AI HW #6

Maximilian Schulten (mls384)

2024-10-13

Exercise 1

a)

We check this by verifying the axioms of probability. Namely, every element $0 \leq P(x = i, y = j) \leq 1$, $\forall i, j \in \{1, 2, 3, 4\}$: This clearly holds by inspection. Moreover that $\sum_{i,j} P(x = i, y = j) = 1$. Clearly demonstrated below:

```
0.06+0.08+0.04+0.02+0.12+0.16+0.08+0.04+0.09+0.12+0.06+0.03+0.03+0.04+0.02+0.01
```

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

b)

Define $P_x(x)$ as the marginal distribution of x.

$$P_x(x = i) := \sum_j p(x = i, y = j), \ \forall i \in \{1, 2, 3, 4\}$$

So we fix x and let y vary. Summing the rows we find:

```
joint_prob <- matrix(c(
  0.06, 0.08, 0.04, 0.02,
  0.12, 0.16, 0.08, 0.04,
  0.09, 0.12, 0.06, 0.03,
  0.03, 0.04, 0.02, 0.01
), nrow = 4, byrow = TRUE)

px <- rowSums(joint_prob)
py <- colSums(joint_prob)

px
```

```
## [1] 0.2 0.4 0.3 0.1
```

```
py
```

```
## [1] 0.3 0.4 0.2 0.1
```

So we conclude:

$$P_x(x) = \begin{cases} 0.2 \,, \ x = 1 \\ 0.4 \,, \ x = 2 \\ 0.3 \,, \ x = 3 \\ 0.1 \,, \ x = 4 \end{cases}$$

c)

X and Y are independent iff their joint probabilities can be factored as a product of their marginal probabilities. Namely we need to check that $P(x = i, y = j) = P_x(x = i) \cdot P_y(y = j)$ for every element of the table. We can easily see that this is the case. Hence X and Y are independent.

Exercise 2

a)

Proposition: $P(a|b, c) = P(b|a, c) \implies P(a|c) = P(b|c)$.

$P(a|b, c) = P(b|a, c) \implies \frac{P(a,b,c)}{P(b,c)} = \frac{P(a,b,c)}{P(a,c)} \implies P(a, c) = P(b, c) \implies P(a|c)P(c) = P(b|c)P(c) \implies P(a|c) = P(b|c)$; **QED.**

b)

Proposition: $P(a|b, c) = P(a) \implies P(b, c) = P(b)$.

$P(a|b, c) = P(a) \iff \frac{P(a,b,c)}{P(b,c)} = P(a) \iff P(b, c) = \frac{P(a,b,c)}{P(a)} = P(b, c|a) \iff P(c|b)P(b) = P(b, c|a) = \frac{P(a|b,c)P(b,c)}{P(a)} \iff P(c|b)P(b) = P(b, c)$

$\therefore P(a|b, c) = P(a) \implies P(b, c) = P(b) \iff P(c|b) = 1$ i.e. c must be guaranteed to occur when b occurs. Hence the proposition is false.

c)

Proposition: $P(a|b) = P(a) \implies P(a|b, c) = P(a|c)$.

$P(a|b, c) = \frac{P(a,b,c)}{P(b,c)} = \frac{P(a,c)P(b|c)}{P(b|c)P(c)} = P(a|c)$; **QED.**

The statement $P(a, b, c) = P(a, c)P(b|c)$ follows from:

$P(a, b, c) = P(a, c)P(b|a, c)$. Since $P(a|b) = P(a)$, a and b are independent. Hence $P(b|a, c) = P(b|c)$. So, $P(a, b, c) = P(a, c)P(b|c)$.

Exercise 3

As a proxy for how indicative the tests are I will say that the test with the higher probability of a patient having the disease given that the test returns positive is the more indicative of the two. To that end I will use Bayesian inference, and define D as the event that a patient is sick.

We are given that, For A:

$$P(T = true|D = true) = 0.95, \ P(T = true|D = false) = 0.1, \ P(D = true) = 0.01$$

Using this information we find:

$$P(D = true|T = true) = \frac{P(T = true|D = true)P(D = true)}{P(T = true)} = \frac{0.95 \cdot 0.01}{\sum_D P(T = true, D)}$$

Notice that:

$$\sum_D P(T = true, D) = \sum_D P(T = true|D)P(D) = 0.95 \cdot 0.01 + 0.1 \cdot 0.99 = 0.1085.$$

Hence:

$$P(D = true|T = true) = \frac{0.95 \cdot 0.01}{0.1085} = 0.08756 = 8.756\%$$

Following a very similar procedure for B, we are given that:

$$P(T = true | D = true) = 0.9, \ P(T = true | D = false) = 0.05, \ P(D = true) = 0.01$$

Using this information we find:

$$P(D = true | T = true) = \frac{P(T = true | D = true) P(D = true)}{P(T = true)} = \frac{0.9 \cdot 0.01}{\sum_D P(T = true, D)}$$

Notice that:

$$\sum_D P(T = true, D) = \sum_D P(T = true | D) P(D) = 0.9 \cdot 0.01 + 0.05 \cdot 0.99 = 0.0585.$$

Hence:

$$P(D = true | T = true) = \frac{0.9 \cdot 0.01}{0.0585} = 0.15385 = 15.385\%$$

We conclude that test B is more indicative of the patient having the virus. This is due to the probabilistic reasoning above yielding almost twice the likelihood of a correct test result given that the disease is being carried when using B versus A.

Exercise 4

a)

Let T represent what color was testified under oath, and let C be the true color of the taxi. We look to find a way to compute the most likely color. We can do this by taking max $\{P(C = green | T = green), 1 - P(C = green | T = green)\}$. Using Bayes' theorem we can say:

$$P(C = blue | T = blue) = \frac{P(T = blue | C = blue) P(C = blue)}{P(T = blue)}$$

However, we have no information on the distribution of taxis in Athens; namely $P(C)$. So, we cannot find this probability!

b)

Now we have $P(C = green) = 0.9$ and implicitly $P(C = blue) = 0.1$. Returning back to the above we find:

$$P(C = blue | T = blue) = \frac{P(T = blue | C = blue) P(C = blue)}{P(T = blue)} = \frac{0.75 \cdot 0.1}{\sum_C P(T = blue, C)}$$

We know:

$$P(T = blue) = \sum_C P(T = blue, C) = \sum_C P(T = blue | C) P(C) = 0.75 \cdot 0.1 + 0.25 \cdot 0.9 = 0.3$$

So:

$$P(C = green | T = green) = \frac{0.75 \cdot 0.1}{0.3} = 0.25 = 25\%$$

It goes without saying then that max $\{P(C = green | T = green), 1 - P(C = green | T = green)\} = P(C = green | T = green) = 0.75$. Therefore, given that we testify the taxi is blue, there is a 25% chance the taxi is in fact green, and a 25% it is blue, mainly by virtue of the rarity of blue taxis.

Exercise 5

We have:

$p(L, G, V)$	L	G	V
0.87318	F	F	F
0.0072	F	F	T
0.049	F	T	F
0.0004	F	T	T
0.00882	T	F	F
0.0108	T	F	T
0.049	T	T	F
0.0016	T	T	T

a)

$P(L) = \sum_{G,V} P(L, G, V) = 0.00882 + 0.0108 + 0.049 + 0.0016 = 0.07022$

$P(L)$	L
0.0696	T
0.9304	F

b)

$P(G) = \sum_{L,V} P(L, G, V) = 0.049 + 0.0004 + 0.049 + 0.0016 = 0.1$

$P(G)$	G
0.1	T
0.9	F

c)

$P(L = T | G = T) = \frac{P(L=T,G=T)}{P(G=T)} = \frac{0.049+0.0016}{0.1} = 0.5$

$P(L = F | G = T) = \frac{P(L=F,G=T)}{P(G=T)} = \frac{0.049+0.0004}{0.1} = 0.5$

$P(L = T | G = F) = \frac{P(L=T,G=F)}{P(G=F)} = \frac{0.0082+0.0108}{0.9} = 0.0215556$

$P(L = F | G = F) = \frac{P(L=F,G=F)}{P(G=F)} = \frac{0.87318+0.0072}{0.9} = 0.9784444$

$P(L|G)$	L	G
0.9784	F	F
0.494	F	T
0.0215556	T	F
0.506	T	T

d)

$P(G = T | L = T \vee V = T) = \frac{P(G=T,L=T\vee V=T)}{P(L=T\vee V=T)} = \frac{0.0004+0.049+0.0016}{0.0072+0.0004+0.00882+0.0108+0.049+0.0016} = 0.6554$

e)

Let us translate the problem statement. We do not have any information on V, hence we treat it as we would if it hadn't been mentioned. $P(L = F | G = T) = 0.0215556$

Exercise 6

a)

Every hypothesis makes a statement about the probability of drawing a certain flavor. In a sense we can see this as a series of Bernoulli trials modeled via a binomial distribution. As we gather more and more data (i.e. draw more candy from the bag) the probability of success, whichever way we may define it, resembles the actual sample more and more for one of the hypotheses. Hence we become more confident in our guesses. From the text we have the formula:

$$P(\mathbf{d}|h_i) = \prod_j P(d_j|h_i)$$

The reason why certain posterior probabilities begin at 0.2, others at 0.1, and one at 0.4 is defined in the problem. When we haven't observed any candy we have no ability to infer beyond the probabilities of each bag type, which is what is modeled at $n = 0$.

b)

The most relevant code snippets are below:

```python
if __name__ == "__main__":
    main(
        bags={
            "P(h_1|d)": {"dist": [1, 0.0], "prob": 0.1},
            "P(h_2|d)": {"dist": [0.75, 0.25], "prob": 0.2},
            "P(h_3|d)": {"dist": [0.50, 0.50], "prob": 0.4},
            "P(h_4|d)": {"dist": [0.25, 0.75], "prob": 0.2},
            "P(h_5|d)": {"dist": [0.0, 1], "prob": 0.1}
        },
        n=15,
        trueBag="P(h_5|d)"
    )

    # Initialize a dictionary to keep track of the respective calculated
    # probabilities at each observation
    probs = {key: [bags.get(key).get("prob")] for key in bags}

    # True bag type
    trueBag = bags.get(trueBag)

    # Keep track of observations
    observations = []

    # Number of observations (x-axis)
    x = np.arange(0, n)

    for i in x:
        # Take a candy from trueBag and add it to the list of observations
        data = take(trueBag)
        observations.append(data)

        for key, hypothesis in bags.items():
            # Calculate the probability of each hypothesis and add it to the
            # dictionary with its corresponding array
            probs.get(key).append(postCalc(prob=hypothesis.get("prob"),
                                            dist=hypothesis.get("dist"),
```

```
                                            observation=data,
                                            observations=observations,
                                            bags=bags))

    for key, probabilities in probs.items():
        plt.plot(np.arange(0, n+1), probabilities, label=rf'${key}$',
                marker='x', linestyle='--')


# Finds p(h_i|d)
def postCalc(prob, dist, observation, observations, bags):
    # Calculating P(d|h_i)
    d_given_h = 1

    # Take product
    for j in observations:
        d_given_h *= dist[j]

    # Multiply by the probability of the hypothesis
    h_given_d = d_given_h * prob

    # Find proportionality constant
    alpha = 0
    for h in bags.values():
        likelihood = 1
        for j in observations:
            likelihood *= h["dist"][j]
        alpha += likelihood * h["prob"]

    return h_given_d / alpha

def take(bag):
    # Select a random index, representing a candy, with given probability
    indices = np.arange(len(bag.get("dist")))

    return np.random.choice(indices, p=bag.get("dist"))
```

Parametrizing the problem   As seen in the call to the `main` function I represented the bags as a dictionary of dictionaries in the context of Python. That is each bag is a dictionary which holds: a probability distribution in the form of an array, and a probability of each bag being the bag we are choosing from. Though it has little bearing on the outcome, I parametrized the number of samples taken from the bag as `n` and specified the bag we are truly drawing from as `trueBag`.

Finding probabilities   Essentially, the simulation (if you will) works by drawing a candy from the bag. The useful `numpy` library makes quick work of this using their random choice methods which allows me to specify the probability of each index being selected. Here the indices represent the candies, as the candies themselves are really just treated as random selection based on the specified probabilities. Once a selection has been made we estimate the probability $P(h_i|d)$ where each hypothesis corresponds precisely to each respective bag. This is done as given in the textbook, where α is calculated to ensure probabilities sum to 1. The function `postCalc` takes care of this arithmetic.

Plotting   Finally, I used `matplotlib` to easily plot the given probabilities of each hypothesis at discrete steps, i.e. every time we pull a candy from the bag.

6
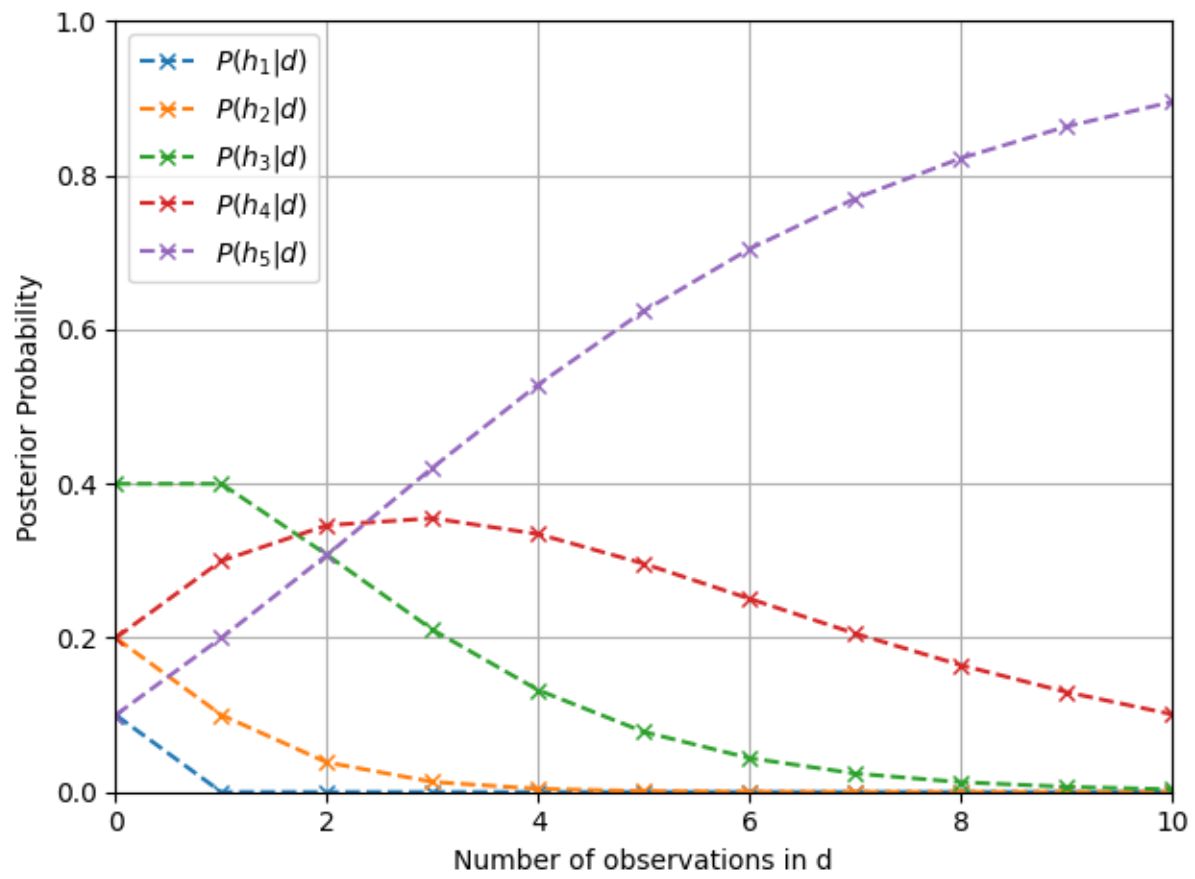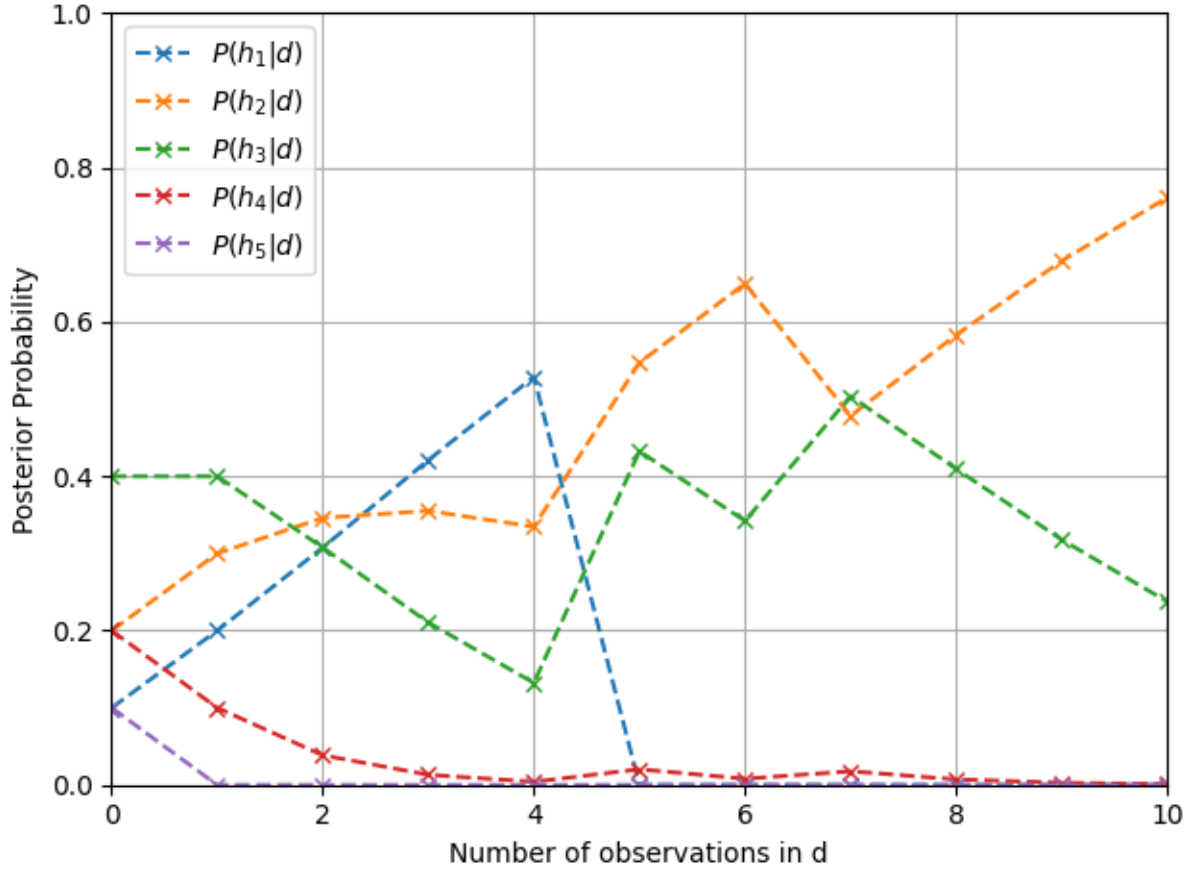
c)

Figure 1: An h_5 Plot

Clearly we are considerably more confident in the h_5 scenario at a much earlier stage. After the 3rd candy is sampled, we are always most confident in the h_5 hypothesis. In contrast, at the 3rd sample during the h_2 plot we are actually more confident in the h_1 hypothesis. Eventually $P(h_2|d)$ starts to dominate, but it is much more erratic and considerably less clear. Just by inspection we see $P(h_5|d) > 0.8$ for its appropriate plot yet $P(h_2|d) < 0.8$ for its plot.

d)

We need:

$$P(h_i|\mathbf{d}) = \alpha P(\mathbf{d}|h_i)P(h_i) > 0.9$$

After substituting the expressions for the given problem we find:

$$\alpha P(h_i) \prod_j P(d_j|h_i) > 0.9 \implies \prod_j P(d_j|h_i) > \frac{0.9}{\alpha P(h_i)}$$

So we need to find j such that the product of the probabilities of each observation assuming the hypothesis h_i. I believe this is the best we can do without a specific example, or a given sequence of pulls from the bag. Namely, if we have an h_3 bag and pull all limes for 100 trials vs. perfectly alternating lime and cherry. Though the former is *extremely* unlikely, this would result in very different values for j (one much much higher) than the latter case.