**MLOps Project**

Auriane BONNOT Emma SCHAPIRA Elrich MIRANDA Solène LEBOUCHER

# Model Overview

Our **ad-click prediction tool**, written in Python, allows companies to predict if customers will engage with a specific type of advertisement shown to them based on the customer’s demographic and online behavior. It can be used by e-commerce platforms for instance to fine-tune their marketing campaigns and advertising strategies. The tool allows the user to select from three types of prediction models, train the model on historical data and use the trained model(s) to predict customer behavior for planned advertisements.

This tool comes packaged with a CLI for ease of use and a Dockerized API which you can deploy within your existing applications.

# Data Requirements

The training and prediction data is to be supplied in the CSV format. Each row must represent a customer who was shown a specific advertisement and some of the following columns useful for the prediction could be:

* Customer region (string)
* Time of the day (float)
* Day of the week (string)
* Device used by the customer (string)
* Average time spent by the customer on the previous website/app before using the app (float)
* No. of previous purchases made by the customer (integer)
* Channel on which the ad is displayed e.g. Instagram, Twitter, in-app (string)

However, the models can work on various datasets. Only the feature engineering and missing values functions are specific to some columns, which are recurrent in datasets for ad-click prediction, but the functions can be tailored to the dataset used.

In addition, the training data must contain a column indicating if the customer engaged with the ad.

# Model Usage

The ad-click prediction tool can be used either through a Command Line Interface (CLI) or via a Dockerized API, providing flexibility depending on the user’s technical environment.

The CLI allows users to train and evaluate models directly from the terminal. After cloning the repository, users can run the provided scripts to load their dataset, select a model, train it on historical data, and generate predictions. Command-line options let users specify input files, model types, and output preferences with ease.

The tool can also be deployed as a web service using Docker. To build and run the container locally, the user needs to do: docker build -t ml-cli-tool and then docker run -p 8000:8000 ml-cli-tool. This starts a FastAPI server on port 8000, which can be accessed through a browser for exemple. Additionally, a docker\_push.sh script is available in the repository to push the image to Docker Hub, enabling easier sharing and deployment across environments. Both operations ensure that we can quickly leverage the tool in different environments.