|  |  |
| --- | --- |
| **Course Code: AI-2002** | **Course: Artificial Intelligence Lab** |
| **Instructor(s):** | **Sania Urooj** |

**National University of Computer & Emerging Sciences, Karachi**

**Computer Science Department**

**Summer – 2023 Lab 09**

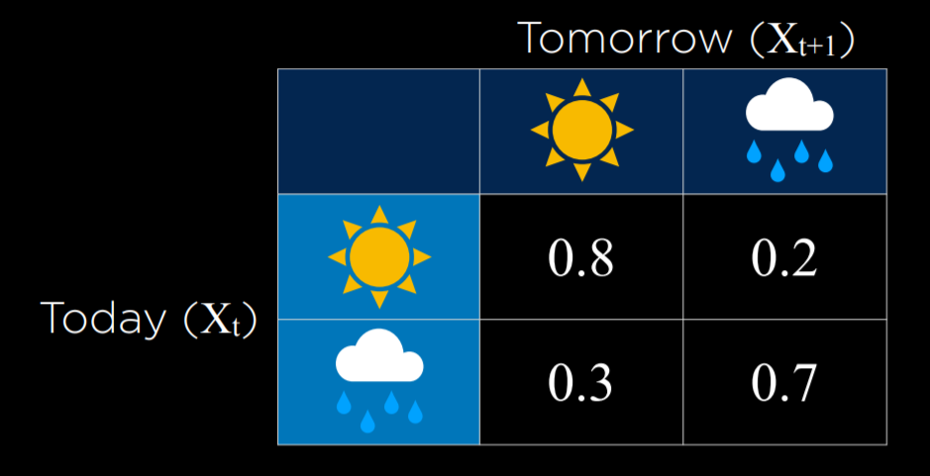
**Markov Model**

Markov models are a type of probabilistic model that is used to predict the future state of a system, based on its current state. In other words, Markov models are used to predict the future state based on the current hidden or observed states. Markov model is a finite-state machine where each state has an associated probability of being in any other state after one step. They can be used to model real-world problems where hidden and observable states are involved. Markov models can be classified into hidden and observable based on the type of information available to use for making predictions or decisions.

To better understand Markov models, let’s look at an example. Say you have a bag of marbles that contains four marbles: two red marbles and two blue marbles. You randomly select a marble from the bag, note its color, and then put it back in the bag. After repeating this process several times, you begin to notice a pattern: The probability of selecting a red marble is always two out of four, or 50%. This is because the probability of selecting a particular color of marble is determined by the number of that color of marble in the bag. In other words, the past history (i.e., the contents of the bag) determines the future state (i.e., the probability of selecting a particular color of marble).

**Markov Chain**

A Markov chain is a sequence of random variables where the distribution of each variable follows the Markov assumption. That is, each event in the chain occurs based on the probability of the event before it.



***Generate markov chains of the above example***

import numpy as np

# Define the weather states and transition probabilities

states = ["sunny", "rainy"]

transition\_matrix = np.array([

[0.8, 0.2],

[0.3, 0.7]

])

# Function to predict the next day's weather based on the current weather

def predict\_weather(current\_weather, num\_days):

current\_state\_index = states.index(current\_weather)

weather\_sequence = [current\_weather]

for \_ in range(num\_days):

# Calculate the next day's weather based on the current state

next\_state\_index = np.random.choice(len(states), p=transition\_matrix[current\_state\_index])

next\_weather = states[next\_state\_index]

# Update the current state for the next iteration

current\_state\_index = next\_state\_index

weather\_sequence.append(next\_weather)

return weather\_sequence

# Perform weather prediction for 5 days starting from a sunny day

predicted\_weather = predict\_weather(current\_weather="sunny", num\_days=5)

print(predicted\_weather)

**Hidden Markov Model**

Hidden Markov models (HMMs) are a type of statistical modeling that has been used for several years. They have been applied in different fields such as medicine, computer science, and data science. The Hidden Markov model (HMM) is the foundation of many modern-day data science algorithms. It has been used in [data](https://vitalflux.com/category/data-science) [science](https://vitalflux.com/category/data-science) to make efficient use of observations for successful predictions or decision- making processes.

The [hidden Markov model (HMM)](https://en.wikipedia.org/wiki/Hidden_Markov_model) is another type of Markov model where there are few states which are hidden. This is where HMM differs from a Markov chain. HMM is a statistical model in which the system being modeled are Markov processes with unobserved or hidden states. It is a hidden variable model which can give an observation of another hidden state with the help of the Markov assumption. The hidden state is the term given to the next possible variable which cannot be directly observed but can be inferred by observing one or more states according to Markov’s assumption. Markov assumption is the assumption that a hidden variable is dependent only on the previous hidden state. Mathematically, the probability of being in a state at a time t depends only on the state at the time (t-1). It is termed a limited horizon assumption. Another Markov assumption states that the conditional distribution over the next state, given the current state, doesn’t change over time. This is also termed a stationary process assumption.

A Markov model is made up of two components: the state transition and hidden random variables that are conditioned on each other. A hidden Markov model consists of four important components:

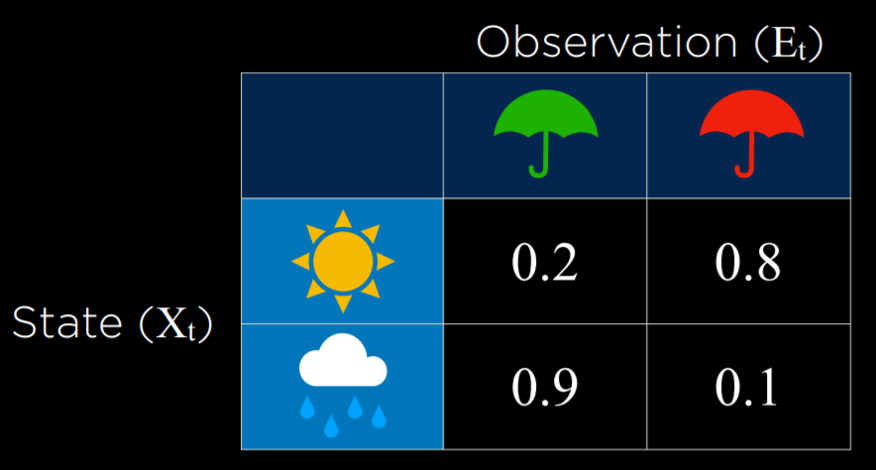
**Key components of Hidden Markov Models:**

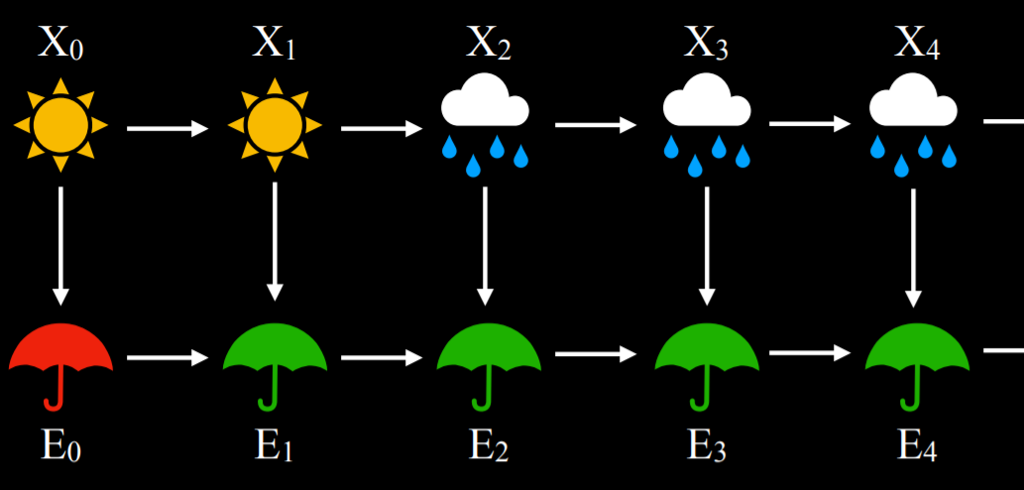
**Hidden States**: These are the unobservable or latent states of the system. Each hidden state represents a different underlying condition or process.

**Observations:** These are the observable emissions or measurements that are generated based on the hidden states. Observations are used to infer the hidden states.

**State Transition Probabilities:** These represent the probabilities of transitioning from one hidden state to another over time. They follow the Markov property, meaning that the future state depends only on the current state.

**Emission Probabilities**: These represent the probabilities of emitting specific observations given the underlying hidden state. Each hidden state can emit different observations with varying probabilities





import numpy as np

from hmmlearn import hmm

# Define observation model for each state

sun = np.array([[0.2, 0.8]])

rain = np.array([[0.9, 0.1]])

# Define transition model

transitions = np.array([

[0.8, 0.2], # Tomorrow's predictions if today = sun

[0.3, 0.7] # Tomorrow's predictions if today = rain

])

# Define starting probabilities

starts = np.array([0.5, 0.5])

# Create the model

model = hmm.CategoricalHMM(n\_components=2, verbose=False)

model.startprob\_ = starts

model.transmat\_ = transitions

model.emissionprob\_ = np.vstack((sun, rain))

# Observed data

observations = np.array([

[0], # umbrella

[0], # umbrella

[1], # no umbrella

[0], # umbrella

[0], # umbrella

[0], # umbrella

[0], # umbrella

[1], # no umbrella

[1] # no umbrella

])

# Predict underlying states

predictions = model.predict(observations)

for prediction in predictions:

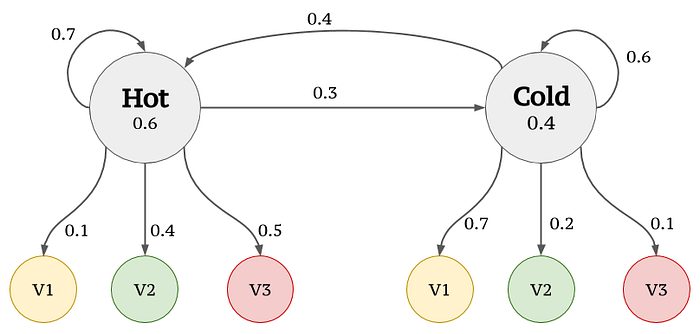
if prediction == 0:

print("sun")

else:

print("rain")

The hidden Markov model in the above diagram represents the process of predicting whether someone will be found to be walking, shopping, or cleaning on a particular day depending upon whether the day is rainy or sunny. The following represents five components of the hidden Markov model in the above diagram:



from hmmlearn import hmm

import numpy as np

# Define the model

model = hmm.CategoricalHMM(n\_components=2, verbose=False)

# Define the transition probabilities

model.transmat\_ = np.array([[0.7, 0.3],

[0.6, 0.4]])

# Define the emission probabilities

model.emissionprob\_ = np.array([[0.1, 0.4,0.5],

[0.7,0.2, 0.1]])

# Define the starting probabilities

model.startprob\_ = np.array([0.6, 0.4])

# Define the observed sequence

observed\_sequence = np.array([0, 1, 0, 0, 1])

# Fit the model to the observed sequence

model.fit(observed\_sequence.reshape(-1, 1))

# Print the model parameters

print("Transition probabilities:")

print(model.transmat\_)

print("\nEmission probabilities:")

print(model.emissionprob\_)

print("\nStarting probabilities:")

print(model.startprob\_)

# Infer the hidden state sequence of the observed sequence

hidden\_states = model.predict(observed\_sequence.reshape(-1, 1))

print("\nHidden state sequence:")

print(hidden\_states)