MLOPS CHECKLIST FOR MODEL DEPLOYMENT

Deploying machine learning (ML) models is a task that seems simple to some and daunting to others. The reality is that mistakes made when building ML applications can lead to serious headaches and wasted effort – that's why machine learning has been called "the high-interest credit card of technical debt."

Whether you're deploying just a few ML models or hundreds of them, these MLOps principles will help you deploy models with agility and reduce the operational burdens of ML. With this checklist, we aim to:

- Identify the most important MLOps practices and framework components
- Explain why each of these elements is important
- Help prioritize effort and infrastructure that will streamline your current or future ML deployments

---

☐ Store models in a registry

- A model registry is a repository for storing and tracking the versions of models, analogous to a version control system like Git, but for ML
- Model registries enable:
  - Data scientists to track their work and perform reproducible research
  - ML engineers to know they're deploying the right version of the right model
  - Organizations to audit ML predictions for compliance and accuracy

☐ Deduplicate effort with a feature store

- A feature store is not always necessary, but helpful when developing many ML models based on the same type of data, such as users, customers, or products
- Feature stores execute data transformations and centralize features into a common location for serving to training and scoring workflows
- A feature store will let data scientists reuse complex features across multiple models and let ML engineers deploy models without rewriting data-transformation logic.

☐ Test code and validate assumptions

- All ML models are built using assumptions about the training data, and many will use custom code
- Assumptions about data should be validated whenever a new dataset is used to train a model
- Custom code should have good unit test coverage and tests should be executed every time the code is updated

☐ Use CI/CD to automate ML deployments

- Manual deployments require increased effort from development teams and increase the risk of human errors
- ML Deployments should be automated using continuous integration and continuous delivery (CI/CD) pipelines
- Automating builds makes sure that updating or releasing models will be business decision rather than a technical challenge

☐ Automate model retraining

- Virtually all ML models will require retraining at some point, some more often than others
- As data scientists develop more and more models, retraining manually could actually come to dominate their time; it also increases the chance of errors
- Model retraining should always be addressed and automated when building ML applications

☐ Capture data and predictions from production

- Many ML deployment examples will generate predictions from a trained ML model and serve up the result without recording any data data
- Capturing input data (features) and output predictions is vital in order to:
  - Evaluate model performance in production
  - Retrain models using new data
  - Iterate on existing models and develop better ones

☐ Monitor for drift

- Deployed ML models generate predictions based on new data that could evolve over time – this is called model drift
- All ML applications should monitor for model drift to reduce the risk of performance degradation

☐ Architect for reliability

- ML predictions can only generate business value when applications are up and running when needed
- ML applications should be architected with reliability in mind by prioritizing high availability and scalability by using serverless architectures or container platforms

☐ Evaluate models continually and iterate

- ML models are built on data, and new insights are commonly exposed
- Data scientists should investigate anomalies in model performance or predictions – use those learnings to develop better models