

Diplomado en ML Cloud - UCB CBBA

Módulo 4: Machine Learning Cloud MLOps

Docente: Ing. Mauricio Alejandro Quezada

Estudiante: Jose Carlos Iriarte

Fecha : Septiembre del 2025

Laboratorio 10

Capturas del DAG con 8 Componentes Reutilizado del Laboratorio 9

The first screenshot shows the `heart_pipeline.py` script in VS Code. The script defines a pipeline with 8 components: `select_cols`, `impute`, `encode`, `scale`, `split`, `train_lr`, `score`, and `evalc`. The components are loaded from YAML files in the `components` directory. The script also sets up Azure credentials and creates a data asset for `heart-disease.csv`.

```
1 from azure.identity import DefaultAzureCredential
2 from azure.ai.ml import MLClient, dsl, Input
3 from azure.ai.ml import load_component
4 from azure.ai.ml.entities import PipelineJob, Data
5
6 SUBSCRIPTION_ID = "f1b1b1b1-b1b1-b1b1-b1b1-b1b1b1b1"
7 RESOURCE_GROUP = "rg-mlops-ucb"
8 WORKSPACE_NAME = "ws-mlops-ucb"
9 COMPUTE_NAME = "cpu-cluster"
10
11 cred = DefaultAzureCredential()
12 ml_client = MLClient(cred, SUBSCRIPTION_ID, RESOURCE_GROUP, WORKSPACE_NAME)
13
14 # Registrar dataset
15 data_asset = Data(
16     name="heart-disease-csv", path="data/heart-disease.csv", type="uri_file"
17 )
18 created = ml_client.data.create_or_update(data_asset)
19 data_ref = f"azureml:{created.name}:{created.version}"
20
21 # Cargar componentes
22 select_cols = load_component("components/select_cols/select_cols.yml")
23 impute = load_component("components/impute/impute.yml")
24 encode = load_component("components/encode/encode.yml")
25 scale = load_component("components/scale/scale.yml")
26 split = load_component("components/split/split.yml")
27 train_lr = load_component("components/train_lr/train_lr.yml")
28 score = load_component("components/score/score.yml")
29 evalc = load_component("components/eval/eval.yml")
30
```

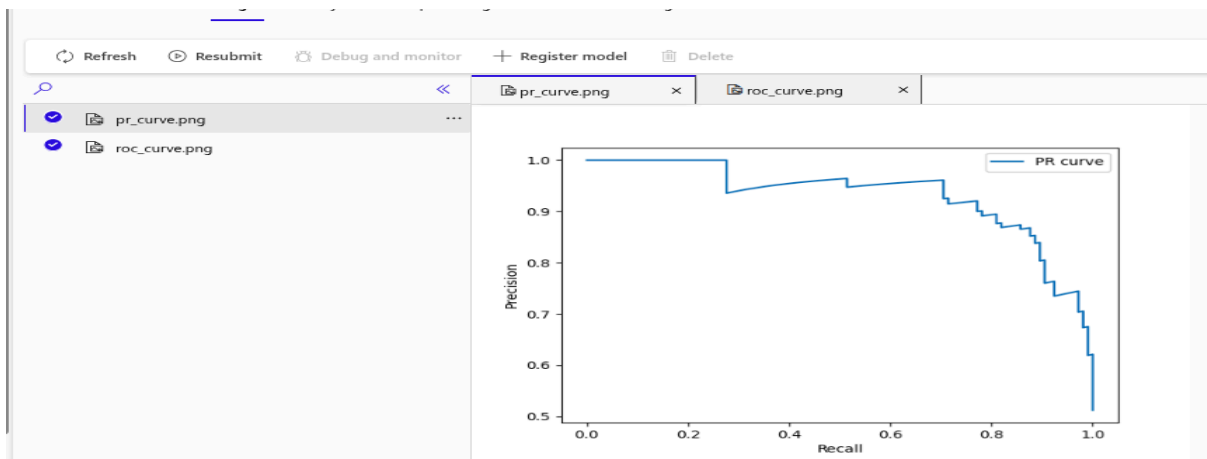
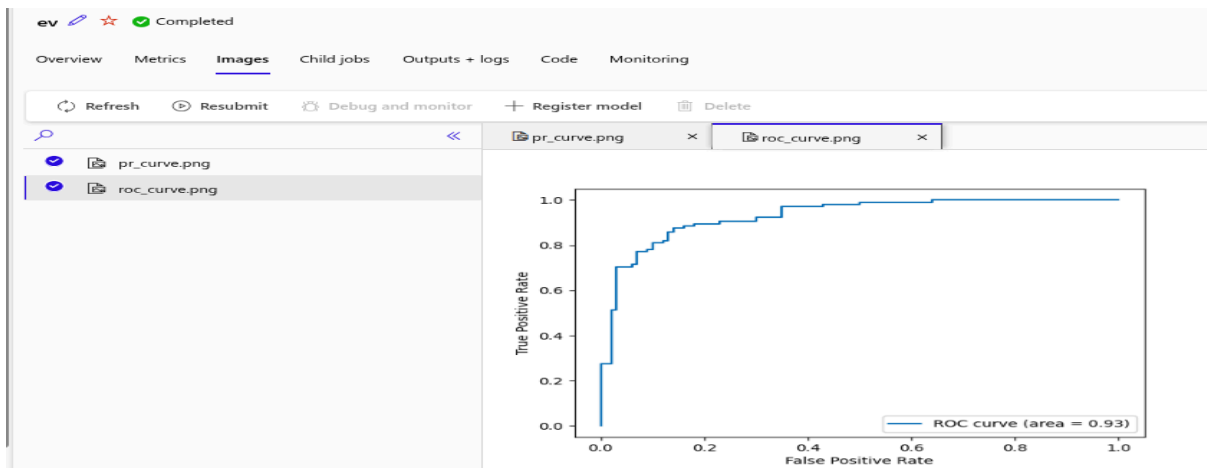
The second screenshot shows the execution output of the pipeline in the VS Code terminal. The output displays the upload progress for each component, indicating that all components were successfully uploaded.

```
uid=0(root) gid=0(root) groups=0(root)
(azureml) root@4293808435f9:/usr/src/app# python heart_pipeline.py
Class AutoDeleteSettingSchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Class AutoDeleteConditionSchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Class BaseAutoDeleteSettingSchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Class IntellectualPropertySchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Class ProtectionLevelSchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Class BaseIntellectualPropertySchema: This is an experimental class, and may change at any time. Please see https://aka.ms/azuremlexperimental for more information.
Uploading train_lr (0.0 MBs): 100% | 1789/1789 [00:00<00:00, 2121.57it/s]
Uploading score (0.0 MBs): 100% | 1405/1405 [00:00<00:00, 1908.95it/s]
Uploading eval (0.0 MBs): 100% | 2313/2313 [00:00<00:00, 2733.64it/s]
```

Captura de metrics con métricas y curvas

The screenshot shows the Azure ML portal interface for the `heart-disease-mlops` pipeline. The `Metrics` tab is selected, displaying a table of metrics for the `ev` job. The metrics are:

Metric	Value
accuracy	0.8097561
f1	0.8311688
precision	0.7619048
recall	0.9142857
roc_auc	0.9297143



Captura de Artefactos

EXPLORER

LABORATORIO-10

- components
 - encode
 - eval
 - impute
 - scale
 - score
 - score.py 3
 - score.yml

conda.yml train_lr.py 6 eval.py 4 score.py 3 outputs.py 2 metrics x

outputs > named-outputs > metrics > metrics > ...

```
1 {
2   "accuracy": 0.8097560975609757,
3   "precision": 0.7619047619047619,
4   "recall": 0.9142857142857143,
5   "f1": 0.8311688311688312,
6   "roc_auc": 0.9297142857142857
7 }
```

EXPLORER

LABORATORIO-10

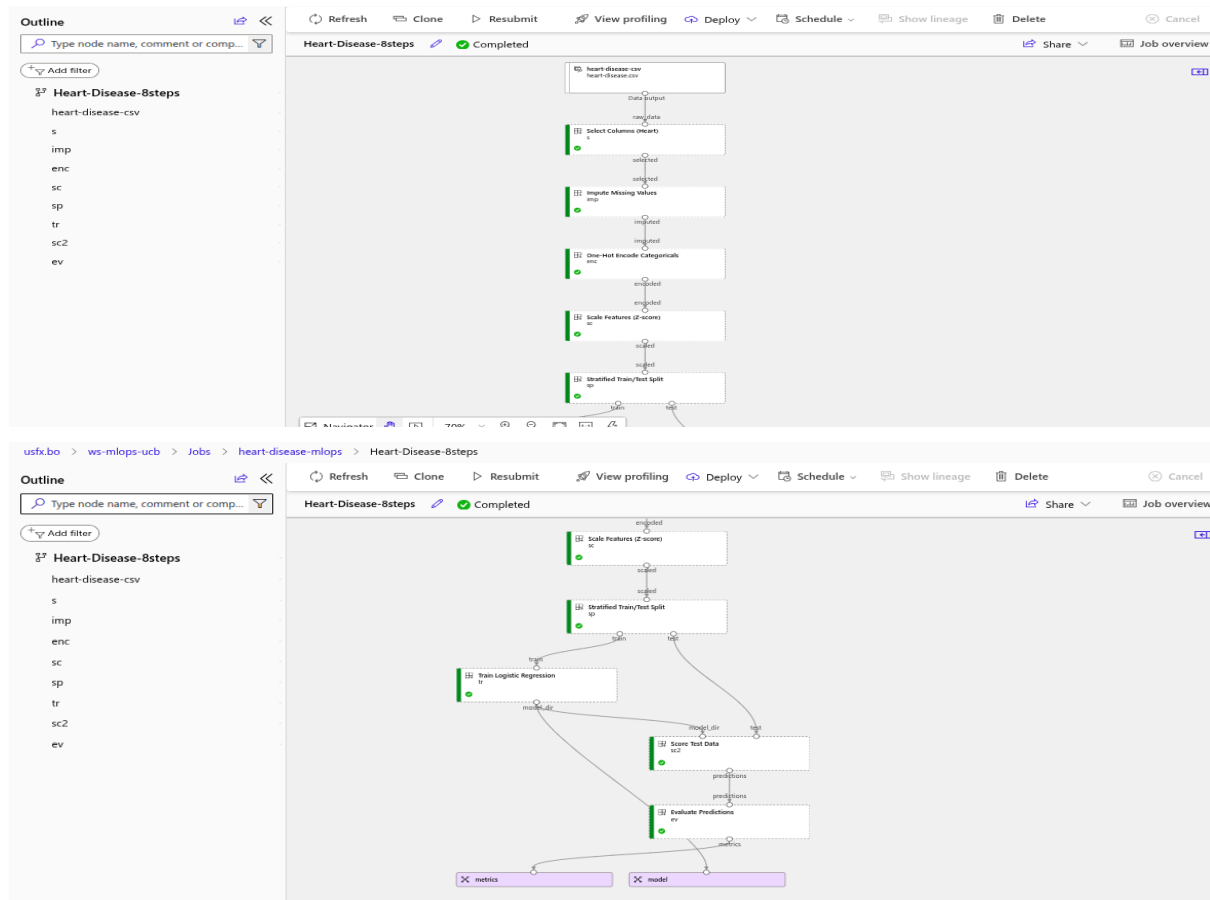
- components
 - encode
 - eval
 - impute
 - scale
 - score
 - select_cols
 - split
 - train_lr
- data
- envs
- outputs/named-outputs
 - metrics
 - metrics
 - model_dir
 - model.pkl
 - predictions
- heart_pipeline.py

conda.yml train_lr.py 6 eval.py 4 score.py 3 outputs.py 2 predictions x

outputs > named-outputs > predictions > predictions

```
1 y_true,y_pred,y_prob
2 0,0,0.011876386426733984
3 1,0,0.3713722768178209
4 0,0,0.00837225367395981
5 1,1,0.7273036986363223
6 0,0,0.2921814444825179
7 0,0,0.019212301183064922
8 1,1,0.829208291968222
9 0,1,0.53995739584032
10 1,1,0.967560405775965
11 1,0,0.3713722768178209
12 0,1,0.7363437019827234
13 1,1,0.9173460561967228
14 0,1,0.53995739584032
15 0,0,0.0727574000362988
16 1,1,0.598054484729409
17 0,0,0.01648827510423169
18 0,0,0.21013187119925933
19 0,0,0.020429666333302416
20 0,0,0.47258212583560505
21 1,1,0.8872526785057695
22 0,0,0.0070132584448762934
```

Capturas del DAG



El modelo de Regresión Logística entrenado sobre el dataset de Heart Disease obtuvo los siguientes indicadores:

- Accuracy: 0.81
- Precision: 0.76
- Recall (Sensibilidad): 0.91
- F1-score: 0.83
- ROC-AUC: 0.93

Interpretación

El modelo presenta un buen equilibrio entre precisión y recall, destacando su alta sensibilidad (91 %), lo que significa que detecta la gran mayoría de los pacientes con enfermedad. El área bajo la curva (AUC=0.93) confirma su capacidad de discriminar entre casos positivos y negativos.

Conclusión

El modelo muestra un desempeño sólido y clínicamente relevante, siendo adecuado como herramienta de apoyo diagnóstico, aunque se recomienda seguir ajustando umbrales o explorar modelos adicionales para mejorar la precisión.