Scientific paper - Assignment 1

1 Introduction

The objective of this project is to clean and treat data via the creation of a data-pipeline, to then make it suitable to be processed by multiple machine-learning programs.

Those machine learning programs will consist of: A genetic algorithm, who will have to represent assumed taste, cost, health impact, and environmental impact. After that, an Auto-encoder will have to be used on features of products inside the chosen dataset

The results of the application of those algorithms will then be used to predict the best possible product in terms of flavor, nutriment balance, price, and environmental impact.

An analysis of the results will then allow us to focus on the best possible product with the goal of generating with a diffusion model a close representation of the product characteristics.

2 Related Work

2.1 Deb et al with NSGA-II (2002)

This thesis was mainly about NSGA-II

Despite not implementing NSGA-II, it was interesting to know about the concept, and some comparisons can be made with parts of my GA.

The early version of my GA is run with a few different weight settings to see the trade-offs, then keep the options that aren't beaten on all goals. I use tournament selection, blend crossover, small random nudges, and strict feature bounds.

2.2 Genetic algorithms (GA) for adaptable design

This thesis by Jansson, Andreas Dyrøy builds an interactive GA for design, a "lock/keep" option for good attributes, and a small preference model that learns from user likes.

The app attached to it was interesting in the way of showing the modularity of the model with the customizable face features.

I borrow the practical bits for my snack search: hard bounds and rules (e.g., caps on sugar/salt, price ranges), seeding near real products, and locking useful traits when I explore good candidates.

I may add a light preference re-ranker later, but for now I select using my overall score plus quick human checks. The goal is realistic and there are suggestions that are possible to try.

3 Methods

3.2 Dataset research

The idea of this part of the project was to find suitable data to gain time on training quality, to get better results. Research was focused not only on a single dataset, but on complementary datasets to complement each other's by merging columns.

3.2.1 Openfoodfacts

Openfoodfacts is one of the biggest free and open datasets on kaggle.com. I choose to use this dataset because of its diversified and large amount of data, who contain food from all European countries, the USA, UK, and many Asiatic countries including China.

3.2.2 CarbonFootprint

This dataset was choosen for its light and accessible data, with the objective of using it as a basis to generate synthetic data with LLM's service, to suit openfoodfacts product list.

3.2.3 Food_fr

Food_fr is a fork of the Openfoodfacts dataset. A lighter version focused on french product who was verry usefull to test implementations of the data-pipeline du to harware limitations.

3.3 Research of a existing project

The goal was to focus on code quality and progressive steps, by starting with an already existing project. The main source of this part of the research was focused on Kaggle, GitHub, and hugging face.

3.3.1 Lee Jacobson tutorial

This tutorial was the basis for the creation of this project auto-encoder. With it's straight foward method, provided code, and explaination, it helped me getting a base ga that i could complexify with the addition of new features.

3.3.2 Autoencoder repository

This repository is about creating a autoencoder with the aim of completing missing data in food/nutriment dataset. This project was choosen as a basis for the creation of an autoencoder, because of its proximity with this paper subject.

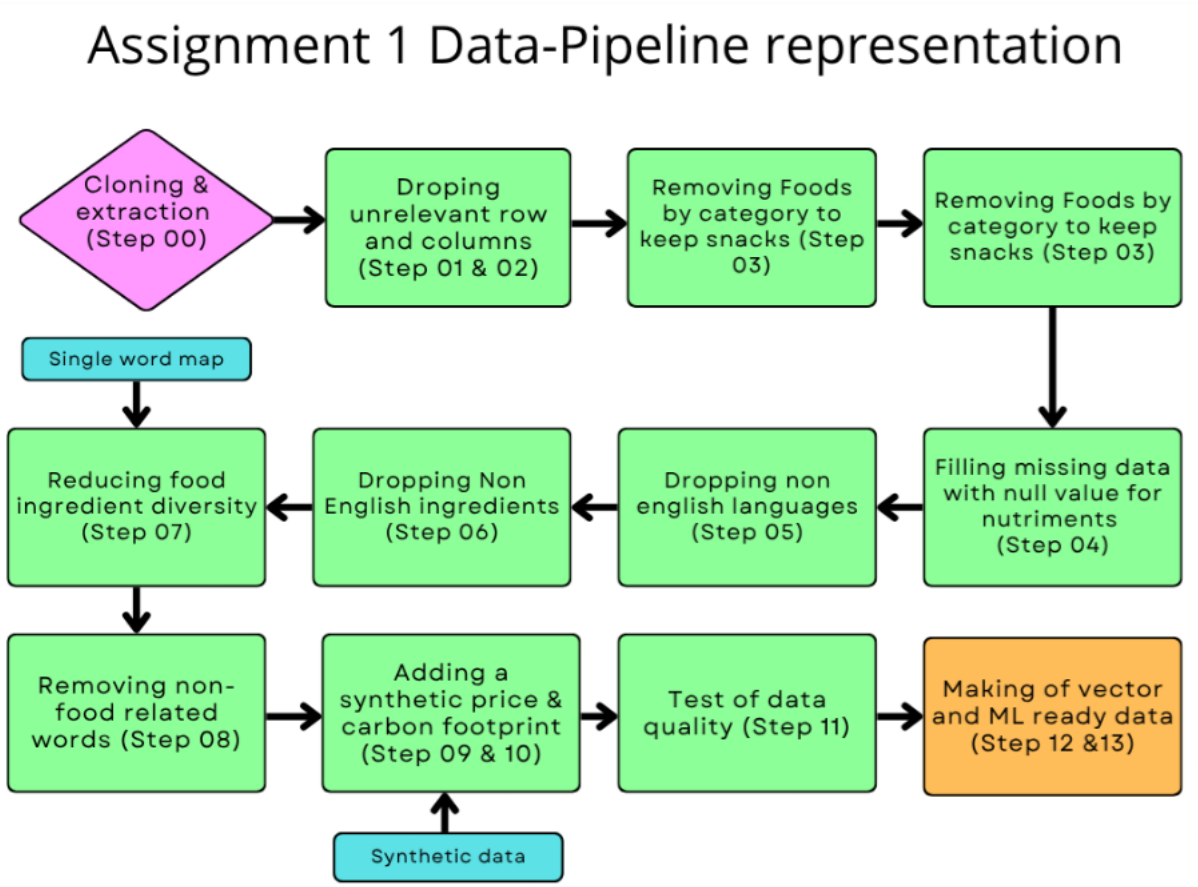
3.4 Data-Pipeline

3.4.1 Structure

Early structures like cutting the main dataset in half were much simpler but didn't provide good data, or worse, deleted too much information.

Because of that I have chosen to create a sequential and more organized structure. Each file of this part of the project does a single step, from non-relevant column to single row cleaning. During the execution of the pipeline, with fallback error management to avoid bad results, but also to follow what's going on in the pipeline during process.

Diagram of the data pipeline



3.4.2 Pipeline Algorithm

The pipeline's algorithm are step-by-step ETL (extract, transform, load).

I first clean and normalize the ingredient tokens, then fill missing nutrient values with safe defaults, scale all numeric features to 0–1 using quantile clipping to handle outliers, enrich each product with synthetic price and CO_2 estimates from small synthetic dictionaries, and finally assemble everything into a compact feature vector ready for the GA and autoencoder.

3.5 Genetic Algorithm

3.5.2 Algorithm used

The genetic algorithm that i implemented is an evolve-and-select loop: I seed a population with real products (plus a little noise), score each candidate with one overall score that balances taste, price, health/processing, and environment, then pick good

parents, mix their values, add small random nudges, clip to realistic bounds, keep a few of the best, and repeat for several generations.

3.6 Autoencoder

3.6.1 Autoencoder

My autoencoder structure was heavely inspired by the auto-encoder repository. It learns a “latent space” of the snack features: it first scale every column to comparable ranges (nutrients, log-minmax for price/CO2, unit scaling for already-normalized fields, and a simple 1–5 health mapping).

i trained a MLP encoder–decoder with a sigmoid output and an extra head that predicts the 1 to 5 health class; I balance classes each epoch, monitor a validation split, and stop early when it stops improving.

After training, I export latent vectors (x_latent.csv), wich was at first intended to try to do a reconstruction back in the original units (x_recon_full.csv), but it was never used in my app and was in the end out of the subject and not working.

3.5 Synthetic data

3.4.1 Use of LLM services (GPT-5)

Synthetic data was created with online LLM services, to create data fast with verry specific content. This method was used to create synthetic prices, carbon footprint, but also the single word map and the non-related word map, from a single word list extracted from the database.

The consequences of this choice resulted in incoherent and partially hallucinated dataset, with some foods items being over evaluated in terms of prices.

4 Results

4.5 Results based on step 4 analysis

3.4.1 two-dimensional map

Distances roughly reflect similarity, but the axes themselves have no fixed meaning. I read the map by local neighborhoods and density: tight pockets suggest similar nutrient/ingredient patterns. I color points by simple labels (category or health/processing band) to make patterns easier to spot and to check that clusters aren't artifacts.

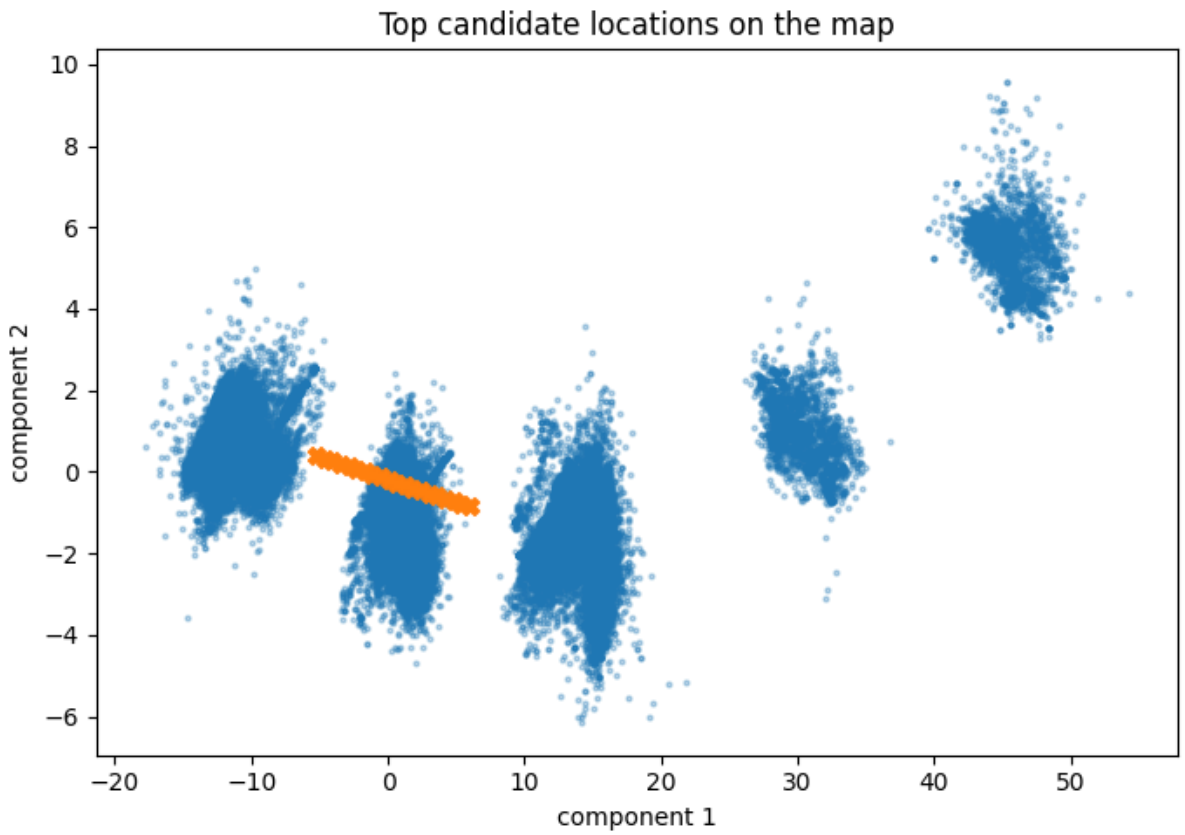
3.4.1 groups

I see clear pockets that line up with product types: a sweet/chocolate area, a salty chips/crisps area, a nuts/seeds area, and a lighter fruit-based area. What separates them is mainly the mix of sugar, fat, salt, and fiber, with a processing/health signal pushing ultra-processed items toward one side and simpler ingredient lists toward another. Price and CO_2 enrichment shift positions slightly when ingredients differ a lot.

3.4.1 how I generated candidates

I used First, interpolation between two strong exemplars to blend desirable traits; decoding keeps me near known, realistic regions. Second, small bounded perturbations around a high scorer to explore safe variations without drifting into nonsense. After decoding, I recheck feasibility (nutrient bounds, price/CO2 sanity) and keep a short trace so I can explain how each

Plot representation of top candidates per locations based on ga v2



candidate was formed.

5 Discussion

5.1 limits and improvements

5.1.1 GA and autoencoder

First iterations of my GA were completely biased, with results showing only successful outputs. This was due by making the mistake of computing already added data from taste, price, and carbon. Since i tried to first predict taste that were originally added together in my pipeline.

The V2 of the Ga also has its fields of improvement, mainly on the optimization of the parameters of generation. The Ga is also affected by the many bad generated synthetic data, who misspelled some products words.

The autoencoder wasn't optimized, and also had room for improvements, and orignal implementations.

Old results, autoencoder V1

```
taste_balance,price_norm,health_score,carbon_norm
1.0,0.0,1,0.0
1.0,0.0,1,0.0
1.0,0.0,1,0.0
1.0,0.0,1,0.0
```

new results, autoencoder V2 with incoherent food combinaisons

```
ingredients_combo,taste_balance,price_norm,health_score,carbon_norm
vanilla + flax + pear + coffee + monostearate + starter + flageolet,0.9702775795935954,0.4169381280313319,4,0.3999326597959562
seed + cream + chickpea + cumin + chestnut + kiwi + chive + shrimp + flageolet,0.9893437737002319,0.5246315642767323,3,0.4780604359994538
```

5.1.1 Pipeline and vector results

The main mistakes of the pipeline were probably to be too complex. But this was also a consequence of the nature of the chosen dataset. With over 200 columns, 4 millions rows, and lots of missing data, this dataset would have need proper implementation of real data from real dataset instead of the rush implementation of synthetic data.

The main results of the over complex structure were many missed steps at the beginning of the project, who translated of a loss of development time, and rushed solutions to improve the data quality as much as possible.

6 Conclusion

Despite its complexities and challenge, this project was verry interesting for practicing Genetic algorithm and discovering its uses and applications.

The project also highlighted the importance of having clean and repayable data, and that the quality of the results of machine learning algorithms is directly dependent of data quality and diversity.

Concerning the Auto-encoder, the limitations came from my understanding of the project, and the verry limited possibilities of improving the work of the inspired original project on GitHub. Despite this, some results have been experimented and extended my interest in this kind of ml algorythm.

Results were verry mitered in my own oppinon, but improvements were made along the many modifications made both in the data pipeline and in the GA itself.

7 References

7.1 papers

https://sci2s.ugr.es/sites/default/files/files/Teaching/OtherPostGraduateCourses/Metaheuristicas/Deb_NSGAII.pdf

7.2 Datasets

https://huggingface.co/datasets/openfoodfacts/product-database/tree/main

https://huggingface.co/datasets/openfoodfacts/product-database/tree/main