



códigofacilito

Módulo 5. Implementación y consumo de modelos con Azure Machine Learning

Implementación de un modelo en un punto de conexión en línea administrado

CINTHYA CABANZO

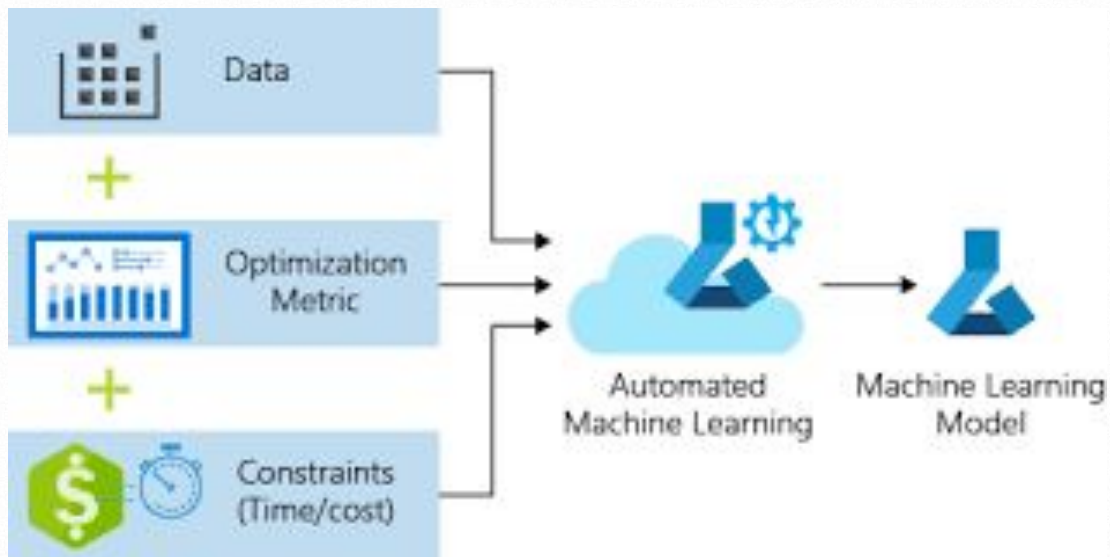




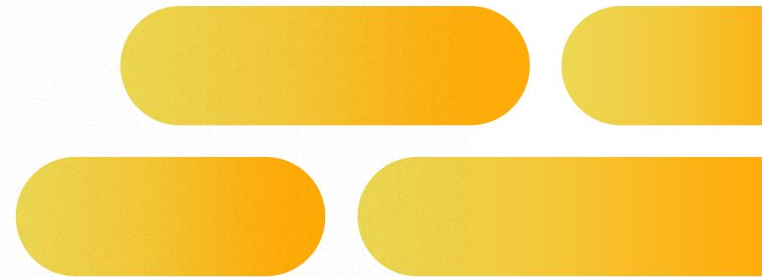
>_ Agenda

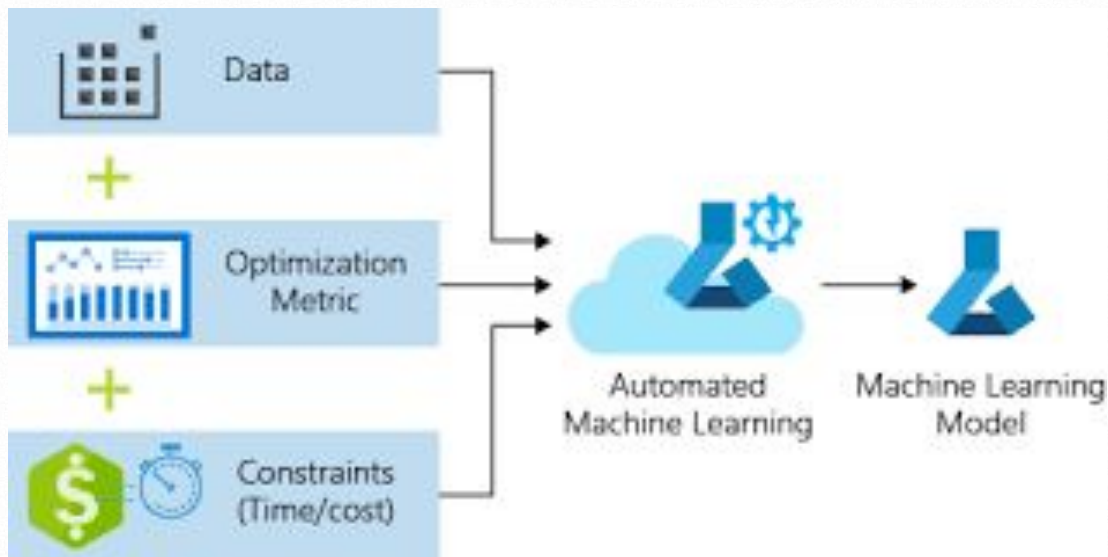
- Introducción
- Exploración de puntos de conexión en línea administrados
- Implementación de un modelo de MLflow en un punto de conexión en línea administrado
- Implementación de un modelo en un punto de conexión en línea administrado
- Prueba de puntos de conexión en línea administrados
- Ejercicio: Implementación de un modelo de MLflow en un punto de conexión en línea



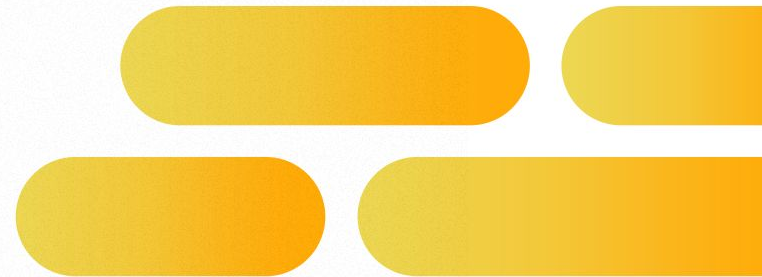


Ya tenemos el
modelo,
y ahora cómo
lo consumo?





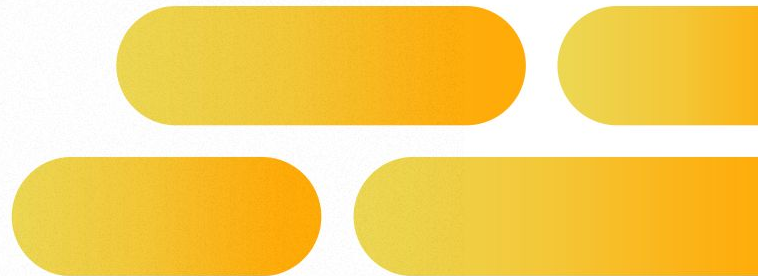
Para que un modelo de aprendizaje automático esté **disponible** en otras aplicaciones, puede **implementarlo** en un **punto de conexión en línea administrado**.



Para que un modelo de aprendizaje automático esté **disponible** en otras aplicaciones, puede **implementarlo** en un **punto de conexión en línea administrado**.



Usar puntos de conexión en línea administrados para predicciones en tiempo real se refiere a emplear servicios de **infraestructura** en la nube que permiten **desplegar modelos** de machine learning (ML) para hacer **predicciones** de manera **inmediata** y sin necesidad de gestionar la infraestructura subyacente.



Para que un modelo de aprendizaje automático esté **disponible** en otras aplicaciones, puede **implementarlo** en un **punto de conexión en línea administrado**.



Conceptos Clave

Puntos de conexión en línea:

Estos son punto de conexión HTTPS (endpoints) accesibles a través de la red (generalmente mediante APIs) donde se puede **enviar datos** y **recibir respuestas** del modelo de ML.

Un punto de conexión en línea está siempre disponible para recibir solicitudes y procesarlas en **tiempo real**.

Los **datos** que envíe al punto de conexión servirán como **entrada** para el **script de puntuación** hospedado en el punto de conexión. El script de puntuación **carga** el **modelo entrenado** para predecir la etiqueta de los nuevos datos de entrada, lo que también se conoce como **inferencia**. La etiqueta forma parte de la salida que se devuelve.



Para que un modelo de aprendizaje automático esté **disponible** en otras aplicaciones, puede **implementarlo** en un **punto de conexión en línea administrado**.



Conceptos Clave

Administrados:

Indica que la **infraestructura** y el **mantenimiento** del servicio son **manejados** por un proveedor de nube (como [Azure](#)).

Esto incluye:

- la escalabilidad
- la disponibilidad
- la seguridad
- las actualizaciones del servicio

Liberando al usuario de estas responsabilidades.

Para que un modelo de aprendizaje automático esté **disponible** en otras aplicaciones, puede **implementarlo** en un **punto de conexión en línea administrado**.



Conceptos Clave

Predicciones en tiempo real:

Se refiere a la capacidad de recibir datos, **procesarlos a través del modelo de ML** y devolver resultados (**predicciones**) de manera instantánea o con una latencia mínima.

Adecuada para aplicaciones que requieren respuestas inmediatas.





Ejemplo Práctico

Imagina que tienes un modelo de ML que predice la **probabilidad** de que una **transacción sea fraudulenta**.

Al usar un punto de conexión en línea administrado, que crees que podrías hacer?

Por favor, **escribe** tus **ideas** en el **chat**!





Ejemplo Práctico

Al usar un punto de conexión en línea administrado, podrías:

1. **Desplegar** el modelo en un servicio en la **nube** (Azure Machine Learning).
2. Obtener un **endpoint**, URL proporcionado por el servicio en la nube.
3. Enviar **solicitudes HTTP** a este endpoint con datos de transacciones en tiempo real.
4. Recibir **predicciones** instantáneamente indicando la **probabilidad** de fraude para cada **transacción**.





Ejemplo Práctico

Este enfoque es beneficioso porque:

- **Escalabilidad automática:** El servicio en la nube puede manejar aumentos en la carga de trabajo sin intervención manual.
- **Mantenimiento reducido:** No necesitas preocuparte por la gestión del hardware o la infraestructura.
- **Tiempo de respuesta:** Diseñado para responder rápidamente, lo cual es crucial para aplicaciones en tiempo real.





Ejemplo Práctico

Este enfoque es beneficioso porque:

- **Escalabilidad automática:** El servicio en la nube puede manejar aumentos en la carga de trabajo sin intervención manual.
- **Mantenimiento reducido:** No necesitas preocuparte por la gestión del hardware o la infraestructura.
- **Tiempo de respuesta:** Diseñado para responder rápidamente, lo cual es crucial para aplicaciones en tiempo real.

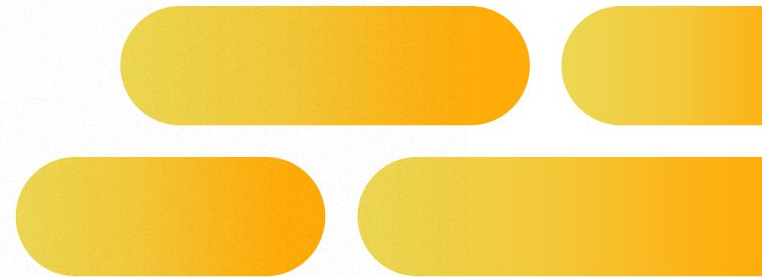


☰ Tipos de puntos de conexión en línea

1. Administrado (Azure Machine Learning)



2. Kubernetes



Tipos de puntos de conexión en línea – Kubernetes

Ventajas

- **Alta Flexibilidad:** Ofrece mayor control sobre la infraestructura y la configuración del despliegue.
- **Escalabilidad Personalizada:** Capacidad para escalar según necesidades específicas, con un control detallado sobre los recursos.
- **Independencia del Proveedor:** Menor dependencia de las soluciones específicas de Azure, lo que permite un enfoque más agnóstico en términos de proveedor.
- **Uso de Contenedores:** Aprovecha la portabilidad y la eficiencia de los contenedores Docker, facilitando la implementación y el escalado de aplicaciones.

Desventajas:

- **Mayor Complejidad:** Requiere una mayor comprensión y gestión de la infraestructura Kubernetes, lo que puede aumentar la complejidad del despliegue y mantenimiento.
- **Mantenimiento y Actualizaciones:** Responsabilidad del usuario para gestionar las actualizaciones de seguridad y mantenimiento del clúster.
- **Costos Operativos:** Aunque Kubernetes puede ser más rentable en términos de uso de recursos, los costos operativos pueden aumentar debido a la necesidad de administrar y mantener la infraestructura.

Tipos de puntos de conexión en línea – AML

Ventajas

- **Facilidad de Uso:** Azure ML proporciona una interfaz fácil de usar y una integración directa con otros servicios de Azure, facilitando el despliegue y la gestión de modelos.
- **Administración Simplificada:** Azure ML maneja automáticamente la infraestructura, la escalabilidad, la seguridad y el mantenimiento.
- **Integración con el Ecosistema de Azure:** Integración nativa con otros servicios de Azure como Azure Data Lake, Azure Data Factory, etc.
- **Actualizaciones Automáticas:** Los puntos de conexión reciben actualizaciones automáticas de seguridad y rendimiento.
- **Monitorización y Mantenimiento:** Proporciona herramientas integradas para el monitoreo y la gestión del rendimiento de los modelos desplegados.

Desventajas:

- **Menos Flexibilidad:** Menos opciones de personalización en comparación con un despliegue en Kubernetes.
- **Costos Potenciales:** Dependiendo del uso, puede ser más costoso debido a la administración y servicios adicionales proporcionados por Azure.
- **Dependencia del Proveedor:** Mayor dependencia del ecosistema y las actualizaciones de Azure.

☰ Tipos de puntos de conexión en línea

1. Administrado (Azure Machine Learning)



2. Kubernetes



Conclusión

La elección entre un punto de conexión en línea administrado por Azure Machine Learning y uno por Kubernetes depende de las necesidades específicas del proyecto, el nivel de **expertise del equipo**, la necesidad de **flexibilidad** y **control**, y las consideraciones de **costos**. Azure ML es ideal para quienes buscan una solución más sencilla y administrada, mientras que Kubernetes es mejor para quienes necesitan un control y flexibilidad máximos en su despliegue.

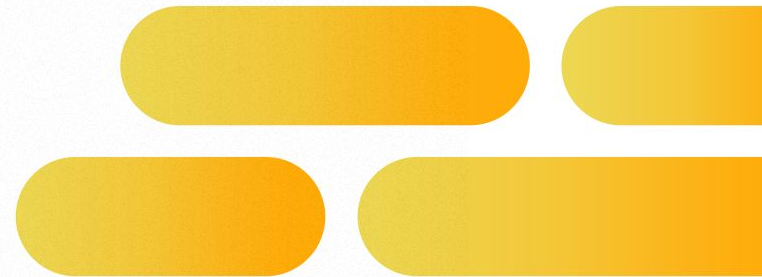


📋 Implementación del modelo en un punto de conexión en línea administrado

- **Recursos de modelo** como el archivo pickle del modelo o un **modelo registrado** en el área de trabajo de Azure Machine Learning.
- **Script de puntuación** que carga el modelo.
- **Entorno** que enumera todos los paquetes necesarios que deben instalarse en el proceso del punto de conexión.
- La **configuración de escalado**, incluidos el **tamaño de proceso** y la **configuración de escalado** para asegurarse de que puede controlar la cantidad de solicitudes que recibirá el punto de conexión.

Importante

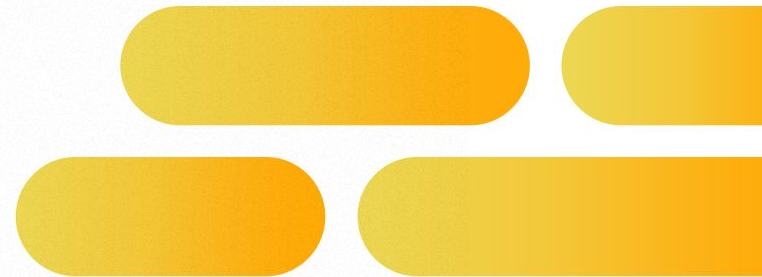
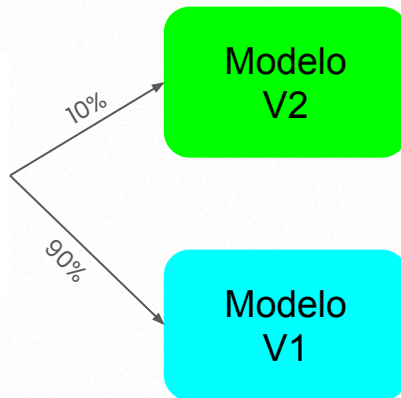
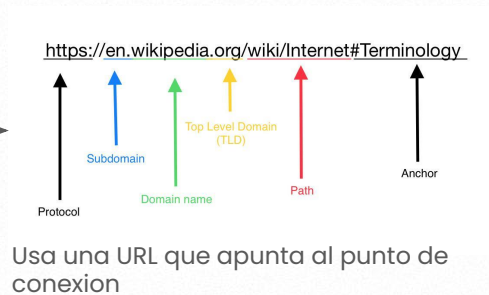
Al implementar **modelos** de **MLFlow** en un punto de conexión en línea, **no es necesario** proporcionar un script de puntuación y un entorno, ya que ambos se generan automáticamente.



Implementación azul-verde



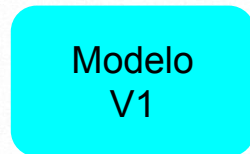
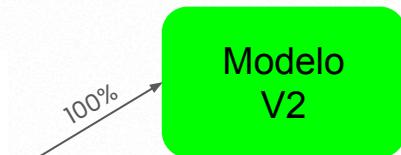
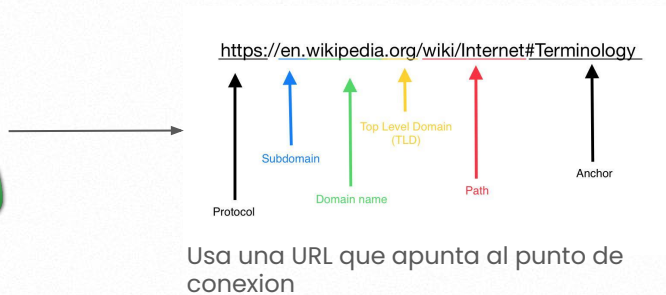
Usuario



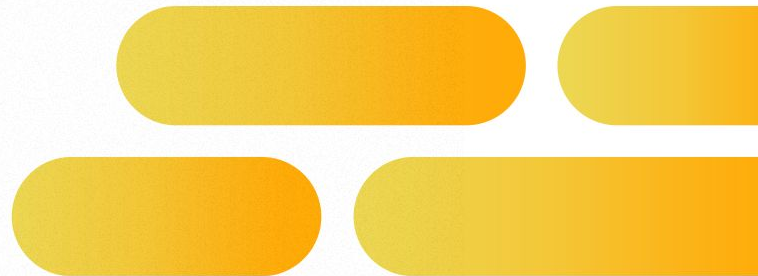
Implementación azul-verde



Usuario



La implementación azul/verde permite implementar **varios modelos** en un punto de conexión. Puede decidir cuánto tráfico se debe desviar a cada modelo implementado. De este modo, puede **cambiar** a una **nueva versión** del modelo **sin interrumpir** el **servicio** al consumidor.





Crear un punto de conexión

ManagedOnlineEndpoint

name: Nombre del punto de conexión. Debe ser único en la región de Azure.

auth_mode: Use **key** para la autenticación basada en claves. Use **aml_token** para la autenticación basada en tokens de Azure Machine Learning.

Python

```
from azure.ai.ml.entities import ManagedOnlineEndpoint
```

```
# create an online endpoint
```

```
endpoint = ManagedOnlineEndpoint(  
    name="endpoint-example",  
    description="Online endpoint",  
    auth_mode="key",  
)
```

```
ml_client.begin_create_or_update(endpoint).result()
```

```
.....
```

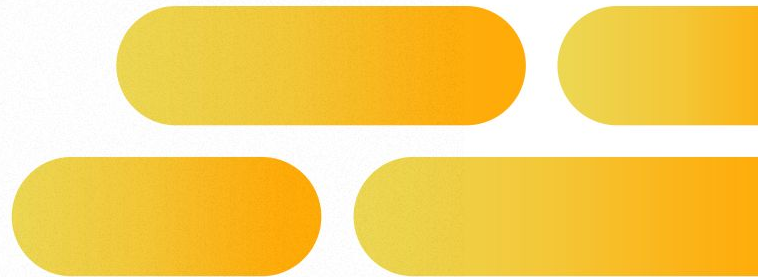
<https://learn.microsoft.com/es-es/python/api/azure-ai-ml/azure.ai.ml.entities.managedonlineendpoint?view=azure-python>

● Implementación de un modelo de MLflow en un punto de conexión en línea administrado

Para implementar un modelo de MLflow, debe tener archivos de modelo almacenados en una ruta de acceso local o con un modelo registrado.

En este ejemplo, vamos a tomar los archivos de modelo de una ruta de acceso local. Todos los archivos se almacenan en una carpeta local denominada model. La carpeta debe incluir el archivo MLmodel, que describe cómo se puede cargar y usar el modelo.

- `instance_type`: tamaño de la máquina virtual (VM) que se va a usar.
- `instance_count`: número de instancias que se van a usar.



Implementación de un modelo de **MLflow** en un punto de conexión en línea administrado

Python

```
from azure.ai.ml.entities import Model, ManagedOnlineDeployment
from azure.ai.ml.constants import AssetTypes

# create a blue deployment
model = Model(
    path="./model",
    type=AssetTypes.MLFLOW_MODEL,
    description="my sample mlflow model",
)

blue_deployment = ManagedOnlineDeployment(
    name="blue",
    endpoint_name="endpoint-example",
    model=model,
    instance_type="Standard_F4s_v2",
    instance_count=1,
)

ml_client.online_deployments.begin_create_or_update(blue_deployment).result()
```

```
from azure.ai.ml.entities import Model, ManagedOnlineDeployment
from azure.ai.ml.constants import AssetTypes

# Importa las clases necesarias para definir el modelo y el despliegue administrado en línea.
# También importa los tipos de activos, en este caso, para especificar que el modelo es un modelo MLflow.

# Crea una instancia de la clase Model que representa el modelo MLflow que deseas desplegar.
model = Model(
    path="./model",          # Especifica la ruta al modelo que deseas desplegar.
    type=AssetTypes.MLFLOW_MODEL, # Indica que el tipo de modelo es un modelo MLflow.
    description="my sample mlflow model", # Proporciona una descripción del modelo.
)

# Crea una instancia de la clase ManagedOnlineDeployment para definir el despliegue del modelo.
blue_deployment = ManagedOnlineDeployment(
    name="blue",              # Nombre del despliegue. Puede ser cualquier nombre que elijas.
    endpoint_name="endpoint-example", # Nombre del endpoint al que se asociará el despliegue.
    model=model,              # Asocia el modelo definido anteriormente con este despliegue.
    instance_type="Standard_F4s_v2", # Especifica el tipo de instancia de Azure VM a usar para el
despliegue.
    instance_count=1,          # Número de instancias de la VM que se desplegarán. Aquí se despliega
una sola instancia.
)

# Inicia la creación o actualización del despliegue en línea gestionado.
ml_client.online_deployments.begin_create_or_update(blue_deployment).result()
# Utiliza el cliente de Azure ML ('ml_client') para iniciar el proceso de creación o actualización del
despliegue en línea.
# 'begin_create_or_update' es un método que inicia la operación de creación o actualización.
# 'result()' espera a que la operación se complete y devuelve el resultado.
```

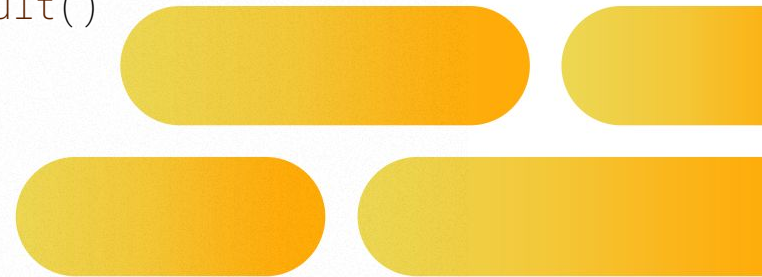
📋 Enrutar el tráfico a una implementación específico

Python

```
# blue deployment takes 100 traffic
endpoint.traffic = {"blue": 100}
ml_client.begin_create_or_update(endpoint).result()
```

Asigna el 100% del tráfico al despliegue 'blue'
`endpoint.traffic = {"blue": 100}`

Utiliza el cliente de Azure ML para actualizar el endpoint con la nueva configuración de tráfico
`ml_client.begin_create_or_update(endpoint).result()`



Eliminar el punto de conexión y todas las implementaciones asociadas

`begin_delete`

Python

```
ml_client.online_endpoints.begin_delete(name="endpoint-example")
```

Este código inicia el proceso de eliminación de un endpoint en línea específico en Azure Machine Learning.

ml_client.online_endpoints es el componente del cliente de Azure ML que maneja operaciones relacionadas con endpoints en línea.

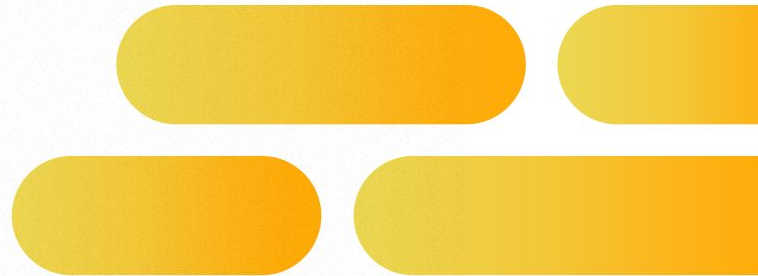
begin_delete(name="endpoint-example") es el método utilizado para comenzar el proceso de eliminación del endpoint llamado "endpoint-example".



📋 Implementación de un modelo en un punto de conexión en línea administrado SIN MLFLOW

Para implementar un modelo, deberá crear el **script de puntuación**, definir el **entorno** necesario durante la inferencia y **archivos de modelo** almacenados en la ruta de acceso local o en el **modelo registrado**.

Para implementar un modelo, debe haber **creado** un **punto de conexión**. A continuación, puede implementar el modelo en el punto de conexión.





Creación del script de puntuación

`init()`, `run()`

Python

```
import json
import joblib
import numpy as np
import os

# called when the deployment is created or updated
def init():
    global model
    # get the path to the registered model file and load it
    model_path = os.path.join(os.getenv('AZUREML_MODEL_DIR'), 'model.pkl')
    model = joblib.load(model_path)

# called when a request is received
def run(raw_data):
    # get the input data as a numpy array
    data = np.array(json.loads(raw_data)['data'])
    # get a prediction from the model
    predictions = model.predict(data)
    # return the predictions as any JSON serializable format
    return predictions.tolist()
```

```
import json # Para manejar datos en formato JSON
import joblib # Para cargar el modelo previamente guardado
import numpy as np # Para manejar y operar con arrays numéricos
import os # Para manejar rutas y variables de entorno del sistema operativo

# Esta función se llama cuando se crea o actualiza el despliegue
def init():
    global model # Declara que `model` es una variable global, accesible desde otras
funciones
    # Obtiene la ruta al archivo del modelo registrado y lo carga
    model_path = os.path.join(os.getenv('AZUREML_MODEL_DIR'), 'model.pkl')
    model = joblib.load(model_path)

# Esta función se llama cuando se recibe una solicitud
def run(raw_data):
    # Convierte los datos de entrada a un array de numpy
    data = np.array(json.loads(raw_data)['data'])
    # Obtiene una predicción del modelo
    predictions = model.predict(data)
    # Devuelve las predicciones en un formato serializable por JSON
    return predictions.tolist()
```

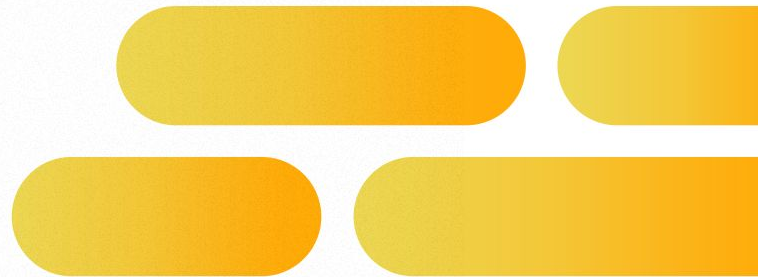

Creación de un entorno

Puede crear un entorno con una imagen de Docker con dependencias de Conda o con Dockerfile.

conda.yml

yml

```
name: basic-env-cpu
channels:
  - conda-forge
dependencies:
  - python=3.7
  - scikit-learn
  - pandas
  - numpy
  - matplotlib
```



Creación de un entorno

Puede crear un entorno con una imagen de Docker con dependencias de Conda o con Dockerfile.

Python

```
from azure.ai.ml.entities import Environment

env = Environment(
    image="mcr.microsoft.com/azureml/openmpi3.1.2-ubuntu18.04",
    conda_file="./src/conda.yml",
    name="deployment-environment",
    description="Environment created from a Docker image plus Conda environment.",
)
ml_client.environments.create_or_update(env)
```



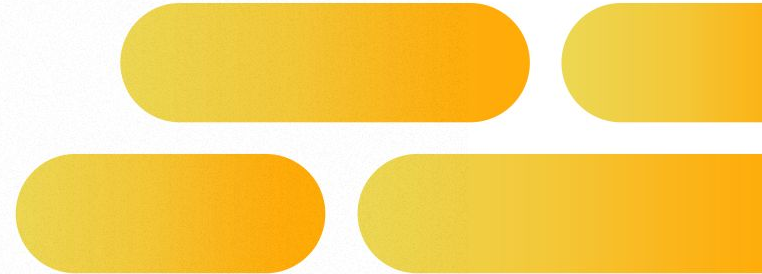
```
from azure.ai.ml.entities import Environment # Importa la clase Environment  
de azure.ai.ml.entities
```

Crea una instancia de la clase Environment, que define el entorno en el que se ejecutará el modelo.

```
env = Environment(  
    image="mcr.microsoft.com/azureml/openmpi3.1.2-ubuntu18.04", # Especifica  
la imagen de Docker a usar como base del entorno.  
    conda_file="./src/conda.yml", # Especifica el archivo conda.yml que  
contiene las dependencias de Python.  
    name="deployment-environment", # Asigna un nombre al entorno.  
    description="Environment created from a Docker image plus Conda  
environment.", # Proporciona una descripción del entorno.  
)
```

Usa el cliente de Azure ML (ml_client) para crear o actualizar el entorno en Azure.

```
ml_client.environments.create_or_update(env)
```





Creación de la implementación

ManagedOnlineDeployment

Python

```
from azure.ai.ml.entities import ManagedOnlineDeployment, CodeConfiguration

model = Model(path="./model",

blue_deployment = ManagedOnlineDeployment(
    name="blue",
    endpoint_name="endpoint-example",
    model=model,
    environment="deployment-environment",
    code_configuration=CodeConfiguration(
        code="./src", scoring_script="score.py"
    ),
    instance_type="Standard_DS2_v2",
    instance_count=1,
)

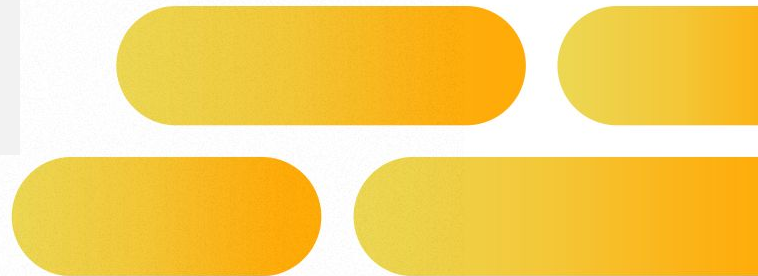
ml_client.online_deployments.begin_create_or_update(blue_deployment).result()
```



Para implementar un modelo en un punto de conexión, puede especificar la configuración de proceso con dos parámetros:

instance_type: tamaño de la máquina virtual (VM) que se va a usar. Revise la lista de tamaños admitidos.

instance_count: número de instancias que se van a usar.



Enrutar el tráfico a una implementación específica

traffic

Python

```
# blue deployment takes 100 traffic
endpoint.traffic = {"blue": 100}
ml_client.begin_create_or_update(endpoint).result()
```

Eliminar el punto de conexión y todas las implementaciones asociadas

begin_create_or_update

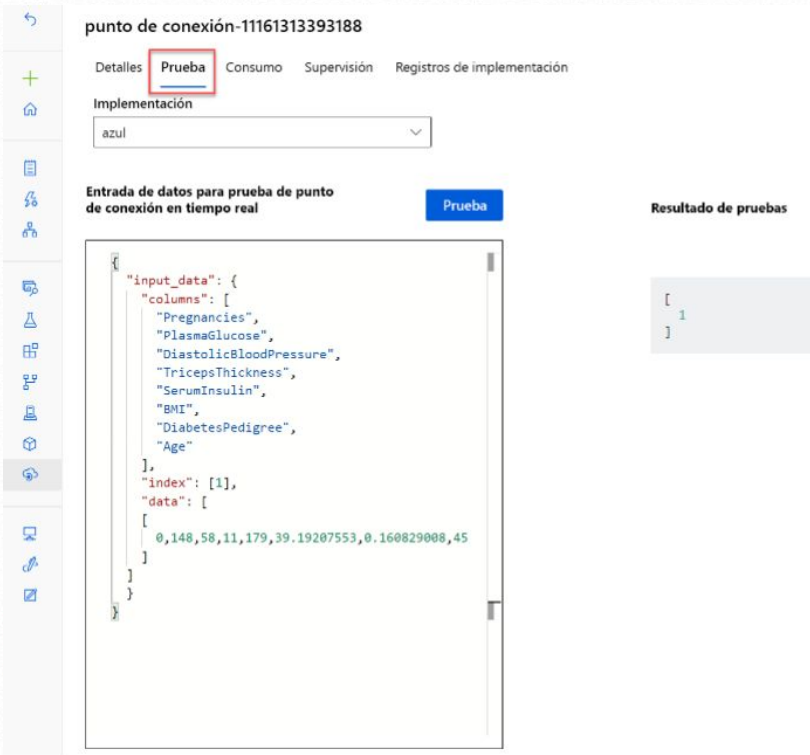
Python

```
ml_client.online_endpoints.begin_delete(name="endpoint-example")
```



Prueba de puntos de conexión en línea administrados

Estudio de Azure Machine Learning



The screenshot shows the Azure Machine Learning portal interface. The top navigation bar includes tabs: Detalles, **Prueba** (highlighted with a red box), Consumo, Supervisión, and Registros de implementación. Below the tabs, there's a dropdown menu for 'Implementación' with 'azul' selected. The main section is titled 'Entrada de datos para prueba de punto de conexión en tiempo real' and features a blue 'Prueba' button. A text area contains a JSON payload for the test request:

```
{
  "input_data": {
    "columns": [
      "Pregnancies",
      "PlasmaGlucose",
      "DiastolicBloodPressure",
      "TricepsThickness",
      "SerumInsulin",
      "BMI",
      "DiabetesPedigree",
      "Age"
    ],
    "index": [1],
    "data": [
      [
        0,148,58,11,179,39.19207553,0.160829008,45
      ]
    ]
  }
}
```

To the right, under the heading 'Resultado de pruebas', a small box displays the response:

```
[
  1
]
```

Estudio de Azure Machine Learning

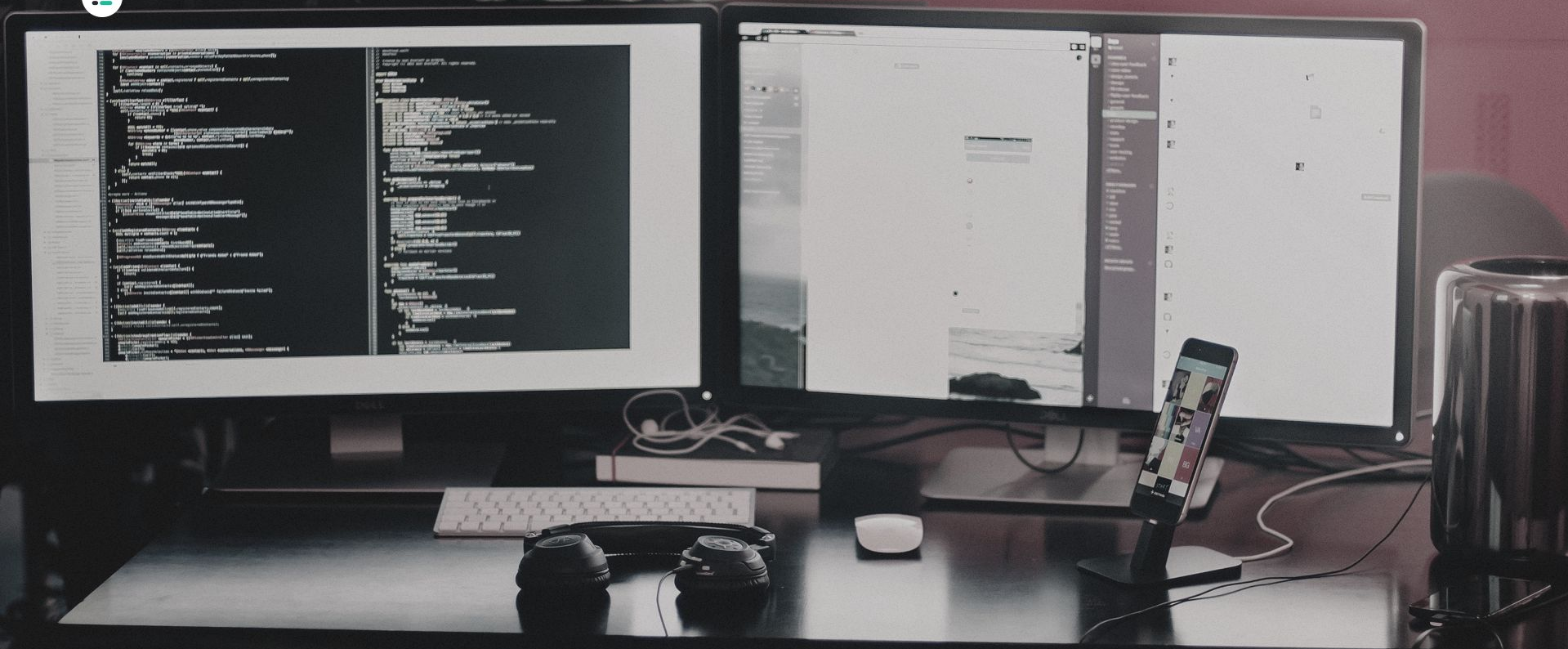
JSON

```
{
  "data": [
    [0.1,2.3,4.1,2.0], // 1st case
    [0.2,1.8,3.9,2.1], // 2nd case,
    ...
  ]
}
```

Python

```
# test the blue deployment with some sample data
response = ml_client.online_endpoints.invoke(
    endpoint_name=online_endpoint_name,
    deployment_name="blue",
    request_file="sample-data.json",
)

if response[1]=='1':
    print("Yes")
else:
    print ("No")
```



Vamos a la practica!!!



“Lo esencial es invisible a los ojos”

Antoine de Saint-Exupéry, autor de El Principito





```
//life motto  
if (sad()===true){  
    sad().stop();  
    beAwesome();  
}
```

