1:03

Alexey

**This week we'll talk about A/B testing and experimentation. We have a special guest today, Jakob. Jakob has more than three years of experience of growing data analytics teams. Now he's a head of analytics at Inkitt. He's passionate about A/B tests, defining metrics, and pizza dough. Welcome, Jakob.**

1:23

Jakob

Yeah, thanks for having me.

# Jakob’s background

1:25

Alexey

**I’m especially curious about the last point – the pizza dough. But maybe this is something that we'll talk about later. But before we go into our main topic of A/B tests, let's start with your background. Can you tell us about your career journey so far?**

1:40

Jakob

Sure, yeah. I guess my journey started with finishing my Master's degree in economics and econometrics –it was a quantitative focus. Back then I was super into economics and really wanted to actually be a practicing economist, which was surprisingly hard on the job market, but I found a job in economic consulting. Basically I did analysis of mergers and cost of damages and stuff like that. You can actually apply and run econometric models, like regression models and similar things. But the overall work environment wasn't super inspiring. What I took away from that was basically the fact that I liked working with data, but not in that industry.

Through a friend, I learned about an opening at King. King is the producer of Candy Crush and these types of mobile games, so they have tons of data. That was basically my start in a proper data science job. That's the first time when I actually started working with SQL, properly with R, and later, Python. I really got to enjoy the topic of A/B testing. Then later on, I moved to Babbel, which is a language learning company. There I pretty much did a similar job, but I would say at a company that had less data tools built out, which is also what was interesting to me. I really wanted to understand how to build up a data organization – how to create the tools and pave the way. I got promoted there to a team lead position as the team leader of product analytics. Later on, my team also took on the mission of building an experimentation platform – so also much more of a focus on A/B testing.

After that, I had the opportunity to join Inkitt, which is where I am now. It’s a smaller company than Babbel. We basically do online publishing, or specifically, we are a platform where users can publish their books. We analyze reading behaviors and have an algorithm that predicts how well these books will perform. If they perform well enough, we put them under contract and move them to the app where users actually pay for them. We also do tons of story-specific testing as well, which is also one of the big reasons why I joined. A small plug – we have positions open so if you're interested, I’d be happy to chat about it.

5:00

Alexey

**Is it for novels? Or for what kind of books?**

5:04

Jakob

Yeah. For novels.

5:09

Alexey

**Not for technical books?**

5:11

Jakob

No. [laughs] Maybe that's something that would be interesting for the future. But so far, it's strictly fiction.

5:19

Alexey

**So you get a manuscript, you run it through a model and then if the model says it's going to be a bestseller, you sign up the contractor?**

5:28

Jakob

Basically. But it is based on how users read it. We put it on the reading platform, and then we have a paid platform. Basically, on the paid app are the successful books from the former.

5:43

Alexey

**Ah, that's quite interesting. So this is more like for indie writers – people who want to publish independently, right? Not go to big publishers. I don't actually know the names of any.**

5:57

Jakob

Yeah, exactly. I mean, in some sense, we are then the publisher. It is kind of like a publishing house. I think, in the past, we sold books on Amazon, but we've pretty much stopped doing that and the focus is really on this app, through which you can read.

6:18

Alexey

**I'm also curious about your econometrics background. You said you first started with that. I remember taking courses about econometrics and what surprised me is how rigorous it is compared to machine learning. In machine learning, you just throw in data, then you do cross-validation and if it works on cross validation, you're fine, right? But in econometrics, you need to do a lot of analysis, you need to find these correlated variables and all these things – you need to be careful with them, remove them if necessary. Is this a correct observation?**

6:55

Jakob

Yeah. I think because the aim of econometrics is more to learn about relationships in the data and less about predicting. I think the whole science behind it is based on the idea that economists want to understand the world and see if their models actually fit. So they collected data and they started kind of using these regression techniques to understand causality like, “Is *this* doing *that*?” And of course, causality is very complicated when you work with observational data. That's also why I think this whole field of econometrics is very specific about regression techniques and stuff like that – how the errors are distributed and where you need to be careful. So I think it's much more about hypothesis testing to some extent.

7:54

Alexey

**Yeah, I remember there are a lot of tests like, “Is this normally distributed?” Then you run a test, and then the test says, “It's not,” or “the p-value of this is *this* much.” This is when you really get into statistics and really get into statistical tests, right?**

8:10

Jakob

Yeah.

# The importance of A/B tests

8:13

Alexey

**Can you tell us what an A/B test actually is? Why should we care about this test if we're not somebody who is in econometrics, but let’s say in data science?**

8:26

Jakob

Sure. So, A/B test – I think of it as what everybody knows at least from popular science, and I guess, particularly from the corona pandemic there are clinical trials – how we actually find out whether a vaccine works or not. These clinical trials are usually split into two groups: you have a treatment group that actually gets the proper vaccine, and then you have a control group that doesn't get the proper vaccine. Maybe they get a placebo or something like that. Here, the participants are kind of randomly split into these two groups, and then you kind of track whatever outcome you're interested in over time. An A/B test is very similar to that, I think.

Well, I don't know where A/B testing strictly starts, but I would say, in the online world, that's where the whole idea came from, or when it started being called an “A/B test”. It is really about taking some unit of randomization – that may be a user or that may be a specific session, for example – and basically testing specific experiences on users. The classic example is a button color. I don't think anybody really does that, because oftentimes (at least at the companies where I worked) we had bigger things that we wanted to improve upon. But the idea is really that some users will get a control version of something – basically, the old version and then another group of users gets something in addition to it – for example, a new feature redesign or something like that.

We track the behavior over time, we measure it and then we basically compare which of these two groups performs better. Usually, that's done with some statistical analysis to control the noise of the experiments and conclude based on that. So that's the overall idea of an A/B test – in some way, a fairly old idea. I think the cool thing about it in the online world is that you get results so much faster when you talk to people that actually work on clinical trials and things like that. They have to run studies for years. If we have enough users on a certain website or a certain product, we can do it in weeks, or even days.

11:20

Alexey

**In clinical trials – the example with vaccines – there are two groups. One gets the proper vaccine and the other gets something that’s not the vaccine. Then there is some metric that is measured, for example, “After half a year, how many of them get the disease?” This is the metric that is measured, and it is expected in the treatment group that this percentage will be lower than in the control group. Is that right?**

11:48

Jakob

Yeah, exactly. The idea behind why we do A/B testing in the first place is really to establish this causal link of the change and the impact. Otherwise, as I mentioned before, econometrics is so sophisticated because economists have to work with observational data. You can't do experiments on entire countries or something like that. That gets very hard – and it's unethical, of course. But in the online world it’s much easier to do. At the same time, we have so many external factors that may otherwise influence our metrics that it often gets really hard to establish whether something had an impact unless it's really big. This is why we need these types of experiments to really make sure of what we're actually doing.

# Statistical noise

12:44

Alexey

**This is what you mentioned. You said “control the noise,” right? So “noise” is all the factors that we don't want to have control of. We want to make sure that whatever we're observing is not affected by that. Because I guess, let's say with the example of the color of a button, you just think, “Okay, maybe a green button will convert better than a blue button.” You change this button – you deploy this change – and then on the next day, you see that more people bought and you think, “Okay, this probably is good.” But maybe something happened. Maybe there was a promotion that your marketing team ran and you didn’t know about it, so that is why more people clicked on this button.**

13:25

Jakob

Definitely. [laughs] I think you reach that point quickly when you're just in a company where it's very hard to keep track of what everybody's doing and there are so many different factors. That just gets very hard. These external influences are a big challenge. The other source of noise is just that every user is going to be a bit different. Even if you look at the exact same demographic, and the exact same situation, they may just behave differently. So on the one hand, with this experiment setup, we can control for all of these external factors – hopefully – if we design it properly and then we just deal with this individual noise. For that, we can use statistics to say whether this is actually a measurable effect, or that what we’re seeing is just by chance.

# A/B test example

14:27

Alexey

**Can you maybe give us an example from something you did recently? An A/B test that you ran – maybe something simple?**

14:38

Jakob

Let's see. Well, something that we did on Inkitt, which was basically a big project, let's say. On our commercial app, called Galatea, to be able to read the next chapter of a book, you either have to wait for six hours, or you basically purchase points. Points are like in-game currency, so to say. You can purchase point packages in our store. But we're thinking about introducing a subscription model, which basically says, “If you have the subscription, you can go on reading as much as you want.”

15:33

Alexey

**This is something that Babbel, and other games from King do quite often, right?**

15:38

Jakob

Exactly. That's a really great example of something that can have a huge positive impact for the business. But at the same time, it has certain risks. What if users are scared of the subscription and don't want to subscribe because it's much more money up front than what they would be doing with this point system? This “revenue per install,” for example, is something that is quite hard to foresee in which direction it will go. Then again, we have so many ways of how to design a subscription model. We can do a twelve month subscription, we can do one month subscription, we can change the prices, we can change the prices of the point packages in comparison to it.

There are so many degrees of freedom, which, of course adds the opportunity that if you get it wrong at first, there's things to change. But this also means if you're not aware of whether this was a success or failure, you can also cause a lot of harm to your business. Whoever works with revenue data in a freemium environment probably knows that this is one of these metrics that fluctuates the most on a user level. It's really hard to measure. Therefore, if you just roll something out, you can’t hope that from one day to another, it just goes up and stays up, or goes down and stays down. These level changes are going to be unlikely, so you really need an experiment to actually compare the differences.

17:24

Alexey

**In this case of the experiment, you have two models – the old model was with points or without? I don't remember.**

17:32

Jakob

The control was with points – so points only and then the test was points and subscription at the same time.

17:39

Alexey

**I guess, here, the metric of the outcome that you wanted to measure is some sort of revenue-based metric, right?**

17:47

Jakob

Yep.

17:48

Alexey

**Then you would look at how much revenue per user each group would generate and based on that, you would decide whether you should stick with the old version or go with the new one. Right?**

18:04

Jakob

Exactly, yeah.

# A/B tests vs expert opinion

18:06

Alexey

**So why can't we just trust our gut feeling? In this case, let's say you have this experience from King and you know that for users, they really like this gamification system, and that some of them would actually pay to be able to go to the next level of Bubble Witch or some other game. So can't you just trust your gut feeling and say, “Okay, trust me. It will work. Let's go ahead and deploy it.”?**

18:34

Jakob

Yeah. I mean, gut feelings are generally great just to come up with ideas. I also think there are probably some decisions where maybe it's too hard to test – maybe it's too much effort to test – or something like that. Then expert opinion is a good way to go. But in any case, I think the challenge with expert opinion is that you can devise, saying “Users like a subscription” or “Users like a certain type of feature,” but then how you implement this general idea of a feature can still go a myriad of different ways. Some implementations may work perfectly and you can trust your gut feeling or your expert vision, but some may also be just not very user friendly or something like that. And those are the ones you probably want to understand or catch early.

I think this “de-risking” of a feature is a huge aspect of why we should be doing this. The other aspect is – if you're working with a product team that is continuously iterating on a certain product component or something like that, building new features and iterating, then it's also good for them to have specific feedback, like “This change that we did gives 10% more revenue,” or something like that, or “This change only gave 5% more revenue and when we tried *this*, we actually failed and revenue reduced.” To have these learnings is actually what builds up expert knowledge. It's basically about the organizational learning as well. The more you test, and the more you test iterations on a similar field, the more you can understand how users function and what makes them tick. At the same time, you learn how you can maybe transfer that knowledge later on.

For sure, with most tests – if you design them in a good way, you can probably abstract some kind of learnings that you could apply somewhere else. Here's an example from when I worked at King. Obviously, King has a whole portfolio of games. They had some huge games like Candy Crush, and then they had some small games. And in small games, it's harder to massively break something. So oftentimes they did little experiments, or more risky stuff on the smaller games. When they found what worked out well, then they could just copy that over to all the games. In that way, I think that's a really great example of how you can multiply and transfer your learnings quite quickly. With other products, it may not always work that directly, but indirectly, I think you can still get to learn a lot about your impact as a team and the end-users that you're serving.

22:04

Alexey

**To summarize why we should go with A/B tests and not our gut feeling – the first thing is that a gut feeling is good at just giving you some initial idea, but there are thousands of ways of implementing this idea and you don't know which one works best. The only way to find out is to iterate and see how users react. This way, you learn from users and you can attribute and see that “This feature brought an X amount of uplift to the metric we care about, for example, revenue.” Then we know “Okay, this is what users really like.” Then we can maybe rank the features by the impact and you have this learning as a result. Was there something else?**

22:46

Jakob

No, I think that's a good summary. Maybe one more aspect is – imagine a situation where from one day to another, your revenue cuts in half. It’s a huge crisis in the company. “What happened?” If during the previous couple of days, you just rolled out new features without A/B testing – you wouldn't know if it’s because of them. Like, “Did they ruin everything?” You roll those releases back and hope for the best. Maybe that fixed it or maybe not. But with an A/B test, you could just open the data, have a quick peek and say, “Okay, both groups go down. It's something else.” So that also helps – just to know that effect.

23:29

Alexey

**Yeah. I realized that you mentioned that they de-risk things at King. If you have a big game like Candy Crush, you want to be very careful with the changes that you introduce because they will affect a lot of users and it brings a huge pile of revenue. So you want to be super careful there and you want to test if every change you make actually brings uplift.**

23:53

Jakob

Yep.

# Traffic splitting, A/A tests, and designing experiments

23:54

Alexey

**Now we know that A/B tests are good. We need to experiment – how do we start with this? Let's say you join a company – a startup. At the startup they do not experiment yet. They already have data in the data engineering team, or maybe a data engineer is a single person who puts together some infrastructure. So the company is already tracking some events and these events, hopefully, end up in a data warehouse, where we can use BigQuery or Athena or some other tool – Snowflake, I think is popular these days. We have this infra, and you join this company as a product analyst.**

**How do you go about setting up A/B tests there? How do you set up an experimentation culture in this company?**

24:44

Jakob

Yeah, that's a good question. I've never been at that stage where I had to set everything up from scratch. One thing that I would probably think about first is that there are two roads to take. In some sense, they have major implications down the line. On the one hand, there are third party tools that you can purchase that give you the “all-in-one” package of doing A/B testing, doing the analysis, and so on. Stuff like Optimizely, Google Optimize, Firebase – they all have varying amounts of features. Probably, Optimizely is the most complete and it serves you a nice analysis dashboard at the end. I have never actually worked at a company that uses these third-party tools.

At King, Babbel, and now also at Inkitt, we kind of have our in-house tools. I would say, it's probably much more effort to actually do it this way. But at the same time, you have full transparency. So I think I can't really give fair advice regarding which road to take when you're at the very beginning. But I would say it depends a bit. Probably at the beginning, the experiment setups are going to be very specific and probably a bit basic. Maybe then it's fine to use an external tool that just splits the traffic quickly and gives you some feedback in terms of events. You can choose either to do the analysis manually or to look at the dashboard or whatever they serve you. The nice thing about that is that you, as the first product analyst, are probably going to be the poor person having to do all the work. [laughs] If there's a tool that already does something like that automatically, maybe it's good. That's one way to go at the beginning.

The other way would be to really work with engineers and build your own traffic splitter – something in the backend that basically can receive an API call and then say “this is user B” or “this is user A” or something like that. Then whatever product we're working with, (let's assume it's an app or something like that) the app can then receive that, and show a specific experience based on that. I mean, a traffic splitter doesn't do something that’s very difficult. You just have to be careful that it actually randomizes whatever it’s doing, and that it randomizes on the right level. “What unit of randomization is it? Is it the user ID? Is it the session ID? A cookie?” Things like that.

27:52

Jakob

My next suggestion would be to build as much tracking around this process as possible. Not only tracking “Here's the assignments. This is user A. This is user B.” But you actually want to understand if the app always calls the traffic split at the right time. “Does the traffic splitter always give a sensible result? Is that received properly?” These sorts of things. Because this is where the first complications that can start. What if the internet connection is bad? What will the app do then? Will it just say, “Well, don't have a call. So I'm just gonna say that this user is in Group A.” That's already the first point where that's not how it should work. Because then every offline user is a control user all of a sudden, and you're going to get very biased results. There are also these sorts of things to consider.

So from scratch, I think you need to build out a system that you can fully monitor and trust. If you don't trust that system, then you can forget about your A/B tests. A good way to understand what's happening there is to do an A/A test, where you just let the traffic split, but the app will just show each group the exact same thing and you just track that. You have two groups and they see the same thing. You should ideally be able to measure the same thing, but the only differences are due to chance – due to users being individuals. But for example, if you've 50/50 split between test and control, then in an A/A test, you can at first understand “Well, are the assignments actually 50/50? Or is it 60/40?” That would be a warning sign that something's going wrong. Or other experiment results were tracking some conversion rates, where in Group A, it's 40% and in group B, it's 60%. Then also probably something's going on with the randomization. So that kind of makes the tool trustworthy.

30:05

Jakob

I think it's also worth doing that with external tools, because at Babbel, we had the experience that we couldn't really trust them. That was actually what was going on. They gave us these not really clean tests where we wanted 50/50 splits and we got 55/45 and we didn't know what was going on. So I think that's kind of like a trust building technique. Then the next step is to really work with a team that does the first test and properly design it. Designing the experiment means properly planning out, “Which groups do we have? Which kind of metrics do we want to make a decision based on?” There should be one metric, there shouldn't be five, or something like that, because then the results won't be very clear.

Ideally, there should be some idea of expected impact. You'll probably want to understand the metrics before you actually run the test. So this means understanding things like, “Are they super noisy? Are they fairly stable?” These sorts of things – to kind of understand and plan the duration. You can apply statistics for that. At the basic level, you need to at least understand, “Okay, revenue per install is probably super noisy. Whereas a click-through rate is maybe something that's fairly stable.”

My recommendation with the first A/B test is to just keep it as simple as possible. Don't plan more than two groups. Don't do fancy metrics, where you don't even know what statistical test to apply later, or something like that. Don't put it in a very strange logical position that makes it hard to track or that makes it challenging for the product to handle the request to the server to receive the assignment. These sorts of things. It should be something where you can understand that the technical environment is not going to give you too many headaches. In the end, it's about making sure that you catch the data in the right way and do the analysis.

32:39

Alexey

**To summarize – there are two routes, which are third party tools and building the tool in-house. So you need to select the route you want to take and then you need to get a good first use case, which should be something simple. Then you need to understand the metrics and maybe this is something we can talk about a bit more. But the first thing, as you mentioned, is that we need to have trust in what we are doing and the first thing we can do is run an A/A test. In an A/A test, the split at the end should be 50/50 and the results in each group should be similar. Once we have that and once we have trust in the tool, then we work closely with the team and design the experiment.**

# Noisy vs stable metrics – test duration and business cycles

33:23

Alexey

**You also mentioned a few things, which are – we need to understand the metric, we need to understand if they are noisy or stable, and what the expected impact is. I have a few questions about this. First of all, what is a noisy metric? What does it mean for a metric to be noisy?**

33:40

Jakob

Well, I think the simplest way to look at noise is to just track a metric over time. Look at conversion rate daily, or revenue daily, or something like that – if it fluctuates a lot like a stock price or something like that, then that means that it's probably very noisy. Think about if you had two of these lines that are going like this [waves hand up and down to signify heavy fluctuation], and you need to be able to distinguish and say just from looking at it, “Is it easy to say whether one line is like clearly outperforming the other line or not?”

In the end, that’s what's gonna be spit out of the experiment and you have to make sense of it. If it's two lines, and they just go like this [draws lines with finger to show clear distinction] then yeah, it’s very easy to say, right? But if they go like this [draws lines with finger to show jumbled results], then it's going to be much harder to say. That's kind of how I would think about noise in an intuitive sense. Of course, you could say “the standard deviation is very high” or something like that, for some statistics. But I think the intuitive way of looking at it is how much variation you see when you look at a time series.

34:59

Alexey

**And the opposite of a stable metric is a noisy metric, right? So stable means that it doesn't oscillate too much – it's not jumping back and forth. When you plot a metric for group A and group B, you can clearly see that this blue line is better than this green line and you can make a decision, “Okay, this group is better.” But if they're constantly overlapping, then it's very difficult. Right?**

35:29

Jakob

Yeah. Another aspect of that, which is relevant for a lot of metrics and that I think you have to really analyze for each product is “Are there any specific seasonality or patterns in the data?” For example, for entertainment products, you're probably going to have higher usage on the weekends rather than the weekdays – these sorts of things. That means, for example, that you probably shouldn't run an experiment that *only* runs on the weekend days and then it's done. You probably want a full weekly cycle to have unbiased results.

36:13

Alexey

**What could be an example of such metrics? Like how much time people spent in games? Probably, they spend more time on the phone during the weekend, right? Because during the work week, they have to go to school or work. Is that right?**

36:31

Jakob

Yeah, something like that. Game rounds per user will probably be higher on the weekend. Retention rate is also a very common metric and will probably also be higher on the weekend. Each of these have these weekly cycles, but there may also be products like TaxFix, which is this company where you can submit your taxes in Germany with their app. Well, they're gonna have a big boom at the beginning of the year, or the first half of the year – but then in December, there’s probably not that much going on.

There are business cycles that are on an even larger scale that you have to anticipate. Of course, we're not going to run an entire year-long experiment to control for that, but I think you at least have to be aware of them. Because that's also going to determine how many users will actually go into the experiments – how much traffic it actually gets.

# Z-tests, T-tests, and time series

37:44

Alexey

**You also mentioned duration. For metrics that have these oscillations – there are more users on the weekend, for example – we need to plan to at least cover the entire week. What do we need to think about when we're talking about duration – when we plan for how long we want to run the experiment?**

38:09

Jakob

Well, in typical frequentist statistics, there's something called power analysis, which is something we could apply. The idea is basically to plug in all the requirements that you have for your experiments and some statistical properties of the metric that you're looking at. Based on that, it will spit out how many observations you need in order to make a decision. The idea is, basically, that we want to make sure that we have enough observations to reasonably detect something – for example a 5% improvement on some metric or something like that. Then, based on that, you can take this 5% improvement of the metric and the metric itself, its standard deviation, and mean – that really depends on how the metric is statistically distributed – but if it's something like a conversion rate, it's fairly straightforward to basically use the idea of a Z-Test. For that, there's a simple formula for the power analysis, and it will spit you out something like, “Okay, you need 2000 observations,” for example.

Based on that result, you can actually look at your daily traffic, like “How many users do I get every day on that page where the A/B test triggers?” And just estimate “How long do I need to run this test, roughly?” That's something that on the one hand, you can use to determine, “Well, at this stage, now I can do my analysis.” But you can also talk to stakeholders and say, “Okay, we need to get this running for roughly three weeks.” Oftentimes, once you have an A/B test running, as an analyst, you're going to get some questions about it from the product manager one day later. Basically, “How's it doing? How’s it doing?” [laughs] It is dangerous to give into these things.

40:23

Alexey

**You mentioned that there is a simple formula. I saw these calculators and you don't even need to look at the formula – you just go there and it's an online calculator. You go to it, you put in some numbers, and then it says, “Okay, you need *this* many samples.” I'm wondering – how much statistics do we actually need to know as data scientists or as product analysts in order to be able to run A/B tests? If we have these simple formulas, or if we have these online calculators, do we even need to bother with knowing the internals or can we just use these calculators?**

41:05

Jakob

To be honest, as I said before, in the most basic format, you could also just look at time series. If they look very different, you probably don't even need statistics for it. For very obvious uplift between test and control, you probably won't need statistics. I think the statistics will only tell you something when you can’t tell with your own eyes any longer regarding what's right and what's wrong. You don't necessarily need somebody who's super advanced in statistics. If you did a university course or something like that and refresh your knowledge on how hypothesis tests work, I don't think it's super necessary to be able to derive a formula or something like that. But, you should know, “What are type one/type two errors? How do I control for them?” Setting the confidence level and understanding what the confidence level means, understanding how to interpret p-values and these sorts of things. I think that's the most important thing.

The next part, if you're only testing these rates that are always between zero and one – like conversion rates and retention rates, I think that's enough. But the challenge is, when you go into metrics that get a bit more difficult to describe statistically – such as revenue per install, which I think is a great example. There's also the T-test, half the world probably applies the T-test to everything. It may also be a bit misleading at times, especially with this data, where for 90% of users the outcome is zero and then for one user, it's like 59 or something like that – you see these very extreme values – that's actually where you have to be really careful in your choice of statistical tests. Then, I think, it really helps having a bit more in-depth knowledge of statistics.

So I think it really depends on how advanced it gets – if you have very complicated metrics, it's definitely worth having somebody that knows a bit more about it and knows how to choose an appropriate distribution and these sorts of things. Oftentimes, nothing super bad will necessarily happen if you don't choose the perfect tests, but you may have to wait way longer or something like that – you may need way more data than necessary. For example, if you can't apply a T-test, so you do a nonparametric test or something like that, in terms of what kind of uplift or increase in the mean it can detect, it'll be less efficient. It'll take so many more observations.

44:39

Alexey

**How can we pick this up? Let's say, for those who have no idea what (T-test is probably quite a widespread thing) but for those who don't know what a T-test is or what a nonparametric test is, but maybe they did some machine learning or some analytics in the past. So what could be a good resource to pick this knowledge up? How much math do we need for that?**

45:09

Jakob

I would say that actually not a lot of math is necessary. If you are familiar with Python or with R, there are going to be tons of packages where you can apply these sorts of tests. I think the more interesting part is to actually look at histograms, see how they fit distributions, and these sorts of things. So that's kind of where the statistical modeling comes in, like, “Does the mean of my data look normally distributed?” If yes, then life is very easy, usually. If it looks similar to normally distributed, but it has very fat tails and you have a much higher chance of seeing extreme values, then it gets more difficult. I think it's mostly about visualizing and making sure that you understand the distribution of your data, and that you maybe know some basic distributions that you can throw on it and compare it with. But I don't think you need to be super into math, unless you're doing something like proper research on statistics.

# A/B test crash course advice

46:23

Alexey

**Do you know of a good crash course on A/B tests that does not involve a lot of math – ones that are more practical? “If your data looks like this, use this test. If your data looks like that, use another test.” Something like that?**

46:37

Jakob

Honestly, I don't know. I'm pretty sure there are tons. I would say there are probably more that you can find online than in real books. I've rarely found a good book on A/B testing that is not essentially a statistics course or how to apply statistics in Python or R. There's one book I can really recommend, but it's more of a practical guide. It touches on almost everything except for statistics, which I think is oftentimes the harder part to obtain. Also, my advice for when you start to do A/B tests, it’s probably only 10% of the effort to think about how things are statistically distributed. There are surprisingly so many other things that are going on and can go wrong.

47:44

Alexey

**Like thinking of metrics and things like that?**

47:46

Jakob

Yeah.

What is a p-value

47:44

Alexey

**We have quite a few questions. We already talked about the frequentist approach and the question is related to that. “Can you please explain it like I'm five years old – what is a p-value?” Or maybe five is a bit tough – maybe like I'm ten years old?**

48:12

Jakob

Well, I'm just going to try to explain it in simple words, but I'm not sure if that 's appropriate for 10 year olds. Basically, it gives you an idea of how likely it is that you would see the results from your test. Let's say the test group has 5% better performance than control. How likely is it that you would see such a result in an A/A test, basically. So there, you don't have a treatment – it's the same thing – you test the exact same thing in both groups from the same user population, and you would still get a 5% uplift. So, the p-value is kind of like an indication of “What are the chances that this is out of the ordinary? That this is not by chance?” So the lower the p-value is, the more likely that there's something else going on than just the typical noise that you see.

49:29

Alexey

**Now, this is such a good explanation, because usually, it’s something like “The p-value is the probability of rejecting the null hypothesis under…” and when I read this my mind just explodes. [laughs] It is very difficult to understand all this “Probability of rejecting…” stuff. But the way you put it is “You compare it with an A/A test, and what's the probability that you would see it under an A/A test?” That makes a lot of sense and this is a lot easier to understand and probably to explain to people who do not have math background – who haven't studied statistics – because if you start telling them about null hypothesis, their mind will just go blank.**

50:12

Jakob

Yeah, exactly. [laughs] That's a funny thing. I think you need to strike a balance. As a data scientist, people are not going to trust you, if you explain things in a complicated way, or they're just going to be impressed. There are also these typical people like, “Oh, I feel you're the smartest guy in the room! Whatever you tell us is probably a super smart idea.” But if you can put it in simple terms, they'll understand it and they'll probably believe in the results more than if you keep it very abstract. When I interview people for these types of analyst positions, I just always ask them to explain [laughs] these sorts of terms in a way that a product manager with zero statistics background could kind of relate to. Of course, it's always going to be a bit simplified or simplistic and a statistician would get nervous or something like that, but at least it's not misleading, you know?

51:24

Alexey

**Do you remember any good answers that you got aside from this A/A test explanation?**

51:34

Jakob

I think it's always like a similar answer. So far, I've actually not had so many. Well, maybe I'm also biased myself, just because I answered it. [laughs] But no, off the bat – I can’t say.

# Frequentist approach vs Bayesian approach

51:55

Alexey

**What do you think about this “other way” of doing tests? The “other way” meaning – we have the frequentist approach to testing and then we have the Bayesian approach to testing, which I think (correct me if I'm wrong) but there is no notion of a P-value in a Bayesian test. Right? What do you think about this?**

52:12

Jakob

Often people take very strong sides and I think Bayesian is a bit more popular because it sounds much more exciting and it's a bit more like the hype these days. I think both approaches work. And both have some pros and cons. The frequentist approach is well adapted, as we mentioned – we have these online calculators, we have methods, or packages implemented in almost all data programming languages that let you analyze data in a very convenient way with frequentist statistics.

Almost all people that had some statistics course will have been exposed to this basic idea of hypothesis testing. They have heard about confidence intervals, about p-values, and these sorts of things. So the terms that you work with are well-established and they are usually easy to calculate. Computationally, it's cheap – you can think about automating A/B testing in a large company where you run hundreds of tests and you want to update the results every day, then it's nothing to worry about, basically. But I think the challenge is, as you said, the formal definition of “What is the confidence interval?” and “What does a p-value mean?” is fairly abstract. It's not something that we relate to in our daily lives.

I think the other drawback is that each test is based on various specific assumptions and you fairly quickly may run into a situation where you just don't know what a good test to fit your data is. Then you can apply these nonparametric tests, as I said, but they may not be super efficient. I think the Bayesian approach, on the other hand, the outputs that you get from it, you can construct very intuitive things. A credible interval is actually something much more intuitive than a confidence interval although they look the same.

55:01

Alexey

**What is a credible interval?**

55:03

Jakob

It just literally tells you “There's a 95% chance that your mean is within *these* boundaries.” Whereas the confidence interval says basically, “If you repeat this experiment 100 times, then 95 out of these times, you get a confidence interval that may not look like this, but the true mean is in there.” [laughs]

55:30

Alexey

**That's super confusing. [laughs]**

55:31

Jakob

Yeah. The credible interval is super easy to work with. You can calculate win and loss probabilities – you can say “The test version of my A/B test has a 60% chance of winning.” With that, it's super easy to talk to product managers and discuss with them, “Hey, we're going to roll this out if we see an 80% winning probability.” So that's a huge pro of the Bayesian approach. The outputs that you get are very intuitive. I think another factor is there's a lot of very explicit modeling in it, but you need to understand a bit more about the statistics behind it.

You can choose priors – you have to think about what priors to pick, from which distribution they come from. Then you have to model the actual data distribution. Based on that, you're going to get some posterior. So there's a lot of modeling choices that you can make and based on that, you can have really great result quality, maybe with less observations than the frequentist approach. But you can also do things [laughs] wrong, very easily. If you just throw some data into some random model, and then it computes something.

57:10

Jakob

And the other thing is, again, you're very quickly going to end up in a world where there's no analytical solution for the approach that you take. This is when you have to do simulations like MCNC modeling, which can get computationally expensive. At Inkitt, for example, we currently run hundreds of story tests. Now imagine, for every test, you have to do simulations – 5000 draws or something like that – just for one group of one test. [laughs] You need a powerful computational resource for that, for sure. That's one of the problems with it, for sure. If you want to automate something like that, it's harder to scale.

Another aspect that is oftentimes forgotten is that you can still run into the same errors that you do with frequentist statistics. Type one and type two errors are still something that you need to deal with, except there’s not such a clear framework for it. You can still make wrong decisions, or misleading decisions, and you can still not detect effects. I don't know, there may be ways of actually controlling that as well, like in Bayesian statistics, but the knowledge is not super available about it at least. I think it's much more of an open field. With frequentist statistics, there is tons of literature about problems and corner cases. And I think with Bayesian approaches, you quickly reach the end and then you have to be really confident in what you're doing yourself.

# A/B/C/D tests

59:08

Alexey

**I see that we are almost running out of time, but there is one question that perhaps you can answer pretty quickly. The question is about A/B/C/D tests. So, “What is an A/B/C/D test with respect to A/B tests? And when do we need this complex test?”**

59:28

Jakob

When do we need it? I think if you talk to product teams that quickly want to iterate on things, that's probably what they want to do all the time. But the idea is basically instead of having one test group, you now have three test groups. Going back to the stupid button color example, you don't only test like green versus red, but you also have a blue and you have yellow on the side, and you just run it all at the same time, and you want to find out which one is the best. You can do that – there's nothing that is methodologically problematic about it, but when you think about duration, splitting a population into two parts and reaching the required sample size for it is going to be faster than doing it if you have like 25% in each group. So the test will run longer to detect the same effect. That's one problem.

Jakob

60:42

Often, we all want to get things over with quickly. The other pitfall that can be – going back to this frequentist approach – basically, you usually set this confidence level to 5% and what that says is “I want to limit the chance of type one errors to 5%.” It’s basically seeing when a test says there’s an impact, even though in reality, it doesn't have an impact – the chances of that are fairly small. Now, when you have to do pairwise comparisons in A/B/C/D tests, then you're not only doing one test decision, you're doing A versus B, you're doing A versus C, A versus D, B versus C, and so on, right? All of a sudden, you have a much higher chance that one of those is wrong.

Maybe that's fine. Maybe you also want a limit and just want to say, “I want to compare A versus B, A versus C, and A versus D. I don't want to compare the three groups in between.” But in reality, of course you will. You want to do that. So then you kind of have to think about how to make sure that with that at some point, you're not going to end up deciding on something that may have an effect just by chance. The simplest example of that is – let's imagine we have 20 groups and you have this 5% chance. if you do 20 tests, and one of them will have an impact. 20 A/A tests by chance. [laughs] These sorts of problems can become very prevalent with multiple test groups.

# Pizza dough and finding Jakob online

62:52

Alexey

**Okay, thanks. I actually wanted to ask you about pizza dough, as well. We didn't have a chance to talk about it. Do you have any resources that you can recommend to us in order to learn more about this topic and A/B tests? Maybe how can we test the best pizza dough with A/B tests?**

63:13

Jakob

[laughs] I have never tested that. That's one of the things where I don't do A/B tests. Maybe I should do it. I think how to get good at pizza dough is just to practice a lot. Invite people over so that you can produce a lot of pizza dough and not just for one or two pizzas. Then, just keep on doing it. But yeah. So, in the summer, I got this portable pizza oven, which is a lot of fun. It heats up high enough that I can make Neapolitan-style pizza in it. Yeah, I'm getting very, very nerdy about dough. [laughs]

63:59

Alexey

**Okay, yeah. [laughs] You mentioned that you are actually hiring for a product analyst role, right? If you have any job descriptions linked to your job portals, please send them and I will include this in the description. For those who are interested, you will find the link in the description. I guess you can just look up Inkitt jobs in your favorite search engine and you will find that. If somebody has a question for you, how can they find you? On LinkedIn or some other platform?**

64:29

Jakob

Yeah. LinkedIn is probably easiest. Under my full name, Jakob Graff. Should be easy enough to find.

64:40

Alexey

**Okay, thanks a lot. Thanks for joining us today. Thanks for sharing your experience with us. Thanks, everyone, for joining us today as well and for asking questions. I wish everyone a great weekend!**