getindata

The MLOps platform that makes you productive, everywhere!

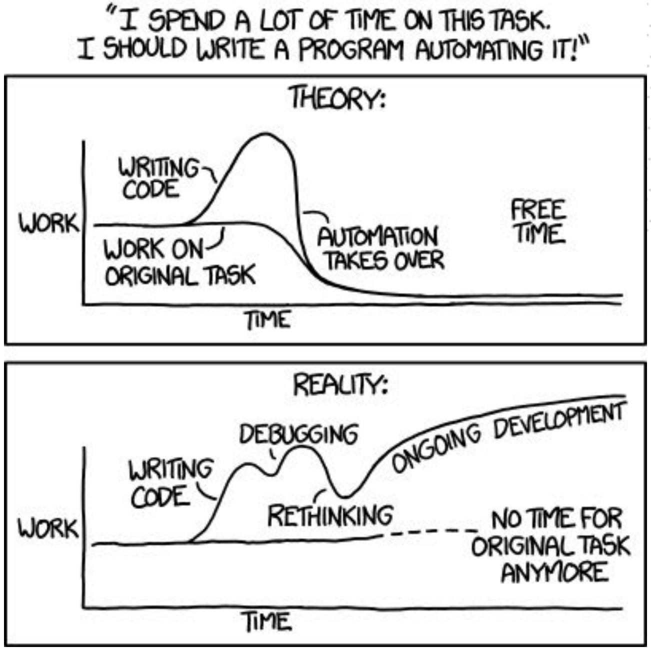
Klarna Meetup

Stockholm, 2023-05-23

What MLOps is (not only) about ?

getindata

- Application of the DevOps principles to ML world
- Managing ML model lifecycle
- Tools and platforms
- *Automation* and processes
- Infrastructure as Code



The ultimate goal is: **PRODUCTIVITY**

GID MLOps "Productivity Manifesto"

getindata

- Machine learning and data science should be ***first-class*** citizens of Data Platforms
- ***Open*** standards and cloud ***agnosticism***
- Short development ***feedback loop*** (incl. local dev)
- ***Fast*** new ML projects bootstrapping and ***standardization***
- Execution environment ***independent*** training pipelines
- Easy ***collaboration***
- … MLOps capabilities provisioned ***in days not months***

ML projects in layers

getindata

Data Scientist

Experimentation + EDA

Machine Learning frameworks

Example technologies:

ML projects in layers

getindata

Data Scientist

Example technologies:

Experimentation + EDA

Machine Learning frameworks

?

Execution environment

Data

MLOps / ML Engineer

Building blocks of the GID MLOps

getindata

Data Scientist

Experimentation + EDA

Machine Learning frameworks

Portable MLOps framework	Experiment tracking and collaboration	IaC and automation
Kedro	mlflow	Terraform

Cloud Integrations (incl. GID Kedro plugins)

Execution environment

Data

MLOps / ML Engineer

Example technologies:

jupyter

scikit learn

XGBoost

GID MLOps platform

Building blocks of the GID MLOps

getindata

Data Scientist

Example technologies:

Experimentation + EDA

Machine Learning frameworks

Portable MLOps framework	Experiment tracking and collaboration	IaC and automation
Kedro	mlflow	Terraform

Cloud Integrations (incl. GID Kedro plugins)

Execution environment

Data

GID MLOps Platform

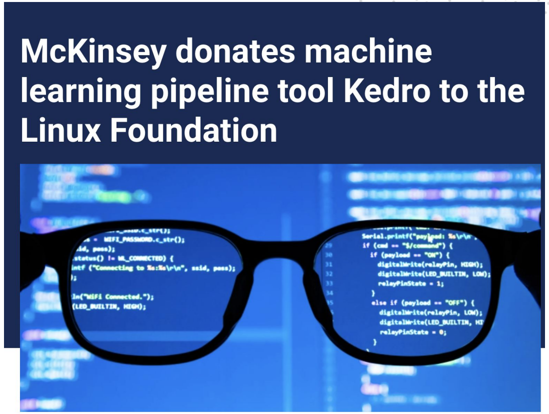
MLOps / ML Engineer

What is Kedro?

getindata

Kedro

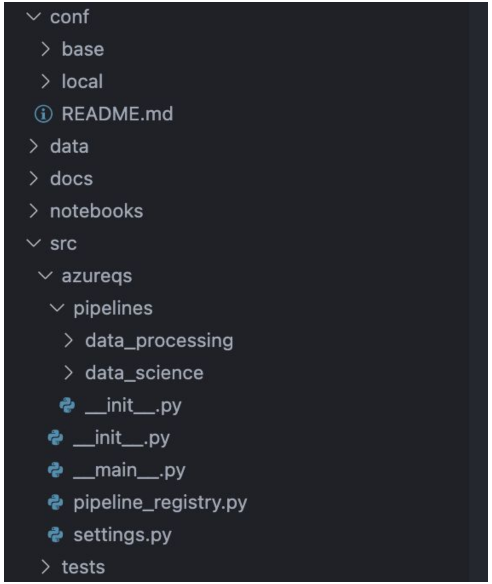
= Software Engineering Principles + Data Science



***Kedro** is an open-source Python framework*
for creating reproducible, maintainable and modular data science code.

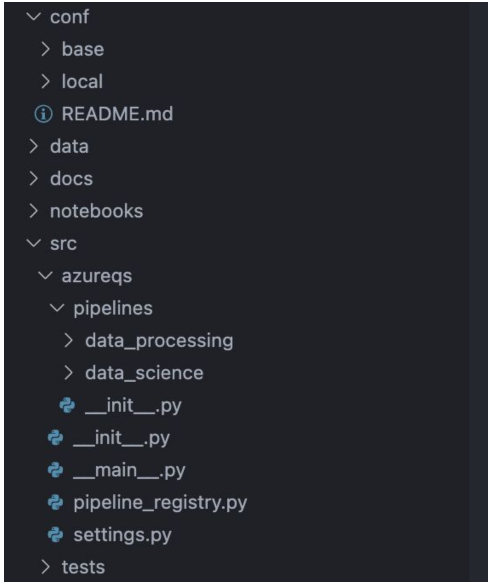
What features does Kedro have? (Part 1)

getindata

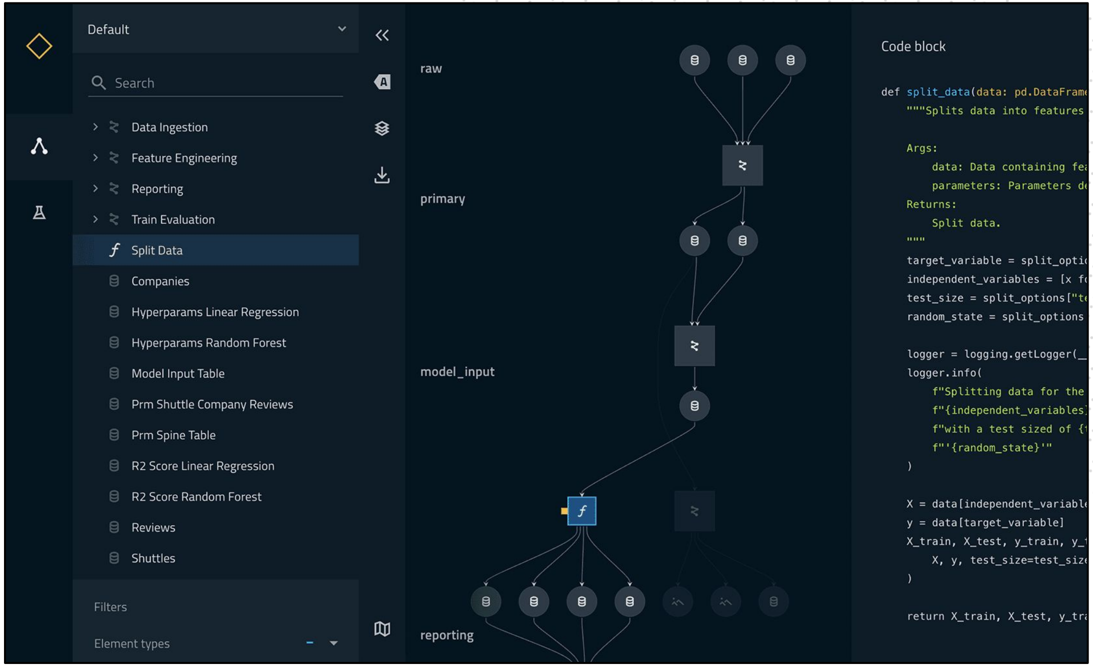


Well defined project structure

What features does Kedro have? (Part 1)



Well defined project structure + project starters



Nodes & pipelines abstractions

Kedro pipeline - data engineering

getindata

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=preprocess_companies,
                inputs="companies",
                outputs="preprocessed_companies",
                name="preprocess_companies_node",
            ),
            node(
                func=preprocess_reviews,
                inputs="reviews",
                outputs="preprocessed_reviews",
                name="preprocess_reviews_node",
            ),
            node(
                func=create_model_input_table,
                inputs=["preprocessed_reviews", "preprocessed_companies", "ratings"],
                outputs="model_input_table",
                name="create_model_input_table_node",
            ),
        ]
    )
```

Kedro pipeline - data science

getindata

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=split_data,
                inputs=["model_input_table", "params:model_options"],
                outputs=["X_train", "X_test", "y_train", "y_test"],
                name="split_data_node",
            ),
            node(
                func=train_model,
                inputs=["X_train", "y_train"],
                outputs="regressor",
                name="train_model_node",
            ),
            node(
                func=evaluate_model,
                inputs=["regressor", "X_test", "y_test"],
                outputs=None,
                name="evaluate_model_node",
            ),
        ]
    )
```

Kedro node

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=preprocess_companies,
                inputs="companies",
                outputs="preprocessed_companies",
                name="preprocess_companies_node",
            ),
            node(
                func=preprocess_reviews,
                inputs="reviews",
                outputs="preprocessed_reviews",
                name="preprocess_reviews_node",
            ),
            node(
                func=create_model_input_table,
                inputs=["preprocessed_reviews", "
                outputs="model_input_table",
                name="create_model_input_table_node",
            ),
        ]
    )
```

```python
def create_model_input_table(
    reviews: pd.DataFrame, companies: pd.DataFrame, ratings: pd.DataFrame
) -> pd.DataFrame:
    """Combines all data to create a model input table.

    Args:
        reviews: Preprocessed data for reviews.
        companies: Preprocessed data for companies.
        ratings: Raw data for ratings.
    Returns:
        Model input table.

    """
    reviews_with_ratings = reviews.merge(ratings, left_on="id", right_on="rating_id")
    model_input_table = reviews_with_ratings.merge(
        companies, left_on="company_id", right_on="id"
    )
    model_input_table = model_input_table.dropna()
    return model_input_table
```

What about parameters?

getindata

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=split_data,
                inputs=["model_input_table", "params:model_options"],
                outputs=["X_train", "X_test", "y_train", "y_test"],
                name="split_data_node",
            ),
            node(
                func=train_model,
                inputs=["X_train", "y_train"],
                outputs="regressor",
                name="train_model_node",
            ),
            node(
                func=evaluate_model,
                inputs=["regressor", "X_test", "y_test"],
                outputs=None,
                name="evaluate_model_node",
            ),
        ]
    )
```

What about parameters?

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=split_data,
                inputs=["model_input_table", "p
                outputs=["X_train", "X_test", "
                name="split_data_node",
            ),
            node(
                func=train_model,
                inputs=["X_train", "y_train"],
                outputs="regressor",
                name="train_model_node",
            ),
            node(
                func=evaluate_model,
                inputs=["regressor", "X_test", "y_test"],
                outputs=None,
                name="evaluate_model_node",
            ),
        ]
    )
```

```
conf
  base
    parameters
      data_processing.yml
      data_science.yml
    azureml.yml
    catalog.yml
    logging.yml
    parameters.yml
  local
data
docs
```

```yaml
model_options:
  test_size: 0.2
  random_state: 3
  features:
    - engines
    - passenger_capacity
    - crew
    - d_check_complete
    - moon_clearance_complete
    - iata_approved
    - company_rating
    - review_scores_rating
```

ML model?

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=split_data,
                inputs=["model_input_table", "params:model_options"],
                outputs=["X_train", "X_test", "y_train", "y_test"],
                name="split_data_node",
            ),
            node(
                func=train_model,
                inputs=["X_train", "y_train"],
                outputs="regressor",
                name="train_model_node",
            ),
            node(
                func=evaluate_model,
                inputs=["regressor", "X_test", "y_test"],
                outputs=None,
                name="evaluate_model_node",
            ),
        ]
    )

```

?

What about data?

getindata

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=preprocess_companies,
                inputs="companies",
                outputs="preprocessed_companies",
                name="preprocess_companies_node",
            ),
            node(
                func=preprocess_reviews,
                inputs="reviews",
                outputs="preprocessed_reviews",
                name="preprocess_reviews_node",
            ),
            node(
                func=create_model_input_table,
                inputs=["preprocessed_reviews", "preprocessed_companies", "ratings"],
                outputs="model_input_table",
                name="create_model_input_table_node",
            ),
        ]
    )
```

Kedro Data Catalog

```python
def create_pipeline(**kwargs) -> Pipeline:
    return pipeline(
        [
            node(
                func=preprocess_companies,
                inputs="companies",
                outputs="preprocessed_co
                name="preprocess_compani
            ),
            node(
                func=preprocess_reviews,
                inputs="reviews",
                outputs="preprocessed_re
                name="preprocess_reviews
            ),
            node(
                func=create_model_input_
                inputs=["preprocessed_reviews", "preprocessed_companies", "ratings"],
                outputs="model_input_table",
                name="create_model_input_table_node",
            ),
        ]
    )
```

```
conf
  base
    parameters
    azureml.yml
    catalog.yml
    logging.yml
    parameters.yml
  local
data
docs
notebooks
src
```

```yaml
companies:
  type: pandas.CSVDataSet
  filepath: data/01_raw/companies.csv

reviews:
  type: pandas.ParquetDataSet
  filepath: data/01_raw/reviews.parquet

pictures:
  type: pillow.ImageDataSet
  filepath: data/01_raw/images/*.jpg
```

What features does Kedro have? (Part 2)

getindata

Local catalog.yml

```yaml
companies:
  type: pandas.CSVDataSet
  filepath: data/01_raw/companies.csv

reviews:
  type: pandas.ParquetDataSet
  filepath: data/01_raw/reviews.parquet

pictures:
  type: pillow.ImageDataSet
  filepath: data/01_raw/images/*.jpg
```

Cloud catalog.yml

```yaml
companies:
  type: pandas.CSVDataSet
  filepath: abfs://my_blob_container/data/01_raw/companies.csv

reviews:
  type: pandas.SQLQueryDataSet
  sql: "select * from reviews;"
  credentials: db_credentials

pictures:
  type: kedro_azureml.AzureMLFileDataSet
  dataset: my_dataset_from_azureml
  filepath: data/01_raw/images/*.jpg
```

Data Catalog

What features does Kedro have? (Part 2)

getindata

Local catalog.yml

```yaml
companies:
  type: pandas.CSVDataSet
  filepath: data/01_raw/companies.csv

reviews:
  type: pandas.ParquetDataSet
  filepath: data/01_raw/reviews.parquet

pictures:
  type: pillow.ImageDataSet
  filepath: data/01_raw/images/*.jpg
```

Cloud catalog.yml

```yaml
companies:
  type: pandas.CSVDataSet
  filepath: abfs://my_blob_container/data/01_raw/companies.csv

reviews:
  type: pandas.SQLQueryDataSet
  sql: "select * from reviews;"
  credentials: db_credentials

pictures:
  type: kedro_azureml.AzureMLFileDataSet
  dataset: my_dataset_from_azureml
  filepath: data/01_raw/images/*.jpg
```

Data Catalog

Extensibility & Integrations

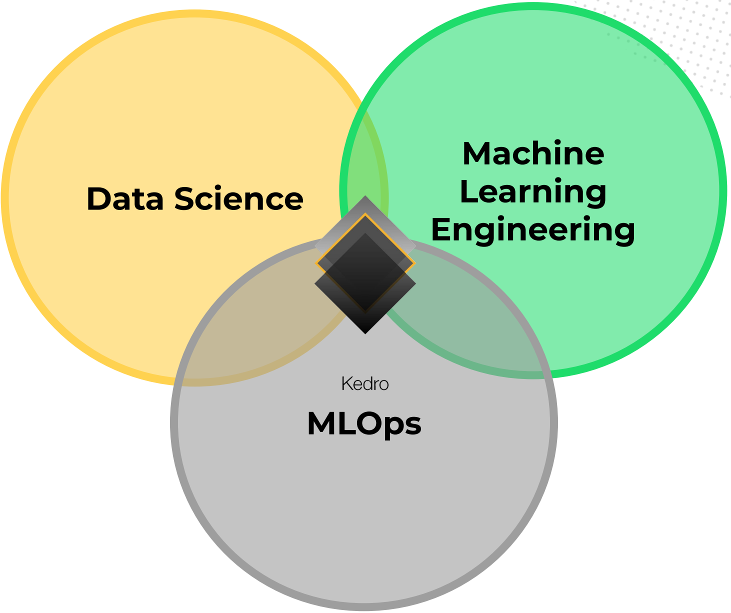
Kedro can be integrated with multiple industry leading solutions, including: Apache Spark, Pandas, Dask, Matplotlib, Plotly, fsspec, Apache Airflow, Jupyter Notebook and Docker.

Cloud agnostic with Kedro

getindata

Machine Learning Frameworks

XGBoost



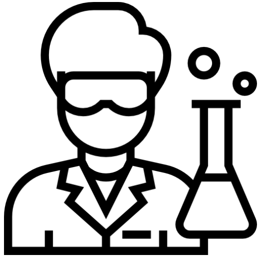
Model serving frameworks

Infrastructure tools

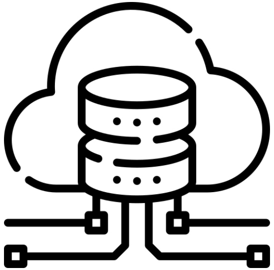
MLOps Platform by GetinData

What we're achieving with MLOps Platform

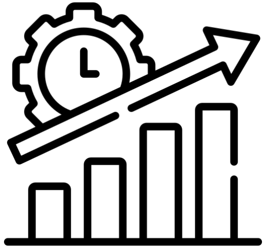
getindata



Reliable experiment tracking



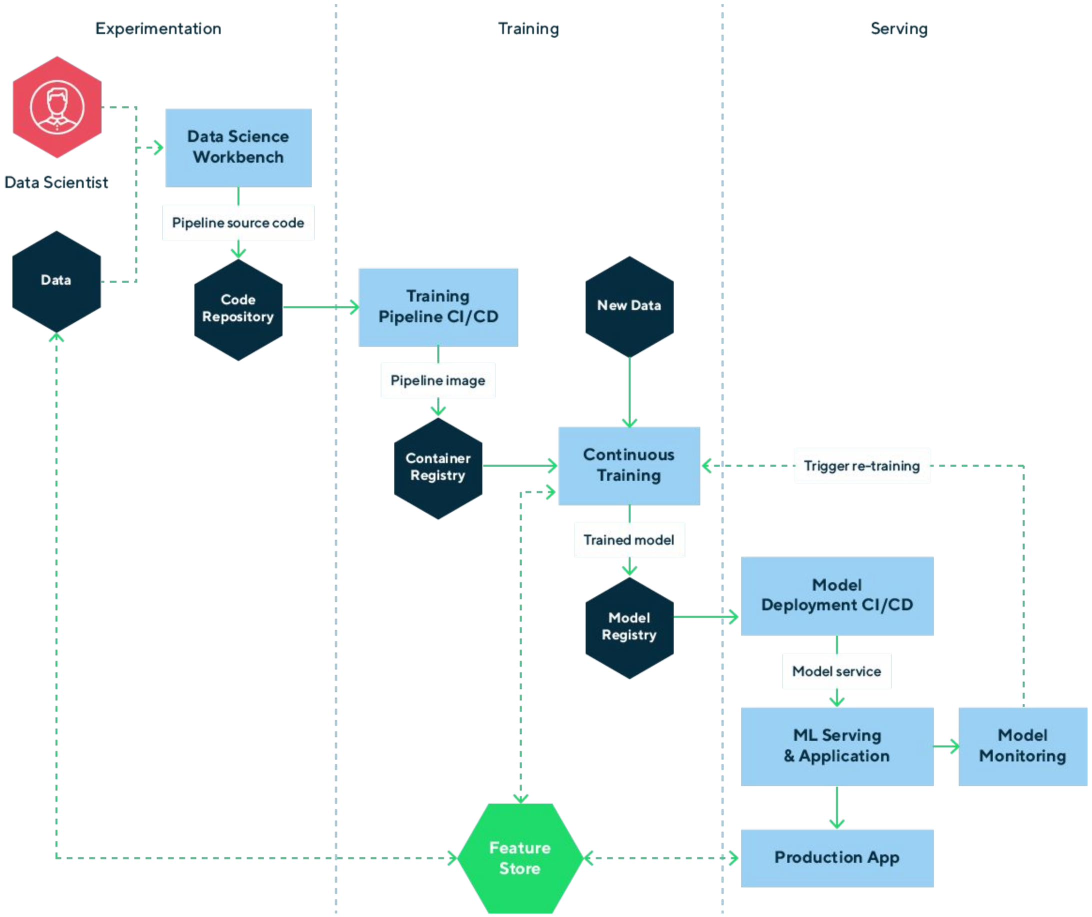
Easy path
from local to cloud



Faster time to market

Model management

Icons made by Freepik and catkuro from www.flaticon.com

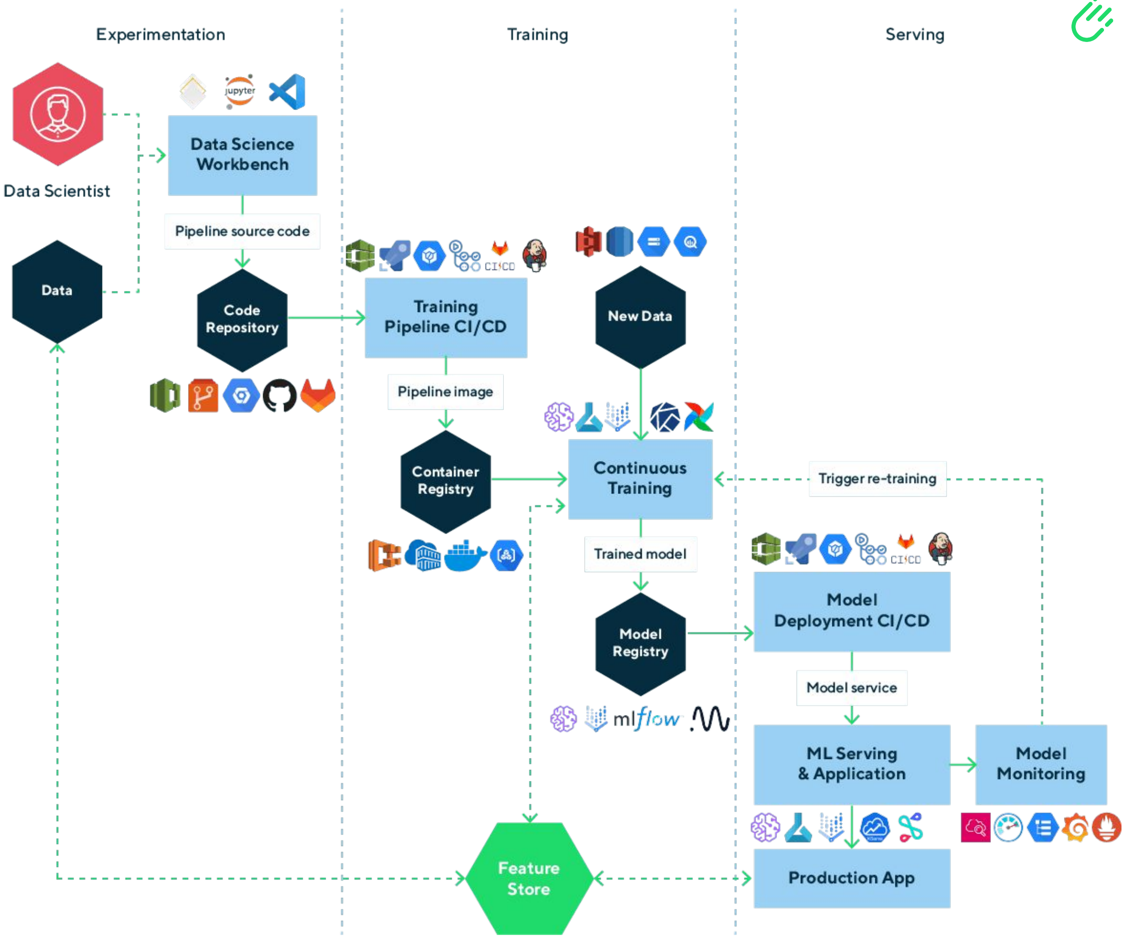


getindata

MLOps Platform

We built a best-of-breed solution instead od all-in-one

getindata



MLOps Platform

A framework of best MLOps practices to make the process of ML experimentation, model training, and model serving efficient, secure and reliable. We built a best-of-bread solution instead of all-in-one

Supported cloud platforms

Future integrations

Write once - run (almost) everywhere

getindata

Kedro

Kedro VertexAI (GCP)
github.com/getindata/kedro-vertexai

Kedro AzureML (Azure)
github.com/getindata/kedro-azureml

Kedro Sagemaker (AWS)
github.com/getindata/kedro-sagemaker

Kedro Airflow (Kubernetes)
github.com/getindata/kedro-airflow-k8s

Kedro Kubeflow (Kubernetes)
github.com/getindata/kedro-kubeflow

In progress:
Kedro Snowflake & Databricks

Read more: https://getindata.com/blog/running-kedro-everywhere-machine-learning-pipelines-kubeflow-vertex-ai-azure-airflow/

How plugins work?

getindata

Pipeline definition

Kedro

```
def train_model(X_train: pd.DataFrame, y_train: pd.Series)
-> LinearRegression:
    regressor = LinearRegression()
    regressor.fit(X_train, y_train)
    return regressor
```

Cloud Native SDKs

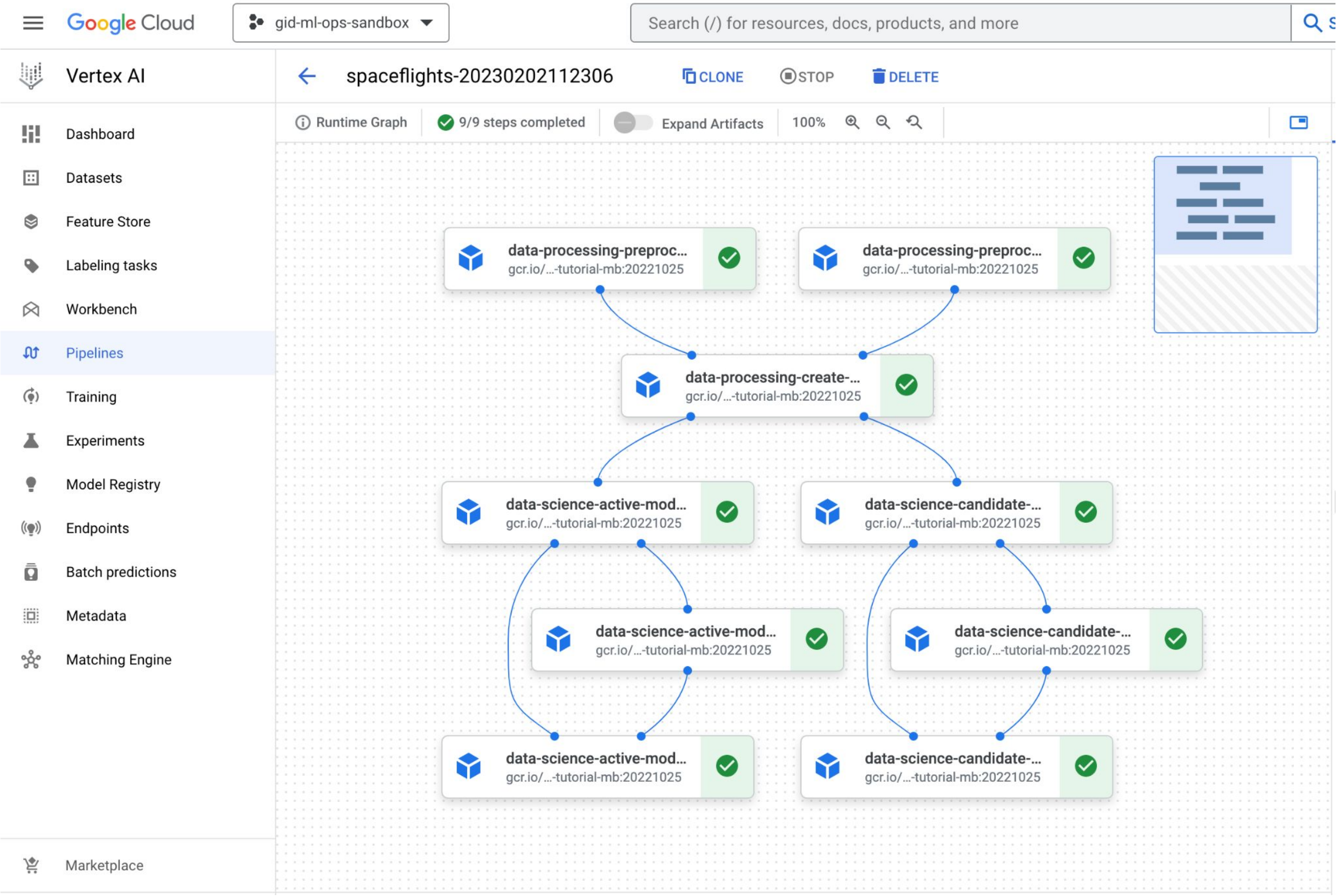
```
@dsl.component(kfp_package_path=_KFP_PACKAGE_PATH)
def train_model(X_train: pd.DataFrame, y_train: pd.Series) -> str:

    …
    return regressor
```

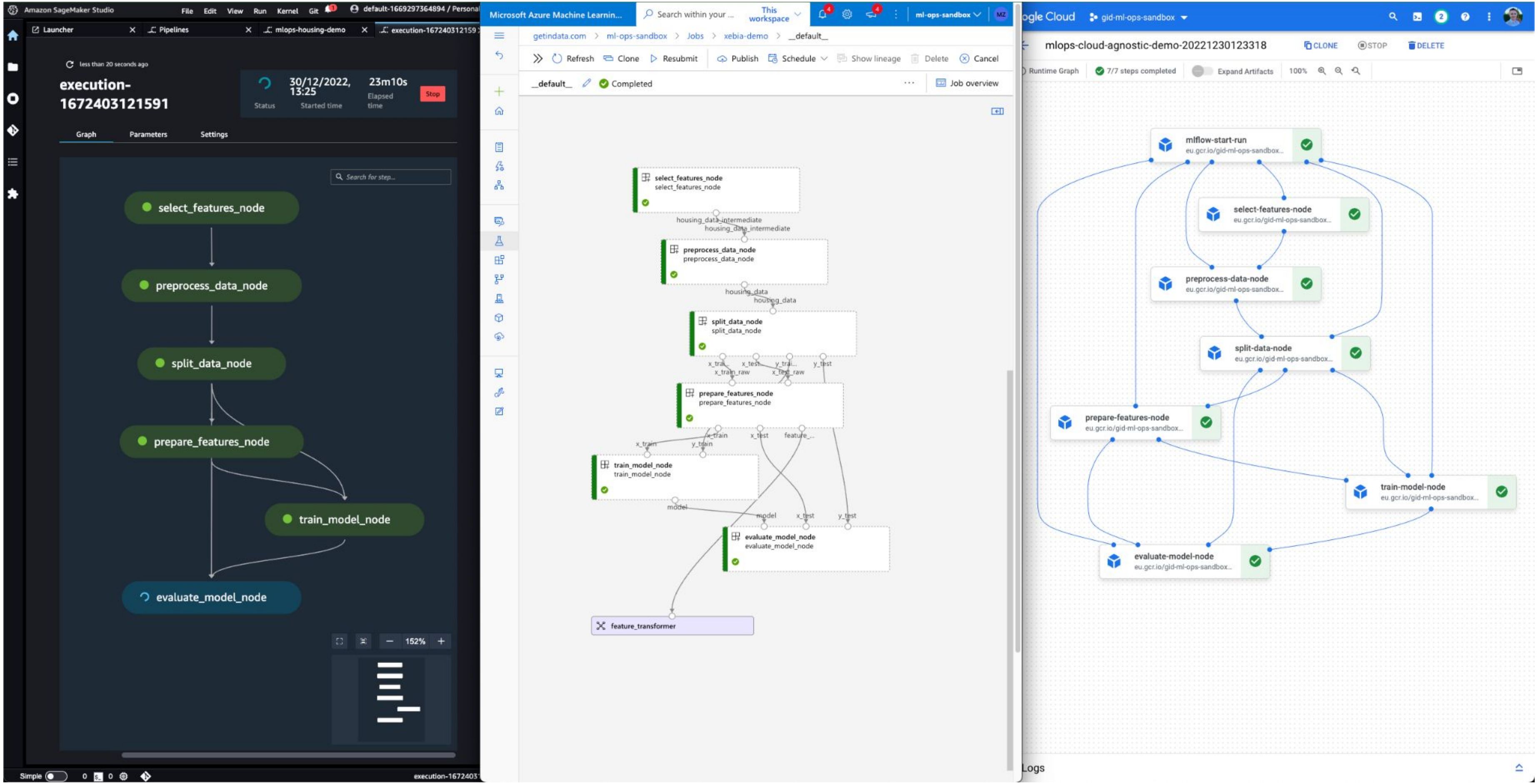
Serialized pipeline deployment

```
"exec-data-science-active-modelling-pipeline-train-model-node": {
    "container": {
        "args": [
            "kedro vertexai -e local initialize-job --params='{\"data_science\": {\"active_modelling_pipeline\":
{\"model_options\":             {\"test_size\": 0.2, \"random_state\": 3, \"features\": [\"engines\",
\"passenger_capacity\", \"crew\", \"d_check_complete\", \"moon_clearance_complete\", \"iata_approved\",
\"company_rating\", \"review_scores_rating\"]}}, \"candidate_modelling_pipeline\": {\"model_options\":
{\"test_size\": 0.2, \"random_state\": 8, \"features\": [\"engines\", \"passenger_capacity\", \"crew\",
\"review_scores_rating\"]}}}}' &&  KEDRO_VERTEXAI_DISABLE_CONFIG_HOOK=false
KEDRO_CONFIG_RUN_ID={{$.pipeline_job_uuid}} KEDRO_CONFIG_JOB_NAME={{$.pipeline_job_name}}
KEDRO_VERTEXAI_RUNNER_CONFIG='{\"storage_root\": \"mb-temp/mlops-webinar-demo\"}'  kedro run -e local --pipeline
__default__ --node \"data_science.active_modelling_pipeline.train_model_node\" --runner
kedro_vertexai.vertex_ai.runner.VertexAIPipelinesRunner --config config.yaml"
            ],...
```

*Our **Plugins** translate the ML Pipeline from Kedro to selected execution environment.*



Google Cloud
gid-ml-ops-sandbox
Search (/) for resources, docs, products, and more
Vertex AI
spaceflights-20230202112306
CLONE
STOP
DELETE
Dashboard
Datasets
Feature Store
Labeling tasks
Workbench
Pipelines
Training
Experiments
Model Registry
Endpoints
Batch predictions
Metadata
Matching Engine
Marketplace
Runtime Graph
9/9 steps completed
Expand Artifacts
100%
data-processing-preproc...
gcr.io/...-tutorial-mb:20221025
data-processing-preproc...
gcr.io/...-tutorial-mb:20221025
data-processing-create-...
gcr.io/...-tutorial-mb:20221025
data-science-active-mod...
gcr.io/...-tutorial-mb:20221025
data-science-candidate-...
gcr.io/...-tutorial-mb:20221025
data-science-active-mod...
gcr.io/...-tutorial-mb:20221025
data-science-candidate-...
gcr.io/...-tutorial-mb:20221025
data-science-active-mod...
gcr.io/...-tutorial-mb:20221025
data-science-candidate-...
gcr.io/...-tutorial-mb:20221025



Amazon SageMaker Studio
File
Edit
View
Run
Kernel
Git
Launcher
Pipelines
mlops-housing-demo
less than 20 seconds ago
execution-1672403121591
30/12/2022, 13:25
23m10s
Stop
Status
Started time
Elapsed time
Graph
Parameters
Settings
Search for step...
select_features_node
preprocess_data_node
split_data_node
prepare_features_node
train_model_node
evaluate_model_node
152%
Simple
Microsoft Azure Machine Learnin...
Search within your ...
This workspace
ml-ops-sandbox
getindata.com
ml-ops-sandbox
Jobs
xebia-demo
__default__
Refresh
Clone
Resubmit
Publish
Schedule
Show lineage
Delete
Cancel
Completed
Job overview
select_features_node
housing_data_intermediate
preprocess_data_node
housing_data
split_data_node
x_train_raw
x_test_raw
y_test
prepare_features_node
x_train
y_train
x_test
train_model_node
model
evaluate_model_node
feature_transformer
gid-ml-ops-sandbox
mlops-cloud-agnostic-demo-20221230123318
CLONE
STOP
DELETE
Runtime Graph
7/7 steps completed
Expand Artifacts
100%
mlflow-start-run
select-features-node
preprocess-data-node
split-data-node
prepare-features-node
train-model-node
evaluate-model-node
Logs

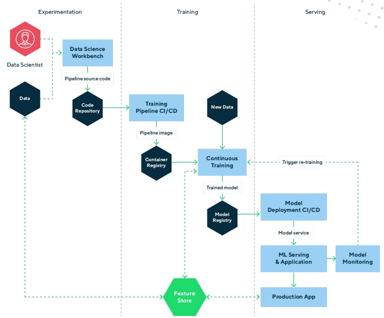
Our approach: cloud agnostic MLOps Platform

getindata

Client 1
Client 2
Client 3

Kedro

MLOps Framework



MLOps Platform

Completed project!

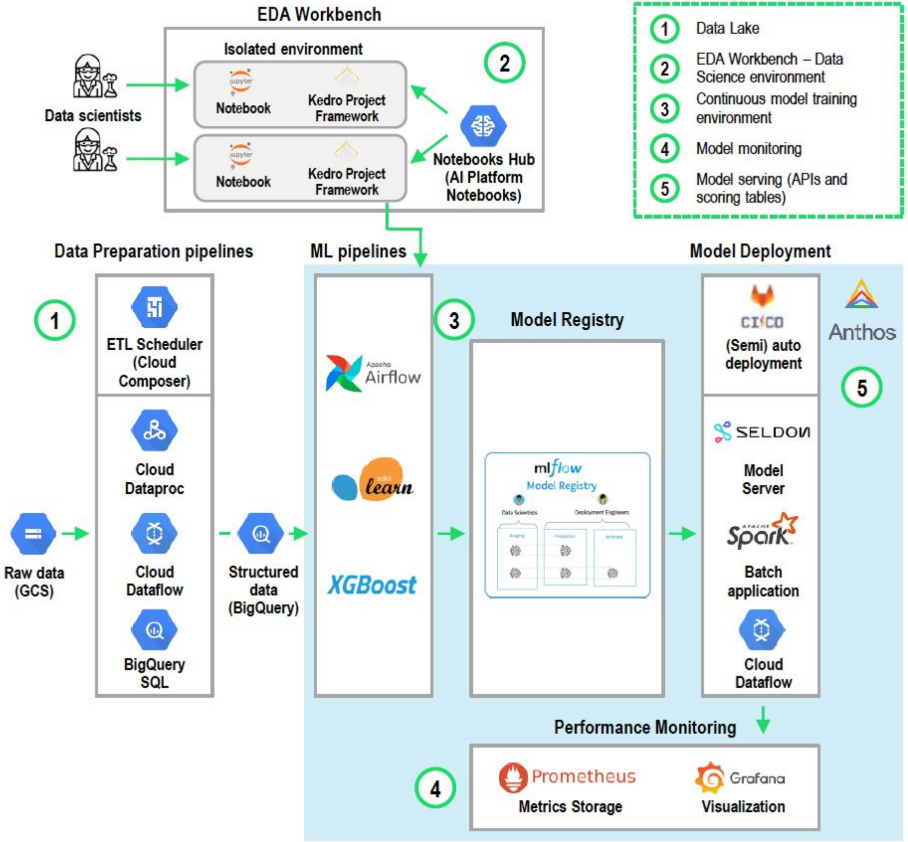
With **cloud agnostic** approach, we can unify MLOps projects with one framework (Kedro) and deploy it into various cloud platforms with Kedro Plugins.

MLOps Platform to Run Production ML/AI Models (banking)

getindata

CONFIDENTIAL

Leading Bank in CEE



Key technologies

Apache Airflow

Google Cloud Platform

jupyter

Apache Spark

mlflow

SELDON

Grafana

Modern Data Platform, analytics and MLOps (FinTech)

getindata

We have delivered **Data Platform** that **supports general analytics** (ad-hoc querying, reporting in BI tools) and Machine Learning initiatives. Data Platform is build on top of Google Cloud services (BigQuery, Cloud Composer, Data Studio) and open-source projects (dbt, Terraform).

We have delivered an **MLOps Platform** that **supports a production-grade ML model lifecycle.** Our solution is using heavily Google Cloud Platform services (Vertex AI, BigQuery, Cloud Build) and open-source projects (mlflow, Kedro).

We have co-developed a number of **production AI/ML models** such as User Suspension Model, Invoice Model, Activation Model.

willa

Willa is a Sweden and U.S.-based FinTech that helps professional freelancers, influencers, and social media content creators get paid immediately by brands for their freelance work and paid collaborations.

Key technologies

Google Cloud Platform

dbt

Google Big Query

mlflow

[Building robust ML & Analytics capability very early at a FinTech]

getindata

MLOps Labs

Michał Bryś
Machine Learning Engineer at Getindata

michal.brys@getindata.com

Marek Wiewiórka
Chief Data Architect at Getindata

marek@getindata.com

Marcin Zabłocki
MLOps Architect at Getindata

marcin.zablocki@getindata.com

Let's meet our (hypothetical) clients

getindata

Client 1

- A few ML models, no pipelines, manual / script based on VMs
- Data Scientists distributed across different teams
- Strict compliance requirements

Client 2

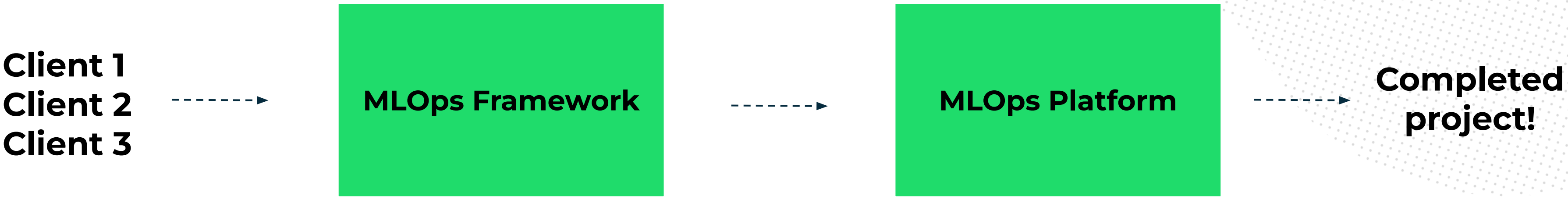
- Small data volume, but this will change quickly as the business grows
- One model, trained on the DS workstation / Jupyter Notebook
- Limited budget for the infrastructure
- Small team, overwhelmed by tasks

Client 3

- Current ML workloads running on-premises on Kubernetes
- Exploring the cloud services
- Lots of technical debt

Our approach: cloud agnostic MLOps Platform

getindata



With **cloud agnostic** approach, we can unify **MLOps** projects with one framework (Kedro) and deploy them to various cloud platforms with Kedro Plugins.

getindata

ML model training in layers

Data Scientist

- **Experimentation + EDA**
- **Machine Learning framework**
- **MLOps Framework**
- **Integrations (plugins)**
- **Execution environment (local, cloud)**
- **The data**

Example technologies:

Kedro abstracts the pipeline from the execution environment SDK.

Under-engineering refers to building with reduced complexity resulting in **a less robust, efficient and capable product**. It happens because of tight deadlines or because of a lack of understanding.

We under-engineer machine-learning prototypes and create code with a lot of technical debt

Technical debt is intentional or accidental decisions that make ***code difficult to understand, maintain, extend and fix errors***. Much like a loan, you pay a higher cost later, as it decreases the team's agility as the project matures.

What features does Kedro have?

getindata

```
project-template    # Project folder
├── conf            # Configuration files
├── data            # Local project data
├── docs            # Documentation
├── logs            # Logs of pipeline runs
├── notebooks       # Exploratory Jupyter notebooks
├── pyproject.toml  # Identifies the project root
├── setup.cfg       # Configuration options for tes
├── README.md       # README.md explaining your pro
├── setup.cfg       # Configuration options for tes
└── src             # Source code for pipelines
```

FEATURES

Project Templates

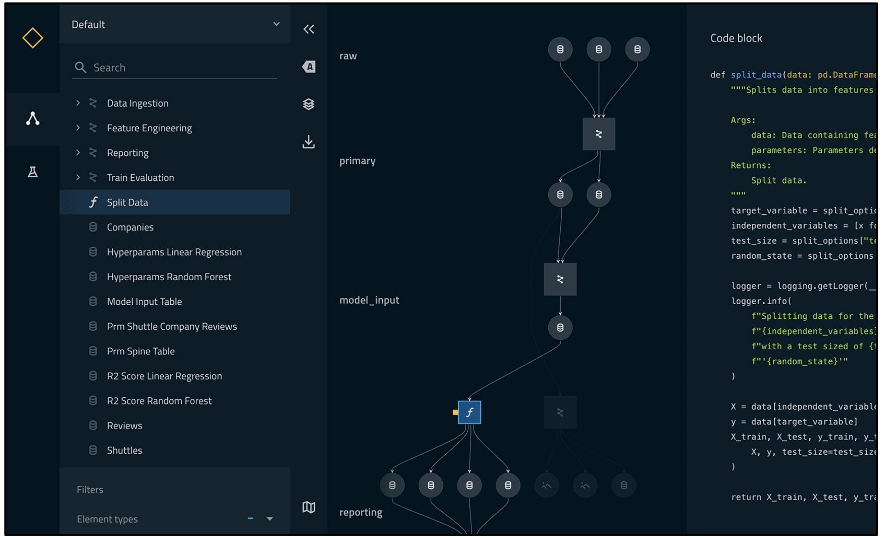
Kedro starter contains code in the form of a Cookiecutter template for a ML project. Metaphorically, a starter is similar to using a pre-defined layout when creating a presentation or document.

FEATURES

Extensibility & Integrations

Kedro can be integrated with multiple industry leading solutions, including Apache Spark, **Pandas**, Dask, Matplotlib, Plotly, fsspec, Apache Airflow, **Jupyter Notebook** and **Docker**.

What features does Kedro have?




```yaml
# Load a Spark DataFrame on S3

flight_patterns:
  type: spark.SparkDataSet
  filepath: s3a://your_bucket/data/01_raw/flight_patterns*
  credentials: dev_s3
  file_format: csv
```

```yaml
# Save an image created with Matplotlib on Google Cloud Storage

results_plot:
  type: matplotlib.MatplotlibWriter
  filepath: gcs://your_bucket/data/08_results/plots/output_1.jpeg
  fs_args:
  project: my-project
  credentials: my_gcp_credentials
```

FEATURES

Pipeline Visualisations

Kedro's pipeline visualisation plugin shows a blueprint of your developing data and machine-learning workflows, provides **data lineage**, keeps track of machine-learning experiments and makes it easier to collaborate with business stakeholders.

FEATURES

Data Catalog

A series of lightweight **data connectors** used to save and load data across many different file formats and file systems.

Pillars of the GID MLOps approach

getindata