Università degli Studi di Milano

MSc. in Data Science and Economics

Bayesian Analysis Exam

A mobile videogame Bayesian Analysis: *Cookie Cats*

Alessia Di Giovanni
alessia.digiovanni2@studenti.unimi.it
Federico Vinci
federico.vinci1@studenti.unimi.it

May 5, 2023

Abstract

"Cookie Cats" is a connected-three mobile game where players encounter gates as they advance through different levels of the game. These *gates* require them to wait for a certain amount of time or buy an in-app purchase to proceed further. The first gate players encounter is at level 30th. Developers are questioning whether moving the gate from level 30th to level 40th would be a wise choice or not. A *Bayesian A/B test* is the optimal choice to figure out which location of the gate would lead to the best retention. The results show that users who encounter the gate at level 40 are 1% and 4% less likely to return after 1 day and 7 days, respectively, to users who encounter the gate at level 30. Moreover, to gain a better understanding of the game, a second question is raised regarding whether users who play more rounds are more likely to return after 7 days. By conducting a *Bayesian Logistic Regression*, it is found out that the number of rounds played can be considered as a reliable indicator of player retention.

Contents

1 Introduction

1.1 Cookie Cats

Cookie Cats is a popular mobile puzzle game developed by *Tactile Entertainment*. It is a classic "connect three"-style puzzle game where the player has to connect tiles of the same color to create a sequence. The goal is to feed cats with the cookies.

Figure 1: *Cookie Cats game*

1.2 Problem statement

As players progress through the levels of the game, they will occasionally encounter *gates* that force them to wait a non-trivial amount of time or make an in-app purchase to progress. In addition to driving in-app purchases, *gates* also serve the important purpose of giving players an enforced break from playing the game, hopefully resulting in an increasing and prolonging player's enjoyment of the game. The question lies on where should the first *gate* be placed. In the original variant of the game the gate is placed at level 30th, however, game creators analyse the possibility of moving it to level 40th.

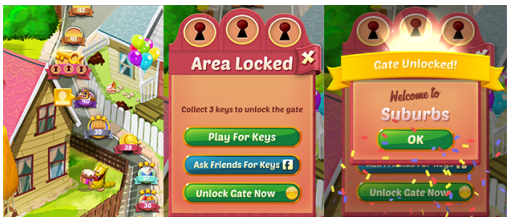


Figure 2: *Levels and gate*

One of the most appropriate solutions to determine where to place the gate is a **Bayesian A/B test**. This statistical experiment involves showing different versions of a game feature to different groups of users to evaluate whether a change will have a positive or negative impact. Group A will see the feature as it currently is, while Group B will see the same feature with an edit. The test lasts for a set period and includes metrics to determine which version of the feature is best.

The Bayesian A/B test on Cookie Cats will see involved:

- **Group A**: users who are shown the original version of the game, in which the gate is placed at the 30th level;
- **Group B**: users who are shown an alternative version of the game, in which the gate is placed at the 40th level.

It is important to remark that the A/B test is conducted from period t to period $t+n$ where the users involved are those that download the game within this period. Therefore, a user that downloads the game at period t could either play version A (`gate_30`) or version B (`gate_40`).

The second part of the project will feature a **Bayesian Logistic Regression** to understand the underlying relationship that lies between the *rounds played* by a player and its chance to come back (ie. *retention*) one week later.

In this case, the different variants of the game will not be considered (there will be no split of users) and the goal will be to understand what's the magnitude of the coefficient, ie. how strong is the relationship between the two variables. The outcome, the Y, is modeled as a binary categorical response variable, converted to a $0-1$ indicator, and the predictor is the number of rounds played.

2 Data

2.1 Dataset

The data comprises 90.189 players that install the game right after the *A/B test* is initialised. Each player is identified by a row and characterised by four variables. The variables are the following:

- `version` - whether the player is put in the control group A (`gate_30` - first gate at level 30) or in the treatment group B, the *shifted-gate group* (`gate_40` - first gate at level 40);
- `sum_gamerounds` - the number of rounds played by each user within the first 14 days after downloading the game;
- `retention_1` - whether the player come back and play 1 day after downloading the game;
- `retention_7` - whether the player come back and play 7 days after downloading the game.

The screenshot of the `df.head()` shows how the data look like.

	version	sum_gamerounds	retention_1	retention_7
0	gate_30	3	0	0
1	gate_30	38	1	0
2	gate_40	165	1	0
3	gate_40	1	0	0
4	gate_40	179	1	1

Figure 3: *An overview of the dataset.*

2.2 Exploratory Data Analysis

Before moving to the modeling phase, it is important to explore the data and manipulate it if needed. The `user`'s label is eliminated, as it does not add any relevant information. Amongst the 90.189 players, there are 3.994 players who install the game but never play any *round* during the A/B test period, thus not having the chance to test themselves on where the gate would influence their decision on whether to return after one day or one week. Finally, the total number of observations is 86.195.

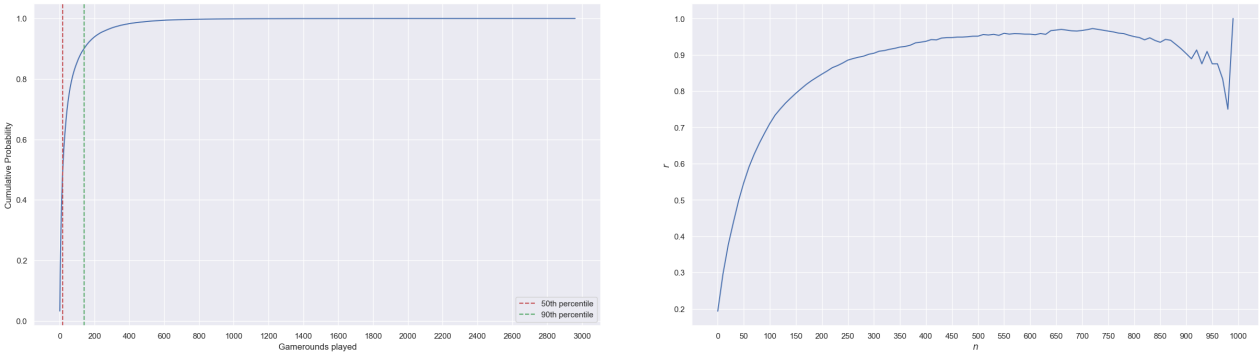
For the A/B test, the dataset is divided into two groups based on the game variants assigned to the players. Variant A, which has the gate at level 30, is assigned to 42.763 players, while variant B, with the gate at level 40, is assigned to 43.432 players. Amongst the players in variant A, $46,75\%$ returns to play after one day, while $46,21\%$ of those in variant B returns. The retention rates after 7 days are much lower, with only $19,84\%$ of players in variant A returning to play and $19,03\%$ of those in variant B.

During the period of the analysis, users play on average 54 rounds, with a minimum of 1 round played and a maximum of 4.900. This could be considered as an outlier and therefore dropped, resulting in 2.961 as the new maximum.
In Figure 4a the cumulative probability of users who play n rounds is displayed. Figure 4b shows the relationship between r and n, where:

$$r = \frac{number\ of\ users\ who\ play\ n\ rounds\ and\ return\ after\ 1\ week}{number\ of\ users\ who\ play\ n\ rounds}$$

and n equal or less than 1000.

The choice of selecting such n is mainly driven by observing the shape of the *CDF*: given its *skewness*, it is noticeable that 90% of users play less than or exactly 140 rounds. Moreover, for any $n \rightarrow$ `max(sum_gamerounds)`, not only the number of players decreases a lot (see 4a), but their rate of 1-week conversion steadily increases until stabilizing around the value of 95%. With this, it is deducible that those few players who play a significant amount of rounds (arbitrarily larger than 140) are almost likely to come back after one week. As for this, those *dedicated gamers* won't be considered for the sake of the second part of this project (namely the Bayesian Logistic Regression) as they do not add any additional information about the state of the retention and will give an overestimation of the parameter of interest. Furthermore, this part of the project mainly focuses on average behaviour of players and not on unlikely ones. Finally, by focusing on this subset, many computationally-intense algorithms will also flow more smoothly. The `dataframe` used for this second part is called `log_df`.



(a) *The percentage of players who play less or equal than n rounds.*

(b) *Relationship between n and r.*

Figure 4

3 Bayesian A/B test

3.1 Frequentist vs Bayesian approach

In frequentist A/B test, a null hypothesis and an alternative hypothesis are established to test the statistical significance of a proposed difference between the two groups. The null hypothesis assumes that there is no significant difference between the two groups being compared, while the alternative hypothesis assumes that there is a statistically significant difference between them. To establish if the null hypothesis is accepted or not, the *p-value* is considered. Moreover, when frequentist A/B test is performed, only the data of the experiment are considered. In Bayesian A/B testing, the information about the current data is meshed with past knowledge of similar experiments, whose knowledge is approximately represented by the **prior distribution**. By multiplying the prior distribution with the **likelihood**, which is a function that represents the probability of observing the data given a set of model parameters, a **posterior distribution** is finally obtained. The posterior distribution gives room for its **moments**, such as its **expected value** or **variance**.

As Matt Gershoff [1] explains:[3]

> *"The difference is that, in the Bayesian approach, the parameters that we are trying to estimate are treated as* ***random variables****. In the frequentist approach, they are fixed. Random variables are governed by their parameters and distributions. The prior is just the prior belief about these parameters. In this way, we can think of the Bayesian approach as treating probabilities as degrees of belief, rather than as frequencies generated by some unknown process."*

So it can be said that the Bayesian approach is preferred for two main reasons: first, the result (ie. the posterior distribution) is not a point estimate but rather a probability distribution;[1] secondly, the prior beliefs allow to preserve the element of uncertainty and the accuracy of the outcome is inversely proportional to the degree of bias

[1]Matt Gershoff is Co-founder of Conductrics, an intelligent decision engine platform (www.conductrics.com). In an interview he shares thoughts about A/B testing and experimentation in this.

present in the prior used. This is mostly true when working with big samples (the number of observations tends to infinity).

3.2 Bayesian statistics

In this section, the **Bayes Theorem** is introduced[29] . Its formula is based on conditional probability: the conditional probability of an event A given the event B is given by:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

where $P(B) > 0$.

If random variables are considered instead of events, the Bayes theorem can be rewritten as:

$$\pi(\theta|\underline{x}) = \frac{f(\underline{x}|\theta)\pi(\theta)}{f(\underline{x})} \tag{2}$$

where $\underline{x}$ is the vector of observations, $\underline{x} = (x_1, x_2, ..., x_n)$, with n number of players, and θ is the parameter to be estimated. The denominator, represented by $f(\underline{x})$, is a function of the observations and it does not depend on the parameter that has to be estimated. For this reason it can be considered as a constant and it allows to rewrite the Bayes theorem as an approximation in this way:

$$\pi(\theta|\underline{x}) = \frac{f(\underline{x}|\theta)\pi(\theta)}{f(\underline{x})} \propto f(\underline{x}|\theta)\pi(\theta) \tag{3}$$

Let's analyse the terms of the equation. The first element $f(\underline{x}|\theta)$, which is the **likelihood**, incorporates the informations given by the data. The prior beliefs are expressed through the **prior distribution** $\pi(\theta)$. What it is obtained is the **posterior distribution** for theta given by $\pi(\theta|\underline{x})$.

The dataset features two *dependent variables*, `retention_1` and `retention_7`. Both are represented as *boolean* outcomes, namely `True` if the player returns after one day (seven days) or `False` otherwise. Note that these variables have been converted into `int`. The observed data for the duration of the A/B test are represented by the number of players who download the game within the period the A/B test is ongoing and the number of players who come back one day (seven days) after installing the game. The dependent variables are modeled by a **Binomial distributions** in this way: for group A

$$retention_1^A \sim \text{Binomial}(N_A, p = \theta_A^1), \quad retention_7^A \sim \text{Binomial}(N_A, p = \theta_A^7)$$

and for group B:

$$retention_1^B \sim \text{Binomial}(N_B, p = \theta_B^1), \quad retention_7^B \sim \text{Binomial}(N_B, p = \theta_B^7)$$

with N_v $(v = A, B)$ the number of players who installed the game and happened to be in game's variant v and θ_v^1 and θ_v^7 are the conversion rates for variant v.

Within the Bayesian context, the prior knowledge, which is supposed to derive from past conversion rates of previous similar experiments, is assumed to be shaped by a *prior distribution*. In this case, the θ_v^1 and θ_v^7 are drawn from **Beta distributions**:

$$\theta_v^1 \sim \theta_v^7 \sim \text{Beta}(\alpha, \beta)$$

The Beta distribution is a family of continuous probability distributions defined on interval $[0, 1]$ and it is characterized by two positive parameters, α and β. The parameter α is known as the *shape* parameter and controls the shape of the distribution. Specifically, higher values of α result in a more peaked and concentrated distribution, while lower values of α produce a flatter and more spread-out distribution. The parameter β is known as the *scale* parameter and controls the location of the distribution. Higher values of β shift the distribution to the right, while lower values of β shift the distribution to the left. Together, these two parameters allow us to specify a wide range of different beta distributions with varying shapes and locations.

Given the beta distribution as the prior distribution, it follows that the posterior distribution of binomial likelihood will also converge towards a beta distribution. This is explained by the fact that beta distribution is a

conjugate prior for the binomial likelihood.

A family F of prior distributions for θ is said to be **conjugate** to $f(\underline{x}|\theta)$ if for each prior distribution $\pi(\theta) \in F$, the posterior distribution $\pi(\theta|\underline{x})$ is also in F. Here follows the proof. Given the likelihood function

$$f(\underline{x}|\theta) = \prod_{i=1}^{n} f_X(x_i|\theta) = \prod_{i=1}^{n} \binom{n}{x_i} \theta^{x_i}(1-\theta)^{n-x_i} \tag{4}$$

and the prior distribution

$$\pi(\theta) = \frac{1}{B(\alpha,\beta)} \theta^{\alpha-1}(1-\theta)^{\beta-1}, \tag{5}$$

the posterior distribution follows as

$$\begin{aligned} \pi(\theta|\underline{x}) \propto & f(\underline{x}|\theta)\pi(\theta) = \prod_{i=1}^{n} f_X(x_i|\theta) \cdot \pi(\theta) = \prod_{i=1}^{n} \binom{n}{x_i} \theta^{x_i}(1-\theta)^{n-x_i} \cdot \frac{1}{B(\alpha,\beta)} \cdot \theta^{\alpha-1}(1-\theta)^{\beta-1} \\ \propto & \prod_{i=1}^{n} \theta^{x_i+\alpha-1}(1-\theta)^{n-x_i+\beta-1} = \theta^{\sum_{i=1}^{n} x_i+\alpha-1}(1-\theta)^{n-\sum_{i=1}^{n} x_i+\beta-1} = Beta\left(\sum_{i=1}^{n} x_i + \alpha, n - \sum_{i=1}^{n} x_i + \beta\right). \end{aligned} \tag{6}$$

This is why there is no need of using Markov Chain Monte Carlo sampling methods to estimate the posterior distribution.

3.3 MCMC sampling methods

Bayesian Data Analysis[6] book defines **Markov Chain Monte Carlo** (MCMC) as a family of computational algorithms for generating samples from a target probability distribution. The basic idea behind MCMC is to construct a Markov Chain (a systematic method for generating a sequence of random variables where the current value is probabilistically dependent on the precedent value), where each state of the chain corresponds to a sample from the target distribution. The chain is constructed such that its stationary distribution is the target distribution of interest. The goal is to estimate the posterior distribution of a set of model parameters given some observed data. The MCMC algorithm generates a sequence of samples from the posterior distribution, which can be used to estimate the posterior mean, variance, quantiles and other summary statistics of interest. In this project, two of MCMC methods will be used: Gibbs Sampler and Metropolis-Hastings.

Gibbs sampler is often used when the full conditional distributions of the parameters are known and can be easily sampled from. It is particularly useful when the joint posterior distribution of the model's parameters has a simple factorization structure. It works first initialising the parameters of interest; then, a value for the first parameter is sampled from its conditional distribution, given the current values of the other parameters. Iteratively, this is done for the other parameters as well. These steps are repeated for a large number of iterations.

Metropolis-Hastings is used when it is not possible (or convenient) to directly sample from the conditional distributions. Also for this algorithm the parameters of interest are firstly initialized. Then, a new value for a parameter is proposed using a proposal distribution and it can be accepted or rejected based on the acceptance probability, designed to ensure that the target distribution is stationary under the Markov Chain defined by the sampler. Iteratively, this is done for the other parameters. These steps are repeated for a large number of iterations.

To summarize, if the posterior distribution has a simple structure, Gibbs sampler can be more efficient and accurate than the Metropolis-Hastings; on the other side, if the posterior distribution is complex, the Metropolis Hastings may be more appropriate.

3.3.1 Model Checking and Diagnostics

Model diagnostics in Bayesian statistics refer to a set of techniques used to assess the performance of the MCMC sampling results. These techniques help to identify potential problems of the model, such as lack of convergence and thus the presence of bias. The goal of model diagnostics is to ensure that the model is providing accurate and reliable estimates of the parameters of interest and to identify any areas where the model or inference procedure may need to be improved. Besides assessing the absence of autocorrelation in the trace, some common techniques used

for model diagnostics include some tests such as the **Gelman-Rubin**.[4] It computes the *between-chain* variance (B) and the *within-chain* variance (W) and assesses whether they are different enough to worry about convergence. Considering m chains, each of length n,

$$B = \frac{n}{m-1}\Sigma_{j=1}^{m}\left(\overline{\theta_j} - \overline{\overline{\theta}}\right)^2, \quad \text{where} \quad \overline{\overline{\theta}} = \frac{1}{m}\Sigma_{j=1}^{m}\overline{\theta_j}, \qquad W = \frac{1}{m}\Sigma_{j=1}^{m}s_j^2, \quad \text{where} \quad s_j^2 = \frac{1}{n-1}\Sigma_{i=1}^{n}\left(\theta_{ij} - \overline{\theta_j}\right)^2 \tag{7}$$

where $\overline{\overline{\theta}}$ is the mean over the chains of the posterior mean of each chain, s_j^2 is the variance of j-th chain, θ_{ij} is the posterior i-th draw of the j-th chain and $\overline{\theta_j}$ is the posterior mean for the j-th chain. Then the variance of the marginal posterior variance of θ distribution is estimated as a weighted average of W and B

$$\widehat{Var(\theta)} = \left(1 - \frac{1}{n}\right)W + \frac{1}{n}B \tag{8}$$

Assuming θ initialized to arbitrary starting points in each chain, $\widehat{Var(\theta)}$ will overestimate the true marginal posterior variance. W will tend to underestimate the within-chain variance early in the sampling run. However, in the limit as $n \to \infty$, both quantities will converge to the true variance of θ. In light of this, the Gelman-Rubin statistic monitors convergence using the ratio:

$$\hat{r} = \sqrt{\frac{\widehat{Var(\theta)}}{W}}, \tag{9}$$

the potential scale reduction. If it close to 1 (less than 1.1), it can be said that a particular estimand has converged.

3.4 Modeling: Bayesian A/B Test

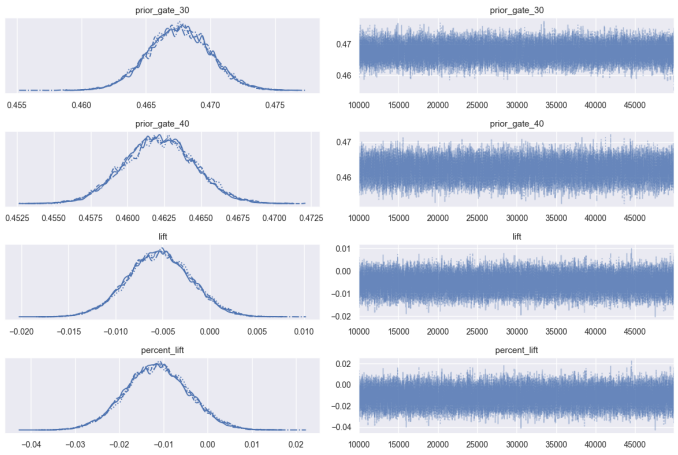
3.4.1 Retention after 1 day

Throughout the entire modeling phase a library called `pymc` has been used. `pymc` is a Python package for Bayesian statistical modeling and probabilistic machine learning which allows users to build models using intuitive, high-level syntax and provides tools for fitting those models to data using a variety of inference algorithms.

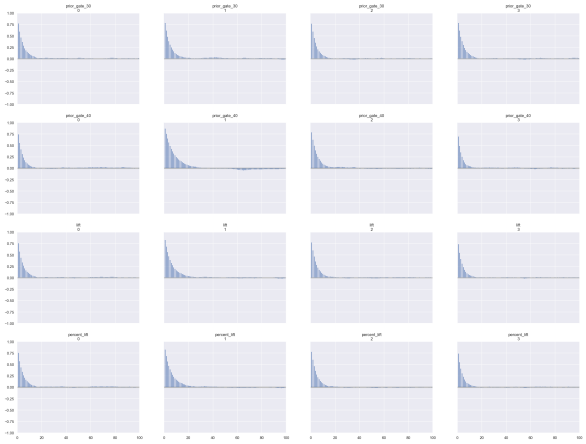
The first model used is called `retention_1`, which is a class of `pymc` library that provides a way to create a new Bayesian model. A Bayesian model is a statistical model that uses Bayes' theorem to update the probability of an hypothesis as more evidence or data becomes available. The model encapsulates:

- `prior_gate_30, prior_gate_40`: `pymc.Beta` class. These variables build two Beta distributions with specified parameters α and β;
- `lift`: `pymc.Deterministic` class. A deterministic variable is a variable that can be expressed as a function of other variables in the model, without any randomness or uncertainty. The role of these variables is to calculate additional model outputs that are not directly observed in the data, but are of interest to the modeler. By defining these variables (deterministic) as functions of the input variables, `pymc` automatically calculates their values during the posterior sampling process, and includes them in the resulting `trace`. In this case, the **lift** variable is computed as follows:
$$L = \mathbb{E}[\pi(\theta_B^1|\underline{x}_B)] - \mathbb{E}[\pi(\theta_A^1|\underline{x}_A)] \tag{10}$$
- `percent_lift`: `pymc.Deterministic` class. **Percent lift** is computed as follows:
$$PL = \frac{\mathbb{E}[\pi(\theta_B^1|\underline{x}_B)]}{\mathbb{E}[\pi(\theta_A^1|\underline{x}_A)]} - 1; \tag{11}$$
- `lik_gate_30, lik_gate_40`: `pymc.Binomial` class. These classes represent two binomial distributions. As stated above, this is the distribution that shapes the likelihood of the data, where the probability p draws values from the specified prior distribution while the observed values are the sum of people who returned to the game after 1 day;
- `trace`: `pymc.sample` method. This method is used to draw posterior samples from a Bayesian model (in this case `retention_1`). It performs MCMC sampling using one of the proposed algorithms. Particularly, since the conjugacy of the distributions has been proved above, the **Gibbs Sampler** is chosen.

The sampling phase had 50000 *draws* for 4 *chains*. The function `plot_trace` by `arviz` library plots the trace of the variables from a bayesian inference analysis of all its chains. The trace refers to a sequence of samples drawn by running a MCMC algorithm (in this case, Gibbs sampler). On Figure 5a we can see the function output: by looking at the right-hand side graph, for each variable, the algorithm converges towards a specific value, which is supposedly the expected value of each posterior distribution. An important step when it comes to sampling phase is checking for *autocorrelation*. By looking at its plot, we can assess the degree of autocorrelation within the MCMC drawn distribution. If there is high autocorrelation, it may be necessary to *thin* the samples or increase the number of iterations to reduce the correlation. However, this would come at the cost of an increase in the computational power and time. A broadly-used alternative could also be to create a *burned trace*. In Figure 5b we can see the `arviz.plot_autocorr` plot that outputs the autocorrelation graph for each variable for the first 100 lags. The acceptable degree of autocorrelation trace depends on the specific problem and the convergence criteria. In general, the goal is to have a trace that is well-mixed and has low autocorrelation, which indicates that the samples are independent and are efficiently exploring the target distribution. A commonly used rule of thumb is that the autocorrelation should be less than 0.1 for every lag beyond the first lag. This means that the autocorrelation should decrease rapidly and be negligible after a small number of lags. Moreover, a good fit should not display excessive "swings" on the posterior distribution's fit. Overall, the posterior draws do not show any excessive sign of autocorrelation and have a good fit.
In Figure 6, the four posterior plots are being displayed along with their respective **94% High Density Interval (HDI)**, which are plot in correspondence of the black line under each graph.

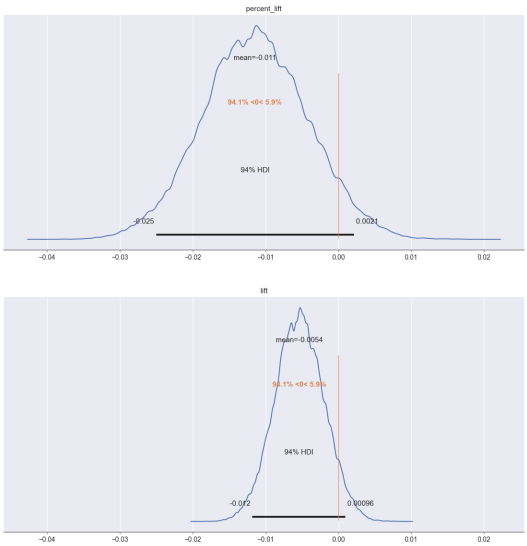


(a) `arviz.plot_trace` *output.*

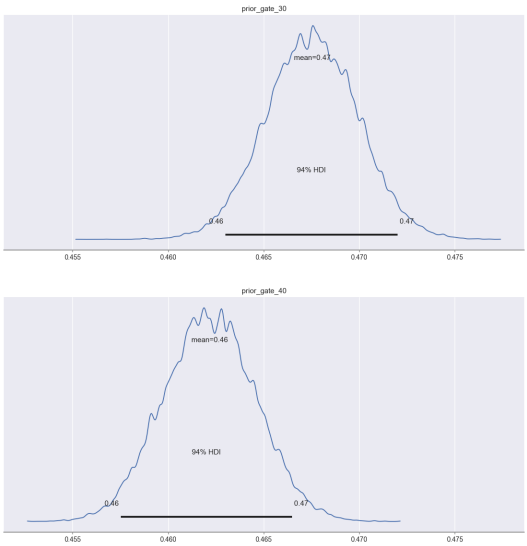


(b) `arviz.plot_autocorr` *function.*

Figure 5



(a) `arviz.plot_posterior` *output. Posterior Distribution of* ***Percent Lift*** *and* ***Lift*** *deterministic variables.*



(b) `arviz.plot_posterior` *function. Posterior Distribution of* $\pi(\theta_A^1|\underline{x}_A)$ *and* $\pi(\theta_B^1|\underline{x}_B)$.

Figure 6

3.4.2 Retention after 7 days

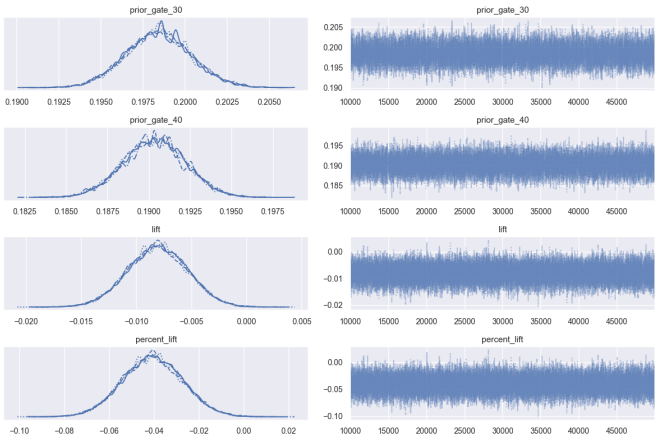
This subsubsection will obviously mirror subsubsection **3.4.1** premises, as both are part of the same A/B Test. This time, a new model called `retention_7` has been created and will copy the exact same instructions given to model `retention_1`. Within this model, the variables are essentially the same as `retention_1`'s ones except for a few changes: the binomial distribution that represents the likelihood of the data will this time consider observed value represented by the sum of people who returned after 7 days. The **lift** and **percent lift** formula will now differ. The **lift** will now be:

$$L = \mathbb{E}[\pi(\theta_B^7|\underline{x}_B)] - \mathbb{E}[\pi(\theta_A^7|\underline{x}_A)] \tag{12}$$

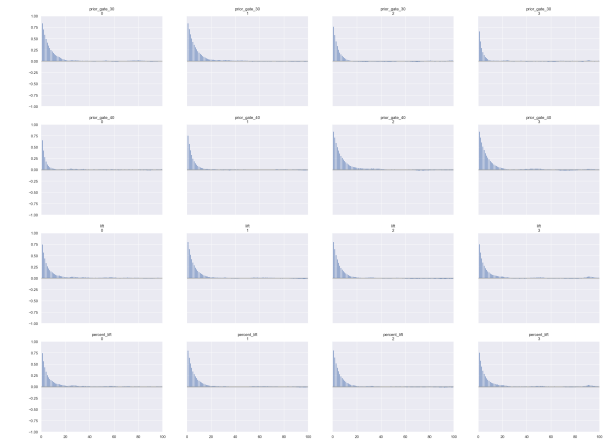
and the **percent lift**:

$$PL = \frac{\mathbb{E}[\pi(\theta_B^7|\underline{x}_B)]}{\mathbb{E}[\pi(\theta_A^7|\underline{x}_A)]} - 1 \tag{13}$$

Below, the plots for `retention_7`'s trace, its autocorrelation, and its sampled distribution.

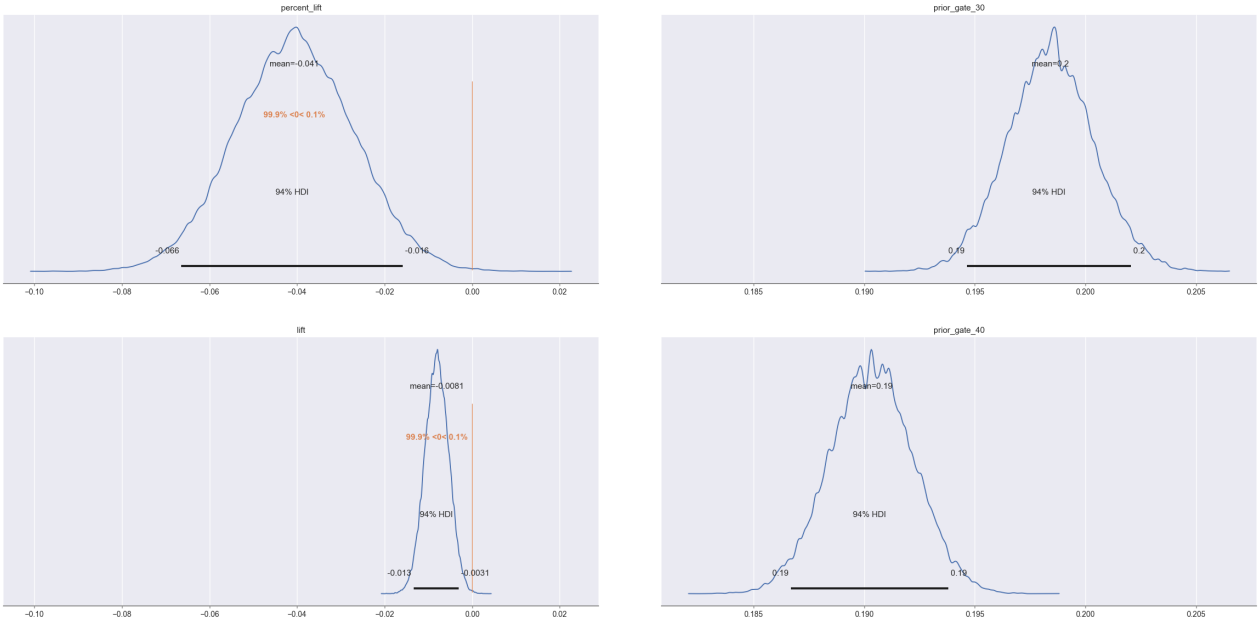


(a) `arviz.plot_trace` *output.*



(b) `arviz.plot_autocorr` *function.*

Figure 7



(a) `arviz.plot_posterior` *output. Posterior Distribution of* ***Percent Lift*** *and* ***Lift*** *deterministic variables.*

(b) `arviz.plot_posterior` *function. Posterior Distribution of* $\pi(\theta_A^7|\underline{x}_A)$ *and* $\pi(\theta_B^7|\underline{x}_B)$.

Figure 8

3.4.3 Results

The table below shows the summary statistics of the posterior distributions for `retention_1`'s model, which comprises the mean of the four variables, along with the standard deviation and their respective highest density intervals. Finally, the `r_hat` will indicate the potential scale reduction.

	mean	sd	hdi_3%	hdi_97%	r_hat
prior_gate_30	0.468	0.002	0.463	0.472	1.002
prior_gate_40	0.462	0.002	0.458	0.467	1.002
lift	-0.005	0.003	-0.012	0.001	1.003
percent_lift	-0.011	0.007	-0.025	0.002	1.002

By definition, a $100(1-\alpha)\%$ highest density interval (**HDI**) for a parameter θ is defined as the region:

$$C\alpha = \theta : \pi(\theta|\underline{x}) \geq \gamma \tag{14}$$

where γ is chosen so that $Pr(\theta \in C_\alpha|\underline{x}) = 1-\alpha$.

Given $\alpha = 0.06$, for the two estimated *Beta posteriors*, there is a 94% probability that the estimated parameters (given the data observed in each respective group) lie within the region delimited by the lower bound value `hdi_3%` and upper bound value `hdi_97%`. The *central* values *are expected* to coincide with their respective mean values. Nevertheless, in order to determine which of the groups better respond during the test, the variable of interests `lift` and `percent_lift` are to be taken into consideration along with its statistics: by observing its mean value, the answer of the question of interest can be obtained. Lift's expected value is -0.005, which means that:

$$\mathbb{E}[\pi(\theta_B^1|\underline{x}_B)] < \mathbb{E}[\pi(\theta_A^1|\underline{x}_A)] \tag{15}$$

given how the `lift` variable has been previously defined.

Following, the mean value of **PL**, the percent lift, accounts for the percentage difference of the two beta posterior's distribution's mean value, *ie.* $\mathbb{E}[\pi(\theta_B^1|\underline{x}_A)]$ and $\mathbb{E}[\pi(\theta_A^1|\underline{x}_B)]$, which, as expected, tuned out to be negative. Its mean value is therefore -0.011. Given the results obtained, it seems that users who played the 40th gate-version of the game (group B) were 1% less likely to come back after 1 day of having installed the game compared to group A.
In conclusion, it is important to remark, however, that these results are obtained given a $Beta(1,1)$ prior, meaning that there is no prior knowledge about the considered event. The results would have certainly been different if it had been considered a different prior scenario. Not only, one has also to take into account the presence, although small, of standard deviation within these estimations. The differences between the two experiments is not large. Therefore, it would be wrong to conclude with extreme evidence the *superiority* of one group over the other.

As the two models mirror each other, similar conclusions can be drawn for `retention_7`'s model. Below, its summary statistics:

	mean	sd	hdi_3%	hdi_97%	r_hat
prior_gate_30	0.198	0.002	0.195	0.202	1.002
prior_gate_40	0.190	0.002	0.187	0.194	1.003
lift	-0.008	0.003	-0.013	-0.003	1.001
percent_lift	-0.041	0.013	-0.066	-0.016	1.003

It can be stated that, once again, the original version of the game, the 30th-gate version of the game, represented by group A, grants an higher rate of retention after 7 days. Users who play the 40th gate-version of the game (group B) are 4% less likely to come back after 7 days of having installed the game, compared to their peers of group A.

Concluding, in both tables above, the `r_hat` is on average 1, meaning that all the variables successfully converge towards their expected posterior distributions.

4 Bayesian Logistic Regression

4.1 Logistic regression model

The goal of this second model, namely the Bayesian Logistic Regression[7],[10] is to predict whether or not there will be retention. The outcome is a binary categorical response variable converted to $0-1$.

$$y_i = \begin{cases} 1 & \text{if there is } \texttt{retention} \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

where i indicates the $i-th$ player. So the response variable can be considered as a Bernoulli random variable and, for the Bernoulli probability distribution, $Pr(y_i = 1) = \pi_i$ whereas $Pr(y_i = 0) = 1 - \pi_i$:

$$y_i|\pi_i \sim Bern(\pi_i) \tag{17}$$

where the expected value $E(yi|\pi_i) = \pi_i$.
For the analysis, only one predictor X_i is considered, which is the variable `sum_gamerounds`, the number of rounds played by each player i. In order to write the mean of the Bernoulli variable as a linear function of the predictor, let's consider a function g$(\cdot)$:

$$g(\pi_i) = \beta_0 + \beta_1 X_i. \tag{18}$$

The right side term spans from $-\infty$ to $+\infty$ so g$(\cdot)$ must be a function that also spans the entire real line. That's why g$(\cdot)$ is choosen as:

$$g(\pi_i) = logit(\pi_i) = log\left(\frac{\pi_i}{(1-\pi_i))}\right) \tag{19}$$

where $log\left(\frac{\pi_i}{(1-\pi_i)}\right)$ is the logit link function and $\frac{\pi_i}{(1-\pi_i)}$ is the odds of π_i, the ratio of a probability to its complement.
Putting 18 and 19 together, the Bayesian Logistic model is obtained:

$$logit(\pi_i) = log\left(\frac{\pi_i}{(1-\pi_i))}\right) = \beta_0 + \beta_1 X_i. \tag{20}$$

Taking the exponential of the both-side terms:

$$\frac{\pi_i}{(1-\pi_i)} = e^{\beta_0+\beta_1 X_i} \tag{21}$$

the Logistic Response function is finally derived:

$$\pi_i = \frac{e^{\beta_0+\beta_1 X_i}}{1+e^{\beta_0+\beta_1 X_i}} = \frac{1}{1+e^{-(\beta_0+\beta_1 X_i)}} \tag{22}$$

When $X_i = 0$, β_0 is the log odds of the event of interest and e^{β_0} is the odds.
If $X_i = x$ (the value of the predictor is fixed), when it increases by 1, from x to $x+1$, β_1 represents the change in log odds, $\beta_1 = log(odds_{x+1}) - log(odds_x)$, and e^{β_1} is the multiplicative change in odds, $e^{\beta_1} = \frac{odds_{x+1}}{odds_x}$: given

$$logg(odds_x) = \beta_0 + \beta_1 x, \qquad log(odds_{x+1}) = \beta_0 + \beta_1(x+1), \tag{23}$$

it follows

$$log(odds_{x+1}) - log(odds_x) = \beta_1(x+1) - \beta_1 x = \beta_1. \tag{24}$$

4.2 Bayesian Logistic Regression

In the Bayesian approach,[5] the prior beliefs are combined with the data to model y_i. The parameter β_0 and β_1 can assume every real values, so Normal prior distributions are appropriate for them. In this way the prior distribution becomes:

$$\pi(\underline{\beta}) = \prod_{j=0}^{1} \pi(\beta_j) = \prod_{j=0}^{1} \frac{1}{\sqrt{2\pi\sigma_j^2}} exp\left(\frac{-1(\beta_j-\mu_j)^2}{2\sigma_j^2}\right). \tag{25}$$

The likelihood function is defined as:

$$f(\underline{y}|\underline{\beta}) = \prod_{i=1}^{n} f_{y_i}(y_i|\underline{\beta}) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{y_i} \tag{26}$$

and the posterior is given by the product, so:

$$\pi(\underline{\beta}|\underline{y}) \propto f(\underline{y}|\underline{\beta})\pi(\underline{\beta}). \tag{27}$$

This expression is not a closed form, since it is not a particular distribution. For this reason, MCMC simulation with Metropolis-Hastings will be used to estimate the posterior distributions of the parameters.

4.3 Modeling: Bayesian Logistic Regression

The following section will try to give a Bayesian interpretation over the magnitude that the estimated coefficient of the variable `sum_gamerounds` has when it comes to predicting whether a player will return after 1 week, ie. if the independent variable `sum_gamerounds` is a *good proxy* for predicting `retention_7` dependent variable. As stated above, the dataframe named `log_df` is used. Recall that this dataframe has been edited according to the reason specified in 2.2.

Once again, the library `pymc` is used. A `pymc.Model` named `logistic`, which formalises mathematically in this way, is deployed:

$$Pr(\texttt{retention\_7} = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \texttt{sum\_gamerounds})}} \tag{28}$$

Here an overview of its variables.

- `prior_beta_0, prior_beta_1`: `pymc.Normal` classes. Once again, there is no past knowledge (ie. *uninformative prior believes*) about any behaviour of the β. Therefore, given the fact that the regression coefficients *usually* assume values that are in the neighbour of 0, the following assumption can be made:

$$\beta_0, \beta_1 \sim \mathcal{N}(0,\, 10)$$

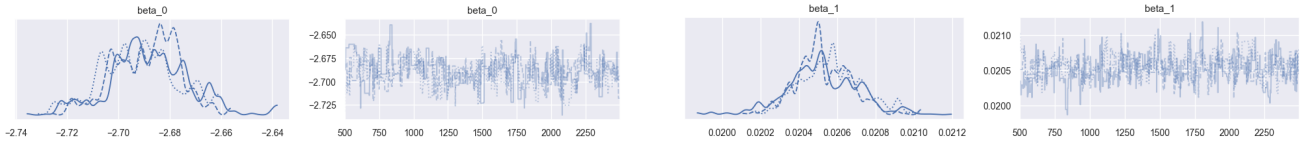
- `p`: `pymc.Deterministic` class. `p` is a deterministic variable that represents the probability of success in a logistic regression model. The logistic regression model is typically used to model the probability of a binary outcome as a function (*sigmoidal* function) of one or more predictor variables. In this case, `p` is the probability of success as a function of the predictor variable $\beta_1 \cdot$ `sum_gamerounds` plus a certain coefficient β_0 which can be seen as intercept. The `pymc.math.sigmoid` function is used to map the linear combination of the predictor variable and the coefficients (i.e., `prior_beta_0` and `prior_beta_1`) onto the interval [0,1], which is the range of possible probabilities.

- `posterior`: `pymc.Binomial` class. `posterior` is a `pymc.Bernoulli` distribution that models the observed data `retention_7` as a binary outcome with probability `p`. The observed parameter in `pymc.Bernoulli` specifies the observed data, which in this case is the retention status of players after 7 days, and the `p` parameter is the probability of retention calculated from the logistic regression model `p`. By including posterior in the `logistic` model, the model can be used to estimate the posterior distribution of the parameters given the observed data.

A Bayesian logistic regression model is defined using the `pymc.Model()` context manager. By considering `logistic` through the "`with`" `python` command, `pymc` receives as input a `model` and samples 4 `chains` that comprises 2500 observations through the MCMC `Metropolis` algorithm. The number of chains (as well as the number of samples) used in an MCMC simulation can have an impact on the accuracy and reliability of the results. In general, it is recommended to use at least 2 chains and a couple of thousands samples in an MCMC simulation. However, due to computational expensiveness of the Metropolis algorithm, it has been decided to choose such low number samples: this might lead to a non-optimal fit of the desired posterior distribution.

4.3.1 Diagnostics

In the next section, several diagnostic plots are presented, including an attempted plot of the desired posterior distributions of the β_0 and β_1 parameters. While there is some sample autocorrelation within each chain (Figure 9a) and the estimate plot may not be as clear as in previous sections, the potential scale reduction factor (`r_hat`) for each variable is close to one, indicating that the variables converge towards certain values.



(a) `arviz.plot_trace of` β_0 (b) `arviz.plot_trace` of β_1

Figure 9

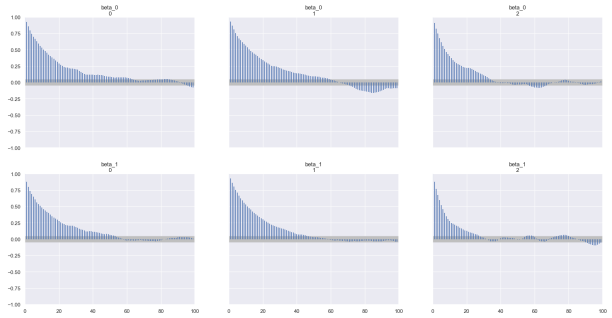


Figure 10: `arviz.plot_autocorr autocorrelation plot`

4.3.2 Results

Given the results obtained,

	mean	sd	hdi_3%	hdi_97%	r_hat
beta_0	-2.690	0.014	-2.719	-2.664	1.02
beta_1	0.021	0.000	0.020	0.021	1.02

it can be affirmed that:

- Considering[8] $\beta_1 = 0$, $\beta_0 = -2.69$, is the **log-odds**. Thus, $e^{\beta_0} = 0.068$ is the odds. From it, π_i can be calculated as $\pi_i = \frac{odds}{1+odds} = 0.064 \ \ \forall \ \ i$. So the probability that the event occurs, regardless `sum_gamerounds` being considered, is approximately $6,36\%$.
- Considering $\beta_1 = 0.02$, for a one-unit increase in the predictor variable, `sum_gamerounds`, the **log(odds)** increases by 0.02. Through exponentiating, the multiplicative change in odds for a one-unit increase in the predictor variable can be obtained, which is 1.02. So considering β_1, it can be said that whenever the predictor $X_i = x$, when $x + 1$ within the predictor's scale, then $logit(\pi)$ goes up by an amount $\beta_1 = 2,1\%$.

5 Conclusion

The goal of this paper was to perform a Bayesian analysis on the game Cookie Cats in order to test whether moving the first gate from level 30 to level 40 would improve player retention and to investigate whether an increase in the number of rounds played by a player would lead to a return after one week on the game platform. To answer the first question, a Bayesian A/B test was conducted, with an uninformative prior established and a likelihood modeled for both variants. By analyzing the lift and percent lift, it was concluded that players who downloaded the version of the game with the gate set at level 40th were 1% and 4% less likely to return to the platform after one day and seven days, respectively, than those who downloaded the level 30 version. This could led the developers to conclude that , despite the magnitude of the evidence being little, no change has to be made to the game. The A/B test was conducted using an MCMC algorithm, and because the posterior distribution was known, a Gibbs sampler was used. In the second part of the project, Bayesian logistic regression was used to investigate whether users who played more rounds were more likely to return to the platform after one week. The coefficient of interest, β_1, was estimated, indicating that an increase of one unit in the sum of rounds played by a player led to a 2.1% increase in the user's logarithmic odds of returning to the platform after one week. This time, given the fact that the posterior distribution was of an unknown form, the Metropolis-Hastings algorithm has been chosen.

References

[1] Bayesian ab testing. https://making.lyst.com/2014/05/10/bayesian-ab-testing/.

[2] Bayesian ab testing. https://towardsdatascience.com/bayesian-ab-testing-ed45cc8c964d/.

[3] Bayesian vs. frequentist a/b testing: What's the difference? https://cxl.com/blog/bayesian-frequentist-ab-testing/.

[4] Model checking and diagnostics. https://pymcmc.readthedocs.io/en/latest/modelchecking.html.

[5] Najla A Al-Khairullah and Tasnim HK Al-Baldawi. Bayesian computational methods of the logistic regression model. In *Journal of Physics: Conference Series*, volume 1804, page 012073. IOP Publishing, 2021.

[6] Hal S. Stern David B. Dunson Aki Vehtari Donald B. Rubin Andrew Gelman, John B. Carlin. *Bayesian Data Analysis*. 2013.

[7] Alicia A Johnson, Miles Q Ott, and Mine Dogucu. *Bayes rules!: an introduction to applied Bayesian modeling*. CRC Press, 2022.

[8] AS Kurz. Doing bayesian data analysis in brms and the tidyverse, 2021.

[9] Rossini Luca. Bayesian analysis. Department of Economics, Management and Quantitative Methods, University of Milan, March 2023.

[10] PA Lukman, S Abdullah, and A Rachman. Bayesian logistic regression and its application for hypothyroid prediction in post-radiation nasopharyngeal cancer patients. In *Journal of Physics: Conference Series*, volume 1725, page 012010. IOP Publishing, 2021.