École Polytechnique Fédérale de Lausanne

Applied data analysis project

Fall semester 2018

Tasty and healthy recipes
Milestone 2

Authors:

Martin Chatton Lionel Pellier Lucas Massemin

November 25, 2018

Contents

1 repository architecture

The repository has been organized in a specific way. Below is a list of the different folders and their target files :

- DATA The datasets and databases files (usda, cookies, and kaggle)
- DOC Files related to the project documentation
- GENERATED code-generated files , mainly results (e.g nut_data.h5).
- NOTEBOOKS The notebooks of the project
- SCRIPTS the Perl scripts of 'From cookies to cooks'

We are aware that the expected format was a simple notebook, but having used multiple datasets, we assumed that it would be considered cleaner to treat each dataset in a different notebook.
To avoid the confusion of 'what to read first?', we state in all sections, and explicitly, the notebooks that contain the code we are referring to.

2 Introduction

The goal of the analysis is to determine what ingredients people like to eat, study their nutritional facts and come up with tasty and healthy recipes recommendations.
Nutrition balance would be an answer to junk food and malnutrition issues across the world, including diabetes and other potentially food-related diseases. Finally, tastiness would ensure the adoption of the recipes and the actual impact of the recommendations on people's lifestyles.

3 Study of tastiness

In order to get an insight into what kind of recipes people like, we decided to study existing recipes and to identify their composition patterns, if any.
We got access to the recipes by using existing datasets, namely the Kaggle "whats cooking" dataset and the "From cookies to cooks" dataset, from which we retrieved 49718 and 47045 recipes respectively. The way we preprocessed the datasets is described below.

3.1 Datasets preprocessing

For each dataset, we first extracted the list of ingredients of each recipe before applying string preprocessing operations to each ingredient. We first present the extraction process and then the string operations.

3.1.1 "From cookies to cooks" recipes extraction

The recipes in the kaggle dataset were easy to retrieve and did not require any complex extraction process. On the other hand, we had to run several scripts to get the second dataset information.
The work was done by running two scripts provided along with the dataset, *extractNutrientsFromRecipes.perl* and *extractIngredientsFromRecipes.perl*.
We first cleaned the *msg.txt* file to transform it into a list of http links stored in the *msg_clean.txt* file.
We then applied the Perl scripts and got a list of nutrients and ingredients, respectively. Those lists were put in two files, *nutrient.txt* and *ingredients.txt*.
The scripts were not run from a jupyter notebook, however, one can see the cleaning of both the *msg.txt* file and the *nutrients.txt* file in the *nutrients_cookies_recipes.ipynb* notebook.

3.1.2 ingredients names processing

Once a list of ingredients was retrieved for each recipe, we focused on mapping similar ingredients on each others and making their names as clear as possible to allow for a nutritional search in the USDA database (see section 4.2).
The work was done on all ingredients names, in the *clean_recipes_dataset.ipynb* notebook, independently on whether the ingredient came from either datasets.
We can roughly summarize the name preprocessing by the following steps :

- DECOMPOSE LINE a line might contain several ingredients
- REMOVE PARENTHESIS CONTENT usually unwanted quantity specification
- REMOVE SPECIAL CHARACTERS whitespaces, digits, etc.
- REMOVE SPECIFIC WORDS mainly quantities, see *to_remove.csv*
- 'SINGULARIZE' WORDS avoid obvious duplicates like 'egg' and 'eggs'

The cleaned files were named *clean_cookies_recipes.json* and *clean_kaggle_recipes.json*.
After cleaning, we managed to keep 98% of the data.

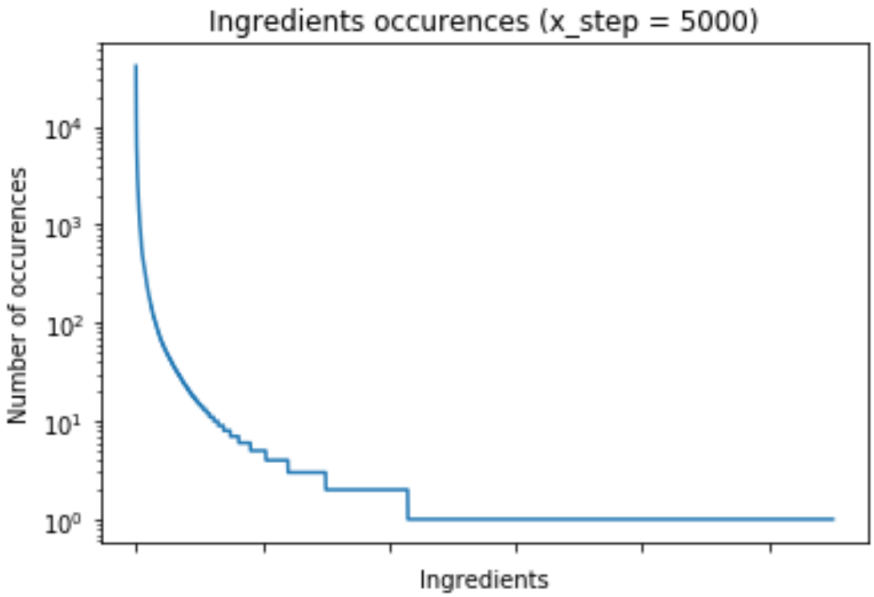

Figure 1: Ingredients distribution

We see from 1 that a lot of ingredients appear only a few times and that they are not very relevant, as the area under the curve is mostly significant for ingredients that appear at least five times. Ingredients that appear less than five times are either ingredients we were not able to parse correctly, or ingredients that are very rarely used in recipes. This is why we chose to consider only ingredients that appear at least five times in the data-set. By doing so, we keep 97% of the cleaned data which corresponds to 95% of the entire data.

	count
salt	41429
onion	25006
butter	24760
water	21339
egg	18975
sugar	18942
black pepper	17001
olive oil	16542
pepper	15976
garlic	14730

Figure 2: Ten most frequent ingredients and the number of recipes they appear in

4 Study of healthiness

The idea is to be able to recommend recipes based on the nutrients their ingredients contain. To do so, we need to have an idea of what is good in terms of nutrients intake, but also to know what nutrients are contained in what quantities in the recipes we got.
Though the ingredients quantities were not retrieved, we can still have a good idea of the recipe nutritional 'profile' by looking at the nutrients contained in its ingredients.
The 'good' amount of nutrient to take per day has been defined according to the US dietary reference intakes (RDI). A table was created under the file *RDI.xlsx* with all nutrients values stated in mg unit.
The challenge lied in finding the nutritional value for the ingredients contained in the recipes we got from the datasets. Our attempts are described below.

4.1 Wikipedia nutrients scrapping

The first attempt consisted in scrapping vegetables and fruits names from Wikipedia, before going to their wikipedia page to retrieve their nutritional values (usually stored in a table). This was done in the *nutritional_facts_parsing.ipynb* notebook, and provided the nutritional facts for 213 food items.

4.2 USDA database linking

One concern with the wikipedia retrieval is that recipes ingredients might as well be meat, baking products and many other types of food not included in our

dataframe. For that reason, we decided to find another way to get nutritional information, one that would provide us with more food items information. Luckily enough, it occurs that the wikipedia nutritional information was retrieved from a larger database, the USDA food composition database.Note that USDA is acronym for United States Department of Agriculture, this department maintains a database including many food items, along with their composition.
We thus chose to get this database and to link it to our ingredients, which was done in *USDA_parsing.ipynb* by completing the tasks described below.

4.2.1 Files description

The first step was to download the ascii files using the download site and to isolate the useful files, namely :

- FOOD_DES.TXT a description for each food item
- FOOD_GROUP.TXT a description of each food group
- NUT_DATA.TXT the nutrient composition of each food item
- NUTR_DEF.TXT the definition of nutrients

4.2.2 Database filtering

The USDA database contains many groups of food items, amongst them 'Beverages', 'Snacks' and 'Pork Products' to cite some. Although the Pork Products category might contain some of our ingredients, it is much less likely to find some in 'Snacks'.
We decided to drop categories to avoid false positive when searching an ingredient, and to make the search quicker.
We also define the comma-separated fields of the *long_description* column as 'categories'. We study those categories and decide to drop the ones that are not likely to describe our ingredients, like 'pan-fried' or 'roasted'.

4.2.3 Search function

"Linking" the USDA database to our recipes means being able to find the USDA food entry corresponding to each ingredient. To do so, we define a search function which, given an ingredient, finds the most-likely-to-match food entry in the USDA database. We first pre-process the food item descriptions to generate search words, and then design a score function to quantify the matching.

The search words are words providing information on each food items entries in USDA. They are generated by cleaning the *common_names* column of the *FOOD_DES.txt* file ('singularization', 'lower-casing', etc.) and by prepending it to the *long_description* column.

The design of the score function starts from three straightforward observations, namely :

- Lots of categories usually mean complex food item
- Categories are ordered by descending relevance
- Some categories (e.g 'raw') do not make the food item more complex

Note that the 'categories' that do not make the food-item more complex were chosen empirically.
Based on the previous observations, we design the score function so that it takes the query (the ingredient entry, a list of words) and does the following for each food item in the USDA database :

```
def search_score(categories, ing_words) :

    # singularize search words
    ing_words = set([singularize_word(x) for x in ing_words])

    #prioritize matching query terms
    nb_matching = len(ing_words.intersection(set(categories)))

    #non_complexificators should not be penalized,ignore them AFTER computing number of matching words
    categories = [c for c in categories if (c not in non_complexificators)]


    #matching keywords one by one
    matching = [len(set([x]).intersection(ing_words)) != 0 for x in categories]


    #first keywords are more important
    weights = np.linspace(2, 1, num=len(matching))
    weights = weights / sum(weights)

    #the query should have as many ingredients words as possible
    score = (10 * nb_matching) + sum([c[0] * c[1] for c in zip(matching, weights)])

    return score
```

Figure 3: USDA score function

In the above example, 'categories' refers to the search words.

4.2.4 Nutrients and RDI linking

Now that our ingredients are mapped to the database entries, we would like to facilitate the study of their nutrients composition in the context of RDI. To do so, we first filter the nutrients of the USDA database (149 distinct !) to match the 35 contained in the RDI file. Then, we create a table which, for each food_id, gives the absolute amount of nutritients, as well as their RDI percentage for men and women.

4.2.5 Results storage

The USDA linking resulted in two different files.
The first file is named *ing_usda_mapping.json*, it consists of the mapping between ingredients names as found in the recipes, and their food_id in the USDA table.
The second file is named *nut_data.h5*, it contains the nutritional table which links the food_id with the food nutrients composition.
You can also find interesting statistics in the *stats.ipynb* notebook.

5 Future work

Now that the datasets have been cleaned, The following actions will be undertaken :

- Run the Apriori algorithm to detect 'classic' ingredients associations
- Apply a community algorithm on the graph of ingredients to see whether we are able to identify several types of ingredients (probably different nationalities)
- Identify similar ingredients through a connectivity analysis, suggest replacements to balance the recipes nutritional values

A more complete list can be found under *doc/research_ideas.txt* in the repository.