Introduction to Bayesian A/B testing

Johann de Boer

2022-10-22

Table of contents

1 Setting the scene

1.1 Randomised Control Trials (RCTs)

A simplistic example:

- Users are assigned at **random** to two groups, A and B, with equal probability.
- Let A be our **control** group and B be our **treatment** group.

We want to know what effect our treatment has.

> Tip
>
> Early on during an experiment, differences between these groups could simply be due to the random allocation of participants. As the groups get larger, these random differences will diminish, bringing us closer to the difference caused by the treatment. Applying Bayesian inference effectively gives the experiment a guided head start by including more data in the form of **priors**.

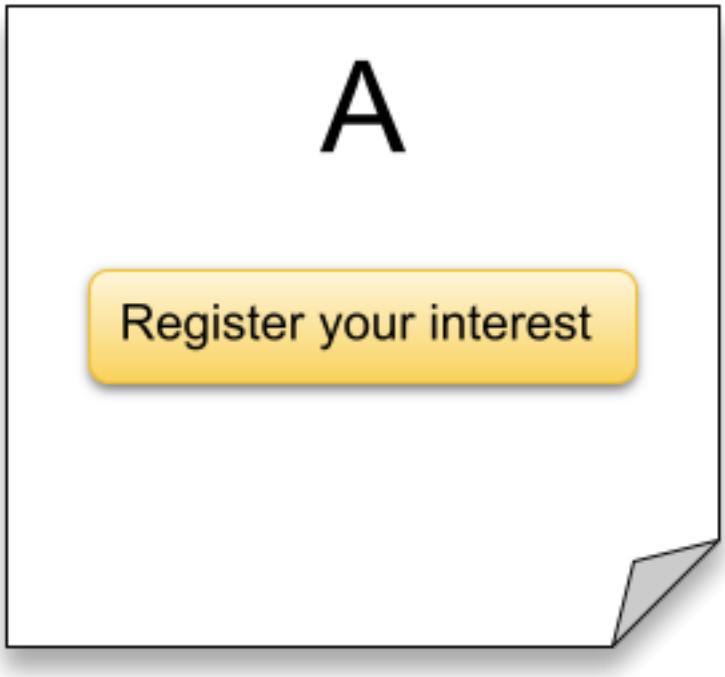
1.2 Hypothetical scenario

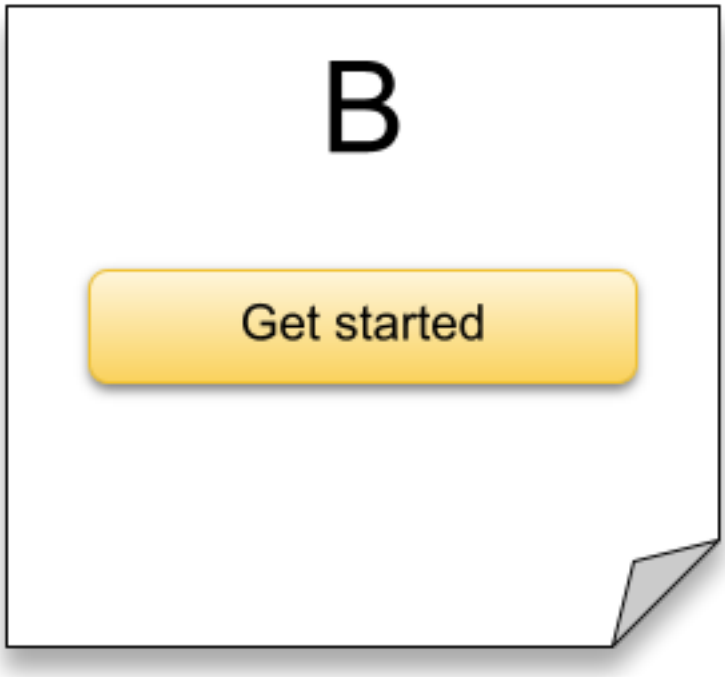
We have a button on a landing page that takes users to a sign up form.

At present, the button is labelled “Register your interest”.

We want to test whether changing it to “Get started” will result in an increased click-through rate (CTR).

The idea of “Get started” was suggested by an experienced and skilled UX design professional.


A
Register your interest


B
Get started

2 Priors and probability distributions

2.1 Prior knowledge and beliefs

Before running an experiment, we form opinions about what we expect to see. We gather evidence such as:

- The baseline click-through rate of the button (with its current label) and knowledge of any outside variables that affects click-through rate, e.g. seasonality
- Effects we have seen from similar previous experiments
- Qualitative research, such as usability tests, focus groups, and surveys that are related to the test
- Opinions (including critical views) from interested parties, including experts

2.2 Priors are probability distributions

We express our prior beliefs about the control group using a **probability distribution**.

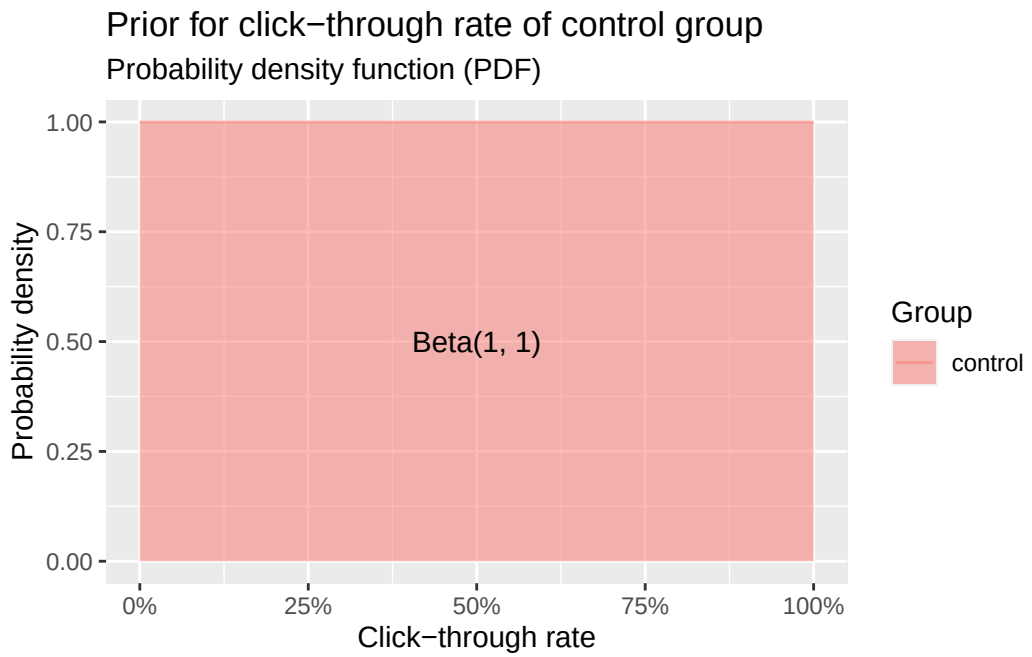
This plot shows an example of an extremely uninformative prior – a uniform prior that says every outcome is equally likely, i.e. naive.

```
plot_beta_pdf <- function(shape1, shape2, group) {
    p <- seq(0, 100, by = 0.1) / 100 # 0 to 100 percent
    df <- pmap_dfr(
        list(shape1, shape2, group),
        function(shape1, shape2, group) {
            tibble(
                p = p,
                d = dbeta(p, shape1, shape2),
                group = group
            )
        }
    )
    labels_df <- tibble(
        group = group,
        p = shape1 / (shape1 + shape2),
        d = dbeta(p, shape1, shape2) / 2,
        label = glue("Beta({shape1}, {shape2})")
    )
    ggplot(df) +
        aes(x = p, y = d, fill = group) +
        geom_area(alpha = 0.5, position = position_identity()) +
        geom_line(aes(colour = group), alpha = 0.5) +
```

```
        geom_text(aes(label = label), data = labels_df) +
        scale_x_continuous(labels = scales::percent) +
        labs(
            subtitle = "Probability density function (PDF)",
            x = "Click-through rate",
            y = "Probability density",
            colour = "Group",
            fill = "Group"
        )
}

experiment_groups <- c("control", "treatment")

plot_beta_pdf(1, 1, group = factor("control", levels = experiment_groups)) +
    labs(
        title = "Prior for click-through rate of control group"
    )
```



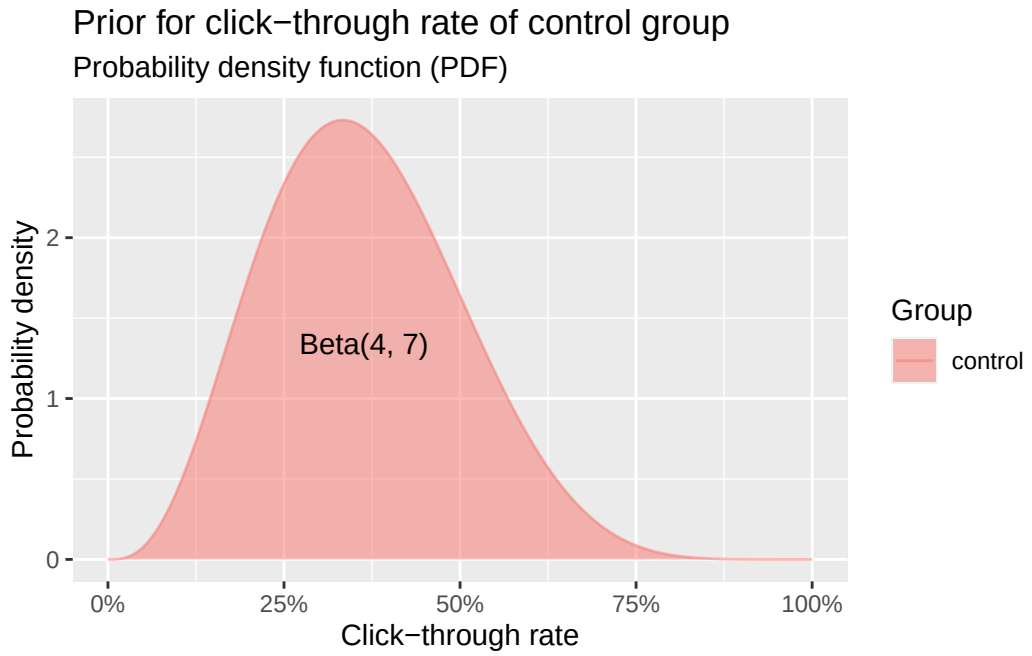
> **Tip**
>
> The **Beta distribution** is a **probability density function (PDF)** with two **shape parameters**: $B(shape1, shape2)$. It's used to describe proportions, such as click-through rate.

The total area under the curve will always add to 100%. That is, the curve represents all

possibilities regardless of what shape parameters are used.

2.3 Something a little more informative

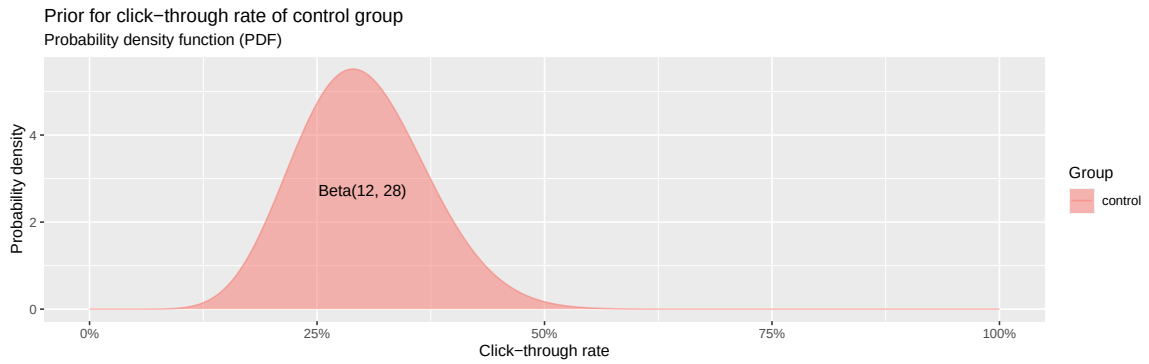
```
plot_beta_pdf(4, 7, group = factor("control", levels = experiment_groups)) +
    labs(
        title = "Prior for click-through rate of control group"
    )
```



As the curve narrows, notice that the shape parameters of the Beta distribution increase.

2.4 Something even more informative

```
plot_beta_pdf(12, 28, group = factor("control", levels = experiment_groups)) +
    labs(
        title = "Prior for click-through rate of control group"
    )
```



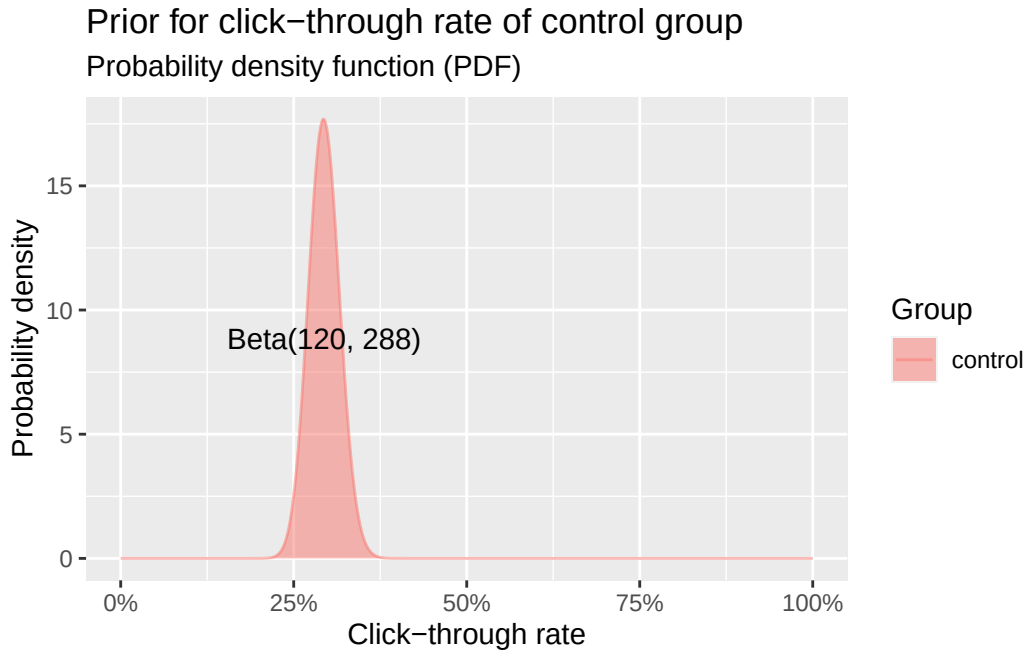
The shape parameters (`shape1` and `shape2`) of the Beta distribution can be considered counts of **successes** and **failures**, respectively. Therefore, the mean probability of success (i.e. average click-through rate) can be calculated by this formula:

$$\frac{shape1}{shape1 + shape2}$$

The shape parameters are actually slightly more than the count of successes and failures, i.e. $successes = \alpha - 1$ and $failures = \beta - 1$, or $successes = \alpha - 0.5$ and $failures = \beta - 0.5$ if using Jeffreys prior.

2.5 Let's say we've settled on this:

```
plot_beta_pdf(
    shape1 = 120, shape2 = 288,
    group = factor("control", levels = experiment_groups)
) +
    labs(
        title = "Prior for click-through rate of control group"
    )
```



The more confident we are about our beliefs, the narrower the curve.

The mean of this curve is: $\mu = \frac{\alpha}{\alpha+\beta}$

The standard deviation of this curve is: $\sigma = \sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}}$

Where α and β represent the first and second shape parameters of the Beta distribution, respectively.

Through substitution and rearrangement, you can determine α and β from μ and σ.

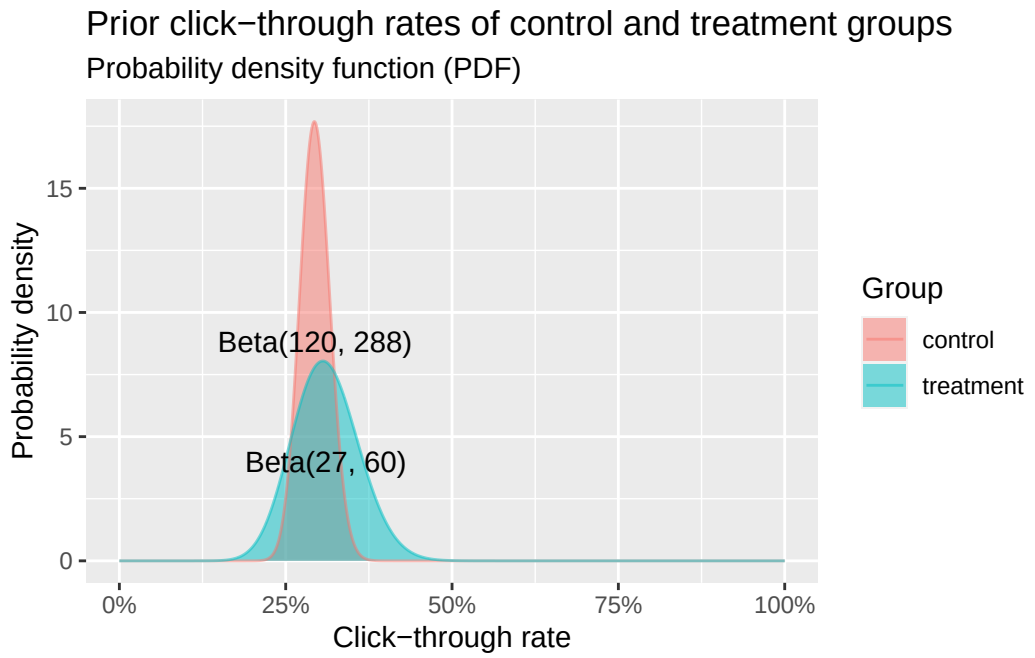
2.6 What about the treatment group?

- We expect that the click-through rates of the treatment and control groups will be correlated.
- We don't know how correlated they will be, but we're not expecting a dramatic difference between them.
- We hope that the click-through rate of the treatment group will be an improvement, but we're open to other possibilities.
- We're more confident than not that the click-through rate of the treatment group will be an improvement over control.
- We don't want to bias the results of the experiment in favour of treatment or control, or towards a conclusion of there being a difference or no difference.

2.7 We've settled on these priors:

```
priors <- tibble(
    group = factor(c("control", "treatment"), levels = experiment_groups),
    shape1 = c(120, 27),
    shape2 = c(288, 60)
)

do.call(plot_beta_pdf, priors) +
    labs(
        title = "Prior click-through rates of control and treatment groups"
    )
```



3 Running a Bayesian A/B test

3.1 Prior agreement

Agreement must be reached on the priors before collecting and analysing data from the experiment.

Once we've agreed on the priors and have locked them in, we can start the experiment.

Here's a summary of the priors we have chosen:

```r
priors %>%
    mutate(
        mean = (shape1 / (shape1 + shape2)) %>%
            scales::percent(),
        sd = sqrt(
            shape1 * shape2 /
                ((shape1 + shape2) ^ 2 * (shape1 + shape2 + 1))
        ) %>%
            scales::percent(suffix = " p.p.")
    )
```

group	shape1	shape2	mean	sd
control	120	288	29.4%	2.3 p.p.
treatment	27	60	31.0%	4.9 p.p.

It's important to not change your original priors after seeing data collected from the experiment. Doing so is effectively double-dipping, whereby your priors are being influenced by the data you have collected.

3.2 Let's run a simulated experiment

```r
true_ctr <- c(control = 32, treatment = 35) / 100
```

Let's pretend that there's some true theoretical click-through rate for the control and treatment groups, 32% and 35% respectively. That equates to a relative uplift of 9.4%.

> **Note**
>
> Remember that this is just a hypothetical simulation. We wouldn't know these in a real experiment ¯_( )_/¯.

If we're successful at applying Bayesian inference then we should hope to see (but can't guarantee due to randomness) results that somewhat match with these theoretical CTRs.

3.3 1 day since the experiment started...

```r
avg_daily_users <- 150

experiment_simulator <- function(rate_of_users, true_ctr) {
    experiment_groups <- names(true_ctr)
    n_users <- rpois(1, lambda = rate_of_users)
```

```
    group_assignment <- rbinom(
        n = n_users,
        size = length(experiment_groups) - 1,
        prob = 0.5
    ) %>%
        factor(labels = experiment_groups)
    group_sizes <- table(group_assignment)
    clicks_by_group <- map2(group_sizes, true_ctr, rbernoulli)
    map_dfr(clicks_by_group, function(clicks) {
        list(
            click_data = list(clicks),
            clicked = sum(clicks),
            not_clicked = sum(!clicks)
        )
    }, .id = "group") %>%
        mutate(group = factor(group, levels = experiment_groups))
}
```

Let's pretend that on average 150 users enter our experiment each day, and we've received the following data from day 1:

```
experiment_data <- experiment_simulator(avg_daily_users, true_ctr)

experiment_data %>%
    mutate(
        CTR = scales::percent(clicked / (clicked + not_clicked)),
        total_users = clicked + not_clicked
    ) %>%
    select(group, total_users, clicked, not_clicked, CTR)
```

group	total_users	clicked	not_clicked	CTR
control	69	24	45	34.8%
treatment	67	30	37	44.8%

Our experiment simulator randomly selects users and assigns them to each group using a Poisson process. It then randomly chooses which users had clicked using Bernoulli trials (e.g. coin flips).

3.4 Let's now incorporate our priors

For each experiment group, we derive our **posterior** shape parameters through simple arithmetic addition:

- Increment the first shape parameter by the count users who had **clicked**
- Increment the second shape parameter by the count users who **didn't click**

```
posterior_update <- function(priors, experiment_data) {
    experiment_data %>%
        left_join(priors, by = "group") %>%
        rename(prior_shape1 = shape1, prior_shape2 = shape2) %>%
        select(-click_data) %>%
        mutate(
            posterior_shape1 = prior_shape1 + clicked,
            posterior_shape2 = prior_shape2 + not_clicked
        )
}

posteriors <- priors %>% posterior_update(experiment_data)

posteriors_long <- posteriors %>%
    gather(key = "count", value = "value", -group, factor_key = TRUE) %>%
    spread(group, value) %>%
    mutate(count = count %>% fct_relevel(
        c(
            "prior_shape1", "clicked", "posterior_shape1",
            "prior_shape2", "not_clicked", "posterior_shape2"
        )
    )) %>%
    arrange(count)
```

```
posteriors_long %>%
    filter(count %in% c("prior_shape1", "clicked", "posterior_shape1"))
```

count	control	treatment
prior_shape1	120	27
clicked	24	30
posterior_shape1	144	57

```
posteriors_long %>%
    filter(count %in% c("prior_shape2", "not_clicked", "posterior_shape2"))
```

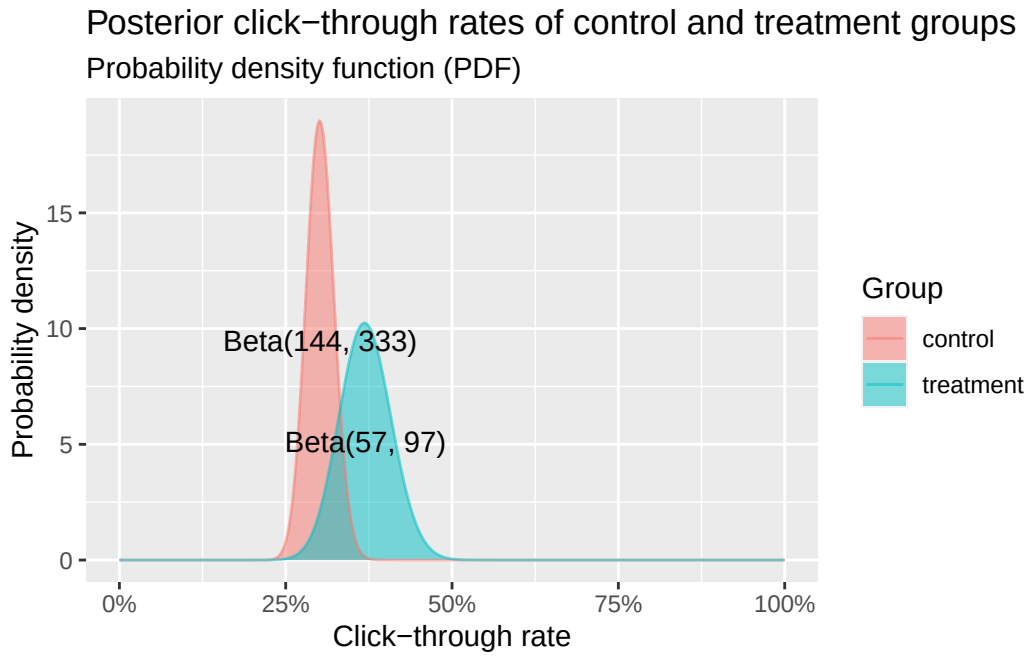
count	control	treatment
prior_shape2	288	60
not_clicked	45	37

count	control	treatment
posterior_shape2	333	97

The process of incorporating data with priors is called Bayesian updating. The data generated follows a Bernoulli distribution (Binomial with 1 trial). The prior follows a Beta distribution, which is conjugate to the Binomial distribution.

3.5 Posterior distribution of each group

```
do.call(
    plot_beta_pdf,
    posteriors %>% select(
        group = group,
        shape1 = posterior_shape1,
        shape2 = posterior_shape2
    )
) +
    labs(
        title = "Posterior click-through rates of control and treatment groups"
    )
```



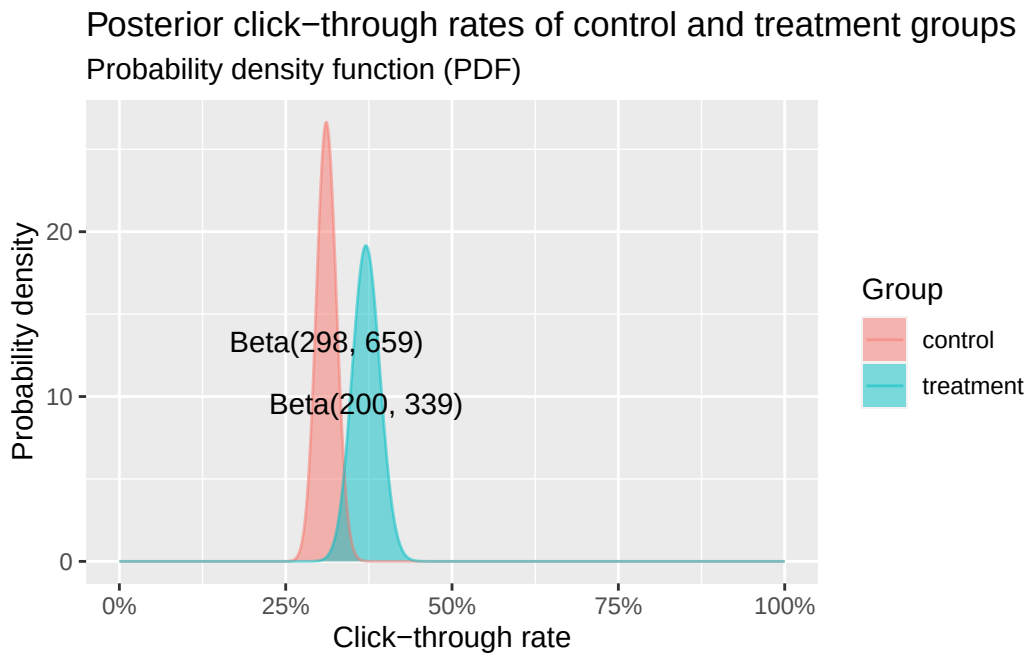
We have now updated our beliefs. These posteriors can now be thought of as our updated priors.

3.6 Another six days later...

We've now collected more data, so let's update our priors to form new posteriors.

```
new_experiment_data <- experiment_simulator(
    avg_daily_users * 6, true_ctr
)
posteriors <- posteriors %>%
    select(
        group = group,
        shape1 = posterior_shape1,
        shape2 = posterior_shape2
    ) %>%
    posterior_update(new_experiment_data)
```

```
do.call(
    plot_beta_pdf,
    posteriors %>% select(
        group = group,
        shape1 = posterior_shape1,
        shape2 = posterior_shape2
    )
) +
    labs(
        title = "Posterior click-through rates of control and treatment groups"
    )
```



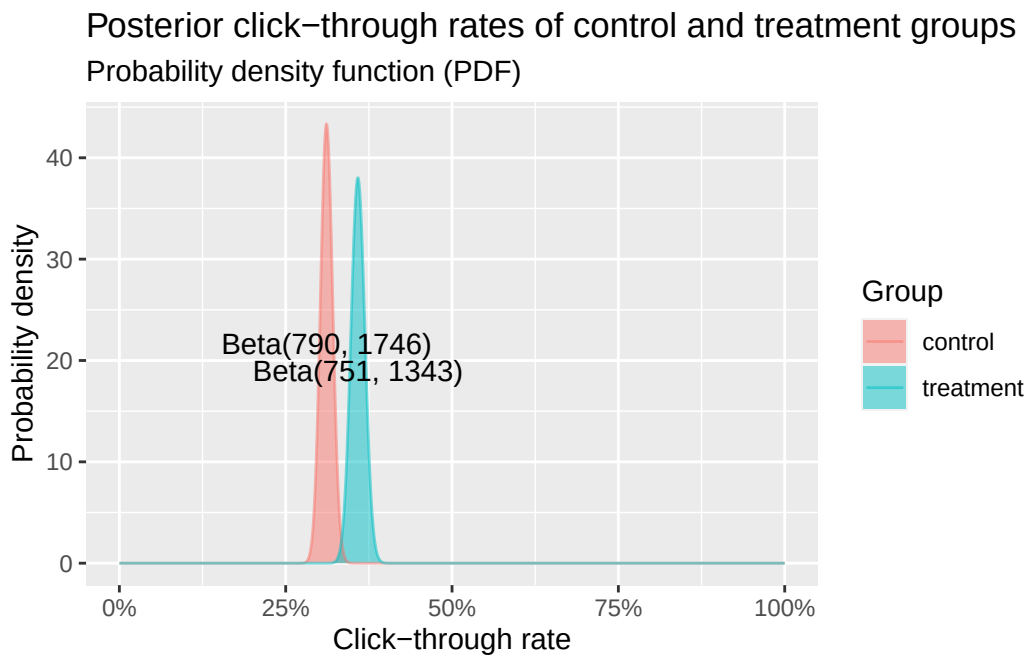
3.7 Another three weeks later...

We've now collected even more data, so let's again update our priors to form new posteriors.

```
new_experiment_data <- experiment_simulator(
    avg_daily_users * 7 * 3, true_ctr
)
posteriors <- posteriors %>%
    select(
        group = group,
        shape1 = posterior_shape1,
        shape2 = posterior_shape2
    ) %>%
    posterior_update(new_experiment_data)
```

```
do.call(
    plot_beta_pdf,
    posteriors %>% select(
        group = group,
        shape1 = posterior_shape1,
        shape2 = posterior_shape2
    )
) +
```

```
    labs(
        title = "Posterior click-through rates of control and treatment groups"
    )
```



4 Posterior analysis

Statistical inferences using the posterior distributions

4.1 Monte Carlo simulation

We can draw a very very large number of random samples from our posterior distributions to make inferences about the experiment.

This is called Monte Carlo simulation – named after a well known casino.

> **Note**
>
> The more samples drawn, the greater the reliability and precision of the inferences you make, but this comes at the cost of computational time and memory. Nowadays, computer processing speed and memory are more than adequate for what we need.

```
simulation_size <- 100
```

Figure 1: Credits: Sam Garza from Los Angeles, USA, CC BY 2.0, via Wikimedia Commons

4.2 100 simulations

Let's start slowly by drawing 100 random samples from our distributions and plotting them using histograms...

```
draw_posterior_samples <- function(posteriors, simulation_size) {
    posteriors %>%
        rowwise() %>%
        mutate(
            sim_id = list(1:simulation_size),
            ctr = list(
                rbeta(
                    n = simulation_size,
                    shape1 = posterior_shape1,
                    shape2 = posterior_shape2
                )
            )
        ) %>%
        select(group, sim_id, ctr) %>%
        unnest(cols = c(sim_id, ctr))
}

posterior_samples <- draw_posterior_samples(posteriors, simulation_size)
```

Here's some of our Monte Carlo samples:

```
posterior_samples %>%
    spread(group, ctr) %>%
    mutate(
        uplift = treatment / control - 1,
        beats_control = treatment > control
    ) %>%
    select(-sim_id) %>%
    head(n = 7) %>%
    knitr::kable(digits = 3)
```

control	treatment	uplift	beats_control
0.305	0.356	0.169	TRUE
0.285	0.356	0.249	TRUE
0.315	0.342	0.087	TRUE
0.299	0.357	0.193	TRUE
0.313	0.364	0.165	TRUE
0.313	0.337	0.077	TRUE
0.316	0.349	0.108	TRUE

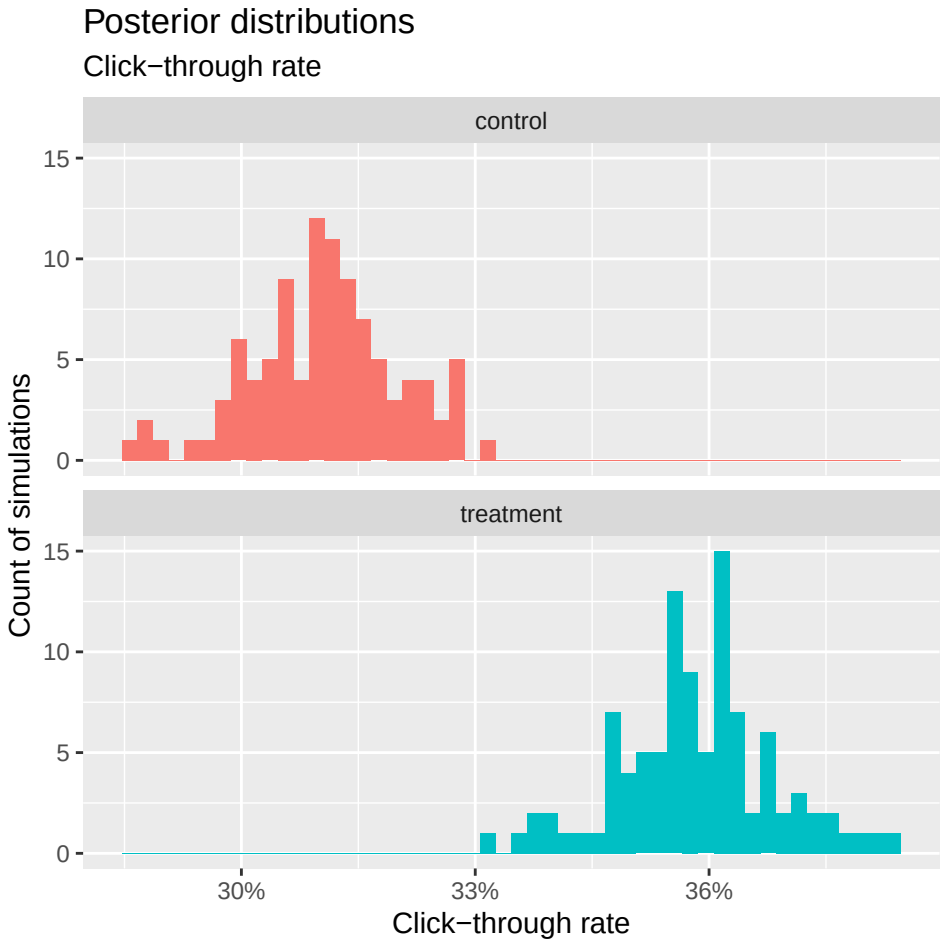
```r
plot_posterior_samples <- function(
        posterior_samples, bins = 100, animate = FALSE
) {
    posterior_samples <- posterior_samples %>%
        mutate(
            frame = cut(log(sim_id), breaks = 10, labels = FALSE)
        )
    p <- ggplot(posterior_samples) +
        facet_wrap(~group, ncol = 1) +
        aes(ctr, fill = group) +
        geom_histogram(bins = bins) +
        scale_x_continuous(labels = scales::percent) +
        labs(
            title = "Posterior distributions",
            subtitle = "Click-through rate",
            x = "Click-through rate",
            fill = "Group",
            y = "Count of simulations"
        ) +
        guides(fill = guide_none())

    if(animate) {
        p <- p +
            transition_manual(frame, cumulative = TRUE) +
            view_follow()
        p <- p %>%
            animate(duration = 5, renderer = gifski_renderer(loop = FALSE))
    }

    p

}

posterior_samples %>%
    plot_posterior_samples(animate = render_animations, bins = 50)
```



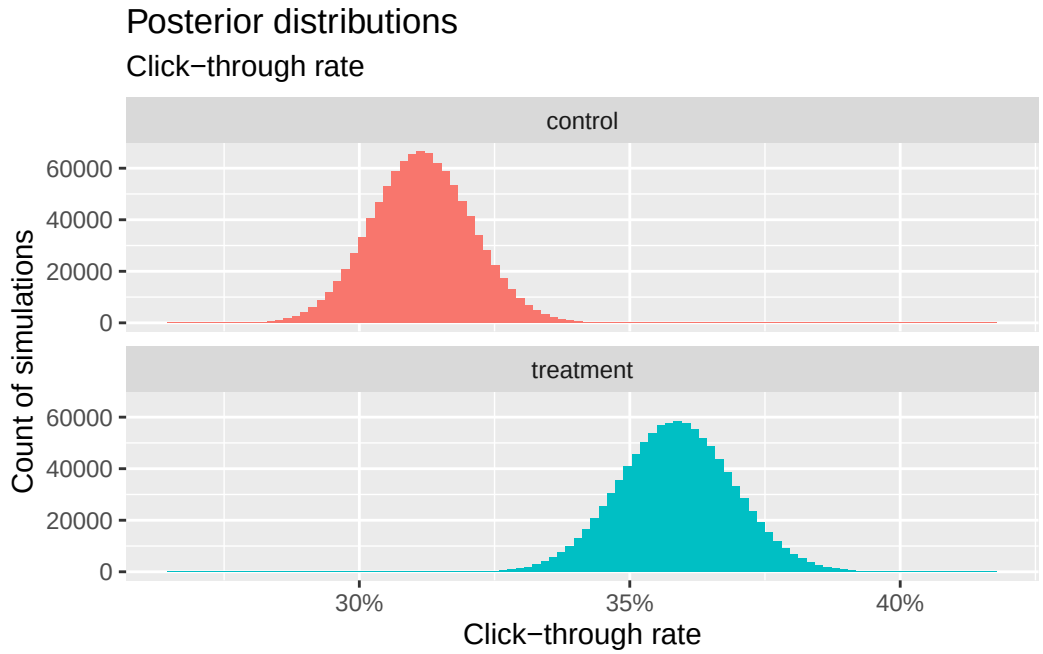
4.3 Let's now beef it up a bit...

```
simulation_size <- 1000000
```

We'll now draw 1 000 000 samples...

```
posterior_samples <- draw_posterior_samples(posteriors, simulation_size)

posterior_samples %>%
    plot_posterior_samples(animate = render_animations)
```



Notice how these histograms follow the same distribution as our posteriors. That is because these samples have been drawn at random according to those posterior distributions.

4.4 We can now make some inferences

Here's a summary of our posterior predictive distributions as a result of the 1 000 000 simulations:

```
posterior_comparison <- posterior_samples %>%
    spread(group, ctr) %>%
    mutate(
        uplift = treatment / control - 1,
        beats_control = treatment > control
    )

posterior_comparison %>%
    select(-sim_id) %>%
    summary()
```

```
   control          treatment          uplift          beats_control
Min.   :0.2652   Min.   :0.3082   Min.   :-0.05328   Mode :logical
1st Qu.:0.3053   1st Qu.:0.3516   1st Qu.: 0.11949   FALSE:352
Median :0.3115   Median :0.3586   Median : 0.15127   TRUE :999648
Mean   :0.3115   Mean   :0.3586   Mean   : 0.15223
```

```
 3rd Qu.:0.3177   3rd Qu.:0.3657   3rd Qu.: 0.18399
 Max.   :0.3544   Max.   :0.4170   Max.   : 0.39710
```

What is the probability that the CTR of the treatment is greater than that of control?

```
with(
    posterior_comparison,
    mean(beats_control)
) %>% scales::percent(0.01)
```

```
[1] "99.96%"
```

Out of our 1 000 000 simulations, we can see how often the treatment bet control. This tells us the probability that treatment is the winner.

If we filtered our simulations to those where control won and calculated the median CTR uplift, and did the same for cases where treatment won, we can determine the expected losses of choosing either variant as the winner. We should prefer the variant with the lowest expected loss or continue to run the experiment longer to improve our confidence.

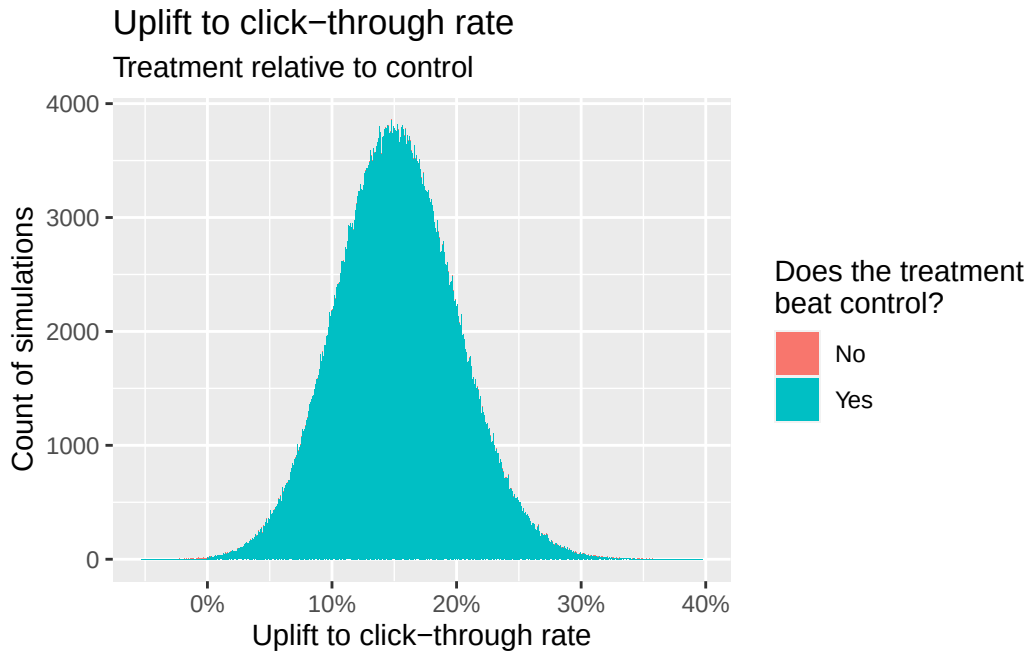
4.5 Posterior distribution of the CTR uplift

```
p <- ggplot(
    posterior_comparison %>%
        mutate(
            frame = cut(log(sim_id), breaks = 10, labels = FALSE)
        )
) +
    aes(
        uplift,
        fill = factor(
            if_else(beats_control, "Yes", "No"),
            levels = c("No", "Yes")
        )
    ) +
    geom_histogram(bins = 1000) +
    scale_x_continuous(labels = scales::percent) +
    labs(
        title = "Uplift to click-through rate",
        subtitle = "Treatment relative to control",
        x = "Uplift to click-through rate",
        y = "Count of simulations",
        fill = "Does the treatment\nbeat control?"
    )
```

```r
if (render_animations) {
    p <- p +
        transition_manual(frame, cumulative = TRUE) +
        view_follow()

    p <- p %>%
        animate(duration = 5, renderer = gifski_renderer(loop = FALSE))
}

p
```



5 When to stop a Bayesian A/B test?

5.1 If using uninformative priors...

If your original priors are uninformative or too weak, then you face the same risks as with frequentist experiments.

Perform **power analysis** ahead of running the experiment. This is to determine the required sample size before any inferences are made.

Prior to the experiment commencing, decide on:

- The **minimum detectable effect** size

- The accepted **false positive rate**
- The accepted **false negative rate**

5.2 If using informative priors...

If your priors are relatively informative and chosen carefully, then this can reduce the chances of false positives and negatives. But:

- Be careful to not bias the results of the experiment.
- Power analysis is still recommended in order to gauge the worse case scenario for how long the experiment might run.

Bayesian inference, with informative priors, can make it possible to end an experiment early.

5.3 If deciding to end early...

Ask yourself:

- Has the experiment run for at least a couple of cycles? (e.g. at least two full weeks)
- Have the results stabilised? Is there a clear winner?
- Could it be worth running longer to learn more?
- What are the risks of continuing or ending now? What if the results you see are just a fluke and are therefore misguiding you? What is the impact of making the wrong choice? What are the chances?

6 Summary and some final remarks

6.1 Before starting an experiment

Gather prior knowledge and articulate beliefs:

- Establish a baseline - what do you know about the control group?
- What do you expect the effect of the treatment to be? How sure are you?

Express those beliefs and knowledge as distributions - these are your priors for your control and treatment groups.

> **Important**
>
> Ensure that the priors encapsulate the collective knowledge and beliefs of all interested parties so that there is agreement. This helps to avoid the results from being challenged later. This is because everyone would have already had an opportunity to provide their opinions.

6.2 Running the experiment

- Start the experiment, gather data, and update your priors to form posteriors
- Draw inferences by running a large number of Monte Carlo simulations using the posteriors
- Know when to end the experiment – try to plan for this ahead of running the experiment

6.3 Final remarks

Null-hypothesis significance testing (NHST) is not what Bayesian is for:

- Bayesian tells you the probability of some effect being within some range, given the data. I.e. Given everything we know so far, what are the risks associated with the choices we have?
- NHST tells you the probability of data at least as extreme as what has been observed, given there is no real effect. I.e. How ridiculous would this outcome be if it were due to chance alone?

NHST is often referred to as the frequentist approach, where decisions are made using p-values and some arbitrary threshold α (i.e. false positive rate).

Unlike NHST, Bayesian A/B testing doesn't give you a yes/no answer – it instead informs you about the probabilities and risks associated with the choices you have.

7 Questions?

> **Tip**
>
> These presentation slides and simulations have been produced in RStudio using Quarto. You can download the source code and slides from Github at: https://github.com/jdeboer/measurecamp2022