Open Food Facts

Link: <https://www.kaggle.com/datasets/openfoodfacts/world-food-facts>

Texts that can be inserted in to the notebook: - Xiaoyun Yu

Assess the dataset:

**Quality: Is the data reliable? On what basis do you make that judgement?**

Yes. The dataset is provided directly by the owner called “open food facts”, and Kaggle team. There are 542 codes already discussing about this dataset. And it even has an AI repository. We see this is a popular dataset, with over 71k downloads.

**Detail: How much detail is there? Is the information helpful?**

There are tremendous details. This dataset includes 356027 rows × 163 columns, although some columns are empty or sparse that should be discarded. Each food entry also has a URL which is a webpage containing tidy visualizations for its brief info. The columns contain not only the food’s ingredients, but also the import country and city, shop, and creator.

**Documentation: How clear is it what the data means? Where was the documentation? Was it easy to find?**

Is lacks of source/provenance, and column descriptions. This is one of its drawbacks. However, we can infer what each column means just by its names. Also, Kaggle’s preview functionality is powerful for us to have an overview for each column’s values. It says that “the data is a combination of crowdsourced information gathered via mobile phones, producer data, and data extracted from the first two sources using various machine learning, OCR, and regex techniques.” (<https://www.kaggle.com/datasets/openfoodfacts/world-food-facts>)

**Interrelation: Would it be useful to connect to other data sets? Which? How easy would that be?**

It is difficult to find another related dataset, since this dataset is mainly based on each of the specific food names. It is rather difficult to find the specific food names. However, we may find another dataset with country-city entries to be combined with this dataset.

**Use: What could you use it for? What questions would you like to ask the dataset but can't? What's missing?**

We can:

(1) Do classification using all necessary columns against “main\_category” column.

(2) Do classification using all necessary columns against “countries\_tags” column.

(3) Use other columns to predict “nutrition-score-uk\_100g” column value.

(4) Use other columns to predict “energy\_100g” column value.

(5) Use other columns to predict “salt\_100g” column value.

(6) Use other columns to predict “calcium\_100g” column value.

(7) Find any relationships between the countries and “nutrition-score-uk\_100g” column.

(8) Environmental Threat Analysis: Use an XPath approach to extract a Boolean flag indicating whether a product poses a species threat.

(9) Nutritional Trends & EDA: Investigate the correlation between high sugar content, additive levels, and nutrition scores across countries. (Inference)

We can’t:

(1) Analyze the Eco-score directly from the table, since that score is a missing column.

**Discoverability: How easy was it to find open data in your chosen domain? Where did you go? Were there many alternatives?**

For food data, there are tons of food images classification data in Kaggle. However there are few with plain data.

**Terms of use / license:**

[Database: Open Database, Contents: Database Contents](http://opendatacommons.org/licenses/dbcl/1.0/)

**Interests:**

**Difficulties:**

(1) Countries and cities are ambiguous. In some columns, they are mixed, separated with commas; in other columns, there are only countries, or there are codes such as “brignemont-haute-garonne-france” that are hard to understand. Since there are non-UDF characters, and multiple natural languages, difficulty of data validation and entity resolution (coreference). For example, “en:GB”, “United Kingdom” and “Royaume-Uni” are the same thing.

(2) There are many columns with collinearity. For example, similar columns: “ingredients\_from\_palm\_oil\_tags”, “ingredients\_from\_palm\_oil”, “ingredients\_from\_palm\_oil\_n”.

(3) “ingredients\_text” column is hard to expand, since there are parentheses and commas (layered structure), and the ingredient’s name may be different for the same ingredient (entity resolution).

(4) Since it’s lack of metadata, it’s hard to infer for some technical terms, such as “E307c - Tocopherol” and “fr:filet-de-boeuf”.

(5) Imbalanced dataset. For example, there are predominant foods which are from the United States.

(6) Because of the different serving\_size for each entry, we can’t directly compare some of the numerical values of each entry.

To solve the issues of some anomalies that may occur during our management of the database, and to maintain ACID properties, we need to normalize the database. Normal anomalies may include: update anomaly, insertion anomaly, deletion anomaly, modification anomaly etc. Normal forms define sets of attributes that ensure relations are properly structured. They are progressive, with each successive normal form imposing increasingly stringent requirements.

|  |  |
| --- | --- |
| 1NF |  |
| ↓ | eliminate the partial functional dependencies of non-prime attributes towards keys |
| 2NF |  |
| ↓ | eliminate the transitive functional dependencies of non-prime attributes towards keys |
| 3NF |  |
| ↓ | eliminate the partial transitive functional dependencies of prime attributes towards keys |
| BCNF |  |

The 1st normal form emphasizes on **Atomicity**. “A relation is in first normal form (1NF) if and only if all the domains in which its attributes are defined contain scalar values only.” (Lewis, Normalisation, p. 113) For this dataset to be in the 1st normal form, we should expand each of the multiple-value element to make each cell only containing one element. The “code” column shouldn’t be set as the unique ID, since there are only 355839 values in that column, and we have 356027 rows. We can set the “product\_id” as the primary key for each row. However, for the “states” columns, we can just leave multiple-value elements there (or separate these into another table), since the “states” is a summary for all the other info.

States:

|  |  |  |  |
| --- | --- | --- | --- |
| states | States\_tags | states\_en | product\_id |
|  |  |  |  |

**Not satisfying 2NF: Some attributes are not determined by the primary key**. For example, creator is not determined by product\_id.

A relation is in second normal form (2NF) if and only if: 1. it is in 1NF; and 2. **every non-key attribute is irreducibly dependent on the primary key** (Lewis, Normalisation, p. 113). It deletes the partial functional dependency of non-prime attributes towards keys.

We need to create one table for the creator, and use product\_id as a foreign key.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Creator\_id | creator | created\_t | created\_datetime | last\_modified\_t | last\_modified\_datetime | product\_id |
|  |  |  |  |  |  |  |

For packaging and brands, we can temporarily see these as properties for each food product.

For city and country, we should create separate tables.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | origin | origin\_tag | manufacturing\_place | manufacturing\_place\_tag | product\_id |
|  |  |  |  |  |  |

City-country:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| id | city | city\_tag | purchase\_place | country | country\_tag | country\_en | product\_id |
|  |  |  |  |  |  |  |  |

Country-food:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | country | country\_tag | country\_en | product\_id |
|  |  |  |  |  |

Categories:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | category | categories\_tag | category\_en | product\_id |
|  |  |  |  |  |
|  |  |  |  |  |

Stores:

|  |  |  |
| --- | --- | --- |
| store\_id | store | product\_id |
|  |  |  |

States (leave as what it is):

|  |  |  |  |
| --- | --- | --- | --- |
| product\_id | states | States\_tags | states\_en |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | emb\_code | emb\_codes\_tag | first\_packaging\_code\_geo | product\_id |
|  |  |  |  |  |