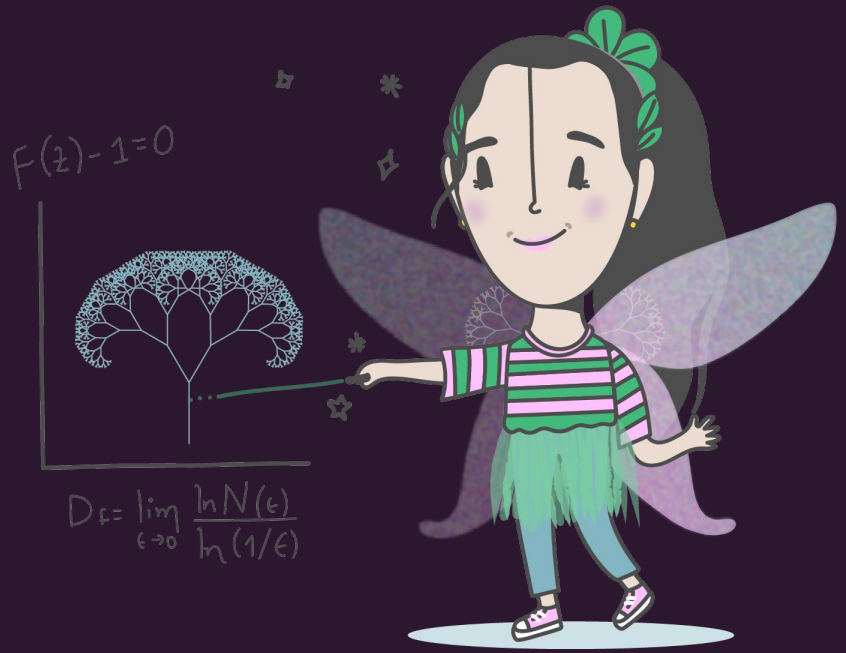
Navegando en Contenedores: IA Potenciada por Docker

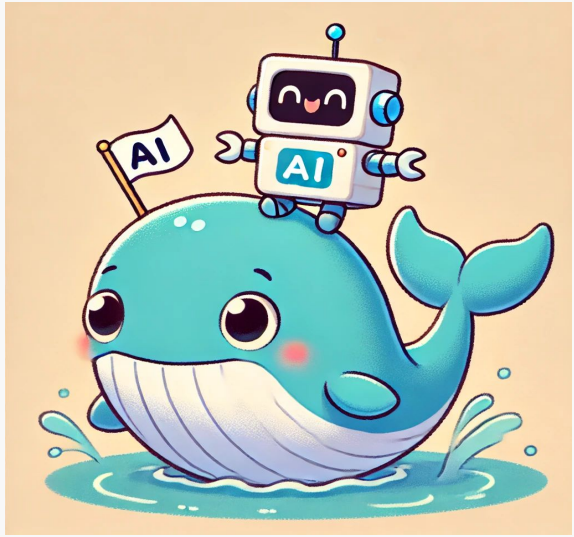
@marisbotero



Hola!!!

Me llamo Maris, juego con datos, hago dibujitos y me gusta la naturaleza.

¿Qué es Docker? – Loncheras mágicas.



Docker es una plataforma que te permite "empaquetar" tus aplicaciones con todas sus dependencias, asegurando que funcionen igual en cualquier entorno.

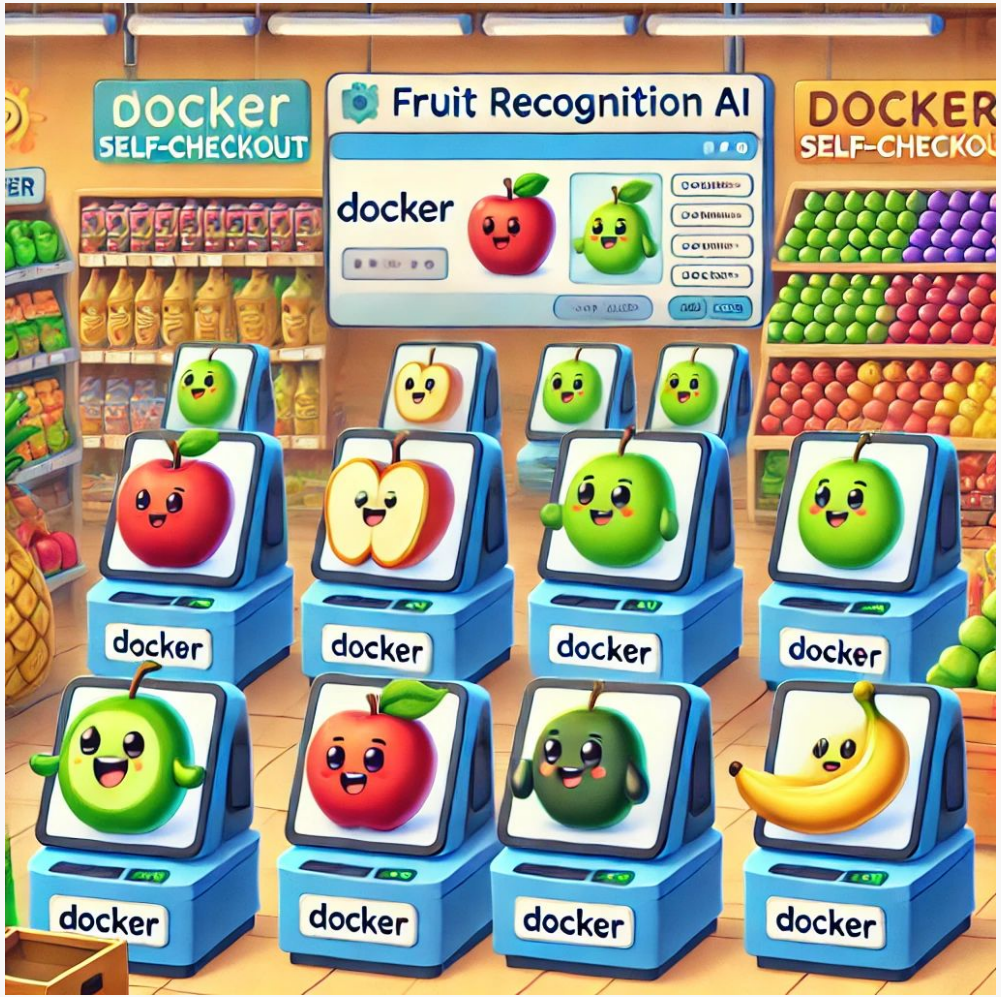

DOCKER
DC
docker
DOCKER

```dockerfile
# Usa la imagen oficial de Python como base
FROM python:3.8-slim

# Copia el archivo de requisitos
COPY requirements.txt /app/requirements.txt

# Instala las dependencias
RUN pip install -r /app/requirements.txt

# Copia el código fuente de la aplicación
COPY . /app

# Define el comando para ejecutar la aplicación
CMD ["python", "/app/main.py"]
```

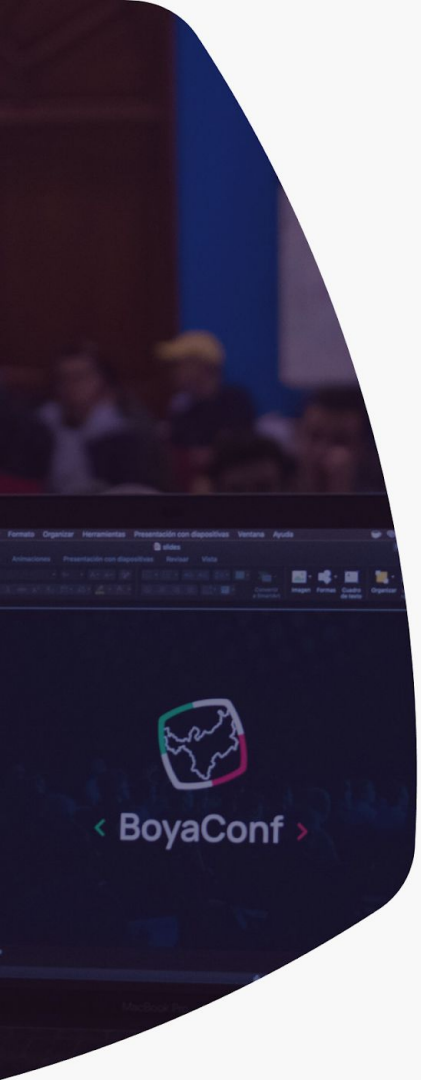

docker
docker
docker
docker
docker
docker
docker
docker
docker
docker

```dockerfile
# Usa TensorFlow como base
FROM tensorflow/tensorflow:2.6.0

# Copia y instala los requisitos específicos del modelo
COPY requirements.txt /app/requirements.txt
RUN pip install -r /app/requirements.txt

# Copia el modelo y el código
COPY . /app

# Define el comando de inicio para el modelo
CMD ["python", "/app/infer.py"]
```


```dockerfile
# Construcción de la imagen y ejecución del contenedor
docker build -t modelo-ia .
docker run -d -p 5000:5000 modelo-ia
```


docker
SELF-CHECKOUT
Fruit Recognition AI
docker
DOCKER
SELF-CHECKOU
docker
docker
docker
docker
docker
docker
docker
docker


```yaml
version: '3'
services:
  modelo1:
    image: modelo-ia
    ports:
      - "5001:5000"
  modelo2:
    image: modelo-ia
    ports:
      - "5002:5000"
  modelo3:
    image: modelo-ia
    ports:
      - "5003:5000"
```

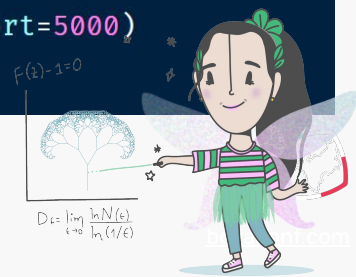


¿Qué es IA? – El chef inteligente.

Cuando entrenamos un modelo de IA, necesitamos bibliotecas específicas (como TensorFlow o PyTorch), además de datos, código, y recursos computacionales. Docker encapsula todo esto para que no tengamos que preocuparnos por errores que pueden ocurrir cuando movemos nuestras aplicaciones de una máquina a otra.

```python
from flask import Flask, request, jsonify
import joblib
import pandas as pd
from sklearn.datasets import load_diabetes
from sklearn.linear_model import LinearRegression

app = Flask(__name__)

# Entrenar el modelo al iniciar la aplicación
def train_model():
    diabetes = load_diabetes()
    X = diabetes.data
    y = diabetes.target
    model = LinearRegression()
    model.fit(X, y)
    # Guardar el modelo entrenado
    joblib.dump(model, 'model.joblib')
    print("Modelo entrenado y guardado.")
```

```python
# Endpoint para hacer predicciones
@app.route('/predict', methods=['POST'])
def predict():
    # Cargar el modelo
    model = joblib.load('model.joblib')
    # Obtener los datos de entrada
    data = request.get_json(force=True)
    input_data = pd.DataFrame(data, index=[0])
    # Realizar la predicción
    prediction = model.predict(input_data)
    # Devolver el resultado
    return jsonify({'prediction': prediction[0]})

if __name__ == '__main__':
    train_model()
    app.run(host='0.0.0.0', port=5000)
```

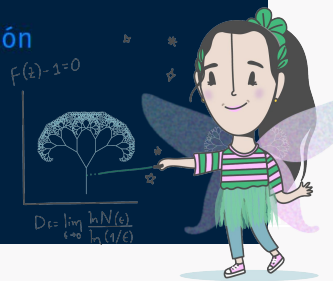


Crear el archivo de requisitos

```
flask
scikit-learn
joblib
pandas
```

Crear el Dockerfile

```
# Usar una imagen base de Python
FROM python:3.8-slim

# Establecer el directorio de trabajo
WORKDIR /app

# Copiar los archivos de requisitos y la
aplicación
COPY requirements.txt requirements.txt
COPY app.py app.py

# Instalar las dependencias
RUN pip install --no-cache-dir -r requirements.txt

# Exponer el puerto de la aplicación
EXPOSE 5000

# Ejecutar la aplicación
CMD ["python", "app.py"]
```



Construir la Imagen de Docker

```
# Construir la imagen y etiquetarla como
'diabetes-predictor'
docker build -t diabetes-predictor .
```

Ejecutar el Contenedor

```
docker run -d -p 5000:5000 diabetes-predictor
```

Probar la Aplicación

```
curl -X POST http://localhost:5000/predict \
     -H 'Content-Type: application/json' \
     -d '{
           "age": 0.0380759064334241,
           "sex": 0.0506801187398187,
           "bmi": 0.0616962065186835,
           "bp": 0.0218723549949558,
           "s1": -0.0442234984244464,
           "s2": -0.0348207628376986,
           "s3": -0.0434008456520269,
           "s4": -0.00259226199818282,
           "s5": 0.0199084208763183,
           "s6": -0.0176461251598052
         }'
```

Respuesta esperada:

```
{"prediction": 150.09261845213342}
```

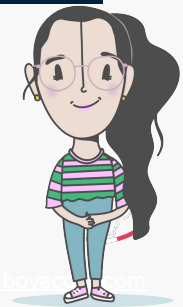
Escalar la Aplicación (Opcional)

Crea un archivo `docker-compose.yml`:

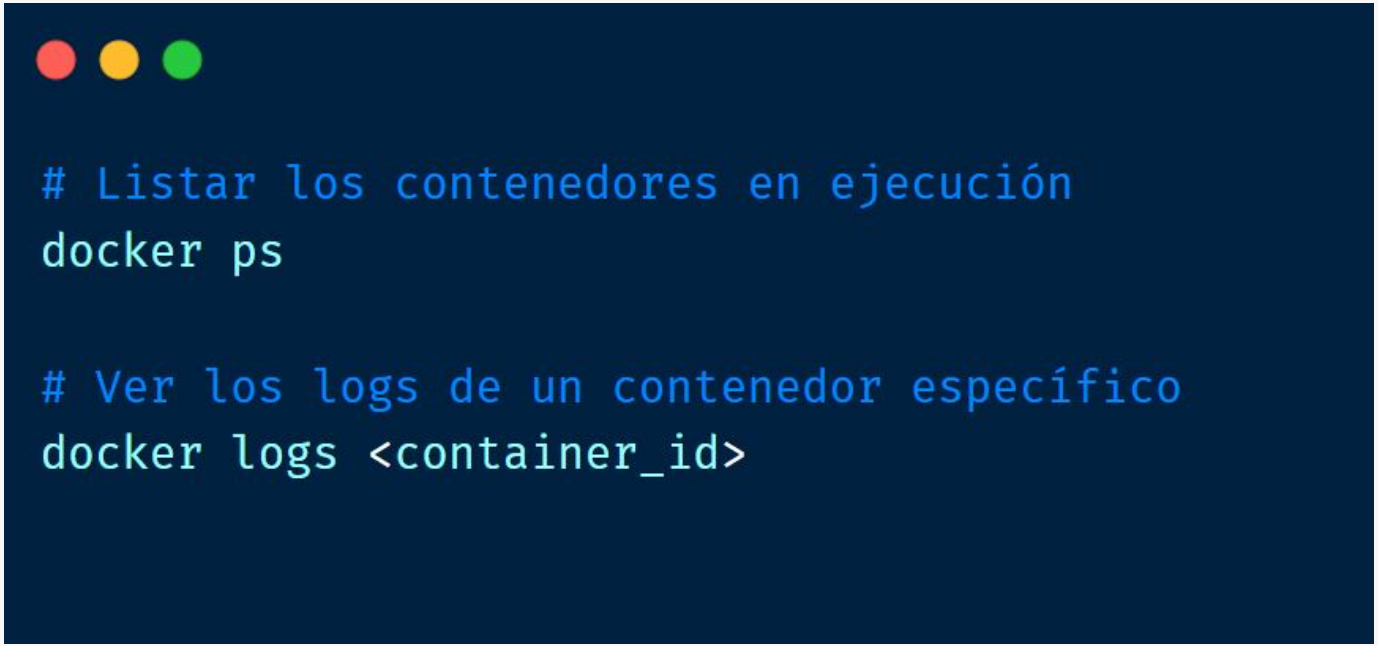
```yaml
version: '3'
services:
  diabetes-predictor:
    build: .
    ports:
      - "5000"
```

Ejecuta Docker Compose para iniciar varias instancias de la aplicación:

```
docker-compose up --scale diabetes-predictor=3
```



Monitorear los Contenedores



Muchas Gracias BoyaConf 💚