Data Analytics

Recipe Recommender

Elnara Kulijeva

March, 2023

Table of Contents

1. Introduction

In today's fast-paced world, people are increasingly looking for quick, easy and delicious meals to cook at home due to time constraints or food costs. With so many recipe options available online, it can be challenging for users to find recipes that match their tastes and preferences. In this project, I aimed to solve this problem by building a recipe recommendation system using data analysis techniques.

The aim of this report is to provide a detailed overview of our recipe recommendation system, including data pre-processing and cleaning, feature engineering, model selection, evaluation metrics and deployment process. I started by exploring a large dataset of recipes and user interactions, which I used to train our recommendation model. The system is designed to provide users with personalized recipe recommendations based on their ingredients.

To achieve this, I will use one of the machine learning models such as cosine similarity and unsupervised clustering, which are common algorithms used in recommendation systems. I will evaluate the performance of my model using the metrics of accuracy, precision, recall and F1 score. I then plan to deploy a model as a web page via 'streamlit' to allow users to access personalized recipe recommendations in real time.

Throughout this report I will provide a detailed overview of my methodology, results and future recommendations for improving my recipe recommendation system. I believe that my system has the potential to revolutionize the way people search for and discover new recipes, making home cooking more accessible and enjoyable for everyone.

2. Plan

- Acquire the data from data source
- Importing the files in Python
    - Preprocessing and cleaning the data
    - Feature engineering for more developed features
    - Exporting to MySQL for processing
- MySQL
    - Building a database and entity relationship
    - Get EER model
    - Create 5 interesting queries
- Explanatory analysis
    - In Python
    - In Tableau
- Building ML model
    - Using Cosine Similarity
    - Trying Clustering Unsupervised learning
- Creating Recipe Recommender
- Implement in Streamlit* (depends on time)

3. Data and data sources

In this project, was used 2 data sets from Kaggle.com, which was scrapped from a famous recipe portal food.com via Python (using Requests and BeautifulSoup). This dataset consists of 231 637 rows and 12 columns for the recipes and 1 132 367 rows and 5 columns for the review from users.

The data exported from Kaggle consist of 2 data files:

- RAW_recipes.csv
- RAW_interactions.csv

The data that 'RAW_recipes.csv' provides:

RAW_recipes.csv provided features	
Name	Recipe name
Id	Unique ID of recipe
Minutes	Minutes to prepare recipe
Contributor_id	User ID who submitted this recipe
Submitted	Date recipe was submitted
Tags	Food.com tags
Nutrition	Nutrition information
N_steps	Number of steps in recipe
Steps	Text for recipe steps
Description	User-provided description

Example of the raw data review:

```
1 raw_recipes.head()
```

Python

	name	id	minutes	contributor_id	submitted	tags	nutrition	n_steps	steps	description	ingredients	n_ingredients
0	arriba baked winter squash mexican style	137739	55	47892	2005-09-16	['60-minutes-or-less', 'time-to-make', 'course...	[51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0]	11	['make a choice and proceed with recipe', 'dep...	autumn is my favorite time of year to cook! th...	['winter squash', 'mexican seasoning', 'mixed ...	7
1	a bit different breakfast pizza	31490	30	26278	2002-06-17	['30-minutes-or-less', 'time-to-make', 'course...	[173.4, 18.0, 0.0, 17.0, 22.0, 35.0, 1.0]	9	['preheat oven to 425 degrees f', 'press dough...	this recipe calls for the crust to be prebaked...	['prepared pizza crust', 'sausage patty', 'egg...	6
2	all in the kitchen chili	112140	130	196586	2005-02-25	['time-to-make', 'course', 'preparation', 'mai...	[269.8, 22.0, 32.0, 48.0, 39.0, 27.0, 5.0]	6	['brown ground beef in large pot', 'add choppe...	this modified version of 'mom's' chili was a h...	['ground beef', 'yellow onions', 'diced tomato...	13
3	alouette potatoes	59389	45	68585	2003-04-14	['60-minutes-or-less', 'time-to-make', 'course...	[368.1, 17.0, 10.0, 2.0, 14.0, 8.0, 20.0]	11	['place potatoes in a large pot of lightly sal...	this is a super easy, great tasting, make ahea...	['spreadable cheese with garlic and herbs', 'n...	11
4	amish tomato ketchup for canning	44061	190	41706	2002-10-25	['weeknight', 'time-to-make', 'course', 'main-...	[352.9, 1.0, 337.0, 23.0, 3.0, 0.0, 28.0]	5	['mix all ingredients& boil for 2 1 / 2 hours ...	my dh's amish mother raised him on this recipe...	['tomato juice', 'apple cider vinegar', 'sugar...	8

At the first look, we could already see the several columns that need to be split and columns that we will not need in further analysis.

The data that ‘RAW_interactions.csv’ provides:

RAW_interactions.csv provided features	
User_id	User ID
Recipe_id	Recipe ID
Date	Date of iteration
Rating	Rating given by user
Review	Review text

Example of raw data review:

```
1  raw_interactions.head()
```

	user_id	recipe_id	date	rating	review
0	38094	40893	2003-02-17	4	Great with a salad. Cooked on top of stove for...
1	1293707	40893	2011-12-21	5	So simple, so delicious! Great for chilly fall...
2	8937	44394	2002-12-01	4	This worked very well and is EASY. I used not...
3	126440	85009	2010-02-27	5	I made the Mexican topping and took it to bunk...
4	57222	85009	2011-10-01	5	Made the cheddar bacon topping, adding a sprin...

4. Data collection

Kaggle is a popular online community for data scientists and machine learning practitioners to share, collaborate and compete on data analytics projects. It provides a vast repository of datasets that are openly available for download and use in data analytics projects. Kaggle's datasets cover a wide range of domains, including health, finance, education, sports, and entertainment, among others.

So, I chose it because of the availability of large and diverse datasets: Kaggle hosts one of the largest collections of datasets publicly available for data analysis. These datasets come from different sources, in my case from a popular food portal "food.com".

Due to the limited time, I decided to spend more time on data analysis, machine learning and recommender in order to provide more interesting insights and build good explanatory analysis based on my research on the given datasets, also trying to implement all the knowledge gained during this bootcamp. As well, additional task will be to check if all recipes match to second dataset and could be merged during the process.

5. Data cleaning and preprocessing for analysis

Stage 1. Data Cleaning

Searching and removing 'missing values' in 'raw_recipes', for this part, I had only 1 missing name of recipe, which I tracked down and filled up, I had as well many 'description' column missing value, due of not exploiting this column, it gives us no problem.

```
raw_recipes.isna().sum()
#filling missing value in name column
raw_recipes.loc[raw_recipes['name'].isna(), 'name'] = 'Salad Dressing'
```

Also, in 'raw_reviews' I found the 'review' column had some missing values, but as I do not use this column, it is not a problem either.

Then I implemented other data preprocessing manipulations, such as:

Apply lowercase to all string values in the DataFrame

```
raw_interactions = raw_interactions.applymap(lambda s: s.lower() if type(s) == str else s)
raw_recipes = raw_recipes.applymap(lambda s: s.lower() if type(s) == str else s)
```

#Convert the list of strings in the ‘steps’ column to a numbered list of strings

```
raw_recipes['steps'] = raw_recipes['steps'].apply(lambda x: [f"{i}. {step.strip()}" for i, step in
enumerate(x.split(','), start=1)])
```

#Remove unwanted characters from the ‘steps’ column

```
unwanted_chars = "[]\"'/\\,"
raw_recipes['steps'] = raw_recipes['steps'].apply(lambda x: x.translate(str.maketrans('', '',
unwanted_chars)))
```

In this dataset no duplicates found, no high correlation and no low variance detected.

Stage 2. Feature engineering in my dataset

To make predictions and improve performance, I decided to transform the data into a feature that would capture the food types across all recipes, as well as nutrition columns that would help visualize the differences between recipes.

I started by creating 'nutrient' features by splitting the column for each nutrient and cleaning it up.

```
#creating columns for nutrition and fill from 'nutrition' column
recipes[['calories','total fat','sugar','sodium','protein','saturated fat','carbohydrates']] =
recipes.nutrition.str.split(",",expand=True)
#removing square brackets from columns
```

```
recipes[['calories', 'carbohydrates']] = recipes[['calories',
'carbohydrates']].applymap(lambda x: x.replace('[','').replace(']',''))
# making nutri columns float
recipes[['calories','total                                    fat','sugar','sodium','protein','saturated
fat','carbohydrates']].astype('float')
#drop the nutri column
recipes.drop(['nutrition'], axis =1, inplace=True)
```

As I moved on, I created a 'food type' column, based on food classification as the result of my own metrics, to have some identification for future use:

```
Gluten-Free                                  76350
Gluten-Free, Low-calories                    71906
Balanced                                     37481
Low-calories                                 28673
Veg dessert, Gluten-Free, Low-calories        3392
Veg dessert, Gluten-Free                      2702
Veg                                           2533
Non-Veg dessert                               2118
Veg dessert, Low-calories                     2062
Non-Veg dessert, Low-calories                 1813
Veg dessert                                   1376
Non-Veg dessert, Gluten-Free                   879
Non-Veg dessert, Gluten-Free, Low-calories     352
          Name: food_types, dtype: int64
```

First of all, I created 'food types' column, and created a list of 'unique_ingredients' from the dataset of recipes, I cleaned it from unwanted characters. After I looked for gluten-free recipes in the dataset and got the answer of 305 gluten-free recipes, so I created research based on common gluten ingredients to try to define the number of these recipes:

```
#creating filter for recipes which may contain gluten
gluten_ingredients = ['wheat flour', 'all-purpose flour', 'breadcrumbs', 'bread',
'barley','rye', 'malt','couscous', 'bulgur', 'spelt', 'farro' , 'cookies']
count_gl= 0
count_nogl = 0
for i in range(len(recipes['ingredients'])):
    if any(gluten_ingredient in recipes['ingredients'][i] for gluten_ingredient in
gluten_ingredients):
        count_gl +=1
    elif any(word in recipes['ingredients'][i] for phrase in gluten_ingredients for word in
phrase.split()):
        count_gl += 1
    else:
        count_nogl += 1
```

```
print(f'So we have {count_nogl} gluten-free recipes, and {count_gl} recipes containing gluten')
```

Response: `So we have 155581 gluten-free recipes, and 76056 recipes containing gluten`

Finally, I started to create all my defined food types according to my formula, as shown below:

```
#looping through data to check for type of food 'veg'
for i in recipes['ingredients'].index:
    if 'eggs' not in recipes['ingredients'][i]:
        if 'ice-cream' in recipes['ingredients'][i] or 'chocolate' in recipes['ingredients'][i] or 'cookies' in recipes['ingredients'][i]:
            recipes['food_types'][i] = 'Veg dessert'
    elif 'eggs' in recipes['ingredients'][i]:
        if 'ice-cream' in recipes['ingredients'][i] or 'chocolate' in recipes['ingredients'][i] or 'cookies' in recipes['ingredients'][i]:
            recipes['food_types'][i] = 'Non-Veg dessert'
```

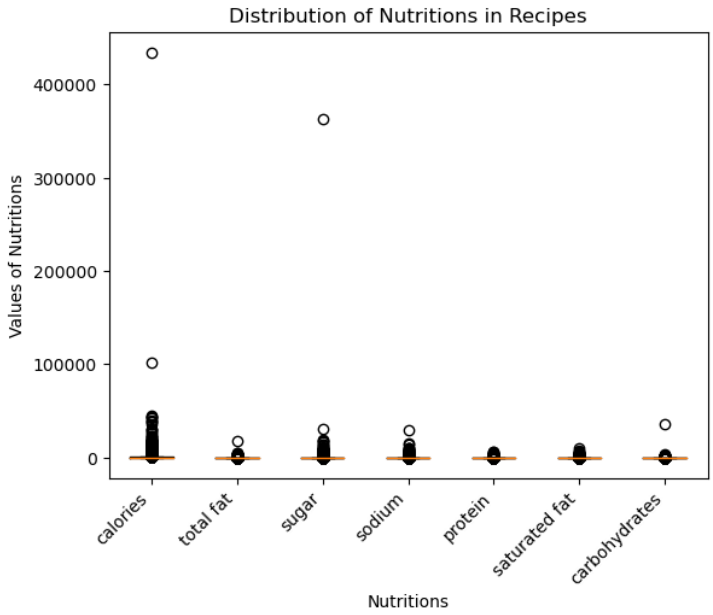
As the result of this feature, I have 13 new classification.

Stage 3. Dealing with the outliers

I started by converting the nutritional columns to numerical ones, to be able to see if there were any problems. After looking at the summary table of the descriptive statistics, I could see, that there were some interesting moments, that needed to be plotted to see more clearly.

	minutes	n_steps	n_ingredients	calories	total fat	sugar	sodium	protein	saturated fat	carbohydrates
count	2.316370e+05	231637.00	231637.00	231637.00	231637.00	231637.00	231637.00	231637.00	231637.00	231637.00
mean	9.398550e+03	9.77	9.05	473.94	36.08	84.30	30.15	34.68	45.59	15.56
std	4.461963e+06	6.00	3.73	1189.71	77.80	800.08	131.96	58.47	98.24	81.82
min	0.000000e+00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25%	2.000000e+01	6.00	6.00	174.40	8.00	9.00	5.00	7.00	7.00	4.00
50%	4.000000e+01	9.00	9.00	313.40	20.00	25.00	14.00	18.00	23.00	9.00
75%	6.500000e+01	12.00	11.00	519.70	41.00	68.00	33.00	51.00	52.00	16.00
max	2.147484e+09	145.00	43.00	434360.20	17183.00	362729.00	29338.00	6552.00	10395.00	36098.00

In the graph below, you can see visible outliers for the numerical columns and for the column of ‘number of ingredients’, which are in question after consulting them in the summary statistics table:



After visual confirmation, the outliers were successfully removed by dropping formula:

```
recipes = recipes.drop(recipes[recipes['sugar'] > 350000].index, axis=0)
recipes = recipes.drop(recipes[recipes['sodium'] > 20000].index, axis=0)
recipes = recipes.drop(recipes[recipes['protein'] > 4500].index, axis=0)
recipes = recipes.drop(recipes[recipes['saturated fat'] > 10000].index, axis=0)
recipes = recipes.drop(recipes[recipes['total fat'] > 17000].index, axis=0)
recipes = recipes.drop(recipes[recipes['n_ingredients'] > 25].index, axis=0)
```

Stage 4. Continue to work on feature engineering

Creating a ‘rating_mean’ in ‘rating_new’ dataset, preparing it to add only the mean rating to main recipe dataset by merging them by ‘recipe id’.

The next step was to create a ‘unique ingredient’ dataset, by creating a unique ID for each unique ingredient found in the main recipe set, splitting each recipe by ingredient, and assigning a unique id to each food type.

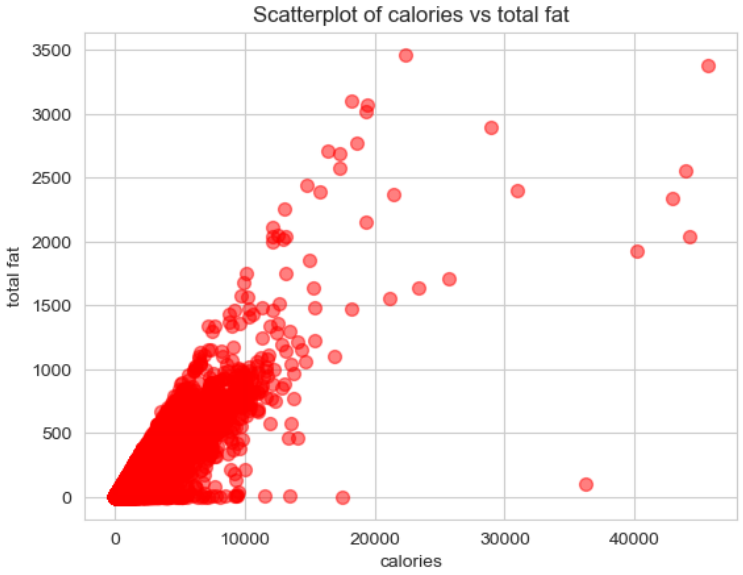
At the end, in possession 5 tables like this:

- food_types - unique food type with its proper ID number (shape 13, 2)
- nutrition – all nutrition for every recipe (by recipe ID) (shape 231 396,8)
- recipe_ingredient – recipes split to ingredients with its unique ID (shape 2 096 962, 2)
- unique_ingr – every unique ingredient with its proper unique ID (shape 14 905, 2)
- recipe – main table, with all ids and features like minutes, number of steps and ingredients, as well as steps by text and mean rating for every recipe (shape 231396, 8)

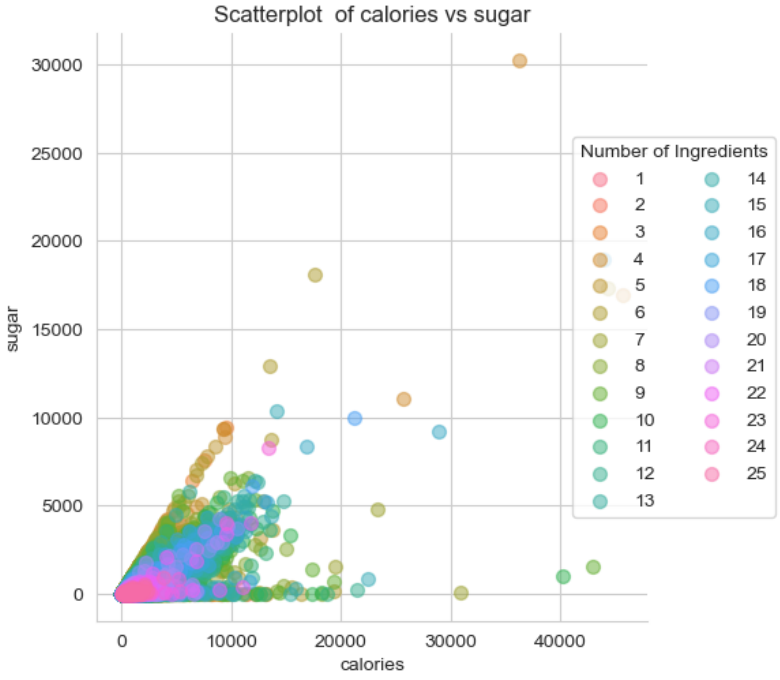
6. Data Exploratory Analysis

Part I. Python

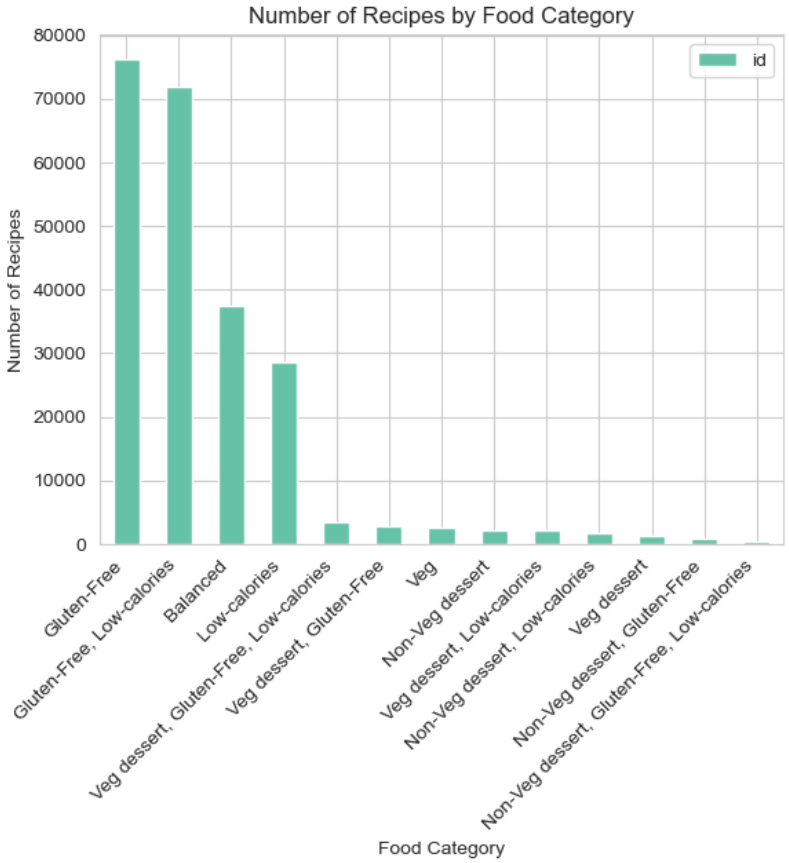
- This chart is useful for identifying patterns and relationships between variables, here to explore the relationship between calorie and fat content in the recipes. Visible linear correlation between ‘total fat’ and ‘calories’ contains by recipes.



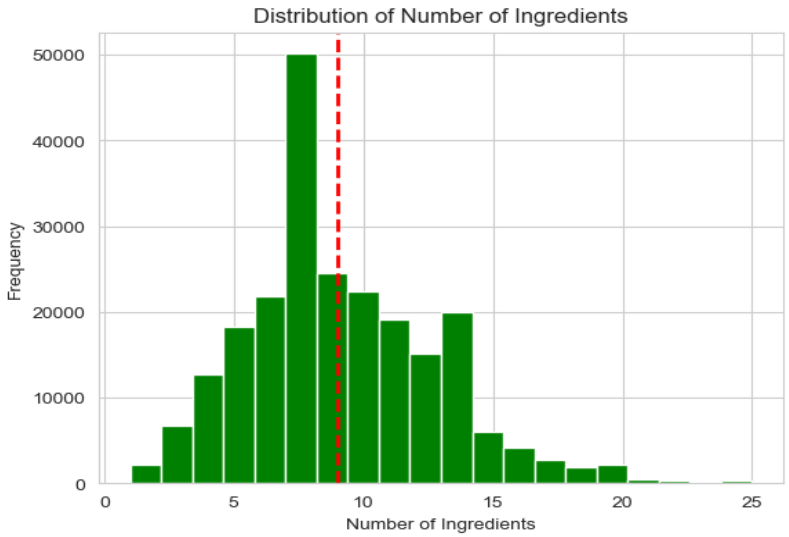
- This chart visualizes the relationship between calories and sugar in the recipes, with the hue which represents the number of ingredients in each recipe. In general, visible dependence of growth in sugar and calories.



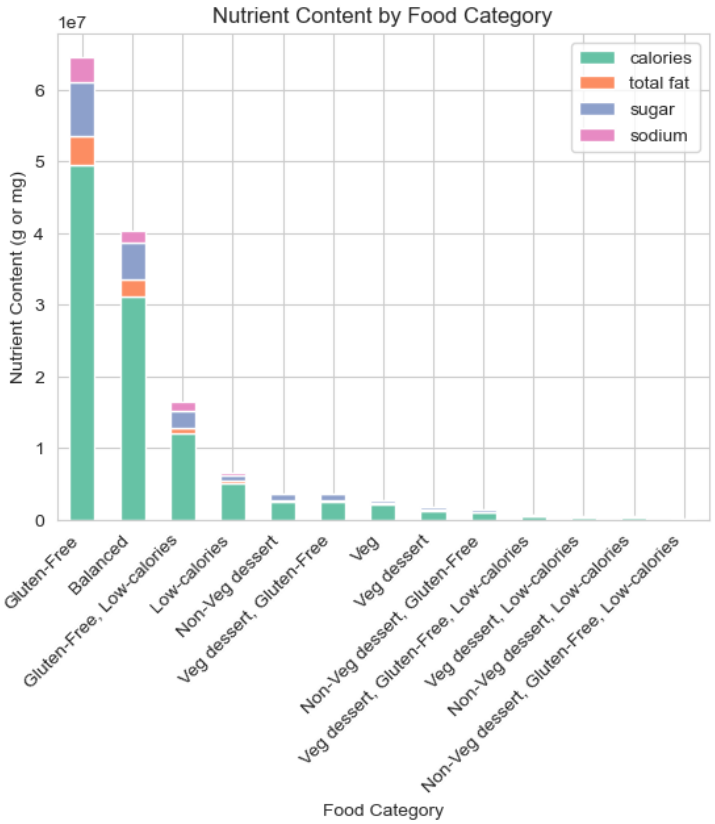
- This chart to show the number of recipes in each food category. We have our top 3: ‘Gluten-free’, ‘Gluten-Free, Low-calories’ and ‘Balanced’ food category.



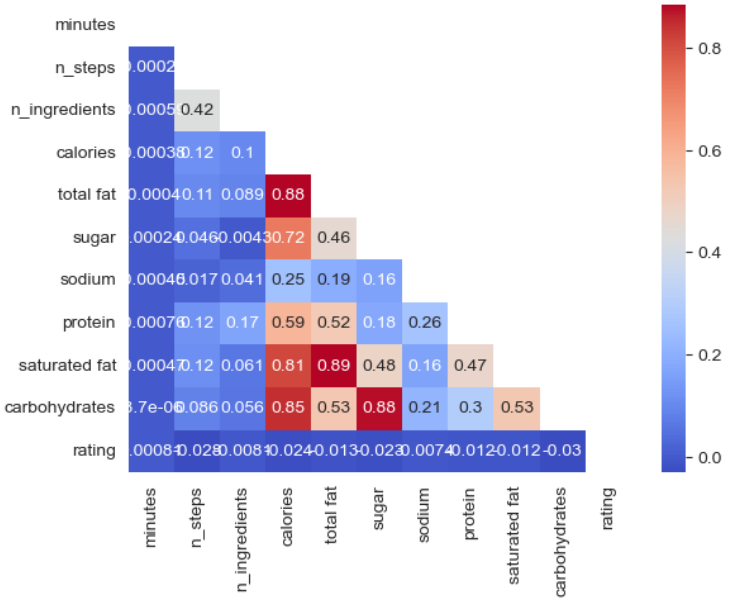
- A chart to show the frequency distribution of the number of ingredients used in the recipes, where red line shows the mean number of ingredients. We can conclude, that the more frequent number of ingredients is 7 ingredients in one recipe, also the red line shows us the mean of 9 for all library of recipes.



- A chart to show the proportion of nutrient content by ‘food category’. So here, we can clearly see the ‘Gluten-Free’ category containing most nutrition, followed by ‘Balanced’ class and ‘Gluten-Free, Low calories’ class.

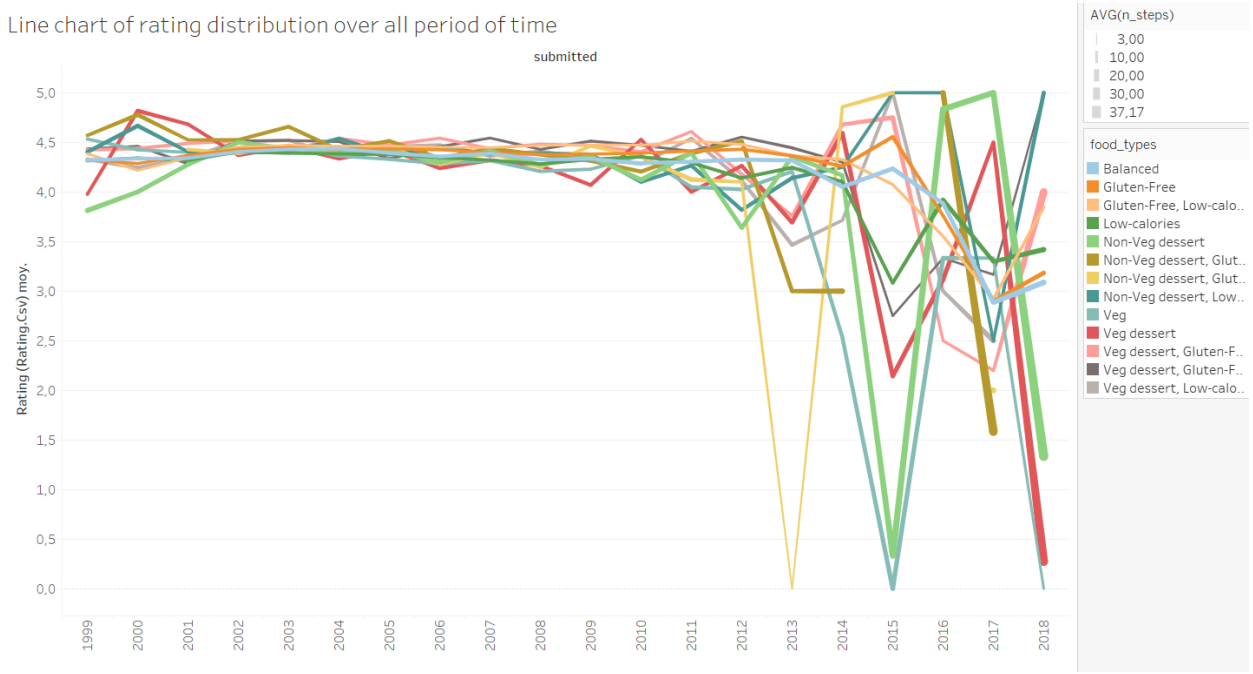


- This heatmap shows the correlation matrix for the different numerical columns. The higher values indicating stronger positive correlations, such as (Total fat and calories) and lower indicating weaker correlations.

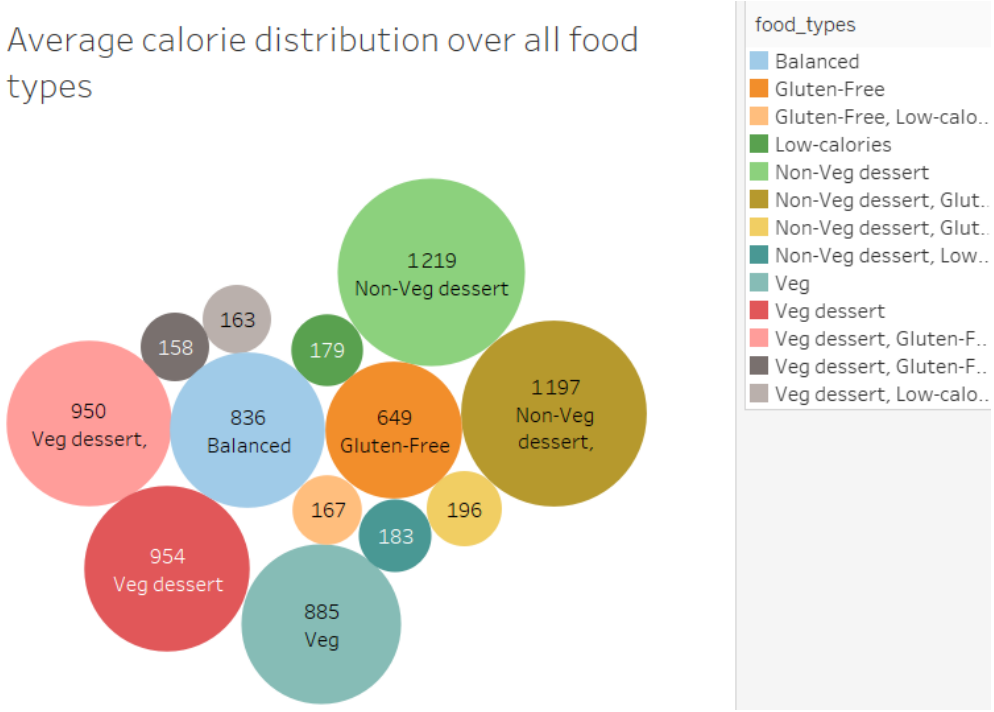


Part II. Tableau

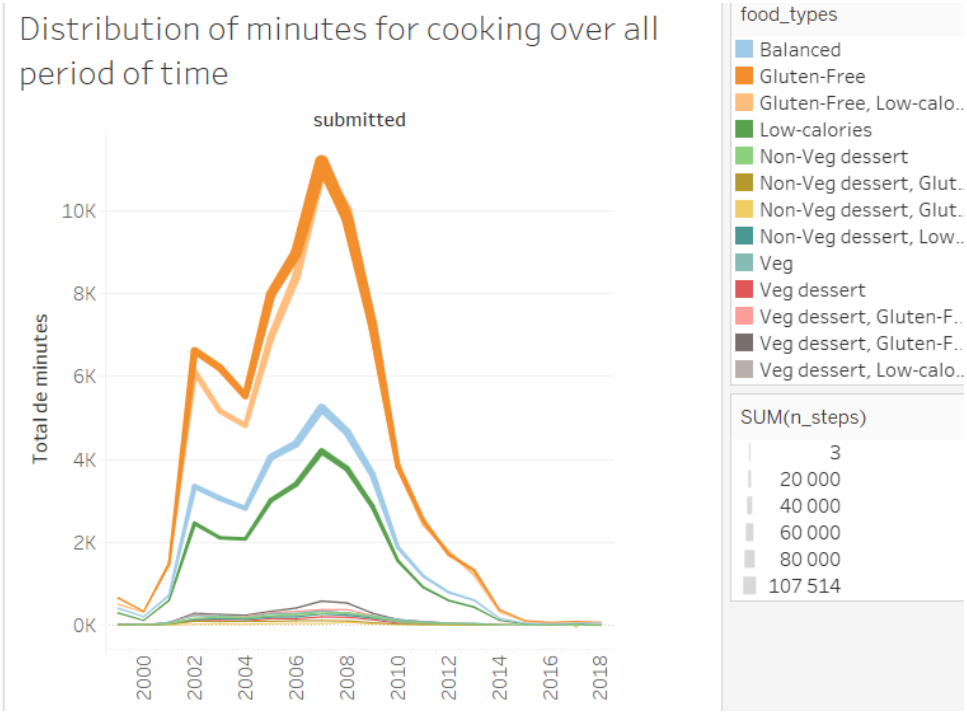
- The line chart, showing the rating distribution for every year from beginning till the last recipe submitted, where size of lines counts average number of steps in recipe and the color explain the food type. That shows us, that when the internet became more accessible, the rating given started to be less predictable, and number of ingredients in recipes became more various.



- Circle diagram of average calories distribution over all possible food types, where color is food type and size are represented by average calorie for each food type



- A line chart, that shows the distribution of minutes over all period of time, where color corresponds to food type and size of line to summary number of steps in each recipe. We can see what minutes consumed to food preparation a lot changes over the time as well as number of steps necessaire to complete the meal.



7. Data base type selection

When it comes to selecting a database type for a project, there are several options available, including relational databases, NoSQL databases, and graph databases. Each type has its unique features and benefits, and the choice ultimately depends on the requirements of the project.

One popular type of database is the relational database, which organizes data into tables and uses SQL for querying and manipulation. Relational databases are suitable for projects that require data consistency and integrity and have a predefined schema. They are widely used in various industries, including finance, healthcare, and e-commerce, among others.

MySQL is one of the most widely used relational database management systems (RDBMS) globally, with over 20 years of development and refinement. MySQL is open-source, free to use, and provides a robust set of features, making it an excellent choice for a wide range of projects.

Here are some reasons why MySQL might be a suitable choice for a project:

Reliability and stability: MySQL have been in development for over two decades, making it a reliable and stable database management system. It has a proven track record of being used in a wide range of projects, from small-scale applications to large, enterprise-level systems.

Scalability: MySQL is highly scalable and can handle a large volume of data and high-traffic applications. It supports both vertical and horizontal scaling and can easily accommodate growing datasets and increased user demand.

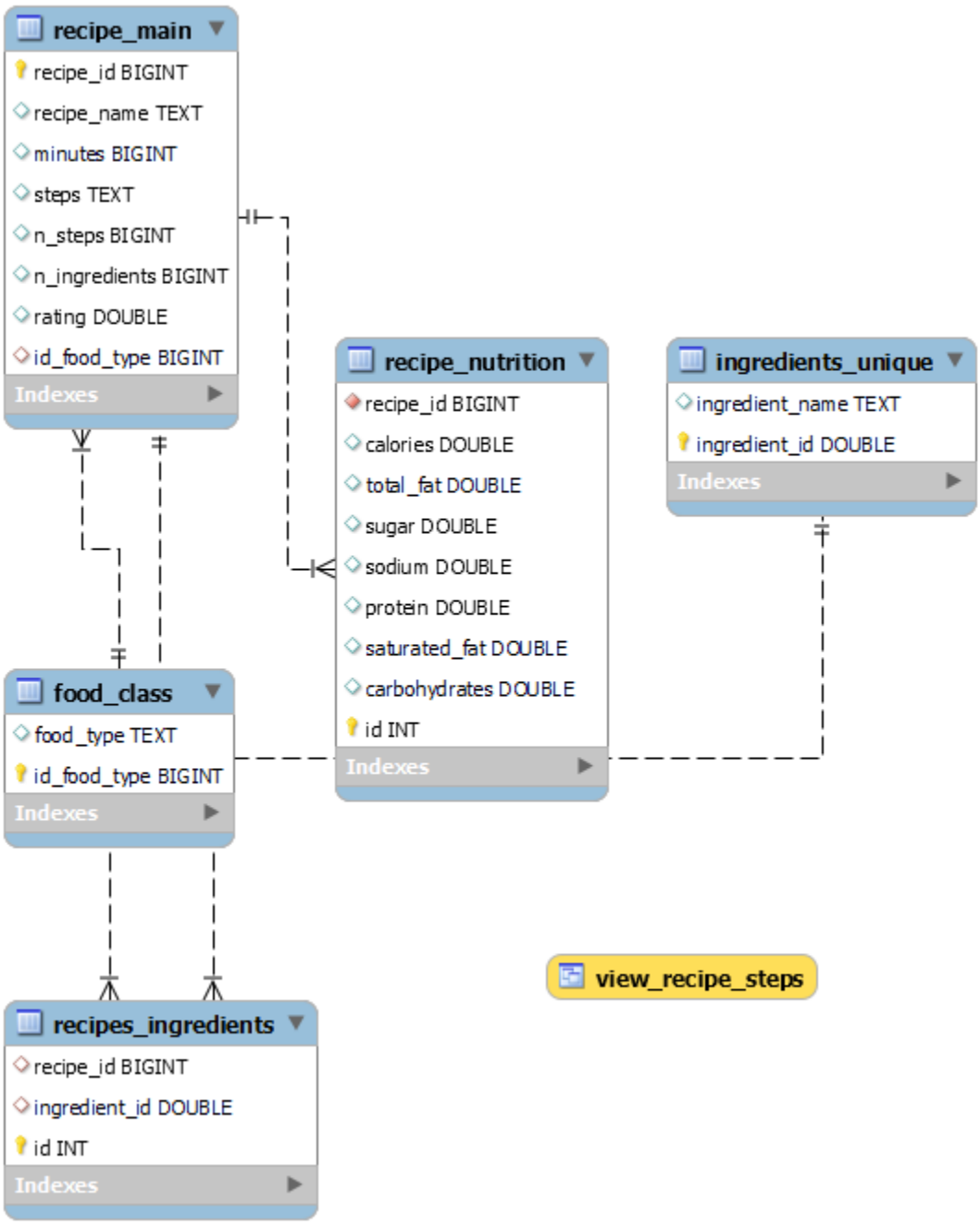
Ease of use: MySQL is easy to install, set up, and configure. It has a straightforward syntax and provides a wide range of tools and resources for database management, including backup and recovery, performance tuning, and security features.

Compatibility: MySQL is compatible with various programming languages, including Python, Java, and PHP, among others. This makes it an ideal choice for projects that require integration with different technologies.

In conclusion, MySQL is a popular and reliable choice for a wide range of projects, thanks to its stability, scalability, ease of use, and compatibility with various programming languages. Its robust set of features, combined with its open-source nature, make it an excellent choice for developers and organizations looking for a reliable, flexible, and cost-effective database management system.

8. Entities. ERD

The entities of the diagram represent my database after the cleaning process. As the final database, I created 5 entities with each have the primary key, most of them have foreign keys to show tables relationship.



9. MySQL Queries

To connect Python file with the MySQL database, I used PyMySQL and SQLalchemy to export data. I ended up with 5 tables to proceed for analysis using SQL.

The MySQL code used to create the database and setting up a connection to work with big tables:

```
############ CREATING DATABASE ##############
CREATE DATABASE IF NOT EXISTS recipes_recomm ;
USE recipes_recomm;

########### SET UP CONNECTION ############
SET GLOBAL connect_timeout=600;
```

Query 1. A procedure to retrieve a list of all recipes that contain a specific ingredient.

```
########### 1.Procedure that retrieves a list of all recipes that contain a specific ingredient
DELIMITER //
CREATE PROCEDURE get_recipes_with_ingredient(IN ingredient_name VARCHAR(50))
BEGIN
    SELECT rm.recipe_name, rm.rating, rm.minutes
    FROM recipe_main rm
    JOIN recipes_ingredients ri ON rm.recipe_id = ri.recipe_id
    JOIN ingredients_unique iu ON ri.ingredient_id = iu.ingredient_id
    WHERE iu.ingredient_name = ingredient_name
    ORDER BY rm.rating DESC;
END //
DELIMITER ;

######### calling procedure ###########
CALL get_recipes_with_ingredient('chicken');
```

Query 1. Result. All possible recipes with its rating in descending order and time of preparation.

recipe_name	rating	minutes
couscous with asparagus chervil white wine	5	25
couscous garbanzo salad	5	15
cousin	5	485
cozy cassoulet	5	180
magic chicken noodle soup	5	40
gujarat chicken	5	35
15 minute chicken tortilla soup	5	12
gumbo potatoes	5	135
guy fieri s lime chicken tequila tailgate sandwiches	5	20
cracker coated oven fried chicken	5	55
beer can chicken in an oven	5	17

Query 2. A procedure to get all ingredients for chosen recipe.

```
########### 2. Get Ingredients by Recipe ##########
DELIMITER //

CREATE PROCEDURE get_ingredients_by_recipe(IN recipe_name VARCHAR(50))
BEGIN
    SELECT iu.ingredient_name, rm.n_steps, rm.n_ingredients
    FROM recipe_main rm
    JOIN recipes_ingredients ri ON rm.recipe_id = ri.recipe_id
    JOIN ingredients_unique iu ON ri.ingredient_id = iu.ingredient_id
    WHERE rm.recipe_name = recipe_name;
END //
DELIMITER ;

######## calling procedure ##########
CALL get_ingredients_by_recipe('cassoulet');
```

Query 2. Result. Table of the ingredients needed for filtered recipe.

ingredient_name	n_steps	n_ingredients
boneless duck breast	15	17
salt pork	15	17
toulouse sausages	15	17
oil	15	17
onions	15	17
garlic cloves	15	17
cannellini beans	15	17
carrots	15	17
diced tomatoes	15	17
tomato paste	15	17
bouquet garni	15	17
dried thyme	15	17
chicken stock	15	17

Query 3. Create a view of the steps of a recipe

```
######## 3. View to see the steps of recipe
CREATE VIEW view_recipe_steps AS
SELECT rm.recipe_name, rm.n_steps, rm.steps, fc.food_type
FROM recipe_main rm
JOIN food_class fc ON rm.id_food_type = fc.id_food_type;

###### view the view table
SELECT * FROM view_recipe_steps WHERE recipe_name = 'Spaghetti Bolognese';
```

Query 3. Result. Filtering by chosen recipe to extract its steps of preparation.

	recipe_name	n_steps	steps	food_type
▶	spaghetti bolognese	14	1. in a large pot 2. heat the oil over medium-hig…	Gluten-Free

Query 4. Most used ingredients in top rated recipes.

```
####### 4. Most used ingredients in top rated recipes
SELECT iu.ingredient_name, COUNT(*) AS frequency
FROM recipe_main rm
JOIN recipes_ingredients ri ON rm.recipe_id = ri.recipe_id
JOIN ingredients_unique iu ON ri.ingredient_id = iu.ingredient_id
WHERE rm.rating = (SELECT MAX(rating) FROM recipe_main)
GROUP BY iu.ingredient_name
ORDER BY frequency DESC limit 20;
```

Query 4. Result. The top 3 used ingredients are Salt, Butter and Sugar, and its frequency.

	ingredient_name	frequency
▶	salt	35208
	butter	23402
	sugar	19192
	onion	15477
	olive oil	14551
	eggs	14191
	water	13853
	garlic cloves	11356
	flour	10225
	milk	10168

Query 5. The average of all nutrition based on food class only for recipes with rating 5.

```
###### 5. The average of all nutritions based on food class only for recipes with rating 5
SELECT fc.food_type, round(AVG(rn.calories), 2) AS avg_calories, round(AVG(rn.protein), 2) AS avg_protein,
                     round(AVG(rn.total_fat),2) AS avg_total_fat, round(AVG(rn.carbohydrates), 2) AS avg_carbohydrates,
                     round(AVG(rn.sugar), 2) AS avg_sugar
FROM (
    SELECT rm.recipe_id, iu.ingredient_id, rm.id_food_type
    FROM recipe_main rm
    JOIN recipes_ingredients ri ON rm.recipe_id = ri.recipe_id
    JOIN ingredients_unique iu ON ri.ingredient_id = iu.ingredient_id
    WHERE rm.rating = 5
) AS t
JOIN food_class fc ON t.id_food_type = fc.id_food_type
JOIN recipe_nutrition rn on t.recipe_id = rn.recipe_id
GROUP BY fc.food_type
ORDER BY avg_calories DESC limit 20;
```

Query 5. Result. Showing all food types in descending order by calories, so the leader is ‘Non-Veg dessert’ food type.

food_type	avg_calories	avg_protein	avg_total_fat	avg_carbohydrates	avg_sugar
Non-Veg dessert	1262.38	34.13	108.19	49.99	399.07
Non-Veg dessert, Gluten-Free	1014.38	28.2	95.3	35.23	285.11
Veg dessert	1006.13	33.6	82.08	40.09	291.34
Veg dessert, Gluten-Free	945.16	30.09	84.4	34.76	294.18
Veg	885.03	45.11	53.74	40.22	122.02
Balanced	844.69	57.79	65.1	28.58	136.32
Gluten-Free	645.96	56.33	53.67	17.52	87.93
Non-Veg dessert, Gluten-Free, Low-calories	206.17	6.51	18.1	7.1	57.25
Non-Veg dessert, Low-calories	189.27	5.12	14.61	7.61	61.54
Low-calories	183.94	12.78	11.37	7.05	31.29

10. Conclusion

In conclusion, this project successfully tackled the problem of recipe recommendation by building a comprehensive system using data analysis techniques. The project started by acquiring data from reliable sources and importing it into Python for pre-processing and cleaning. Feature engineering was then used to improve the quality of the data, and it was exported to MySQL for processing.

In MySQL, a database and entity relationship were established, and five interesting queries were created to extract useful information from the data. Explanatory analysis was then performed in Python to generate insights, and Tableau visualization was used to present the results in an easy-to-understand format.

Finally, a machine learning model, based on natural language processing and unsupervised machine learning model, were built to recommend recipes to users based on their preferences. The accuracy of the model was tested and improved until it reached a satisfactory level of performance. Overall, this project demonstrated how data analysis techniques can be used to provide personalized and accurate recipe recommendations, helping users save time and money while enjoying delicious meals.

11. Links

GitHub link

https://github.com/EljMorgan/ProjectFinal_8

Google drive with larges files

https://drive.google.com/drive/u/0/folders/1TuYJeDAGhHxq_NCDJ3QM8PA84OecCg6x

Trello board

https://trello.com/invite/b/mGWPU2aW/ATTIa86565030eb4a057ed310655b226f186FB547648/finalproject