat matriks reward (M)
MATRIX_SIZE = 11
M = np.matrix(np.ones(shape=(MATRIX_SIZE, MATRIX_SIZE)))
M *= -1

for point in edges:
    if point[1] == goal:
        M[point] = 100
    else:
        M[point] = 0

    if point[0] == goal:
        M[point[::-1]] = 100
    else:
        M[point[::-1]] = 0

M[goal, goal] = 100

# --- tampilkan tabel reward ---
print("Tabel Reward (M):")
print(np.asarray(M, dtype=int))
```

```
Tabel Reward (M):
[[ -1   0  -1  -1  -1  -1   0  -1  -1  -1  -1]
 [  0  -1   0   0  -1   0  -1   0  -1  -1  -1]
 [ -1   0  -1  -1   0  -1  -1  -1  -1  -1  -1]
 [ -1   0  -1  -1  -1  -1  -1  -1  -1   0  -1]
 [ -1  -1   0  -1  -1   0  -1  -1  -1  -1  -1]
 [ -1   0  -1  -1   0  -1   0  -1  -1  -1  -1]
 [  0  -1  -1  -1  -1   0  -1   0  -1  -1  -1]
 [ -1   0  -1  -1  -1  -1   0  -1   0  -1  -1]
 [ -1  -1  -1  -1  -1  -1  -1   0  -1   0  -1]
 [ -1  -1  -1   0  -1  -1  -1  -1   0  -1 100]
 [ -1  -1  -1  -1  -1  -1  -1  -1  -1   0 100]]
```

KODE PROGRAM Q-LEARNING

```python
#Proses latih dan uji q learning
scores = []
for i in range(1000):
    current_state = np.random.randint(0, int(Q.shape[0]))
    available_action = available_actions(current_state)
    action = sample_next_action(available_action)
    score = update(current_state, action, gamma)
    scores.append(score)

# Testing
current_state = 0
steps = [current_state]

while current_state != 10:

    next_step_index = np.where(Q[current_state, ] == np.max(Q[current_state, ]))[1]
    if next_step_index.shape[0] > 1:
        next_step_index = int(np.random.choice(next_step_index, size = 1))
    else:
        next_step_index = int(next_step_index)
    steps.append(next_step_index)
    current_state = next_step_index

print("Rute paling efisien yang ditemukan:")
print(steps)

pl.plot(scores)
pl.xlabel('Jumlah Iterasi Pelatihan')
pl.ylabel('Nilai Reward')
pl.show()
```

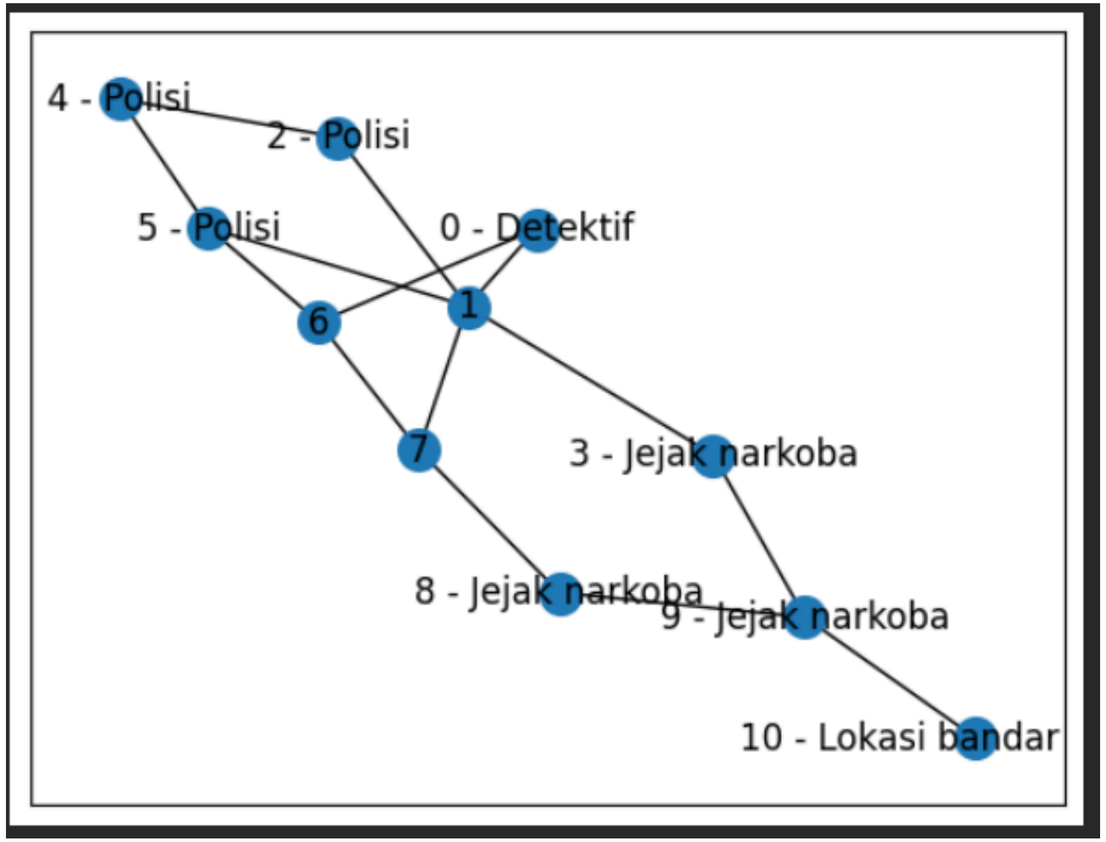
```
Rute paling efisien yang ditemukan:
[0, 1, 3, 9, 10]
```



Model menghasilkan rute paling efisien 0–1–3–9–10, yaitu jalur tercepat menuju lokasi bandar narkoba setelah proses pembelajaran Q-Learning. Grafik reward menunjukkan peningkatan tajam di awal, lalu stabil di akhir, menandakan jika agen sudah menemukan strategi optimal dan proses belajar sudah konvergen.

KODE PROGRAM Q-LEARNING

```python
# Menentukan lokasi polisi dan jejak narkoba
police = [2, 4, 5]
drug_traces = [3, 8, 9]

G = nx.Graph()
G.add_edges_from(edges)

mapping = {
    0: '0 - Detektif',
    1: '1',
    2: '2 - Polisi',
    3: '3 - Jejak narkoba',
    4: '4 - Polisi',
    5: '5 - Polisi',
    6: '6',
    7: '7',
    8: '8 - Jejak narkoba',
    9: '9 - Jejak narkoba',
    10: '10 - Lokasi bandar narkoba'
}

H = nx.relabel_nodes(G, mapping)
pos = nx.spring_layout(H)
nx.draw_networkx_nodes(H, pos, node_size=200)
nx.draw_networkx_edges(H, pos)
nx.draw_networkx_labels(H, pos)
pl.show()
```



KODE PROGRAM Q-LEARNING

```python
# Melatih agen dan mencatat posisi polisi serta jejak narkoba
scores = []
for i in range(1000):
    current_state = np.random.randint(0, int(Q.shape[0]))
    available_action = available_actions(current_state)
    action = sample_next_action(available_action)
    score = update(current_state, action, gamma)

#Menampilkan Q tabel
print("\nTabel Q setelah training:")
Q_rounded = np.round(np.asarray(Q, dtype=float), 2)
print(Q_rounded)

print('Lokasi Polisi yang Terdeteksi')
print(env_police)
print('')
print('Lokasi Jejak Narkoba yang Terdeteksi')
print(env_drugs)
```

```
Tabel Q setelah training:
[[-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [-1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]]
Lokasi Polisi yang Terdeteksi
[[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]

Lokasi Jejak Narkoba yang Terdeteksi
[[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

KESIMPULAN

Dapat diambil kesimpulan bahwa Q-Learning adalah metode reinforcement learning berbasis tabel (Q-Table) yang memungkinkan agen belajar mengambil keputusan optimal melalui proses trial-and-error . Q-Learning bekerja dengan merepresentasikan masalah dalam bentuk state, action, reward, policy, dan Q-value, lalu secara berulang memperbarui nilai Q menggunakan persamaan Bellman dan Temporal Difference Learning sampai nilai Q stabil. Pada studi kasus pencarian rute menuju node tujuan dan perluasan kasus dengan node polisi serta jejak narkoba, terlihat bahwa agen mampu belajar memilih jalur yang paling menguntungkan sekaligus menghindari jalur berisiko berdasarkan reward dan bantuan informasi lingkungan. Hal ini menunjukkan bahwa Q-Learning efektif digunakan untuk masalah pengambilan keputusan bertahap, seperti penentuan rute, perencanaan langkah, maupun sistem cerdas lain yang membutuhkan strategi optimal berbasis pengalaman.

TERIMA KASIH