

732A96 Advanced Machine Learning

LAB 2:: HIDDEN MARKOV MODELS

Johannes Hedström

STIMA
Department of Computer and Information Science
Linköpings universitet

2024-09-16

Contents

1	Description	1
2	Question 1	1
3	Question 2	2
4	Question 3	3
5	Question 4	3
6	Question 5	3
7	Question 6	3
8	Question 7	4

```
library(HMM)
library(knitr)
```

1 Description

The purpose of the lab is to put in practice some of the concepts covered in the lectures. To do so, you are asked to model the behavior of a robot that walks around a ring. The ring is divided into 10 sectors. At any given time point, the robot is in one of the sectors and decides with equal probability to stay in that sector or move to the next sector. You do not have direct observation of the robot. However, the robot is equipped with a tracking device that you can access. The device is not very accurate though: If the robot is in the sector i , then the device will report that the robot is in the sectors $[i - 2, i + 2]$ with equal probability.

2 Question 1

Build a hidden Markov model (HMM) for the scenario described above. Note that the documentation for the `initHMM` function says that the `emissionProbs` matrix should be of size `[number of states]x[number of states]`. This is wrong. It should be of size `[number of states]x[number of symbols]`. The package works correctly, though. It is just the documentation that is wrong.

```
# States
states <- as.character(1:10)
# Movements for the robot
sym <- paste0('State_',states)
# starting probabilities, equal for all states
startprobs <- rep(1/10,10)

# Transitions probabilities 1/2 probability to stay or change

t_prob <- matrix(0,nrow=10,ncol=10) # empty matrix
diag(t_prob) <- 0.5 # 0.5 to stay
diag(t_prob[, -1]) <- 0.5 # 0.5 to move to NEXT sector
t_prob[10,1] <- 0.5 # left bottom corner(step from 10 to 1)

# emissionprobs
e_prob <- matrix(0,nrow=10, ncol=10) # empty matrix

for (i in 1:10) { # filling the matrix with probabilities, each row sums to 1
  j <- (i-2):(i+2)
  if(11 %in% j){ # if we are going over 10 then go back to 1 and upwards
    j_1 <- j[1:which(j==10)]
    j <- c(1:(length(j) - length(j_1)),j_1)
  }
  if(0 %in% j){ # if we are getting a value under 1 then go from 10 and downwards
    j_1 <- j[which(j==1):length(j)]
    j <- c((10-(length(j) - length(j_1))+1):10,j_1)
  }
}
```

```
e_prob[i,j] <- 1/5
}

hmm <- initHMM(states,sym, startprobs,transProbs=t_prob,e_prob)

knitr::kable(t_prob, caption='Transitions probabilities')
```

Table 1: Transitions probabilities

0.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.5	0.5	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.5	0.5	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.5	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.5	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.5
0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5

```
knitr::kable(e_prob, caption='Emission probabilities[i - 2, i + 2]')
```

Table 2: Emission probabilities[i - 2, i + 2]

0.2	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.2
0.2	0.2	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.2
0.2	0.2	0.2	0.2	0.2	0.0	0.0	0.0	0.0	0.0
0.0	0.2	0.2	0.2	0.2	0.2	0.0	0.0	0.0	0.0
0.0	0.0	0.2	0.2	0.2	0.2	0.2	0.0	0.0	0.0
0.0	0.0	0.0	0.2	0.2	0.2	0.2	0.2	0.0	0.0
0.0	0.0	0.0	0.0	0.2	0.2	0.2	0.2	0.2	0.0
0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.2	0.2	0.2
0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.2	0.2
0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.2

3 Question 2

Simulate the HMM for 100 time steps.

```
sims <- simHMM(hmm,100) # simulate 100 steps

sim_state <- sims$states
sim_obs <- sims$observation

knitr::kable(sim_obs[1:10],caption='First ten simulated observations')
```

Table 3: First ten simulated observations

x
State_8
State_8
State_9
State_10
State_7
State_10
State_9
State_1
State_7
State_9

4 Question 3

Discard the hidden states from the sample obtained above. Use the remaining observations to compute the filtered and smoothed probability distributions for each of the 100 time points. Compute also the most probable path.

5 Question 4

Compute the accuracy of the filtered and smoothed probability distributions, and of the most probable path. That is, compute the percentage of the true hidden states that are guessed by each method.

Hint: Note that the function `forward` in the HMM package returns probabilities in log scale. You may need to use the functions `exp` and `prop.table` in order to obtain a normalized probability distribution. You may also want to use the functions `apply` and `which.max` to find out the most probable states. Finally, recall that you can compare two vectors `A` and `B` element wise as `A==B`, and that the function `table` will count the number of times that the different elements in a vector occur in the vector

6 Question 5

Repeat the previous exercise with different simulated samples. In general, the smoothed distributions should be more accurate than the filtered distributions. Why ? In general, the smoothed distributions should be more accurate than the most probable paths, too. Why ?

7 Question 6

Is it always true that the later in time (i.e., the more observations you have received) the better you know where the robot is ? Hint: You may want to compute the entropy of the filtered distributions with the function `entropy.empirical` of the package `entropy`.

8 Question 7

Consider any of the samples above of length 100. Compute the probabilities of the hidden states for the time step 101.