1:08

Alexey

**This week we'll talk about doing software engineering in academia. We have a special guest today, Johanna. Johanna has a formal background in psychology and computational neuroscience. She's now about to complete her PhD in the field of machine learning in clinical neuroimaging. She's joining us from the University of Melbourne, in Australia, where she discovered the field of research software engineering.**

**In addition to doing research software engineering and in academia, Johanna has contributed to several open source projects and she's an advocate for open source and open science. Also, Johanna is a very avid listener of this podcast. She once mentioned in a LinkedIn comment that she listened to every single episode of this podcast and I thought “I have to invite her.” [Johanna chuckles]**

**Also, you probably heard in a few recent ones that the questions were prepared by Johanna Bayer and this is the Johanna. That helps with preparation as well. It's a big pleasure to have you here on this podcast. Hi.**

2:19

Johanna

Hi. Thanks for having me. I'm really excited to be here.

# Johanna’s background

2:24

Alexey

**Thanks for being here. Before we go into our main topic of doing software engineering in academia, let's start with your background. Can you tell us about your career journey so far?**

2:34

Johanna

Yeah, sure. As you've already mentioned, my formal background is psychology and clinical neuroscience, with more on the clinical side. I started with a Bachelor's in psychology. Back then already, my favorite subjects were statistics, the biology subject of psychology, and also I had an elective, which was about artificial intelligence and the brain. I found it super interesting. Already back then I started also sitting in on computer science classes while doing psychology. That was in Bamberg, in Germany.

I'm German. Then I got into a kind of very elite Master’s in Munich that taught psychology and neuroscience. I continued there in Munich and I also took classes there. In the German system, it's quite easy to just do additional stuff compared to the Australian system. Maybe that's something you want to talk about later.

When I was in Munich, I applied for Bernstein Center for computational neuroscience, they have a program where they give you between your Master’s and your PhD, one year of funding to go to any university that you really like and I went to the Technical University in Berlin, where I did some modeling for neuroimaging, and then to the Translational Neuromodeling Unit in Zurich, where I also did some modeling for neuroimaging like auditory stimuli in the brain, basically. While I was in Zurich, one of my friends forwarded me a position that was in Australia and it sounded really cool.

I applied for this position, which was about creating a normative model of the brain and then comparing how people with depression score under this normative model. Basically, it's modeling a biomarker for depression in the brain. So I applied and I got the position. Basically, I decided within a week that I would move to Australia. I was there like half a year later, because of the visa and stuff. And I've been here ever since. I’m about to finish my PhD. I’m working already as a postdoc in predicting and profiling, depression, anxiety and schizophrenia.

Also, I work with biological features and markers – so brain stuff and also behavioral stuff. Apart from that, I do a bit of stuff on the side, like you said, open source projects. I've worked for NASA. I've worked for MathWorks, which is, let's say, a MATLAB kind of company. I've contributed to open source, been a teaching assistant, and I've led hackathons. And I've continued studying computer science, but at a German university. That's basically me.

5:27

Alexey

**Interesting. As a teaching assistant, which subjects do you teach?**

5:32

Johanna

Yeah, it's open science. So that's also what I do. I'm really passionate about this idea to make science more reproducible, which is a huge issue and also relates to our topic, actually, research software engineering. Research software engineering can help to make science more reproducible.

5:58

Alexey

**As a teaching assistant, you need to help with homework and answer questions from students, right?**

6:06

Johanna

Yeah. It's actually very funny, because I'm a teaching assistant at a university in California. So it's remote. We have a Slack community, and they get materials from the university. Then I also watch materials on YouTube and I answer questions. I taught some subjects, like intro to Git, how to contribute to open source projects, and stuff like that, and a bit of machine learning.

6:37

Alexey

**Is any of this public?**

6:40

Johanna

Yeah. The course that they put out is public, but the teaching stuff is not. It’s Zoom face to face. But there is a lot of free stuff that you can get into. This course is also free – you can just apply and get into it.

7:01

Alexey

**I'm asking because in the courses we have at DataTalks.Club, we usually assume some familiarity with Git and things like this, so there are some prerequisites. When people without these prerequisites come to learn from our courses, it's a bit difficult. The learning curve for them is steeper, some catch up and finish the courses, but some do not because they need to pick up extra stuff in addition to what we are teaching. Now it's good to know that you teach Git, so we can know where to send them.**

7:39

Johanna

Exactly. Also, I just recently completed an instructor course for The Carpentries? So I don't know whether you've heard about The Carpentries – it's a software company. They have scheduled curriculum for very beginner computer science and software programming courses. They also have an intro to Git course. It's very structured and you can also take these courses online. I'm basically almost a certified instructor now. I need to teach my first course. For these courses, people can just go through them by themselves – it's open. We can put the link in the show notes. I think that's a really nice place for people to begin, because it takes people where they are.

# Open science course and reproducible papers

8:30

Alexey

**I'm really curious to ask you now about this open science course. I know we planned to talk about this a bit later, but since we're talking about this now, I wanted to ask. In this open science course, what are the things that students study? What is the curriculum for this course?**

8:51

Johanna

The first one was Git. The next one kind of fits with the topic of the episode, actually. There's this idea about reproducible publication. Basically, the big problem that we're facing in academia is a reproducibility crisis. People put out papers, but then if other people want to reproduce their findings, it's just not possible, especially in my field. Neuroimaging, almost died from this. We're still recovering. Basically, there are papers that have titles like, “Why all research findings up to now are wrong”. Stuff like that – really, really sad things. [nervous chuckle]

There's this idea about reproducible papers, where you have stuff like, for example, markdown. You can write your thesis or your paper with your code. You basically just send this thing to someone and someone can, from scratch, from the data, reproduce the paper that you submitted. So how to do that, for example, how to put your code on Git, how to write tests.

It's not very common in academia to write tests to make your code bulletproof. Usually students are not so skilled with programming and software engineering in general, so their code is less reproducible. So it’s about teaching them sometimes very basic skills. But that's the idea. What else did we teach them? Open source – how to contribute to open source projects. I think I've said that. Basically, that's it.

10:42

Alexey

**How do you teach your students to contribute to open source? Do you say, “Okay, here's an open source project. Go contribute.” Or is it more guided?**

10:52

Johanna

Yeah, it's more guided. I started, of course, with kids. But what I like to do and what I think really helps is, if you’re in a Zoom session, where you have just a very simple repo and you teach people to make pull requests. You accept them and they see what happens. Because Git is such an interactive tool, it's so hard to learn it by yourself. So that's how I teach them. In my field, we have a couple of repositories that are not as big as, for example, NumPy, or SciKit Learn – these repositories are massive, and even skilled people get very overwhelmed with how to start.

But there are smaller repositories. For example, one of them is the Turing-made book. It's a book on reproducible research – a Jupyter book by the Alan Turing Institute – and people can just really easily contribute, because it's like a book. You can just contribute text or put some citations. This is usually something that people from academia are familiar with, like writing a little paragraph and putting some citations. That's a very good way for people to start. Usually people are way less intimidated if I show them these resources.

# Research software engineering

12:10

Alexey

**Yeah, pretty interesting. So, since the topic today is doing software engineering in academia (I think we more or less started talking about this) I wanted to ask you – what is research software engineering? How are research and software engineering connected?**

12:28

Johanna

Yeah, that's a very good question. It's actually a very new term. The old ways, as I've already indicated, is that people have some data, they make some analysis, they write a paper, and then they forget about the data and all the analysis. Analysis, in some fields, it's done in SPSS, so it's not at all. I don't know whether you know what SPSS is, but it's more like a GUI thing where you do statistics.

12:55

Alexey

**It’s like Excel, but more powerful, right? [Johanna confirms] It’s a tool from IBM. I think I used it in my data mining classes or something like that.**

13:05

Johanna

Yeah. You click a lot, but it's not bad. The models that are in there are really good, but you click a lot. So how would you reproduce that? This is a problem. Research software engineering is basically people who take on doing proper analysis – they really focus on the methods and on the analysis. That is one part. They still focus on publishing papers.

The other part of research software engineering is people who publish software as part of their academic output. So they don't publish papers anymore, but they put a toolbox on GitHub. I think this can really help because these toolboxes and what they produce, it's really tailored to academia. It's done in a very good way and these people are really passionate about the software. I’m not saying that the other people are not passionate about the analysis, but it is not their main focus.

14:10

Alexey

**These people who publish software as part of their work in academia, are they PhD students or are they hired specifically to just program – to code?**

14:23

Johanna

No, they're usually PhD students. PhD students like me, for example. I didn’t publish a toolbox, but rather a method that corrects for side effect differences. Basically, when you collect data at different scanning sites, you have differences due to the magnetic field strength in the scanner or the vendor of the scanner. These differences can screw up your analysis. I published a tool to correct for these differences. It has nothing to do with the outcome. For example, differences between clinical populations and healthy controls are usually the topic of my field. It's really about the method.

Usually it’s people from academia. For me, I always thought that I'm this weird person who is really interested in these “middle” things – not really looking at the clinical population, not really the topic of my field, but kind of getting there. But then I learned that I'm probably a research software engineer. I'm just really interested in writing software as my research output. It can also be the other side, like you said. I think this comes about more and more often.

People from academia just work in their fields and they are forced to write software and sometimes they're not really interested in this aspect. This can also happen. And also they’re not really skilled in this aspect. It might actually happen that this comes about more and more often that, hopefully, software engineers get hired to do these things.

15:57

Alexey

**When you got hired for this position, what was the name for this position? Was it already called research software engineer? Or was it something else?**

16:09

Johanna

No, no. I have three paper applications for my PhD and maybe when we talk about it, we talk about the application paper, which is where I do stuff with clinical populations and a huge dataset and then there’s something we call the methods papers. But really it’s just writing software.

16:32

Alexey

**I see. So you still need to publish papers, right?**

16:36

Johanna

Yeah, that's the other thing. That's what we are hoping will also come about – software as research. This output is more and more, hopefully, acknowledged as output. For example, you can now get a DOI, a digital identifier for your software as well. Most people are happy to publish a manual-like paper or something like that. It's just that they don't necessarily want to publish in their field anymore.

# Convincing a professor to work on software instead of papers

17:10

Alexey

**I'm curious, how do you convince your professor to let you work on software instead of publishing papers? I guess many professors are used to only one KPI, which is the number of papers published by their department. Probably it should be like IA Star conferences or whatever – high impact publications. So how do you go and convince them that, “Hey, doing papers is good, but we also need to publish some software.”**

17:49

Johanna

Yeah, it is very hard. There are definitely people (PI's) that are more willing to do this, and some are less willing to do it. I was relatively lucky with my supervisor. We ran into this problem and I made it one of my topics and I'm very happy that she let me do this. But I have a really good supervisor – she's very flexible. But yeah, that's an issue. Often people do it in their free time because they're really interested in it, which shouldn't be the case. In academia, it's also – I don't know whether you know that, but academia has lots of fluctuations. The contracts are very short. So if you don't like where you are, you just move. [chuckles]

18:42

Alexey

**If you just move, then you have to start your PhD research from scratch?**

18:48

Johanna

Yeah, that's true. I mean after that.

18:55

Alexey

**I see. [chuckles] So it's more of a bottom-up approach – when PhD students approach their supervisors and say, “Look, there’s this problem in our field. We need to make our research reproducible.” Does it also happen in a top-down manner, for example, when the professor says, “Okay. Listen, everyone. Now in our department, we all care about reproducibility.”?**

19:18

Johanna

I know some professors, like one Stanford who kind of started this whole idea. He definitely does that. But it's less common. One reason for that is that many people who are in PI positions just didn't grow up with software or analysis like we do now, in general. They're a bit more scared of it. Also, they have less time. They're not doing the analysis anymore. Yeah, it's definitely a problem. On the other hand, like you said, it's a grassroots approach. There’s also these hackathons that we organize – they try to teach people and pull people in from the bottom to apply these principles.

# The importance of reproducible analysis

20:05

Alexey

**What you are talking about reminds me of the state of data science like 10-20 years ago, when it just started. Industry companies would hire data scientists who are just fresh from academia – very smart people with PhDs in physics, mathematics, and so on – and they would tell them, “Okay, there’s this dataset that we have, now go figure out how to make it valuable.” And then it didn't work out.**

**Companies realized that just hiring academics is not enough and that they actually need software engineers. Now it's usually expected that data scientists know how to program, that they know Git, that they know all these reproducibility things. On top of that, we also have these machine learning engineers, data engineers – all these engineers that help data scientists. Do you think I got it right that it's similar in a way?**

21:03

Johanna

Yeah. Like you said, it's a bit like 10 years ago. I think that we are at the verge and it's inevitable because digitalization is just everywhere. Regardless of which field you're in – even like in humanities or social sciences – at some point, you will have to do analysis and at some point, someone will ask you to be able to reproduce it. I think it's coming. Also it's very interesting, because I also think industry and also other fields have already solved a lot of the problems that we're still facing. For example, I attended the Machine Learning Zoomcamp and MLflow – it's great. [laughs] I'd never heard about this before. If you don't look out, if you don't search yourself as a scientist, academia doesn't teach you these things. The scientists themselves have to take the initiative at the moment. It's not good. It should be more top-down, but it's not.

22:09

Alexey

**Do you now use MLflow in your experiments?**

22:12

Johanna

Yeah, I really like it. [chuckles] It’s very useful.

# Why academia is behind on software engineering

22:16

Alexey

**Why do you think academia is behind? Why is nobody teaching these things?**

22:21

Johanna

I think there are many reasons. Like I said, I think many PI's really didn't grow up in their academic career with this and they're coming from an era where it was really okay if you just published a paper and burned your analysis. Nobody would ever ask you about this. But the other thing is that we have this problem that the digitalization of any academic field is there, but the people that work there are actually more interested in their topic. Someone who does social sciences or some humanities field – they don't want to do analysis, they want to work on whatever.

Take a social worker, for example, they don't want to deal with t-tests and statistics and modeling – they want to care about people. But they are now forced to do this and I think that's the problem. People are sometimes a bit reluctant. They're also a bit scared, they work in a different field, and now they're kind of forced to do this. It can also be scary. You put out work and then you basically give people everything, and they can find errors. They can say “Your analysis is wrong.” And then you have to reject a paper that you've been working on for a year or something. This can be scary. But in the end, I think it would help academia to have to kind of enforce reproducibility throughout.

23:54

# The problems with open science publishing in academia

Alexey

**What do you think is still missing? What kind of tools, like MLflow, do you want to adopt in academia, or you think should be adopted in academia, to solve these problems? Or at least start solving them?**

24:09

Johanna

Yeah. Tests. [chuckles] I've never seen anyone writing tests. It's just not common at all. And then, the idea of the reproducible paper – I think that would be really good. But the problem that I have is that I'm desperately trying to find a space to host it, because if you put it on Amazon (AWS) or whatever, you have to pay for it. For example, I make a figure and I want people to be able to interact with this figure – this is easily possible. In Python, you can do that locally on your computer – you can turn it around and hover over it and look at data points.

But if I wanted to make this available to other people, I would have to put it somewhere and then I would probably have to pay for it. This concerns not only short-term, but long-term because this paper will be out for a couple of years and people will still want to look at this figure. And I think there's no solution for this. I think the universities should provide some space for the researchers, basically.

25:14

Alexey

**Maybe Google Colab could be an alternative?**

25:17

Johanna

Yeah, definitely. But then… [cross-talk]

25:21

Alexey

**Then you have to scroll down all the way to find this figure and to interact with it. So it’s not a paper, right?**

25:28

Johanna

Yeah, it's not a paper. The idea of this reproducible paper would be that you provide the data as well. You provide the data, you provide all the scripts, and people just click, and from the raw data, it runs through the analysis, and it gives you all the figures. Being able to put that somewhere will be great, so I could just say, “Take this and just click a button.” Then people wouldn't have to run a Docker or whatever – just click a button and it will all come down. That would be great. [chuckles]

26:01

Alexey

**Do you use Python or R? [Johanna confirms both] I remember there was a time when I also used R – it was long ago – but remember that in R Studio, there was a way to publish. You write in the R Markdown and then you publish it to something called RPubs (or something like that). [Johanna confirms] But it’s still not interactive, right?**

26:28

Johanna

It is. I have done that. I did it with one figure. It basically gets pushed to an instance or something but then you can only run this instance for 25 hours per month for free and then you have to pay. It would be perfect. Another problem is, of course, the data that I'm using is imaging data – it's pretty big. [chuckles] I pushed it to this R instance and it was just this one little brain figure that you could rotate. It's a big figure and I was like, “Okay, this is one out of 15 that I have in my paper. [laughs] Okay. Maybe not.”

# The importance of standard coding practices

27:15

Alexey

**I see. So the things we need to bridge the gap between industry and academia, or at least start to, are: researchers need to learn how to write tests and we need to have a way to make papers reproducible – so there should be a tool that is free for researchers. Anything else?**

27:38

Johanna

I guess the teaching of standard coding practices to all academic fields. There should be one introduction to programming or statistics class for everyone. Stuff like the proper modularization of code, for example, how to write a package, how to set up an environment – these types of things – how to write a requirements text file. Very basic things. I think that will be good.

# How Johanna got into research software engineering

28:12

Alexey

**How did you learn these things? How did it happen to you?**

28:18

Johanna

Like I said, I was already interested in computer science very early. But I think what really brought me to this open science/open source field was Brainhack. It was actually during the pandemic. I don't know whether you know, but in Australia, we had a massive lockdown. For eight months you were not allowed to go more than five kilometers from your house, and like two hours outside per day. So it was very isolated. I don't have family here. My field hosts this hackathon, it's actually an organization called Brainhack. It's about hackathons for neuroscience, basically. This is a complete grassroots organization and the way it works is like a typical hackathon. You join and either you pitch a project or you can join a project.

I joined a project – I don't even remember what it was about – like implementing a canonical correlation in Python or something like that. [chuckles] I really liked the vibe there and I really liked the people. The next one was right around the corner, so I decided to help organize it. That really got me into open science and open source because I got tasked with things like how to create the website for the new event. You go to GitHub and you tweak the current website – you make a copy of the current website and tweak it for the next website – so you learn a bit about web development. Then you see what else is on GitHub and you talk to people who have all these skills and they teach you.

Then I also applied as a secretary of a global open science group, which was a role I had for a year. Basically, they were looking for people to join them at the Brainhack. So all this basically started like that. Then I just never stopped. [chuckles] I just joined community after community. And I like it a lot.

30:34

Alexey

**So for you it was more learning by doing rather than just attending a course that teaches you everything that you need to know. Right?**

30:44

Johanna

Yeah, also the other thing is – I think I've mentioned already – it was a bit out of necessity. I'm still doing my computer science degree at the University Hagen Like – that's the one that sends you stuff. Here in Australia, you basically pay for every course. I got a scholarship for my PhD and also a living allowance, but if you want to sit another course, you pay for it.

Usually it’s quite a sum, similar to American tuition fees. So it's impossible to just sit another course here. I would have done it, of course. So you just have to get by with what you find on the internet, which is actually a lot. You can totally have a full computer science degree (or an equivalent [chuckles]) just doing stuff on the internet.

31:36

Alexey

**You need to have the discipline. In university, if you don't do the courses, you don't get credits and if you don't get credits, you don't graduate. [Johanna agrees] But if you're just studying by yourself, then you have to have the discipline and you don't get the credits. That's a different kind of motivation, I guess.**

31:57

Johanna

It is, yeah. The other thing that I also started is freelancing. That also puts a bit of pressure. [laughs] If you get paid for doing stuff, you basically better deliver at some point. [laughs]

# Effective ways of learning software engineering skills

32:15

Alexey

**Do you think what you did is the most effective way of picking up these skills? Just starting freelancing and taking part in hackathons and learning by doing? Or are there maybe better ways researchers can learn these things?**

32:33

Johanna

Yes, good question. I guess there are better ways. If you want to learn something and if you are in a country where it's easy just to sit another course, I would totally do that. As you said, it's way more structured, you have peers that do the same. I would totally do that. Apart from that, I don't think that it's the worst way. It’s definitely a lot of fun doing it this way, I would say.

33:04

Alexey

**I noticed that we have a question. The question is, “Does anyone revise your code? Do you work alone or is there somebody in your team with whom you can discuss different things?”**

33:20

Johanna

That's a very good question. In academia, sadly, when you're not in a computer science or a technical field lab, usually, you work alone. I have a supervisor (not the one that I have here but another one) who is really technically skilled. He has developed this idea about the type of model that I use on the brain, so he could help me. But he's in the Netherlands. Usually, you work alone, which makes it even more necessary that you put your code on GitHub [chuckles] so that someone can actually have a look at it.

34:05

Alexey

**You said that you're a part of many communities. I imagine that there are communities where people also care about this topic. If you work alone, then you can put some code to GitHub, and then perhaps ask other researchers from these communities to take a look. Does this happen?**

34:24

Johanna

Yes, it does happen. It happens especially when you publish your code with your paper. I get requests from people that read my paper and they say, “Look. Your code – I don't understand this part. Can you explain?” And then you say “Oh, yeah. I can see why you don't understand. Maybe it wasn't explained that well in my report.”

34:49

Alexey

**But this adds a lot of work on top of what you do. In a typical scenario, you publish a paper, you go to a conference, you talk about this paper, or maybe you have a poster session. This is two days. Then maybe you get a few emails, you answer these emails and then you focus on the next paper. But here, you get comments, “Okay, I don't understand your code.” It just adds a lot more stuff to your plate, right?**

35:23

Johanna

Yeah, it does. It's definitely often driven by a lot of idealism, trying to like science better, I would say. The entire field of academia is probably populated by people who don't do things just for the money and effectiveness, because then they would move to industry where they would get paid way more for what they're doing. I think that's just the case.

35:55

Alexey

**Okay. So you just know that it will take more effort and you know that this is for the greater good, so you're willing to put more time into this.**

36:05

Johanna

Yeah. And I do enjoy it. I like it. I like working on these things. I like collaborating with people. Often, it's also quite good. They give you new ideas or new insights, or they think like, “Oh, I saw this in your code and I thought it would be nice to extend it by this. What do you think about this? Can we collaborate?” It can also help you in many ways. There's definitely studies that show that if you publish your code – there's this really cool paid webpage called Papers With Code, which is a collection of papers where the code is shared – it shows that these kinds of papers get more citations, they get more recognition. So it definitely helps you. Also, of course, it helps you if you want to go into a more software engineering direction. Further on it, of course, also helps you if you have repositories that you can share and showcase.

# Providing data and analysis for your project

37:01

Alexey

**Yeah, I see that there is a comment that says “I asked a researcher about his code and he never replied. The code was buggy.” Do you think that this kind of situation happens often? [Johanna confirms] Well, at least there's code, which is already a good thing.**

37:19

Johanna

Yeah, it gets even funnier. There was this thread on GitHub where, just for experimental purposes, this one researcher… Usually people add to the manuscripts, “Data can be obtained upon request.” They asked 250 researchers who had put this statement under their paper for the data and basically showcased the replies to this kind of request. A couple of them provided the data and for others, it was like, “Yeah, the data left our lab with the last PhD student,” or like, “We couldn't find the data anymore, so I just didn’t reply.” [chuckles] Really, it's a bit ridiculous. [laughs] But that's the current state.

38:07

Alexey

**I guess there are conferences where you *have* to provide code and data in order to publish there. [Johanna confirms] Conferences and journals, right?**

38:17

Johanna

Yes, if you want to apply for grants, that's usually a requirement that you adhere to open science principles. Also, many, many journals actually ask for both the code and the data. Like I said, it's coming because people are recognizing the issues in the field. It's not a good state. It's what contributed to this open science and reproducibility crisis.

# Johanna’s initial experience with software engineering in a project

38:50

Alexey

**Maybe you can tell us a few examples of projects that you worked on and how adding these engineering practices helped? Because from what I understood, you didn't start as a research software engineer, you started as a normal PhD student, but you had this interest in open science/open source computer science.**

**Maybe you can tell us about one of the projects that you did and then how you realize that you need to start adding these things and how it looked like. What did you add first, what did you add second, and so on? And how did it help you in the end?**

39:27

Johanna

Yeah. I think a good example is my current and my oldest PhD project, which is about the normative model of the brain, basically. I'm working on a very big depression dataset. It's massive, at least for clinical standards. I went in and my first issue was, “How should I organize the folder structure?” I found Cookie Cutter, which is a repository that has examples for the structures – for different fields. You can go there and it creates one. So that was very helpful. You do that and create an environment for your code. That helps. Then, of course, Git. If you push stuff on Git, there's also a principle that I learned. You have at least two branches – one is a main branch and the other one is a dev branch – and you never push new stuff to the main branch, but you push it to the dev branch. It's just very simple things.

What else did I do? Then code formatting standards. Of course, there are tools that do that. In R, you can just click “format my code,” but there's Flake, Flake8, and I think Black is the one in Python that you can use. What else? Version control, for example, for your models MLflow. And then of course, just reading a lot of good code can also have an effect. What else did I do? I also have to say, at the beginning. I probably deleted all my experiments and redid them from scratch like three times. [chuckles] Because it was just so messy. But that's just how it is. Now I'm kind of at a stage where I start a project and I know what to do. I know how to set it up. I know how to keep things clean.

41:52

Alexey

**So you didn't do this project knowing that you needed to add all these best practices, right? For you, you realized it while working on the project – I guess when you deleted your experiments for the first time, you thought “Okay, how do I make sure that this doesn't happen?”**

42:11

Johanna

Yeah. It was definitely a learning process. But like I said, it's kind of just academia – where you figure things out. [laughs]

# Working with sensitive data and the nuances of publishing it

42:22

Alexey

**I remember for my Master’s thesis – I don't know how representative this is – but usually, the way to organize exploratory data analysis for many ad hoc tasks is quite messy. I have a ton of notebooks there, they have very cryptic names. Then I know that I need to push them to Git but… sometimes there are tokens, some sensitive information that I don't want to push – I need to clean them. They stay without being pushed to Git and then something happens, like the file gets corrupted for whatever reasons, and then the notebook is lost. Yeah. Terrible. [chuckles] So I guess you have to solve this and this is how you discovered all these best practices.**

43:11

Johanna

Yeah. Also, often in academia, at least in my field, you have data that is sensitive, so you just can't put any data to any repository. But you can put metadata or parameters from your models that you fitted – they are quite nonspecific, like a mean or whatever.

43:41

Alexey

**Well, for your project, you mentioned that the data is about depression, which can be quite sensitive, right? Can you publish this data? Or you cannot?**

43:50

Johanna

No. The data is from a consortium that you can get access to if you write a proposal. But no, I wouldn't be able to publish the data.

44:04

Alexey

**And if you need to submit a paper based on this data somewhere – to an open science conference – how do you do this?**

44:12

Johanna

Yeah. You can present figures. Like I said, you can present the mean of whatever. My research is about the thickness of the cortex, so that's what I focus on, whether the thicker thickness of the cortex is smaller in people with depression or not. So you can provide a mean. You can also provide de-identified data like it's just *some* depressed brain. [chuckles]

Of course, there’s no personal data, but even I don't have that data. I don't have this information. But still, you wouldn't publish the whole… I mean, there are some datasets that are definitely about this. But for my data, it's part of this consortium, and you need to write a proposal. So not everybody can have this data.

45:07

Alexey

**Interesting. When it comes to healthcare data, it's always tricky. You always need to have to take extra care when you deal with this data.**

# How often Johanna does hackathons, open source, and freelancing

45:24

Alexey

**From what you said, it looks like the easiest way to get skills is to take part in hackathons, contribute to open source, and also freelance. Right? [Johanna confirms] How often do you actually do these things?**

45:42

Johanna

At the moment less often than I would like to. I already work full-time as a research fellow. I'm trying to finish my PhD in the hours before and after that [chuckles] during the day. I just recently contributed to a code sprint of this Turing repository. There's a very nice community. And that's the good thing about living in Australia – all the stuff that happens during the day in Europe happens during the evening here, so you can do stuff after work. I think at least once or twice per month, usually. There are just a lot of things that I contribute to. I'm almost daily on Git just looking for new stuff. I'm in a lot of communities where they also always bring up new stuff or things that you can contribute to. They look for people to give a talk or things like that.

46:54

Alexey

**So you made it a part of your daily routine to check GitHub?**

46:59

Johanna

Yeah, I check trending repositories and I love the “awesome” lists. I love them. [chuckles].

47:10

Alexey

**I think when I asked you about how you found out about the DataTalks.Club community, you said that you found a trending GitHub repo that was from our course. Is that right?**

47:22

Johanna

Yeah. It was from the course. I think it was the ML Zoomcamp.

47:27

Alexey

**Probably the MLOps Zoomcamp, right? The one with MLflow.**

47:30

Johanna

Yeah. That was the one.

47:33

Alexey

**Cool. So there are actually people who check trending.**

47:37

Johanna

Yes. Me. [laughs] I love GitHub.

# Social media as a source of repos and Johanna’s favorite communities

47:42

Alexey

**Instead of checking Twitter or LinkedIn (these are the two social networks I check) you come from work or maybe at the beginning of your workday, you go to Git and you check the trending repos?**

47:57

Johanna

No, I have the GitHub app. I have it on my phone. I also actually check Twitter, LinkedIn, Mastodon, Slack and GitHub.

48:13

Alexey

**What kind of communities are you part of?**

48:17

Johanna

One of them is the Carpentries that I already told you about – this software teaching program. Then the research software engineering community – we have an Australian chapter, but there's also a global chapter and there's also a German chapter and a UK chapter. My open science community. DataTalks.club, of course. [chuckles] Then there's another podcast that I think is called MLOps Podcast. I'm not very active in that one. Kaggle – that's also one. They also have, I think, a very unofficial Slack channel.

48:57

Alexey

**KaggleNoobs or something like that, right?**

48:59

Johanna

Yes. What else? Just a couple of more science-y Slack channels. For example, I work on physiological noise collection for fMRI. We have a Slack channel all there. Then there’s my teaching course. And then the Alan Turing Institute. That's also a really good one.

49:28

Alexey

**That's quite a lot of different things.**

49:32

Johanna

Yeah. [chuckles] And then the other one was where I worked for NASA. I developed a curriculum on open science for NASA and we have a Slack channel where we continue this work.

# Contributing to Git repos

49:46

Alexey

**So let's say that in one of these communities or in GitHub trending, you come across a Git repo that is interesting for you and you want to contribute. What happens next?**

50:00

Johanna

Usually I look at the readme first of all, then the contributing. Then it depends on how big the repo is. It can be a repo that’s structured in releases, then it's usually quite well organized, but also quite big. For smaller repos, I always look at the dev branch, because that's usually where the more interesting stuff happens. Then I would probably look at the documentation and they have. One of the repos that I'm preparing to contribute to at the moment is SciKit Learn and they have a really nice intro for new developers.

Basically, they guide you. They also show you what coding principles you have to adhere to, how to format your code, what tests to run, and stuff like that. So I would look at that. Then of course, you look at the issues, and you look at stuff that needs to be done. Then download the repo and look whether you can give this a fix. It's also always good to actually talk to people. If you see someone that has opened an issue, you can just talk to them saying like, “So what did you think about this? How should we approach this? I would be really interested. Are you already working on this?” Git is a very collaborative tool. It's made to get in touch with people.

51:35

Alexey

**It seems that this recommendation from SciKit Learn is helpful in general, not just for contributing to SciKit Learn. Right?**

51:42

Johanna

Yeah, it's very helpful. But also, they have a really nice section that guides new developers on how they want contributions to look like.

51:57

Alexey

**Do you have your own open source projects?**

51:59

Johanna

I have my paper that I published. And I'm about to launch a blog, actually, where I’m going to write about open science things. And of course, it's a GitHub repository, probably.

52:18

Alexey

**That’s the easiest way to start a blog these days, right? Because it's free.**

# Publishing in the open in academia vs industry

52:22

Johanna

Exactly. [chuckles] Yeah. Apart from that, like I said, at the moment a lot of my time is actually spent on my work. That's the good thing about academia is that you work on stuff that is very easily converted into an open science project. It's meant to be this way. It's a good thing if you do this – if you add open science on top of it. If you do an analysis for a paper and if you then go to your professor or whatever, and say, “I would really like to put this code on GitHub. We would make sure that all sensitive information is removed. Would you mind that?” They probably wouldn't say no.

If you just say, “I’ll take care of this.” They wouldn't say no. I think it's quite easy. I think academia is also really nice because it provides you with a very structured approach to projects. I’ve heard that for many people in industry, it's quite hard to have a portfolio of projects that you can showcase. Academia works this way. You start with some data, you do something about it, and you publish it.

53:33

Alexey

**I guess in academia, the reason people are in academia is to do research and then share it with the world. There is no reason a professor would say, “No, you shouldn't publish the code.” Because that's the reason you are in academia – to share what you do with the rest of the world. While in industry, there could be things like, “Okay, it's actually giving us a competitive edge. We don't want to share it because our competitors will take our code and use it and get more money than us.” In academia, this doesn't exist. Does it?**

54:11

Johanna

Yeah – I just wanted to say… There are definitely arguments when people say, “If I put this on GitHub, I get scooped.” I haven't experienced that from people in academia. What I have experienced is, when I worked at a startup, indeed, I was asked whether I could just use some code from my one of my supervisors and I said no. [chuckles] I mean, it's a no-brainer because this code is A) not mine and B) under license that wouldn't allow this. It's an absolute no-brainer, but then this person just said, “Yeah, this happens all the time. We just rewrite it in a different language.” And I said no. Yeah, it shouldn’t happen, but it's a danger. But I think it's not the rule. Usually people would rather approach you to collaborate, I think.

# Johanna’s book and resource recommendations

55:12

Alexey

**I see. It seems like it's a new topic, but there are already courses for open science and research software engineering. Are there already books about this topic that you can recommend?**

55:30

Johanna

Books? Not yet. I'm actually thinking about writing one. [chuckles] Definitely, the Alan Turing book is about open science. And what else? Yeah, definitely a couple of papers about it. I think that's mostly it.

55:58

Alexey

**Is there something you would recommend anyways? Maybe not related to this particular topic, but in general? Maybe you have some good book recommendations or other resource recommendations?**

56:10

Johanna

Yeah. It's very funny because, of course, I knew this question would come as I was the one who suggested this question. [laughs] What I also do is… Oh! I forgot to mention one social media – Discord. I made a Discord channel from Pact (the publisher) and they sometimes give books that are about to be published for people to read. I'm part of this channel.

One of them is the Machine Learning Solutions Architect Handbook. I was one of the people who read this book. It's really good. It's pretty comprehensive – I think like 800 pages. But it also has hands-on exercises, which I really like, after each chapter. I would recommend that, I think. I really like that book.

57:03

Alexey

**800 pages! That’s like this. [gestures with hands]**

57:08

Johanna

Yeah. It's fairly comprehensive. It's pretty big. I got the ebook, of course.

57:15

Alexey

**So what does the table of contents look like there? What are the topics they cover?**

57:21

Johanna

Everything about machine learning, but then also how to set it up on AWS. Yeah, it's a pretty good book. I had one month – I actually didn't read all of it. I still need to go through the rest because it was just so much. [chuckles]

57:36

Alexey

**Yeah. 800 pages.**

57:39

Johanna

Yeah. I recommended it for publication.

57:42

Alexey

**Business use cases for machine learning, science, tools and infrastructure, platform for ML, ML algorithms, data management, open source machine learning libraries, Kubernetes. Yeah.. and I just started the list. There's so much stuff.**

58:00

Johanna

Yeah, it's really big.

# Conclusion

58:03

Alexey

**Okay. Did I forget to ask you anything that you wanted to talk about?**

58:10

Johanna

No, I don't think so.

58:12

Alexey

**We covered everything, right?**

58:14

Johanna

We covered everything, yeah. It was really, really cool. The time flew by, really.

58:20

Alexey

**It did, yes. I hope your cat enjoyed this.**

58:24

Johanna

Yeah, I think she went… she always lives in my cupboard. I think she went there to sleep.

58:36

Alexey

**Johanna, thanks for joining us today. That was fun talking to you. It seems like you enjoyed it. That's good. [chuckles]**

58:44

Johanna

Yeah. It was really interesting, yeah.

58:46

Alexey

**You said you will not listen to this when it's published?**

58:51

Johanna

[chuckles] We’ll see.

58:54

Alexey

**There will be a transcript that you can just read. [chuckles]**

58:56

Johanna

Yeah, I can just read and then like, “Oh, did I really say this?” [laughs]

59:02

Alexey

**[chuckles] Yeah. Thanks for joining us today. Thanks, everyone, also for joining us today, for asking… We got one question in the comments, so thanks for being active. I know it's a pretty early time. Have a great weekend, everyone.**

59:17

Johanna

Thank you.

59:18

Alexey

**Bye.**