Python [conda env:DataScience]

From understanding to preparing data.

Dataset Description:

- The data was compiled by a researcher named Yong-Yeol Ahn, who extracted tens of thousands of food recipes (cuisines and ingredients) from three different websites, namely:
  - www.allrecipes.com
  - www.epicurious.com
  - www.menupan.com
- All the collected data was grouped into: ***recipes.csv***
- Our dataset consists of various recipes and their respective ingredients.
- Each row represents a recipe, and for each recipe, the corresponding geographic area and whether or not the ingredients exist in the recipe are documented, starting with **almonds** and ending with **zucchini**.
- We want to determine which ingredients are most commonly used in each geographic area.
- For this reason, the name of each recipe was removed, leaving only the geographic area and ingredients.

Required packages and libraries:

In [1]:
```python
import pandas as pd
import numpy as np
import re

# Disable warnings
import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', None)

print('All packages imported!')
```

```
All packages imported!
```

In [2]:
```python
recipes = pd.read_csv("recipes.csv")
print("Data readed to 'recipes' DataFrame!")
```

```
Data readed to 'recipes' DataFrame!
```

Check the first rows.

In [3]:
```python
recipes.head()
```

Out[3]:

	country	almond	angelica	anise	anise_seed	apple	apple_brandy	apricot	ar
0	Vietnamese	No	No	No	No	No	No	No	
1	Vietnamese	No	No	No	No	No	No	No	
2	Vietnamese	No	No	No	No	No	No	No	
3	Vietnamese	No	No	No	No	No	No	No	
4	Vietnamese	No	No	No	No	No	No	No	

Get the dimensions of the DataFrame.

In [4]:
```
recipes.shape
```

Out[4]:
```
(57691, 384)
```

So our dataset consists of 57,691 recipes. Each row represents a recipe, and for each recipe, the corresponding cuisine is documented and whether or not there are 384 ingredients in the recipe, starting with almonds and ending with zucchini.

Data Preparation:

We will prepare the data for modeling, this stage involves exploring the data further and making sure it is in the right format for the machine learning algorithm we will select in the analytical approach stage.

- We will check the Data to see if it needs to be cleaned.

In [5]:
```
totals = recipes["country"].value_counts()

for item, value in totals.items():
    print(item, value)
```

```
American 40150
Mexico 1754
Italian 1715
Italy 1461
Asian 1176
French 996
east_asian 951
Canada 774
korean 767
Mexican 622
western 450
Southern_SoulFood 346
India 324
Jewish 320
Spanish_Portuguese 291
Mediterranean 289
UK-and-Ireland 282
Indian 274
France 268
MiddleEastern 248
Central_SouthAmerican 241
Germany 237
Eastern-Europe 235
Chinese 226
Greek 225
English_Scottish 204
Caribbean 183
Thai 164
Scandinavia 158
EasternEuropean_Russian 146
Cajun_Creole 146
Moroccan 137
Japanese 136
China 130
Thailand 125
African 115
Southwestern 108
South-America 103
japanese 99
Scandinavian 92
chinese 86
Irish 86
Japan 85
Spain 75
italian 74
Vietnamese 65
North-African 60
German 52
Portugal 50
Philippines 43
Korea 32
Netherlands 32
Lebanon 31
Vietnam 30
Austria 21
Iran 21
Switzerland 20
Pakistan 19
Malaysia 18
asian 17
```

```
Turkey 16
South-African 16
mexico 14
West-African 13
Indonesia 12
Belgium 11
East-African 11
Israel 9
Bangladesh 4
```

Looking at the data, we can draw the following conclusions:

- The cuisine column is labeled as country, which is inaccurate.
- The names of the cuisines are not consistent as they do not all start with a capitalized first letter.
- Some cuisines are duplicated as a variation of the country name, such as Vietnam and Vietnamese.
- Some cuisines have very few recipes.

Let's fix these issues.

- We corrected the name of the country column to 'cuisine'.

In [6]:
```
columns_names = recipes.columns.values
columns_names[0] = "cooks"
recipes.columns = columns_names
```

- We changed all the names in the 'cooks' column to start with capital letters.

In [7]:
```
recipes["cooks"] = recipes["cooks"].str.capitalize()

cooks = recipes['cooks'].unique()

for cook in cooks:
    print(cook)
```

```
Vietnamese
Indian
Spanish_portuguese
Jewish
French
Central_southamerican
Cajun_creole
Thai
Scandinavian
Greek
American
African
Middleeastern
Easterneuropean_russian
Italian
Irish
Mexican
Chinese
German
Mediterranean
Japanese
Moroccan
Southern_soulfood
English_scottish
Asian
Southwestern
Mexico
East_asian
Western
Korean
Canada
Turkey
Caribbean
Bangladesh
India
France
Italy
Israel
Korea
Iran
Eastern-europe
South-african
Uk-and-ireland
China
Belgium
Germany
South-america
Spain
Netherlands
Scandinavia
Philippines
Indonesia
East-african
Vietnam
Thailand
Switzerland
West-african
North-african
Pakistan
Portugal
```

```
Lebanon
Malaysia
Austria
Japan
```

Make the cuisine names consistent.

In [8]:
```
recipes.loc[recipes["cooks"] == "American", "cooks"] = "North-america"
recipes.loc[recipes["cooks"] == "Austria", "cooks"] = "Austrian"
recipes.loc[recipes["cooks"] == "Belgium", "cooks"] = "Belgian"
recipes.loc[recipes["cooks"] == "China", "cooks"] = "Chinese"
recipes.loc[recipes["cooks"] == "Canada", "cooks"] = "North-america"
recipes.loc[recipes["cooks"] == "Netherlands", "cooks"] = "Dutch"
recipes.loc[recipes["cooks"] == "France", "cooks"] = "French"
recipes.loc[recipes["cooks"] == "Germany", "cooks"] = "German"
recipes.loc[recipes["cooks"] == "India", "cooks"] = "Indian"
recipes.loc[recipes["cooks"] == "Indonesia", "cooks"] = "Indonesian"
recipes.loc[recipes["cooks"] == "Iran", "cooks"] = "Iranian"
recipes.loc[recipes["cooks"] == "Italy", "cooks"] = "Italian"
recipes.loc[recipes["cooks"] == "Japan", "cooks"] = "Japanese"
recipes.loc[recipes["cooks"] == "Israel", "cooks"] = "Israeli"
recipes.loc[recipes["cooks"] == "Korea", "cooks"] = "Korean"
recipes.loc[recipes["cooks"] == "Lebanon", "cooks"] = "Lebanese"
recipes.loc[recipes["cooks"] == "Malaysia", "cooks"] = "Malaysian"
recipes.loc[recipes["cooks"] == "Mexico", "cooks"] = "Mexican"
recipes.loc[recipes["cooks"] == "Pakistan", "cooks"] = "Pakistani"
recipes.loc[recipes["cooks"] == "Philippines", "cooks"] = "Philippine"
recipes.loc[recipes["cooks"] == "Scandinavia", "cooks"] = "Scandinavian"
recipes.loc[recipes["cooks"] == "Spain", "cooks"] = "Spanish_portuguese"
recipes.loc[recipes["cooks"] == "Portugal", "cooks"] = "Spanish_portugues
recipes.loc[recipes["cooks"] == "Switzerland", "cooks"] = "Swiss"
recipes.loc[recipes["cooks"] == "Thailand", "cooks"] = "Thai"
recipes.loc[recipes["cooks"] == "Turkey", "cooks"] = "Turkish"
recipes.loc[recipes["cooks"] == "Vietnam", "cooks"] = "Vietnamese"
recipes.loc[recipes["cooks"] == "Uk-and-ireland", "cooks"] = "Uk-and-iris
recipes.loc[recipes["cooks"] == "Irish", "cooks"] = "Uk-and-irish"

recipes
```

Out[8]:

	cooks	almond	angelica	anise	anise_seed	apple	apple_brandy	apricot
0	Vietnamese	No	No	No	No	No	No	No
1	Vietnamese	No	No	No	No	No	No	No
2	Vietnamese	No	No	No	No	No	No	No
3	Vietnamese	No	No	No	No	No	No	No
4	Vietnamese	No	No	No	No	No	No	No
...	...	...	...	...	...	...	...	...
57686	Japanese	No	No	No	No	No	No	No
57687	Japanese	No	No	No	No	No	No	No
57688	Japanese	No	No	No	No	No	No	No
57689	Japanese	No	No	No	No	No	No	No
57690	Japanese	No	No	No	No	No	No	No

57691 rows × 384 columns

- We eliminated cooks with <50 recipes.

In [9]:
```
# Get a list of recipes to keep.
recipes_count = recipes["cooks"].value_counts()
cooks_index = recipes_count > 50

cooks_to_keep = list(np.array(recipes_count.index.values)[np.array(cooks_
```

In [10]:
```
rows_before = recipes.shape[0] # Original number of rows in the DataFrame
print(f"Original DataFrame rows number: {rows_before}.")

recipes = recipes.loc[recipes['cooks'].isin(cooks_to_keep)] # We filter o

rows_after = recipes.shape[0] # Number of rows in the processed DataFrame
print(f"Processed DataFrame rows number: {rows_after}.")

print(f"{rows_before - rows_after} rows removed!")
```

```
Original DataFrame rows number: 57691.
Processed DataFrame rows number: 57394.
297 rows removed!
```

- Encode all Yes in 1 and No in 0

In [11]:
```
recipes = recipes.replace(to_replace = "Yes", value = 1)
recipes = recipes.replace(to_replace = "No", value = 0)
```

In [12]:
```
recipes.head()
```

Out[12]:

	cooks	almond	angelica	anise	anise_seed	apple	apple_brandy	apricot	ari
0	Vietnamese	0	0	0	0	0	0	0	
1	Vietnamese	0	0	0	0	0	0	0	
2	Vietnamese	0	0	0	0	0	0	0	
3	Vietnamese	0	0	0	0	0	0	0	
4	Vietnamese	0	0	0	0	0	0	0	

Let's analyze the data a little more to get to know it better and note any interesting preliminary observations.

- We'll check out recipes containing **rice**, **soy_sauce**, **wasabi**, **seaweed**, and **carrot** and analyze the results.

In [13]:

```
recipes_check = recipes.loc[(recipes["rice"] == 1) & (recipes["soy_sauce"
                                (recipes["wasabi"] == 1) & (recipes["seaw

recipes_check
```

Out[13]:

	cooks	almond	angelica	anise	anise_seed	apple	apple_brandy	apricot
11321	Japanese	0	0	0	0	0	0	0
11361	Japanese	0	0	0	0	0	0	0
12171	Asian	0	0	0	0	0	0	0
12385	Asian	0	0	0	0	0	0	0
13159	Asian	0	0	0	0	0	0	0
13586	Japanese	0	0	0	0	0	0	0
14495	East_asian	0	0	0	0	0	0	0

- Based on the results obtained, the recipes with: **rice**, **soy sauce**, **wasabi**, **seaweed** and **carrot** belong to the Japan, Asia and East Asia Region.

Count the ingredients in all recipes.

In [14]:

```
# We add each column.
ingredient_present = recipes.iloc[:, 1:].sum(axis=0)
```

In [15]:

```
# Define each column as a pandas array.
ingredient = pd.Series(ingredient_present.index.values, index = np.arange
count = pd.Series(list(ingredient_present), index = np.arange(len(ingredi

# Create a DataFrame
ingredients_df = pd.DataFrame(dict(ingredient = ingredient, count = count
ingredients_df = ingredients_df[["ingredient", "count"]]
print(ingredients_df.to_string())
```

	ingredient	count
0	almond	2306
1	angelica	1
2	anise	223
3	anise_seed	87
4	apple	2420
5	apple_brandy	37
6	apricot	620
7	armagnac	11
8	artemisia	13
9	artichoke	391
10	asparagus	460
11	avocado	660
12	bacon	2169
13	baked_potato	9
14	balm	3
15	banana	989
16	barley	266
17	bartlett_pear	23
18	basil	3842
19	bay	1463
20	bean	1992
21	beech	1
22	beef	4902
23	beef_broth	845
24	beef_liver	10
25	beer	307
26	beet	233
27	bell_pepper	5979
28	bergamot	7
29	berry	183
30	bitter_orange	85
31	black_bean	494
32	black_currant	11
33	black_mustard_seed_oil	30
34	black_pepper	9825
35	black_raspberry	8
36	black_sesame_seed	26
37	black_tea	44
38	blackberry	170
39	blackberry_brandy	4
40	blue_cheese	396
41	blueberry	466
42	bone_oil	50
43	bourbon_whiskey	156
44	brandy	395
45	brassica	114
46	bread	4571
47	broccoli	929
48	brown_rice	346
49	brussels_sprout	92
50	buckwheat	90
51	butter	20715
52	buttermilk	1634
53	cabbage	1026
54	cabernet_sauvignon_wine	17
55	cacao	35
56	camembert_cheese	12
57	cane_molasses	7741
58	caraway	233

59	cardamom	352
60	carnation	3
61	carob	7
62	carrot	3689
63	cashew	208
64	cassava	19
65	catfish	71
66	cauliflower	332
67	caviar	28
68	cayenne	8253
69	celery	3625
70	celery_oil	1005
71	cereal	204
72	chamomile	3
73	champagne_wine	100
74	chayote	27
75	cheddar_cheese	3027
76	cheese	3278
77	cherry	1082
78	cherry_brandy	32
79	chervil	52
80	chicken	5436
81	chicken_broth	3603
82	chicken_liver	52
83	chickpea	402
84	chicory	156
85	chinese_cabbage	166
86	chive	1333
87	cider	1132
88	cilantro	2473
89	cinnamon	5593
90	citrus	167
91	citrus_peel	4
92	clam	476
93	clove	10
94	cocoa	4798
95	coconut	1801
96	coconut_oil	17
97	cod	180
98	coffee	719
99	cognac	67
100	concord_grape	12
101	condiment	9
102	coriander	1647
103	corn	4828
104	corn_flake	225
105	corn_grit	163
106	cottage_cheese	347
107	crab	574
108	cranberry	920
109	cream	10170
110	cream_cheese	2840
111	cucumber	1895
112	cumin	3274
113	cured_pork	315
114	currant	241
115	date	377
116	dill	1105
117	durian	0
118	eel	20

119	egg	21022
120	egg_noodle	317
121	elderberry	5
122	emmental_cheese	1
123	endive	115
124	enokidake	106
125	fennel	912
126	fenugreek	924
127	feta_cheese	623
128	fig	139
129	fish	2110
130	flower	32
131	frankfurter	37
132	fruit	479
133	galanga	49
134	gardenia	9
135	garlic	17351
136	gelatin	1417
137	geranium	1
138	gin	68
139	ginger	4358
140	goat_cheese	260
141	grape	346
142	grape_brandy	8
143	grape_juice	824
144	grapefruit	121
145	green_bell_pepper	2582
146	green_tea	35
147	gruyere_cheese	45
148	guava	13
149	haddock	31
150	ham	1300
151	hazelnut	284
152	herring	10
153	holy_basil	3
154	honey	2551
155	hop	3
156	horseradish	396
157	huckleberry	10
158	jamaican_rum	1
159	japanese_plum	13
160	jasmine	8
161	jasmine_tea	2
162	juniper_berry	33
163	kaffir_lime	1
164	kale	100
165	katsuobushi	63
166	kelp	179
167	kidney_bean	442
168	kiwi	109
169	kohlrabi	6
170	kumquat	33
171	lamb	482
172	lard	3051
173	laurel	2
174	lavender	62
175	leaf	9
176	leek	422
177	lemon	3043
178	lemon_juice	5065

179	lemon_peel	729
180	lemongrass	217
181	lentil	247
182	lettuce	1206
183	licorice	21
184	lilac_flower_oil	1
185	lima_bean	149
186	lime	1160
187	lime_juice	1618
188	lime_peel_oil	108
189	lingonberry	9
190	litchi	12
191	liver	42
192	lobster	131
193	long_pepper	2
194	lovage	142
195	macadamia_nut	102
196	macaroni	3115
197	mace	117
198	mackerel	44
199	malt	37
200	mandarin	279
201	mandarin_peel	15
202	mango	418
203	maple_syrup	478
204	marjoram	527
205	mate	1
206	matsutake	57
207	meat	987
208	melon	163
209	milk	12869
210	milk_fat	959
211	mint	1012
212	mozzarella_cheese	1288
213	mung_bean	24
214	munster_cheese	27
215	muscat_grape	1
216	mushroom	3370
217	mussel	168
218	mustard	4119
219	mutton	3
220	nectarine	51
221	nira	67
222	nut	1255
223	nutmeg	2506
224	oat	1265
225	oatmeal	61
226	octopus	45
227	okra	102
228	olive	1798
229	olive_oil	9874
230	onion	18078
231	orange	1724
232	orange_flower	17
233	orange_juice	1726
234	orange_peel	596
235	oregano	3180
236	ouzo	9
237	oyster	406
238	palm	46

239	papaya	57
240	parmesan_cheese	3173
241	parsley	5550
242	parsnip	139
243	passion_fruit	20
244	pea	1180
245	peach	531
246	peanut	509
247	peanut_butter	1014
248	peanut_oil	308
249	pear	484
250	pear_brandy	11
251	pecan	2176
252	pelargonium	1
253	pepper	9230
254	peppermint	142
255	peppermint_oil	8
256	pimenta	2
257	pimento	270
258	pineapple	1638
259	pistachio	219
260	plum	288
261	popcorn	97
262	porcini	106
263	pork	2056
264	pork_liver	5
265	pork_sausage	1369
266	port_wine	49
267	potato	3528
268	potato_chip	65
269	prawn	24
270	prickly_pear	20
271	provolone_cheese	168
272	pumpkin	803
273	quince	29
274	radish	525
275	raisin	1889
276	rapeseed	3
277	raspberry	784
278	raw_beef	2
279	red_algae	2
280	red_bean	33
281	red_kidney_bean	59
282	red_wine	1395
283	rhubarb	169
284	rice	3856
285	roasted_almond	3
286	roasted_beef	227
287	roasted_hazelnut	1
288	roasted_meat	15
289	roasted_nut	1
290	roasted_peanut	202
291	roasted_pecan	1
292	roasted_pork	124
293	roasted_sesame_seed	593
294	romano_cheese	275
295	root	101
296	roquefort_cheese	23
297	rose	56
298	rosemary	1892

299	rum	599
300	rutabaga	34
301	rye_bread	92
302	rye_flour	131
303	saffron	234
304	sage	904
305	sake	680
306	salmon	451
307	salmon_roe	15
308	sassafras	18
309	sauerkraut	185
310	savory	128
311	scallion	4782
312	scallop	300
313	sea_algae	4
314	seaweed	215
315	seed	1340
316	sesame_oil	1693
317	sesame_seed	778
318	shallot	1304
319	sheep_cheese	2
320	shellfish	27
321	sherry	706
322	shiitake	595
323	shrimp	1679
324	smoke	463
325	smoked_fish	6
326	smoked_salmon	100
327	smoked_sausage	268
328	sour_cherry	50
329	sour_milk	46
330	soy_sauce	3799
331	soybean	1195
332	soybean_oil	2
333	spearmint	6
334	squash	572
335	squid	238
336	star_anise	131
337	starch	2731
338	strawberry	1080
339	strawberry_jam	1
340	strawberry_juice	2
341	sturgeon_caviar	1
342	sumac	11
343	sunflower_oil	8
344	sweet_potato	529
345	swiss_cheese	519
346	tabasco_pepper	976
347	tamarind	1672
348	tangerine	52
349	tarragon	478
350	tea	108
351	tequila	142
352	thai_pepper	136
353	thyme	3043
354	tomato	9920
355	tomato_juice	176
356	truffle	52
357	tuna	463
358	turkey	900

```
359                 turmeric   1290
360                   turnip    188
361                  vanilla   9009
362                     veal    197
363                vegetable   1703
364            vegetable_oil  11103
365                  vinegar   8060
366                   violet      5
367                   walnut   2728
368                   wasabi    135
369               watercress    150
370               watermelon    110
371                    wheat  20775
372              wheat_bread     82
373                  whiskey    148
374              white_bread    370
375               white_wine   2205
376  whole_grain_wheat_flour    731
377                     wine   1026
378                     wood     33
379                      yam     85
380                    yeast   3385
381                   yogurt   1033
382                 zucchini   1102
```

- We now have a data frame of ingredients and their total counts across all recipes.
- Let's sort this data frame in descending order.

In [16]:
```
ingredients_df.sort_values(["count"], ascending=False, inplace=True)
ingredients_df.reset_index(inplace=True, drop=True)

print(ingredients_df)
```

```
          ingredient  count
0                egg  21022
1              wheat  20775
2             butter  20715
3              onion  18078
4             garlic  17351
..               ...    ...
378   strawberry_jam      1
379  sturgeon_caviar      1
380       kaffir_lime      1
381            beech      1
382           durian      0

[383 rows x 2 columns]
```

- Top 5 ingredients most used in recipes are:
  - Egg: 21022 times
  - Wheat: 20775 times
  - Butter: 20715 times
  - Onion: 18078 times
  - Garlic: 17351 times

In the table above, there are approximately 40,000 recipes from North America in our dataset, meaning the data is skewed towards ingredients from that region.

Therefore, we will do a more objective summary of the ingredients by looking at the ingredients by cuisine.

Let's create a *profile* for each cuisine.

We try to find out what ingredients Chinese people usually use and what **Caribbean** food is, for example.

In [17]:
```
cooks = recipes.groupby("cooks").mean()
cooks.head()
```

Out[17]:

cooks	almond	angelica	anise	anise_seed	apple	apple_brar
African	0.156522	0.0	0.000000	0.000000	0.034783	
Asian	0.007544	0.0	0.000838	0.002515	0.012573	
Cajun_creole	0.000000	0.0	0.000000	0.000000	0.006849	
Caribbean	0.016393	0.0	0.010929	0.000000	0.010929	
Central_southamerican	0.053942	0.0	0.000000	0.020747	0.000000	

We create a data frame where each row is a cuisine and each column (except the first column) is an ingredient, and the values in the rows represent the percentage of each ingredient in the corresponding cuisine.

For example:

- Almond is present in 15.65% of all **African** recipes.
- Garlic is present in 56.84% of all **Central_Southamerican** recipes.

Let's print out a profile of each cuisine showing the four main ingredients of each one.

In [18]:
```
total_ingredients = 4 # number of main ingredients to print.

# Print the main ingredients of each cuisine Function
def print_most_used_ingredients(row):
    print(f'{row.name.upper()}:')
    ordered_row = row.sort_values(ascending=False)*100
    most_used_ingredients = list(ordered_row.index.values)[0:total_ingred
    ordered_row = list(ordered_row)[0:total_ingredients]

    for index, ingredient in enumerate(most_used_ingredients):
        print(f'{ingredient}: {ordered_row[index]:.0f}%')
    print("\n")

# apply function to cuisines dataframe
create_kitchen_profiles = cooks.apply(print_most_used_ingredients, axis=1
```

```
AFRICAN:
onion: 53%
olive_oil: 52%
garlic: 50%
cumin: 43%


ASIAN:
soy_sauce: 50%
ginger: 49%
garlic: 48%
rice: 41%


CAJUN_CREOLE:
onion: 70%
cayenne: 56%
garlic: 49%
butter: 36%


CARIBBEAN:
onion: 51%
garlic: 51%
vegetable_oil: 31%
black_pepper: 31%


CENTRAL_SOUTHAMERICAN:
garlic: 57%
onion: 54%
cayenne: 52%
tomato: 41%


CHINESE:
soy_sauce: 69%
ginger: 53%
garlic: 53%
scallion: 48%


EAST_ASIAN:
garlic: 55%
soy_sauce: 50%
scallion: 50%
cayenne: 48%


EASTERN-EUROPE:
wheat: 53%
egg: 52%
butter: 48%
onion: 45%


EASTERNEUROPEAN_RUSSIAN:
butter: 60%
egg: 51%
wheat: 49%
```

```
onion: 38%


ENGLISH_SCOTTISH:
butter: 67%
wheat: 62%
egg: 53%
cream: 41%


FRENCH:
butter: 50%
egg: 44%
wheat: 37%
olive_oil: 28%


GERMAN:
wheat: 65%
egg: 61%
butter: 47%
onion: 35%


GREEK:
olive_oil: 76%
garlic: 44%
onion: 36%
lemon_juice: 34%


INDIAN:
cumin: 60%
turmeric: 51%
onion: 50%
coriander: 48%


ITALIAN:
olive_oil: 61%
garlic: 53%
tomato: 39%
onion: 33%


JAPANESE:
soy_sauce: 57%
rice: 44%
vinegar: 37%
vegetable_oil: 35%


JEWISH:
egg: 59%
wheat: 49%
butter: 31%
onion: 30%


KOREAN:
```

```
garlic: 59%
scallion: 52%
cayenne: 52%
soy_sauce: 49%


MEDITERRANEAN:
olive_oil: 80%
garlic: 51%
onion: 39%
tomato: 35%


MEXICAN:
cayenne: 74%
onion: 68%
garlic: 62%
tomato: 59%


MIDDLEEASTERN:
olive_oil: 60%
garlic: 47%
wheat: 38%
lemon_juice: 36%


MOROCCAN:
olive_oil: 73%
cumin: 55%
onion: 50%
garlic: 46%


NORTH-AFRICAN:
onion: 55%
olive_oil: 50%
cumin: 48%
garlic: 47%


NORTH-AMERICA:
butter: 41%
egg: 40%
wheat: 40%
onion: 29%


SCANDINAVIAN:
butter: 64%
wheat: 58%
egg: 53%
cream: 29%


SOUTH-AMERICA:
onion: 43%
garlic: 37%
egg: 35%
milk: 31%
```

```
SOUTHERN_SOULFOOD:
butter: 58%
wheat: 49%
egg: 42%
corn: 30%


SOUTHWESTERN:
cayenne: 81%
garlic: 62%
onion: 61%
cilantro: 52%


SPANISH_PORTUGUESE:
olive_oil: 58%
garlic: 54%
onion: 47%
bell_pepper: 35%


THAI:
garlic: 60%
fish: 53%
cayenne: 47%
cilantro: 42%


UK-AND-IRISH:
butter: 60%
wheat: 58%
egg: 48%
milk: 33%


VIETNAMESE:
fish: 74%
garlic: 73%
rice: 49%
cilantro: 43%


WESTERN:
egg: 51%
wheat: 46%
butter: 46%
black_pepper: 36%
```

Changelog

Date (DD/MM/YYYY)	Version	Description of change
15/15/2023	1.0	Notebook creation

Date (DD/MM/YYYY)	Version	Description of change
23/09/2023	1.1	Function added: print_most_used_ingredients

In [ ]:

```

```