Day21 in 2023s

SPR Day 19

Inferring the Most Likely State Sequence under an HMM using Viterbi algorithm

Reading: Bishop PRML 13.2.5

Topics:

- Motivation to find most likely hidden state sequence
- Formal Problem Statement (+ why brute force is hard)
- Intuition for a recursive algorithm
- Viterbi algorithm

Motivation: Inferring Sequences of Hidden States

Common task in many sequential data applications

given data $x_1, x_2, \cdots x_T$

produce a "good" guess about system state over time $z_1, z_2, \cdots z_T$

$z_t \in \{1, 2, \cdots K\}$

Examples

E1: Messages over noisy channels

true states: English alphabet $\{a, b, c, d, \cdots x, y, z\}$

observed data: binary codes

x_1	x_2	x_3	$\cdots$	x_T
0	0	0		1
0	1	0		1
0	0	1		1
0	1	0		0
0	1	0		0
	1	0		0

$p(x|z)$ some chance this is a 0 instead but got corrupted

Inference goal:

Which is more likely the message?

a: the bpitish wixl strikeatmidyight

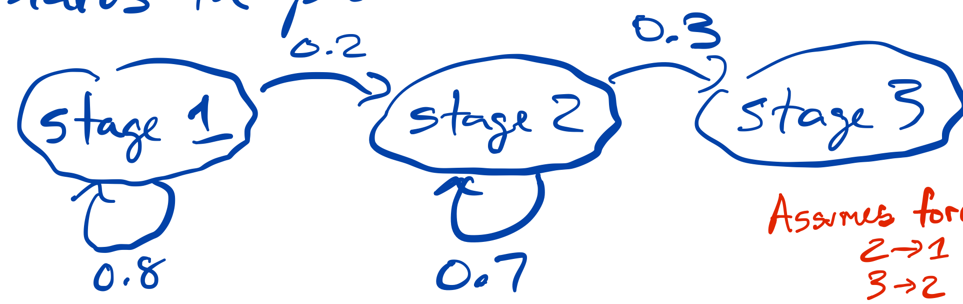
b: the british will strike at midnight

Known probabilities of transitions between letters useful!

E2: Disease status in patient over time

Model

Markov model for disease progression



Assumes forward progression
2→1 NOT allowed
3→2 NOT allowed

Observations

symptoms at each time x_t

Goal:

Given symptom history, what stage was patient at in each timestep?

$$z_1 \quad z_2 \quad \cdots \quad z_T$$

$$z_{1:T} = 1, 1, 2, 2, 2, 2, 3, 3$$

1 = stage 1
2 = stage 2
3 = stage 3

Key idea: need to do inference of all $z_1, z_2 \cdots z_T$ values <u>jointly</u>, will give better, more useful results.

Want to avoid implausible configurations, e.g.

$$z_{1:T} = 1, 2, \boxed{2, 1}, 2, 2, 3, 3$$

Our Markov model says cannot go back from stage 2 to stage 1

Formal Problem Statement
Inference of Most Likely Hidden State Seq.

Given: Data at each timestep: $x_1, x_2 \cdots x_T$

HMM Parameters (assume Gaussian data-given-state distributions)

- $\pi \in \Delta^K$ initial state probas
- $A = \{A_j\}_{j=1}^K$ $A_j \in \Delta^K$ transition probas
- $\mu = \{\mu_k\}_{k=1}^K$ means and variances
  $\sigma = \{\sigma_k\}_{k=1}^K$

Infer: $\hat{z}_1, \hat{z}_2, \cdots \hat{z}_T$, a state sequence that satisfies:

- $$\hat{z}_{1:T} = \underset{z_{1:T} \in \Omega}{\operatorname{argmax}} \; p(z_{1:T} \mid x_{1:T}, \theta)$$

space of all possible sequences of size T w/ K possible symbols

We could call this "MAP estimation" because we are finding the sequence $z_{1:T}$ this is most likely under posterior $p(z \mid x, \theta)$

Brute Force Method

We could enumerate all $z_{1:T} \in \Omega$, and for each one compute $\log p(x_{1:T}, z_{1:T} | \theta)$ [the complete log likelihood] $\Rightarrow$ costs $O(TK^2)$ runtime for each $z_{1:T}$

E.g. for $T=4$, $K=3$ we would do

$z_{1:T}$	$\log p(x_{1:T}, z_{1:T} \mid \theta)$
1, 1, 1, 1	0.001
1, 1, 1, 2	0.003
1, 1, 1, 3	**0.023**
⋮	⋮
2, 1, 1, 1	-0.001
2, 1, 1, 2	0.001
⋮	⋮
4, 4, 4, 3	-0.001
4, 4, 4, 4	-0.002

return $z_{1:T}$ with largest value

Equivalent goals

$$\max_{z_{1:T}} p(z_{1:T} | x_{1:T})$$

$$= \max_{z_{1:T}} \frac{p(z_{1:T}, x_{1:T})}{\text{const}} \quad \text{Bayes rule}$$

$$= \max_{z_{1:T}} \log p(z_{1:T}, x_{1:T}) \quad \text{log is monotonic}$$

Problem: Each row costs $O(TK^2)$

And there are K^T possible rows.

Once $K > 10$ and $T > 10$, this brute force becomes way too impractical

Idea: Approach like FORWARD and BACKWARD algo.

Find subproblems easier to solve, use those to build up overall solution

Base case: $T=1$

This is do-able!

$$\hat{z}_1 = \underset{z_1 \in \{1,2,\dots K\}}{\operatorname{argmax}} \log p(x_1, z_1 | \theta)$$

$$= \underset{k \in \{1,2,\dots K\}}{\operatorname{argmax}} \log p(z_1 = k | \theta) + \log p(x_1 | z_1 = k, \theta)$$

$$\log \pi_k + \underbrace{\log \mathrm{NormPDF}(x_1 | \mu_k, \sigma_k)}_{L_{1k}}$$

Notation:

$$w_{1j} = \log p(z_1 = j) + \log p(x_1 | z_1 = j)$$

Joint log proba of x_1 and $z_1 = j$

Lets define for timesteps $t \in 1 \dots T$

$$L_{tk} = \log p(x_t | z_t = k, \theta)$$

from here on, for easy notation

Next, consider $T=2$

ignore dependence on θ for now...

$$\hat{z}_1, \hat{z}_2 = \underset{k \in \{1,2,\dots K\}}{\operatorname{argmax}} \; \underset{j \in \{1,2,\dots K\}}{\operatorname{argmax}} \log p(z_1 = j, z_2 = k, x_1, x_2)$$

$$= \underset{k}{\operatorname{argmax}} \log p(x_2 | z_2 = k) + \underset{j}{\operatorname{argmax}} \log p(z_2 = k | z_1 = j) + w_{1j}$$

can't do j argmax before k argmax

Track score: Joint log proba of $x_{1:2}$, $z_2 = k$, and best choice for z_1

$$w_{2k} = \log p(x_2 | z_2 = k) + \max_j \left[\log p(z_2 = k | z_1 = j) + w_{1j} \right]$$

Next, consider $T=3$

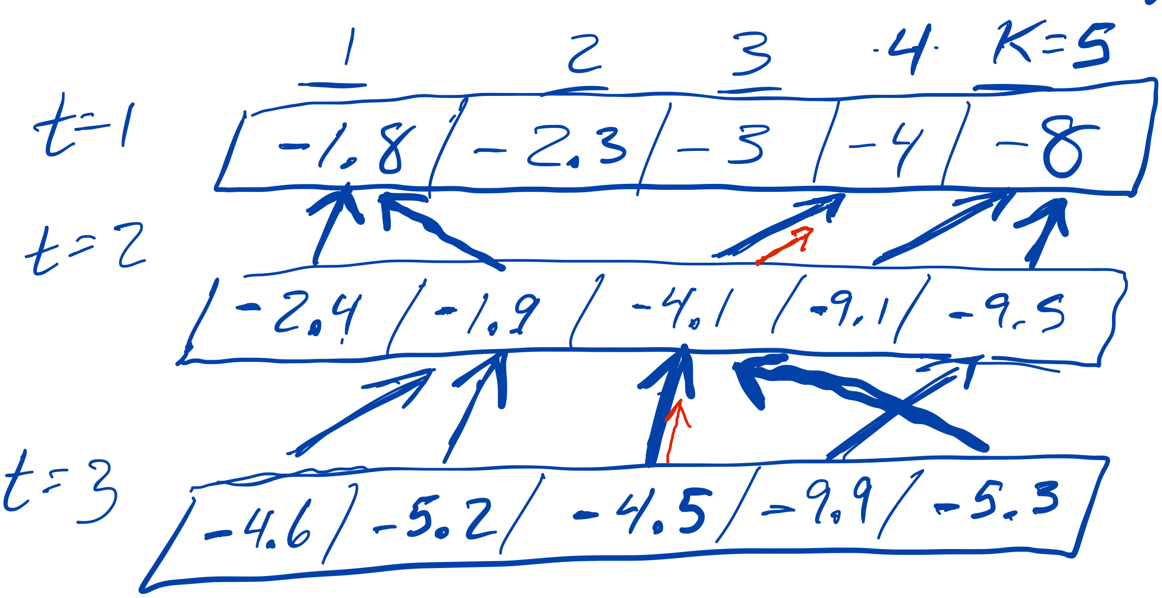
$$\underset{j}{\hat z_1}, \underset{k}{\hat z_2}, \underset{l}{\hat z_3} = \underset{l\in\{1,2,\dots K\}}{\operatorname{argmax}}\Big[\overbrace{\log p(x_3|z_3=l)}^{L_{3l}} + \underset{k\in\{1,\dots K\}}{\operatorname{argmax}} \overbrace{\boxed{\log p(z_3=l|z_2=k)}}^{\log A_{kl}} + \underbrace{\log p(z_1,z_2=k,x_1,x_2)}_{w_{2k}}$$

w_{2k} can tell us this

See the recursive structure?

For each possible timestep t and state k, we track:

- score $w_{tk} \in \mathbb{R}$, the complete log likelihood of best sequence that terminates at k at time t
- backpointer $b_{tk} \in \{1,2,\dots K\}$ which state j the best possible path uses at previous time $t-1$



numbers represent w_{tk} scores

arrows represent backpointers

given this configuration, we find the best sequence by looking at last row ($T=3$) and picking overall highest score (highest joint log likelihood).

We pick -4.5 so $z_3 = 3$.

Then following back pointers we have $z_2 = 3$, $z_1 = 4$

So given w_t and b_t, we can solve the problem

This insight gives us the

VITERBI Algorithm

Dynamic Programming

Input: π, A: HMM parameters

$L = T \times K$ array where $L_{tk} = \log p(x_t \mid z_t = k, \theta)$

Think of L as deterministic summary of x, μ, σ^2

Initial: $t=1$

$$w_{1k} = L_{1k} + \log \pi_k \quad \text{for } k \text{ in } 1 \ldots K$$

$$b_{1k} = -1$$

(unused, no backpointer at $t=1$)

Recursive update: $t = 2, 3, \ldots, T$

for t in $2, \ldots T$:

score
$$w_{tk} = L_{tk} + \max_j \left(\log A_{jk} + w_{t-1,j} \right) \quad \forall k$$

backpointer
$$b_{tk} = \operatorname{argmax}_j \log A_{jk} + w_{t-1,j} \quad \forall k$$

Backward sweep to identify state sequence $T, T-1, \ldots, 2, 1$

$$\hat{z}_T = \operatorname{argmax}_k w_{Tk}$$

for $t = T-1, T-2, \cdots 1$:

$$\hat{z}_t = b_{t+1, \hat{z}_{t+1}}$$

follow back pointer

Return $\hat{z}_1, \hat{z}_2, \ldots \hat{z}_{T-1}, \hat{z}_T$

Runtime: Linear in T
Quadratic in K

need to visit each k and perform $\max_{j \in \{1, \ldots K\}}$

$O(TK^2)$