Automated model retraining

Milica Mladjenovic

Automated model retraining

- Model retraining refers to the process of updating a deployed machine learning model with new data
- This process can be done manually but it can also be automated

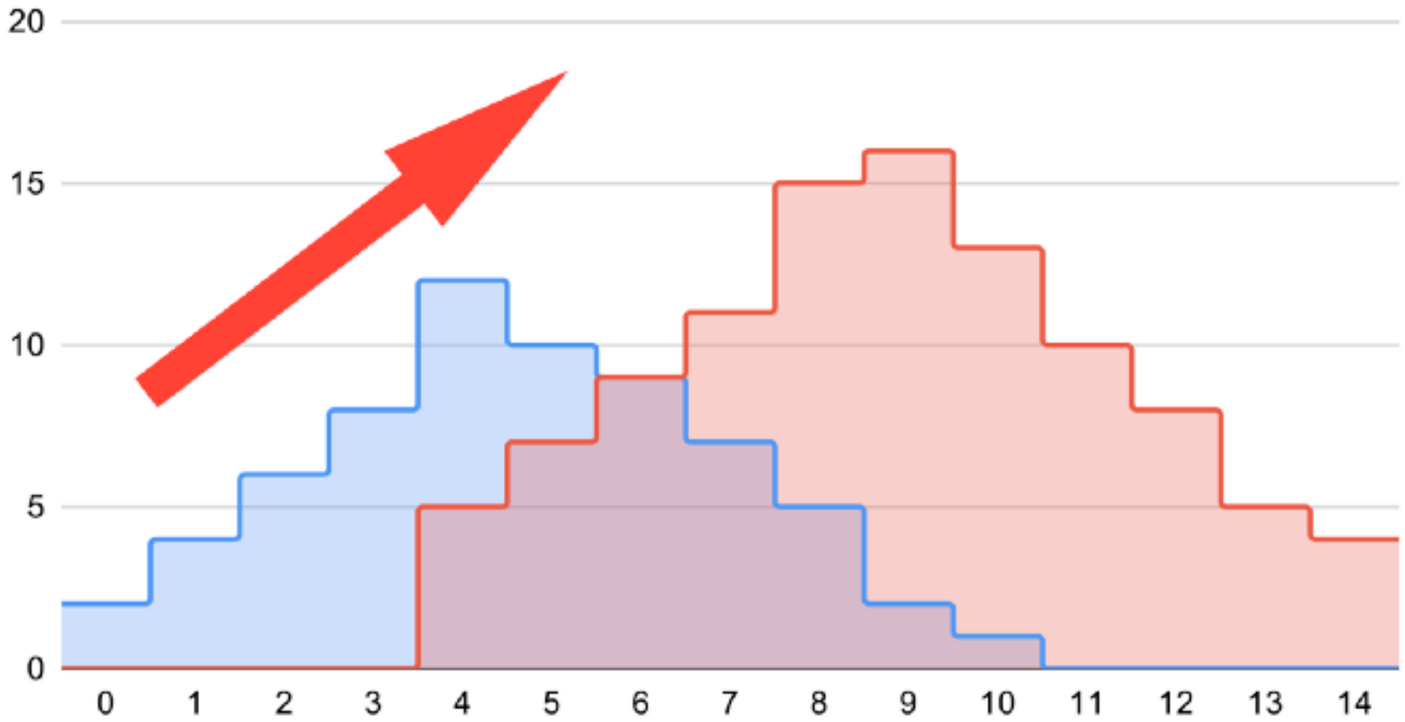
Why is model retraining necessary?

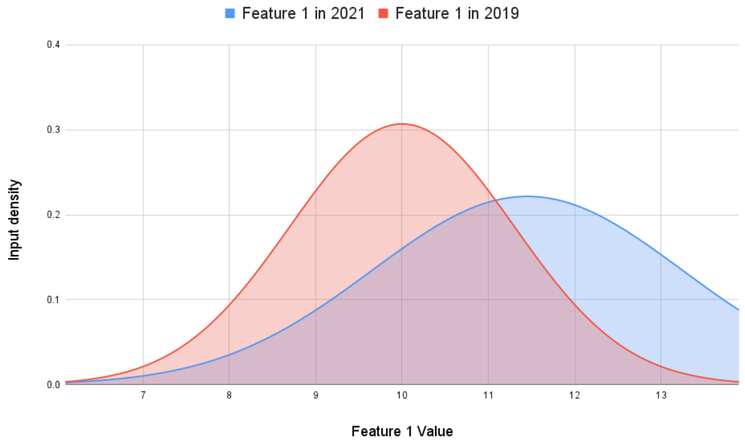
- Good models have a good generalisation error and in order for our model to have good performance, the training data needs to be a good representation of the outside world
- Over time, as the world changes, we will notice a degradation of the model performance over time, referred to as a model drift
- Retraining is required to present this drift and to ensure that the model will stay consistent with it’s performance

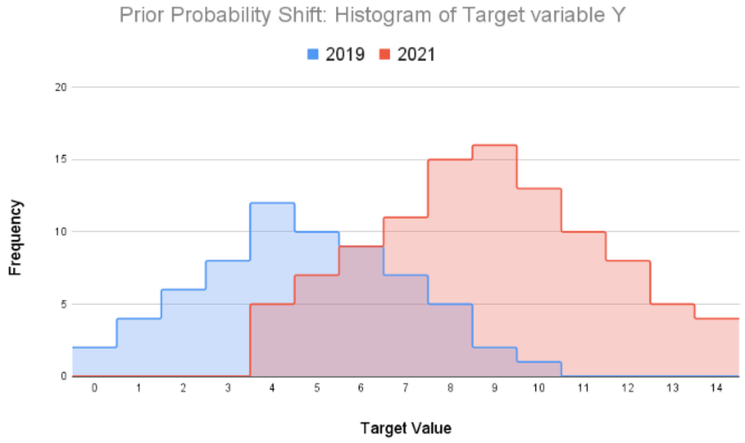
Model drift

- There are two types of model drift: concept drift and data drift
- Concept Drift: The link between the input variables and target variables changes over time - training on old data
- Data Drift: The characteristics of the input data changes

Covariate and prior probability shift



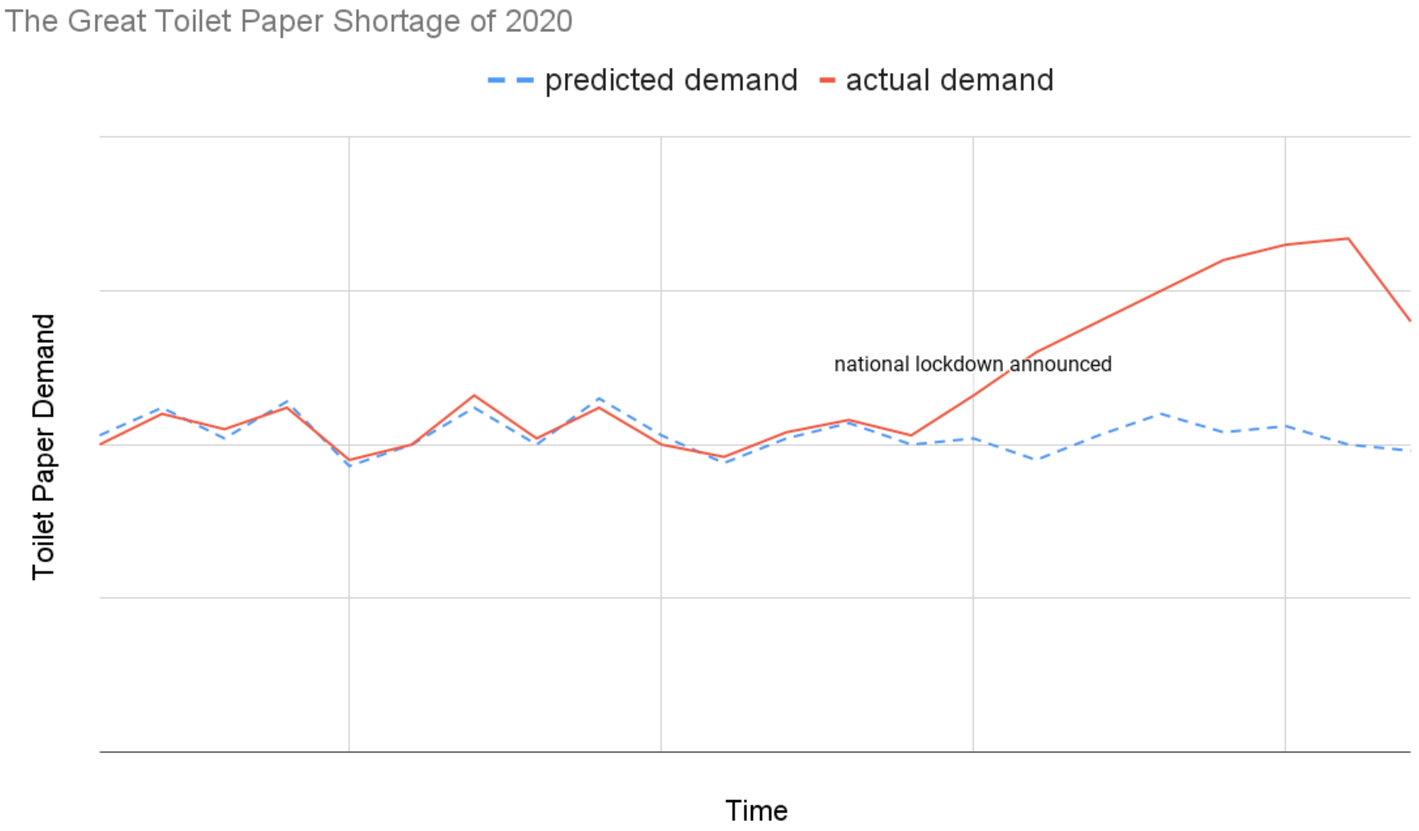




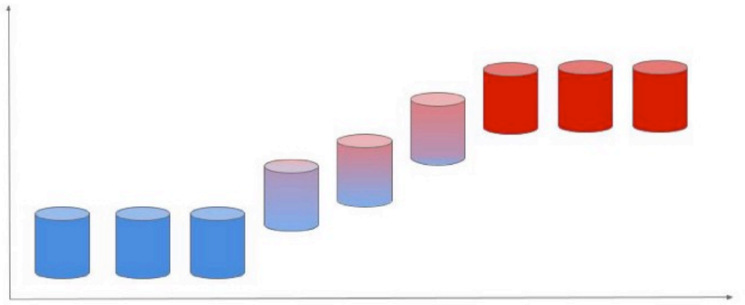
- Covariate shift- distribution of feature X has changed over time
- The model in production sees a different age demographic (the mean and variance are different)
- Prior probability shift - the distribution of the target variables Y change over time

Concept shift

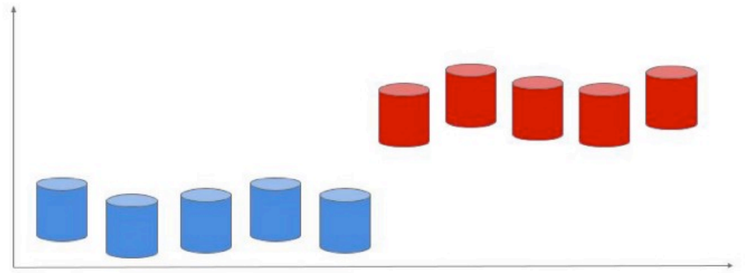
The Great Toilet Paper Shortage



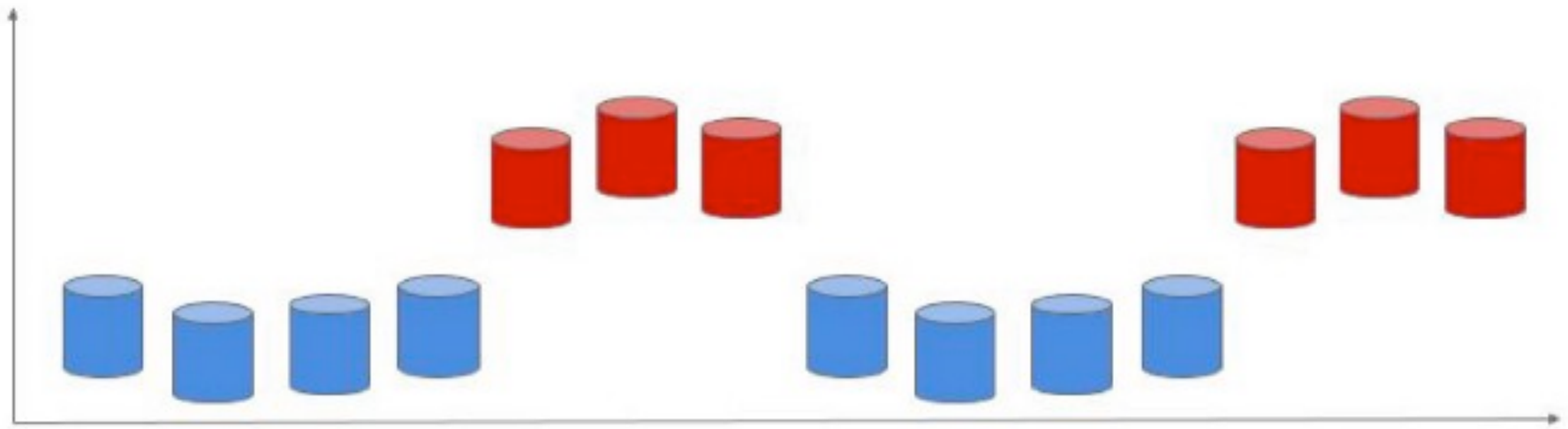
Gradual concept shift



Sudden concept shift



- Concept shift - the relationship between X and Y change
- Gradual shift - launch of alternative products, economic changes
- Sudden shift - major changes to road network, new equipment added to the production line



Reccurent concept drift

- “Seasonality” of data
- The data is different on the weekends, different behaviour over seasons etc

What should be retrained?

- If a concept drift occurred, it’s better to replace the whole dataset (offline learning)
- If there is a constant stream of new training data, we can leverage a time window that will exclude the oldest data and include the new data thus periodically retain the model (online learning)

When should the model be retrained?

- Periodic retraining: The model is retrained at a time interval
- Trigger based retraining: retrain the model when the model’s performance drops

When to not?

- Usually our initial dataset contains rich data and summarises that knowledge as efficiently as possible
- What if the quality of newly gathered data is not good enough?
- Should we retrain on the old data as well?
- Does the concept we are trying to predict change so often that we need to retrain our model every day/week/month?

How to get a handle on it?

- Find the optimal frequency for retraining
- Montior the model and check accuracy, precision or recall over time
- Keep a close eye on the prediction outputs: does the min/max mean or median value change over time?
- Statistical approach, model based approach, adaptive windowing

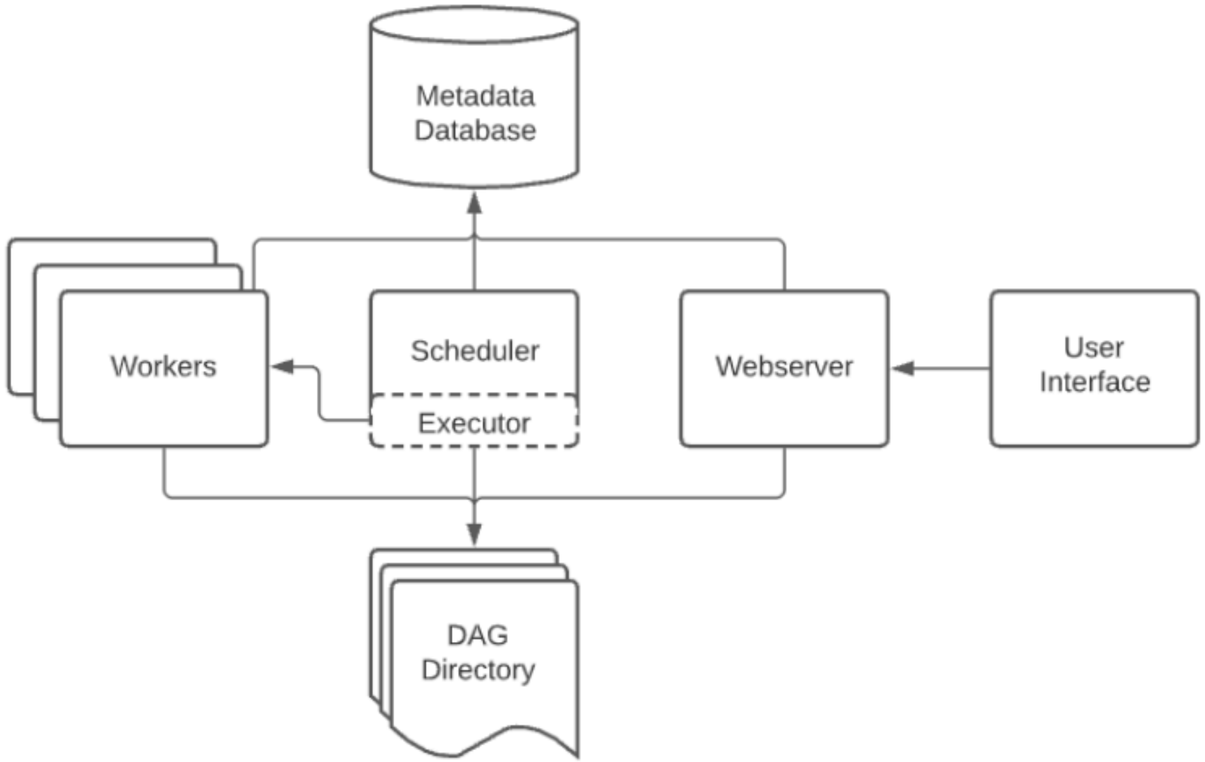
Automated model retraining tools

- MLOPS has become a popular topic in the recent years so there many tools and solutions that can help in implementing automated model retraining

- The following slides contain some examples and suggestions on how to select the right tool for your use-case

Apache Airflow

- Created by Airbnb
- Generic workflow orchestrator
- Workflows defined as DAGs
- Can run anywhere



Airflow architecture

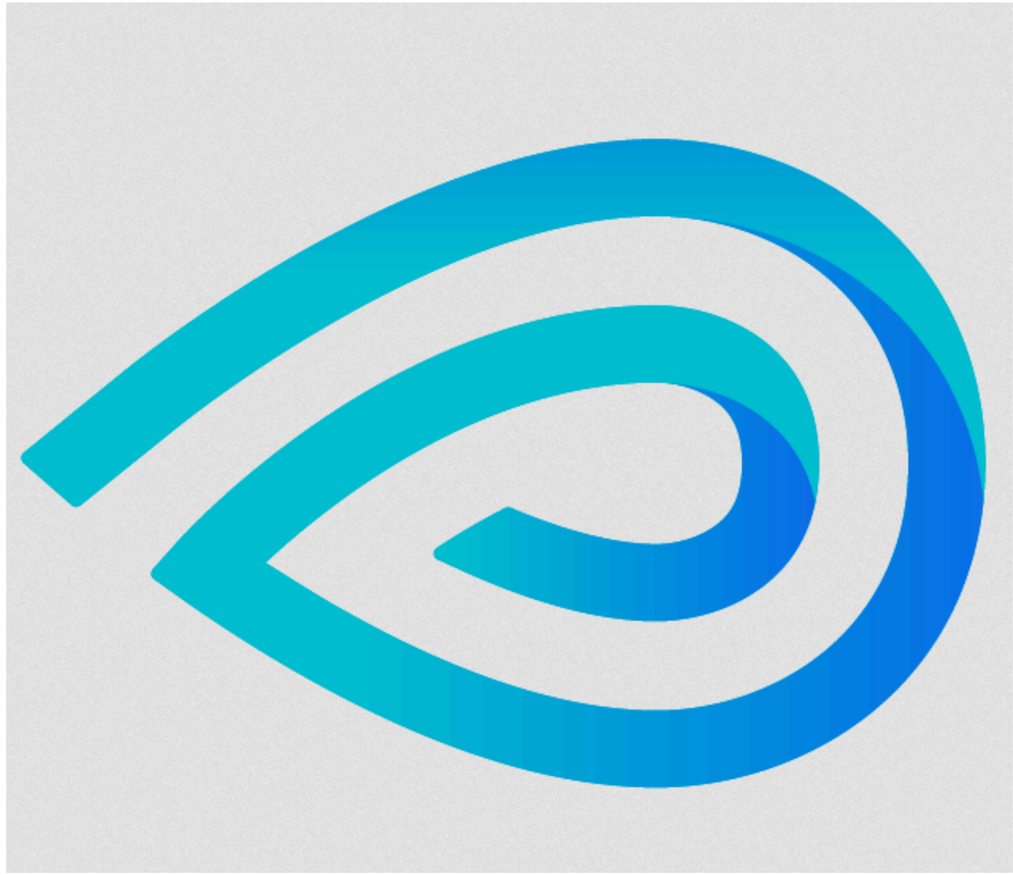
Airflow Pros/Cons

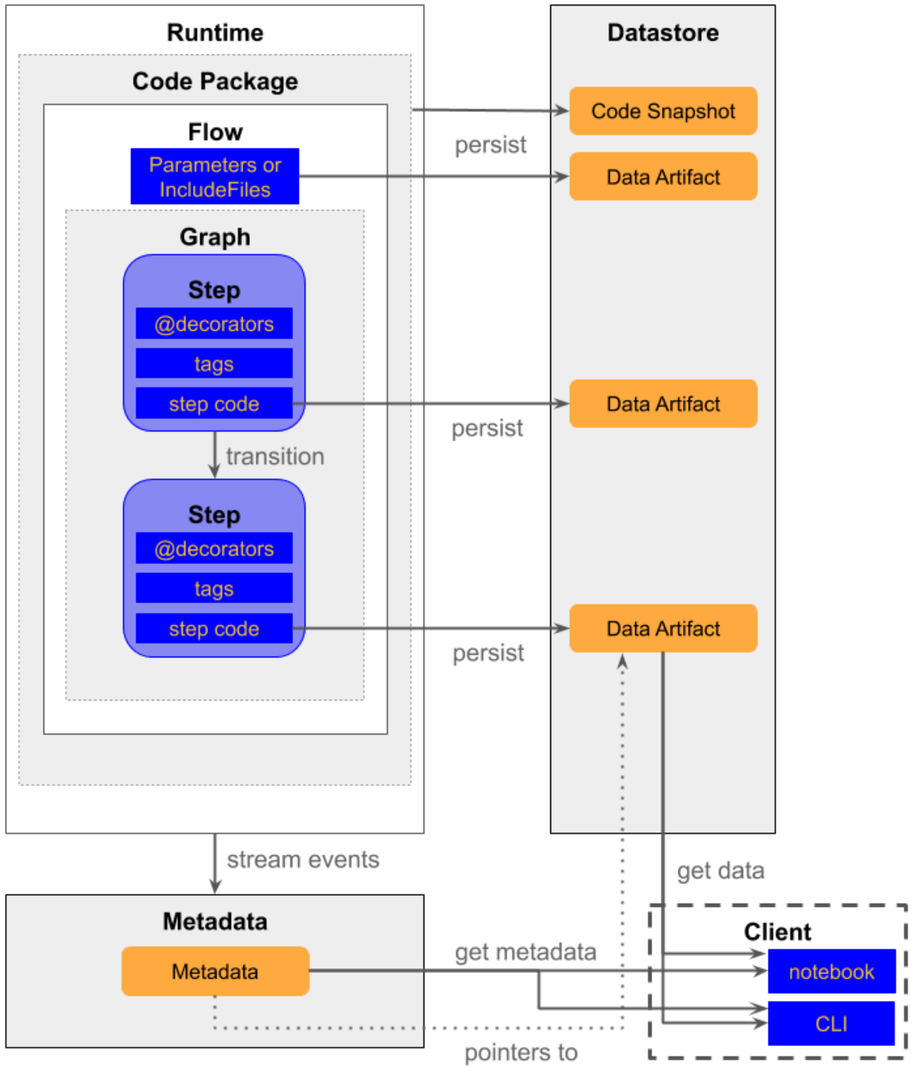
- Many ways to install it
- Can run in K8s, container, locally
- Not just for ML workflows
- Big community support

- Not friendly to prototyping
- Not multi environment friendly
- Requires good knowledge of the framework
- You need to handle synchronisation of DAGs and configs across nodes
- More suited for data and ETLs

Metaflow

- Developed at Netflix
- Workflows as DAGs
- human friendly python library





Metaflow architecture

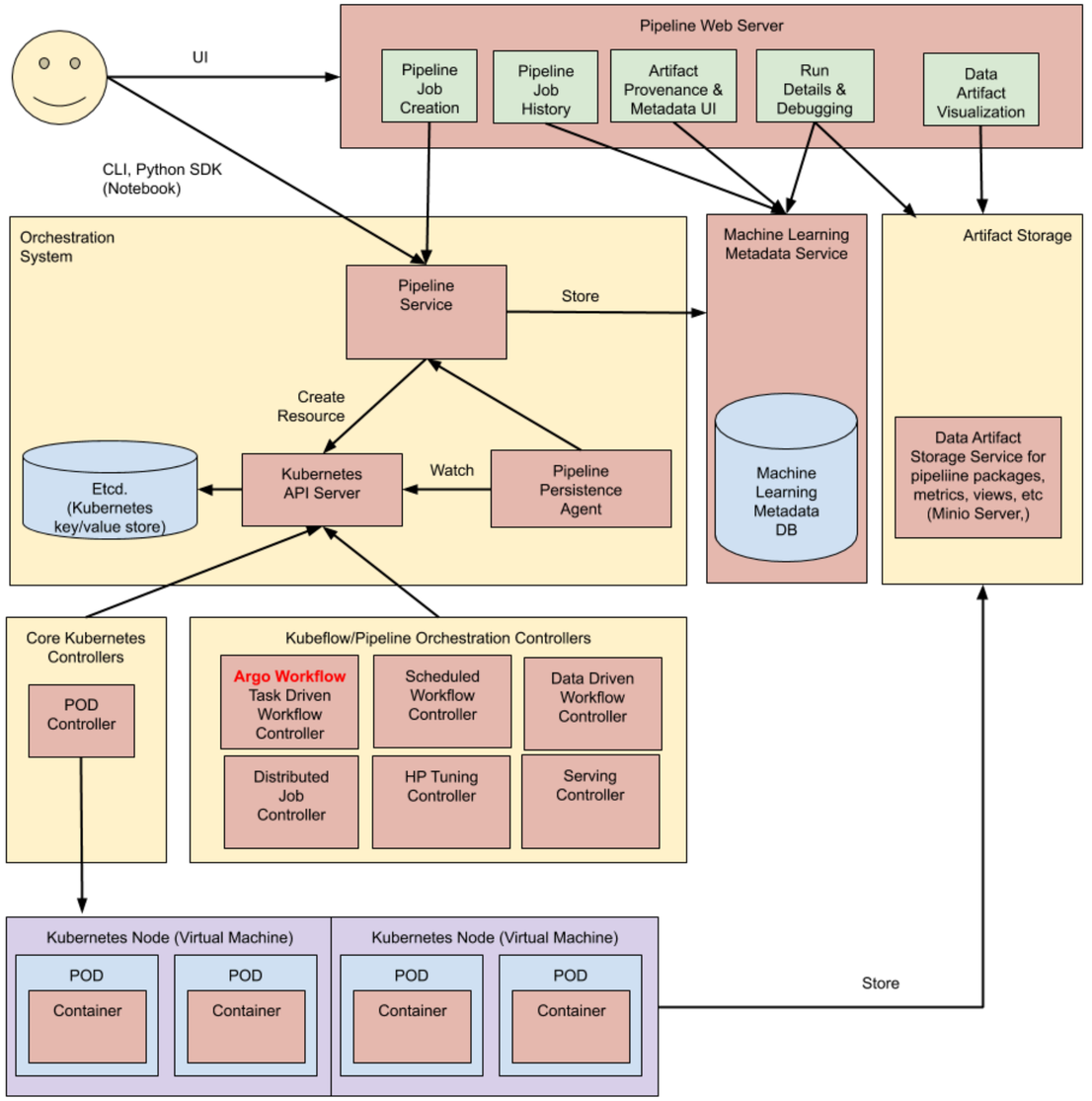
Pros/Cons

- Easiest to learn
- Data-science centric python library
- Prototyping friendly
- Everything is code
- Super easy to use
- Argo Workflows support

- Tightly coupled with AWS
- ~~No Kubernetes support~~
- Small community support

Kubeflow

- Developed by Google
- ML workflows on Kubernetes
- Made just for Kubernetes
- Workflows as graphs



Kubeflow components

Pros/Cons

- Kubernetes handles the resources
- Code can be written in Jupiter Notebooks
- Has support for all cloud providers
- E2e solution for ML workflows
- Good support for remote work

- Can run just in K8s
- Not big community
- Requires a lot of K8s knowledge
- High complexity
- Really hard to setup on a local machine, so not dev friendly

Questions?