Machine Learning Operations (MLOps) Clase 3

Leticia Rodríguez

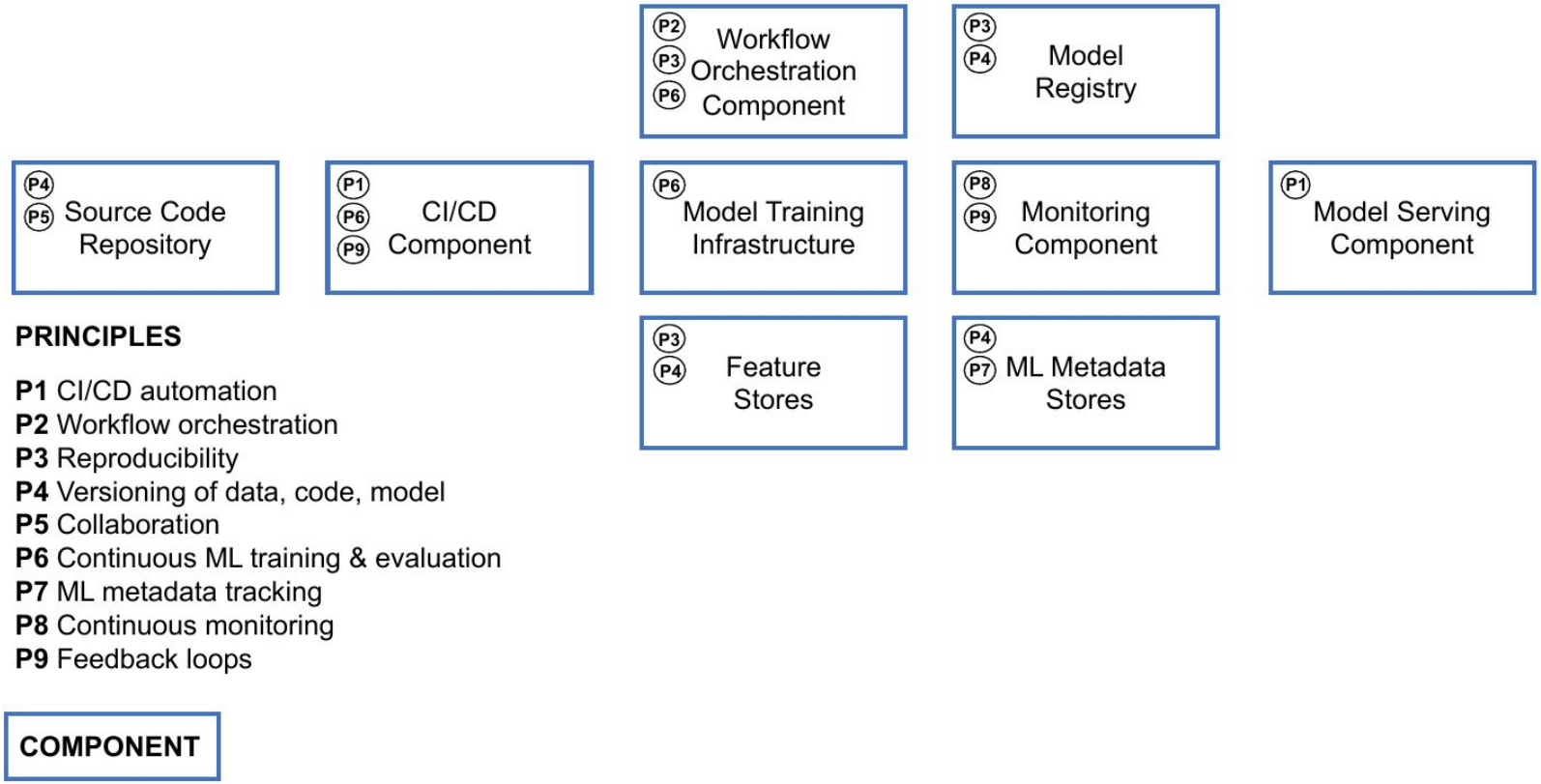
Septiembre 2024 - 2do Cuatrimestre - 4to. Bimestre

Universidad de Buenos Aires - FCEyN - Departamento de Computación

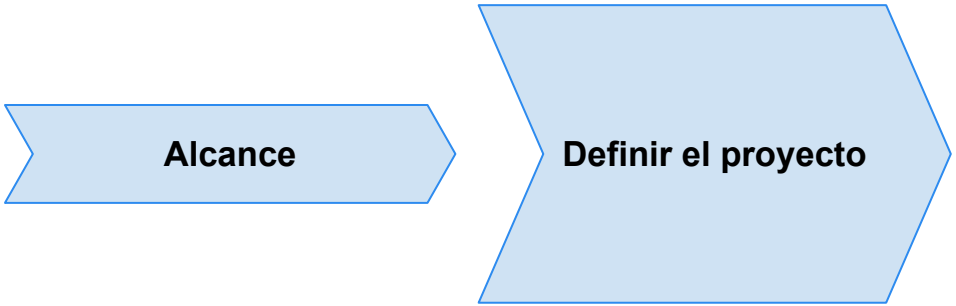
Encuesta y Asistencia

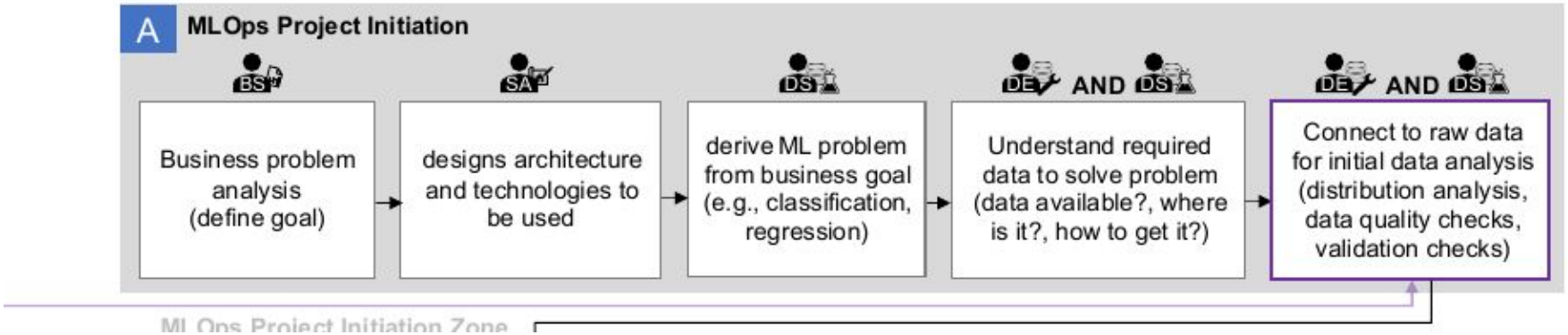
Kahoot de respaso

MLOps - Principios y Componentes

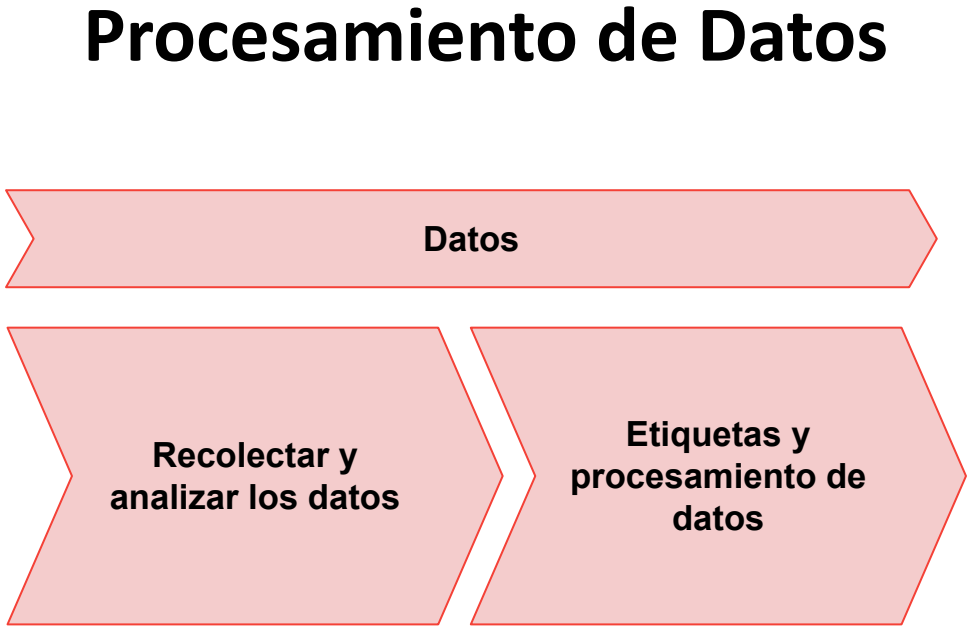


MLOps - La clase pasada



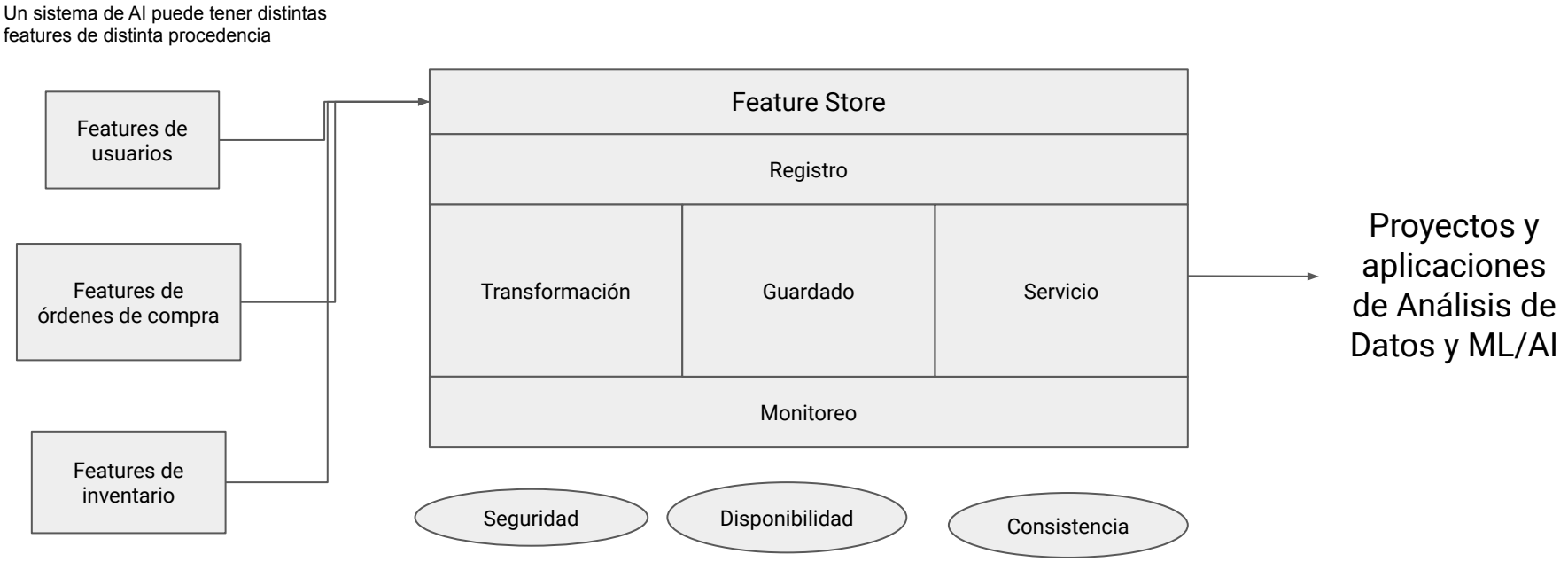


MLOps - La clase pasada y continuamos



Feature Store

- Constituyen una fuente de datos única para proyectos de Análisis de Datos y Aprendizaje Automático

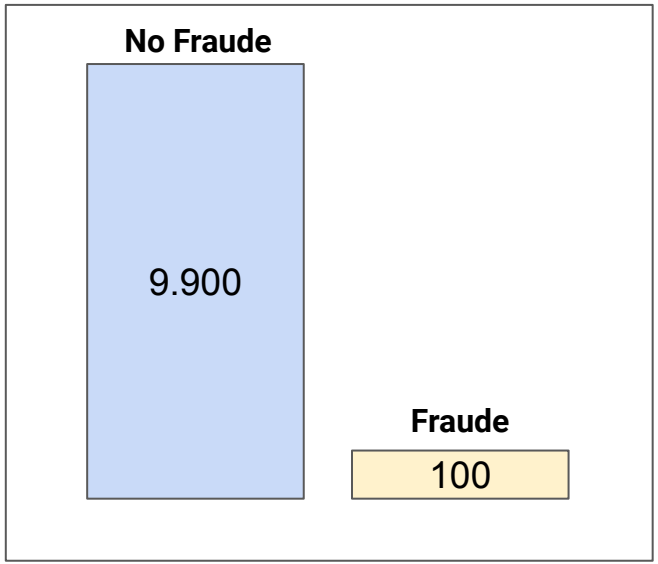


Actividad: Desbalanceo de Clases

En grupos analicen el paper, “Approaches to handle Data Imbalance Problem in Predictive Machine Learning Models: A Comprehensive Review” de Govind M. Poddar at el. siguiendo las consignas:

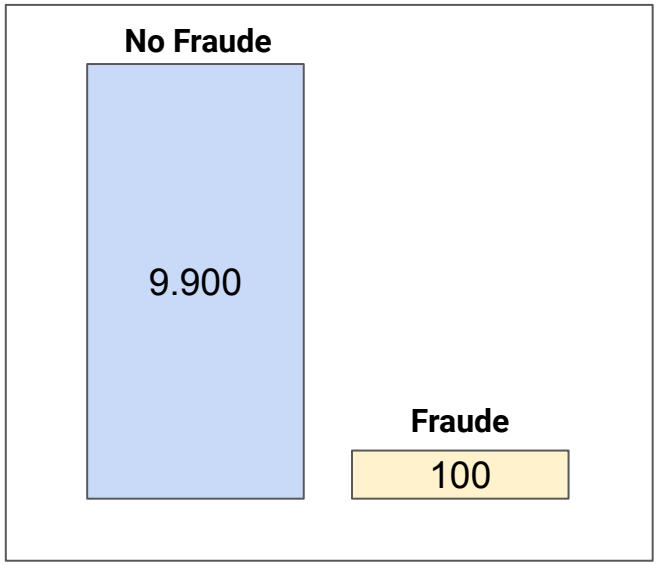
1. Lean y discutan la sección “3. Even Distribution Versus Uneven Distribution” que trata sobre el concepto de desbalanceo de clases. Compartan sus experiencias.
2. Lean la sección “6. Different approaches to tackle Class Imbalance Problem” sólo la introducción y luego, elijan al menos 3 métodos entre los que figuran en la tabla 1 de undersample, tabla 2 de oversample o algorithmic approaches. Los métodos de Ensamble los dejaremos para más adelante.
3. Por último, lean la sección 7: “7. Performance Evaluation Metrics for Class Imbalanced Datasets”
4. Compartan su trabajo con el resto de la clase.

Desbalanceo de Clases



- Si tenemos un modelo con **accuracy 99%,** podemos pensar que es muy bueno.
- Pero ese 99% de exactitud podría estar ubicado exclusivamente en detectar la clase No Fraude por lo cual, para datasets desbalanceados no es una buena métrica.
- Tenemos que consider además el recall.

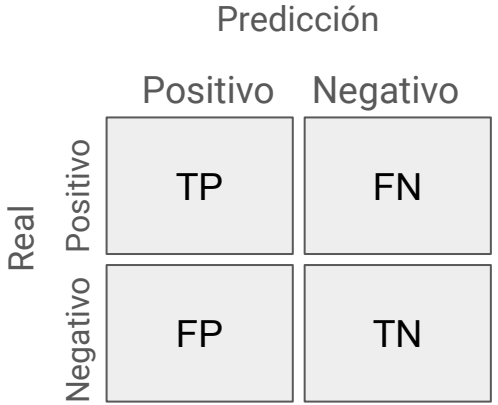
Desbalanceo de Clases



- Recall = TP / (TP+FN)
- Si tenemos un modelo con recall 99%, podemos pensar que es muy bueno.
- Pero ese 99% de recall no nos dice nada de los falsos positivos
- Entonces podría estar detectando los No Fraude como Fraude

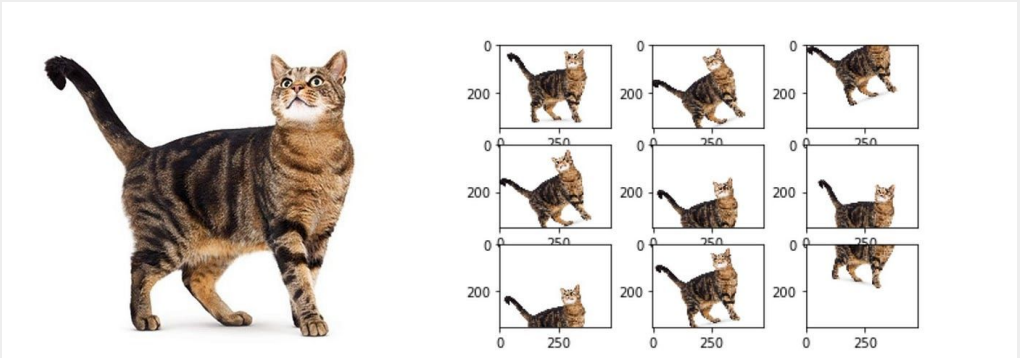
Desbalanceo de Clases: Posibles Soluciones

- **Elección de las métricas correctas**
  - Evaluar el tradeoff recall-precision y decidir el threshold acorde el negocio
  - **Precisión: TP / (TP+FP) Y Recall: TP / (TP+FN)**
  - Curva AUC PR
  - Matrices de confusión
- **Usar métodos a nivel de datos: Resampleo de Datos**
  - SMOTE - Undersampling - Oversampling
- **Usar métodos a nivel de algoritmos**
  - Cost-sensitive learning
  - Class-balanced loss
  - Focal loss



Data Augmentation

- Distintas técnicas que permiten incrementar la cantidad de datos de entrenamiento y mejoran el aprendizaje con bajo costo. A veces, este incremento se hace desde los mismos datos de entrenamiento actuales.
- Se volvió fundamental para la creación de modelos en Computer Vision, y recientemente en NLP.
- Depende altamente del formato de los datos, es distinta para texto e imágenes.



Simple Label-Preserving Transformations

Perturbation:
Agregar noise (ruido) como ejemplo adversario

Data Synthesis:
Generar data sintética

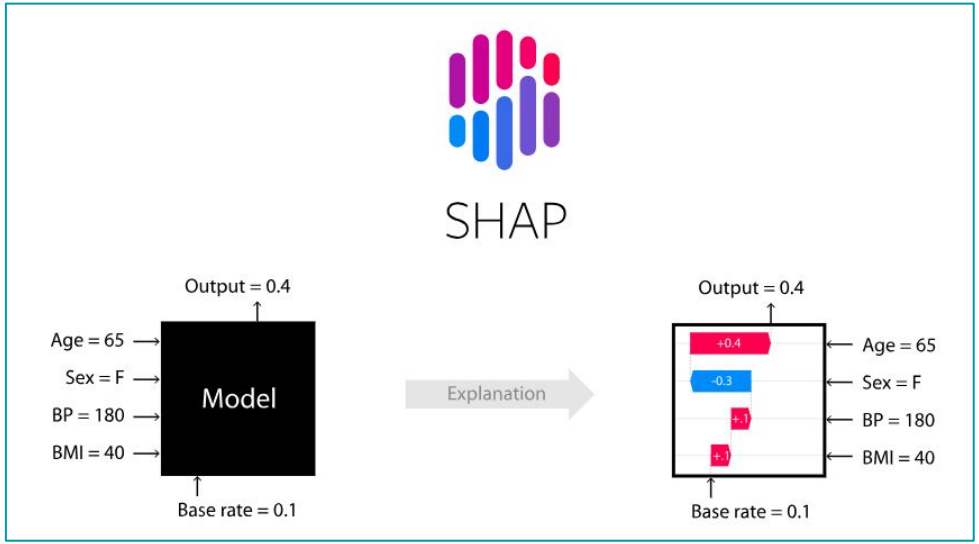
Simple Label-Preserving Transformations:
Aplicar transformaciones (rotaciones, mover, flip, crop) a las imagenes del training set.

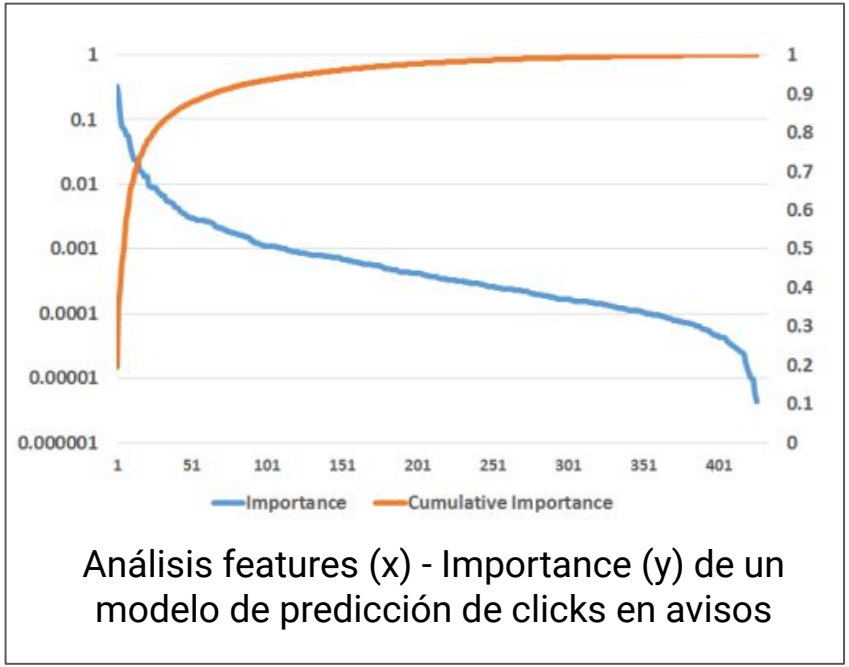
Data Leakage

- Se llama Data Leakage al fenómeno en el cual los labels (etiquetas) de un aprendizaje supervisado se cuelan de alguna forma dentro de los features haciendo que el modelo a entrenar tenga el sesgo de ese dato para predecir.
- Consecuencias:
  - Performance demasiado optimista
  - Generalización pobre
  - Conocimiento engañoso
- Causas:
  - **Distribuir los datos de forma aleatorio y no por tiempo**
  - **Scalar antes de dividir los sets**: error común es usar toda la data para sacar las estadísticas (media, varianza) para escalar. Solución: tomar la estadística en training y usarla para escalar todos los splits.
  - **Llenar datos faltantes con estadísticas del split de testeo.**
  - **Manejo pobre de los datos duplicados antes de dividir los sets**: sacarlos antes de dividir los sets
  - **Group leakage**: datos similares del mismo grupo con diferencia de tiempo o minima cae uno en testeo y otro en training.
  - **Leakeage derivado de la generación de datos**
- Formas de detectarlo:
  - Estudiar la correlación entre label y features
  - Hacer pruebas sacando algún feature y viendo si degrada demasiado la performance del modelo
  - Estar atento a los features que se van agregando al modelo

IA Explicable

- Feature Importance: es una técnica que nos dice mediante un score que tan importante es un feature en relación a otros features para el modelo.
- Algunas emplementaciones de árboles como XGBoost traen la funcionalidad implementada
- SHAP (SHarply Additive exPlanations) es agnostico al modelo y no solo mide la importancia, sino también la contribución en una predicción

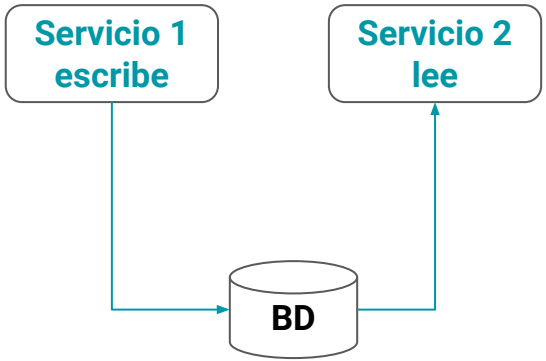




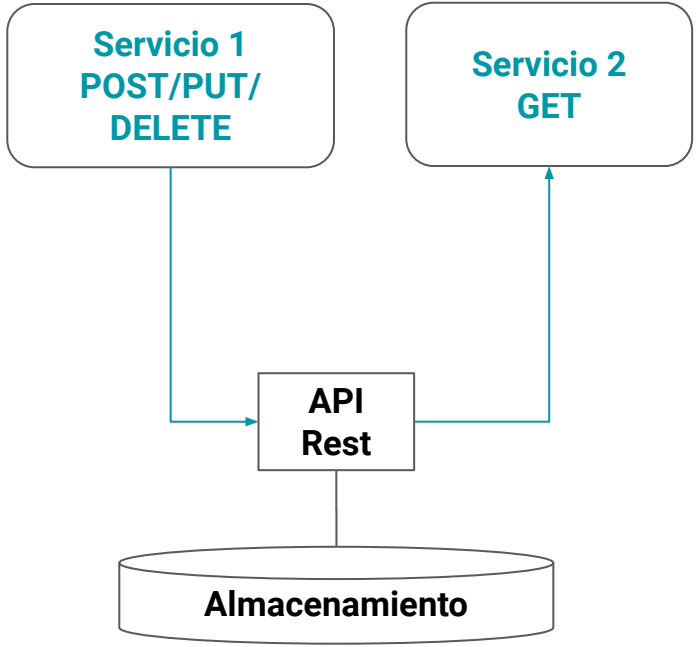
Análisis features (x) - Importance (y) de un modelo de predicción de clicks en avisos

Flujo de los Datos (DataFlow)

Pasaje a través de Base de Datos

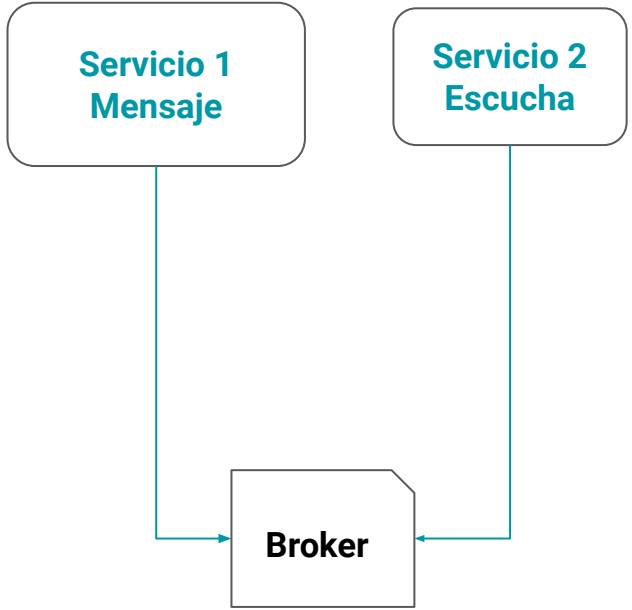


Pasaje a través de Sevicios



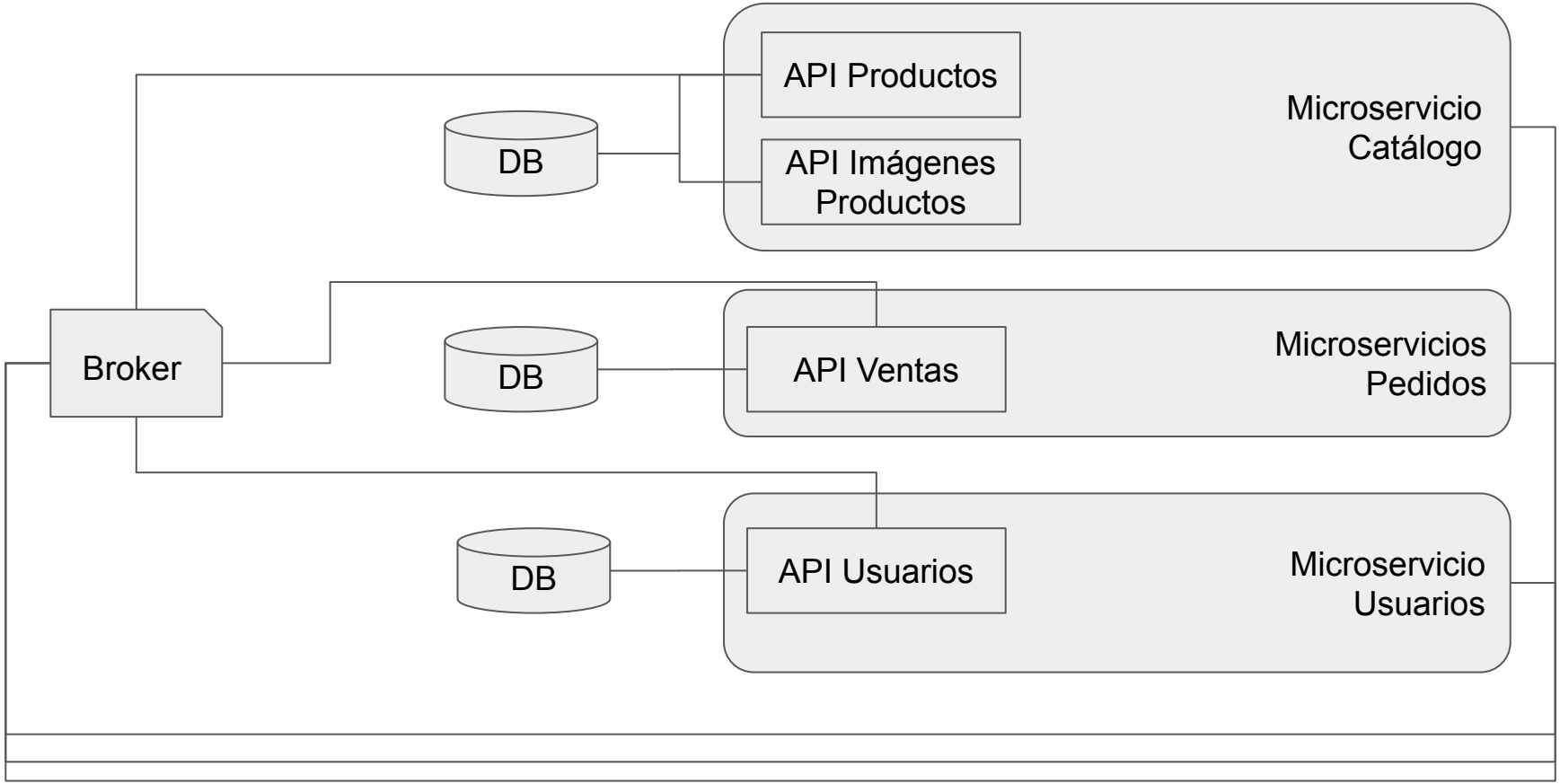
El servicio puede ser RPC u otro

Pasaje a través de un transporte de tiempo real



Kafka, RabbitMQ ejemplos

Flujo de Datos y Microservicios - Ejemplo



Arquitectura simplificada que combina dos tipos de pasajes de datos

Data Pipelines

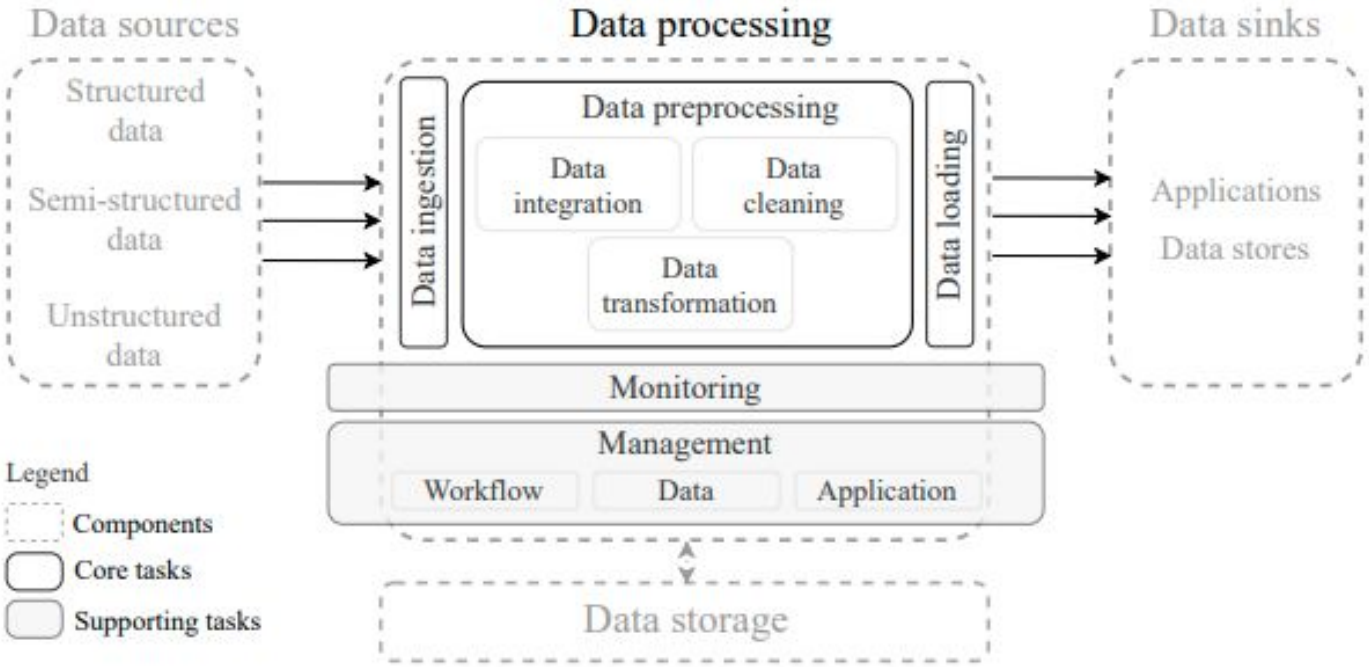


Figure 1: High-level data pipeline architecture

Data Pipeline Quality: Influencing Factors, Root Causes of Data-related Issues, and Processing Problem Areas for Developers

Actividad: Lectura del paper de Data Pipelines

En grupos, leamos la sección 1. Introduction. Luego, responder:

1. ¿Qué es un data pipeline?
2. ¿Cuáles son los desafíos de estos pipelines en producción?
3. De la sección 4 Use Cases, elijan 2 de los pipelines en alguna empresa A,B,C y comentenlos en el grupo.
4. De la sección 5 Challenges to Data Pipeline Management lean 1 de las 3 secciones: Infraestructure, Organizational or Data Quality y coméntenla entre uds.
5. De la sección 6 Opportunities selección 3 oportunidades y coméntenlas entre uds.

Todos juntos revisemos lo más interesante que leyó cada grupo

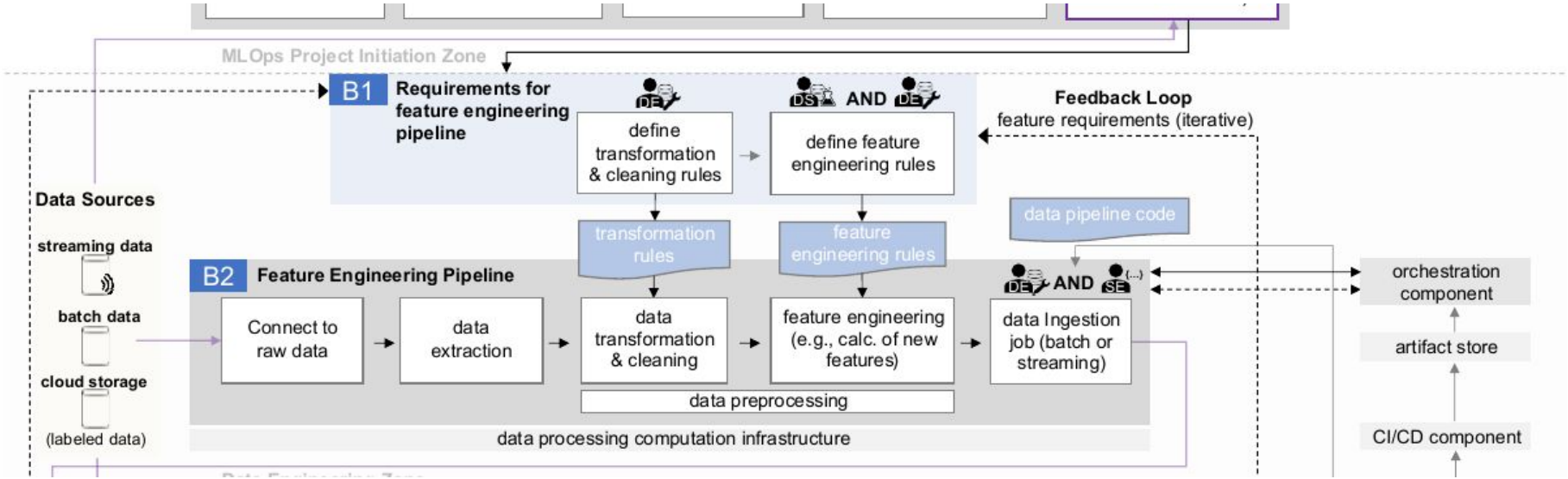
Batch vs. Online

Existen dos formas de procesamiento de datos en forma Batch u Online.

- Online: procesar los datos al momento que llegan
  - requerimientos de infraestructura para la variabilidad de la cantidad los pedidos
  - servicios de mensajería: Kafka, RabbitMQ, servicios cloud para este fin.

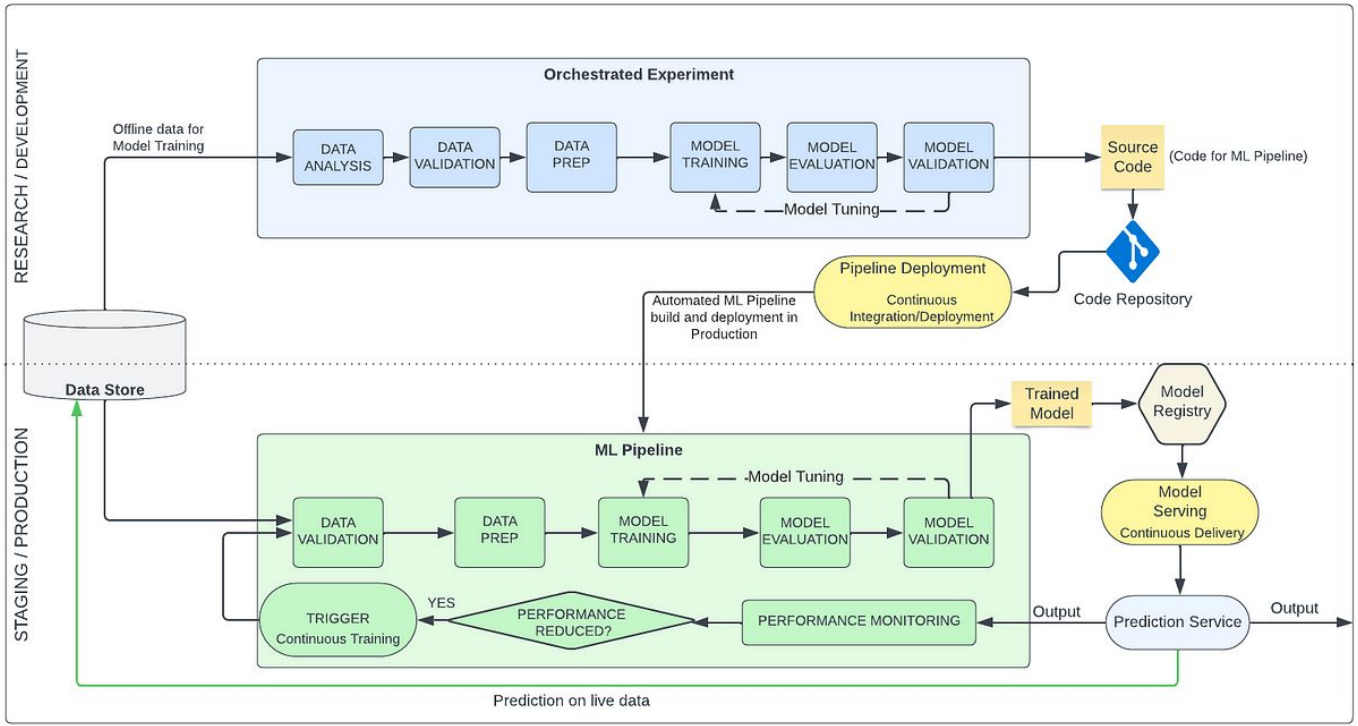
- Batch: procesar los datos en lote
  - procesos programados que corren en horarios o frecuencia determinada
  - procesan grandes cantidades de datos
  - tener en cuenta temas lockeo de datos
  - infraestructura escalable acorde la cantidad de datos a procesar

Pipeline de Datos en el paper de MLOps



Feature Store

Creando un pipeline de datos para ML



Metadatos y traceo de los datos

- Data Lineage: es la técnica de traquear el origen de los datos y sus etiquetas.
- Los datos de un dataset pueden tener diverso origen y haber pasado por diversos procesos de equitado
- El etiquetado puede requerir contratar equipos externos que manualmente lo realizan o mediante diferentes algoritmos y técnicas automáticas
- Llevar registro del origen de los datos y qué datos se usaron para entrenar a que modelos es fundamental para cuando haya que buscar evidencia sobre errores o problemas e incluso para entender comportamientos del modelo.
- También la data no estructurada puede ir acompañada de metadata estructurada que de más información sobre el origen de los datos y simplique la construcción de modelos (ej. audios, con metadata que indique duración, interlocutores u otros tags)
- Esto lleva también almacenamiento estructurado adicional

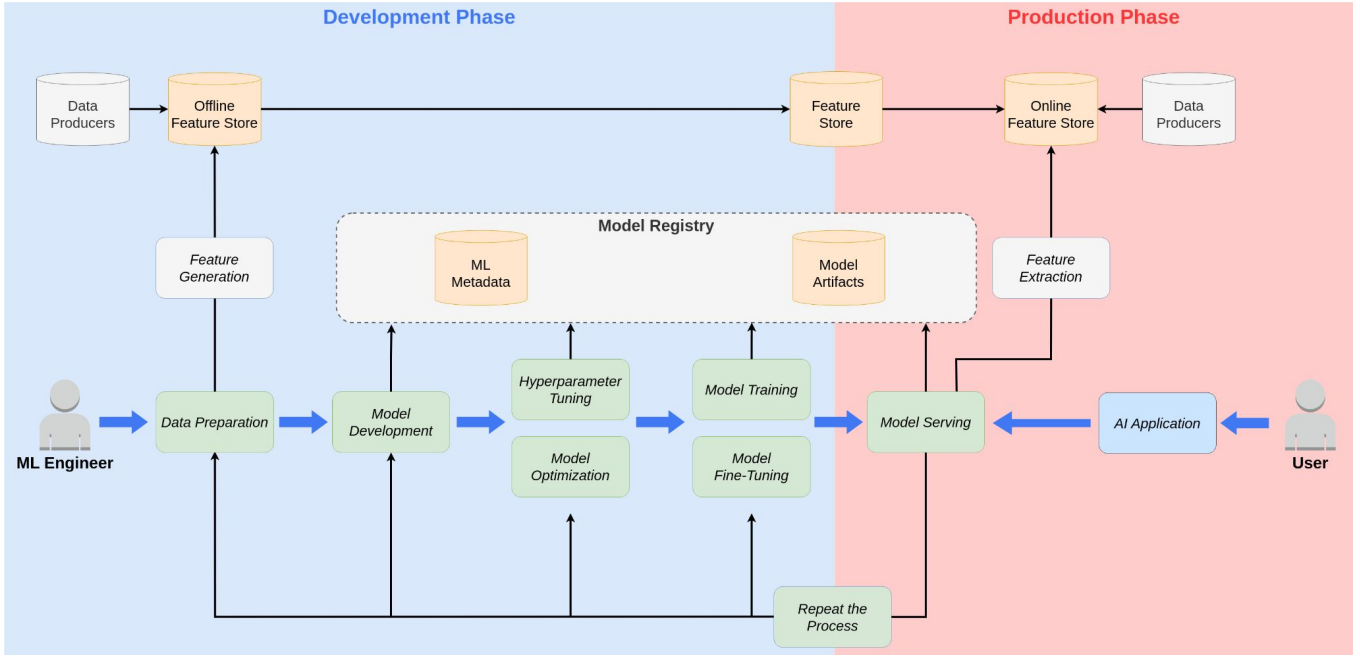
Los cambios en los datos - Drifts

- Imaginemos un sistema de ecomerce que funciona hace 10 años… sus datos cambiaron en dos dimensiones
  - Contenido. Por ejemplo, podria haber más usuarios
  - Schema - Esquema . Por ejemplo, recienmente se agregó un campo para especificar un link a redes sociales
- A su vez, un mismo campo de datos puede tomar valores distintos o nuevo, por ejemplo, medio de pago tradicionalmente era contado, tarjeta y agrega billetera electrónica
- También, podría haber cambios en la distribución de los datos. Anteriormente, las transacciones eran en su mayoría al contado, hoy la mayor parte son con billetera electrónica.
- Podría haber cambio en la significacia del dato (esto sería más raro)
- Incluso nuevos requerimientos del lado del software puedan agregar datos o features nuevas que serían de utilidad
- **Los datos y su significado cambian a través del tiempo**

Herramientas opensource para Pipelines de ML

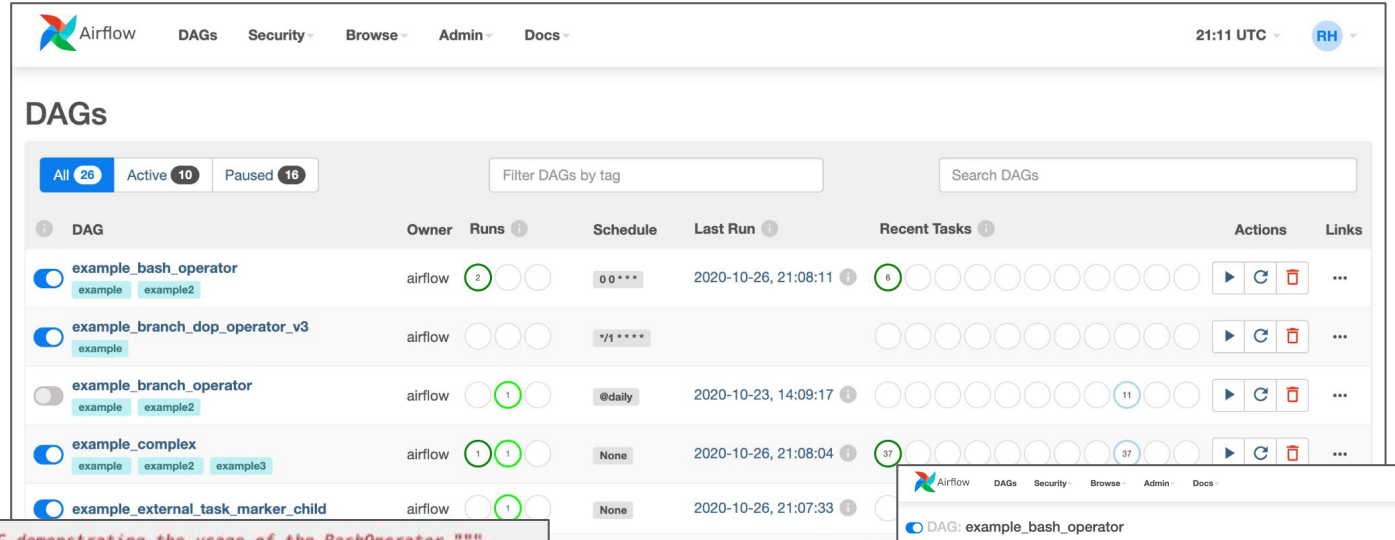
Existen distintas herramientas para armar flujos de datos y de Machine Learning
Las más populares son:

- Apache Airflow
- Kubeflow

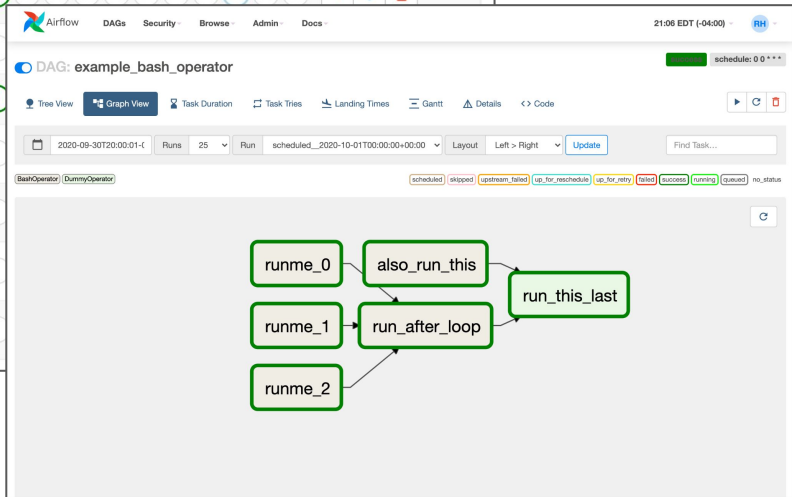


De la documentación de Kubeflow

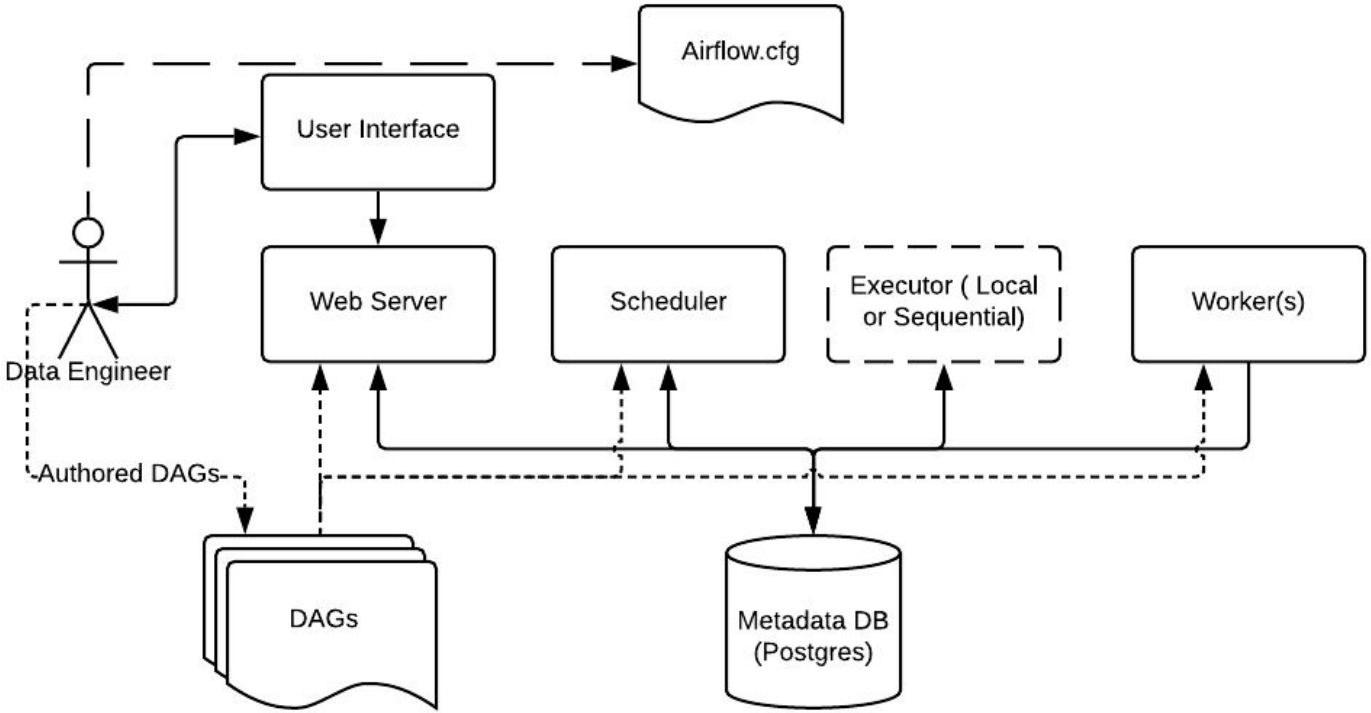
Apache Airflow




```
"""Example DAG demonstrating the usage of the BashOperator."""

from datetime import timedelta

from airflow import DAG
from airflow.operators.bash import BashOperator
from airflow.operators.dummy_operator import DummyOperator
from airflow.utils.dates import days_ago

args = {
    'owner': 'airflow',
}

dag = DAG(
    dag_id='example_bash_operator',
    default_args=args,
    schedule_interval='0 0 * * *',
```



Apache Airflow

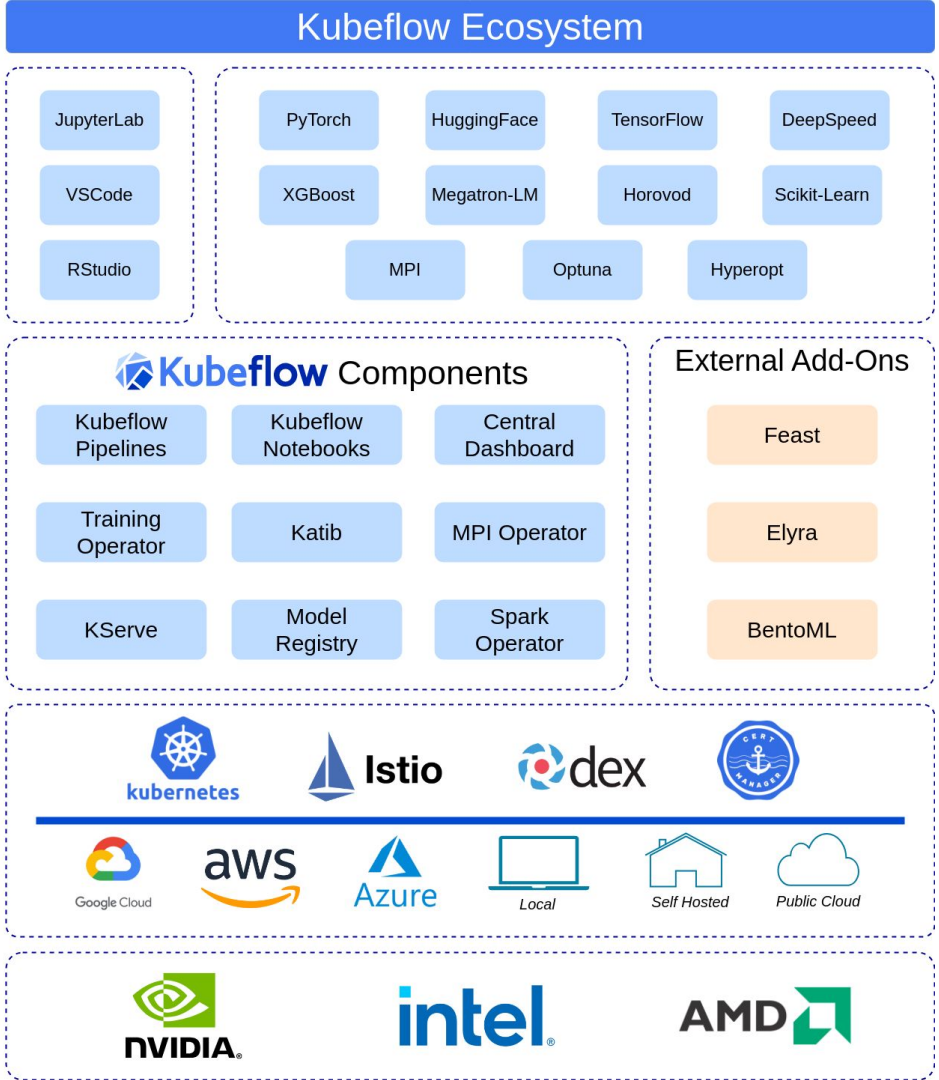


https://airflow.apache.org/docs/apache-airflow/2.0.1/concepts.html

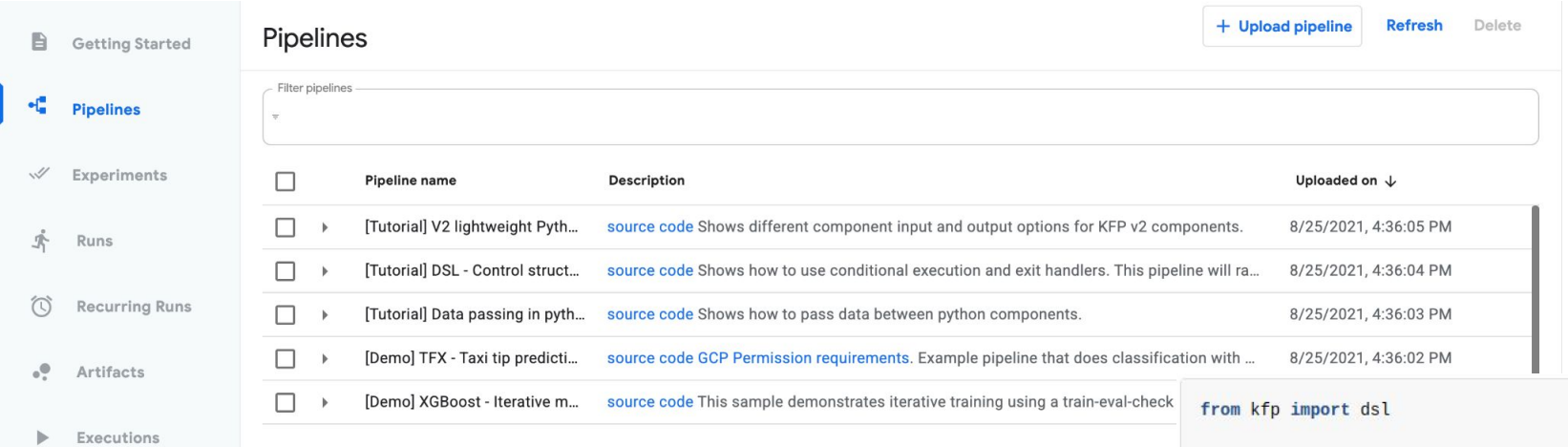
Kubeflow

Kubeflow Pipelines (KFP) es una plataforma para crear y deployar flujos de Machine Learning

Un pipeline es la definición de un flujo de datos con uno o más componentes formando un grafo.
Esta definición se ejecuta en un **contenedor** y puede preprocesar datos hasta crear modelo de Machine Learning.



Kubeflow - Pipelines

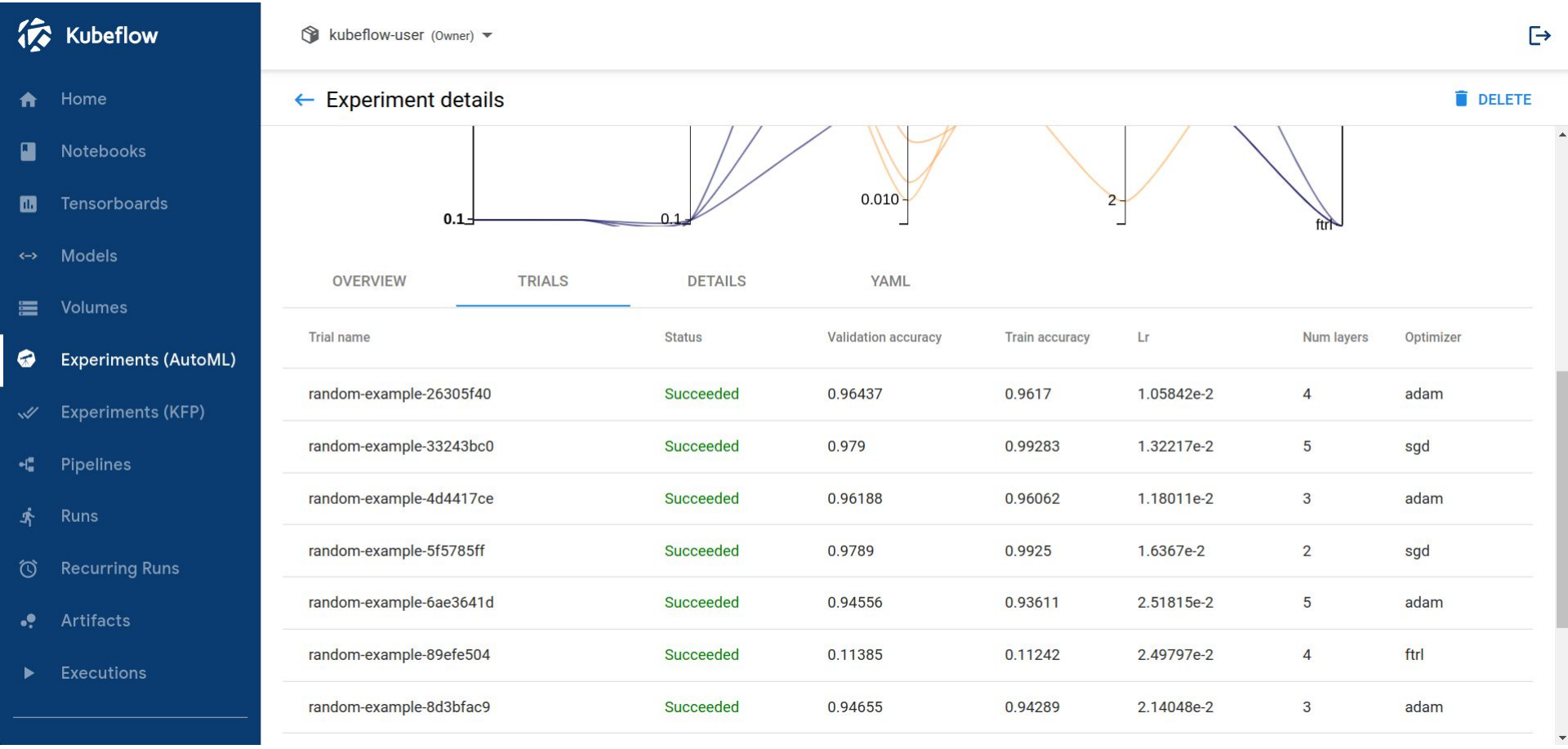



```python
from kfp import dsl

@dsl.component
def say_hello(name: str) -> str:
    hello_text = f'Hello, {name}!'
    print(hello_text)
    return hello_text

@dsl.pipeline
def hello_pipeline(recipient: str) -> str:
    hello_task = say_hello(name=recipient)
    return hello_task.output
```

You can compile the pipeline to YAML with the KFP SDK DSL `Compiler`:

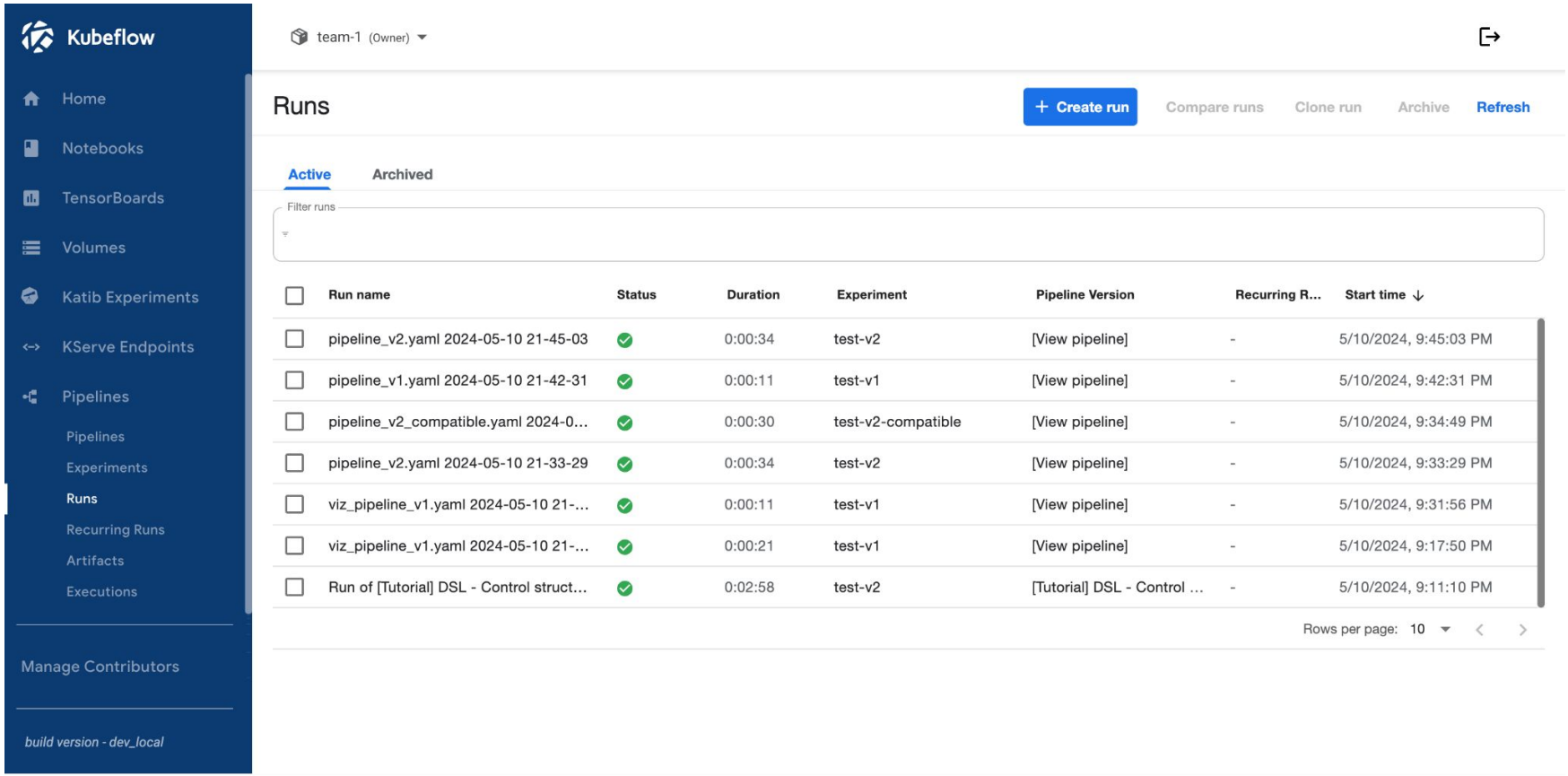
```python
from kfp import compiler
```

Kubeflow - Experiments

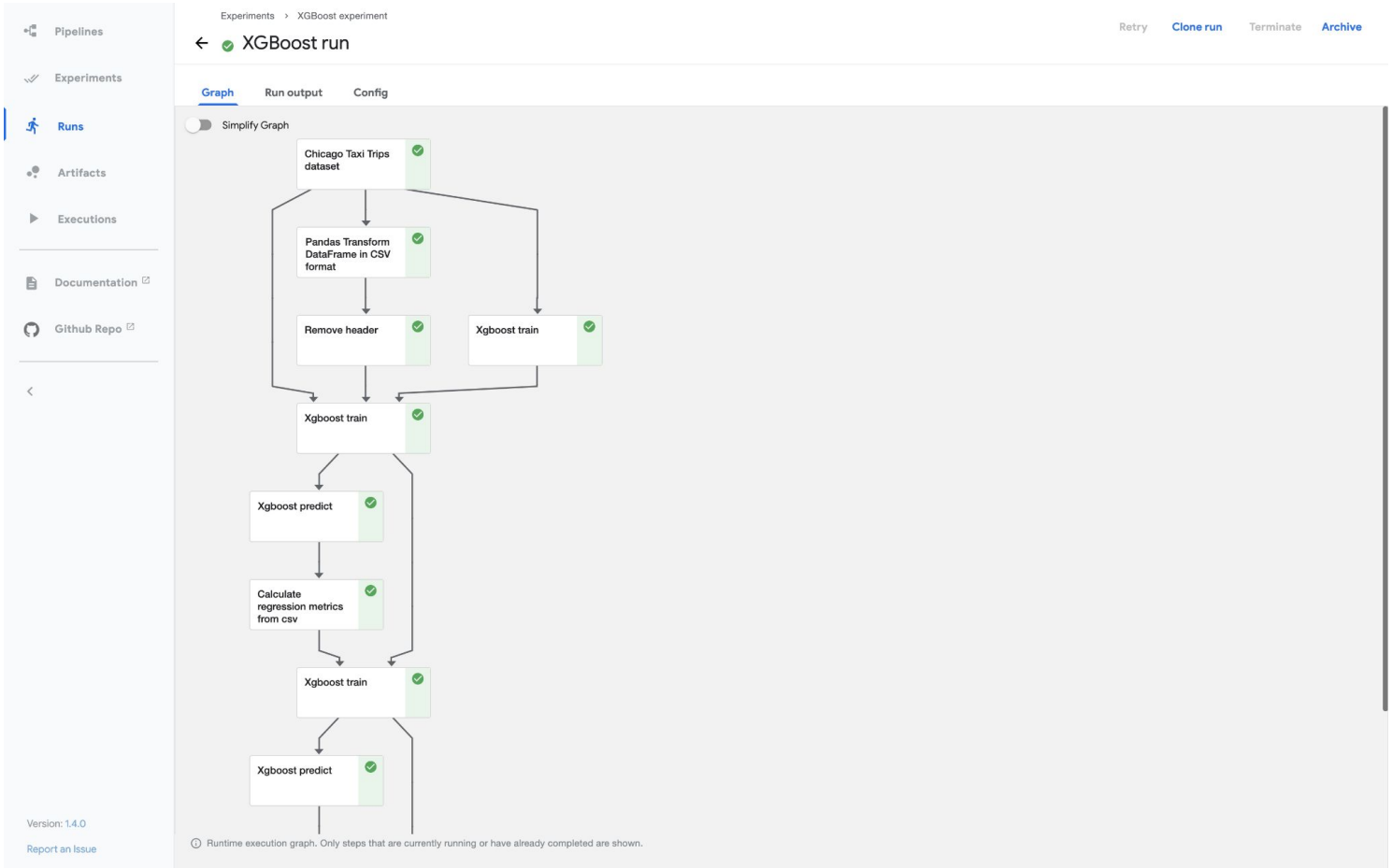


Trial name	Status	Validation accuracy	Train accuracy	Lr	Num layers	Optimizer
random-example-26305f40	Succeeded	0.96437	0.9617	1.05842e-2	4	adam
random-example-33243bc0	Succeeded	0.979	0.99283	1.32217e-2	5	sgd
random-example-4d4417ce	Succeeded	0.96188	0.96062	1.18011e-2	3	adam
random-example-5f5785ff	Succeeded	0.9789	0.9925	1.6367e-2	2	sgd
random-example-6ae3641d	Succeeded	0.94556	0.93611	2.51815e-2	5	adam
random-example-89efe504	Succeeded	0.11385	0.11242	2.49797e-2	4	ftrl
random-example-8d3bfac9	Succeeded	0.94655	0.94289	2.14048e-2	3	adam

Kubeflow - Runs

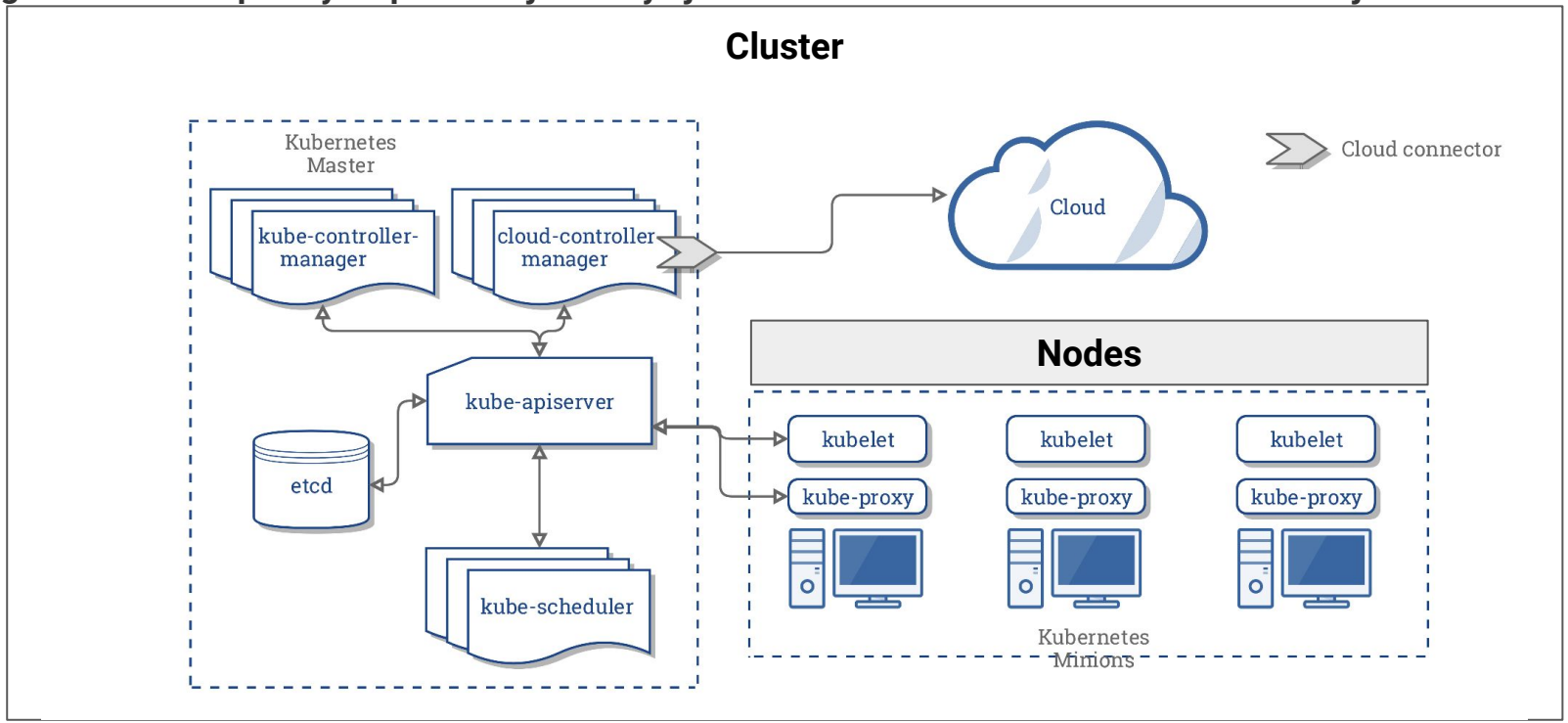


Kubeflow - Runs



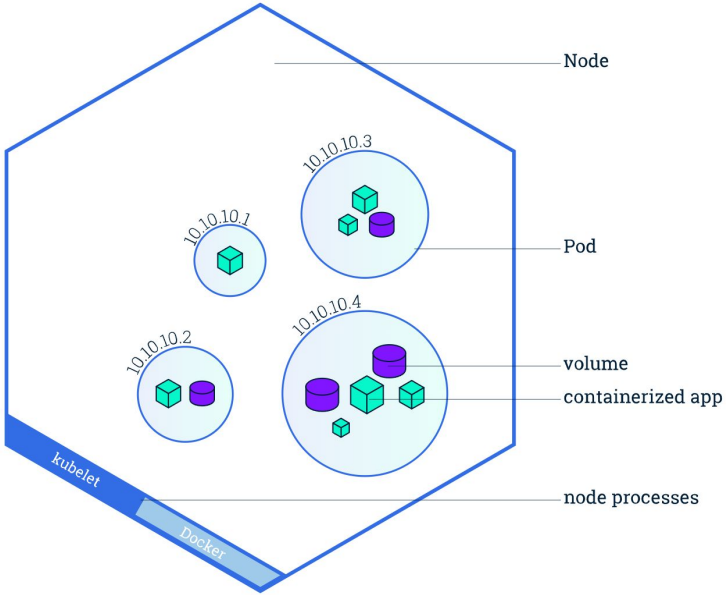
Kubernetes

Kubernetes es un software de código abierto que le permite implementar y administrar aplicaciones en contenedores a escala. Kubernetes administra un **clúster** y **programa contenedores** para que se ejecuten en el clúster en función de los recursos informáticos disponibles y de los requisitos de recursos de cada contenedor. Los **contenedores se ejecutan en agrupaciones lógicas llamadas pods y es posible ejecutar y ajustar la escala de uno o más contenedores juntos como un pod.**

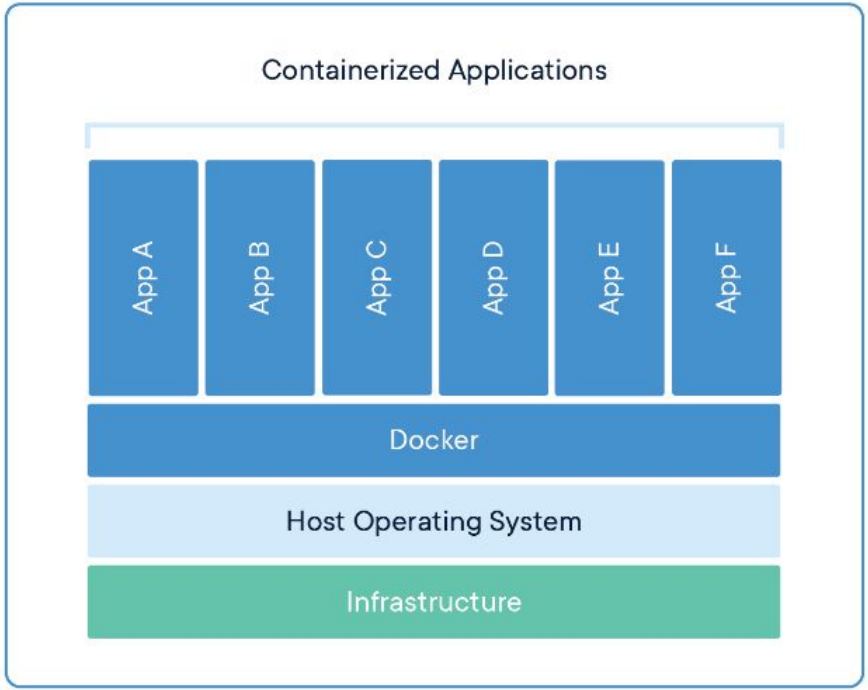


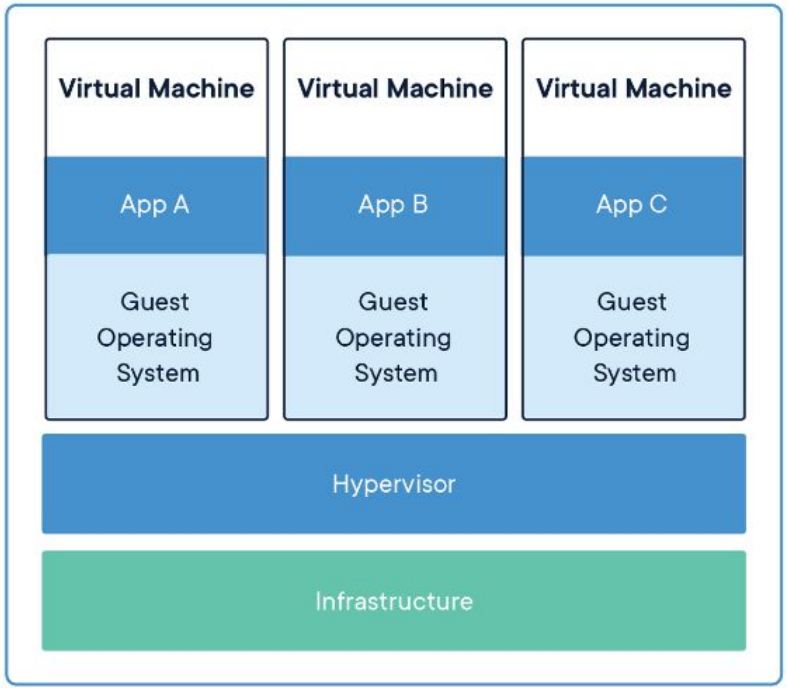
Kubernetes - Pods

Un Pod es un grupo de uno o más contenedores, con recursos de red y almacenamiento compartidos, y una especificación sobre cómo ejecutar los contenedores.
El contenido de un Pod siempre está ubicado y programado en el mismo lugar, y se ejecuta en un contexto compartido. En contextos que no son de nube, las aplicaciones ejecutadas en la misma máquina física o virtual son análogas a las aplicaciones de nube ejecutadas en el mismo host lógico.



Docker

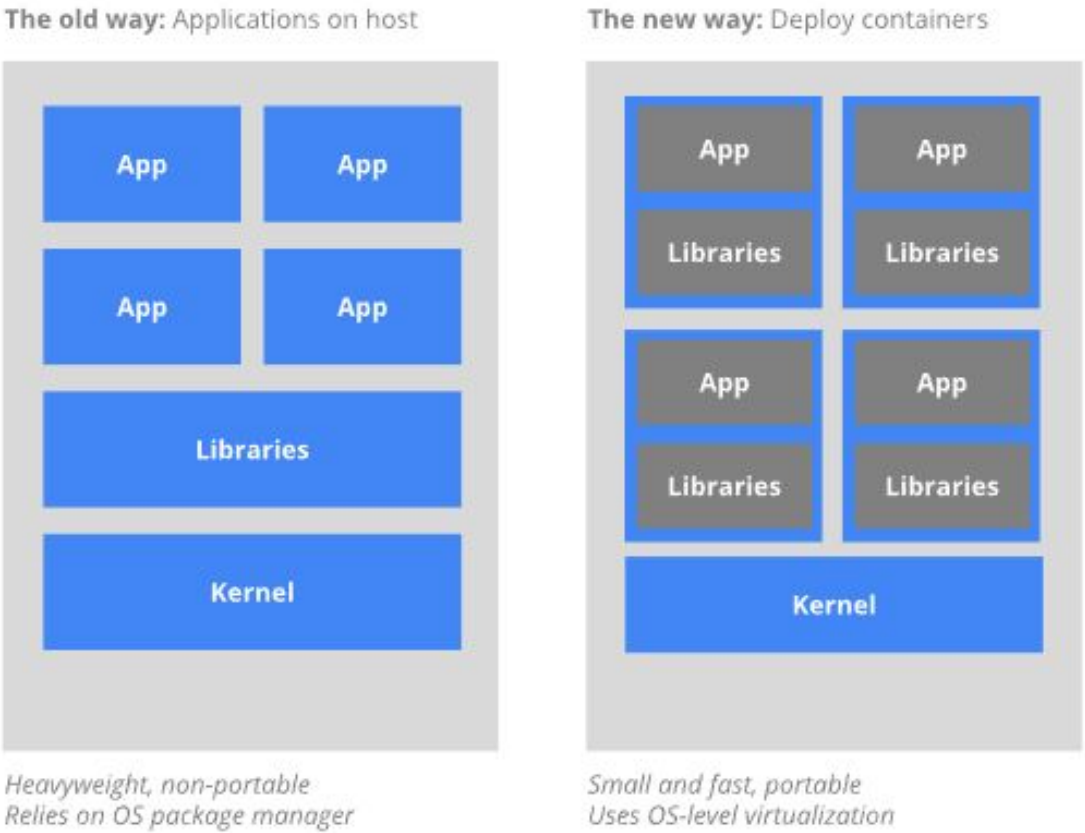




El contenedor es una abstracción en la capa de aplicación. Varias aplicaciones corren en una misma máquina compartiendo el kernel del sistema operativo. La imagen de un contendor es más pequeña, MBs.

Máquina Virtual es una abstracción que se crea sobre una misma infraestructura que contiene el sistema operativo y diferentes aplicaciones.

Contenedores



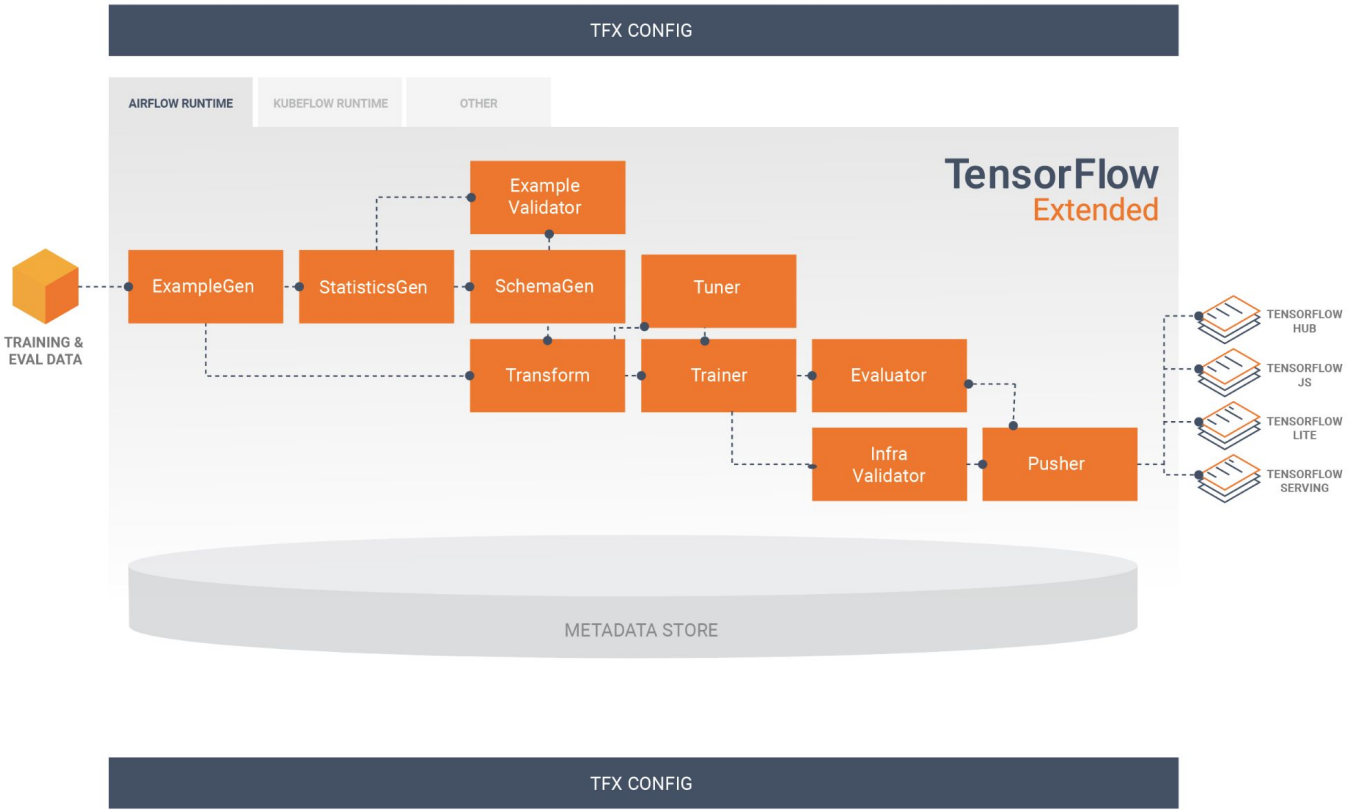
Virtualización a nivel del sistema operativo, en vez del hardware.

Estos contenedores están aislados entre ellos y con el servidor anfitrión: tienen sus propios sistemas de archivos, no ven los procesos de los demás y el uso de recursos puede ser limitado. Son más fáciles de construir que una máquina virtual, y porque no están acoplados a la infraestructura y sistema de archivos del anfitrión, pueden llevarse entre nubes y distribuciones de sistema operativo.

Generar una imagen de contenedor al momento de la compilación permite tener un entorno consistente que va desde desarrollo hasta producción.

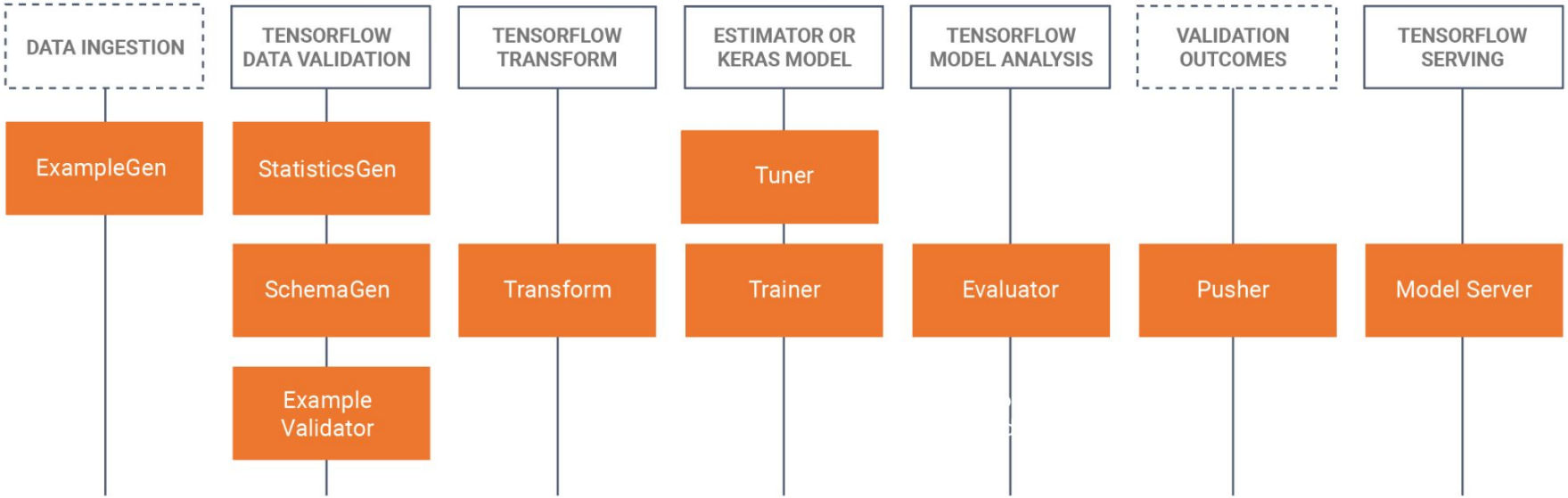
TFX - Tensorflow Extended

Herramienta para crear y gestionar flujos de MLOps.- Basado en Tensorflow.



TFX - Tensorflow Extended

Posee distintos componentes para el armado de flujos de Machine Learning



TFX utiliza Apache Beam para implementar procesamiento de datos en paralelo.
Opcional: Orquestadores como Apache Airflow y Kubeflow facilitan la configuración, operación, monitoreo y mantenimiento .
Portatil y diseñado para multiples entornos

Material Recomendado de esta semana

Most Popular Feature Stores In 2023

Chip Huyen, Designing Machine Learning Systems, 2022 - Chapter 4: Training Data - Class Imbalance - Pág. 102-112

Chip Huyen, Designing Machine Learning Systems, 2022 - Chapter 4: Training Data - Data Augmentation - Pág. 113-117

Kubernetes Tutorial

Otros Recursos

FEAST - Feature store opensource

SHAP Interpretable ML Book

Scott M. Lundberg et. al., A Unified Approach to Interpreting Model Predictions, 2017

SHAP Project API

Kubeflow Arquitectura

Apache Airflow Arquitectura

Docker que es un container

¿Qué es Kubernetes K8s?

Borg: The Predecessor to Kubernetes

TensorFlow Extended TFX