**Multi-armed bandit**

The multi-armed bandit problem is one of the most basic problems in decision science. Namely, this is the problem of the optimal allocation of resources under uncertainty. The very name "multi-armed bandit" came from the old slot machines, which were controlled with pens. These machines were nicknamed "bandits" because after interacting with them, people usually felt robbed. Now imagine that there are several such machines and each of them has a different chance of winning. Since we're going to play with these machines, we want to determine which one of them has the highest chance and exploit it more than the others.

**The problem is this:** how can we most effectively figure out which one is the best for us? This is not some theoretical problem, this is a problem that business and companies face every day. For example, a company has several options for messages that need to be shown to users (messages, for example, include advertising, websites, images) so that the selected messages maximize a certain business task, **conversion**.

**The conversion** is calculated as a percentage of the total number of visitors who completed an action. An action can be filling out a form on the landing page, making a purchase in an online store, registering, subscribing to news, clicking on a link and many other actions.

**Naive approaches**

A typical way to solve this problem is to run A/B tests.

This test has the following form: we take all our variants (banners, for example) and test each of them for the same amount of time. For example, if there are 2 banners and 100 tests, then each banner will be tested 50 times. At the end of the experiment, judging by the statistics of successes and failures, we will be able to choose the best banner. This method is suitable when there are few options, perhaps 2 or 4. But when there are many options, this approach becomes inefficient – both in terms of **lost time** and **lost profits**.

**Such happens because:** A/B tests should not be interrupted until they have finished. This means that the experimenter does not know which option is better until the testing is over. This means that by extending A/B tests, **for a big amount of visitors** we publish **not the best banner, the ineffective ones** thereby **losing our profit**.

So It would be better if you could quickly discard bad options in reality, and only then, when there are few options, use A / B tests for them.

Exploration-exploitation balance

Thus, we come to understanding **that if we want to find the optimal strategy** we must strike a balance between exploring the space of variants (banners, bandits, options) and using the best of them.

- The space of options must be explored in order to have an idea of ​​which option is optimal. If we first discovered the best option, and then use it all the time, we maximize the total reward.

- On the other hand, we also want to explore other possible options - what if they turn out to be better in the future, but we just don't know yet? In other words, we want to hedge against potential losses by experimenting a little with sub-optimal options in order to clarify for ourselves their payback. And if their payback is actually higher, they can be used more often.

The main problem, therefore, is to decide how to get out of the dilemma between exploration and exploitation the best way (exploration-exploitation tradeoff).

Epsilon-greedy algorithm

A typical way to get out of this dilemma is to use the epsilon-greedy algorithm.

"Greedy" means exactly what you think. An algorithm is "greedy" insofar as it chooses immediate profit without considering the long-term outcome.

After some initial period where we randomly try every variant, on every iteration algorithm starts to choose between the best option and random one from the others. And, for example, if epsilon = 5%, the algorithm chooses the best option 95% of the time.

**In fact, this is a fairly efficient algorithm, however, it may not explore the space of options well enough**, and therefore it can get stuck on a **suboptimal option**.

**Thompson sampling**

Thompson Sampling rather than just trying to calculate an estimate of the mean reward for any bandit(or banner in our case) it instead builds **up a probability model from the obtained rewards**, and then samples from this to choose an action.

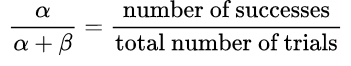
In this way, for every bandit algorithm uses not only increasingly accurate estimate of the possible probability of success, but the whole **probability distribution model** which also provides a level of confidence in this reward, and this confidence increases as more samples are collected.

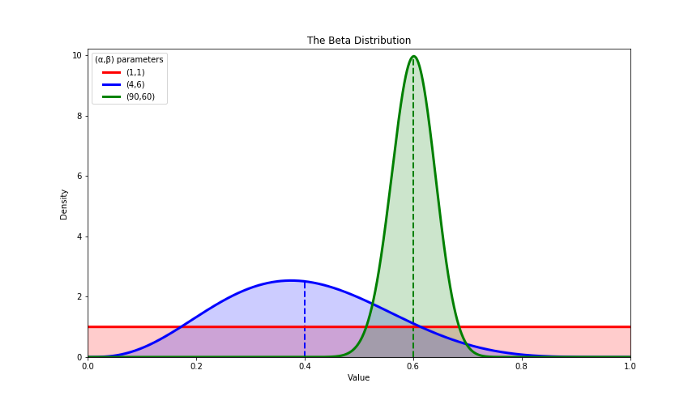
Bernoulli Thompson Sampling andBeta-distribution

Our banners problem has a such structure: each banner if we use it either will have a conversion (success) or won’t have it so the reward have only **two possible values**: 1 and 0. When a random variable has only two possible outcomes its behavior can be described by the [**Bernoulli distribution**](https://en.wikipedia.org/wiki/Bernoulli_distribution).

As already mentioned, Thompson Sampling generates a model of the reward probabilities. When, as in this case, the available rewards are binary (win or lose, yes or no) then the [**Beta distribution**](https://en.wikipedia.org/wiki/Beta_distribution) is ideal to model this type of probability.

The Beta distribution takes two parameters, ‘*α*’ (alpha) and ‘*β*’ (beta) which can be thought of as respectively the count of successes and failures. Additionally, a Beta distribution has a **mean value** given by:





The Beta distribution for various values of alpha and beta.

Initially we can start by setting both ‘*α*’ and ‘*β*’ to one, which produces a flat line [**Uniform distribution**](https://en.wikipedia.org/wiki/Uniform_distribution_(continuous))(shown as the flat, red, line).

This initial guess at the probability of the banner producing a conversion is known as the [***Prior Probability***](https://en.wikipedia.org/wiki/Prior_probability); it is the probability of the specific event occurring before we have collected any evidence and in this case is represented by the Beta distribution *Beta(1,1)*.

Once we test a banner, and obtain a reward, we can modify our belief in the likelihood of success for it. This new probability, after some evidence has been collected, is known as the [***Posterior Probability***](https://en.wikipedia.org/wiki/Posterior_probability). Again this is given by a Beta distribution, but now the values of ‘*α*’ and ‘*β*’ are updated with the value of the returned reward.

As more data is collected the Beta distribution moves from being a flat line to become an increasingly accurate model of the probability of the mean reward. By maintaining the values of ‘*α*’ and ‘*β*’ a **Thompson sampling algorithm** is able to describe the estimated mean reward and the level of confidence in this estimate.

In contrast to the Greedy algorithm, which at each time step selects the action with the highest estimated reward, even if the confidence in that estimate is low, **Thompson sampling instead samples from the Beta** distribution of each action and chooses the action with the highest returned value. Since actions that have been tried infrequently have wide distributions (see the blue curve in *figure*), they have a larger range of possible values. In this way, a banner that currently has a low estimated mean reward, but has been tested fewer times than a banner with a higher estimated mean, can return a larger sample value and therefore may be selected at this time step.

In the graph above, the blue curve has a lower estimated mean reward than the green curve. Therefore, **under Greedy selection**, green would be chosen and the blue banner would never be selected. In contrast, **Thompson Sampling** effectively considers the full width of the curve, which for the blue banner can be seen to extend beyond that of the green one. In this case the blue one may be selected in preference to the green.

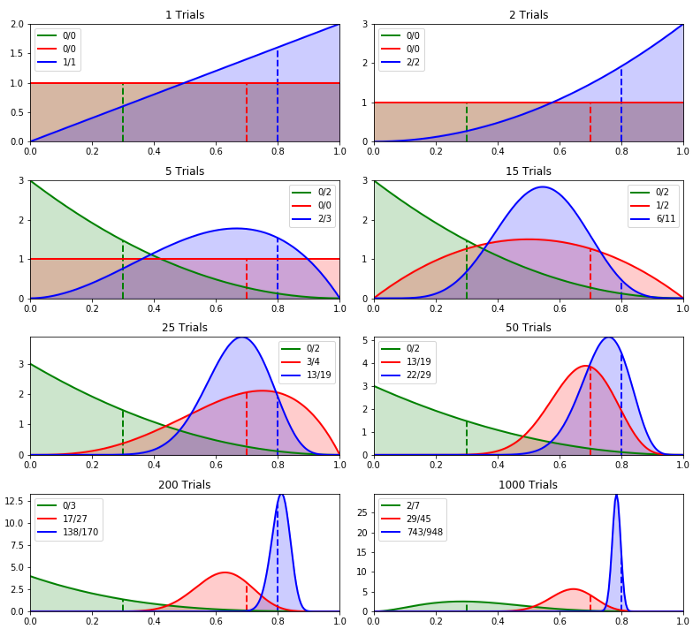
As the number of trials of a banner increases the confidence in the estimated mean increases, the probability distribution becomes narrower and the sampled value will then be drawn from a range of values that are closer to the true mean (see the green curve in *figure*). As a result, **exploration decreases and exploitation increases**.

On the other hand, banners with a low estimated mean will start to be selected less frequently and will tend to be dropped early from the selection process. *Since we are only interested in finding the best banner as quickly as possible, we don’t care about poorly performing banners.*

Experiments results

The evolution of the Beta distribution for each banner can be seen in *Figure*.

To keep things simple, we’ve reduced the number of banners to three and these have true probabilities 0.3 (green), 0.7(red) and 0.8 (blue) of success when tested.



In *Figure* above, : Thompson sampling using Beta distributions for 3 probabilistic banners, with true probabilities of 0.3, 0.7 and 0.8. The true means are shown by the dashed lines. The legend displays the number of trials for each socket and the number of successes that have resulted from these trials.

Curve – дуга, характеристика, кривая

The main points to note from *Figure* are the following:

* At time step 0 all Beta distributions give a flat Uniform distribution.
* Since all banners have equal distribution, at time step 1 the blue one is selected randomly. When tested it gives a reward, so its ***α***value gets incremented by 1 and its probability density curve shifts to the right.
* At the second time step the blue banner is again selected and again it returns a reward. The blue curve squeezes slightly more to the right
* By the 5th trial the blue one has been selected once more, but this time it failed to give a reward. As a result the probability that it always returns a value drops to zero (at probability = 1.0). On the other hand, the green banner has now been tested twice and is yet to return a value, hence its probability density curve is shifted to the left with its highest value at probability =0, as there’s still a chance this one never returns a reward.
* At 15 trials the red banner has now been tried a couple of times. Since it’s returned a reward once, it has an estimated mean reward probability of 0.5. At this stage the blue banner has been tried 11 times and has returned a reward on 6 of these trials, giving it a slightly higher estimated reward probability of 0.54. **In a Greedy system the blue banner** would therefore be the chosen one, however because the red banner has been tried less times than the blue one, it can be seen to have a much wider probability density curve, giving it a good chance of being selected in preference to the blue one.
* The more times a banner is tested, the more confident we are in its estimate and the narrower its probability density curve becomes. **The best one will then be used more** often and testing of the sub**-optimal variants will tail off**. This behavior can be seen at the end of our test, when the blue banner has been tried much more often than others.

**Comparison of methods**