Differential privacy in practice (easy version)

2024-06-19

This document implements Damien Desfontaines blog, “A friendly, non-technical introduction to differential privacy’ ’. Here is the link.

First we load our top commands:

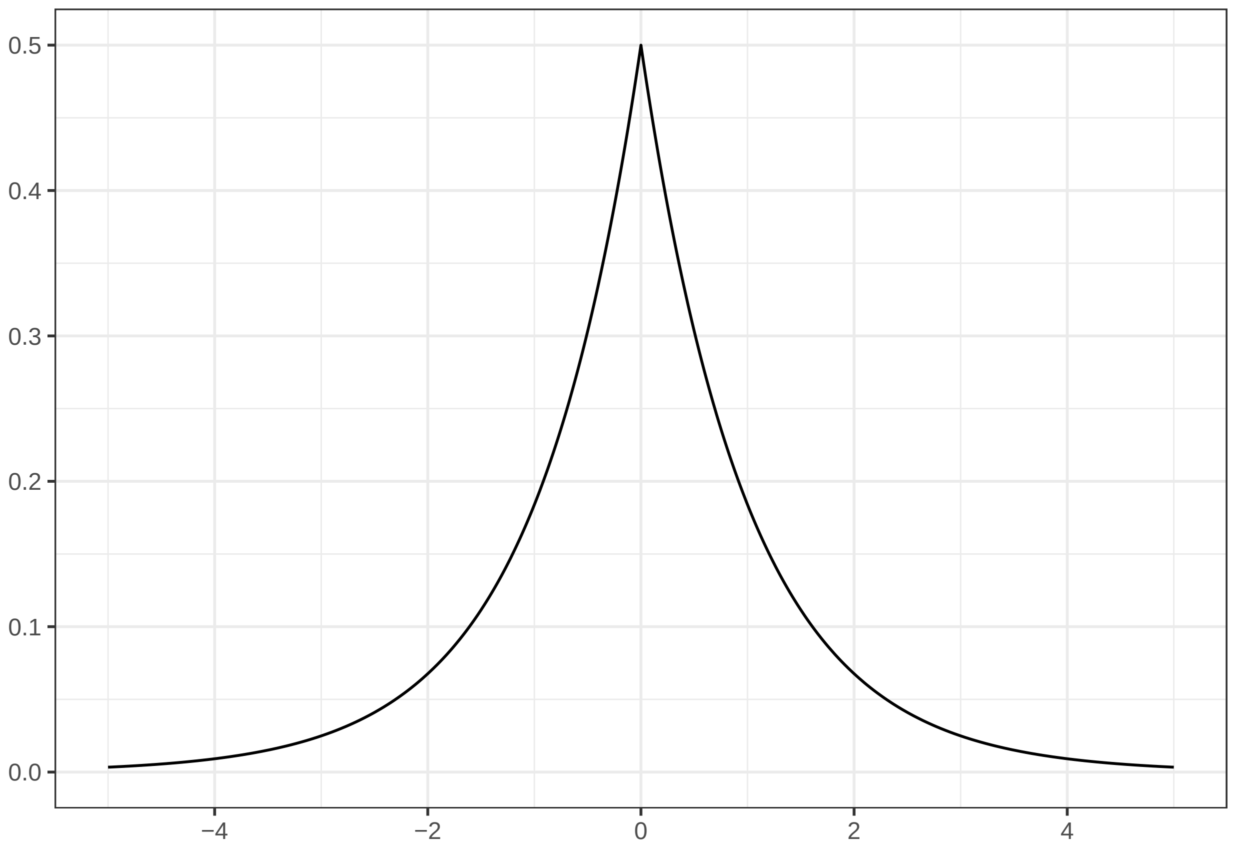
```r
# top commands
set.seed(123)

# load libraries
library(tidyverse)

# create functions
dlaplace <- function(x, mean, scale) {
  return(1/(2*scale) * exp(-abs(x - mean)/scale))
}
```

Counting unique users

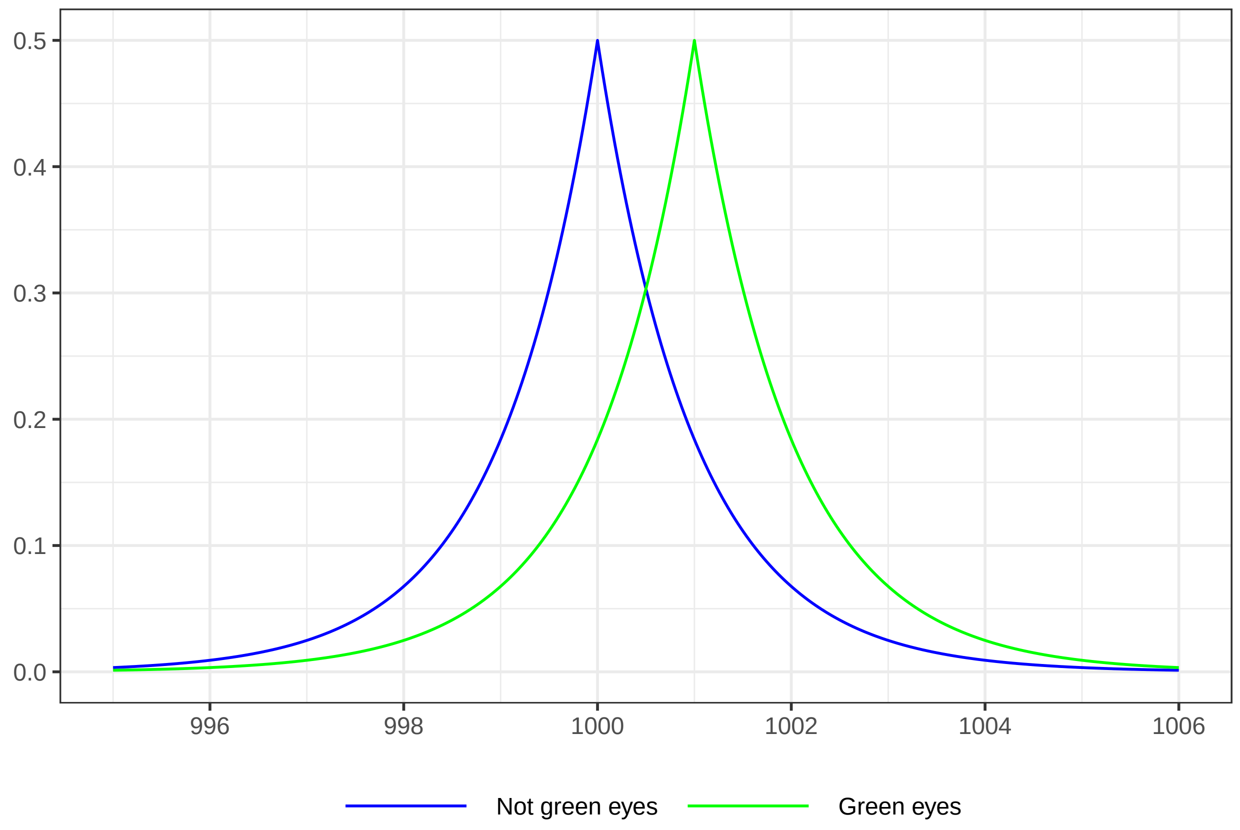
Suppose you have a database, and you want to publish how many people in there satisfy a given condition. Say, how many have green eyes? Even if you have many people in your database, you can’t just publish the true answer. Let’s take a moment to understand why.

With differential privacy, we assume that the attacker knows almost all elements. They only have uncertainty about their target. Say they want to know whether their target has green eyes. If you output the real number k, they can compare it with the number of people with green eyes among the people they know. If it’s $k - 1$, then the target has green eyes. If it’s k, then the target does not.

So, what do we do? We compute the exact answer, and we add noise. This noise will come from a probability distribution called the Laplace distribution. This distribution has a parameter, its scale, which determines how “flat” it is. It looks like this:



So, to get ϵ-differential privacy, we pick a random value according to Laplace$(1/\epsilon)$, and we add this noise to the real value. Why does it work? Let's look at the distribution of the number we return, depending on whether the true count is $k = 1000$ (blue line, the target doesn't have green eyes) or $k = 1001$ (green line, the target has green eyes).



Let's say the real number is $k = 1001$, and after adding noise, we published 1003. Let's put ourselves in the attacker's shoes. What's the likelihood that the original number was 1001 vs. 1000? The hypothesis "$k = 1001$' ' is a bit more likely: generating a noise of 2 is more likely than a noise of 3. How much more likely? It turns out that the ratio between these likelihoods is … e^{ϵ}! So the ratio of probabilities of differential privacy is satisfied. Here's how to quantify this:

Differential Privacy Likelihood Calculation

To determine the likelihood that the original number was 1000 versus 1001 from the perspective of an attacker, given that a noisy count of 1003 was published, we use the properties of differential privacy, particularly the noise distribution.

Given Information

- The noisy count $y = 1003$
- The possible true counts $x_0 = 1000$ and $x_1 = 1001$
- The noise follows a Laplace distribution with parameter $b = 1$

Likelihood Calculation

The probability density function of the Laplace distribution centered at μ with scale parameter b is:

$$f(x|\mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

Given the noisy count $y = 1003$:

$$P(y = 1003|x = 1000) = \frac{1}{2 \cdot 1} \exp\left(-\frac{|1003 - 1000|}{1}\right) = \frac{1}{2} \exp(-3) = \frac{1}{2e^3}$$

$$P(y = 1003|x = 1001) = \frac{1}{2 \cdot 1} \exp\left(-\frac{|1003 - 1001|}{1}\right) = \frac{1}{2} \exp(-2) = \frac{1}{2e^2}$$

Likelihood Ratio

To compare the likelihoods, we calculate the ratio of these probabilities:

$$\frac{P(y = 1003|x = 1001)}{P(y = 1003|x = 1000)} = \frac{\frac{1}{2e^2}}{\frac{1}{2e^3}} = \frac{1}{e^2} \cdot e^3 = e$$

Thus, the likelihood ratio is e, which is approximately 2.718.

Conclusion

Given the published count of 1003, it is approximately 2.718 times more likely that the true count was 1001 than 1000. We can also transform this into a function.

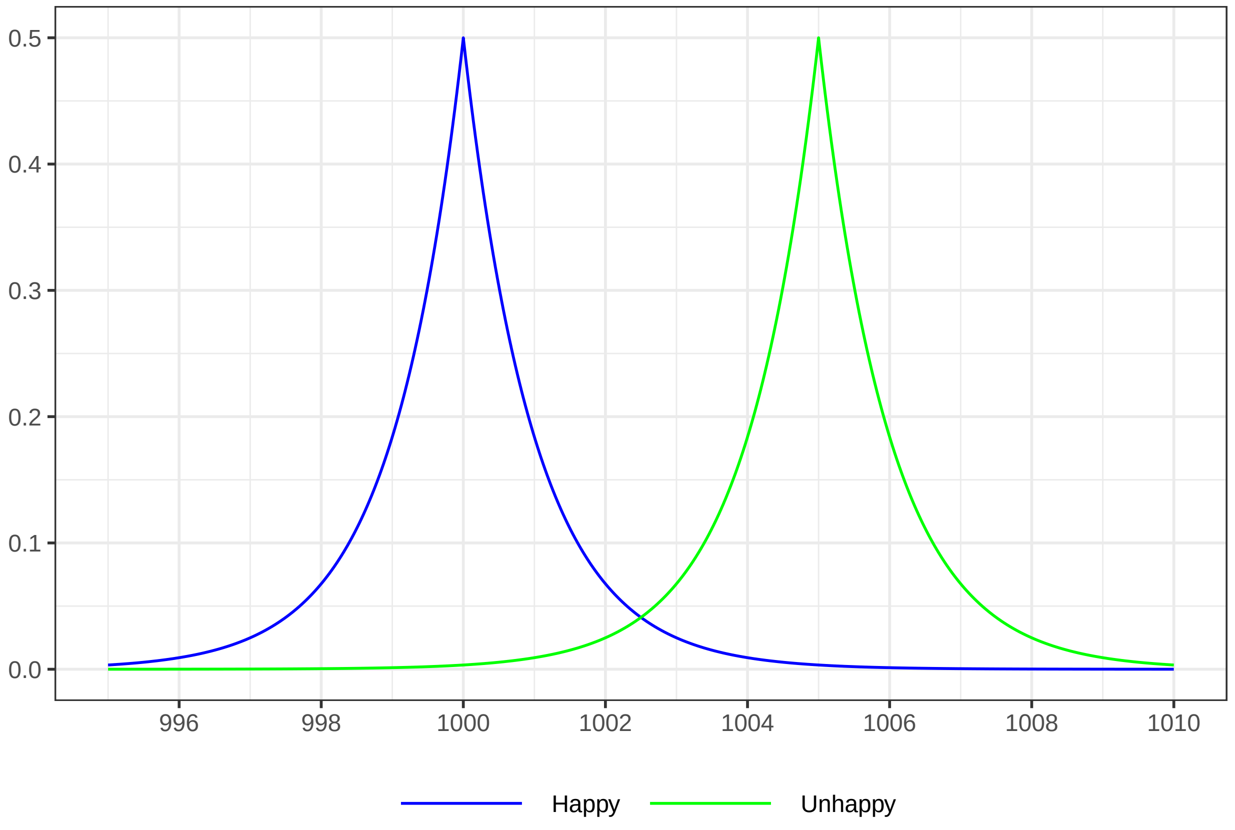
```r
# Define the function to calculate the likelihood ratio
calculate_likelihood_ratio <- function(y, x1, x2, b) {
  # Calculate the likelihood for x1
  likelihood_x1 <- (1 / (2 * b)) * exp(-abs(y - x1) / b)

  # Calculate the likelihood for x2
  likelihood_x2 <- (1 / (2 * b)) * exp(-abs(y - x2) / b)

  # Calculate the likelihood ratio
  likelihood_ratio <- likelihood_x2 / likelihood_x1

  return(likelihood_ratio)
}

# Example usage
y <- 1003 # published count
x1 <- 1003 # possible true count
x2 <- 1005 # possible true count
b <- 1

# Calculate the likelihood ratio
likelihood_ratio <- calculate_likelihood_ratio(y, x1, x2, b)
likelihood_ratio
```

```
## [1] 0.1353353
```

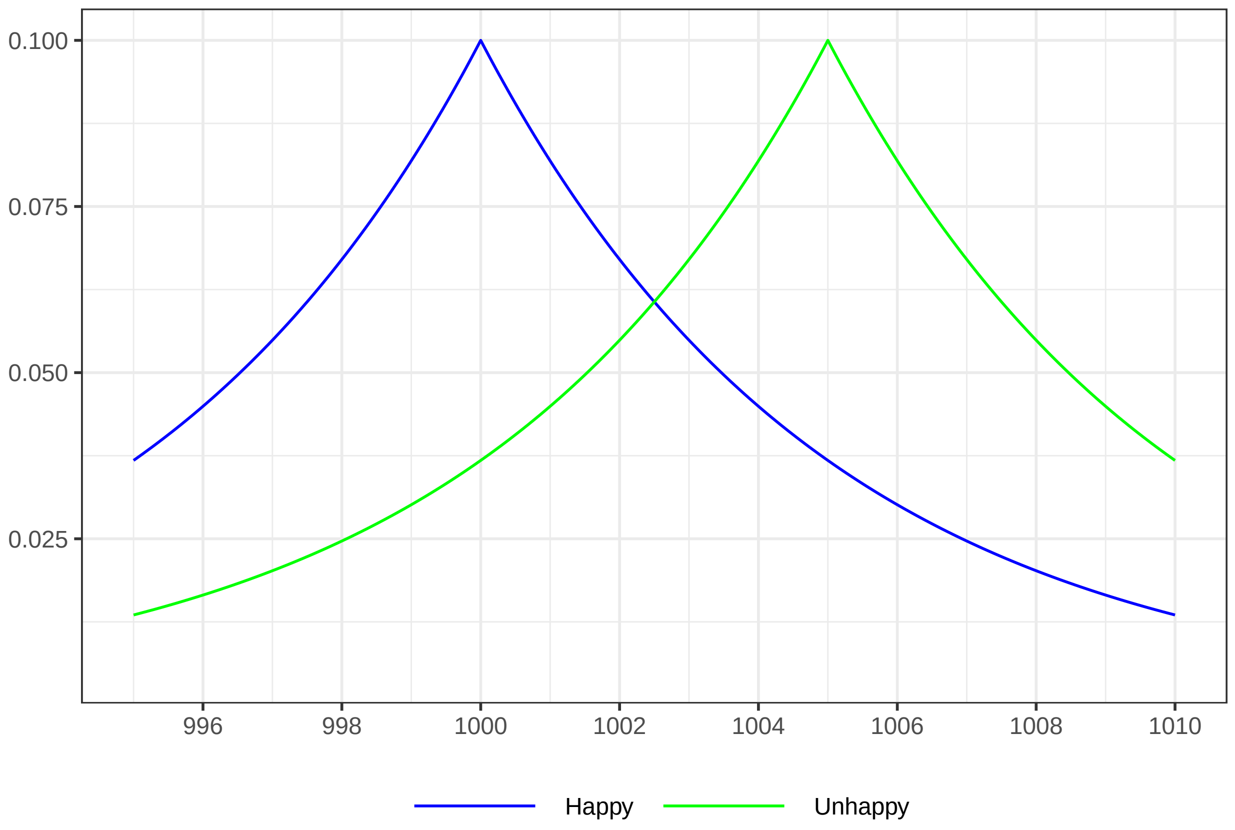
Counting things

OK, so counting unique users was pretty easy. Counting things must also be straightforward, right? Let's say you have a database of suggestions that people sent to your company using a feedback form. You want to publish the number of suggestions you received on a given day. Meanwhile, the attacker wants to get an idea of how many complaints their target published.

What's different about the previous scenario? Can't we just add noise picked from Laplace($1/\epsilon$) and get ϵ-differential privacy? There's a catch: what if someone sent more than one complaint during one day? Let's say someone was super unhappy and sent five complaints. The other 1000 customers sent one complaint each. The influence of this one disgruntled customer will be larger than before. The two distributions now look like this:



The difference between the curves is much larger than before. Their ratio is at most $e^{5\epsilon}$, so using a parameter of $1/\epsilon$ only gives 5ϵ-differential privacy. To fix this, we need to add more noise. How much more? It depends on the maximum contribution of one individual user. If the maximum amount of complaints in one day is 5, you must add 5 times the amount of noise. In this example, using Laplace$(5/\epsilon)$ would give you ϵ-differential privacy.



Note that you can't fully automate this process: you need to know what the largest contribution can be. A human, with some knowledge over the process, must make a judgment call. In our case, this could be "users won't post more than five complaints per day' '.

What happens if that judgment call is wrong, and a user later decides to post 10 complaints in one day? To preserve the desired level of privacy, you need to *clamp* all values to the estimated maximum. In other words, for this outlier user, you would only count 5 complaints in the non-noisy sum.

This process can introduce unexpected bias in the data. So, be careful when estimating the largest contribution! If clamping only happens very rarely, you should be fine.

Summing or averaging numbers

Let's say each of your users gives your customer service a rating, between -10 and 10. You want to release the average rating. Computing an average is pretty much the same as computing a sum — add all ratings, then divide by the number of users [1]. So, what do we do to the sum to achieve differential privacy?

[1] If you want to be extra pedantic, you might also want to add noise to your total number of users. That depends on the flavor of definition that you choose. I'm not going to that level of detail here, and you probably shouldn't either.