

# ML Pipelines, Reproducibility and Experimentation

Alex Kim @alex000kim 

# We have a Jupyter notebook...



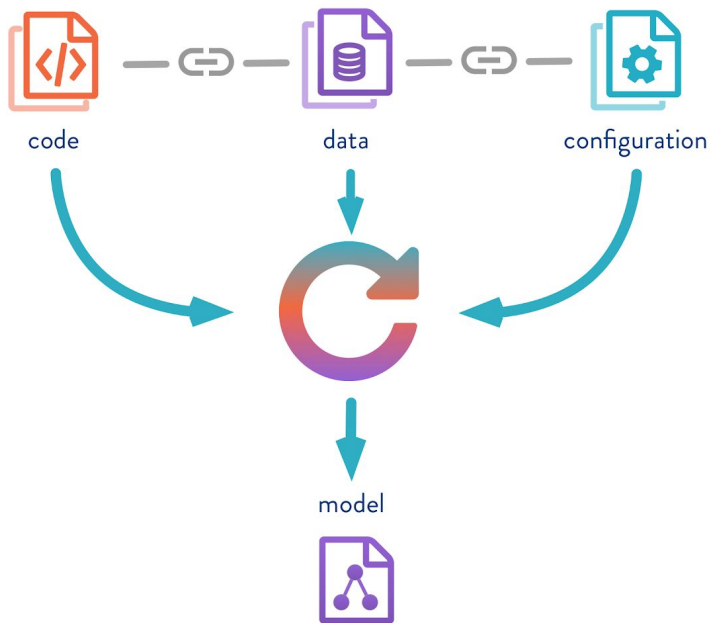
- ◇ Data loading
- ◇ Feature Engineering
- ◇ Model Training
- ◇ Model Evaluation

# Can you easily answer these questions?

- What exactly was used to produce a particular model?
- Can you easily compare many ML experiments?
- Will you be able to reproduce them later?

# Goal #1: Achieve best performance

- Running many experiments
- **Experiment** = a particular combination of **Code & Data & Config**



## Goal #2: Ensure reproducibility

- **Improving model performance:**  
you can't improve what you can't reproduce
- **Transparency and team collaboration:**  
know everything your team members did to achieve certain performance
- **Auditability (laws and regulations):**  
e.g. what *exactly* went into the models that prescribes treatment to patients or determines creditworthiness of bank customers

# Goal #3: Minimal setup and dependency of 3rd party services

Problems:

1. **Vendor lock-in:** instrument code with framework-specific code
2. **Maintenance & cost:** maintain your own ML tracking server (or pay them to take care of it)
3. **Security concerns:** send data to an external service or database

Most ML tracking solutions (MLflow, W&B, comet.ml, etc) have at least 2 of these problems

Fact:  
It's difficult to  
achieve all three  
goals

Can we do all of the following?

1. Iterate quickly i.e. generate many experiments
2. Automatically track **all** changes to code, configs and data
3. Avoid dependency on 3rd party services to store data, metrics and params

# Same experiments, but different metrics?



W&B

alex000kim > Projects > bank-customer-churn > Table

Runs (5)

Filter Group Sort Tag Move Create Sweep

	Name (5 visualized)	Created	f1	roc_auc	max_depth	n_estimators	model_type	ID
	hearty-gorge-9	2m ago	0.6084	0.8585	5	50	random-forest	1joxzkd8
	lucky-water-8	3m ago	0.5779	0.8658	5	50	random-forest	z85kgiiu
	bright-galaxy-7	4m ago	0.5282	0.8324	5	50	random-forest	1vf5dcy4
	denim-surf-6	6m ago	0.2899	0.7621	5	50	random-forest	xyz1k8t3
	crimson-glitter-5	7m ago	0.2461	0.7685	5	50	random-forest	6o3mzc6i



# Reproducibility VS. Experimentation?



- DVC pipelines for generating many experiments
- Achieve complete reproducibility by versioning **everything!**
  - code and configs --> Git
  - dataset, models, other artifacts --> DVC remote storage (cloud buckets, NAS, SFTP, etc)
- VS Code as a convenient UI for experiment management
- No need to maintain (or pay for) additional services



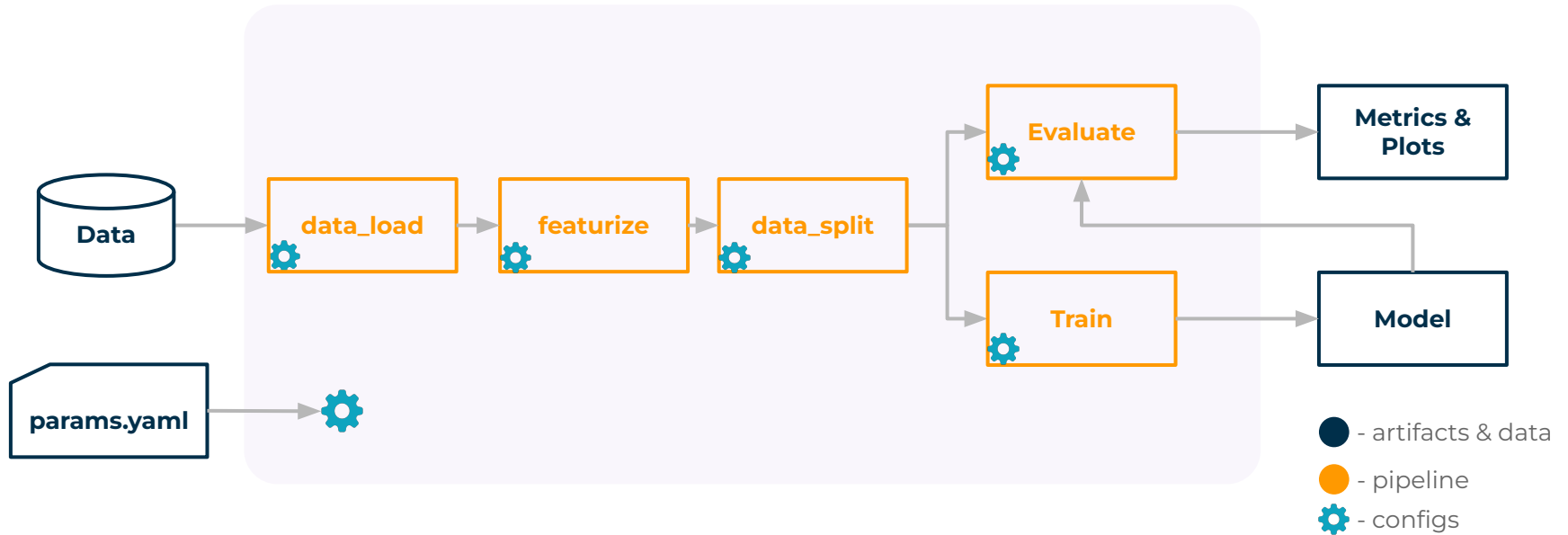
Visual Studio Code



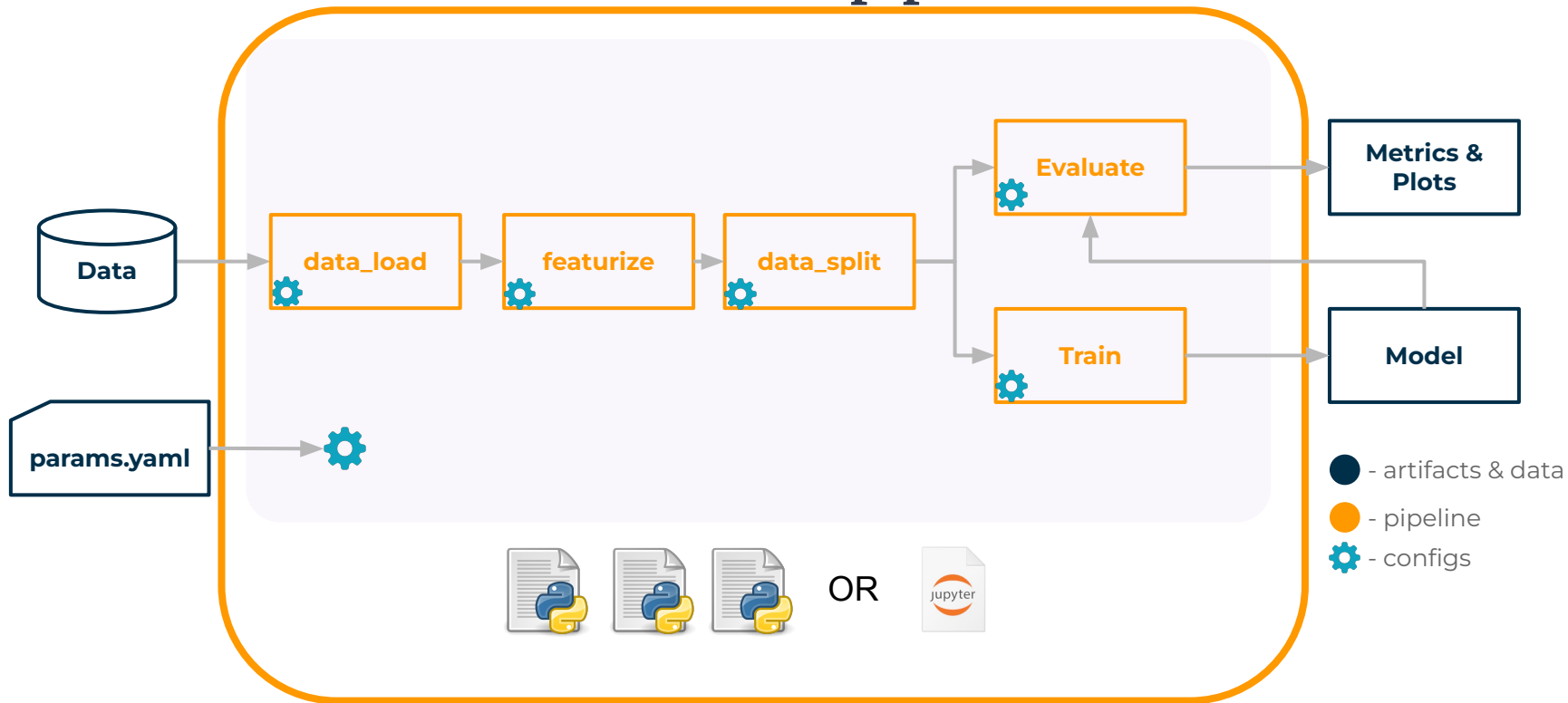
git



# What are DVC pipelines?






# What are DVC pipelines?




# DVC pipeline (defined in `dvc.yaml`) as:

a sequence of Python modules



```
stages:
  data_preprocessing_stage:
    cmd: python process_data.py
    deps:
      - process_data.py
      - path/to/raw/data
    outs:
      - path/to/processed/data
    params:
      - preprocessing_params
  train_stage:
    cmd: python train.py
    deps:
      - train.py
      - path/to/processed/data
    outs:
      - my_model.pickle
    params:
      - train_params
  eval_stage:
    cmd: python eval.py
    deps:
      - eval.py
      - path/to/processed/data
      - my_model.pickle
    params:
      - eval_params
```

a Jupyter notebook



```
stages:
  run_notebook_stage:
    cmd: papermill MyNotebook.ipynb MyNotebook_out.ipynb
      -p n_estimators ${n_estimators}
      -p max_depth ${max_depth}
    deps:
      - MyNotebook.ipynb
      - path/to/raw/data
    outs:
      - my_model.pickle
```

# Run an experiment

```
$ dvc exp run -S train.params.n_estimators=120
```

```
'data/Churn_Modelling.csv.dvc' didn't change, skipping
```

```
Running stage 'run_notebook':
```

```
> papermill TrainChurnModel.ipynb TrainChurnModel_out.ipynb -p n_estimators 120 -p max_depth 10 -p model_type lightgbm
```

```
Input Notebook: TrainChurnModel.ipynb
```

```
Output Notebook: TrainChurnModel_out.ipynb
```

```
Black is not installed, parameters wont be formatted
```

```
Executing: 0%|
```

```
| 0/27
```

```
[00:00<?, ?cell/s]Executing notebook with kernel: python3
```

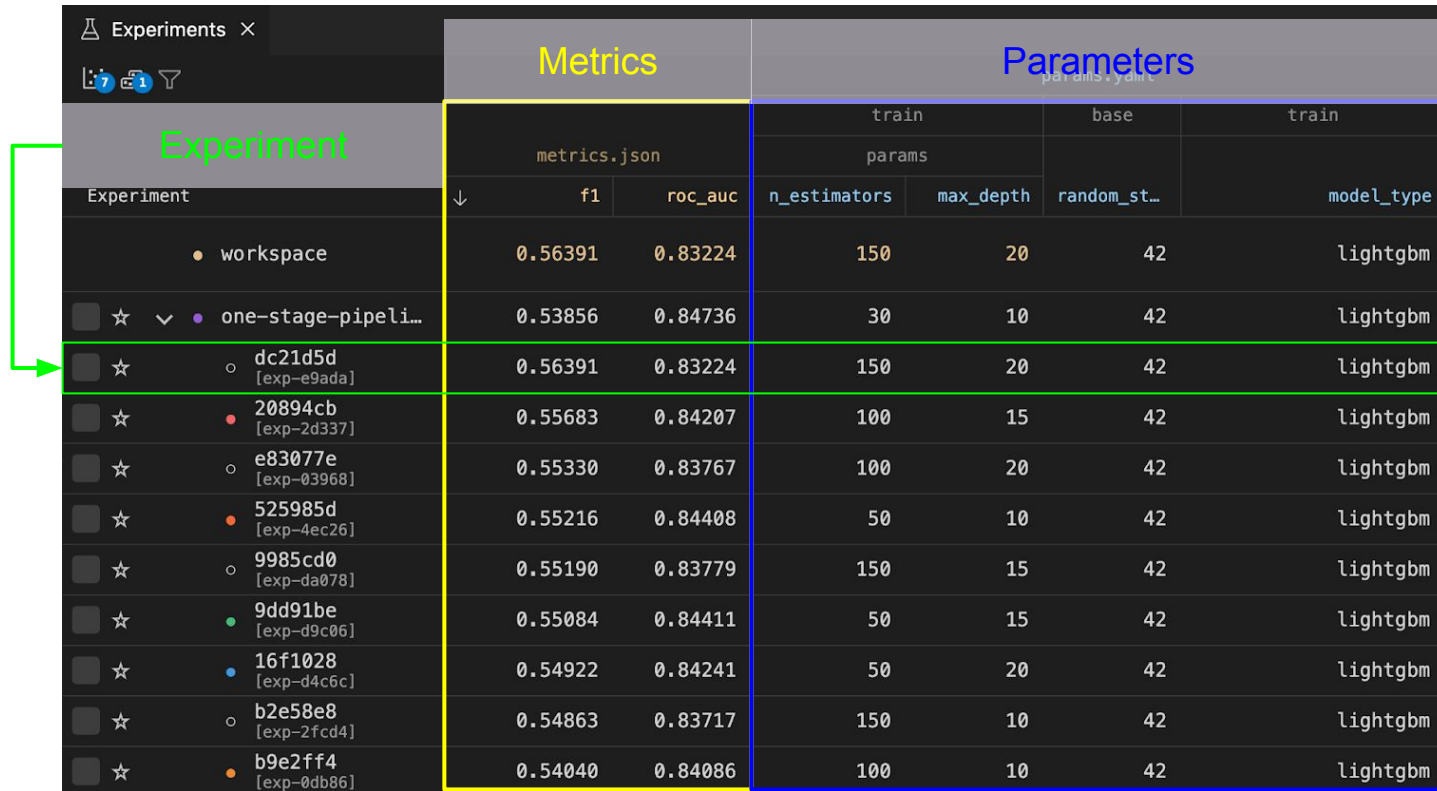
```
Executing: 100%|
```

```
| 27/27
```

```
[00:03<00:00, 7.58cell/s]
```

```
Updating lock file 'dvc.lock'
```

# Track and manage many experiments



The screenshot shows a web interface for tracking experiments. The main table is divided into three sections: 'Experiment', 'Metrics', and 'Parameters'. The 'Experiment' column lists various experiments, including a 'workspace' and several individual experiments with IDs and names. The 'Metrics' column shows 'f1' and 'roc\_auc' values for each experiment. The 'Parameters' column shows 'train', 'base', and 'train' parameters, with sub-columns for 'n\_estimators', 'max\_depth', 'random\_st...', and 'model\_type'. A green arrow points to the 'dc21d5d' experiment row, and a yellow box highlights the 'Metrics' column. A blue box highlights the 'Parameters' column.

Experiment	Metrics		Parameters			
	f1	roc_auc	n_estimators	max_depth	random_st...	model_type
workspace	0.56391	0.83224	150	20	42	lightgbm
one-stage-pipeli...	0.53856	0.84736	30	10	42	lightgbm
dc21d5d [exp-e9ada]	0.56391	0.83224	150	20	42	lightgbm
20894cb [exp-2d337]	0.55683	0.84207	100	15	42	lightgbm
e83077e [exp-03968]	0.55330	0.83767	100	20	42	lightgbm
525985d [exp-4ec26]	0.55216	0.84408	50	10	42	lightgbm
9985cd0 [exp-da078]	0.55190	0.83779	150	15	42	lightgbm
9dd91be [exp-d9c06]	0.55084	0.84411	50	15	42	lightgbm
16f1028 [exp-d4c6c]	0.54922	0.84241	50	20	42	lightgbm
b2e58e8 [exp-2fcd4]	0.54863	0.83717	150	10	42	lightgbm
b9e2ff4 [exp-0db86]	0.54040	0.84086	100	10	42	lightgbm

# Practice time!

