MLOps Project

Mathias Goris & Arthur Hofman

## Explanation of our project

This project builds upon a machine learning case we explored during a previous course. The context involves a luxury hotel in the Bahamas screening **Smurf guests** for admission. Smurfs are notorious for generating both high profits and significant property damages during their stays.

### Data context

The original dataset includes records for 5000 past Smurf guests. For each guest, the dataset contains:

* **Profit made** during their last stay (outcome\_profit)
* **Whether damage was caused** (outcome\_damage\_inc)
* **Damage cost** (outcome\_damage\_amount)
* A variety of features like previous stay history, hotel facility usage, demographic data, and staff behavior scores

### Our approach

To simplify the project and improve focus on the model and deployment pipeline, we used a **cleaned version** of the dataset.

* **Model Used:** GradientBoostingRegressor from scikit-learn
* **Reason for Choice:** This model yielded the best performance in our earlier analysis of this dataset. We reused the best hyperparameters identified during that project.
* **Preprocessing:** The cleaned dataset already handled all of the missing and anomalous data. We ensured correct feature scaling and converted categorical variables where necessary before training.

## Task 1 – Cloud AI Services (Azure Machine Learning)

To train and register our machine learning model in the cloud, we used **Azure Machine Learning** services.

### Resource Setup

We first created a dedicated Azure resource group and workspace:

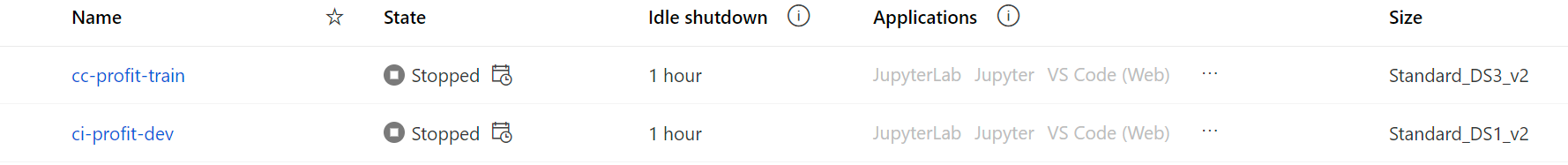
* **Resource Group:** smurf-profit-ml-gr
* **Workspace:** profit-model-ws

A white rectangular object with black lines

AI-generated content may be incorrect.

Within this workspace, we provisioned two compute resources:

* cc-profit-train: Dedicated for running Azure ML pipelines and training jobs
* ci-profit-dev: Used for development and experimentation with Jupyter notebooks



### Data Assets

We uploaded our **cleaned dataset** as a registered data asset inside the Azure Machine Learning workspace. This dataset serves as the input for the training pipeline.

A screenshot of a computer

AI-generated content may be incorrect.

### Development Workflow

For a more flexible development experience, we transitioned to **Visual Studio Code** using the **Azure CLI**. This allowed us to define and control the training pipeline programmatically.

Key configuration and training files:

* **smurf-train-env.yaml** and **conda.yaml**: Define the Python environment for training (dependencies, versions)
* **train\_regressor.yaml**: Defines the training component, including inputs, outputs, and environment
* **train.py**: Contains the training logic using GradientBoostingRegressor
* **pipeline.yaml**: Defines the pipeline which runs train.py and registers the trained model as an Azure ML model artifact

With these YAML and Python files, we were able to submit the pipeline from the command line using the Azure CLI, allowing automated model training and registration.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

### GitHub Integration

To enable GitHub Actions to interact with our Azure ML workspace (for automated training and deployment), we created an App Registration:

* **App Registration Name:** github-ml-pipeline

This registration provides secure credentials that allow GitHub to authenticate against Azure for running ML workflows from CI/CD pipelines.

A screenshot of a computer

AI-generated content may be incorrect.