**Hidden Markov Models**

Viterbi Algorithm

[[Ellie Arbab](https://medium.com/@ellie.arbab?source=post_page-----580bdc4844fa--------------------------------)](https://medium.com/@ellie.arbab?source=post_page-----580bdc4844fa--------------------------------)

[Ellie Arbab](https://medium.com/@ellie.arbab?source=post_page-----580bdc4844fa--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F32c26bf626d0&operation=register&redirect=https%3A%2F%2Fmedium.com%2F%40ellie.arbab%2Fhidden-markov-models-580bdc4844fa&user=Ellie+Arbab&userId=32c26bf626d0&source=post_page-32c26bf626d0----580bdc4844fa---------------------post_header-----------)

6 min read

·

Jun 1

11

Markov Processes and Hidden Markov Models (HMM) are almost always part of the conversation in sequence models. While extremely intuitive, they offer a powerful inference framework. In this post, after introducing the Markov Process and Hidden Markov Models (HMM), we go over a use case example in Sports Betting. Then we describe the Viterbi Algorithm, a dynamic programming approach to finding the optimum path over a multi-thread of potential sequence values.

**Markov Process and HMM**

A Markov Process is one in which the probability distribution for every state only depends on the previous state in the system. In other words, every state includes all the required information from previous states in order to determine the likelihood of the next state.

Hidden Markov Models are an extension to Markov Processes in which our observations are directly a result of a Markov Process but we don’t have visibility to it, hence the name.

In other words, a HMM have the following two properties:

1. It consist of *Unobservable (****X****)*Markov process and *Observable* (***Y***) process. The unobservable Markov process is defined by the *Transition* function.



1. The observables, ***Y,*** is known to be influenced only by the hidden Markov state, ***X***. This dependency is described by the *Emission* function:



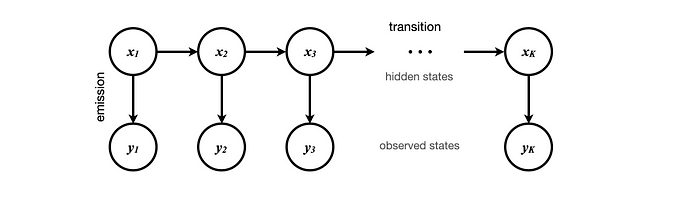


Figure 1: General Architecture of a Hidden Markov Model.

**Example: Sports Betting**

We know that a horse performance is highly correlated to its physical fitness: *Healthy* or *Recovering*. This information is kept confidential, unobservable ***X***. Based on the horse physical state, the manager decides which one of the three Jockeys will ride the race, observable ***Y***.

While a sports bettor cannot know the health status of a horse, the Jockey is announced publicly. Knowing the most likely health state of a horse highly impacts the likelihood of winning and hence the bettors’ action. Can a sport bettor decide the most likely physical fitness history of a hours only through a series of Jockey choices?

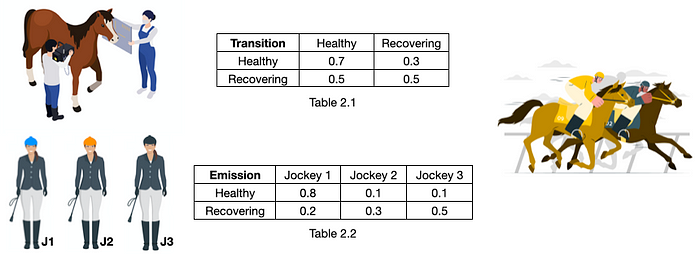


Figure 2: Hidden Markov Model Example, [Images](https://www.freepik.com/free-vector/farm-animals-veterinary-compositions_26762016.htm#query=horse%20veterinary&position=14&from_view=author) by [macrovector](https://www.freepik.com/free-vector/set-flat-color-icons-with-white-background-depicting-jockey-equipment-horse_3797298.htm#query=horses&position=17&from_view=author) and [storyset](https://www.freepik.com/free-vector/horse-riding-competition-concept-illustration_26388719.htm#query=jockey&position=38&from_view=search&track=sph) on [freepik](https://www.freepik.com/).

Imagine Table 2.1 and Table 2.2 of Figure (2) are given. Table 2.1 tells us the likelihood of the horse fitness status transitioning between Healthy and Recovery in two consecutive races. For example, if the horse enters a race healthy, there’s 70% chance that it retains its health for the next race and 30% chance that it will go into recovery status. On the other hand, if the horse enters a race while in recovery state, it is equally likely to be in either recovery or healthy state after that race.

Table 2.2 describes the relationship between fitness status and choice of Jockey. It tells us that if the horse is healthy, there is 80% chance that Jockey 1 (J1) will ride the race and only 10% chance that Jockey 2 (J2) or Jockey 3 (J3) will. If the horse is recovering, then Jockey 1, Jockey 2 and Jockey 3 will ride the race with 20%, 30% and 50% probability, respectively.

To accommodate the case where a horse is running its first race of the season, we include a start state ***S***and estimate the transitions probabilities from it to the hidden states. Figure (3) has the full Hidden Markov Model.

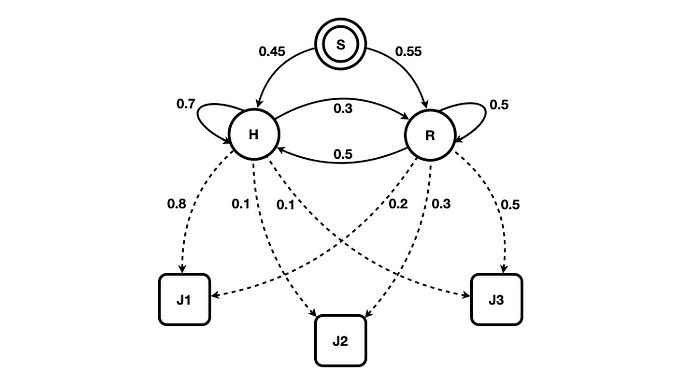
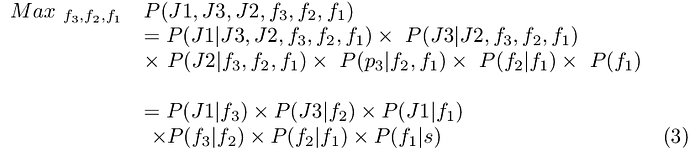


Figure 3: Hidden Markov Model of Sports Betting example.

Note the solid arrows indicate transition and dashed arrows the emission probabilities.

Let’s imagine that Jockey 2 rode the first race of season, Jockey 3 rode the second one and Jockey 1 rode the third race. Our sports bettor then can calculate the most likely sequence for the horse fitness, *f*, as follows:



Without loss of generality, observable and unobservable variables in Equation (3) are arranged to accommodate the most straight forward algebraic expansion.

In small samples, one might solve Equation (3) by brute force over the possible values. It is not difficult to see that in this specific example, *Recovering, Healthy, Healthy*is the sequence of hidden states that maximizes the probability of observing J2, J3, J1.

What if the brute force optimization of Equation (3) was infeasible? With longer sequence of observations or larger ***X*** and ***Y***sets, brute force quickly becomes intractable.

**Viterbi Algorithm**

Viterbi Algorithm uses dynamic programming principles to compute a posteriori probabilities of hidden states given a sequence of observables and also identifies the path that leads to such maximum probabilities, aka *Viterbi Path*.

The core idea is to understand how only the path with highest probability thus far at a node, should be pursued as potential solution among all other paths that share the same node.

Let’s demonstrate this by calculating the optimized values for the Sports Betting example.

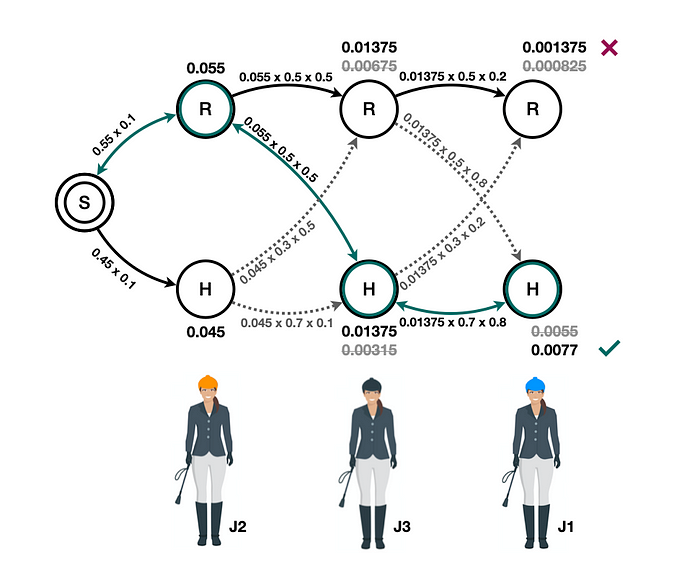


Figure 4: Viterbi Algorithm of Sport Betting Example.

Figure (4) demonstrates the steps of Viterbi Algorithm in discovering the most likely sequence of hidden states given J2, J3, J1 observation sequence.

Starting off, it computes the respective probabilities of arriving to either Healthy or Recovery state from start given J2 rode the first race, *0.45*x *0.1*and *0.55*x*0.1* respectively. The probability of being in second Healthy state given J3 rode the second race then becomes either (*0.055* x *0.5* x *0.5*) or (*0.045* x *0.7* x *0.1*) depending on whether the first state was Recovery or Healthy, respectively. Since arriving at Healthy second state from a Healthy first state (*0.00315*) is less likely than Recovery first state (*0.01375*), we don’t pursue this path any further.

Following a similar logic, we arrive at the maximum probability at the third and final state and identify the path that leads to it as the most likely sequence of hidden states.

function VITERBI(H: hidden\_states, Y: observed\_states, T: transition\_matrix,  
 E:emission\_matrix) -> X: hidden\_state\_index  
  
 for h in H:   
 x[h, 0] = 0  
 m[h, 0] = T[0,h] x E[h, Y[0]]  
  
 for y in Y:  
 for h in H:  
 for j in H:  
 temp = m[j, y-1] x T[j, h] x E[h, y]  
  
 m[h, y] = max(temp)  
 x[h, y] = argmax(temp)  
   
 X[len(Y)-1] = argmax(m[:,Y[len(Y)-1]])  
 for i in range(len(Y)-1, 1, -1):  
 X[i] = x[X[i+1], Y[i]]  
  
 return X

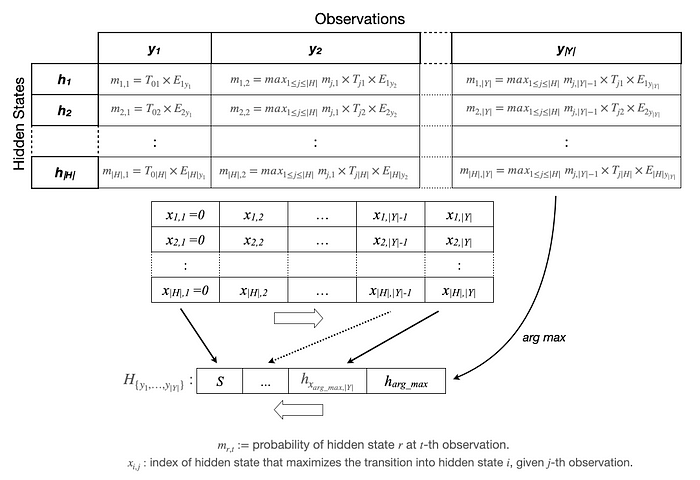


Figure 5: Viterbi Algorithm walkthrough.

Recap  
***Markov Processes*** and ***HMM***s lend themselves to model many sequence events. They are intuitive and powerful and enable making inferences based on sequence of observations.

HMM’s usually deal with identifying the most likely hidden states of the system, which in itself involves optimization to execute. ***Viterbi Algorithm*** incorporates the observation that the most likely path is the one that generates the highest probability among all paths that share a node.

An understanding of ***Transition*** and ***Emission*** probabilities is key in creating HMM models. These probabilities, as per Table 2.1 and Table 2.2 of Figure (2) are learned and estimate quantities based on some raw data or at the training phase.

While we examined one application of HMMs, can you envision how a similar approach can lead to identifying top 2 potential realizations of hidden state sequence? How can one expand this model to incorporate confidence intervals? How about modeling more complex scenarios where multiple hidden layers are stacked atop observed states?