Radancy Data Science & ML Engineer Case Study Challenge

Welcome to the Radancy Data Science Case Study Challenge. This exercise is designed to assess your ability to apply data science and machine learning techniques to real-world programmatic advertising problems. You'll have the opportunity to demonstrate both technical proficiency and business acumen while tackling common challenges faced by our Data Science Programmatic Team.

Time Commitment

- Estimated effort: 2–4 hours
- Desired format: PowerPoint & concise Python notebook or Python script, including only essential code and explanatory comments
    - reports, charts, or other formats are acceptable in addition
- It's ok if you don't do everything. Just make sure what you do works well.

Business Problem

Radancy runs hundreds of programmatic recruitment advertising campaigns across multiple publishers and geographic markets. Each campaign is tailored to a specific customer and is designed to generate job applications ("conversions") at a minimized (as possible) cost per application (CPA).
However, campaign performance varies significantly depending on factors like the market (location), the publisher (ad platform), and the category of the job or the client's industry. This creates a challenge: How do we decide where to run campaigns to achieve maximum efficiency?
You've been brought in to help us:

- Predict CPA outcomes before running a campaign
- Understand what drives CPA performance
- Recommend which market and publisher combinations are likely to yield the most cost-effective results for each campaign

To solve this, you will build a CPA prediction model and use it to develop a decision-making algorithm that can guide future campaign decisions.

Dataset

`ds_challenge_data.csv`
Each row represents daily campaign-level performance across various publishers.

Column Name	Description
`date`	Date of the event
`campaign_id`	Unique campaign identifier
`category_id`	Category assigned to the campaign
`industry`	Industry of the customer
`customer_id`	Unique customer identifier
`publisher`	Platform or inventory source where the ad was shown
`market_id`	Unique market (location) identifier
`cost`	Total cost incurred
`clicks`	Total clicks received
`conversions`	Total applications

Objective

- Maximize conversions, while
- Minimizing CPA (cost per application)

Manipulable Parameters

You may experiment with and optimize:

- Publishers – e.g., identifying higher-performing platforms
- Markets – e.g., identifying higher-performing locations
- Categories – what works best for each campaign category that yields the best results
- CPC (Cost-per-click) – when we post a campaign in any ad platform, we send a bid, which is the highest cost-per-click (CPC) we're willing to pay, the actual CPC value can be treated as a variable representing the bid sent.

Technical Task

You are asked to:

1. Build a CPA Prediction Model
    - Use Python to build a model that predicts CPA based on the dataset's features.
    - Provide relevant performance metrics.
    - Justify your model's decision and describe your methodology
2. Build a Decision-Making Configuration Algorithm

- Using the trained model above, create an algorithm that selects which market and publisher to assign a campaign
- For each `customer_id`, `industry` and `category_id`, your algorithm should recommend:
    i. The best `market_id`
    ii. The best `publisher`
- Consider CPA efficiency, performance trade-offs, and any implicit constraints
- What happens for the case of new `customer_id`, `industry` or `category_id`?

3. Expose Your Model via an API (**only the CPA Prediction Model, not the Decision-Making Configuration Algorithm**)
    - Create a training script to train the model
    - Create an inference script to predict using the trained model
    - Expose the trained model as an API where you pass the features to an endpoint (an endpoint that returns predicted CPA given the input features)
        i. Make sure you provide clear instructions (in a README, script, or test) on how to call the API
4. Containerize It (**only the CPA Prediction Model, not the Decision-Making Configuration Algorithm**)
    - Create a simple docker container to encapsulate the server
    - Bonus: YAML to create a deployment and a service on a Kubernetes cluster

Evaluation Criteria

We will evaluate your submission on:

Criterion	Description
Model Performance	Accuracy and robustness of CPA predictions and decision-making derived from that model.
Business Insight	Ability to derive actionable insights and strategic recommendations
Code Quality	Readability, documentation, and structure of script
API Design	Usability and clarity of your prediction endpoint
Deployment Readiness	Dockerization and (optional) Kubernetes deployment readiness
Communication	Clarity and conciseness of your presentation or documentation