

The rise of MLOps

Catarina Silva





The rise of MLOps

- ML in real world
- MLOps problems
- MLOps tools
- MLOps maturity





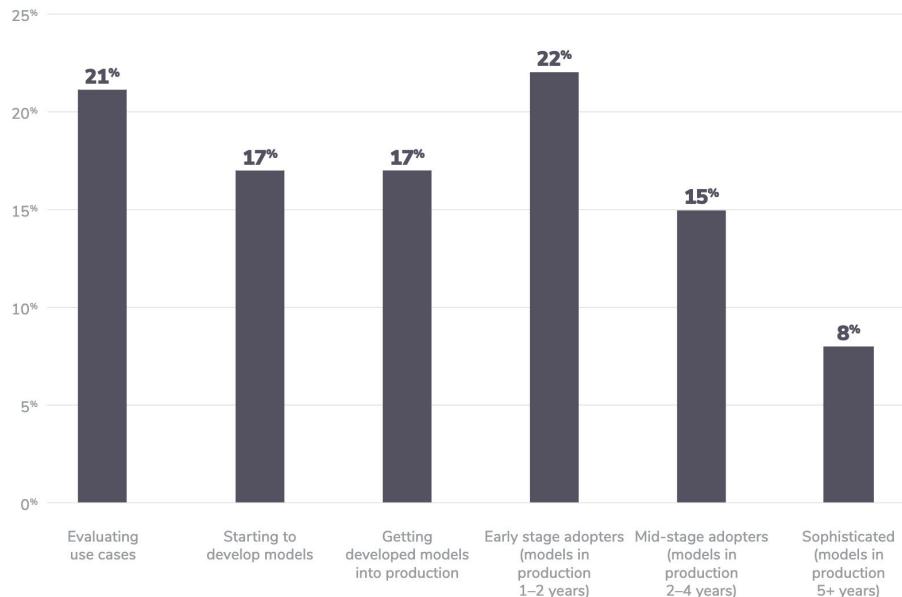
The era of Machine Learning

- Recent improvements in several different fields
- New techniques to improve resource usage
- Huge economical motivation
- Hype - AI for the sake of AI

The era of Machine Learning

Influx of new job titles such as

- Machine Learning Engineer,
- ML Developer,
- ML Architect,
- Data Engineer,
- Machine Learning Operations (ML Ops),
- AI Ops



https://info.algorithmia.com/hubfs/2019/Whitepapers/The-State-of-Enterprise-ML-2020/Algorithmia_2020_State_of_Enterprise_ML.pdf



Around 90% of machine learning projects never make it into production



Main blockers for ML Success

- Data comes with its own set of issues



Main blockers for ML Success

- Data comes with its own set of issues
- Models have different sets of requirements



Main blockers for ML Success

- Data comes with its own set of issues
- Models have different sets of requirements
- AI is currently like a one hit wonder - deploy once with no follow up



Main blockers for ML Success

- Data comes with its own set of issues
- Models have different sets of requirements
- AI is currently like a one hit wonder - deploy once with no follow up
- There is a gap between research processes and production-focused processes



Main blockers for ML Success

- Data comes with its own set of issues
- Models have different sets of requirements
- AI is currently like a one hit wonder - deploy once with no follow up
- There is a gap between research processes and production-focused processes
- Get one of the pieces wrong, and the others won't fit
 - High interest debt accumulator

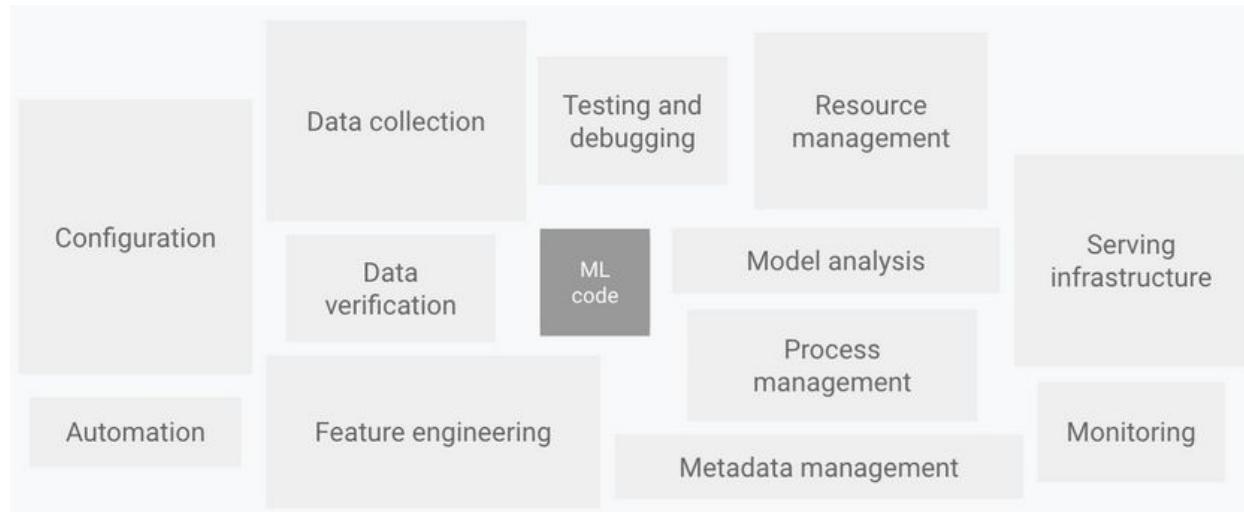


Main blockers for ML Success

- Data comes with its own set of issues
- Models have different sets of requirements
- AI is currently like a one hit wonder - deploy once with no follow up
- There is a gap between research processes and production-focused processes
- Get one of the pieces wrong, and the others won't fit
 - High interest debt accumulator
- Silos between data scientists building models and remaining teams
 - People are very attached to their ways of working



Main blockers for ML Success

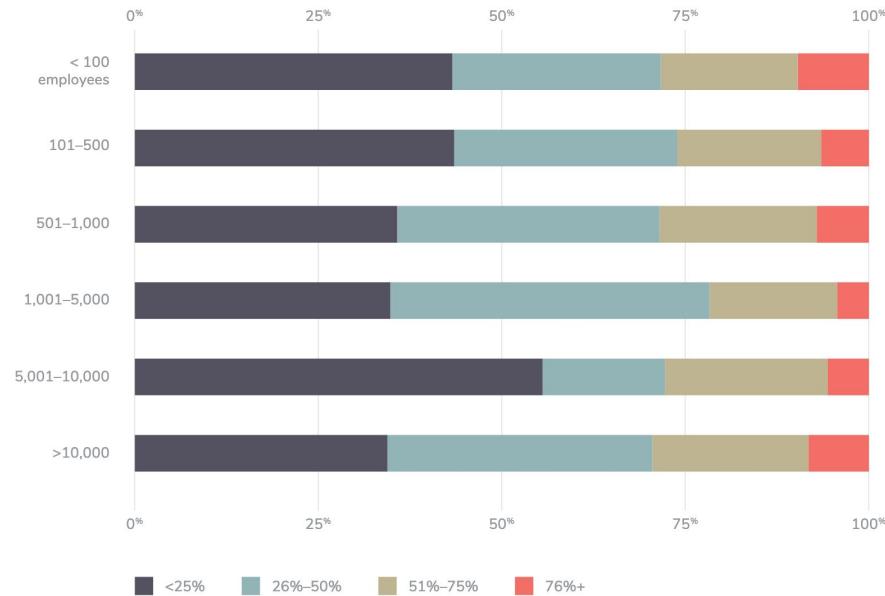


<https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>

Main blockers for ML Success

- Deploying a model should be as easy as deploying code
- DevOps brought a change to code deployment, can MLOps bring the same change?

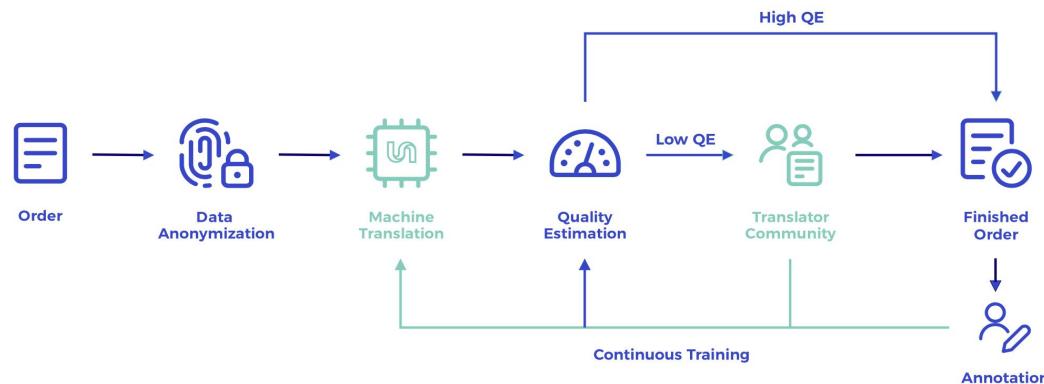
Time data scientists spend deploying models by company size



<https://info.algorithmia.com/hubfs/2019/Whitepapers/The-State-of-Enterprise-ML-2020/Algorithmia-2020-State-of-Enterprise-ML.pdf>

Where I'm coming from

- A company where AI is one of the central pieces
- We have high cardinality of models (several models / several different projects)
- We've scaled and are still scaling - growing pains





Where I'm coming from

- I'm an instructor here
- Teaching model deployment and other concepts around
- Much much lower scale
- Completely different type of applications



Specialization #6 - Data Science in the Real World

BLU13 Basic model Deployment	BLU14 Deployment in the real world	BLU15 Model CSI	Hackathon #6
------------------------------------	--	--------------------	--------------



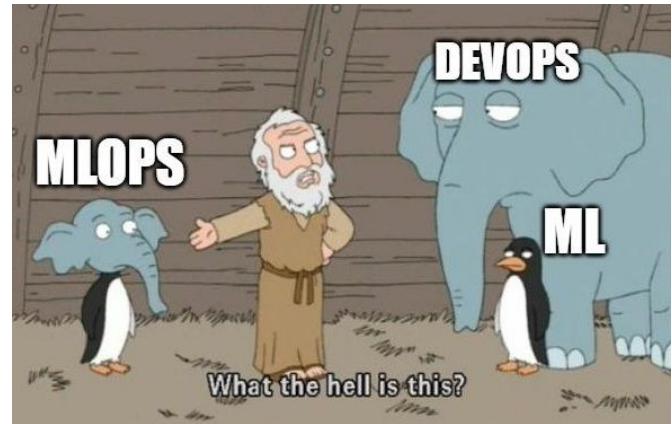
What is MLOps?



What is MLOps ?

MLOps is the set of Machine Learning and Engineering culture and practices that have as a goal to unify the development and operation of Machine Learning systems.

Similarly to devops, it aims to shorten the systems development life cycle and increase an organization's ability to deliver high quality machine learning systems quickly and regularly.





MLOps differs from DevOps where Machine Learning
differs from Software Engineering



Traditional vs. ML Engineering

Traditional Programming



Machine Learning



<https://machinelearningmastery.com/basic-concepts-in-machine-learning/>

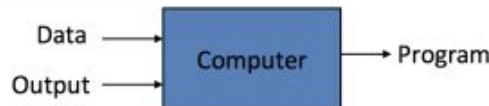
Traditional vs. ML Engineering

Traditional Programming



- ✓ Desired behavior is well defined
- ✓ Program is well known

Machine Learning



<https://machinelearningmastery.com/basic-concepts-in-machine-learning/>

Traditional vs. ML Engineering

Traditional Programming



✓ Desired behavior is well defined

✓ Program is well known

Machine Learning



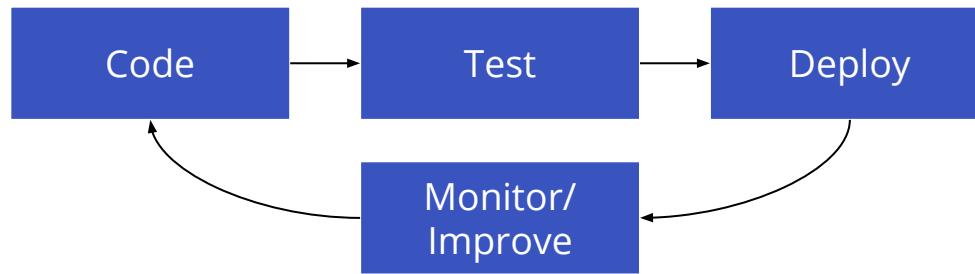
⚠ Desired behavior is not 100% defined

⚠ Program is the result of the algorithm



Traditional Software Engineering

SoftEng





Traditional Software Engineering

SoftEng



- ✓ Continuous development
- ✓ Continuous testing
- ✓ Continuous integration

Traditional Software Engineering

SoftEng



- ✓ Continuous development
- ✓ Continuous testing
- ✓ Continuous integration

- ✓ Continuous deployment
- ✓ Continuous monitoring
- ✓ Infrastructure as code



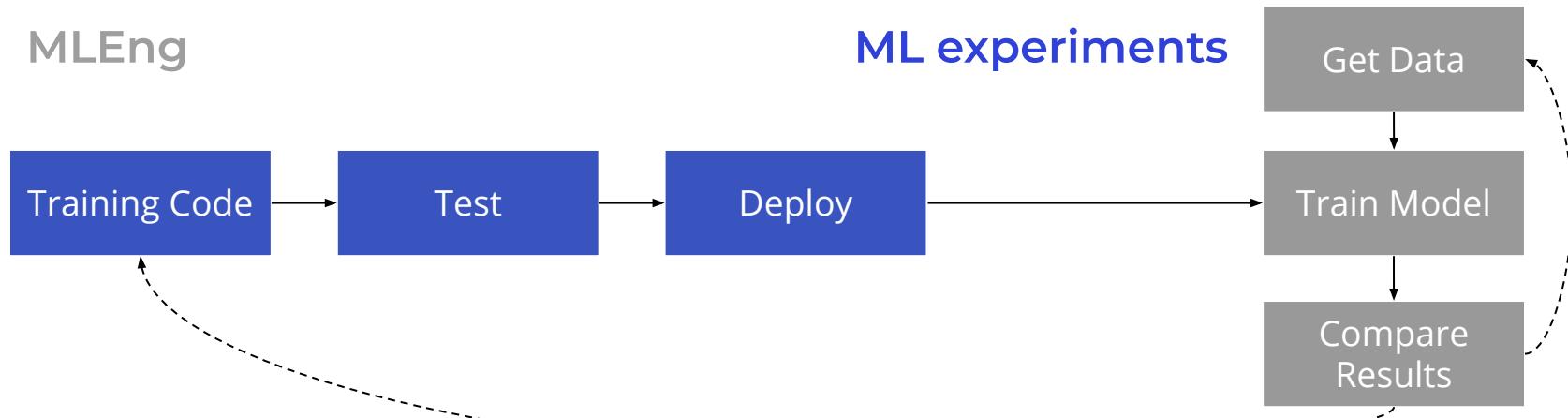
Machine Learning Engineering

MLEng



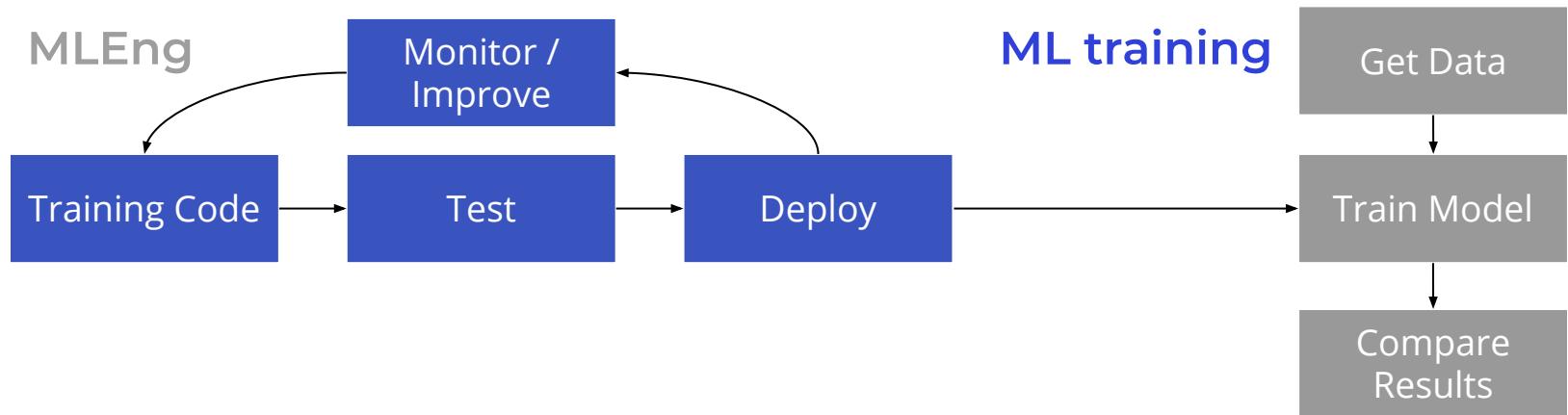


Machine Learning Engineering



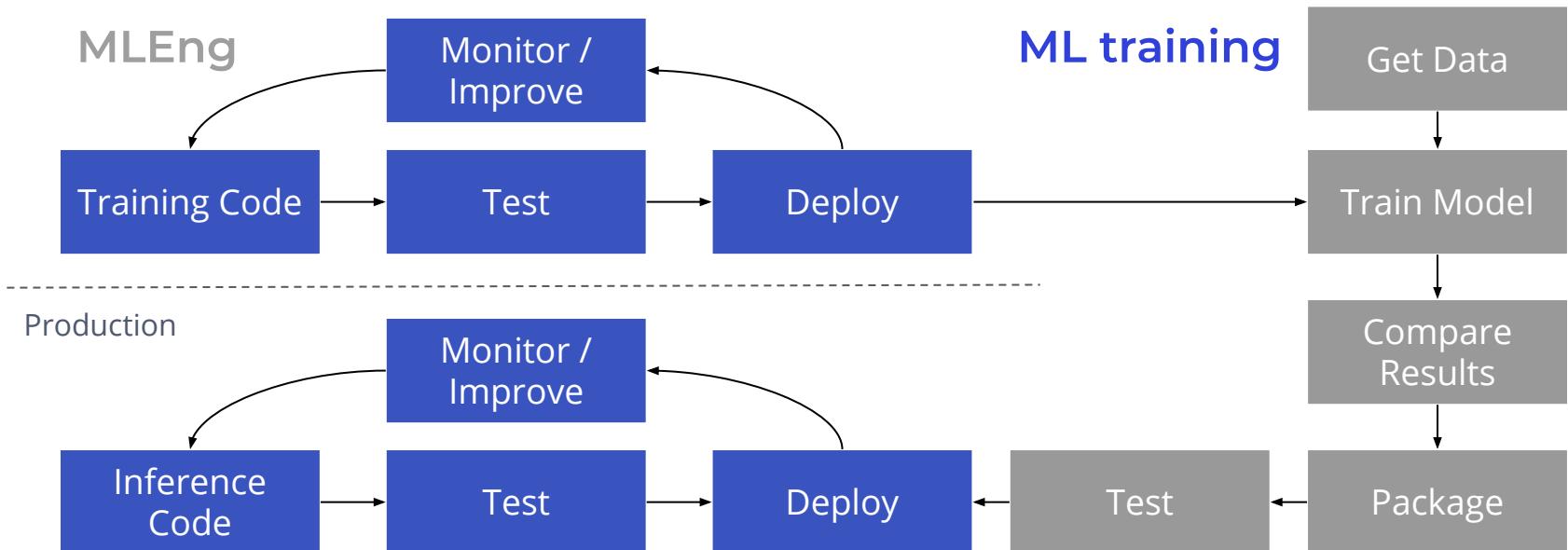


Machine Learning Engineering





Machine Learning Engineering





Machine Learning is essentially experimental

Models are often black boxes

- You can't differentiate between two models, so how do you compare them?
- Evolving surroundings have an impact on performance



<https://xkcd.com/1838/>

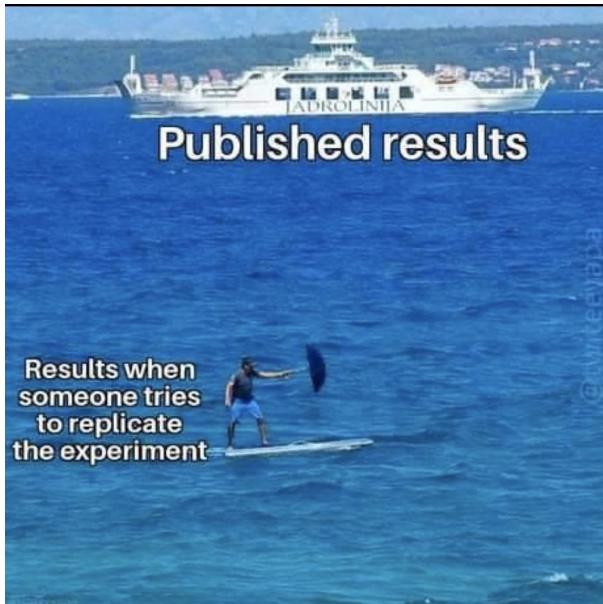


Models are hard to scale

- If a model works great in a small environment it doesn't mean it will work great in any case
- Usually datasets are very big
- Models are also somewhat big, and getting bigger everyday
- Inference can be slow



Models are hard to reproduce



Models are a combination of data, parameters and algorithms, including random seed pick

Results may not scale well from one scenario to the other, even when they are close

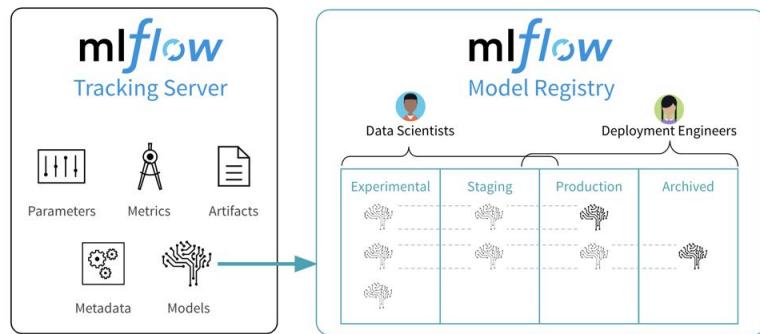


Reproducibility leads to efficiency

- ✓ Experiment tracking
- ✓ Model and dataset management
- ✓ Data Provenance

Models are hard to reproduce

- ✓ Experiment tracking
- ✓ Model and dataset management
- ✓ Data Provenance



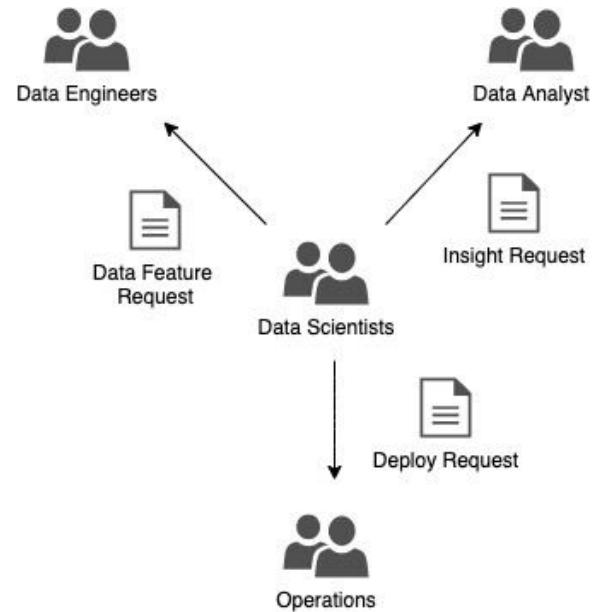
<https://databricks.com/product/managed-mlflow>



Models are only as good as the data they see

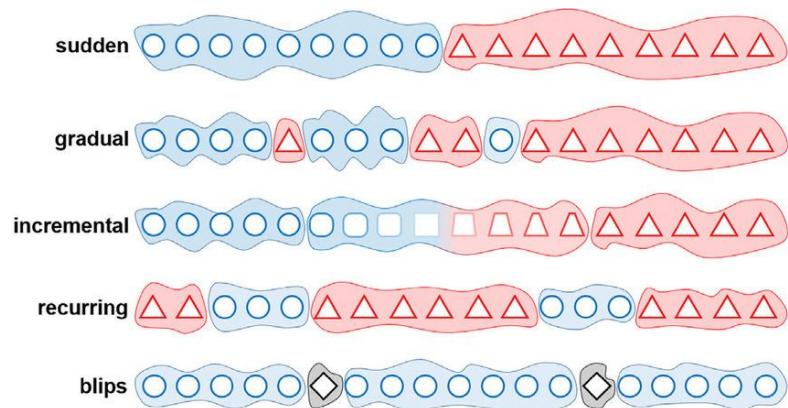
Data ingestion

- Data comes from many different sources
- Sometimes there are processes that block access to data
- A solid data platform can be a game changer for data science teams



Model Monitoring

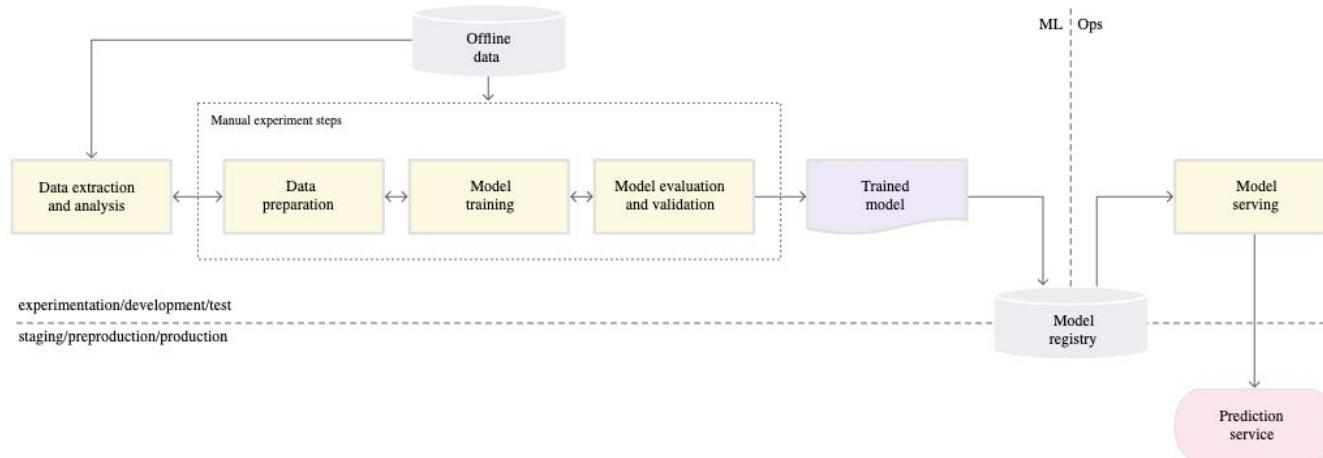
- Real world data is constantly changing
- **Model Degradation** can be caused by data shifts



[Krawczyk, Bartosz & Cano, Alberto. \(2018\). Online Ensemble Learning with Abstaining Classifiers for Drifting and Noisy Data Streams. Applied Soft Computing. 68. 677-692. 10.1016/j.asoc.2017.12.008.](https://doi.org/10.1016/j.asoc.2017.12.008)

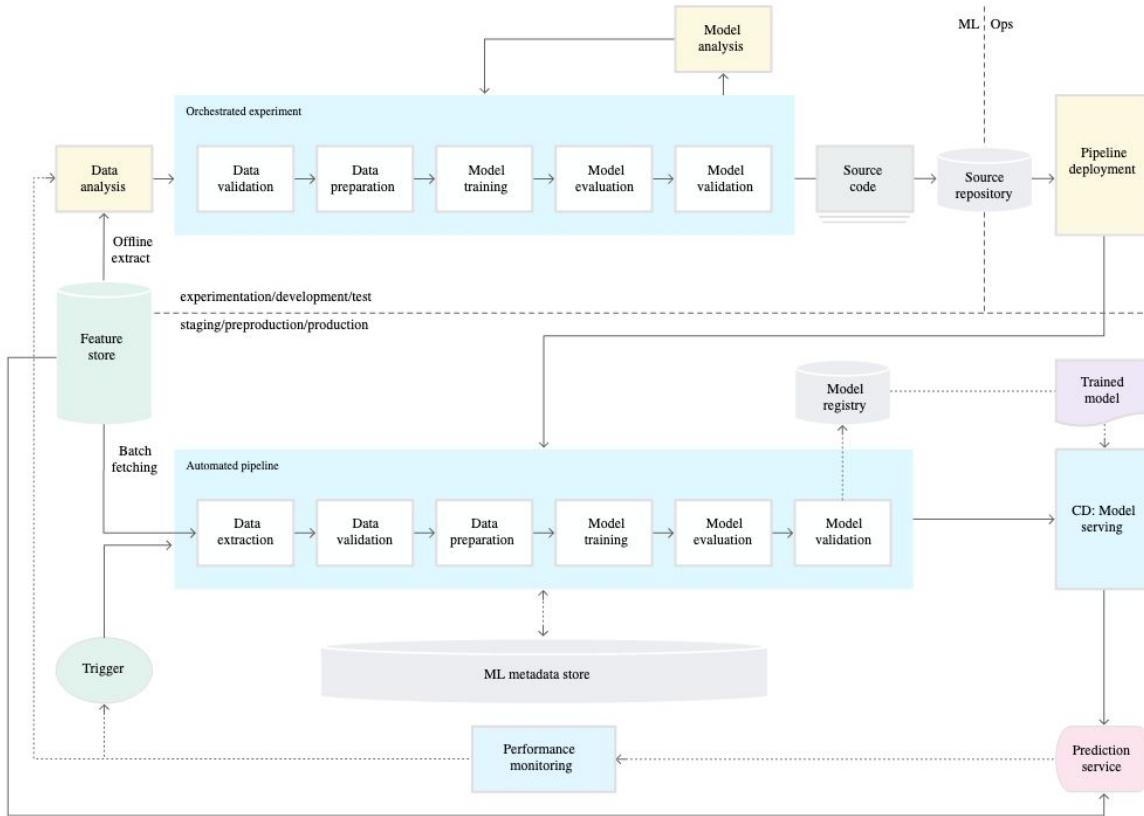
Continuous Training

A regular experimental pipeline, if well established, is easy to run manually. However, if we just want to re-run every day/week/etc with the newest data, it becomes inefficient.



<https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>

The rise of MLOps



<https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>



Training infrastructure

- Most machine learning applications need different infrastructure requirements - eg. GPU
- Usually costly and has peaks of usage
- Typically shared among several researchers and relies on some queue system



Continuous Training is key

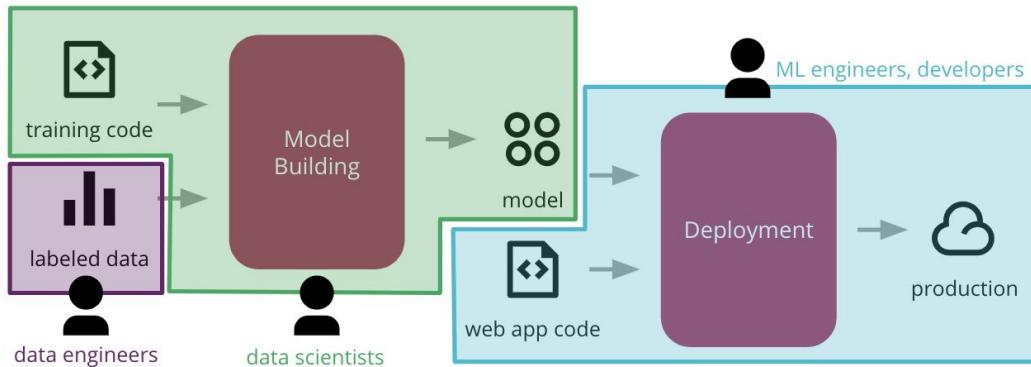
- ✓ Data accessibility and Platform thinking
- ✓ Continuous training and pipeline monitoring
- ✓ Continuous ML monitoring
- ✓ Model degradation detection
- ✓ Training infrastructure management 💰 💰



It takes a village to deliver successful machine
learning projects

MLOps

Putting all these tools and practices in place is not the work only of ML Engineers, but of the whole team. This means that ML engineers are not solely responsible by these pipelines, but are empowered to do their job better and are more aware and accountable for the models they produce.



<https://martinfowler.com/articles/cd4ml.html>



MLOps

- Continuous development
- Continuous testing
- Continuous integration
- Continuous deployment
- Continuous monitoring
- Infrastructure as code
- Experiment tracking
- Asset management
- Data Provenance
- Platform thinking
- Continuous training
- Continuous ML monitoring
- Model degradation detection
- Training infrastructure management



How can I start using MLOps?





Importance of MLOps for you

- MLOps can bring high value, but there is not a one-size-fits-all
- Different maturity levels, different approaches
- There's a trade-off between the value it brings and the costs of implementation

AI with and without MLOps

"The model seems to work!"

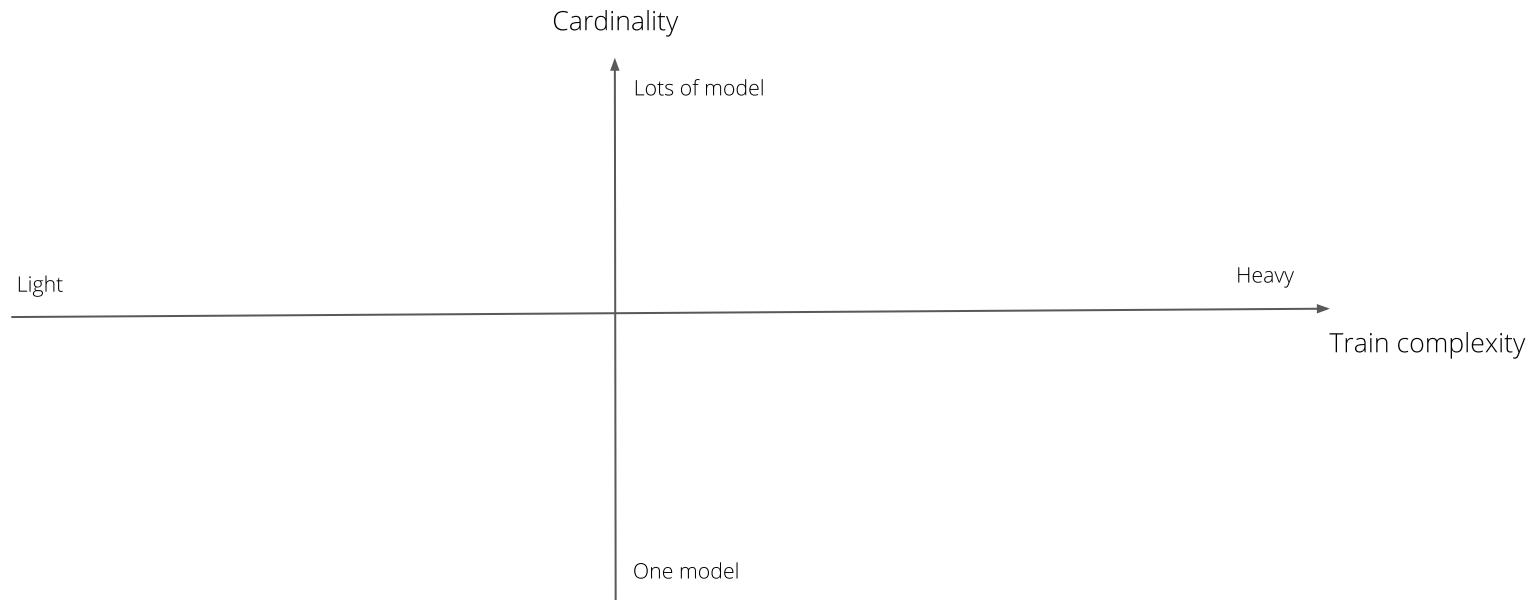
WITHOUT
MLOps

Retraining	Data ingest
Accuracy tracking	Security
Model versioning	Scaling
API serving	Pipeline monitoring
"The model seems to work!"	Efficient GPU utilization

WITH
MLOps

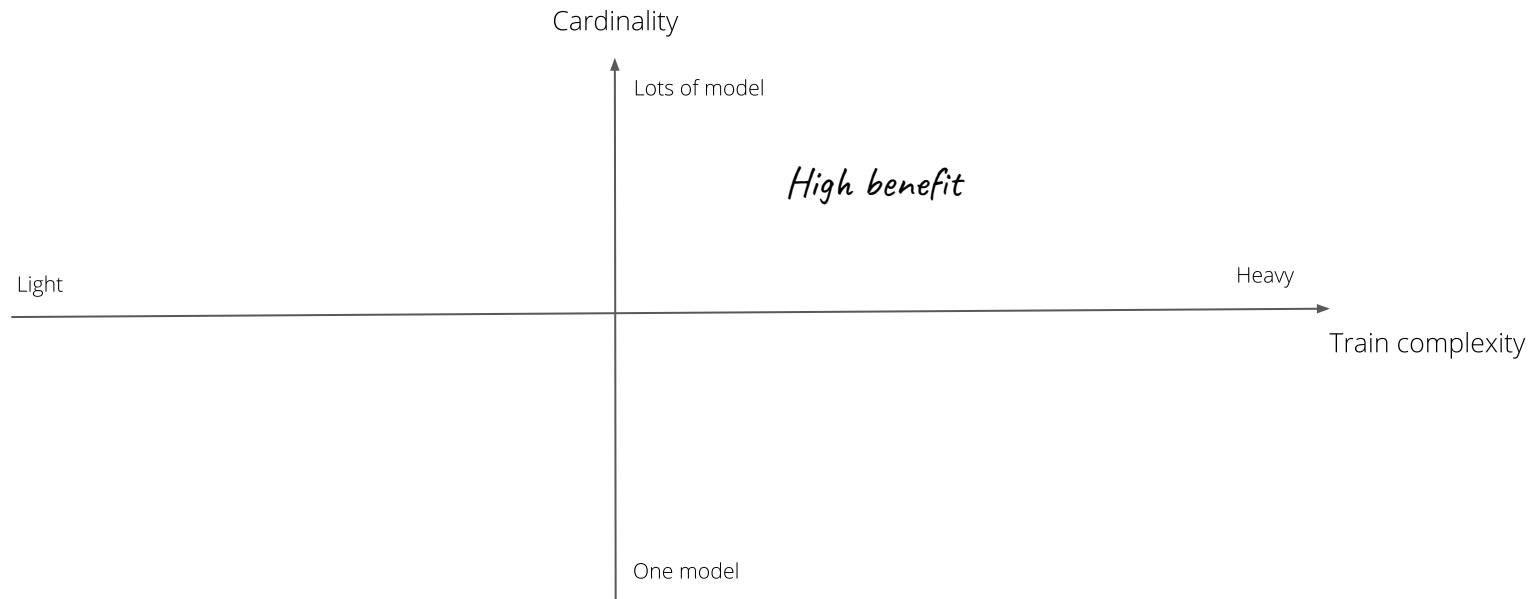


Importance of MLOps for you



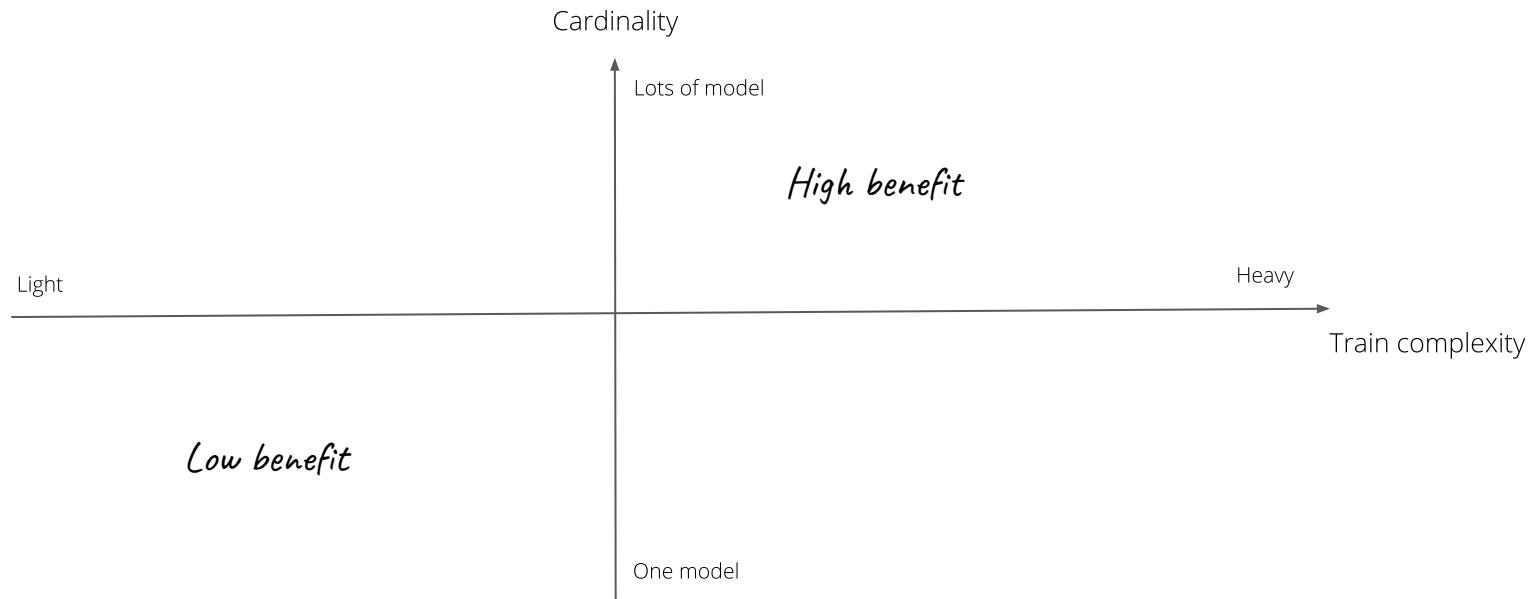


Importance of MLOps for you



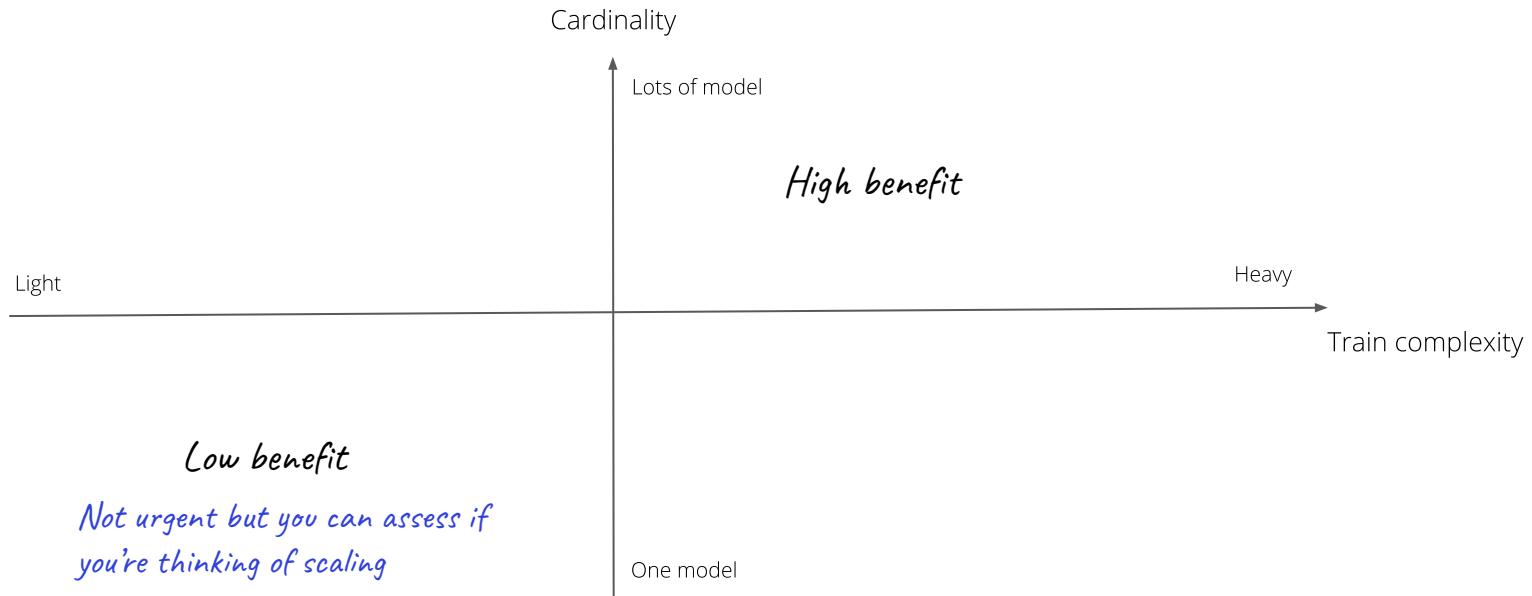


Importance of MLOps for you



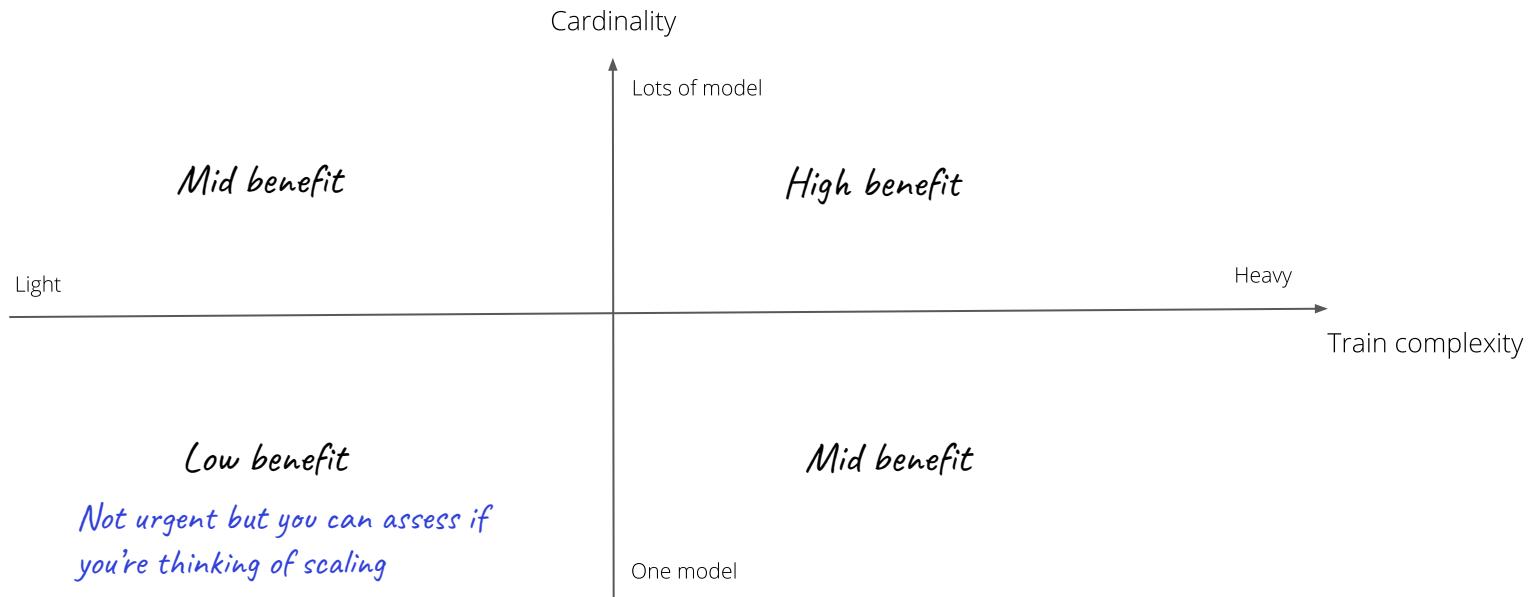


Importance of MLOps for you



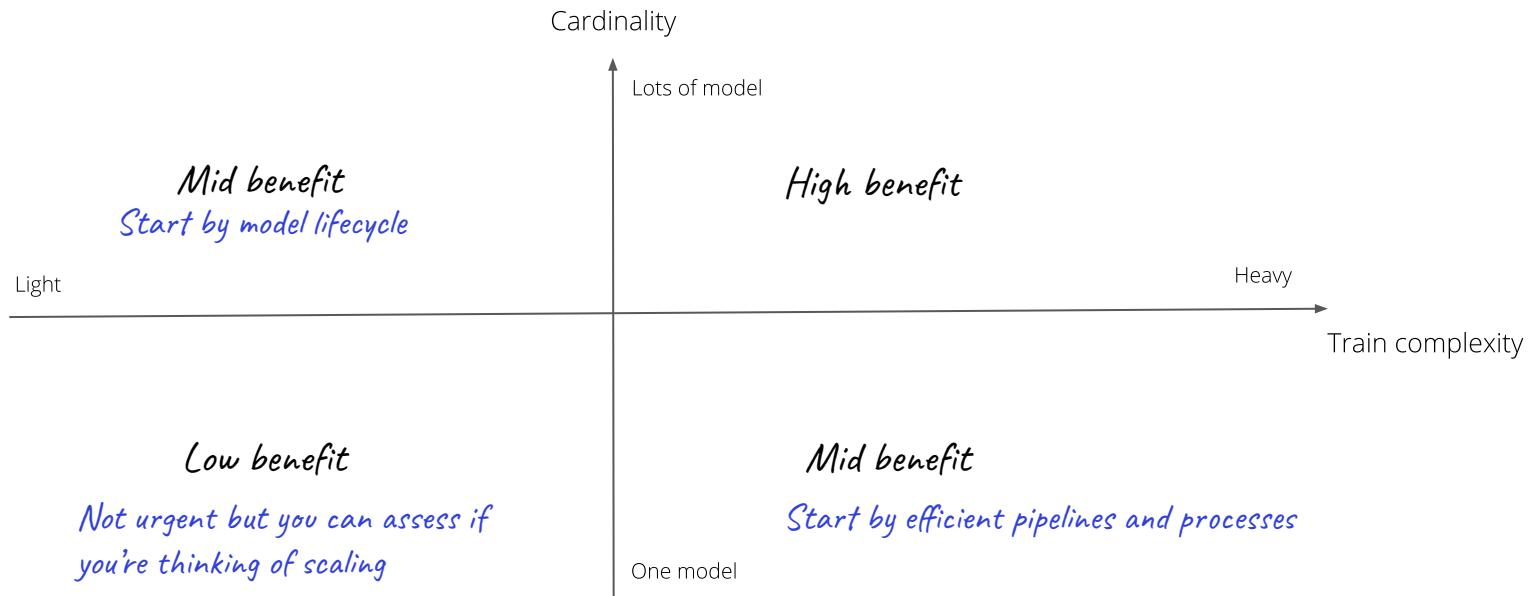


Importance of MLOps for you



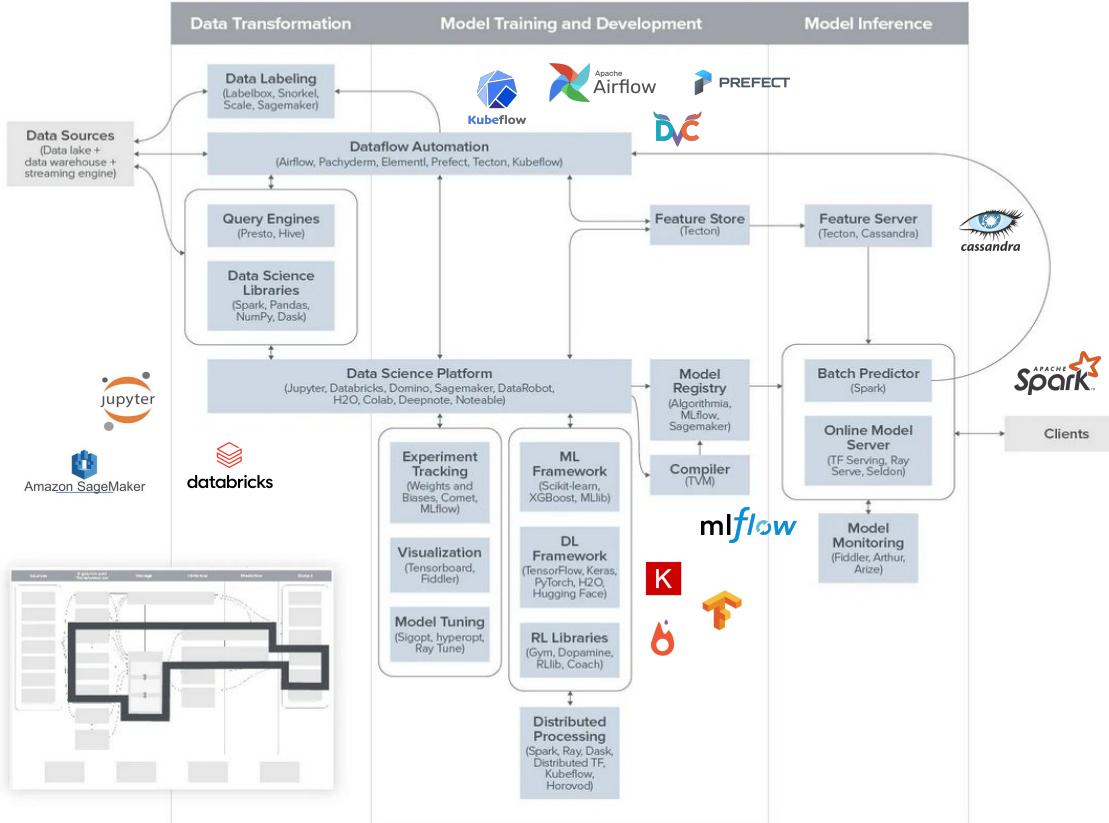


Importance of MLOps for you

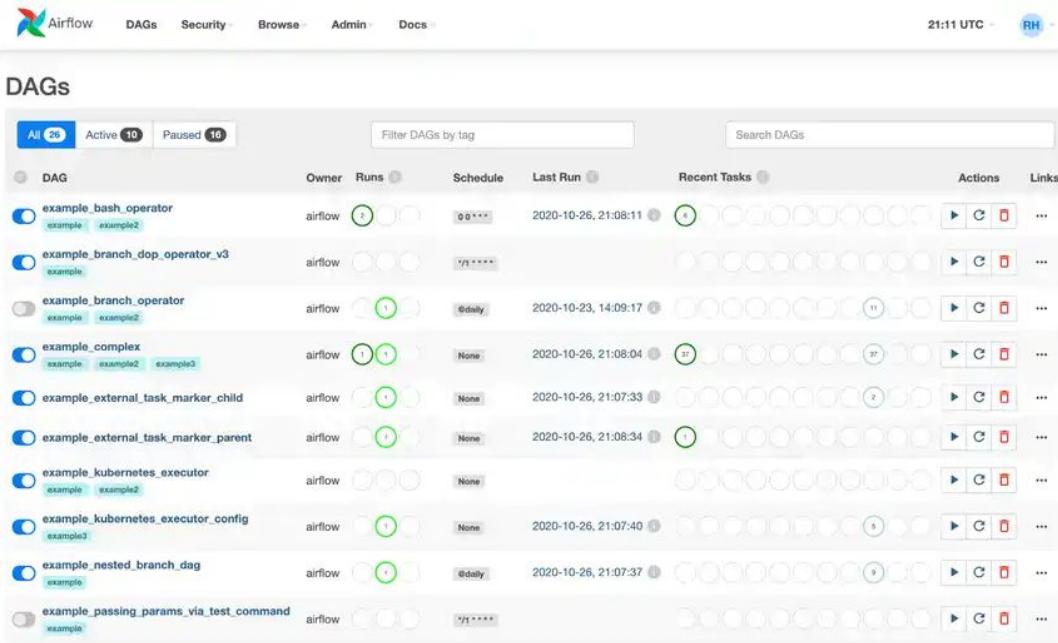


Some tools for MLOps

3. AI and ML Blueprint



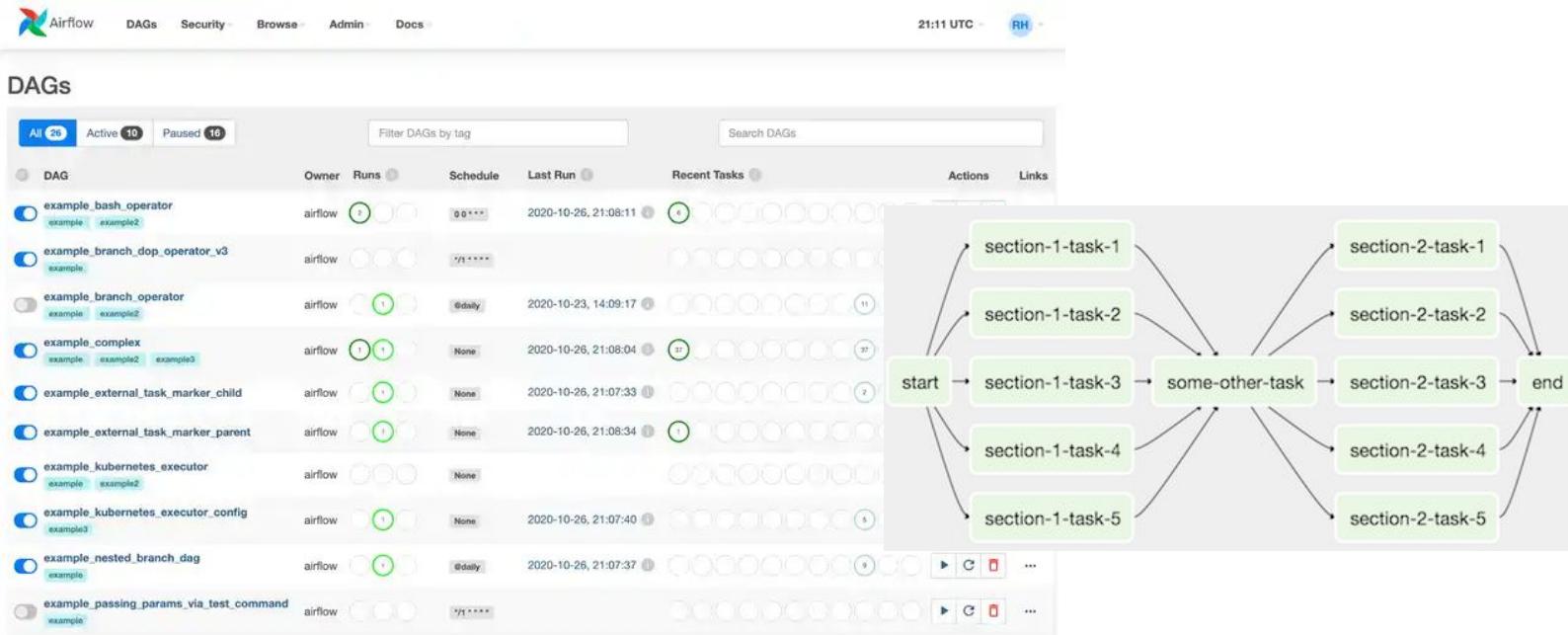
Example: Pipeline management



DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
example_bash_operator	airflow	100	0 * * * *	2020-10-26, 21:08:11	100		
example_branch_dop_operator_v3	airflow	0	*/* * * * *		0		
example_branch_operator	airflow	1	@daily	2020-10-23, 14:09:17	1		
example_complex	airflow	1	None	2020-10-26, 21:08:04	27		
example_external_task_marker_child	airflow	0	None	2020-10-26, 21:07:33	0		
example_external_task_marker_parent	airflow	1	None	2020-10-26, 21:08:34	1		
example_kubernetes_executor	airflow	0	None		0		
example_kubernetes_executor_config	airflow	1	None	2020-10-26, 21:07:40	5		
example_nested_branch_dag	airflow	1	@daily	2020-10-26, 21:07:37	9		
example_passing_params_via_test_command	airflow	0	*/* * * * *		0		

<https://godatadriven.com/blog/apache-airflow-tutorial-for-data-pipelines/>

Example: Pipeline management

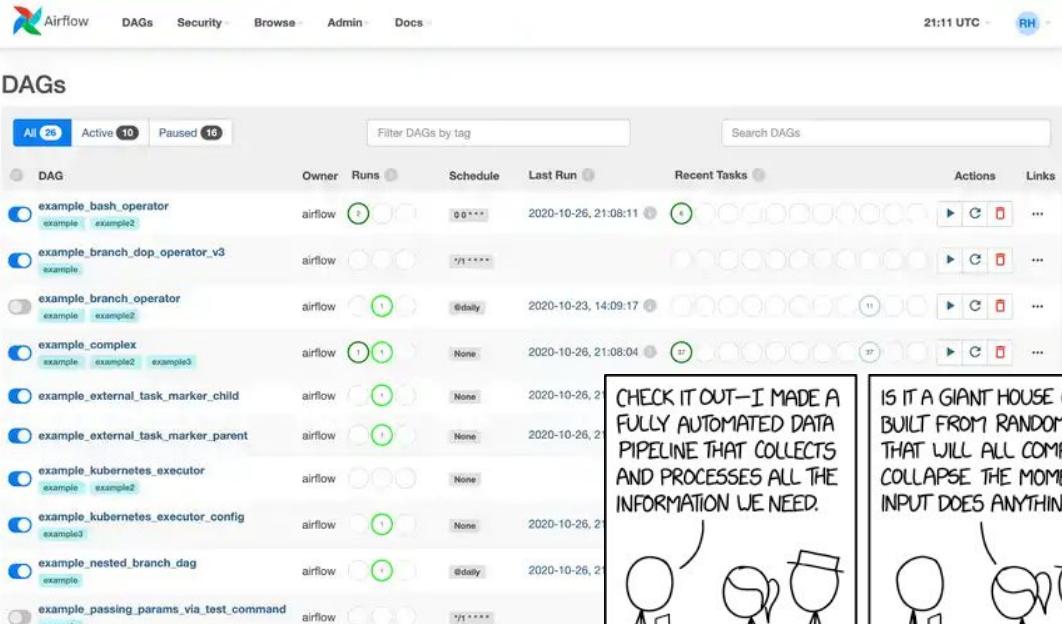


The screenshot shows the Airflow web interface with the following details:

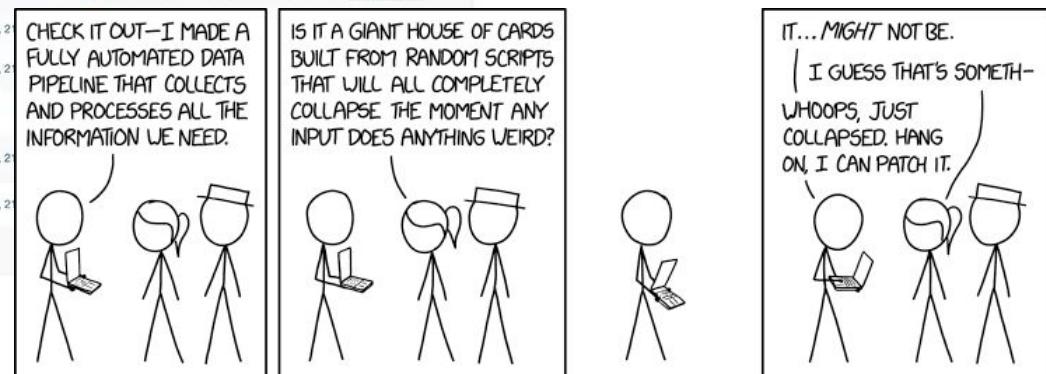
- DAGs Overview:** Shows 26 total DAGs, 10 active, and 16 paused. A search bar and filter by tag are available.
- DAG List:** A table listing 13 example DAGs, each with columns for Owner (airflow), Runs (green circle icon), Schedule (e.g., @daily, None, specific dates), Last Run (date/time), Recent Tasks (green circles), Actions (dropdown menu), and Links (dropdown menu).
- Detailed DAG Visualization:** A large diagram on the right illustrates a complex DAG structure. It starts with a green box labeled "start". Arrows point from "start" to "section-1-task-3" and "some-other-task". From "section-1-task-3", arrows point to "section-1-task-1", "section-1-task-2", "section-1-task-4", and "section-1-task-5". From "some-other-task", arrows point to "section-2-task-1", "section-2-task-2", "section-2-task-3", and "section-2-task-4". Finally, arrows point from "section-2-task-3" and "section-2-task-4" to a green box labeled "end".

<https://godatadriven.com/blog/apache-airflow-tutorial-for-data-pipelines/>

Example: Pipeline management

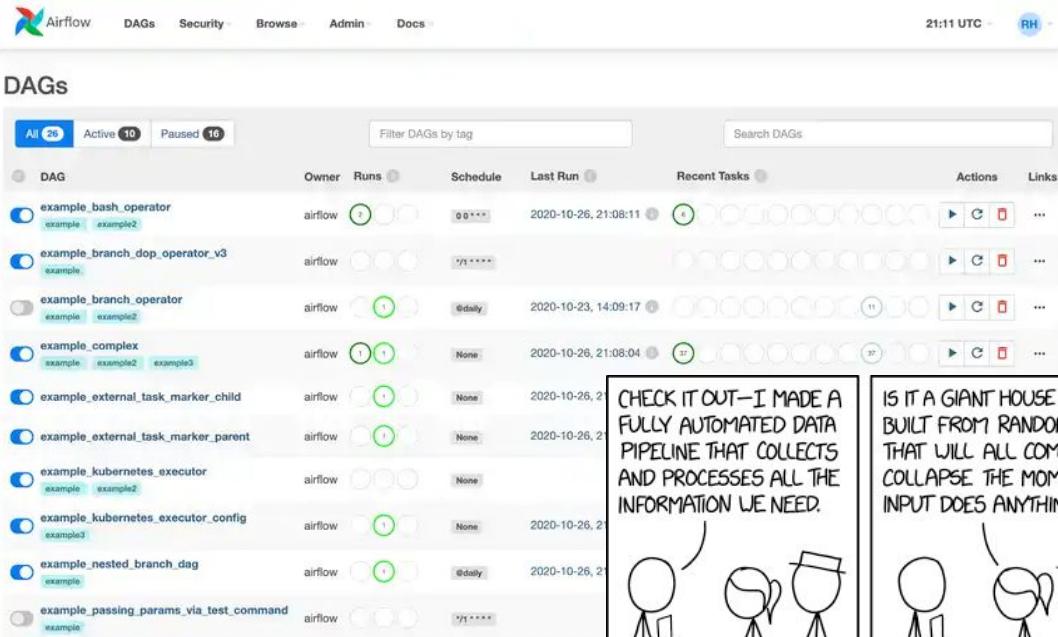


DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links	
DAG	airflow	1	0 0 * * *	2020-10-26, 21:08:11	1			
example_bash_operator	airflow	1	*/1 * * * *		1			
example_branch_dop_operator_v3	airflow	0	@daily		0			
example_branch_operator	airflow	1	@daily	2020-10-23, 14:09:17	11			
example_complex	airflow	1	None	2020-10-26, 21:08:04	27			
example_external_task_marker_child	airflow	0	None	2020-10-26, 21:08:04	0			
example_external_task_marker_parent	airflow	1	None	2020-10-26, 21:08:04	0			
example_kubernetes_executor	airflow	0	None		0			
example_kubernetes_executor_config	airflow	0	None		0			
example_nested_branch_dag	airflow	1	@daily	2020-10-26, 21:08:04	0			
example_passing_params_via_test_command	airflow	0	*/1 * * * *		0			

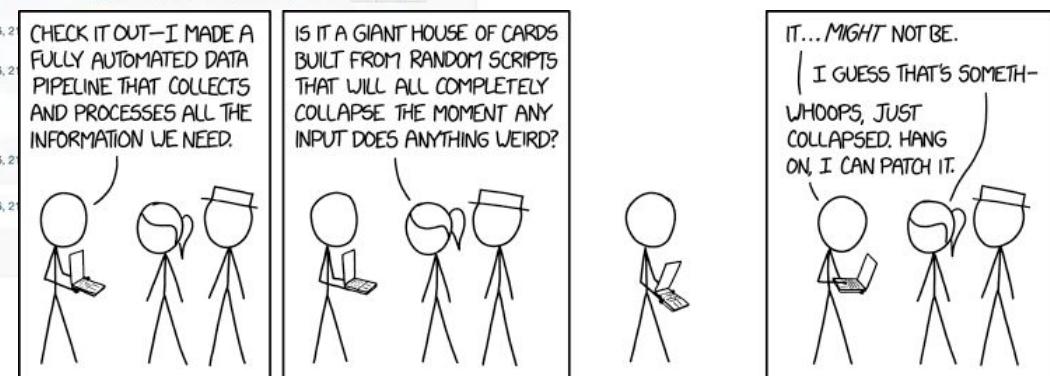


<https://xkcd.com/2054/>

Example: Pipeline management



DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
DAG							
example_bash_operator	airflow	10	0 * * * *	2020-10-26, 21:08:11	10	▶ C ⚡	...
example_branch_dop_operator_v3	airflow	0	*/* * * * *		0	▶ C ⚡	...
example_branch_operator	airflow	1	@daily	2020-10-23, 14:09:17	1	▶ C ⚡	...
example_complex	airflow	1	None	2020-10-26, 21:08:04	1	▶ C ⚡	...
example_external_task_marker_child	airflow	0	None	2020-10-26, 21:08:04	0	▶ C ⚡	...
example_external_task_marker_parent	airflow	0	None	2020-10-26, 21:08:04	0	▶ C ⚡	...
example_kubernetes_executor	airflow	0	None		0	▶ C ⚡	...
example_kubernetes_executor_config	airflow	0	None	2020-10-26, 21:08:04	0	▶ C ⚡	...
example_nested_branch_dag	airflow	1	@daily	2020-10-26, 21:08:04	1	▶ C ⚡	...
example_passing_params_via_test_command	airflow	0	*/* * * * *		0	▶ C ⚡	...



<https://xkcd.com/2054/>



Current tool status

- There's a lot of tools, and it's a fast changing world
- The maturity level of most tools is still quite low
- There's a big tendency for some of these tools to try to solve everything



Maturity Framework

0: No MLOps

The teams are disparate and releases are painful

Manual builds, deployments, testing, training

1: DevOps but no MLOps

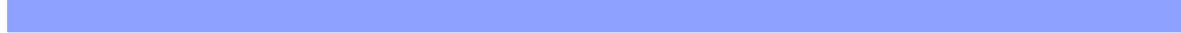
Releases are less painful
Difficult to trace/reproduce results

Automated builds and tests for application

2: Automated Training

Training environment is fully managed/traceable
Easy to reproduce model

Automated model training
Centralized tracking of model performance

3: Automated Model Deployment

Full traceability from deployment back to data
Environment managed: train > test > production

Automated tests for all code
Data provenance

4: Fully Automated Operations

Full system automated and easily monitored
Production systems providing information and automatically improving models

Automated model training and testing
Verbose, centralized metrics



Above all, it's about collaboration

While many of the DevOps patterns shown in this book require automation, **DevOps also requires cultural norms** and an architecture that allows for the shared goals to be achieved throughout the IT value stream. This **goes far beyond just automation**.

The DevOps Handbook

For MLOps to learn from DevOps, **we must center the needs of data scientists** and the people that are impacted by their models first. [...] When looking for a solution to automate, consider if you're only reducing the work required for manual processes or if you're also **enabling data scientists to focus on the hard problems they're trained to tackle**. [\[1\]](#)



What does success look like?

- 🚀 Time to value of new models and projects
- 🚀 Costs of training infrastructure
- 🚀 Efficiency of data scientists/machine learning engineers



Questions?

