RECIPE RECOMMENDER

ELNARA KULIJEVA
IRONHACK
MARCH 2023

98%

Table of Content

Objective of the Project

- Investigate the recipes construction
- Find out most approved food classification
- Is more caloric food is the most popular?
- Try to build good recommender
- Tool for users to discover new recipes

Project Planning

- Data collecting and exploring
- Data preprocessing and cleaning
- Creating a database in MySQL
- MySQL queries
- Exploratory analysis in Python
- Tableau visualization
- Building ML model
- Streamlit app

Data Collection

- 2 data sets from Kaggle.com
- Recipe portal food.com
- Scrapped via Python (using Requests and BeautifulSoup)
- 231 637 rows
- 12 columns

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

```
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

Data Collection

- 1 132 367 rows
- 5 columns

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

```
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...

Data Cleaning raw_recipes

- 1 missing value in 'name' column
- 4979 missings values in 'description' column
- No duplicates
- No low variance

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

- Removed characters
- Put all in lowercases
- Added steps numbers

Data Cleaning raw_review

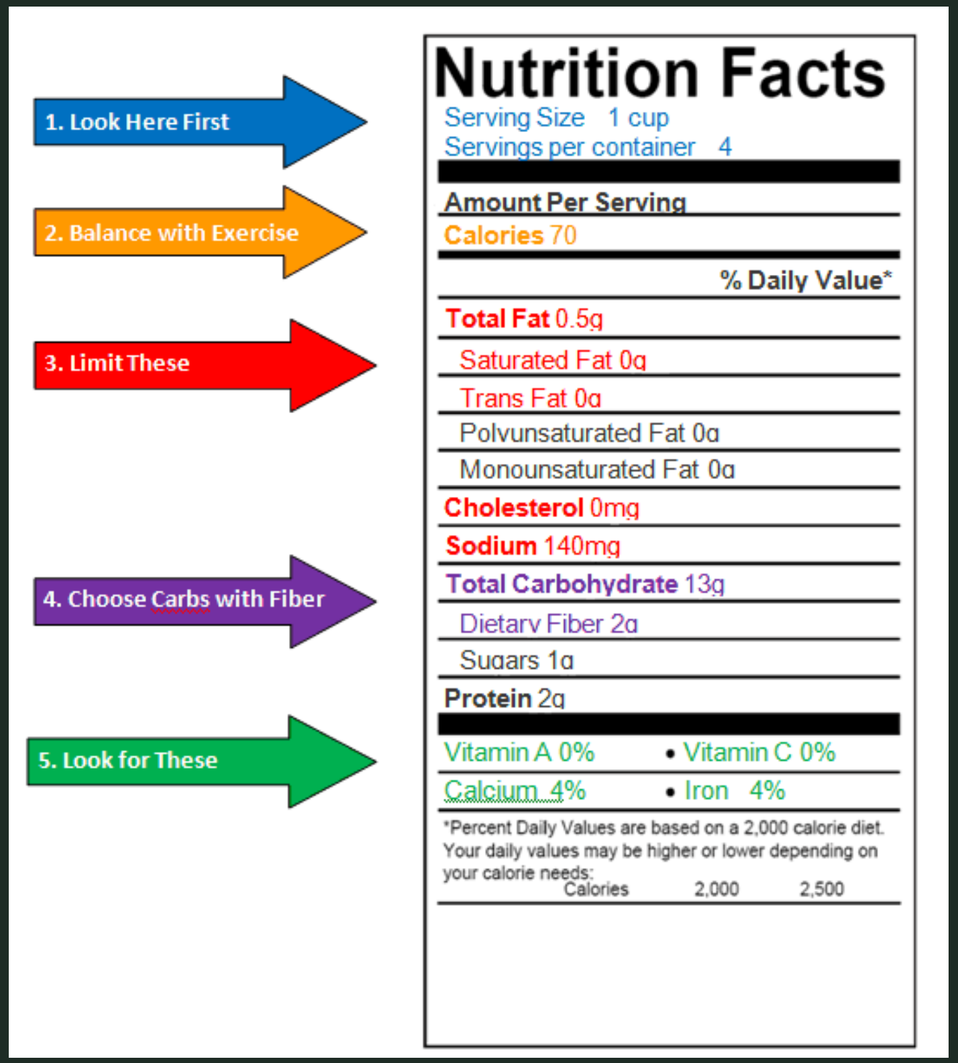
- 169 missing values in 'review' column
- No duplicates
- No low variance

- Finding unique_ingredients
- 29 580 length
- No duplicates
- Removing unwanted characters ['] "
- 155 581 gluten-free recipes
- 76 056 contains gluten

```
recipes = raw_recipes.drop(['contributor_id', 'tags'], axis=1)
rating = raw_interactions.drop(['user_id','review'],  axis=1)
```

Data Preprocessing

- Creating new features in recipes
- 8 new columns
- Each nutrition for each recipe
  - Calories
  - Total fat
  - Sugar
  - Sodium
  - Protein
  - Saturated fat
  - Carbohydrates



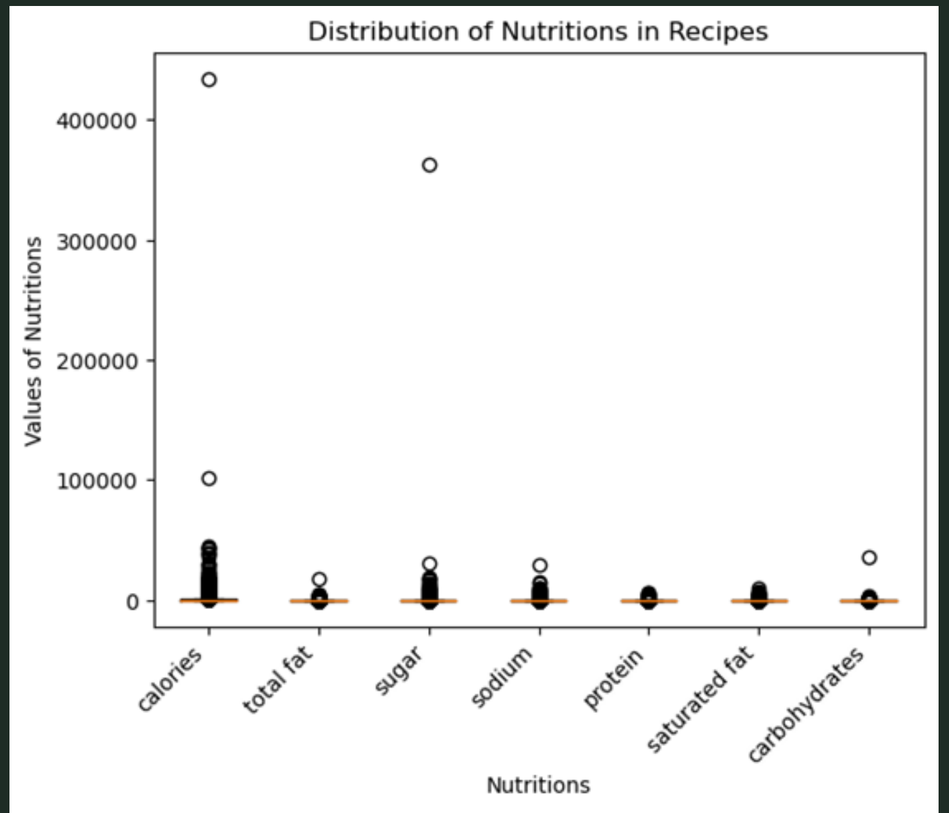
Data Preprocessing

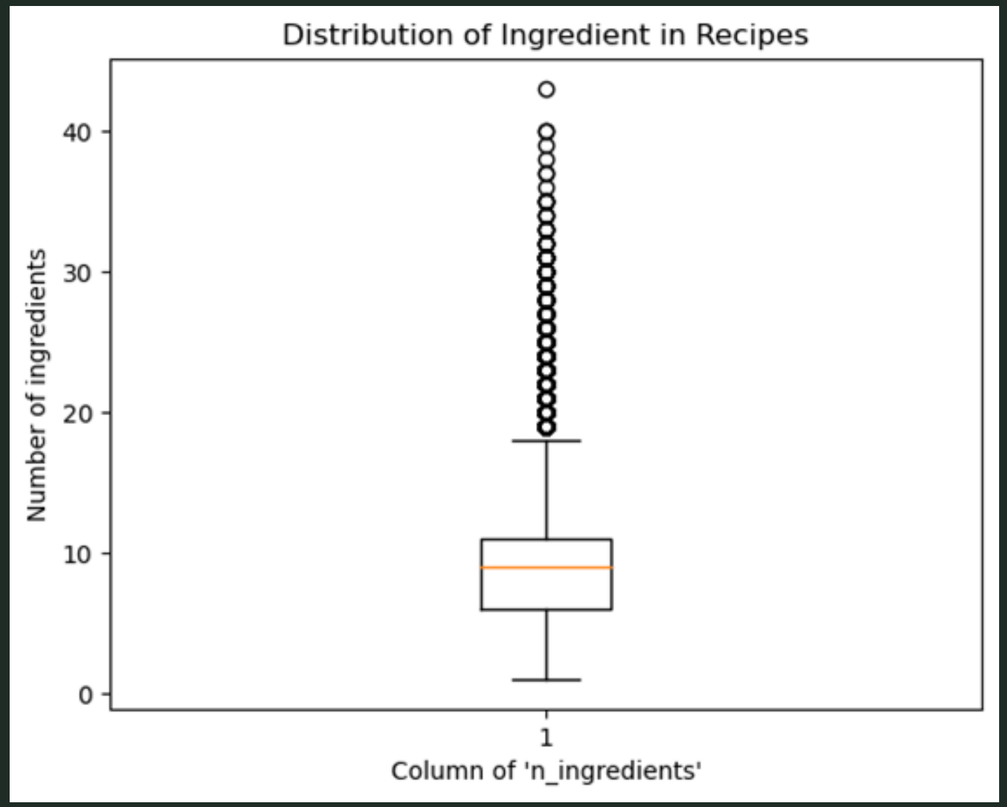
- Creating new features in recipes
- 1 new column
- Food type categorical metric for each recipe
- 13 classifications, for Low-Calories it is less than 300

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

Data Processing - Outliers

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





Data Processing - Feature engeneering

```python
#Creating unique ingredient df
unique_ingredient = pd.DataFrame(unique_ingredients, columns=['Ingredient'])
ingr_df = unique_ingredient.reset_index(drop=True)
ingr_df.index += 1

# Generate unique integer IDs for each row
ingr_df['ID'] = range(1, len(ingr_df) + 1)
# Filter out duplicate ingredients in ingr_df
ingr_df = ingr_df.drop_duplicates(subset='Ingredient')
ingr_df.head()
```

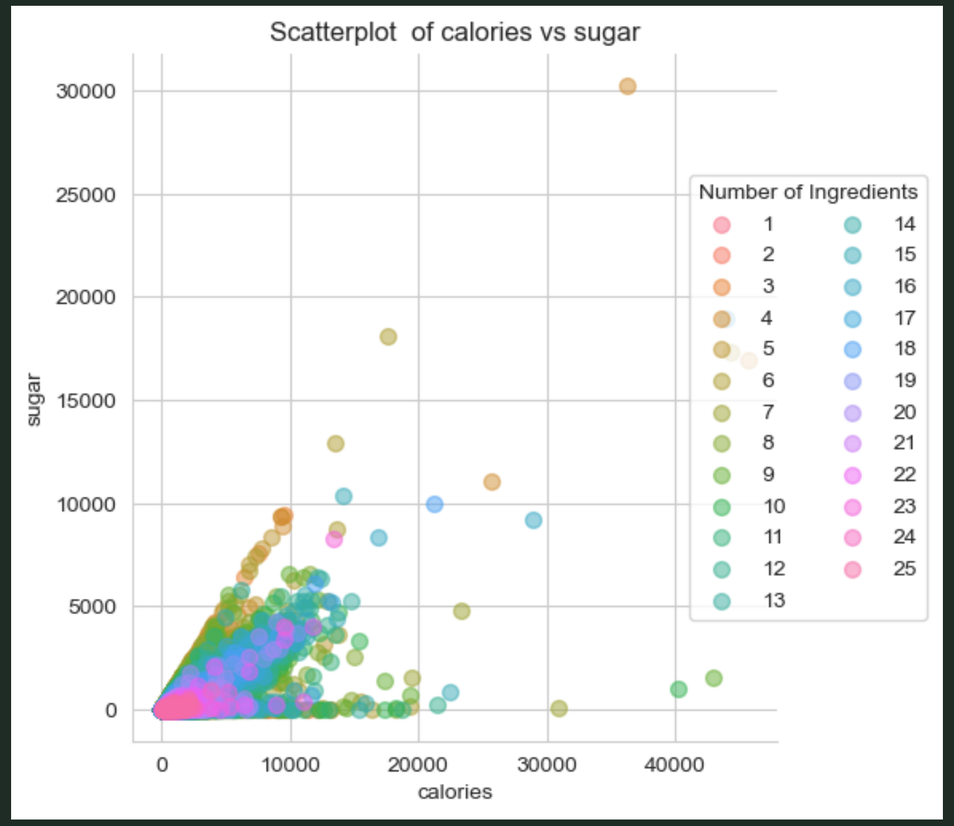
Ingredient	ID
winter squash	1
mexican seasoning	2
mixed spice	3
honey	4
butter	5

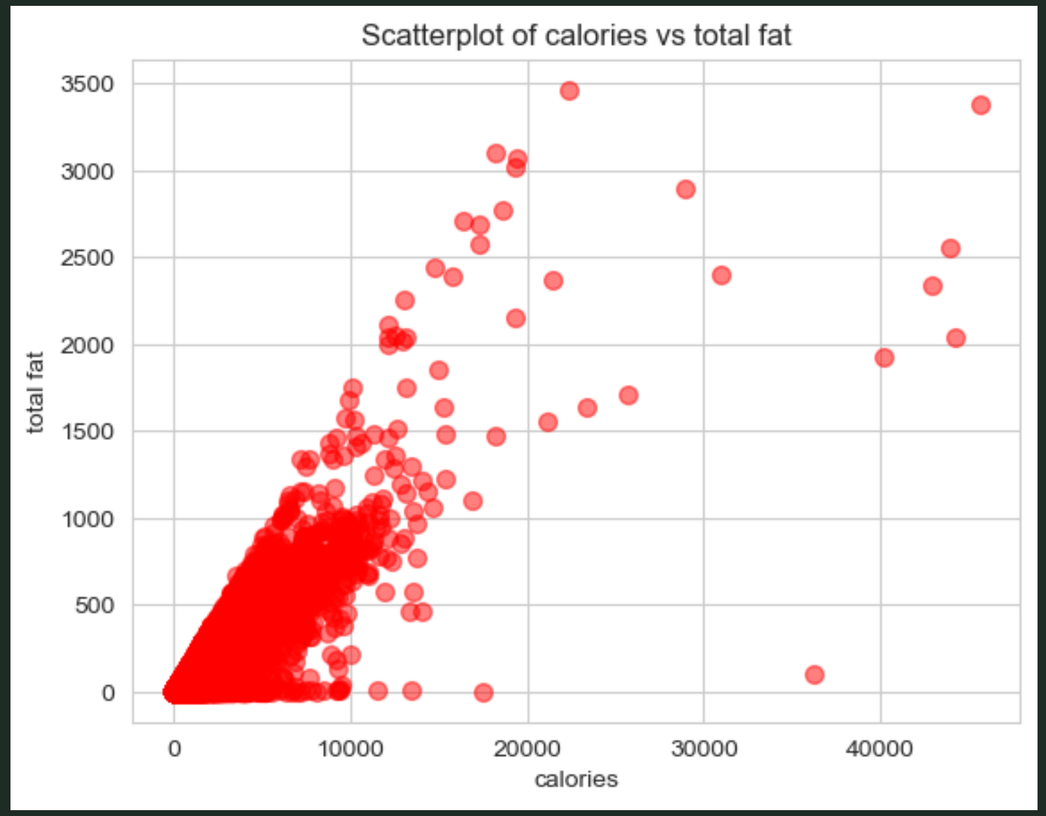
```python
# rename the column using the rename() method
food_type = food_type.rename(columns={0: 'food_type'})
# Generate unique integer IDs for each row
food_type['id_food_type'] = range(1, len(food_type) + 1)
food_type.head()
```

food_type	id_food_type
Gluten-Free, Low-calories	1
Gluten-Free	2
Veg dessert	3
Balanced	4
Low-calories	5

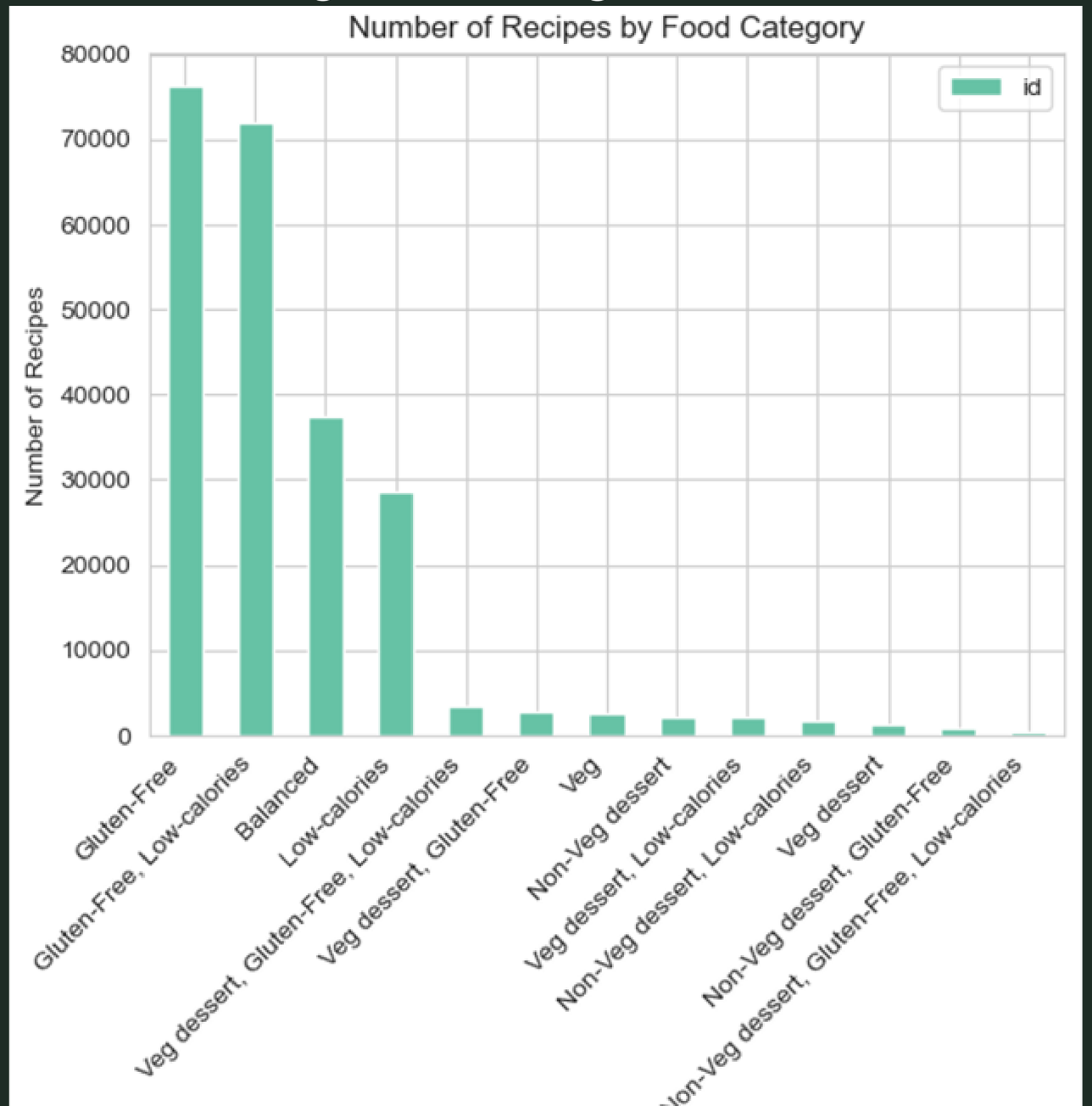
- food_types (shape 13, 2)
- nutrition (shape 231 396, 8)
- recipe_ingredient (2 096 962, 2)
- unique_ingr (14 905, 2) with every unique ingredient with it unique ID
- recipe_main (231 396, 8) with mean rating for every recipe

Data Exploratory Analysis

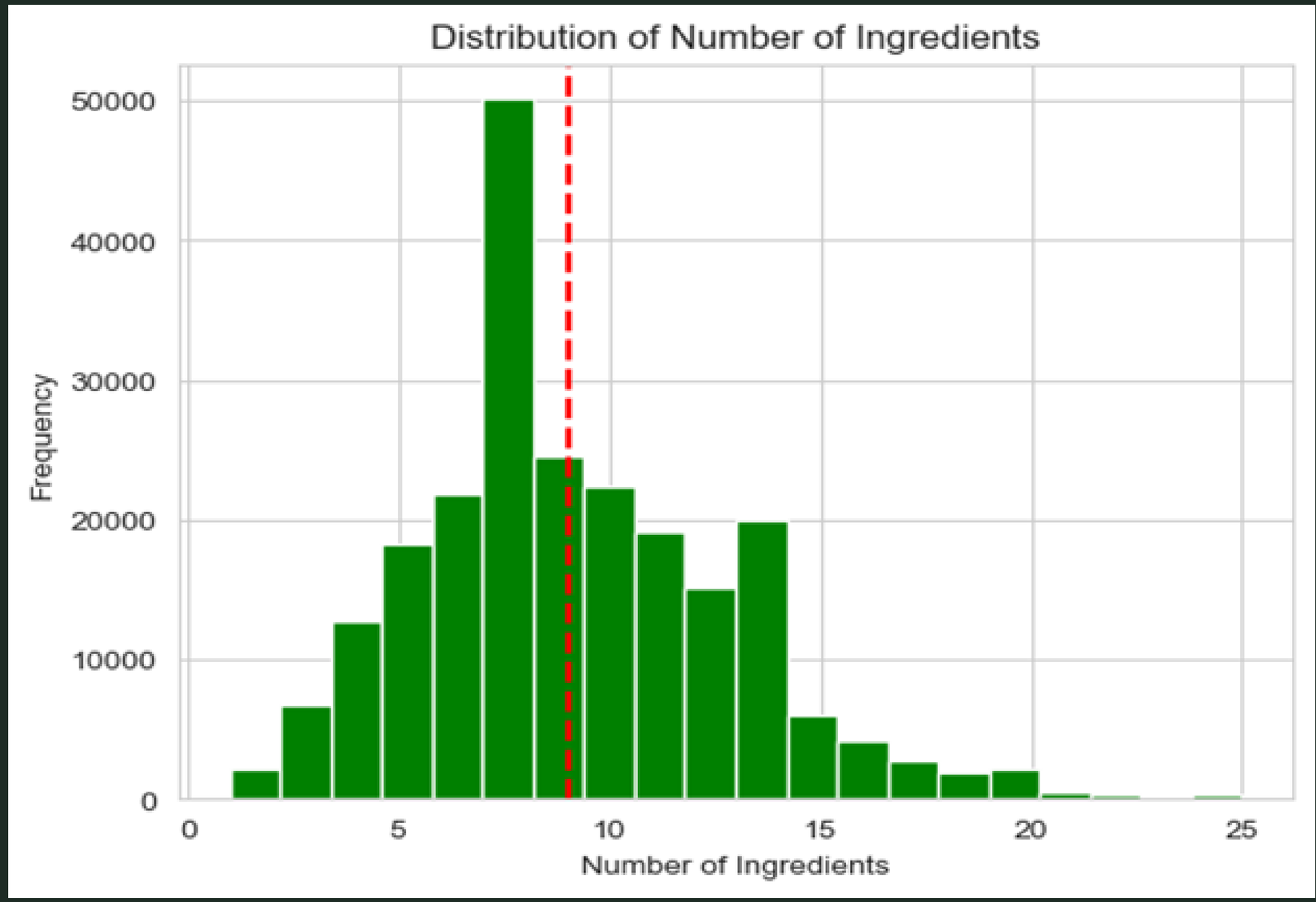




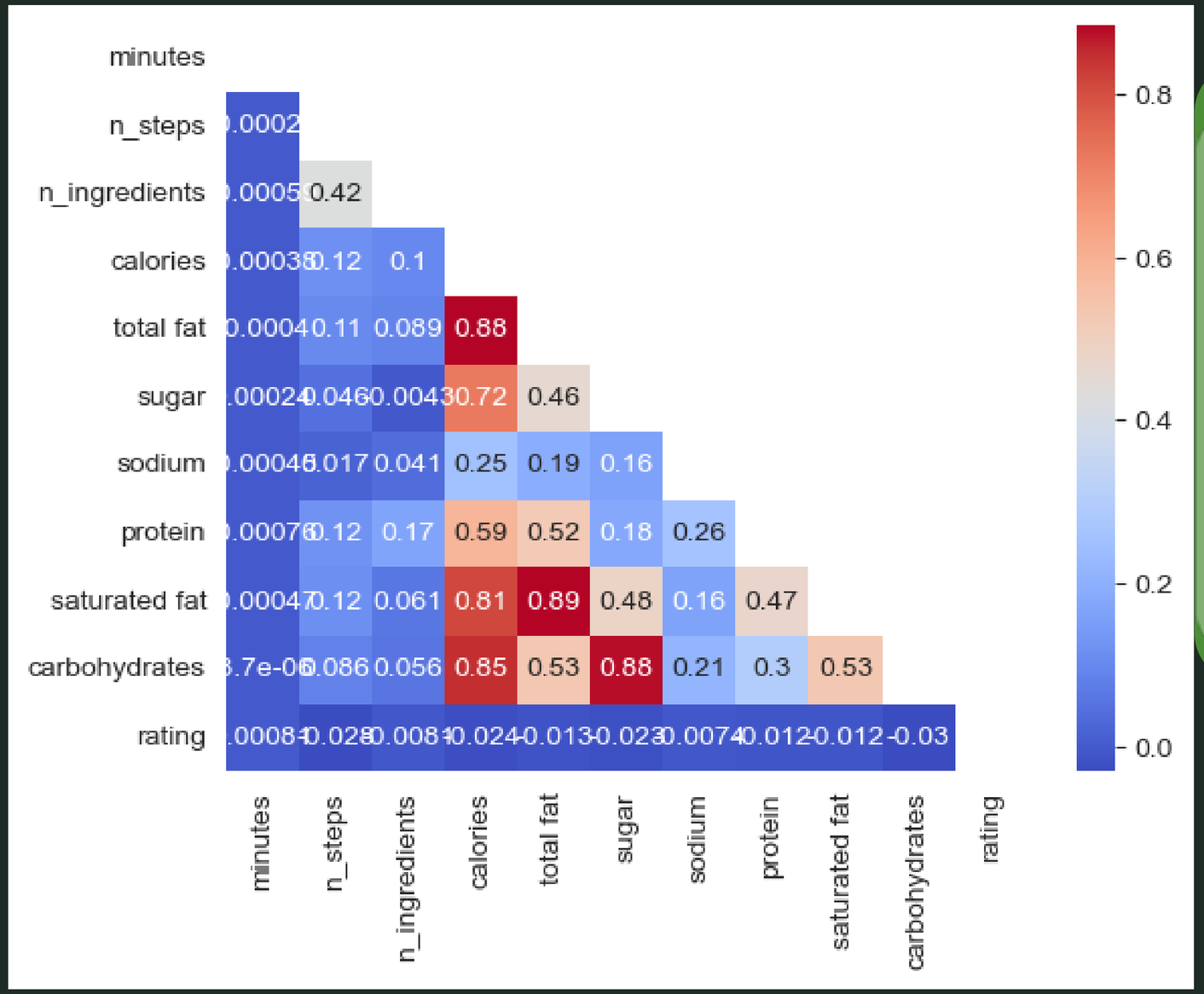
Data Exploratory Analysis

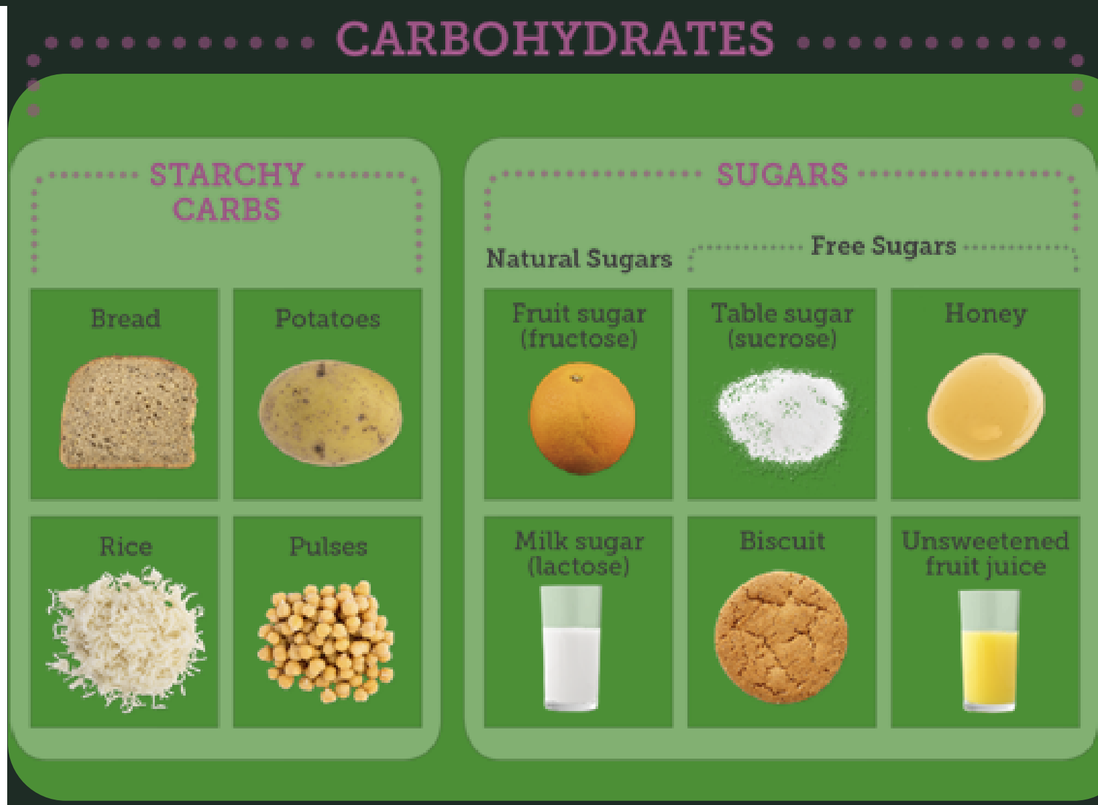


Data Exploratory Analysis

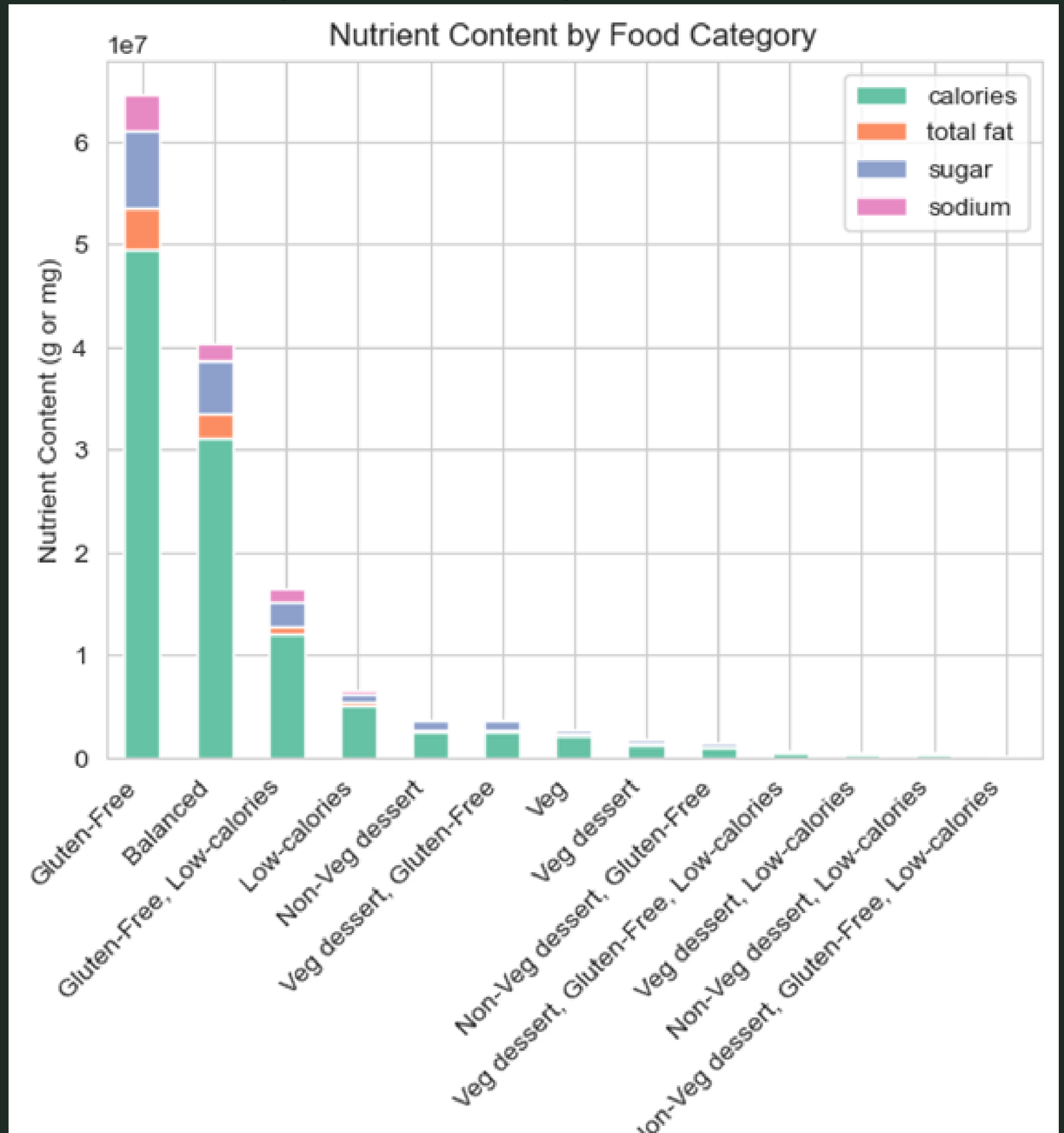


Data Exploratory Analysis

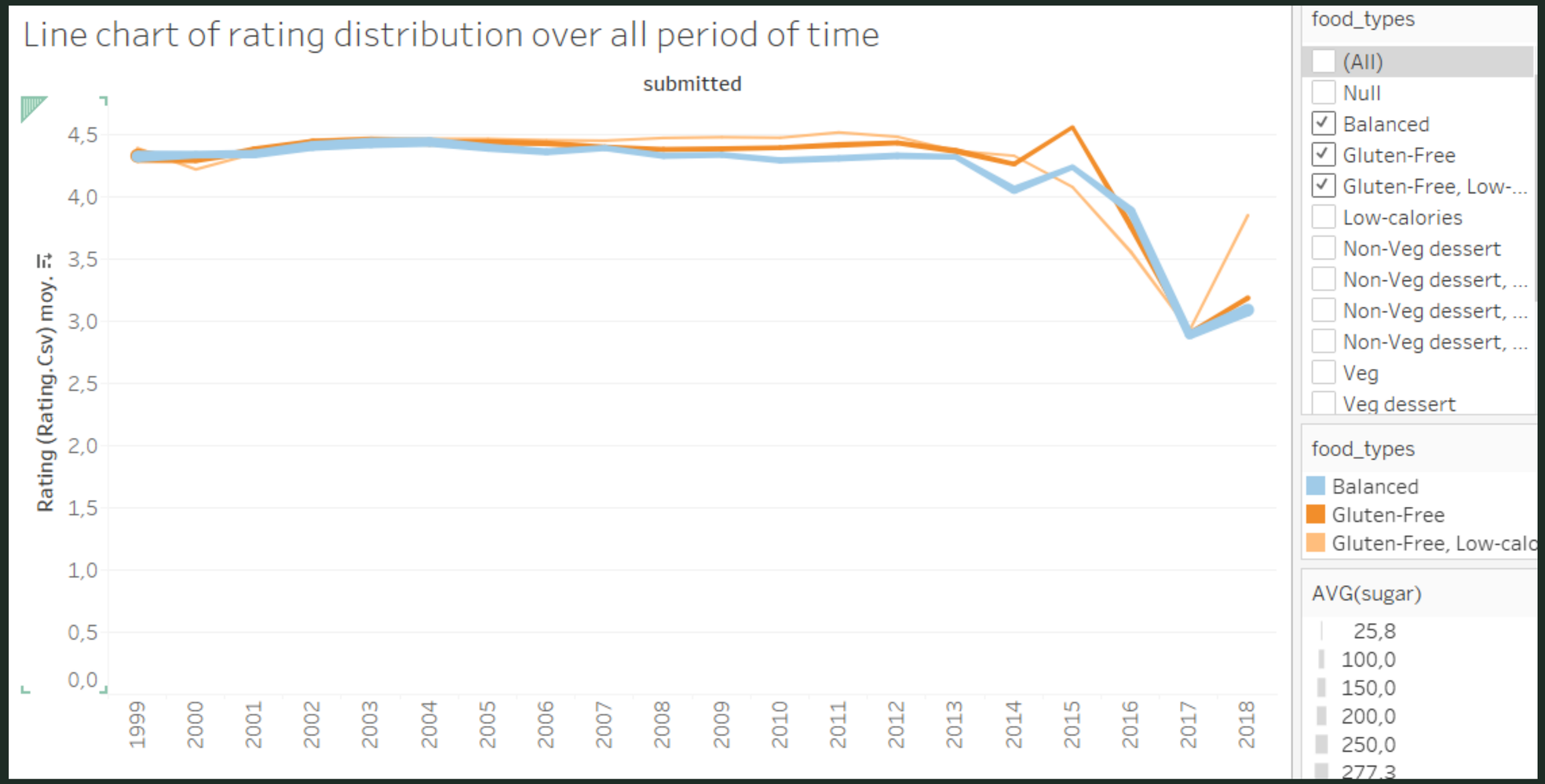




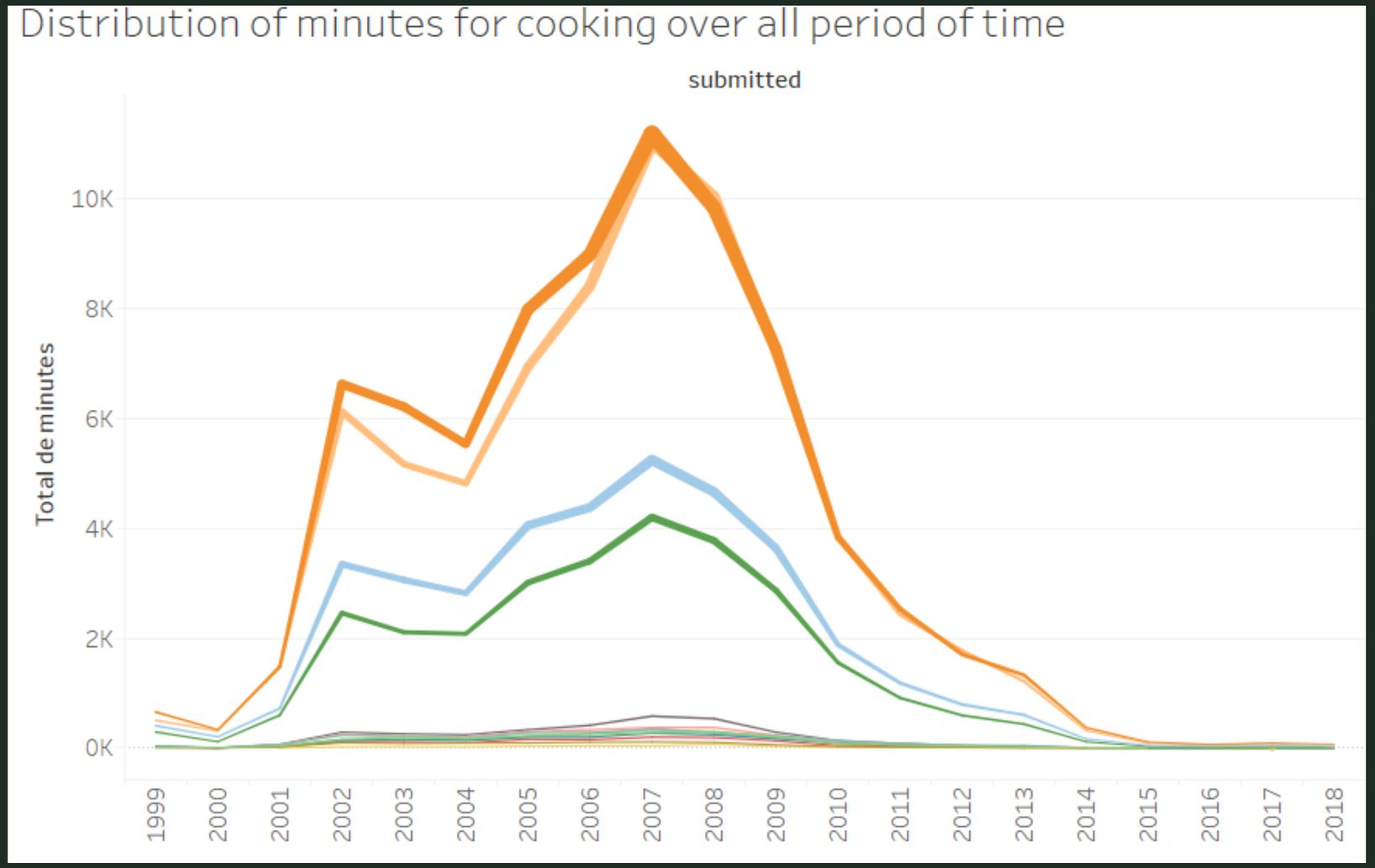
Data Exploratory Analysis



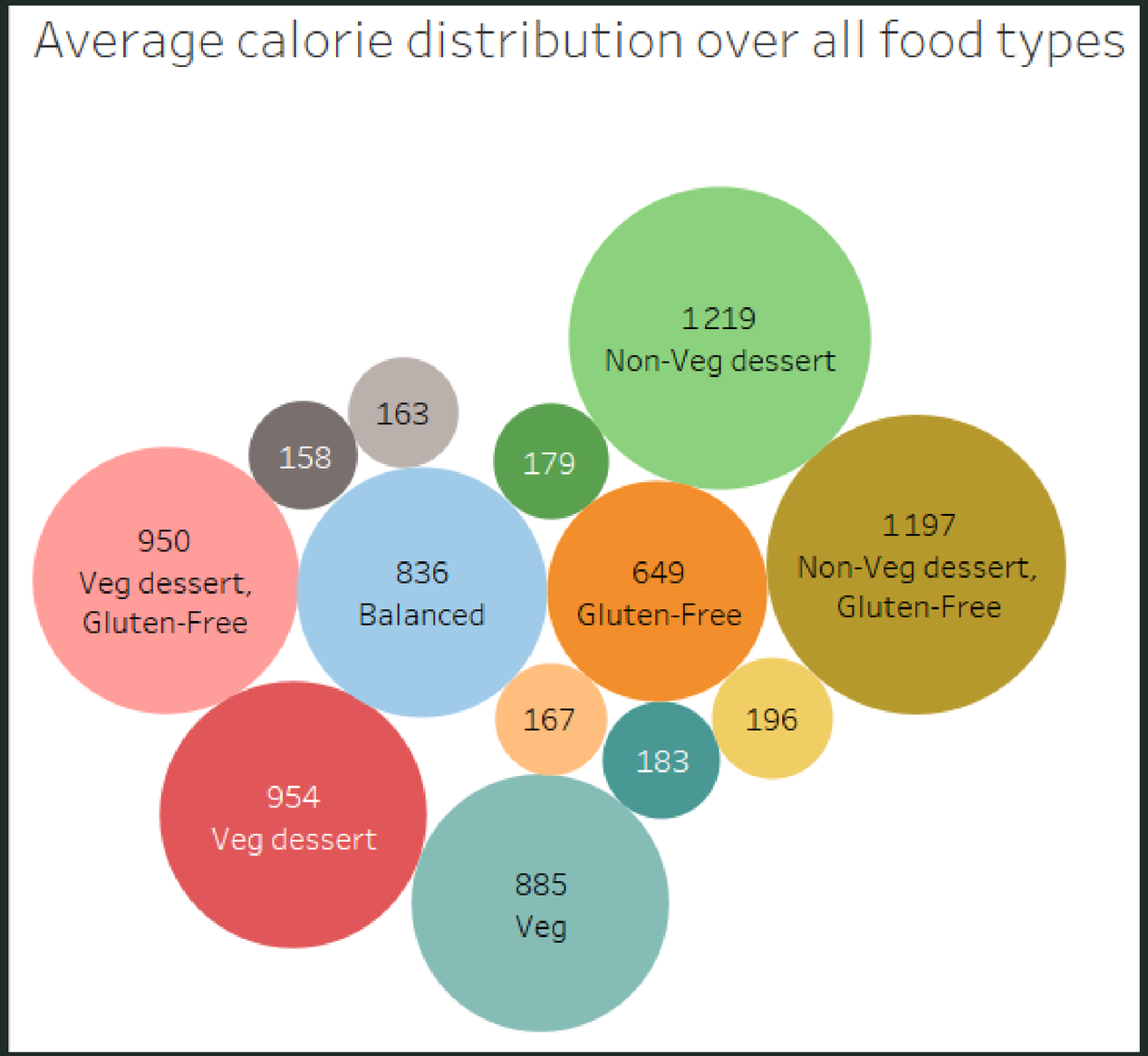
Data Exploratory Analysis



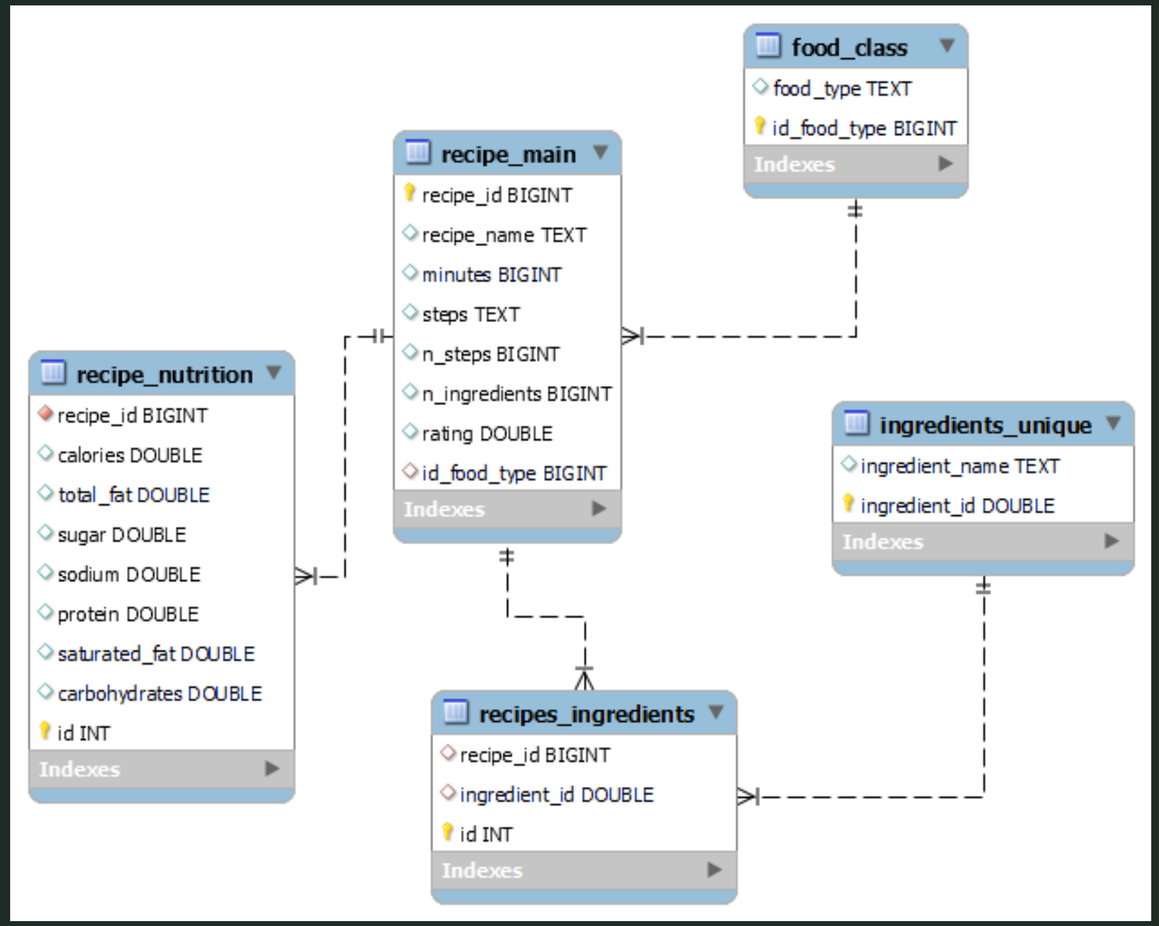
Data Exploratory Analysis



Data Exploratory Analysis



Data Base Creation



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

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

MySQL Queries

```
########### 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');
```

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

MySQL Queries

```
####### 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;
```

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

MySQL Queries

```sql
###### 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;
```

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

Data Modeling

```
1  recipes.isna().sum()
```

name	0
id	0
minutes	0
submitted	0
n_steps	0
steps	0
description	4978
ingredients	0
n_ingredients	0
calories	0
total fat	0
sugar	0
sodium	0
protein	0
saturated fat	0
carbohydrates	0
food_types	0
rating	0


```
salt,;94345
pepper,;85954
sugar,;80676
fresh;75239
oil,;72059
ground;64643
butter,;64077
cheese,;58555
flour,;55847
onion,;55799
garlic;52466
powder,;48634
olive;41719
sauce,;40746
red;39827
water,;39349
chicken;38725
black;37459
milk,;36952
juice,;35271
baking;34591
eggs,;33683
pepper;32468
green;32392
salt;31333
cloves,;29549
dried;29131
white;28083
lemon;27654
brown;27052
```

Data Modeling

```python
from fuzzywuzzy import fuzz, process

def find_similar_recipe_names(name, subset_recipe, n=3):
    # Get all recipe names in the dataframe
    all_names = subset_recipe['name'].values.tolist()

    # Find the most similar names using fuzzy matching
    similar_names = process.extract(name, all_names, scorer=fuzz.token_sort_ratio, limit=n)

    # Extract the names and similarity scores
    similar_names = [(n, s) for n, s in process.extract(name, all_names, scorer=fuzz.token_sort_ratio, limit=n)]

    return similar_names

```

✓ 0.0s Python

```python
findy = find_similar_recipe_names('broccoli', subset_recipe)
findy
```

✓ 0.6s Python

```
[('broccoli pie', 80), ('broccoli pork', 76), ('broccoli slaw', 76)]
```

Data Modeling

- 3 columns ['name]+['ingredients']+ + ['description']
- Removing stop words
- TfidfVectorizer
- cosine_similarity
- Final dataset 15 000

```
# Select two recipes to compare
recipe1 = subset_recipe.iloc[9473]
recipe2 = subset_recipe.iloc[8048]

# Compute the cosine similarity between the two recipes
cosine_sim_12 = cosine_sim[9473, 8048]

print(f"The cosine similarity between '{recipe1['name']}' and '{recipe2['name']}' is {cosine_sim_12:.2f}")

```

✓ 0.0s Pytho

```
The cosine similarity between 'broccoli pork' and 'pork tenderloin with cumberland sauce' is 0.43
```

Data Modeling

```
i = 0
print(f'Top 5 similar recipes to "try" are:\n')
if sorted_similar:
    for element in sorted_similar:
        print(get_name_from_index(element[0]))
        i += 1
        if i > 4:
            break
else:
    print('No similar recipes found.')

✓ 0.0s
```

'broccoli pie'

```
Top 5 similar recipes to "try" are:

cream of broccoli cheese soup
broccoli cheese soup
grandma s egg custard pie
broccoli parmesan
nestle toll house pie
```

Data Modeling

```
1  top_similar = get_similar(subset_recipe, cosine_sim)
2  top_similar
```

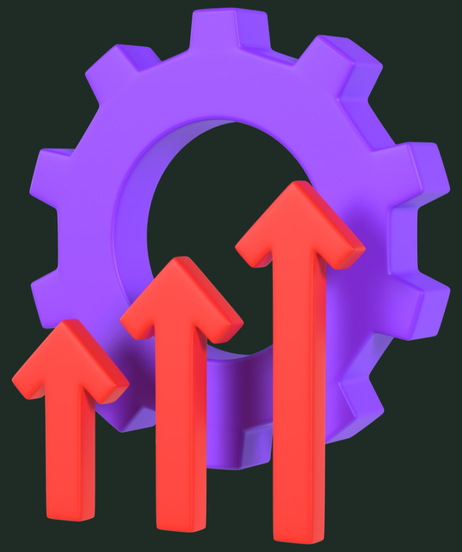
✓ 7.6s

```
Top 5 similar recipes to 'chicken thai curry' are:

1. green coconut curry with vegetables
2. curry chicken  fast  easy
3. thai vegetable curry
4. gaeng keow wan gai  thai green curry chicken
5. thai green chicken curry  diabetic friendly  sugarless

curry chicken  fast  easy

Ingredients: cooking oil, garlic, curry powder, onions, whole chicken, water, salt, potatoes

Steps: 1. in a large pot heat up cooking oil add chopped garlic and onion lightly fry 2. add curry powder a
```

'chicken thai curry'

Challenges and Highlights

- RAM of the notebook
- Code debugging time
- Expectations were higher than a result
- Planning of solo project
- Use of acquired knowledge
- Implementing own vision of problem



Improvement plan

- Try anothers models
- Be able to search by ingredients
- Make more interactive page of Streamlit
- Try to run from a cloud to be able to manipulate with large datasets
- Create option for time preparation choosing
- Add photo of recipe and make search by photo

Streamlit demo

Conclusion

Power of data analytics
Even simple model can be usefull
Working with NLP is very specific
Friendly app
Practical project for users

THANK YOU