Data Science Test

recipe similarity scoring

Background

At Gousto we sell recipe kits with the necessary ingredients in exact portions – helping our customers cook meals at home while minimising food waste. Gousto currently operates with a weekly updated *menu* of 24 *recipes*. Subscription customers receive a box with either 2, 3 or 4 different recipes with the necessary ingredients and cooking instructions, where the ingredients come in quantities to cook the recipes for either 2 or 4 people. Subscription customers can either choose their recipes for the week directly on the website or receive a box with recipes chosen for them, in their preferred quantities. In addition, people may have paused their subscription while maintaining an active user account and decide to order boxes on a transactional basis whenever they like. Recipes can be ordered through the website (through browsers on any device) or via the iOS or Android app.

The data science team at Gousto is responsible for various data products that help drive operations, target marketing efforts, as well as enable product and service features. One of those products is the *recommendation engine*. The recommendation engine consists of a collection of algorithms that work together in generating personalised recipe recommendations for each Gousto customer across all available recipes. The team is constantly exploring other ways to integrate new approaches in order to generate more accurate recommendations.

Now imagine that the team recently had a meeting with the CTO. During the meeting, a suggestion was raised to enrich the product data we leverage for recommendations with a measure of pairwise similarity between all recipes. At the end of the meeting you got tasked with finding a minimum viable approach to compute these pairwise similarities given available data.

Your objectives

In this test your task is to write a program that efficiently retrieves and processes relevant data, and generates a *similarity matrix* to be fed into the recommendation engine. Below you will find details concerning the data you may use to construct this matrix. The data comprises two randomised data sets, respectively containing data on subjective similarity evaluations given by Gousto employees ('similarity scoring data') and data on a subset of the product features we track for all recipes ('recipe feature data'). Each element of the similarity matrix should give a pairwise similarity score or coefficient between recipes - see the example below. The similarity matrix should have results for *all recipes in the recipe feature data set*. Note that the similarity scoring data only has data for a subset of these recipes If you're using Python, please return the similarity matrix as a numpy matrix, otherwise use an equivalent array-like object in your preferred language.

	Recipe A	Recipe B	Recipe C	Recipe D	....
Recipe A	1	0.67	0.3	0.47	
Recipe B	0.67	1	0.8	0.33	
Recipe C	0.3	0.8	1	0.1	
Recipe D	0.47	0.33	0.1	1	
....					

Table1: Similarity matrix example

- Similarity scoring data (similarity_scores.csv)

  This data resulted from an experiment conducted in-house with Gousto employees as test subjects. Subjects were presented with the photos for two recipes and requested to score the similarity of the recipes shown. Subject could score the similarity with either 1 (not similar at all), 2 (not really similar), 3 (similar) and 4 (very similar). Requests were sent out to random subjects at random times over a period of 6 weeks using our messaging platform. Subjects were incentivised to give multiple responses to cover as many different recipes.

  - Column names: *user_id, user_name, message_ts, action_ts, recipe_a, recipe_b, score*
  - *user_id:* unique string identifier for each test participant
  - *message_ts:* timestamp (YYYY-MM-DD hh:mm:ss) for the time of sending request to the test
  - *action_ts:* timestamp (YYYY-MM-DD hh:mm:ss) for the time of response to request by subject
  - *recipe_a:* bigint identifier for first recipe shown
  - *recipe_b:* bigint identifier for first recipe shown
  - *score:* integer score given by test subject

- Recipe feature data (recipes_info.csv)

  This is a subset of the data stored in our database for every recipe developed by our Food team. Field values are manually inputted in recipe development.

  - Column names: *recipe_id, country, country_secondary, dish_category, dish_type, diet_type, carbohydrate_base, carbohydrate_category, protein, protein_cut, protein_type, family_friendly, spice_level, prep_time*
  - *recipe_id:* identifier for recipe
  - *country:* country of origin of recipe
  - *country_secondary:* secondary country of origin of recipe - if there is not a secondary country, primary country of origin is repeated
  - *dish_category:* broad dish category of the recipe

  - *dish_type:* specific dish type of the recipe
  - *diet_type*: field indicating whether recipe is vegetarian, or contains fish or meat
  - *carbohydrate_base:* carbohydrate base of recipe
  - *carbohydrate_category:* general category of carbohydrate base
  - *protein:* main protein in the dish
  - *protein_cut:* cut of the main protein in the dish
  - *protein_type:* general category of protein in the dish
  - *family_friendly:* field indicating whether dish can be considered family friendly
  - *spice_level:* field indicating the spice level of the recipe
  - *prep_time:* time taken to prepare recipe

Some hints to help you along

1. Compare and contrast: We encourage you to try different approaches and see how the similarity matrices compare. If you can, provide a commentary on why the results may be different.
2. Combine your data sources: It may be possible to combine the subjective similarity scoring data with the recipe feature data. Consider, for instance, using the similarity scoring to generate weights in a feature-based similarity computation.
3. Aim for deployment: At Gousto, engineers as well as data scientists 'own' the code they built. This means we build and submit production-ready code that can perform at scale, handle errors and is easy to maintain for other data scientists working on new iterations in the future.
4. Ask away! Feel free to reach out and ask us any questions you may have. We'll do our best respond as soon as possible.

Time to get started!

This test is intended to give you an opportunity to demonstrate how you would tackle this problem. You are free to take your own approach to achieve the objective! Although we are interested in the results, we care most about your approach in getting the results. Therefore, it would be great to see a short overview or presentation (preferably PDF) of your key findings as well as the code (preferably in Python) that produced the results.