Lab 2

Axel Holmberg (axeho681)

9/22/2020

Task 1

Build a hidden Markov model (HMM) for the scenario described above.

```
## [1] "States:"

##  [1]  1  2  3  4  5  6  7  8  9 10

## [1] "Symbol states:"

##  [1]  1  2  3  4  5  6  7  8  9 10

## [1] "Transition matrix:"

##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]  0.5  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0   0.5
##  [2,]  0.5  0.5  0.0  0.0  0.0  0.0  0.0  0.0  0.0   0.0
##  [3,]  0.0  0.5  0.5  0.0  0.0  0.0  0.0  0.0  0.0   0.0
##  [4,]  0.0  0.0  0.5  0.5  0.0  0.0  0.0  0.0  0.0   0.0
##  [5,]  0.0  0.0  0.0  0.5  0.5  0.0  0.0  0.0  0.0   0.0
##  [6,]  0.0  0.0  0.0  0.0  0.5  0.5  0.0  0.0  0.0   0.0
##  [7,]  0.0  0.0  0.0  0.0  0.0  0.5  0.5  0.0  0.0   0.0
##  [8,]  0.0  0.0  0.0  0.0  0.0  0.0  0.5  0.5  0.0   0.0
##  [9,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.5  0.5   0.0
## [10,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.5   0.5

## [1] "Emission matrix:"

##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]  0.2  0.2  0.2  0.0  0.0  0.0  0.0  0.0  0.2   0.2
##  [2,]  0.2  0.2  0.2  0.2  0.0  0.0  0.0  0.0  0.0   0.2
##  [3,]  0.2  0.2  0.2  0.2  0.2  0.0  0.0  0.0  0.0   0.0
##  [4,]  0.0  0.2  0.2  0.2  0.2  0.2  0.0  0.0  0.0   0.0
##  [5,]  0.0  0.0  0.2  0.2  0.2  0.2  0.2  0.0  0.0   0.0
##  [6,]  0.0  0.0  0.0  0.2  0.2  0.2  0.2  0.2  0.0   0.0
##  [7,]  0.0  0.0  0.0  0.0  0.2  0.2  0.2  0.2  0.2   0.0
##  [8,]  0.0  0.0  0.0  0.0  0.0  0.2  0.2  0.2  0.2   0.2
##  [9,]  0.2  0.0  0.0  0.0  0.0  0.0  0.2  0.2  0.2   0.2
## [10,]  0.2  0.2  0.0  0.0  0.0  0.0  0.0  0.2  0.2   0.2

## [1] "Summary of HMM:"

##              Length Class  Mode
## States        10    -none- numeric
## Symbols       10    -none- numeric
## startProbs    10    -none- numeric
## transProbs   100    -none- numeric
```

```
## emissionProbs 100    -none- numeric
```

Task 2

Simulate the HMM for 100 time steps.

```
## [1] "The observed simulated steps are:"
```

```
##   [1]  7 10  8 10 10  6  4  6  3  3  2  4 10  4 10  3  3  1  2  2  1  1 10 10 10
##  [26]  6  6  8  7  4  7  2  3 10  3  3 10 10  9  6 10  9  7  9  8  6  6  5  6  3
##  [51]  6  5  2  4  4 10 10  1  3 10  1  2  8  8  7 10  6  8  6  7  6  8  4  7  7
##  [76]  4  4  6  4  6  4  5  2  4  3  3  4  3 10 10 10  8  7  7  6  8  7  9  7  5
```

Task 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.

The most probable path is:

```
##   [1]  8  8  8  8  8  7  6  5  4  3  2  2  2  2  1  1  1  1  1 10  9  9  8  8  8
##  [26]  7  6  6  5  5  5  4  3  2  1  1 10  9  8  8  8  7  7  7  6  5  4  4  4  4
##  [51]  4  3  2  2  2  1  1  1  1  1  1 10  9  8  8  8  7  6  6  6  6  6  5  5  5
##  [76]  4  4  4  4  4  3  3  2  2  2  2  2  1 10  9  8  7  7  7  7  7  7  7  6  5
```

Task 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.

```
## [1] "Filtered:"

## [1] 0.36

## [1] "Smoothed:"

## [1] 0.64

## [1] "Most probable path:"

## [1] 0.28
```

Task 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?

```
## [1] 0.4

## [1] 0.66

## [1] 0.04
```

The smoothed distribution is more accurate as it takes both α and β from the forward and the backward part of the algorithm with the formula $p(z^t|x^{0:t}) = (\alpha(z^t)\beta(z^t))/(\sum_z t\alpha(z^t)\beta(z^t))$ compared to the filtering which only takes α in to account $p(z^t|x^{0:t}) = \alpha(z^t)/\sum_z t\alpha(z^t)$.

Task 6

Is it true that the more observations you have the better you know where the robot is?

The entropy for the case with 100 simluated cases is:

```
## [1] 4.439829
```

The entropy for the case with 300 simluated cases is:

```
## [1] 4.439829
```

Task 7

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

The 101st step would be:

```
##               [,1]
##  [1,] 0.000000e+00
##  [2,] 0.000000e+00
##  [3,] 0.000000e+00
##  [4,] 0.000000e+00
##  [5,] 1.697914e-81
##  [6,] 5.521304e-81
##  [7,] 6.326287e-81
##  [8,] 2.502896e-81
##  [9,] 0.000000e+00
## [10,] 0.000000e+00
```

Appendix for code

```r
library(HMM)

##### TASK 1 #####

# 10 Different places
states <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
symbol_states <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

# Equal probability of it moving and staying
transition_probs <- matrix(
                   c(0.5, 0.5, 0, 0, 0, 0, 0, 0, 0, 0,
                     0, 0.5, 0.5, 0, 0, 0, 0, 0, 0, 0,
                     0, 0, 0.5, 0.5, 0, 0, 0, 0, 0, 0,
                     0, 0, 0, 0.5, 0.5, 0, 0, 0, 0, 0,
                     0, 0, 0, 0, 0.5, 0.5, 0, 0, 0, 0,
                     0, 0, 0, 0, 0, 0.5, 0.5, 0, 0, 0,
                     0, 0, 0, 0, 0, 0, 0.5, 0.5, 0, 0,
                     0, 0, 0, 0, 0, 0, 0, 0.5, 0.5, 0,
                     0, 0, 0, 0, 0, 0, 0, 0, 0.5, 0.5,
                     0.5, 0, 0, 0, 0, 0, 0, 0, 0, 0.5),
                     nrow=10, ncol=10)


# Equal probability of it being i-2,i-1,i,i+1 and i+2
emission_probs <-  matrix(
                    c(0.2, 0.2, 0.2, 0, 0, 0, 0, 0, 0.2, 0.2,
                      0.2, 0.2, 0.2, 0.2, 0, 0, 0, 0, 0, 0.2,
                      0.2, 0.2, 0.2, 0.2, 0.2, 0, 0, 0, 0, 0,
                      0, 0.2, 0.2, 0.2, 0.2, 0.2, 0, 0, 0, 0,
                      0, 0, 0.2, 0.2, 0.2, 0.2, 0.2, 0, 0, 0,
                      0, 0, 0, 0.2, 0.2, 0.2, 0.2, 0.2, 0, 0,
                      0, 0, 0, 0, 0.2, 0.2, 0.2, 0.2, 0.2, 0,
                      0, 0, 0, 0, 0, 0.2, 0.2, 0.2, 0.2, 0.2,
                      0.2, 0, 0, 0, 0, 0, 0.2, 0.2, 0.2, 0.2,
                      0.2, 0.2, 0, 0, 0, 0, 0, 0.2, 0.2, 0.2),
                      nrow=10, ncol=10)


hmm <-
  initHMM(States = states,
          Symbols = symbol_states,
          startProbs = rep(0.1, 10),
          transProbs = transition_probs,
          emissionProbs = emission_probs
          )



##### TASK 2 #####

set.seed(12345)
simulated_steps <- simHMM(hmm,100)
```

```r
##### TASK 3 #####

filter_function <- function(alphas) {
  filtered <- matrix(NA,
                     nrow = dim(alphas)[1],
                     ncol = dim(alphas)[2])
  for (t in 1:dim(alphas)[1]) {
    filtered[t,] <- alphas[t,] / sum(alphas[t,])
  }

  return(filtered)
}


smooth_function <- function(alphas, betas) {
  smoothed <- matrix(NA,
                     nrow = dim(alphas)[1],
                     ncol = dim(alphas)[2])

  for (t in 1:dim(alphas)[1]) {
    smoothed[t, ] <-
      (alphas[t, ] * betas[t, ]) / (sum(alphas[t, ] * betas[t, ]))
  }

  return (smoothed)
}


hmm_forward_alpha <-
  exp(forward(
    hmm = hmm,
    observation = simulated_steps$observation
  ))

hmm_backward_beta <-
 exp( backward(
    hmm = hmm,
    observation = simulated_steps$observation
  ))

filtered <- filter_function(hmm_forward_alpha)

smoothed <-
  smooth_function(hmm_forward_alpha, hmm_backward_beta)

viterbi <- viterbi(hmm, simulated_steps$observation)


##### TASK 4 #####

accuracy_function <- function(prediction, true) {
  confusion_matrix <- table(prediction, true)
  accuracy <- sum(diag(confusion_matrix))/sum(confusion_matrix)
```

```r
  return (accuracy)
}

filtered_prediction <- apply(t(filtered), MARGIN = 1, which.max)
smoothed_prediction <- apply(t(smoothed), MARGIN = 1, which.max)


accuracy_function(filtered_prediction, simulated_steps$states)
accuracy_function(smoothed_prediction, simulated_steps$states)
accuracy_function(viterbi, simulated_steps$states)


##### TASK 5 #####
#Different seed
set.seed(67890)
simulated_steps_seeded <- simHMM(hmm,100)

hmm_forward_alpha_seeded <-
  exp(forward(
    hmm = hmm,
    observation = simulated_steps_seeded$observation
  ))

hmm_backward_beta_seeded <-
  exp( backward(
    hmm = hmm,
    observation = simulated_steps_seeded$observation
  ))

filtered_seeded <- filter_function(hmm_forward_alpha_seeded)

smoothed_seeded <-
  smooth_function(hmm_forward_alpha_seeded, hmm_backward_beta_seeded)

viterbi_seeded <- viterbi(hmm, simulated_steps$observation)

filtered_prediction_seeded <- apply(t(filtered_seeded), MARGIN = 1, which.max)
smoothed_prediction_seeded <- apply(t(smoothed_seeded), MARGIN = 1, which.max)


accuracy_function(filtered_prediction_seeded, simulated_steps_seeded$states)
accuracy_function(smoothed_prediction_seeded, simulated_steps_seeded$states)
accuracy_function(viterbi_seeded, simulated_steps_seeded$states)


##### TASK 6 #####
library(entropy)


simulated_steps_300 <- simHMM(hmm, 300)

hmm_forward_alpha_300 <-
  exp(forward(hmm = hmm,
```

```r
             observation = simulated_steps_300$observation))

hmm_backward_beta_300 <-
  exp(backward(hmm = hmm,
               observation = simulated_steps_300$observation))

filtered_300 <- filter_function(hmm_forward_alpha)

smoothed_300 <-
  smooth_function(hmm_forward_alpha, hmm_backward_beta)

viterbi_300 <- viterbi(hmm, simulated_steps_300$observation)

filtered_prediction_300 <- apply(t(filtered), MARGIN = 1, which.max)

entropy_100 <- entropy.empirical(filtered_prediction)
entropy_300 <- entropy.empirical(filtered_prediction_300)

entropy_100
entropy_300


##### TASK 7 #####

step_101 <- transition_probs %*% t(filtered)[100, ]

print(step_101)
```