🧪 ML Engineering – Take Home Exercise without explicit mention to GitHub

Context

Our mission as the Machine Learning Engineering team at Yape is to design and build systems that continuously deliver a delightful and safe experience to yaperos and merchants. One of our biggest opportunities is building recommender systems that personalize the user experience across the platform.

The purpose of this exercise is to assess how you would go about building a recommender system using a real-world data set provided by Instacart. It is meant to give you a taste of a type of problem that you would work on as part of our team, and to give us an idea of how you would approach tackling it.

Problem Statement

Your goal is to use the data from the Instacart Market Basket Analysis competition to develop a recommender system that predicts the products that a user is more likely to purchase next. Note that this problem statement is slightly different than the one from the Kaggle competition above, where the goal was to predict the products contained in each customer's next order.

Conceptually, for a given user ID, the system must return a ranked list of the top N recommended products for that user.

```
recommend(user_id) -> [top_recommended_product_id_1,
                                                 top_recommended_product_id_
                                                 ...,
                                                 top_recommended_product_id_

```

Instructions

> These instructions are not overly prescriptive by design: we want to provide you enough details so you know what we expect, while also giving you freedom to choose how you tackle the problem.

1. Visit the Instacart Market Basket Analysis competition page on Kaggle. Read the details of the data and download it. *If you don't have a Kaggle account, you will need to create one in order to download the data.*
2. You must use Python as the programming language to develop your solution in notebooks and/or Python modules, but you are free to use any open source libraries and tools.
3. Solving this problem in production, at scale, is difficult. The intention behind this take home exercise is to see how you would approach building a quick solution to it (while learning new topics you may not be too familiar with), so **please timebox your efforts to ~6 hours**. We do not expect you to deliver production-quality code, but try to use best-practices when implementing your solution. *Tip: if you are running low on time, aim to have a suboptimal, but working solution. You can always walk us through your ideas to improve it in a follow-up meeting.*
4. **Submitting your solution**
    - You have 72 hours to submit your solution.
    - Once you're happy with your solution, share it with jorgeaescobedo@yape.com.pe and rodrigoepizarro@yape.com.pe. Please include all the notebooks and/or modules you end up writing, and include instructions to run your solution.
    - **Stretch goal:** You will get bonus points if you include a bare-bones web app as part of your solution. The wep app should take a user ID as input and display two lists:
        - The top N products recommended for the user by the recommender system, and
        - The last M products purchased by the user.

What we will evaluate

- Your overall approach to tackling the problem.
- If you can submit a working solution, or how close you get to it.
- How well you structure your solution.
- Your ideas to improve your solution if you were tasked with productionizing it at scale.
- Any ideas you may have to use this type of system in Yape.