



**MIAD**

Maestría  
en Inteligencia  
Analítica de Datos



# Taller MLflow

## Instrucciones:

La primera parte de este taller consiste en instalar MLflow en una máquina virtual y emplearlo para llevar el versionamiento de modelos. En la segunda parte se muestra cómo crear una cuenta en Databricks, donde también podrá usar MLflow.

La entrega de este taller consiste en un reporte en formato de documento de texto, donde pueda incorporar fácilmente capturas de pantalla, textos y elementos similares. El formato de entrega del reporte es PDF.

## 1. Instale y pruebe MLflow en una máquina virtual

1. Lance una instancia en AWS EC2. Se recomienda una máquina t2.medium, con sistema operativo Ubuntu y 20GB de disco. **Incluya un pantallazo de la consola de AWS EC2 con la máquina en ejecución en su reporte.**
2. Conéctese a la máquina:

a) Abra una terminal: En windows, escriba *cmd* y *Enter*. En macOS, abra la aplicación llamada *Terminal*.

b) En la terminal emita el comando

```
ssh -i /path/to/llave.pem ubuntu@IP
```

donde */path/to/* se refiere a la ubicación del archivo *llave.pem* que descargó, e *IP* es la dirección IP de la instancia EC2 que lanzó. Si prefiere, en la terminal puede navegar a la ubicación del archivo *llave.pem* y emitir el comando

```
ssh -i llave.pem ubuntu@IP
```

**Note que usamos en este caso *ubuntu* en vez de *ec2-user*, pues éste es el usuario creado por defecto como administrador con sistema operativo Ubuntu server. Incluya en su reporte un *screenshot* de la conexión a la máquina virtual.**

3. Actualice las ubicaciones de los paquetes con el siguiente comando:



```
sudo apt update
```

4. Instale pip para python3

```
sudo apt install python3-pip
```

5. Instale sklearn

```
pip install scikit-learn
```

6. Instale mlflow

```
pip install mlflow
```

Incluya la dirección `/home/ubuntu/.local/bin` en el PATH para facilitar su ejecución

```
PATH=$PATH:/home/ubuntu/.local/bin
```

**Incluya en su reporte un *screenshot* de la instalación de mlflow.**

7. Descargue el archivo `mlflow-diab.py` que encontrará en Coursera. Estudie con cuidado el código y describa en su reporte qué hace.

8. Comente la línea de registro de mlflow (línea 26)

```
mlflow.set_tracking_uri("http://0.0.0.0:5000")
```

9. En una terminal aparte, suba el archivo `mlflow-diab.py` a la máquina con un comando como

```
scp -i llave.pem mlflow-diab.py ubuntu@IP:/home/ubuntu
```

10. En la terminal de la máquina ejecute mlflow como servidor y exponga el servicio por el puerto 8050.

```
mlflow server -h 0.0.0.0 -p 8050
```

**Tome un pantallazo de la salida de la terminal con el servicio de mlflow en ejecución e inclúyalo en su reporte.**

11. La interfaz web debe quedar disponible a través del navegador en el puerto 8050. Abra el puerto en el grupo de seguridad de la máquina y compruebe que funciona.

12. Ejecute el archivo de python para entrenar y evaluar el modelo (en una nueva terminal conectada a la máquina virtual)

```
python3 mlflow-diab.py
```

13. Modifique alguno de los parámetros del modelo y vuelva a correr el script. Para ello puede usar el comando `nano`.

```
nano mlflow-diab.py
```



Esto abre un editor de texto. Usando las flechas navegue hasta la parte en que se definen los parámetros del modelo.

Modifique algún parámetro, para esto tenga en cuenta la siguiente información:

- a) Número de estimadores (`n_estimators`): Este parámetro controla la cantidad de árboles en el bosque aleatorio. Puedes probar diferentes valores para ver cómo afecta al rendimiento y la precisión del modelo. Por ejemplo, puede probar con 100, 500 o incluso más árboles.
- b) Profundidad máxima (`max_depth`): Este parámetro controla la profundidad máxima de cada árbol en el bosque aleatorio. Cambiar este parámetro puede afectar la capacidad del modelo para ajustarse a los datos. Puedes probar diferentes valores o incluso dejarlo sin límite (`None`) para permitir que los árboles crezcan hasta que todas las hojas sean puras.
- c) Número máximo de características (`max_features`): Este parámetro controla el número máximo de características que se consideran al buscar la mejor división en cada nodo del árbol. Cambiar este valor puede afectar la diversidad de los árboles y su capacidad para capturar diferentes aspectos de los datos.

Para cerrar el archivo CTRL+X, escriba Y para confirmar que quiere guardar los cambios y ENTER, y regresará a la terminal principal. Para verificar el cambio puede usar el comando

```
cat mlflow-diab.py
```

- 14. Repita este procedimiento varias veces cambiando distintos parámetros.
- 15. Ahora visite la interfaz web de MLflow e identifique el experimento y las dos corridas realizadas. Compare los dos modelos en términos de MSE (error cuadrático medio) usando las funciones de la interfaz. **Incluya en su reporte una gráfica comparando los dos modelos y justifique el comportamiento de acuerdo a los parámetros explorados.**
- 16. Repita este procedimiento con el notebook **mlflow-mnist** que encontrará en Coursera. Para esto tenga en cuenta la siguiente información:
  - a) Tamaño del lote (`batch_size`) : Es el número de ejemplos de entrenamiento que se utilizan en una iteración antes de que los pesos del modelo se actualicen. Especifica cuántos ejemplos se procesan a la vez antes de realizar una actualización en el modelo.
  - b) Épocas (epochs): Es el número de veces que el modelo recorrerá todo el conjunto de entrenamiento durante el entrenamiento. Una época completa significa que cada ejemplo de entrenamiento se ha utilizado una vez para ajustar los pesos del modelo.
  - c) Tasa de aprendizaje (`learning_rate`): Es el factor que controla cuánto se ajustan los pesos del modelo en función del error durante el entrenamiento. Determina la magnitud de los cambios que se aplican a los pesos en cada actualización. Una tasa de aprendizaje alta puede hacer que



el modelo se ajuste rápidamente, pero puede ser propenso a saltar por encima de los mínimos locales. Una tasa de aprendizaje baja puede hacer que el modelo converja más lentamente pero con mayor precisión.

- d) **Número de unidades ocultas (num\_hidden\_units):** Es el número de neuronas en cada capa oculta de la red neuronal. Cuantas más unidades ocultas haya, más capacidad tendrá el modelo para aprender representaciones más complejas de los datos, pero también puede aumentar la complejidad computacional y el riesgo de sobreajuste.
  - e) **Número de capas ocultas (num\_hidden\_layers):** Es el número de capas ocultas en la red neuronal. Cada capa oculta contiene un número determinado de unidades ocultas. Aumentar el número de capas ocultas puede permitir al modelo aprender características más abstractas y complejas, pero también puede aumentar la complejidad del modelo y requerir más tiempo de entrenamiento.
  - f) **Dropout:** Es una técnica de regularización utilizada durante el entrenamiento de redes neuronales para reducir el sobreajuste. Especifica la fracción de unidades en cada capa oculta que se “apaga” de manera aleatoria durante el entrenamiento. El dropout evita que las neuronas dependan demasiado de las demás y promueve una representación más robusta.
  - g) **Momentum:** Es un parámetro utilizado en algunos algoritmos de optimización, como el descenso de gradiente estocástico con momento (SGDM), para acelerar la convergencia y evitar oscilaciones. Controla la proporción en la que se tiene en cuenta el cambio de los pasos anteriores en la dirección del gradiente. Un valor alto de momentum permite que el modelo se “mueva” más rápidamente a lo largo de las regiones con pendientes consistentes, mientras que un valor bajo permite que el modelo “explote” más las regiones con cambios bruscos.
17. **En su reporte explique en qué consiste este último notebook y los modelos que allí se entrenan.**
18. **Incluya en su reporte una gráfica comparando los modelos entrenados y justifique el comportamiento de acuerdo a los parámetros explorados.**

## 2. MLflow en Databricks

1. Cree una cuenta en Databricks Free Edition, usando el botón de Sign Up en <https://www.databricks.com/learn/free-edition>. Si ya tiene una cuenta puede usar la opción de Login to Free Edition.
2. Proporcione una dirección de correo electrónico y recibirá un código de identificación.



3. Una vez haya creado la cuenta, ingrese y familiarícese con la consola principal. **Incluya en su reporte un screenshot de su cuenta de Databricks** donde sea **visible** el **correo** que usó para crear la cuenta (esquina superior derecha).
4. En el menú de la izquierda de click en **New** y luego en **Notebook** para crear un nuevo cuaderno de jupyter.
5. En la parte superior izquierda del editor, asigne un nombre al notebook y asegúrese de tener Python como lenguaje seleccionado (por defecto).
6. En la parte superior derecha del editor use el botón **Connect** para conectarse al servicio de cómputo Serverless. Esto hará que el cuaderno use este servicio para su ejecución. Note que esta inicialización puede tomar algunos minutos. Mientras se completa esta inicialización puede continuar con los siguientes pasos.
7. Con el notebook abierto, copie las instrucciones del notebook **mlflow-diab** que encontrará en Coursera.
8. Revise y comprenda las instrucciones de cada celda. Aún no las ejecute.
9. Comente la línea de registro de mlflow:  

```
mlflow.set_tracking_uri("http://0.0.0.0:5000")
```

ya que esta línea no es necesaria en Databricks.
10. Ajuste también el nombre del experimento en la línea  

```
experiment = mlflow.set_experiment("sklearn-diab")
```

Incluya un slash / al inicio del nombre, pues en Databricks todos los nombres deben tener una dirección absoluta que inicie con un slash. Agregue además su primer nombre al final del nombre del experimento. Por ejemplo, esta línea puede quedar como  

```
experiment = mlflow.set_experiment("/sklearn-diab-juan")
```
11. Ahora sí, ejecute cada celda y explore los resultados.
12. En el menú de la izquierda seleccione Experiments.
13. Seleccione el experimento realizado y navegue la documentación asociada (versión del modelo, librerías, python). Es posible que le pida seleccionar el tipo de experimento, entre GenAI y Machine Learning. Seleccione Machine Learning.
14. Vuelva al cuaderno y cambie los parámetros del entrenamiento del modelo y ejecute un nuevo experimento. Para esto tenga en cuenta la siguiente información:
  - a) Número de estimadores (n\_estimators): Este parámetro controla la cantidad de árboles en el bosque aleatorio. Puedes probar diferentes valores para ver cómo afecta al rendimiento y la precisión del modelo. Por ejemplo, puede probar con 100, 500 o incluso más árboles.



- b) Profundidad máxima (`max_depth`): Este parámetro controla la profundidad máxima de cada árbol en el bosque aleatorio. Cambiar este parámetro puede afectar la capacidad del modelo para ajustarse a los datos. Puedes probar diferentes valores o incluso dejarlo sin límite (`None`) para permitir que los árboles crezcan hasta que todas las hojas sean puras.
  - c) Número máximo de características (`max_features`): Este parámetro controla el número máximo de características que se consideran al buscar la mejor división en cada nodo del árbol. Cambiar este valor puede afectar la diversidad de los árboles y su capacidad para capturar diferentes aspectos de los datos.
15. Repita este procedimiento varias veces. Revise los resultados en Experiments, genere gráficas comparativas. Compare los dos modelos en términos de MSE (error cuadrático medio). **Incluya en su reporte una gráfica comparando los dos modelos y justifique el comportamiento de acuerdo a los parámetros explorados.**
  16. Puede emplear Databricks para realizar más experimentos con éste y otros notebooks como el `mlflow-mnist` que encontrará en Coursera.