**How to find the right Machine Learning team**

**Questions you should ask and red flags you should avoid**



Photo by [Adi Goldstein](https://unsplash.com/@adigold1?utm_source=medium&utm_medium=referral) on [Unsplash](https://unsplash.com?utm_source=medium&utm_medium=referral)

As Machine Learning professional, navigating the diverse landscape of ML roles within the industry can be confusing. Job titles are usually not a big help because they change depending on the company and also depending on the organization within a company. Job titles tend to change over time as…

In order to navigate the job market and find potential roles for yourself, you therefore need to have a list of probing question. Here are 3:

* what kind of ML team are they?
* what’s the skill gap?
* what do they own and who are their customers?

Let’s dive a little bit deeper into each of these 3 probing questions, and why they should always be on your mind when surveying the ML job market.

**What kind of ML team are they?**

In a matured tech company (not necessarily in startups though), you’ll typically find 3 types of ML teams, infra, applied, and research:

**Infra ML teams** build services for model development that they serve to other teams via APIs or UIs. One team may own a modeling service, another team a feature engineering service, another team an inference service, and so on. ML infra teams work on problems such as:

* scalability: how can we scale our services to the entire suite of ML models owned by the company?
* efficiency: how can we reduce the computational cost of training and serving our models?
* integration: how can we integrate the model predictions with the product? How do we handle error cases where the model inference call fails?
* automation and abstraction: how can we automate and abstract away most of the heavy lifting around model development? How can we build user-friendly self-service tools such that even non-technical business partners can build and deploy their own models?

**Applied ML teams** design, develop, test, deploy, and iteratively improve ML models that solve concrete business problems, using tools owned by ML infra teams where suitable. These teams work on problems such as:

* framing: how can we frame a business problem as an ML problem?
* data and feature discovery: what data do we need to solve this problem? How do we get the right labels, and make sure they are reliable? What features do we need, and what’s the coverage of these features?
* experimentation and [A/B testing](https://towardsdatascience.com/the-joy-of-a-b-testing-theory-practice-and-pitfalls-de58acbdb04a): which model works best for our use-case?
* continuous improvement: how often should we re-train the model? How can we improve the next model version with more features, more labeled data, better sampling, or a better model architecture?

**ML research teams** invent new algorithms or model architectures, with the main goal of publishing their findings in academic journals and conferences. They’re the birthplace of innovations such as the ADAM optimizer, the attention mechanism, or particular model architectures such as AlexNet or BERT. Most of ML research never finds its way into production, but when it does, it can create a paradigm shift with massive performance gains. Research teams work on problems such as:

* how can we beat the latest state-of-the-art on a public benchmark dataset?
* what are the empirical scaling laws that describe the behavior of large neural networks?
* [why](https://towardsdatascience.com/why-does-deep-learning-work-so-well-6550f3aa22c6) does deep learning work in the first place, and what are its limitations?
* how exactly does [fine-tuning](https://towardsdatascience.com/what-exactly-happens-when-we-fine-tune-bert-f5dc32885d76) of a large language model work?

Choose a team that aligns best with the kind of ML work that you want to do. Based on my own observations, infra ML teams tend to attract people with a software engineering background, while applied and research ML teams tend to attract people with an academic background, oftentimes PhDs. This may be because applied and research ML roles are more heavily driven by experimentation, which is something that may be more natural to a PhD scientist than to an engineer.

Lastly, avoid [pin factories](https://towardsdatascience.com/data-science-anti-patterns-you-should-know-1d8711eef72d), where model developers build model artifacts and hand these over to engineers for deployment. This is frustrating for everyone because it introduces communication overhead, slows down iteration cycles, and creates unclear ownership and finger-pointing when things break in production.

**What’s the skill gap?**

“I absolutely know it is hard, but we’ll learn how to do it.” — Jeff Bezos

There’s never going to be a role for which you have all the required skills. Instead, there will always be a gap between your personal set of skills and those that define the role.

This introduces a trade-off: if the skill gap is large, you’ll learn more, but it will take you longer to ramp up. If the skill gap is small, you learn less, but you’ll ramp up faster and make contributions sooner.

Choose a role with a skill gap that’s small enough that you feel confident to be able to make contributions within a reasonable amount of time, but avoid roles in which there’s really nothing new for you to learn. For example, if a new applied ML team uses the same modeling technologies and the same tech stack that you’re currently working with, only on a different problem domain, there’s nothing technically new for your learn, and it may not be the best career move.

When thinking about skill gaps in potential new roles, it’s also useful to adopt what psychologists call a ‘[growth mindset](https://online.hbs.edu/blog/post/growth-mindset-vs-fixed-mindset" \t "_blank)’ as compared to a ‘fixed mindset’: trust that you’ll learn the skills you need on the fly.

**What do they own and who are their customers?**

Every team should be owning something, and they should have customers. In particular,

* infra ML teams own services, and their customers are either other infra ML teams or applied ML teams,
* applied ML teams own models, and their customers are the users of the company’s apps and websites,
* research ML teams own research domains, and their customers are mostly other research teams, and in rare cases, if an innovation is practical enough to productize, infra and applied ML teams.

Be careful about teams that either don’t own anything directly or own something but have no customers. For example, if an applied ML team does not own models directly, but instead makes proposals to models that are being owned by other teams, the impact in that team will always be limited by the goodwill of others. Or, if an infra ML team owns a service that has very few internal customers, sooner or later the question may be asked whether that service, and hence the team, are really still needed.

‘What do they own and who are their customers’ is therefore an extremely useful red-flag-detector, and you should always have it on your mind when surveying new ML teams.

**Final thoughts**

When looking for your next (or first) ML team, always keep these 3 questions in mind:

* what kind of ML team are they? Infra, applied, or research?
* what’s the skill gap, and are you comfortable with that? Are there new things for you to learn?
* what do they own and who are their customers? If they don’t own anything, how do they create impact? If they don’t have customers, why are they needed?

Lastly, just as roles and titles change over time, so will your own preferences. Personally, I started off my journey in ML research, but soon switched to applied ML because I liked the prospects of creating real-world impact much more than those of publishing papers that may soon be forgotten. I also know of peers that moved from applied ML to infra ML because they didn’t like the unpredictable nature of running ML experiments, or because they wanted to learn new skills.

Always remember, this is your career, and if you don’t look out for your own best interests no one else will. Make the choices that get you where you want to be.