OfferFit

Make the Leap from A/B to AI

A Community of Bandits

How OfferFit automates experimentation for lifecycle marketers

George Khachatryan, Ph.D.
Co-Founder & CEO

Victor Kostyuk, Ph.D.
Co-Founder & CTO

Nathaniel Rounds, Ph.D.
Product Marketer

OfferFit
June 2023

Contents

Introduction

AI breakthroughs are changing marketing. Generative AI will help marketers create content and content variants, but marketers still need to experiment and learn what works with actual customers. Traditionally marketers have tested their ideas with **A/B testing**, but A/B testing is slow and doesn't scale. A different kind of AI called **reinforcement learning**, is helping marketers automate and accelerate the process of experimentation.

OfferFit lets lifecycle marketers unlock the value of their first-party data to make 1:1 decisions for each individual customer. In this white paper, we look "under the hood" of OfferFit's AI, which replaces manual A/B testing with reinforcement learning, and we explain in detail how OfferFit's **community of bandits** helps lifecycle marketers automate experimentation.

The problem with lifecycle marketing

No business can survive without new customers, but now's not the time to overlook retention. Customers worried about inflation or recession are looking for products to prune or subscriptions to cut. It may seem difficult to devote scarce resources to existing customers when the need for acquisition looms, but good retention marketing increases the value of each new customer. As marketers monetize the customers they have, their business case for acquisition spend grows stronger. The greater the CLV (customer lifetime value), the more valuable each marginal customer becomes. Higher CAC (cost to acquire a customer) is sustainable if marketers are confident in the value of that customer. To put it plainly: there's no point in pouring water into a leaky bucket. That means investing in lifecycle marketing to existing, identified customers.

The good news is that marketers have everything they need to engage with their existing identified customers.

- **Comms to existing customers are cheap**. Marketers own the channels (email, direct mail, SMS, push notifications, etc.) for existing customers, whereas they have to pay for ads or SEM to reach new prospects.
- **Data on existing customers is rich**. Marketers have a wealth of first-party data on their identified customers – what they've bought, how they've engaged in-app or interacted with emails.

So in some sense, retention marketing should be easy. All a marketer needs to do is send each customer the perfect message, with the perfect offer, through the perfect channel, with the optimal frequency to maximize revenue, purchases, or whatever metric matters most to the business. So why does this story sound more like a fairy tale?

In practice, there are three barriers to a lifecycle marketer's dream approaching reality.

1. The **data integration bottleneck**. In theory, a business' rich first-party data powers lifecycle marketing. In practice, data might be incomplete, jumbled, scattered, or disconnected from the platforms marketers use to orchestrate campaigns and communicate with customers.
2. The **content creation bottleneck**. To leverage the value of their customer data, marketers need to craft messages, subject lines, offers, and creative that will resonate with each individual. But making lots of variants takes time!
3. The **experimentation bottleneck**. Even if marketers have content, and have data to suggest which content and offers would best resonate with each customer, they still need to test and learn what works in practice. Traditionally marketers have tested their ideas using A/B testing, but A/B testing is slow and doesn't scale.

But AI breakthroughs are changing the game. **Generative AI** built on **large language models (LLMs)**, such as Google's Bard and OpenAI's ChatGPT, are poised to solve the content creation bottleneck, and help with the **data integration** bottleneck as well. New tools are already being built and released which will automate data transformations and build and maintain pipelines between systems. However, generative AI is unlikely to be much help with the **experimentation bottleneck** – marketers will still need to test and learn what works with actual customers. To push through this bottleneck, savvy marketers are relying on a different kind of AI called **reinforcement learning** to automate the process of experimentation.

Automated Experimentation

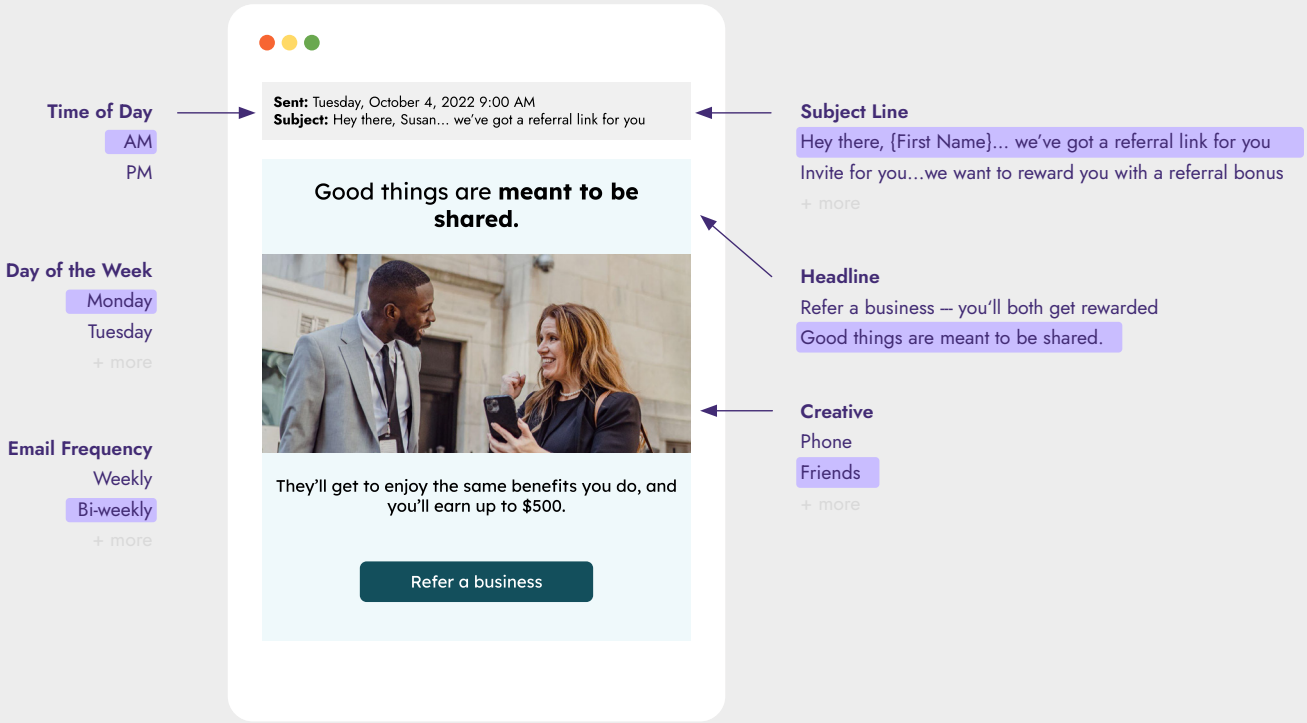
Lifecycle marketers know that when it comes to engaging their customers, practice beats theory. No matter how brilliant a campaign, email subject, ad, or creative seems on paper, the only way to know if an idea works is to try it out with customers. Traditionally, marketers have tried out their ideas through A/B testing, but A/B testing is slow and doesn't scale with the number of variables marketers need to test. OfferFit's **reinforcement learning** platform allows marketers to automate the process of experimentation.

Marketers use OfferFit to optimize campaigns to existing identified customers, with business goals such as cross-sell, upsell, repurchase, retention, renewal, referral, and winback. To implement OfferFit, marketers:

1. Choose a **success metric** they want to maximize, such as revenue, conversions, ARPU, or any other KPI they can measure from their data. This is the metric OfferFit's AI will try to maximize.

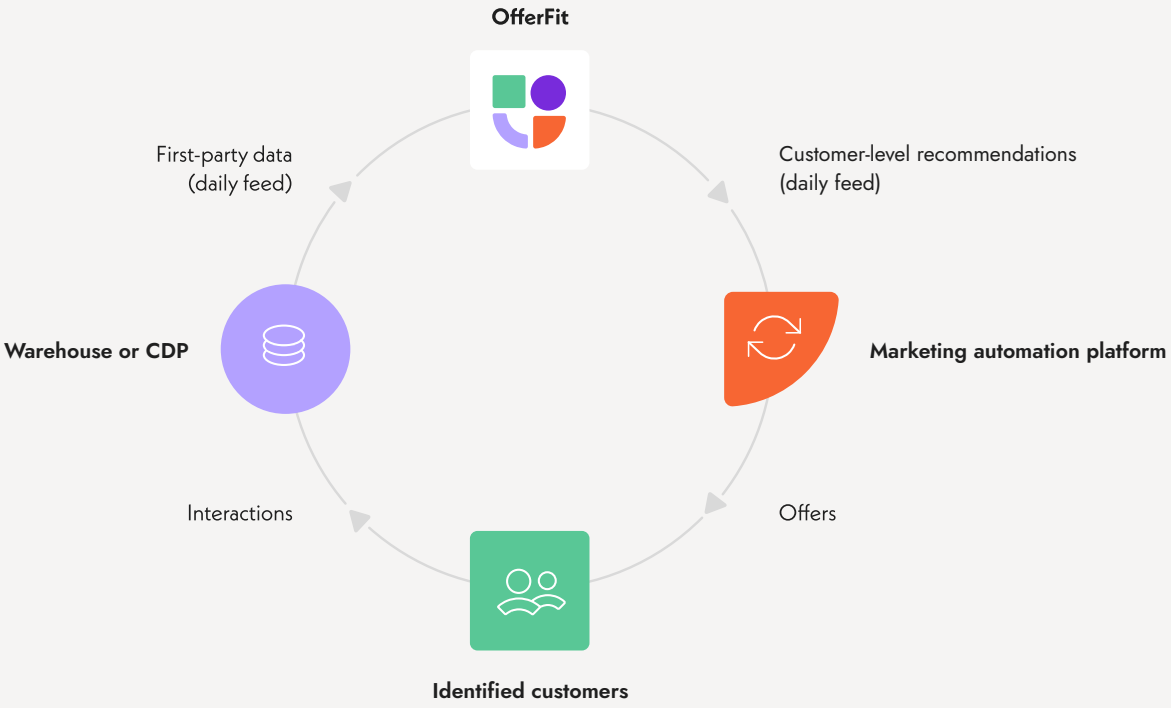
2. Choose **dimensions** along which to test (e.g., offer, subject line, creative, channel, time, day, frequency, etc.)

3. Finally, choose the **options** available for each dimension. For example the options for the "channel" dimension might be email, SMS and push; while the options for the "frequency" dimension might be daily, twice a week, and weekly.



A bank uses OfferFit to maximize business credit card referrals. Every day, OfferFit's AI chooses a subject line, headline, creative, time of day, day of the week, and frequency for each individual customer.

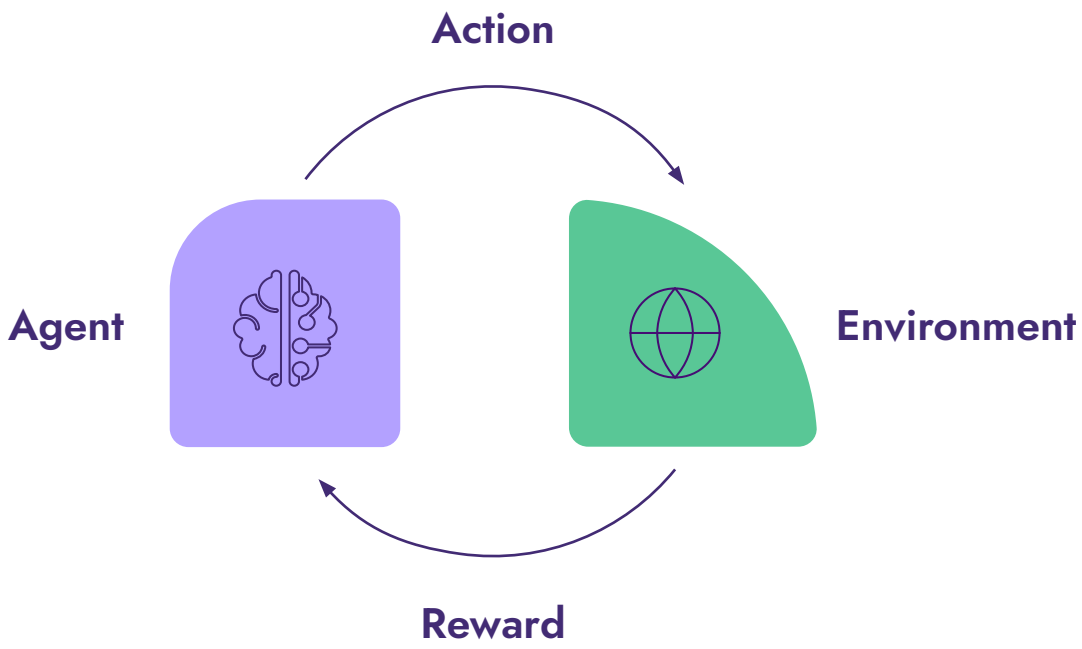
OfferFit's AI then automates the process of experimentation. OfferFit makes daily recommendations for each customer, seeking to maximize the chosen success metric. The AI learns from every customer interaction, and applies these insights to the next day's recommendations.



The rest of this paper will be devoted to a detailed explanation of how OfferFit's AI learns and how the AI makes decisions for each individual customer.

Reinforcement learning

OfferFit's uses a type of machine learning called **reinforcement learning**, or **self-learning AI**. A reinforcement learning model tries to learn from its environment — it iteratively chooses from a set of **actions**, receives some **reward** in response, and then chooses another action based on what it's learned.[1] As the model learns from the rewards it receives, it updates its policy for selecting future actions with the goal of maximizing the reward achieved over time.



In the case of OfferFit's AI, the **actions** are the set of choices for each dimension. For example, the actions for the "day of the week" dimension might simply be the days of the week, while the actions for the "creative" dimension might be a bank of dozens or even hundreds of emails. The **reward** the agents receive is determined by the success metric chosen by the marketer implementing OfferFit. The reward for a renewal campaign, for example, might be the marginal ARPU generated from a new contract. Rewards can also take into account multiple metrics, for example the contract length as well the marginal ARPU, and negative events, say adding a penalty for unsubscribes.

[1] Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Reinforcement learning models, which experiment and learn as they go, are well-suited for making customer-level decisions in marketing. As theory meets practice, however, several challenges emerge.

1. **The set of possible choices is large**. Suppose a marketer running a renewal campaign has 4 plans, each with 5 corresponding creative, which could be sent on 5 possible days of the week, at 4 possible times of day, with 3 different possible frequencies. This relatively simple situation results in 1200 possibilities per customer! This set of possibilities, the **action space**, quickly becomes large in practical marketing applications.

2. **AI models need to learn fast**. An AI can certainly explore a large action space. In fact, the earliest applications of reinforcement learning include teaching AIs to play games like chess and go, which famously have a vast space of possible moves. But an AI learning chess can play itself billions of times to generate data. Marketers have no such luxury – they can't afford to rack up unsubscribes while an AI learns. Marketers need their models to be **sample efficient** – to use data efficiently and learn quickly from a small sample.

3. **Rich data means the signal can overpower the noise**. When it comes time to train a reinforcement learning model, marketers' rich first-party data can be both a blessing and a curse. Suppose a marketer sends out 1000 emails, and 10 customers convert. Marketers have hundreds of data points about each customer – everything from last purchase to zip code. Which of those data points is most likely to explain the conversion? A model which tries too hard to fit the model to preliminary data will be unlikely to generalize and make good predictions in the future. This danger is what data scientists call **overfitting**. (In the US we see examples of overfitting every presidential election. Someone always trots out a heuristic that perfectly predicts the last 20 presidential elections – perhaps using the rate of change of the unemployment rate in October, or who leads in Ohio – but which is no better than a coin flip at predicting the next winner.)

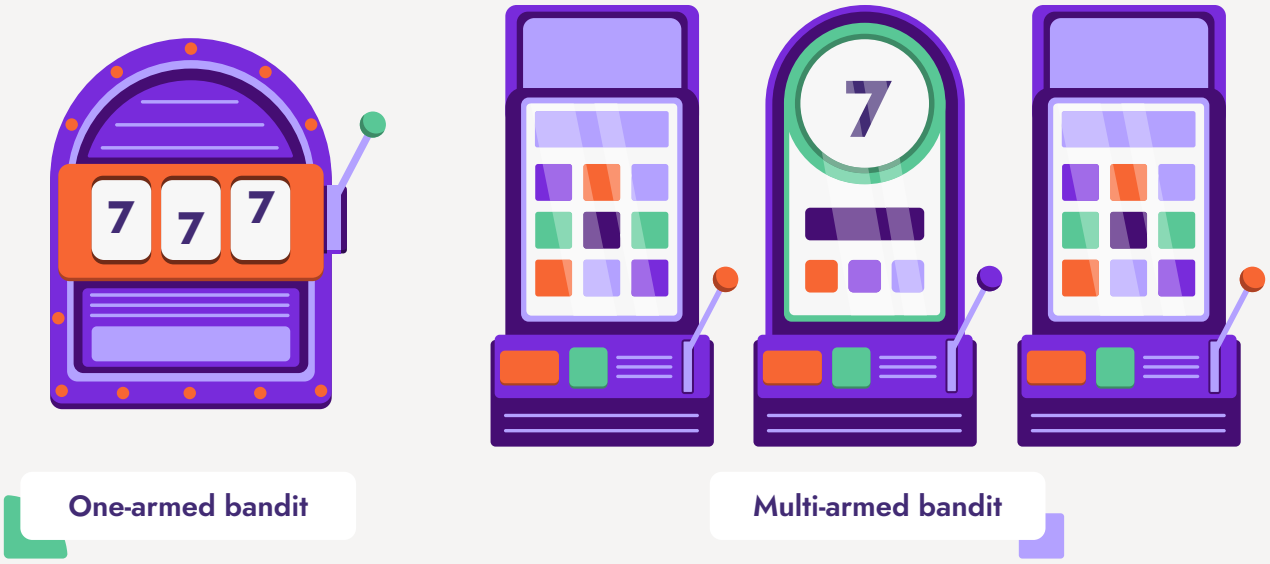
To be useful to marketers, a reinforcement learning AI must quickly and efficiently explore a large action space, and separate the signal from the noise. OfferFit's AI threads this needle with our proprietary **community of bandits**. This collection of reinforcement learning models, called bandits, collaborate with each other to quickly learn the best choice for each individual customer. To understand OfferFit's approach, we need to meet the bandits.

Multi-armed bandits

Let's start with one of the simplest types of reinforcement learning, called a **multi-armed bandit**. The name comes from an old joke: a slot machine has a lever to pull and takes your money — they should call it a one-armed bandit! Classical game theory poses the following problem:

> *You are facing an array of 10 slot machines. You have 1000 coins, each of which can be used for one pull on any of the ten machines. Your goal, of course, is to make as much money as possible. What do you do?*

Faced with this lineup in a casino, your optimal choice is probably to walk away. But in this hypothetical game, you are promised that some of the slot machines would never be allowed in a real casino — they pay out often enough that you could turn a profit. The problem: you don't know which slot machines are which, and you don't know how big your potential gains could be. Since you're facing not one arm but 10, this conundrum is called the **multi-armed bandit problem**.



One-armed bandit

Multi-armed bandit

Suppose you put your first coin in machine #7, and you get five coins back. Now you know that this machine has a (still unknown) chance of paying out. You are faced with a choice: do you **explore** your other options by putting coins in other machines, or **exploit** the knowledge you already have about which option gives the best rewards? As you continue to spend your coins — as you continue to explore the **action space** — you might settle on a particular machine you think will give the best payout, or you might keep trying to learn more about other machines. This **explore-exploit tradeoff** turns out to be a central problem in reinforcement learning.

A **multi-armed bandit (MAB)** is a reinforcement learning model that tries to solve a multi-armed bandit problem.[2] In a marketing context, the actions — the array of slot machines — are the various choices a marketer might make — creative, sending time, frequency, etc. Instead of learning which slot machines pay out the most money, the multi-armed bandit is learning which subject lines (for example) generate the most conversions. Just as in the case of the slot machines, the central problem is how to balance **exploration** and **exploitation**.

One of the simplest ways to navigate the explore-exploit tradeoff is the **epsilon-greedy** algorithm, which works as follows:

1. Choose a small parameter **epsilon**, say 10%.

2. The multi-armed bandit is **greedy** 90% of the time, picking the action which generates the best possible reward, as far as the bandit knows so far.

3. The other 10% of the time, the bandit **explores** by making a random choice.

In other words, **exploit** most of the time — get the best possible reward you know about — but occasionally **explore** — try to find something better. This simple algorithm — the epsilon-greedy multi-armed bandit — is surprisingly effective, even in real-world marketing contexts.

The exploration strategy can be improved in various ways. For example, a bandit could be greedy 90% of the time, choose randomly from among the top 20% of options 5% of the time, and only make a truly random choice 5% of the time. These parameters can also be varied as the bandit learns, so that the bandit explores less as it learns more, a strategy known as **epsilon decay**. Other more sophisticated methods include **Thompson sampling**,[3] a Bayesian method which handles uncertainty about which options give the best rewards, and **upper confidence bound** strategies, which take into account both the predicted reward and the model's degree of confidence in that prediction.

But no matter how sophisticated the exploration strategy, all multi-armed bandits have a fundamental, and devastating, limitation: **multi-armed bandits cannot tell customers apart**. Let's suppose that a multi-armed bandit has learned that referral emails convert most often if they are sent on Wednesdays. The bandit will likely choose to send future emails on Wednesdays most of the time, which is the strategy it knows to maximize rewards. But a marketer, or a multi-armed bandit, that adopts the "winning" option is simply letting the majority rule — some customers are more likely to engage with an email that arrives on Saturday. In an election, getting 51% of the vote means you win. In a marketing campaign, engaging 51% of your customers is probably a losing strategy — 49% of your customers aren't getting what they want.

One popular solution to this problem is called **MAB by segment** — divide customers into segments, and let a multi-armed bandit experiment to find the best choices for each segment. Dividing customers into segments is better than nothing, but it doesn't solve the problem. A marketer is still deciding how to engage each customer based on a single data point — their segment. Today's lifecycle marketers have rich first-party data moldering in their data warehouses and CDPs. MAB by segment lets the depth of that data go to waste.

---

[2] Kuleshov, Volodymyr and Doina Precup. "Algorithms for multi-armed bandit problems." arXiv preprint arXiv:1402.6028, 2014.

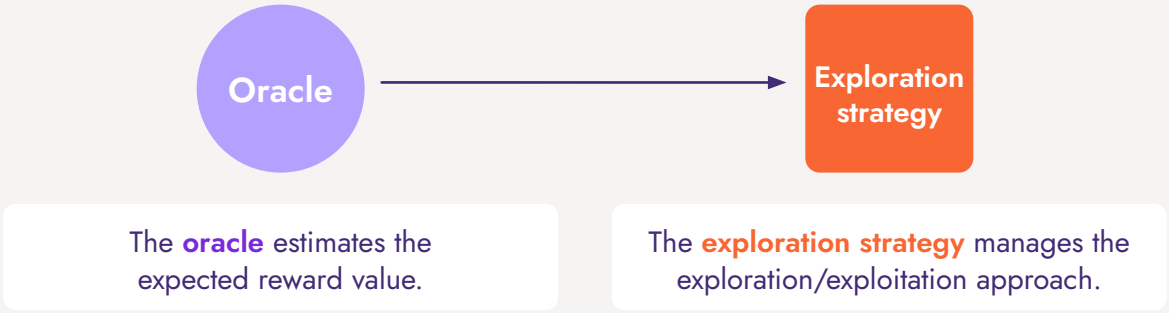
[3] Russo, Daniel J., et al. "A tutorial on Thompson sampling." *Foundations and Trends in Machine Learning* 11.1, 2018: 1-96.

[4] Nguyen, Trong T., and Hady W. Lauw. "Dynamic clustering of contextual multi-armed bandits." *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, 2014.

Contextual bandits

To make 1:1 decisions for each individual customer, OfferFit relies on a family of more advanced reinforcement learning models called **contextual bandits**.[5] These models are well-suited for experimentation in marketing because they are extremely **sample efficient** – they learn quickly.

A contextual bandit has two elements: an **oracle** and an **exploration strategy**. The exploration strategy is the bandit's plan for navigating the **explore-exploit tradeoff**, and could be any of the methods a multi-armed bandit might use, such as epsilon-greedy or Thompson sampling. The **oracle** is the bandit's way of predicting the **reward** for each of its available **actions**. Where a multi-armed bandit makes a single prediction of the value of each action which it uses for all customers, a contextual bandit makes a unique prediction for each individual customer.



Contextual bandits understand customers using **customer features** – the collection of relevant attributes that marketers can glean from their first-party data. These features could be built from data such as a customer's:

- Purchase history,
- Average spend per month,
- Current contract,
- Months remaining on a contract,
- Engagements with an app, website, or loyalty program,
- Viewing history,
- Type of device,
- History of emails or phone calls to customer supporter, or
- Any data which a marketer believes could differentiate customers.

[5] Bietti, Alberto, Alekh Agarwal, and John Langford. "A contextual bandit bake-off." *The Journal of Machine Learning Research* 22.1, 2021: 5928-5976.

In general it's better for a model to use too many features rather than too few – if a particular feature ends up not doing much to predict customer behavior, the model will learn to ignore it. OfferFit's AI typically uses hundreds of features.

In a similar way, contextual bandits understand their available actions using **action features**. For example, an offer for an internet plan might have features like:

- The length of the contract,
- The upload and download speeds of the plan,
- The monthly price, and
- The terms of a discount or promotion.

These parameters allow the model to learn, for example extrapolating the expected reward for a 10% discount based on customer responses to 5% and 20% discounts.

The set of all possible values of all customer features is known as the **customer feature space**. An **oracle** predicts the expected reward for each action for every possible customer – that is, given a point in the **customer feature space** and a point in **action feature space**, the oracle predicts the expected reward. For this reason, an oracle can make predictions even for new customers it has not encountered before – the model built on data from existing customers extends to other possible values of the customer features.

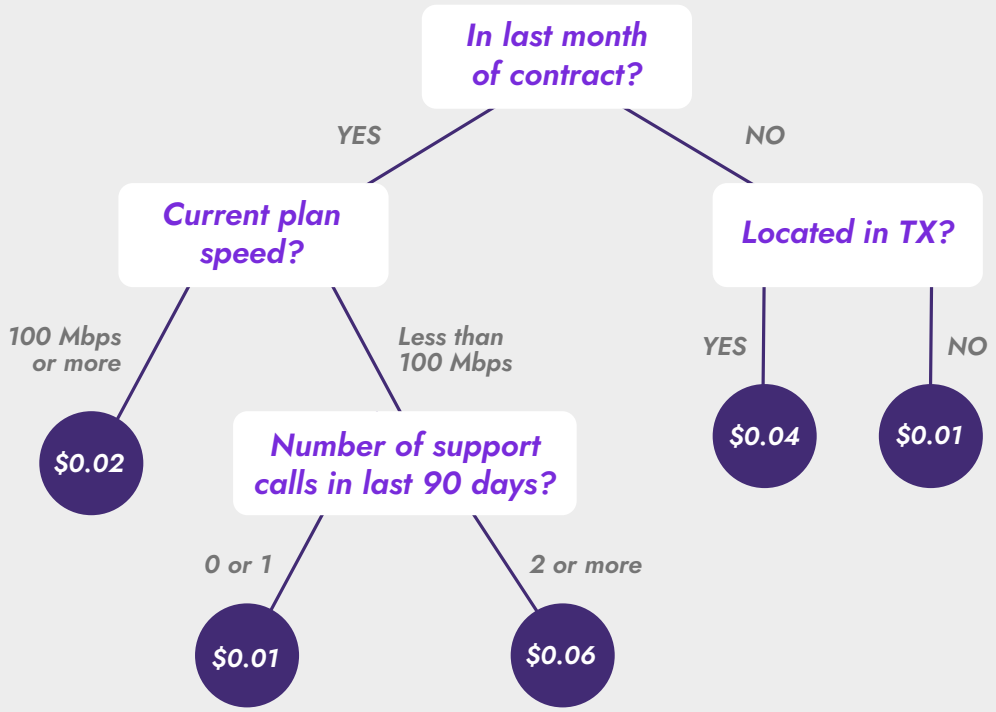
An oracle's prediction is fundamentally a model, built to fit existing data on prior customer behavior, with the goal of predicting the reward for future customer actions. A contextual bandit could in principle use any kind of model, but in practice OfferFit's bandits rely on two types of oracles: **elastic nets** and **gradient-boosted decision trees**. This combination of models allows OfferFit's AI to navigate the **bias-variance tradeoff** – using simpler models when data is sparse to avoid overfitting, and more powerful models when data is rich to maximize performance.

Machine learning models deployed in marketing must take care that differentiating between customers does not lead to unfair outcomes, or worse, illegal discrimination. Even a model that doesn't "know" protected characteristics, like race or national origin, might end up making decisions based on customer features which correlate with these characteristics. (One infamous example in US history is the practice of basing mortgage decisions on zip codes, which are a proxy for race – a discriminatory practice which is now illegal.) Of course appropriate use of customer data varies by context and industry – offering customers different financial terms based on race or gender is unethical and likely criminal; offering different beauty products based on race or gender might just be common sense.

AI ethics is a complicated topic, but OfferFit starts with a simple question – would the personalization in this marketing campaign result in unfairness in meaningful individual-level outcomes? If so, the campaign is likely unethical, and is not an appropriate use of OfferFit's AI. OfferFit does not accept personally identifying information (PII), and takes care that when we use data to personalize, we give our customers' customers differentiated outcomes only in ways that are legal, ethical, and fair.

1. **Elastic nets**. Perhaps the simplest way to model data is linear regression — choosing appropriate weights for each feature to give the "best-fit line" to the existing data. Customer feature spaces are too large for naive linear regression to work well — there are too many parameters to fit. An elastic net is a linear model with two additional constraints: i) most of the weights must be 0 and ii) the average weight must be small. These constraints mean that the linear model is allowed to depend on only a small subset of the hundreds of features and cannot let any one feature dominate the model.

2. **Gradient-boosted decision trees**. A decision tree is a series of "if-then" statements arranged in a tree — a flow chart without any loops. Contextual bandits can use decision trees to model the predicted reward for each action. In effect, a decision tree is a segmentation of the customer feature space and a predicted reward for the customers in each segment. The more decisions encoded in the tree, the finer the segmentation becomes. Gradient boosting is the process of using not one decision tree, but a chain of many trees. The first tree predicts rewards, and each successive tree tries to minimize the error of the previous tree.



A decision tree which predicts the marginal increase in ARPU from an internet upgrade offer. The decision tree partitions customer feature space — in this case into 5 segments — and predicts the expected reward for sending the given offer to a customer in each segment.

These different varieties of contextual bandit make different tradeoffs between making the best possible prediction for each customer and overfitting the data. When data is sparse, a simple model like an elastic net is best. As data accumulates, more complex models like gradient-boosted decision trees will make better predictions. Moreover, the **hyperparameters** of a decision tree oracle — such as the maximum depth of each tree, and the number of trees to use — can be allowed to increase as data becomes richer. If the oracle segments the customer feature space too finely when data is sparse, the model is likely to overfit the data and not generalize well to new situations. As the model learns, it's appropriate to use more powerful and flexible oracles.

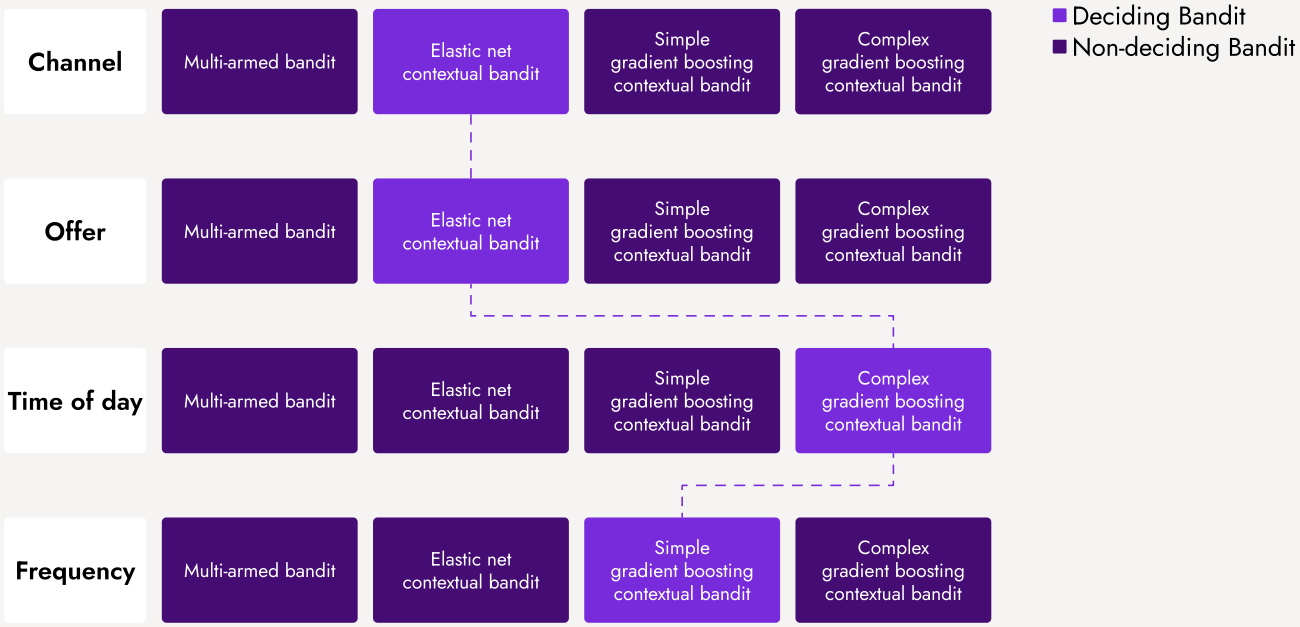
OfferFit's community of bandits

Now that we've met our bandits, let's see how OfferFit uses them to make 1:1 decisions for each customer. The bandits need to explore a large **action space**, learn quickly to be extremely **sample efficient**, and avoid **overfitting**. OfferFit achieves these objectives with three key innovations.

1. We use not one bandit, but **a team of four**: a multi-armed bandit, an elastic net bandit, a simple gradient-boosting bandit, and a complex gradient-boosting bandit. The complex gradient-boosting bandit is allowed to use deeper decision trees with more nodes, and also to chain together more trees in gradient boosting. (Note that while the bandits are named after the type of *oracle* they use, each bandit also has its own *exploration strategy*.)

2. We partition the action space by **dimension** and assign a team of bandits to each dimension. Imagine a marketer is using OfferFit to find the best channel, offer, and time of day for each customer. The channel dimension bandits first select a channel, say email. This decision is passed as *additional context* to the bandits making a decision about which offer to recommend. The offer bandits' oracles must then predict the reward for every set of customer features *and* every possible choice made by the channel bandits. The time of day bandits, in turn, must predict the expected reward for every possible customer, channel, *and offer*. Each bandit team is thus responsible for learning and exploring only the action space for its own dimension, which means quicker and more reliable learning.

3. For each dimension we have several active bandits, and each one makes its own prediction of the expected rewards for each action and customer. OfferFit's AI must choose which of these recommendations to follow — for this customer and this dimension, and on this day, which bandit's advice is best? The deciding bandit for each dimension is chosen by a **meta-bandit**. OfferFit's meta-bandits are special contextual bandits that decide, for each customer, which bandit's recommendation to follow. Suppose that OfferFit's AI is deciding whether to contact given customer by email, phone or SMS. The channel bandits each make their own prediction as to which channel will give the best reward for that customer. The channel meta-bandit must then choose not from the set of channels, but *the set of channel bandits*. The meta-bandit empirically discovers not the best *action* for each customer, but the best *bandit* to make a decision about that customer. For customers about whom we don't know very much — a customer in a sparse region of customer feature space — the meta-bandit might choose the elastic net bandit or even the multi-armed bandit. These simpler models are likely to perform better because they will avoid overfitting. As data accumulates and the bandits learn, the more powerful bandits will perform better, and the meta-bandit will choose them to be the deciding bandits more and more often.



OfferFit's meta-bandit chooses the deciding bandit for each dimension and each customer.

OfferFit's proprietary **community of bandits** learns reliably and flexibly, quickly finding the best action for each customer which brings the best reward.

An automated experimentation success story

A top North American bank with over $10B in revenue wanted to optimize business credit card referrals. Their marketing team knew that a single new business account was extremely valuable, and so even a small increase in conversions would translate into significant gains in revenue. They conducted extensive A/B testing, worked with consultants to optimize their messaging, and were very confident they had learned which creative, messaging, sending times, and days of the week were most effective for their email referral campaign.

Choosing the options that worked best for their customer population as a whole, however, meant making tradeoffs that weren't optimal for each individual. For example, the bank knew that more frequent emails meant more conversions, but also more unsubscribes. So they "split the difference" by emailing customers every other week. OfferFit's **community of bandits** was able to make 1:1 decisions, emailing some customers weekly and others only monthly while also varying message, creative, time of day, and day of the week. As the model learned, OfferFit's emails reached a **92%** increase in conversion rate compared with the bank's business-as-usual approach, worth a staggering **$16M** in annualized value.

Uplift in conversion rate from business credit card referrals.

More elementary methods would not have been likely to realize these gains. Multi-armed bandits, for example, would have struggled to see uplift on a campaign that had already been significantly optimized through A/B testing. The bank needed the full power of **automated experimentation** with OfferFit's community of contextual bandits.

Why marketers need reinforcement learning models integrated into a SaaS product

In practice there are three ways that marketers can leverage a reinforcement learning model for experimentation — work with a consultancy, build a solution in-house, or implement a SaaS product. OfferFit co-founder Victor Kostyuk worked as a Lead Data Scientist at BCG Gamma, the data science arm for the Boston Consulting Group, where he built reinforcement learning models for Fortune 500 companies. Victor saw both the power of reinforcement learning and the challenges inherent in building and maintaining custom models. He came to believe that the best way to empower marketers is not to create custom models for each situation, but to offer a robust SaaS product that can be configured to companies' specific needs. OfferFit was founded on this belief. Why is a product a stronger approach than a custom build?

1. **Wrangling data takes infrastructure and tooling**. Marketers have rich first-party data in their data warehouses and CDPs, but this data needs to be transformed before it can power a machine learning model. A data warehouse might log events such as email clicks or purchases. These raw events must be processed into **customer features**, like "number of clicks in the last 30 days'' or "total dollar value of in-app purchases in the last 60 days." Then machine learning models need data aggregated in some standard format, such as one row per customer and one column per customer feature. OfferFit quickly and seamlessly performs **feature engineering** — transforming raw data into the collection of customer features that OfferFit's AI will use to differentiate individual customers — and aggregates data as necessary to feed the machine learning model.

2. **Reinforcement learning is hard to get right**. In-house data science teams typically have experience building models based on supervised and unsupervised learning, such as churn prediction or cluster analysis. These types of machine learning models are well supported by "off the shelf" tools, and can be stood up relatively quickly. Reinforcement learning models, such as those that power OfferFit, are a different story. Implementing reinforcement learning requires making dozens of choices about which models to use, and how to tune their hyperparameters. OfferFit has deployed the models best suited to experimentation in marketing. Even within our curated and fine-tuned community of bandits, OfferFit uses a meta-bandit to choose the best model to make a decision for each individual customer. Without OfferFit, *you are the meta-bandit*, and countless companies have tried and failed to get reinforcement learning right.

3. **Marketers need to see what the machine is doing**. Experimentation in a marketing campaign means testing dozens or hundreds of variants. Marketers want to see what is actually being implemented and what recommendations are being made for their customers.

4. **Marketers need reporting for performance and insights**. Reinforcement learning models are driven by the reward the agents are trying to maximize, typically a financial metric like referrals, renewals, conversions, repurchases etc. But marketers need to know if the machine is working. OfferFit provides built-in reporting that clearly demonstrates lift compared with a control group — marketers can watch

the community of bandits learn and create value in real time. Marketers can learn, moreover, *how* the AI is making gains – what insights the AI has discovered about which actions work best with which customers.

OfferFit lets marketers deploy the power of reinforcement learning without implementing complex code or hiring expensive consultants, but some marketers may be inclined to build their own solution. Before deciding to build, companies should consider:

- **Cost**. Standing up an engineering and data science team is expensive, and their time is scarce and valuable. And what will happen when data science and engineering teams have moved on to other projects? Marketers need a plan to maintain their ML solutions over time.

- **Risk**. Reinforcement learning is a relatively new and specialized field of machine learning. Moreover simpler implementations, like multi-armed bandit by segment, may not give strong results to a team that has already optimized significantly with manual A/B testing. Marketers should be sure they have the necessary expertise in-house before deciding to build a more sophisticated solution.

- **Time to value**. Building an automated experimentation capability from scratch, even one intended for only one application, might take a dedicated engineering team a year or more. Other priorities might come up that could push back the schedule, and the longer an ML project goes on without demonstrating value, the likelier it becomes that someone decides it's time to pull the plug.

Implementing a reinforcement learning solution like OfferFit will often get results more quickly and affordably, and give a solution that's easier to maintain over time.

Conclusion: reinforcement learning is the future of experimentation in marketing

Lifecycle marketers are closer than they've ever been to the dream of 1:1 decisioning for every customer, thanks to the twin technologies of generative AI and reinforcement learning. Generative AI will help marketers integrate data and create variants, while reinforcement learning helps them automate experimentation. A/B testing will never scale to match the speed of the market and the number of variables marketers need to test. To test and learn at the level of individual customers, marketers need to harness the power of reinforcement learning.

Humans have always learned through trial and error, and marketers are no exception. Marketers have traditionally used A/B testing to empirically discover the best action for each customer, but the era of A/B testing is over. The future of experimentation in marketing will be automated testing and learning using reinforcement learning. Marketers can start that journey with OfferFit's community of bandits.

About OfferFit

OfferFit accelerates the creation of knowledge. Trial and error has always been the core of human progress. At OfferFit, we automate experimentation using reinforcement learning, a type of self-learning AI, to make knowledge creation faster than ever before.

A/B testing can be effective for lifecycle marketing, but it's slow and doesn't scale. Lifecycle marketers use OfferFit to radically accelerate experimentation. Marketers choose options for multiple dimensions such as messaging, creative, incentive, channel, and timing. Then OfferFit experiments to discover the best-performing recommendations for each customer.

OfferFit's Automated Experimentation Platform lets marketers unlock the value in their first-party data and maximize whichever KPIs are most important to their business.

OfferFit

Make the Leap from A/B to AI

offerfit.ai

hello@offerfit.ai

About the Authors

George Khachatryan
Co-Founder & CEO

Before founding **OfferFit**, George was an Associate Partner at McKinsey & Company, where he led transformations of some of the world's largest companies. Previously, George co-founded a technology company whose products are used today by over 1 million users. He holds a PhD in Mathematics from Cornell University.

Victor Kostyuk
Co-Founder & CTO

Before founding **OfferFit**, Victor was a Lead Data Scientist at the Boston Consulting Group. He is an expert in reinforcement learning, with extensive experience serving Fortune 500 companies on offer personalization. He holds a Ph.D. in Mathematics from Cornell University

Nathaniel Rounds
Product Marketer

Before joining **OfferFit**, Nathaniel spent a decade leading SaaS product design and development. He holds a Ph.D. in Mathematics from Stony Brook University.