MANAGING MACHINE LEARNING PROJECTS

How to manage and estimate an uncertain project to help build successful ML solutions

Good practices in managing machine learning projects

Being of R&D nature, ML projects involve some risk: not everything has to go as planned. How to estimate an uncertain project? Are there good management practices to help mitigate risk?

We talked to Jakub Romanowski, CEO & co-founder of Evorain, about managing ML projects.

***

Let's start with the basics: can you briefly describe how ML is used in your company?

ML is the core of our business, we provide services mainly in this area. We create research solutions packed in an implementation-ready product. We solve various problems, sometimes more and sometimes less "typical", such as recommendation systems or image recognition. We also create frameworks for AI from scratch, including research on new algorithmic methods or modification of existing ones to increase effectiveness and efficiency — internally, we call it "surgery in AI". This doesn't mean that we exclude popular frameworks or cloud solutions — on the contrary, we work with them in our daily work. We also have internal projects, including our sub-company that's focused on personal profiling — and there, we implemented a project based on machine learning. We've completed several other projects for our own needs, including the prediction of Bitcoin rates.

When we think about building a machine learning team, are there any competencies that are most important to its members? Is there anything that you should pay particular attention to?

That depends on the level and the issue we're dealing with. If we are dealing with a project where we provide a full service, then people with different competencies are needed: we need someone to validate a dataset, or build such a set, or normalize the data, and such people do not have to be experts. They must certainly have a working knowledge of specific techniques to know how to prepare

Interview with

Jakub Romanowski

CEO & co-founder of Evorain, data scientist, specializing in image processing, recommendation systems, algorithms development, and matching the tech skills with a business-oriented approach

this data, but the key issue that we take into account when recruiting new team members and developing our competencies is multidimensional logical thinking. Programming experience is important, and with such a base, this thinking combined with the willingness to develop towards machine learning make up a mix that's sufficient: such a person can be prepared for this job, and sometimes it is more effective to train someone than to look for a person with a specific skill set on the market. This does not mean that we are completely closed to recruiting new people with a specific set of skills — we try and do that too, then we talk to a potential candidate about specific issues related to AI and ML. In such a case, the conversation begins at a much higher level of algorithmic and scientific nuances, and the exchange of project experiences, which usually gives us a pretty good overview of the situation.

We also pay close attention to skills around working with databases - it's crucial that team members are skilled in this type of work. When it comes to integration with existing systems, we need people who have somewhat mixed skills: they program but they can also join a company, fit in, adapt, and build software around the issues they're working with.

At Evorain, we combine our scientific background with programming skills. As partners, we met while working at the University. Each of us has worked in various IT companies, mainly as programmers. This combination of two worlds, typical research and software development makes it possible to blend these two branches together, taking into account the needs and appropriate communication between these "worlds". Thanks to this, we can build solutions that work as intended and bring benefits, and that's what ultimately matters: that these projects are not put away in the drawer, which can sometimes happen with research projects. That's a skill that allows us to respond to complex customer needs since it's not always the case that a business problem can be solved immediately — then we can make use of our scientific expertise and conduct an R&D project to find this solution.

Are skills that are more business-oriented or on the soft side also important?

When it comes to business matters, it's my role to prepare and present everything properly. We usually work with two types of customers: those who are technically aware - you should talk to them about technical details, it's something they expect, and the other group is not as aware but willing to gain more understanding — the key here will be to explain everything appropriately not to overwhelm the client but also to give them a sense of security in the implementation of the project despite the high level of abstraction of the issue. The latter case is more difficult from our perspective because even the communication part of the project is tailor-made, taking into account what the client wants and needs to know.

> **The key is to explain everything appropriately not to overwhelm the client but also to give them a sense of security in the implementation of the project despite the high level of abstraction of the issue.**

Sometimes it's not enough to say that we will use this and this machine learning method, you also need to know what it will give the client: it will save them this much money, increase efficiency by a given percentage — I have to be able to translate this technology into numbers that matter to the company. And that's where I go back to my scientific experiences because thanks to them, I learned to estimate these tasks over time, translate them into numbers, and as a consequence, I am able to talk about it all in a way that's easily digestible for the end-user.

Listening and being open to the client's needs is also essential, this gives you a good starting point to choose the right words.

When it comes to project management, it is believed that ML projects pose some difficulties. Would you say that managing such projects can be difficult?

There is no clear answer: yes and no. Simpler projects, such as the aforementioned recommendation systems, if we get a specific set of data and know that it is closed, then such a project and its management are not troublesome.

We are able to accurately estimate it, but that's also a consequence of the experiences from previous projects that we've implemented. Thanks to this, we no longer feel the risk associated with research work. We've already conducted a number of similar projects, so we are able to assess how the project will proceed.

Every member of the expert team estimates the same task and I am the nitpicking one, I look for whatever could go wrong: maybe we'll fail here, or maybe this won't go as planned.

This approach definitely works in more complex challenges. Naturally, identifying as many risks as possible and estimating them increases the certainty of the project's success or quite the opposite — it indicates that, unfortunately, the project should be suspended. Of course, these projects are not all implemented according to one template, because they are always different in some ways, but with some experience, it is much easier for us to

assess how much time it will take and what risks may arise.

Typical research work, where it'd be best to have an agreement like "OK, we can build your own rocket-science thing, but we have to develop it for an infinite amount of time" is problematic and we do need some time frame. Despite the above, in the last 5 years, we have never had the actual time of the research stage in the project exceed the estimate by more than 20%. What works for us and provides the ability to determine the time of work is the selection of 3 dark horses. **We select 3 research and technological solutions and develop them successively to the stage at which we can tell the client that the solution is sufficient and meets the assumptions.** It often happens that the first of the three typical solutions is relevant and in such cases, we already have the approximate effectiveness of the algorithms determined.

ML projects are always fraught with some risk but over time, when we grow more and more experienced, the risk is not that high anymore technologically speaking. There is also that other side, namely communication and the situation around it when the project is a cross-section of many fields and connects people who are specialists in their field. For example, when we join a project from a completely different field of science, e.g. the humanities, we always start with workshops since this allows us to reduce the risk, assess the current situation, decide what issue to develop or not. Naturally, the people we work with in the project do not have to understand all issues related to the software, how neural networks learn, etc. Then, there's an additional risk that the stage during which we explain things will take longer but then we just talk a lot and check each other's knowledge to be sure that we're all on the same page.

In a group of people with a non-technical skill set, training is another important factor: it's not enough to collect the requirements and go do our own thing, we also show them how it works. If we don't devote this time to education and communication, and let's say 4 out of 10 people from the team did not understand a given issue or we didn't explain it right, then, as a consequence, data sets change during the project, or new data comes in - something that seemed irrelevant to one party but with time, it becomes clear that it actually plays a significant role.

Training is also an important factor: it's not enough to collect the requirements and go do our own thing, we need to show how it works.

I don't want to present it as if we're always right and mistakes happen somewhere else, that's not true. It's great if there's commitment and the willingness to understand in our coworkers, but **it's our responsibility to present and describe everything that has an impact on the project's success.** When we're not on the same page from the beginning, it gets difficult because the requirements may change, new ideas come in, and this makes the process longer and more difficult. There's no point in going through that when we can start off well and save ourselves much trouble. However, the thing that's so awesome about people — diversity — can play tricks on us.

So in situations like that, you have to step into the client's shoes, get on board with their business, and advise them on what they can win with a given technology?

Definitely. We most certainly want to, even must, get to the level of specialists in a given business. I really enjoy this part of our work but it's also challenging at times. For example, we need to learn how cells divide in biopsy samples or understand individual markets and brands in e-commerce. We need to get to know these areas inside out to build a solution that can improve a given company. We could, of course, just collect general requirements, but then we come back to the problems of misunderstanding and constant changes in the scope of work. It's important for us to have a deeper understanding of the domain, which takes some time, but apart from allowing for more efficient cooperation it also gives us a lot of joy and ultimately brings the best results.

Are there any fundamental differences in how ML projects are managed compared to "standard" software development projects?

For sure there are some, but having been in this business for a longer while allows us to bring it to similar norms. When team members know each other well like we do, it's quite easy to decide who does what and we sometimes get each other without words — so the process naturally becomes smooth.

One major difference could be the research issues mentioned earlier: we usually estimate these with a 20% error margin, and typical software projects can often be estimated extremely accurately, although it is always worth having some margin of error. I am glad that we've never had a situation when we weren't able to find a solution during the research stage — so there aren't any big differences in managing these projects.

The difference, however, was more obvious at the beginning of our "adventure". For example, we proposed something that we weren't certain about (with the assumption that when we promise something, we'll do it), and then "crisis management" comes into play. From my point of view, there are no significant differences now, of course, except for implementation issues because ML projects also have to ensure data consistency, data has to be verified, but these are early stages of the project.

The difference may also be in the level of customer awareness as it's easier to talk about software in general than specialized areas such as artificial intelligence.

There's also room for challenges when we join a company's team, create some part of the component, and work with the staff there. Sometimes communication is difficult and doesn't happen instantly — people have many different issues on their plate and sharing data can be an issue since it has to follow a process involving a number of people. As a consequence, reporting our needs related to the project also takes a long time and may be difficult, but of course, everything has to be worked out.

It all translates directly into project management and it may sound a bit like a list of complaints, but in fact it's just what we've experienced over the years.

Before the interview, I asked some clients what they would point out — and summing it all up, I see some conclusions: having more conversations, digging deeper into some areas, more frequent and more granular adjustments. Well, we are still learning and I'm grateful for these comments because they are invaluable — that's a good wrap-up on the project management and matters that directly influence the process.

“

Sometimes it's not enough to say that we will use this and this machine learning method, you also need to know what it will give the client: it will save them this much money, increase efficiency by a given percentage - I have to be able to translate this technology into numbers that matter to the company. And that's where I go back to my scientific experiences because thanks to them, I learned to estimate these tasks over time, translate them into numbers.

You mentioned that project management is easier now, but it wasn't easy at the beginning. Having all these experiences that allowed you to develop certain processes, do you still see difficulty in the fact that research projects may simply not work out? Clients can be concerned about that and when we combine it with findings that around 80% of AI projects do not bring the promised benefits, it is hard not to feel the risk.

There are situations when clients are not aware of the risk, or they are aware and have concerns about it, but what really helps us here is our approach: we select 3 solutions and within 2-3 months, we can check whether the uncertain thing can live up to expectations in the development stage. This is a certain risk on the client's side because they invest their time in this project, however, if there's uncertainty, we'd rather verify it. If someone says that they can solve any problem, that's a stretch, and I'd estimate such work as that 80%. You have to be aware that it can be different, and on our part, if we don't even see a glimpse of hope, we don't recommend going into the development stage.

Several years in this field and observations of what is happening with AI projects make us aware of how the recipients look at it and I am not surprised. I really hope that we have a share in the successful 20%, mainly thanks to our assumption that projects are not meant to end up hiding in drawers. Acting with respect towards those who invest their resources into research endeavors, we're fully transparent, and when we don't see any chance of success, we don't start the project.

What about estimating the time of work in ML projects?

I think that it's our big advantage that our estimates are reliable and accurate. We have academic and commercial experience, and it's a battle-tested approach with our 3 main development paths, so our estimates are usually on point. Although sometimes we also face challenges, e.g. in a project related to explainable AI — here, the work took longer, but in the project, we agreed that we have a certain time frame when we can observe what's happening. In this project, we created new algorithms that were based not only on neural networks but also on neural-fuzzy logic. In this case, we couldn't rely on the frameworks available on the market. The project was successful but it was a strictly research project and we had that given time to see what would come out of it.

Challenges related to more commercial, popular ML solutions are easier to estimate but we're always transparent about keeping it within some range, and we land somewhere in the middle.

As for the approach I've mentioned — the 3 research methods — it's worth elaborating on. We believe that the 3 assumptions of solving the problem are the maximum on our side in terms of exposing someone to a risk. If we are not able to solve the issue in those 3 attempts, that's the red line that indicates the next steps.

Do you think a project manager is needed in ML projects? Should it be a technical or a non-technical person?

I think a project manager will definitely bring value to every project. It's great if such a person has some programming background and basic knowledge of machine learning, so they don't get anything wrong. **In-house training certainly addresses this need.** As long as the number of projects is not overwhelming, it is OK when the team deals with the issues of project management on their own, and we're in touch anyway to discuss issues and catch up. The assumption is that each of our teams must have a leader-expert, so this person bears some decision-making responsibility. With time, however, there may be too many of such duties, and then, a PM is necessary. I would definitely prefer to educate a project manager in ML than to involve a person who creates solutions in full-time project management. If there is a person in the ML team who delivers, they should stay there, as long as it is consistent with their professional goals.

How would you help a project manager with no prior ML experience manage this process?

In our company, we organize **regular internal meetings to exchange knowledge.** We talk about our achievements, solutions, problems — and that allows us to keep learning. A person undergoing training listens, gains knowledge and experience by discussing case studies from our work, and this seems to be the best solution. We could, possibly, have a person who would learn "live" right away in the project, but I think such a case would require us to have the client agree to that. Anyway, education is the key, a certain level of technical knowledge will simply be needed for this person, even to be able to tell whether a project can be estimated in the first place. Such a project manager must be very aware and well-trained.

Do you think it is possible to work out good practices that can be replicated in ML projects?

Yes, but I wouldn't have said that 2 years ago. It is thanks to our experience that we were able to develop a framework for it. At the heart of the ML project is data, so it is important to have a process of preparing, processing, normalizing data, and that's a starting point. The client has to be aware that this data must be there because without it, nothing can be done. The client should also know that it's best to provide us with specialists from their company who will be able to consult us in their domain. Later, we implement the plan that I mentioned earlier, i.e. we choose promising solutions, build models incrementally, test them as often as possible, validate them with specialists from the client's team, check the effectiveness of the models and try to maintain it at the highest possible level. That's our template, a simplified recipe to deliver the project and finish it well.

SOFTWAREMILL

evorain