

 <p>Universidad Pontificia Bolivariana <small>SECCIONAL BUCARAMANGA</small></p>	<h2 style="text-align: center;">Laboratorio 3.5: Implementación de un modelo</h2>		
Curso:	Analíticas de Datos - Opt	Docente:	Omar Pinzón Ardila
Grupo:		Fecha:	

Integrantes	ID

Descripción del trabajo a realizar
<p>En este laboratorio se diseña un flujo completo de almacenamiento, consumo y transformación por lotes de un modelo de Machine Learning en un entorno local. Para ello, utilizarán MongoDB con GridFS para alojar el artefacto entrenado y una API REST para consumirlo mediante predicciones individuales y en lote.</p> <p>El trabajo emula prácticas de MLOps que permiten separar el entrenamiento, el versionado y el consumo de modelos en aplicaciones reales.</p>
Objetivos
<ol style="list-style-type: none"> 1. Alojar modelos en una base de datos local usando GridFS, entendiendo cómo se gestionan archivos grandes en MongoDB. 2. Exponer un modelo a través de un servicio API REST que permita predicciones en línea (online inference). 3. Implementar transformaciones por lotes (batch inference) simulando un escenario de predicción masiva sobre datasets nuevos. 4. Documentar el proceso completo, analizando dificultades técnicas y reflexionando sobre buenas prácticas en despliegues locales.
Actividades por pasos
<p>Paso 1: Alojar el modelo</p> <ul style="list-style-type: none"> • Entrenar un modelo con un dataset (a elección del grupo o sugerido por el profesor). • Serializar el modelo en un formato estándar. • Guardar el modelo en MongoDB GridFS, registrando, además: <ul style="list-style-type: none"> ○ Nombre del modelo. ○ Versión. ○ Métricas de desempeño. ○ Fecha de creación. • Verificar que el modelo se pueda recuperar desde la base de datos.

Paso 2: Realizar predicciones (inferencia en línea)

- Implementar un **endpoint REST /predict** para consumir el modelo activo.
- Permitir que el servicio reciba registros en formato JSON y devuelva la predicción y la probabilidad asociada.
- Probar el endpoint con datos de ejemplo y registrar los resultados.

Paso 3: Gestión de modelos

- Listar los modelos disponibles en la base de datos (endpoint /models).
- Permitir activar un modelo específico como el “en producción” (endpoint /models/activate).
- Verificar que las predicciones usen siempre el modelo activo.

Paso 4: Realizar una transformación por lotes (batch inference)

- Implementar un **endpoint /batch-predict** que reciba un archivo CSV con nuevos datos.
- Procesar el archivo completo usando el modelo activo.
- Guardar los resultados de las predicciones en un archivo de salida (ej. predicciones.csv).
- Entregar evidencias de ejecución (archivo generado, ejemplos de entrada y salida).

Entregables**Entregable 1:** Presentación conceptual y evidencia de ejecución del laboratorio de AWS

El estudiante desarrollará el laboratorio propuesto en la plataforma AWS Academy, siguiendo paso a paso las instrucciones del cuaderno original. La entrega final se realizará mediante una presentación directa al docente, en la cual el estudiante explicará los resultados obtenidos, analizará las visualizaciones generadas y expondrá las conclusiones del ejercicio.

Se debe prestar especial atención a la aplicación de las inferencias en línea y por lotes. Es importante entender como se aloja el modelo y como puede descargarse manualmente. También es importante comprender como se consume el modelo. Verificar si lo puedo hacer desde una red externa y si no que debería ajustar para realizarlo.

Entregable 2: Ejecución local con adaptación del laboratorio

El estudiante debe adaptar el código del laboratorio para ejecutarlo localmente, incorporando una arquitectura en la cual el modelo entrenado se almacene en

MongoDB utilizando GridFS¹ y se despliegue a través de una **API REST²** que permita consumirlo tanto en predicciones individuales como en transformaciones por lotes.

Se utilizará un conjunto de datos alternativo descargado del repositorio UCI: [Vertebral Column — UCI Machine Learning Repository](#)

El flujo a implementar debe contemplar:

1. **Alojar el modelo:** entrenamiento, serialización y almacenamiento en GridFS con registro de metadatos.
2. **Realizar predicciones:** habilitar un endpoint REST /predict que consuma el modelo activo.
3. **Realizar transformación por lotes:** habilitar un endpoint /batch-predict que reciba un archivo CSV y genere un archivo de salida con las predicciones.

Forma de entrega: La entrega final se realiza mediante la **presentación directa al docente**, mostrando en vivo:

- El entrenamiento y registro de un modelo en GridFS.
- La consulta de modelos disponibles y la activación de un modelo específico.
- La ejecución de predicciones individuales (/predict).
- La transformación por lotes sobre un dataset de prueba (/batch-predict).
- Visualizaciones y métricas obtenidas durante el entrenamiento.
- Conclusiones del análisis.

Es importante tener claro los siguientes aspectos:

- **Cambios realizados al código original**
 - Adaptación del entrenamiento para trabajar con el dataset.
 - Incorporación de conexión a MongoDB y uso de GridFS.
 - Definición de endpoints REST para consumir el modelo (predicción individual y por lotes).
- **Dificultades o ajustes presentados durante la adaptación**
 - Problemas técnicos al conectar con MongoDB o al guardar modelos grandes en GridFS.
 - Errores en el esquema de datos entre entrenamiento y predicción.
 - Ajustes en los pipelines de preprocesamiento para soportar predicciones en línea y en lote.
 - Retos al versionar modelos y activar uno como “en producción”.

¹ Una opción es utilizar **MongoDB Atlas** o crear una versión local.

² Puede utilizar una plataforma como **Replit** o crear una local.