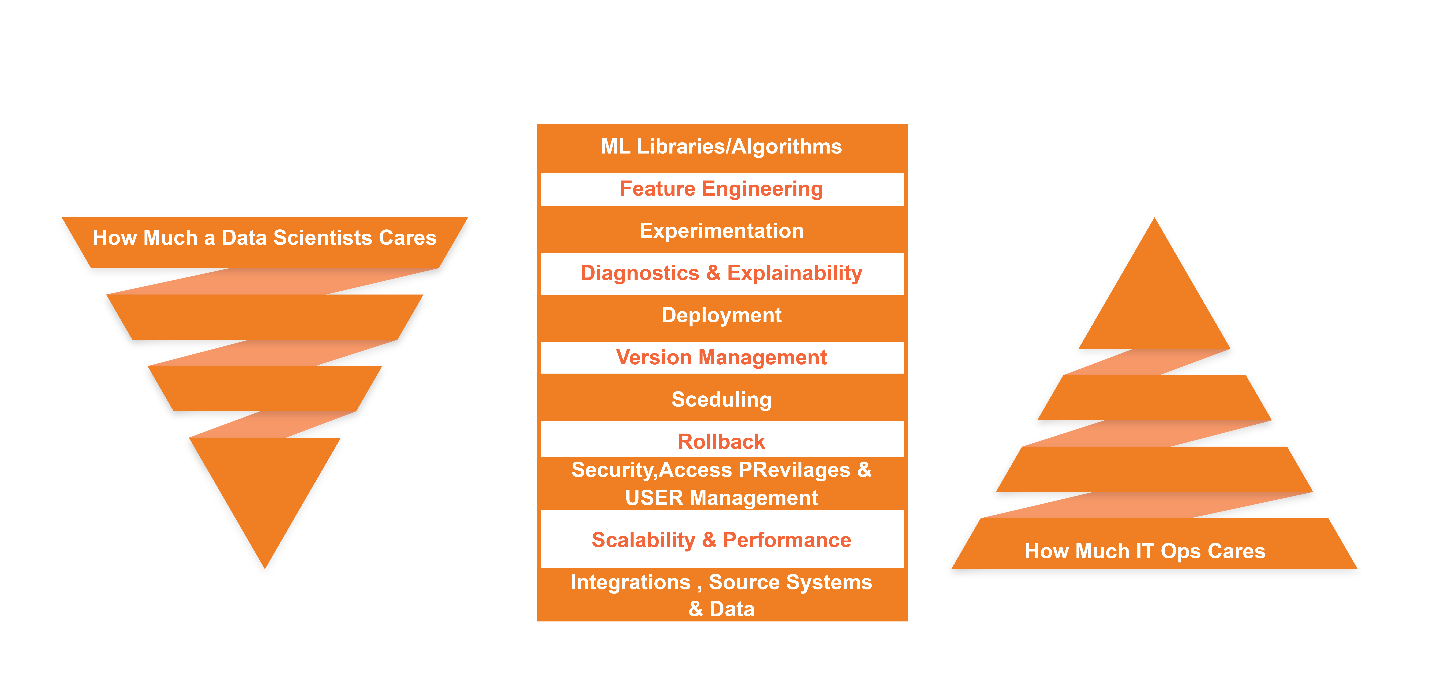
[MLOps – The Rise of MLo16n (ML Operationalization)](https://www.tredence.com/blog/mlops-the-rise-of-mlo16n-ml-operationalization/)

Let me start this post with a provocative statement: More machine learning models than ever, but are they making it into Production? Or more provocative even: once the ML models make it to production, are the IT Support teams equipped to run it, maintain it, manage it and above all safeguard the investments made on these breakaway enterprise assets?

The above provocation stems from observing trends that clearly highlights the amount of innovation and new products crowding the ML development space in contrast to only a few focused on how to operationalize these ML models in production. Why So?

Getting an ML model to deliver exciting results in a controlled environment (sandboxes, PoCs, Pilots, etc) is one thing. To deliver sustainable business value for the enterprise and that too at scale is another thing. The ML model has to be deployed in production, monitored, managed and continuously improved upon, this is where rubber hits the road and all or nothing game begins.

I don’t think a lot of data scientists have really understood the sheer implications of what all goes into making their models production ready. “Prediction” is not just about .predict(), it also means how things will work at scale, how well it is packaged to be deployed, how well it will be monitored and updated, etc. Without a systematic approach and skills needed to make the models production ready, the ever growing data scientists community (including the citizen data scientists) are stuck in their notebooks. Models live there as PoCs, performance gets reported with power point narrations and other visualization tools, and the ML lifecycle stays broken and incomplete.



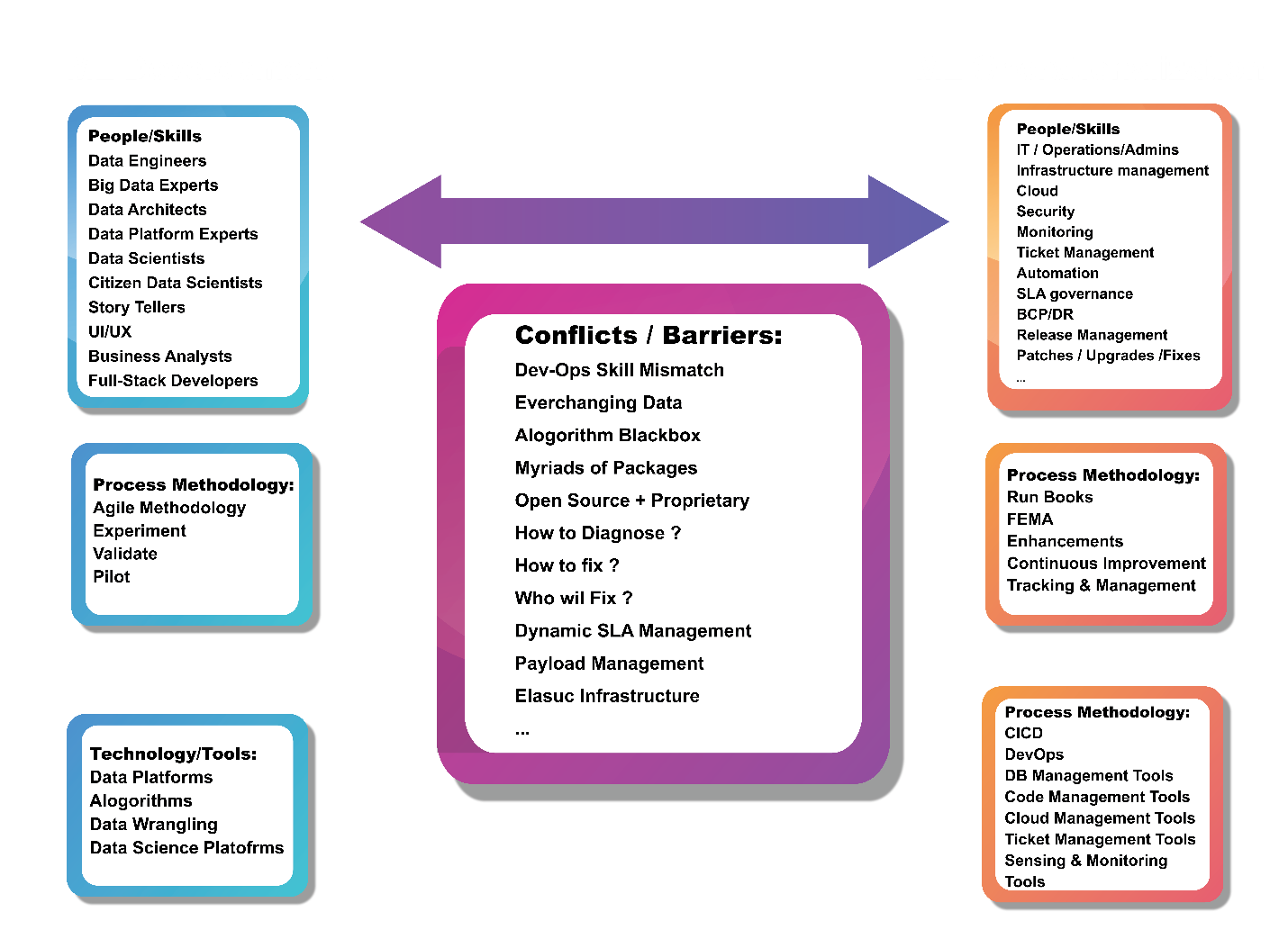
Deploying, managing, and optimizing ML in production means being prepared for additional challenges:

* **ML Application = ML Code + Data:** The ML application that you end up putting in production, is created by applying an algorithm to a mass of training data. Crucially, the algorithm’s output also depends on the input data that it receives at prediction time, which you can’t know in advance. While ML code can be carefully crafted in a controlled development environment, data in the real world never stops changing, and you can’t control how it will change.
* **Entropy at Play :** Entropy means lack of order or predictability leading to gradual decline into disorder. That’s precisely why the operations world literally scoffs at anything new that comes into production. For the IT operations and support teams, scalability, maintainability, traceability, resilience, etc are far more important than what novel problem the new piece of code is solving.
* **Diagnostics Challenges:** Unlike other systems and applications, ML algorithms are predictive models hence the outcomes do not always exhibit consistently “correct” results, which only a data scientist can appreciate. For the operations and support teams, inconsistent results means there are some serious flaws in the software.

The Tipping Point

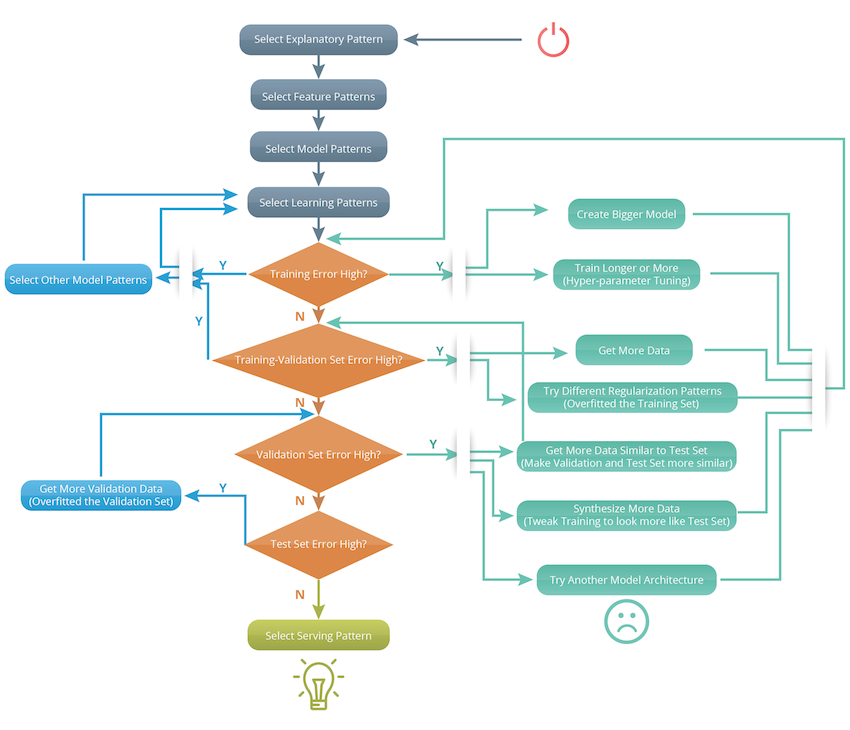
What we have seen so far is the explosion of model development activities in the data science development world; these models might have shown promising business returns in a controlled environment, however till these models are deployed into production and starts serving the enterprise at scale, all RoI promises are still in the air. Now, as the number of data scientists increases, as democratization of data and AutoML tools improve data science productivity, and as compute power grows making it easier to test new algorithms in development platforms, more and more models will continue to get developed, subsequently each one awaiting the move into production.

In data science projects, the derivation of business value follows the Pareto Principle, where the vast majority of the business value is generated only when you operationalize the project, till then the ML models have no tangible business benefits.



Typically, to get the model deployed, coordination is required across individuals or teams, where the responsibility for model development and model deployment lies with different people.

Let’s take a closer look at the ML model development life cycle.



After all these iterations and by the time your model is finally deployed, you face the next set of significant challenges: tracking and monitoring its performance over time. Don’t get surprised that by the time your model hits production running, it is outdated because the data demographics has changed. What options you have? You re-train your model with a newer slice of data and then do a fresh deployment. The problem doesn’t stop there, now the question is which model versions are in play in production, which features were used, what are the expected outcomes, etc; all these become an ever more demanding task – and this is just for one model. Now imagine if you have tens or hundreds?

What constitutes MLOps?

ML Ops is a set of practices, frameworks and tools that combines Machine Learning, DevOps and Data Engineering to deploy and maintain ML models in production reliably and efficiently.

Let’s now see what this actually means by delving deeper into the individual components that can be used to achieve ML Ops’ objectives.

* **The 360\* View:** A user interface that provides real time visibility into what is deployed and where, with deployment status, infrastructure health, and monitoring available at a glance.
* **Versioning, Monitoring, and Auditing:** A rich metadata repository to record and maintain lineage and traceability including timeline of model deployment: when a model was deployed, what are the associated data pipelines, what are the features, who deployed it, what are the access rights and privileges, what are the SLAs and what are the infrastructure usage and performance expectations.
* **Deployment Tools and Automation:** Set of tools to: quickly deploy or rollback a version of a model in response to changing data or business needs, infrastructure agnostic to enable deployments on-premises or to the cloud with Docker and Kubernetes so that the essential complexity of packaging, configuration and parameters tuning are abstracted from development and IT Ops teams.
* **End-to-End Integrated Workflows:** Interfaces and workflows to seamlessly integrate with data ingestion pipelines, data preparation and cleansing modules, model development, training and validation including a/b testing. A complete orchestrated view where final deployments can be traced right back to the initial datasets and processes and actions that took place within the development life cycle.
* **Cross Functional Teams:** Productionizing an ML model requires a set of skills that so far were considered separate. In order to be successful we need a hybrid team that, together, covers a range of skills: Data Scientist or ML Engineer, a DevOps Engineer and a Data Engineer.

MLOps is a brand new discipline, it will evolve over time, for sure. In terms of best practices, we can leverage and extend few of the adjacent disciplines to further strengthen MLOps best practices.

The following table takes a holistic view of DevOps, Data Engineering and ML Engineering as they seem applicable to MLOps:

|  |  |  |  |
| --- | --- | --- | --- |
| MLOps Practices | DevOps Practices | Data Engineering Practices | ML Engineering Practices |
| Pipeline | APIs, Integrations | Data Ingestion Pipelines, ETL | Training ML Pipelines, Test ML Pipelines, APIs, Workflows |
| Parameters | Rule Engines | Transformation Repositories | Feature Engineering Repositories |
| Versioning | Code Version Control | Jobs Version Controls, Data Lineage, Metadata Repository | ML Code Version Control + Model Version Control |
| Output Validation | Test Cases | Test Cases | Model Validation |
| CI/CD | Deploys Code to Production | Deploys Code to Production | Deploys Code to Production + All the related pipelines and Dataset versioning |
| Data Validation | NA | Business Validation | Statistical Validation |
| Monitoring | SLA Based | SLA Based | SLA + Data Drift + Temporal monitoring + parameters performance monitoring |

Serverless technologies allow us to write code and specification which automagically translates itself to auto-scaling production workloads. Here is an interesting take on how with open-source technologies (MLRun+Nuclio+KubeFlow), you can take advantage of serverless functions and make your Data Science projects ready for real-time, extreme scale data-analytics and machine learning.

In Summary

As data science teams grow and businesses increase the ML models adoption, the challenges that organizations will face in their operationalization will also increase. MLOps solves many of these challenges by optimising time to production, increasing collaboration and reducing cross-functional dependencies, improve reliability and reducing error and providing, at all times, visibility and traceability.

At Tredence, we have formed a cross functional practice for MLOps bringing in multiple disciplines (DevOps, Data Engineering, ML Engineering, Automation, Cloud and Infrastructure Management) to develop MLOps offerings for our clients.

Author: [**Soumendra Mohanty**](https://in.linkedin.com/in/soumendra-mohanty-43756448), *EVP – Analytics*