HMM

What is a Markov model

A Markov Model or a Markov Chain is a stochastic model that represents a system where the probability of transitioning from one state to another depends only on the current state and not on the sequence of events that preceded it. This property is often referred to as the Markov property or Markovian property.

In a Markov Model, the system is assumed to exhibit the Markov property, meaning the future behaviour of the system depends only on its current state, and not on how it arrived at that state. This makes Markov Models particularly useful for modelling processes where the past history is not relevant, and only the current state matters.

There are different types of Markov Models:

1. **First-order Markov Model**: In this model, the probability of transitioning to a certain state depends only on the current state. Mathematically, this can be represented as $P(X_{n+1}=x|X_n=y, X_{n-1}, \ldots, X_0) = P(X_{n+1}=x|X_n=y)$, where X_n represents the state at time n, and x and y represent specific states.
2. **Higher-order Markov Model**: In higher-order Markov Models, the probability of transitioning to a certain state depends on the current state as well as a fixed number of preceding states. For example, in a second-order Markov Model, the probability of transitioning to a certain state depends on the current state and the previous state.
3. **Hidden Markov Model (HMM)**: A special case of a Markov Model where the states are not directly observable. Instead, the system emits observable symbols or observations, and the goal is to infer the sequence of hidden states based on the observed data. HMMs are widely used in various fields such as speech recognition, natural language processing, and bioinformatics.

Markov Models find applications in a wide range of fields, including finance, biology, telecommunications, and more. They are particularly useful for modelling systems with stochastic behaviour and are employed in tasks such as predicting stock prices, analysing DNA sequences, modelling text data, and more.

First-order Markov model

A first-order Markov model, also known as a first-order Markov chain, is a stochastic model that describes a sequence of events where the probability of transitioning to a certain state depends only on the current state. It's a fundamental concept in probability theory and has applications in various fields including computer science, economics, and biology.

Let's break down the key components of a first-order Markov model:

1. **States**: A set of possible conditions or situations that the system can be in at any given time. These states are often denoted as $S = \{S_1, S_2, \ldots, S_n\}$.

2. **Transition Probabilities**: For each pair of states S_i and S_j, there is a probability P_{ij} of transitioning from state S_i to state S_j. These transition probabilities are typically represented as a transition matrix.
3. **Initial State Probability Distribution**: The probabilities of starting in each state. This is usually represented as a probability vector denoted by $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$, where π_i represents the probability of starting in state S_i.
4. **Memoryless Property**: In a first-order Markov model, the probability of transitioning to a certain state at time $t+1$ depends only on the state at time t, and not on any earlier states. This property is also known as the Markov property.
5. **Time Homogeneity**: The transition probabilities do not change over time. This means that the probability of transitioning from one state to another remains constant regardless of the time step.

First-order Markov models are often represented using state transition diagrams or transition matrices. They are used in various applications such as:

- Natural language processing: Modelling text data, part-of-speech tagging.
- Economics: Modelling market behaviour, predicting financial time series.
- Biology: Modelling DNA sequences, protein structure prediction.
- Engineering: Modelling system reliability, queueing systems.

Overall, first-order Markov models provide a simple yet powerful framework for modeling sequential data and understanding probabilistic dependencies between events.

hidden Markov model

Hidden Markov Models (HMMs) are statistical models used for modelling sequential data, where you have a sequence of observable events but there are hidden states underlying these observations. It's a type of probabilistic graphical model.

In an HMM, you have two main components:

1. **Hidden States**: These are the underlying states of the system that you cannot observe directly. Each hidden state emits observable symbols with certain probabilities.
2. **Observations**: These are the events or symbols that you can observe. Each observation is emitted by one of the hidden states.

The key assumptions of an HMM are:

- Markov property: The probability of transitioning from one state to another depends only on the current state.
- Stationarity: The transition probabilities between hidden states do not change over time.
- Output independence: The probability of emitting an observation depends only on the current hidden state and is independent of previous observations.

HMMs are used in various applications such as speech recognition, natural language processing, bioinformatics, and more, where the underlying structure of the data is assumed to have hidden states influencing the observed data. They're particularly useful when the underlying system is not directly observable, but we can observe some outcomes that depend on it.

Training an HMM typically involves two main tasks:

1. **Learning**: Estimating the parameters of the model (transition probabilities between states and emission probabilities of observations) from the observed data, often using algorithms like the Baum-Welch algorithm (a variant of the Expectation-Maximization algorithm).
2. **Inference**: Given a sequence of observations, determining the most likely sequence of hidden states that generated them, usually accomplished using the Viterbi algorithm.

HMMs have been instrumental in various fields due to their ability to model sequential data with hidden structures. They are powerful tools for tasks involving pattern recognition, time series analysis, and understanding dynamic systems.

An example

Let's consider a simple example of a weather system represented by a Hidden Markov Model (HMM). In this example, we have two hidden states representing the weather conditions: "Sunny" and "Rainy". These hidden states are not directly observable. Instead, we can only observe whether people are carrying umbrellas or not.

Here's the breakdown:

1. **Hidden States**:
    - State 1: Sunny
    - State 2: Rainy
2. **Observations**:
    - Observation 1: No umbrella
    - Observation 2: Umbrella

Now, let's define the parameters of our HMM:

- **Initial probabilities**: The probability of starting in each hidden state.
- Probability of starting in Sunny: 0.6
- Probability of starting in Rainy: 0.4
- **Transition probabilities**: The probability of transitioning from one hidden state to another.
- Transition from Sunny to Sunny: 0.7
- Transition from Sunny to Rainy: 0.3
- Transition from Rainy to Rainy: 0.6
- Transition from Rainy to Sunny: 0.4
- **Emission probabilities**: The probability of observing a certain symbol (umbrella or no umbrella) given the hidden state.

- Probability of No umbrella given Sunny: 0.2
- Probability of Umbrella given Sunny: 0.8
- Probability of No umbrella given Rainy: 0.6
- Probability of Umbrella given Rainy: 0.4

Now, let's say we observe a sequence of events over a few days:

Day 1: No umbrella Day 2: Umbrella Day 3: Umbrella Day 4: No umbrella

Given this sequence of observations, we want to infer the most likely sequence of hidden states (weather conditions) that generated these observations. We can use the Viterbi algorithm for this purpose.

By applying the Viterbi algorithm, we can find the most likely sequence of hidden states:

- Day 1: Sunny
- Day 2: Rainy
- Day 3: Rainy
- Day 4: Sunny

So, according to the model, the most likely weather conditions on these days were: Sunny, Rainy, Rainy, Sunny.

This is just a simple example to illustrate how an HMM works. In real-world applications, HMMs can be much more complex and are used for tasks such as speech recognition, part-of-speech tagging, bioinformatics and more.