

Despliegue de un modelo de Machine Learning en Azure aplicado a un conjunto de datos de predicción de cáncer (maligno o benigno)

María Fernanda Plaza Giraldo

Claudia Yaneth Valencia Morales

Trabajo final

Data Streaming y Servicios en la nube

Especialización en Analítica y Ciencia de Datos

Profesor

Reyson Mauricio Diaz González

Universidad de Antioquia

Facultad de Ingeniería

Departamento académico de Ingeniería de Sistemas

Medellín, Colombia

2021-2

Despliegue de un modelo de Machine Learning en Azure aplicado a un conjunto de datos de predicción de cáncer (maligno o benigno)

Para la implementación del modelo de machine learning se utilizó un conjunto de datos que contiene características de unas muestras de células humanas extraídas de pacientes que se creía que estaban en riesgo de desarrollar cáncer. Este modelo se entrenó con un RandomForestClassifier y la clase a predecir sería maligno o benigno.

Ciclo MLOps:

1. Creación y estructura del proyecto

Se creó un grupo de recursos (Workspace en Azure), el cual es un agrupador de componentes en un espacio de trabajo de azure ML y está compuesto por:

- Application Insights: genera las métricas de desempeño de los modelos de machine learning
- Key Vault: es una bodega que almacena las credenciales
- Storage account: es un repositorio para almacenar los conjuntos de datos y los recursos utilizados para los modelos de machine learning
- Machine Learning: genera los flujos de machine learning
- Container registry: es un repositorio para guardar imágenes de los modelos

En las figuras que se muestran a continuación están los pasos que se hicieron para la creación y la estructura del proyecto

Figura 1: Creación del grupo de recursos

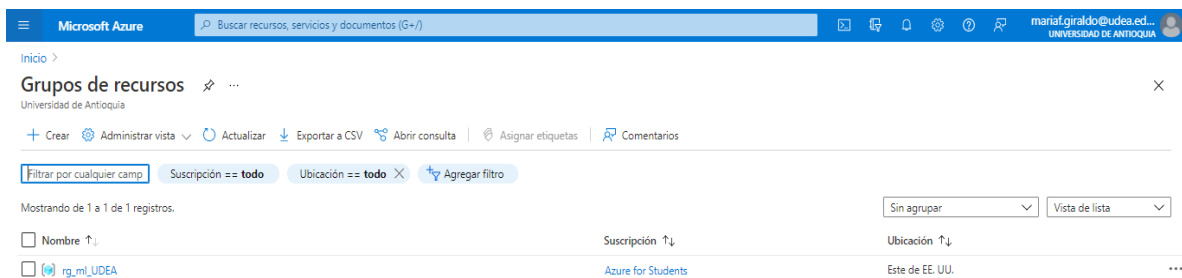


Figura 2: Espacio de trabajo de Machine Learning

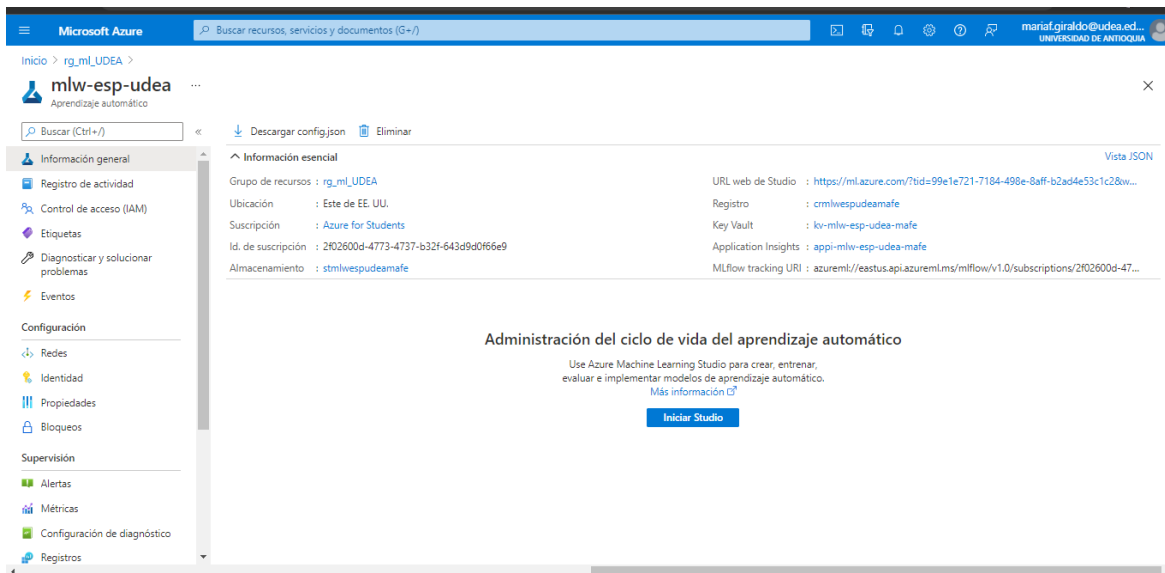


Figura 3: Creación de los ambientes de trabajo

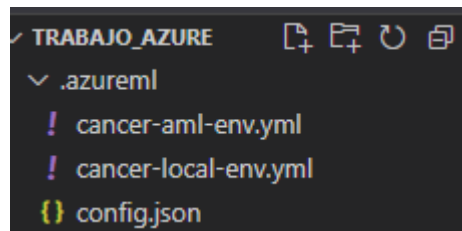


Figura 4: Configuración de los ambientes de trabajo

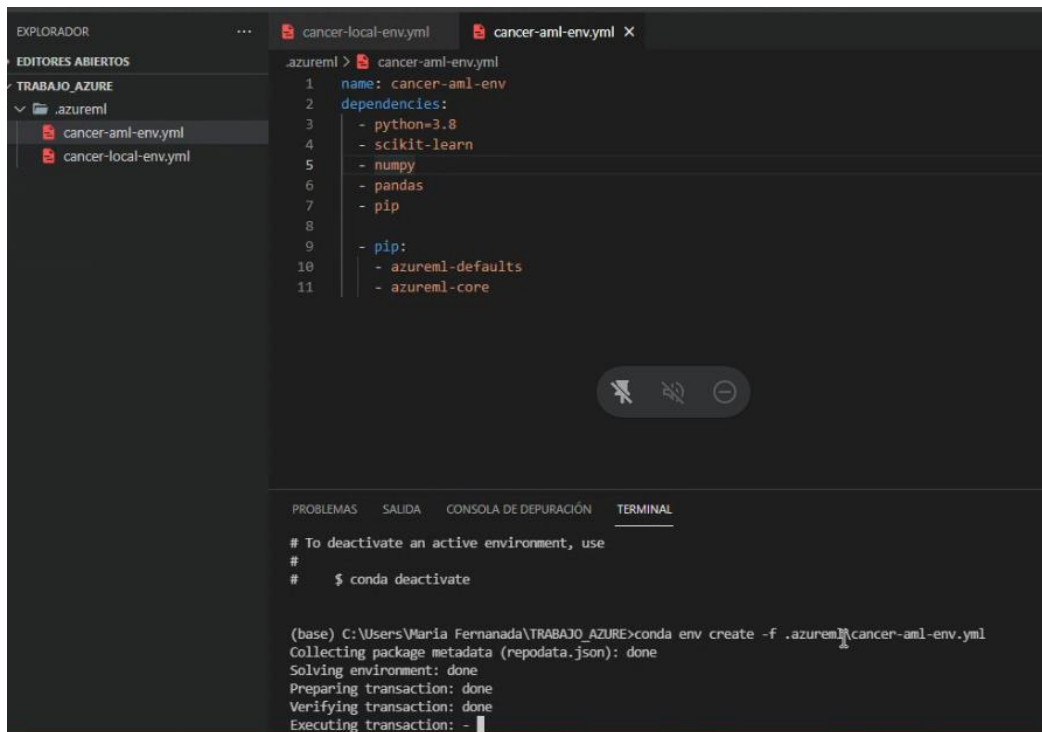


Figura 5: estructura del modelo

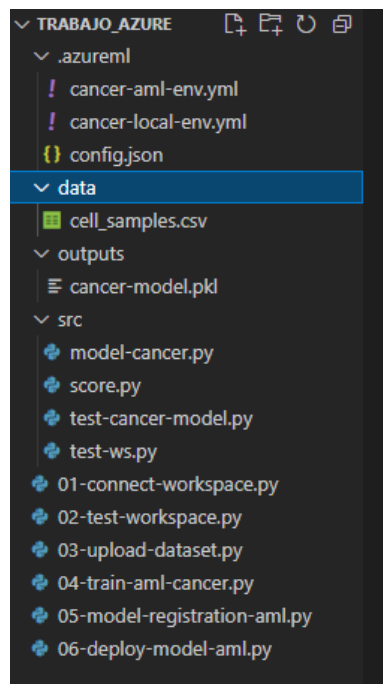
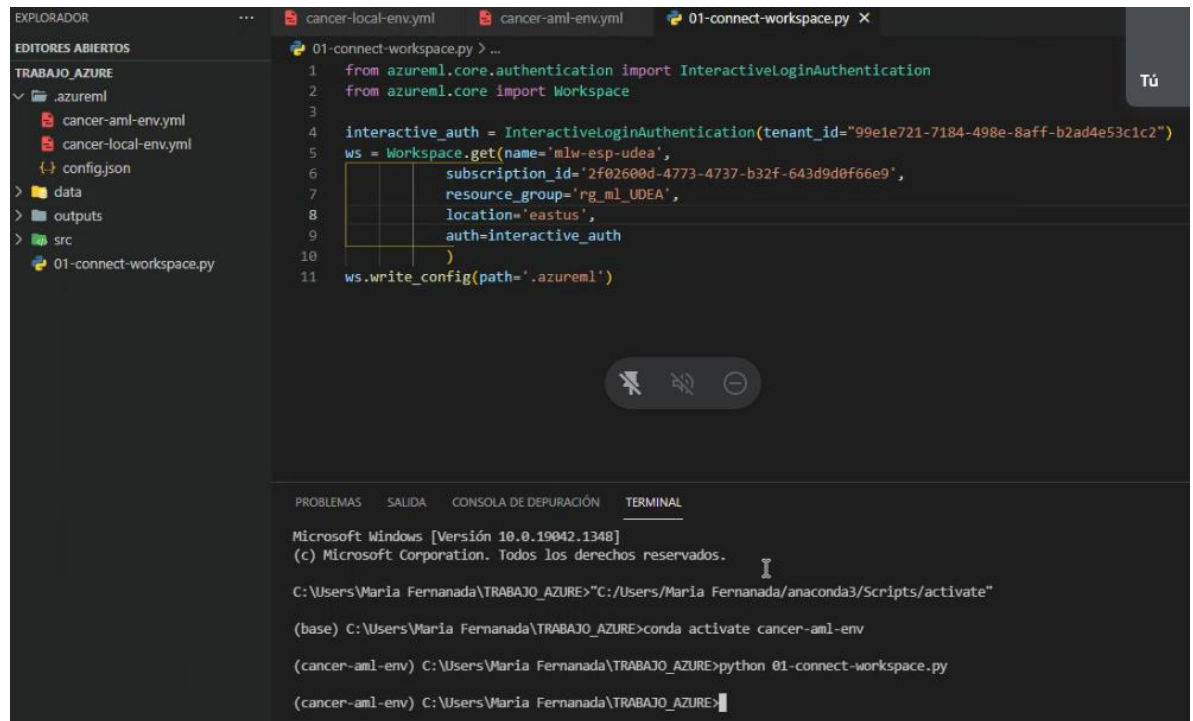


Figura 6. Conexión al workspace



The screenshot displays the Visual Studio Code interface. On the left, the Explorer pane shows a project named 'TRABAJO_AZURE' with a folder '.azureml' containing files 'cancer-aml-env.yml' and 'cancer-local-env.yml', and subfolders 'data', 'outputs', and 'src'. The file '01-connect-workspace.py' is open in the editor. The code in the editor is as follows:

```
1 from azureml.core.authentication import InteractiveLoginAuthentication
2 from azureml.core import Workspace
3
4 interactive_auth = InteractiveLoginAuthentication(tenant_id="99e1e721-7184-498e-8aff-b2ad4e53c1c2")
5 ws = Workspace.get(name="mlw-esp-udea",
6                   subscription_id="2f02600d-4773-4737-b32f-643d9d0f66e9",
7                   resource_group="rg_ml_UDEA",
8                   location="eastus",
9                   auth=interactive_auth)
10
11 ws.write_config(path='.azureml')
```

Below the editor, the TERMINAL pane shows the following commands and output:

```
Microsoft Windows [Versión 10.0.19042.1348]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Maria Fernanada\TRABAJO_AZURE>"C:/Users/Maria Fernanada/anaconda3/Scripts/activate"

(base) C:\Users\Maria Fernanada\TRABAJO_AZURE>conda activate cancer-aml-env

(cancer-aml-env) C:\Users\Maria Fernanada\TRABAJO_AZURE>python 01-connect-workspace.py

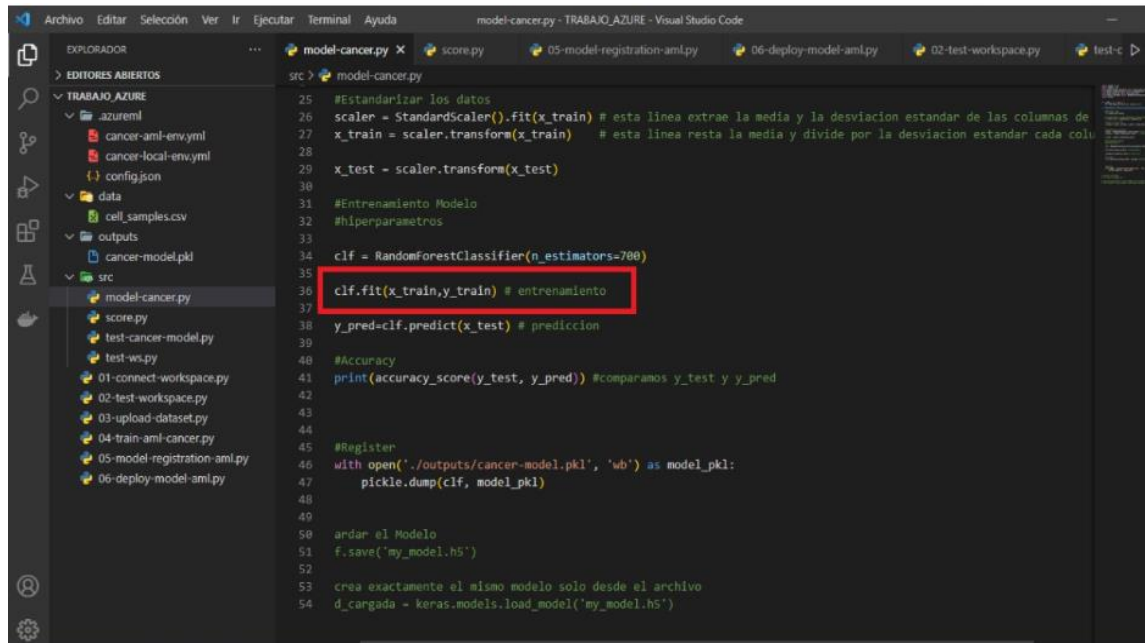
(cancer-aml-env) C:\Users\Maria Fernanada\TRABAJO_AZURE>
```

Todo lo anterior se hizo con el objetivo de generar el espacio de trabajo y la conexión entre el ambiente de trabajo local a través de Visual Studio Code (VSC) y la plataforma de Azure

2. Entrenamiento y pruebas del modelo

Se entrena el modelo en el ambiente local y se guarda en el archivo cancer-model.pkl. Este entrenamiento da una predicción del 95,9% como se muestra en la *figura 7 y 8*.

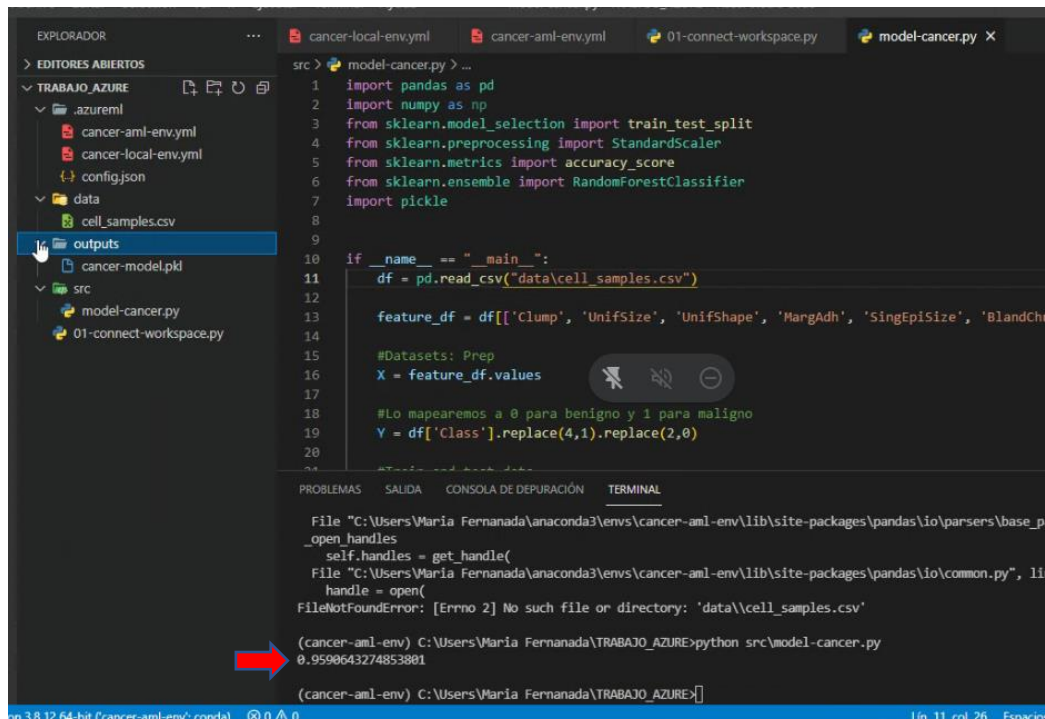
Figura 7: entrenamiento del modelo local



The screenshot shows the Visual Studio Code interface with the file explorer on the left and the code editor in the center. The file explorer shows a project structure with folders like .azureml, data, outputs, and src. The code editor displays the file model-cancer.py with Python code for training a Random Forest Classifier. The line `clf.fit(x_train, y_train) # entrenamiento` is highlighted with a red box. The code includes imports for StandardScaler, RandomForestClassifier, and pickle, and it saves the trained model to a file named my_model.h5.

```
25 #Estandarizar los datos
26 scaler = StandardScaler().fit(x_train) # esta linea extrae la media y la desviacion estandar de las columnas de
27 x_train = scaler.transform(x_train) # esta linea resta la media y divide por la desviacion estandar cada colu
28
29 x_test = scaler.transform(x_test)
30
31 #Entrenamiento Modelo
32 #hiperparametros
33
34 clf = RandomForestClassifier(n_estimators=700)
35
36 clf.fit(x_train, y_train) # entrenamiento
37
38 y_pred=clf.predict(x_test) # prediccion
39
40 #Accuracy
41 print(accuracy_score(y_test, y_pred)) #comparamos y_test y y_pred
42
43
44
45 #Register
46 with open("./outputs/cancer-model.pkl", 'wb') as model_pkl:
47     pickle.dump(clf, model_pkl)
48
49
50 #Guardar el Modelo
51 f.save('my_model.h5')
52
53 #carga exactamente el mismo modelo solo desde el archivo
54 d_cargada = keras.models.load_model('my_model.h5')
```

Figura 8: resultados del entrenamiento del modelo local



The screenshot shows the Visual Studio Code interface with the file explorer on the left and the code editor in the center. The file explorer shows the same project structure as Figure 7. The code editor displays the file model-cancer.py with Python code for loading and preprocessing data. The code includes imports for pandas, numpy, train_test_split, StandardScaler, accuracy_score, and RandomForestClassifier. The code reads data from a CSV file named cell_samples.csv and preprocesses it. The terminal at the bottom shows the execution of the script, which results in a FileNotFoundError: [Errno 2] No such file or directory: 'data\\cell_samples.csv'. A red arrow points to the error message in the terminal.

```
1 import pandas as pd
2 import numpy as np
3 from sklearn.model_selection import train_test_split
4 from sklearn.preprocessing import StandardScaler
5 from sklearn.metrics import accuracy_score
6 from sklearn.ensemble import RandomForestClassifier
7 import pickle
8
9
10 if __name__ == "__main__":
11     df = pd.read_csv("data/cell_samples.csv")
12
13     feature_df = df[['Clump', 'UnifSize', 'UnifShape', 'MargAdh', 'SingEpiSize', 'BlandCh
14
15     #Datasets: Prep
16     X = feature_df.values
17
18     #Lo mapearemos a 0 para benigno y 1 para maligno
19     Y = df['Class'].replace(4,1).replace(2,0)
20
21 #Train and Test data
```

File "C:\Users\Maria Fernanda\anaconda3\envs\cancer-aml-env\lib\site-packages\pandas\io\parsers\base_p...
_open_handles
self.handles = get_handle(
File "C:\Users\Maria Fernanda\anaconda3\envs\cancer-aml-env\lib\site-packages\pandas\io\common.py", 11...
handle = open(
FileNotFoundError: [Errno 2] No such file or directory: 'data\\cell_samples.csv'

(cancer-aml-env) C:\Users\Maria Fernanda\TRABAJO_AZURE>python src\model-cancer.py
0.9590643274853801

(cancer-aml-env) C:\Users\Maria Fernanda\TRABAJO_AZURE>

Posteriormente, se realizaron los experimentos del modelo y esto se evidencia en las figuras que se nombran a continuación:

Figura 9: ejecutando el experimento desde VSC a Azure

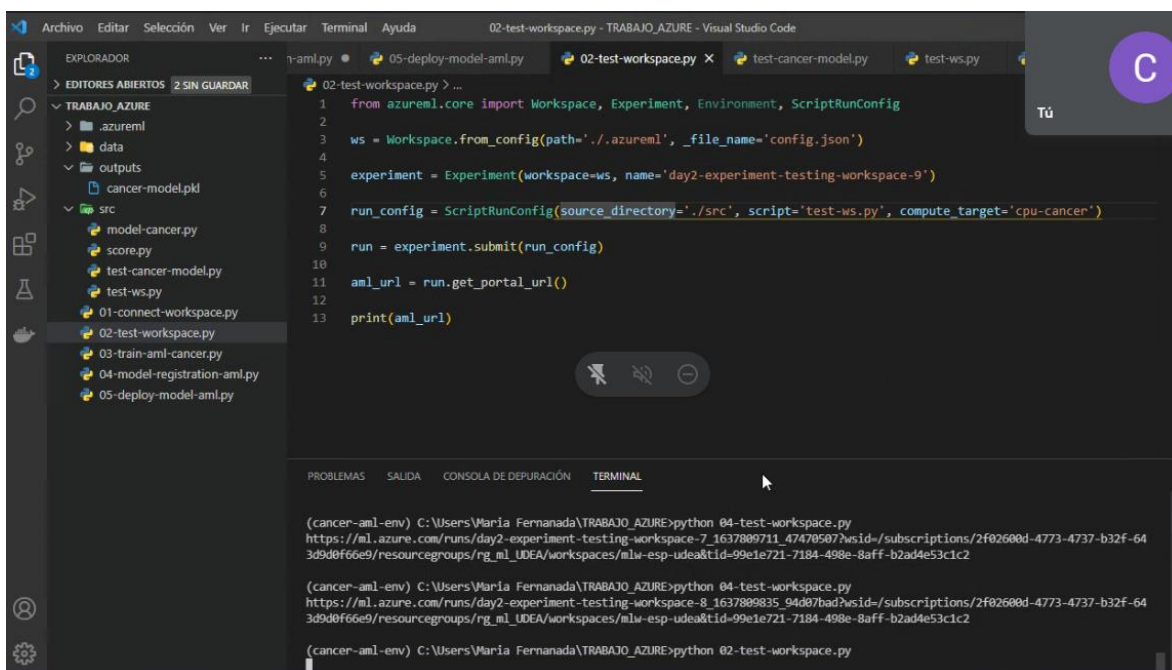
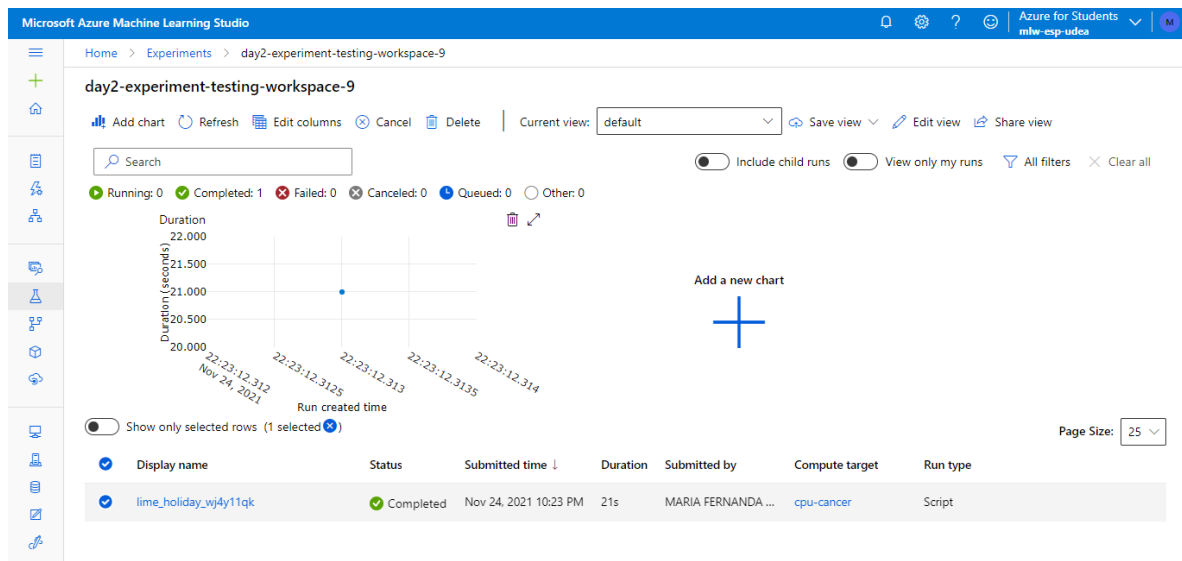


Figura 10: experimentos del modelo en la nube

The screenshot shows the Microsoft Azure Machine Learning Studio interface. The 'Experiments' tab is selected, and a list of experiments is displayed. The table includes columns for Experiment name, Latest run, Last submitted, and Created. The experiments listed are 'train-cancer', 'day2-experiment-testing-workspace-9', 'day2-experiment-testing-workspace-8', 'day2-experiment-testing-workspace-7', 'day1-experiment-testing-workspace-6', and 'day1-experiment-testing-workspace-4'.

Experiment	Latest run	Last submitted	Created
train-cancer	polite_rainbow_b85dt0l4	Nov 24, 2021 10:26 PM	Nov 24, 2021 10:26 PM
day2-experiment-testing-workspace-9	lime_holiday_wj4y11qk	Nov 24, 2021 10:23 PM	Nov 24, 2021 10:22 PM
day2-experiment-testing-workspace-8	sad_dress_qmwqs6ct	Nov 24, 2021 10:10 PM	Nov 24, 2021 10:10 PM
day2-experiment-testing-workspace-7	mango_room_ds9460z8	Nov 24, 2021 10:09 PM	Nov 24, 2021 9:43 PM
day1-experiment-testing-workspace-6	lucid_animal_pwty10jf	Nov 24, 2021 8:05 PM	Nov 24, 2021 8:04 PM
day1-experiment-testing-workspace-4	lime_tail_m7mqpsd	Nov 24, 2021 6:29 PM	Nov 24, 2021 1:19 PM

Figura 11: resultados del experimento en la nube



Luego, en la **figura 12** se muestra el dataset subido a Azure

Figura 12: Datos cargados a Azure

Microsoft Azure Machine Learning Studio

Home > Datastores > workspaceblobstore

workspaceblobstore (Default)

Overview Browse (preview)

Create dataset Refresh Update authentication Set as default datastore

This is a preview with limited file settings available. More options exist during dataset creation.

Path: https://stmlwespudeamafe.blob.core.windows.net/azureml-blobstore-86a22022-c7e5-4397-b64e-000a24fc5fdb/datasets/cifar10/cell_samples.csv

File size: 19.74 KiB

With column header: ☐

ID	Column1	Column2	Column3	Column4	Column5	Column6
1	ID	Clump	UnifSize	UnifShape	MargAdh	SingEpiSize
2	1000025	5	1	UnifSize	1	2
3	1002945	5	4	4	5	7
4	1015425	3	1	1	1	2
5	1016277	6	8	8	1	3

3. Registro del modelo

Ahora se procede a realizar el registro del modelo con el nombre de model-cancer-finalfinal

Figura 13: registrando el modelo desde VSC

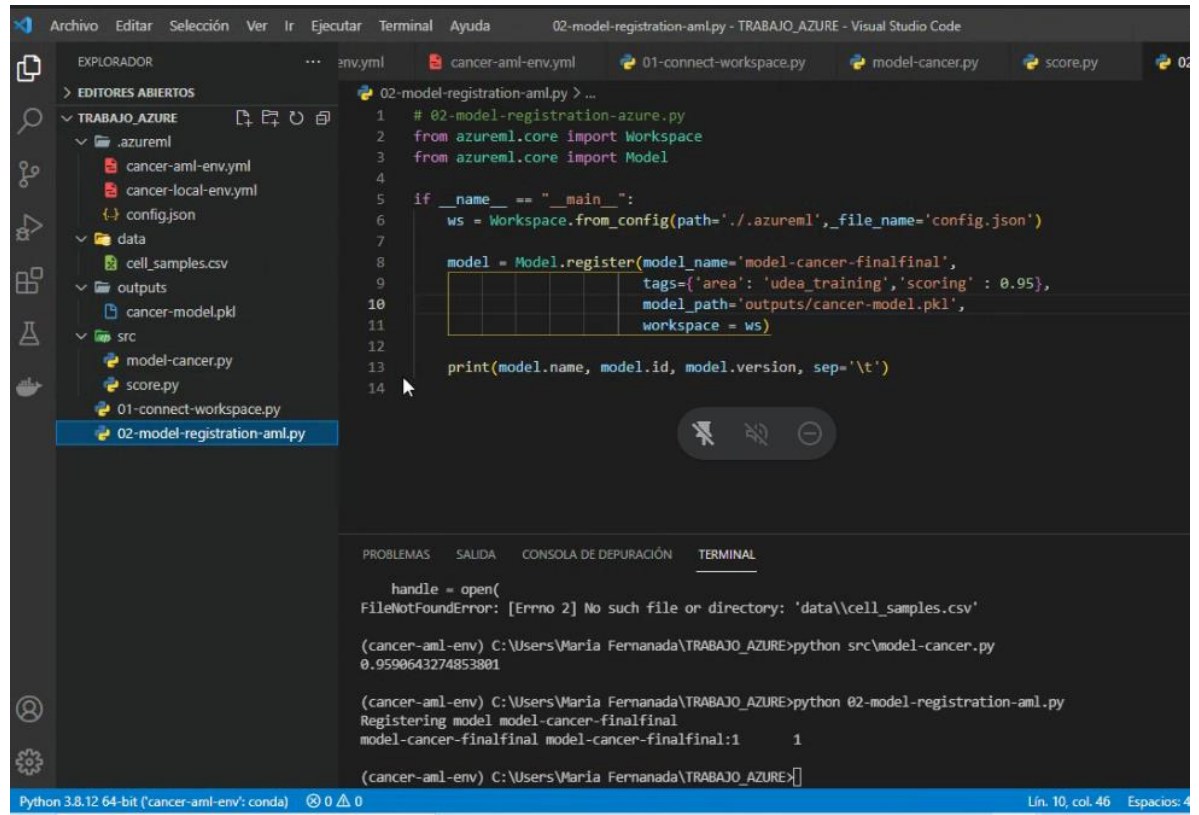
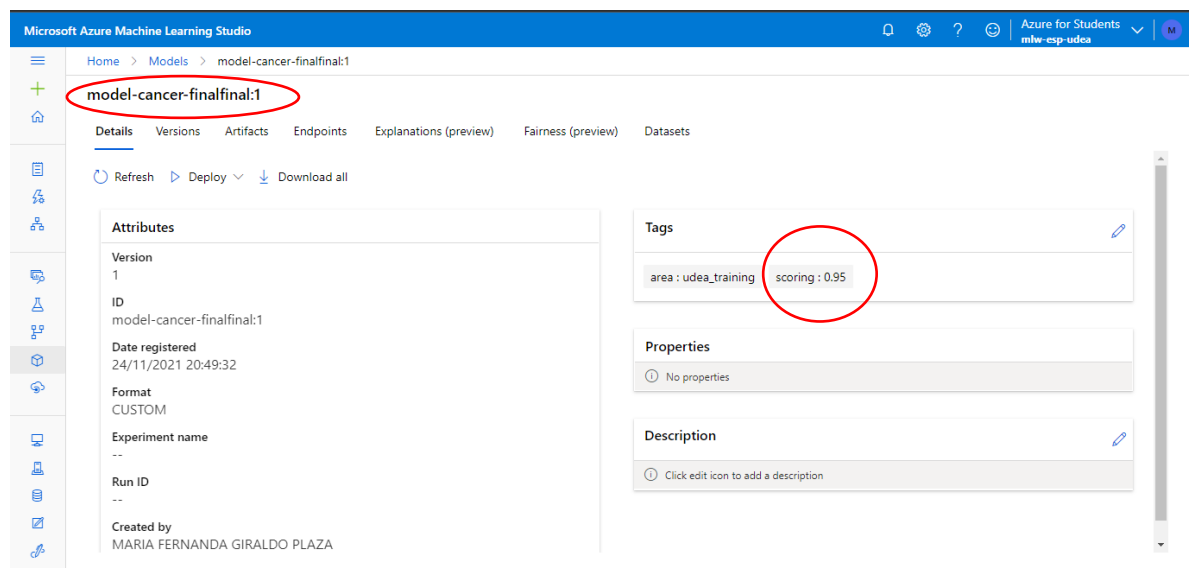


Figura 14: registro del modelo



4. Despliegue

Se despliega el modelo creado llamado cancer-model.py y este proceso se indica en las siguientes figuras

Figura 15: desplegando el modelo desde VSC

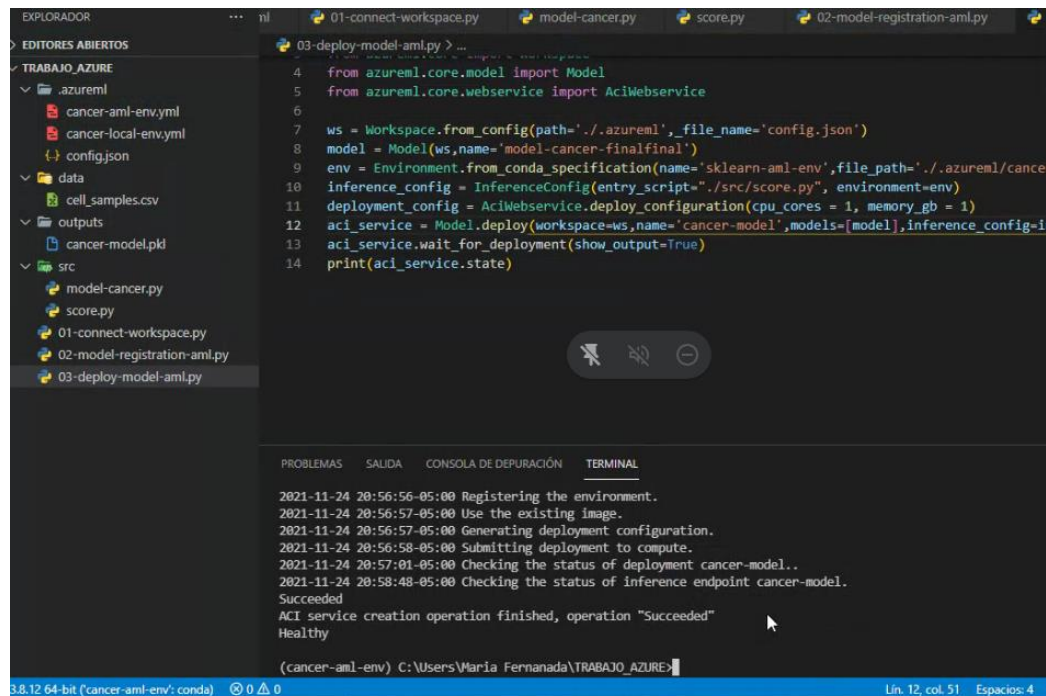
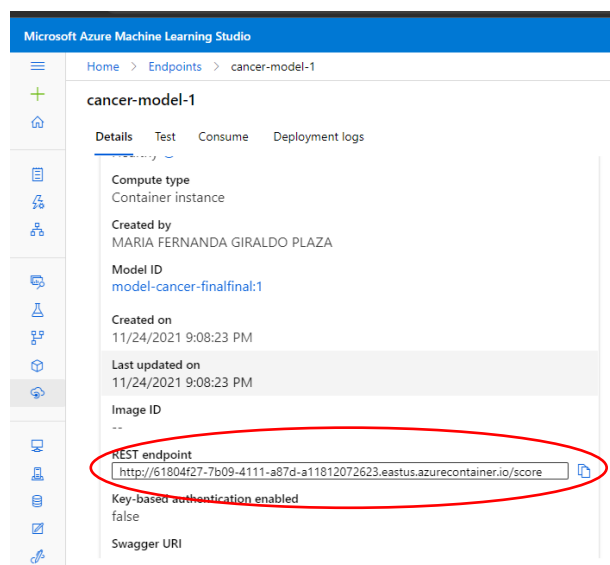


Figura 16: despliegue del modelo en la nube



Después de desplegar este modelo en la nube se genera el siguiente web service (Endpoint):

<http://61804f27-7b09-4111-a87d-a11812072623.eastus.azurecontainer.io/score>

Este endpoint se prueba en postman con los siguientes datos:

```
{
  "data":
  [
    [10,7,7,3,8,7,4,3]
  ]
}
```

Finalmente, con la información suministrada anteriormente se llega a la siguiente prueba en postman; como se muestra en la *figura17*.

Figura 17: prueba en postman

