Lab2_Thijs

Thijs Quast (thiqu264)

9/24/2019

Contents

Question 1

```
library(HMM)

# Parameter initialization, vectors
states <- as.character(c(1:10))
symbols <- as.character(c(1:10))
startprobs <- rep((1/length(states)), length(states))

# Parameter initialization, matrices

# transprobs

transprobs <- diag(x = 0.5, nrow = 10, ncol = 10)
transprobs[1,2] <- 0.5
transprobs[2,3] <- 0.5
transprobs[3,4] <- 0.5
transprobs[4,5] <- 0.5
transprobs[5,6] <- 0.5
transprobs[6,7] <- 0.5
transprobs[7,8] <- 0.5
transprobs[8,9] <- 0.5
transprobs[9,10] <- 0.5
transprobs[10,1] <- 0.5

colnames(transprobs) <- as.character(c(1:10))
rownames(transprobs) <- as.character(c(1:10))


# emissionProbs
emissionprobs <- matrix(data = 0, nrow = 10, ncol = 10)
emissionprobs[1, c(1,2,3,9,10)] <- 0.2
emissionprobs[2, c(1,2,3,4,10)] <- 0.2
emissionprobs[3, c(1,2,3,4,5)] <- 0.2
emissionprobs[4, c(2,3,4,5,6)] <- 0.2
emissionprobs[5, c(3,4,5,6,7)] <- 0.2
emissionprobs[6, c(4,5,6,7,8)] <- 0.2
emissionprobs[7, c(5,6,7,8,9)] <- 0.2
emissionprobs[8, c(6,7,8,9,10)] <- 0.2
emissionprobs[9, c(7,8,9,10,1)] <- 0.2
emissionprobs[10, c(8,9,10,1,2)] <- 0.2


colnames(emissionprobs) <- as.character(c(1:10))
rownames(emissionprobs) <- as.character(c(1:10))
```

```
# Init HMM
HMM <- initHMM(States = states, Symbols = symbols, startProbs = startprobs,
               transProbs = transprobs,
               emissionProbs = emissionprobs)
```

Question 2

```
simulation <- simHMM(HMM, length = 100)
simulation
```

```
## $states
##   [1] "9"  "10" "10" "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "8"  "8"  "8"
##  [15] "8"  "8"  "9"  "9"  "10" "1"  "2"  "3"  "4"  "5"  "6"  "6"  "6"  "7"
##  [29] "7"  "8"  "9"  "10" "1"  "2"  "3"  "3"  "3"  "3"  "3"  "4"  "5"  "5"
##  [43] "5"  "6"  "7"  "7"  "8"  "9"  "10" "10" "10" "10" "1"  "1"  "1"  "1"
##  [57] "2"  "2"  "3"  "4"  "5"  "5"  "5"  "5"  "6"  "7"  "7"  "7"  "8"  "8"
##  [71] "8"  "8"  "9"  "9"  "10" "10" "1"  "2"  "3"  "3"  "3"  "4"  "5"  "6"
##  [85] "7"  "8"  "8"  "9"  "10" "10" "1"  "2"  "2"  "3"  "3"  "3"  "3"  "4"
##  [99] "4"  "4"
##
## $observation
##   [1] "1"  "10" "1"  "2"  "10" "4"  "4"  "4"  "4"  "7"  "10" "6"  "8"  "10"
##  [15] "7"  "10" "9"  "7"  "1"  "3"  "10" "2"  "4"  "4"  "4"  "4"  "7"  "7"
##  [29] "9"  "10" "8"  "9"  "3"  "2"  "4"  "5"  "4"  "5"  "5"  "2"  "4"  "3"
##  [43] "5"  "7"  "8"  "7"  "8"  "9"  "10" "1"  "1"  "2"  "10" "10" "2"  "2"
##  [57] "10" "4"  "2"  "3"  "4"  "4"  "6"  "5"  "8"  "7"  "5"  "9"  "8"  "7"
##  [71] "8"  "8"  "8"  "8"  "8"  "2"  "3"  "3"  "4"  "3"  "4"  "2"  "7"  "4"
##  [85] "8"  "10" "7"  "7"  "10" "9"  "10" "4"  "3"  "3"  "2"  "2"  "3"  "4"
##  [99] "2"  "5"
```

Question 3

```
# Generate alphas
alphas <- exp(forward(hmm = HMM, observation = simulation$observation))

# Generate betas
betas <- exp(backward(hmm = HMM, observation = simulation$observation))
```

```
# Filtered probability distributions:
filtered <- prop.table(alphas, margin = 2)

# Check if filtered probabilities sum to 1:
apply(filtered, 2, sum)
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

```
##  91  92  93  94  95  96  97  98  99 100
##   1   1   1   1   1   1   1   1   1   1
```

```r
# Smoothed probability distributions:
alphas_betas <- alphas*betas
smoothed <- prop.table(alphas_betas, margin = 2)

# Check if smoothed probabilites sum to 1:
apply(smoothed, 2, sum)
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  91  92  93  94  95  96  97  98  99 100
##   1   1   1   1   1   1   1   1   1   1
```

```r
# Most probably path:
most_probable_path <- viterbi(hmm = HMM, observation = simulation$observation)
```

Question 4

```r
# Guessed paths:
guessed_filtered <- as.character(apply(filtered, MARGIN = 2, FUN=which.max))
guessed_smoothed <- as.character(apply(smoothed, MARGIN = 2, FUN=which.max))
```

```r
# Accuracy filtered:
filterd_table <- table(guessed_filtered==simulation$states)
accuracy_filtered <- filterd_table[2]/sum(filterd_table)
```

```r
# Accuracy smoothed:
smoothed_table <- table(guessed_smoothed == simulation$states)
accuracy_smoothed <- smoothed_table[2]/sum(smoothed_table)
```

```r
# Accuracy most probable path:
probable_table <- table(most_probable_path == simulation$states)
accuracy_probable <- probable_table[2]/sum(probable_table)
```

```r
library(knitr)
df <- as.data.frame(cbind(accuracy_filtered, accuracy_smoothed, accuracy_probable))
colnames(df) <- c("Filtered", "Smoothed", "Probable")
rownames(df) <- c("Accuracy")
kable(df, caption = "Accuracy table")
```

Table 1: Accuracy table

	Filtered	Smoothed	Probable
Accuracy	0.46	0.75	0.47

Question 5

```
df2 <- as.data.frame(matrix(data = NA, nrow = 50, ncol = 3))
colnames(df2) <- c("Filtered", "Smoothed", "Probable")
```

```
for (i in 1:50){
  simulation <- simHMM(HMM, length = 100)

  # Generate alphas
  alphas <- exp(forward(hmm = HMM, observation = simulation$observation))

  # Generate betas
  betas <- exp(backward(hmm = HMM, observation = simulation$observation))

  # Filtered probability distributions:
  filtered <- prop.table(alphas, margin = 2)

  # Smoothed probability distributions:
  alphas_betas <- alphas*betas
  smoothed <- prop.table(alphas_betas, margin = 2)

  # Most probably path:
  most_probable_path <- viterbi(hmm = HMM, observation = simulation$observation)

  # Guessed paths:
  guessed_filtered <- as.character(apply(filtered, MARGIN = 2, FUN=which.max))
  guessed_smoothed <- as.character(apply(smoothed, MARGIN = 2, FUN=which.max))

  # Accuracy filtered:
  filterd_table <- table(guessed_filtered==simulation$states)
  accuracy_filtered <- filterd_table[2]/sum(filterd_table)

  # Accuracy smoothed:
  smoothed_table <- table(guessed_smoothed == simulation$states)
  accuracy_smoothed <- smoothed_table[2]/sum(smoothed_table)

  # Accuracy most probable path:
  probable_table <- table(most_probable_path == simulation$states)
  accuracy_probable <- probable_table[2]/sum(probable_table)

  df2[i,] <- cbind(accuracy_filtered, accuracy_smoothed, accuracy_probable)
}
```

```r
df2$Sample <- c(1:50)
```

```r
kable(df2, caption = "Accuracy table, 50 samples")
```

Table 2: Accuracy table, 50 samples

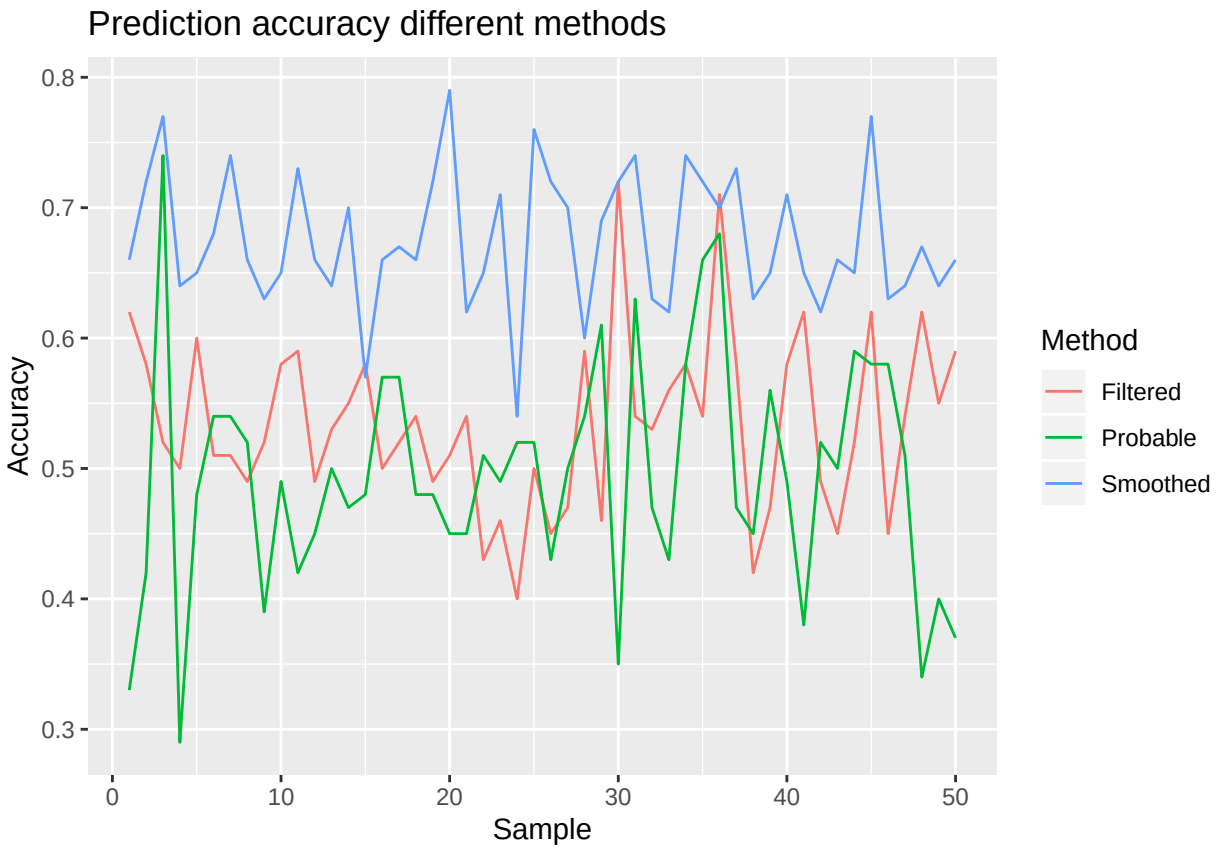
Filtered	Smoothed	Probable	Sample
0.62	0.66	0.33	1
0.58	0.72	0.42	2
0.52	0.77	0.74	3
0.50	0.64	0.29	4
0.60	0.65	0.48	5
0.51	0.68	0.54	6
0.51	0.74	0.54	7
0.49	0.66	0.52	8
0.52	0.63	0.39	9
0.58	0.65	0.49	10
0.59	0.73	0.42	11
0.49	0.66	0.45	12
0.53	0.64	0.50	13
0.55	0.70	0.47	14
0.58	0.57	0.48	15
0.50	0.66	0.57	16
0.52	0.67	0.57	17
0.54	0.66	0.48	18
0.49	0.72	0.48	19
0.51	0.79	0.45	20
0.54	0.62	0.45	21
0.43	0.65	0.51	22
0.46	0.71	0.49	23
0.40	0.54	0.52	24
0.50	0.76	0.52	25
0.45	0.72	0.43	26
0.47	0.70	0.50	27
0.59	0.60	0.54	28
0.46	0.69	0.61	29
0.72	0.72	0.35	30
0.54	0.74	0.63	31
0.53	0.63	0.47	32
0.56	0.62	0.43	33
0.58	0.74	0.58	34
0.54	0.72	0.66	35
0.71	0.70	0.68	36
0.58	0.73	0.47	37
0.42	0.63	0.45	38
0.47	0.65	0.56	39
0.58	0.71	0.49	40
0.62	0.65	0.38	41
0.49	0.62	0.52	42
0.45	0.66	0.50	43
0.52	0.65	0.59	44
0.62	0.77	0.58	45

Filtered	Smoothed	Probable	Sample
0.45	0.63	0.58	46
0.54	0.64	0.51	47
0.62	0.67	0.34	48
0.55	0.64	0.40	49
0.59	0.66	0.37	50

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(ggplot2)

df2 <- gather(df2, key = "Method", value = "Accuracy", -Sample)

plot <- ggplot(df2, aes(Sample, Accuracy, col=Method)) + geom_line() +
  ggtitle("Prediction accuracy different methods")
plot
```

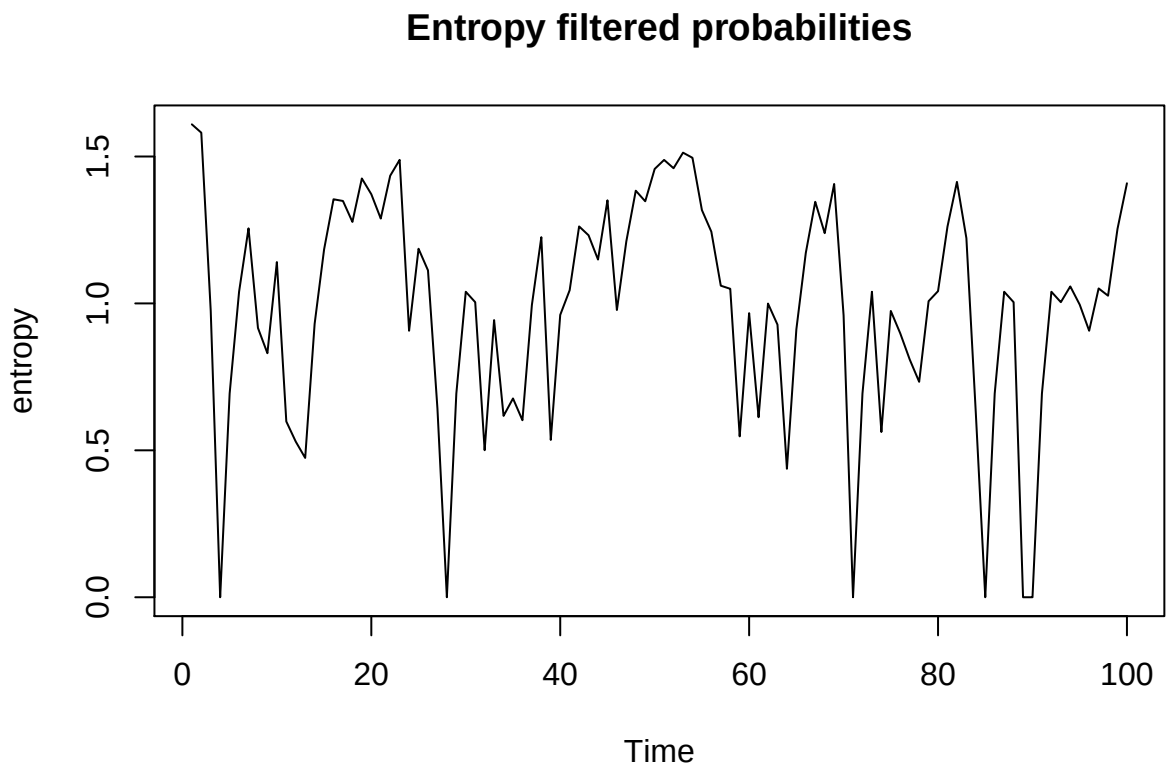


Smoothed probabilities show higher accuracies, probably this is due to the fact that according to the formula it uses all observations (0:T). Whilst, filtered only used 0:t. Viterbi algorithm (most probable path) has to deal with the constraint that it has to come up with a feasible paths. I.e. no unrealistic steps.

Question 6

```
library(entropy)

# Entropy is level of uncertainty, the higher the uncertainty, the more information
entropy <- apply(filtered, MARGIN = 2, FUN = entropy.empirical)
```

```
plot(entropy, type = "l", main = "Entropy filtered probabilities", xlab = "Time")
```



No, the entropy plot shows fluctuations. So, also at later time points when the algorithm can use more information it is still not always certain.

Question 7

```
time_101 <- filtered[,100] %*% transprobs
time_101
```

```
##      1 2 3    4         5         6     7         8          9 10
## [1,] 0 0 0 0.05 0.2043478 0.3336957 0.275 0.1163043 0.02065217  0
```

Using the estimated predictions where the robot could be. From here on the robot moves one step further according to the transition matrix.