Tecnologías Bonus

Máster en IA, Cloud Computing y DevOps

Módulo: Machine Learning y Deep Learning

Contenido

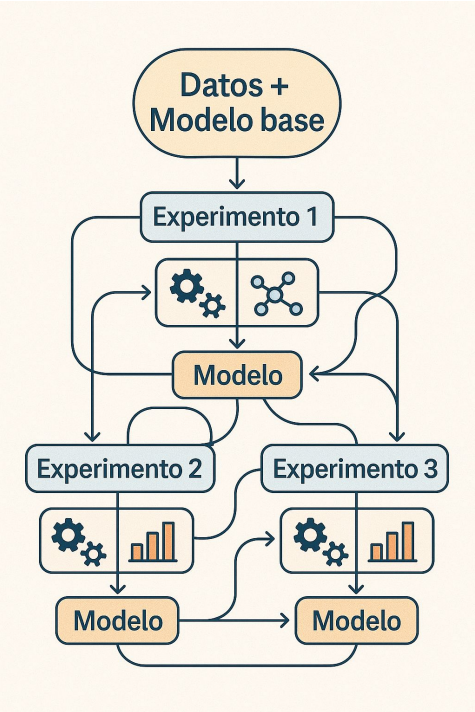
01. MLflow y el trackeo de experimentos
02. Machine Learning low-code: PyCaret

01. La dificultad del entrenamiento iterativo de modelos

MLflow y el trackeo de experimentos

Dentro del entrenamiento de modelos de IA, la gestión de las diferentes iteraciones es compleja, dado que se comienza partiendo de una primera prueba que acaba teniendo infinidad de variantes.

Además, la gestión tanto de los cambios hechos en los datos, como los hiper-parámetros usado en los diferentes modelos introduce complejidad en la gestión del flujo de trabajo.

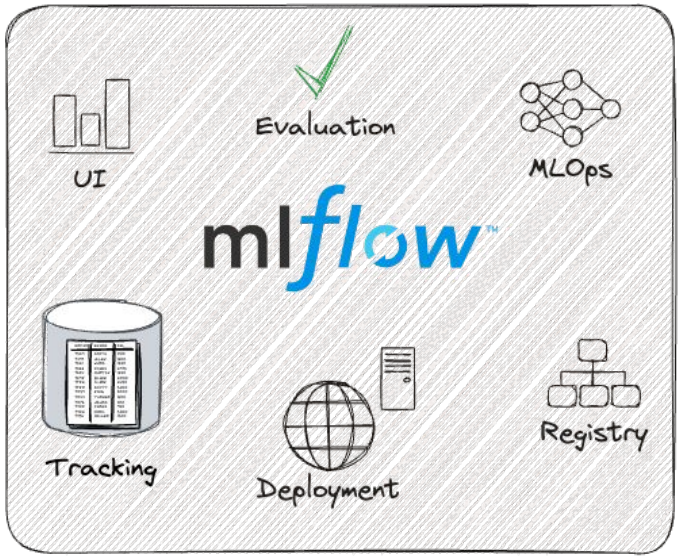


01. MLflow: Herramienta para gestionar el ciclo de vida de ML

MLflow y el trackeo de experimentos

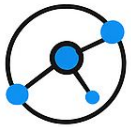
MLflow es una plataforma de código abierto que nació con el objetivo es facilitar la **gestión del ciclo de vida de los modelos** de Machine Learning. Bien es cierto que no se han quedado a atrás y han integrado funcionalidades específicas para poder trabajar con GenAI.

Además, permite trabajar con diferentes lenguajes y frameworks además de Python como R o Java.



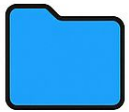
01. Módulos de MLflow

MLflow y el trackeo de experimentos



Tracking

Registro de experimentos incluyendo código, parámetros y configuración



Projects

Paquetes del código de modelos de una forma reutilizable

Models

Gestión del despliegue de los modelos de diferentes tipos de librerías

Registry

Repositorio central que permite manejar el ciclo de vida de los modelos y su versión

01. Ventajas y desventajas

MLflow y el trackeo de experimentos

Ventajas	Desventajas
Reproducibilidad MLflow permite registrar detalles como el código, datos, hiper-parámetros o métricas de cada modelo	**Complejidad inicial** A pesar de ser una herramienta potente, tiene muchas funcionalidades que pueden resultar complicadas de entender en un inicio
Escalabilidad Utiliza diversas librerías (Python, R o Java), herramientas de despliegue o frameworks comunes en la industria (Databricks)	**Instalación y mantenimiento** No a nivel de un usuario individual sino a nivel de equipos u organizaciones, si no se utiliza un entorno ya gestionado como Databricks, no es sencillo de configurar
Flexibilidad Soporta diversos lenguajes y frameworks de programación	**Complejidad para proyectos pequeños** Quizás en proyectos de poca envergadura el balance entre su configuración frente a la eficiencia para poder tener resultados rápido no es positivo

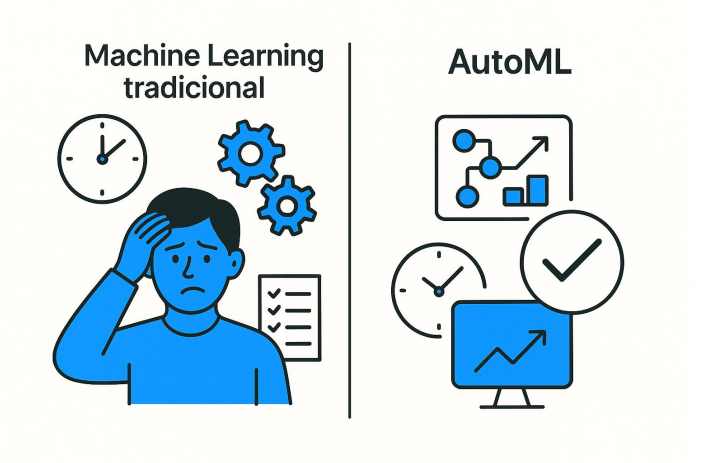
02. Qué es el AutoML

Machine Learning low-code: PyCaret

Gracias a la automatización de tareas repetitivas de escritura de código, con muy poco código (low-code) y a través de interfaces en muchas ocasiones es posible reducir la barrera de entrada así como agilizar enormemente las etapas del ciclo de vida de ML.

Algunas de esos procesos que agiliza son:

- Preprocesamiento e ingeniería de los datos
- Realizar Análisis Exploratorio de los Datos (EDA)
- Selección automática de los algoritmos más apropiados al problema
- Ajuste automático de hiperparámetros
- Evaluación y comparación de múltiples modelos
- Despliegue y puesta en producción del mejor modelo



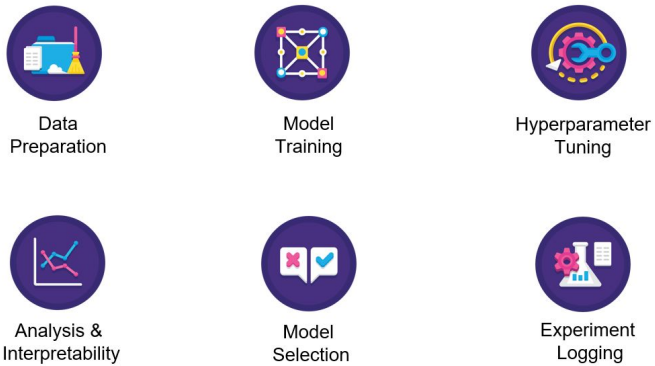
02. Qué es el PyCaret

Machine Learning low-code: PyCaret

PyCaret es una librería de código abierto de Python que gracias a la implementación de AutoML permite trabajar con modelos de regresión, clasificación, clustering, series temporales y detección de anomalías. En el pasado también NLP pero lo deprecaron por bajo uso.

Algunas sus características son:

- Se necesitan muy pocas líneas de código para los procesos básicos de entrenamiento e inferencia de un modelo
- Integra librerías populares como scikit-learn, XGBoost, LightGBM, etc
- Permite realizar EDA de manera automática
- Implementa funcionalidades de MLOps para poder desplegar el modelo con Docker o servirlo a través de una API mediante FastAPI



02. Ciclo de vida de un modelo

Machine Learning low-code: PyCaret

```python
# Classification OOP API Example

# loading sample dataset
from pycaret.datasets import get_data
data = get_data('juice')

# init setup
from pycaret.classification import ClassificationExperiment
s = ClassificationExperiment()
s.setup(data, target = 'Purchase', session_id = 123)

# model training and selection
best = s.compare_models()

# evaluate trained model
s.evaluate_model(best)

# predict on hold-out/test set
pred_holdout = s.predict_model(best)

# predict on new data
new_data = data.copy().drop('Purchase', axis = 1)
predictions = s.predict_model(best, data = new_data)

# save model
s.save_model(best, 'best_pipeline')
```

Gracias

Sergio Benito Martín

www.pontia.tech

pontia@sergiobenito.com

99. Links útiles

Anexo

- MLflow:
  - Página oficial - Quickstart: MLflow Tracking Quickstart
  - Tutoriales externos MLflow:
    - A Comprehensive Guide to MLflow: What It Is, Its Pros and Cons, and How to Use It in Your Python Projects | by Anna | Medium
    - Experiment Tracking with MLflow in 10 Minutes | Towards Data Science
    - MLOps: How MLflow effortlessly tracks your experiments and helps you compare them? | by LittleBigCode
- PyCaret
  - Página oficial - Quickstart: https://pycaret.gitbook.io/docs/get-started/quickstart
  - Repositorio de Github: GitHub - pycaret/pycaret: An open-source, low-code machine learning library in Python