**The Art of Machine Learning Experimentation**

5 simple strategies to help you get the most out of your ML experiments

Experimentation is at the heart of the Machine Learning profession. We progress because we experiment.

However, not all experiments are equally meaningful. Some create more business impact than others. Yet, the art of selecting, executing, and iterating on experiments with a focus on impact isn’t usually covered in standard ML curricula.

This creates a lot of confusion. New ML practitioners may get the impression that you’re supposed to simply throw everything at a problem and “see what sticks”. That’s not how it works.

To be clear, I’m not talking about the statistics of offline and online tests and their variants, such as [A/B testing](https://towardsdatascience.com/the-joy-of-a-b-testing-theory-practice-and-pitfalls-de58acbdb04a). I’m talking about what happens before and after the experiment is done. How do we select what to experiment on? What do we do if the outcome is negative? How do we iterate as efficiently as possible?

More broadly, how can you get the most out of your ML experiments? Here are 5 simple strategies that you can adopt.

**1 — Know when to experiment**

As ML practitioner, there will always be a Million questions on your mind. What if we dropped this feature? What if we added that neural network layer? What if we used this other library which claims to be faster? The possibilities for spending your time are endless.

How should you decide what to experiment on, given your limited time budget? Here are some practical tips:

**Prioritize experiments with the highest expected gain.** Take your time to fully understand the existing model and find out where the largest gaps are: that’s where you want to focus your efforts. For example, if a model uses just a handful of features, the best experiments are probably around feature discovery. If a model is a simple logistic regression model, work on model architecture may be more promising.

**Don’t experiment to learn things that are already known.** Before even thinking about launching any experiments, do your research. If there’s a broad consensus in the literature about a question you’re trying to answer, then you probably don’t need to design an experiment around it. Trust the consensus, unless you have strong reasons not to.

**Define clear success criteria prior to the experiment.** If you don’t have clear success criteria, you’ll never know when you’re done. It’s as simple as that. I’ve seen too many models that were never deployed because the launch criteria changed after the experiments were run. Avoid this pitfall by defining and communicating clear criteria prior to running any experiments.

**2 — Always start with a hypothesis**

Scientific experimentation always starts with a hypothesis. We hypothesize first, and then run an experiment that will either confirm it or rule it out. Either way, we have gained knowledge. That’s how science works.

A scientific hypothesis has to be a statement, usually with the word “because” in it. It can’t be a question. “Which model is better?” Is not an hypothesis.

A hypothesis could be:

* “I hypothesize that a BERT model works better for this problem because the context of words matters, not just their frequencies”,
* “I hypothesize that a neural net works better than logistic regression for this problem because the dependencies between the features and the target is non-linear”,
* “I hypothesize that adding this set of features will improve model performance because they’re also used in that other, related use-case”,

and so on.

Too often I’ve seen people run large numbers of experiments and present the results in long spreadsheets, without a clear conclusion. When asked “Why is this number bigger than that number?”, the answer is often some form of ad-hoc guess. This is HARKing, hypothesizing after the results are known.

HARKing is the opposite of science, it’s pseudoscience. It’s dangerous because it can result in [statistical flukes](https://towardsdatascience.com/is-my-model-really-better-560e729f81d2), results that appear to be real but are simply the result of chance alone (and don’t materialize in production).

The scientific method — formulating a hypothesis prior to the experiment — is the best guard against fluke discoveries.

**3 — Create tight feedback loops**

Changing one thing in your ML pipeline should be as simple as changing one line of code and executing a submit script in a terminal. If it’s much more complicated than that, it’s a good idea to first tighten your feedback loop. A tight feedback loop simply means that you can test ideas quickly, without any complicated stunts.

Here are some ideas on how to do that:

* **Automate naming.** Time spent thinking about how to name something (a model, a trial, a dataset, an experiment, etc) is time that’s not spent actually experimenting. Instead of trying to come up with clever and insightful names such as “BERT\_lr0p05\_batchsize64\_morefeatures\_bugfix\_v2”, automate naming with libraries such as [coolname](https://pypi.org/project/coolname/), and instead simply dump the parameters into logfiles.
* **Log generously.** When logging experimental parameters, err on the side of logging more than you need. Logging is cheap, but re-running experiments because you don’t remember which knobs you’ve changed is expensive.
* **Avoid notebooks.** Notebooks are hard to version, hard to share, and mix code with logs, making you scroll up and down each time you want to change something. They do have their use-cases, for example in exploratory data analysis and visualization, but in ML experimentation, scripts are usually better: they can be versioned, shared, and create a clear boundary between code and logs, i.e. inputs and outputs.
* **Start small and fail fast.** It’s a good idea run an experiment first on a small, sub-sampled, dataset. This allows you to get quick feedback without losing too much time, and “fail fast”: if the idea isn’t working, you’ll want to know as soon as possible.
* **Change one thing at a time.** If you change multiple things at the same time, you simply can’t know which of these things caused the change in model performance that you’re seeing. Make your life easier by changing just one thing at a time.

**4 — Avoid “shiny new thing” bias.**

All too often I’ve seen people getting overly excited about the latest ML research paper and trying to force it into their particular use-case. The reality is that the problems we tackle in ML production are oftentimes much different from the problems studied in ML research.

For example, large language models such as BERT drastically moved the needle on academic benchmark datasets such as as GLUE, which contains linguistically tricky problems such as

*“The trophy did not fit into the suitcase because it was too small. What was too small, the trophy or the suitcase?”*

However, a typical business problem may be as simple as detecting all products in an e-commerce catalog that contain batteries, a problem for which a simple bag-of-words approach is perfectly fine, and BERT may be overkill.

The antidote to “shiny new thing” bias is, once again, to rigorously follow the scientific method and formulate clear hypotheses prior to running any experiments.

“It’s a new model” is not a hypothesis.

**5 — Avoid experiment purgatory**

The outcome of an experiment can be either positive (we confirm the hypothesis) or negative (we reject it), and either outcome is equally valuable. Positive outcomes improve our production models and hence our business metrics, while negative outcomes narrow down our search space.

Too many times I’ve seen peers stuck in “experiment purgatory”: the experimental outcome was negative (the idea didn’t work), yet instead of wrapping up and moving on, they kept trying different modifications of the original idea, perhaps because of organizational pressure, perhaps because of “sunk cost” bias, or who knows why.

Experiment purgatory prevents you from moving on to other, more fruitful ideas. Accept that negative experimental results are simply part of the process, and move on when you need to. It’s how an empirical science is supposed to progress.

**Coda: always be experimenting**

To summarize,

* **know when to experiment**: prioritize experiments with the most expected gain,
* **always start with a hypothesis**: avoid the pseudoscience of HARKing,
* **create tight feedback loops**: make it as easy as possible for you to test ideas quickly,
* **avoid “shiny new thing” bias**: remember that success on academic problems isn’t necessarily a good indicator for success on business problems,
* **avoid experiment purgatory**: accept that negative results are part of the process and move on when you need to.

Let me end with a piece of advice that my science manager at Amazon once gave me:

The machines of the best ML scientists are rarely idle.

What he meant was that the best ML scientists always have a backlog of experiments that they want to run, which correspond to different hypotheses that they’ve formulated and want to test. Whenever their machines are about to sit idle (such as when they’re about to take off for the weekend), they simply submit experiments from their backlog before they log off.

Machine Learning is an empirical field. More experimentation leads to more knowledge and ultimately more expertise. Master the art of impactful ML experimentation, and you’re on your way to become an ML expert.