Full Stack AI Projects

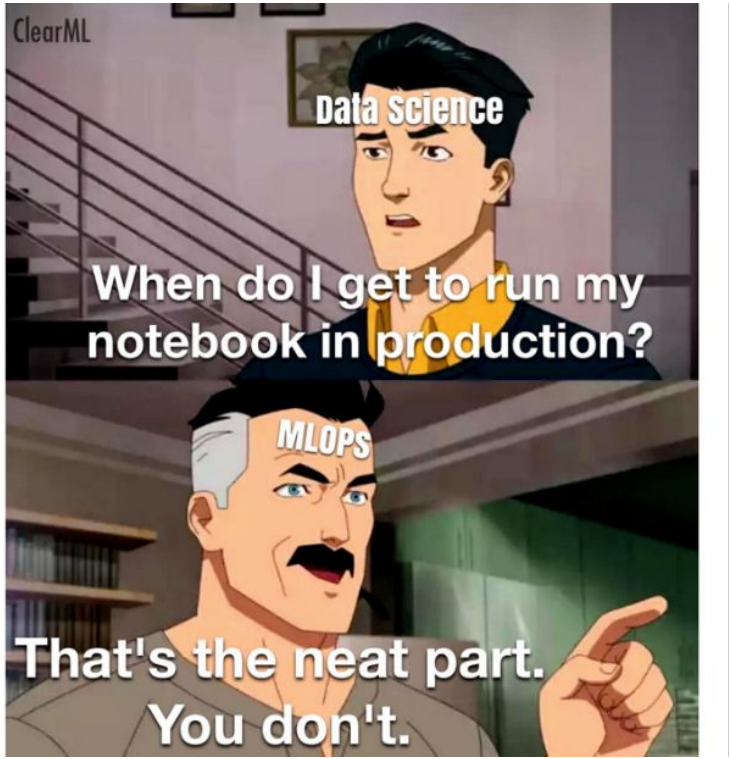
Infrastructure and tooling

Henry Ruiz
GDE ML
@devharuiz
https://haruiz.github.io/

What we have discussed so far?

What is machine learning system design?

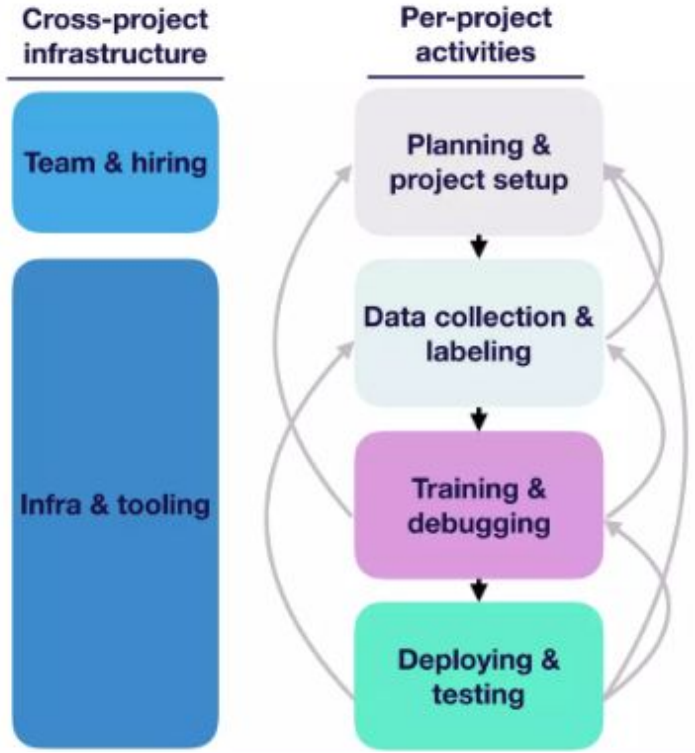
The process of **defining the <u>interface, algorithms, data, infrastructure, and hardware</u>** for a machine learning system to satisfy specified requirements.



What we have discussed so far?

So, What is required to design and build an ML system?

Like any other software solution, ML systems require a **well-structured methodology** to maximize the success rate of the implementation.



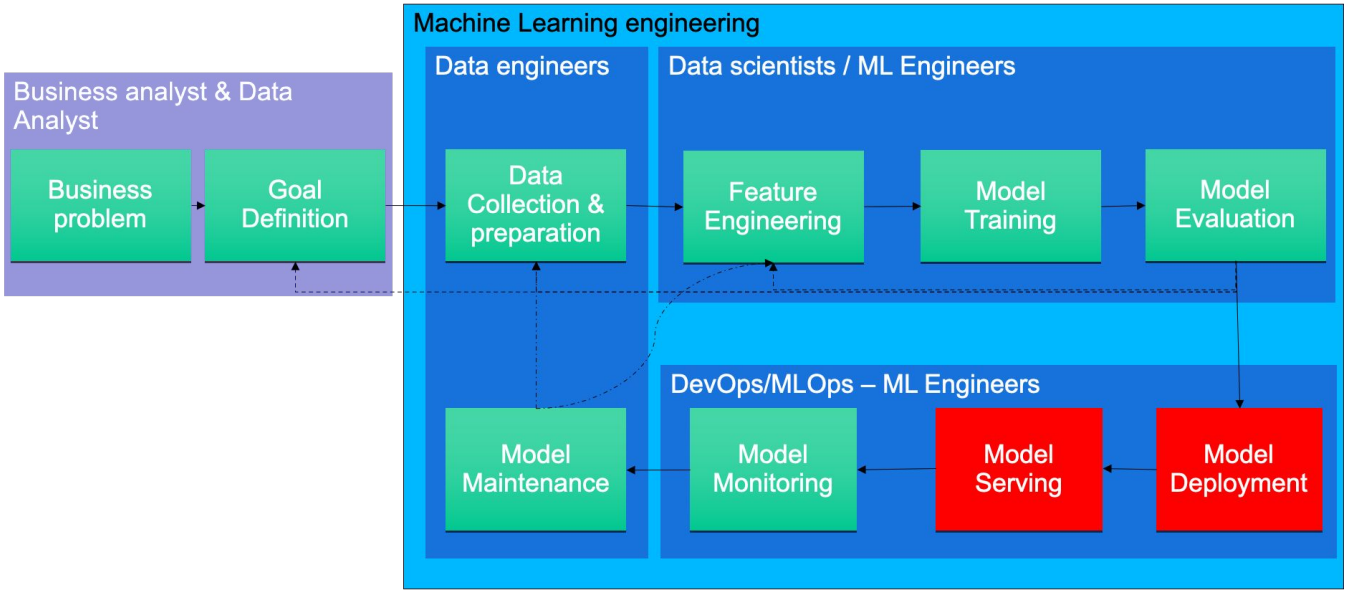
What we have discussed so far?

So, What is required to design and build an ML system?

Set up a team

Managing and leading ML and Data Science teams require unique skills.

ML teams have diverse roles.

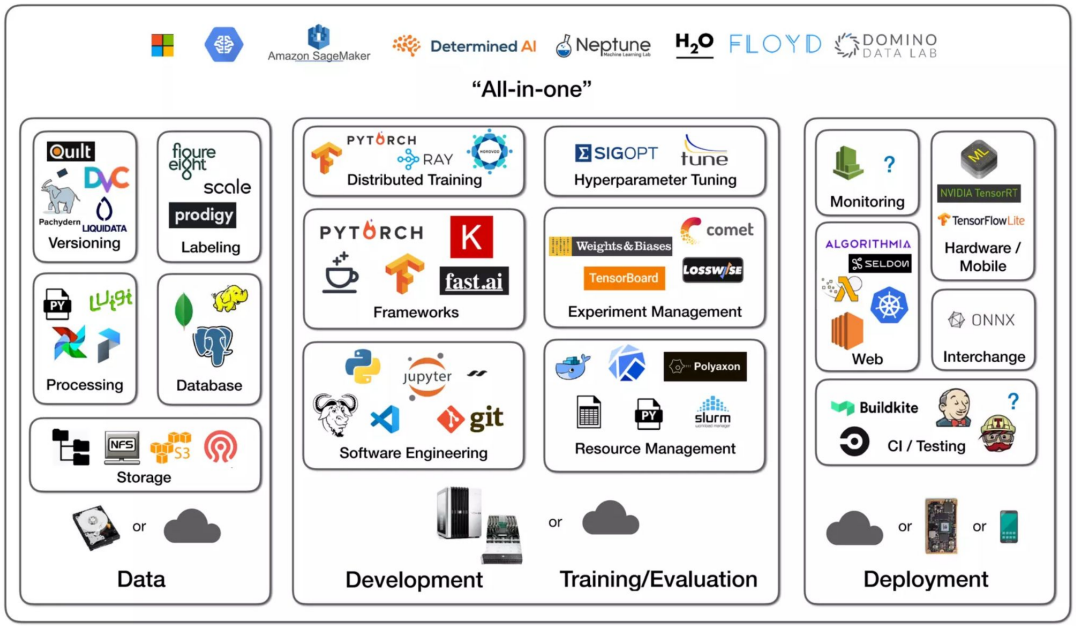


What we have discussed so far?

Define Development infrastructure & Tooling

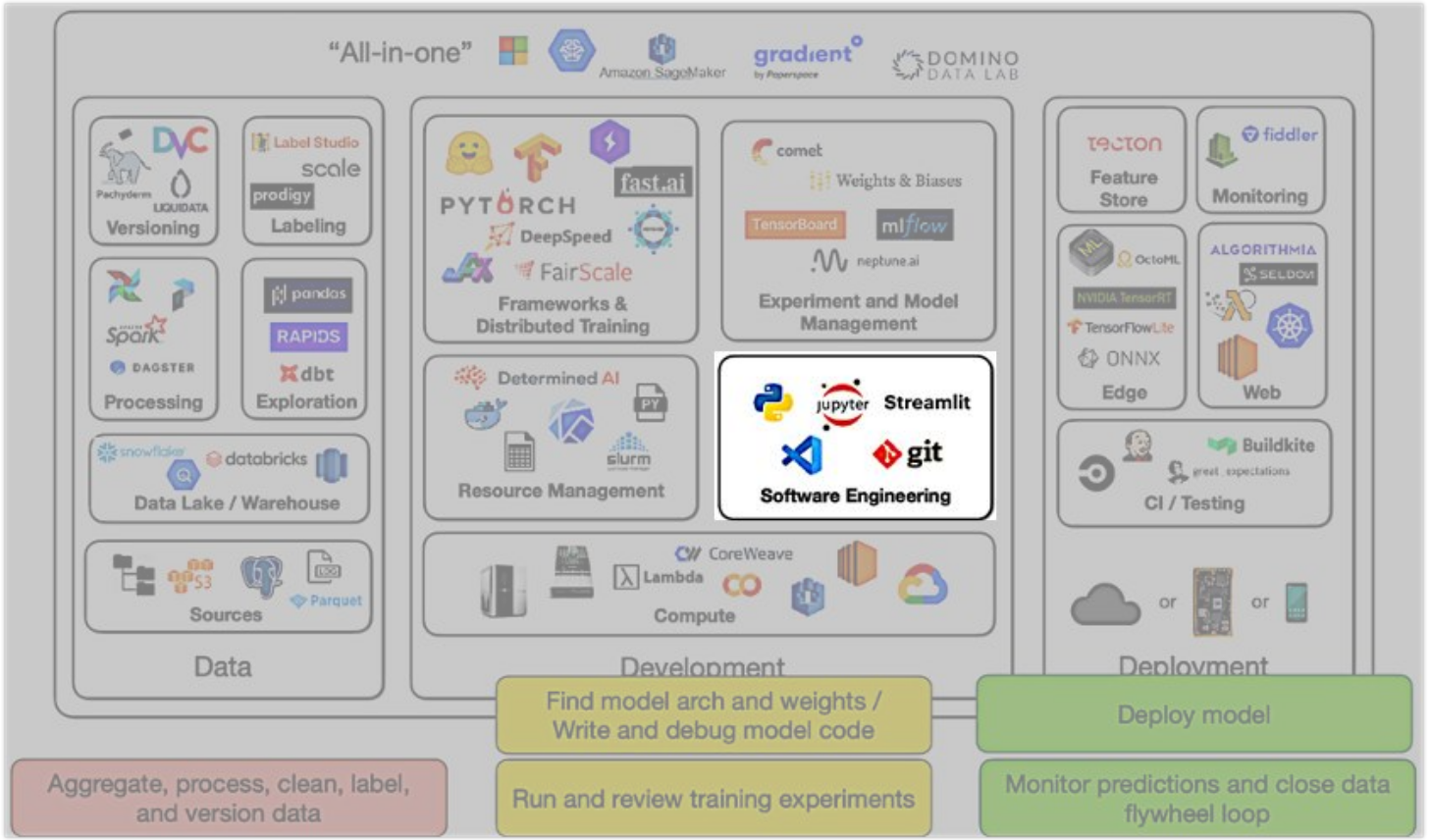
The number of **available tools to work with ML seems endless.**

Selecting the appropriate tools depends on:
the kind of problem, **type of solution, deployment scenario, capacity building**, team experience, cost, hardware and software infrastructure, etc.



Credits: https://fullstackdeeplearning.com/

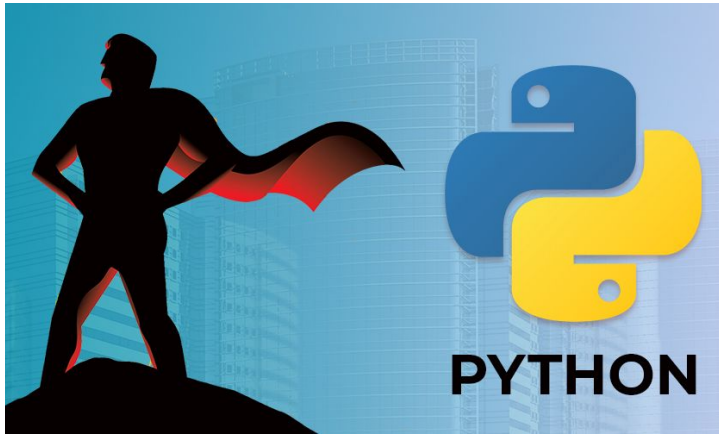
Software Engineering Tools



Software Engineering Tools

Language, why Python?

- **Ease of use:** Python is a high-level programming language that is easy to learn and use. Its syntax is simple and readable, making it accessible to beginners.
- **Large and active community:** Python has a large and active community of developers, which means that there are many resources and libraries available for data scientists to use.
- **Abundance of libraries and tools:** Python has a rich ecosystem of libraries and tools specifically designed for data science, including NumPy, Pandas, Scikit-learn, Matplotlib, TensorFlow, and PyTorch, to name a few.
- **Flexibility and versatility:** Python can be used for a wide range of data science tasks, including data cleaning, data analysis, data visualization, machine learning, and deep learning.
- **Interoperability:** Python is designed to be easily interoperable with other languages and tools, which means that data scientists can easily integrate it with other technologies and frameworks.
- **Open source and free:** Python is an open-source programming language, which means that it is free to use and distribute, making it accessible to everyone, regardless of budget.



Software Engineering Tools

Code editor

A software application that is specifically designed for writing and editing source code by offering features such as:

- **Syntax highlighting:** Code editors use syntax highlighting to color the keywords used by the programming language, making it easier to read and understand.
- **Auto-completion:** Many code editors provide auto-completion functionality that suggests code snippets, functions, and variable names based on the context.
- **Debugging support:** Code editors often provide debugging tools and features to help developers identify and fix errors and bugs in their code.
- **Integrated development environment (IDE) features:** Some code editors provide additional features that are typically found in full-fledged IDEs, such as build tools, version control, and project management.
- **Customization options:** Code editors are highly customizable, and users can typically customize the interface, keyboard shortcuts, and other settings to maximize the productivity of the coder.
- **Support for multiple languages:** Code editors typically support multiple programming languages, making them versatile and useful for a wide range of projects.
-

Software Engineering Tools

Software Versioning

web-based platform that provides hosting for software development and version control using Git

- **Git repository hosting:** GitHub provides a platform for hosting Git repositories, allowing developers to collaborate on code and track changes over time.
- **Issue tracking:** GitHub includes a robust issue tracking system that allows developers to create and manage tasks, bugs, and feature requests.
- **Pull requests:** GitHub provides a streamlined process for making changes to a codebase and submitting those changes for review through pull requests.
- **Collaboration tools:** GitHub includes a range of collaboration tools, such as wikis, project management tools, and team management features, to help teams work together more effectively.
- **Open source community:** GitHub has a large and active open source community, making it a great platform for contributing to open source projects and collaborating with other developers.
- **Integration with other tools:** GitHub integrates with a wide range of other tools and services, such as continuous integration and deployment tools, project management tools, and chat applications.
- **Code review tools:** GitHub includes a robust code review system that allows developers to provide feedback and comments on each other's code, helping to ensure code quality and maintainability.

packaging, distributing and managing your code dependencies

Transforming your code into a deployment unit

Isolating python dependencies using Pyenv and Poetry

Pyenv

- Pyenv is a simple **Python version management tool.**
- It allows users to easily switch between different versions of Python on the same system.
- Pyenv can be used to manage both system-wide and per-user installations of Python.
- It can be used to install different versions of Python, and to set the global and local versions of Python for a specific project.

Poetry

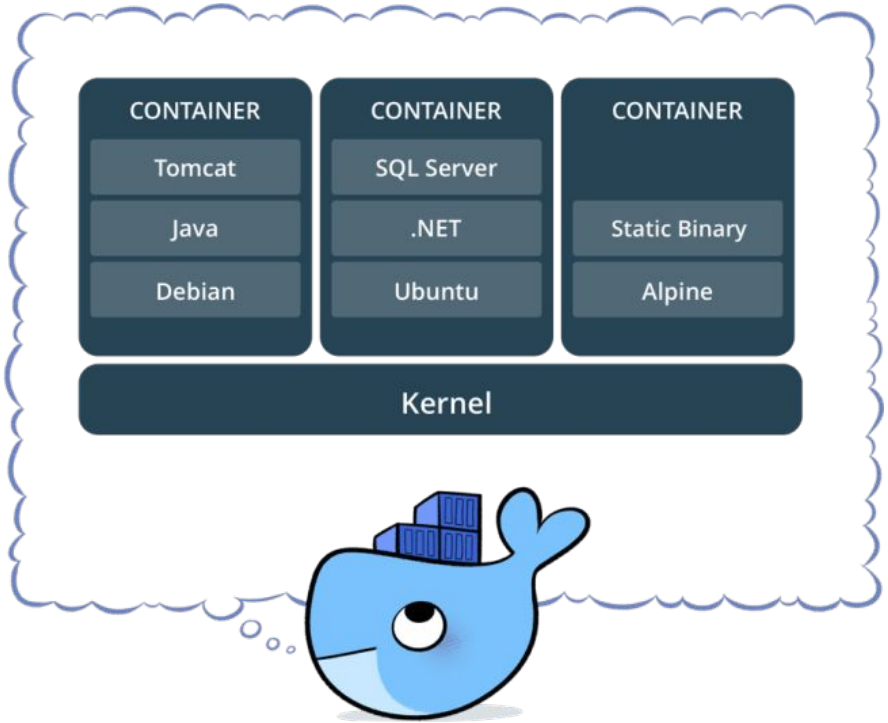
- Poetry is a **package and dependency manager for Python.**
- It provides a simple and convenient way to manage dependencies, packages, and virtual environments for a Python project.
- Poetry can be used to create new projects, to add and manage dependencies, and to build and publish packages.
- It provides a consistent and repeatable way to manage dependencies, making it easier to test and deploy Python projects.
- Poetry uses a pyproject.toml file to manage the dependencies and configuration of a Python project, and it integrates with Pyenv to provide an easy way to manage multiple Python versions and virtual environments.

Isolating App dependencies using Docker

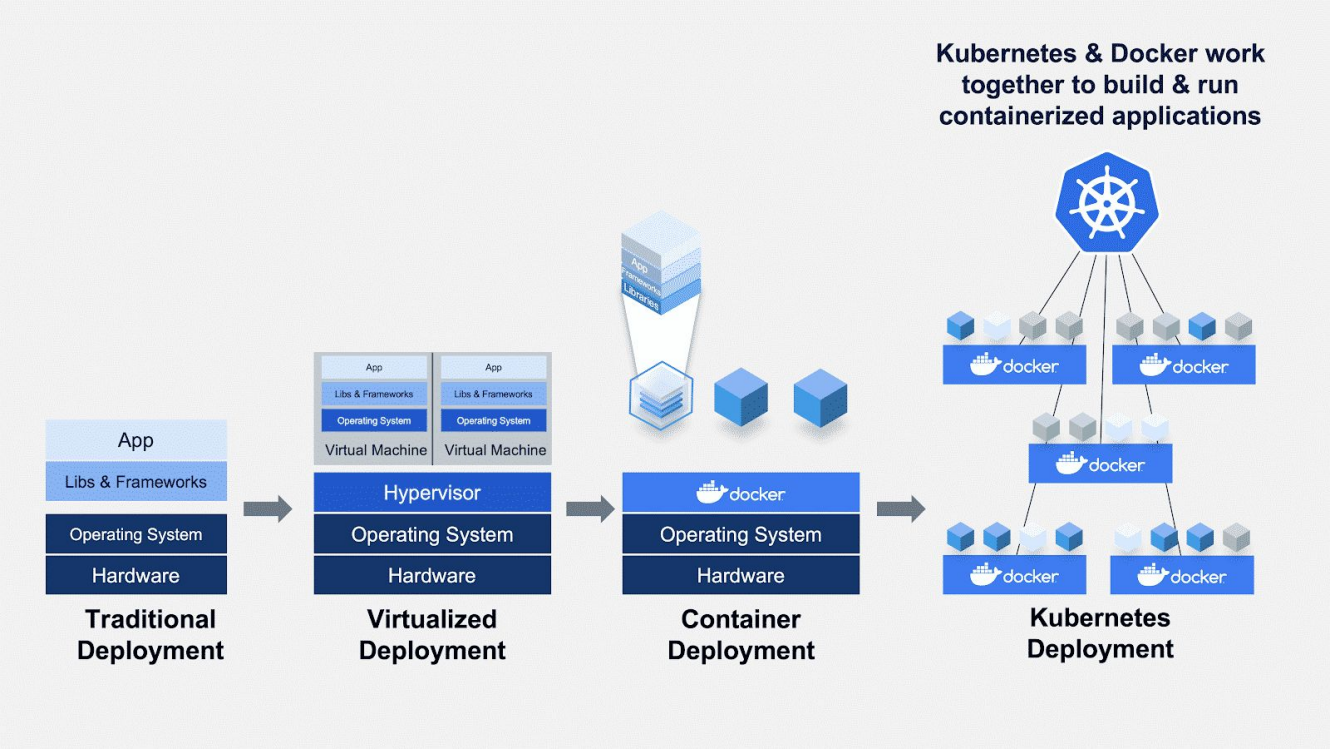
Docker is a platform for building, shipping, and running distributed applications.

It uses containers to package software into isolated environments that can run on any system with the Docker engine installed.

This makes it easy to run the same application in different environments, such as development, testing, and production, without worrying about dependencies, configurations, or compatibility issues.



orchestrating containerized applications



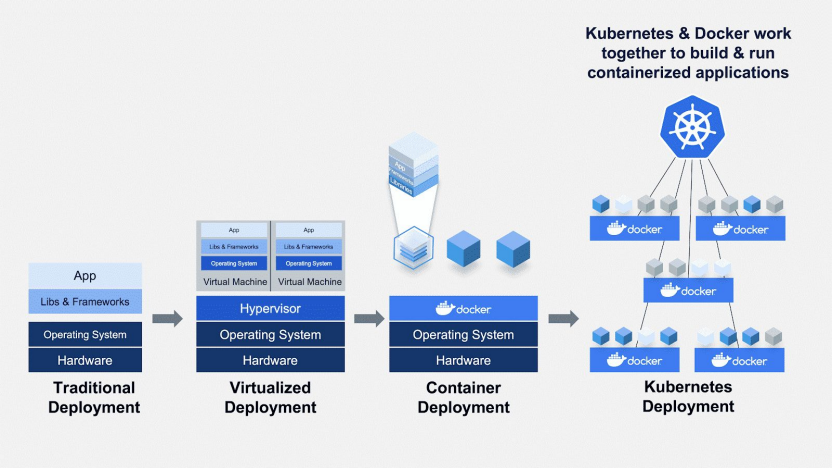
https://testingclouds.wordpress.com/2021/03/09/migrating-from-docker-compose-to-skaffold/

Docker compose vs Kubernetes

Docker Compose and Kubernetes are both popular tools for managing multi-container applications, **but they are designed for different use cases and have different architectures.**

Docker Compose is a tool for defining and running multi-container applications on a single host. It is easy to use and requires minimal setup, making it a popular choice for developers who want to quickly set up and test their applications locally. Docker Compose provides a simple and convenient way to define, configure, and run multi-container applications, but it is limited to a single host and does not provide the same level of scalability and resource management as Kubernetes.

Kubernetes, on the other hand, is a production-ready platform for deploying, scaling, and managing containerized applications. It provides a powerful and flexible architecture for managing multi-container applications at scale, and it includes features for automatic scaling, rolling updates, self-healing, and resource management. Kubernetes is designed for large, complex, and mission-critical applications, and it provides a high degree of availability and resilience.



kubernetes + docker

Experiment and Model Management

Experiment management refers to **tools and processes that help us keep track of code, model parameters, and data sets that are iterated on during the model development lifecycle**. Such tools are essential to effective model development. There are several solutions here:

- TensorBoard: A non-exclusive Google solution effective at one-off experiment tracking. It is difficult to manage many experiments.
- MLflow: A non-exclusive Databricks project that includes model packaging and more, in addition to experiment management. It must be self-hosted.
- Weights and Biases: An easy-to-use solution that is free for personal and academic projects! Logging starts simply with an "experiment config" command.
- Other options include Neptune AI, Comet ML, and Determined AI, all of which have solid experiment tracking options.