AI_p10.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

21162101012_CBA_Yash_Lakhtariya

AI Practical 10

AIM : Implement Viterbi HMM Algorithm and find the sequence based on observation :

Consider a small town where people are either:

- healthy
- have fever

Only the doctor can differentiate or identify the people having fever from healthy people. He does so by inquiring about their symptoms. People can have one of the following answers:

- normal
- dizzy
- cold

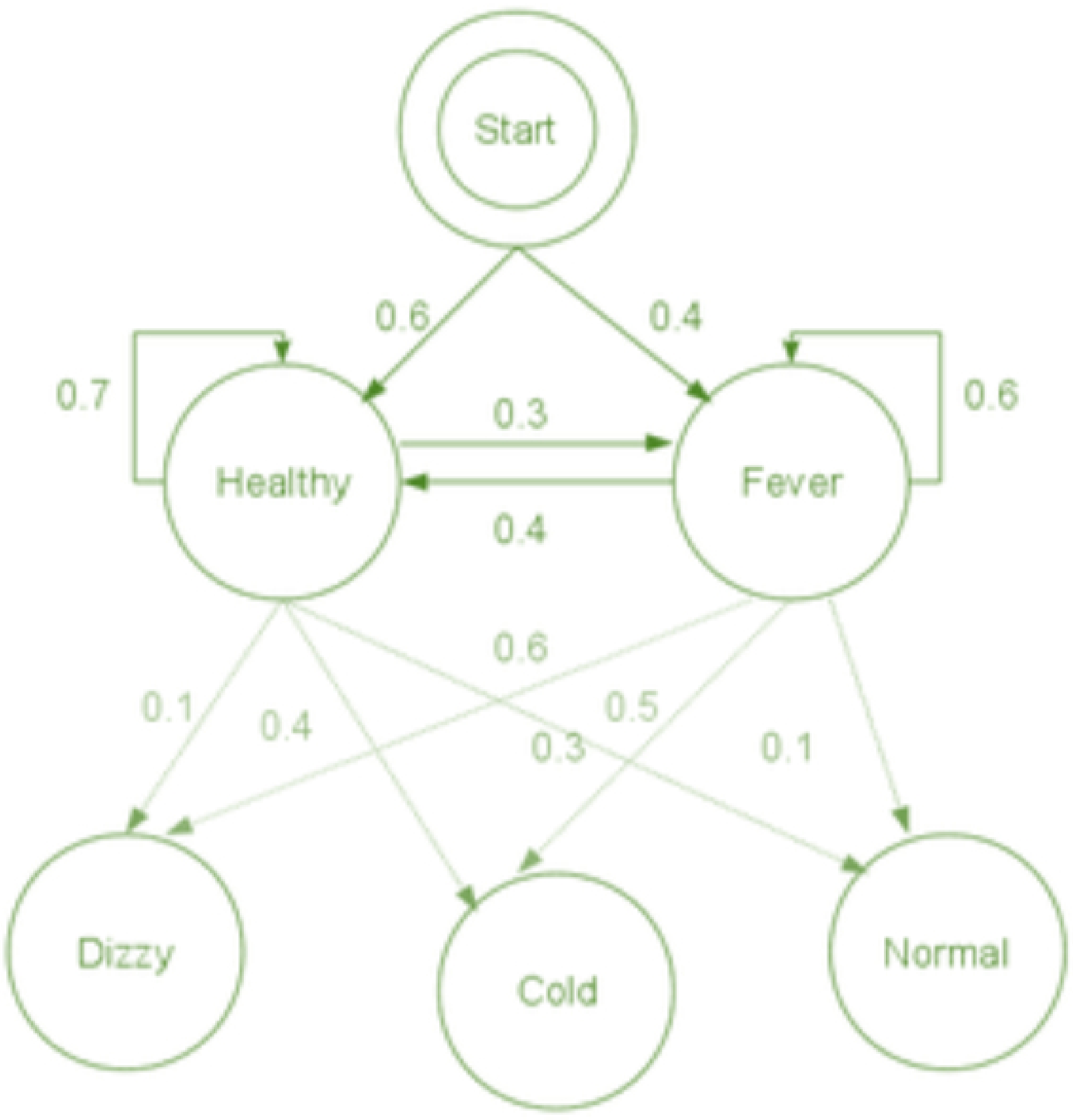
The doctor believes that the health condition of his patients operates as a discrete Markov chain. The two states are:

- healthy
- fever

However, these states are 'hidden' from the doctor. The observations can be :

- normal
- dizzy
- cold

Now, consider a situation where a patient visits three days consecutively. All three days, he shows different symptoms (normal/cold/dizzy). The doctor's question here will be : **What is the most likely sequence of health conditions of the patient that would explain these observations?**



```
[15] s = ('Healthy', 'Fever')
     start_p = {'Healthy': 0.6, 'Fever': 0.4}
     trans_p = {'Healthy': {'Healthy': 0.7, 'Fever': 0.3}, 'Fever': {'Healthy': 0.4, 'Fever': 0.6}}
     emit_p = {'Healthy': {'dizzy': 0.1, 'cold': 0.4, 'normal': 0.3}, 'Fever': {'dizzy': 0.6, 'cold': 0.5, 'normal': 0.1}}
```

```
[16] def viterbi(obs, states, start_prob, trans_prob, emit_prob):
         V = [{}]
         path = {}

         for state in states:
             V[0][state] = start_prob[state] * emit_prob[state][obs[0]]
             path[state] = [state]

         for t in range(1, len(obs)):
             V.append({})
             new_path = {}

             for curr_state in states:
                 max_prob, prev_state = max((V[t - 1][prev_state] * trans_prob[prev_state][curr_state] * emit_prob[curr_state][obs[t]], prev_state) for prev_state in states)
                 V[t][curr_state] = max_prob
                 new_path[curr_state] = path[prev_state] + [curr_state]
             path = new_path

         max_prob, final_state = max((V[-1][state], state) for state in states)
         return max_prob, path[final_state]
```

```
[17] user_input = input("Enter the observed conditions separated by spaces (e.g., dizzy cold normal): ")
     obs = tuple(user_input.strip().split())
```

```
     Enter the observed conditions separated by spaces (e.g., dizzy cold normal): normal cold dizzy
```

```
[20] if not all(o in emit_p['Healthy'] for o in obs):
         print("\n\tInvalid input. Please enter valid conditions: normal, cold, dizzy.")
     else:
         prob, paths = viterbi(obs, s, start_p, trans_p, emit_p)
         print(f"\n\tMost likely path of health conditions : ", end='')
         for i,y in enumerate(paths):
           if i == len(paths)-1:
             print(f' {y}', end='')

           else:
             print(f' {y}', end=' ➡️')
```

```
            Most likely path of health conditions :  Healthy ➡️ Fever ➡️ Fever
```

Colab paid products - Cancel contracts here

✓ 0s completed at 7:55 PM