That's me!

A User Story

A book for children, folks with lots of imagination,

and people in product marketing.

aampe

At Aampe, we do some hard things, but we don't think those things should be hard to understand.

That's why we wrote this story.

Part 1

The problem!

This is a user.

If we met this user in real life, we’d notice **all kinds** of interesting things that make him or her unique.

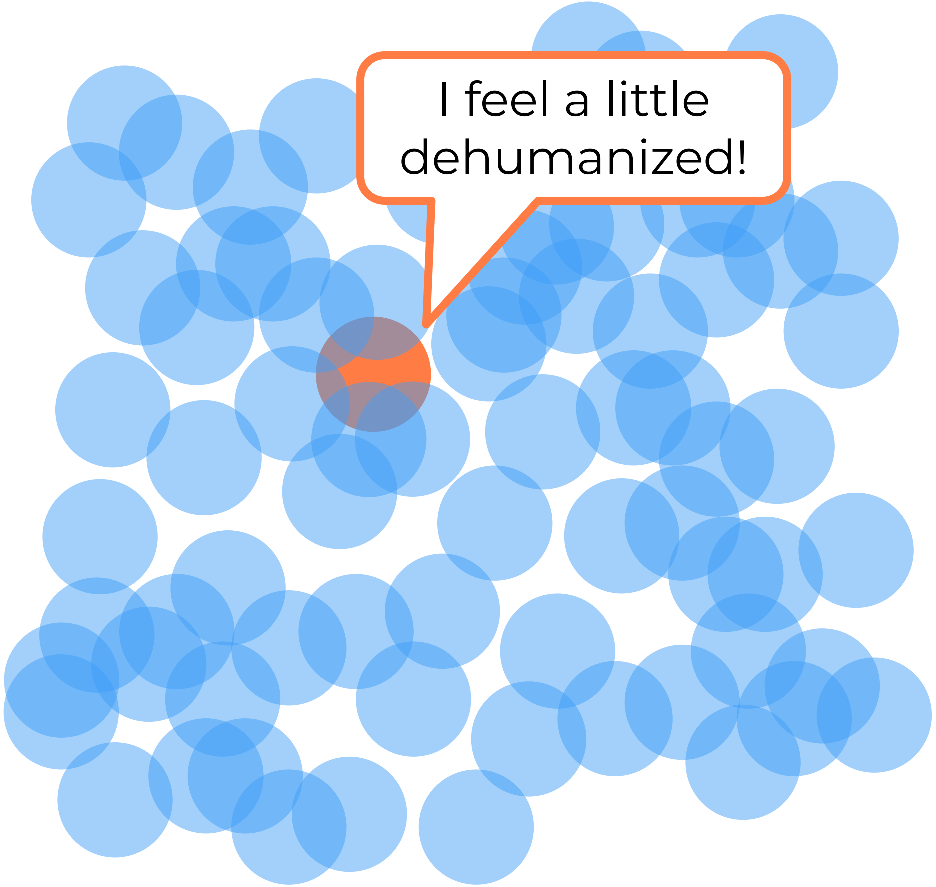
Those interesting things would make us talk to this user in **different** ways than we would talk to other users.

This is this user’s story.

But we don't meet our users in real life

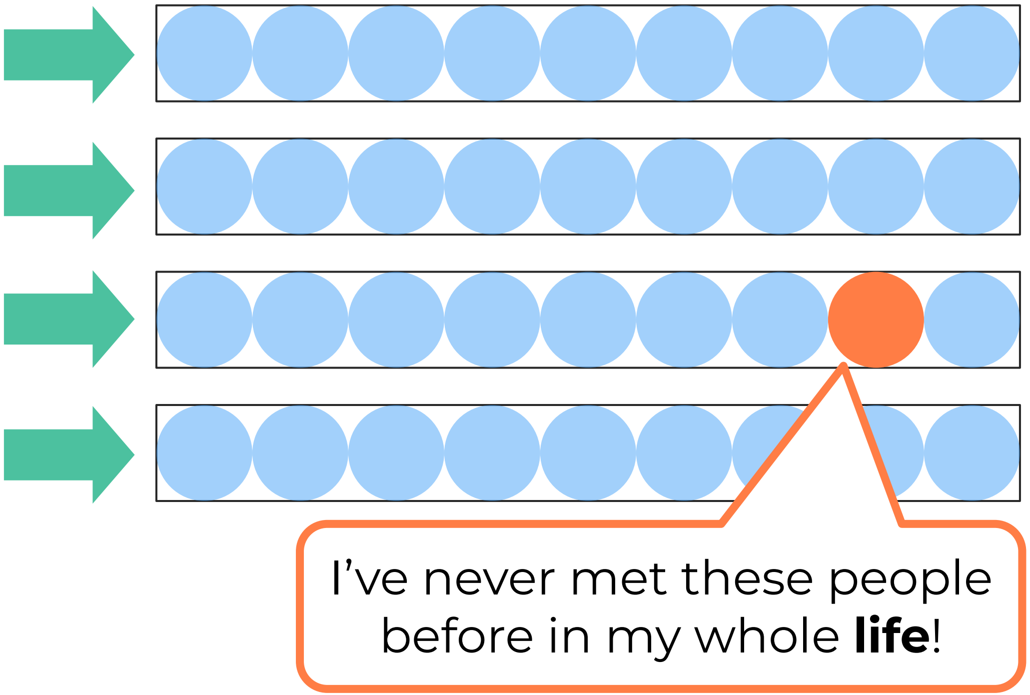
We meet them on our platform, along with all of our other users.

Because there are **so many** of them, and because we **only** ever see them in a very specific setting, **it's hard to tell users apart**, or learn what makes them unique, or treat them as individuals.



These are **segments**.

A segment is a **box** that is supposed to contain users who are all alike in some important way.



Segments are **clumsy**. If you make too many, they become **too hard** to use. If you have just a few, they **cover up** too much important information. Once you put a user in a box, all you can see is the box.

Think of it this way:

Which is more useful to know?

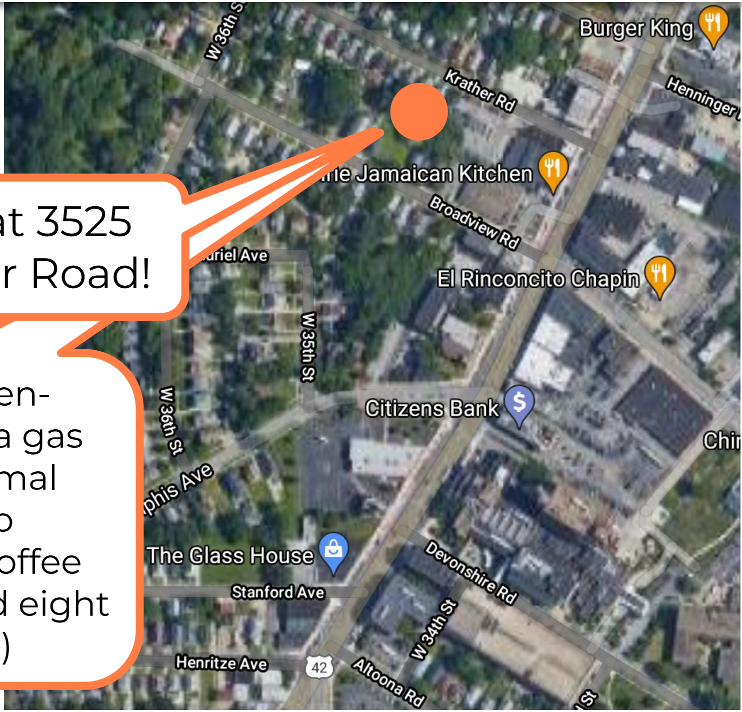
A person's **home town** VS. A person's **street address**

Clarkston, MI
Clearwater, FL
Cleburne, TX
Clemson, SC
Clermont, FL
Cleveland, OH
Clifton, NJ
Clinton, IA
Coarsegold, CA
Cody, WY
Cohasset, MA
Colchester, VT
Coldwater, MI
College Park, MD
College Station, TX
Collingswood, NJ
Colorado Springs, CO

I live in Cleveland!

I live at 3525 Krather Road!

(Within a fifteen-minute walk of a gas station, an animal hospital, two pharmacies, a coffee shop, a bank, and eight restaurants!)

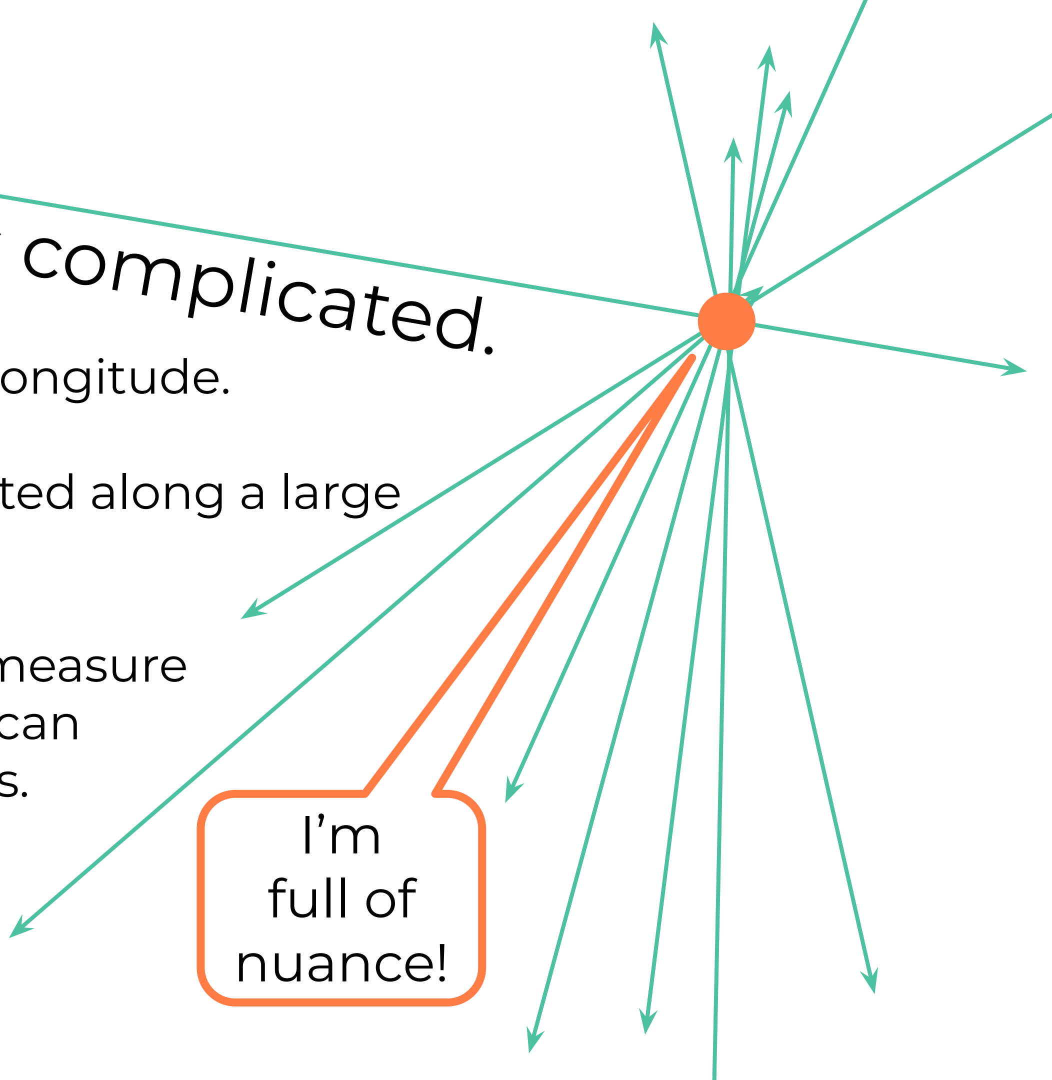


“Locating” behavior is complicated.

To locate a user in physical space, we just need latitude and longitude.

But a user’s **behavior** can be located along a large number of different **dimensions**.

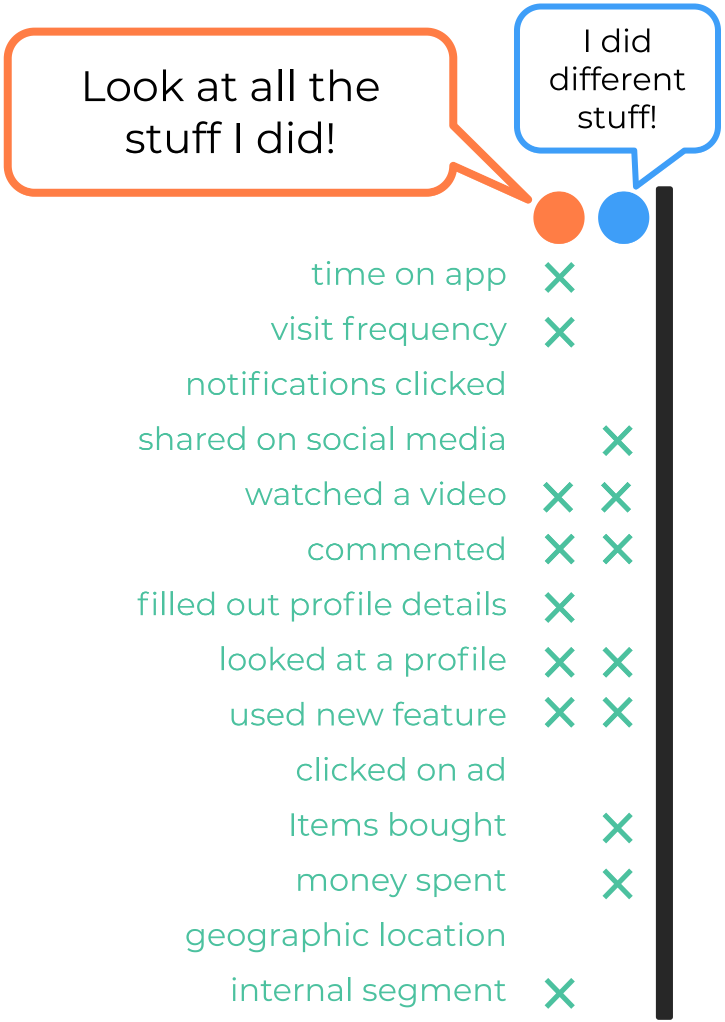
A dimension is anything we can measure about what a user does - and we can measure **a whole bunch** of things.



We need a map.

Part 2

The user landscape!



Everyone does stuff.

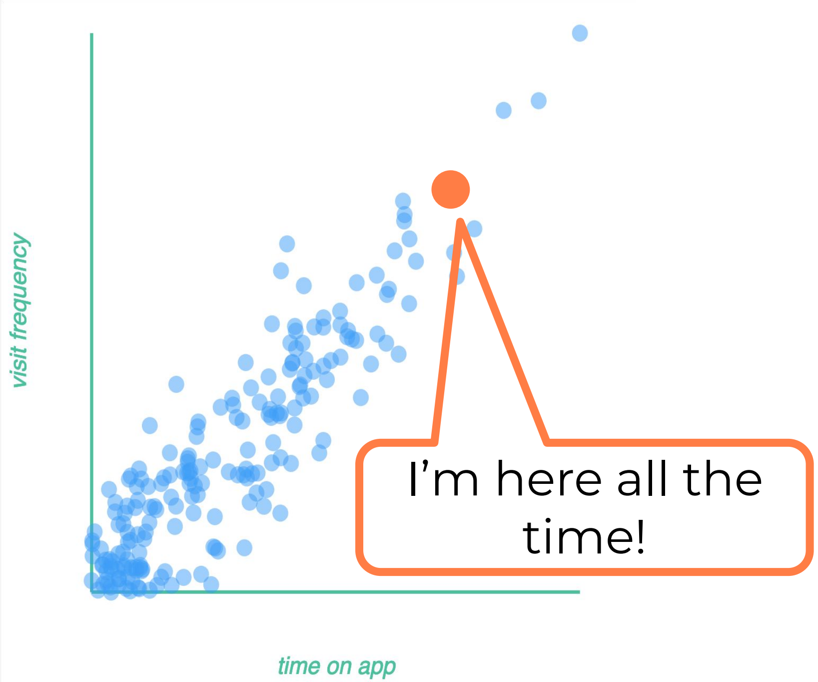
It's usually pretty **easy** to measure what users do in an app.

All those measurements are really **important**, but only if we can figure out what they mean.

That's where things get **difficult**.

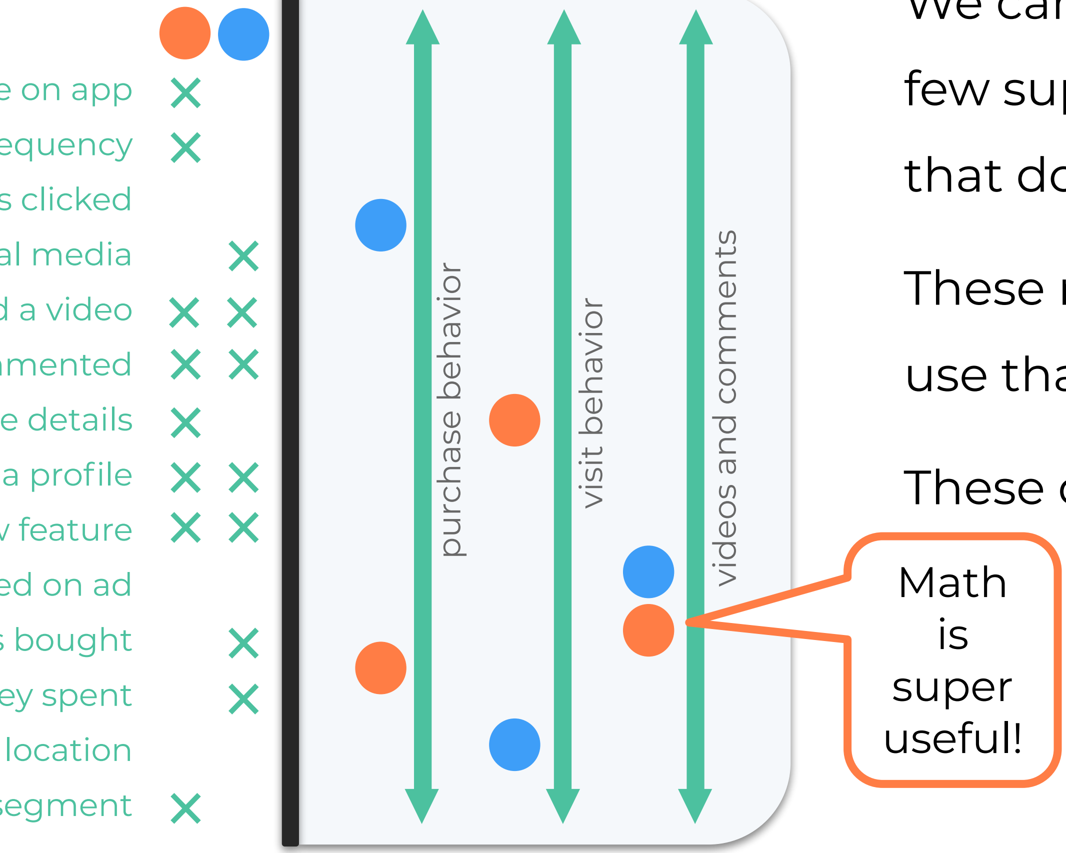
Behaviors can overlap.

Some dimensions are almost just two versions of the **same** thing:



Some dimensions are **hardly related** at all:

We turn raw behavior measures into a map.



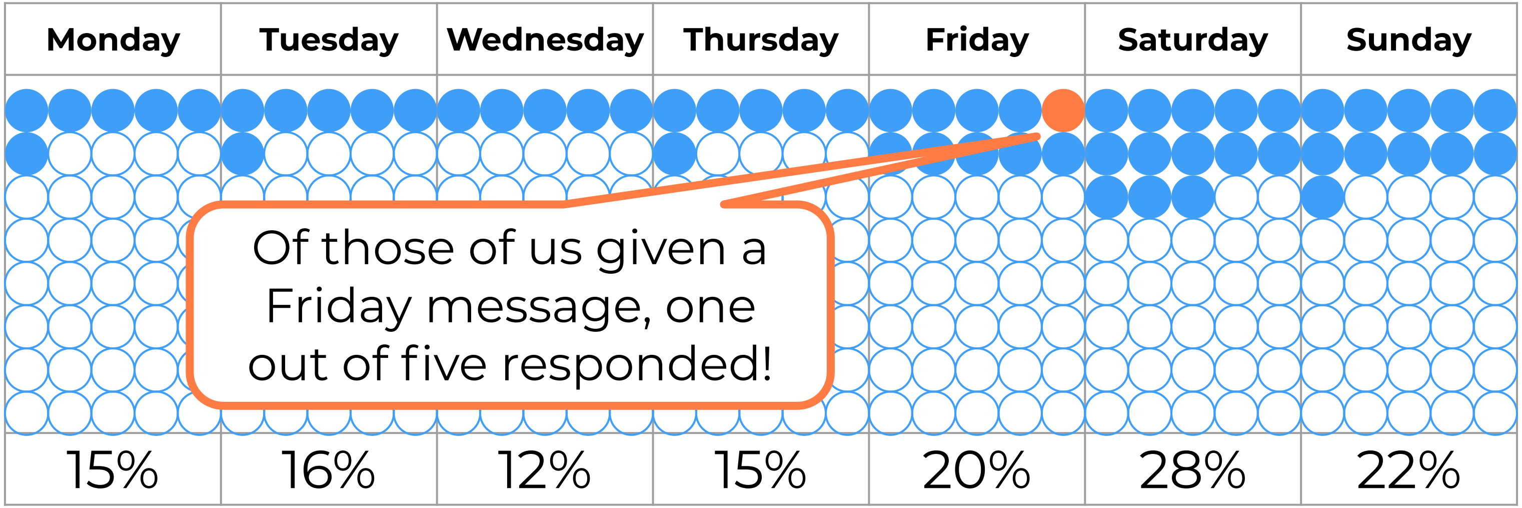
We can use some math to create a few super-informative **dimensions** that don't overlap (much).

These new dimensions are easier to use than the raw measures.

These dimensions form a **landscape** on which we can locate user in terms of their behavior instead of their geography.

It may seem easy to learn about behavior.

Say we want to learn what day of the week is best to send a message to users. We can send **different users** the **same message** on **different days** and see which performs better.



If only it were that simple.

Random differences in behavior can fool us.

For example: we **randomly** choose users to get a message on either **Friday** or **Saturday**. But some users just tend to click more on messages in general - they're more **clicky**. Here are the users listed in order of clickiness. The filled blue dots clicked on our message.

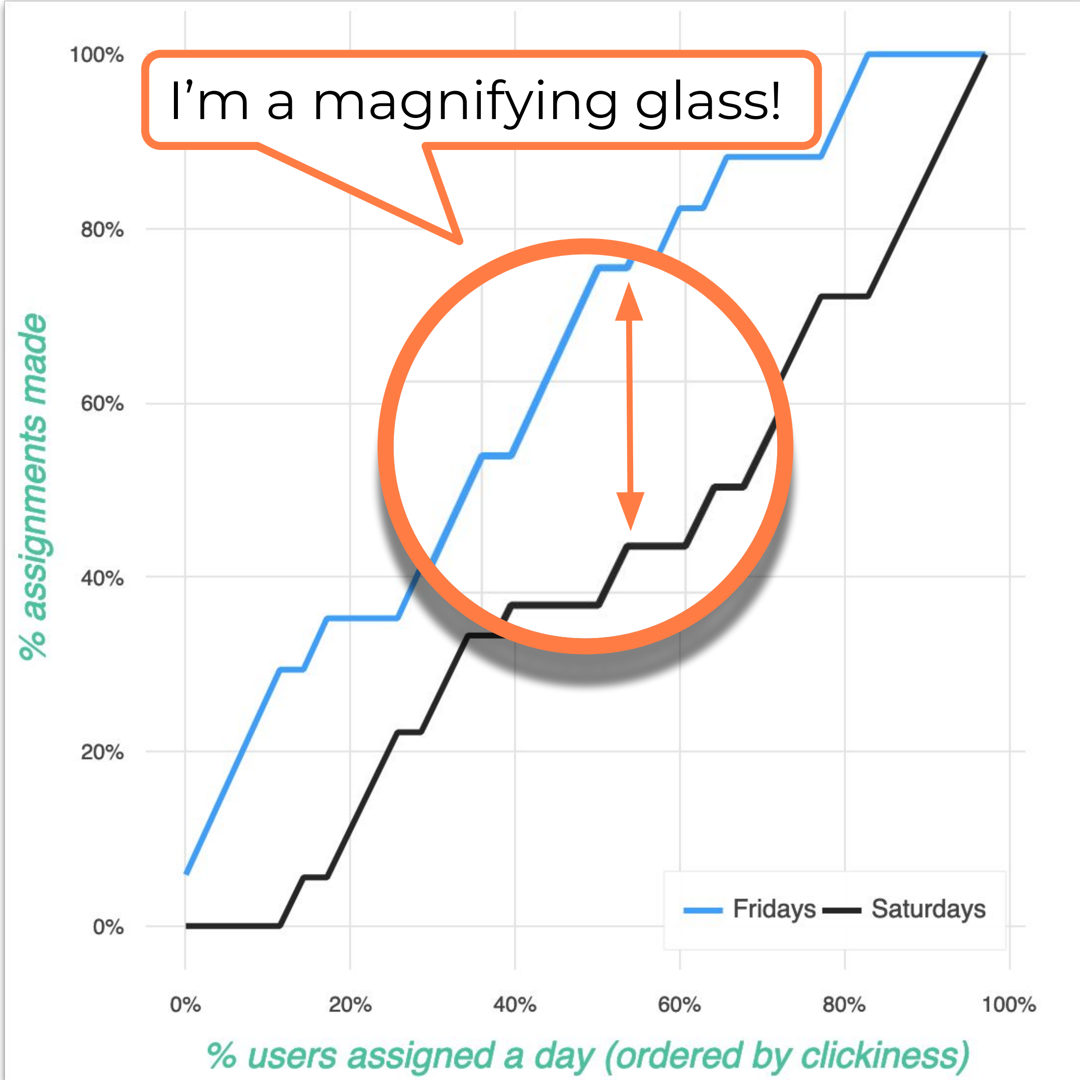
8 of the 17 users (47%) with Friday messages clicked, and 10 of the 18 users (55%) with Saturday messages clicked. **So Saturday is better, right?**

Don’t be fooled.

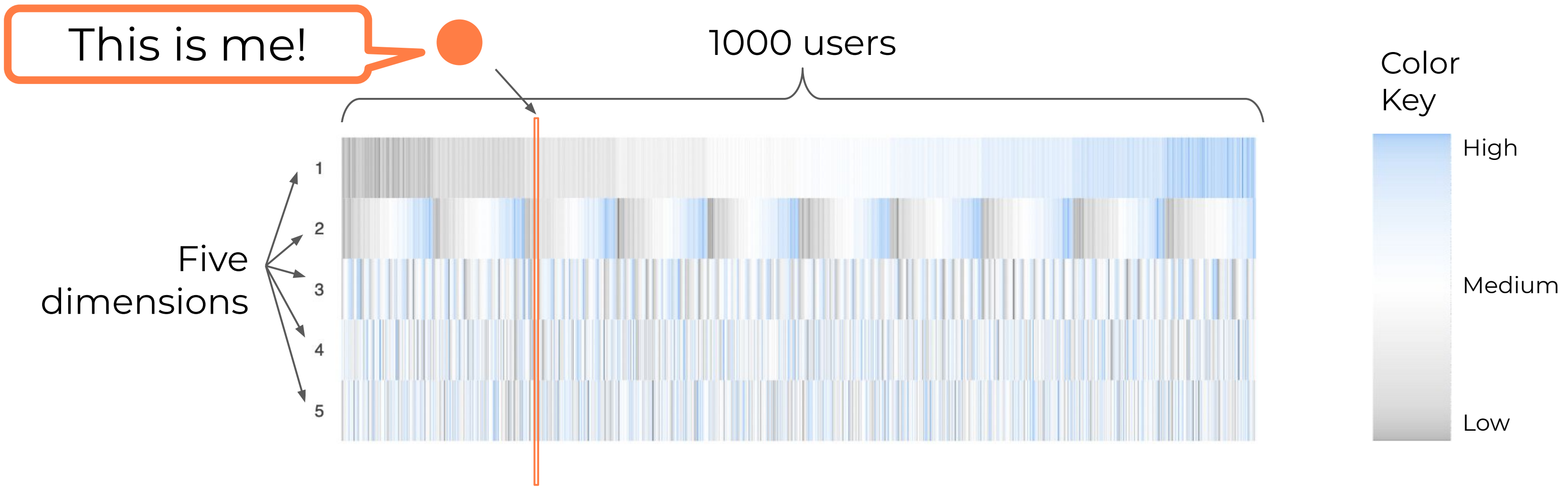
The least-clicky 50% of users got 70% of our Friday assignments, but less than 40% of our Saturday assignments.

Saturday wasn’t better.

We just assigned Saturday more often to more clicky users.

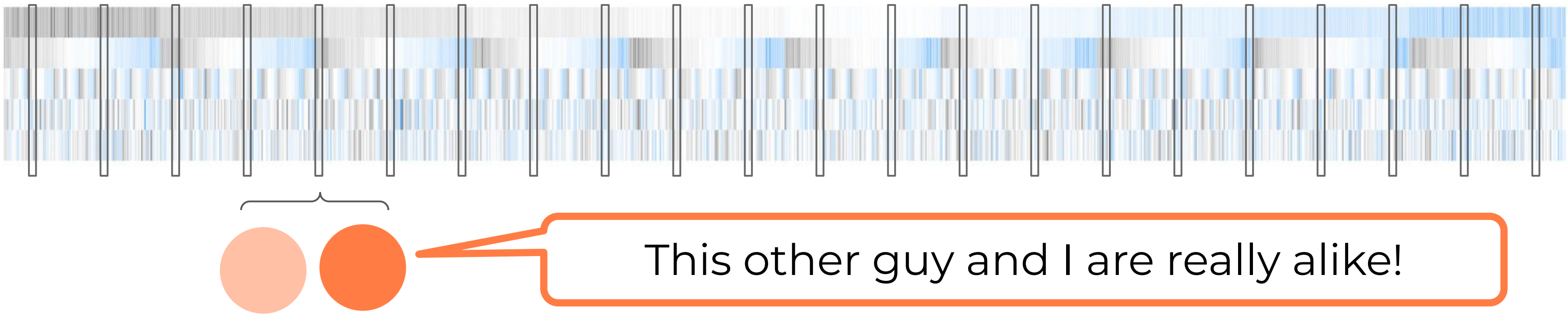


A behavior map keeps us from being fooled.



This is the user landscape. Notice how, on the first dimension, the users run from really low to really high. On the second, they run from low to high in big, repeating groups. On the third, they run low to high in smaller repeating groups. And so on.

Clusters of users keep us from being fooled.



If we want to test a Friday message against a Saturday message, we can **cluster** the user landscape in groups of two, and assign each day in each cluster: Friday, Saturday, Friday, Saturday, Friday, Saturday, and so on.

Clustered assignment guarantees that **different** messages are assigned to **similar** users, and picking clusters from across the whole landscape - from the far left to the far right - guarantees that **different** users get assigned **similar** messages.

Complicated tests are ok.

Say we wanted to test **several things** at once:

How often? Every 1, 2, 3, 4, 5, 6, or 7 days?

What time of day? 8:00am, 11:00am, 2:00pm, 5:00pm, or 8:00pm?

Which call to action? Excited ("Let's do this!"), bossy ("Try now."), or simple ("Ok")?

That's [7 frequency levels] ✕ [5 times of day] ✕ [3 calls to action] = 105 unique combinations. In that case, we'd create clusters of 105 users each, and assign **one** of each combination to **each** cluster.


```
['every 1 day', '8:00am - 11:00am', 'excited']
['every 1 day', '8:00am - 11:00am', 'bossy']
['every 1 day', '8:00am - 11:00am', 'simple']
['every 1 day', '11:00am - 2:00pm', 'excited']
['every 1 day', '11:00am - 2:00pm', 'bossy']
['every 1 day', '11:00am - 2:00pm', 'simple']
['every 1 day', '2:00pm - 5:00pm', 'excited']
['every 1 day', '2:00pm - 5:00pm', 'bossy']
['every 1 day', '2:00pm - 5:00pm', 'simple']
['every 1 day', '5:00pm - 8:00pm', 'excited']
['every 1 day', '5:00pm - 8:00pm', 'bossy']
['every 1 day', '5:00pm - 8:00pm', 'simple']
['every 1 day', '8:00pm - 11:00pm', 'excited']
['every 1 day', '8:00pm - 11:00pm', 'bossy']
['every 1 day', '8:00pm - 11:00pm', 'simple']
['every 2 days', '8:00am - 11:00am', 'excited']
['every 2 days', '8:00am - 11:00am', 'bossy']
['every 2 days', '8:00am - 11:00am', 'simple']
['every 2 days', '11:00am - 2:00pm', 'excited']
['every 2 days', '8:00pm - 11:00
['every 3 days', '8:00am - 11:00am
['every 3 days', '8:00am - 11:00am',
['every 3 days', '8:00am - 11:00am',
['every 3 days', '11:00am - 2:00pm',
['every 3 days', '11:00am - 2:00pm
['every 3 days', '11:00am - 2:00pm'
['every 3 days', '2:00pm - 5:00pm',
['every 3 days', '2:00pm - 5:00pm',
['every 3 days', '2:00pm - 5:00pm',
['every 3 days', '5:00pm - 8:00pm',
['every 3 days', '5:00pm - 8:00pm',
```


More tests at once just means bigger clusters!

The user landscape matters more than algorithms.

Science is hard!

If we just look at stuff with no context, we'll convince ourselves that we see things that aren't really there and miss a lot of important details, too.

If we **assign** messages based on the user landscape, we'll **actively create** the information we need. That's **more important** than the math we use to make sense of that information.

And so is one other thing...

Part 3

The rewards menu!

Let’s talk about dogs.

You might have heard of **Pavlov’s dog**. (It was actually *dogs* - plural - he had lots.) Pavlov studied what makes dogs drool.

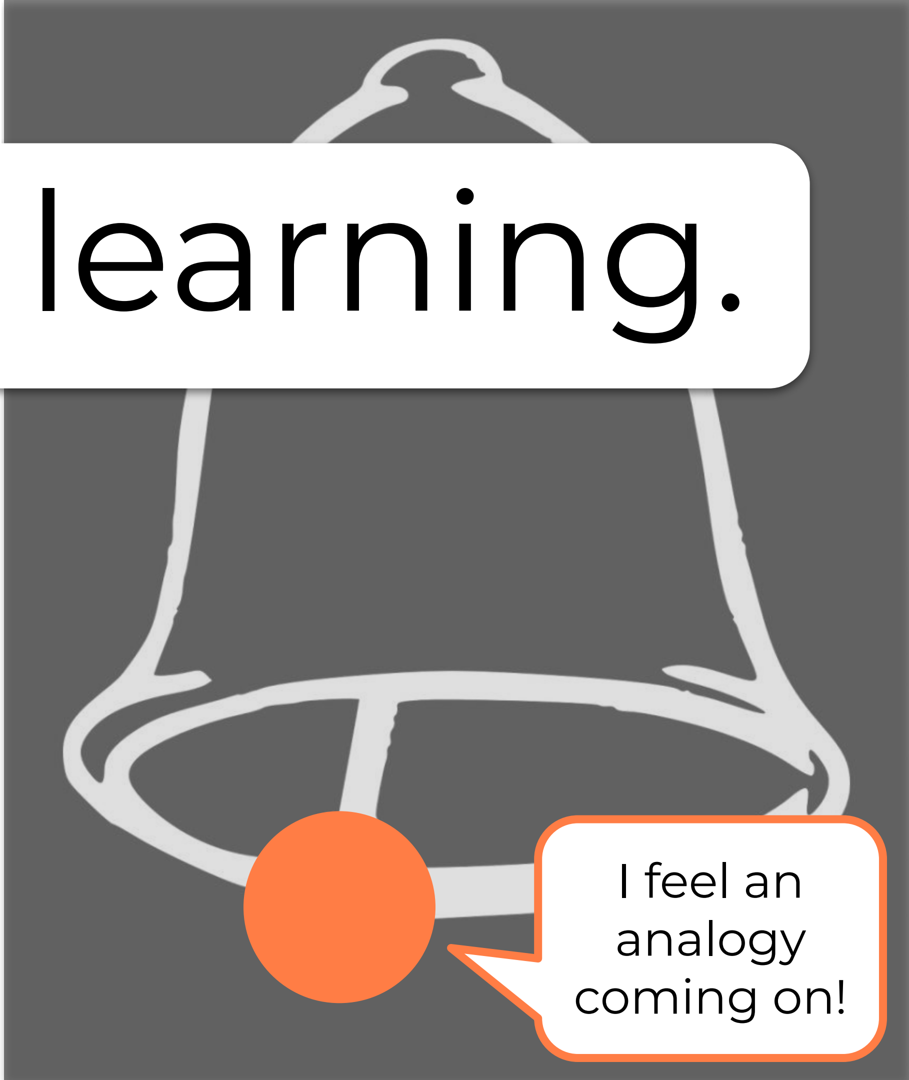
Science is weird!

They drooled when he gave them **food** (not surprising), but, over time, they started to drool even when they heard nothing but his **footsteps** (kind of surprising).

Now let's talk about

associative learning.

Pavlov found that if he took a normally-uninteresting thing (say, a bell) and paired it lots of times with something the dog wanted (food), the dog eventually started responding to bells the same as it responded to food.

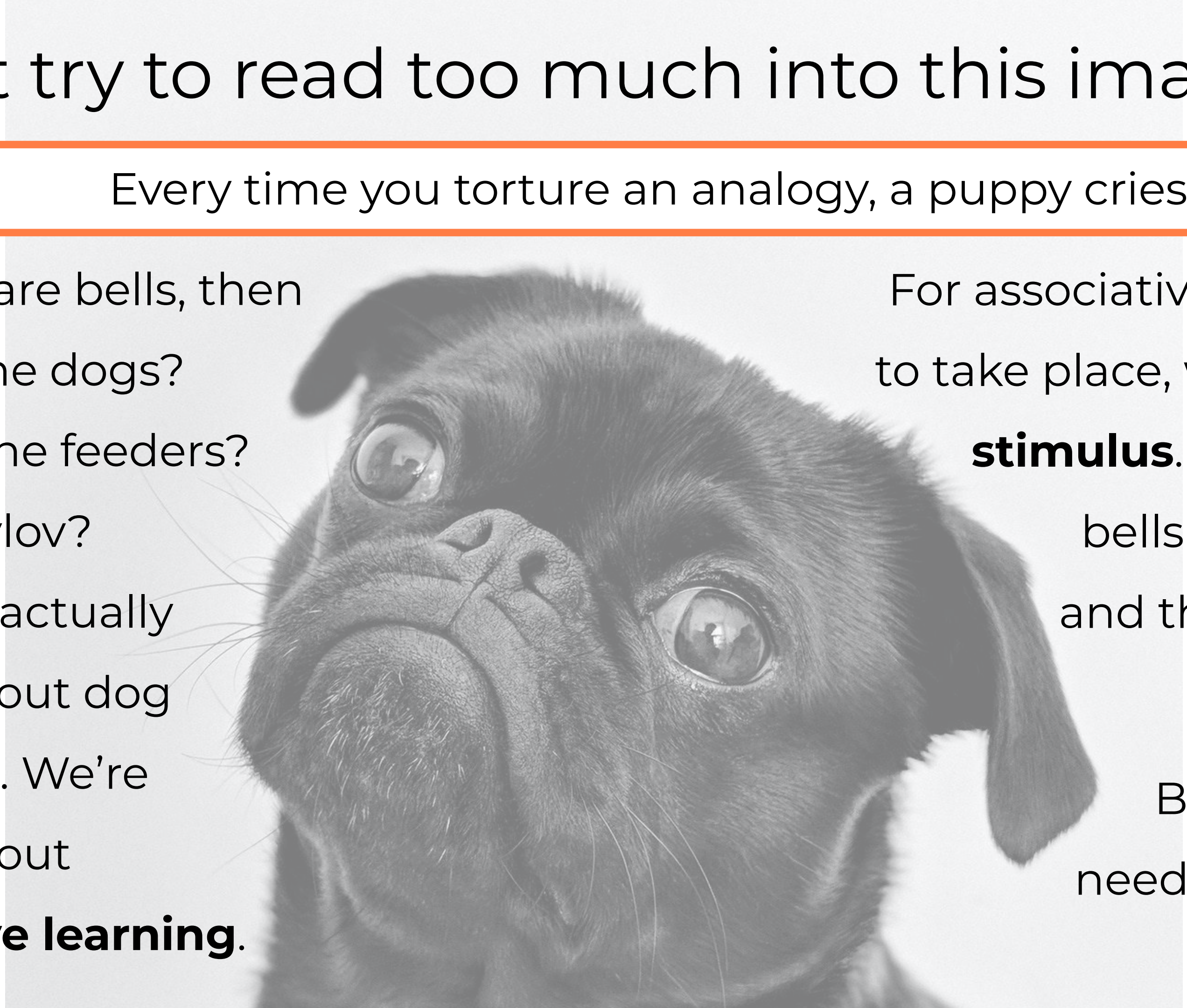


Message tests are bells.

Every way we can change our communication with users is a bell.

Don't try to read too much into this imagery.

Every time you torture an analogy, a puppy cries!



If policies are bells, then who are the dogs? Who are the feeders? Who's Pavlov?

We're not actually talking about dog drool here. We're talking about **associative learning**.

For associative learning to take place, we need a **stimulus**. That's the bells for Pavlov and the policies for us.

But we also need a **reward**.

Users reward us when they like what we offer.

Every time a user does something we want, they reward us.

Our job is to figure out what we can do to get more rewards.

Because those rewards come in many different forms, it's useful to think of a "menu" of rewards.

I like you a lot!
But not on Sunday mornings.

If you bug me on Sunday mornings, you get nothing but my disdain!

Assign menu items different “prices”.

I do all sorts of stuff.
You decide what
matters to you!

We can assign **points** to user actions that reflect how **important** each action is to us.

If an action has its own measure of value (say, money or amount of time spent), we can multiply the points by that measure.

Menu

Visit the app	1 point
Look at a profile	1 point
Search for content	2 points
Fill out profile details	2 points
Click a notification	3 points
Share on social media	4 points
Comment on a video	4 points
Click on an ad	7 points
Watch a video	8 points × # minutes
Buy an item	10 points × # dollars

This is the point where machine learning comes in handy.

We can see how users are similar or different. We can use that landscape to assign smart messaging tests. And we can price user behavior to match what we really care about.

Now, when we use a model, it can gives us results that **actually mean** something.

To-do list:

✓ User landscape

✓ Message assignments

✓ Rewards menu

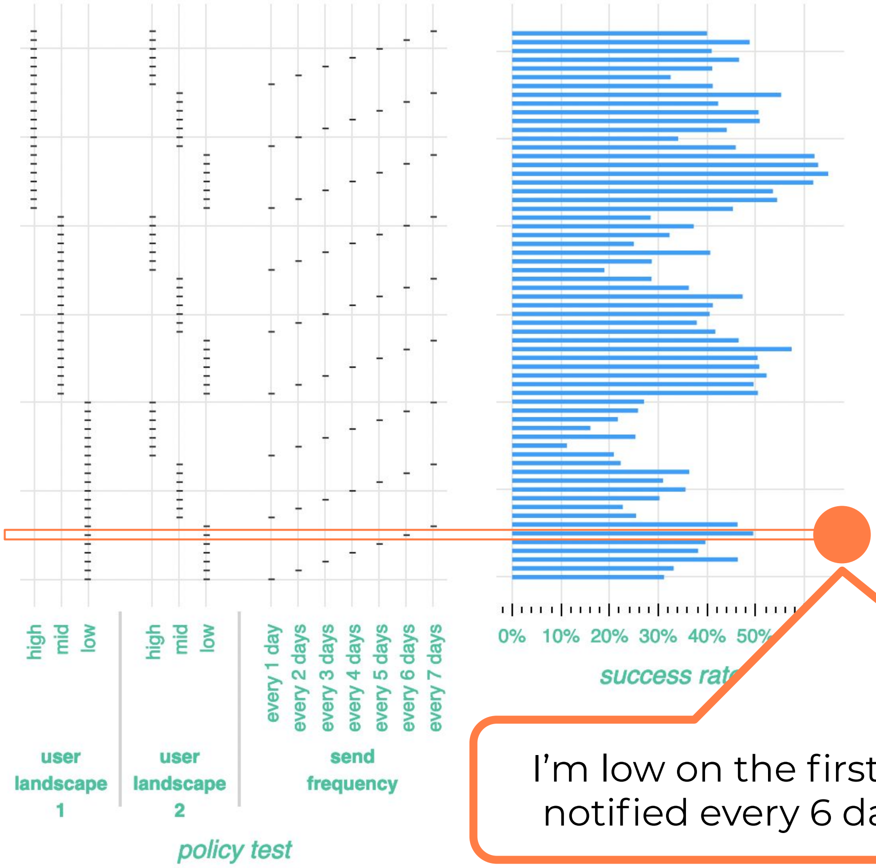
Fancy statistical model

Thanks for not just blindly feeding me into a computer!

Part 4

The *what-if?* model!

Models don’t get bored.



Look at every combination of message and landscape and count up how many rewards each combination gets.

A human can’t do that well - it’s **too many** combinations.

That’s why a **model** is useful**.**

I’m low on the first landscape dimension, low on the second, and notified every 6 days. That group has almost a 50% success rate!

A *what-if?* model has an imagination.

Say a particular user got a message with an excited call to action ("Let's do this!"). ***What if*** they had gotten the bossy call to action ("Try now.") instead?

Because our model saw people who got that other wording, and our user landscape tells us how similar our particular user is to all other users, the model can answer our question.

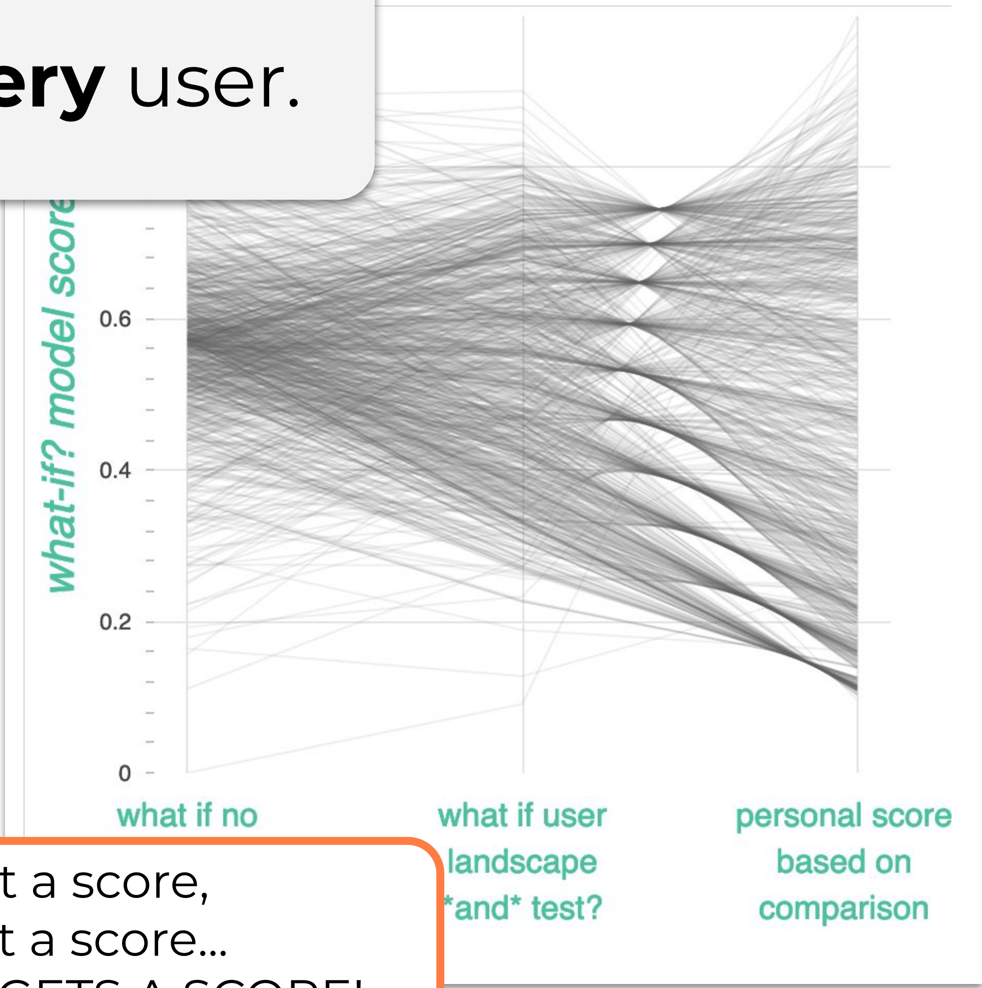
Look at three users.

We had the model estimate the chance each user would respond based **only** on their location in the user landscape. Then we asked what if those users were **also** given a specific test message.

The test was the only thing that could make a differences. **They all started out the same.**

Our model looks at **every** user.

Every test affects every user differently, because every user is different from all others. **Every user** get a personal score for each test, even those not in the original test. We can ask the model what **would have** happened if they had been included.

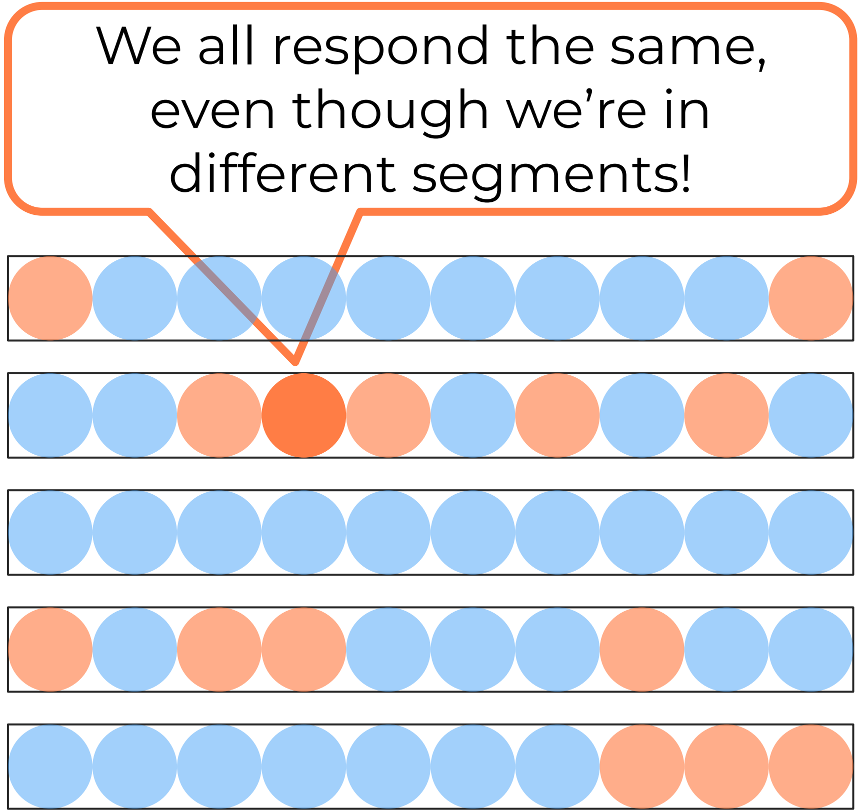


You get a score,
YOU get a score…
EVERYBODY GETS A SCORE!

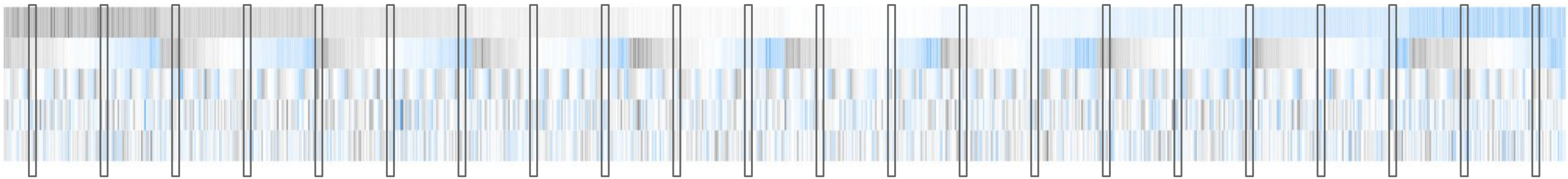
This is better than segments.

A segment is just a box with a label. It doesn't tell us **how** similar users are, or what differences **matter**.

A personal score takes something we can actually do, and tells us how likely each user is to respond to us doing it.



Remember this picture?

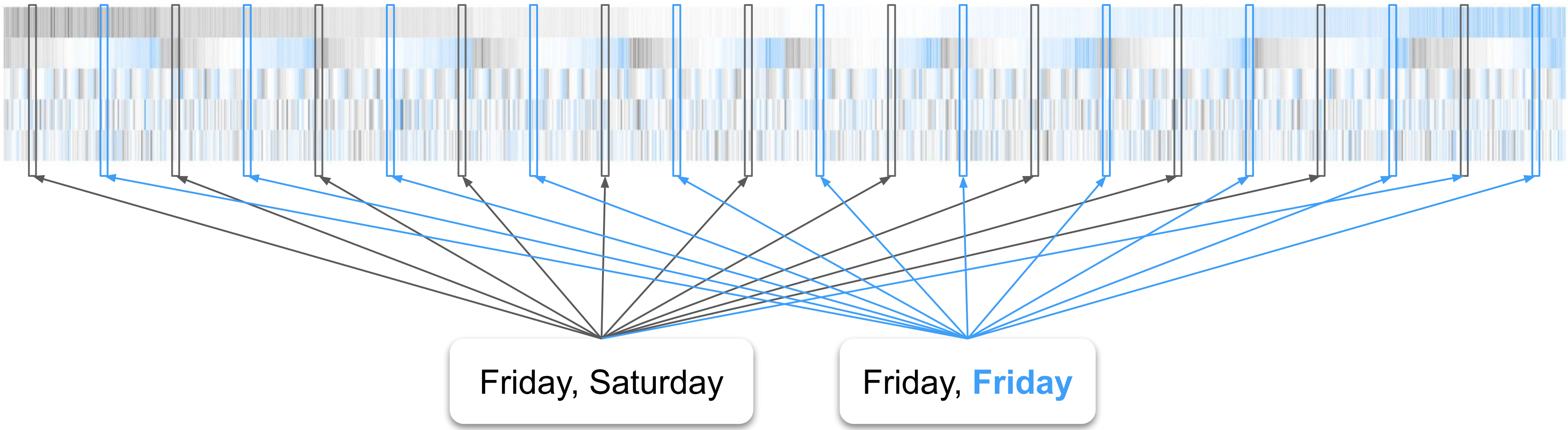


We can **adapt** to talk to our users the way they **want** to be talked to, simply by keeping the messages running.

We assign tests within clusters of similar users, just like we did in the beginning, but now we can assign users messages for which they have **high** personal scores.

I’ve already told you about what I like. I want to see you’re listening!

Fill messages with winning policies.



If users generally **respond** three times as well to Friday messages than Saturday messages, then **assign** Friday messages three times as many times as we assign Saturday messages.

That makes our tests not completely random. That’s ok.

We include **each** user’s personal score in the model. Like we said, the model has a **really good imagination** when given the right data.

The adjustment in how we make assignments allows us to start **acting** on what we’ve already learned, even though we never stop **learning**.

Learn and do at the same time!

Keep getting better

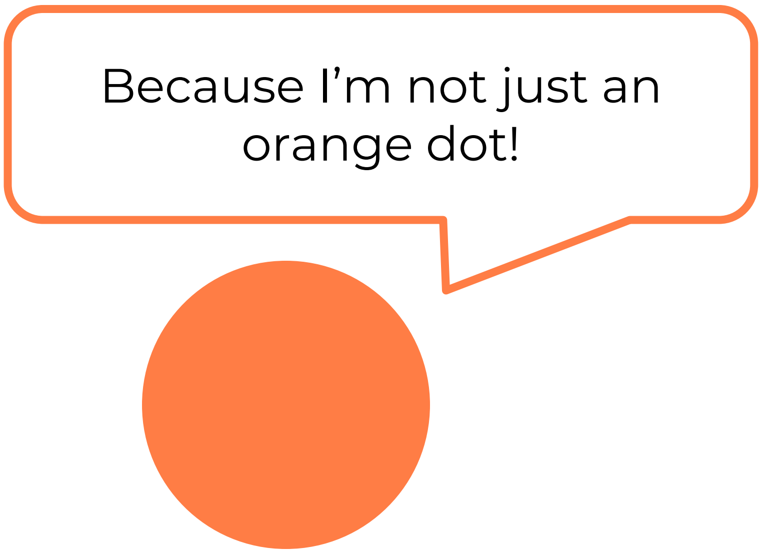
The more messages we send, the more we learn what users like. The more we learn what users like, the more our next messages reflect what we’ve learned.

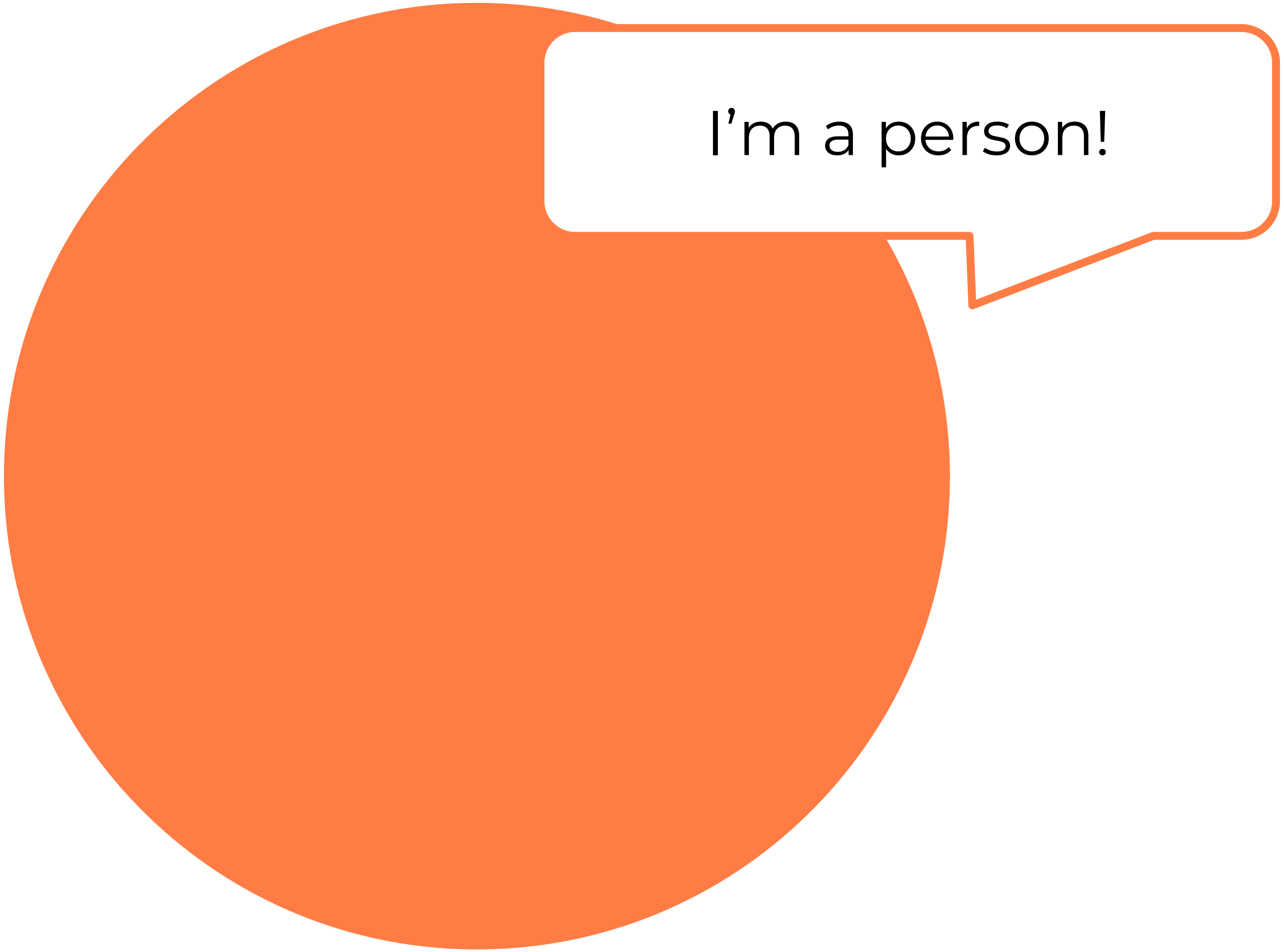
Over time, our messages comes to match user preferences more and more, automatically.

Part 5

The point!

All of this is really hard work.
Why go to all this effort?




I’m a person!

And I want to be
TREATED LIKE A PERSON.

Users are people.

Most people expect pretty **basic** things from someone who says they want to talk:

1. Check in regularly.
2. Listen.
3. Show you listened.

Everyone knows this.

It’s not rocket science!

Most tools don't treat users as people.

Tools like segments, marketing automation, and predictive analytics (when used without a user landscape and planned message tests) do a lot of talking and almost no listening.

Listening and meaningfully responding requires effort when you're only talking to one person.

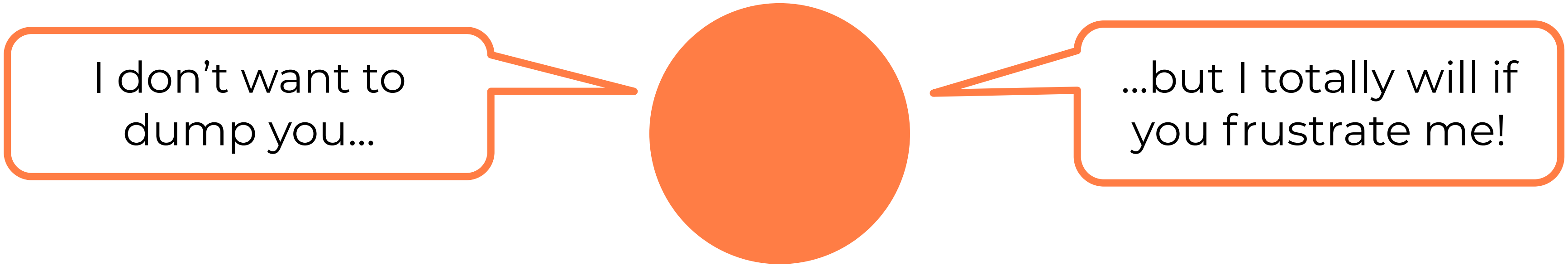
When you're talking to hundreds or thousands or hundreds of thousands, it requires more than just effort.

Ok, so this part is a little more like rocket science!

Users are ready to **walk** if you bug them.

Users give their attention to whatever makes them happy or helps them get things done. If you don't do that better than someone else, your users will go to the **other** guy.

That's how it should be. It's **their** attention. They can spend it however they want.



Treat your users how they want to be treated.

You need a system to do big, hard things!

If you want to both **listen** and **talk** to all of your users, you need to write **lots** and **lots** of messages, decide **what** to talk about and **when** to change the subject, and **connect** all of those pieces together over and over again.

Aampe lets you do that.

Talk to us. Your users will thank you.

aampe

The logo looks like me,
but with better hair!

This is the back cover. You've either finished, or you've started from the wrong end!