Thompson Sampling Ad Auction

Jared Brooks

9/28/2020

Skills Practice

This is for practicing some DS skills. I am following a post from Count Bayesie here: https://www.countbayesie.com/blog/2020/9/26/learn-thompson-sampling-by-building-an-ad-auction

Data Setup

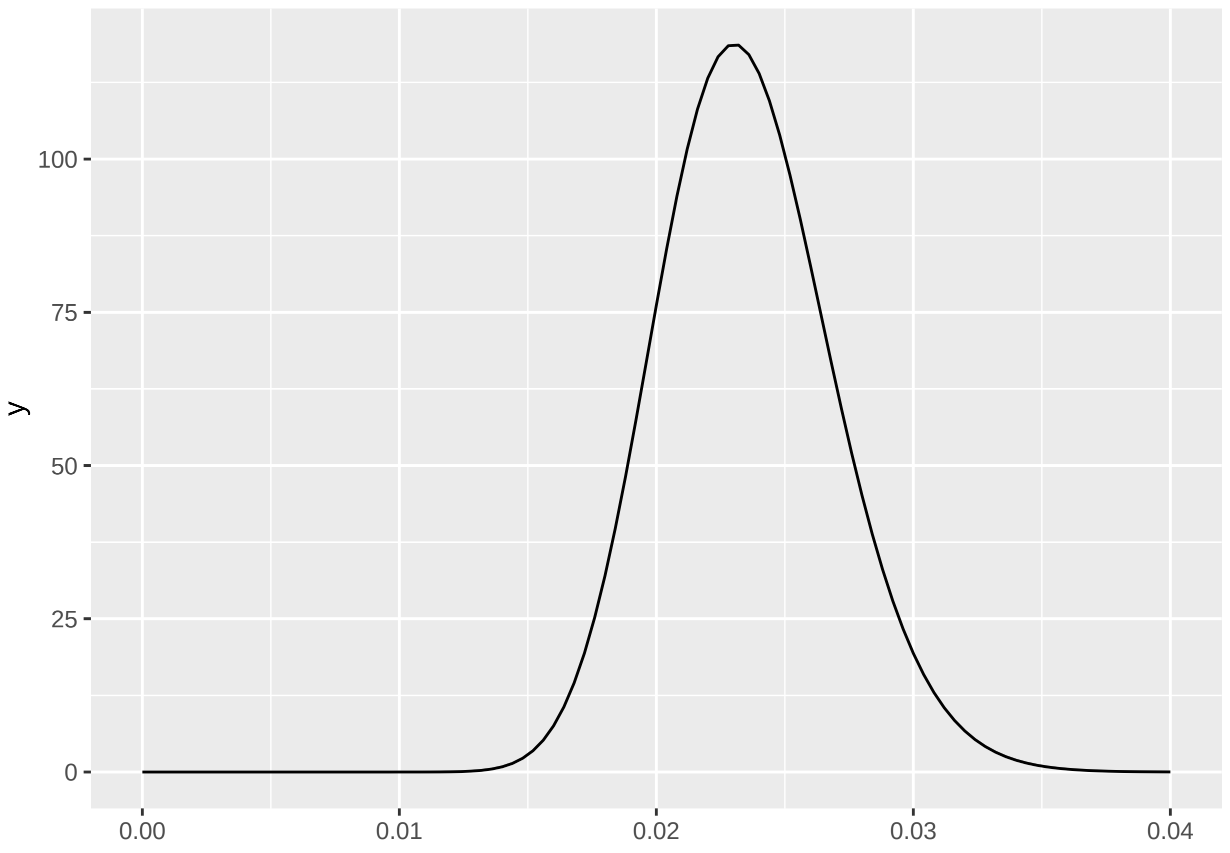
First I'm just setting up example data to play with

```r
set.seed(80085)
ad_info <- data.frame("advertiser" = factor(c("buymystuff.com", "eyespyonyou.com",
                                            "commodityfetishismllc.com"),
                                          levels = c("buymystuff.com", "eyespyonyou.com",
                                                     "commodityfetishismllc.com")),
                      "bid" = c(0.7, 0.5, 0.4), "views" = c(2000, 0,0),
                      "clicks" = c(47,0,0), "ctr" = c(0.025, 0.012, 0.045))
library(knitr)
kable(ad_info)
```

advertiser	bid	views	clicks	ctr
buymystuff.com	0.7	2000	47	0.025
eyespyonyou.com	0.5	0	0	0.012
commodityfetishismllc.com	0.4	0	0	0.045

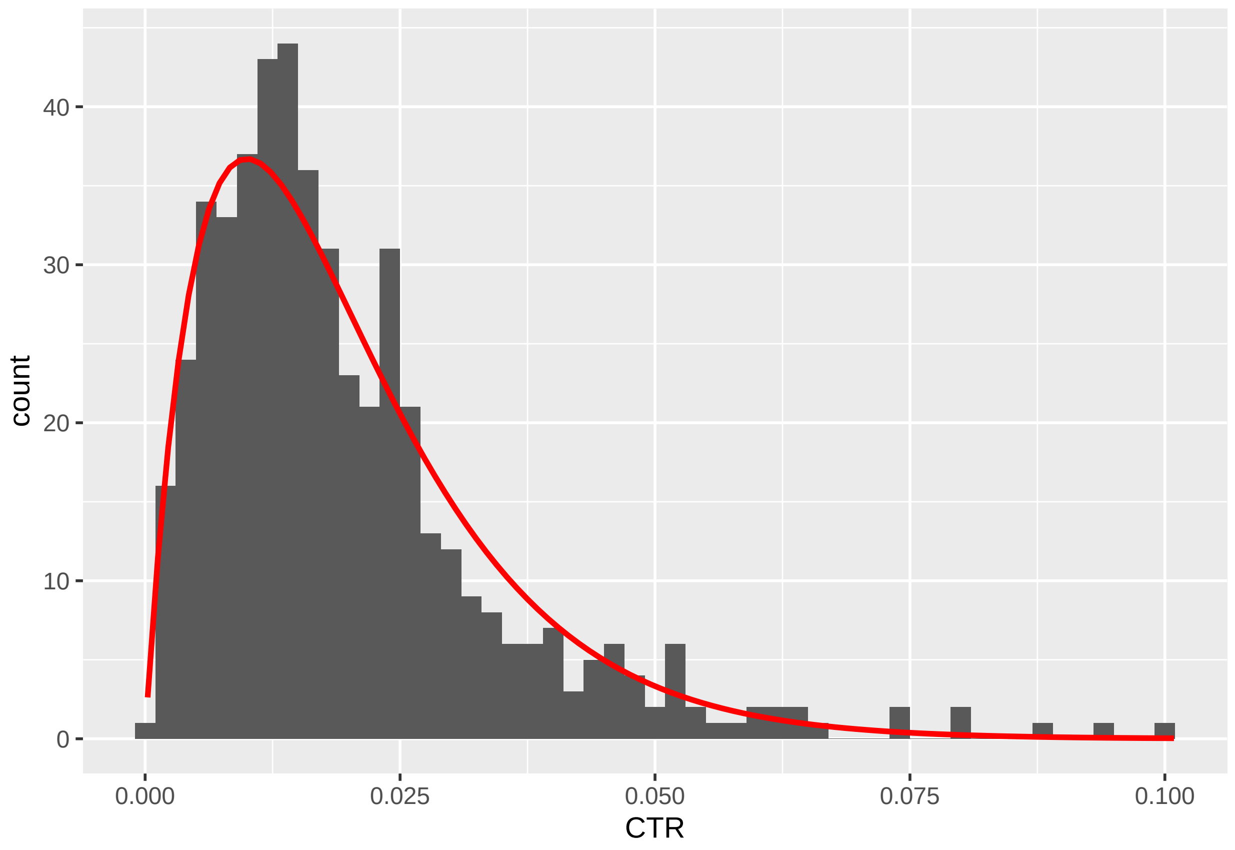
Given the previous views and clicks for "buymystuff.com" we can look at a likelihood estimate plot for its click-through-rate

```r
library(ggplot2)
ggplot() + stat_function(fun = function(x) dbeta(x, 47,1953), color="black") +
  xlim(0, 0.04)
```



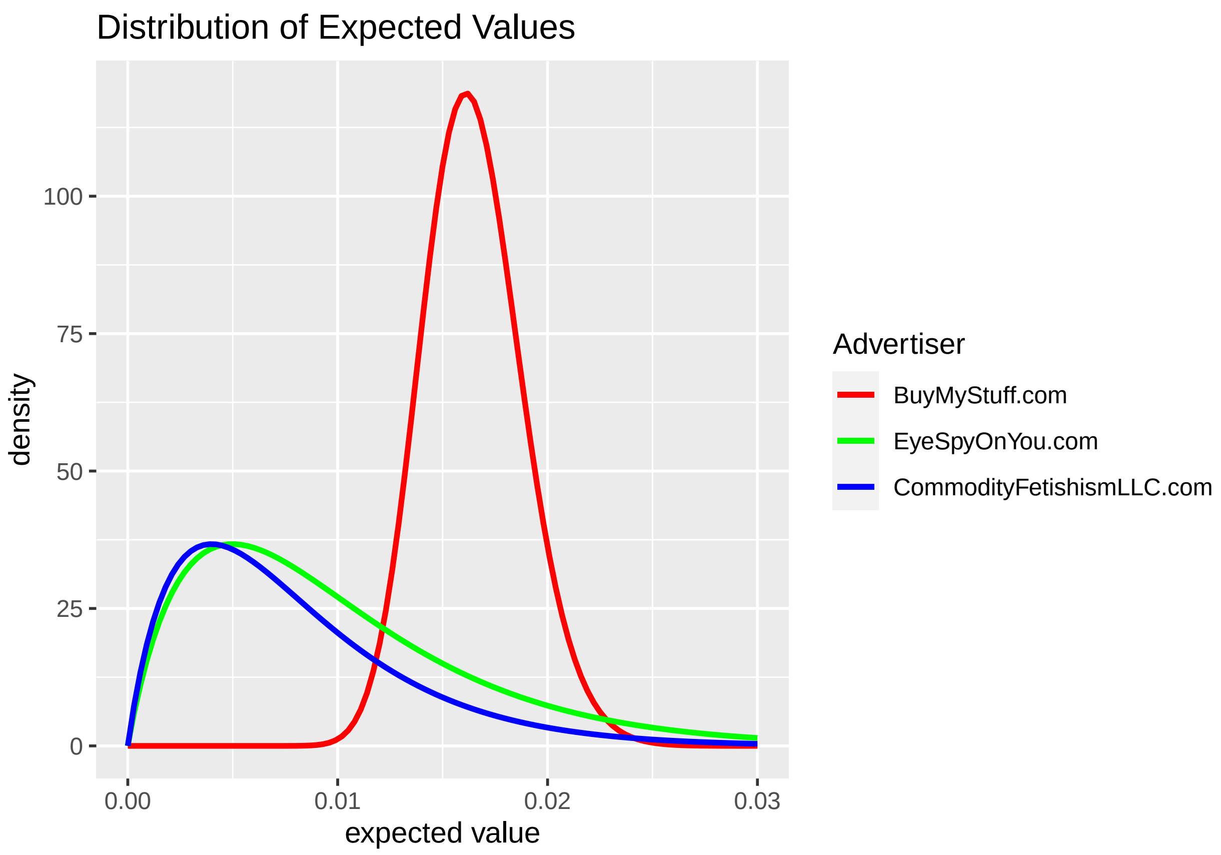
Now we pretend like we don't know the CTRs for the other two ad bidders, but that we do have a lot of information about how campaigns perform in general. We generate this using a beta distribution assuming an average CTR of 0.02 from 500 campaigns.

```
past_ctrs <- rbeta(500, 2, 98, ncp = 0)

prior_est <- MASS::fitdistr(past_ctrs, dbeta,
#choosing these shape parameters because our mean(past_ctrs) is about 0.02
                            start = list(shape1 = 2,
                                         shape2 = 98))
alpha_prior <- prior_est$estimate[1]
beta_prior <- prior_est$estimate[2]

ggplot() + geom_histogram(aes(past_ctrs), binwidth = .002) +
  stat_function(fun = function(x) dbeta(x, alpha_prior, beta_prior),
                color = "red",size = 1) +
  xlab("CTR")
```



Now, given the CTR from one bidder and using a beta distribution on the previous campaigns to estimate the CTRs from the other two bidders, and including their bids, we can plot the distributions of expected values

```r
ggplot() + stat_function(fun = function(x) dbeta(x/ad_info$bid[1], ad_info$clicks[1],
                                                  (ad_info$views[1]-ad_info$clicks[1])),
                         size=1, aes(color = "BuyMyStuff.com")) +
  stat_function(fun = function(x) dbeta(x/ad_info$bid[2], alpha_prior,
                                        beta_prior), size=1, aes(color = "EyeSpyOnYou.com") ) +
  stat_function(fun = function(x) dbeta(x/ad_info$bid[3], alpha_prior,
                                        beta_prior), size=1, aes(color = "CommodityFetishismLLC.com"))
  xlim(0,0.03) + ggtitle("Distribution of Expected Values") +
  xlab('expected value') + ylab('density') +
  scale_color_manual(name = "Advertiser",
                     breaks = c("BuyMyStuff.com", "EyeSpyOnYou.com",
                                "CommodityFetishismLLC.com"),
                     values = c("BuyMyStuff.com" = "red", "EyeSpyOnYou.com" = "green",
                                "CommodityFetishismLLC.com" = "blue") )
```



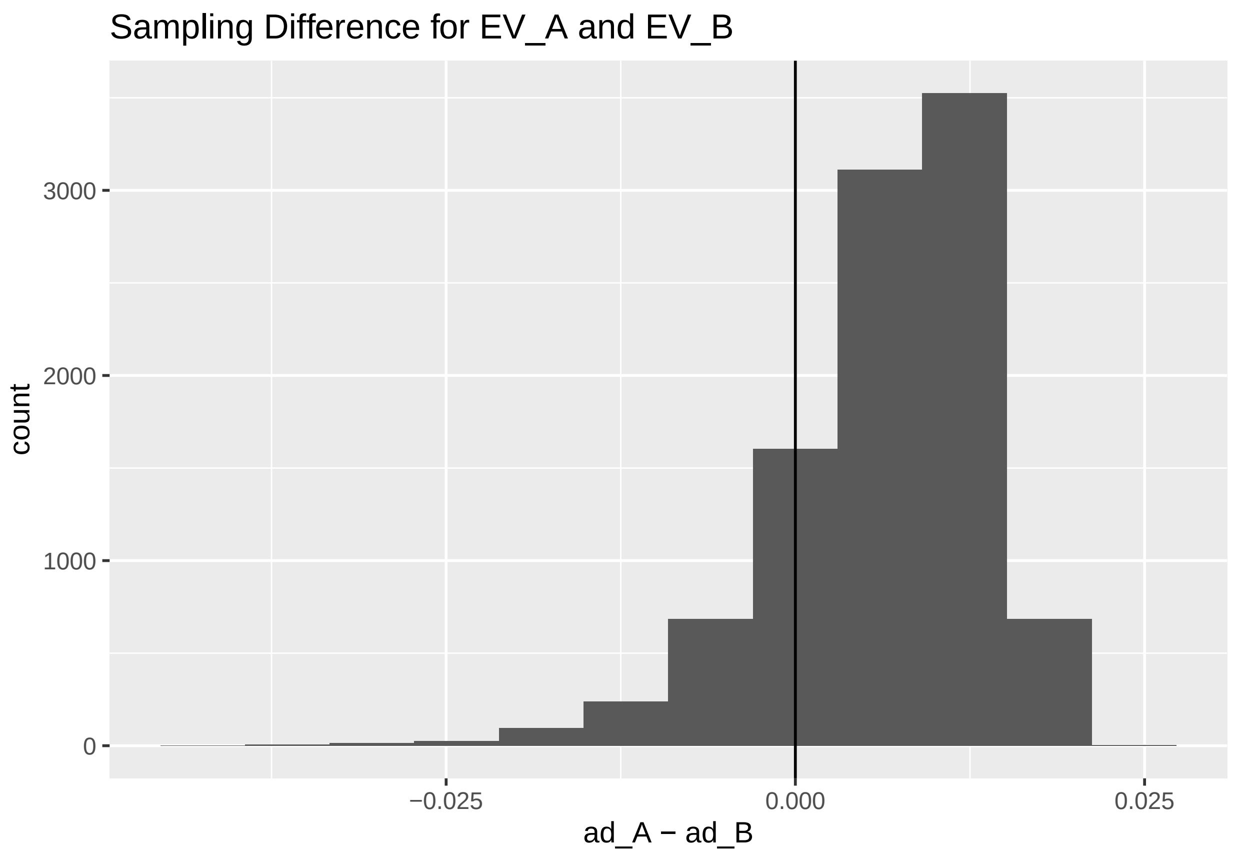
Sampling Frequency for who wins the bids

We now have enough to make estimates of the expected values to see who will win the bids by sampling from our distributions.

```
#number of samples we want
N <- 10000
#BuyMyStuff.com
ad_A <- ad_info$bid[1]*rbeta(N,
                             ad_info$clicks[1] + alpha_prior,
                             (ad_info$views[1]-ad_info$clicks[1]) +
                             beta_prior)

#EyeSpyOnYou.com
ad_B <- ad_info$bid[2]*rbeta(N,
                             ad_info$clicks[2] + alpha_prior,
                             (ad_info$views[2]-ad_info$clicks[2]) +
                             beta_prior)

#CommodityFetishismLLC.com
ad_C <- ad_info$bid[3]*rbeta(N,
                             ad_info$clicks[3] + alpha_prior,
                             (ad_info$views[3]-ad_info$clicks[3]) +
                             beta_prior)
```

Let's compare the sampling frequency of A and B

```r
ggplot() + geom_histogram(aes(ad_A-ad_B), bins = 12) +
  geom_vline(xintercept = 0) +
  ggtitle("Sampling Difference for EV_A and EV_B")
```



Let's compare B and C against A to see their probably of getting picked over A

```r
ad_info$p_gt_A <- c(0.5, sum(ad_A < ad_B)/N, sum(ad_A < ad_C)/N)
```

Ad A gets 0.5 as it is being compared against itself.

Now lets normalize these to get weighted probabilities that add to 1

```r
ad_info$weighted_p_gt_A <- ad_info$p_gt_A/(sum(ad_info$p_gt_A))
```

Sampling from Calculating Sampling Probabilities

Now we sample from these calculated probabilites to see how many new views each ad gets, and use our "known" CTRs to estimate the new clicks for each add

```r
library(dplyr)
winners_df <- sample(ad_info$advertiser,
                     prob = ad_info$weighted_p_gt_A,
                     replace = TRUE,
                     size=500) %>%
  table() %>%
  as.data.frame()

ad_info$new_views <- winners_df$Freq
```

```r
ad_info$new_clicks <- rbinom(c(1,1,1),ad_info$new_views,ad_info$ctr)
```

Results of first round of 500 views

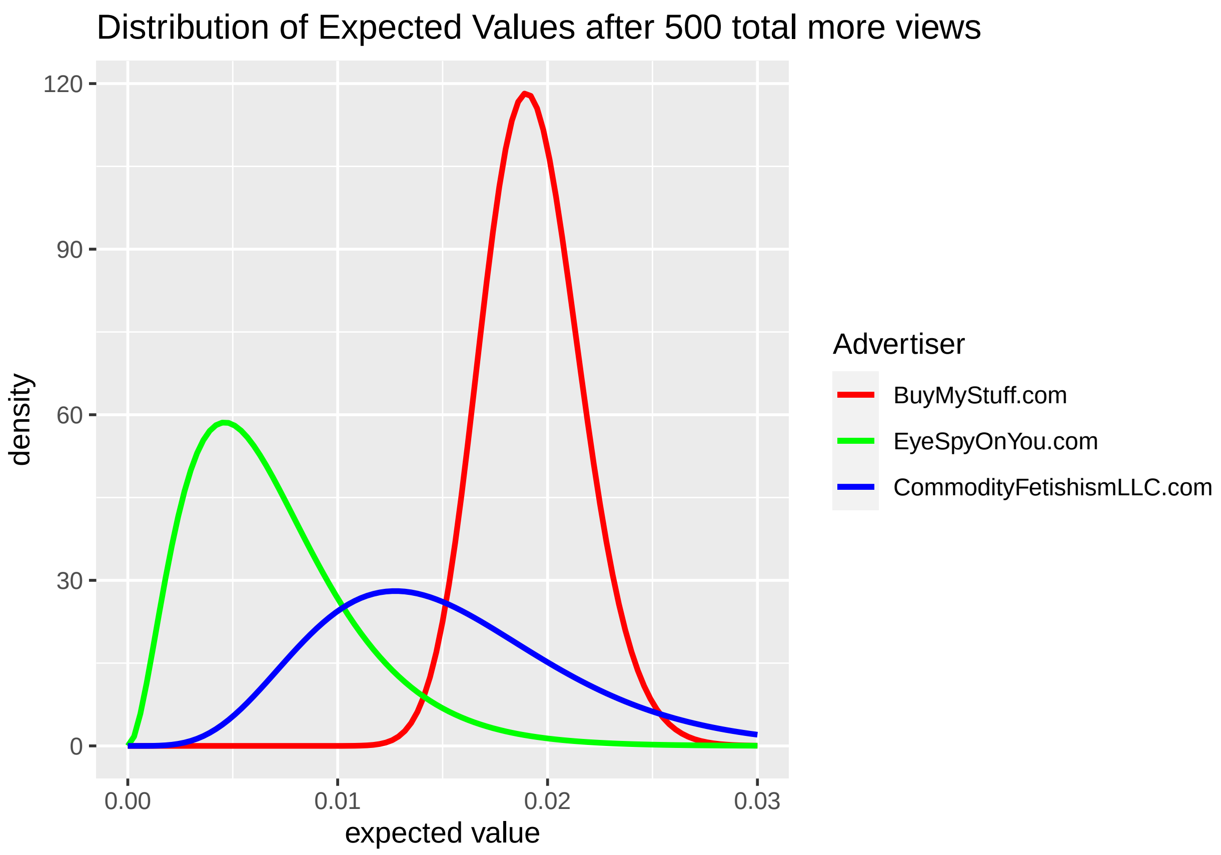
Now lets look at a table of the results

```r
kable(ad_info[,c(1,2,3,4,8,9)])
```

advertiser	bid	views	clicks	new_views	new_clicks
buymystuff.com	0.7	2000	47	325	17
eyespyonyou.com	0.5	0	0	116	1
commodityfetishismllc.com	0.4	0	0	59	4

And we can see the updated distributions

```r
ggplot() +
  stat_function(fun = function(x) dbeta(x/ad_info$bid[1], ad_info$clicks[1] + ad_info$new_clicks[1],
                                        (ad_info$views[1]-ad_info$clicks[1]) +
                                          (ad_info$new_views[1]-ad_info$new_clicks[1])),
                size=1, aes(color = "BuyMyStuff.com")) +

  stat_function(fun = function(x) dbeta(x/ad_info$bid[2], alpha_prior + ad_info$new_clicks[2],
                                        beta_prior + (ad_info$new_views[2]-ad_info$new_clicks[2])),
                size=1, aes(color = "EyeSpyOnYou.com")) +

  stat_function(fun = function(x) dbeta(x/ad_info$bid[3], alpha_prior + ad_info$new_clicks[3],
                                        beta_prior + (ad_info$new_views[3]-ad_info$new_clicks[3])),
                size=1, aes(color = "CommodityFetishismLLC.com")) +
  xlim(0,0.03) +

  ggtitle("Distribution of Expected Values after 500 total more views") +
  xlab('expected value') + ylab('density') +
  scale_color_manual(name = "Advertiser",
                     breaks = c("BuyMyStuff.com", "EyeSpyOnYou.com",
                                "CommodityFetishismLLC.com"),
                     values = c("BuyMyStuff.com" = "red", "EyeSpyOnYou.com" = "green",
                                "CommodityFetishismLLC.com" = "blue") )
```



Now we can use this to calculate and look at new sampling probabilities

```
#number of samples we want
N <- 10000
#BuyMyStuff.com
ad_A <- ad_info$bid[1]*rbeta(N,
                             ad_info$clicks[1] + ad_info$new_clicks[1] + alpha_prior,
                             (ad_info$views[1]-ad_info$clicks[1]) + (ad_info$new_views[1]-ad_info$new_cl
                             beta_prior)

#EyeSpyOnYou.com
ad_B <- ad_info$bid[2]*rbeta(N,
                             ad_info$clicks[2] + ad_info$new_clicks[2] + alpha_prior,
                             (ad_info$views[2]-ad_info$clicks[2]) + (ad_info$new_views[2]-ad_info$new_cl
                             beta_prior)

#CommodityFetishismLLC.com
ad_C <- ad_info$bid[3]*rbeta(N,
                             ad_info$clicks[3] + ad_info$new_clicks[3] + alpha_prior,
                             (ad_info$views[3]-ad_info$clicks[3]) + (ad_info$new_views[3]-ad_info$new_cl
                             beta_prior)

ad_info$p_gt_A <- c(0.5, sum(ad_A < ad_B)/N, sum(ad_A < ad_C)/N)
ad_info$weighted_p_gt_A <- ad_info$p_gt_A/(sum(ad_info$p_gt_A))

kable(ad_info[,c(1,6,7)])
```

advertiser	p_gt_A	weighted_p_gt_A
buymystuff.com	0.5000	0.6613757
eyespyonyou.com	0.0144	0.0190476
commodityfetishismllc.com	0.2416	0.3195767

Continuous updating

Now we see how distributions and sampling proportions change with continuous update.

We'll update for every 100 views 50 times, for a total of 5000 new views.

```r
library(reshape2)
# Set up initial info

## Set data for advertisers

ad_info <- data.frame("advertiser" = factor(c("buymystuff.com", "eyespyonyou.com",
                                             "commodityfetishismllc.com"),
                                           levels = c("buymystuff.com", "eyespyonyou.com",
                                                      "commodityfetishismllc.com")),
                      "bid" = c(0.7, 0.5, 0.4), "views" = c(2000, 0,0),
                      "clicks" = c(47,0,0), "ctr" = c(0.025, 0.012, 0.045))

## Set data for previous campaigns

past_ctrs <- rbeta(500, 2, 98, ncp = 0)

## Fit data from previous campaigns to beta distr

prior_est <- MASS::fitdistr(past_ctrs, dbeta, start = list(shape1 = 2, shape2 = 98))
alpha_prior <- prior_est$estimate[1]
beta_prior <- prior_est$estimate[2]

## Set posteriors

alpha_post <- c(1,1,1)*alpha_prior + ad_info$clicks
beta_post <- c(1,1,1)*beta_prior + (ad_info$views - ad_info$clicks)

## Set sampling probabilities

N <- 10000

ad_A <- ad_info$bid[1]*rbeta(N, alpha_post[1], beta_post[1])
ad_B <- ad_info$bid[2]*rbeta(N, alpha_post[2], beta_post[2])
ad_C <- ad_info$bid[3]*rbeta(N, alpha_post[3], beta_post[3])

ad_info$p_gt_A <- c(0.5, sum(ad_A < ad_B)/N, sum(ad_A < ad_C)/N)

ad_info$weighted_p_gt_A <- ad_info$p_gt_A/(sum(ad_info$p_gt_A))


samp_prop = data.frame(t(ad_info$weighted_p_gt_A))
```

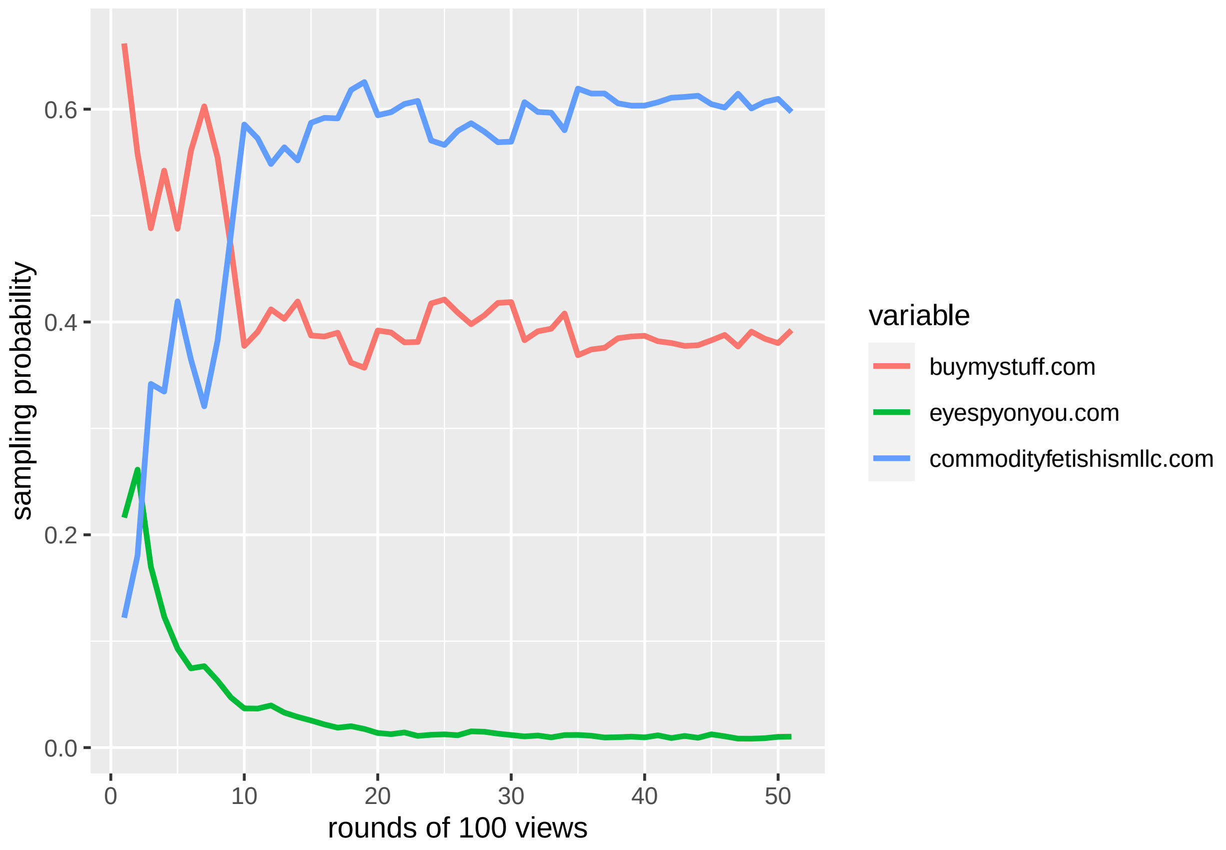
```r
# Using sampling probabilities and underlying CTRs to set new views and clicks

ad_info$total_views <- ad_info$views
ad_info$total_clicks <- ad_info$clicks

i <- 1

while (i<51) {

  winners_df <- sample(ad_info$advertiser,
                       prob = probs <- ad_info$weighted_p_gt_A,
                       replace = TRUE,
                       size=100) %>%
      table() %>%
      as.data.frame()


  ad_info$new_views <- winners_df$Freq

  ## Set new clicks using the calculated new views from sampling probabilites and the underlying CTR
  ad_info$new_clicks <- rbinom(c(1,1,1),ad_info$new_views,ad_info$ctr)

  ## Add new values to totals
  ad_info$total_views <- ad_info$total_views + ad_info$new_views
  ad_info$total_clicks <- ad_info$total_clicks + ad_info$new_clicks

  ## Use new clicks and views to update posteriors
  alpha_post <- alpha_post + ad_info$new_clicks
  beta_post <- beta_post + (ad_info$new_views - ad_info$new_clicks)

  ## Calculate new sampling frequencies
  ad_A <- ad_info$bid[1]*rbeta(N, alpha_post[1], beta_post[1])
  ad_B <- ad_info$bid[2]*rbeta(N, alpha_post[2], beta_post[2])
  ad_C <- ad_info$bid[3]*rbeta(N, alpha_post[3], beta_post[3])

  ad_info$p_gt_A <- c(0.5, sum(ad_A < ad_B)/N, sum(ad_A < ad_C)/N)
  ad_info$weighted_p_gt_A <- ad_info$p_gt_A/(sum(ad_info$p_gt_A))

  ## Add new sampling frequencies to dataframe for plotting
  samp_prop[nrow(samp_prop)+1,] = t(ad_info$weighted_p_gt_A)

  i <- i+1
}
```

Change of Sampling Probability over time

After 50 rounds of 100 views we can see how our sampling probabilities changed. Even though "CommodityFetishismLLC.com" started out with the lowest EV and therefore the smalled sampling probability, its underlying CTR helped it to rise to the largets sampling probability after only 10 rounds.

```r
names(samp_prop) <- ad_info$advertiser
samp_prop$id = 1:nrow(samp_prop)
df_long <- melt(samp_prop, id.vars = 'id')
```

```
ggplot() + geom_line(data = df_long, aes(x = id, y = value, color = variable), size = 1) +
  xlab('rounds of 100 views') + ylab('sampling probability')
```



The table below shows the total clicks and views of each advertiser after 50 rounds.

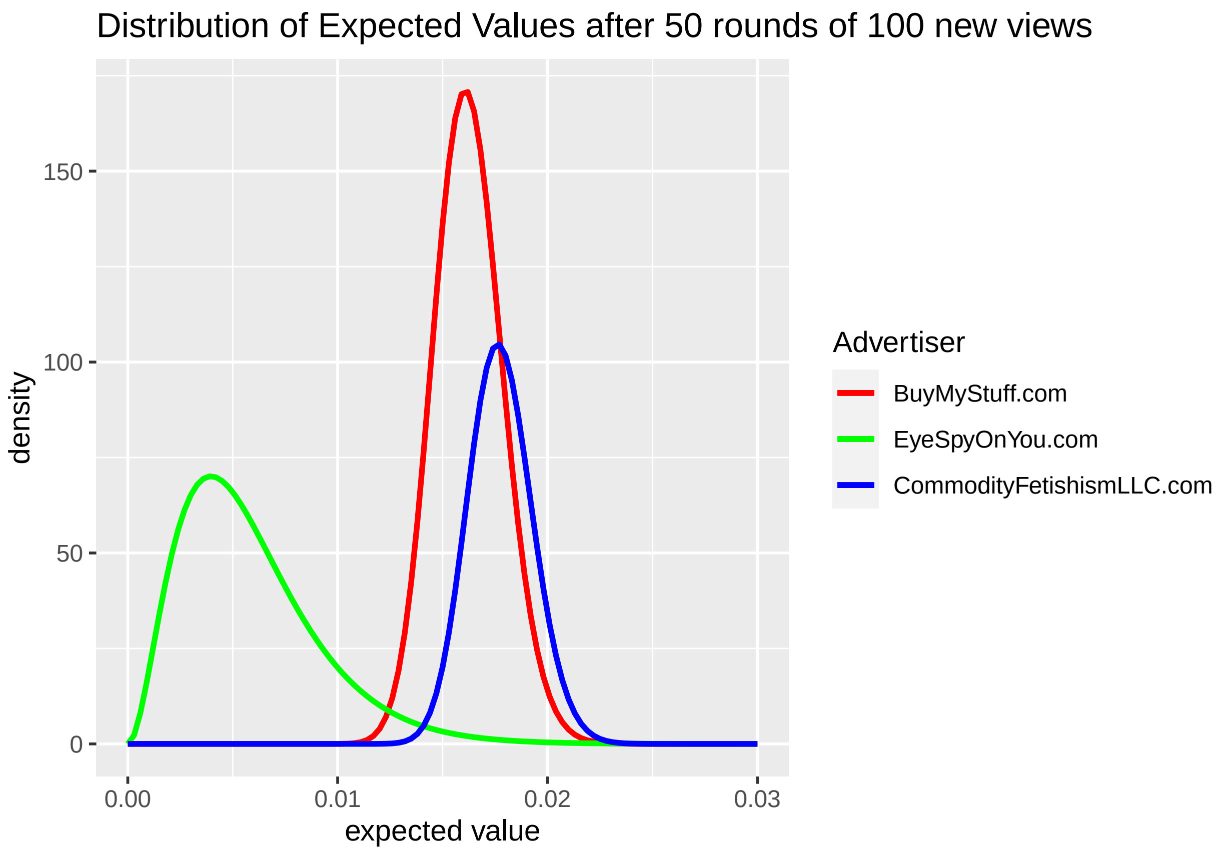
```
kable(ad_info[,c(1,2,5,7,8,9)])
```

advertiser	bid	ctr	weighted_p_gt_A	total_views	total_clicks
buymystuff.com	0.7	0.025	0.3922799	4037	94
eyespyonyou.com	0.5	0.012	0.0102777	159	1
commodityfetishismllc.com	0.4	0.045	0.5974423	2804	127

New Distributions

Now that we have so much more data for all three advertisers, we can get a much more reliable set of distributions for the expected values of their ads. We can see that they are tighter and more closely reflect their underlying CTRs.

```
ggplot() +
  stat_function(fun = function(x) dbeta(x/ad_info$bid[1], alpha_post[1], beta_post[1]),
                size=1, aes(color = "BuyMyStuff.com")) +

  stat_function(fun = function(x) dbeta(x/ad_info$bid[2], alpha_post[2], beta_post[2]),
                size=1, aes(color = "EyeSpyOnYou.com")) +

  stat_function(fun = function(x) dbeta(x/ad_info$bid[3], alpha_post[3], beta_post[3]),
                size=1, aes(color = "CommodityFetishismLLC.com")) +
  xlim(0,0.03) +

  ggtitle("Distribution of Expected Values after 50 rounds of 100 new views") +
  xlab('expected value') + ylab('density') +
  scale_color_manual(name = "Advertiser",
                         breaks = c("BuyMyStuff.com", "EyeSpyOnYou.com",
                                    "CommodityFetishismLLC.com"),
                         values = c("BuyMyStuff.com" = "red", "EyeSpyOnYou.com" = "green",
                                    "CommodityFetishismLLC.com" = "blue") )
```



Looking towards the future

As we get more views, these distributions will tighten further. But we want to allow for CTRs (and, of course, the EVs) to changes over time, so we might want to restrict the age of view/click data we wish to allow into our calculations. Maybe only the last week or month of data, depending on the timescale you expect these values to change.

We would also change the computation of the sampling probabilities such that the other ads would get compared against the current leader, instead of always being compared against ad_A