2023.02.17 ML Ops framework

**Ideal framework with features/modules (5):** Where does customization come in?

* Data processing: *Real time feature store & data exploration*
  + Feature store
  + Data exploration
  + Model training
  + ML metadata, & artifact repository: Data and pipeline lineage
  + Experimentation: Optimize models with hyperparameters
  + Model registry
* Model development & CT
  + Notebooks
  + Package configuration
  + Triggers
  + Visualizing model predictions and performing model evaluation with tables
* CI/CD deployment – model registry: Model lineage and metadata
  + Testing: Lint and Unit
  + Triggers
* Serving/Orchestration: Human AI champion-challenge
  + Model Serving
  + Online experimentation
  + Retrieve models from registry and Orchestrate on another platform
* Monitoring: Drift, observability, & explainability
  + Model monitoring
  + Metrics dashboard
  + Automated retraining triggers
  + Collaboration: Commenting, shared reports, & versioning
* Infrastructure: Vector matching engine – scalability – Compute and storage
  + AWS, Azure, and Google

**Knowledge repo**

* Business
  + How to convert business use cases into measurable ML objectives?
  + Which option to choose from? Proprietary, cloud solutions, or private solutions?
  + How do the cost breakdown for each of these options?
  + What are the different evaluation criteria/features one should be looking into?
  + Who are the different stakeholders and what is needed from each one?
* Data processing
  + How to create an ideal feature store? Is there a particular schema?
  + Why need as model store? Unify data processing for both offline processes such as model training/evaluation to online processes such as model inference
  + Is experimentation part of this phase as well. If so, what is the difference?
  + How to avoid data leakages?
  + Where does data and model management come into play? Is the model registry the same?
* Models build CT
  + How to evaluate models during development (offline) and in production (online/live)?
  + Continual training to accommodate data drifts
* CI/CD
  + How does model registry tie in out here? Is it at the first stage or only after one deploys it onto the model serving server?
  + How to apply Kubeflow demo? [Machine learning operations with GitHub Actions and Kubernetes - GitHub Universe 2019 - YouTube](https://www.youtube.com/watch?v=Ll50l3fsoYs)
  + Model registry (Lineage, model management) is needed for implementing criteria to promote models from development to staging and then to production.
* Model serving
  + How to build an efficient and effective experimentation engine? What is the data schema required?
  + How do I serve the model? Batch, online with batch features, online real-time, & online near real-time
  + How to ensure a smooth user experience?
* Model monitoring
  + How to automate the data drift and concept drift with retraining triggers?
  + How to consider orchestration and reproducibility?
* Infrastructure
  + What is the initial configuration needed for your machine learning system?
  + How do we scale? Some canonical methods borrowed from software engineering principles include horizontal-vertical scaling, load balancing, localization, replication, & asynchrony – Applying this to batch and streaming data

Resources:

* Links
  + <https://www.wandb.courses/courses/effective-mlops-model-development>
  + Weights and biases integration with AWS: <https://aws.amazon.com/blogs/machine-learning/improve-ml-developer-productivity-with-weights-biases-a-computer-vision-example-on-amazon-sagemaker/>
  + Tracking experiments: <https://wandb.ai/iamleonie/Intro-to-MLOps/reports/Intro-to-MLOps-Machine-Learning-Experiment-Tracking--VmlldzozMDE4NzUw>
  + <https://stanford-cs329s.github.io/syllabus.html> (Stanford system design)
  + Serverless is a cloud-native development model that allows developers to build and run applications without having to manage servers.
  + Headless is “decoupling” the front end and back end, implementing a best-in-class API architecture.
  + <https://neptune.ai/blog/machine-learning-as-a-service-what-it-is-when-to-use-it-and-what-are-the-best-tools-out-there>
  + Source: https://neptune.ai/blog/end-to-end-mlops-platforms
  + Source: <https://neptune.ai/blog/learn-mlops-books-articles-podcasts>
  + MLOps stack: <https://neptune.ai/mlops-tool-stack>
  + Foundation papers: Google, databricks, azure, & aws
  + Survey: [[2209.09125] Operationalizing Machine Learning: An Interview Study (arxiv.org)](https://arxiv.org/abs/2209.09125)
  + Facilitate machine learning with GitHub: [Related Projects & Examples | Machine Learning Ops (githubapp.com)](https://mlops.githubapp.com/examples)
* Books
  + Introduction MLOps riley
  + What is MLOps
  + Practical MLOps
  + Reliable Machine Learning
  + Designing Machine Learning systems
  + Machine learning engineering
* Courses
  + Udemy: MLOps Azure
  + Udemy: MLOps Google
  + Udemy: MLOps AWS
  + Weights and biases: Effective ML Ops
  + Full stack deep learning
  + MLOps Udacity nanodegree
  + MLOps coursera Ng
  + Stanford: Designing ML systems
  + Sphere bootcamp
* Companies
  + Weights and biases
  + Abacus.io
  + Neptune.ai
  + DataRobot
  + Dataiku
  + H20
  + Iguazio
  + Algorithmia
  + Allegro.io
  + Cnvrg.io
  + Kubeflow
  + Pachyderm
  + Ployaxon
  + Valohai

Evaluation of providers

Table

Description automatically generated

**MLOps: Insights DevOps – Elasticsearch**

* Package management and provisioning: Packer, Terraform
* Containers/Orchestration: Docker, Kubernetes
* CI/CD: Jenkins, TravisCS, Circle CI
* Configuration management: Puppet, Chef, Ansible
* Scheduling: Luigi, Airflow, Azkaban
* Monitoring: Honeycomb, Prometheus, New Relic
* Log aggregation: Sumo Logic, LogDNA, & The ETL stack
* Experimentation and reliability engineering
* Infrastructure as code: Terraform
* Cloud computing: GCP, AWS, & Azure
* Model registry and metadata artifacts: Where does Kubeflow and MLflow fit in?
* Governance: Security, compliance, & model service catalogue

Weclouddata: <https://weclouddata.com/courses/online/ml-engineering-bootcamp/>

NYC data science academy: <https://nycdatascience.com/courses/designing-and-implementing-production-machine-learning-systems-mlops/>

* Kyle Gallatin: <https://medium.com/@kylegallatin>
* <https://www.pluralsight.com/paths/machine-learning-engineering>
* Go to guide to MLOps analytics Vidhya: <https://www.analyticsvidhya.com/blog/2022/04/your-go-to-guide-on-machine-learning-operations-mlops/?utm_source=related_WP&utm_medium=https://www.analyticsvidhya.com/blog/2021/04/bring-devops-to-data-science-with-continuous-mlops/>
* Bring DevOps to Data science: https://www.analyticsvidhya.com/blog/2021/04/bring-devops-to-data-science-with-continuous-mlops/

**Analytics Vidhya: Happyfeet123**

* https://www.analyticsvidhya.com/blog/2021/04/bring-devops-to-data-science-with-continuous-mlops/
* <https://www.analyticsvidhya.com/blog/2022/04/your-go-to-guide-on-machine-learning-operations-mlops/?utm_source=related_WP&utm_medium=https://www.analyticsvidhya.com/blog/2021/04/bring-devops-to-data-science-with-continuous-mlops/>
* How to become MLOps (Project Pro): <https://www.projectpro.io/article/how-to-become-an-mlops-engineer/478>
* Workfall roadmap: https://www.workfall.com/learning/blog/roadmap-to-become-a-successful-mlops-engineer/