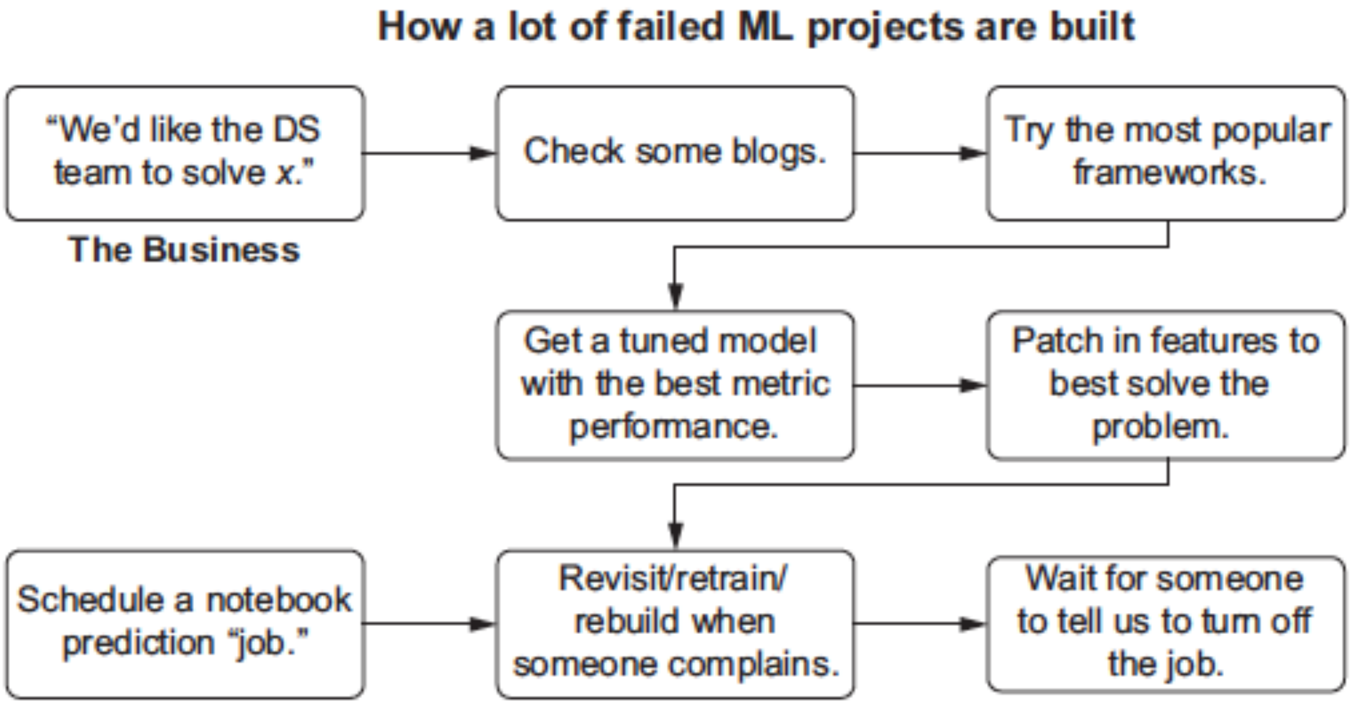
Machine Learning Systems Design



How a lot of failed ML projects are built

“We'd like the DS team to solve *x*.”

The Business

Check some blogs.

Try the most popular frameworks.

Get a tuned model with the best metric performance.

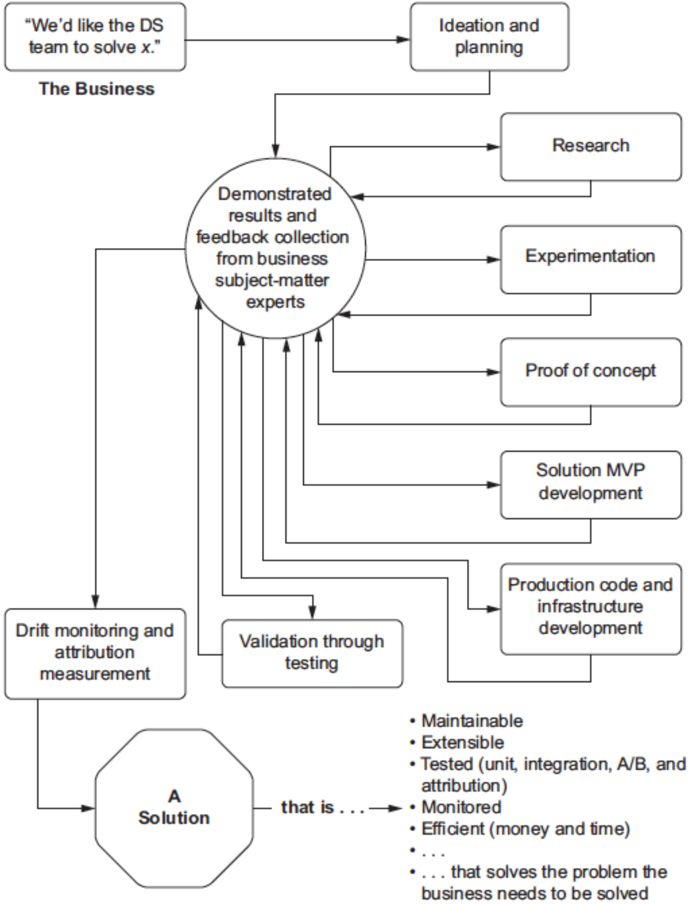
Patch in features to best solve the problem.

Schedule a notebook prediction “job.”

Revisit/retrain/ rebuild when someone complains.

Wait for someone to tell us to turn off the job.

“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.



"We'd like the DS team to solve x."
The Business
Ideation and planning
Research
Demonstrated results and feedback collection from business subject-matter experts
Experimentation
Proof of concept
Solution MVP development
Production code and infrastructure development
Drift monitoring and attribution measurement
Validation through testing
A Solution
that is . . .

- Maintainable
- Extensible
- Tested (unit, integration, A/B, and attribution)
- Monitored
- Efficient (money and time)
- . . .
- . . . that solves the problem the business needs to be solved

"Machine Learning Engineering in Action", Ben Wilson, Manning, 2022.

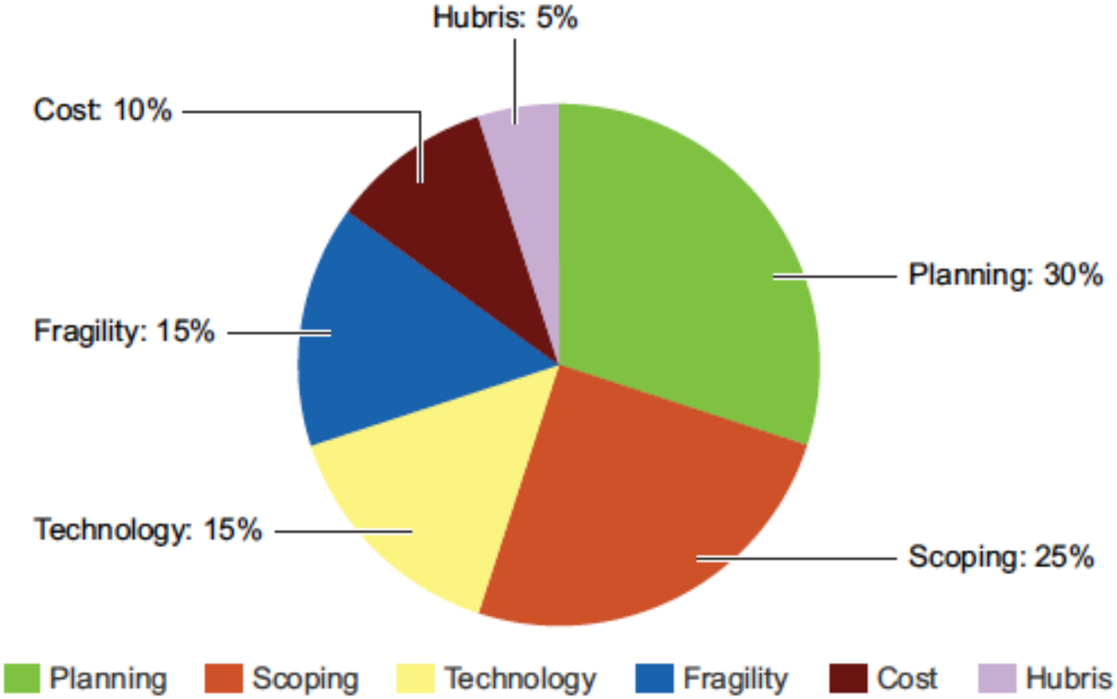


Figure 1.2 My estimation of why ML projects fail, from the hundreds I've worked on and advised others on

"Machine Learning Engineering in Action", Ben Wilson, Manning, 2022.

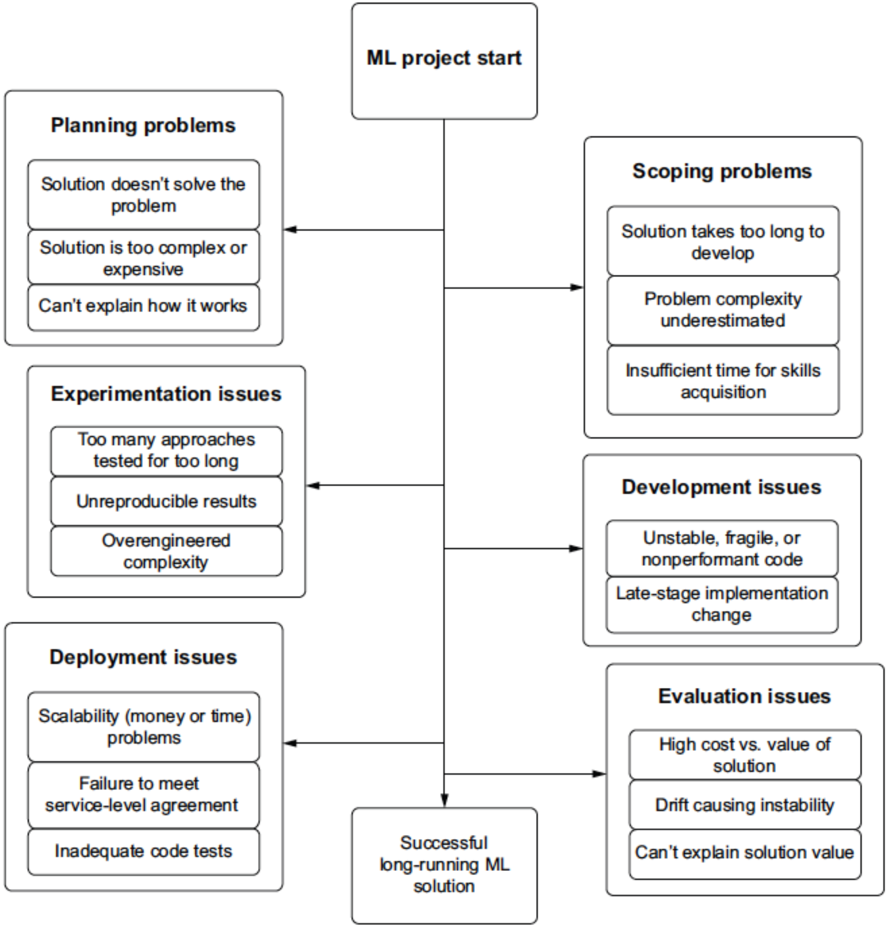
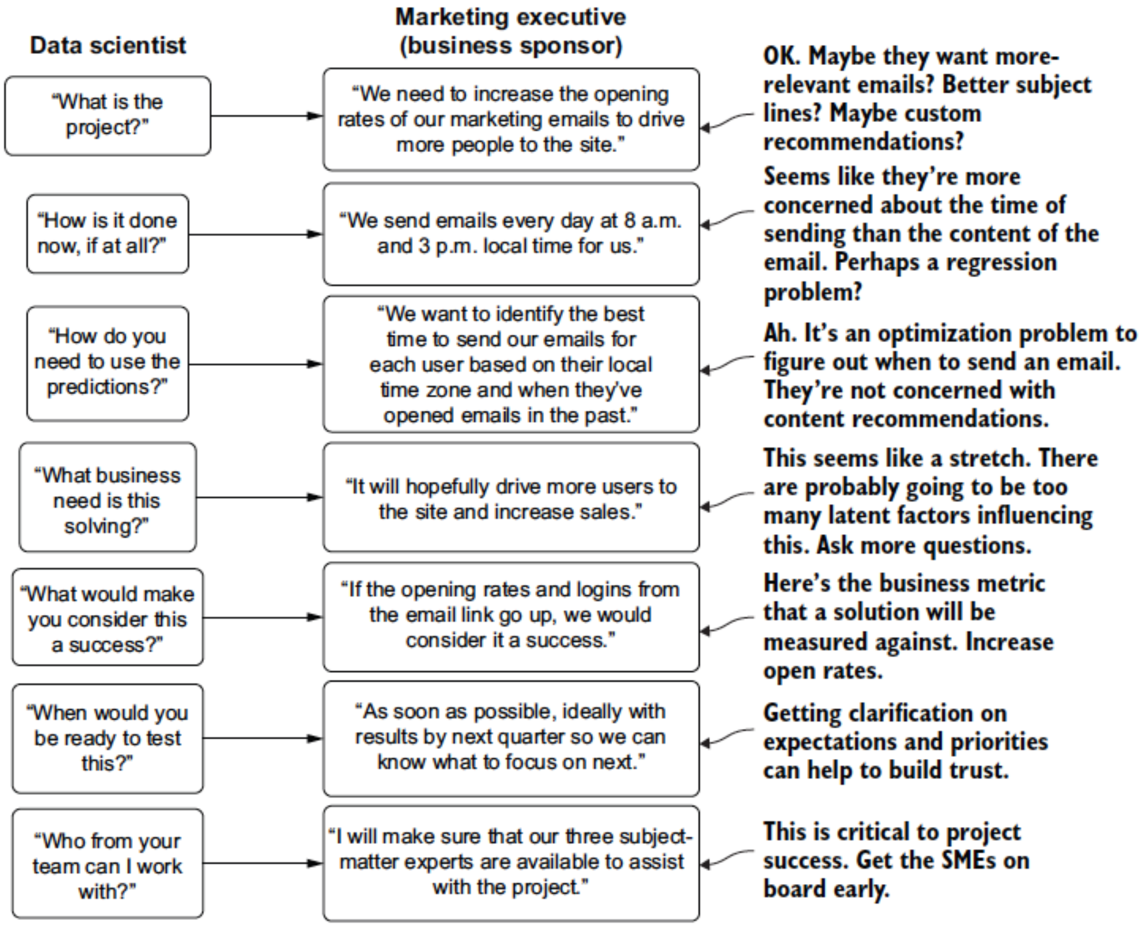

ML project start
Planning problems
Solution doesn't solve the problem
Solution is too complex or expensive
Can't explain how it works
Scoping problems
Solution takes too long to develop
Problem complexity underestimated
Insufficient time for skills acquisition
Experimentation issues
Too many approaches tested for too long
Unreproducible results
Overengineered complexity
Development issues
Unstable, fragile, or nonperformant code
Late-stage implementation change
Deployment issues
Scalability (money or time) problems
Failure to meet service-level agreement
Inadequate code tests
Evaluation issues
High cost vs. value of solution
Drift causing instability
Can't explain solution value
Successful long-running ML solution

Figure 1.3 **ML project detours that lead to project failure**

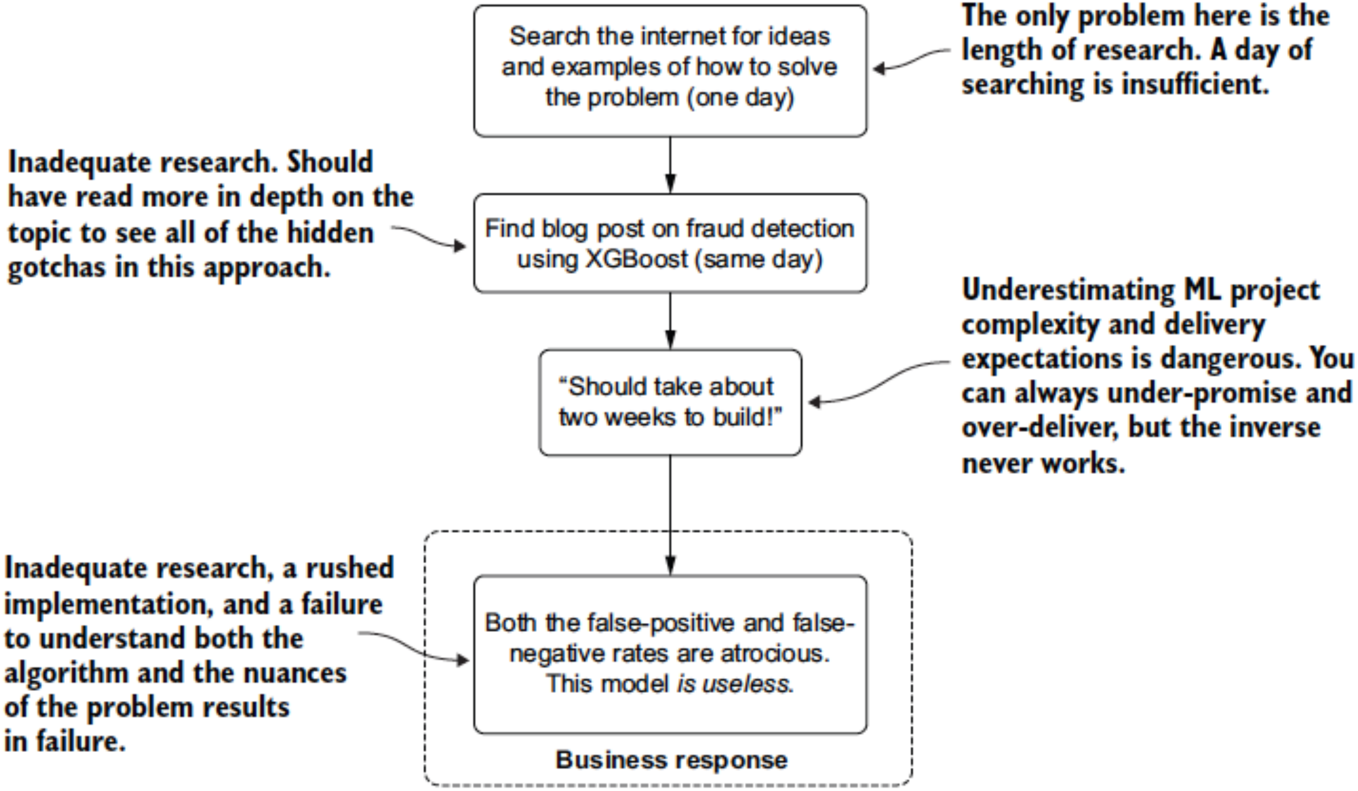
“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.



Data scientist	Marketing executive (business sponsor)	
"What is the project?"	"We need to increase the opening rates of our marketing emails to drive more people to the site."	OK. Maybe they want more-relevant emails? Better subject lines? Maybe custom recommendations?
"How is it done now, if at all?"	"We send emails every day at 8 a.m. and 3 p.m. local time for us."	Seems like they're more concerned about the time of sending than the content of the email. Perhaps a regression problem?
"How do you need to use the predictions?"	"We want to identify the best time to send our emails for each user based on their local time zone and when they've opened emails in the past."	Ah. It's an optimization problem to figure out when to send an email. They're not concerned with content recommendations.
"What business need is this solving?"	"It will hopefully drive more users to the site and increase sales."	This seems like a stretch. There are probably going to be too many latent factors influencing this. Ask more questions.
"What would make you consider this a success?"	"If the opening rates and logins from the email link go up, we would consider it a success."	Here's the business metric that a solution will be measured against. Increase open rates.
"When would you be ready to test this?"	"As soon as possible, ideally with results by next quarter so we can know what to focus on next."	Getting clarification on expectations and priorities can help to build trust.
"Who from your team can I work with?"	"I will make sure that our three subject-matter experts are available to assist with the project."	This is critical to project success. Get the SMEs on board early.

Figure 1.4 A simplified planning discussion diagram

"Machine Learning Engineering in Action", Ben Wilson, Manning, 2022.



Search the internet for ideas and examples of how to solve the problem (one day)

The only problem here is the length of research. A day of searching is insufficient.

Inadequate research. Should have read more in depth on the topic to see all of the hidden gotchas in this approach.

Find blog post on fraud detection using XGBoost (same day)

“Should take about two weeks to build!”

Underestimating ML project complexity and delivery expectations is dangerous. You can always under-promise and over-deliver, but the inverse never works.

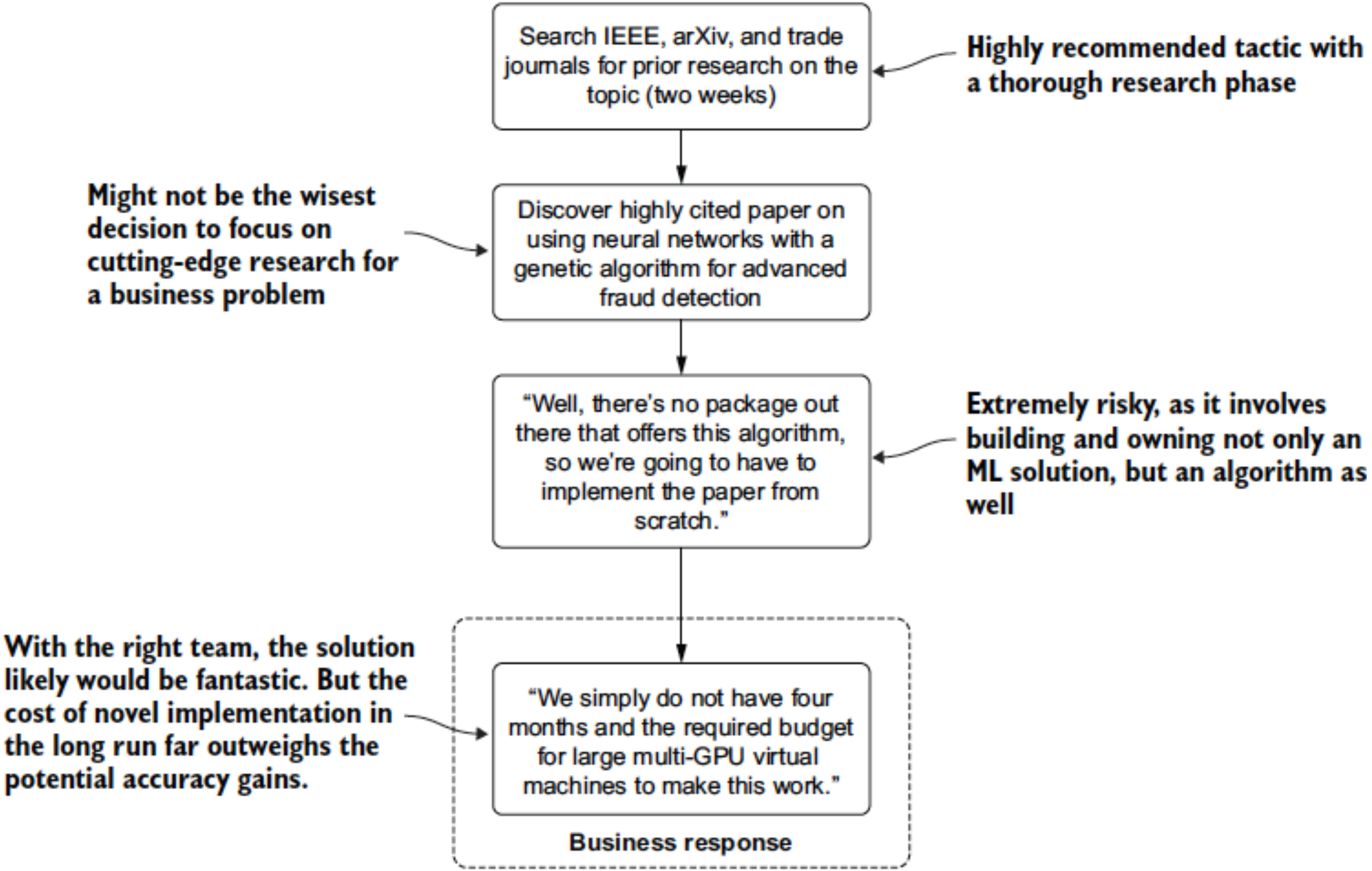
Inadequate research, a rushed implementation, and a failure to understand both the algorithm and the nuances of the problem results in failure.

Both the false-positive and false-negative rates are atrocious. This model *is useless*.

Business response

Figure 1.5 Research and scoping of a fraud-detection problem for a junior team of well-intentioned but inexperienced data scientists

“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.



Search IEEE, arXiv, and trade journals for prior research on the topic (two weeks)

Highly recommended tactic with a thorough research phase

Discover highly cited paper on using neural networks with a genetic algorithm for advanced fraud detection

Might not be the wisest decision to focus on cutting-edge research for a business problem

“Well, there's no package out there that offers this algorithm, so we're going to have to implement the paper from scratch.”

Extremely risky, as it involves building and owning not only an ML solution, but an algorithm as well

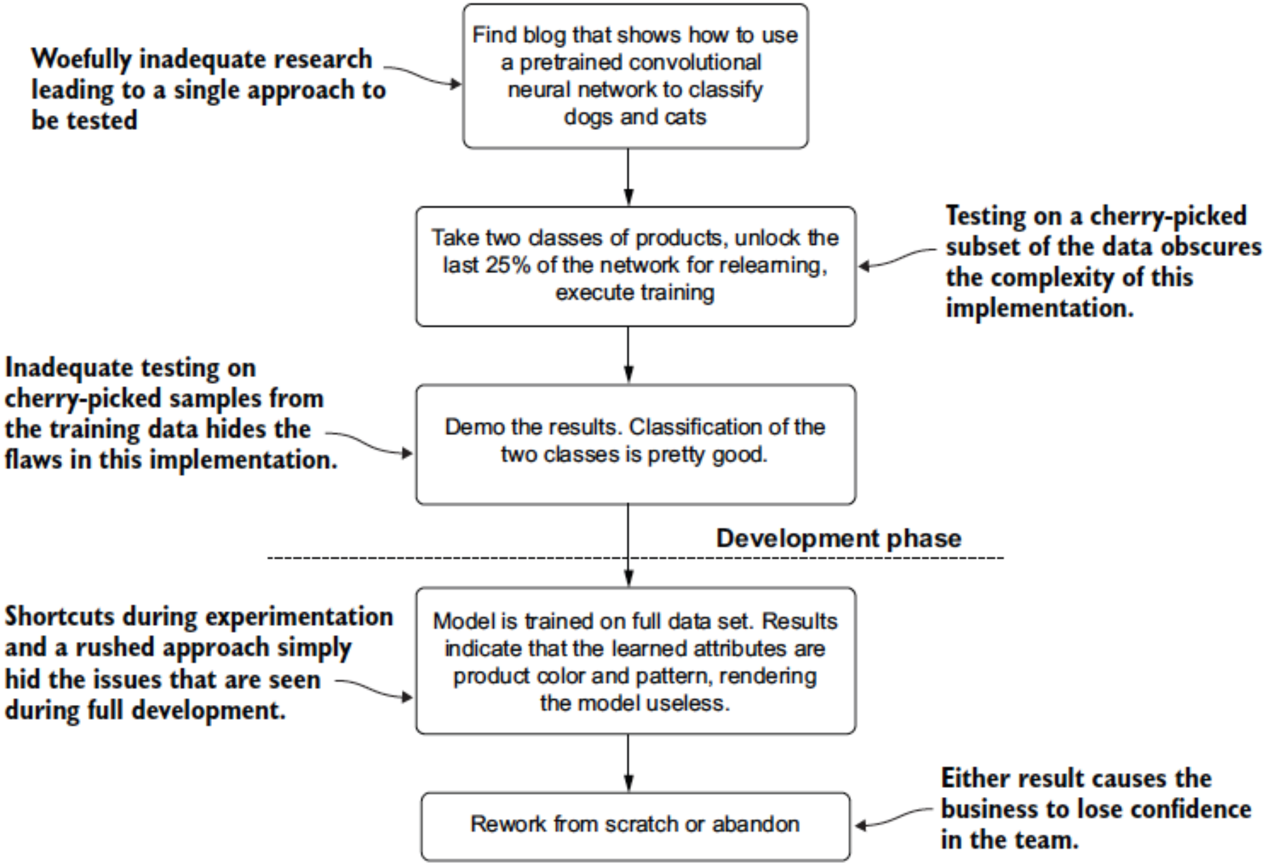
“We simply do not have four months and the required budget for large multi-GPU virtual machines to make this work.”

Business response

With the right team, the solution likely would be fantastic. But the cost of novel implementation in the long run far outweighs the potential accuracy gains.

Figure 1.6 Research and scoping for an academia-focused group of researchers for the fraud-detection problem

“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.



Woefully inadequate research leading to a single approach to be tested

Find blog that shows how to use a pretrained convolutional neural network to classify dogs and cats

Take two classes of products, unlock the last 25% of the network for relearning, execute training

Testing on a cherry-picked subset of the data obscures the complexity of this implementation.

Inadequate testing on cherry-picked samples from the training data hides the flaws in this implementation.

Demo the results. Classification of the two classes is pretty good.

Development phase

Shortcuts during experimentation and a rushed approach simply hid the issues that are seen during full development.

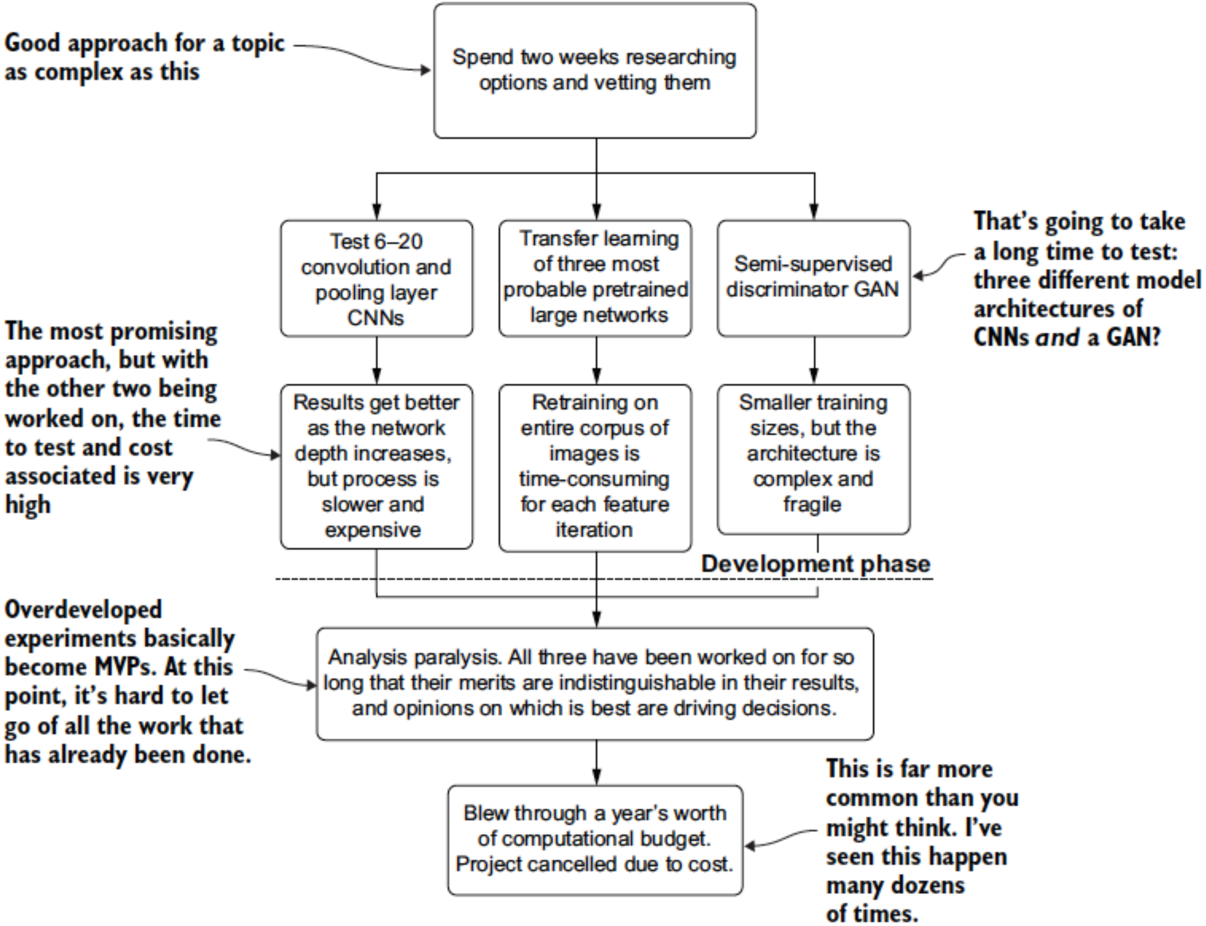
Model is trained on full data set. Results indicate that the learned attributes are product color and pattern, rendering the model useless.

Rework from scratch or abandon

Either result causes the business to lose confidence in the team.

Figure 1.7 A rushed experimentation phase by a team of inexperienced data scientists

“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.



Good approach for a topic as complex as this

Spend two weeks researching options and vetting them

Test 6–20 convolution and pooling layer CNNs

Transfer learning of three most probable pretrained large networks

Semi-supervised discriminator GAN

That's going to take a long time to test: three different model architectures of CNNs *and* a GAN?

The most promising approach, but with the other two being worked on, the time to test and cost associated is very high

Results get better as the network depth increases, but process is slower and expensive

Retraining on entire corpus of images is time-consuming for each feature iteration

Smaller training sizes, but the architecture is complex and fragile

Development phase

Overdeveloped experiments basically become MVPs. At this point, it's hard to let go of all the work that has already been done.

Analysis paralysis. All three have been worked on for so long that their merits are indistinguishable in their results, and opinions on which is best are driving decisions.

Blew through a year's worth of computational budget. Project cancelled due to cost.

This is far more common than you might think. I've seen this happen many dozens of times.

Figure 1.8 A case of too much testing in the experimentation phase of a project

"Machine Learning Engineering in Action", Ben Wilson, Manning, 2022.

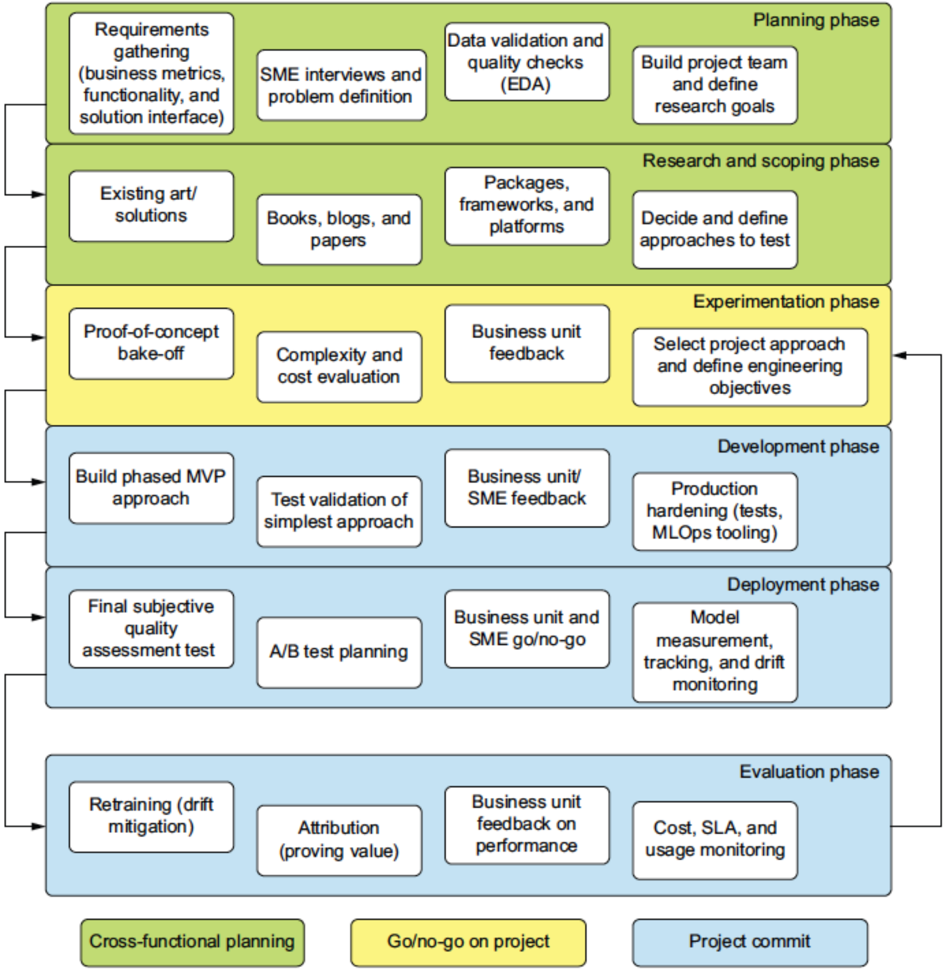

Planning phase
Requirements gathering (business metrics, functionality, and solution interface)
SME interviews and problem definition
Data validation and quality checks (EDA)
Build project team and define research goals
Research and scoping phase
Existing art/ solutions
Books, blogs, and papers
Packages, frameworks, and platforms
Decide and define approaches to test
Experimentation phase
Proof-of-concept bake-off
Complexity and cost evaluation
Business unit feedback
Select project approach and define engineering objectives
Development phase
Build phased MVP approach
Test validation of simplest approach
Business unit/ SME feedback
Production hardening (tests, MLOps tooling)
Deployment phase
Final subjective quality assessment test
A/B test planning
Business unit and SME go/no-go
Model measurement, tracking, and drift monitoring
Evaluation phase
Retraining (drift mitigation)
Attribution (proving value)
Business unit feedback on performance
Cost, SLA, and usage monitoring
Cross-functional planning
Go/no-go on project
Project commit

Figure 1.15 The ML engineering methodology component map

“Machine Learning Engineering in Action”, Ben Wilson, Manning, 2022.

Why data science projects fail?

https://github.com/xLaszlo/datascience-fails

CONGRATS
YOU FAILED
imgflip.com