CS109 assignment 2

andreea.a.musat@gmail.com

January 2019

Warmup

1. Say in Silicon Valley, 36% of engineers program in Java and 24% of the engineers who program in Java also program in C++. Furthermore, 33% of engineers program in C++.
a. What is the probability that a randomly selected engineer programs in Java and C++?
b. What is the conditional probability that a randomly selected engineer programs in Java given that he/she programs in C++?

Solution

a. $P(J,C) = P(J) * P(C|J) = 0.36 * 0.24 = 0.0864$

b. $P(J|C) = \frac{P(C|J)P(J)}{P(C)} = \frac{0.24*0.36}{0.33} = 0.2618$

2. Two cards are randomly chosen without replacement from an ordinary deck of 52 cards. Let E be the event that both cards are Aces. Let F be the event that the first card chosen is the Ace of Spades. Compute $P(E|F)$.

Solution

Intuitively, we'd say that if we already know that the first card is Ace of Spades, there are 51 cards left to choose from and 3 of those would make event E happen, so the probability is $\frac{3}{51}$. We can check this result if we apply Bayes' rule: $P(E|F) = \frac{P(F|E)P(E)}{P(F)}$. We know that $P(F) = \frac{1}{52}$, as there is exactly one Ace of Spades in the deck. Also, $P(E) = P(\text{first card is Ace})P(\text{second card is Ace}) = \frac{4}{52}\frac{3}{51}$. Given that we have a pair of Aces, the probability that first one in the pair is the Ace of Spades is $\frac{1}{4} = P(F|E)$. Replacing the values in Bayes' rule above, we get the same result.

3. Five servers are located in a computer cluster. After one year, each server independently is still working with probability p, and otherwise fails (with probability 1 − p).
a. What is the probability that at least 1 server is still working after one year?
b. What is the probability that exactly 3 servers are still working after one year?
c. What is the probability that at least 3 servers are still working after one year?

Solution

a. $P(\#\text{working clusters} \geq 1) = 1 - P(\#\text{working clusters} = 0) = 1 - (1-p)^5$.
b. $P(\#\text{working clusters} = 3) = \binom{5}{3}p^3(1-p)^2 = 10p^3(1-p)^2$

c.

$$P(\#\text{working clusters} >= 3) = P(\#\text{working clusters} = 3) + P(\#\text{working clusters} = 4) + P(\#\text{working clusters} = 5) = \binom{5}{3}p^3(1-p)^2 + \binom{5}{4}p^4(1-p)^1 + \binom{5}{5}p^5(1-p)^0$$

4. A website wants to detect if a visitor is a robot. They give the visitor

three CAPTCHA tests that are hard for robots, but easy for humans. If the visitor fails in one of the tests, they are flagged as a robot. The probability that a human succeeds at a single test is 0.95, while a robot only succeeds with probability 0.3. Assume all tests are independent.
a. If a robot visits the website, what is the probability they get flagged?
b. If a visitor is human, what is the probability they get flagged?
c. The fraction of visitors on the site that are robots is 1/10. Suppose a visitor gets flagged. What is the probability that visitor is a robot?

Solution

a. $P(\text{R is flagged}) = 1 - P(\text{R gets all 3 tests correct}) = 1 - 0.3^3 = 0.973.$

b. $P(\text{H is flagged}) = 1 - P(\text{H gets all 3 tests correct}) = 1 - 0.95^3 = 0.1426.$

c. $P(\text{V=R}|\text{V is flagged}) = \frac{P(\text{V is flagged}|\text{V=R})P(\text{V=R})}{P(\text{V is flagged})} = \frac{P(\text{R is flagged})P(\text{V=R})}{P(\text{R is flagged})P(\text{V=R})+P(\text{H is flagged})P(\text{V=H})} = \frac{0.973*0.1}{0.973*0.1+0.1426*0.9} = 0.4312$

5. Recall the game set-up in the "St. Petersburg's paradox" discussed in class: there is a fair coin which comes up "heads" with probability p = 0.5. The coin is flipped repeatedly until the first "tails" appears. Let N = number of coin flips before the first "tails" appears (i.e., N = the number of consecutive "heads" that appear). Given that no one really has infinite money to offer as payoff for the game, consider a variant of the game where you win $\text{MIN}(2^N, X)$, where X is the maximum amount that the game provider will pay you after playing. Compute the expected payoff of the game for the following values of X. Show your work.
a. X = \$5.
b. X = \$500.
c. X = \$4096.

Solution

In the most general way, we can write:

$$\mathbb{E}[\text{payoff}] = \sum_{i=0}^{i=\infty} (\frac{1}{2})^i \frac{1}{2} \min(2^i, X) = \sum_{i=0}^{i=\lfloor \log_2 X \rfloor} (\frac{1}{2^{i+1}})2^i + \sum_{i>\lfloor \log_2 X \rfloor} \frac{1}{2^{i+1}} X =$$
$$\frac{1}{2}(1 + \lfloor \log_2 X \rfloor) + \frac{1}{2^{\lfloor \log_2 X \rfloor + 1}} X \sum_{i>0} \frac{1}{2^i} = \frac{1}{2}(1 + \lfloor \log_2 X \rfloor) + \frac{X}{2^{\lfloor \log_2 X \rfloor}}$$

a. Using X=\$5 in the equation above, we get: $\mathbb{E}[\text{payoff}] = \frac{3}{2} + \frac{5}{4} = 2.75.$

b. $\mathbb{E}[\text{payoff}|X = \$500] = \frac{9}{2} + \frac{500}{256} = 6.45.$

c. $\mathbb{E}[\text{payoff}|X = \$4096] = \frac{13}{2} + \frac{4096}{4096} = 7.5.$

6. A bit string of length n is generated randomly such that each bit is generated independently with probability p that the bit is a 1 (and 0 otherwise). How large does n need to be (in terms of p) so that the probability that there is at least one 1 in the string is at least 0.7?

Solution

$P(\#_1(string) \geq 1) = 1 - P(\#_1(string) = 0) = 1 - (1-p)^n$. Thus, $P(\#_1(string) \geq 1) \geq 0.7$ is equivalent to: $1 - (1-p)^n \geq 0.7 \iff 0.3 \geq (1-p)^n \iff \log_{1-p} 0.3 \leq n$

7. The probability that a Netflix user likes a movie Mi from the "Tearjerker" genre, given that they like the Tearjerker genre, is p_i. The probability that a user likes M_i given that they do not like the Tearjerker genre, is q_i. Of all Netflix users, 60% like the Tearjerker genre. Assume that, conditioned on knowing a user's preference for the genre (either liking the genre or not liking it), liking movie M_1 , M_2 and M_3 are independent events. Express all your answers in terms of qs and ps. What is the probability:
a. That a user likes all three movies M_1 , M_2 and M_3 given that they like the Tearjerker genre?
b. That they like at least one movie M_1 , M_2 and M_3 given that they like the Tearjerker genre?
c. That they like the Tearjerker genre given that they like M_1 , M_2 and M_3?

Solution

TBD

8. Suppose we want to write an algorithm fairRandom for randomly generating a 0 or a 1 with equal probability (= 0.5). Unfortunately, all we have available to us is a function: **int unknownRandom()** that randomly generates bits, where on each call a 1 is returned with some unknown probability p that need not be equal to 0.5 (and a 0 is returned with probability 1 − p). Consider the following algorithm for **fairRandom**:

```
def fairRandom():
    while True:
        r1 = unknownRandom()
        r2 = unknownRandom()
        if (r1 != r2): break
    return r2;
```

a. Show mathematically that **fairRandom** does indeed return a 0 or a 1 with equal probability.

Solution
The event that **fairRandom** returns 1 happens when the while loop terminates and the last r_2 is 1: That is,

$$P(\textbf{fairRandom()} \text{ returns } 1) = P(\text{while loop terminates} \wedge r_2^{(\text{last})} = 1).$$

The while loop terminates if it ends at the first iteration or at the second iteration or so on. We denote by $St(i)$ the event that the while loop terminates at iteration i. We can write:

$$P(\text{while loop terminates}) = P(\vee_{i=1}^{\infty} St(i)).$$

Now, the loop terminates at iteration i iff for the first i - 1 iterations we generate the same r_1 and r_2 and then at the i^{th} iteration we generate different numbers. Let $Eq(i)$ be the event that r_1 generated at timestep i is equal to r_2 generated at timestep i. Then,

$$P(St(i)) = P((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i)).$$

Using these, we re-write $P(\textbf{fairRandom()} \text{ returns } 1)$:

$$\begin{gathered}
P(\textbf{fairRandom()} \text{ returns } 1) = P(\text{while loop terminates} \wedge r_2^{(\text{last})} = 1) = \\
P(\text{while loop terminates} | r_2^{(\text{last})} = 1) P(r_2^{(\text{last})} = 1) = \\
P(\vee_{i=1}^{\infty} St(i) | r_2^{(\text{last})} = 1) P(r_2^{(\text{last})} = 1) = \\
P(r_2^{(\text{last})} = 1) \sum_{i=1}^{\infty} P(St(i) | r_2^{(i)} = 1) = \\
P(r_2^{(\text{last})} = 1) \sum_{i=1}^{\infty} P\big(\big(\wedge_{k=1}^{i-1} Eq(k)\big) \wedge \neg Eq(i) | r_2^{(i)} = 1 \big) = \\
P(r_2^{(\text{last})} = 1) \sum_{i=1}^{\infty} P(\neg Eq(i) | r_2^{(i)} = 1) \prod_{k=1}^{i-1} P(Eq(k) | r_2^{(i)} = 1)
\end{gathered}$$

where we used the fact that the events $St(i)$ are independent for all i (and also the events $Eq(i)$ are independent). We know that $P(r_2 = 1) = p$ and that:

$$\begin{gathered}
P(Eq(k)) = P(r_1^{(k)} = r_2^{(k)}) = P(r_1^{(k)} = 0 \wedge r_2^{(k)} = 0) + P(r_1^{(k)} = 1 \wedge r_2^{(k)} = 1) = \\
(1-p)^2 + p^2,
\end{gathered}$$

and obviously:

$$P(\neg Eq(i) | r_2^{(i)} = 1) = P(r_1^{(i)} = 0) = 1 - p$$

We also note that $P(\neg Eq(k)) = 2p(1-p)$. Also, $Eq(k)$ and $r_2^{(i)} = 1$ are independent $\forall k < i$, so replacing these into the above formula we obtain:

$$\begin{gathered}
P(\textbf{fairRandom()} \text{ returns } 1) = p \sum_{i=1}^{\infty} \Big((1-p) \prod_{k=1}^{i-1} \big((1-p)^2 + p^2 \big) \Big) = \\
p \sum_{i=1}^{\infty} \Big((1-p) \big((1-p)^2 + p^2 \big)^{i-1} \Big) = p(1-p) \sum_{i=1}^{\infty} \big((1-p)^2 + p^2 \big)^{i-1} = \\
p(1-p) \frac{1}{1 - \big((1-p)^2 + p^2 \big)} = \frac{p(1-p)}{2p(1-p)} = \frac{1}{2}
\end{gathered}$$

Similarily, we could also show that $P(\textbf{fairRandom()} \text{ returns } 0) = \frac{1}{2}$ (which is an intuitive result, as the while loop always terminates).

b. Say we want to simplify the function, so we write the **simpleRandom** function below. Would the **simpleRandom** function also generate 0's and 1's with equal probability? Explain why or why not. Determine P(**simpleRandom** returns 1) in terms of p.

```
int simpleRandom() {
    r1 = unknownRandom()
    while True:
        r2 = unknownRandom()
        if (r1 != r2): break
        r1 = r2
    return r2
```

Solution

The function returns a 1 if the while loop terminates and r_2 is 1. Looking at the function, we observe that the loop breaks only when r_2 is different from r_1 and r_1 is only generated once, at the beginning. This means that the loop breaks when we generate the first number different from r_1, so we can write:

$$P(\textbf{simpleRandom()} \text{ generates } 1) = P(\text{while loop terminates} \wedge r_2^{(last)} = 1)$$

Denoting by $St(i)$ the event that the loop terminates at iteration i and by $Eq(i)$ the event that r_1 is equal to r_2 generated at timestep i ($r_2^{(i)}$), we can continue:

$$P(\textbf{simpleRandom()} \text{ generates } 1) = P\Big(\vee_{i=1}^{\infty} \big(St(i) \wedge r_2^{(i)} = 1\big)\Big) =$$
$$P\Big(\vee_{i=1}^{\infty} \big((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)\Big)$$

We use the law of total probability with the partitions $r_1 = 0$ and $r_1 = 1$ and we get:

$$P(\textbf{simpleRandom()} \text{ generates } 1) =$$
$$P\Big(\vee_{i=1}^{\infty} \big((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)|r_1 = 0\Big)P(r_1 = 0)+$$
$$P\Big(\vee_{i=1}^{\infty} \big((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)|r_1 = 1\Big)P(r_1 = 1) =$$
$$P\Big(\vee_{i=1}^{\infty} \big((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)|r_1 = 0\Big)P(r_1 = 0),$$

as the second term disappears because $\neg Eq(i) \wedge r_2^{(i)} = 1$ and $r_1 = 1$ are contradictory events (from the definition of $Eq(i)$).

Finally,

$$P(\textbf{simpleRandom()} \text{ generates } 1) =$$
$$P\Big(\vee_{i=1}^{\infty} \big((\wedge_{k=1}^{i-1} Eq(k)) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)|r_1 = 0\Big)P(r_1 = 0) =$$
$$P(r_1 = 0) \sum_{i=1}^{\infty} P\big((\wedge_{k=1}^{i-1} Eq(k) \wedge \neg Eq(i) \wedge r_2^{(i)} = 1\big)|r_1 = 0)$$

Given that $r_1 = 0$ and $r_2^{(i)} = 1$, it follows that $\neg Eq(i)$, so this event is redundant and we can remove it:

$$P(\textbf{simpleRandom()} \text{ generates } 1) =$$
$$P(r_1 = 0) \sum_{i=1}^{\infty} P\big((\wedge_{k=1}^{i-1} Eq(k) \wedge r_2^{(i)} = 1)\big| r_1 = 0) =$$
$$P(r_1 = 0) \sum_{i=1}^{\infty} \big(P(r_2^{(i)} = 1 | r_1 = 0) \prod_{k=1}^{i-1} P(Eq(k)|r_1 = 0)\big) =$$
$$(1-p) \sum_{i=1}^{\infty} \big(p \prod_{k=1}^{i-1} P(Eq(k)|r_1 = 0)\big) = p(1-p) \sum_{i=1}^{\infty} \prod_{k=1}^{i-1} P(r_2^{(k)} = 0) =$$
$$p(1-p) \sum_{i=1}^{\infty} (1-p)^{i-1} = p(1-p) \frac{1}{1-(1-p)} = \frac{p(1-p)}{p} = 1 - p$$

The intuitive explanation for this is that whenever we start with $r_1 = 0$, we return $r_2 = 1$ (as the loop only terminates when we encounter the first different number) and this happens with probability 1-p.