Open in app Get started

Eugine Kang Follow

Sep 1, 2017 · 5 min read · Listen

Save

Hidden Markov Model

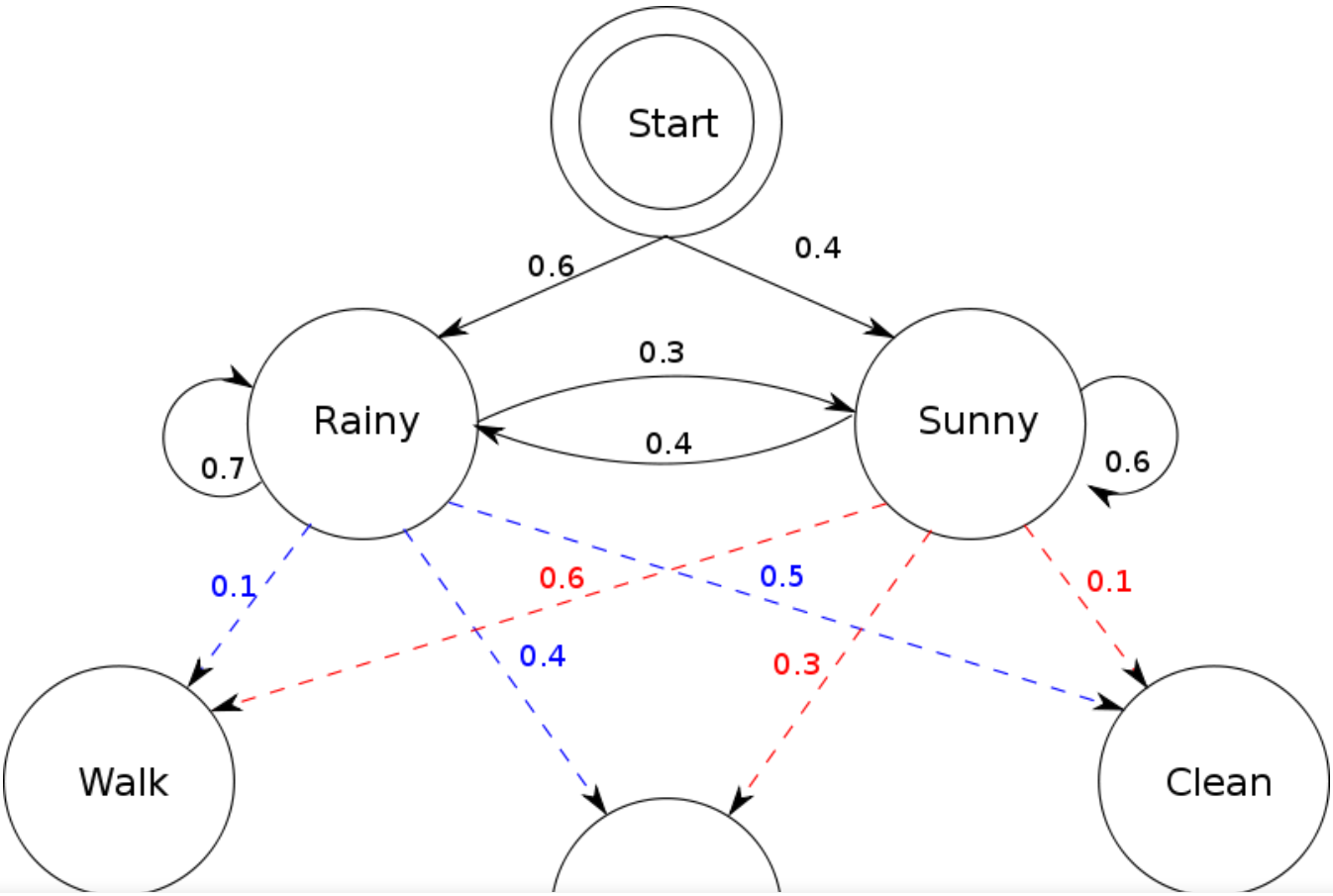
Open in app Get started

Hidden Markov Model (**HMM**) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e. *hidden*) states.

Hidden Markov models are especially known for their application in reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, musical score following, partial discharges and bioinformatics.

Terminology in HMM

The term hidden refers to the first order Markov process behind the observation. Observation refers to the data we know and can observe. Markov process is shown by the interaction between "Rainy" and "Sunny" in the below diagram and each of these are **HIDDEN STATES**.



Open in app Get started

OBSERVATIONS are known data and refers to “Walk”, “Shop”, and “Clean” in the above diagram. In machine learning sense, observation is our training data, and the number of hidden states is our hyper parameter for our model. Evaluation of the model will be discussed later.

$$
\begin{aligned}
T &= \text{length of the observation sequence} \\
N &= \text{number of states in the model} \\
M &= \text{number of observation symbols} \\
Q &= \{q_0, q_1, \ldots, q_{N-1}\} = \text{distinct states of the Markov process} \\
V &= \{0, 1, \ldots, M-1\} = \text{set of possible observations} \\
A &= \text{state transition probabilities} \\
B &= \text{observation probability matrix} \\
\pi &= \text{initial state distribution} \\
\mathcal{O} &= (\mathcal{O}_0, \mathcal{O}_1, \ldots, \mathcal{O}_{T-1}) = \text{observation sequence.}
\end{aligned}
$$

T = don’t have any observation ye = {“Rainy”, “Sunny”}, V = {“Walk”, “Shop”, “Clean”}

1.2K 5

State transition probabilities are the arrows pointing to each hidden state. **Observation probability matrix** are the blue and red arrows pointing to each observations from each hidden state. The matrix are row stochastic meaning the rows add up to 1.

$$a_{ij} = P(\text{state } q_j \text{ at } t+1 \,|\, \text{state } q_i \text{ at } t)$$

$$b_j(k) = P(\text{observation } k \text{ at } t \,|\, \text{state } q_j \text{ at } t)$$

The matrix explains what the probability is from going to one state to another, or going from one state to an observation.

Initial state distribution gets the model going by starting at a hidden state.

Open in app Get started

$$\lambda = (A, B, \pi)$$

How can we build the above model in Python?

```
states = ('Rainy', 'Sunny')

observations = ('walk', 'shop', 'clean')

start_probability = {'Rainy': 0.6, 'Sunny': 0.4}

transition_probability = {
   'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3},
   'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
   }

emission_probability = {
   'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
   'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},
   }

from hmmlearn import hmm
import numpy as np

model = MultinomialHMM(n_components=2)
model.startprob_ = np.array([0.6, 0.4])
model.transmat_ = np.array([[0.7, 0.3],
                            [0.4, 0.6]])
model.emissionprob_ = np.array([[0.1, 0.4, 0.5],
                                [0.6, 0.3, 0.1]])
```

In the above case, emissions are discrete {“Walk”, “Shop”, “Clean”}. MultinomialHMM from the hmmlearn library is used for the above model. GaussianHMM and GMMHMM are other models in the library.

Now with the HMM what are some key problems to solve?

1. **Problem 1**, Given a known model what is the likelihood of sequence O happening?

Open in app Get started

3. **Problem 3**, Given sequence O and number of hidden states, what is the optimal model which maximizes the probability of O?

Problem 1 in Python

```
import math

math.exp(model.score(np.array([[0]])))
# 0.30000000000000004
math.exp(model.score(np.array([[1]])))
# 0.36000000000000004
math.exp(model.score(np.array([[2]])))
# 0.3400000000000001
math.exp(model.score(np.array([[2,2,2]])))
# 0.04590400000000001
```

hmm_2.py hosted with ❤ by GitHub view raw

The probability of the first observation being “Walk” equals to the multiplication of the initial state distribution and emission probability matrix. 0.6 x 0.1 + 0.4 x 0.6 = 0.30 (30%). The log likelihood is provided from calling .score.

Problem 2 in Python

Open in app Get started

```
0.01512
[0 0 1]
0.03675
[0 0 0]
```

Given the known model and the observation {“Shop”, “Clean”, “Walk”}, the weather was most likely {“Rainy”, “Rainy”, “Sunny”} with ~1.5% probability.

Given the known model and the observation {“Clean”, “Clean”, “Clean”}, the weather was most likely {“Rainy”, “Rainy”, “Rainy”} with ~3.6% probability.

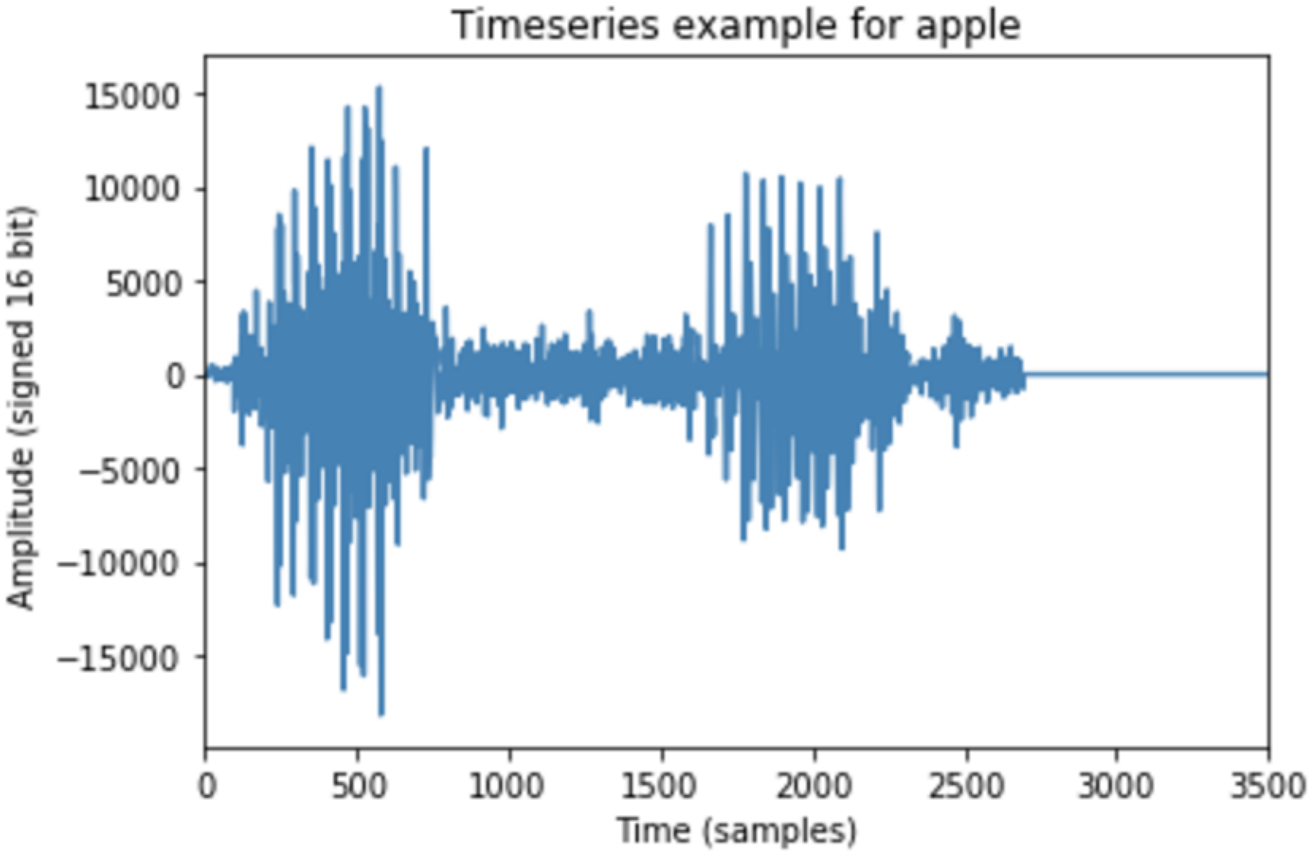
Intuitively, when “Walk” occurs the weather will most likely not be “Rainy”.

Problem 3 in Python

Speech recognition with Audio File: Predict these words

[‘apple’, ‘banana’, ‘kiwi’, ‘lime’, ‘orange’, ‘peach’, ‘pineapple’]

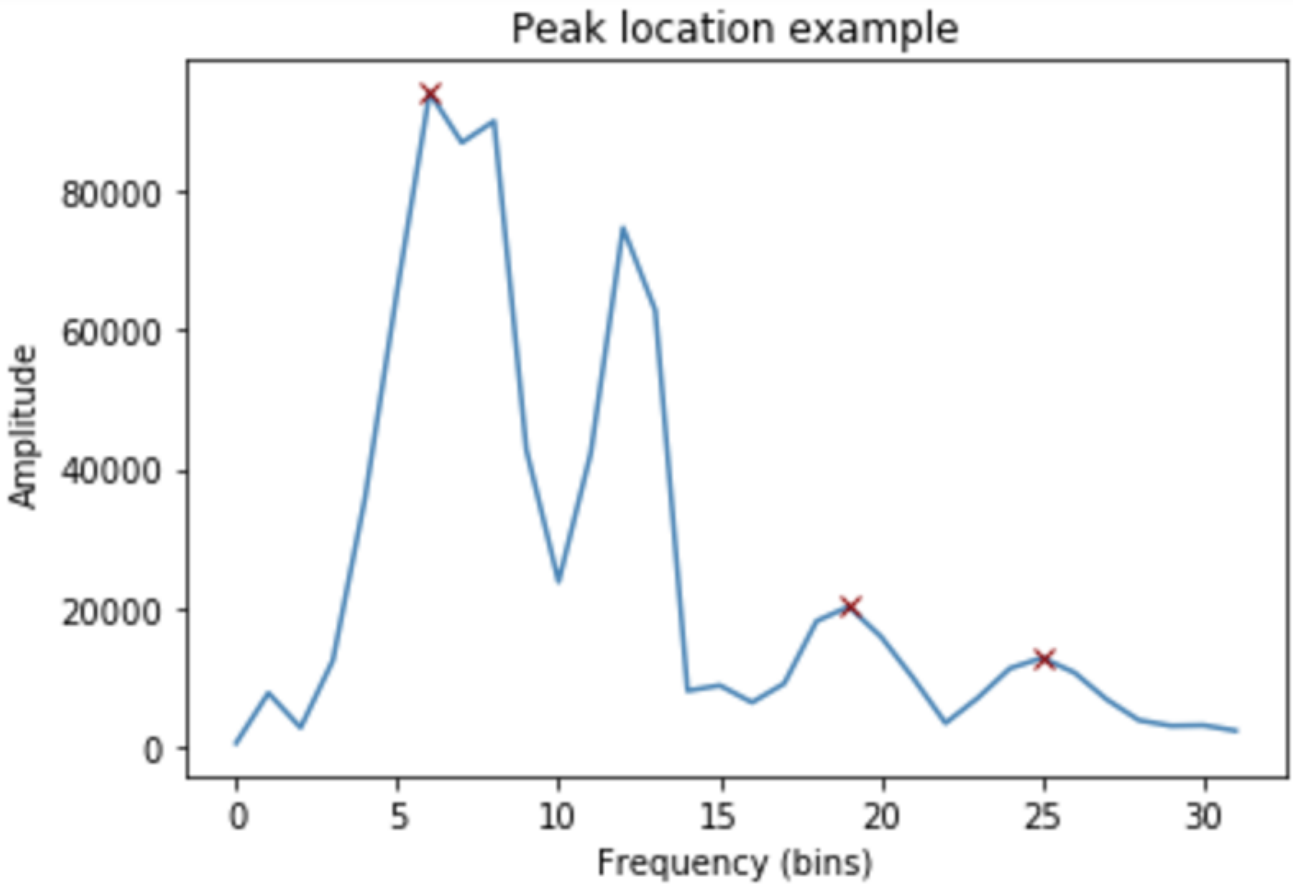

Open in app Get started



Amplitude can be used as the OBSERVATION for HMM, but feature engineering will

Open in app Get started


Open in app Get started



Function **stft** and **peakfind** generates feature for audio signal.

Open in app Get started

The example above was taken from here. Kyle Kastner built HMM class that takes in 3d arrays, I'm using hmmlearn which only allows 2d arrays. This is why I'm reducing the features generated by Kyle Kastner as X_test.mean(axis=2).

Going through this modeling took a lot of time to understand. I had the impression that the target variable needs to be the observation. This is true for time-series.
Classification is done by building HMM for each class and compare the output by calculating the logprob for your input.

Mathematical Solution to Problem 1: Forward Algorithm

$$\alpha_t(i) = P(\mathcal{O}_0, \mathcal{O}_1, \ldots, \mathcal{O}_t, x_t = q_i \mid \lambda)$$

Alpha pass is the probability of OBSERVATION and STATE sequence given model.

$$\alpha_0(i) = \pi_i b_i(\mathcal{O}_0)$$

Alpha pass at time (t) = 0, initial state distribution to i and from there to first observation O0.

$$\alpha_t(i) = \left[\sum_{j=0}^{N-1} \alpha_{t-1}(j) a_{ji}\right] b_i(\mathcal{O}_t)$$


Open in app Get started

$$P(\mathcal{O} \mid \lambda) = \sum_{i=0}^{N-1} \alpha_{T-1}(i).$$

Mathematical Solution to Problem 2: Backward Algorithm

$$\beta_t(i) = P(\mathcal{O}_{t+1}, \mathcal{O}_{t+2}, \ldots, \mathcal{O}_{T-1} \mid x_t = q_i, \lambda).$$

$$\beta_{T-1}(i) = 1, \text{ for } i = 0, 1, \ldots, N-1.$$

$$\beta_t(i) = \sum_{j=0}^{N-1} a_{ij} b_j(\mathcal{O}_{t+1}) \beta_{t+1}(j).$$

$$\gamma_t(i) = P(x_t = q_i \mid \mathcal{O}, \lambda)$$

Given model and observation, probability of being at state qi at time t.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathcal{O} \mid \lambda)}.$$

Mathematical Solution to Problem 3: Forward-Backward Algorithm

Open in app Get started

1. Initialize, $\lambda = (A, B, \pi)$.
2. Compute $\alpha_t(i)$, $\beta_t(i)$, $\gamma_t(i,j)$ and $\gamma_t(i)$.
3. Re-estimate the model $\lambda = (A, B, \pi)$.
4. If $P(\mathcal{O} \mid \lambda)$ increases, goto 2.

$$\gamma_t(i,j) = P(x_t = q_i, x_{t+1} = q_j \mid \mathcal{O}, \lambda)$$

Probability of from state qi to qj at time t with given model and observation

$$\gamma_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)}{P(\mathcal{O} \mid \lambda)}$$

$$\gamma_t(i) = \sum_{j=0}^{N-1} \gamma_t(i,j)$$

Sum of all transition probability from i to j.

$$a_{ij} = \sum_{t=0}^{T-2} \gamma_t(i,j) \Big/ \sum_{t=0}^{T-2} \gamma_t(i) .$$


Open in app Get started

$$b_j(k) = \sum_{\substack{t \in \{0,1,\ldots,T-1\} \\ \mathcal{O}_t = k}} \gamma_t(j) \Big/ \sum_{t=0}^{T-1} \gamma_t(j)$$

Transition and emission probability matrix are estimated with di-gamma.

Iterate if probability for P(O | model) increases

About Help Terms Privacy

Get the Medium app

Download on the App Store GET IT ON Google Play