Seven Steps to Success
Machine Learning in Practice

Daoud Clarke

Project failures in IT are all too common. The risks are higher if you are adopting a new technology that is unfamiliar to your organisation. Machine learning has been around for a long time in academia, but awareness and development of the technology has only recently reached a point at which its benefits are becoming attractive to business. There is huge potential to reduce costs and find new revenue by applying this technology correctly, but there are also pitfalls.

This guide will help you apply machine learning effectively to solve practical problems within your organisation. I'll talk about issues that I've encountered applying machine learning in industry. My experience is in applying machine learning to analysis of text, however I believe the lessons I have learnt are generally applicable. I have been able to deliver significant and measurable benefits through applying machine learning, and I hope that I can enable you to do the same.

I will assume that you know the basics of machine learning, and that you have a real-world problem that you want to apply it to. This is not an introduction to machine learning (there are already plenty of those), however I don't assume that you're a machine learning expert. A lot of the advice is non-technical and would be just as useful to a product manager wanting to understand the technology as a software developer creating a solution.

1 Clearly understand the business need

Understanding the business need is important for any project, but it is easy to get blinded by technological possibilities. Is machine learning really going to benefit the company, or is it possible to achieve the same goals (or most of them) with some simple rules? The goal is to build a solution, not to do machine learning for the sake of it.

Try and identify all the metrics that are important to the business. The metrics we are optimising for have a profound effect on the solution we choose, so it is important to identify these early on. It also affects what alternatives there are to machine learning.

In the case of **classification** problems, potential metrics to consider are

- **accuracy**: the proportion of all instances classified correctly. Note that this can be very misleading if the data is biased (if 90% of the data is from class 1, we can get 90% accuracy by simply classifying everying as being from that class). Real data is normally biased in some way. For this reason, you may want to consider an average of the accuracy on each class, or some other measure.

- high **precision** is needed when the results need to look good, for example if they are being presented to customers without any manual filtering after the machine learning phase.

- high **recall** is important when combining machine learning with manual analysis to produce a combined system with high overall accuracy.

- **F_1 score**, or more generally F_β score is useful when a trade-off between precision and recall is needed, and β can be adjusted to prefer one over the other.

Customer Service at Direct Electric

Direct Electric[a] are a large electricity company based in the south of England. Dave, the head of customer service, is concerned about response times for upset customers who contact the company online. He wants to ensure that if a customer sends an angry email, a representative will get back to them quickly.

"At the moment, it takes about two days to respond, and I'd like to get that down to half a day," he explains to Samantha, the resident machine learning expert on the software development team. Dave has heard about automated sentiment analysis, and wonders if that could be used to quickly identify the emails of interest, so that they can be prioritised by the customer service team.

"What we could do," suggests Samantha, "is try and identify the emails that are likely to carry negative sentiment automatically, and send those to your team to look at first."

"That sounds good!"

"The thing is," says Samantha, "A machine-learning based system isn't going to get everything right. Would it matter if we missed some of the negative sentiment emails?" Samantha thinks a high precision system may be what they are looking for. In this case, we will most likely have to sacrifice recall, and miss some of the emails of interest.

"Well, not really," says Dave, "it's only really useful to us if it finds them all."

"Well, if you want to guarantee you find all of them," says Samantha, "the only way to do that is to examine them manually." Dave looks crestfallen. "But," she continues, "we could probably get nearly all of them. Would it matter if we accidentally prioritised some articles that aren't really negative?" She is thinking of trying to build a system with high recall, which will probably mean lower precision.

"That would be fine," says Dave. "After all, at the moment, we're reading them all."

[a] Examples are fictional; no resemblance to real persons or organisations are intended.

2 Know what's possible

A common problem that I've encountered in working with business is that non-technical people will expect machine learning to solve all their problems. It won't. Machine learning will never do the dishes or the laundry, and there are many tasks in business that will still need to be done at least partially manually. The key is to work out how (and if) machine learning can benefit you, and often the solution is to combine manual analysis with technology.

The first step is to know what machine learning can and can't do for your problem. By its nature, machine learning will never give you 100% accuracy on any given task. You can get an idea of the level of accuracy to expect from a machine learning system by looking at some papers on state of the art systems for related problems in academia. It is important to translate these figures into something that is meaningful to the business.

You then have to consider whether that level of accuracy is acceptable to your business. If it's not, it may still be possible to benefit from machine learning by performing manual analysis after an automated process.

Metrics

"Well, I've done some research," says Samantha, "and it seems that state of the art accuracy for sentiment analysis is about 85%. I'm not sure whether it's possible yet, but if we could get that level of recall on our data, would it be acceptable?" She couldn't find much research that mentioned recall, but she's hoping that as all the experiments were done on unbiased data, 85% accuracy will translate to roughly the same level of recall.

"Hmm, recall," says Dave, "I know what that means." Dave has been reading up on machine learning. "You're saying that we're going to miss 15% of the important emails? I'm not sure that would be acceptable."

"Well, what kind of value *would* be acceptable?" asks Samantha.

"Well, I suppose we could get away with 95% recall."

3 Know the data

Knowing the data the business is dealing with is essential to building an effective solution. Some questions to consider are:

- What features of the data are relevant in relation to the end goal? How is this data structured and how accessible is it?

- For **classification** problems, how **biased** is the data towards each class? For example, in our sentiment analysis situation, if only 0.01% of the emails are negative, it is going to be very hard to build a system with high precision based only on machine learning.

- How does the data vary with time? For example in a classification problem, if our system is presented with a new instance that is nothing like anything in the **training set**, it cannot be expected to classify it correctly. In this case we will need to feed new training data to the system continuously by manually analysing a proportion of the data.

- What levels of granularity are there? In our example, there may be different departments that can all receive support requests, which may vary in content considerably. Should we build a classifier per department, or a single classifier for the whole system (or both)?

There are often unexpected benefits to looking at the data. Often, a simple analysis of the data will uncover vast opportunities for improvement in company processes that have been overlooked, and these are benefits that can often be realised without doing any machine learning, or even any software development!

Analysis

"Ok, I looked at that set of support emails you gave me," says Samantha.

"It covers a whole month, so it should give you a pretty good picture of what we're dealing with," says Dave.

"Yes, it's pretty interesting. Did you know that 65% of them are for the online department, and 90% of those are for people who need help logging in?"

"Yup, that sounds about right!" Dave knows only too well how annoying the company's login process is.

"We should do something about that!"

"We asked for the website to be changed to make login simpler, but it seems it's not on the priority list right now," says Dave.

"Typical," says Samantha. "We should try and get that done before this project. Anyway, the rest of the data is interesting too. Most of the angry emails are from people who spent ages waiting in a phone queue."

"No surprise there," says Dave.

"Ok. Well, given we can't fix the underlying problems, I reckon we can identify the unhappy customer emails with 70% precision at 95% recall."

"You mean 30% of the ones your system says are unhappy customers will actually be satisfied ones?"

"Erm yes. That's what you get for requiring such high recall."

"I see. No such thing as a free lunch, eh."

“The good news is that in this sample the topics of interest seem to be fairly static,” says Samantha.

“I know, people always complain about the same things,” says Dave glumly.

4 Plan for change

There are several aspects to consider when thinking about how to integrate machine learning into your business:

- How will the machine learning component fit into the existing business? Will manual work be necessary on top of machine learning? If so, at what point will this happen, and what will the work consist of? Who could potentially do this work?
- What technical changes will be needed? If machine learning is to be integrated into an existing system which is large and complex, then this task can be challenging.
- What changes to business processes will be needed? For example, will the machine learning judgments need to be monitored? Will someone decide whether machine learning should be enabled, or when models should be retrained?
- Are there opportunities to adapt business processes or technology to enable the collection of more data that may enable new benefits by applying machine learning?

Integration

"So, I've been thinking how the machine learning component could integrate with the existing system," says Samantha.

"Ok," says Dave.

"We need to hook into the support software that your team uses. We can get to the emails before they're presented to your team. A simple solution would be to change the subject to indicate that machine learning thinks they are important."

"Then we can just search for that tag — neat."

"Yup. Since the data seems fairly static, I don't think we need to retrain the models that are used, we can just use the fixed dataset that I used. But we'll need to have a process in place to ensure that the tags added by the new component are correct — can you ensure that your team lets us know if there are any problems?"

"Of course."

"Ok. If there are, we may need to train a new model based on more data."

5 Avoid premature optimisation

This maxim applies as much to business optimisation as it does to software engineering. Machine learning is business optimisation. Don't overdo it.

I strongly recommend taking an Agile approach to software development. This will force you to divide your work into small deliverable chunks. This is typically hard with machine learning, especially if it needs to integrate with an existing system, as there are often several components that need to work together. However, if you are able to do it, you will deliver value to the business much more quickly, and potentially avoid doing unnecessary work by over-engineering your system.

In terms of software engineering, take care to build a clean, elegant system, and use common sense when evaluating your options. In particular:

- Bottlenecks will be where you don't expect them. If possible, scale gradually, and iron out bottlenecks as you encounter them. This should be possible if your code is cleanly structured and unit tested.
- Use big O estimates to reason about possible approaches.
- Try and estimate the likely sizes of the datasets to verify that your storage strategy is appropriate.

Planning

Dave and Samantha come up with a set of features that Dave would like implemented. With the development team, they break down these 'epics' into small 'user stories'; features that her team can implement in a few days at most.

"As the head of customer service, I want to respond quickly to upset customers who contact us by email, so that I can make our customers happy."

"Pretty good as an epic," says Samantha. "We can break it down by making it more specific. How about, 'I want to respond quickly to customers who have trouble logging in'? That way, we can just do a simple regular expression check on the emails to deliver this story, and get the integration part of the work done while we are still developing the machine learning component."

"Perfect," says Dave.

"Also, integrating with the support system is looking a bit tricky. As an initial system, would you be happy if we got the system to send you an email with links to the identified cases? Then we can do the integration as a separate piece of work."

"Ok, so the initial system will just send me an email containing customers who have problems logging in?" says Dave. "That's sounds good."

After estimating the size of the stories, the team decides they can commit to delivering the initial system in the first one-week sprint.

"You'll have a working system," explains Samantha, "and we'll build incrementally from there."

6 Mitigate risks

The biggest risk is that you fail to deliver the project — following an Agile approach as mentioned previously is the best way to mitigate this.

The next biggest risk is that you deliver a system that doesn't get used. Ensure there is buy-in for what you are delivering from as high up in the business as possible, and from those that will actually be using it. Make one of these people the product owner, explain to them the business impact of each potential feature so that they can prioritise them. Don't expect them to understand the technical details.

All software has bugs. You want to catch as many of them as you can before release. Ideally, in addition to extensive unit testing (e.g. from test driven development), the system should be tested by users and/or dedicated testers before release.

Planning

Before every sprint planning session, Dave makes sure the stories in the backlog are in priority order. He knows roughly how much work is involved in each one because the team have estimated them using story points.

Dave is able to measure the benefits delivered by the system against the original proposal, and track how these change as new features are delivered. Any problems or bugs are identified by Dave's team, and Samantha's team can fix them quickly because the code is fresh in their minds.

7 Use common sense

If you are a scientist, it is very easy to get blinded by data. Remember that you only have a sample of the data, and future data may not look anything like the past data. If you are building something that augments or replaces a manual task, then understand the mental processes involved in that manual task. Do the task yourself. You will learn about all the idiosyncrasies and exceptions that require special business processes. The automatic system will have to deal with these somehow. There are some things that a human wouldn't do, because it's just common sense. Don't let your system do these things.

Planning

The machine learning component has been released and Dave is concerned.

"I think we've got a problem. We had a lot of angry customers this morning and your system didn't identify them. It's a recurring problem with our renewals system, it really upsets our customers."

When Samantha looks at data she finds that there were no emails for the renewals department in the training set used to build the model that is used in production. It's not surprising that the system missed them.

"The type of emails you get in each department are quite different," explains Dave.

"Ok, we'll check we have training data for each department," says Samantha.

Conclusion

We have only touched on some of the issues you are likely to encounter when implementing a system which uses machine learning. Many of these issues are common to software projects in general; in fact, my final message is that a project that uses machine learning is still primarily a software project. The actual code relating to machine learning in such a project is likely to be as little as 10%. Treat the project as you would any other software project, follow industry best practices, and know what you're talking about, and you'll be fine.

daoud.clarke@gmail.com